

**MEMBANGUN APLIKASI REMOTE PRESENTATION
"VuDroid Wispa" BERBASIS ANDROID MENGGUNAKAN
GOOGLE SPEECH API**

SKRIPSI



**Disusun Oleh :
MUCHAMMAD ROFIQ
10.18.109**

**PROGRAM STUDI TEKNIK INFORMATIKA S-1
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2014**

RECEIVED THE HONORABLE MEMBERS OF
THE HOUSE OF REPRESENTATIVES
WASHINGTON, D.C.
JANUARY 1, 1903

1271000

THE HONORABLE
MEMBERS OF THE HOUSE OF REPRESENTATIVES
WASHINGTON, D.C.

THE HONORABLE MEMBERS OF THE HOUSE OF REPRESENTATIVES
WASHINGTON, D.C.
JANUARY 1, 1903

LEMBAR PERSETUJUAN DAN PENGESAHAN

MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN GOOGLE SPEECH API

SKRIPSI

*Disusun dan Diajukan untuk melengkapi dan memenuhi persyaratan guna
mencapai Gelar Sarjana Komputer Strata Satu (S-1)*

Disusun Oleh :
Muchammad Rofiq
10.18.109

Diperiksa dan Disetujui,

Dosen Pembimbing I

Joseph Dedy Irawan, ST., MT.
NIP. 197404162005011002

Dosen Pembimbing II

Suryo Adi Wibowo, ST., MT.
NIP.P. 1031000438

Program Studi Teknik Informatika S-1
Ketua

Joseph Dedy Irawan, ST., MT.
NIP. 197404162005011002

PROGRAM STUDI TEKNIK INFORMATIKA S-1
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2014

MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN GOOGLE SPEECH API

Muchammad Rofiq

Program Studi Teknik Informatika S-1
Fakultas Teknologi Industri
Institut Teknologi Nasional Malang
Jl. Raya Karanglo Km. 2 Tasikmadu-Malang
Email : snappers77@yahoo.com

**Dosen Pembimbing: 1. Joseph Dedy Irawan, ST., MT.
2. Suryo Adi Wibowo, ST, MT**

Abstrak

Pengenalan ucapan atau pengenalan wicara dalam istilah bahasa Inggrisnya, Automatic Speech Recognition (ASR) adalah suatu pengembangan teknik dan sistem yang memungkinkan komputer untuk menerima masukan berupa kata yang diucapkan.

Aplikasi remote presentation VuDroid Wispa adalah aplikasi yang berfungsi untuk mengendalikan slide presentasi yang berjalan pada program Ms.Power Point dari jarak jauh. Aplikasi ini menggunakan konektifitas Bluetooth atau Wifi sebagai media penghubung antara aplikasi client dengan aplikasi server serta memanfaatkan fitur speech recognition untuk input perintah pengendaliannya. Input perintah yang didapatkan dari speech recognition akan diproses untuk mengetahui apakah input sesuai dengan perintah yang tersimpan dalam database SQLite. Setiap perintah yang sesuai akan dieksekusi oleh program dengan mengirimkan paket data bertipe string ke aplikasi server yang nantinya akan diproses lebih lanjut pada aplikasi server untuk mengendalikan slide presentasi sesuai dengan perintah yang diterima dari aplikasi client.

Aplikasi remote presentation VuDroid Wispa ini dapat berjalan dengan maksimal pada smartphone dengan sistem operasi Android 4.1-Jelly Bean atau versi yang lebih tinggi. Fitur speech recognition pada aplikasi ini dapat memudahkan pengguna untuk mengendalikan presentasi yang berjalan pada Ms. Power Point tanpa perlu menyentuh smartphone. Jarak rata-rata pemanfaatan konektifitas bluetooth adalah 15 meter, sedangkan pemanfaatan konektifitas wifi mencapai jarak 19 meter. Hasil pengujian aplikasi ini secara keseluruhan telah dinilai baik oleh 20 user dengan prosentase rata-rata sebesar 75%.

Kata kunci : Speech recognition, remote, presentation, bluetooth, wifi, VuDroid Wispa.



KATA PENGANTAR

Puji syukur kehadiran Allah yang maha kuasa, karena telah memberikan rahmat dan hidayah-Nya sehingga penyusun dapat menyelesaikan skripsi dengan judul **MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN GOOGLE SPEECH API** sesuai dengan waktu yang ditentukan.

Penyusunan skripsi ini merupakan salah satu persyaratan untuk menyelesaikan program pendidikan Strata Satu (S-1) Teknik Informatika, Fakultas Teknologi Industri di Institut Teknologi Nasional Malang.

Pada penyusunan skripsi ini penulis mengucapkan terima kasih sebesar-besarnya kepada :

1. Husin dan Zakiyah yang merupakan kedua orang tua dan pendukung utama dari segi moril maupun materiil.
2. Ir.Soeparno Djiwo, MT, selaku Rektor Institut Teknologi Nasional Malang.
3. Ir. Anang Subardi, MT, selaku Dekan Fakultas Teknologi Industri, Institut Teknologi Nasional Malang.
4. Joseph Dedy Irawan, ST, MT, selaku Ketua Program Studi Teknik Informatika, Institut Teknologi Nasional Malang.
5. Sonny Prasetyo, ST, MT, selaku Sekertaris Program Studi Teknik Informatika, Institut Teknologi Nasional Malang.
6. Joseph Dedy Irawan, ST, MT, selaku Dosen Pembimbing I, yang selalu memberikan masukan.
7. Suryo Adi Wibowo, ST, MT selaku Dosen Pembimbing II dan sekaligus Kepala Laboratorium Robotika.
8. Laboratorium Robotika dan Asisten Lab. yang merupakan tempat tinggal kedua dan merupakan tempat bernaung dalam melakukan riset dan bersosialisasi.
9. Semua dosen Program Studi Teknik Informatika yang telah membantu dalam penulisan dan masukan.

10. Semua teman seperjuangan yang telah membantu dalam terselesaikannya skripsi ini.

Penyusun menyadari bahwa skripsi ini masih jauh dari sempurna, oleh karena itu penyusun mengharapkan kritik dan saran dari pembaca, Semoga skripsi ini bisa bermanfaat.

Malang, Agustus 2014

Penyusun

Lembar Persembahan



Alhamdulillah Robbil 'Alamin, Sujud dan Syukur kepada



*atas Segala Karunia dan Kasih SayangNYA, telah Memberikan Ilmu,
Kesabaran dan Kemudahan sehingga Skripsi yang Sederhana ini Dapat
Terselesaikan.*

Sholawat dan salam akan selalu dihaturkan kepada nabi kita, Rasulullah



Terima Kasih...

kepada Kedua Orang Tua Tercinta Husin dan Zakiyah

*atas Ketulusan Mu Merawat dan Membesarkan ku, serta Sabar Memberi Kasih
Sayang yang Tak Ada Batasnya.*

Terima Kasih...

kepada Kakak ku Ali dan Helmy serta Adik ku Tersayang Fatimah

*atas Semangat dan Dukungan di Setiap Langkah ku
dan atas Nasehat-nasehat yang telah Kalian Berikan.*

Terima Kasih...

*kepada Seluruh Keluarga Besar ku atas Segala Do'a dan Dukungan yang Selalu
Terucap untuk ku.*

Terima Kasih...

*kepada Cahya Luciet, bang Shu Lied, Natsun Gem, mbak Anna, Kaorin-chan,
Dante Widodo, Ical, Irei, Maya-san, Risa, Hiz, Hafid,
(Alm.) Shima Agus Tri Cahyo, Aya-san, Galvez, Red, dll
atas Keceriaan, Kebersamaan, Keseruan, Kecintaan yang Sama Terhadap Suikoden,
dan Semua Kebajikan yang telah Kalian Berikan di Salah Satu Rumah Terindah ku*

Castle Of Suikoden

Terima Kasih...

*kepada Teman-teman ku Suzudi, Erika Mika, Wayang, Obet, Dae, Kim, Galih
Reggae, Hana-chu, Hafid Shagame, seluruh member The Eternity dan Ninja Ranger
atas Semua Kenangan Indah selama Berada di*

Nindou International

Terima Kasih...

*atas Keseruan, Canda Tawa dan Kebersamaan kepada
Teman-teman ku Haris AzeGie, Celestine, Claire, Roys, Lican, Alda, Lutfi Helmi,
Adit, Della, Zannie, Guild Arcane dan Kenalan-kenalan lainnya di*

Elsword Indonesia

Terima Kasih...

kepada Pinky Miliasari... terima kasih~

Terima Kasih...

*kepada Deny, Zico, Yosi, Azam, Wahid, Ratri, Putri, Fahmi, (Alm.) Davi, Gobiez,
Rusda, Imam, Teman Seperjuangan Informatika S1-2010, dan Aslab Robotika.
atas Bantuan, Do'a, Nasehat, Semangat, Hiburan, Bully-an, Karaoke-an, Traktiran
serta Kenangan yang Telah Kalian Berikan.*

Terima Kasih kepada Snapp... karna Telah Menjadi Diriku Yang Lain Selama Ini

*Ku Persembahkan Karya ini Untuk Kalian, Semoga Allah SWT Membalas
Kebajikan dan Memberikan Kemudahan dalam Segala Hal... aamiin~*

DAFTAR ISI

Halaman Judul.....	i
Lembar Persetujuan Dan Pengesahan.....	ii
Abstrak.....	iii
Kata Pengantar	iv
Lembar Persembahan.....	vi
Daftar Isi	viii
Daftar Gambar.....	x
Daftar Tabel	xi
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah.....	2
1.3 Tujuan	3
1.4 Batasan Masalah	3
1.5 Manfaat Penelitian	3
1.6 Metodologi Penelitian	4
1.7 Sistematika Penulisan	5
BAB II LANDASAN TEORI	7
2.1 Presentasi.....	7
2.2 Microsoft Power Point	7
2.3 Remote Control.....	8
2.4 Aplikasi Mobile	9
2.5 Aplikasi Desktop.....	10
2.6 Android OS.....	10
2.7 Android Studio	12
2.8 Java Development Kit (JDK)	13
2.9 Software Development Kit (SDK).....	15
2.10 Android Virtual Device (AVD).....	17
2.11 Gradle.....	17
2.12 Bahasa Pemrograman Java	18
2.13 Client-Server Model.....	21
2.14 Speech Recognition	21
2.15 Bluetooth	24
2.16 Wireless Fidelity (Wi-fi)	25
2.17 Database SQLite	26
BAB III ANALISIS DAN PERANCANGAN SISTEM.....	29
3.1 Analisis Sistem	29
3.1.1 Analisis Kebutuhan Fungsional	29
3.1.2 Analisis Kebutuhan Non Fungsional.....	29
3.1.2.1 Analisis Kebutuhan Hardware.....	30
3.1.2.2 Analisis Kebutuhan Software	30
3.1.2.3 Analisis Kebutuhan Brainware	31
3.2 Perancangan Sistem	31
3.2.1 Perancangan Struktur Menu	31
3.2.2 Struktur Basis Data.....	32

3.2.3	Proses Interaksi.....	32
3.2.4	Proses Speech Recognition	33
3.2.5	Flowchart	34
3.2.6	Perancangan Interface.....	36
BAB IV	IMPLEMENTASI DAN PENGUJIAN	43
4.1	Implementasi	43
4.1.1	Implementasi Perangkat Keras.....	43
4.1.2	Implementasi Perangkat Lunak	43
4.1.3	Implementasi Aplikasi	44
4.1.4	Implementasi Aplikasi Server	44
4.1.5	Implementasi Opening Program.....	45
4.1.6	Implementasi Menu Utama	46
4.1.7	Implementasi Menu Setting	46
4.1.7.1	Setting Command List.....	47
4.1.7.2	Setting Theme Styles	48
4.1.8	Implementasi Menu About.....	49
4.1.9	Implementasi Menu Help.....	50
4.1.10	Implementasi Menu Connection	50
4.1.10.1	Bluetooth Connection	51
4.1.10.2	Wifi Connection.....	52
4.1.11	Implementasi Koneksi Server	53
4.1.12	Implementasi Speech Recognition	54
4.1.13	Implementasi Button Remote.....	55
4.1.14	Implementasi Menu Exit.....	55
4.2	Pengujian	56
4.2.1	Pengujian Fungsional.....	56
4.2.2	Pengujian Non Fungsional	62
BAB V	PENUTUP	65
5.1	Kesimpulan.....	65
5.2	Saran.....	66
	DAFTAR PUSTAKA	67
	LAMPIRAN	68

DAFTAR GAMBAR

Gambar 2.1	Versi Android	11
Gambar 2.2	Proses Speech Recognition	22
Gambar 2.3	Proses Komunikasi Bluetooth	24
Gambar 2.4	Proses Komunikasi Wifi	25
Gambar 3.1	Rancangan Struktur Menu.....	31
Gambar 3.2	Proses Interaksi.....	33
Gambar 3.3	Proses Speech Recognition Google	33
Gambar 3.4	Flowchart.....	34
Gambar 3.5	Flowchart Aplikasi Server.....	35
Gambar 3.6	Perancangan Tampilan Aplikasi Server.....	36
Gambar 3.7	Perancangan Tampilan Opening Aplikasi Client	37
Gambar 3.8	Perancangan Tampilan Menu Utama.....	37
Gambar 3.9	Perancangan Tampilan Setting	38
Gambar 3.10	Perancangan Tampilan Command List.....	38
Gambar 3.11	Perancangan Tampilan About	39
Gambar 3.12	Perancangan Tampilan Help	39
Gambar 3.13	Perancangan Tampilan Menu Connection	40
Gambar 3.14	Perancangan Tampilan Remote Menu	40
Gambar 3.15	Perancangan Tampilan Speech Recognition	41
Gambar 3.16	Perancangan Tampilan Button Remote.....	41
Gambar 3.17	Perancangan Tampilan Search Scan Bluetooth dan Input IP	42
Gambar 4.1	Tampilan Aplikasi Server.....	45
Gambar 4.2	Tampilan Opening Program	45
Gambar 4.3	Tampilan Menu Utama	46
Gambar 4.4	Tampilan Menu Setting.....	47
Gambar 4.5	Tampilan Setting Command List.....	47
Gambar 4.6	Tampilan Pengaturan Bahasa	48
Gambar 4.7	Tampilan Setting Theme Styles.....	49
Gambar 4.8	Tampilan Menu About.....	49
Gambar 4.9	Tampilan Menu Help	50
Gambar 4.10	Tampilan Menu Connection.....	50
Gambar 4.11	Tampilan Bluetooth Connection.....	51
Gambar 4.12	Tampilan Pengaktifan Device Bluetooth	51
Gambar 4.13	Tampilan Wifi Connection.....	52
Gambar 4.14	Tampilan Konfirmasi Wifi.....	52
Gambar 4.15	Tampilan Tombol Search.....	53
Gambar 4.16	Tampilan Search Scan Bluetooth.....	53
Gambar 4.17	Tampilan Search Input IP.....	54
Gambar 4.18	Tampilan Remote Speech Recognition.....	54
Gambar 4.19	Tampilan Button Remote	55
Gambar 4.20	Tampilan Konfirmasi Exit.....	55
Gambar 4.21	Gambaran Konektifitas Bluetooth	59
Gambar 4.22	Gambaran Konektifitas Wifi	60

DAFTAR TABEL

Tabel 3.1	Struktur Basis Data Command.....	32
Tabel 4.1	Implementasi Perangkat Keras.....	41
Tabel 4.2	Pengujian Fungsional	56
Tabel 4.3	Pengujian Resolusi Hardware	58
Tabel 4.4	Pengujian Konektifitas Bluetooth	58
Tabel 4.5	Pengujian Konektifitas Wifi	60
Tabel 4.6	Pengujian Input Speech Recognition.....	61
Tabel 4.7	Pengujian Aplikasi Server	62
Tabel 4.8	Pengujian Pengguna	62

BAB I

PENDAHULUAN

1.1 Latar Belakang

Peranan *notebook* dan *smartphone* pada saat ini semakin dibutuhkan dan telah menjadi suatu bagian yang penting dalam kehidupan sehari-hari masyarakat. *Notebook* memberikan berbagai macam fasilitas yang pada dasarnya sama dengan sebuah *Personal Computer (PC)*, namun dengan ukuran yang lebih kecil sehingga mudah untuk dibawa kemana saja. Di sisi lain, *smartphone* memberikan fasilitas dasar yang dimiliki oleh sebuah *mobile phone* pada umumnya, dan didukung dengan beberapa fasilitas yang dimiliki oleh *PC*. Sebagian besar masyarakat modern yang mengikuti perkembangan teknologi telah memiliki kedua alat tersebut dan selalu membawanya kemana saja.

Adapun pemanfaatan yang sering digunakan dari *notebook* atau *PC* salah satunya adalah untuk presentasi yang ditampilkan melalui LCD Proyektor. Presentasi merupakan suatu kegiatan berbicara di hadapan banyak hadirin atau salah satu bentuk komunikasi. Presentasi merupakan kegiatan pengajuan suatu topik, pendapat atau informasi kepada orang lain. Berbeda dengan pidato yang lebih sering dibawakan dalam acara resmi dan acara politik, presentasi lebih sering dibawakan dalam acara bisnis ataupun penyampaian materi perkuliahan.

Melihat kondisi tersebut, maka keberadaan *smartphone* juga bisa dimanfaatkan sebagai salah satu alternatif pendukung untuk mengatasi kesulitan yang sering dihadapi seseorang dalam melakukan presentasi, yaitu sebagai alat kontrol presentasi jarak jauh. Dengan merancang sebuah aplikasi ini, maka *smartphone* dapat berperan sebagai sebuah remot kontrol sehingga tidak perlu lagi mendekati *notebook* untuk mengendalikan *slide* presentasi. Aplikasi tersebut akan memanfaatkan fasilitas *Wireless Fidelity (Wi-Fi)* dan *Bluetooth* sebagai media koneksi antara *smartphone* dengan *notebook* dan memanfaatkan fitur *Speech Recognition* untuk pengendaliannya.

Pengenalan ucapan atau pengenalan wicara dalam istilah bahasa Inggrisnya, *Speech Recognition (SR)* adalah suatu pengembangan teknik dan sistem yang memungkinkan komputer untuk menerima masukan berupa kata yang diucapkan. Teknologi ini memungkinkan suatu perangkat untuk mengenali dan memahami kata-kata yang diucapkan dengan cara digitalisasi kata dan mencocokkan sinyal digital tersebut dengan suatu pola tertentu yang tersimpan dalam suatu perangkat. Kata-kata yang diucapkan diubah bentuknya menjadi sinyal digital dengan cara mengubah gelombang suara menjadi sekumpulan angka yang kemudian disesuaikan dengan kode-kode tertentu untuk mengidentifikasi kata-kata tersebut. Hasil dari identifikasi kata yang diucapkan dapat ditampilkan dalam bentuk tulisan atau dapat dibaca oleh perangkat teknologi sebagai sebuah komando untuk melakukan suatu pekerjaan.

Aplikasi ini diberi nama “VuDroid Wispa”, VuDroid diambil dari kata *voodoo* atau *vudu* yang merupakan ilmu sihir untuk mengendalikan objek dari jarak jauh dan Android yang merupakan suatu sistem operasi *smartphone* sehingga memiliki makna *smartphone* pengendali jarak jauh, serta Wispa yang diambil dari kata *whisper* memiliki arti berbisik sehingga “VuDroid Wispa” bermakna aplikasi remot kontrol dengan cara mengucapkan sesuatu (menggunakan suara) yang berbasis Android. Aplikasi “VuDroid Wispa” ini terdiri dari dua aplikasi yang berjalan di *server (notebook)* dan *client (smartphone)*. Aplikasi *server* akan berjalan pada *notebook* atau komputer dengan sistem operasi Microsoft Windows, sedangkan aplikasi *client* berjalan pada *smartphone* dengan sistem operasi Android.

1.2 Rumusan Masalah

Berdasarkan pada latar belakang masalah diatas, maka didapatkan rumusan masalah yang akan dibahas yaitu bagaimana membuat aplikasi yang dapat mempermudah proses pengontrolan *slide* presentasi pada *software* Ms. Power Point dengan *Speech Recognition* menggunakan android *smartphone*.

1.3 Tujuan

Adapun tujuan penelitian ini adalah untuk membuat aplikasi yang berfungsi sebagai remot kontrol presentasi dengan Ms. Power Point melalui Android *smartphone*.

1.4 Batasan Masalah

Batasan masalah dalam membangun aplikasi *remote* presentasi Ms. Power Point ini antara lain :

1. Aplikasi dibuat dengan menggunakan bahasa pemrograman Java.
2. Jenis sistem operasi yang digunakan untuk aplikasi *client* yaitu AndroidOS dan untuk aplikasi *server* menggunakan sistem operasi Windows.
3. *Software* yang digunakan dalam pembuatan aplikasi ini adalah Android Studio untuk aplikasi *client* dan NetBeans untuk aplikasi *server*.
4. Pengkoneksian antara *Server* dan *Client* dilakukan menggunakan *Wireless Fidelity (Wi-Fi)* dan *Bluetooth*.
5. Aplikasi ini membutuhkan koneksi internet untuk menggunakan *Google Speech API* sebagai fitur *Speech Recognition*.
6. Presentasi yang dikontrol adalah presentasi yang berjalan pada *software* Ms. Power Point.

1.5 Manfaat

Adapun manfaat yang dapat diambil dari aplikasi yang berbasis android ini adalah :

1. Membantu pengguna untuk melakukan presentasi pada *software* Ms. Power Point dengan bebas tanpa harus berada di dekat *notebook* atau *PC* terus menerus.
2. Mempermudah pengendalian presentasi pada *software* Ms. Power Point dengan menggunakan fitur *Speech Recognition*.
3. Sebagai bahan referensi bahwa *smartphone* dan *notebook* atau *PC* dapat diintegrasikan satu sama lain untuk berbagai macam hal yang baru antara lain sebagai aplikasi remot kontrol presentasi.

1.6 Metode Penelitian

Untuk dapat mencapai keinginan dalam pembuatan aplikasi “VuDroid Wispa” sebagai media *remote presentation* berbasis android ini, maka perlu dilakukan dengan langkah-langkah sebagai berikut :

1. Studi Literatur

Penelitian ini dimulai dengan studi literatur yaitu pengumpulan data yang berhubungan dengan permasalahan yang dibahas sehingga dapat membantu penyelesaian masalah dalam perancangan aplikasi dari sumber-sumber bacaan seperti, buku, jurnal, referensi, *web page*, blog, dan karya tulis ilmiah.

2. Analisis Permasalahan

Pada tahap ini dilakukan analisis terhadap hasil dari studi literatur yang telah dilakukan untuk mengetahui pemahaman mengenai konsep pembuatan program android dan konektifitas antara *client-server* serta penerapannya.

3. Perancangan Sistem

Pada tahap ini dilakukan perancangan terhadap sistem yang akan dibangun antara lain :

- a. Perancangan *opening layout* aplikasi “VuDroid Wispa”.
- b. Perancangan *layout* utama yang berisi 3 macam tombol yaitu tombol *connect* tombol *setting*, tombol *about*, serta 2 tombol pada menu yaitu tombol *help* dan tombol *exit*.
- c. Perancangan menu *connect* yang berisi 2 macam pilihan koneksi yaitu *Wi-Fi* atau *bluetooth*.
- d. Perancangan menu kontrol untuk *remote presentation*.
- e. Pengimplementasian fitur *speech recognition* untuk *remote presentation*.

Dalam tahap ini merupakan tahapan perancangan sistem secara keseluruhan, maka tahapan ini merupakan tahapan terpenting dalam pembuatan aplikasi ini.

4. Pembuatan Aplikasi

Pada tahap ini sistem yang telah dirancang kemudian diimplementasikan kedalam bahasa pemrograman yakni bahasa pemrograman Java.

5. Pengujian dan Evaluasi

Uji coba ini bertujuan untuk memastikan bahwa masing-masing bagian dari sistem ini dapat bekerja sesuai yang diharapkan.

6. Penyusunan Laporan dan Kesimpulan Akhir

Pada tahap ini dilakukan penyusunan laporan tentang seluruh isi penelitian dan analisisnya ke dalam format penulisan tugas akhir dengan disertai kesimpulan akhir.

1.7 Sistematika Penulisan

Sistematika penyusunan proposal skripsi ini untuk memberikan gambaran secara umum dari proposal skripsi yang dibuat yang meliputi bab-bab sebagai berikut :

BAB I : PENDAHULUAN

Berisi Latar Belakang, Rumusan Masalah, Tujuan, Manfaat, Batasan Masalah, Metode Penelitian, dan Sistematika Penulisan.

BAB II : LANDASAN TEORI

Membahas tentang *software* yang akan digunakan, bahasa pemrograman yang digunakan, metode yang digunakan serta berbagai macam *software* pendukung yang akan digunakan dalam pembuatan aplikasi ini.

BAB III : ANALISIS DAN PERANCANGAN SISTEM

Membahas tentang gambaran umum dari tata letak menu dan tampilan yang akan dibuat. Dalam perancangan sistem akan ditentukan rancangan struktur menu, Proses Interaksi, Flowchart dan *Layout Prototype*.

BAB IV : IMPLEMENTASI DAN PENGUJIAN

Membahas tentang hasil implementasi dan analisis hasil uji coba program. Serta memaparkan hasil-hasil dari tahap penelitian, tahap analisis, desain, implementasi desain, hasil testing dan

implementasinya, berupa penjelasan teoritik, baik secara kualitatif, kuantitatif, atau secara statistik.

BAB V : PENUTUP

Berisi kesimpulan dan saran. Kesimpulan didapat dari ulasan data-data penelitian, menyimpulkan bukti-bukti yang dapat diperoleh dan akhirnya menarik intisari apakah hasil yang didapat layak untuk digunakan dan diimplementasikan.

BAB II

LANDASAN TEORI

2.1 Presentasi

Presentasi merupakan suatu kegiatan berbicara di hadapan banyak hadirin atau salah satu bentuk komunikasi. Presentasi merupakan kegiatan pengajuan suatu topik, pendapat atau informasi kepada orang lain. Berbeda dengan pidato yang lebih sering dibawakan dalam acara resmi dan acara politik, presentasi lebih sering dibawakan dalam acara bisnis ataupun penyampaian materi perkuliahan.

Teknik Presentasi merupakan salah satu elemen dalam desain komunikasi Visual khususnya di bidang multimedia. Pada awalnya ketika jurusan Desain Komunikasi Visual ini masih baru, teknik presentasi ditafsirkan dengan beberapa macam penafsiran, misalnya pada penerapan bidang desain produk. Teknik presentasi mungkin ditafsirkan cara menampilkan barang-barang produk yang ingin dipresentasikan ke hadapan *audience*. Presentasi untuk bidang karya ilmiah, misalnya disampaikan dengan cara mendemonstrasikan hasil karya tersebut sehingga dipahami oleh *audience*. Akan tetapi, akhirnya berhasil disepakati bahwa bahasan tentang teknik (penyajian) presentasi dan media interaktif adalah cara seseorang menyajikan penjelasan terhadap data, uraian proses, maupun pembelajaran, baik disajikan di muka *audience* dengan bantuan alat peraga berupa *slide show*, program aplikasi yang menyajikan informasi interaktif yang dapat diakses secara personal, maupun presentasi dalam bentuk cetakan yang dibagikan kepada semua penerima informasi. (Kusrianto, 2007)

2.2 Microsoft Power Point

PowerPoint adalah sebuah program komputer untuk presentasi yang dikembangkan oleh Microsoft di dalam paket aplikasi kantor yaitu Microsoft Office. PowerPoint berjalan di atas komputer *PC* berbasis sistem operasi Microsoft Windows dan juga Apple Macintosh yang menggunakan sistem operasi Apple Mac OS. Aplikasi ini sangat banyak digunakan, apalagi oleh kalangan perkantoran dan pebisnis, para pendidik, siswa, dan trainer. Dalam Power Point, seperti halnya perangkat lunak pengolah presentasi lainnya, objek teks, grafik,

video, suara, dan objek-objek lainnya diposisikan dalam beberapa halaman individual yang disebut dengan "*slide*". Istilah *slide* dalam Power Point ini memiliki analogi yang sama dengan *slide* dalam proyektor biasa, yang telah kuno, akibat munculnya perangkat lunak komputer yang mampu mengolah presentasi semacam PowerPoint dan Impress. Setiap *slide* dapat dicetak atau ditampilkan dalam layar dan dapat dinavigasikan melalui perintah dari si presenter. *Slide* juga dapat membentuk dasar *webcast* (sebuah siaran di *World Wide Web*).

Power Point menawarkan dua jenis properti pergerakan, yakni *Custom Animations* dan *Transition*. Properti pergerakan *Entrance*, *Emphasis*, dan *Exit* objek dalam sebuah *slide* dapat diatur oleh *Custom Animation*, sementara *Transition* mengatur pergerakan dari satu *slide* ke *slide* lainnya. Semuanya dapat dianimaskan dalam banyak cara. Desain keseluruhan dari sebuah presentasi dapat diatur dengan menggunakan *Master Slide*, dan struktur keseluruhan dari presentasi dapat disunting dengan menggunakan *Primitive Outliner (Outline)*.

PowerPoint dapat menyimpan presentasi dalam beberapa format, yakni sebagai berikut.

1. *.PPT (*Power Point Presentation*), yang merupakan data biner dan tersedia dalam semua versi Power Point (termasuk Power Point 12)
2. *.PPS (*Power Point Show*), yang merupakan data biner dan tersedia dalam semua versi Power Point (termasuk Power Point 12)
3. *.POT (*Power Point Template*), yang merupakan data biner dan tersedia dalam semua versi Power Point (termasuk Power Point 12)
4. *.PPTX (*Power Point Presentation*), yang merupakan data dalam bentuk XML dan hanya tersedia dalam Power Point 12.

2.3 *Remote Control*

Remote control atau pengendali jarak jauh adalah sebuah alat elektronik yang digunakan untuk mengoperasikan sebuah mesin dari jarak jauh. Istilah *remote control* juga sering disingkat menjadi "*remote*" saja. *Remote* juga sering kali mengacu pada istilah "*controller*", "*donker*", "*doofer*", "*zapper*" "*click-buzz*", "*box*", "*flipper*", "*zippity*", "*clicker*" atau "*changer*".

Pada umumnya, pengendali jarak jauh digunakan untuk memberikan perintah dari kejauhan kepada televisi atau barang-barang elektronik lainnya seperti sistem stereo dan pemutar DVD. *Remote control* untuk perangkat-perangkat ini biasanya berupa benda kecil nirkabel yang dipegang dalam tangan dengan sederetan tombol untuk menyesuaikan berbagai setting, seperti misalnya saluran televisi, nomor trek, dan volume suara. Malah, pada kebanyakan peranti modern dengan kontrol seperti ini, remot kontrolnya memiliki segala kontrol fungsi sementara perangkat yang dikendalikan itu sendiri hanya mempunyai sedikit kontrol utama yang mendasar. Kebanyakan *remote* berkomunikasi dengan perangkatnya masing-masing melalui sinyal-sinyal infra merah dan beberapa saja melalui sinyal radio. *Remote control* biasanya menggunakan baterai AAA yang kecil atau AA sebagai sumber dayanya.

2.4 Aplikasi *Mobile*

Pengertian aplikasi adalah program yang digunakan orang untuk melakukan sesuatu pada sistem komputer. *Mobile* dapat diartikan sebagai perpindahan yang mudah dari satu tempat ke tempat yang lain, misalnya telepon *mobile* berarti bahwa terminal telepon yang dapat berpindah dengan mudah dari satu tempat ke tempat lain tanpa terjadi pemutusan atau terputusnya komunikasi.. Aplikasi ini dapat diakses melalui perangkat nirkabel seperti pager, seperti telepon seluler dan PDA. Adapun karakteristik perangkat *mobile* yaitu:

1. Ukuran yang kecil : Perangkat *mobile* memiliki ukuran yang kecil. Konsumen menginginkan perangkat yang terkecil untuk kenyamanan dan mobilitas mereka.
2. *Memory* yang terbatas : Perangkat *mobile* juga memiliki *memory* yang kecil, yaitu *primary* (RAM) dan *secondary* (disk).
3. Daya proses yang terbatas : Sistem *mobile* tidaklah setangguh rekan mereka yaitu *desktop*.
4. Mengonsumsi daya yang rendah : Perangkat *mobile* menghabiskan sedikit daya dibandingkan dengan mesin *desktop*.
5. Kuat dan dapat diandalkan : Karena perangkat *mobile* selalu dibawa kemana saja, mereka harus cukup kuat untuk menghadapi benturan-benturan, gerakan, dan sesekali tetesan-tetesan air.

6. Konektivitas yang terbatas : Perangkat *mobile* memiliki bandwidth rendah, beberapa dari mereka bahkan tidak tersambung.
7. Masa hidup yang pendek : Perangkat-perangkat konsumen ini menyala dalam hitungan detik kebanyakan dari mereka selalu menyala. (Huda, 2013)

2.5 Aplikasi *Desktop*

Aplikasi *Desktop* adalah suatu aplikasi yang dapat berjalan sendiri atau independen tanpa menggunakan browser atau koneksi internet di suatu komputer otonom. Aplikasi *Desktop* memiliki keunggulan dan juga kekurangan dalam penggunaannya. Keunggulan Aplikasi *Desktop* antara lain :

1. Dapat berjalan dengan independen, tanpa perlu menggunakan browser.
2. Tidak perlu koneksi internet, karena semua file yang diperlukan untuk menjalankan aplikasinya sudah terinstall sebelumnya.
3. Dapat dengan mudah memodifikasi settingannya.
4. Prosesnya lebih cepat.

Adapun Kekurangan Aplikasi *Desktop* antara lain :

1. Apabila akan menjalankan aplikasi, harus diinstal terlebih dahulu di komputer.
2. Bermasalah dengan lisensi. Hal ini membutuhkan lisensi yang banyak pada setiap komputer.
3. Aplikasi tidak dapat dibuka di komputer lain, jika belum diinstall.
4. Biasanya memerlukan *hardware* dengan spesifikasi tinggi. (Huda, 2013)

2.6 Android OS

Android adalah sistem operasi berbasis Linux yang dirancang untuk perangkat seluler layar sentuh seperti *smartphone* dan komputer tablet. Android menyediakan *platform* terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak.

Awalnya, Google Inc. membeli Android Inc., pendatang baru yang membuat peranti lunak untuk ponsel. Kemudian untuk mengembangkan Android, dibentuklah *Open Handset Alliance*, konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk Google, HTC, Intel, Motorola,

Qualcomm, T-Mobile, dan Nvidia. Pada saat perilisan perdana Android, 5 November 2007, Android bersama *Open Handset Alliance* menyatakan mendukung pengembangan standar terbuka pada perangkat seluler. Di lain pihak, Google merilis kode-kode Android di bawah lisensi Apache, sebuah lisensi perangkat lunak dan standar terbuka perangkat seluler. Di dunia ini terdapat dua jenis distributor sistem operasi Android. Pertama yang mendapat dukungan penuh dari Google atau *Google Mail Services* (GMS) dan kedua adalah yang benar-benar bebas distribusinya tanpa dukungan langsung Google atau dikenal sebagai *Open Handset Distribution* (OHD).

Sejak Android dirilis telah dilakukan berbagai pembaruan berupa perbaikan *bug* dan penambahan fitur baru. Telepon pertama yang memakai sistem operasi Android adalah HTC *Dream*, yang dirilis pada 22 Oktober 2008. Pada penghujung tahun 2009 diperkirakan di dunia ini paling sedikit terdapat 18 jenis telepon seluler yang menggunakan Android. Versi dari Android adalah sebagai berikut :

Versi	Nama kode	Tanggal rilis	Level API	Distribusi
1.5	<i>Cupcake</i>	30 April 2009	3	0%
1.6	<i>Donut</i>	15 September 2009	4	0%
2.0–2.1	<i>Eclair</i>	26 Oktober 2009	7	0%
2.2	<i>Froyo</i>	20 Mei 2010	8	1,7%
2.3.3–2.3.7	<i>Gingerbread</i>	9 Februari 2011	10	26,3%
2.3–2.3.2	<i>Gingerbread</i>	6 Desember 2010	9	0%
3.2	<i>Honeycomb</i>	15 Juli 2011	13	0,1%
3.1	<i>Honeycomb</i>	10 Mei 2011	12	0,0%
4.0.3–4.0.4	<i>Ice Cream Sandwich</i>	16 Desember 2011	15	19,8%
4.3.x	<i>Jelly Bean</i>	24 Juli 2013	18	2,3%
4.2.x	<i>Jelly Bean</i>	13 November 2012	17	12,5%
4.1.x	<i>Jelly Bean</i>	9 Juli 2012	16	37,3%
4.4	<i>KitKat</i> ^[131]	31 Oktober 2013 ^[132]	19	

Gambar 2.1 : Versi Android

Fitur yang tersedia di Android adalah :

1. Kerangka aplikasi : itu memungkinkan penggunaan dan penghapusan komponen yang tersedia.
2. Dalvik mesin virtual : mesin virtual dioptimalkan untuk perangkat telepon seluler.

