IMPLEMENTASI *IoT* DENGAN PENDEKATAN PEMROGRAMAN BERBASIS *BLOCK* MENGGUNAKAN MIT APP INVENTOR

Okky Prasetia

Program Studi Teknik Informatika
Fakultas Ilmu Komputer, Universitas Pamulang, Indonesia
Jl. Raya Puspitek No.11, Buaran, Serpong, Kota Tangerang Selatan, Banten 15310

E-mail: dosen02837@unpam.ac.id

ABSTRAK

IMPLEMENTASI IOT DENGAN PENDEKATAN PEMROGRAMAN BERBASIS *BLOCK* MENGGUNAKAN MIT APP INVENTOR Internet of Things (IoT) mengintegrasikan perangkat fisik dan menciptakan peluang bagi manusia untuk berinteraksi dengan lingkungan sekitar . Sementara sejumlah pendekatan berbasis *block* yang sudah ada untuk antar muka beberapa program perangkat keras seperti Ide Arduino, Microblock, dan Scratch, antar muka nya ini masih merupakan ranah yang belum dijelajahi. Dalam penelitian ini, penulis menggunakan pendekatan pemrograman berbasis *block* menggunakan MIT App Inventor untuk dapat dikembangkan dalam membangun aplikasi seluler yang terintegrasi dengan teknologi IoT. MIT App Inventor sangat ideal untuk mengembangkan aplikasi IoT dengan antarmuka pemrograman visual berbasis blok. Dengan App Inventor, pengguna cukup memilih, menarik dan melepas blok fungsional ke platform berbasis browser web untuk membuat aplikasi seluler Android Penulis juga menggunakan bahasa C++ dan Database realtime Fire Base yang diterapkan untuk IoT. Metode yang digunakan peneliti adalah metode research and development , dengan tahapan pengumpulan data, perencananaan, perancangan, implementasi dan pengujian . Tujuan Hasil penelitian dengan program berbasis *block* bisa menginspirasi masyarakat untuk berkreasi dengan IoT .

Kata Kunci : *Internet of Things*, Riset dan pengembangan , pemrograman berbasis blok, MIT App Inventor

ABSTRACT

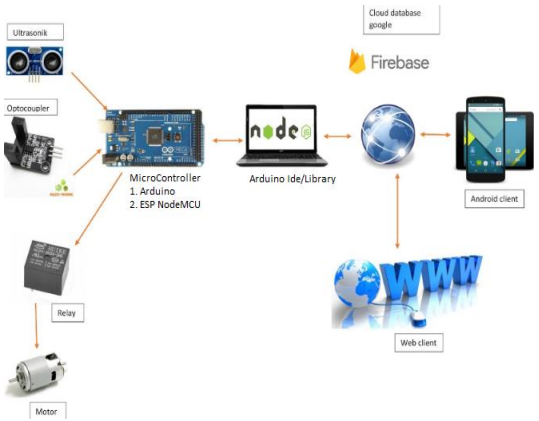
IMPLEMENTATION OF IOT WITH A PROGRAMMING APPROACH BLOCK BASED USING MIT APP INVENTOR. *Internet of Things (IoT) integrates physical devices and creates opportunities for humans to interact with the surrounding environment. While a number of block-based approaches already exist to interface with hardware programs such as Arduino Idea, Microblock, and Scratch, these interfaces are still an unexplored realm. In this research, the author uses a block-based programming approach using MIT App Inventor to develop mobile applications that are integrated with IoT technology. MIT App Inventor is ideal for developing IoT applications with a block-based visual programming interface. With App Inventor, users simply select, drag and drop functional blocks into a web browser-based platform to create Android mobile applications. The author also uses the C++ language and Fire Base realtime database applied for IoT. The research method used is the research and development method, with stages of data collection, planning, design, implementation and testing. Objective: The results of research using block-based programs can inspire people to be creative with IoT.*

Keywords: Internet of Things, Research and development, block-based programming, MIT App Inventor

1. PENDAHULUAN

MIT App Inventor adalah antar muka bahasa pemrograman berbasis blok yang intuitif dimana programmer pemula dapat membangun aplikasi fungsional tanpa *coding* pada ponsel pintar dan tablet. MIT App Inventor adalah sebuah tools pemrograman berbasis blocks yang memungkinkan para pemula untuk memulai pemrograman dan membangun aplikasi untuk perangkat mobile Android. Block disini adalah kumpulan atau code block berbentuk graphic seperti puzzle, dimana didalamnya terdapat komponen komponen *Logic, Control, Math, Text, Lists, Colors, Variables*, dan *Procedures*. Untuk para Pendatang baru ini dapat mengembangkan dan menjalankan aplikasi pertamanya pada MIT App Inventor dalam waktu kurang dari satu jam, dan dapat memprogram aplikasi yang lebih kompleks dalam waktu yang jauh lebih singkat dibandingkan dengan aplikasi berbasis teks dengan bahasa pemrograman tradisional.

