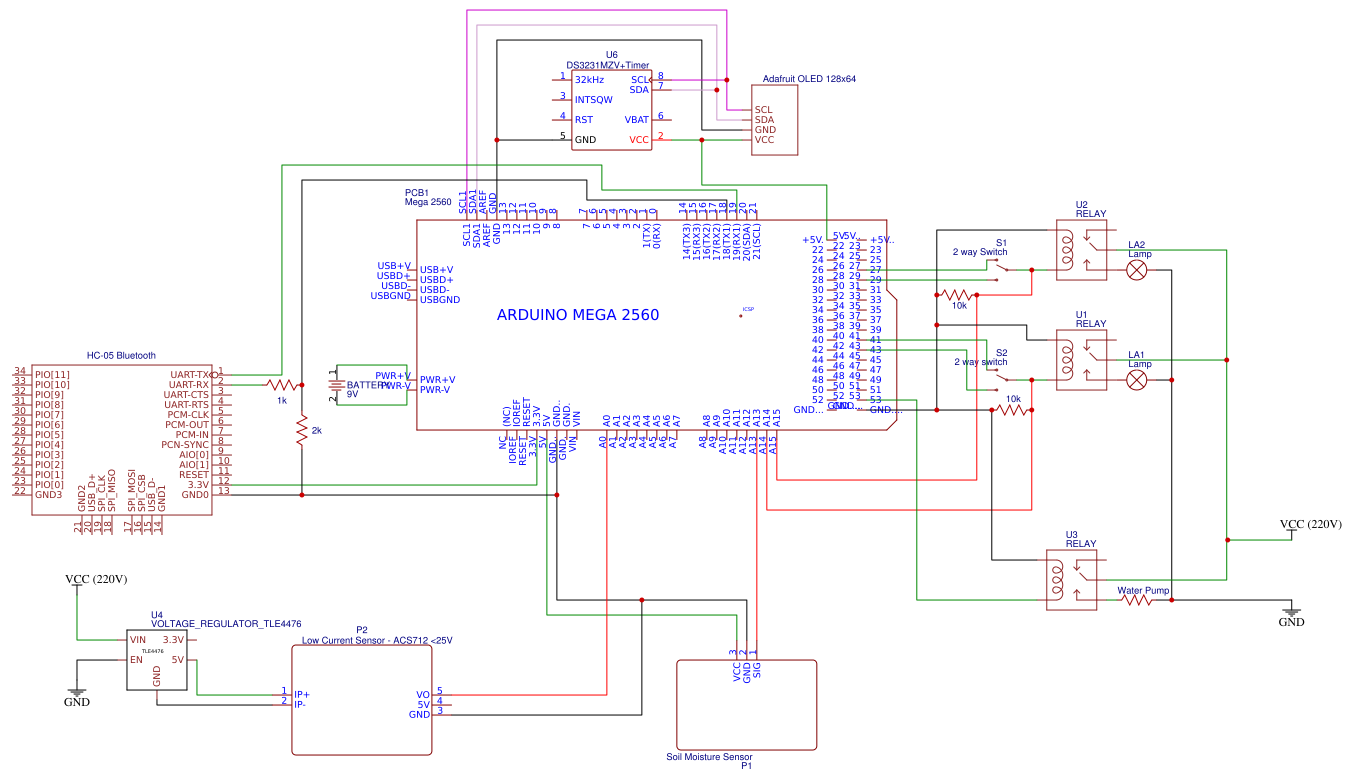
**Project report 17.9.2019 – 13.10.2019**

Emre Özincegedik

**Schematics 17.9.2019 – 19.9.2019**

 Due to complexity of modules and also high number of wires, the visualizing wouldn’t be done in paint. So a more professional looking website is used for showing the schematics, an online website – www.easyada.com is used. Below is the schematic for how the modules are wired together:

**Libraries 19.9.2019 – 20.9.2019**

Most of the code is handled thanks to Arduino’s own library. After examining the modules only 2 of them need additional libraries to utilize. The modules which require additional libraries are 128x64 OLED screen and DS3231 Clock module. The libraries are:

