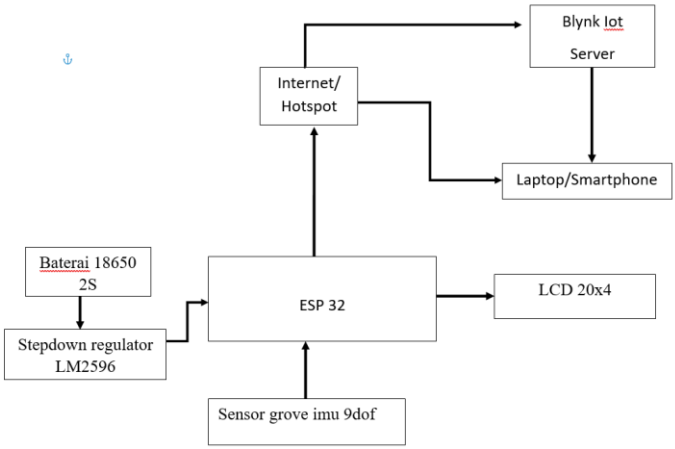
BAB III
METODOLOGI PENELITIAN

3.1 Perancangan Sistem

Bab ini menjelaskan tentang tahapan tahapan penelitian berupa perancangan sistem yang dibagi menjadi dua yaitu, perancangan perangkat keras *(hardware)* dan perancangan perangkat lunak *(software)*. Bagian tersebut disusun dan dirancang secara sistematis untuk menciptakan sebuah sistem yang sesuai dengan perancangan yang dapat berjalan sesuai dengan fungsinya.



Gambar 3. 1 Blok diagram sistem

Seperti yang terlihat pada gambar 3.1 mengenai blok diagram sistem pada penelitian ini komponen yang digunakan untuk melakukan penelitian yaitu sebagai berikut:

1. Baterai 18650 2S digunakan sebagai *input* catu daya alat.
2. *Stepdown regulator* berfungsi untuk menurunkan tegangan dari baterai 18650 2S dengan keluaran tegangan 8,4V menjadi 5 V dengan memutar potensiometer hingga pas 5 V yang kemudian tegangan *output* dilanjutkan untuk menghidupkan ESP 32, sensor IMU 9 Dof, dan LCD.
3. *Sensor* IMU 9 Dof berfungsi untuk mengirimkan hasil monitoring pergerakan tanah.

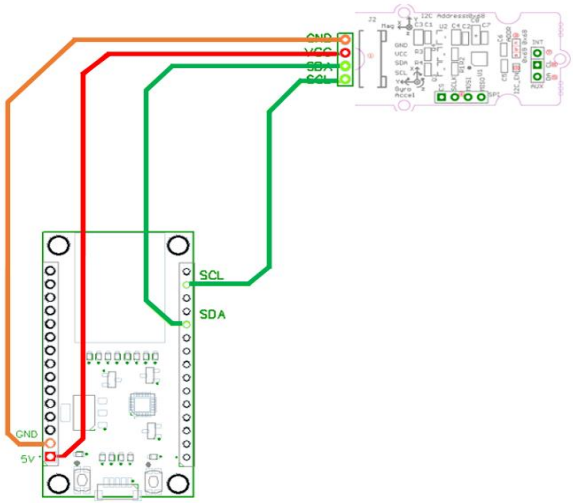
4. ESP 32 digunakan untuk memproses data dari sensor IMU 9 Dof dan mengirim data tersebut ke Blynk *server* sebagai *software* dari pembacaan sensor IMU 9 Dof.
5. LCD sebagai *hardware* yang berfungsi untuk menampilkan data pergeseran tanah.
6. Internet berfungsi untuk menghubungkan data dari ESP 32 ke Blynk *server.*
7. Blynk IoT *server* berfungsi untuk menampilkan data grafik yang berasal dari ESP 32.
8. Laptop/ *smartphone* berfungsi untuk melihat data pergeseran tanah

3.2 Perancangan Perangkat Keras (Hardware)

Tahap ini menjelaskan tentang tahap perancangan perangkat keras *(hardware)* dan rangkaian masing - masing sensor dengan ESP32 dan suplai daya.

3.2.1 Rangkaian Sensor IMU 9 Dof

Sensor IMU 9 Dof ini berfungsi untuk mengetahui pergeseran keadaan tanah. Perubahan dari pergeseran tersebut dengan melihat 3 *axis gyroscope* dan 3 *axis accelerometer s.*



Gambar 3.2 Rangkaian sensor IMU 9 Dof dengan ESP 32

Pada gambar 3.2 menampilkan cara menghubungkan antara ESP 32 dengan sensor IMU 9 Dof degan menggunakan pin SDA (*serial data)* dan SCL *(serial clock)* ke pin GPIO 22 *(general purpose input/ output)* dan GPIO 21.