Berbagai aplikasi yang dapat dibangun menggunakan MIT App Inventor salah satunya adalah penerapan aplikasi berbasis Iot dengan Bahasa pendukung lainnya seperti IDE Arduino serta Database realtime menggunakan FireBase yang dapat berintegrasi untuk merancang dan membuat aplikasi serta berinteraksi dengan perangkat fisik.



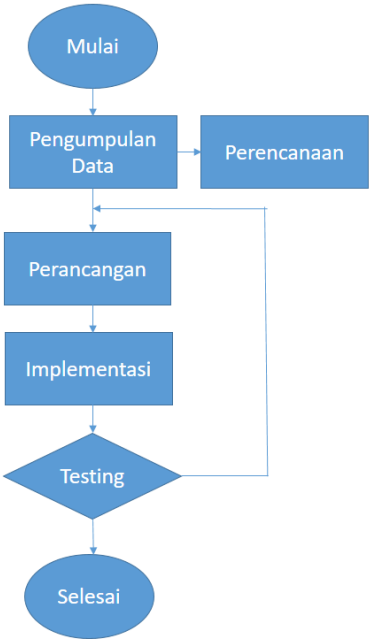
Gambar 1 Perangkat Dasar Iot dan Perangkat lunak pendukung.

Sejumlah pendekatan berbasis *block* yang sudah ada untuk antar muka beberapa program perangkat keras seperti Ide Arduino, Microblock, dan Scratch, antar muka nya ini masih merupakan ranah yang masih belum banyak dijelajahi. Untuk itu penulis melakukan penelitian penerapan Iot dengan pendekatan Antar muka berbasis blok dengan MIT App Inventor .

2. METODE

2.1 Metode Penelitian

Metode penelitian yang digunakan penulis menggunakan metode RND atau Penelitian dan pengembangan , menurut sugiyanto menjelaskan bahwa metode penelitian dan pengembangan atau Research & Development merupakan sebuah metode penelitian yang menghasilkan sebuah produk tertentu sekaligus menguji efektivitas produk yang dihasilkan. Hal ini juga sejalan dengan yang dijelaskan oleh Puslitjaknov bahwa model pengembangan ini merupakan sebuah dasar dalam mengembangkan produk yang akan dihasilkan. Produk yang dikembangkan dapat berupa produk baru maupun penyempurnaan dari produk yang telah ada. Dapat disimpulkan bahwa metode pengembangan atau research and development ini merupakan metode yang hasil akhirnya merupakan sebuah produk baru maupun hasil penyempurnaan dari produk yang telah ada.



Gambar 2. Tahapan metode penelitian

2.2 Gambaran Umum Sistem

Gambaran umum Penelitian ini adalah mengontrol Relay 4 *channel* menggunakan Microcontroller ESP8266 Wemos D1 mini dan dengan interface aplikasi Android yang dikembangkan dengan bantuan MIT App Inventor.

ESP8266 (ESP-01) adalah modul kecil yang memungkinkan mikrokontroler terhubung ke jaringan Wi-Fi dan membuat koneksi IP (Internet Protocol) sederhana. Dapat diprogram dengan menggunakan Arduino, NodeMCU IDE atau ESP8266 SDK. Beberapa modul lain seperti ESP-02, ESP-07 juga dirilis. Semua ini pada dasarnya adalah keluarga dari pada ESP8266, perbedaan adalah jumlah pin GPIO.

Dengan menginstal Aplikasi yang dibuat berbasis block pada Ponsel Android ini dapat mengontrol relay lampu menggunakan ESP8266 yang terhubung ke jaringan WiFi yang sama dengan ponsel Anda.

