1. **Short description of your challenge/problem:**

Nowadays people often go grocery shopping to buy food for the family and friends. Furthermore those people also want to improve the environment, such as decreasing pollution. The problem is, that people usually do not know how bad the waste and carbon footprint is, while buying the food at the grocery store.
First of all there is a lot of fuel, water etc needed to grow vegetables. Second, since far too many people are picky with the looks of the food, they sometimes just get thrown away, when they are optically unpleasant. The latter is already being taken care of, by selling them for a lower price, but sometimes it is just not enough. Furthermore to keep them looking good, the food usually has some chemicals,which are not healthy. Last but not least, the transport is a big pollution problem. Food is being transported all over the world, mainly because some foods cannot be grown anywhere, due to climate reasons mostly but also because of production costs.

2. **Short description of your solution:**

Food unfortunately often has a big carbon and waste footprint. People just want to have a lot of food, that have to be flown, shipped and corgoed in and require a lot of work, food and soil. So the solution should include decreasing all the mentioned problems as much as possible. Vertical Farming could be the solution or at least the footstep for it. Our vertical farm for instance can be built in any household, which means the carbon footprint problem decrease extremely, because the full grown food does not have to be transported.
Furthermore, the owners of the vertical farm would not have to put chemicals into the food, because they decide what they want to have when to eat. The food no longer requires chemicals which give longer durability to the food until it gets bad. The latter means that the food is grown naturally, which is definitely a healthy way to grow food that gets eaten. Another argument for the vertical farm against the normal field growing food is the fact that the vertical farm is inside, it is much easier to control the lighting and temperature. The climate can be manipulated within the product, which means no food has to be delivered from the other side of the world.
Water wastage also gets decreased as a matter of fact. When food gets grown, they normally grow in huge fields, which means that water and soil gets wasted, because it is just a lot of work to do everything precisely and technology is just not near to be perfect. Vertical farms can be grown in any household, which means less food, soil and water are needed. Furthermore the watering technology can be used to its full potential, since there are lesser plants in a vertical farm, than on a field.

3. Description of method and tools used (software, hardware):

For the project on the basic structure of the hardware we used laser cut wood. 3d printed aqueducts were used for water irrigation system. LED strips (RGB and UV) that can emit essential lights are used to manipulate the sun function to let the herbs grow to its full potential. We have chosen basil and leaf lettuce for our project, because it is known to grow fast, which is easier for us to show that our project works. To cultivate plants automatically, several sensors are used. A laser sensor is used to detect if plants are not interfering with the light distribution. Furthermore multiple moisture sensors (2 per shelf) detect the humidity of the soil and provide a so called “dryness level” to our arduino ranging from 0 to 100. The watering system waters the plants according to the dryness level of each shelf. They are watered separately so one shelf could get 100ml while the other gets 150ml. The machine has a clock (Real Time Clock module), which even though it is turned off, shows the right time, when the machine is back on. When plugging in the machine, it will start working from the actual current time instead of a static set one.

Our system uses 2 arduinos, where one is used for some general purpose functions and time communications to the other arduino. We chose for this because if the system were to be expanded one only has to add arduinos or other microcontrollers for the sensors and lighting and you can connect that microcontroller to the central unit that communicates the same base values to all other arduinos.