3. Grafik : grafik di 2D dan grafis 3D berdasarkan pustaka *OpenGL*.
4. *SQLite* : untuk penyimpanan data.
5. Mendukung media : *audio*, *video*, dan berbagai format gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF).
6. GSM, *Bluetooth*, EDGE, 3G, 4G dan *WiFi* (tergantung piranti keras).
7. Kamera, *Global Positioning System (GPS)*, kompas, NFC dan *accelerometer* (tergantung piranti keras).

Android memiliki berbagai keunggulan sebagai piranti lunak yang memakai basis kode komputer yang bisa didistribusikan secara terbuka (*open source*) sehingga pengguna bisa membuat aplikasi baru di dalamnya. Android memiliki aplikasi *native* Google yang terintegrasi seperti pushmail *Gmail*, *Google Maps*, dan *Google Calendar*. Para penggemar *open source* kemudian membangun komunitas yang membangun dan berbagi Android berbasis *firmware* dengan sejumlah penyesuaian dan fitur-fitur tambahan, seperti *FLAC lossless audio* dan kemampuan untuk menyimpan download aplikasi pada *microSD card*. Mereka sering memperbaharui paket-paket *firmware* dan menggabungkan elemen-elemen fungsi Android yang belum resmi diluncurkan dalam suatu *carrier-sanction firmware*. (Wikipedia, 2014)

2.7 Android Studio

Android Studio adalah sebuah IDE (*Integrated Development Environment*) untuk mengembangkan aplikasi untuk *platform* Android. Diumumkan pada tanggal 16 Mei 2013 pada Google *I/O* konferensi oleh *Product Manager* Google, Ellie Powers. Pada Juni 2013, Android Studio tersedia bagi pengguna untuk mencobanya (gratis) sebagai pratinjau akses awal. Berdasarkan pada JetBrains' *IntelliJ IDEA Software*, Studio yang dirancang khusus untuk pengembangan Android. Android Studio tersedia untuk didownload pada Windows, Mac OS X dan Linux.

Ada beberapa fitur yang diharapkan akan diluncurkan kepada pengguna sebagai dalam pengembangan *software* ini, adapun fitur-fitur yang disediakan untuk saat ini, antara lain:

1. *Live Layout: WYSIWYG Editor - Live Coding - Real-time App Rendering.*
2. *Developer Console:* tips optimasi, bantuan untuk penerjemahan, pelacakan rujukan, kampanye dan promosi - Metrik Penggunaan..
3. Penetapan beta rilis dan jadwal penyambutan.
4. *Build Support* berbasis Gradle.
5. *Android-specific refactoring* dan perbaikann cepat.
6. *Lint tools* untuk menganalisa kinerja, kegunaan, kesesuaian versi dan masalah lainnya.
7. Kemampuan *ProGuard* dan *app-signing*.
8. *Template-based wizard* untuk membuat desain Android dan komponen yang umum.
9. Sebuah *layout editor* kaya yang memungkinkan pengguna untuk *drag-and-drop* komponen UI, opsi untuk menampilkan *layout* pada konfigurasi *multiple screen*.

2.8 Java Development Kit (JDK)

Java Development Kit (JDK) merupakan implementasi dari salah satu dari Java SE, Java EE atau Java ME *platform* yang dirilis oleh Oracle Corporation dalam bentuk produk biner yang ditujukan untuk pengembang Java pada Solaris, Linux, Mac OS X atau Windows. Sejak diperkenalkannya *platform* Java, JDK merupakan *Software Development Kit* (SDK) yang paling banyak digunakan sejauh ini. Pada 17 November 2006, Sun mengumumkan bahwa JDK akan dirilis di bawah GNU *General Public License* (GPL), sehingga membuat JDK menjadi perangkat lunak bebas. Hal ini terjadi sekitar tanggal 8 Mei 2007, ketika Sun memberikan kontribusi kode sumber untuk OpenJDK.

JDK memiliki komponen utama dan koleksi *tools* pemrograman, termasuk diantaranya :

1. *appletviewer* - alat ini dapat digunakan untuk menjalankan dan men-debug Java Applet tanpa *web browser*.
2. *apt* - alat anotasi pengolahan.
3. *extcheck* - utilitas yang dapat mendeteksi konflik JAR *file*.

4. *idlj - IDL-to-Java compiler*. Utilitas ini menghasilkan pengikatan Java dari file yang diberikan Java IDL.
5. *jabswitch - Jembatan Java Access*. Paparan teknologi pendukung pada sistem Microsoft Windows.
6. *java - loader* untuk aplikasi Java. Alat ini merupakan interpreter dan dapat menginterpretasikan file kelas yang dihasilkan oleh *compiler* *javac*. Sekarang sebuah peluncur tunggal digunakan untuk kedua pengembangan dan penyebaran. Peluncur penyebaran yang lama, JRE, tidak lagi datang dengan Sun JDK, dan sebagai gantinya telah digantikan oleh *java loader*.
7. *javac - compiler* Java, yang mengubah kode sumber menjadi Java *bytecode*.
8. *javadoc - generator dokumentasi*, yang secara otomatis menghasilkan dokumentasi dari komentar kode sumber.
9. *jar - Pengarsip*, paket perpustakaan kelas yang terkait ke dalam sebuah file JAR tunggal. Alat ini juga membantu mengelola file JAR.
10. *javafxpackager - alat* untuk memaketkan dan menandatangani aplikasi JavaFX.
11. *jarsigner - penandatanganan jar dan verifikasi alat*.
12. *jawah - header C dan stub generator* digunakan untuk menulis metode asli
13. *javap - file kelas disassembler*.
14. *javaws - peluncur Java Web Start* untuk aplikasi JNLP (*Java Network Launching Protocol*).
15. *JConsole - Java Monitoring dan Management Console*.
16. *jdb - debugger*.
17. *jhat - Analisis Java Heap Tool*. (Percobaan).
18. *Jinfo - Utilitas* ini mendapat informasi konfigurasi dari proses Java yang berjalan atau *crash dump*. (Percobaan).
19. *jmap - Utilitas* ini menampilkan pemetaan memori untuk Java dan dapat mencetak peta memori *shared object* atau rincian memori tumpukan pada suatu proses atau *core dump*. (Percobaan).
20. *jmc - Java Mission Control*.
21. *jps - Java Virtual Machine Process Status Tools* mendaftarkan instrumentasi HotSpot Java Virtual Machines (JVMs) pada sistem target. (Percobaan).

22. *jruncscript* - Java *command-line shell script*.
23. *jstack* - utilitas yang mencetak jejak Java *stack* dari Java *thread*. (Percobaan).
24. *jstat* - Java *Virtual Machine* alat pemantauan statistik. (Percobaan).
25. *jstatd* - daemon *jstat*. (Percobaan).
26. *keytool* - alat untuk memanipulasi *keystore*.
27. *pack200* - alat kompresi JAR.
28. *policytool* - penciptaan kebijakan dan manajemen alat, yang dapat menentukan kebijakan untuk *runtime* Java, yang menetapkan hak akses yang tersedia untuk kode dari berbagai sumber.
29. *VisualVM* - visual yang mengintegrasikan beberapa alat JDK alat baris perintah dan kinerja ringan serta kemampuan pencatatan memori.
30. *wsimport* - menghasilkan *portable* JAX-WS artefak untuk memanggil layanan *web*.
31. *xjc* - Bagian dari API Java untuk XML *Binding* (JAXB) API. Ia menerima skema XML dan menghasilkan kelas Java.

Alat percobaan mungkin tidak tersedia dalam versi masa depan dari JDK. JDK juga dilengkapi dengan *Java Runtime Environment* lengkap, biasanya disebut *runtime* pribadi, karena fakta bahwa itu terpisah dari JRE “biasa” dan memiliki isi tambahan. Ini terdiri dari *Java Virtual Machine* dan semua kelas *library* dalam lingkungan produksi, serta *library* tambahan hanya berguna untuk pengembang, seperti perpustakaan internasionalisasi dan perpustakaan IDL. Salinan JDK juga mencakup berbagai pilihan contoh program yang menunjukkan penggunaan hampir semua bagian dari API Java. (Oracle, 2014)

2.9 *Software Development Kit* (SDK)

Software Development Kit merupakan satu set alat pengembangan perangkat lunak yang memungkinkan untuk pembuatan aplikasi untuk paket perangkat lunak tertentu, kerangka kerja perangkat lunak, *platform* perangkat keras, sistem komputer, *video game console*, sistem operasi, atau *platform* pengembangan yang sama.

SDK merupakan sesuatu yang sederhana seperti pelaksanaan satu atau lebih *Application Programming Interface* (API) dalam bentuk beberapa *library* untuk

antarmuka pada bahasa pemrograman tertentu atau untuk memasukkan *hardware* canggih yang dapat berkomunikasi dengan sistem tertanam tertentu. Alat umum mencakup fasilitas *debugging* dan utilitas lain, sering disajikan dalam sebuah lingkungan pengembangan terpadu atau IDE (*Integrated Development Environment*). SDK juga termasuk kode contoh dan catatan pendukung teknis atau dokumentasi pendukung lainnya untuk membantu menjelaskan poin yang dibuat oleh bahan referensi utama.

SDK mungkin telah memiliki izin yang membuat mereka tidak cocok untuk membangun perangkat lunak yang ditujukan untuk dikembangkan di bawah lisensi yang tidak kompatibel. Misalnya, SDK pribadi mungkin akan tidak sesuai dengan pengembangan perangkat lunak bebas, sementara SDK dengan lisensi GPL bisa tidak sesuai dengan pengembangan perangkat lunak pribadi. SDK LGPL (*Lesser General Public License*) biasanya aman untuk pengembangan pribadi.

Seorang programmer perangkat lunak biasanya menerima SDK dari pengembang sistem target. Seringkali SDK dapat didownload langsung melalui Internet. Banyak SDK yang disediakan secara gratis untuk mendorong pengembang untuk menggunakan sistem atau bahasa. Kadang-kadang ini digunakan sebagai alat pemasaran.

Sebuah SDK untuk sistem operasi add-on (misalnya, *QuickTime* untuk Mac OS) mungkin termasuk perangkat lunak *add-on* itu sendiri, yang akan digunakan untuk tujuan pembangunan jika tidak tentu untuk redistribusi bersama-sama dengan produk yang dikembangkan. Sebuah situasi yang menarik muncul di sini antara *platform* di mana dimungkinkan untuk mengembangkan aplikasi yang setidaknya dapat memulai pada konfigurasi sistem tanpa *add-on* yang diinstal, dan menggunakan *Gestalt-style run-time environment* untuk menentukan apakah *add-on* ada, dan di mana aplikasi akan gagal untuk dijalankan. Dengan kata lain, adalah mungkin untuk membangun sebuah biner tunggal yang akan berjalan pada konfigurasi dengan dan tanpa *add-on* ini, meskipun operasi dengan fungsionalitas yang berkurang dalam situasi tertentu.

Penyedia SDK untuk sistem atau subsistem tertentu kadang-kadang menggantikan istilah yang lebih spesifik, bukan perangkat lunak. Sebagai contoh, baik Microsoft dan Apple menyediakan *Driver Development Kit* (DDK) untuk mengembangkan *driver* perangkat. (Oracle, 2014)

2.10 *Android Virtual Device* (AVD)

AVD kependekan dari *Android Virtual Device* adalah konfigurasi emulator yang memungkinkan Anda membuat model perangkat seperti perangkat yang sebenarnya dengan mendefinisikan pilihan *hardware* dan *software* yang akan ditiru oleh Emulator Android.

Sebuah AVD terdiri dari:

1. Sebuah profil *hardware*: Mendefinisikan fitur perangkat keras dari perangkat virtual. Sebagai contoh, Anda dapat menentukan apakah perangkat memiliki kamera, apakah menggunakan *keyboard QWERTY* fisik atau pad panggilan, berapa banyak memori yang dimilikinya, dan sebagainya.
2. Sebuah pemetaan untuk gambar sistem: Anda dapat menentukan apa versi *platform* Android yang akan berjalan pada perangkat virtual. Anda dapat memilih versi *platform* Android standar atau gambar sistem dikemas dengan SDK *add-on*.
3. Pilihan lain: Anda dapat menentukan *skin* emulator yang ingin Anda gunakan dengan AVD, yang memungkinkan Anda mengontrol dimensi layar, penampilan, dan sebagainya. Anda juga dapat menentukan kartu SD tiruan untuk digunakan dengan AVD.
4. Sebuah tempat penyimpanan khusus pada komputer anda: data pengguna perangkat (aplikasi yang diinstal, pengaturan, dan sebagainya) dan kartu SD tiruan disimpan di daerah ini.

2.11 Gradle

Gradle adalah alat otomatisasi proyek yang dibangun berdasarkan konsep Apache Ant dan Maven Apache dan memperkenalkan Groovy-based *Domain Specific Language* (DSL), bukan dalam bentuk XML yang lebih tradisional untuk mendeklarasikan konfigurasi proyek.

Tidak seperti Apache Maven, yang mendefinisikan siklus hidup, dan Apache Ant, dimana target yang dipanggil berdasarkan *depends-on partial ordering* (ketergantungan order parsial), Gradle menggunakan *directed acyclic graph* ("DAG") untuk menentukan urutan dimana tugas dapat dijalankan.

Gradle dirancang untuk *multi-project builds* yang dapat tumbuh menjadi cukup besar, dan mendukung *incremental builds* (build tambahan) dengan cerdas menentukan bagian-bagian dari *build tree* yang *up-to-date*, sehingga setiap bagian-bagian tugas yang tergantung pada bagian itu tidak perlu dieksekusi kembali.

Plugin awal sangat terfokus di sekitar pengembangan dan penyebaran Java, Groovy, serta Scala, tetapi memiliki bahasa yang lebih banyak dan alur kerja proyek lebih akurat.

2.12 Bahasa Pemrograman Java

Java adalah bahasa pemrograman yang dapat dijalankan di berbagai komputer termasuk telepon genggam. Bahasa ini awalnya dibuat oleh James Gosling saat masih bergabung di Sun Microsystems saat ini merupakan bagian dari Oracle dan dirilis tahun 1995. Bahasa ini banyak mengadopsi sintaksis yang terdapat pada C dan C++ namun dengan sintaksis model objek yang lebih sederhana. Aplikasi-aplikasi berbasis java umumnya dikompilasi ke dalam p-code (*bytecode*) dan dapat dijalankan pada berbagai *Java Virtual Machine* (JVM). Java merupakan bahasa pemrograman yang bersifat umum/non-spesifik (*general purpose*), dan secara khusus didisain untuk memanfaatkan dependensi implementasi seminimal mungkin. Karena fungsionalitasnya yang memungkinkan aplikasi java mampu berjalan di beberapa *platform* sistem operasi yang berbeda, java dikenal pula dengan slogannya, "Tulis sekali, jalankan di mana pun". Saat ini java merupakan bahasa pemrograman yang paling populer digunakan, dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi.

Versi awal Java pada tahun 1996 sudah merupakan versi release sehingga dinamakan Java Versi 1.0. Java versi ini menyertakan banyak paket standar awal yang terus dikembangkan pada versi selanjutnya:

1. `java.lang` : Peruntukan kelas elemen-elemen dasar.
2. `java.io` : Peruntukan kelas input dan output, termasuk penggunaan berkas.
3. `java.util` : Peruntukan kelas pelengkap seperti kelas struktur data dan kelas kelas penanggalan.
4. `java.net` : Peruntukan kelas TCP/IP, yang memungkinkan berkomunikasi dengan komputer lain menggunakan jaringan TCP/IP.
5. `java.awt`: Kelas dasar untuk aplikasi *Graphical User Interface* (GUI).
6. `java.applet`: Kelas dasar aplikasi antar muka untuk diterapkan pada penjelajah *web*.

Java juga mempunyai beberapa kelebihan dan kekurangan dalam penggunaannya.

Kelebihan Java antara lain :

1. **Multiplatform.** Kelebihan utama dari Java ialah dapat dijalankan di beberapa *platform*/sistem operasi komputer, sesuai dengan prinsip tulis sekali, jalankan di mana saja. Dengan kelebihan ini pemrogram cukup menulis sebuah program Java dan dikompilasi (diubah, dari bahasa yang dimengerti manusia menjadi bahasa mesin/*bytecode*) sekali lalu hasilnya dapat dijalankan di atas beberapa *platform* tanpa perubahan. Kelebihan ini memungkinkan sebuah program berbasis java dikerjakan diatas *operating system* Linux tetapi dijalankan dengan baik di atas Microsoft Windows. *Platform* yang didukung sampai saat ini adalah Microsoft Windows, Linux, Mac OS dan Sun Solaris. Penyebabnya adalah setiap sistem operasi menggunakan programnya sendiri-sendiri (yang dapat diunduh dari situs Java) untuk menginterpretasikan *bytecode* tersebut.
2. **OOP (*Object Oriented Programming* - Pemrogram Berorientasi Objek)**
3. **Perpustakaan Kelas Yang Lengkap,** Java terkenal dengan kelengkapan *library*/perpustakaan (kumpulan program program yang disertakan dalam pemrograman java) yang sangat memudahkan dalam penggunaan oleh para pemrogram untuk membangun aplikasinya. Kelengkapan perpustakaan ini ditambah dengan keberadaan komunitas Java yang besar yang terus menerus membuat perpustakaan-perpustakaan baru untuk melingkupi seluruh kebutuhan pembangunan aplikasi.

4. **Bergaya C++**, memiliki sintaks seperti bahasa pemrograman C++ sehingga menarik banyak pemrogram C++ untuk pindah ke Java. Saat ini pengguna Java sangat banyak, sebagian besar adalah pemrogram C++ yang pindah ke Java. Universitas-universitas di Amerika Serikat juga mulai berpindah dengan mengajarkan Java kepada murid-murid yang baru karena lebih mudah dipahami oleh murid dan dapat berguna juga bagi mereka yang bukan mengambil jurusan komputer.
5. **Pengumpulan sampah otomatis**, memiliki fasilitas pengaturan penggunaan memori sehingga para pemrogram tidak perlu melakukan pengaturan memori secara langsung (seperti halnya dalam bahasa C++ yang dipakai secara luas).

Adapun kekurangan Java diantaranya :

1. Tulis sekali, jalankan di mana saja - Masih ada beberapa hal yang tidak kompatibel antara *platform* satu dengan *platform* lain. Untuk J2SE, misalnya SWT-AWT *bridge* yang sampai sekarang tidak berfungsi pada Mac OS X.
2. Mudah didekompilasi. Dekompilasi adalah proses membalikkan dari kode jadi menjadi kode sumber. Ini dimungkinkan karena kode jadi Java merupakan *bytecode* yang menyimpan banyak atribut bahasa tingkat tinggi, seperti nama-nama kelas, metode, dan tipe data. Hal yang sama juga terjadi pada Microsoft .NET *Platform*. Dengan demikian, algoritma yang digunakan program akan lebih sulit disembunyikan dan mudah dibajak/direverse-engineer.
3. Penggunaan memori yang banyak. Penggunaan memori untuk program berbasis Java jauh lebih besar daripada bahasa tingkat tinggi generasi sebelumnya seperti C/C++ dan Pascal (lebih spesifik lagi, Delphi dan Object Pascal). Biasanya ini bukan merupakan masalah bagi pihak yang menggunakan teknologi terbaru (karena trend memori terpasang makin murah), tetapi menjadi masalah bagi mereka yang masih harus berkutat dengan mesin komputer berumur lebih dari 4 tahun. (Kadir, 2005)

2.13 *Client-Server Model*

Client-Server merupakan sebuah paradigma dalam teknologi informasi yang merujuk kepada cara untuk mendistribusikan aplikasi ke dalam dua pihak: pihak *client* dan pihak *server*.

Dalam model *client-server*, sebuah aplikasi dibagi menjadi dua bagian yang terpisah, tapi masih merupakan sebuah kesatuan yakni komponen *client* dan komponen *server*. Komponen *client* juga sering disebut sebagai front-end, sementara komponen *server* disebut sebagai back-end. Komponen *client* dari aplikasi tersebut dijalankan dalam sebuah *workstation* dan menerima masukan data dari pengguna. Komponen *client* tersebut akan menyiapkan data yang dimasukkan oleh pengguna dengan menggunakan teknologi pemrosesan tertentu dan mengirimkannya kepada komponen *server* yang dijalankan di atas mesin *server*, umumnya dalam bentuk *request* terhadap beberapa layanan yang dimiliki oleh *server*. Komponen *server* akan menerima *request* dari *client*, dan langsung memprosesnya dan mengembalikan hasil pemrosesan tersebut kepada *client*. *Client* pun menerima informasi hasil pemrosesan data yang dilakukan *server* dan menampilkannya kepada pengguna, dengan menggunakan aplikasi yang berinteraksi dengan pengguna. (Ramadhan, 2006)

2.14 *Speech Recognition*

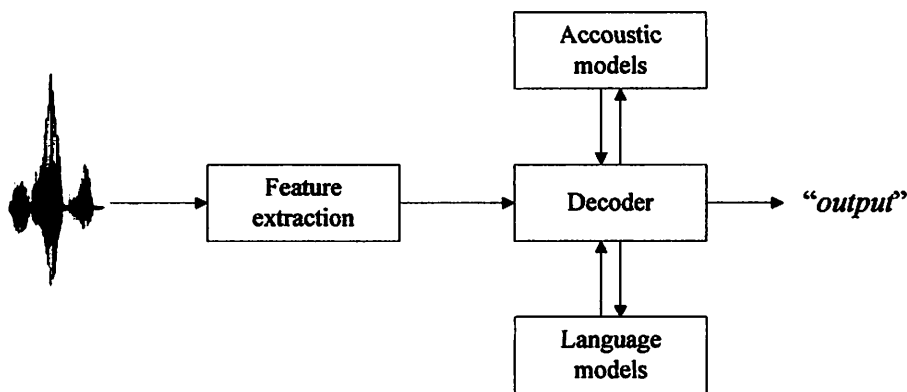
Pengenalan ucapan atau pengenalan wicara dalam istilah bahasa Inggrisnya, *Speech Recognition (SR)* adalah suatu pengembangan teknik dan sistem yang memungkinkan komputer untuk menerima masukan berupa kata yang diucapkan. Teknologi ini memungkinkan suatu perangkat untuk mengenali dan memahami kata-kata yang diucapkan dengan cara digitalisasi kata dan mencocokkan sinyal digital tersebut dengan suatu pola tertentu yang tersimpan dalam suatu perangkat. Kata-kata yang diucapkan diubah bentuknya menjadi sinyal digital dengan cara mengubah gelombang suara menjadi sekumpulan angka yang kemudian disesuaikan dengan kode-kode tertentu untuk mengidentifikasi kata-kata tersebut. Hasil dari identifikasi kata yang diucapkan dapat ditampilkan dalam bentuk tulisan atau dapat dibaca oleh perangkat teknologi sebagai sebuah

komando untuk melakukan suatu pekerjaan, misalnya penekanan tombol pada telepon genggam yang dilakukan secara otomatis dengan komando suara.

Empat dasar cara untuk melakukan speech recognition antara lain :

1. *Template based approaches*, dimana masukan *speech* yang ada dibandingkan dengan database yang ada untuk mendapatkan hasil yang paling cocok.
2. *Knowledge based approaches*, untuk mengenali *speech* dengan proses pembelajaran pada sistem.
3. *Stochastic approaches*, dimana melihat data statistika darisuatu individual *speech*.
4. *Connected approaches*, dimana menggunakan jaringan dari sejumlah contoh data, penghubung seluruh node yang ditraining untuk mengenali *speech*.

Adapun gambaran proses Speech Recognition secara umum adalah sebagai berikut :



Gambar 2.2 : Proses Speech Recognition.

Penjelasan Proses :

1. Penerimaan Masukan

Masukan berupa kata-kata yang diucapkan melalui pengeras suara atau *microphone* untuk mendapatkan gelombang suara.

2. Pengekstraksi Ciri

Tahap ini berfungsi untuk melakukan penyimpanan masukan yang berupa suara sekaligus pembuatan basis data sebagai pola. Proses ini juga dilakukan untuk mendapatkan sederetan besaran pada bagian sinyal masukan untuk menetapkan pola pembelajaran atau pola uji.

3. Pengenalan Pola

Satu atau lebih pola pembelajaran yang berhubungan dengan bunyi ucapan dari kelas yang sama, digunakan untuk membuat pola representatif dari ciri-ciri kelas tersebut. Hasilnya yang biasa disebut dengan pola referensi, dapat menjadi sebuah model yang mempunyai karakteristik bentuk statistik dari ciri-ciri pola referensi.

Pola referensi tersebut dapat menggunakan dua jenis sumber yaitu, *acoustic model* dan *linguistic model*. HMM yang digunakan untuk menangkap fitur akustik bunyi ujaran dan model bahasa menggunakan proses *stochastic* atau kumpulan variabel acak yang digunakan untuk mengetahui model linguistik.

4. Perbandingan input dengan Pola Model

Pola uji yang akan dikenali dibandingkan dengan setiap kelas pola referensi. Kesamaan besaran antara pola uji dengan setiap pola referensi akan dihitung.

Tahap ini merupakan tahap pencocokan data baru dengan data suara (pencocokan tata bahasa) pada pola. Tahap ini dimulai dengan proses konversi sinyal suara digital hasil dari proses ekstraksi ke dalam bentuk spektrum suara yang akan dianalisa dengan membandingkannya dengan pola suara pada basis data. Sebelumnya, data suara masukan dipilah-pilah dan diproses satu per satu berdasarkan urutannya. Pemilihan ini dilakukan agar proses analisis dapat dilakukan secara paralel. Proses yang pertama kali dilakukan ialah memproses gelombang kontinu spektrum suara ke dalam bentuk diskrit. Langkah berikutnya ialah proses kalkulasi yang dibagi menjadi dua bagian :

1. Transformasi gelombang diskrit menjadi data yang terurut

Gelombang diskrit berbentuk masukan berukuran n yang menjadi objek yang akan dibagi pada proses konversi dengan cara pembagian rincian waktu.

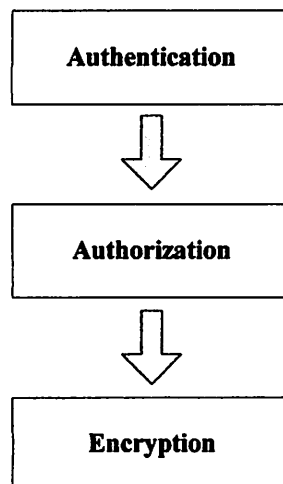
2. Menghitung frekuensi pada tiap elemen data yang terurut.

5. Logic Decision

Menentukan kelas pola referensi mana yang paling cocok untuk pola uji berdasarkan klasifikasi pola. Selanjutnya tiap elemen dari data yang terurut tersebut dikonversi ke dalam bentuk bilangan biner. Data biner tersebut nantinya akan dibandingkan dengan pola data suara dan kemudian diterjemahkan sebagai keluaran yang dapat berbentuk tulisan ataupun perintah pada perangkat.

2.15 Bluetooth

Bluetooth adalah spesifikasi industri untuk jaringan kawasan pribadi (*Personal Area Networks* atau PAN) tanpa kabel. *Bluetooth* menghubungkan dan dapat dipakai untuk melakukan tukar-menukar informasi di antara peralatan-peralatan. Spesifikasi dari peralatan *Bluetooth* ini dikembangkan dan didistribusikan oleh kelompok *Bluetooth Special Interest Group*. *Bluetooth* beroperasi dalam pita frekuensi 2,4 Ghz dengan menggunakan sebuah *frequency hopping traceiver* yang mampu menyediakan layanan komunikasi data dan suara secara *real time* antara *host-host bluetooth* dengan jarak terbatas. Kelemahan teknologi ini adalah jangkauannya yang pendek dan kemampuan transfer data yang rendah. Terdapat 3 tahap proses komunikasi *Bluetooth* seperti yang ditunjukkan pada gambar 2.3.



Gambar 2.3 : Proses Komunikasi Bluetooth.

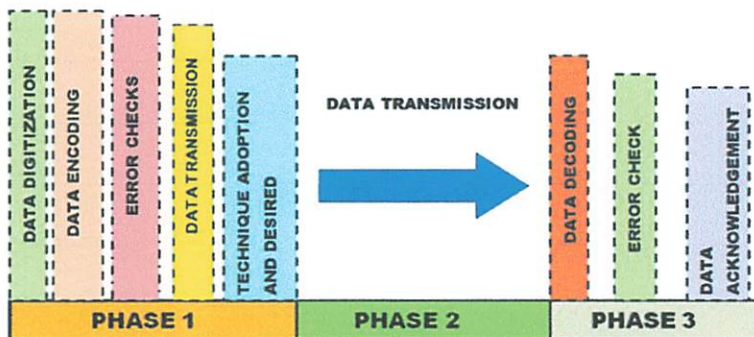
Dari gambar 2.3 menjelaskan tentang urutan dalam menggunakan komunikasi *Bluetooth* diantaranya :

1. **Authentication** : Akses hanya diberikan otomatis untuk perangkat terpercaya (yaitu, perangkat ditandai sebagai tersebut dalam database perangkat) atau perangkat tidak dipercaya setelah prosedur otorisasi. Otorisasi selalu membutuhkan otentikasi untuk memverifikasi bahwa perangkat remote adalah yang benar.
2. **Authorization** : Sebelum menghubungkan ke aplikasi, perangkat remote harus disahkan untuk mengizinkan komunikasi data.
3. **Encryption** : Link harus diubah ke mode terenkripsi, sebelum akses ke *service* ini memungkinkan. Enkripsi digunakan sebagai keamanan komunikasi yang sedang berjalan. (Opel, 2003)

2.16 Wireless Fidelity (Wi-Fi)

Wi-Fi (juga ditulis Wifi atau WiFi) adalah sebuah teknologi terkenal yang memanfaatkan peralatan elektronik untuk bertukar data secara nirkabel (menggunakan gelombang radio) melalui sebuah jaringan komputer, termasuk koneksi internet berkecepatan tinggi. *Wi-Fi Alliance* mendefinisikan *Wi-Fi* sebagai "produk jaringan wilayah lokal nirkabel (WLAN) apapun yang didasarkan pada standar *Institute of Electrical and Electronics Engineers* (IEEE) 802.11". Meski begitu, karena kebanyakan WLAN zaman sekarang didasarkan pada standar tersebut, istilah "*Wi-Fi*" dipakai dalam bahasa Inggris umum sebagai sinonim "WLAN".

Komunikasi pertukaran data pada wifi dibagi menjadi 3 fase seperti yang ditunjukkan pada gambar 2.4.



Gambar 2.4 : Proses Komunikasi WIFI.

- Tahap I : Dimana data yang disiapkan untuk transmisi itu dikodekan, berubah menjadi frame (sinyal digital yang dikirim dalam frame untuk QoS yang lebih baik). Frekuensi untuk transmisi data juga dipilih tergantung pada teknik yang digunakan untuk mengirim sinyal nirkabel.
- Tahap II : Dimana data ditransmisikan dengan udara sebagai media transmisi gelombang.
- Tahap III : Dimana data diterima, diterjemahkan, diakui dan kemudian digunakan.

Semua fase ini menerapkan beberapa teknik populer *Digital Communications Spread Spectrum* untuk sinyal *multiplexing* (*FHSS*, *Infrared*, *OFDM* dll) menggunakan metode keamanan (*WEP*, *WPA*). (Abimanyu, 2012)

2.17 Database *SQLite*

SQLite merupakan sebuah sistem manajemen basisdata relasional yang bersifat *ACID-compliant* dan memiliki ukuran pustaka kode yang relatif kecil, ditulis dalam bahasa C.

SQLite merupakan proyek yang bersifat *public domain* yang dikerjakan oleh D. Richard Hipp. Tidak seperti pada paradigma *client-server* umumnya, inti *SQLite* bukanlah sebuah sistem yang mandiri yang berkomunikasi dengan sebuah program, melainkan sebagai bagian integral dari sebuah program secara keseluruhan. Sehingga protokol komunikasi utama yang digunakan adalah melalui pemanggilan API secara langsung melalui bahasa pemrograman. Mekanisme seperti ini tentunya membawa keuntungan karena dapat mereduksi *overhead*, *latency times*, dan secara keseluruhan lebih sederhana. Seluruh elemen basisdata (definisi data, tabel, indeks, dan data) disimpan sebagai sebuah file. Kesederhanaan dari sisi disain tersebut bisa diraih dengan cara mengunci keseluruhan file basis data pada saat sebuah transaksi dimulai.

Pustaka *SQLite* mengimplementasikan hampir seluruh elemen-elemen standar yang berlaku pada SQL-92, termasuk transaksi yang bersifat atomic, konsistensi basisdata, isolasi, dan durabilitas (dalam bahasa Inggris lebih sering

disebut ACID), *trigger*, dan kueri-kueri yang kompleks. Tidak ada pengecekan tipe sehingga data bisa dientrikan dalam bentuk string untuk sebuah kolom bertipe integer. Beberapa kalangan melihat hal ini sebagai sebuah inovasi yang menambah nilai guna dari sebuah basisdata, utamanya ketika digunakan dalam bahasa pemrograman berbasis script (PHP, Perl), sementara kalangan lain melihat hal tersebut sebagai sebuah kekurangan.

Beberapa proses ataupun *thread* dapat berjalan secara bersamaan dan mengakses basisdata yang sama tanpa mengalami masalah. Hal ini disebabkan karena akses baca data dilakukan secara paralel. Sementara itu akses tulis data hanya bisa dilakukan jika tidak ada proses tulis lain yang sedang dilakukan, jika tidak, proses tulis tersebut akan gagal dan mengembalikan kode kesalahan (atau bisa juga secara otomatis akan mencobanya kembali sampai sejumlah nilai waktu yang ditentukan habis). Hanya saja ketika sebuah tabel temporer dibuat, mekanisme penguncian pada proses *multithread* akan menyebabkan masalah. *Update* yang terkini (versi 3.3.4) dikatakan telah memperbaiki masalah ini.

Sebuah program yang mandiri dinamakan *sqlite* disediakan dan bisa digunakan untuk mengeksekusi kueri dan manajemen file-file basisdata *SQLite*. Program tersebut juga merupakan contoh implementasi penulisan aplikasi yang menggunakan pustaka *SQLite*.

Integrasi *SQLite* dengan beberapa bahasa pemrograman, antara lain :

1. *SQLite* termasuk dalam *framework REALbasic*, yang memungkinkan aplikasi yang dikembangkan dengan menggunakan *REALbasic* dapat memanfaatkan basisdata *SQLite*.
2. Pustaka *SQLite* bisa digunakan secara langsung pada bahasa C/C++, namun untuk Tcl dan beberapa bahasa pemrograman berbasis *script* juga tersedia.
3. Modul DBI/DBD untuk Perl juga tersedia pada CPAN, *DBD:SQLite*, namun modul ini bukanlah antarmuka dengan *SQLite* melainkan memasukkan *SQLite* secara keseluruhan dalam modul tersebut.
4. Modul Python juga tersedia (PySQLite) yang diimplementasikan pada DB API Python versi 2.0 (PEP 249).

5. PHP dimulai dengan PHP5 telah memasukkan *SQLite*, versi PHP4 sebelumnya bisa juga digunakan untuk mengakses *SQLite*, namun modul *SQLite* tidak dimasukan secara standar bawaan.
6. Dimulai Lazarus versi 0.9.8 dan Free Pascal 2.0.0, *SQLite* didukung untuk digunakan oleh para *programmer*. Tutorial tersebut bisa didapatkan disini.
7. Meskipun borland tidak mempaketkan *SQLite* secara standar bawaan, Delphi sudah mendukung *SQLite* juga menggunakan pustaka yang dibuat oleh pihak ketiga (Aducom dan Zeos). Banyak program administrasi basisdata *SQLite* baik komersil ataupun bebas yang menawarkan kemudahan-kemudahan dalam memanajemen basisdata *SQLite* terdapat di internet dibuat dengan menggunakan bahasa ini.
8. *SQLite* juga termasuk dalam paket yang dibundle secara standar bawaan pada Mac OS X, dan digunakan sebagai salah satu pilihan mekanisme penyimpanan data pada API Apple. (Newman, 2005)

BAB III

ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Sistem

Untuk membuat sistem yang akan dibangun dapat dideskripsikan kedalam fungsi-fungsi yang menjadi kebutuhan dari sistem. Adapun fungsi-fungsi yang dibutuhkan akan dijelaskan pada kebutuhan fungsional dan kebutuhan non fungsional.

