Practical and Innovative Applications of IoT & IoT Networks
Smart Home

TITLE:

To Design and implement IoT system for the applications like: Home Automatiom

THEORY:

In the present era, there is a strong emphasis on enhancing comfort, and one effective solution is the integration of IoT (Internet of Things) devices. These devices enable us to streamline our lives by offering remote control over various aspects. For instance, household appliances, door locks, and machinery can be conveniently managed through smartphones or web servers. A practical approach to achieving this is by creating an Android application using MIT App Inventor.

The process involves programming a NodeMCU, which serves as the central hub, to establish an HTTP web server. This server enables the control of home appliances. The communication between the NodeMCU and the Android app primarily employs the HTTP GET method.

To determine the functionality of the web server, a straightforward method is to open a web browser and utilize specific URLs. These URLs are configured to trigger actions such as turning lights on or off. This provides a seamless way to interact with and monitor the connected devices.

http://172.128.10.40/gpio/1

http://172.128.10.40:/gpio/0

The IP address of the NodeMCU is 172.128.10.40. You can easily locate your NodeMCU's IP address by checking the serial monitor. Upon running the code in the Arduino IDE, the device's IP address will be displayed in the serial monitor. This process ensures the verification of the web server's functionality.

MIT App Inventor stands as an open-source online platform tailored for Android applications. Although initially developed by Google, it is now overseen by the Massachusetts Institute of Technology (MIT). This tool serves as an accessible option, even for novices, to craft software applications for the Android platform. Its user-friendly approach hinges on a graphical interface, enabling users to effortlessly construct applications through a drag-and-drop method, all of which are compatible with Android devices.

Subsequent to the design phase within MIT App Inventor, the application can be acquired on an Android phone through a QR code, or alternatively, its APK can be downloaded onto a computer for later installation on a smartphone. Once this is accomplished, the next step involves establishing a connection between the application and the ESP8266 module, facilitating the control of various home appliances.

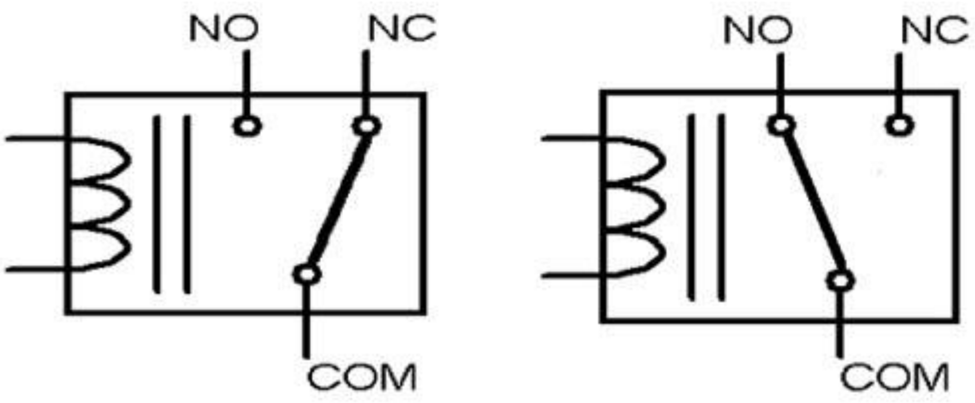
How Relay works?

A relay serves as an electrical switch that is activated through an electrical signal. It proves invaluable in scenarios where multiple circuits need to be managed using a single input. By utilizing a relay, it becomes feasible to control the operation of an electrical circuit, toggling it on or off. The functioning of a relay hinges on a low-level current that is capable of controlling the switching of a higher magnitude current.

Typically featuring five terminals, a relay's configuration is as follows:

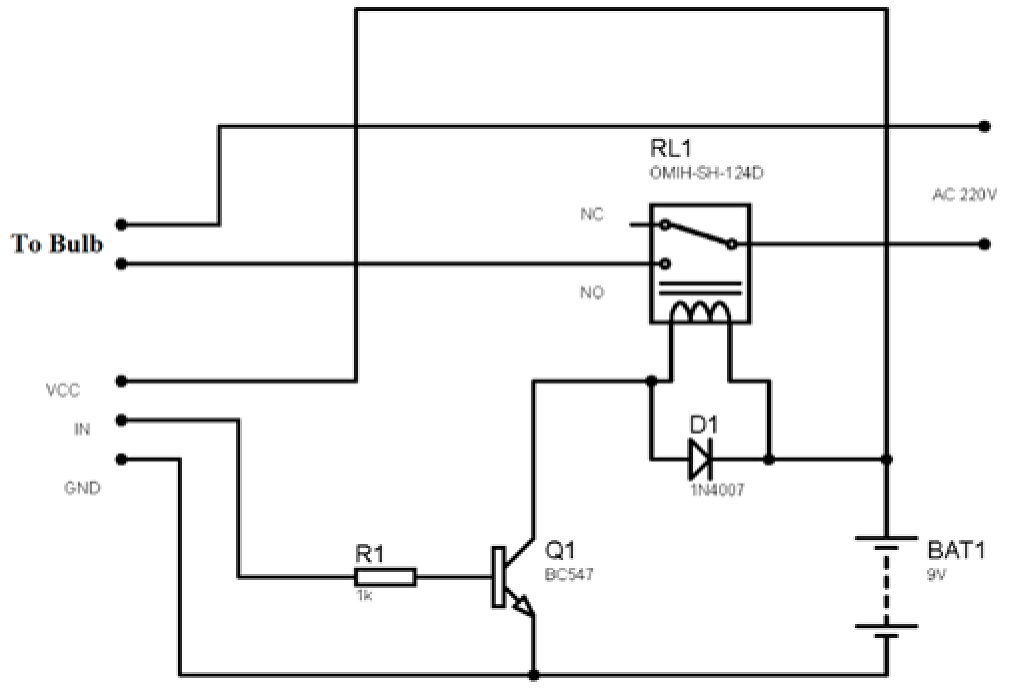
In the absence of voltage applied to the coil, the COM (common) terminal establishes a connection with the NC (normally closed) terminal.