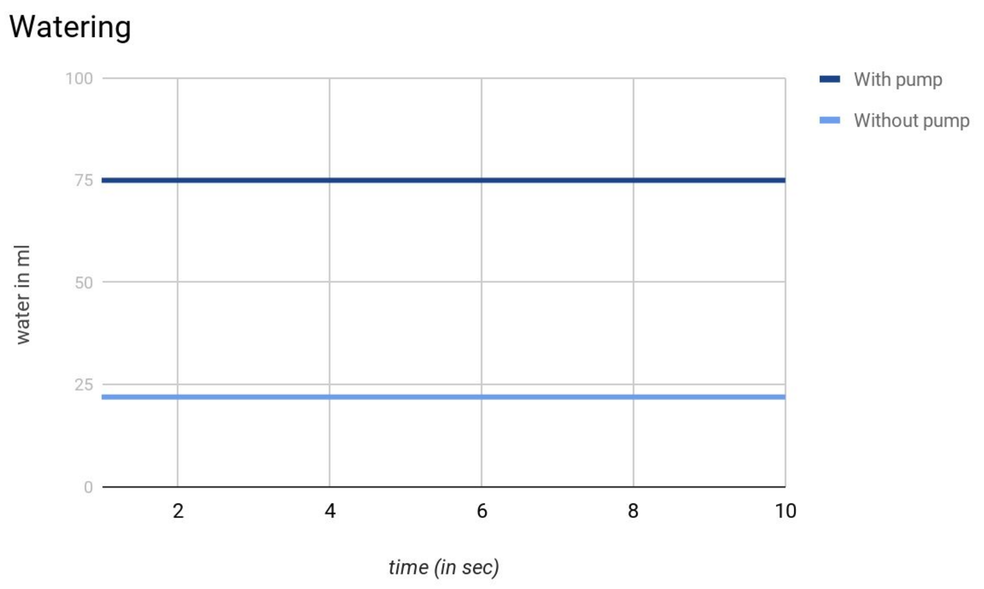
We have added microswitches to the shelves, which detects when the shelf is placed correctly or not. Provided the shelf is well placed, the watering and lighting system goes on. If the shelf is placed incorrectly the whole machine will not be activated.

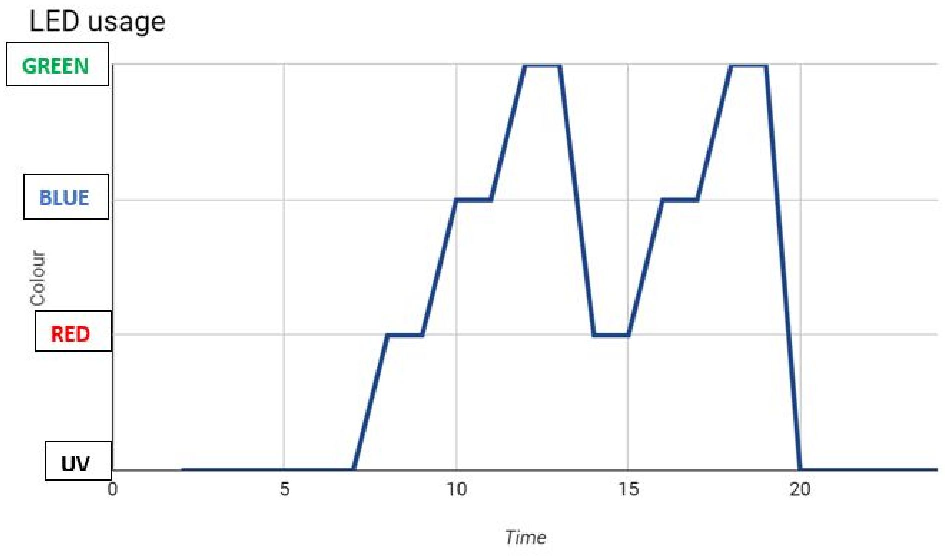
This is a list of all the components that are in the system:

- Controlled Led strips
- Relays
- Motorized valves
- Moisture sensors
- Height sensors
- Laser with laser receiver
- Indication leds
- Arduino uno boards
- Laser cut structures out of wood
- 3D printed parts
- Plastic bottles as holding containers for water
- Containers for leaf lettuce and basil
- Plastic tubes for water transport
- Containers that will capture water remainder