3.1.1 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional menggambarkan proses kegiatan yang akan diterapkan dalam sebuah sistem dan menjelaskan kebutuhan yang diperlukan sistem agar sistem berjalan dengan baik sesuai kebutuhan.

1. Aplikasi menampilkan pilihan-pilihan menu ataupun perintah yang jelas dan mudah dimengerti oleh pengguna agar bisa menjalankan sistem utama dari aplikasi.
2. Aplikasi mempunyai tombol “?” yang merupakan menu bantuan untuk menggunakan aplikasi.
3. Aplikasi Android (*client*) dapat melakukan konektifitas dengan aplikasi *server* menggunakan *Bluetooth* atau *Wifi*.
4. *Database* harus mempunyai *default command* yang otomatis tersimpan saat program pertama kali dijalankan dan dapat diubah melalui menu *Setting*.
5. Inputan suara dengan fitur *Speech Recognition* yang sesuai dengan *command* yang tersimpan di dalam *Database* dapat mengendalikan *slide presentasi* yang berjalan pada komputer *server*.
6. *Google Speech API* dibutuhkan untuk mendukung fitur *Speech Recognition*.
7. Aplikasi *Server* dapat menerima masukan perintah yang dikirimkan oleh Aplikasi Android (*client*) dan memproses perintah tersebut.

3.1.2 Analisis Kebutuhan Non-Fungsional

Pada analisis kebutuhan non-fungsional ini dijelaskan analisis kebutuhan perangkat keras, analisis kebutuhan perangkat lunak, dan analisis pengguna.

3.1.2.1 Analisis Kebutuhan Perangkat Keras (*Hardware*)

1. Spesifikasi minimum perangkat keras yang dibutuhkan oleh pengembang untuk membangun aplikasi VuDroid Wispa adalah :
 - a. Prosesor dengan kecepatan 2.0 GH
 - b. RAM 2 GB
 - c. HDD 100 GB
 - d. VGA Card 256MB
 - e. Wireless Network Adaptor (Bluetooth & Wifi)
 - f. Monitor
 - g. Mouse dan Keyboard
2. Spesifikasi minimal perangkat keras *smartphone* yang digunakan adalah sebagai berikut :
 - a. CPU : 1 GHz
 - b. Internal Memory : 4 GB, 512 RAM
 - c. Android OS 4.1 (Jelly Bean)
 - d. Bluetooth & Wifi Adaptor
 - e. Internet Connection

3.1.2.2 Analisis Kebutuhan Perangkat Lunak (*Software*)

1. Perangkat lunak yang dibutuhkan oleh pengembang untuk membangun aplikasi VuDroid Wispa adalah :
 - a. Sistem Operasi Windows 7
 - b. NetBeans IDE
 - c. Android Studio
 - d. *Android Software Development Kit* (Android SDK)
 - e. *Java Development Kit* (JDK)
 - f. *Android Development Tools* (ADT)
 - g. Gradle
 - h. *SQLite Database*
2. Perangkat lunak yang dibutuhkan untuk *smartphone* agar aplikasi VuDroid Wispa bisa berjalan secara maksimal antara lain :
 - a. Android OS 4.1 (Jelly Bean)
 - b. *Google Voice Search*

3.1.2.3 Analisis Pengguna (*Brainware*)

Dalam menjalankan aplikasi tentu tak lepas dari peran pengguna, berikut adalah kebutuhan pengguna yang diperlukan untuk menjalankan aplikasi berbasis android ini :

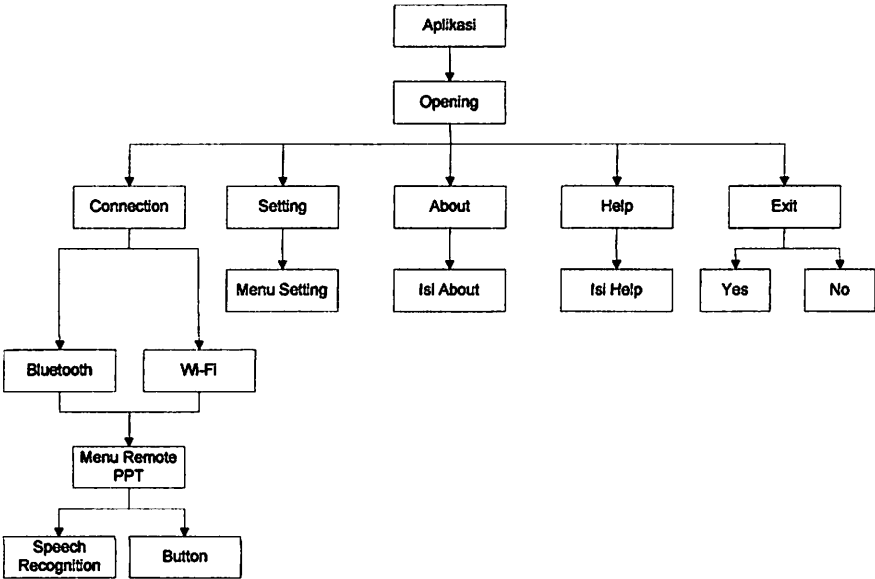
- 1. Pemanfaatan fitur *Speech Recognition* hanya bisa digunakan oleh orang yang mampu berbicara dengan jelas, yaitu selain penderita Tuna Wicara dan Tuna Rungu.
- 2. Pengguna umum bisa mengoperasikan *smartphone* Android.

3.2 Perancangan Sistem

Perancangan sistem adalah bagian dari metodologi pengembangan suatu perangkat lunak yang dilakukan untuk memberikan gambaran secara terperinci. Tahap ini merupakan langkah awal yang menentukan bagaimana alur dan tampilan pada aplikasi *remote presentation* VuDroid Wispa. Langkah awal ini merupakan langkah yang sangat menentukan tingkat kesuksesan dalam pembuatan sistem.

3.2.1 Perancangan Struktur Menu

Perancangan struktur menu adalah perancangan tata urutan menu dari aplikasi *remote presentation* VuDroid Wispa. Rancangan struktur menu program Android (*client*) VuDroid Wispa ditunjukkan pada gambar 3.1.



Gambar 3.1 : Rancangan Struktur Menu.

Dalam perancangan struktur menu pada gambar 3.1, program diawali dengan opening *splash screen* yang kemudian dilanjutkan masuk ke menu utama dimana di sini terdapat 3 *button* yaitu *Connection*, *Setting*, dan *About*, serta 2 menu yaitu *Help* dan *Exit*.

Program utama ada di dalam menu *Connection*, terdapat 2 pilihan koneksi yaitu *Bluetooth* atau *Wi-Fi*, setelah pemilihan koneksi akan masuk ke tampilan menu pemilihan jenis remote ppt antara *Speech Recognition* atau *Button*.

Selanjutnya adalah menu *Setting*, menu ini memiliki 2 fungsi yaitu melakukann perubahan pada perintah untuk memproses inputan *speech recognition* yang tersimpan dalam database, fitur lainnya adalah untuk melakukan perubahan pada tema *background* aplikasi. Selain itu juga terdapat menu untuk menuju pengaturan bahasa yang akan digunakan pada fitur *speech recognition*.

Menu ketiga adalah menu *About*, menu ini menjelaskan tentang program dan pembuat program. Menu keempat adalah menu *Help*, menu ini menampilkan petunjuk penggunaan aplikasi, serta fitur-fitur yang bisa digunakan oleh pengguna. Menu terakhir adalah menu *Exit*, menu ini digunakan apabila pengguna ingin keluar dari program, menu *exit* akan menutup semua proses yang berjalan dan akan menutup program.

3.2.2 Struktur Basis Data

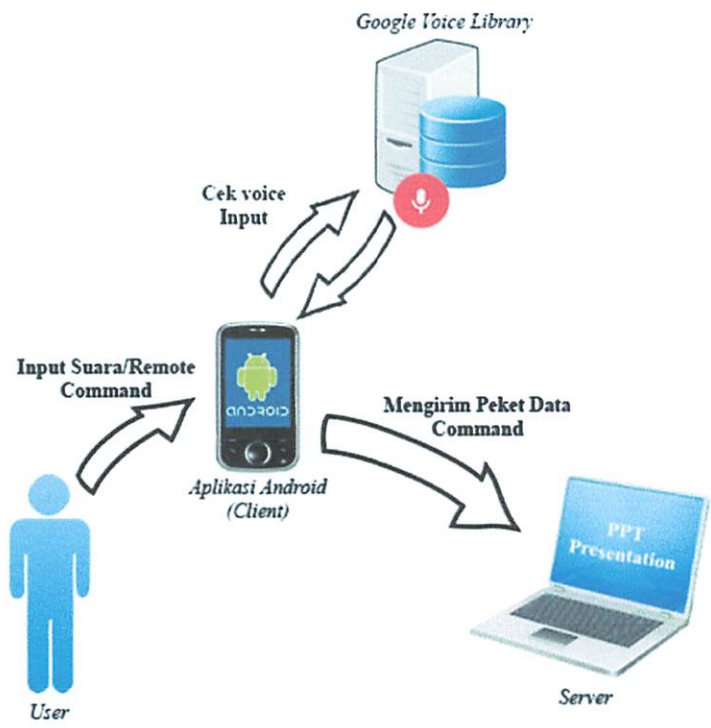
Struktur basis data untuk penyimpanan data pada aplikasi Android (*client*) adalah sebagai berikut :

Table 3.1 Struktur Basis Data *Command*

Kolom	Tipe Data	Key
ROW_ID	Int	Primary Key
COMMAND	Text	-

3.2.3 Proses Interaksi

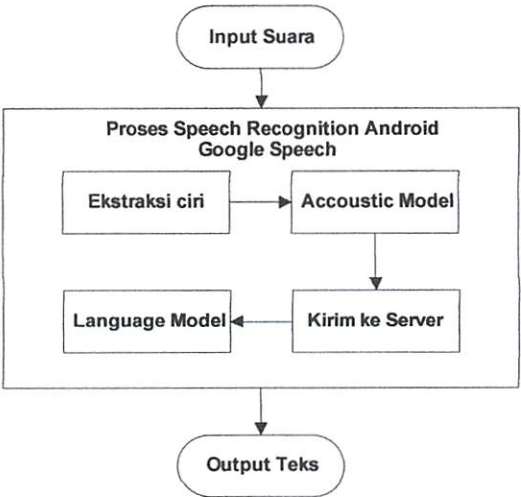
Proses interaksi memaparkan tentang gambaran fungsionalitas dari suatu sistem, sehingga *user* paham dan mengerti mengenai kegunaan aplikasi yang akan dibangun yang ditunjukkan pada gambar 3.2.



Gambar 3.2 : Proses Interaksi.

3.2.4 Proses *Speech Recognition*

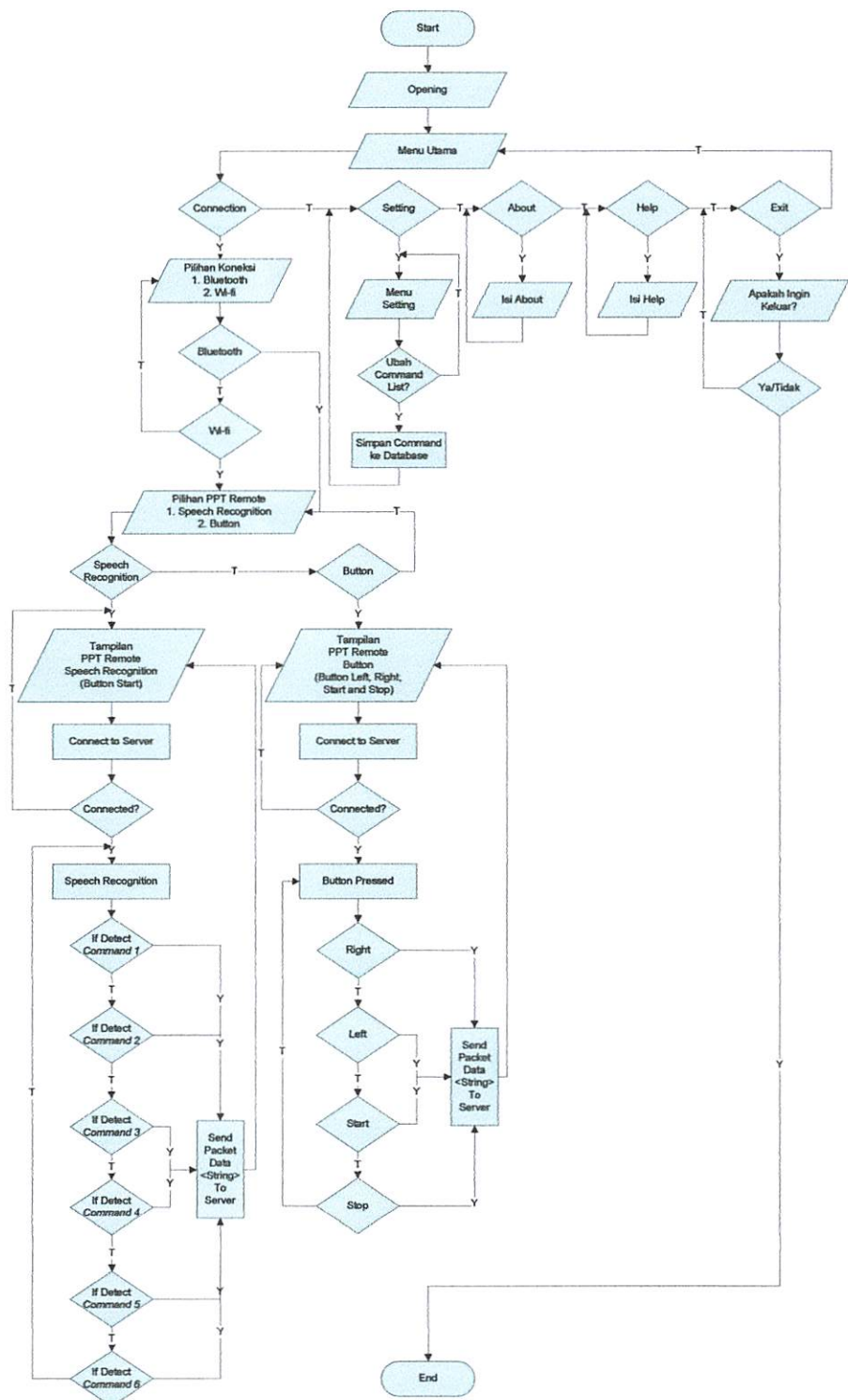
Blok diagram ini berfungsi untuk menggambarkan proses *speech recognition* atau pengenalan ucapan yang dilakukan *Google Speech* bagaimana inputan suara diproses sehingga dapat menghasilkan *output* berupa teks digital. Blok diagram ditunjukkan pada gambar 3.3.



Gambar 3.3 : Proses *Speech Recognition* Google.

3.2.5 Flowchart

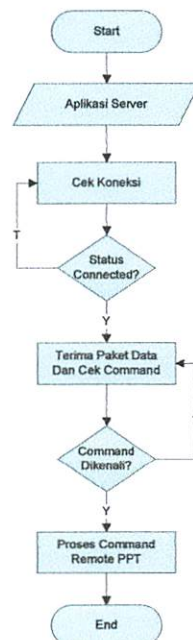
Alur sistem ini menjelaskan tentang proses berjalannya aplikasi yang ditunjukkan dengan *flowchart* pada gambar 3.4.



Gambar 3.4 : Flowchart Aplikasi Android (Client).

Seperti yang terlihat pada gambar 3.4, program dimulai dari *start* program yang kemudian dilanjutkan pada menu utama yaitu menu tampilan untuk memilih *Connection*, *Setting*, *About*, *Help* serta *Exit*. Pada menu *Setting* akan terdapat beberapa pengaturan untuk menentukan *command* dan bahasa yang akan digunakan pada fitur *Speech Recognition*. Menu *About* dan *Help* menampilkan informasi tentang program, pembuat program serta informasi cara menggunakan program. Kemudian opsi *Exit* digunakan apabila pengguna ingin keluar dan mengakhiri program.

Apabila pengguna memilih menu *Connection*, maka akan menampilkan tampilan pemilihan koneksi yang ingin digunakan, yaitu antara koneksi *Bluetooth* atau koneksi *Wifi*. Setelah menentukan koneksi yang diinginkan, pilihan selanjutnya adalah pilihan *remote* program yaitu menggunakan *speech recognition* atau *button remote*. Selanjutnya pengguna harus mengkoneksikan dengan aplikasi *server* terlebih dahulu yaitu dengan melakukan *scanning Bluetooth* (koneksi *Bluetooth*) atau pun dengan meng-inputkan *IP server* (koneksi *Wifi*). Jika koneksi dengan server berhasil, maka program akan mengirimkan paket data perintah ke aplikasi *server*. Proses penerimaan paket data oleh *server* ditunjukkan dalam bentuk *flowchart* pada gambar 3.5.



Gambar 3.5 : Flowchart Aplikasi Server.

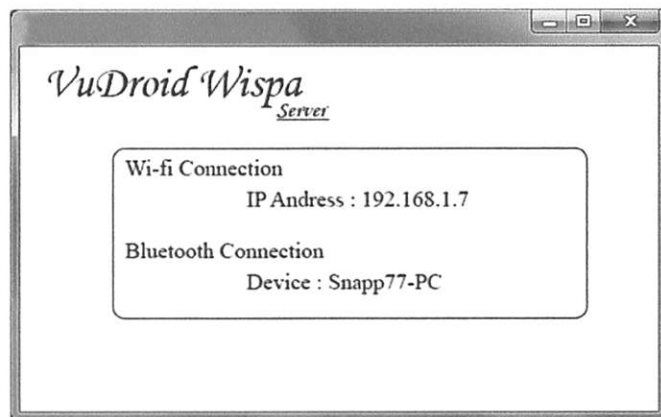
Pada gambar 3.5 dijelaskan saat aplikasi *server* berjalan, program akan melakukan cek koneksi apakah sudah terhubung dengan aplikasi Android *client* atau belum. Jika sudah terkoneksi, maka program akan menunggu untuk menerima paket data *command* yang dikirimkan dari aplikasi *client* untuk diproses sebagai pengendali presentasi seperti pengendalian untuk *start slide*, *stop slide*, *next slide*, *previous slide*, *first slide* dan *last slide*.

3.2.6 Perancangan Interface

Perancangan *Interface* bertujuan untuk memberikan gambaran tentang bentuk tampilan dari aplikasi sesuai dengan tata letak menu program *remote presentation* yang akan dibuat.

1. Perancangan Aplikasi Server

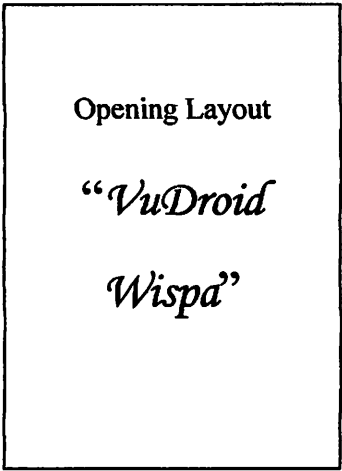
Aplikasi *server* berfungsi sebagai penerima dan pemroses perintah yang didapatkan dari Aplikasi Android (*client*) untuk mengendalikan presentasi pada Ms. Power Point yang dijalankan di *notebook* atau *PC*. Rancangan aplikasi *server* ditunjukkan pada gambar 3.6.



Gambar 3.6 : Perancangan Tampilan Aplikasi Server.

2. Opening Layout

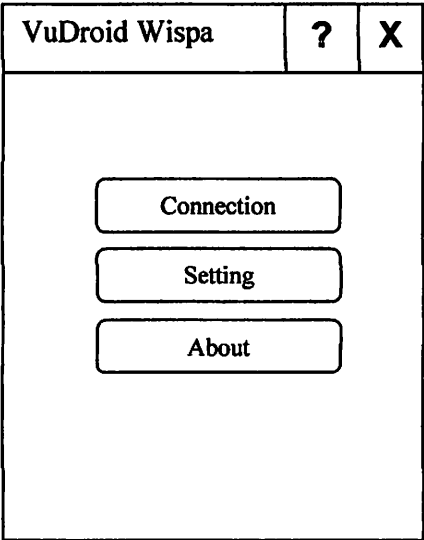
Opening layout merupakan tampilan awal ketika aplikasi ini dijalankan. Pada layout ini hanya berisi gambar pengenalan dari aplikasi android *client* seperti ditunjukkan pada gambar 3.7.



Gambar 3.7 : Perancangan Tampilan Opening Aplikasi Android (Client)

3. Interface Menu Utama

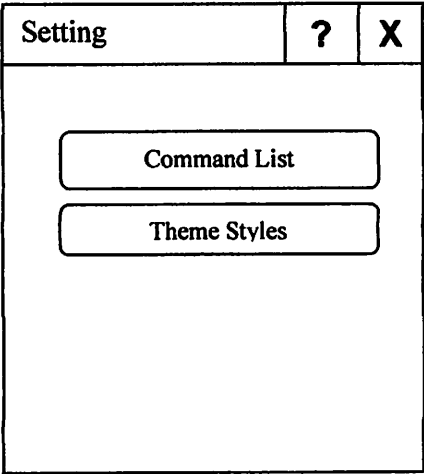
Interface menu utama adalah tampilan menu setelah *Opening layout*. Berisi tentang menu inti dari aplikasi *client* seperti *connection*, *setting*, *about*, *help* dan *exit*. Menu *Connection* digunakan untuk memulai program, yaitu menampilkan tampilan pilihan koneksi yang ingin digunakan antara *Bluetooth* atau *Wifi*. Menu *Setting* untuk menampilkan *interface setting* program. Menu *About* dan “?”/*Help* menampilkan *interface About* serta *interface Help*. Menu “X”/*Exit* digunakan untuk keluar dan menghentikan program. Adapun tampilan menu utama seperti ditunjukkan pada gambar 3.8.



Gambar 3.8 : Perancangan Tampilan Menu Utama.

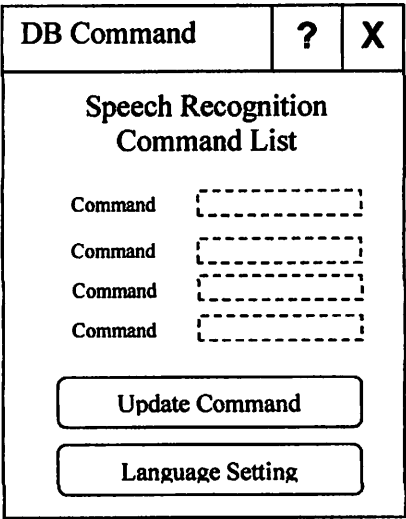
4. *Interface Setting*

Interface ini menampilkan pengaturan program, yaitu pengaturan *Command List* yang nantinya digunakan untuk menyesuaikan *command* yang di-inputkan melalui fitur *speech recognition*. *Interface Setting* ditunjukkan pada gambar 3.9.



Gambar 3.9 : Perancangan Tampilan Setting.

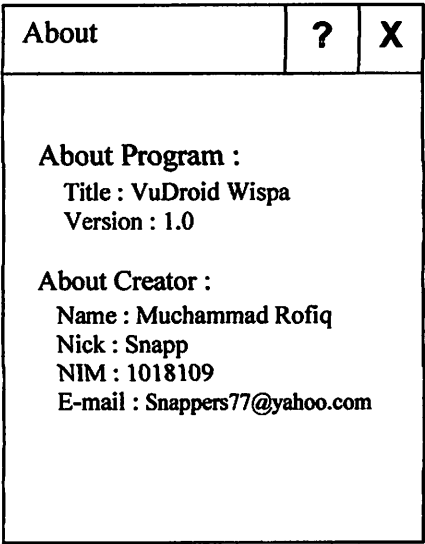
Tombol *Command List* menampilkan data perintah yang akan dieksekusi oleh fitur *speech recognition* yang disimpan dalam database. Data awal yang tersimpan adalah *next*, *back*, *start* dan *stop*. Dalam pengaturan ini juga terdapat tombol untuk memperbarui data *command* serta tombol untuk mengubah bahasa yang digunakan untuk fitur *speech recognition*. Tampilan *command list* yang tersimpan dalam database ditunjukkan pada gambar 3.10.



Gambar 3.10 : Perancangan Tampilan Command List.

5. *Interface About*

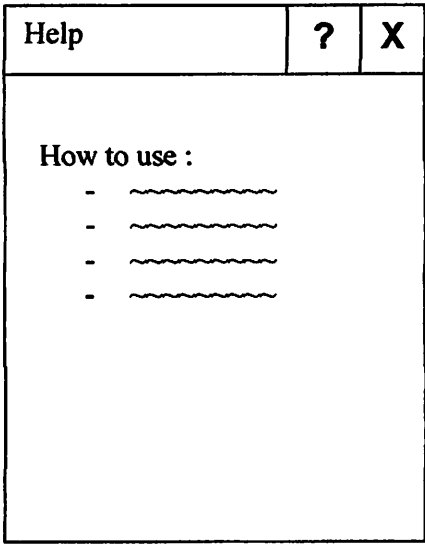
Interface About menampilkan informasi program dan informasi tentang pembuat program. Perancangan *interface about* ditunjukkan oleh gambar 3.11.



Gambar 3.11 : Perancangan Tampilan About.

6. *Interface Help*

Interface “?”/Help menampilkan informasi cara menggunakan program serta informasi fitur-fitur program. Perancangan *interface help* ditunjukkan oleh gambar 3.12.



Gambar 3.12 : Perancangan Tampilan Help.

7. *Interface Menu Connection dan Menu Remote*

Tampilan yang berada dalam menu *connection* adalah menu pemilihan koneksi. Ada 2 jenis koneksi yang bisa digunakan untuk menghubungkan aplikasi *client* dengan *server*, yaitu antara *Bluetooth* atau *Wi-Fi*. Setelah pemilihan koneksi, program akan berlanjut ke pemilihan jenis pengendalian(*remote*) yang akan digunakan yaitu *Speech Recognition* atau *Button*. Tampilan perancangan pemilihan koneksi dan *remote* menu ditunjukkan pada gambar 3.13 dan gambar 3.14.

Connection Menu	?	X
Select Connection : Bluetooth Wireless Fidelity		

Gambar 3.13 : Perancangan Tampilan Menu Connection.

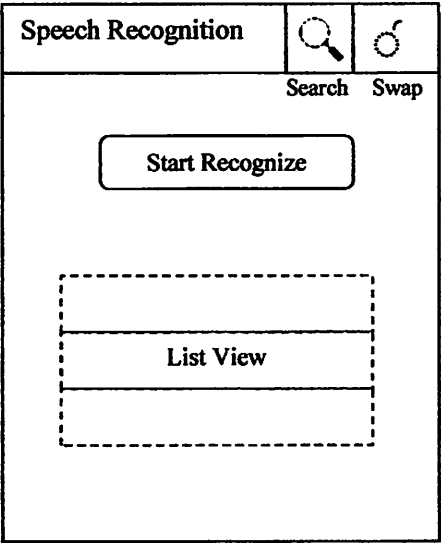
WiFi Connection	?	X
Remote Menu : Speech Recognition Button Remote		

BT Connection	?	X
Remote Menu : Speech Recognition Button Remote		

Gambar 3.14 : Perancangan Tampilan Remote Menu.

8. *Interface Program Remote PPT Speech Recognition*

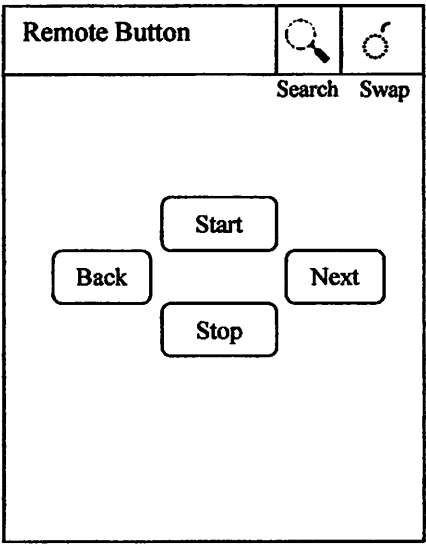
Interface ini menampilkan tombol-tombol dan fitur yang bisa digunakan untuk mengendalikan ppt presentasi berdasarkan inputan suara dengan cara mengirimkan paket data perintah ke aplikasi *server* untuk diproses lebih lanjut. Tampilan *remote Speech Recognition* ditunjukkan pada gambar 3.15.



Gambar 3.15 : Perancangan Tampilan Speech Recognition.

9. *Interface Program Remote PPT Button Remote*

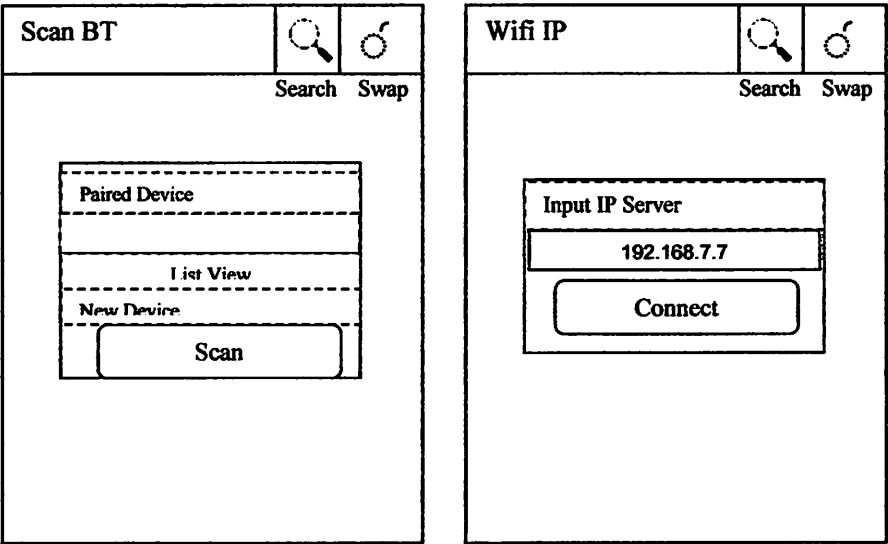
Interface ini menampilkan tombol-tombol dan fitur yang bisa digunakan untuk mengendalikan ppt presentasi berdasarkan tombol yang ditekan. Tampilan *Button Remote* ditunjukkan pada gambar 3.16.



Gambar 3.16 : Perancangan Tampilan Button Remote.

10. *Interface Search Scan Bluetooth dan Input IP Server (Wifi)*

Interface ini muncul ketika tombol *Search* ditekan, berfungsi untuk menampilkan daftar *Bluetooth* atau memasukkan *IP Wifi* untuk mengkoneksikan ke aplikasi *server* sebelum menggunakan *remote* ppt. Fitur ini harus dilakukan terlebih dahulu agar aplikasi *client* dapat mengirimkan paket data perintah ke aplikasi *server* untuk diproses sebagai pengendali presentasi yang berjalan pada komputer *server*. Tampilan *Search* ditunjukkan pada gambar 3.17.



Gambar 3.17 : Perancangan Tampilan Search Scan Bluetooth dan Input IP.

BAB IV

IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi

Implementasi bertujuan untuk menerapkan perancangan yang telah dilakukan terhadap sistem sehingga *user* dapat memberi masukan demi berkembangnya sistem yang telah dibangun. Perangkat keras dan perangkat lunak yang diperlukan dalam pembuatan aplikasi ini adalah sebagai berikut :

4.1.1 Implementasi Perangkat Keras

Perangkat keras yang diperlukan untuk pembuatan aplikasi ini merupakan komputer dengan spesifikasi yang cukup tinggi. Hal ini disebabkan pada penggunaan *resource* CPU dan *memory* yang cukup banyak. Kebutuhan spesifikasi komputer yang tinggi hanya dibutuhkan oleh pembuat aplikasi saja, namun untuk penggunaanya cukup dengan komputer dengan spesifikasi sedang. Adapun spesifikasi komputer pendukung yang digunakan untuk pembuatan aplikasi ini dapat dilihat pada tabel 4.1.

Tabel 4.1 Implementasi Perangkat Keras

<i>Processor</i>	Intel Core i5-2410M @2.30GHz
RAM	4GB
HDD	500GB
Resolusi Layar	1366 x 768
<i>Keyboard</i>	1
<i>Mouse</i>	1
<i>Bluetooth & Wifi Adaptor</i>	YA

4.1.2 Implementasi Perangkat Lunak

Perangkat lunak yang digunakan untuk membuat aplikasi *remote presentation* VuDroid Wispa ini adalah sebagai berikut :

1. Sistem Operasi : Microsoft Windows 7
2. Android Studio
3. NetBeans IDE
4. Adobe Photoshop CS6.

4.1.3 Implementasi Aplikasi

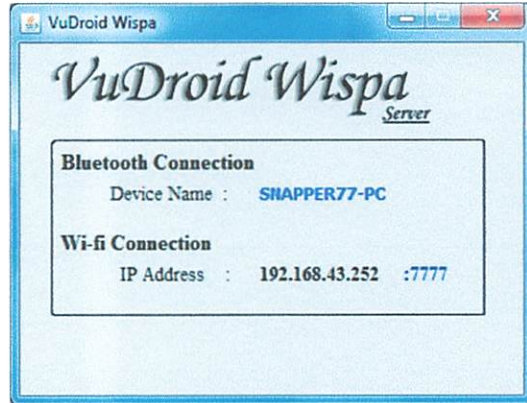
Aplikasi *remote presentation* VuDroid Wispa ini merupakan sebuah aplikasi untuk mengendalikan *slide* presentasi yang berjalan pada MS. Power Point. Aplikasi *client* perlu diinstall pada android *smartphone* dengan OS Android 4.1 (Jelly Beans). Untuk aplikasi *server* tidak perlu dilakukan instalasi, cukup dengan melakukan *double click* pada aplikasi ini (*jar*). Akan tetapi perlu diingat, pada sistem operasi Windows harus sudah terinstall Java JRE (*Java Runtime Environment*) agar aplikasi *server* dapat berjalan. Apabila Java JRE belum terpasang, maka aplikasi *server* tidak dapat dijalankan karena file aplikasi *server* ini berekstensi *.jar* yang akan berjalan dengan dukungan Java JRE.

4.1.4 Implementasi Aplikasi Server

Aplikasi *server* berfungsi sebagai penerima dan pemroses perintah yang didapatkan dari aplikasi Android (*client*) untuk mengendalikan presentasi pada Ms. Power Point yang dijalankan di *notebook* atau *PC*. Aplikasi *server* bekerja dengan cara menerima *command* atau perintah melalui jaringan *Bluetooth* atau *Wifi* bertipe *string*, yang kemudian akan diproses dan dikonversi dari *string* menjadi *key_event* pada Java untuk robot pengendalian tombol *keyboard*.

```
Robot robot = new Robot();
switch (command) {
    case KEY_RIGHT:
        robot.keyPress(KeyEvent.VK_DOWN);
        break;
    case KEY_LEFT:
        robot.keyPress(KeyEvent.VK_UP);
        break;
    case KEY_START:
        robot.keyPress(KeyEvent.VK_F5);
        break;
    case KEY_STOP:
        robot.keyPress(KeyEvent.VK_ESCAPE);
        break;
    case KEY_FIRST:
        robot.keyPress(KeyEvent.VK_HOME);
        break;
    case KEY_LAST:
        robot.keyPress(KeyEvent.VK_END);
        break;
}
```

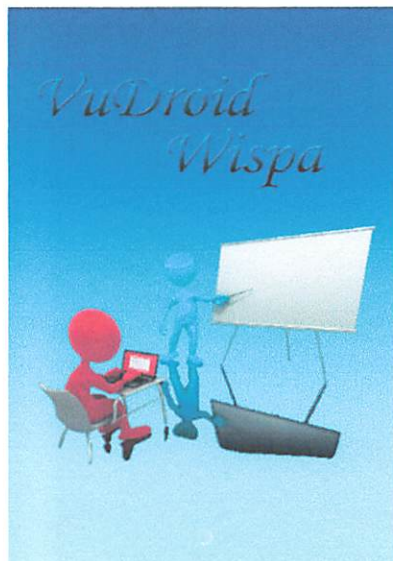
Tombol konversi *keyboard* yang digunakan adalah tombol umum yang digunakan pada aplikasi untuk presentasi seperti Ms. Power Point yaitu seperti tombol *arrow down*, *arrow up*, F5, Esc, Home dan tombol End. Tampilan aplikasi *server* ditunjukkan pada gambar 4.1.



Gambar 4.1 : Tampilan Aplikasi Server.

