TECHNICAL DOCUMENTATION - AUTOHOME

Group 15: Mahmoud Bachir (Leader), Aaron Christie, Smruthi Srikumar, Gunbir Singh, Brendan Li, Kyle Ross, Jimmy Wen, Karthikey Thapliyal

AUTOHOME
HTTPS://GITHUB.COM/MAHMOUDBACHIR/HOMEAUTOMATION
HTTPS://AUTOHOME.HOST/

Arduino Code

Note that the file "Arduino_integration.ino" is the file containing the final code. This file encompasses the code that controls the LED lights, the RGB LEDs and the Piezo buzzer that uses a Hall Effect sensor. The Arduino code can simply be uploaded as a sketch to an Arduino to be run repeatedly. As of now, this code enables the user to turn an LED bulb on or off, start and stop a RGB loop for the RGB LED and to enable the buzzer that turns on or off depending on the presence or absence of a magnet near the Hall Effect sensor. The Arduino must be used in conjunction with an ESP8266-12E WiFi module. The ESP8266-12E WiFi module must also have its own code uploaded and running for the output to work properly.

Should the user wish to enable the buzzer and Hall Effect sensor alone, he or she may upload the file "ArduinoHallEffect.ino" to the Arduino. The same may be done with "RGB_LED.ino" if the user would like to use the RGB LEDs alone. However, there is no apparent need to do this, as any of these outputs could be tested individually with "Arduino_integration.ino" by using buttons on the local server.

The hardware connections for the RGB LED are as follows:

1. Connect the RED cathode to pin 3
2. Connect the GREEN cathode to pin 5
3. Connect the BLUE cathode to pin 6
4. Connect the anode to 5V from the Arduino

The hardware connections for the Piezo buzzer are as follows:

1. Connect one side to the negative rail (GND)
2. Connect the opposite side to pin 13 of the Arduino

The hardware connections for the Hall Effect Sensor are as follows:

1.

The hardware connections for the Arduino are as follows:

1. Connect 5V to the positive rail of the breadboard
2. Connect GND to the negative rail of the breadboard
3. Connect digital pin 3 to the RED cathode of the RGB LED
4. Connect digital pin 5 to the GREEN cathode of the RGB LED
5. Connect digital pin 6 to the BLUE cathode of the RGB LED
6. Connect digital pin 7 to GPIO 14 (pin D5) of the ESP8266-12E

ESP8266-12E Code

The ESP8266-12E code can also be uploaded as a sketch from the Arduino IDE. To use this code properly, replace the string value of ssid with the user's network SSID and replace the string value of password with the network's corresponding password. To check if the device connected successfully, open the Serial Monitor in the Arduino IDE. The output on the Serial Monitor should be "Connected to [ssid]\nIP

address: [local IP address]" where the items in brackets are replaced by the user's choice of SSID and the actual local IP address,
respectively, and there is a new line where '\n' is placed. Note that for the output to be visible on the hardware, the anode of the single
color LED must be connected to GPIO 13 of the ESP8266-12E. GPIO 14 of the ESP8266-12E must be connected to pin 7 of the Arduino, as this is
the input that determines if the LED code is to run or not.

Server Code

Initially,we set up our Raspberry Pi as a local server by installing Apache, PHP, and MySQL and then tested it. We then converted the local server into a web server with IP address (100.8.176.108) through port forwarding technique from the router. After performing port forwarding, the router points to the raspberry pi which enabled us to access it through the internet.

To handle the administration of MySql and controlling the database we installed the PhpMyAdmin to our Raspberry Pi.

PhpMyAdmin can be accessed using the username ‘root’ and password ‘sqlpass’ which we chose while setting up the MySQL server.

Database Code

Step 1: Creating the Database Table

We created our database named members and executed the the SQL query by importing the file named db.sql to the MySQL Databases using PhpMyAdmin.

Steps to follow to import the SQL file

- In phpMyAdmin you will see a list of your databases on the left side. Select the database you would like to import to.
- Once the database is selected, click the "Import" tab at the top.
- In this subsection, a "browse" button will appear. Select this, and choose the desired .sql file on your PC.
- After you have selected the file, click "Go" at the bottom right; it will import your .sql file for you.

Step 2: Creating the Config File

After executing the Sql query we were able to create tables.We then created a PHP script to connect to the MySQL database server and named it config.php and within the file we entered the credentials pertaining to our database.

Website Code

Step 1: Creating the Sign Up Form

Using php script we created a form for the user to enter the credentials and verify them in order to access the members page.We implemented number of checks.

We are checking, if the form is submitted or not. In this step we have two logic's.

- What if the form is submitted.
    1. If the form is submitted, we are assigning the posted values to variables and checking the values are existing in the database or not.
    2. If the values submitted in the form and the values in the database are equal, then we will create a session for the user.
    3. If the values are not equal then it will display an error message.
    4. And then we checks for the session, if the session exists we will great him with the username, otherwise the form will be displayed.
- What if not the form is submitted.
    1. When the user comes first time, he will be displayed with a simple form.
    2. User Name, Email Address, Password and a submit button

Step 2: Creating the Login Form

Using php script we created a login form where user could enter their username and password. When user submits this form the inputs are verified against the credentials stored in the database, if the username and password match, the user is authorized and granted access to the site, otherwise the login attempt is rejected.