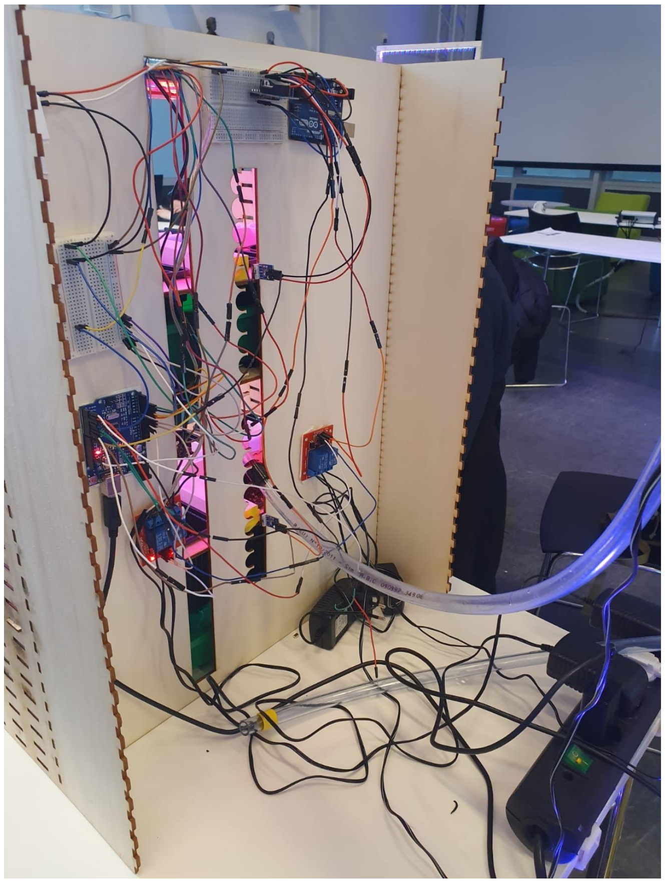
4. **Results/observation/data gathered/graphs:**

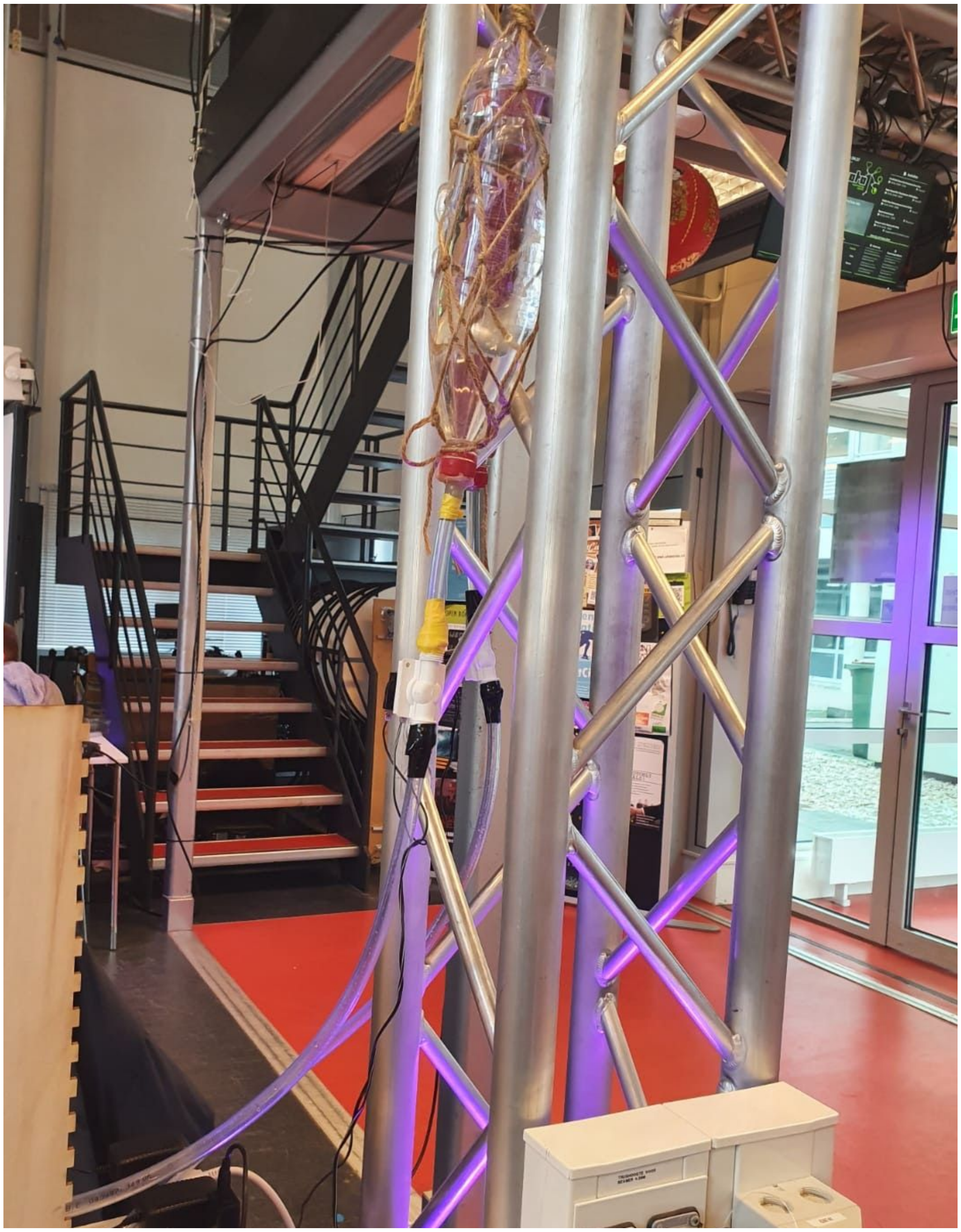
We have been growing the plants for about a week now. As a matter of fact the plants have been proportionally been growing faster when the light and watering system worked, than when we watered it manually and put in the dutch "sunlight". Furthermore the plants have not died yet, so till now we could say the product is working well. We used regular LED strips for the lighting. Our idea of customizable lights suited to each plant should work in theory but because we did not use a spectrometer to check if the lights are exactly the lights that we need it won't work as efficiently as with the proper tested lights.

