A Simple Home Monitoring System

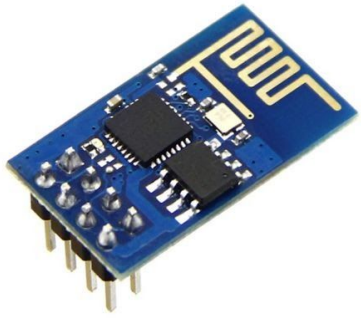
IoT Systems Project (2020)

This project is a simple model of an IoT-based home monitoring system. It consists of three parts: 1) sensors/actuators, 2) database, 3) web control panel. We make the first part in the Tinkercad simulator and the other two parts in Google Firebase service. The setup consists of 3 lamps, a temperature sensor, and a light sensor similar to what we may have in a room. We want to see the temperature and light sensor values on a web control panel. Additionally, we want to be able to turn the lamps on and off not only with a button in the room but also with a button in the web control panel. We also want to report the power consumption of our lamps in the web control panel.

1- Sensors/actuators

In addition to **3 lamps**, **temperature**, and **light sensor**, we need some other components in the Tinkercad to make this project. For example, the bulb lamps in Tinkercad are 8V and their current consumption is more than what Arduino pins can provide. Therefore we should use an **external power source** and **3 relays** to switch the lamps on and off with a **push button**. The button turns the first lamp on with one touch. With the next touch, it turns two lamps on, and with the third touch, it turns all three lamps on. The fourth touch turns all lamps off.
A **seven-segment** shows the number of turned-on lamps.
A **shunt resistor** is required to measure the current consumption of the lamps. One of the Arduino's analog inputs measures the voltage drop over this resistor and then we can calculate the current consumption.

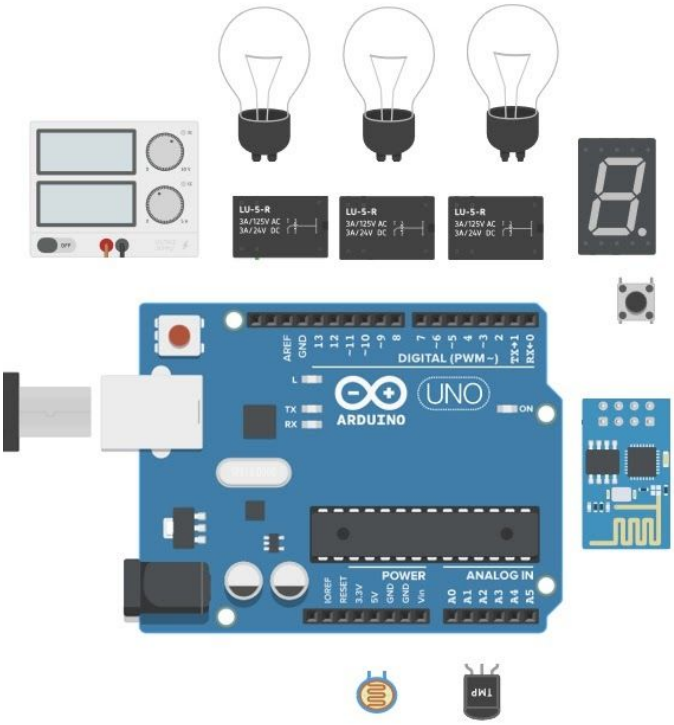
The key component in the setup is a famous WiFi module named **ESP8266**. The utilization of this module is a little tricky not only in the Tinkercad but also with the real component. The ESP8266 component in the Tinkercad is one of the earliest models of this module and has its original firmware that works only with AT commands. For the real component, there are plenty of tools available that make it possible to program this module even with C language in Arduino IDE, but this simulated Tinkercad module works only with a limited set of AT commands. Here are some hints for using this module:



- The module receives and replies to AT commands via Serial communication. Search for the ESP8266 AT commands and examples online.
- The baud rate for Serial communication should be exactly 115200 bps.
- For a real ESP8266 component, we were able to use the *Softwareserial* library on Arduino to communicate with the module over two custom digital pins and keep the Arduino's main serial pins to monitor the results of running an AT command. But

Tinkercad *Softwareserial* library cannot provide 115200 bps baud rate and we need to use the main Arduino RX/TX pins both for sending the AT command to the ESP8266 and for printing the results to the Serial Monitor.