4.1.5 Implementasi Opening Program

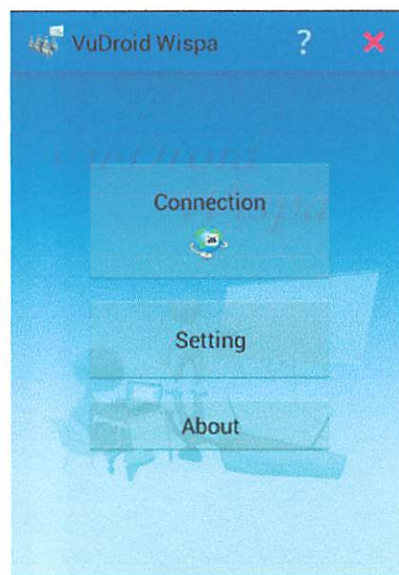
Opening Program adalah tampilan awal ketika program Android *client* dijalankan. Tampilan ini hanya muncul selama beberapa detik saja sebelum berpindah ke tampilan program selanjutnya, untuk mempercepat proses tampilan *opening* juga bisa dilakukan dengan menekan layar lcd dari *smartphone*. Pada tampilan ini hanya berisi gambar dan nama program sebagai pengenalan dari aplikasi android *client* yang ditunjukkan pada gambar 4.2.



Gambar 4.2 : Tampilan Opening Program.

4.1.6 Implementasi Menu Utama

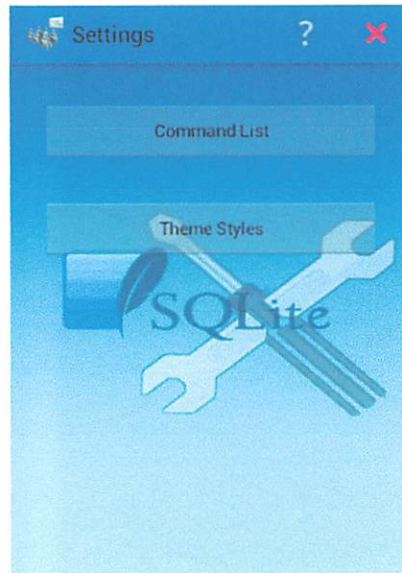
Menu Utama adalah tampilan menu setelah *Opening Program*. Berisi tentang menu inti dari aplikasi *client* seperti *connection*, *setting*, *about*, *help* dan *exit*. Menu *Connection* digunakan untuk memulai program, yaitu menampilkan tampilan pilihan koneksi yang ingin digunakan antara *Bluetooth* atau *Wifi*. Menu *Setting* untuk menampilkan *setting* program. Menu *About* menampilkan informasi program dan informasi tentang pembuat program. Menu “?”/*Help* menampilkan informasi fitur-fitur dan cara penggunaan program. Menu “X”/*Exit* digunakan untuk keluar dan menghentikan program. Tampilan menu utama ditunjukkan seperti pada gambar 4.3.



Gambar 4.3 : Tampilan Menu Utama.

4.1.7 Implementasi Menu Setting

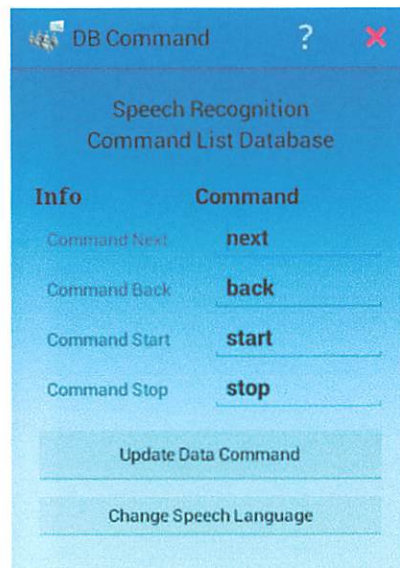
Menu *Setting* atau pengaturan program menampilkan 2 pilihan pengaturan yaitu pengaturan *Command List* dan pengaturan *Theme Styles*. Tampilan Menu *Setting* ditunjukkan pada gambar 4.4.



Gambar 4.4 : Tampilan Menu Setting.

4.1.7.1 Setting Command List

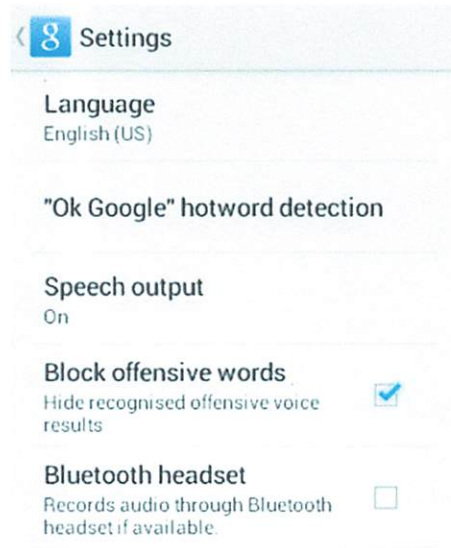
Pengaturan ini digunakan untuk menyesuaikan *command* atau perintah yang di-inputkan melalui fitur *speech recognition* dan data *command* tersebut akan disimpan dalam database program. Tampilan untuk merubah data *command list* ditunjukkan pada gambar 4.5.



Gambar 4.5 : Tampilan Setting Command List.

Data pada kolom “**Command**” merupakan data yang tersimpan dalam database dan untuk mengubahnya hanya perlu merubah isi *command* kemudian

menekan tombol *Update Data Command* untuk memperbarui database. Data *command* atau perintah yang digunakan sebaiknya disesuaikan dengan bahasa yang diinginkan. Tombol *Change Speech Language* akan menampilkan pengaturan bahasa yang ingin digunakan untuk fitur *speech recognition* seperti ditampilkan pada gambar 4.6.



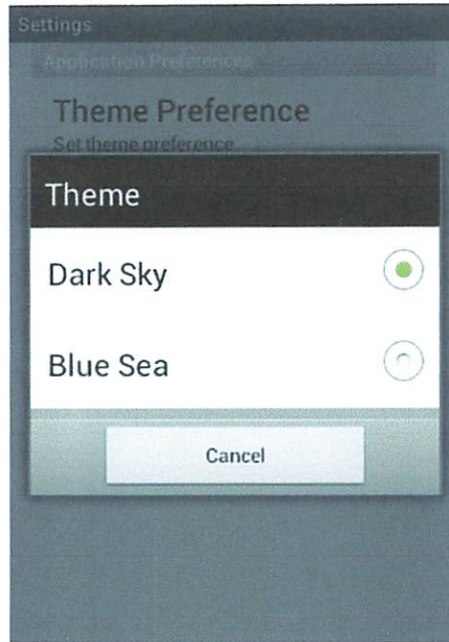
Gambar 4.6 : Tampilan Pengaturan Bahasa.

4.1.7.2 *Setting Theme Styles*

Pengaturan ini digunakan untuk mengubah tema tampilan program. Fitur ini berfungsi untuk merubah keseluruhan *background* dan warna teks program. Ada 2 pilihan tema yang digunakan yaitu :

1. Tema "*Dark Sky*" : Tema dengan dominan *background* berwarna hitam dan teks berwarna putih.
2. Tema "*Blue Sea*" : Tema dengan dominan *background* berwarna biru dan teks berwarna hitam.

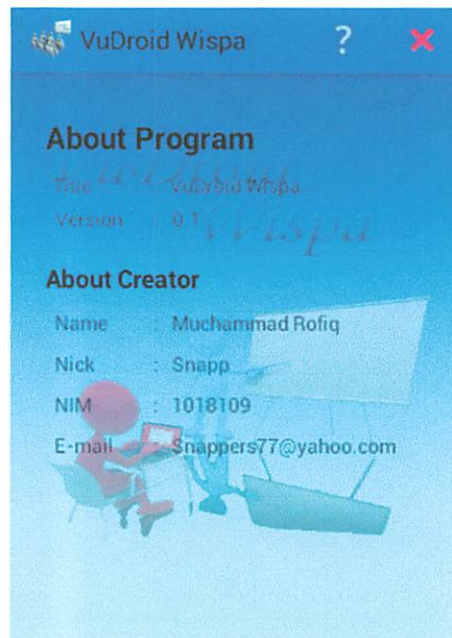
Adapun tampilan *Setting Theme Styles* ditunjukkan pada gambar 4.7.



Gambar 4.7 : Tampilan Setting Theme Styles.

4.1.8 Implementasi Menu *About*

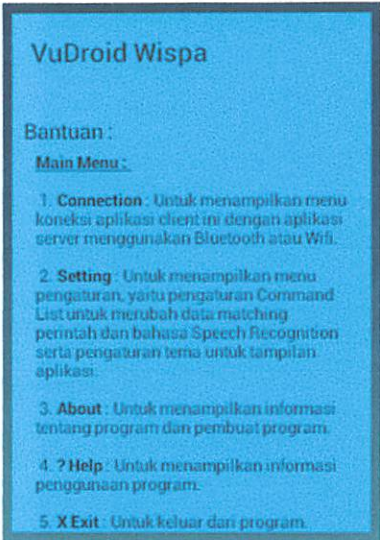
Menu *About* menampilkan informasi program dan informasi tentang pembuat program. Tampilan menu *About* ditunjukkan seperti pada gambar 4.8.



Gambar 4.8 : Tampilan Menu About.

4.1.9 Implementasi Menu *Help*

Menu “?”/*Help* menampilkan informasi fitur-fitur dan cara penggunaan program. Tampilan menu *Help* ditunjukkan oleh gambar 4.9.



Gambar 4.9 : Tampilan Menu “?” *Help*.

4.1.10 Implementasi Menu *Connection*

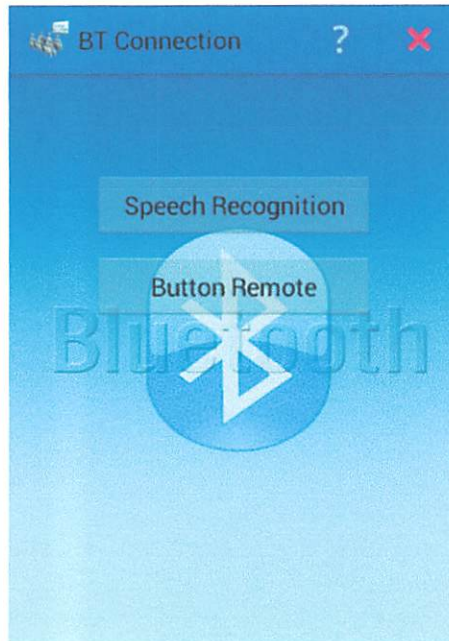
Tampilan yang ada dalam menu *Connection* adalah menu pemilihan koneksi. Ada 2 jenis koneksi yang bisa digunakan untuk menghubungkan aplikasi *client* dengan *server*, yaitu antara *Bluetooth* atau *Wi-Fi*. Tampilan menu *Connection* ditunjukkan seperti pada gambar 4.10.



Gambar 4.10 : Tampilan Menu *Connection*.

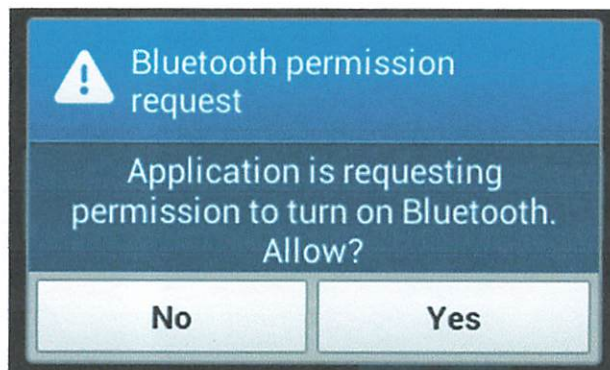
4.1.10.1 Bluetooth Connection

Pada pemilihan koneksi *bluetooth*, program akan berlanjut ke pemilihan jenis pengendalian (*remote*) yang bisa dipilih yaitu *Speech Recognition* atau *Button Remote*. Pemilihan menu *Bluetooth Connection* ditampilkan pada gambar 4.11.



Gambar 4.11 : Tampilan Bluetooth Connection.

Sebelumnya untuk memasuki pilihan *remote Bluetooth* yang ditunjukkan pada gambar 4.11 di atas, pengguna diharuskan untuk mengaktifkan *device Bluetooth* pada *smartphone* terlebih dahulu. Adapun tampilan permintaan untuk mengaktifkan *Bluetooth* ditunjukkan pada gambar 4.12.



Gambar 4.12 : Tampilan pengaktifan device Bluetooth.

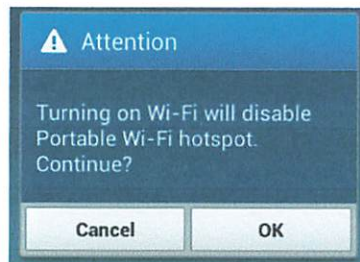
4.1.10.2 Wifi Connection

Pada pemilihan koneksi *wifi*, program akan berlanjut ke pemilihan jenis pengendalian (*remote*) yang bisa dipilih sama seperti *Bluetooth Connection* yaitu *Speech Recognition* atau *Button Remote*. Pemilihan menu *Wifi Connection* ditampilkan pada gambar 4.13.



Gambar 4.13 : Tampilan Wifi Connection.

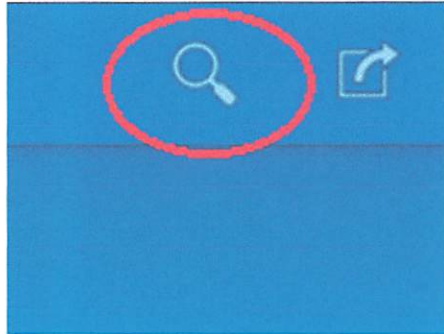
Memilih *Wifi Connection* akan otomatis mengaktifkan *device wifi* untuk dikoneksikan ke media penghubung seperti *Access Point* untuk mendapatkan alamat IP. Pengguna juga bisa memanfaatkan fitur “*Tethering and Portable Hotspot*” yang berfungsi untuk menjadikan android *smartphone* menjadi seperti *Access Point*. Perlu diingat bahwa jika fitur “*Tethering and Portable Hotspot*” telah aktif dan memilih *Wifi Connection*, maka program akan mengkonfirmasi apakah akan mengaktifkan *device wifi* atau tidak karena fitur *wifi* dan “*Tethering and Portable Hotspot*” tidak bisa dijalankan bersamaan. Tampilan konfirmasi tersebut ditunjukkan pada gambar 4.14.



Gambar 4.14 : Tampilan Konfirmasi Wifi.

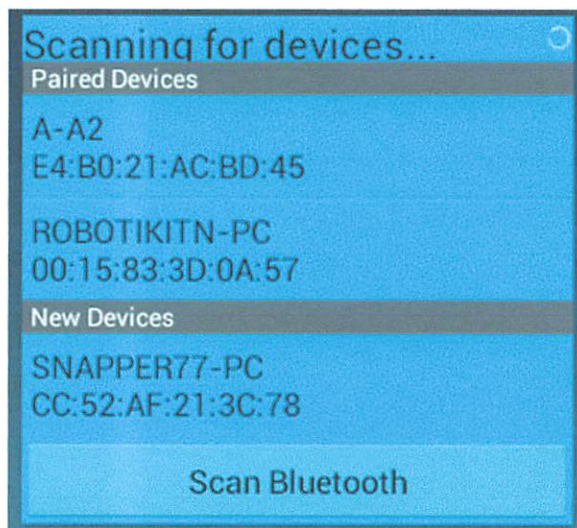
4.1.11 Implementasi Koneksi ke Server

Didalam menu pemilihan *remote* seperti menu *Speech Recognition* atau *Button Remote*, terdapat tombol “Search” yang berfungsi untuk menghubungkan antara aplikasi Android *client* dengan aplikasi yang berjalan pada komputer *server*. Konektifitas aplikasi harus dihubungkan terlebih dahulu agar *command* atau perintah dapat dikirimkan sehingga *remote presentation* berjalan. Tampilan tombol “Search” ditunjukkan pada gambar 4.15.



Gambar 4.15 : Tampilan Tombol Search.

Tampilan koneksi penghubung antara *Bluetooth* dan *wifi* juga berbeda, pada koneksi menggunakan *Bluetooth* tombol “Search” akan menampilkan daftar *device Bluetooth* yang pernah terhubung ataupun *device Bluetooth* yang baru ditemukan dengan menekan tombol *Scan*. Tampilan *Search Bluetooth* ditunjukkan oleh gambar 4.16.



Gambar 4.16 : Tampilan Search Scan Bluetooth.

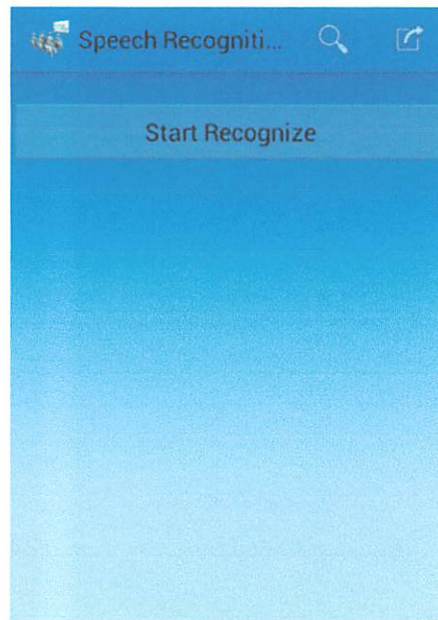
Pada koneksi menggunakan *wifi*, tombol “*Search*” akan menampilkan dialog untuk menginputkan alamat IP sesuai dengan IP yang ditampilkan aplikasi *server* dan menekan tombol *Connect* untuk menghubungkan antara *client* dan *server*. Tampilan *Search Wifi* ditunjukkan oleh gambar 4.17.



Gambar 4.17 : Tampilan Search Input IP Server.

4.1.12 Implementasi *Speech Recognition*

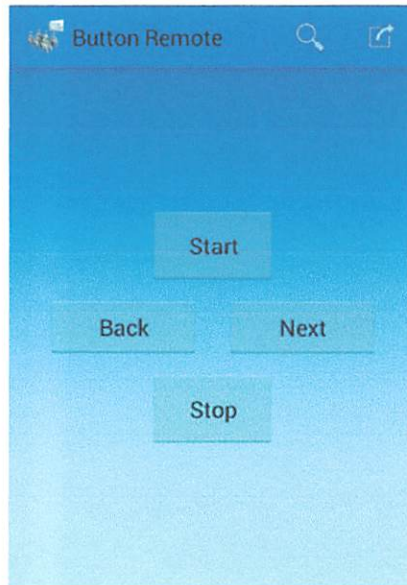
Setelah proses pengkoneksian menggunakan *Bluetooth* atau *Wifi*, pada *remote* menggunakan *Speech Recognition* terdapat tombol untuk memulai mengenali suara. Fitur *Speech Recognition* ini bisa digunakan untuk mengendalikan ppt presentasi berdasarkan inputan suara sesuai dengan *command* atau perintah yang tersimpan pada database *Command List* dan jika perintah yang di-inputkan dikenali, program akan mengirimkan paket data perintah ke aplikasi *server* untuk diproses lebih lanjut. Tampilan *remote Speech Recognition* ditunjukkan pada gambar 4.18.



Gambar 4.18 : Tampilan Remote Speech Recognition.

4.1.13 Implementasi *Button Remote*

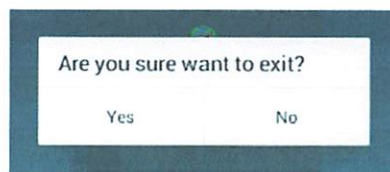
Sama halnya seperti *remote Speech Recognition*, fitur *Button Remote* juga berfungsi untuk mengendalikan ppt presentasi berdasarkan tombol yang ditekan, terdapat 4 tombol yaitu tombol *Next*, *Back*, *Start* dan *Stop* pada program *remote* ini. Pengguna juga bisa menggunakan tombol *volume +* dan *-* untuk memindahkan *slide* presentasi. Setiap penekanan tombol akan mengirimkan paket data perintah ke aplikasi *server* untuk diproses lebih lanjut. Tampilan *Button Remote* ditunjukkan seperti pada gambar 4.19.



Gambar 4.19 : Tampilan *Button Remote*.

4.1.14 Implementasi Menu *Exit*

Menu "*X*"/*Exit* pada tiap halaman program selain menu utama berfungsi untuk menutup *layout* program yang berjalan dan kembali ke halaman program sebelumnya. Menu "*X*"/*Exit* pada menu utama mempunyai fungsi yang sedikit berbeda, yaitu untuk menampilkan konfirmasi apakah ingin menutup dan mengakhiri program sepenuhnya atau tidak. Tampilan konfirmasi *Exit* ditunjukkan oleh gambar 4.20.



Gambar 4.20 : Tampilan Konfirmasi *Exit Program*.

4.2 Pengujian

Pengujian dilakukan terhadap aplikasi untuk memastikan bahwa aplikasi berjalan dengan benar sesuai dengan kebutuhan dan tujuan yang diharapkan

4.2.1 Pengujian Fungsional

Pengujian fungsional merupakan proses untuk melihat kesesuaian fungsi dari setiap fitur yang disediakan oleh aplikasi apakah sudah berjalan dengan baik atau tidak. Pengujian ini juga dilakukan untuk mendeteksi kesalahan-kesalahan yang mungkin terjadi pada program yang telah dibuat untuk evaluasi selanjutnya.

1. Pengujian Fungsional

Pengujian fungsional aplikasi ini dilakukan untuk mengetahui apakah semua fitur dapat berjalan pada beberapa tipe *smartphone* Android seperti yang ditunjukkan pada table 4.2.

Tabel 4.2 Pengujian Fungsional

No.	Menu	Posisi <i>Layout</i>	Proses	Hasil	
				A	B
1	Opening Layout	Starting Program	ke Menu Utama	✓	✓
2	Tombol Connection	Menu Utama	ke Menu Connection	✓	✓
3	Tombol Setting	Menu Utama	ke Settings	✓	✓
4	Tombol About	Menu Utama	ke About	✓	✓
5	Tombol “?” / Help	Menu Utama	ke Help	✓	✓
6	Tombol “X” / Exit	Menu Utama	ke Konfirmasi Exit	✓	✓
7	Tombol Bluetooth	Menu Connection	ke Menu Remote	✓	✓
8	Tombol Wifi	Menu Connection	ke Menu Remote	✓	✓
9	Tombol Speech Recognition	Menu Remote	ke Speech Recognition Remote	✓	✓
10	Tombol Button Remote	Menu Remote	ke Button Remote	✓	✓
11	Tombol Start Recognize	Speech Recognition Remote	Fungsi Speech Recognition	✓	✓
12	Tombol Next	Button Remote	Fungsi Tombol Next	✓	✓
13	Tombol Back	Button Remote	Fungsi Tombol Back	✓	✓
14	Tombol Start	Button Remote	Fungsi Tombol Start	✓	✓
15	Tombol Stop	Button Remote	Fungsi Tombol Stop	✓	✓

No.	Menu	Posisi <i>Layout</i>	Proses	Hasil	
				A	B
16	Tombol Search	Speech Recognition/Button Remote	ke Scan Bluetooth/Input IP	✓	✓
17	Tombol Swap	Speech Recognition/Button Remote	ke Speech Recognition/Button Remote	✓	✓
18	Tombol Command List	Settings	ke DB Command List	✓	✓
19	Tombol Theme Styles	Settings	ke Menu Ubah Tema	✓	✓
20	Tombol Update Data Command	DB Command List	Fungsi Upadate Database Command	✓	✓
21	Tombol Change Speech Language	DB Command List	ke Speech Setting	✓	✓
22	Fungsi Scan Bluetooth	Speech Recognition/Button Remote	Fungsi Scan dan Connect Server via BT	✓	✓
23	Fungsi Input IP	Speech Recognition/Button Remote	Fungsi Input dan Connect Server via Wifi	✓	✓
24	Fungsi Speech Recognition	Speech Recognition	Start Google Speech Recognition	✓	✓
25	Fungsi Connect via Bluetooth	Speech Recognition/Button Remote	ke Aplikasi Server	✓	✓
26	Fungsi Connect via Wifi	Speech Recognition/Button Remote	ke Aplikasi Server	✓	✓
27	Fungsi Theme Styles	Settings	Ubah Tema Aplikasi	✓	✓

Keterangan Simbol :

- ✓ : Berfungsi
- * : Tidak Berfungsi

A : Samsung Galaxy Fame GT-S6810 (Android 4.1 - Jelly Bean)

B : ASUS Zenfone 4 (Android 4.3 – Jelly Bean)

Dari informasi tabel di atas dapat disimpulkan bahwa program dapat berjalan maksimal pada *smartphone* Android dengan sistem operasi Jelly Bean dan versi di atasnya karena versi minimum SDK yang dibutuhkan adalah API level 16.

2. Pengujian Resolusi Hardware

Pengujian ini merupakan pengujian untuk melihat kesesuaian desain aplikasi dari berbagai resolusi *hardware* yang berbeda pada *smartphone* android yang ditunjukkan pada table 4.3.

Tabel 4.3 Pengujian Resolusi Hardware

No.	Type Smartphone	Layout Design (Pixel)	Portrait Orientation	Landscape Orientation
1	Samsung Galaxy Fame GT-S6810	320 x 480	✓	✗
2	ASUS Zenfone 4	480 x 800	✓	✗
3	Sony Xperia L	480 x 854	✓	✗
4	Smartfren Andromax U2	540 x 960	✓	✗
5	Lenovo Idea Tab A1000	600 x 1024	✓	✗

Keterangan Simbol :

- ✓ : Berfungsi
- ✗ : Tidak Berfungsi

Dari tabel 4.3 dapat disimpulkan bahwa desain *layout* program dapat berjalan maksimal pada berbagai resolusi *smartphone* dengan orientasi *portrait*. Program tidak dapat menggunakan orientasi *landscape* karena sistem program tidak menyediakan desain *layout* dengan orientasi tersebut.

3. Pengujian Konektifitas Bluetooth

Pengujian ini dilakukan untuk mengetahui pemanfaatan koneksi *bluetooth* yang digunakan. Hasil pengujian ini ditunjukkan pada tabel 4.4.

Tabel 4.4 Pengujian Konektifitas Bluetooth

No.	Type Smartphone (Network Adaptor)	Type Laptop (Network Adaptor)	Jarak Maksimal (Tanpa Halangan)	Jarak Maksimal (Ada Halangan)	Multiple Connection
1	Samsung Galaxy Fame GT-S6810 (Bluetooth v4.0)	HP Pavilion G4-1003TX (Broadcom 802.11n Wireless Network Adapter)	17 meter	15 meter	✓
2	ASUS Zenfone 4 (Bluetooth v4.0)	Dell Vostro 1088 (Dell Wireless Bluetooth module)	18 meter	14 meter	✓
3	Sony Xperia L (Bluetooth v4.0)	Fujitsu Lifebook LH532 (Bluetooth v4.0)	16 meter	15 meter	✓

No.	Tipe Smartphone (Network Adaptor)	Tipe Laptop (Network Adaptor)	Jarak Maksimal (Tanpa Halangan)	Jarak Maksimal (Ada Halangan)	Multiple Connection
4	Smartfren Andromax U2 (Bluetooth v4.0)	Dell Inspiron n4050 (Optional Combo Dell Wireless Card 802.11b/g/n, Bluetooth v3.0+HS)	17 meter	14 meter	✓
5	Lenovo Idea Tab A1000 (Bluetooth v4.0)	Asus K43BY (Built-in Bluetooth V2.1+EDR)	17 meter	15 meter	✓

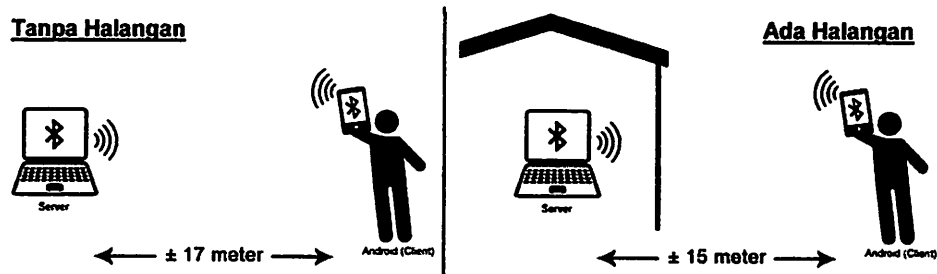
Keterangan Simbol :

✓ : Berfungsi

✗ : Tidak Berfungsi

Dari Tabel 4.4 disimpulkan bahwa konektifitas menggunakan *Bluetooth* bisa digunakan untuk lebih dari satu aplikasi *client* dengan jarak seperti yang tertera pada tabel.

Gambaran konektifitas menggunakan *Bluetooth* ketika terdapat halangan dan tanpa halangan ditunjukkan pada gambar 4.21.



Gambar 4.21 : Gambaran Konektifitas Bluetooth.

4. Pengujian Konektifitas Wifi

Pengujian ini dilakukan untuk mengetahui pemanfaatan koneksi *wifi* yang digunakan. Untuk pengujian konektifitas dengan *wifi* dibutuhkan *access point* untuk memperoleh alamat IP dan sebagai penghubung koneksi jaringan. Hasil pengujian ini ditunjukkan pada tabel 4.5.

Tabel 4.5 Pengujian Konektifitas Wifi

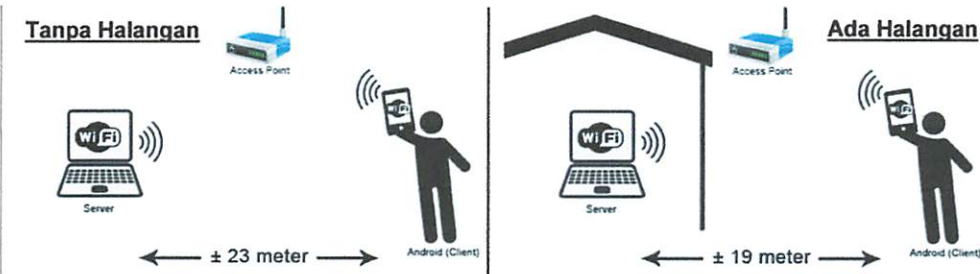
No.	Tipe Smartphone (Network Adaptor)	Tipe Laptop (Network Adaptor)	Jarak Maksimal (Tanpa Halangan)	Jarak Maksimal (Ada Halangan)	Multiple Connection
1	Samsung Galaxy Fame GT-S6810 (WLAN 802.11 b/g/n)	HP Pavilion G4-1003TX (Broadcom 802.11n Wireless Network Adapter)	24 meter	19 meter	✓
2	ASUS Zenfone 4 (WLAN 802.11b/g/n)	Dell Vostro 1088 (Wireless LAN 802.11b/g/n)	23 meter	20 meter	✓
3	Sony Xperia L (WLAN 802.11 b/g/n)	Fujitsu Lifebook LH532 (WLAN 802.11b/g/n)	23 meter	18 meter	✓
4	Smartfren Andromax U2 (WLAN 802.11 b/g/n)	Dell Inspiron n4050 (Standard Dell Wireless 802.11b/g/n)	21 meter	19 meter	✓
5	Lenovo Idea Tab A1000 (WLAN 802.11 b/g/n)	Asus K43BY (Integrated WLAN 802.11 b/g/n)	24 meter	18 meter	✓

Keterangan Simbol :

- ✓ : Berfungsi
- ✗ : Tidak Berfungsi

Dari Tabel 4.5 disimpulkan bahwa konektifitas menggunakan Wifi bisa digunakan untuk lebih dari satu aplikasi client dengan jarak device berada dalam radius rata-rata 19-23 meter.

Gambaran konektifitas menggunakan Wifi ketika terdapat halangan dan tanpa halangan ditunjukkan pada gambar 4.22.



Gambar 4.22 : Gambaran Konektifitas Wifi.

5. Pengujian *Input Speech Recognition*

Pengujian ini dilakukan untuk mengetahui keberhasilan aplikasi *client* dalam menerima inputan ucapan yang dapat terdeteksi oleh sistem *speech recognition*. Mikrofon *smartphone* itu diselaraskan dengan suara manusia yaitu dengan frekuensi antara 300-3400Hz, serta amplitudo antara 40-60dB. Suara sangat keras misalnya lebih dari 100 dB tidak dapat dikenali. Pengujian ini dilakukan pada frekuensi dan amplitudo umum yang dapat dikenali mikrofon *smartphone* yang ditunjukkan pada tabel 4.6.

Tabel 4.6 Pengujian *Input Speech Recognition*

No.	Fungsi	Jarak	Pengujian	Berhasil	Tidak Berhasil
1	<i>Next Slide</i>	1 meter	10x	7	3
2	<i>Previous Slide</i>	1 meter	10x	7	3
3	<i>Start Slide Show</i>	1 meter	10x	8	2
4	<i>Stop Slide Show</i>	1 meter	10x	6	4
5	<i>First Slide</i>	1 meter	10x	8	2
6	<i>Last Slide</i>	1 meter	10x	5	5

Prosentase Keberhasilan Pengujian *Input Speech Recognition*

$$\frac{41}{6} \times 100\% = 68.3\%$$

Keterangan :

Dari tabel 4.6 dapat disimpulkan bahwa pendeteksian suara dapat dilakukan dengan baik. Prosentase keberhasilan *input speech recognition* pada suara dengan jarak pengucapan perintah sejauh 1 meter adalah sekitar 68%.

6. Pengujian Aplikasi *Server*

Pengujian ini dilakukan untuk mengetahui keberhasilan aplikasi *server* dalam menerima perintah dari aplikasi *client* serta melakukan pengendalian terhadap tombol *keyboard* untuk mengendalikan aplikasi presentasi Ms. Power Point yang ditunjukkan pada tabel 4.7.

Tabel 4.7 Pengujian Aplikasi *Server*

No.	Fungsi	Hasil
1	<i>Next Slide</i>	✓
2	<i>Previous Slide</i>	✓
3	<i>Start Slide Show</i>	✓
4	<i>Stop Slide Show</i>	✓
5	<i>First Slide</i>	✓
6	<i>Last Slide</i>	✓
7	<i>Bluetooth Connect</i>	✓
8	<i>Wifi Connect</i>	✓

Keterangan Simbol :

- ✓ : Berfungsi
 ✕ : Tidak Berfungsi

Dari tabel 4.7 dapat disimpulkan bahwa aplikasi *server* bisa berjalan sesuai fungsi yang diharapkan untuk dapat mengendalikan program presentasi yang berjalan pada komputer *server*.

4.2.2 Pengujian Non Fungsional

Pengujian non fungsional dilakukan untuk mengetahui kepuasan pengguna (*user*) dalam menggunakan aplikasi ini dari segi kinerja, tampilan, kemudahan penggunaan, keberhasilan fungsi serta manfaat aplikasi yang disediakan. Pengujian ini dilakukan dengan memberikan beberapa pertanyaan kepada 10 pengguna (*user*) agar dapat memberikan penilaian terhadap aplikasi *remote presentation* VuDroid Wispa ini. Adapun hasil dari pengujian aplikasi kepada pengguna (*user*) ditunjukkan pada tabel 4.8.

Tabel 4.8 Pengujian Pengguna (*User*)

No.	Pertanyaan	Baik	Cukup	Kurang
1	Kinerja	16	4	-
2	Tampilan	16	4	-
3	Tingkat Kemudahan	12	6	2
4	Fungsi-fungsi Aplikasi	14	6	-
5	Manfaat Aplikasi	17	3	-

Prosentase Pengujian Pengguna (*User*)

1. Kinerja

$$\frac{16}{20} \times 100\% = 80\%$$

Keterangan :

Pengujian menunjukkan bahwa 16 dari 20 *user* memilih baik untuk kinerja aplikasi, sehingga memperoleh prosentase 80%, selebihnya *user* memilih cukup.

2. Tampilan

$$\frac{16}{20} \times 100\% = 80\%$$

Keterangan :

Pengujian menunjukkan bahwa 16 dari 20 *user* memilih baik untuk tampilan aplikasi, sehingga memperoleh prosentase 80%, selebihnya *user* memilih cukup.

3. Tingkat Kemudahan

$$\frac{12}{20} \times 100\% = 60\%$$

Keterangan :

Pengujian menunjukkan bahwa 12 dari 20 *user* memilih baik untuk tingkat kemudahan aplikasi, sehingga memperoleh prosentase 60%, 6 *user* memilih cukup dan 2 *user* lainnya memilih kurang.

4. Fungsi-fungsi Aplikasi

$$\frac{14}{20} \times 100\% = 70\%$$

Keterangan :

Pengujian menunjukkan bahwa 14 dari 20 *user* memilih baik untuk fungsi-fungsi yang ada pada aplikasi, sehingga memperoleh prosentase 70%, selebihnya *user* memilih cukup.