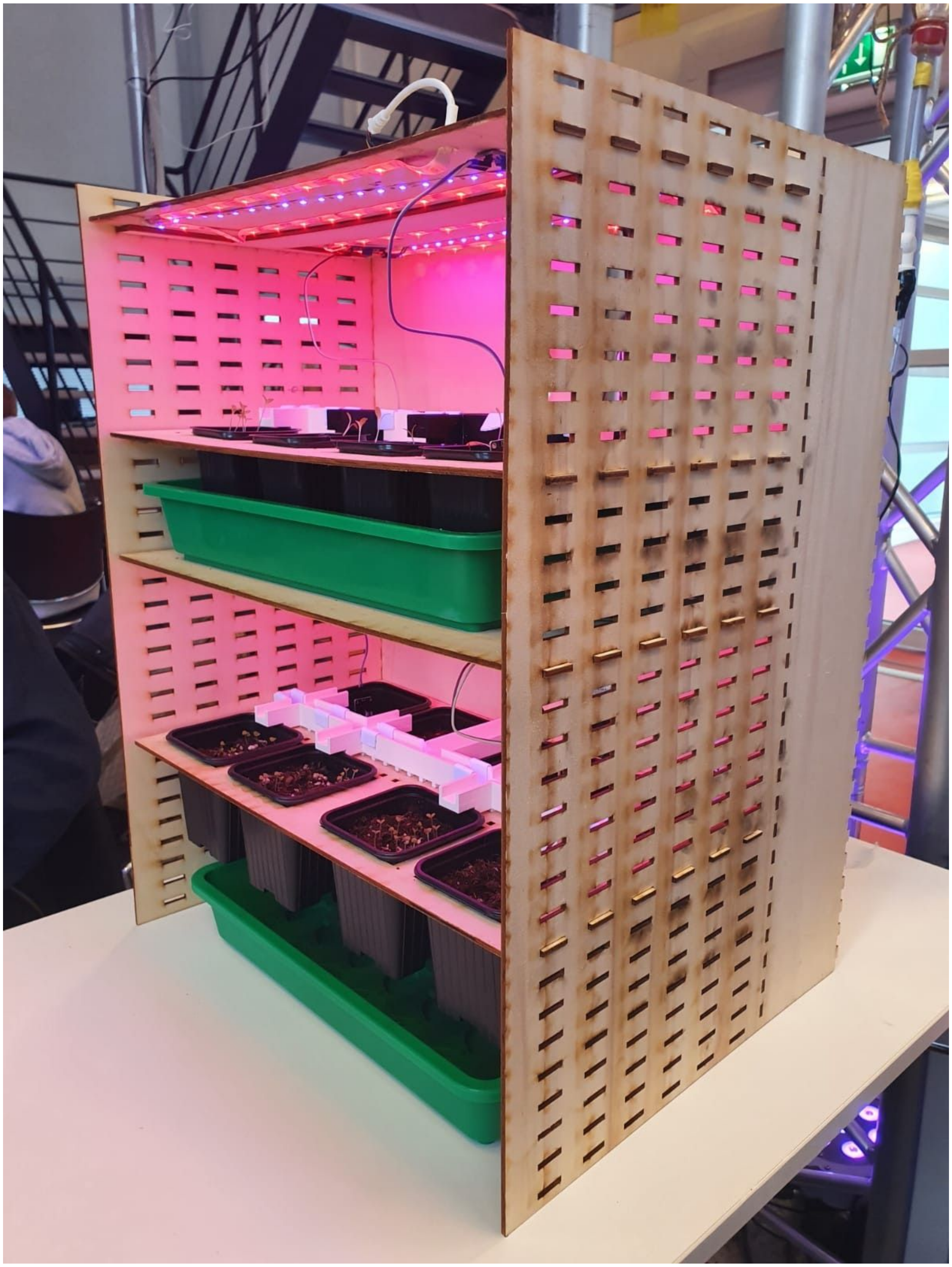


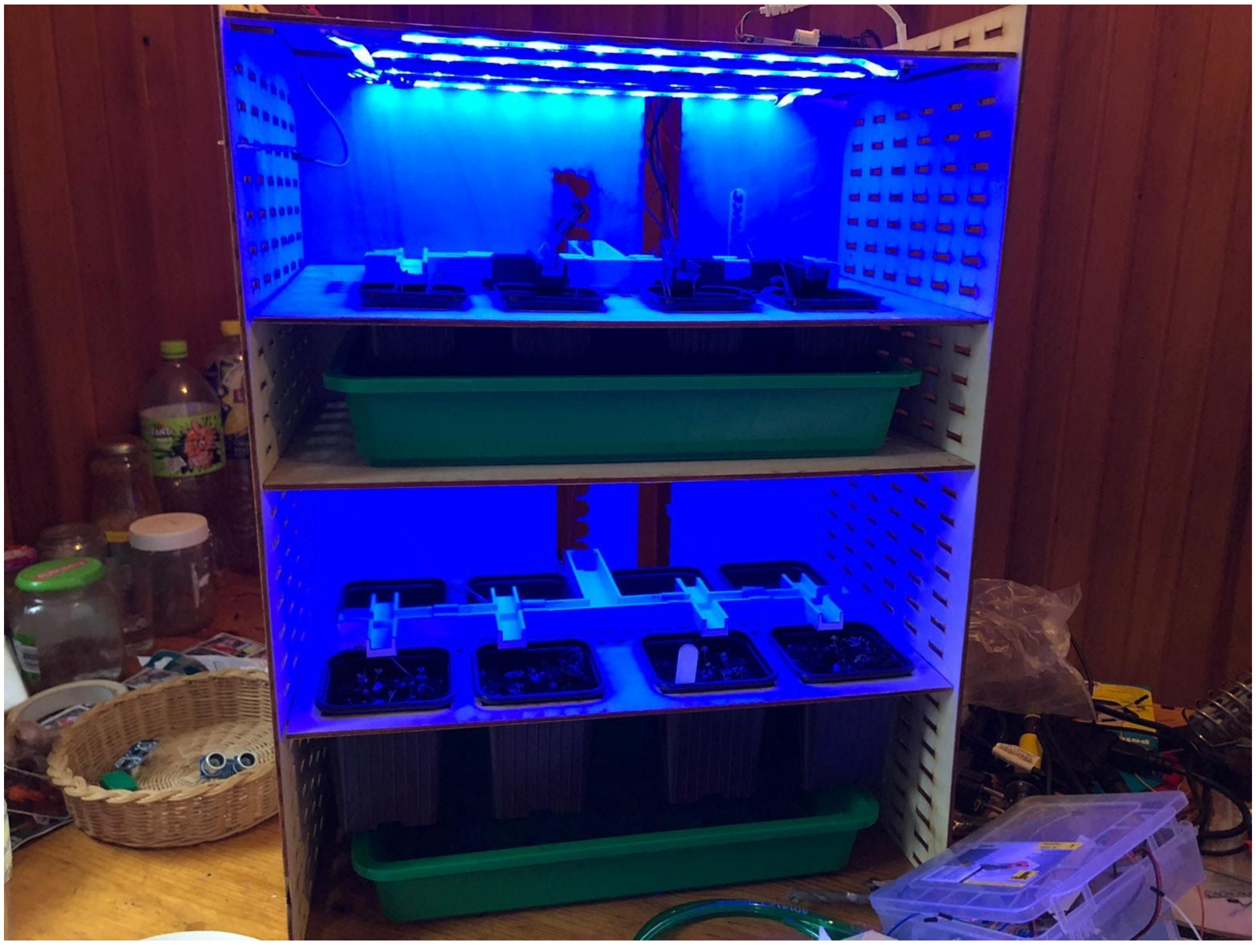


5. Up to 5 photos of your prototype









Code for the timekeeping hub:

```
#include "Wire.h"
#define DS3231_I2C_ADDRESS 0x68
// Convert normal decimal numbers to binary coded decimal
byte decToBcd(byte val)
{
  return( (val/10*16) + (val%10) );
}
// Convert binary coded decimal to normal decimal numbers
byte bcdToDec(byte val)
{
  return( (val/16*10) + (val%16) );
}
void setup()
{
  Wire.begin();
  Serial.begin(9600);
  // if you activate the next two lines then you can change the start settings of the realtime clock
  // DS3231 seconds, minutes, hours, day, date, month, year
```

```
   //setDS3231time(30,42,21,4,26,11,14);
}
void setDS3231time(byte second, byte minute, byte hour, byte dayOfWeek, byte