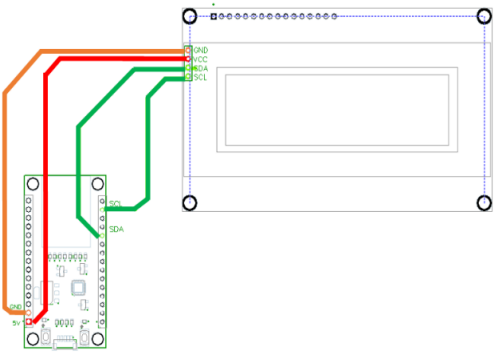
Tabel 3.1 Konfigurasi pin sensor IMU 9 Dof

Sensor Imu 9 Dof	**Esp 32**
Gnd	Gnd
Vcc	Vcc
SDA	GPIO 21
SCL	GPIO 22

Pada tabel 3.1 Pin SDA berfungsi untuk mengirimkan data dan pin SCL digunakan untuk mengatur waktu *transfer* data dan sinkronisasi antara ESP32 dan sensor IMU 9, sedangkan GPIO pada ESP32 adalah pin yang dapat diatur sebagai *input* atau *output* data.

3.2.2 Rangkaian LCD

Rangkaian ini berfungsi untuk menampilkan data pergeseran tanah dari sensor IMU 9 Dof pada LCD.



Gambar 3.3 Rangkaian LCD dengan ESP 32

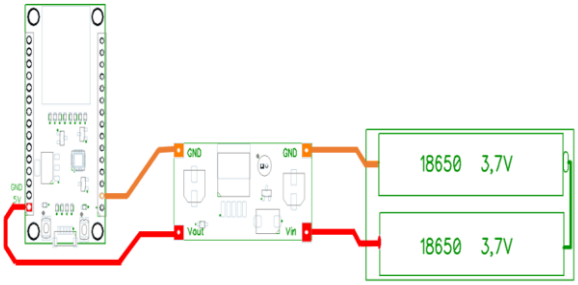
Pada gambar 3.3 menampilkan cara menghubungkan antara ESP 32 dan LCD dengan menggunakan pin SDA (serial data) dan SCL (serial clock) pada LCD ke pin SDA dan pin GPIO 21 dan pin GPIO 22 pada ESP 32.

Tabel 3.2 Konfigurasi pin LCD dan ESP 32

LCD	ESP 32
Gnd	Gnd
Vcc	Vcc
SDA	GPIO 21
SCL	GPIO 22

Pada tabel 3.2 Pin SDA berfungsi untuk mengirimkan data dan pin SCL digunakan untuk mengatur waktu *transfer* data dan sinkronisasi antara ESP32 dan LCD, sedangkan GPIO pada ESP32 adalah pin yang dapat diatur sebagai *input* atau *output* data.

3.2.3 Rangkaian ESP 32 dengan Catudaya



Gambar 3.4 Rangkaian ESP 32 dengan catudaya

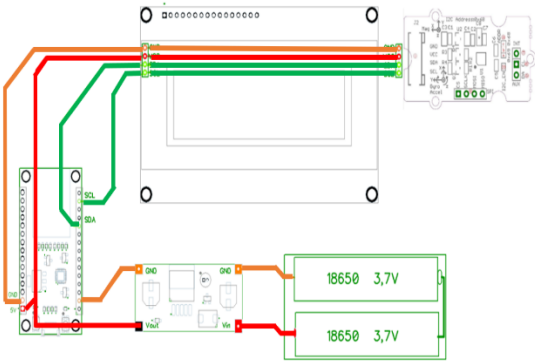
Pada gambar 3.4 menampilam cara merangkaian yang rangkaian ini berfungsi untuk mengatur *output* tegangan 8,4 V dari baterai dan diteruskan melalui *stepdown regulator* menjadi 5 V yang akan disalurkan ke ESP 32.

Tabel 3.3 Konfigurasi pin ESP 32, *stepdown regulator* dan baterai

ESP 32	*Stepdown regulator*	Baterai
V out 5 V Gnd	V out Gnd	V in Gnd

Pada tabel 3.3 pin V *out* 5V pada ESP 32 dihubungkan ke V *out stepdown regulator* dan V in baterai yang berfungsi sebagai sumber tegangan untuk ESP 32 dan sensor lainnya.