5. Manfaat Aplikasi

$$\frac{17}{20} \times 100\% = 85\%$$

Keterangan :

Pengujian menunjukkan bahwa 17 dari 20 *user* memilih baik untuk manfaat aplikasi, sehingga memperoleh prosentase 85%, selebihnya *user* memilih cukup.

Prosentase Rata-rata Hasil Pengujian Aplikasi Keseluruhan**1. Baik**

$$\frac{75}{100} \times 100\% = 75\%$$

2. Cukup

$$\frac{23}{100} \times 100\% = 23\%$$

3. Kurang

$$\frac{2}{100} \times 100\% = 2\%$$

Keterangan :

Dari pengujian sistem kepada pengguna (*user*) sebanyak 20 orang, dapat ditarik kesimpulan bahwa aplikasi *remote presentation* VuDroid Wispa ini secara keseluruhan dinilai baik dengan rata-rata sebesar 75%, dinilai cukup dengan rata-rata sebesar 23% dan 2% lainnya menilai kurang.

BAB V

PENUTUP

5.1 Kesimpulan

Berdasarkan penelitian dan pengujian yang telah dilakukan, maka dapat diambil beberapa kesimpulan antara lain :

1. Aplikasi *remote presentation* VuDroid Wispa yang telah dibuat dapat memudahkan pengguna dalam mengendalikan presentasi yang berjalan pada Ms. Power Point dari jarak jauh menggunakan android *smartphone*.
2. Aplikasi dapat melakukan pengkoneksian antara aplikasi *client* dengan aplikasi *server* menggunakan *Bluetooth* untuk mengirim dan menerima perintah pengendalian presentasi dengan jarak rata-rata jika ada halangan sejauh 15 meter dan tanpa halangan sejauh 17 meter.
3. Aplikasi dapat melakukan pengkoneksian antara aplikasi *client* dengan aplikasi *server* menggunakan *Wifi* untuk mengirim dan menerima perintah pengendalian presentasi dengan jarak rata-rata jika ada halangan sejauh 19 meter dan tanpa halangan sejauh 23 meter.
4. Fitur *speech recognition* bisa digunakan sebagai pengendali presentasi dengan meng-inputkan suara sesuai *command* atau perintah yang tersimpan dalam *database* dengan tingkat keberhasilan sebesar 68%.
5. Fitur dan fungsional aplikasi *client* dapat berjalan maksimal pada *smartphone* Android dengan versi sistem operasi Android 4.1 - Jelly Bean atau versi yang lebih tinggi.
6. Tampilan aplikasi *client* pada *smartphone* maupun *tablet* dapat berjalan dengan sempurna pada resolusi layar antara lain 320x480 *pixel*, 480x800 *pixel*, 480x854 *pixel*, 540x960 *pixel* dan 600x1024 *pixel*.
7. Aplikasi *server* dapat berjalan pada komputer dengan sistem operasi Microsoft Windows yang sudah terinstall JRE (*Java Runtime Environment*).
8. Dari pengujian sistem kepada pengguna (*user*) dapat ditarik kesimpulan bahwa aplikasi *remote presentation* VuDroid Wispa ini secara keseluruhan dinilai baik oleh *user* dengan rata-rata sebesar 75%.

5.2 Saran

Dalam pembuatan aplikasi *remote presentation* VuDroid Wispa ini masih memiliki banyak kekurangan sehingga untuk pengembangan lebih lanjut disarankan :

1. Pengkoneksian aplikasi *client* dengan aplikasi *server* menjadi lebih mudah untuk meningkatkan kemauan dan kenyamanan *user* dalam menggunakan aplikasi ini.
2. Pemanfaatan fitur *speech recognition* menjadi lebih akurat dan fleksibel sehingga *user* dapat dengan mudah melakukan pengendalian presentasi.
3. Menambahkan fitur-fitur tambahan yang dibutuhkan aplikasi *server* untuk kemudahan pengecekan pembaruan *IP address* dan *device Bluetooth*.
4. Menambahkan fitur pengaturan pada aplikasi *server* untuk mengubah proses inputan perintah.
5. Melakukan peningkatan pengendalian aplikasi *server* untuk presentasi yang sedang berjalan pada komputer dengan perintah-perintah yang lebih banyak.

DAFTAR PUSTAKA

- Kadir, Abdul. (2005). *"Dasar Pemrograman Java 2"*. Yogyakarta : ANDI Offset.
- Kusrianto, A. (2007). *"Presentasi Sukses Dengan Powerpoint"*. PT. Elex Media Komputindo. Jakarta.
- Newman, C. (2005). *"SQLite Developer's library"*. Sams.
- Ramadhan, A. (2006). *"Pengenaln Jaringan Komputer"*. PT. Elex Media Komputindo. Jakarta.
- Abimanyu, Mathur. (2012). *"What Is Wifi Technology"*. [Online]. Tersedia : <http://www.engineersgarage.com/articles/what-is-wifi-technology> (7 Januari 2014)
- Anonim. (2008). *"Automatic Speech Recognition Using Hidden Markov Models"*. [Online]. Tersedia : masters.donntu.edu.ua/2008/fvti/verenich/library/th_eng.htm (25 Juli 2014)
- Huda, Muhammad. (2013). *"4 Perbedaan Aplikasi Mobile Dengan Aplikasi Web Desktop"*. [Online]. Tersedia : <http://mkhuda.com/teknologi/4-perbedaan-aplikasi-mobile-dengan-aplikasi-web-desktop/> (16 April 2014)
- Opel, Alexander. (2003). *"Bluetooth"*. [PDF]. Tersedia : www.toengel.net/studium/mm_and_sec/bluetooth.pdf (3 Januari 2014)
- Oracle. (2014). *"Learn About Java"*. [Online]. Tersedia : <https://www.oracle.com/java/index.html> (16 Mei 2014)
- Wikipedia Indonesia. (2014). *"Android (sistem operasi)"*. [Online]. Tersedia : http://id.wikipedia.org/wiki/Android_%28sistem_operasi%29 (28 Desember 2013)

LAMPIRAN

LEMBAR KEASLIAN
PERNYATAAN KEASLIAN SKRIPSI

Saya yang bertanda tangan di bawah ini:

Nama : Muchammad Rofiq
Nim : 10.18.109
Program Studi : Teknik Informatika S-1
Fakultas : Teknologi Industri

Menyatakan dengan sesungguhnya bahwa Skripsi saya yang berjudul:

**MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa”
BERBASIS ANDROID MENGGUNAKAN GOOGLE SPEECH API**

Adalah skripsi saya sendiri bukan duplikat serta mengutip atau menyadur seluruhnya karya orang lain kecuali dari sumber aslinya.

Malang, 16 Agustus 2014

Yang Membuat Pernyataan,



Muchammad Rofiq



INSTITUT TEKNOLOGI NASIONAL MALANG
Fakultas Teknologi Industri
Program Studi Teknik Informatika S1

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

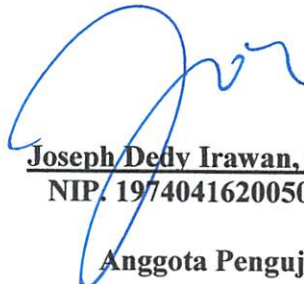
Nama : Muchammad Rofiq
NIM : 1018109
Jurusan : Teknik Informatika S-1
Judul : Membangun Aplikasi Remote Presentation "Vudroid Wispa"
Berbasis Android Menggunakan Google Speech API

Dipertahankan dihadapan Majelis Penguji Skripsi Jenjang Strata Satu (S-1)
pada :

Hari : Selasa
Tanggal : 19 Agustus 2014
Tempat : Ruang Rapat Dosen Teknik Informatika S-1
Nilai : (A)

Panitia Ujian Skripsi :

Ketua Majelis Penguji



Joseph Dedy Irawan, ST, MT
NIP. 197404162005011002

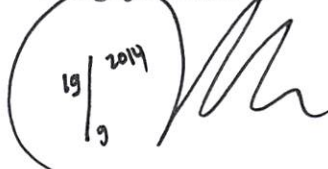
Anggota Penguji :

Penguji Pertama



Sonny Prasetyo, ST, MT
NIP.P. 1031000433

Penguji Kedua



Nurlaily Vandyansyah, ST



INSTITUT TEKNOLOGI NASIONAL MALANG
Fakultas Teknologi Industri
Program Studi Teknik Informatika S1

FORMULIR PERBAIKAN SKRIPSI

Nama : Muchammad Rofiq
NIM : 1018109
Jurusan : Teknik Informatika S-1
Judul : Membangun Aplikasi Remote Presentation "Vudroid Wispa"
Berbasis Android Menggunakan Google Speech API

Tanggal	Penguji	Uraian	Paraf
19 Agustus 2014	I	- Perbaikan Judul - Pengujian User Ditambah Menjadi 20 Responden - Perbaiki Kesimpulan - Perbaiki Abstrak - Gambarkan Ilustrasi Pengujian Bluetooth dan Wifi	
19 Agustus 2014	II	- Tambahkan Teori Komunikasi Data - Pengujian Fungsi Ilustrasikan dalam Bentuk Gambar dan Narasi - Tambahkan Proses Speech Recognition - Analisa Fungsi (Speech Recognition0 - Tambahkan Implementasi Google Speech API - Lampiran Hasil Responden/User - Penambahan Saran - Sitasi Landasan Teori dengan Daftar Pustaka	

Anggota Penguji :

Penguji Pertama

Sonny Prasetyo, ST, MT.
NIP.P. 1031000433

Penguji Kedua

Nurlaily Vendyansyah, ST

Mengetahui

Dosen Pembimbing I

Joseph Dedy Irawan, ST, MT.
NIP. 197404162005011002

Dosen Pembimbing II

Suryo Adi Wibowo, ST, MT
NIP.P. 1031000438



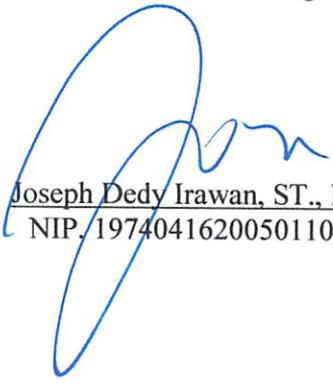
INSTITUT TEKNOLOGI NASIONAL MALANG
Fakultas Teknologi Industri
Program Studi Teknik Informatika S1

FORMULIR BIMBINGAN SKRIPSI

Nama : Muchammad Rofiq
NIM : 1018109
Masa Bimbingan : 19 Maret 2014 s/d 19 Agustus 2014
Judul Skripsi : Membangun Aplikasi Remote Presentation "VuDroid Wispa"
Berbasis Android Menggunakan Google Speech API

No.	TANGGAL	URAIAN	PARAF PEMBIMBING
1	16-07-2014	Demo Program	
2	21-07-2014	Revisi Program	
3	24-07-2014	Bimbingan Jurnal Seminar Hasil	
4	27-07-2014	Revisi Jurnal Seminar Hasil	
5	14-08-2014	Bimbingan Laporan Bab 1-5	
6	15-08-2014	Bimbingan Laporan Bab 1-5	
7	16-08-2014	Revisi Laporan Bab 1-5	

Malang, 16 Agustus 2014
Dosen Pembimbing


Joseph Dedy Irawan, ST., MT.
NIP. 197404162005011002



INSTITUT TEKNOLOGI NASIONAL MALANG
Fakultas Teknologi Industri
Program Studi Teknik Informatika S1

FORMULIR BIMBINGAN SKRIPSI

Nama : Muchammad Rofiq
NIM : 1018109
Masa Bimbingan : 19 Maret 2014 s/d 19 Agustus 2014
Judul Skripsi : Membangun Aplikasi Remote Presentation "VuDroid Wispa"
Berbasis Android Menggunakan Google Speech API

No.	TANGGAL	URAIAN	PARAF PEMBIMBING
1	10-07-2014	Bimbingan Laporan Bab 1-3	
2	12-07-2014	Revisi Laporan Bab 1-3	
3	18-07-2014	Bimbingan Laporan Bab 1-3	
4	18-07-2014	Demo Program	
5	21-07-2014	Bimbingan Laporan Bab 4-5	
6	22-07-2014	Revisi Laporan Bab 4-5	
7	23-07-2014	Bimbingan Jurnal Seminar Hasil	
8	25-07-2014	Revisi Jurnal Seminar Hasil	
9	15-08-2014	Bimbingan Laporan Bab 1-5	
10	16-08-2014	Bimbingan Laporan Bab 1-5	

Malang, 16 Agustus 2014

Dosen Pembimbing

Suryo Adi Wibowo, ST., MT.
NIP. P. 1031000438



PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG

**FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK**

PT. BNI (PERSERO) MALANG
BANK NIAGA MALANG

Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. RAYA Karanglo, Km2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

Malang, 21 April 2014

Nomor : ITN-253/INF/TA/2014
Lampiran : ---
Perihal : Bimbingan Skripsi

Kepada : Yth. Bpk/Ibu Joseph Dedy Irawan, ST, MT
Dosen Pembina Program Studi Teknik Informatika S-1
Institut Teknologi Nasional
Malang

Dengan Hormat,

Sesuai dengan permohonan dan persetujuan dalam proposal skripsi untuk mahasiswa :

Nama : MUCHAMMAD ROFIQ
Nim : 1018109
Prodi : Teknik Informatika S-1
Fakultas : Teknologi Industri

Maka dengan ini pembimbingan kami serahkan sepenuhnya kepada Saudara/i selama waktu 6 (enam) bulan, terhitung mulai tanggal :

21 April 2014 S/D 21 September 2014

Sebagai satu syarat untuk menempuh Ujian Akhir Sarjana Teknik, Program Studi Teknik Informatika S-1.

Demikian agar maklum dan atas perhatian serta bantuannya kami sampaikan terima kasih.

Mengetahui
Program Studi Teknik Informatika S-1
Ketua,



Joseph Dedy Irawan, ST., MT.
NIP. 197404162005021002

Form S-4a



PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG

**FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK**

PT. BNI (PERSERO) MALANG
BANK NIAGA MALANG

Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. RAYA Karanglo, Km2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

Malang, 21 April 2014

Nomor : ITN-253/INF/TA/2014

Lampiran : ---

Perihal : Bimbingan Skripsi

Kepada : Yth. Bpk/Ibu Suryo Adi Wibowo, ST.MT
Dosen Pembina Program Studi Teknik Informatika S-1
Institut Teknologi Nasional
Malang

Dengan Hormat,

Sesuai dengan permohonan dan persetujuan dalam proposal skripsi untuk mahasiswa :

Nama : MUCHAMMAD ROFIQ
Nim : 1018109
Prodi : Teknik Informatika S-1
Fakultas : Teknologi Industri

Maka dengan ini pembimbingan kami serahkan sepenuhnya kepada Saudara/i selama waktu 6 (enam) bulan, terhitung mulai tanggal :

21 April 2014 S/D 21 September 2014

Sebagai satu syarat untuk menempuh Ujian Akhir Sarjana Teknik, Program Studi Teknik Informatika S-1.

Demikian agar maklum dan atas perhatian serta bantuannya kami sampaikan terima kasih.

Mengetahui
Program Studi Teknik Informatika S-1
Ketua,



Joseph Dedy Irawan, ST., MT.

NIP. 197404162005021002

Form S-4a

Lampiran Script Program

1. Script Aplikasi Server

a. Main Program

```
public VuDroidWispa() {
    initComponents();
    setLocationRelativeTo(null);

    LocalDevice localDevice = null;
    try {
        localDevice = LocalDevice.getLocalDevice();
    } catch (BluetoothStateException ex) {

        Logger.getLogger(VuDroidWispa.class.getName()).log(Level.SEVERE, null, ex);
    }
    Btname.setText(localDevice.getFriendlyName());
    Btname.setForeground(Color.BLUE);

    try {
        InetAddress addr = InetAddress.getLocalHost();
        String ipAdd = addr.getHostAddress();
        ip.setText(ipAdd);
    } catch (UnknownHostException ex) {

        Logger.getLogger(VuDroidWispa.class.getName()).log(Level.SEVERE, null, ex);
    }
}

public static void main(String args[]) {
    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        @Override
        public void run() {
            new VuDroidWispa().setVisible(true);
            Thread runThread = new Thread(new
RunThread());

            runThread.start();
            Thread wifiThread = new Thread(new
WifiThread());

            wifiThread.start();
        }
    });
}
```

b. Bluetooth Thread

```
class RunThread extends VuDroidWispa implements Runnable {

    //Constructor
    public RunThread() {
    }

    @Override
```



```

public void onCreate() {
    super.onCreate();
    setContentView(R.layout.activity_main);

    // Initialize the local device
    LocalDevice localDevice = null;
    try {
        localDevice = LocalDevice.getLocalDevice();
    } catch (IOException ex) {
        // Log the exception
        Logger.getLogger(LocalDevice.class.getName()).log(Level.SEVERE,
            "Error getting local device: " + ex.getMessage(), ex);
    }

    // Initialize the remote device
    RemoteDevice remoteDevice = null;
    try {
        remoteDevice = RemoteDevice.getRemoteDevice();
    } catch (IOException ex) {
        // Log the exception
        Logger.getLogger(RemoteDevice.class.getName()).log(Level.SEVERE,
            "Error getting remote device: " + ex.getMessage(), ex);
    }

    // Initialize the connection
    Connection connection = null;
    try {
        connection = Connection.getConnection(localDevice, remoteDevice);
    } catch (IOException ex) {
        // Log the exception
        Logger.getLogger(Connection.class.getName()).log(Level.SEVERE,
            "Error getting connection: " + ex.getMessage(), ex);
    }

    // Initialize the data
    Data data = null;
    try {
        data = Data.getData(connection);
    } catch (IOException ex) {
        // Log the exception
        Logger.getLogger(Data.class.getName()).log(Level.SEVERE,
            "Error getting data: " + ex.getMessage(), ex);
    }

    // Initialize the UI
    TextView textView = (TextView) findViewById(R.id.textView);
    textView.setText(data.getText());
}

@Override
public void onStart() {
    super.onStart();
    // Start the connection
    startConnection();
}

@Override
public void onResume() {
    super.onResume();
    // Resume the connection
    resumeConnection();
}

@Override
public void onPause() {
    super.onPause();
    // Pause the connection
    pauseConnection();
}

@Override
public void onStop() {
    super.onStop();
    // Stop the connection
    stopConnection();
}

@Override
public void onDestroy() {
    super.onDestroy();
    // Destroy the connection
    destroyConnection();
}

private void startConnection() {
    // Start the connection
}

private void resumeConnection() {
    // Resume the connection
}

private void pauseConnection() {
    // Pause the connection
}

private void stopConnection() {
    // Stop the connection
}

private void destroyConnection() {
    // Destroy the connection
}

```



```

class RemoteDevice implements Runnable {
    // Remote device
    private RemoteDevice() {
        // Remote device
    }

    @Override
    public void run() {
        // Remote device
    }
}

```

```

public void run(){
    waitForBTConnect();
}

private void waitForBTConnect(){

    //Initialize local BT Device
    LocalDevice BTDevice = null;

    StreamConnectionNotifier notifier = null;
    StreamConnection connection = null;

    // setup the server to listen for connection
    try{
        BTDevice = LocalDevice.getLocalDevice(); //get
local BT Device
        BTDevice.setDiscoverable(DiscoveryAgent.GIAC);
        UUID uuid = new UUID(80087355); //"04c6093b-0000-
1000-8000-00805f9b34fb"
        String url = "btspp://localhost:" +
uuid.toString() + ";name=VuDroid Wispa";
        notifier =
(StreamConnectionNotifier)Connector.open(url);
    }catch(IOException e){
    }

    //waiting for connection
    while(true){
        try {
            //System.out.println("waiting for
connection...");
            connection = notifier.acceptAndOpen();

            Thread processThread = new Thread(new
ProcessConnectionThread(connection));
            processThread.start();
        } catch (IOException e) {
            return;
        }
    } //while
}
}

```

c. Bluetooth Process Command

```

class ProcessConnectionThread implements Runnable {

    private StreamConnection mConnection;

    //Constant that indicate command from device
    private static final int EXIT_CMD = -1;
    private static final int KEY_RIGHT = 1;
    private static final int KEY_LEFT = 2;
    private static final int KEY_START = 3;
    private static final int KEY_STOP = 4;

```

```

        public void run() {
            waitForBTConnect();
        }

        private void waitForBTConnect() {
            //Initialise local BT Device
            LocalDevice HfDevice = null;

            StreamConnectionNotifier n = null;
            StreamConnection connection = null;

            // Setup the server to listen for connection
            try {
                HfDevice = LocalDevice.getLocalDevice(); //Get
                Local BT Device
                BTPDevice btpDevice = new BTPDevice(HfDevice);
                UUID uuid = new UUID(80087355); //"\0408087355-0000-1000-8000-00805F9B34FB"
                String url = "btp://:" + btpDevice.getDeviceName();
                n = new StreamConnectionNotifier(url);
                connection = n.accept();
            } catch (IOException e) {
            }

            //Waiting for connection
            while (true) {
                try {
                    //System.out.println("Waiting for connection...");
                    connection = n.accept();
                } catch (IOException e) {
                }
            }

            Thread processThread = new Thread(new ProcessConnectionThread(connection));
            processThread.start();
        } catch (IOException e) {
        }
    }
}

```

c. Bluetooth Process Command

```

class ProcessConnectionThread implements Runnable {
    private StreamConnection mConnection;

    //Constants that indicate command from device
    private static final int EXIT_CMD = -1;
    private static final int KEY_RIGHT = 1;
    private static final int KEY_LEFT = 2;
    private static final int KEY_START = 3;
    private static final int KEY_STOP = 4;
}

```

```

private static final int KEY_FIRST = 5;
private static final int KEY_LAST = 6;

public ProcessConnectionThread(StreamConnection
connection) {
    mConnection = connection;
}

@Override
public void run(){
    try{
        //prepare to receive data
        InputStream inputStream =
mConnection.openInputStream();

        //System.out.println("Waiting for input..");

        while(true){
            int command = inputStream.read();

            if(command == EXIT_CMD){
                //System.out.println("finish process");
                break;
            }
            processCommand(command);
        }
    }catch(IOException e){
    }
}

/* Process the command from client
 * @param command the command code
 */
private void processCommand(int command) {
    try{
        Robot robot = new Robot();
        switch (command) {
            case KEY_RIGHT:
                robot.keyPress(KeyEvent.VK_DOWN);
                //System.out.println("Next");
                break;
            case KEY_LEFT:
                robot.keyPress(KeyEvent.VK_UP);
                //System.out.println("Back");
                break;
            case KEY_START:
                robot.keyPress(KeyEvent.VK_F5);
                //System.out.println("Start");
                break;
            case KEY_STOP:
                robot.keyPress(KeyEvent.VK_ESCAPE);
                //System.out.println("Stop");
                break;
            case KEY_FIRST:
                robot.keyPress(KeyEvent.VK_HOME);
                //System.out.println("Home");
                break;
            case KEY_LAST:
                robot.keyPress(KeyEvent.VK_END);

```

```

private static final int KEY_FIRST = 0;
private static final int KEY_LAST = 6;

public ProcessConnection(Socket s) {
    connection = s;
}

@Override
public void run() {
    try {
        //prepare to receive data
        InputStream inputStream =
            connection.getInputStream();

        //System.out.println("Waiting for input...");

        while(true) {
            int command = inputStream.read();

            if(command == KEY_FIRST ||
                //System.out.println("Finish process");
                break;
            }

            processCommand(command);
        }
    } catch(IOException e) {
    }
}

/* Process the command from client
 * @param command the command code
 */
private void processCommand(int command) {
    try {
        Robot robot = new Robot();

        switch (command) {
            case KEY_RIGHT:
                robot.keyPress(KeyEvent.VK_DOWN);
                //System.out.println("Next");
                break;
            case KEY_LEFT:
                robot.keyPress(KeyEvent.VK_UP);
                //System.out.println("Back");
                break;
            case KEY_START:
                robot.keyPress(KeyEvent.VK_F5);
                //System.out.println("Start");
                break;
            case KEY_STOP:
                robot.keyPress(KeyEvent.VK_ESCAPE);
                //System.out.println("Stop");
                break;
            case KEY_FIRST:
                robot.keyPress(KeyEvent.VK_HOME);
                //System.out.println("Home");
                break;
            case KEY_LAST:
                robot.keyPress(KeyEvent.VK_END);
        }
    }
}

```



```

        break;
        case "Start":
            robot.keyPress(KeyEvent.VK_F5);
            System.out.println(message);
            break;
        case "Stop":
            robot.keyPress(KeyEvent.VK_ESCAPE);
            System.out.println(message);
            break;
        case "First":
            robot.keyPress(KeyEvent.VK_HOME);
            System.out.println(message);
            break;
        case "Last":
            robot.keyPress(KeyEvent.VK_END);
            System.out.println(message);
            break;
    }
} catch (AWTException ex) {

Logger.getLogger(WifiThread.class.getName()).log(Level.SEVERE,
null, ex);

    }
    inputStreamReader.close();
    clientSocket.close();
} catch (IOException ex) {
    System.out.println("Problem in message
reading");
}
}
}
}
}

```

2. Script Aplikasi Client Android

a. *Android Manifest*

```

<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
    package="skripsi.vudroidwispa">

    <uses-permission
android:name="android.permission.BLUETOOTH" />

    <uses-permission
android:name="android.permission.BLUETOOTH_ADMIN" />

    <uses-permission
android:name="android.permission.ACCESS_WIFI_STATE" />

    <uses-permission

```



```

        android:name="android.permission.CHANGE_WIFI_STATE" />
        <uses-permission
        android:name="android.permission.CHANGE_NETWORK_STATE" />
        <uses-permission
        android:name="android.permission.ACCESS_NETWORK_STATE" />
        <uses-permission
        android:name="android.permission.INTERNET" />
        <uses-permission
        android:name="android.permission.RECORD_AUDIO" />

        <application
            android:allowBackup="true"
            android:icon="@drawable/ic_launcher"
            android:label="@string/app_name"
            android:theme="@style/AppTheme"
            android:screenOrientation="portrait">

            <activity
                android:name="skripsi.vudroidwispa.MainActivity"
                android:label="@string/app_name"
                android:screenOrientation="portrait">
                <intent-filter>
                    <action
                        android:name="android.intent.action.MAIN" />
                    <category
                        android:name="android.intent.category.LAUNCHER" />
                </intent-filter>
            </activity>
            <activity
                android:name=".MenuActivity"
                android:label="@string/app_name"
                android:screenOrientation="portrait"></activity>
            <activity
                android:name=".ConnectionMenuActivity"
                android:label="@string/con_menu"
                android:screenOrientation="portrait"></activity>

            //BT
            <activity

```

```

        android:name=".RemoteBTMenuActivity"
        android:label="@string/bt_con"
        android:screenOrientation="portrait">/activity>
    <activity
        android:name=".BTListActivity"
        android:label="@string/app_name"

android:theme="@android:style/Theme.Dialog">/activity>
    <activity
        android:name=".SpeechRecognitionBT"
        android:label="@string/speech_recog"
        android:screenOrientation="portrait">/activity>
    <activity
        android:name=".ButtonRemoteBT"
        android:label="@string/remote_btn"
        android:screenOrientation="portrait">/activity>
//End BT

//Wifi
<activity
    android:name=".RemoteWIFIMenuActivity"
    android:label="@string/wifi_con"
    android:screenOrientation="portrait">/activity>
<activity
    android:name=".WifiIPActivity"
    android:label="@string/app_name"

android:theme="@android:style/Theme.Dialog">/activity>
    <activity
        android:name=".SpeechRecognitionWIFI"
        android:label="@string/speech_recog"
        android:screenOrientation="portrait">/activity>
    <activity
        android:name=".ButtonRemoteWIFI"
        android:label="@string/remote_btn"
        android:screenOrientation="portrait">/activity>
//End Wifi

//Setting

```

```

        android:text="RemoteWiFiActivity"
        android:label="@string/wifi_con"
    android:screenOrientation="portrait"></activity>

<activity
    android:name=".WiFiActivity"
    android:label="@string/wifi_name"

    android:theme="@android:style/Theme.Dialog"/></activity>

<activity
    android:name=".SpeechRecognition"
    android:label="@string/speech_recog"
    android:screenOrientation="portrait"></activity>

<activity
    android:name=".ButtonRemoteWiFi"
    android:label="@string/remote_btn"
    android:screenOrientation="portrait"></activity>

\\End WiFi

\\Setting

```

```

<activity
    android:name=".SettingActivity"
    android:label="@string/action_settings"
    android:screenOrientation="portrait"></activity>
<activity
    android:name=".Prefs"
    android:label="@string/action_settings"
    android:screenOrientation="portrait"></activity>
<activity
    android:name=".SQLiteView"
    android:label="@string/db_com"
    android:screenOrientation="portrait"></activity>
//End Setting

//About
<activity
    android:name=".AboutActivity"
    android:label="@string/app_name"
    android:screenOrientation="portrait"></activity>
//End About

//About
<activity
    android:name=".HelpActivity"
    android:label="@string/app_name"

android:theme="@android:style/Theme.Dialog"></activity>
//End About
</application>

</manifest>

```

b. Database Adapter

```

public class DBAdapter {
    public static final String KEY_ROWID = "_id";
    public static final String KEY_COMMAND = "command_word";

    private static final String DB_NAME = "myDB";
    private static final String DB_TABLE = "CommandTable";
    private static final int DB_VERSION = 1;
    private final Context myContext;
    private DBHelper myHelper;

```

```

<activity
    android:name=".SettingActivity"
    android:label="@string/action_settings"
    android:screenOrientation="portrait"></activity>

<activity
    android:name=".Profile"
    android:label="@string/action_settings"
    android:screenOrientation="portrait"></activity>

<activity
    android:name=".SQLiteOpenHelper"
    android:label="@string/app_name"
    android:screenOrientation="portrait"></activity>

</end setting

//About
<activity
    android:name=".AboutActivity"
    android:label="@string/app_name"
    android:screenOrientation="portrait"></activity>

</end about

//About
<activity
    android:name=".HelpActivity"
    android:label="@string/app_name"
    android:theme="@android:style/Theme.Holo"></activity>

</end about
</action>

</manifest>

```

4. Database Helper

```

public class DBHelper {
    public static final String KEY_ROWID = "id";
    public static final String KEY_COMMAND = "command_word";

    private static final String DB_NAME = "mydb";
    private static final String DB_TABLE = "CommandTable";
    private static final int DB_VERSION = 1;
    private final Context myContext;
    private DBHelper myHelper;
}

```

```

private SQLiteDatabase myDatabase;

public DBAdapter(Context c) {
    myContext = c;
}

public DBAdapter open() throws SQLException {
    myHelper = new DBHelper(myContext);
    myDatabase = myHelper.getWritableDatabase();
    return this;
}

public void close() {
    myHelper.close();
}

//try
public String getCommandNext(long l1) throws SQLException
{
    String[] column = new String[]{KEY_ROWID,
KEY_COMMAND};
    Cursor c = myDatabase.query(DB_TABLE, column,
KEY_ROWID + "=" + l1, null, null, null, null);

    if (c != null) {
        c.moveToFirst();
        String command = c.getString(1); //column
position in array from getData
        return command;
    }
    return null;
}

public String getCommandBack(long l2) throws SQLException
{
    String[] column = new String[]{KEY_ROWID,
KEY_COMMAND};
    Cursor c = myDatabase.query(DB_TABLE, column,
KEY_ROWID + "=" + l2, null, null, null, null);

    if (c != null) {
        c.moveToFirst();
        String command = c.getString(1); //column
position in array from getData
        return command;
    }
    return null;
}

public String getCommandStart(long l3) throws
SQLException {
    String[] column = new String[]{KEY_ROWID,
KEY_COMMAND};
    Cursor c = myDatabase.query(DB_TABLE, column,
KEY_ROWID + "=" + l3, null, null, null, null);

    if (c != null) {
        c.moveToFirst();
        String command = c.getString(1); //column

```

[illegible]

```

        position in array from getData
        return command;
    }
    return null;
}

    public String getCommandStop(long l4) throws SQLException
    {
        String[] column = new String[]{KEY_ROWID,
KEY_COMMAND};
        Cursor c = myDatabase.query(DB_TABLE, column,
KEY_ROWID + "=" + l4, null, null, null, null);

        if (c != null) {
            c.moveToFirst();
            String command = c.getString(1); //column
position in array from getData
            return command;
        }
        return null;
    }

    public String getCommandFirst(long l5) throws
SQLException {
        String[] column = new String[]{KEY_ROWID,
KEY_COMMAND};
        Cursor c = myDatabase.query(DB_TABLE, column,
KEY_ROWID + "=" + l5, null, null, null, null);

        if (c != null) {
            c.moveToFirst();
            String command = c.getString(1); //column
position in array from getData
            return command;
        }
        return null;
    }

    public String getCommandLast(long l6) throws SQLException
    {
        String[] column = new String[]{KEY_ROWID,
KEY_COMMAND};
        Cursor c = myDatabase.query(DB_TABLE, column,
KEY_ROWID + "=" + l6, null, null, null, null);

        if (c != null) {
            c.moveToFirst();
            String command = c.getString(1); //column
position in array from getData
            return command;
        }
        return null;
    }

    public long modifCommand1(long lRow1, String mCommand1)
throws SQLException {
        ContentValues cvModif = new ContentValues();
        cvModif.put(KEY_COMMAND, mCommand1);
        return myDatabase.update(DB_TABLE, cvModif, KEY_ROWID

```

```

        кергэн шүрээсрэгэ обьекта (DB ЛУГТЕ' САМОУТЕ' КЕА КОМІД
        САМОУТЕ' БОР (КЕА СОМЖИД' ИСОМЖИДУУТ) :
        СОМЖИДУУТЕА САМОУТЕ' ИЕМ СОМЖИДУУТЕА() :
    эрхэмэ гэдэхсөөртөн {
        бүртгэ юмд ИСОМЖИДУУТЕА (ЮМД ЛУКОМ' ГЭГҮД ИСОМЖИДУУТ)

    }

    кергэн ИЛУУ :
    }

    кергэн СОМЖИДУУ :
    БОМЖИДУУ ИУ ЭХЭЛЭГ ЭТОМ БӨГРӨГЭ
    ГЭГҮД СОМЖИДУУ = О'БӨГРӨГЭГҮД ( ) : \ \СОМЖИДУУ
    О'ШОЛӨГӨГҮДЭР ( ) :
    ГЭ (О Л= ИЛУУ) :

    КЕА КОМІД + "==" + ГЭ' ИЛУУ' ИЛУУ' ИЛУУ' ИЛУУ :
    ОЛХӨӨР О = ШҮРЭЭСРЭГЭ' ДӨХЛЭ (DB ЛУГТЕ' СОМЖИДУУ'
    КЕА СОМЖИДУУ) :
    ГЭГҮД ( ) СОМЖИДУУ = ИЕМ ГЭГҮД ( ) (КЕА КОМІД'
    {
        бүртгэ гэгүд ИСОМЖИДУУТЕА (ЮМД ЛУ) ЭРХЭМЭ ГЭДЭХСӨӨРТӨН

    }

    кергэн ИЛУУ :
    }

    кергэн СОМЖИДУУ :
    БОМЖИДУУ ИУ ЭХЭЛЭГ ЭТОМ БӨГРӨГЭ
    ГЭГҮД СОМЖИДУУ = О'БӨГРӨГЭГҮД ( ) : \ \СОМЖИДУУ
    О'ШОЛӨГӨГҮДЭР ( ) :
    ГЭ (О Л= ИЛУУ) :

    КЕА КОМІД + "==" + ГЭ' ИЛУУ' ИЛУУ' ИЛУУ' ИЛУУ :
    ОЛХӨӨР О = ШҮРЭЭСРЭГЭ' ДӨХЛЭ (DB ЛУГТЕ' СОМЖИДУУ'
    КЕА СОМЖИДУУ) :
    ГЭГҮД ( ) СОМЖИДУУ = ИЕМ ГЭГҮД ( ) (КЕА КОМІД'
    ГЭДЭХСӨӨРТӨН {
        бүртгэ гэгүд ИСОМЖИДУУТЕА (ЮМД ЛУ) ЭРХЭМЭ

    }

    кергэн ИЛУУ :
    }

    кергэн СОМЖИДУУ :
    БОМЖИДУУ ИУ ЭХЭЛЭГ ЭТОМ БӨГРӨГЭ