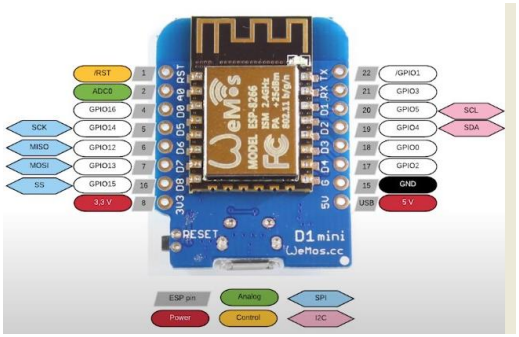
3. PERANCANGAN SISTEM

Tahapan Perancangan Sistem pada penelitian ini dibentuk menjadi dua bagian, yaitu perancangan perangkat keras serta perancangan perangkat lunak.

3.1. Perancangan Perangkat Keras

Perancangan perangkat yang dibutuhkan adalah perangkat Iot terdiri dari

1. NodeMCU ESP2866 seri Wemos D1 mini



Gambar 3 Node MCU ESP8266

2. Relay 4 channel

Gambar 4 Relay 4 Channel

3. Lampu

Gambar 4 Lampu Bolamp

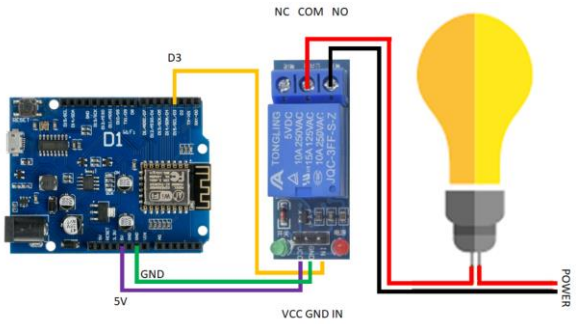
4. Kabel Jumper

Gambar 5 Kabel Jumper

5. Project Board

Gambar 6 Project Board

6. Wiring Perancangan



Gambar 7 Wiring Perancangan

3.2 Perancangan Perangkat Lunak

Perancangan perangkat lunak dibutuhkan 3 perangkat lunak yaitu :

1. Arduino IDE
2. FireBase
3. MIT App Inventor

1. Arduino IDE

Arduino IDE (Integrated Development Environment) digunakan untuk menulis kode program dan mengunggah kode ini ke perangkat fisik IoT. Arduino IDE sangat sederhana dan kesederhanaan ini mungkin merupakan salah satu alasan utama Arduino menjadi begitu populer. Kompatibilitas dengan Arduino IDE kini menjadi salah satu persyaratan utama untuk board MCU baru. Selama bertahun-tahun, banyak fitur berguna telah ditambahkan ke Arduino IDE dan kini Anda dapat mengelola library pihak ketiga dari IDE, dan tetap menjaga kesederhanaan pemrograman .