* Wire.h library: This library enables SDA/SCL communication protocols for Arduino for sending and receiving data more than only 0 and 1s. This library is used both for timer module and OLED screen module.
* RTClib.h: This library is for DS3231 Timer Module, which enables Arduino to get the current time and act accordingly to certain events
* Adafruit\_SSD1306.h: This library enables Arduino to utilize OLED Screen for showing data, which is received by other modules

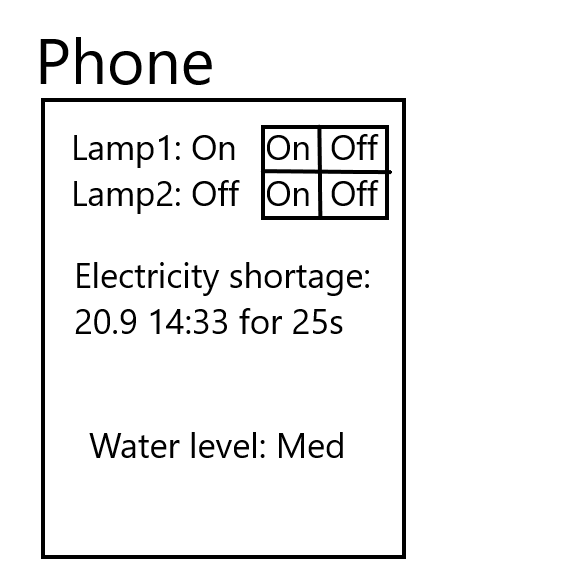
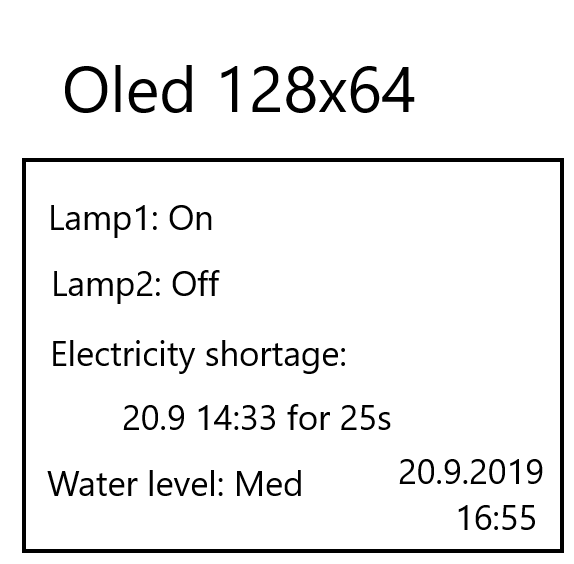
**Algorithm 20.9.2019 – 25.9.2019**

Algorithm for each module is currently separated from each other for simplicity and they will be combined while implementing code. Algorithms for modules will work as explained below:

* Arduino will communicate with timer module via SDA/SCL communication protocol and get current time and date. Time will be shown in OLED screen
* Soil moisture module will send data to Arduino, Arduino will process the data and if the water level is low, Arduino will send to water pump’s relay and water pump will open. When the water level is high enough Arduino will stop pump’s relay which will stop pump. Water level will be shown in both OLED screen and phone. (Phone uses bluetooth module with serial communication)
* Arduino will read electricity status from electricity module. If power is cut from Arduino, a timer will start and Arduino will show when the last power cut happened on OLED screen and phone and for how long it happened
* Arduino reads lamps status and shows on both phone and OLED screen
* Phone sends signal through bluetooth and Arduino processes incoming signal. According to incoming signal, either lamp1 or lamp2 will be opened or closed.

This part ended 5 days early than expected (Expected time was 10 days) The reason it ended so fast is looking at the schematics enabled me to produce the logic way faster than I anticipated.