```

```

+ "=" + lRow1, null);
    }

    public long modifCommand2(long lRow2, String mCommand2)
    throws SQLException {
        ContentValues cvModif = new ContentValues();
        cvModif.put(KEY_COMMAND, mCommand2);
        return myDatabase.update(DB_TABLE, cvModif, KEY_ROWID
+ "=" + lRow2, null);
    }

    public long modifCommand3(long lRow3, String mCommand3)
    throws SQLException {
        ContentValues cvModif = new ContentValues();
        cvModif.put(KEY_COMMAND, mCommand3);
        return myDatabase.update(DB_TABLE, cvModif, KEY_ROWID
+ "=" + lRow3, null);
    }

    public long modifCommand4(long lRow4, String mCommand4)
    throws SQLException {
        ContentValues cvModif = new ContentValues();
        cvModif.put(KEY_COMMAND, mCommand4);
        return myDatabase.update(DB_TABLE, cvModif, KEY_ROWID
+ "=" + lRow4, null);
    }

    public long modifCommand5(long lRow5, String mCommand5)
    throws SQLException {
        ContentValues cvModif = new ContentValues();
        cvModif.put(KEY_COMMAND, mCommand5);
        return myDatabase.update(DB_TABLE, cvModif, KEY_ROWID
+ "=" + lRow5, null);
    }

    public long modifCommand6(long lRow6, String mCommand6)
    throws SQLException {
        ContentValues cvModif = new ContentValues();
        cvModif.put(KEY_COMMAND, mCommand6);
        return myDatabase.update(DB_TABLE, cvModif, KEY_ROWID
+ "=" + lRow6, null);
    }

    private static class DBHelper extends SQLiteOpenHelper {
        public DBHelper(Context context) {
            super(context, DB_NAME, null, DB_VERSION);
        }

        @Override
        public void onCreate(SQLiteDatabase sqLiteDatabase) {
            sqLiteDatabase.execSQL("CREATE TABLE " + DB_TABLE
+ " (" +
                                KEY_ROWID + " INTEGER PRIMARY KEY
AUTOINCREMENT, " +
                                KEY_COMMAND + " TEXT NOT NULL);"
            );

            sqLiteDatabase.execSQL("insert into " + DB_TABLE
+ " (" + KEY_COMMAND + ") values ('next')");

```

```

+ "=" + rowId, null);
}

public long modifyCommand3(long rowId, String command3)
throws SQLException {
    ContentValues cvModif = new ContentValues();
    cvModif.put(KEY_COMMAND, command3);
    return myDatabase.update(DB_TABLE, cvModif, KEY_ROWID
+ "=" + rowId, null);
}

public long modifyCommand3(long rowId, String command3)
throws SQLException {
    ContentValues cvModif = new ContentValues();
    cvModif.put(KEY_COMMAND, command3);
    return myDatabase.update(DB_TABLE, cvModif, KEY_ROWID
+ "=" + rowId, null);
}

public long modifyCommand4(long rowId, String command4)
throws SQLException {
    ContentValues cvModif = new ContentValues();
    cvModif.put(KEY_COMMAND, command4);
    return myDatabase.update(DB_TABLE, cvModif, KEY_ROWID
+ "=" + rowId, null);
}

public long modifyCommand5(long rowId, String command5)
throws SQLException {
    ContentValues cvModif = new ContentValues();
    cvModif.put(KEY_COMMAND, command5);
    return myDatabase.update(DB_TABLE, cvModif, KEY_ROWID
+ "=" + rowId, null);
}

public long modifyCommand6(long rowId, String command6)
throws SQLException {
    ContentValues cvModif = new ContentValues();
    cvModif.put(KEY_COMMAND, command6);
    return myDatabase.update(DB_TABLE, cvModif, KEY_ROWID
+ "=" + rowId, null);
}

private static class DBHelper extends SQLiteOpenHelper {
    public DBHelper(Context context) {
        super(context, DB_NAME, null, DB_VERSION);
    }

    @Override
    public void onCreate(SQLiteDatabase sqLiteDatabase) {
        sqLiteDatabase.execSQL("CREATE TABLE " + DB_TABLE
+ " (" +
        KEY_ROWID + " INTEGER PRIMARY KEY
        AUTOINCREMENT, " +
        KEY_COMMAND + " TEXT NOT NULL)");
    }

    @Override
    public void onUpgrade(SQLiteDatabase sqLiteDatabase,
        int oldVersion, int newVersion) {
        // Nothing to do here
    }
}

```

```

        SQLiteDatabase.execSQL("insert into " + DB_TABLE
+ " (" + KEY_COMMAND + ") values ('back')");
        SQLiteDatabase.execSQL("insert into " + DB_TABLE
+ " (" + KEY_COMMAND + ") values ('start')");
        SQLiteDatabase.execSQL("insert into " + DB_TABLE
+ " (" + KEY_COMMAND + ") values ('stop')");
        SQLiteDatabase.execSQL("insert into " + DB_TABLE
+ " (" + KEY_COMMAND + ") values ('first')");
        SQLiteDatabase.execSQL("insert into " + DB_TABLE
+ " (" + KEY_COMMAND + ") values ('last')");
    }

    @Override
    public void onUpgrade(SQLiteDatabase sqLiteDatabase,
int i, int i2) {
        SQLiteDatabase.execSQL("DROP TABLE IF EXISTS " +
DB_TABLE);
        onCreate(sqLiteDatabase);
    }
}
}

```

c. Main Menu

```

public class MenuActivity extends FragmentActivity {

    private boolean isLight;
    private int currentTheme;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        // Following options to change the Theme must precede
        setContentView().
        toggleTheme();

        setContentView(R.layout.main_menu);

        Button ConnectionButton = (Button)
        findViewById(R.id.btnConnection);
        Button SettingButton = (Button)
        findViewById(R.id.btnSetting);
        Button AboutButton = (Button)
        findViewById(R.id.btnAbout);

        ConnectionButton.setOnClickListener(new
        View.OnClickListener() {
            public void onClick(View v) {
                Intent Connection = new
                Intent(MenuActivity.this, ConnectionMenuActivity.class);
                startActivity(Connection);
            }
        });

        SettingButton.setOnClickListener(new
        View.OnClickListener() {
            public void onClick(View v) {

```



```

        Intent Setting = new
Intent(MenuActivity.this, SettingActivity.class);
        startActivity(Setting);
    }
});

    AboutButton.setOnClickListener(new
View.OnClickListener() {
        public void onClick(View v) {
            Intent About = new Intent(MenuActivity.this,
AboutActivity.class);
            startActivity(About);
        }
    });
}

@Override
protected void onDestroy() {
    super.onDestroy();

    finish();
}

/**
 * onResume is called when the activity is going to
foreground.
 */
@Override
public void onResume() {
    super.onResume();
    toggleTheme();
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    // Inflate the menu; this adds items to the action
bar if it is present.
    getMenuInflater().inflate(R.menu.main, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    // Handle action bar item clicks here. The action bar
will
    // automatically handle clicks on the Home/Up button,
so long
    // as you specify a parent activity in
AndroidManifest.xml.
    /*int id = item.getItemId();
    if (id == R.id.action_help) {
        return true;
    }*/
    switch (item.getItemId()) {
        case R.id.action_help:
            Intent Help = new Intent(MenuActivity.this,
HelpActivity.class);
            startActivity(Help);
            return true;

```

```

IntentSetting = new
Intent(MenusActivity.this, SettingActivity.class);
startActivity(intent);

@Override
protected void onCreate() {
    super.onCreate();
    setContentView(R.layout.activity_main);
    ActionBar actionBar = getSupportActionBar();
    actionBar.setTitle("Menu");
    actionBar.setHomeButtonVisible(true);
    actionBar.setHomeAsUpIndicator(R.drawable.ic_menu);
    actionBar.setNavigationMode(ActionBar.NAVIGATION_MODE_LIST);
    List<MenuItem> menuItems = new ArrayList<MenuItem>();
    menuItems.add(new MenuItem().setTitle("Home").setIntent(new Intent(this, MainActivity.class)).setPriority(1));
    menuItems.add(new MenuItem().setTitle("Settings").setIntent(new Intent(this, SettingActivity.class)).setPriority(2));
    menuItems.add(new MenuItem().setTitle("About").setIntent(new Intent(this, AboutActivity.class)).setPriority(3));
    menuItems.add(new MenuItem().setTitle("Help").setIntent(new Intent(this, HelpActivity.class)).setPriority(4));
    actionBar.setListItems(menuItems);
}

@Override
public boolean onOptionsItemSelected() {
    // Handle action bar item clicks here. The action bar
    // will automatically handle clicks on the Home/Up button,
    // so long as you specify a parent activity in
    AndroidManifest.xml.
    int id = item.getItemId();
    if (id == R.id.action_help) {
        return true;
    }
    switch (item.getItemId()) {
        case R.id.action_help:
            Intent help = new Intent(MenusActivity.this,
            HelpActivity.class);
            startActivity(help);
            return true;
    }
    return false;
}

```

```

        case R.id.action_exit:
            exit();
            return true;
        }
        return super.onOptionsItemSelected(item);
    }

    @Override
    public boolean onKeyDown(int keyCode, KeyEvent event) {
        if (keyCode == KeyEvent.KEYCODE_BACK) {
            exit();
            return true;
        }
        return false;
    }

    public void exit() {
        AlertDialog.Builder alertDialogBuilder = new
        AlertDialog.Builder(this);

        // set dialog message
        alertDialogBuilder
            .setMessage("Are you sure want to exit?")
            .setCancelable(false)

            .setNegativeButton("Yes", new
        DialogInterface.OnClickListener() {
            public void onClick(DialogInterface
        dialog, int id) {
                finish();
            }
        })
            .setPositiveButton("No", new
        DialogInterface.OnClickListener() {
            public void onClick(DialogInterface
        dialog, int id) {
                dialog.cancel();
            }
        });
        AlertDialog alertDialog =
        alertDialogBuilder.create();
        alertDialog.show();
    }

    // Method to check SharedPreferences and set the current
    theme
    private void toggleTheme() {

        // Following options to change the Theme must precede
        setContentView().

        SharedPreferences sharedPref =
        PreferenceManager.getDefaultSharedPreferences(this);
        String lister =
        sharedPref.getString("list_preference", "1");

        currentTheme = Integer.parseInt(lister);
        if (currentTheme == 1) {
            isLight = false;

```

```

        case RtlDownKey:
            return super.OnKeyDown(keyCode, KeyEvent event);
        }

        @Override
        public boolean onKeyDown(int keyCode, KeyEvent event) {
            if (keyCode == KeyEvent.KEYCODE_BACK) {
                exit();
                return true;
            }
            return false;
        }

        public void exit() {
            AlertDialog.Builder builder = new
            AlertDialog.Builder(this);

            // set dialog message
            builder.setMessage("Are you sure want to exit?");
            builder.setCancelable(false);

            builder.setPositiveButton("Yes", new
            DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int id) {
                    finish();
                }
            });

            builder.setNegativeButton("No", new
            DialogInterface.OnClickListener() {
                public void onClick(DialogInterface dialog, int id) {
                    dialog.cancel();
                }
            });

            AlertDialog alertDialog =
            builder.create();
            alertDialog.show();
        }

        // Method to show SharedPreferences and set the content
        private void showDialog() {
            // Following options to change the theme must precede
            setContentView(R.layout.dialog);

            SharedPreferences sharedPreferences =
            PreferenceManager.getDefaultSharedPreferences(this);
            String lister =
            sharedPreferences.getString("list_preference", "1");

            sharedPreferences = sharedPreferences;
            if (listPreference == 1) {
                listPreference = false;
            }
        }
    }

```

```

    } else {
        isLight = true;
    }

    if (isLight) {
        setTheme(R.style.BlueSea);
    } else {
        setTheme(R.style.DarkSky);
    }
}
}

```

d. Setting Data Command

```

public class SQLiteView extends Activity {

    private boolean isLight;
    private int currentTheme;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        toggleTheme();
        setContentView(R.layout.sqliteview);

        final TextView tvNext = (TextView)
        findViewById(R.id.txtViewNext);
        final TextView tvBack = (TextView)
        findViewById(R.id.txtViewBack);
        final TextView tvStart = (TextView)
        findViewById(R.id.txtViewStart);
        final TextView tvStop = (TextView)
        findViewById(R.id.txtViewStop);
        final TextView tvFirst = (TextView)
        findViewById(R.id.txtViewFirst);
        final TextView tvLast = (TextView)
        findViewById(R.id.txtViewLast);

        try {
            final String s1 = new String("1");
            long l1 = Long.parseLong(s1);

            final String s2 = new String("2");
            long l2 = Long.parseLong(s2);

            final String s3 = new String("3");
            long l3 = Long.parseLong(s3);

            final String s4 = new String("4");
            long l4 = Long.parseLong(s4);

            final String s5 = new String("5");
            long l5 = Long.parseLong(s5);

            final String s6 = new String("6");
            long l6 = Long.parseLong(s6);

            final DBAdapter db = new

```



```

DBAdapter(SQLiteView.this);
    try {
        db.open();
    } catch (SQLException e) {
        e.printStackTrace();
    }
    String returnedCommandNext =
db.getCommandNext(11);
    String returnedCommandBack =
db.getCommandBack(12);
    String returnedCommandStart =
db.getCommandStart(13);
    String returnedCommandStop =
db.getCommandStop(14);
    String returnedCommandFirst =
db.getCommandStop(15);
    String returnedCommandLast =
db.getCommandStop(16);

    tvNext.setText(returnedCommandNext);
    tvBack.setText(returnedCommandBack);
    tvStart.setText(returnedCommandStart);
    tvStop.setText(returnedCommandStop);
    tvFirst.setText(returnedCommandFirst);
    tvLast.setText(returnedCommandLast);

    Button updateCommand = (Button)
findViewById(R.id.btnUpdateCommand);
    updateCommand.setOnClickListener(new
View.OnClickListener() {
        @Override
        public void onClick(View v) {
            try {
                String mCommand1 =
tvNext.getText().toString();
                long lRow1 = Long.parseLong(s1);
                String mCommand2 =
tvBack.getText().toString();
                long lRow2 = Long.parseLong(s2);
                String mCommand3 =
tvStart.getText().toString();
                long lRow3 = Long.parseLong(s3);
                String mCommand4 =
tvStop.getText().toString();
                long lRow4 = Long.parseLong(s4);
                String mCommand5 =
tvFirst.getText().toString();
                long lRow5 = Long.parseLong(s5);
                String mCommand6 =
tvLast.getText().toString();
                long lRow6 = Long.parseLong(s6);

                Dialog d = new
Dialog(SQLiteView.this);
                d.setTitle("DB Info");
                TextView tv = new
TextView(SQLiteView.this);
                tv.setText("Update Success!");
                d.setContentView(tv);

```

[illegible]

```

        d.show();

        db.modifCommand1(1Row1, mCommand1);
        db.modifCommand2(1Row2, mCommand2);
        db.modifCommand3(1Row3, mCommand3);
        db.modifCommand4(1Row4, mCommand4);
        db.modifCommand5(1Row5, mCommand5);
        db.modifCommand6(1Row6, mCommand6);
        db.close();
    } catch (Exception e) {
        String error = e.toString();

        Dialog d = new
Dialog(SQLiteView.this);
        d.setTitle("Update Failed!");
        TextView tv = new
TextView(SQLiteView.this);
        tv.setText(error);
        d.setContentView(tv);
        d.show();
    }
    });
} catch (Exception e) {
}

    Button voiceSearchSetting = (Button)
findViewById(R.id.voiceSearch);
    voiceSearchSetting.setOnClickListener(new
View.OnClickListener() {
        public void onClick(View v) {
            try {
                final Intent vsIntent = new
Intent(Intent.ACTION_MAIN);
                vsIntent.setComponent(new
ComponentName("com.google.android.voicesearch",
"com.google.android.voicesearch.VoiceSearchPreferences"));

                vsIntent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
                startActivity(vsIntent);

            } catch (final Exception e) {

                try {
                    final Intent vsjIntent = new
Intent(Intent.ACTION_MAIN); //jelly beans
                    vsjIntent.setComponent(new
ComponentName("com.google.android.googlequicksearchbox",
"com.google.android.voicesearch.VoiceSearchPreferences"));

                    vsjIntent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
                    startActivity(vsjIntent);

                } catch (final Exception e1) {
                    e1.printStackTrace();
                }
            }
        }
    }
}

```

```

        d.show();
    }

    db.modifiCommand1(fRow1, mCommand1);
    db.modifiCommand2(fRow2, mCommand2);
    db.modifiCommand3(fRow3, mCommand3);
    db.modifiCommand4(fRow4, mCommand4);
    db.modifiCommand5(fRow5, mCommand5);
    db.modifiCommand6(fRow6, mCommand6);
    d.close();
    } catch (Exception e) {
        String error = e.getMessage();
    }

    Created d = new
        Dialog(SQLiteDatabase);
        d.setTitle("Update Failed!");
        TextView tv = new
            TextView(SQLiteDatabase);
            tv.setText(error);
            d.setContentTextView(tv);
            d.show();
        }
    }
    } catch (Exception e) {
    }

    Button voiceSearchSetting = (Button)
        findViewById(R.id.voiceSearchSetting);
        voiceSearchSetting.setOnClickListener(
            new View.OnClickListener() {
                public void onClick(View v) {
                    try {
                        final Intent voiceIntent = new
                            Intent(Intent.ACTION_MAIN);
                            voiceIntent.setComponentName(
                                "com.google.android.voicesearch");
                            "com.google.android.voicesearch.VoiceSearchPreferences");
                            voiceIntent.putExtra(Intent.EXTRA_ACTIVITY_NEW_TASK);
                            startActivity(voiceIntent);
                    } catch (final Exception e) {
                    }
                    try {
                        final Intent voiceIntent = new
                            Intent(Intent.ACTION_MAIN);
                            voiceIntent.setComponentName(
                                "com.google.android.voicesearch.VoiceSearchPreferences");
                            "com.google.android.voicesearch.VoiceSearchPreferences");
                            voiceIntent.putExtra(Intent.EXTRA_ACTIVITY_NEW_TASK);
                            startActivity(voiceIntent);
                    } catch (final Exception e1) {
                        e1.printStackTrace();
                    }
                }
            }
        );

```

```

    });
}

protected void onDestroy() {
    super.onDestroy();

    finish();
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {

    // Inflate the menu; this adds items to the action
    bar if it is present.
    getMenuInflater().inflate(R.menu.main, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    // Handle action bar item clicks here. The action bar
    will
    // automatically handle clicks on the Home/Up button,
    so long
    // as you specify a parent activity in
    AndroidManifest.xml.
    switch (item.getItemId()) {
        case R.id.action_help:
            Intent Help = new Intent(SQLiteView.this,
            HelpActivity.class);
            startActivity(Help);
            return true;
        case R.id.action_exit:
            onDestroy();
            return true;
    }
    return super.onOptionsItemSelected(item);
}
}

```

e. Bluetooth Service

```

public class BTService {

    // Constants that indicate the current connectivity state
    public static final int STATE_NONE = 0;        // we're
    doing nothing
    public static final int STATE_LISTEN = 1;      // now
    listening for incoming connections
    public static final int STATE_CONNECTING = 2; // now
    initiating an outgoing connectivity
    public static final int STATE_CONNECTED = 3; // now
    connected to a remote device
    // Constants that indicate command to computer
    public static final int EXIT_CMD = -1;
    public static final int KEY_RIGHT = 1;
    public static final int KEY_LEFT = 2;
    public static final int KEY_START = 3;
    public static final int KEY_STOP = 4;
}

```



```

    public static final int KEY_FIRST = 5;
    public static final int KEY_LAST = 6;
    //    private BluetoothDevice mSavedDevice;
    //    private int mConnectionLostCount;

    // Debugging
    private static final String TAG =
"BluetoothCommandService";
    private static final boolean D = true;
    // Unique UUID for this application
    private static final UUID MY_UUID =
UUID.fromString("04c6093b-0000-1000-8000-00805f9b34fb");
    // Member fields
    private final BluetoothAdapter mAdapter;
    private final Handler mHandler;
    private ConnectThread mConnectThread;
    private ConnectedThread mConnectedThread;
    private int mState;
    //public static final int MOUSE_MOVE = 3;

    /**
     * Constructor. Prepares a new BluetoothChat session.
     *
     * @param context The UI Activity Context
     * @param handler A Handler to send messages back to the
UI Activity
     */
    public BTService(Context context, Handler handler) {
        mAdapter = BluetoothAdapter.getDefaultAdapter();
        mState = STATE_NONE;
        //mConnectionLostCount = 0;
        mHandler = handler;
    }

    /**
     * Return the current connectivity state.
     */
    public synchronized int getState() {
        return mState;
    }

    /**
     * Set the current state of the chat connectivity
     *
     * @param state An integer defining the current
connectivity state
     */
    private synchronized void setState(int state) {
        if (D) Log.d(TAG, "setState() " + mState + " -> " +
state);
        mState = state;

        // Give the new state to the Handler so the UI
Activity can update

        mHandler.obtainMessage(ButtonRemoteBT.MESSAGE_STATE_CHANGE,
state, -1).sendToTarget();
    }

```

```

        public static final int KEY_FIRST = 0;
        public static final int KEY_LAST = 0;
        private BluetoothDevice mDevice;
        private int mConnectionState;

        // Deinitializing
        private static final String TAG =
            "BluetoothCommandService";
        private static final boolean D = false;
        // Unique UUID for this application
        private static final UUID MY_UUID =
            UUID.fromString("00000000-0000-1000-8000-00805F9B34FB");
        // Member fields
        private final BluetoothAdapter mAdapter;
        private final Handler mHandler;
        private ConnectThread mConnectThread;
        private ConnectedThread mConnectedThread;
        private int mState;
        //public static final int MOUSE_MOVE = 0;

        //
        * Connection: Represents a new BluetoothChat session.
        *
        * Opened context: The UI Activity Context
        * @param handler: A handler to send messages back to the
        UI Activity
        */
        public BluetoothChatService(Handler handler) {
            mAdapter = BluetoothAdapter.getDefaultAdapter();
            mState = STATE_NONE;
            //mConnectionState = 0;
            mHandler = handler;
        }

        /**
         * Return the current connectivity state.
         */
        public synchronized int getState() {
            return mState;
        }

        /**
         * Set the current state of the chat connectivity
         *
         * @param state: An integer defining the current
            connectivity state
         */
        private synchronized void setState(int state) {
            if (D) Log.d(TAG, "setState() " + mState + " -> " + state);
            mState = state;

            // Give the new state to the Handler so the UI
            Activity can update
            mHandler.obtainMessage(BT_CONNECT_STATE_CHANGE,
                state, 0, null).sendToTarget();
        }
    }

```

```

/**
 * Start the chat service. Specifically start
 * AcceptThread to begin a
 * session in listening (server) mode. Called by the
 * Activity onResume()
 */
public synchronized void start() {
    if (D) Log.d(TAG, "start");

    // Cancel any thread attempting to make a
    connectivity
    if (mConnectThread != null) {
        mConnectThread.cancel();
        mConnectThread = null;
    }

    // Cancel any thread currently running a connectivity
    if (mConnectedThread != null) {
        mConnectedThread.cancel();
        mConnectedThread = null;
    }

    setState(STATE_LISTEN);
}

/**
 * Start the ConnectThread to initiate a connectivity to
 * a remote device.
 *
 * @param device The BluetoothDevice to connect
 */
public synchronized void connect(BluetoothDevice device)
{
    if (D) Log.d(TAG, "connect to: " + device);

    // Cancel any thread attempting to make a
    connectivity
    if (mState == STATE_CONNECTING) {
        if (mConnectThread != null) {
            mConnectThread.cancel();
            mConnectThread = null;
        }
    }

    // Cancel any thread currently running a connectivity
    if (mConnectedThread != null) {
        mConnectedThread.cancel();
        mConnectedThread = null;
    }

    // Start the thread to connect with the given device
    mConnectThread = new ConnectThread(device);
    mConnectThread.start();
    setState(STATE_CONNECTING);
}

/**
 * Start the ConnectedThread to begin managing a
 * Bluetooth connectivity

```

ВНЕШНЕГО СОВЕЩАНИЯ

• 2000-01-01 00:00:00 to 2000-01-01 00:00:00

1. 2. 3.

2652476 (STYLE CONNECTING) :

WOUNDED, KILLED, MISSING () .

CONNECTIONS = FOR CONNECTIONS (FEAT06)

11 agree the policy to connect with the district

1

UNCLASSIFIED//FOR OFFICIAL USE ONLY

UNCLASSIFIED//FOR OFFICIAL USE ONLY (U//FOUO) :

IT (recovered) (= 0.07) (

1) Өзіндік және бірлескен қаржы қолдануға тиым салу және қолданылуы

...

WOLFEHARTEN ISLAND

REF ID: A66541

TE (WONDERFULNESS) (= BUTT) (

```

11 (usage == STATE_COMPILE)

```

COMPTON AT FL

✓ cannot not please everybody to make a

77 (D) god'qlyue' "connected to: " + qeayge):

1

Бројте аписане наредби и списоци (успешно) (10)

•

ABSTRACT QUALITY AND HYPERCOMPUTATION DO COMBINE

24

7 1000, 5 054700

2025: THE CONNECTIONS TO THE 1970S & CONTEMPORARY

44

1

261205 PG (CIVIL RIGHTS)

•

UNCLASSIFIED//FOR OFFICIAL USE ONLY

WCONB60F5701599 CSUC67 () :

71 (WCDWCCFQJPT69C [E-2011])

11. расселит сибиряков сибирских племен : составлять

1

WCOUJGJUTBSG = UOJJA

WOMEN'S GROUPS' CHARTER :

IT INCONVENIENTESE = UNITI

CONJUGATELY

11 cannot say whether or not we

7E (D) 100-9 (372' "afbx") :

ВЕРХНЕ-КАМЫШЕНСКИЙ РАЙОН

4

УСТАТЪА ОУКЪЕШЕ()

1. The following information was obtained from the above mentioned source:

УДК 62-50:62-50:62-50

* 2011-2012 ANNUAL REPORT

、 米 子

```

*
* @param socket The BluetoothSocket on which the
connectivity was made
* @param device The BluetoothDevice that has been
connected
*/
public synchronized void connected(BluetoothSocket
socket, BluetoothDevice device) {
    if (D) Log.d(TAG, "connected");

    // Cancel the thread that completed the connectivity
    if (mConnectThread != null) {
        mConnectThread.cancel();
        mConnectThread = null;
    }

    // Cancel any thread currently running a connectivity
    if (mConnectedThread != null) {
        mConnectedThread.cancel();
        mConnectedThread = null;
    }

    // Start the thread to manage the connectivity and
perform transmissions
    mConnectedThread = new ConnectedThread(socket);
    mConnectedThread.start();

    // Send the name of the connected device back to the
UI Activity
    Message msg =
mHandler.obtainMessage(ButtonRemoteBT.MESSAGE_DEVICE_NAME);
    Bundle bundle = new Bundle();
    bundle.putString(ButtonRemoteBT.DEVICE_NAME,
device.getName());
    msg.setData(bundle);
    mHandler.sendMessage(msg);

    // save connected device
    //mSavedDevice = device;
    // reset connectivity lost count
    //mConnectionLostCount = 0;

    setState(STATE_CONNECTED);
}

/**
 * Stop all threads
 */
public synchronized void stop() {
    if (D) Log.d(TAG, "stop");
    if (mConnectThread != null) {
        mConnectThread.cancel();
        mConnectThread = null;
    }
    if (mConnectedThread != null) {
        mConnectedThread.cancel();
        mConnectedThread = null;
    }
}

```

```

        * @param socket The BluetoothSocket on which the
        * connectivity was made
        * @param device The BluetoothDevice that has been
        * connected
        *
        public synchronized void connected(BluetoothSocket
        socket, BluetoothDevice device) {
            if (D) Log.d(TAG, "connected");

            // Cancel the thread that completed the connectivity
            if (mConnectedThread != null) {
                mConnectedThread.cancel();
                mConnectedThread = null;
            }

            // Cancel any thread currently running a connectivity
            if (mConnectedThread != null) {
                mConnectedThread.cancel();
                mConnectedThread = null;
            }

            // Start the thread to manage the connectivity and
            // perform transactions
            mConnectedThread = new ConnectedThread(socket);
            mConnectedThread.start();

            // Send the name of the connected device back to the
            // UI Activity
            Message msg =
            mHandler.obtainMessage(BroadcastReceiver.MESSAGE_DEVICE_NAME);
            Bundle bundle = new Bundle();
            bundle.putString(BroadcastReceiver.DEVICE_NAME,
            device.getName());
            msg.putExtras(bundle);
            mHandler.sendMessage(msg);

            // save connected device
            mSavedDevice = device;
            // reset connectivity flag count
            mConnectionCount = 0;

            setState(STATE_CONNECTED);
        }

        *
        * Stop all threads
        *
        public synchronized void stop() {
            if (D) Log.d(TAG, "stop");
            if (mConnectedThread != null) {
                mConnectedThread.cancel();
                mConnectedThread = null;
            }
            if (mConnectedThread != null) {
                mConnectedThread.cancel();
                mConnectedThread = null;
            }
        }
    }

```

```

        setState(STATE_NONE);
    }

    /**
     * Write to the ConnectedThread in an unsynchronized
    manner
     *
     * @param out The bytes to write
     * @see ConnectedThread#write(byte[])
     */
    /* public void write(byte[] out) {
        // Create temporary object
        ConnectedThread r;
        // Synchronize a copy of the ConnectedThread
        synchronized (this) {
            if (mState != STATE_CONNECTED) return;
            r = mConnectedThread;
        }
        // Perform the write unsynchronized
        r.write(out);
    }
    */

    public void write(int out) {
        // Create temporary object
        ConnectedThread r;
        // Synchronize a copy of the ConnectedThread
        synchronized (this) {
            if (mState != STATE_CONNECTED) return;
            r = mConnectedThread;
        }
        // Perform the write unsynchronized
        r.write(out);
    }

    /**
     * Indicate that the connectivity attempt failed and
    notify the UI Activity.
     */
    private void connectionFailed() {
        setState(STATE_LISTEN);

        // Send a failure message back to the Activity
        Message msg =
        mHandler.obtainMessage(ButtonRemoteBT.MESSAGE_TOAST);
        Bundle bundle = new Bundle();
        bundle.putString(ButtonRemoteBT.TOAST, "Unable to
    connect device");
        msg.setData(bundle);
        mHandler.sendMessage(msg);
    }

    /**
     * Indicate that the connectivity was lost and notify the
    UI Activity.
     */
    private void connectionLost() {
        setState(STATE_LISTEN);
        // Send a failure message back to the Activity
        Message msg =

```

```

mHandler.obtainMessage(ButtonRemoteBT.MESSAGE_TOAST);
    Bundle bundle = new Bundle();
    bundle.putString(ButtonRemoteBT.TOAST, "Device
connectivity was lost");
    msg.setData(bundle);
    mHandler.sendMessage(msg);
}

/**
 * This thread runs while attempting to make an outgoing
connectivity
 * with a device. It runs straight through; the
connectivity either
 * succeeds or fails.
 */
private class ConnectThread extends Thread {
    private final BluetoothSocket mmSocket;
    private final BluetoothDevice mmDevice;

    public ConnectThread(BluetoothDevice device) {
        mmDevice = device;
        BluetoothSocket tmp = null;

        // Get a BluetoothSocket for a connectivity with
the
        // given BluetoothDevice
        try {
            tmp =
device.createRfcommSocketToServiceRecord(MY_UUID);
        } catch (IOException e) {
            Log.e(TAG, "create() failed", e);
        }
        mmSocket = tmp;
    }

    public void run() {
        Log.i(TAG, "BEGIN mConnectThread");
        setName("ConnectThread");

        // Always cancel discovery because it will slow
down a connectivity
        mAdapter.cancelDiscovery();

        // Make a connectivity to the BluetoothSocket
        try {
            // This is a blocking call and will only
return on a
            // successful connectivity or an exception
            mmSocket.connect();
        } catch (IOException e) {
            connectionFailed();
            // Close the socket
            try {
                mmSocket.close();
            } catch (IOException e2) {
                Log.e(TAG, "unable to close() socket
during connectivity failure", e2);
            }
            // Start the service over to restart

```

```

// Stand the service over to restart
}
during connectivity failure", e2);
Log.e(TAG, "unable to close() socket
        } catch (IOException e2) {
            mSocket.close();
        }
        try {
            mSocket.close();
        }
        // Close the socket
        // connectionFailed();
        } catch (IOException e) {
            mSocket.connect();
        }
        // successful connectivity or an exception
        return on a
    }
    // This is a blocking call and will only
    try {
        // takes a connectivity to the BluetoothSocket
        mSocket.connect();
    }
    // Always cancel discovery because it will slow
    // down a connectivity
    mSocket.cancelDiscovery();
    setName("ConnectThread");
    Log.i(TAG, "BEGIN mConnectThread");
    public void run() {
        mSocket = null;
    }
    Log.e(TAG, "connect() failed", e);
    } catch (IOException e) {
        device.close();
    }
    try {
        // given BluetoothDevice
        // Get a BluetoothSocket for a connectivity with
        BluetoothSocket soc = null;
        mDevice = device;
    }
    public BluetoothDevice device;
}
private final BluetoothDevice mDevice;
private final BluetoothSocket mSocket;
private static ConnectThread extends Thread
}
// succeeds or fails
connectivity of other
* with a device. It runs straight through; the
connectivity
// This thread runs while attempting to make an outgoing
**
}
mHandler.sendMessage(msg);
msg.setObj(mDevice);
connectivity was lost";
}
device.putString(BroadcastReceiver.EXTRA_MESSAGE_TOAST, "Device
        Bundle bundle = new Bundle();
        mHandler.sendMessage(BroadcastReceiver.EXTRA_MESSAGE_TOAST);
}

```

```

listening mode
    BService.this.start();
    return;
}

// Reset the ConnectThread because we're done
synchronized (BService.this) {
    mConnectThread = null;
}

// Start the connected thread
connected(mmSocket, mmDevice);
}

public void cancel() {
    try {
        mmSocket.close();
    } catch (IOException e) {
        Log.e(TAG, "close() of connect socket
failed", e);
    }
}

/**
 * This thread runs during a connectivity with a remote
device.
 * It handles all incoming and outgoing transmissions.
 */
private class ConnectedThread extends Thread {
    private final BluetoothSocket mmSocket;
    private final InputStream mmInStream;
    private final OutputStream mmOutStream;

    public ConnectedThread(BluetoothSocket socket) {
        Log.d(TAG, "create ConnectedThread");
        mmSocket = socket;
        InputStream tmpIn = null;
        OutputStream tmpOut = null;

        // Get the BluetoothSocket input and output
streams
        try {
            tmpIn = socket.getInputStream();
            tmpOut = socket.getOutputStream();
        } catch (IOException e) {
            Log.e(TAG, "temp sockets not created", e);
        }

        mmInStream = tmpIn;
        mmOutStream = tmpOut;
    }

    public void run() {
        Log.i(TAG, "BEGIN mConnectedThread");
        byte[] buffer = new byte[1024];

        // Keep listening to the InputStream while
connected

```

[illegible]

```

        while (true) {
            try {
                // Read from the InputStream
                int bytes = mmInStream.read(buffer);

                // Send the obtained bytes to the UI
Activity
mHandler.obtainMessage(ButtonRemoteBT.MESSAGE_READ, bytes, -
1, buffer)
                    .sendToTarget();
            } catch (IOException e) {
                Log.e(TAG, "disconnected", e);
                connectionLost();
                break;
            }
        }

    /**
     * Write to the connected OutputStream.
     *
     * @param buffer The bytes to write
     */
    public void write(byte[] buffer) {
        try {
            mmOutputStream.write(buffer);

            // Share the sent message back to the UI
Activity
//
mHandler.obtainMessage(BluetoothChat.MESSAGE_WRITE, -1, -1,
buffer)
//
                    .sendToTarget();
            } catch (IOException e) {
                Log.e(TAG, "Exception during write", e);
            }
        }

        public void write(int out) {
            try {
                mmOutputStream.write(out);

                // Share the sent message back to the UI
Activity
//
mHandler.obtainMessage(BluetoothChat.MESSAGE_WRITE, -1, -1,
buffer)
//
                    .sendToTarget();
            } catch (IOException e) {
                Log.e(TAG, "Exception during write", e);
            }
        }

        public void cancel() {
            try {
                mmOutputStream.write(EXIT_CMD);
                mmSocket.close();
            } catch (IOException e) {