Berikut ini code untuk relay 4 channel dan NodeMCU ESP2866 Wemos D1 Mini :

```
#include <ESP8266WiFi.h>
#include <Firebase_ESP_Client.h>

//Provide the token generation process info.
#include "addons/TokenHelper.h"
//Provide the RTDB payload printing info and other helper functions.
#include "addons/RTDBHelper.h"

// Insert your network credentials
#define WIFI_SSID "Galaxy"
#define WIFI_PASSWORD "1234512345"

// Insert Firebase project API Key
#define API_KEY "AIzaSyB--cVmsxYGUH81BK-H3IFZ8w36vj8XDvQ"

// Insert RTDB URLefine the RTDB URL */
#define DATABASE_URL "https://relay4channel-be769-default-rtdb.firebaseio.com/"

//Define Firebase Data object
FirebaseData fbdo;

FirebaseAuth auth;
FirebaseConfig config;

//some importent variables
String sValue, sValue2;
bool signupOK = false;

 const int relay1=16; //D0
const int relay2=14; //D5
const int relay3=12; //D6
const int relay4=13; //D7


void setup() {
Serial.begin(115200);

 pinMode (relay1, OUTPUT);
 pinMode (relay2, OUTPUT);
 pinMode (relay3, OUTPUT);
 pinMode (relay4, OUTPUT);

 WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED) {
    Serial.print(".");
    delay(300);
  }
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();
```

```
  /* Assign the api key (required) */
  config.api_key = API_KEY;

  /* Assign the RTDB URL (required) */
  config.database_url = DATABASE_URL;

  /* Sign up */
  if (Firebase.signUp(&config, &auth, "", "")) {
    Serial.println("ok");
    signupOK = true;
  }
  else {
    Serial.printf("%s\n", config.signer.signupError.mes-
sage.c_str());
  }
  /* Assign the callback function for the long running to-
ken generation task */
  config.token_status_callback = tokenStatusCallback;
//see addons/TokenHelper.h

  Firebase.begin(&config, &auth);
  Firebase.reconnectWiFi(true);
    digitalWrite(relay1,HIGH);
    digitalWrite(relay2,HIGH);
    digitalWrite(relay3,HIGH);
    digitalWrite(relay4,HIGH);
}

void loop() {


  if (Firebase.ready() && signupOK ) {


    if (Firebase.RTDB.getString(&fbdo, "COLED/L1")) {
      if (fbdo.dataType() == "string") {
        sValue = fbdo.stringData();
        int a = sValue.toInt();
        Serial.println(a);
        if (a == 1){
          digitalWrite(relay1,HIGH);
        }else{
          digitalWrite(relay1,LOW);
        }
      }
    }
      else {
        Serial.println(fbdo.errorReason());
      }

if (Firebase.RTDB.getString(&fbdo, "COLED/L2")) {
      if (fbdo.dataType() == "string") {
        sValue2 = fbdo.stringData();
        int b = sValue2.toInt();
        Serial.println(b);
        if (b == 1){
          digitalWrite(relay2,HIGH);
        }else{
          digitalWrite(relay2,LOW);
        }
      }
    }
    else {
      Serial.println(fbdo.errorReason());
    }

if (Firebase.RTDB.getString(&fbdo, "COLED/L3")) {
      if (fbdo.dataType() == "string") {
        sValue2 = fbdo.stringData();
        int c = sValue2.toInt();
        Serial.println(c);
        if (c == 1){
          digitalWrite(relay3,HIGH);
        }else{
          digitalWrite(relay3,LOW);
        }
      }
    }
    else {
      Serial.println(fbdo.errorReason());
    }

if (Firebase.RTDB.getString(&fbdo, "COLED/L4")) {
      if (fbdo.dataType() == "string") {
        sValue2 = fbdo.stringData();
        int d = sValue2.toInt();
        Serial.println(d);
        if (d == 1){
          digitalWrite(relay4,HIGH);
        }else{
          digitalWrite(relay4,LOW);
        }
      }
```

```
  }
  else {
    Serial.println(fbdo.errorReason());
  }

 }

}
```

2. Perancangan Realtime Cloud DB Fire Base

Firebase menawarkan beberap layanan berbasis cloud mulai dari auten kasi hingga database, penyimpanan, da hosting. Untuk pengembangan aplika IoT real-time,

Dalam pengembangan IoT, ki dapat menghubungkan perangkat pint dengan Realtime Database dan memb atnya mengirimkan data secara berkala ke aplikasi. Mencapai hal seperti itu dengan tipe database lain agak rumit dan memerlukan pemrograman yang ek-stensif.

Layanan Hosting yang ditawarkan oleh Firebase dapat digunakan untuk menghosting aplikasi IoT real-time kita.

4. HASIL DAN PEMBAHASAN

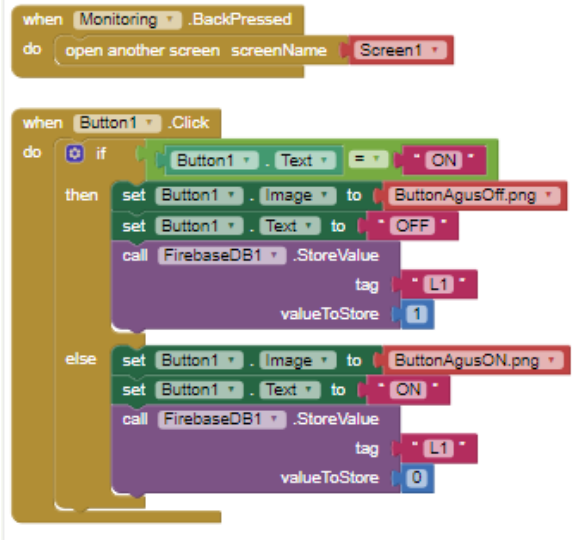
4.1 Desain Implementasi

Sebelum menggunakan block berikut ini adalah desain antar muka pada MIT App Inventor :

