2. The back end, which involves a microcontroller to monitor devices used in the residence for the purpose of home automation by using appropriate connectivity protocols.

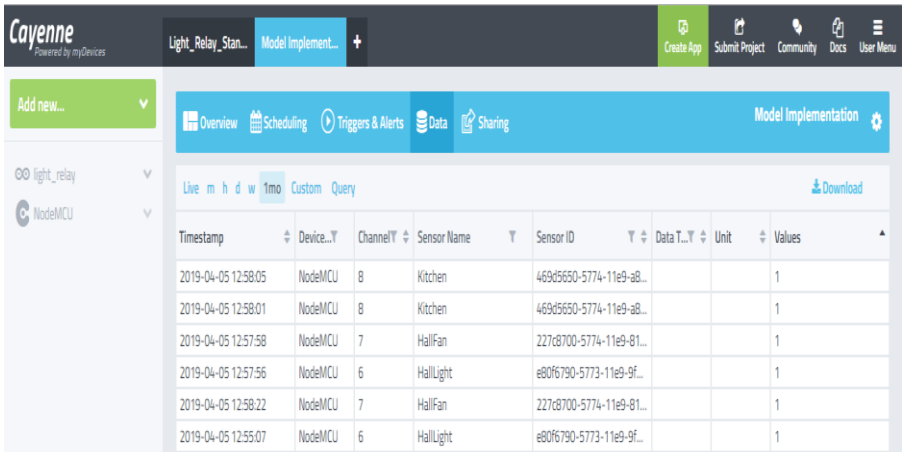


*Figure 6.3: Back end*

A system to be designed must prove flexible and compatible to the work environment and operate accordingly. The modules implemented must provide security by notifying and performing actions in the field of home automation for any chance of malicious activity.

The system provides cost effective features controlled by a centralized system. Here, smart phone acts as the centralized system which will receive updates and status during the working of the automated system. The master device, NodeMCU is used to control:

- Gate/Door
- Light, Fan, AC Thermostat
- Water Level
- Geyser
- Garden Irrigation.

For the Gate Automation, RFID technology is being used. So, the main gate will open and close based on RFID installed on vehicles of the user, but the priority is given to the remote control unit (i.e. mobile app). When the RFID reader detects a radio wave from the RFID tag and if the tag is accepted, only then will the gate open. The Door Automation makes use of relays and mobile app. The relay is connected to the microcontroller. Whenever the user uses the mobile app to unlock or lock the door, the

relay switch functions and the door gets unlocked or locked respectively. The main power supply is connected to the main door. As soon as the main door gets opened or if there is a person in the house, the power is supplied else, the power is not supplied to the house. This power supply can also be controlled by the mobile application.

The lights, fans and AC thermostat are controlled by the dashboard. In this system, depending on the presence and absence of people in a particular room, the devices are switched ON and OFF respectively. With respect to the AC thermostat, the temperature of the surrounding is sensed and the room conditions can be set accordingly.

Microcontroller based automatic water controlling system comes to use, when a minimum amount of water level is reached, the user gets a notification and the water supply to the garden gets minimized. When the sump is getting filled up, the water level is noted and the flow of water is monitored in such a way that the sediments do not get disturbed. In order to adjust the temperature of the water, geysers are also controlled by the user by specifying criteria on the dashboard, thereby saving energy and time.

Garden Irrigation is linked to water controlling system wherein, according to the weather conditions, the level of water in the tank, and moisture of the soil, the water supply to the garden is monitored.

## 6.2 Flow Diagram of the Proposed System

Flow diagram is a collective term for a diagram representing a flow or set of dynamic relationships in a system. The term flow diagram is also used as a synonym for flowchart, and sometimes as a counterpart of the flowchart. Flow diagrams are used to structure and order a complex system, or to reveal the underlying structure of the elements and their interaction.

According to the flow diagram, the functioning of the house is divided into five modules which will be dealt with separately but with respect to the functioning of the house, every module's status matters simultaneously. This is possible when we combine or integrate the logic of all the modules into one program and simultaneously checking the status of the required modules.