dayOfMonth, byte month, byte year)
{
  // sets time and date data to DS3231
  Wire.beginTransmission(DS3231_I2C_ADDRESS); // here are the start settings of the realtime clock.
  Wire.write(0); // set next input to start at the seconds register
  Wire.write(decToBcd(20)); // set seconds
  Wire.write(decToBcd(35)); // set minutes
  Wire.write(decToBcd(18)); // set hours
  Wire.write(decToBcd(6)); // set day of week
  Wire.write(decToBcd(24)); // set date
  Wire.write(decToBcd(1)); // set month
  Wire.write(decToBcd(20)); // set year
  Wire.endTransmission();
}
void readDS3231time(byte *second,
byte *minute,
byte *hour,
byte *dayOfWeek,
byte *dayOfMonth,
byte *month,
byte *year)
{
  Wire.beginTransmission(DS3231_I2C_ADDRESS);
  Wire.write(0); // set DS3231 register pointer to 00h
  Wire.endTransmission();
  Wire.requestFrom(DS3231_I2C_ADDRESS, 7);
  // request seven bytes of data from DS3231 starting from register 00h
  *second = bcdToDec(Wire.read() & 0x7f);
  *minute = bcdToDec(Wire.read());
  *hour = bcdToDec(Wire.read() & 0x3f);
  *dayOfWeek = bcdToDec(Wire.read());
  *dayOfMonth = bcdToDec(Wire.read());
  *month = bcdToDec(Wire.read());
  *year = bcdToDec(Wire.read());
}
void displayTime()
{
  byte second, minute, hour, dayOfWeek, dayOfMonth, month, year;
  // retrieve data from DS3231
  readDS3231time(&second, &minute, &hour, &dayOfWeek, &dayOfMonth, &month,
  &year);
  // send it to the serial monitor

  Serial.write(hour);
  //Serial.println(hour);
```

```
}
void loop()
{
  displayTime(); // display the real-time clock data on the Serial Monitor,
  delay(1000); // every second
}
```