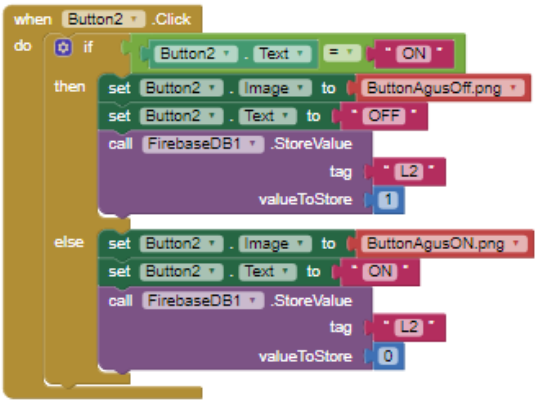
Gambar 8 Desain Interface

4.2 Pemrograman Block

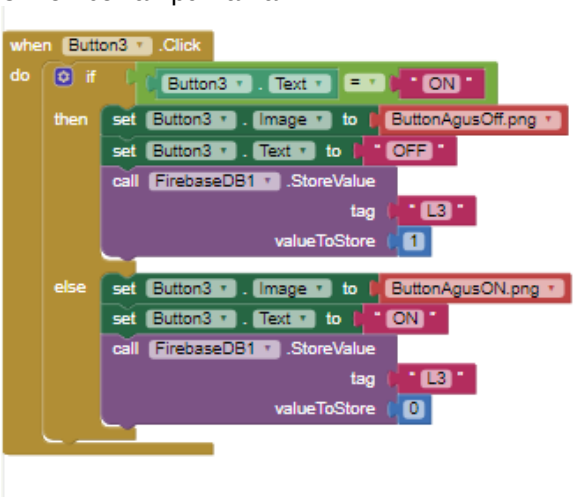
1. Tombol Lampu Teras



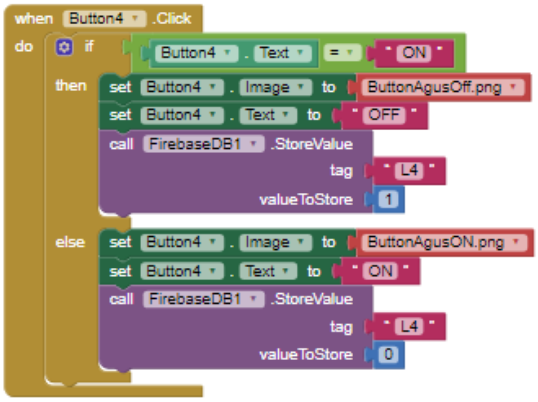
2. Tombol Lampu Ruangan Tamu



3. Tombol lampu Kamar



4. Tombol Lampu Kipas Angin



5. KESIMPULAN

Ada beberapa lingkungan pemrograman berbasis *block* yang dapat membangun aplikasi dan memiliki fitur perspektif Sebagai salah satu platform pemrograman visual yang intuitif , MIT App Inventor memiliki keunggulan.

Fitur fitur pada MIT App yang berfokus pada aplikasi seluler dapat diperluas lebih banyak fungsi terkait IoT , MIT App Inventor dengan kesederhanaan dan kemudahan pemrograman blocknya memudahkan untuk programmer pemula dan bahkan anak-anak

Drag dan drop untuk menggabungkan *block* adalah sangat simple dan mudah tanpa hambatan berarti, ini memungkinkan orang untuk menikmati pengalaman belajar dan bermain dengan IoT.

Blok memungkinkan seseorang membuat kode tanpa sintaks pengkodean yang sulit . Termasuk gambar atau teks , ini mengurangi beban kognitif, Sangat bermanfaat bagi pemula dan anak-anak.

DAFTAR PUSTAKA

[1]. Agus Suharto , Tutorial Mudah Membuat Aplikasi Android Dengan MIT APP INVENTOR (AI2). Penerbit Adab CV Adanu, 2022

[2] Wenxi , Evan W. Patton, *Block-Based Approaches to Internet of Things in MIT*, MIT CSAIL, Massachusetts Institute of Technology 2018 Cambridge, MA, USA

[2]. Sugiyono. , Metode Penelitian Kuantitatif, Kualitatif, dan R&D. Bandung: CV Alfabeta, 2016

[3]. Arduino Integrated Development Environment (IDE) v1, dapat diakses pada link https://docs.arduino.cc/software/ide-v1/tutorials/arduino-ide-v1-basics 2021

[4].Bagaimana Mengembangkan Aplikasi IoT Real-Time Menggunakan Google Firebase? , https://www.ashutec.com/blog/how-to-develop-real-time-iot-apps-using-google-firebase-8318ad001a31,