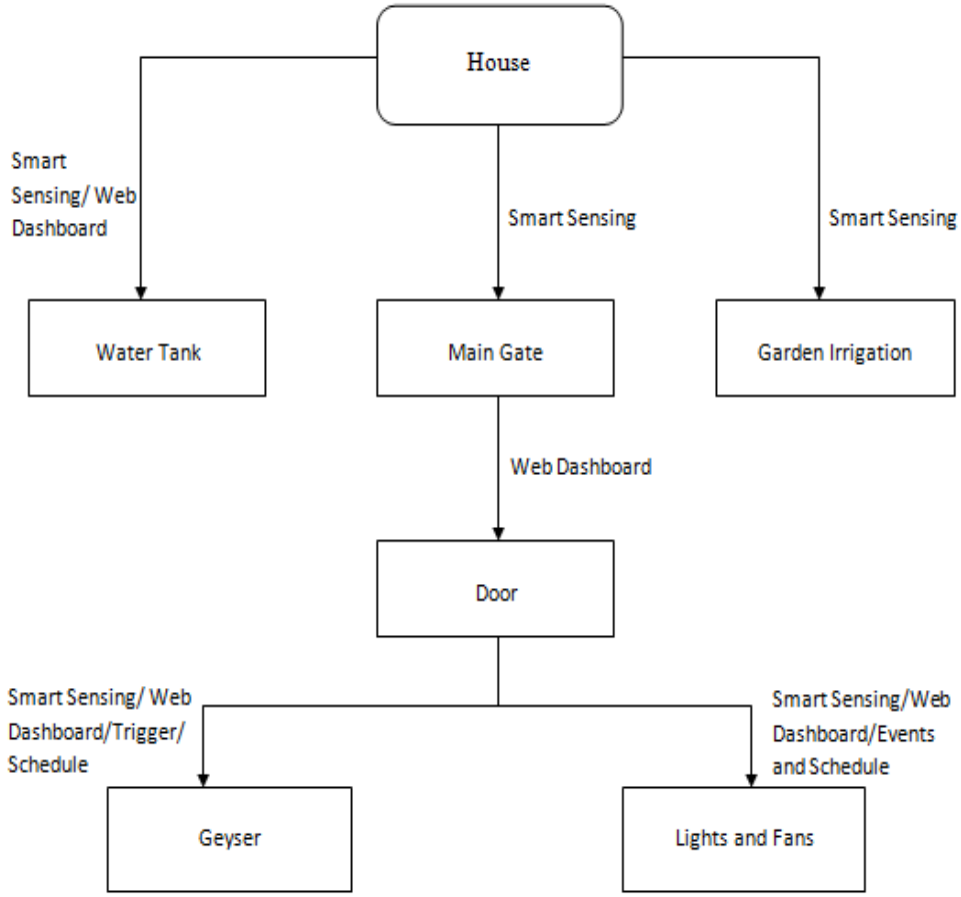


*Figure 6.4: Flow Diagram of the System*

According to the flow diagram, the house is connected to the main gate, water tank and garden irrigation system. The main gate is then connected to the door, which gives way to the inner part of the house, followed by the rooms with light, fan, geyser and AC. Smart Sensing concept is used in the entire automation wherein when a particular parameter is sensed, then the value will be checked in the algorithm or code to perform its function. Apart from Smart Sensing, the priority is given to the Web API which facilitates the user to control the functioning of the house during special cases. The flow diagram is with respect to the smart sensing concept.

## 6.3 Schedule, Trigger and Alert

### Schedule

A schedule is a time management tool consisting of a list of times at which events are to occur, or an order in which they are to occur.Schedules are program events for connected single board computers, microcontrollers, sensors and actuators. The event screen provides a simplified approach to creating schedules. Name the event, date and

time of occurrence, and the frequency. You can also setup an alert for a post-event notification. You can easily keep track of all your events. Simply go the events section where you can view these events by year, month, week, day, and as a complete list. You can also filter by device.

**Trigger**

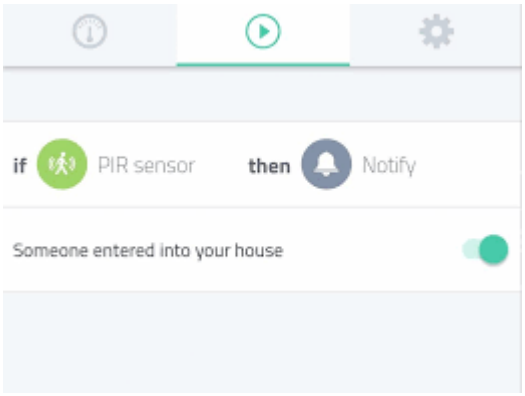


*Figure 6.5: Trigger Configuration*

A trigger is procedural code that is automatically executed in response to certain events on a particular table or view in a database. The trigger is mostly used for maintaining the integrity of the information on the database. It favors automation in a few easy steps with powerful if-then statements based on real-time data and actions. Device data capture in real-time enables the execution of triggered events.