3.2.4 Rangkaian Keseluruhan Perangkat Keras



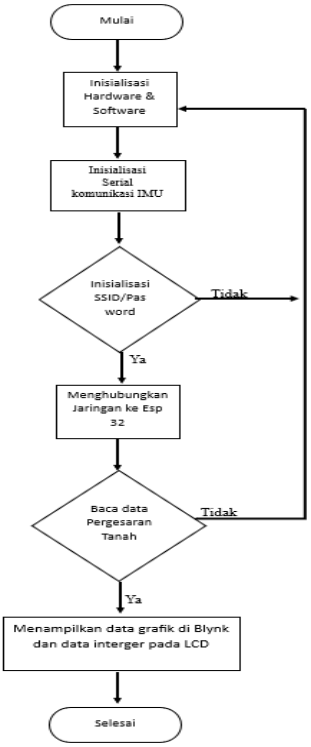
Gambar 3.5 Rangkaian sistem keseluruhan

Pada gambar 3.5 adalah seluruh rangkaian mulai dari rangkaian sensor IMU 9 Dof yang terhubung ke ESP 32, rangkaian LCD yang terhubung ke ESP 32, dan rangkaian ESP 32 yang terhubung pada catudaya diintergrasikan menjadi satu rangkaian sistem keseluruhan.

3.3 Perancangan Perangkat Lunak

Tahap perancangan perangkat lunak ini menjelaskan tentang program yang terhubung antara Nodemcu ESP 32, aplikasi Bylnk IoT dan LCD dengan menggunakan bahasa pemrograman C++ dan hasil pembacaan dari sensor IMU 9 Dof akan ditampilkan pada aplikasi Bylnk Iot sebagai *software* dan LCD sebagai tampilan *hardware.* Apabila tidak terhubung maka perlu dicek kembali pada *software* SSID, *password wifi*, dan *source code* yang digunakan dalam pengaplikasian sudah sesuai atau belum, dan pada *hardware* pastikan semua pin sudah sesuai dengan blok diagram yang ada pada gambar 3.5 rancangan keseluruhan perangkat keras.

3.3.1 Flowchart



Gambar 3. 6 *Flowchart*

Dari gambar 3.6 dapat kita ketahui bahwa cara kerja alat, dimulai dari sinkronisasi antara *hardware* (perangkat keras) yaitu rangkaian alat dan *software* (perangkat lunak) yaitu *coding* untuk rangkaian kemudian terhubung dengan sensor IMU 9 Dof dilanjutkan dengan pengecekan SSID/ *password* apa bila tidak terkoneksi maka akan kembali lagi ke sinkronisasi *hardware dan software* dan jika sudah terkoneksi maka dilanjutkan ke ESP 32 dan pada ESP 32 akan memproses *output* dari pembacaan sensor IMU 9 Dof apabila belum ada data yang keluar kemungkinan jaringan terputus atau tidak stabil, apabila ESP 32 dapat membaca dan mengolah data *output* pembacaan dari sensor IMU 9 Dof maka data grafik akan ditampilkan pada Blynk dan data *interger* akan muncul pada layar LCD.

3.3.2 Program Sensor IMU 9 Dof

Program sensor IMU 9 Dof ini digunakan untuk memonitoring pergeseran tanah. Keluaran dari sensor ini berupa sinyal *Analog to Digital Converter* (ADC). tiga ADC 16-bit untuk mendigitalkan keluaran *gyroscope* dan tiga ADC 16-bit untuk mendigitalkan keluaran akselerometer dan tiga ADC 16-bit untuk mendigitalkan keluaran *magnetometer*. Program pembacaan datanya menggunakan bantuan *library* bawaan dari *seed studio.*

```
Serial.println("Gyro(degress/s) of X,Y,Z:");
Serial.println(String () + "Derajat x:"+ Gxyz[0]); Blynk.virtualWrite(V6, Gxyz[0]);
Serial.println(String () + "Derajat y:"+ Gxyz[1]); Blynk.virtualWrite(V7, Gxyz[1]);
Serial.println(String () + "Derajat z:"+ Gxyz[2]); Blynk.virtualWrite(V8, Gxyz[2]);
Serial.println("     ");
lcd.setCursor(10,0);lcd.print("Gyro(s)");
lcd.setCursor(10,1);lcd.print(String () +"X:" + Gxyz[0] );
lcd.setCursor(10,2);lcd.print(String () +"Y:" + Gxyz[1] );
lcd.setCursor(10,3);lcd.print(String () +"Z:" + Gxyz[2] );
```