**Interface Design 25.9.2019 – 26.9.2019**

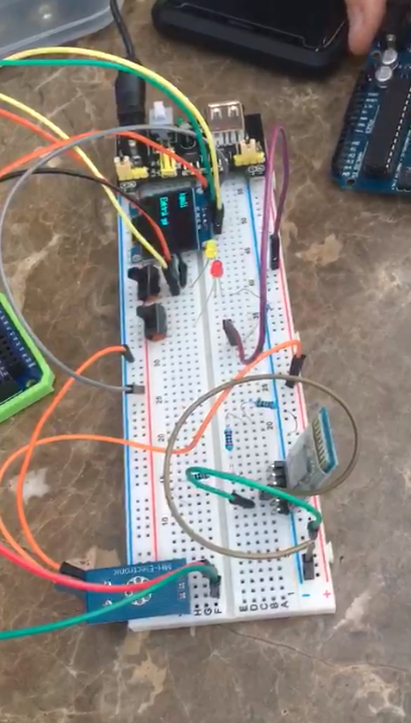
 Since this is not implementing, but rather designing how the OLED screen and phone will show the data to user. How the visuals will look in screen and phone are shown with Microsoft’s Paint program:

**Design Summary 26.9.2019 – 27.9.2019**

In short, this part is not coding but making short way for how coding logic and interface will look like

**Wiring Components 27.9.2019 – 1.10.2019**

Due to physical size of the project, the lamp and relay’s cables which connect to the wall’s electricity got disconnected a lot while moving the project in workspace. So LED lamps will be used for testing. The prototype will have real lamps.



**Exams 1.10.2019 – 8.10.2019**

Project is delayed for 1 week because of school exams.

**Code Implementation 9.10.2019 – 13.10.2019**

Only code snippets for modules’ algorithm logic are done in this part. They all will be combined later in the Combining section

**Soil moisture code snippet (30 locs):**

#define moisture\_pin A13

#define water\_pump 53

void setup() {

pinMode(moisture\_pin, INPUT);

pinMode(water\_pump, OUTPUT);

digitalWrite(water\_pump, LOW);

Serial.begin(9600);

}

char level[];

void loop() {

int moisture=analogRead(moisture\_pin); //moisture is 0-1023

if (moisture<200){

pinMode(water\_pump, HIGH);

level="LOW";

}

else if(moisture<600){

level="MED";

}

else{

pinMode(water\_pump,LOW);

level="HIGH";

}

Serial.print("Water level: ");

Serial.println(level);

//Also OLED screen

}

**Time and Oled Screen code snippet (52 locs):**

#include <Wire.h>

#include "RTClib.h"

#include <Adafruit\_SSD1306.h>

#define OLED\_RESET 4

Adafruit\_SSD1306 display(OLED\_RESET);

RTC\_DS1307 RTC;

void setup () {

Serial.begin(9600);

display.begin(SSD1306\_SWITCHCAPVCC, 0x3C);

display.clearDisplay();

Wire.begin();

RTC.begin();

if (! RTC.isrunning()) {

Serial.println("RTC is NOT running!");

RTC.adjust(DateTime(F(\_\_DATE\_\_), F(\_\_TIME\_\_)));

}

}

void loop () {

DateTime now = RTC.now();

Serial.print(now.year(), DEC);

Serial.print('/');

Serial.print(now.month(), DEC);

Serial.print('/');

Serial.print(now.day(), DEC);

Serial.print(' ');

Serial.print(now.hour(), DEC);

Serial.print(':');

Serial.print(now.minute(), DEC);

Serial.print(':');

Serial.print(now.second(), DEC);

Serial.println();

delay(1000);

display.clearDisplay();

display.setTextSize(1);

display.setTextColor(WHITE,BLACK);

display.setCursor(95,45);

display.print(now.hour(), DEC);

display.print(':');

display.print(now.minute(), DEC);

//display.print(':');

//display.print(now.second(), DEC);

display.setCursor(95,15);

display.print(now.day(), DEC);

display.print('/');

display.print(now.month(), DEC);

display.print('/');

display.print(now.year(), DEC);

display.display();

}

**Lamps status only snippet (26 locs):**

#define lamp1\_read A15

#define lamp2\_read A14

void setup(){

Serial.begin(9600);

pinMode(lamp1\_read, INPUT);

pinMode(lamp2\_read, INPUT);

}

void loop(){

int lamp1\_status = analogRead(lamp1\_read); //0-1023

int lamp2\_status = analogRead(lamp2\_read);

if (lamp1\_status>600)

{

Serial.println("Lamp1: On")

}

else

{

Serial.println("Lamp1: Off")

}

if (lamp2\_status>600)

{

Serial.println("Lamp2: On")

}

else

{

Serial.println("Lamp2: Off")

}

}

**Project is 2 days slower than anticipated**