```

```

        Log.e(TAG, "close() of connect socket
failed", e);
    }
}
}
}

```

f. Bluetooth List Scan

```

public class BTListActivity extends Activity {

    //Debugging
    private static final String TAG = "BTListActivity";
    private static final boolean D = true;
    //Return Intent extra
    public static String EXTRA_DEVICE_ADDRESS =
"device_address";
    //OnClick listener for all BT devices in ListView
    private AdapterView.OnItemClickListener
mDeviceClickListener = new AdapterView.OnItemClickListener()
{
    public void onItemClick(AdapterView<?> av, View view,
int arg2, long arg3) {
        //cancel discovery because will create p2p
connectivity
        mBtAdapter.cancelDiscovery();

        //get BT device MAC Address, the last 17char in
view
        String BTinfo = ((TextView)
view).getText().toString();
        String BTaddress =
BTinfo.substring(BTinfo.length() - 17);

        //create the result intent and include MAC
address
        Intent BTintent = new Intent();
        BTintent.putExtra(EXTRA_DEVICE_ADDRESS,
BTaddress);

        //set result and finish activity
        setResult(Activity.RESULT_OK, BTintent);
        finish();
    }
};

//BroadcastReceiver that listens for discovered BT
devices n change the title when discovery is finished
private final BroadcastReceiver mReceiver = new
BroadcastReceiver() {
    @Override
    public void onReceive(Context context, Intent intent)
{
        String BTaction = intent.getAction();

        //when discovery finds a device
        if
(BluetoothDevice.ACTION_FOUND.equals(BTaction)) {
            //get BT device object from intent
            BluetoothDevice BTReceiveDevice =

```

```

        Log.e(TAG, "close() of connect socket
        failed");
    }
}

```

4 Bluetooth List View

```

public class BTDiscovery extends Activity {

    //Declaring
    private static final String TAG = "BTDiscovery";
    private static final boolean D = true;
    //Return Intent extra
    public static String EXTRA_DEVICE_ADDRESS =
        "device_address";
    //onClick listener for all BT devices in listview
    private AdapterView.OnItemClickListener
        onItemClick = new AdapterView.OnItemClickListener() {
            public void onItemClick(AdapterView<?> av, View view,
                int arg2, long arg3) {
                //cancel discovery because will create gap
                connectivity
                    mBluetooth.cancelDiscovery();

                //get BT device MAC address, the last character in
                view
                String BTinfo = ((TextView)
                    view).getText().toString();
                String BTaddress =
                    BTinfo.substring(BTinfo.length() - 17);

                //store the result intent and include MAC
                address
                Intent BtIntent = new Intent();
                BtIntent.putExtra(EXTRA_DEVICE_ADDRESS,
                    BTaddress);

                //set result and finish activity
                setResult(Activity.RESULT_OK, BtIntent);
                finish();
            }
        };

    //BroadcastReceiver that listens for discovered BT
    //device changes the title when discovery is finished
    private final BroadcastReceiver mReceiver = new
        BroadcastReceiver() {
            @Override
            public void onReceive(Context context, Intent intent) {
                String BTaction = intent.getAction();

                //when discovery finds a device
                if (BluetoothAdapter.ACTION_DISCOVERY_FINISHED.equals(BTaction)) {
                    //get BT device object from intent
                    BluetoothAdapter

```

```

intent.getParcelableExtra(BluetoothDevice.EXTRA_DEVICE);
        //if already paired, skip it, because already
        listed
        if (BTReceiveDevice.getBondState() !=
BluetoothDevice.BOND_BONDED) {

mNewDevicesArrayAdapter.add(BTReceiveDevice.getName() + "\n"
+ BTReceiveDevice.getAddress());
        }
        //when discovery finished, change activity
        title
        } else if
(BluetoothAdapter.ACTION_DISCOVERY_FINISHED.equals(BTaction))
{
        setProgressBarIndeterminateVisibility(false);
        setTitle(R.string.select_BTdevice);
        if (mNewDevicesArrayAdapter.getCount() == 0)
        {
                String noBTDevices =
getResources().getText(R.string.no_BTfound).toString();
                mNewDevicesArrayAdapter.add(noBTDevices);
        }
        }
};
private boolean isLight;
private int currentTheme;
//Member fields
private BluetoothAdapter mBtAdapter;
private ArrayAdapter<String> mPairedDevicesArrayAdapter;
private ArrayAdapter<String> mNewDevicesArrayAdapter;

@Override
protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        toggleTheme();

        //Setup the window

requestWindowFeature(Window.FEATURE_INDETERMINATE_PROGRESS);
        setContentView(R.layout.btdevice_list);

        //Set result CANCELED in case the user backs out
        setResult(Activity.RESULT_CANCELED);

        //Initialize the button to perform device discovery
        Button ScanBT = (Button)
findViewById(R.id.btnScanBT);
        ScanBT.setOnClickListener(new View.OnClickListener()
        {
                @Override
                public void onClick(View view) {
                        scanBluetooth();
                        view.setVisibility(View.GONE);
                }
        });

        //Initialize ArrayAdapters, One for already pairedBT

```

[illegible]

```

and one for newBT
    mPairedDevicesArrayAdapter = new
    ArrayAdapter<String>(this, R.layout.bt_name);
    mNewDevicesArrayAdapter = new
    ArrayAdapter<String>(this, R.layout.bt_name);

    //find and set up ListView for pairedBT
    ListView pairedBTlist = (ListView)
    findViewById(R.id.lvPairedBT);
    pairedBTlist.setAdapter(mPairedDevicesArrayAdapter);

    pairedBTlist.setOnItemClickListener(mDeviceClickListener);

    //find and set up ListView for newBT
    ListView newBTlist = (ListView)
    findViewById(R.id.lvNewBT);
    newBTlist.setAdapter(mNewDevicesArrayAdapter);

    newBTlist.setOnItemClickListener(mDeviceClickListener);

    //Get the local BT adapter
    mBtAdapter = BluetoothAdapter.getDefaultAdapter();

    //Register for broadcasts when BT device is
    discovered
    IntentFilter filter = new
    IntentFilter(BluetoothDevice.ACTION_FOUND);
    this.registerReceiver(mReceiver, filter);

    //Register for broadcast when BT discovery is
    finished
    filter = new
    IntentFilter(BluetoothAdapter.ACTION_DISCOVERY_FINISHED);
    this.registerReceiver(mReceiver, filter);

    //Get a set of currently paired BT devices
    Set<BluetoothDevice> pairedBT =
    mBtAdapter.getBondedDevices();

    //if there are paired BT, add each one to
    ArrayAdapter
    if (pairedBT.size() > 0) {

    findViewById(R.id.textBTpaired).setVisibility(View.VISIBLE);
        for (BluetoothDevice BTdevice : pairedBT) {

    mPairedDevicesArrayAdapter.add(BTdevice.getName() + "\n" +
    BTdevice.getAddress());
        }
    } else {
        String noBTDevices = (String)
    getResources().getText(R.string.no_BTpaired);
        mPairedDevicesArrayAdapter.add(noBTDevices);
    }
}

@Override
protected void onDestroy() {
    super.onDestroy();
}

```

```

        and one for newBT
        mPairDeviceArrayAdapter = new
        ArrayAdapter<String>(this, R.layout.bt_name);
        mNewDeviceArrayAdapter = new
        ArrayAdapter<String>(this, R.layout.bt_name);

        //Find and set up ListView for pairedBT
        ListView pairedBTlist = (ListView)
        findViewById(R.id.listViewBT);
        pairedBTlist.setAdapter(mPairDeviceArrayAdapter);

        pairedBTlist.setOnItemClickListener(mDeviceClickListener);

        //Find and set up ListView for newBT
        ListView newBTlist = (ListView)
        findViewById(R.id.listViewBT);
        newBTlist.setAdapter(mNewDeviceArrayAdapter);

        newBTlist.setOnItemClickListener(mDeviceClickListener);

        //Get the local BT adapter
        mBTAdapter = BluetoothAdapter.getDefaultAdapter();

        //Register for broadcasts when BT device is
        discovered
        IntentFilter filter = new
        IntentFilter(BluetoothAdapter.ACTION_FOUND);
        this.registerReceiver(mReceiver, filter);

        //Register for broadcasts when BT discovery is
        finished
        filter = new
        IntentFilter(BluetoothAdapter.ACTION_DISCOVERY_FINISHED);
        this.registerReceiver(mReceiver, filter);

        //Get a list of currently paired BT devices
        Set<BluetoothDevice> pairedBT =
        mBTAdapter.getPairedDevices();

        //If there are paired BT add each one to
        mArrayAdapter
        if (pairedBT.size() > 0) {
            findViewById(R.id.listViewBT).setVisibility(View.VISIBLE);
            for (BluetoothDevice BTDevice : pairedBT) {
                mPairDeviceArrayAdapter.add(BTDevice.getName() + "/" +
                BTDevice.getAddress());
            }
            if (mArrayAdapter.getCount() > 0) {
                mNewDeviceArrayAdapter.add(mArrayAdapter.getItem(0));
            }
        }
    }

    @Override
    protected void onPause() {
        super.onPause();
    }

```

```

        //make sure app not doing BT discovery
        if (mBtAdapter != null) {
            mBtAdapter.cancelDiscovery();
        }
        //unregister broadcast listener
        this.unregisterReceiver(mReceiver);
    }

    //BT scan
    private void scanBluetooth() {
        if (D) Log.d(TAG, "scanBluetooth()");

        //indicate scanning in the title
        setProgressBarIndeterminateVisibility(true);
        setTitle(R.string.scanningBT);

        //Turn on sub-title for new devices

        findViewById(R.id.textBTNew).setVisibility(View.VISIBLE);

        //if already discovering, stop the process
        if (mBtAdapter.isDiscovering()) {
            mBtAdapter.cancelDiscovery();
        }

        //Request discover from BTAdapter
        mBtAdapter.startDiscovery();
    }
}

```

g. Speech Recognition Bluetooth

```

public class SpeechRecognitionBT extends Activity implements
RecognitionListener {

    static final int check = 1111;

    // Message types sent from the BluetoothService Handler
    public static final int MESSAGE_STATE_CHANGE = 1;
    public static final int MESSAGE_DEVICE_NAME = 4;
    public static final int MESSAGE_TOAST = 5;

    // Key names received from the BluetoothCommandService
    Handler
    public static final String DEVICE_NAME = "bt_name";
    public static final String TOAST = "toast";
    private static final String TAG = "MyVoiceNoBox";

    // Intent request codes
    private static final int REQUEST_CONNECT_DEVICE = 1;
    private static final int REQUEST_ENABLE_BT = 2;

    //SpeechRecognitionBT component
    TextView errorID, errorStatus;
    Button StartRecognize;
    ListView lv;
    SpeechRecognizer sr;

    private boolean isLight;

```

```

        //BT scan
        private void scanBluetooth() {
            if (D) Log.d(TAG, "scanBluetooth()");

            //Indicate scanning in the title
            setTitle(R.string.scanningBT);
            self.progressBarIndeterminate.setVisibility(View.VISIBLE);

            //Turn on app- this for new devices
            findViewById(R.id.textNew).setVisibility(View.VISIBLE);

            //If already discovering, stop the process
            if (mBluetoothAdapter.isDiscovering()) {
                mBluetoothAdapter.cancelDiscovery();
            }

            //Request discover from BluetoothAdapter
            mBluetoothAdapter.startDiscovery();
        }
    }

    //make sure app not doing BT discovery
    if (mBluetoothAdapter == null) {
        mBluetoothAdapter = BluetoothAdapter.getDefaultAdapter();
    }
}

```

g. Speech Recognition Bluetooth

```

public class SpeechRecognitionBluetooth extends AppCompatActivity {
    RecognitionListener mRecognitionListener;

    static final int check = 1111;

    // Message types sent from the BluetoothService Handler
    public static final int MESSAGE_STATE_CHANGE = 1;
    public static final int MESSAGE_DEVICE_NAME = 2;
    public static final int MESSAGE_TOAST = 3;

    // Key names received from the BluetoothCommandService
    Handler mHandler;
    private static final String TAG = "SpeechRecognition";
    public static final String DEVICE_NAME = "bt_name";
    public static final String TOAST = "toast";

    // Intent request codes
    private static final int REQUEST_CONNECT_DEVICE = 1;
    private static final int REQUEST_ENABLE_BT = 2;

    //SpeechRecognitionBluetooth component
    TextView errorID, errorStatus;
    Button startSpeechRecognition;
    ListView lv;
    SpeechRecognizer recognizer;

    private boolean isPaired;
}

```

```

private int currentTheme;

// Name of the connected device
private String mConnectedDeviceName = null;

// The Handler that gets information back from the
BluetoothChatService
private final Handler mHandler = new Handler() {
    @Override
    public void handleMessage(Message msg) {
        switch (msg.what) {
            case MESSAGE_STATE_CHANGE:
                switch (msg.arg1) {
                    case BTService.STATE_CONNECTED:
                        break;
                    case BTService.STATE_CONNECTING:
                        break;
                    case BTService.STATE_LISTEN:
                    case BTService.STATE_NONE:
                        break;
                }
                break;
            case MESSAGE_DEVICE_NAME:
                // save the connected device's name
                mConnectedDeviceName =
msg.getData().getString(DEVICE_NAME);
                Toast.makeText(getApplicationContext(),
"Connected to "
                                + mConnectedDeviceName,
Toast.LENGTH_SHORT).show();
                break;
            case MESSAGE_TOAST:
                Toast.makeText(getApplicationContext(),
msg.getData().getString(TOAST),
                                Toast.LENGTH_SHORT).show();
                break;
        }
    }
};

// Local Bluetooth adapter
private BluetoothAdapter mBluetoothAdapter = null;
// Member object for Bluetooth Command Service
private BTService mCommandService = null;
};

@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    // Set up the window layout
    toggleTheme();
    setContentView(R.layout.speech_recog);

    // Set SpeechRecognitionBT component
    errorID = (TextView) findViewById(R.id.tvErrorID);
    errorStatus =
(TextView) findViewById(R.id.tvErrorStatus);
    lv = (ListView) findViewById(R.id.lvVoiceReturn);
    StartRecognize = (Button) findViewById(R.id.bVoice);

```

```

private int currentTheme;

// Name of the connected device
private String mConnectedDeviceName = null;

// The Handler that gets information back from the
BluetoothGattService
private final Handler mHandler = new Handler() {
    @Override
    public void handleMessage(Message msg) {
        switch (msg.what) {
            case MESSAGE_STATE_CHANGE:
                switch (mGatt.getState()) {
                    case BluetoothGatt.STATE_CONNECTED:
                        break;
                    case BluetoothGatt.STATE_CONNECTING:
                        break;
                    case BluetoothGatt.STATE_LISTENING:
                        break;
                    case BluetoothGatt.STATE_NONE:
                        break;
                }
                break;
            case MESSAGE_DEVICE_NAME:
                // save the connected device's name
                mConnectedDeviceName = msg.getData().getString("DEVICE_NAME");
                Toast.makeText(getApplicationContext(),
                    "Connected to "
                    + mConnectedDeviceName,
                    Toast.LENGTH_SHORT).show();
                break;
            case MESSAGE_TOAST:
                Toast.makeText(getApplicationContext(),
                    msg.getData().getString("TOAST"),
                    Toast.LENGTH_SHORT).show();
                break;
        }
    }
};

// Local Bluetooth adapter
private BluetoothAdapter mAdapter = null;
// Member object for Bluetooth Command Service
private BluetoothCommandService mService = null;

@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    // Set up the window layout
    setContentView(R.layout.activity_main);
    // Set SpeedResponseBT component
    findViewById(R.id.textView1).setOnClickListener();
    findViewById(R.id.textView2).setOnClickListener();
    findViewById(R.id.textView3).setOnClickListener();
    findViewById(R.id.textView4).setOnClickListener();
    findViewById(R.id.textView5).setOnClickListener();
    findViewById(R.id.textView6).setOnClickListener();
    findViewById(R.id.textView7).setOnClickListener();
    findViewById(R.id.textView8).setOnClickListener();
    findViewById(R.id.textView9).setOnClickListener();
    findViewById(R.id.textView10).setOnClickListener();
    findViewById(R.id.textView11).setOnClickListener();
    findViewById(R.id.textView12).setOnClickListener();
    findViewById(R.id.textView13).setOnClickListener();
    findViewById(R.id.textView14).setOnClickListener();
    findViewById(R.id.textView15).setOnClickListener();
    findViewById(R.id.textView16).setOnClickListener();
    findViewById(R.id.textView17).setOnClickListener();
    findViewById(R.id.textView18).setOnClickListener();
    findViewById(R.id.textView19).setOnClickListener();
    findViewById(R.id.textView20).setOnClickListener();
    findViewById(R.id.textView21).setOnClickListener();
    findViewById(R.id.textView22).setOnClickListener();
    findViewById(R.id.textView23).setOnClickListener();
    findViewById(R.id.textView24).setOnClickListener();
    findViewById(R.id.textView25).setOnClickListener();
    findViewById(R.id.textView26).setOnClickListener();
    findViewById(R.id.textView27).setOnClickListener();
    findViewById(R.id.textView28).setOnClickListener();
    findViewById(R.id.textView29).setOnClickListener();
    findViewById(R.id.textView30).setOnClickListener();
    findViewById(R.id.textView31).setOnClickListener();
    findViewById(R.id.textView32).setOnClickListener();
    findViewById(R.id.textView33).setOnClickListener();
    findViewById(R.id.textView34).setOnClickListener();
    findViewById(R.id.textView35).setOnClickListener();
    findViewById(R.id.textView36).setOnClickListener();
    findViewById(R.id.textView37).setOnClickListener();
    findViewById(R.id.textView38).setOnClickListener();
    findViewById(R.id.textView39).setOnClickListener();
    findViewById(R.id.textView40).setOnClickListener();
    findViewById(R.id.textView41).setOnClickListener();
    findViewById(R.id.textView42).setOnClickListener();
    findViewById(R.id.textView43).setOnClickListener();
    findViewById(R.id.textView44).setOnClickListener();
    findViewById(R.id.textView45).setOnClickListener();
    findViewById(R.id.textView46).setOnClickListener();
    findViewById(R.id.textView47).setOnClickListener();
    findViewById(R.id.textView48).setOnClickListener();
    findViewById(R.id.textView49).setOnClickListener();
    findViewById(R.id.textView50).setOnClickListener();
    findViewById(R.id.textView51).setOnClickListener();
    findViewById(R.id.textView52).setOnClickListener();
    findViewById(R.id.textView53).setOnClickListener();
    findViewById(R.id.textView54).setOnClickListener();
    findViewById(R.id.textView55).setOnClickListener();
    findViewById(R.id.textView56).setOnClickListener();
    findViewById(R.id.textView57).setOnClickListener();
    findViewById(R.id.textView58).setOnClickListener();
    findViewById(R.id.textView59).setOnClickListener();
    findViewById(R.id.textView60).setOnClickListener();
    findViewById(R.id.textView61).setOnClickListener();
    findViewById(R.id.textView62).setOnClickListener();
    findViewById(R.id.textView63).setOnClickListener();
    findViewById(R.id.textView64).setOnClickListener();
    findViewById(R.id.textView65).setOnClickListener();
    findViewById(R.id.textView66).setOnClickListener();
    findViewById(R.id.textView67).setOnClickListener();
    findViewById(R.id.textView68).setOnClickListener();
    findViewById(R.id.textView69).setOnClickListener();
    findViewById(R.id.textView70).setOnClickListener();
    findViewById(R.id.textView71).setOnClickListener();
    findViewById(R.id.textView72).setOnClickListener();
    findViewById(R.id.textView73).setOnClickListener();
    findViewById(R.id.textView74).setOnClickListener();
    findViewById(R.id.textView75).setOnClickListener();
    findViewById(R.id.textView76).setOnClickListener();
    findViewById(R.id.textView77).setOnClickListener();
    findViewById(R.id.textView78).setOnClickListener();
    findViewById(R.id.textView79).setOnClickListener();
    findViewById(R.id.textView80).setOnClickListener();
    findViewById(R.id.textView81).setOnClickListener();
    findViewById(R.id.textView82).setOnClickListener();
    findViewById(R.id.textView83).setOnClickListener();
    findViewById(R.id.textView84).setOnClickListener();
    findViewById(R.id.textView85).setOnClickListener();
    findViewById(R.id.textView86).setOnClickListener();
    findViewById(R.id.textView87).setOnClickListener();
    findViewById(R.id.textView88).setOnClickListener();
    findViewById(R.id.textView89).setOnClickListener();
    findViewById(R.id.textView90).setOnClickListener();
    findViewById(R.id.textView91).setOnClickListener();
    findViewById(R.id.textView92).setOnClickListener();
    findViewById(R.id.textView93).setOnClickListener();
    findViewById(R.id.textView94).setOnClickListener();
    findViewById(R.id.textView95).setOnClickListener();
    findViewById(R.id.textView96).setOnClickListener();
    findViewById(R.id.textView97).setOnClickListener();
    findViewById(R.id.textView98).setOnClickListener();
    findViewById(R.id.textView99).setOnClickListener();
    findViewById(R.id.textView100).setOnClickListener();
}

```

```

        sr = SpeechRecognizer.createSpeechRecognizer(this);
        sr.setRecognitionListener(this);

        // Get local Bluetooth adapter
        mBluetoothAdapter =
BluetoothAdapter.getDefaultAdapter();

    }

    @Override
    protected void onStart() {
        super.onStart();
        if (mCommandService == null) {
            setupCommand();
        }
    }

    @Override
    protected void onResume() {
        super.onResume();
        // Performing this check in onResume() covers the case
in which BT was
        // not enabled during onStart(), so we were paused to
enable it...
        // onResume() will be called when
ACTION_REQUEST_ENABLE activity returns.
        if (mCommandService != null) {
            if (mCommandService.getState() ==
BTService.STATE_NONE) {
                mCommandService.start();
            }
        }
    }

    private void setupCommand() {
        // Initialize the BluetoothChatService to perform
bluetooth connections
        mCommandService = new BTService(this, mHandler);
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        sr.destroy();
        if (mCommandService != null)
            mCommandService.stop();
    }

    @Override
    public boolean onKeyDown(int keyCode, KeyEvent event) {
        if (keyCode == KeyEvent.KEYCODE_BACK) {
            String btnText =
StartRecognize.getText().toString();
            if (btnText == "Stop Recognize") {
                sr.destroy();
                finish();
                startActivity(getIntent());
            } else {

```

[illegible]

```

        finish();
    }
    return true;
}
return false;
}

private void openBTButtonR() {
    // Launch BT Button Remote
    Intent BTButtonR = new Intent(this,
ButtonRemoteBT.class);
    startActivity(BTButtonR);
    this.finish();
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.option_menu, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {
        case R.id.scan:
            // Launch the DeviceListActivity to see
            devices and do scan
            Intent serverIntent = new Intent(this,
BTListActivity.class);
            startActivityForResult(serverIntent,
REQUEST_CONNECT_DEVICE);
            return true;
        case R.id.swapRemote:
            openBTButtonR();
            return true;
    }
    return false;
}

public void onActivityResult(int requestCode, int
resultCode, Intent data) {
    switch (requestCode) {
        case REQUEST_CONNECT_DEVICE:
            // When DeviceListActivity returns with a
            device to connect
            if (resultCode == Activity.RESULT_OK) {
                // Get the device MAC address
                String address = data.getExtras()

.getString(BTListActivity.EXTRA_DEVICE_ADDRESS);
                // Get the BLuetoothDevice object
                BluetoothDevice device =
mBluetoothAdapter.getRemoteDevice(address);
                // Attempt to connect to the device
                mCommandService.connect(device);
            }
            break;
        case REQUEST_ENABLE_BT:

```

```

        finish();
    }

    return false;
}

private void openBluetooth() {
    // Launch BT Adapter Remote
    Intent intent = new Intent("android.bluetooth.adapter.action.requestEnable");
    startActivity(intent);
    this.finish();
}

@Override
public boolean onOptionsItemSelected() {
    MenuItem item = getMenuItem();
    if (item != null) {
        startActivity(intent);
        return true;
    }
}

@Override
public boolean onOptionsItemSelected() {
    switch (item.getItemId()) {
        case R.id.action_settings:
            // Launch the DeviceActivity to see
            devices and do scan
            Intent intent = new Intent(this,
                DeviceActivity.class);
            startActivity(intent);
            return true;
        case R.id.action_scan:
            openBluetooth();
            return true;
    }
    return false;
}

public void onActivityResult(int requestCode, int resultCode, Intent data) {
    switch (requestCode) {
        case REQUEST_CONNECT_DEVICE:
            // When DeviceActivity returns with a
            device to connect
            if (resultCode == Activity.RESULT_OK) {
                // Get the device MAC address
                String address = data.getStringExtra("address");
                // Get the BluetoothDevice object
                BluetoothDevice device = BluetoothAdapter.getDefaultAdapter().getRemoteDevice(address);
                // Attempt to connect to the device
                mCommandService.connect(device);
            }
            break;
        case REQUEST_ENABLE_BT:
            // When DeviceActivity returns with a
            device to connect
            if (resultCode == Activity.RESULT_OK) {
                // Get the device MAC address
                String address = data.getStringExtra("address");
                // Get the BluetoothDevice object
                BluetoothDevice device = BluetoothAdapter.getDefaultAdapter().getRemoteDevice(address);
                // Attempt to connect to the device
                mCommandService.connect(device);
            }
            break;
    }
}

```

```

// When the request to enable Bluetooth
returns
    if (resultCode == Activity.RESULT_OK) {
        // Bluetooth is now enabled, so set up a
chat session
        setupCommand();
    } else {
        // User did not enable Bluetooth or an
error occurred
        Toast.makeText(this,
R.string.bt_not_enabled_leaving, Toast.LENGTH_SHORT).show();
        finish();
    }
    break;
}

}

public void StartRecog(View view) {
    StartRecognize.setText("Stop Recognize");
    String btnText = StartRecognize.getText().toString();
    AutoRecognize();
    if(btnText == "Stop Recognize") {
        StartRecognize.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {
                sr.destroy();
                finish();
                startActivity(getIntent());
            }
        });
    }
}

public void AutoRecognize(){
    Intent intent = new
Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);

    intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL, Recogniz
erIntent.LANGUAGE_MODEL_FREE_FORM);

    intent.putExtra(RecognizerIntent.EXTRA_CALLING_PACKAGE, "skrips
i.vudroidwispa");

    intent.putExtra(RecognizerIntent.EXTRA_MAX_RESULTS, 5);

    intent.putExtra(RecognizerIntent.EXTRA_SPEECH_INPUT_COMPLETE_S
ILENCE_LENGTH_MILLIS, 2000); //The amount of time that it
should take after we stop hearing speech to consider the input
complete.

    intent.putExtra(RecognizerIntent.EXTRA_SPEECH_INPUT_MINIMUM_LE
NGTH_MILLIS, 2000); //The minimum length of an utterance.

    intent.putExtra(RecognizerIntent.EXTRA_SPEECH_INPUT_POSSIBLY_C
OMplete_SILENCE_LENGTH_MILLIS, 2000); //The amount of time
that it should take after we stop hearing speech to consider
the input possibly complete.
    sr.startListening(intent);
}

```

[illegible]

```

        Log.i("111111","11111111");
    }

    //Speech Recognition Listener
    public void onReadyForSpeech(Bundle params) //Called when
    the endpointer is ready for the user to start speaking.
    {
        Log.d(TAG, "onReadyForSpeech");
    }
    public void onBeginningOfSpeech() //The user has started
    to speak.
    {
        Log.d(TAG, "onBeginningOfSpeech");
    }
    public void onRmsChanged(float rmsdB) //Used for receiving
    notifications from the SpeechRecognizer when the recognition
    related events occur. All the callbacks are executed on the
    Application main thread
    {
        Log.d(TAG, "onRmsChanged");
    }
    public void onBufferReceived(byte[] buffer) //More sound
    has been received.
    {
        Log.d(TAG, "onBufferReceived");
    }
    public void onEndOfSpeech() //Called after the user stops
    speaking.
    {
        Log.d(TAG, "onEndofSpeech");
    }
    public void onError(int error) //A network or recognition
    error occurred.
    {
        Log.d(TAG, "error " + error);
        errorID.setText("error " + error);
        if (error == 1) {
            errorStatus.setText(" ERROR_NETWORK_TIMEOUT");
        } else if (error == 2) {
            errorStatus.setText(" ERROR_NETWORK ");
        } else if (error == 3) {
            errorStatus.setText(" ERROR_AUDIO ");
        } else if (error == 4) {
            errorStatus.setText(" ERROR_SERVER ");
        } else if (error == 5) {
            errorStatus.setText(" ERROR_CLIENT ");
        } else if (error == 6) {
            errorStatus.setText(" ERROR_SPEECH_TIMEOUT ");
        } else if (error == 7) {
            errorStatus.setText(" ERROR_NO_MATCH ");
        } else if (error == 8) {
            errorStatus.setText(" ERROR_RECOGNIZER_BUSY ");
        } else if (error == 9) {
            errorStatus.setText("
ERROR_INSUFFICIENT_PERMISSIONS ");
        }
        try {
            Thread.sleep(2000);
        } catch (InterruptedException e) {

```

```

try {
    Thread.sleep(2000);
} catch (InterruptedException e) {
    // ignored
}

//Speech Recognition Listener
public void onReadyForSpeech() { //Called when
    the endpointer is ready for the user to start speaking.
}

log.d(TAG, "onReadyForSpeech");

public void onBeginningOfSpeech() { //The user has started
    to speak.
}

log.d(TAG, "onBeginningOfSpeech");

public void onRmsChanged(float rmsdB) { //Used for receiving
    notifications from the SpeechRecognizer when the recognition
    related events occur. All the callbacks are executed on the
    Application main thread
}

log.d(TAG, "onRmsChanged");

public void onBufferReceived(byte[] buffer) { //More sound
    has been received.
}

log.d(TAG, "onBufferReceived");

public void onEndOfSpeech() { //Called after the user stops
    speaking.
}

log.d(TAG, "onEndOfSpeech");

public void onEvent(int error) { //A network or recognition
    error occurred.
}

log.d(TAG, "error " + error);
error += "\n" + error;
if (error == 1) {
    errorStatus.setText("ERROR_NETWORK_TIMEOUT");
} else if (error == 2) {
    errorStatus.setText("ERROR_NETWORK");
} else if (error == 3) {
    errorStatus.setText("ERROR_AUDIO");
} else if (error == 4) {
    errorStatus.setText("ERROR_SERVER");
} else if (error == 5) {
    errorStatus.setText("ERROR_UNKNOWN");
} else if (error == 6) {
    errorStatus.setText("ERROR_SPEECH_TIMEOUT");
} else if (error == 7) {
    errorStatus.setText("ERROR_NO_MATCH");
} else if (error == 8) {
    errorStatus.setText("ERROR_RECOGNIZER_BUSY");
} else if (error == 9) {
    errorStatus.setText("ERROR_INSUFFICIENT_PERMISSIONS");
}
}

```

```

        e.printStackTrace();
    } finally {
        sr.startListening(getIntent());
    }
}

public void onResults(Bundle results) {
    Log.d(TAG, "onResults " + results);
    errorID.setText("");
    errorStatus.setText("");
    //read data from db
    try {
        String s1 = new String("1");
        long l1 = Long.parseLong(s1);

        String s2 = new String("2");
        long l2 = Long.parseLong(s2);

        String s3 = new String("3");
        long l3 = Long.parseLong(s3);

        String s4 = new String("4");
        long l4 = Long.parseLong(s4);

        String s5 = new String("5");
        long l5 = Long.parseLong(s5);

        String s6 = new String("6");
        long l6 = Long.parseLong(s6);

        DBAdapter db = new DBAdapter(this);
        try {
            db.open();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        String returnedCommandNext =
db.getCommandNext(l1);
        String returnedCommandBack =
db.getCommandBack(l2);
        String returnedCommandStart =
db.getCommandStart(l3);
        String returnedCommandStop =
db.getCommandStop(l4);
        String returnedCommandFirst =
db.getCommandStop(l5);
        String returnedCommandLast =
db.getCommandStop(l6);
        db.close();
        //end read data db, check for result
        ArrayList<String> data =
results.getStringArrayList(SpeechRecognizer.RESULTS_RECOGNITIO
N);

        lv.setAdapter(new
ArrayAdapter<String>(getApplicationContext(),
android.R.layout.simple_list_item_1, data));

        if (data.contains(returnedCommandNext)) {
            mCommandService.write(BTService.KEY_RIGHT);

```

```

        e.printStackTrace();
    } finally {
        sr.startListening(getIntent());
    }

    public void onStart() {
        Log.d(TAG, "onStart");
        errorID.setText("");
        errorText.setText("");
        //read data from db
        try {
            String s1 = new String("1");
            long l1 = Long.parseLong(s1);

            String s2 = new String("2");
            long l2 = Long.parseLong(s2);

            String s3 = new String("3");
            long l3 = Long.parseLong(s3);

            String s4 = new String("4");
            long l4 = Long.parseLong(s4);

            String s5 = new String("5");
            long l5 = Long.parseLong(s5);

            String s6 = new String("6");
            long l6 = Long.parseLong(s6);

            DBHelper db = new DBHelper(this);
            try {
                db.open();
                e.printStackTrace();
            }
            String returnedCommandText =
                db.getCommandText(1);
            String returnedCommandText =
                db.getCommandText(2);
            String returnedCommandText =
                db.getCommandText(3);
            String returnedCommandText =
                db.getCommandText(4);
            String returnedCommandText =
                db.getCommandText(5);
            String returnedCommandText =
                db.getCommandText(6);
            db.close();
            //and read data db, check for result
            ArrayList<String> data =
                results.getStringArrayList(SpeechRecognizer.RESULTS_RECOGNITION);
            if (data != null) {
                ArrayList<String> (dataApplicationContext);
                ArrayList<String> (data);
            }
            if (data != null) {
                if (data.contains(returnedCommandText))
                    commandService.write(BTService.KEY_RIGHT);
            }
        }
    }

```

```

        } else if (data.contains(returnedCommandBack)) {
            mCommandService.write(BTService.KEY_LEFT);
        } else if (data.contains(returnedCommandStart)) {
            mCommandService.write(BTService.KEY_START);
        } else if (data.contains(returnedCommandStop)) {
            mCommandService.write(BTService.KEY_STOP);
        } else if (data.contains(returnedCommandFirst)) {
            mCommandService.write(BTService.KEY_FIRST);
        } else if (data.contains(returnedCommandLast)) {
            mCommandService.write(BTService.KEY_LAST);
        } else {
            //do nothing
        }
        try {
            Thread.sleep(2000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        } finally {
            sr.startListening(getIntent());
        }
    } catch (Exception e) {
    }
}

public void onPartialResults(Bundle partialResults)
//Called when partial recognition results are available.
{
    Log.d(TAG, "onPartialResults");
}

public void onEvent(int eventType, Bundle params)
//Reserved for adding future events.
{
    Log.d(TAG, "onEvent " + eventType);
}

//End of Speech Recognition Listener
}

```

h. Input IP Wifi

```

public class WifiIPActivity extends Activity {

    //Debugging
    private static final String TAG = "WifiIPActivity";
    private static final boolean D = true;
    //Return Intent extra
    public static String IP_ADDRESS = "ip_address";
    private boolean isLight;
    private int currentTheme;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        toggleTheme();

        //Setup the window

        requestWindowFeature(Window.FEATURE_INDETERMINATE_PROGRESS);
        setContentView(R.layout.wifi_ip);
    }
}

```



```

        //Set result CANCELED in case the user backs out
        setResult(Activity.RESULT_CANCELED);

        //Get IP from textview
        final TextView textIP = (TextView)
        findViewById(R.id.textIPAddress);

        //Initialize the button to perform device discovery
        Button ConnectIP = (Button)
        findViewById(R.id.btnConnectIP);
        ConnectIP.setOnClickListener(new
        View.OnClickListener() {
            @Override
            public void onClick(View view) {
                String IPaddress =
                textIP.getText().toString();

                //create the result intent
                Intent IPintent = new Intent();
                IPintent.putExtra(IP_ADDRESS, IPaddress);

                //set result and finish activity
                setResult(Activity.RESULT_OK, IPintent);
                finish();
            }
        });
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
    }
}

```

i. Speech Recognition Wifi

```

public class SpeechRecognitionWIFI extends Activity implements
RecognitionListener {

    static final int check = 1111;

    // Intent request codes
    private static final int REQUEST_CONNECT_IP = 1;

    private static final String TAG = "MyVoiceNoBox";

    //SpeechRecognition component
    TextView errorID, errorStatus;
    CheckBox AutoRecog;
    Button StartRecognize;
    ListView lv;
    SpeechRecognizer sr;

    private boolean isLight;
    private int currentTheme;
    private Socket client;
    private PrintWriter printwriter;
    private String message, address;