**Alert**

Alerts enable text messages and Email notifications based on triggered events. Cayenne has a built in a rules engine to easily set up email and text alerts when a device or sensor reaches a predetermined state. With Cayenne fully integrated rules engine all you have to do is drag-and-drop devices into the Triggers & Alerts section to create threshold notifications to a single recipient or a group. Creating alerts powered by triggered events is easy. Alerts can be managed by naming the trigger and customizing the threshold limits from any data associated with connected devices.

## 6.4 Protocols Used

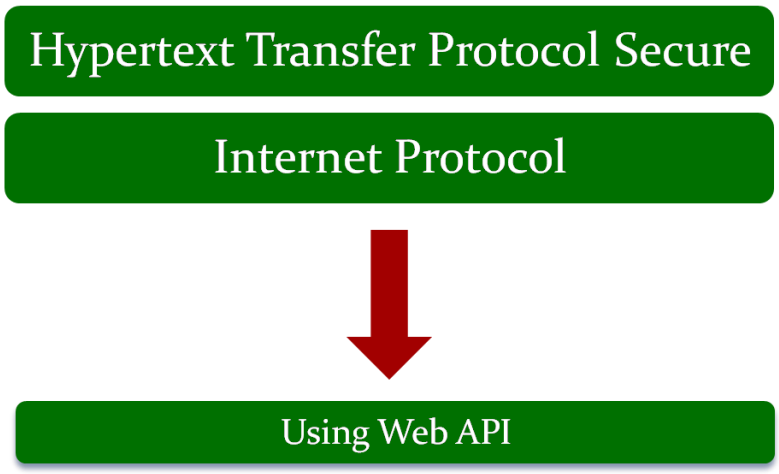


*Figure 6.6: Protocols*

The entire system makes use of two main protocols, namely, Hypertext Transfer Protocol Secure (HTTPS) and Internet Protocol (IP) using Web Application Interface.

**Hypertext Transfer Protocol Secure (HTTPS)**

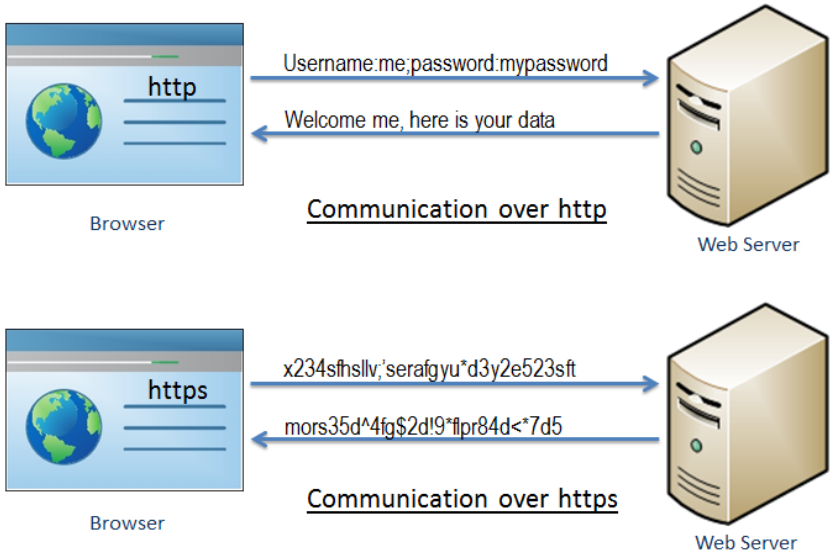


*Figure 6.7: HTTPS Working*

The Uniform Resource Identifier (URI) scheme HTTPS has identical usage syntax to the HTTP scheme. However, HTTPS signals the browser to use an added encryption layer of SSL/TLS to protect the traffic. SSL/TLS is especially suited for HTTP, since it can provide some protection even if only one side of the communication is authenticated.

This is the case with HTTP transactions over the Internet, where typically only the server is authenticated (by the client examining the server's certificate).

HTTPS creates a secure channel over an insecure network. This ensures reasonable protection from eavesdroppers and man-in-the-middle attacks, provided that adequate cipher suites are used and that the server certificate is verified and trusted.