The code for the watering and LEDs. :

```
/*"Supherb's" automatic rack based plant growing system with dynamic lighting and automatic
watering. Before use, test the amount of water your particular solution (so water bottle+gravity
 * or a reservoir with pump) pump per second. Fill proper values in for the integers "small" "medium"
and "large". This is the time the valves stay open. ALSO VERY IMPORTANT: check the moisture levels
through.
 * the serial monitor and fill in the according values in at the integers "lowlevel", "highlevel" and
"midlevel". These are the values that communicate the amount of water that is needed at watering
time. Make
 * sure to test this with the soil you are going to use.
*/

 #include <IRremote.h>

IRsend irsend;
int mPin[] = {0, 1, 2, 3}; // The four moisture sensors
int waterVal[] = {A0, A1, B0, B1};
int waterPer[] = {0, 1, 2, 3};
int rPin1 = 4;
int rPin2 = 5;
int ledPinH = 10;
int ledPinL = 11;
int upperLayer;
int lowerLayer;
int valHigh;
int valLow;
int highPin = 8;
int lowPin = 7;
int EMERGENCYLEVEL = 95;
int LOWLEVEL = 50;
int MIDLEVEL = 70;
int HIGHLEVEL = 85;
int cycle = 0;
int x = 0;
int UVPin = 6;
int ledPin = 2;
int harvestLed = 9;
int val1 = 0;
int resetPin = 13;
int morning = 8;
int evening = 20;
int hour;
int small;
int medium;
int large;
unsigned long current;
unsigned long currentTimeA;
unsigned long currentTimeB;
unsigned long currentTimeC;
unsigned long currentTimeD;
unsigned long currentTimeE;
unsigned long currentTimeF;
unsigned long currentTimeG;
unsigned long currentTimeH;
unsigned long currentTimeI;
unsigned long IR;
boolean water = true;
boolean waterL = true;
boolean on = false;
boolean interrupt = false;
boolean change = true;
boolean Stime = false;


void setup() {
  Serial.begin(9600);
  pinMode(5, INPUT);
  pinMode(rPin1, OUTPUT);
  pinMode(rPin2, OUTPUT);
  pinMode(ledPinL, OUTPUT);
  pinMode(ledPinH, OUTPUT);
  pinMode(ledPin, OUTPUT);
  pinMode(harvestLed, OUTPUT);
  pinMode(UVPin, OUTPUT);
  pinMode(resetPin, INPUT);
  pinMode(lowPin, INPUT);
  pinMode(highPin, INPUT);
  currentTimeI = millis();
}
void loop() {
  if (Serial.available() > 0) {
    hour = Serial.read();
    Serial.println(hour);
  }
  valHigh = digitalRead(highPin);
  valLow = digitalRead(lowPin);
```

```
if (valHigh == 1 && interrupt == false) {
  digitalWrite(ledPinH, HIGH);
} else {
  digitalWrite(ledPinH, LOW);
}
if (valLow == 1 && interrupt == false) {
  digitalWrite(ledPinL, HIGH);
} else {
  digitalWrite(ledPinL, LOW);
}

if (millis() > currentTimeI + 750) {
  for (int i = 0; i < 4; i++) {
    waterVal[i] = analogRead(mPin[i]);
    waterPer[i] = map(waterVal[i], 250, 1023, 0, 100);
  }
  upperLayer = waterPer[0] + waterPer[1];
  upperLayer = upperLayer / 2;
  lowerLayer = waterPer[2] + waterPer[3];
  lowerLayer = lowerLayer / 2;
  Serial.print("UP");
  Serial.println(upperLayer);
  Serial.print("lOW");
  Serial.println(lowerLayer);
  Serial.println(valHigh);
  Serial.println(valLow);
  currentTimeI = millis();
}
if (cycle < 10) {
  if (hour >= morning && hour <= evening) {
    if (on == false) {
      digitalWrite(ledPin, HIGH);
      x = morning;
      on = true;
    }
    if (hour == x || hour == x + 1) {
      digitalWrite(ledPin, HIGH);
      for (int i = 0; i < 10; i++) {
        if (millis() - IR >= 2000) {
          irsend.sendNEC(0xF720DF, 32);  //RED
          IR = millis();
        }
      }
    }
    if (hour == x + 2 || hour == x + 3) {
      digitalWrite(ledPin, HIGH);
      for (int i = 0; i < 10; i++) {
        if (millis() - IR >= 2000) {
```

```
      irsend.sendNEC(0xF7609F, 32);  //BLUE
      IR = millis();
    }
  }
}
if (hour == x + 4 || hour == x + 5) {