```



```

@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    // Set up the window layout
    toggleTheme();
    setContentView(R.layout.speech_recog);

    // Set SpeechRecognitionBT component
    errorID = (TextView) findViewById(R.id.tvErrorID);
    errorStatus =
(TextView) findViewById(R.id.tvErrorStatus);
    lv = (ListView) findViewById(R.id.lvVoiceReturn);
    StartRecognize = (Button) findViewById(R.id.bVoice);
    sr = SpeechRecognizer.createSpeechRecognizer(this);
    sr.setRecognitionListener(this);
}

@Override
protected void onDestroy() {
    super.onDestroy();
    sr.destroy();
}

@Override
public boolean onKeyDown(int keyCode, KeyEvent event) {
    if (keyCode == KeyEvent.KEYCODE_BACK) {
        String btnText =
StartRecognize.getText().toString();
        if(btnText == "Stop Recognize") {
            sr.destroy();
            finish();
            startActivity(getIntent());
        }else{
            finish();
        }
        return true;
    }
    return false;
}

private void openWIFIButtonR() {
    // Launch BT Speech Recognition
    Intent WIFIButtonR = new Intent(this,
ButtonRemoteWIFI.class);
    startActivity(WIFIButtonR);
    this.finish();
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.option_menu, menu);
    return true;
}

@Override
public boolean onOptionsItemSelected(MenuItem item) {
    switch (item.getItemId()) {

```



```

        case R.id.scan:
            // Launch the WifiIPActivity to input IP
            Intent serverIPIntent = new Intent(this,
WifiIPActivity.class);
            startActivityForResult(serverIPIntent,
REQUEST_CONNECT_IP);
            return true;
        case R.id.swapRemote:
            openWIFIButtonR();
            return true;
    }
    return false;
}

public void onActivityResult(int requestCode, int
resultCode, Intent data) {
    switch (requestCode) {
        case REQUEST_CONNECT_IP:
            // When DeviceListActivity returns with a
device to connect
            if (resultCode == Activity.RESULT_OK) {
                // Get the device MAC address
                address = data.getExtras()

.getString(WifiIPActivity.IP_ADDRESS);
            }
            break;
    }
}

public void StartRecog(View view) {
    StartRecognize.setText("Stop Recognize");
    String btnText = StartRecognize.getText().toString();
    AutoRecognize();
    if(btnText == "Stop Recognize") {
        StartRecognize.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {
                sr.destroy();
                finish();
                startActivity(getIntent());
            }
        });
    }
}

public void AutoRecognize(){
    Intent intent = new
Intent(RecognizerIntent.ACTION_RECOGNIZE_SPEECH);

intent.putExtra(RecognizerIntent.EXTRA_LANGUAGE_MODEL,Recogniz
erIntent.LANGUAGE_MODEL_FREE_FORM);

intent.putExtra(RecognizerIntent.EXTRA_CALLING_PACKAGE,"skrips
i.vudroidwispa");

    intent.putExtra(RecognizerIntent.EXTRA_MAX_RESULTS,5);
}

```

[illegible]

```
intent.putExtra(RecognizerIntent.EXTRA_SPEECH_INPUT_COMPLETE_SILENCE_LENGTH_MILLIS, 2000); //The amount of time that it should take after we stop hearing speech to consider the input complete.
```

```
intent.putExtra(RecognizerIntent.EXTRA_SPEECH_INPUT_MINIMUM_LENGTH_MILLIS, 2000); //The minimum length of an utterance.
```

```
intent.putExtra(RecognizerIntent.EXTRA_SPEECH_INPUT_POSSIBLY_COMPLETE_SILENCE_LENGTH_MILLIS, 2000); //The amount of time that it should take after we stop hearing speech to consider the input possibly complete.
```

```
    sr.startListening(intent);  
    Log.i("111111", "11111111");
```

```
}
```

```
//Speech Recognition Listener  
public void onReadyForSpeech(Bundle params) //Called when the endpointer is ready for the user to start speaking.
```

```
{  
    Log.d(TAG, "onReadyForSpeech");  
}
```

```
public void onBeginningOfSpeech() //The user has started to speak.
```

```
{  
    Log.d(TAG, "onBeginningOfSpeech");  
}
```

```
public void onRmsChanged(float rmsdB) //Used for receiving notifications from the SpeechRecognizer when the recognition related events occur. All the callbacks are executed on the Application main thread
```

```
{  
    Log.d(TAG, "onRmsChanged");  
}
```

```
public void onBufferReceived(byte[] buffer) //More sound has been received.
```

```
{  
    Log.d(TAG, "onBufferReceived");  
}
```

```
public void onEndOfSpeech() //Called after the user stops speaking.
```

```
{  
    Log.d(TAG, "onEndofSpeech");  
}
```

```
public void onError(int error) //A network or recognition error occurred.
```

```
{  
    Log.d(TAG, "error " + error);  
    errorID.setText("error " + error);  
    if (error == 1) {  
        errorStatus.setText(" ERROR_NETWORK_TIMEOUT");  
    } else if (error == 2) {  
        errorStatus.setText(" ERROR_NETWORK ");  
    } else if (error == 3) {  
        errorStatus.setText(" ERROR_AUDIO ");  
    } else if (error == 4) {  
        errorStatus.setText(" ERROR_SERVER ");  
    } else if (error == 5) {  
        errorStatus.setText(" ERROR_CLIENT ");  
    }  
}
```

```

        intent.putExtra(Intent.EXTRA_SPEECH_INPUT_COMPLETE,
            intent.getBooleanExtra(Intent.EXTRA_SPEECH_INPUT_COMPLETE, false));
        intent.putExtra(Intent.EXTRA_SPEECH_INPUT_MINIMUM_LENGTH,
            intent.getIntExtra(Intent.EXTRA_SPEECH_INPUT_MINIMUM_LENGTH, 0));
        //The amount of time that it
        should take after we stop hearing speech to consider the input
        complete.
    }

```

```

        intent.putExtra(Intent.EXTRA_SPEECH_INPUT_MINIMUM_LENGTH,
            intent.getIntExtra(Intent.EXTRA_SPEECH_INPUT_MINIMUM_LENGTH, 0));
        //The minimum length of an utterance.
    }

```

```

        intent.putExtra(Intent.EXTRA_SPEECH_INPUT_COMPLETE,
            intent.getBooleanExtra(Intent.EXTRA_SPEECH_INPUT_COMPLETE, false));
        intent.putExtra(Intent.EXTRA_SPEECH_INPUT_MINIMUM_LENGTH,
            intent.getIntExtra(Intent.EXTRA_SPEECH_INPUT_MINIMUM_LENGTH, 0));
        //The amount of time
        that it should take after we stop hearing speech to consider
        the input complete.
        startListening(intent);
        Log.d(TAG, "startListening");
    }

```

```

        //Speech Recognition Listener
        public void onReadyToSpeak() {
            //Called when
            the recognizer is ready for the user to start speaking.
        }

```

```

        Log.d(TAG, "onReadyToSpeak");
    }

    public void onBeginnningOfSpeech() {
        //The user has started
        to speak.
    }

```

```

        Log.d(TAG, "onBeginnningOfSpeech");
    }

    public void onRecognized(float aDB) {
        //Used for receiving
        notifications from the SpeechRecognizer when the recognition
        related events occur. All the callbacks are executed on the
        Application main thread.
    }

```

```

        Log.d(TAG, "onRecognized");
    }

    public void onUnrecognized(float aDB) {
        //Used for receiving
        notifications from the SpeechRecognizer when the recognition
        related events occur. All the callbacks are executed on the
        Application main thread.
    }

```

```

        Log.d(TAG, "onUnrecognized");
    }

    public void onEndOfSpeech() {
        //Called after the user stops
        speaking.
    }

```

```

        Log.d(TAG, "onEndOfSpeech");
    }

    public void onError(int error) {
        //A network or recognition
        error occurred.
    }

```

```

        Log.d(TAG, "onError: " + error);
        Log.d(TAG, "onError: " + error);
        if (error == 1) {
            Log.d(TAG, "ERROR_NETWORK_TIMEOUT");
        } else if (error == 2) {
            Log.d(TAG, "ERROR_NETWORK");
        } else if (error == 3) {
            Log.d(TAG, "ERROR_AUDIO");
        } else if (error == 4) {
            Log.d(TAG, "ERROR_SERVER");
        } else if (error == 5) {
            Log.d(TAG, "ERROR_CLIENT");
        }
    }

```

```

    } else if (error == 6) {
        errorStatus.setText(" ERROR_SPEECH_TIMEOUT ");
    } else if (error == 7) {
        errorStatus.setText(" ERROR_NO_MATCH ");
    } else if (error == 8) {
        errorStatus.setText(" ERROR_RECOGNIZER_BUSY ");
    } else if (error == 9) {
        errorStatus.setText("
ERROR_INSUFFICIENT_PERMISSIONS ");
    }
    try {
        Thread.sleep(2000);
    } catch (InterruptedException e) {
        e.printStackTrace();
    } finally {
        sr.startListening(getIntent());
    }
}

public void onResults(Bundle results) {
    Log.d(TAG, "onResults " + results);
    errorID.setText("");
    errorStatus.setText("");
    //read data from db
    try {
        String s1 = new String("1");
        long l1 = Long.parseLong(s1);

        String s2 = new String("2");
        long l2 = Long.parseLong(s2);

        String s3 = new String("3");
        long l3 = Long.parseLong(s3);

        String s4 = new String("4");
        long l4 = Long.parseLong(s4);

        String s5 = new String("5");
        long l5 = Long.parseLong(s5);

        String s6 = new String("6");
        long l6 = Long.parseLong(s6);

        DBAdapter db = new DBAdapter(this);
        try {
            db.open();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        String returnedCommandNext =
db.getCommandNext(l1);
        String returnedCommandBack =
db.getCommandBack(l2);
        String returnedCommandStart =
db.getCommandStart(l3);
        String returnedCommandStop =
db.getCommandStop(l4);
        String returnedCommandFirst =
db.getCommandStop(l5);

```

```

        } else if (error == 3) {
            errorStatus.setText(" ERROR_SPEECH_TIMEOUT ");
        } else if (error == 7) {
            errorStatus.setText(" ERROR_NO_MATCH ");
        } else if (error == 8) {
            errorStatus.setText(" ERROR_RECOGNIZE_BUSY ");
        } else if (error == 9) {
            errorStatus.setText(" ERROR_INSUFFICIENT_PERMISSIONS ");
        }
    }

    try {
        Thread.sleep(2000);
    } catch (InterruptedException e) {
        e.printStackTrace();
    } finally {
        startListening(getIntent());
    }

    public void onResult(Bundle results) {
        Log.d(TAG, "onResult: " + results);
        errorStatus.setText("");
        //read data from db
        try {
            String s1 = new String("1");
            long l1 = Long.parseLong(s1);

            String s2 = new String("2");
            long l2 = Long.parseLong(s2);

            String s3 = new String("3");
            long l3 = Long.parseLong(s3);

            String s4 = new String("4");
            long l4 = Long.parseLong(s4);

            String s5 = new String("5");
            long l5 = Long.parseLong(s5);

            String s6 = new String("6");
            long l6 = Long.parseLong(s6);

            Database db = new Database(this);
            try {
                db.open();
            } catch (SQLException e) {
                e.printStackTrace();
            }

            String returnedCommand1 = db.getCommandBack(1);
            String returnedCommand2 = db.getCommandBack(2);
            String returnedCommand3 = db.getCommandBack(3);
            String returnedCommand4 = db.getCommandBack(4);
            String returnedCommand5 = db.getCommandBack(5);
            String returnedCommand6 = db.getCommandBack(6);
        }
    }

```

```

        String returnedCommandLast =
db.getCommandStop(16);
        db.close();
        //end read data db, check for result
        ArrayList<String> data =
results.getStringArrayList(SpeechRecognizer.RESULTS_RECOGNITION);
        lv.setAdapter(new
ArrayAdapter<String>(getApplicationContext(),
android.R.layout.simple_list_item_1, data));

        if (data.contains(returnedCommandNext)) {
            message = "Next".toString();
            SendMessage sendMessageTask = new
SendMessage();
            sendMessageTask.execute();
        } else if (data.contains(returnedCommandBack)) {
            message = "Back".toString();
            SendMessage sendMessageTask = new
SendMessage();
            sendMessageTask.execute();
        } else if (data.contains(returnedCommandStart)) {
            message = "Start".toString();
            SendMessage sendMessageTask = new
SendMessage();
            sendMessageTask.execute();
        } else if (data.contains(returnedCommandStop)) {
            message = "Stop".toString();
            SendMessage sendMessageTask = new
SendMessage();
            sendMessageTask.execute();
        } else if (data.contains(returnedCommandFirst)) {
            message = "First".toString();
            SendMessage sendMessageTask = new
SendMessage();
            sendMessageTask.execute();
        } else if (data.contains(returnedCommandLast)) {
            message = "Last".toString();
            SendMessage sendMessageTask = new
SendMessage();
            sendMessageTask.execute();
        } else {
            //do nothing
        }
        try {
            Thread.sleep(2000);
        } catch (InterruptedException e) {
            e.printStackTrace();
        } finally {
            sr.startListening(getIntent());
        }
    } catch (Exception e) {
    }
}

public void onPartialResults(Bundle partialResults)
//Called when partial recognition results are available.
{
    Log.d(TAG, "onPartialResults");
}

```

[illegible]

```

    }
    public void onEvent(int eventType, Bundle params)
    //Reserved for adding future events.
    {
        Log.d(TAG, "onEvent " + eventType);
    }
    //End of Speech Recognition Listener

    private class SendMessage extends AsyncTask<Void, Void,
    Void> {

        @Override
        protected Void doInBackground(Void... params) {
            try {

                client = new Socket(address, 7777); // connect
to the server
                printwriter = new
                PrintWriter(client.getOutputStream(), true);
                printwriter.write(message); // write the
message to output stream

                printwriter.flush();
                printwriter.close();
                client.close(); // closing the connectivity

            } catch (UnknownHostException e) {
                e.printStackTrace();
            } catch (IOException e) {
                e.printStackTrace();
            }
            return null;
        }
    }
}

```

ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN SPEECH RECOGNITION

NAMA RESPONDEN : AKH. FARID

ALAMAT : lumajang

PEKERJAAN : mahasiswa.



AKH. FARID

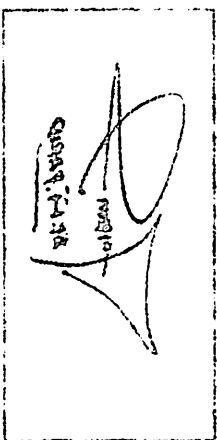
1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
2. Bagaimana *user interface* atau tampilan aplikasi ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang

ԻՄՔԱՅԵՐԵՐԿԱ

Խոն 7- ՀԱՅԿԱՊԱՏԵՐԻ ԵՐԵՎԱՆԻ ԶԵՆԻՏԱԿԱՆ ԱՍՏՈՒԿԱՆԻՆԻ : ԼԻՍՏԻ
ՄՈՋԵՐԵՆ ԿԱՌԱՐԱՐՈՒՄԸ ԳՐԱԴԱՆԱ ՏԻՐԱԿՐԵՐԻ ՆԳԻՐ

ՆՈՒՄԱՐԱԿԱՆ

ԳՐԱԴԱՆԱԿԱՆ : ՀԱՅԿԱՊԱՏԵՐԻ ԱՐԿԱՆ
ԲԱՐՍԵՐՈՒՄ : 7776111
ՇՆՈՐՀԱՅԵՐԿԱ : ԿԱՌԱՐԱՐՈՒՄ



1. Բաճարման քննիչի կողմից կատարված զննման արդյունքները համաձայնեցվել են հետևյալ կերպով:

Մեր խնդիրը անհրաժեշտ է պարզաբանել և համաձայնեցվել է հետևյալ կերպով:

☒ Եսկ

☐ Շեղծ

☐ Եսկ

☐ Շեղծ

2. Բաճարման քննիչի կողմից կատարված զննման արդյունքները համաձայնեցվել են հետևյալ կերպով:

☒ Եսկ

☐ Շեղծ

3. Բաճարման քննիչի կողմից կատարված զննման արդյունքները համաձայնեցվել են հետևյալ կերպով:

☒ Եսկ

☐ Շեղծ

4. Բաճարման քննիչի կողմից կատարված զննման արդյունքները համաձայնեցվել են հետևյալ կերպով:

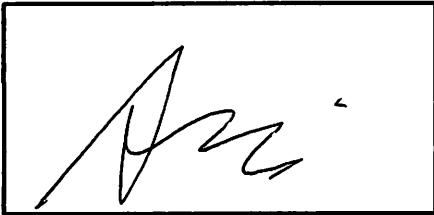
☒ Եսկ

☐ Շեղծ

ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN SPEECH RECOGNITION

NAMA RESPONDEN : Fahmi
ALAMAT : Malang
PEKERJAAN : Mahasiswa

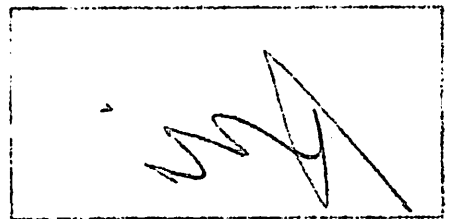


1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
2. Bagaimana *user interface* atau tampilan aplikasi ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☐ a) Baik
☒ b) Cukup
☐ c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
☐ a) Baik
☒ b) Cukup
☐ c) Kurang

ANGKET SURvei

JUDUL : MEMBANGUN APLIKASI BEKOTE PRESENTATION SYSTEM
WISPA, BERKAS, ANBOR, MENGGUNAKAN SPEECH
RECOGNITION

NAME RESPONDER : Fahmi
ALAMAT : Malang
PEKERJAAN : Mahasiswa



1. Bagaimana kinerja dari aplikasi yang presentation Word ini?
a) Baik ☒
b) Cukup ☐
c) Kurang ☐
2. Bagaimana cara bekerja atau tampilan aplikasi ini?
a) Baik ☒
b) Cukup ☐
c) Kurang ☐
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
a) Baik ☒
b) Cukup ☐
c) Kurang ☐
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah sudah baik atau tidak?
a) Baik ☒
b) Cukup ☐
c) Kurang ☐
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
a) Baik ☒
b) Cukup ☐
c) Kurang ☐

ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN SPEECH RECOGNITION

NAMA RESPONDEN : M. Alfian Ulinnuha

ALAMAT : Mulyorejo, Gg Keramat - Sukun - Malang.

PEKERJAAN : Mahasiswa

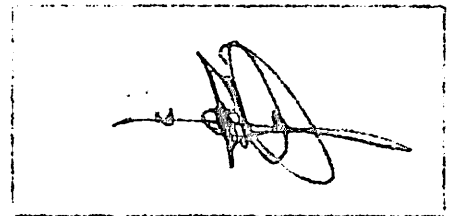


1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
2. Bagaimana *user interface* atau tampilan aplikasi ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang

ANGKET PENELITIAN

JUDUL : PENELITIAN TENTANG PERUBAHAN PERILAKU KESEHATAN MASYARAKAT
LOKASI : KOTA SURABAYA
WAKTU : 10 Hari

KELOMPOK : M. Alfian Alimulhuda
ANGKATAN : M. Alfian Alimulhuda - 2010 - 2011
MATERI : M. Alfian Alimulhuda



1. Bagaimana pendapat anda tentang penelitian ini?
☒ Baik
☐ Buruk
2. Bagaimana pendapat anda tentang penelitian ini?
☒ Baik
☐ Buruk
3. Bagaimana pendapat anda tentang penelitian ini?
☒ Baik
☐ Buruk
4. Bagaimana pendapat anda tentang penelitian ini?
☒ Baik
☐ Buruk
5. Bagaimana pendapat anda tentang penelitian ini?
☒ Baik
☐ Buruk

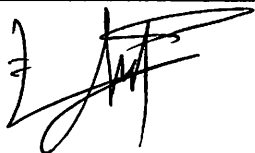
ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN SPEECH RECOGNITION

NAMA RESPONDEN : EDI Dwi HARONO

ALAMAT : Malang

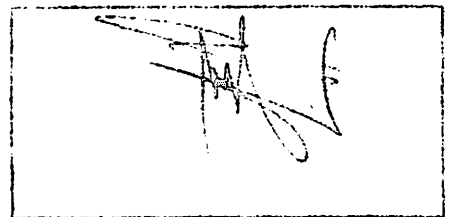
PEKERJAAN : Mahasiswa



1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
☒ a) Baik ☐ c) Kurang
☐ b) Cukup
2. Bagaimana *user interface* atau tampilan aplikasi ini?
☒ a) Baik ☐ c) Kurang
☐ b) Cukup
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☐ a) Baik ☐ c) Kurang
☒ b) Cukup
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
☒ a) Baik ☐ c) Kurang
☐ b) Cukup
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
☒ a) Baik ☐ c) Kurang
☐ b) Cukup

RECOGNITION
WISPR, BERBASIS ANDROID MENGGUNAKAN SPEECH
JUDUL : MEMBANGUN APLIKASI RELATIF PRESENTATION 22 MAREK

NAMA RESPONDEN : EDI WU HARORO
ALAMAT : KAWUNG
PEKERJAAN : KAWUNG



1. Bagaimana kinerja dari aplikasi tersebut pada perangkat Android Wispr ini?
a) Baik ☒
b) Cukup ☐
c) Kurang ☐
2. Bagaimana cara pengguna atau tampilan aplikasi ini?
a) Baik ☒
b) Cukup ☐
c) Kurang ☐
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
a) Baik ☒
b) Cukup ☐
c) Kurang ☐
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini sesuai dengan baik atau tidak?
a) Baik ☒
b) Cukup ☐
c) Kurang ☐
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
a) Baik ☒
b) Cukup ☐
c) Kurang ☐

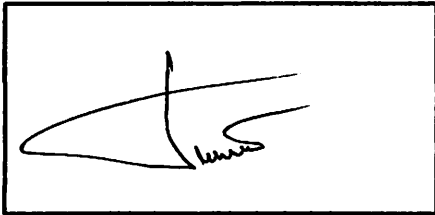
ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN SPEECH RECOGNITION

NAMA RESPONDEN : Yosi Chandra Kurniawan

ALAMAT : Ds. Gembrung

PEKERJAAN : Mahasiswa

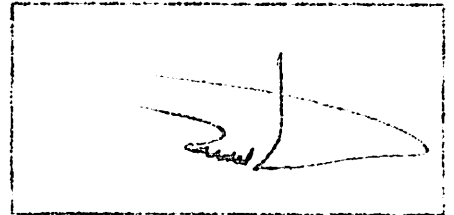


1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
2. Bagaimana *user interface* atau tampilan aplikasi ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang

ANNEX 1

RECOGNITION
TUGAS BERBASIS ANDROID NEGOTIATION
JUDUL : MEMBANGUN APLIKASI NEGOTIATION

NAME : Yosi Chandra Kusumawan
ALAMAT : Jl. Cempaka
PEKERJAAN : Mahasiswa

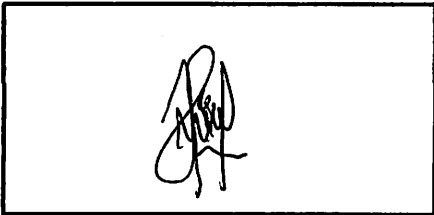


1. Bagaimana kinerja dari aplikasi ini? (berdasarkan Wawancara)
☒ (a) Baik
☐ (b) Cukup
☐ (c) Kurang
2. Bagaimana cara pengguna atau tampilan aplikasi ini?
☒ (a) Baik
☐ (b) Cukup
☐ (c) Kurang
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☒ (a) Baik
☐ (b) Cukup
☐ (c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah sejalan dengan hasil yang diharapkan?
☒ (a) Baik
☐ (b) Cukup
☐ (c) Kurang
5. Bagaimana menurut Anda tampilan yang diberikan oleh aplikasi ini?
☒ (a) Baik
☐ (b) Cukup
☐ (c) Kurang

ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN SPEECH RECOGNITION

NAMA RESPONDEN : Denny Arif Kurniawan
ALAMAT : Pasuruan
PEKERJAAN : Mahasiswa



1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
2. Bagaimana *user interface* atau tampilan aplikasi ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☐ a) Baik ☒ b) Cukup ☐ c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
☐ a) Baik ☒ b) Cukup ☐ c) Kurang
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang

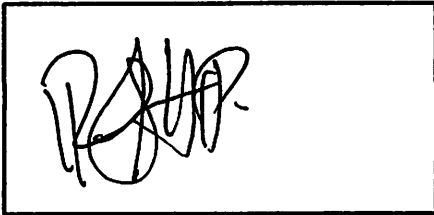
ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN SPEECH RECOGNITION

NAMA RESPONDEN : Rizki Utama

ALAMAT : K. Ploso

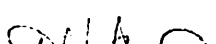
PEKERJAAN : Mahasiswa



1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
2. Bagaimana *user interface* atau tampilan aplikasi ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang

RECOGNITION

1. **RESEARCHER** : Mr. John Doe
 2. **RESEARCHER** : Mr. John Doe
 3. **RESEARCHER** : Mr. John Doe



1. Bagaimana kinerja dan kualitas pelayanan / obidit W / apa ini?
 a) Baik ~~X~~
 b) Cukup
 c) Kurang
2. Bagaimana kerja wawakel atau isipile / pilikid ini?
 a) Baik ~~X~~
 b) Cukup
 c) Kurang
3. Bagaimana tingkat kermatiran dalam menggunakan aplikasi ini?
 a) Baik ~~X~~
 b) Cukup
 c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
 a) Baik ~~X~~
 b) Cukup
 c) Kurang
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
 a) Baik ~~X~~
 b) Cukup
 c) Kurang

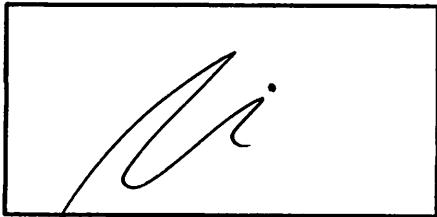
ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN SPEECH RECOGNITION

NAMA RESPONDEN : A.ZAMROZ

ALAMAT : Limasan

PEKERJAAN : Mahasiswa



1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
a) Baik
☒ b) Cukup
c) Kurang
2. Bagaimana *user interface* atau tampilan aplikasi ini?
a) Baik
☒ b) Cukup
c) Kurang
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☒ a) Baik
b) Cukup
c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
☒ a) Baik
b) Cukup
c) Kurang
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
☒ a) Baik
b) Cukup
c) Kurang

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION "ANDREW"
MATERI BERBASIS ANDROID MENGGUNAKAN SPEECH
RECOGNITION

NAMA RESPONDEN : A. A. A.

ALAMAT : ...

PEKERJAAN : ...



1. Bagaimana kinerja dari aplikasi remote presentation / andrew ini?

- a) Baik ☒
b) Cukup ☒
c) Kurang ☐

2. Bagaimana cara kerja dari aplikasi ini?

- a) Baik ☒
b) Cukup ☒
c) Kurang ☐

3. Bagaimana tingkat kenyamanan dalam menggunakan aplikasi ini?

- a) Baik ☒
b) Cukup ☒
c) Kurang ☐

4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?

- a) Baik ☒
b) Cukup ☒
c) Kurang ☐

5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?

- a) Baik ☒
b) Cukup ☒
c) Kurang ☐

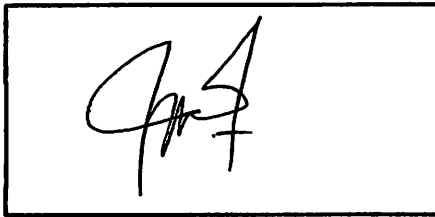
ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN SPEECH RECOGNITION

NAMA RESPONDEN : Rega Aditya. P

ALAMAT : Nganjuk

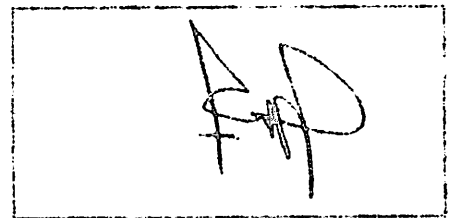
PEKERJAAN : Mahasiswa



1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
a) Baik
☒ b) Cukup
c) Kurang
2. Bagaimana *user interface* atau tampilan aplikasi ini?
a) Baik
☒ b) Cukup
c) Kurang
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☒ a) Baik
b) Cukup
c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
a) Baik
☒ b) Cukup
c) Kurang
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
☒ a) Baik
b) Cukup
c) Kurang

TITLE : MEMBRANGUN APLIKASI REMOTE PRESENTATION "VISION"
BERBASIS ANDROID MELALUI APLIKASI RESPON
RECOGNITION

NAMA RESPONDEN : Rodo Agatha. P
ALAMAT : Hologram
PROFESI : Mahasiswa



1. Bagaimana cara anda aplikasi remote presentation / android / java ini?
a) Baik
b) Cukup ☒
c) Kurang
2. Bagaimana cara anda aplikasi remote presentation / android / java ini?
a) Baik
b) Cukup ☒
c) Kurang
3. Bagaimana fungsi remote presentation / android / java ini?
a) Baik
b) Cukup ☒
c) Kurang
4. Bagaimana fungsi remote presentation / android / java ini apakah sudah baik atau tidak?
a) Baik
b) Cukup ☒
c) Kurang
5. Bagaimana menurut Anda materi yang diberikan oleh aplikasi ini?
a) Baik
b) Cukup ☒
c) Kurang

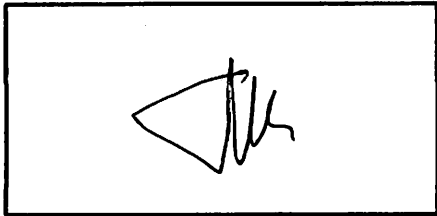
ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN SPEECH RECOGNITION

NAMA RESPONDEN : Ruwatho Azzalawon

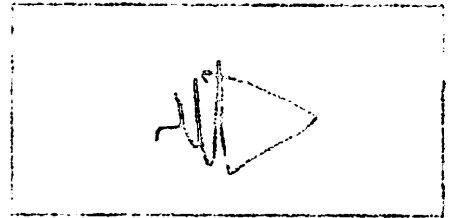
ALAMAT : Trenggalek

PEKERJAAN : mahasiswa



1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
a) Baik
☒ b) Cukup
c) Kurang
2. Bagaimana *user interface* atau tampilan aplikasi ini?
a) Baik
☒ b) Cukup
c) Kurang
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☒ a) Baik
b) Cukup
c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
☒ a) Baik
b) Cukup
c) Kurang
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
☒ a) Baik
b) Cukup
c) Kurang

NAMIA REPPINDEN : Puntad Assalamun
 ALIANT : Puntaduk
 PUNTERIAAN : Puntadun



1. Bagaimana kinerja dari aplikasi Kewas-weras dalam 7 hari ini?
a) Baik
b) Cukup
c) Kurang
2. Bagaimana user menyukai atau tampilan aplikasi ini?
a) Baik
b) Cukup
c) Kurang
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
a) Baik
b) Cukup
c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
a) Baik
b) Cukup
c) Kurang
5. Bagaimana menurut Anda masalah yang diberikan user aplikasi ini?
a) Baik
b) Cukup
c) Kurang

ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN GOOGLE SPEECH API

NAMA RESPONDEN : Aand April
ALAMAT : Jln. Simpang Golf
PEKERJAAN : Mahasiswa



1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
☒ a) Baik
b) Cukup
c) Kurang
2. Bagaimana *user interface* atau tampilan aplikasi ini?
☒ a) Baik
b) Cukup
c) Kurang
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☒ a) Baik
b) Cukup
c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
a) Baik
☒ b) Cukup
c) Kurang
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
☒ a) Baik
b) Cukup
c) Kurang

ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN GOOGLE SPEECH API

NAMA RESPONDEN : Mega Putri Darliana

ALAMAT : Jl. Golf 73 Tasikradu Kec. Lembang

PEKERJAAN : Mahasiswa

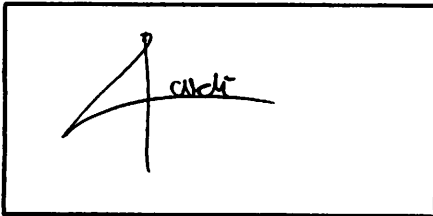


1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
2. Bagaimana *user interface* atau tampilan aplikasi ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang

ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN GOOGLE SPEECH API

NAMA RESPONDEN : Imam Hardi L
ALAMAT : Jl bordabur Agung
PEKERJAAN : Mahasiswa

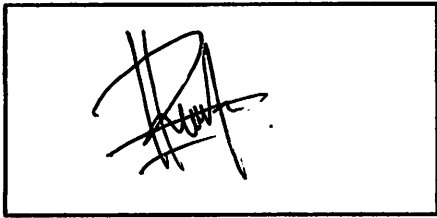


1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
2. Bagaimana *user interface* atau tampilan aplikasi ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang

ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN GOOGLE SPEECH API

NAMA RESPONDEN : Syahrizal Jusriah Agmar
ALAMAT : Jln. Sigura - gura
PEKERJAAN : Mahasiswa



1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
2. Bagaimana *user interface* atau tampilan aplikasi ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang

ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN GOOGLE SPEECH API

NAMA RESPONDEN : Edwin Tunanto
ALAMAT : Pandaan Jawa timur
PEKERJAAN : mahasiswa



1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
2. Bagaimana *user interface* atau tampilan aplikasi ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang

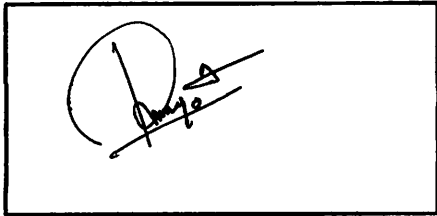
ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN GOOGLE SPEECH API

NAMA RESPONDEN : Ridha Fahrozi

ALAMAT : Malang

PEKERJAAN : Mahasiswa



1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
2. Bagaimana *user interface* atau tampilan aplikasi ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang

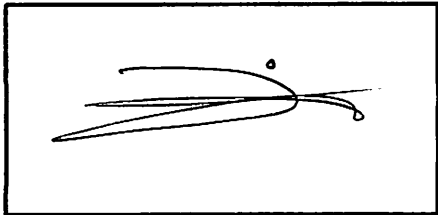
ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN GOOGLE SPEECH API

NAMA RESPONDEN : Jaka Rahmadi

ALAMAT : Jl. Karanglo

PEKERJAAN : Mahasiswa



1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
2. Bagaimana *user interface* atau tampilan aplikasi ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
☒ a) Baik
☐ b) Cukup
☐ c) Kurang

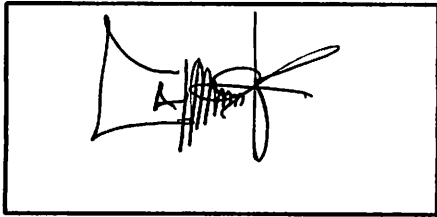
ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN GOOGLE SPEECH API

NAMA RESPONDEN : Diryan Almas

ALAMAT : Malang

PEKERJAAN : Mahasiswa



1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
2. Bagaimana *user interface* atau tampilan aplikasi ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
☒ a) Baik ☐ b) Cukup ☐ c) Kurang

ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN GOOGLE SPEECH API

NAMA RESPONDEN : Bondan Prakosa Waas

ALAMAT : Jalan simpang golf no. 75 Tasik Madu

PEKERJAAN : Mahasiswa



1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
☒ a) Baik ☐ c) Kurang
☐ b) Cukup
2. Bagaimana *user interface* atau tampilan aplikasi ini?
☒ a) Baik ☐ c) Kurang
☐ b) Cukup
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
☐ a) Baik ☐ c) Kurang
☒ b) Cukup
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
☒ a) Baik ☐ c) Kurang
☐ b) Cukup
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
☒ a) Baik ☐ c) Kurang
☐ b) Cukup

ANGKET SKRIPSI

JUDUL : MEMBANGUN APLIKASI REMOTE PRESENTATION “VuDroid Wispa” BERBASIS ANDROID MENGGUNAKAN GOOGLE SPEECH API

NAMA RESPONDEN : Fahriz Za'iem A

ALAMAT : Jl. Bulak Banteng lor Bhineka D/2g

PEKERJAAN : Mahasiswa.



1. Bagaimana kinerja dari aplikasi *remote presentation* VuDroid Wispa ini?
a) Baik c) Kurang
~~b) Cukup~~
2. Bagaimana *user interface* atau tampilan aplikasi ini?
~~a) Baik~~ c) Kurang
b) Cukup
3. Bagaimana tingkat kemudahan dalam menggunakan aplikasi ini?
~~a) Baik~~ c) Kurang
b) Cukup
4. Bagaimana fungsi-fungsi yang ada dalam aplikasi ini apakah berjalan dengan baik atau tidak?
a) Baik c) Kurang
~~b) Cukup~~
5. Bagaimana menurut Anda manfaat yang diberikan oleh aplikasi ini?
~~a) Baik~~ c) Kurang
b) Cukup