HTTPS is especially important over insecure networks (such as public Wi-Fi access points), as anyone on the same local network can packet-sniff and discover sensitive information not protected by HTTPS. Additionally, many free to use and paid WLAN networks engage in packet injection in order to serve their own ads on web pages. However, this can be exploited maliciously in many ways, such as injecting malware onto web pages and stealing users' private information.

**Internet Protocol (IP)**

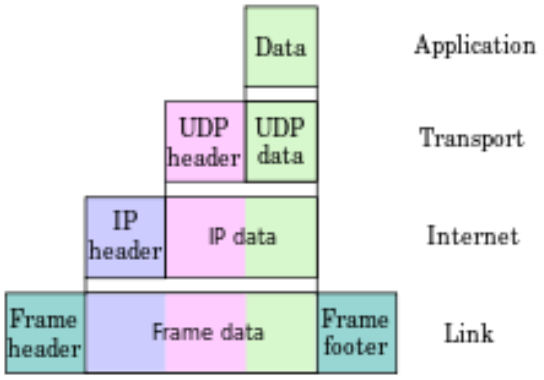


*Figure 6.8: Sample encapsulation of application data from UDP to a Link protocol frame*

The Internet Protocol (IP) is the method or protocol by which data is sent from one computer to another on the Internet. Each computer (known as a host) on the Internet has at least one IP address that uniquely identifies it from all other computers on the Internet. When you send or receive data (for example, an e-mail note or a Web page), the message gets divided into little chunks called packets. Each of these packets contains both the sender's Internet address and the receiver's address. Any packet is sent first to a gateway computer that understands a small part of the Internet. The gateway computer reads the destination address and forwards the packet to an adjacent gateway that in turn reads the destination address and so forth across the Internet until one gateway recognizes the packet as belonging to a computer within its immediate neighborhood or domain. That gateway then forwards the packet directly to the computer whose address is specified.

Because a message is divided into a number of packets, each packet can, if necessary, be sent by a different route across the Internet. Packets can arrive in a different order than the order they were sent in. The Internet Protocol just delivers them. It's up to another protocol, the Transmission Control Protocol (TCP) to put them back in the right order.

IP is a connectionless protocol, which means that there is no continuing connection between the end points that are communicating. Each packet that travels through the Internet is treated as an independent unit of data without any relation to any other unit of data. (The reason the packets do get put in the right order is because of TCP, the connection-oriented protocol that keeps track of the packet sequence in a message.) In the Open Systems Interconnection (OSI) communication model, IP is in layer 3, the Networking Layer.

**Web API**

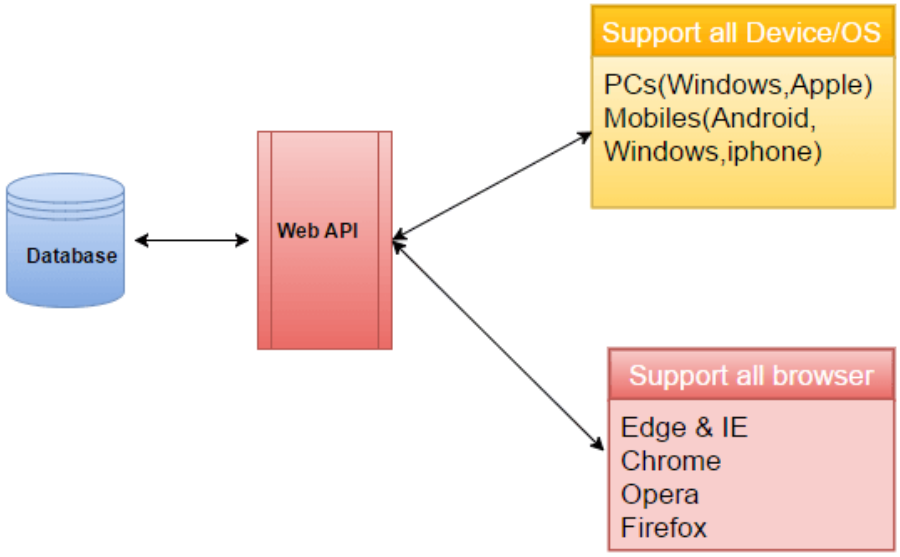


*Figure 6.9: Working of Web API*

In computer programming, an application programming interface (API) is a set of sub routine definitions, protocols, and tools for building software and applications. To put it in simple terms, API is some kind of interface which has a set of functions that allow programmers to access specific features or data of an application, operating system or other services. Web API as the name suggests, is an API over the web which can be

accessed using HTTP protocol. It is a concept and not a technology. We can build Web API using different technologies such as Java, .NET etc.