- The voltage level of Arduino RX/TX pins signal is 5V, but ESP8266 module powers on with 3.3v and thus the signal voltage level for TX/RX pins is 3.3v. There is no level shifter component available in the Tinkercad and we should use two resistors to change the signal level. There are some examples available online to find how to do this.
- There is a simulated WiFi network available in the Tinkercad environment which provides internet connection for our setup. The WiFi network name is “Simulator Wifi” and it works without a password.

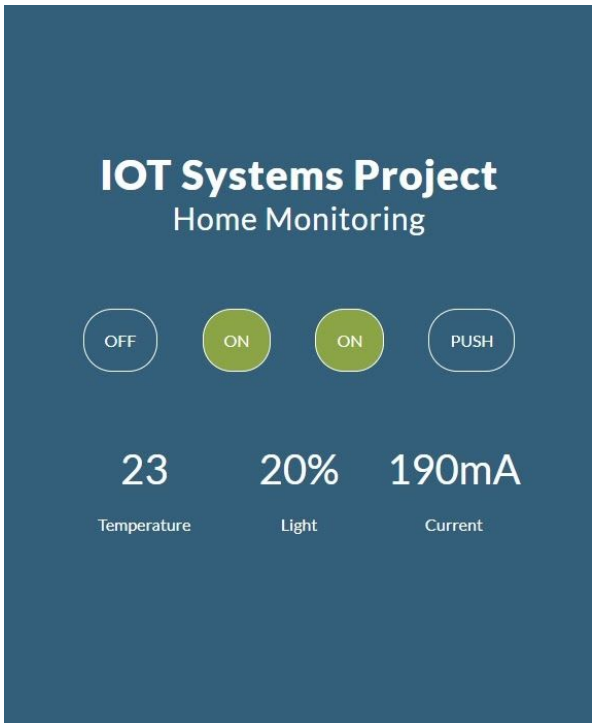


2- Database

We use Google Firebase service for our database. Google Firebase service is free and easy to set up via a google account. It provides a JSON structured database with a web address which we can send our data as a JSON HTTP PUT request to it and receive the contents of the database via an HTTP GET request. We expect the database to keep the record of the current room temperature/light, the current consumption of our setup, and the number of turned-on lamps. We may also need another field in this database to check if we have received a lamp turn on/off command from the control panel or not. This will help our setup to recognize which button is valid for turning lamps on/off, the button in the Tinkercad, or the button in the web control panel.

3- Control panel

Here we use the web hosting feature of the Google Firebase to make a simple web control panel. The Control panel shows the temperature, light, current consumption, and the status of the lamps. There is also a push-button on the panel and it should be able to turn the Tinkercad lamps on/off remotely, exactly like the push button in the Tinkercad. Firebase web hosting provides us a web address and we need to install *firebase-tools* on our computer to deploy our code. For this part, some knowledge about HTML, CSS, JavaScript is required.



Notes:

- Please self organize to groups of 1-3. Moodle discussion forum can be used for this purpose if needed.

- Grading is based on effort as well as end result. It is not required that everything works perfectly to pass this assignment. Good effort and demonstration of basic understanding of fundamental concepts is enough to pass and a working execution gives the best project work grade.

- Tinkercad simulator is very slow and AT commands are slow too. It is completely acceptable if there is up to 10 seconds delay between what happens in the room and what is shown on the web control panel.

- You may want to skip part 3 and just store sensor values and lamps status in the database if you want to work on this project alone.