Upon the application of voltage to the coil, an electromagnetic field is generated, attracting the armature. This action causes the COM terminal to connect with the NO (normally open) terminal.



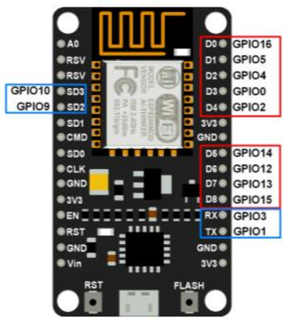
Relay Driver:

A compact driver circuit employs a Transistor, Diode, and Resistor to configure the relay. The Transistor serves to enhance the current, the Resistor provides the necessary bias to the transistor, and in instances when the transistor is in the off state, the Diode is employed to avert any reverse current flow. It's important to note that a 5V Relay module is utilized in this setup.



COMPONENTS REQUIRED:

- ESP8266 (NodeMCU)
- Lamp
- 5V Relay
- Breadboard
- Connecting Wires

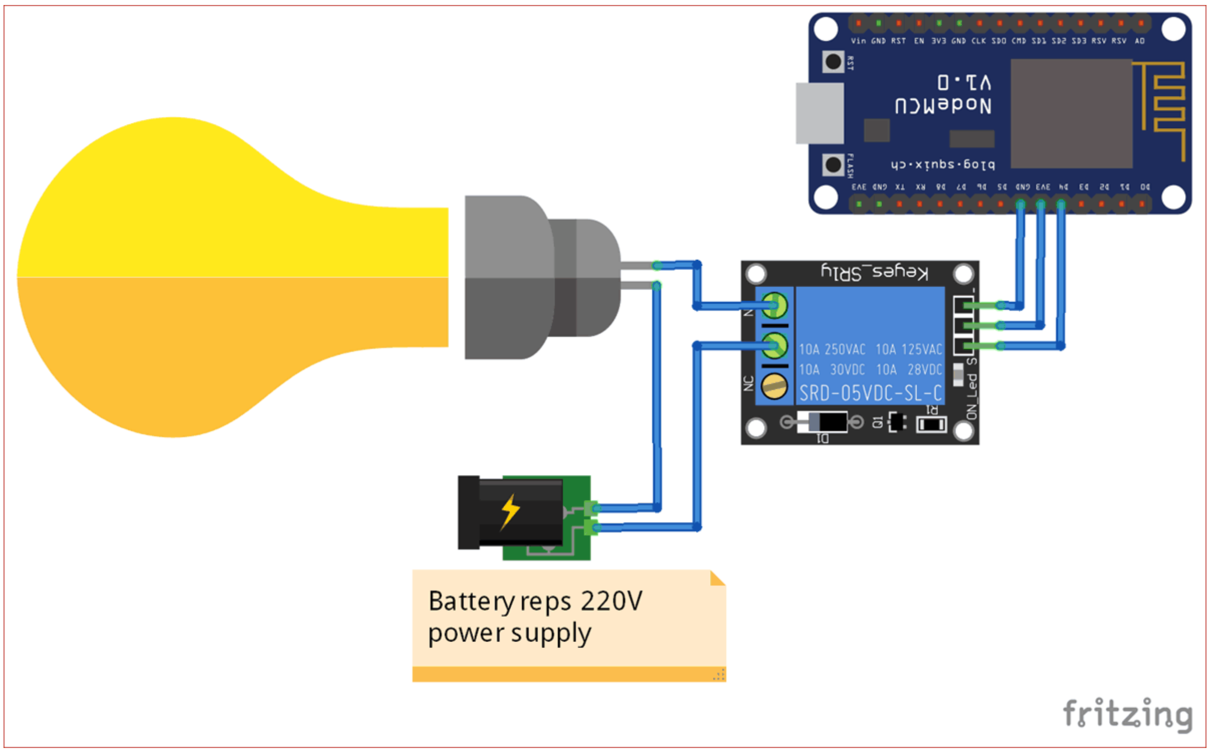


Circuit Diagram

In this experiment, we are going to log and monitor data over internet using MIT app Inventer IoT server. And we can view the logged data over time on Mobile app. System is made using ESP8266 WiFi module and Relay module. ESP8266 WiFi chip reads the current date and sends it to MIT app inventer server for live monitoring.

In this experiment, relay module can be used with esp8266 or Nodemcu and program from Arduino IDE, the mobile data has used to send data to MIT app and it was really productive, results are quite good.