## 6.5 Budget

Budgeting is the process of creating a plan to spend money. As mentioned before that the project is affordable with a reasonable cost, the following shows a list of gadgets that were used in the automation with the cost.

| COMPONENTS | PRICE with GST (Rs.) |
|---|---|
| Node MCU | 320 |
| Relay (4 channel) | 660 |
| PIR sensor | 150 |
| LDR | 50 |
| Servo Motor | 200 |
| Moisture Sensor | 200 |
| Water level sensor | 100 |
| Ultrasonic range finder sensor | 150 |
| Temperature-Humidity sensor (DHT11) | 130 |
| Bread board | 50 |
| Router with Internet connection | Depends on service provider |
| RFID Kit | 240 |

*Figure 6.10: Budget*

The proposed system has resulted in a Grand Total of Rs. 2,250 without router and internet expense.

# CHAPTER 7

# IMPLEMENTATION

The implementation or execution of the entire home automation system was done in Arduino IDE software. This makes us of the Embedded C language. This language makes use of two mandatory functions, namely:

- Setup () function - This is called when a sketch starts. It is used to initialize variables, pin modes, start using libraries, etc. This will only run once, after each power up or reset of the NodeMCU board.
- Loop () function - This does precisely what its name suggests, and loops consecutively, allowing your program to change and respond. Use it to actively control the NodeMCU board.

Below is the snippets of the modules that have been used in the proposed system.

## 7.1 Gate Automation

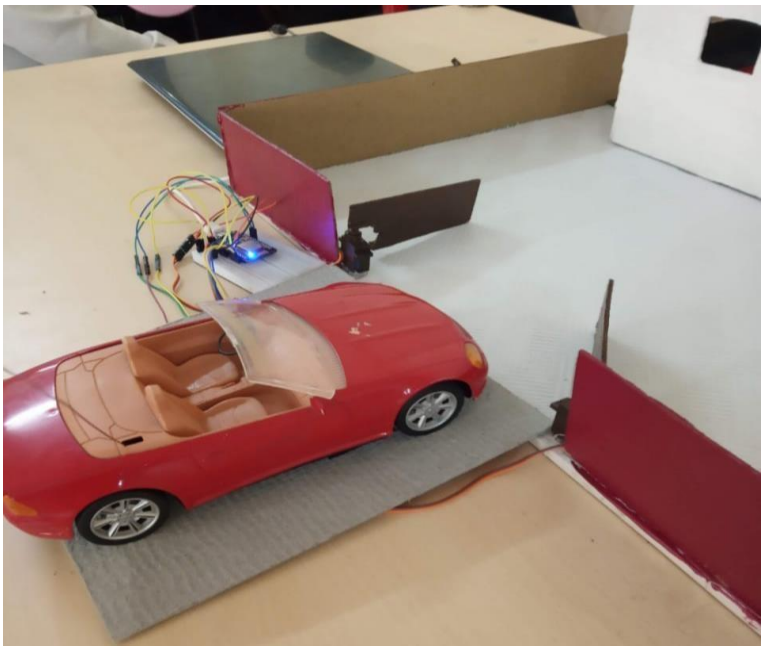

*Figure 7.1: RFID is read and Gate opens*

The following snippet code implies that once the RFID reader reads from the right RFID tag, then the gate has to open. The opening of the gate happens due to the servo motors that are attached to the gates.

```
void loop()
{
  // Look for new cards
  if ( ! mfrc522.PICC_IsNewCardPresent())
  {
    return;
  }
  // Select one of the cards
  if ( ! mfrc522.PICC_ReadCardSerial())
  {
    return;
  }
  //Show UID on serial monitor
  Serial.println();
  Serial.print(" UID tag :");
  String content= "";
  byte letter;
  for (byte i = 0; i < mfrc522.uid.size; i++)
  {
     Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
     Serial.print(mfrc522.uid.uidByte[i], HEX);
     content.concat(String(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " "));
     content.concat(String(mfrc522.uid.uidByte[i], HEX));
  }
  content.toUpperCase();
  Serial.println();
  int pos;
  if (content.substring(1) == "56 99 5E 1F") //change UID of the card that you want to
give access
  {
    Serial.println(" Access Granted ");
    Serial.println(" IOT GROUP!!! ");
    delay(1000);
    Serial.println();
```