Gambar 3.7 *Source code* pembacaan *axis gyroscope*

Untuk pembacaan *axis gyroscope* masukan *source code* seperti pada gambar 3.7 pada aplikasi arduino IDE.

```
Serial.println("Acceleration(g) of X,Y,Z:");
Serial.print(Axyz[0]);
Serial.print(",");
Blynk.virtualWrite(V3, Axyz[0]);
Serial.print(Axyz[1]);
Serial.print(",");
Blynk.virtualWrite(V4, Axyz[1]);
Serial.println(Axyz[2]);
Serial.println("     ");
Blynk.virtualWrite(V5, Axyz[2]);
lcd.setCursor(0,0);lcd.print("Accel(g)");
lcd.setCursor(0,1);lcd.print(String () +"X:" + Axyz[0] );
lcd.setCursor(0,2);lcd.print(String () +"Y:" + Axyz[1] );
lcd.setCursor(0,3);lcd.print(String () +"Z:" + Axyz[2] );
```

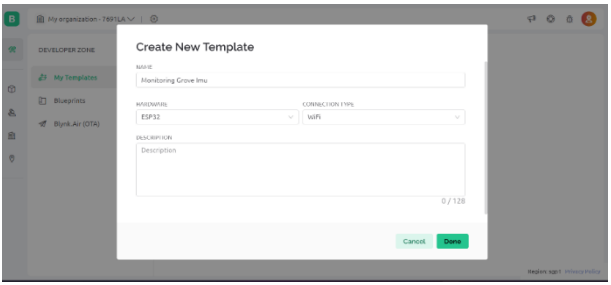
Gambar 3.8 *Source code* pembacaan *axis accelerometer*

Untuk pembacaan *axis accelerometer* masukan *source code* seperti pada gambar 3.8 pada aplikasi arduino IDE.

3.3.3 Menghubungkan ESP 32 Ke Blynk IoT

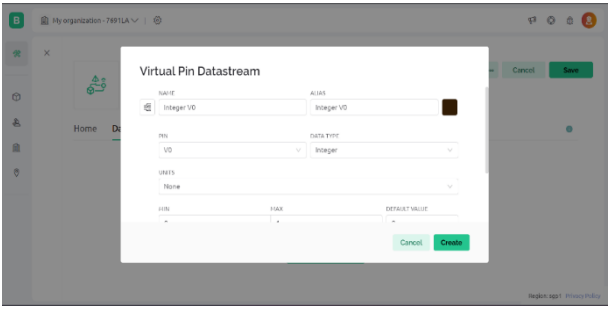
Cara menghubungkan ESP 32 dengan *platform* Blynk IoT memerlukan program untuk pengambilan data pada tiap sensor secara *online*. Program tersebut dihubungkan dengan jaringan internet dan nantinnya akan digunakan untuk menghubungkan ke

s*erver* Blynk IoT. kemudian dilanjutkan meunuju *platform* Blynk IoT. Pada tampilan utama *platform* Blynk IoT terdapat beberapa tahap untuk mengatur ke *cloud internet,* dan ada beberapa fitur atau menu yang dapat digunakan untuk menghubungkan dan mengkonfigurasi perangkat, menambahkan variabel dan membuat *data streams* di ESP 32. Pada tahap pertama adalah membuat *channel* pada laman Blynk IoT yang nantinya akan digunakan untuk menambahkan data data atau variabel.



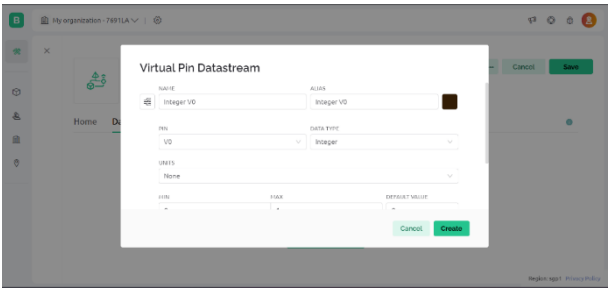
Gambar 3.9 Pembuatan *template* baru di Blynk IoT

Tahap pertama seperti pada gambar 3.9 tahap ini yaitu membuat *channel* pada laman aplikasi Blynk IoT yang nantinya akan digunakan untuk menambahkan data data atau variabel.



Gambar 3. 10 Pembuatan *virtual data streams*

Pada tahap kedua seperti pada gambar 3.10 tahap ini membuat virtual data streams. Dimana pada setiap sensor akan ditambahkan variabel untuk pengambilan data pada program yang sebelumnya sudah dibuat dan terhubung melalui jaringan internet agar bisa saling berkomunikasiantara NodeMCU ESP32 dengan Blynk IoT.



Gambar 3. 11 Membuat *virtual data streams* di Blynk IoT

Tahap ketiga seperti pada gambar 3.10 tahap ini yaitu pembuatan *dashboard* untuk *monitoring* data pergeseran tanah. Pada *dashboard* ini data - data bisa divisualisasikan dalam bentuk angka ataupun grafik dan nantinya akan menghasilkan data pengeluaran yang akan di*monitoring* melalui *platform* Blynk IoT.