Connection:



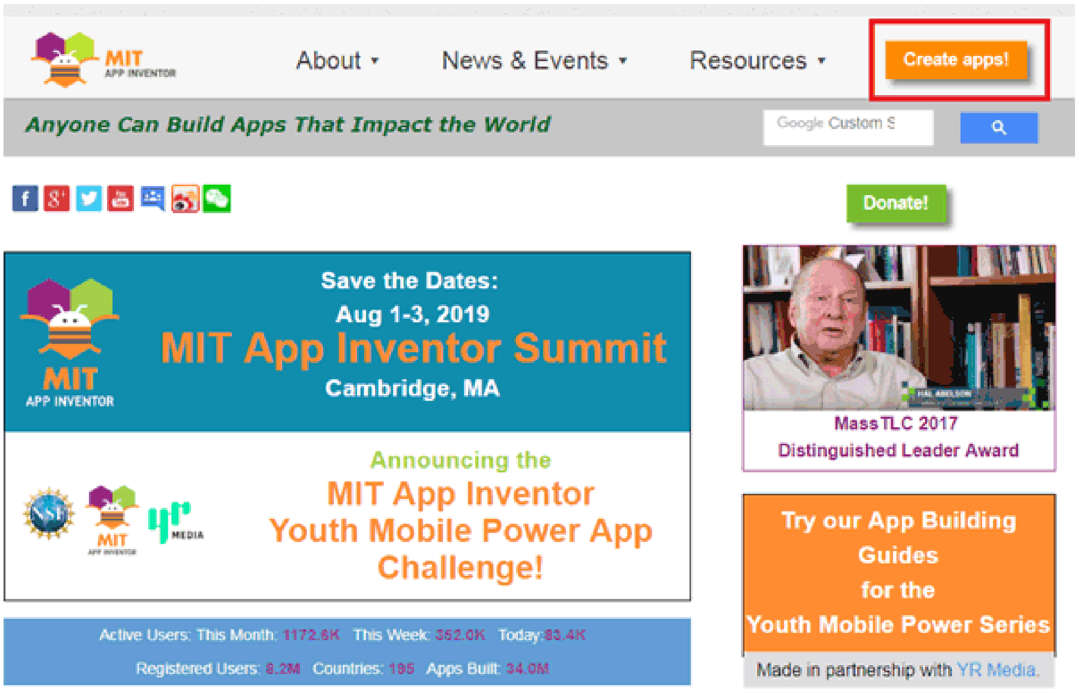
Create an Android APP using MIT App Inventor:

To initiate the process of crafting an Android app for light control, we'll utilize MIT App Inventor through the subsequent steps:

Begin by accessing the MIT App Inventor's official website:

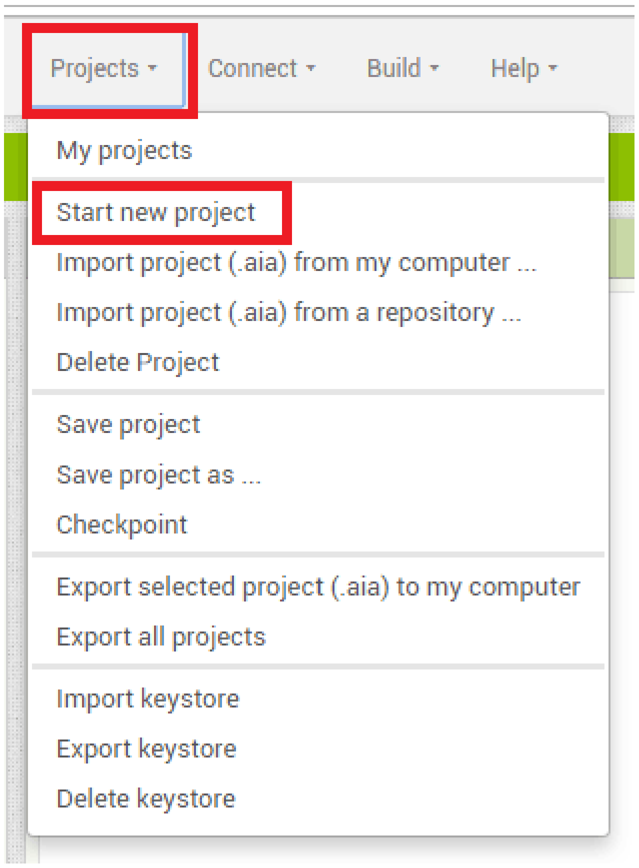
http://ai2.appinventor.mit.edu/

Proceed by selecting the 'Create apps' option situated at the upper right-hand corner of the interface.


MIT APP INVENTOR
About
News & Events
Resources
Create apps!
Anyone Can Build Apps That Impact the World
Google Custom S
Donate!
Save the Dates:
Aug 1-3, 2019
MIT App Inventor Summit
Cambridge, MA
Announcing the
MIT App Inventor
Youth Mobile Power App
Challenge!
Active Users: This Month: 1172.6K This Week: 352.0K Today: 83.4K
Registered Users: 8.2M Countries: 195 Apps Built: 34.0M
Mass TLC 2017
Distinguished Leader Award
Try our App Building Guides for the Youth Mobile Power Series
Made in partnership with YR Media.

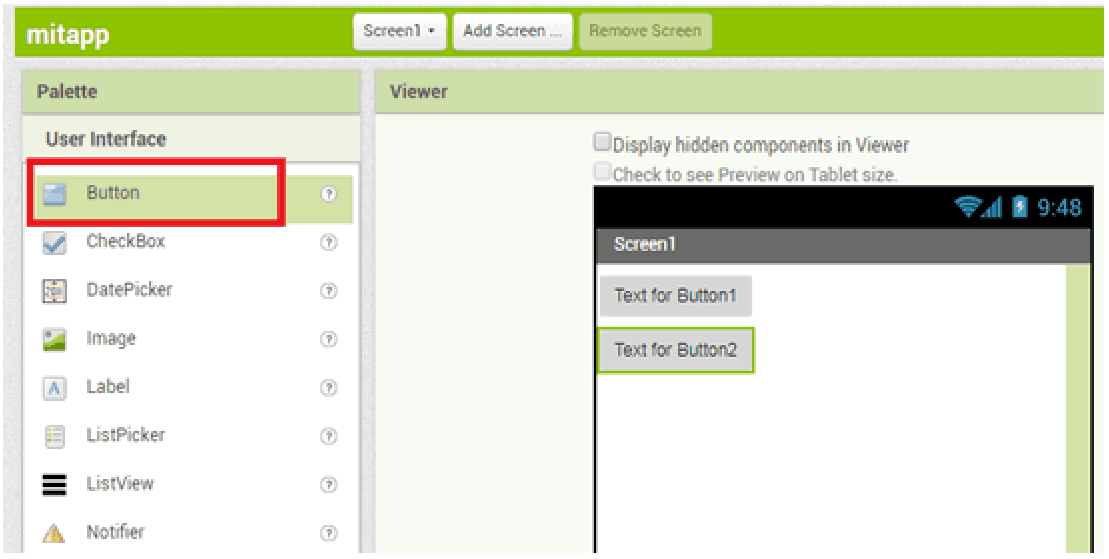
Create an Android APP using MIT App Inventor:

Moving forward, on the subsequent screen, navigate to 'Projects' and subsequently choose 'Start new project'.



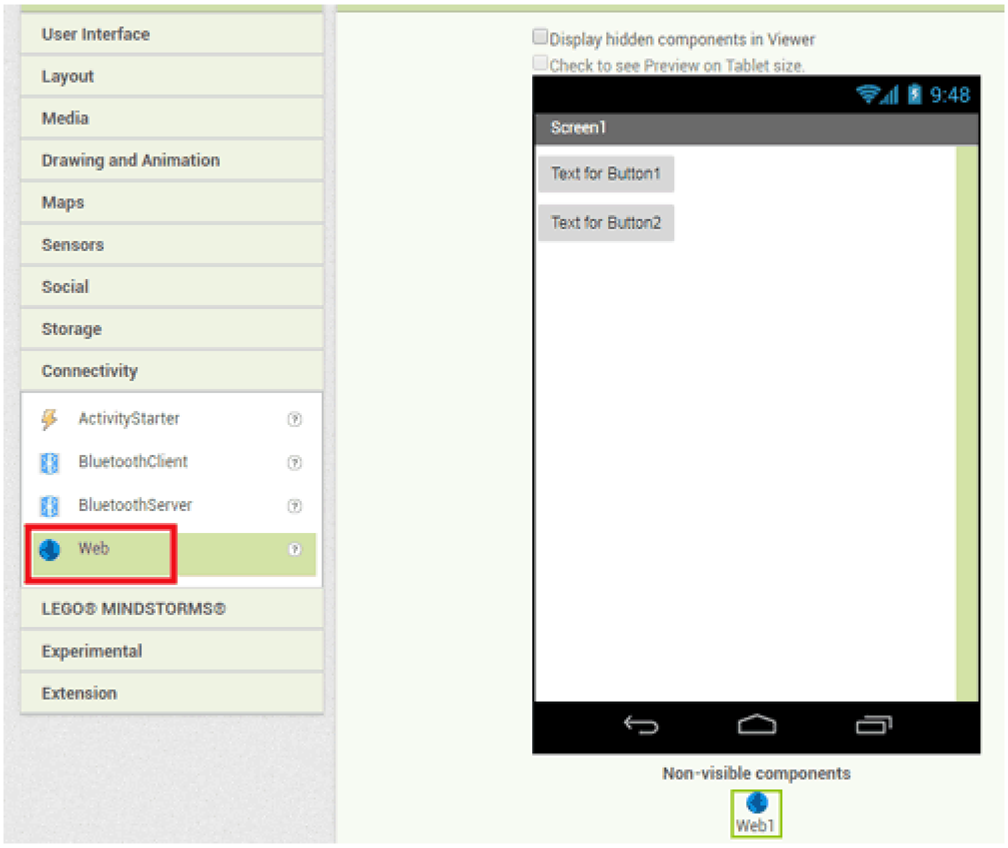
Create an Android APP using MIT App Inventor

Next, select the 'Button' element and proceed to drag and drop two buttons onto the primary screen. To personalize the buttons, you can input your preferred names from the available options located on the right-hand side.



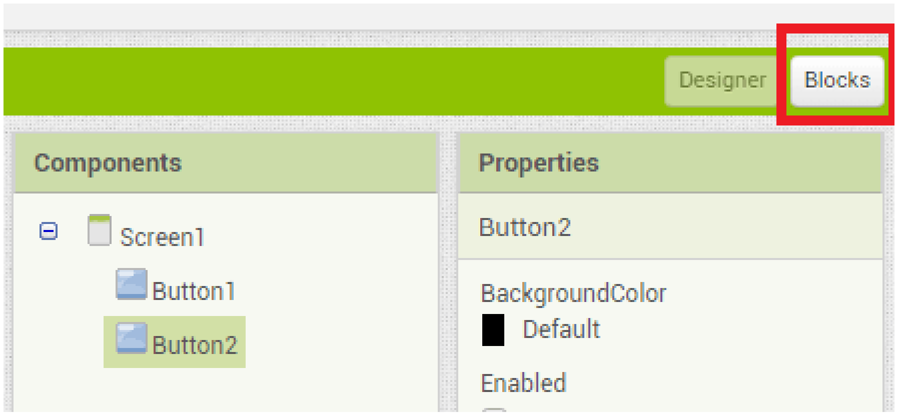
Create an Android APP using MIT App Inventor

Following this step, access the 'connectivity' section and proceed to drag and drop the web component onto the main screen.



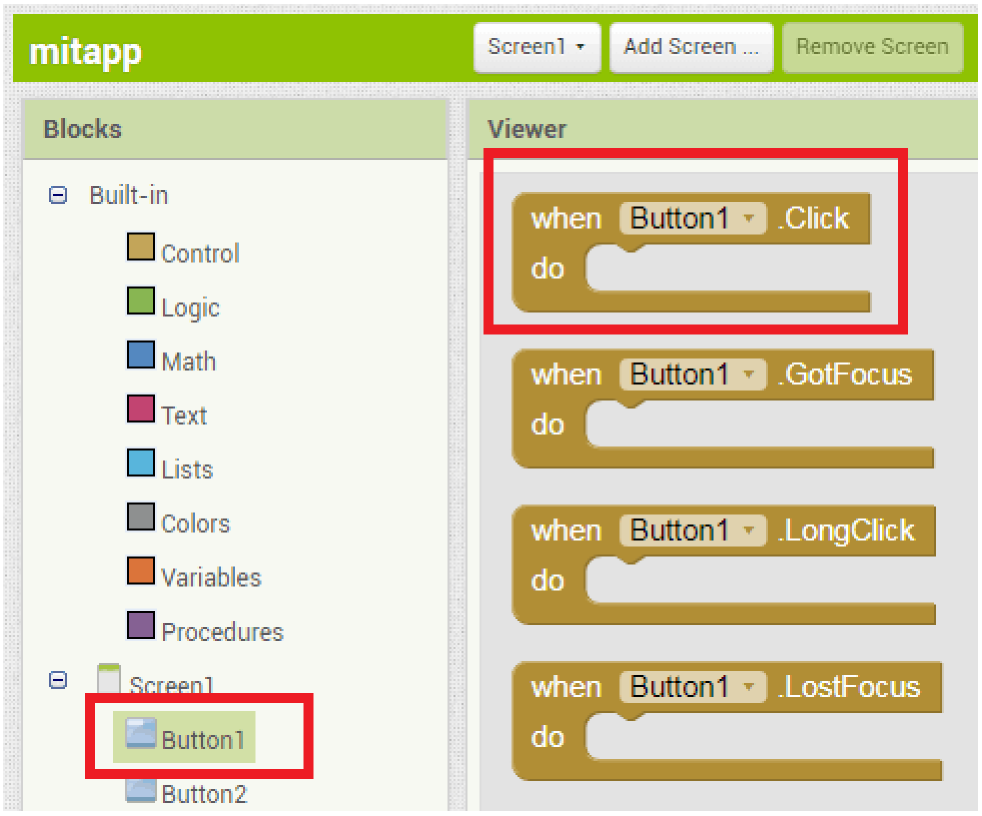
Create an Android APP using MIT App Inventor

Now, access the 'Blocks' section to incorporate blocks into your application.



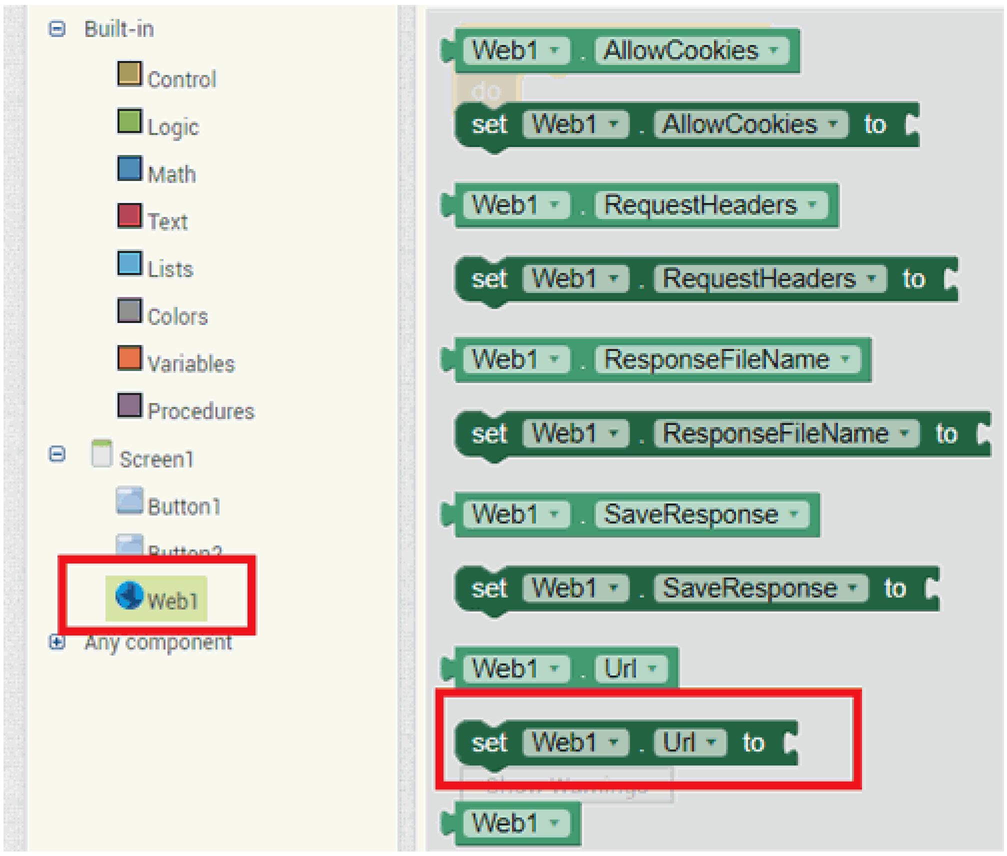
Create an Android APP using MIT App Inventor

Within the blocks menu, select 'button1,' and then choose the highlighted red option.



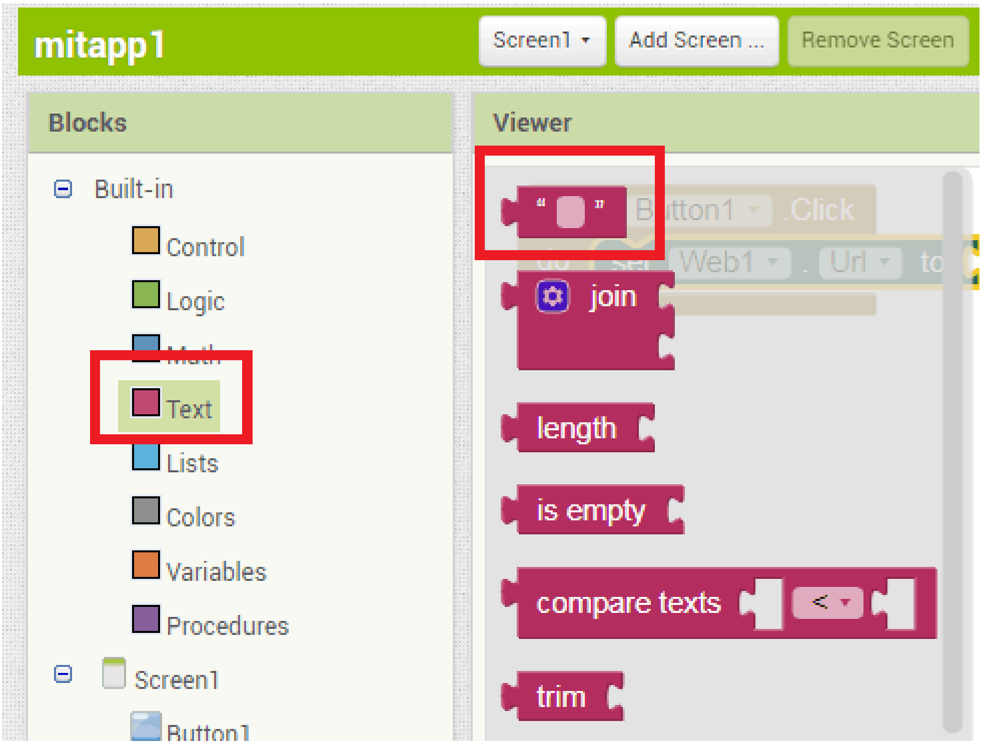
Create an Android APP using MIT App Inventor

Subsequently, click on 'web1.' Scroll down and opt for the block indicated with a red mark.



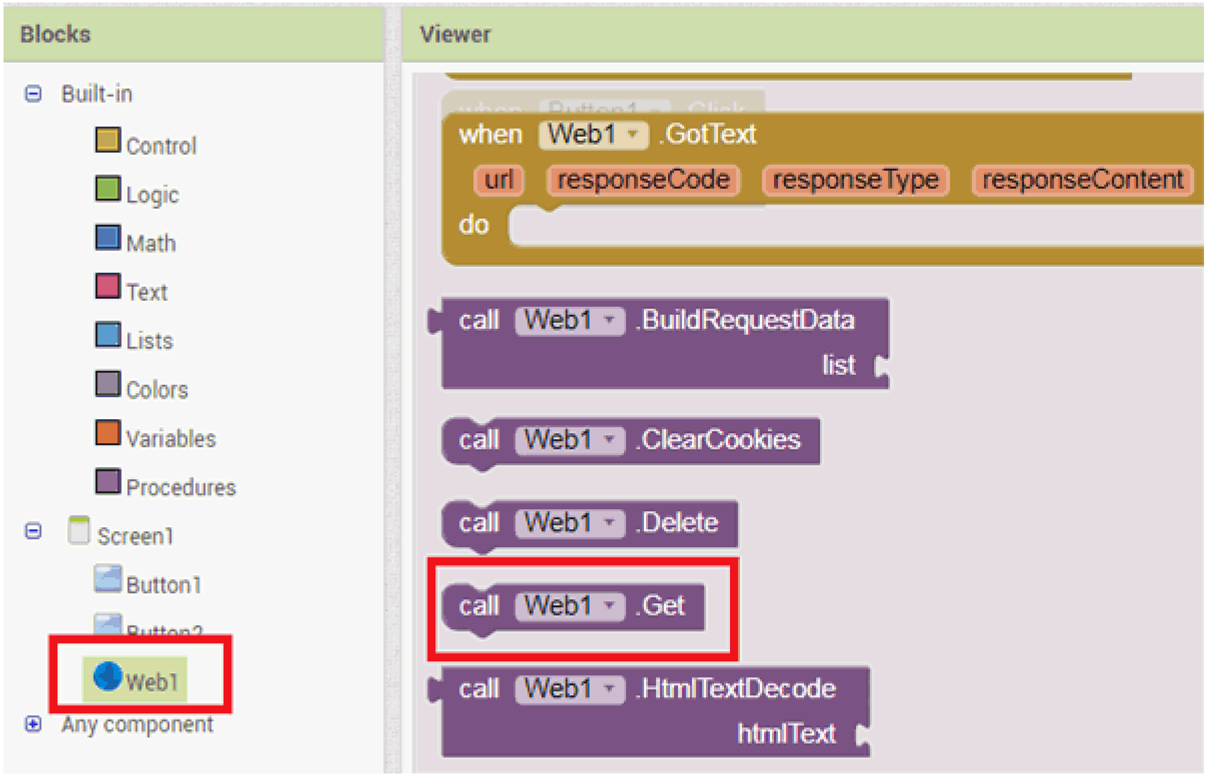
Create an Android APP using MIT App Inventor

Now, access the 'Text' menu and select the initial option. Enter your desired URL within the text menu.



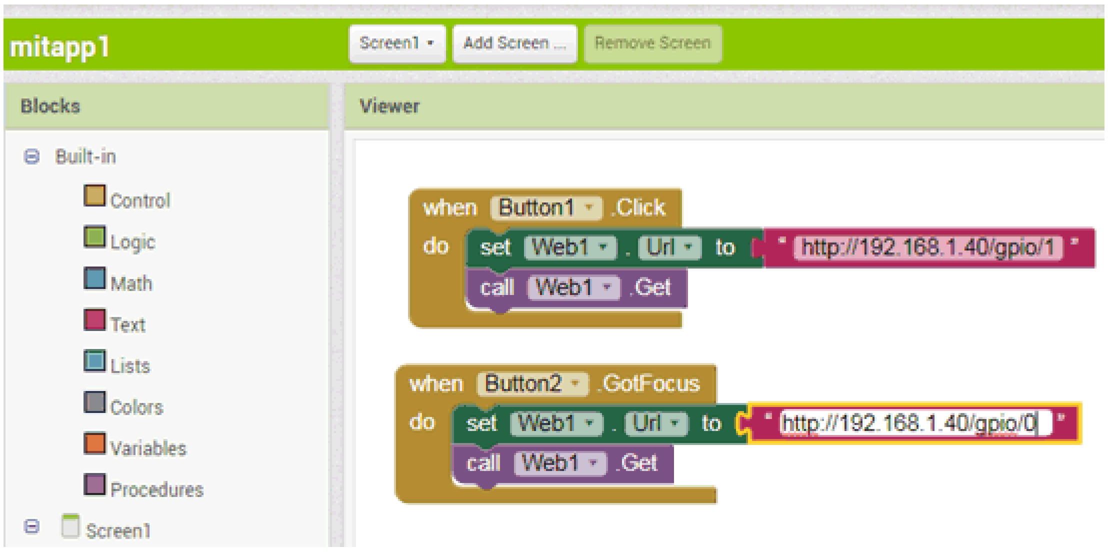
Create an Android APP using MIT App Inventor

Next, click on 'web1' once more and proceed to select the option highlighted in red.



Create an Android APP using MIT App Inventor

Apply the identical procedure for 'Button2.'

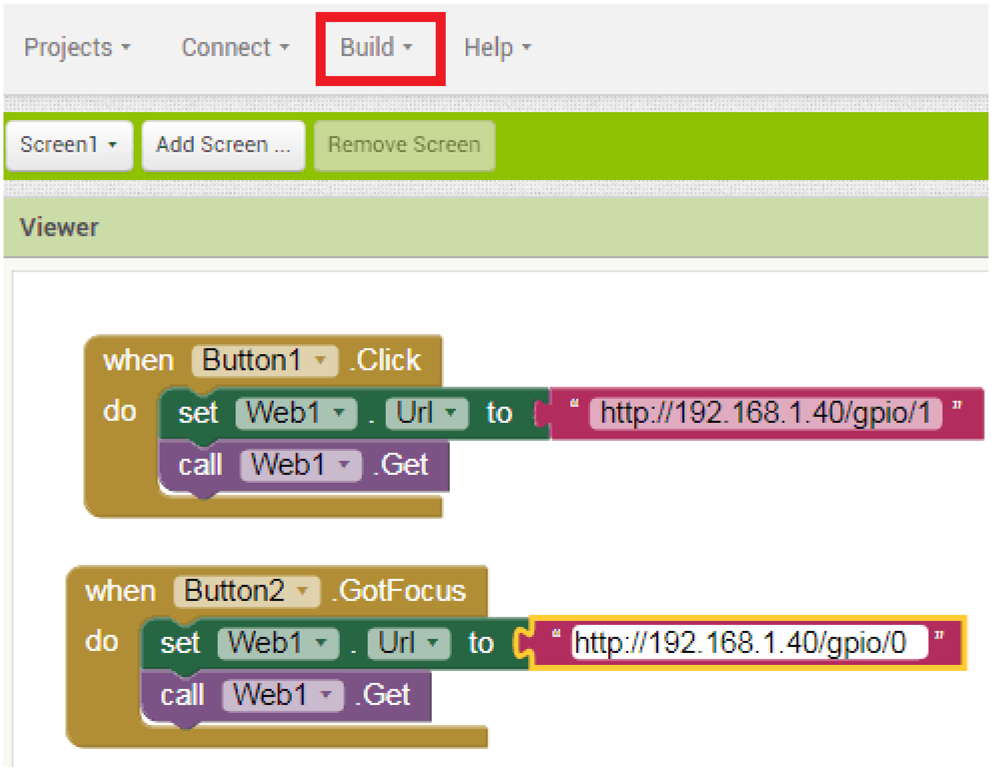


```
when Button1 .Click
do  set Web1 . Url to "http://192.168.1.40/gpio/1"
    call Web1 .Get

when Button2 .GotFocus
do  set Web1 . Url to "http://192.168.1.40/gpio/0"
    call Web1 .Get
```

Create an Android APP using MIT App Inventor

The app is now prepared for download. Just click on the 'Build' option to obtain the APK file. There are two methods available to acquire the app APK: through the QR code or directly on your PC. Subsequently, you can install the downloaded APK on your Android device.

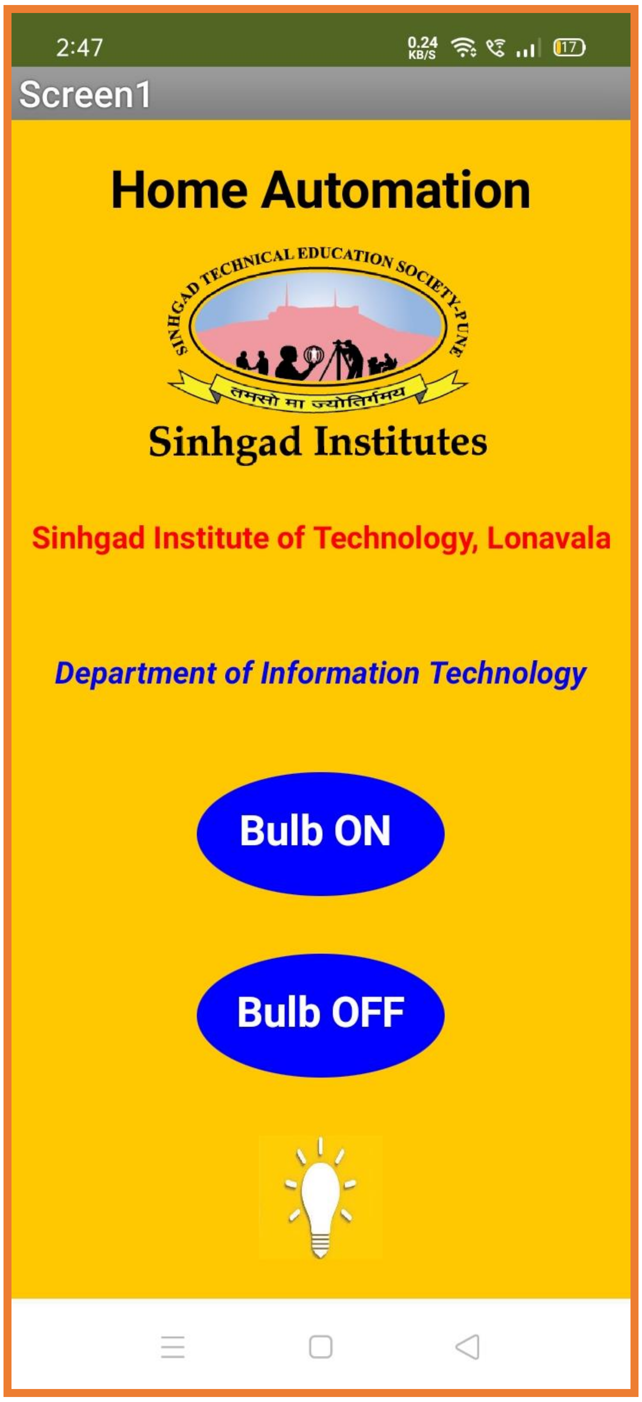


```
when Button1 .Click
do  set Web1 . Url to "http://192.168.1.40/gpio/1"
    call Web1 .Get

when Button2 .GotFocus
do  set Web1 . Url to "http://192.168.1.40/gpio/0"
    call Web1 .Get
```

Create an Android APP using MIT App Inventor

Your application is now fully prepared, granting you the ability to control the lighting using the on-off buttons featured within the app.


2:47
Screen1
Home Automation
SINHGAD TECHNICAL EDUCATION SOCIETY-PUNE
तमसो मा ज्योतिर्गमय
Sinhgad Institutes
Sinhgad Institute of Technology, Lonavala
Department of Information Technology
Bulb ON
Bulb OFF

Program:

```
#include <ESP8266WiFi.h>
 const char* ssid = "OPPO_pcl";
const char* password = "pclatane123";
WiFiServer server(80);
 void setup() {
  Serial.begin(115200); //Default Baud Rate for NodeMCU
  delay(10);
  pinMode(2, OUTPUT);  // Connect Relay to NodeMCU's D4 Pin
  digitalWrite(2, 0);
  // Connect to WiFi network
  Serial.println();
  Serial.println();
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
  delay(500);
  Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");
  // Start the server
  server.begin();
  Serial.println("Server started");
  // Print the IP address
  Serial.println(WiFi.localIP());
}
 void loop() {
  // Check if a client has connected
  WiFiClient client = server.available();
  if (!client) {
    return;
  }
  // Wait until the client sends some data
  Serial.println("new client");
  while(!client.available()){
  delay(1);
  }
  String req = client.readStringUntil('\r');
  Serial.println(req);
  client.flush();
int val;
  if (req.indexOf("/gpio/0") != -1)
    val = 0;
  else if (req.indexOf("/gpio/1") != -1)
    val = 1;
  else {
    Serial.println("invalid request");
    client.stop();
    return;
  }
  String s = "HTTP/1.1 200 OK\r\nContent-Type:
 text/html\r\n\r\n<!DOCTYPE HTML>\r\n<html>\r\nGPIO is now ";
  s += (val)?"high":"low";
  s += "</html>\n";
// Send the response to the client
client.print(s);
delay(1);
Serial.println("Client disonnected");
}
```

CONCLUSION:

After the study of this assignment, we are familiar with the home automation and use of MIT app Inventer to upload data on cloud.