  digitalWrite(ledPin, HIGH);
  digitalWrite(UVPin, HIGH); //UV
  for (int i = 0; i < 10; i++) {
    if (millis() - IR >= 2000) {
      irsend.sendNEC(0xF7A05F, 32); //GREEN
      IR = millis();
    }
  }
}
if (hour == x + 6 || hour == x + 7) {
  digitalWrite(ledPin, HIGH);
  digitalWrite(UVPin, HIGH); //UV
  for (int i = 0; i < 10; i++) {
    if (millis() - IR >= 2000) {
      irsend.sendNEC(0xF720DF, 32);  //RED
      IR = millis();
    }
  }
}
if (hour == x + 8 || hour == x + 9) {
  digitalWrite(ledPin, HIGH);
  digitalWrite(UVPin, HIGH); //UV
  for (int i = 0; i < 10; i++) {
    if (millis() - IR >= 2000) {
      irsend.sendNEC(0xF7609F, 32);  //BLUE
      IR = millis();
    }
  }

}
if (hour == x + 10 || hour == x + 11) {
  digitalWrite(ledPin, HIGH);
  digitalWrite(UVPin, LOW); //UV
  for (int i = 0; i < 10; i++) {
    if (millis() - IR >= 2000) {
      irsend.sendNEC(0xF7A05F, 32); //GREEN
      IR = millis();
    }
  }
}
if (hour >= x + 12) {
  digitalWrite(ledPin, LOW);
```

```
        digitalWrite(UVPin, LOW); //UV
        on = false;
      }
      cycle + 1;
    }
  }
  else if (cycle < 20) {
    if (hour >= morning && hour <= evening) {
      if (on == false) {
        digitalWrite(ledPin, HIGH);
        x = morning;
        on = true;
      }
      if (hour == x || hour == x + 1) {
        for (int i = 0; i < 10; i++) {
          if (millis() - IR >= 2000) {
            irsend.sendNEC(0xF7A05F, 32); //GREEN
            IR = millis();
          }
        }
      }
      if (hour == x + 2 || hour == x + 3) {
        digitalWrite(UVPin, HIGH); //UV
        for (int i = 0; i < 10; i++) {
          if (millis() - IR >= 2000) {
            irsend.sendNEC(0xF7609F, 32);  //BLUE
            IR = millis();
          }
        }
      }
      if (hour == x + 4 || hour == x + 5) {
        digitalWrite(UVPin, HIGH); //UV
        for (int i = 0; i < 10; i++) {
          if (millis() - IR >= 2000) {
            irsend.sendNEC(0xF720DF, 32);  //RED
            IR = millis();
          }
        }
      }
      if (hour == x + 6 || hour == x + 7) {
        digitalWrite(UVPin, HIGH); //UV
        for (int i = 0; i < 10; i++) {
          if (millis() - IR >= 2000) {
            irsend.sendNEC(0xF7609F, 32);  //BLUE
            IR = millis();
          }
        }
      }
      if (hour == x + 8 || hour == x + 9) {
        digitalWrite(UVPin, LOW); //UV
      }
      if (hour == x + 10 || hour == x + 11) {
        for (int i = 0; i < 10; i++) {
          if (millis() - IR >= 2000) {
            irsend.sendNEC(0xF720DF, 32);  //RED
            IR = millis();
          }
        }
      }
      if (hour == x + 12) {
        digitalWrite(ledPin, LOW);
        on = false;
      }
      cycle + 1;
    }
  }
  else {
    if (hour >= morning && hour <= evening) {
      if (on == false) {
        digitalWrite(ledPin, HIGH);
        x = morning;
        on = true;
      }
      if (hour == x) {
        for (int i = 0; i < 10; i++) {
          if (millis() - IR >= 2000) {
            irsend.sendNEC(0xF7A05F, 32); //GREEN
            IR = millis();
          }
        }
      }
      if (hour == x + 1) {
        for (int i = 0; i < 10; i++) {
          if (millis() - IR >= 2000) {
            irsend.sendNEC(0xF7609F, 32);  //BLUE
            IR = millis();
          }
        }
      }
      if (hour == x + 2 || hour == x + 3) {
        digitalWrite(UVPin, HIGH); //UV
        for (int i = 0; i < 10; i++) {
          if (millis() - IR >= 2000) {
            irsend.sendNEC(0xF720DF, 32);  //RED
            IR = millis();
          }
      }
    }
    if (hour == x + 4 || hour == x + 5) {
      digitalWrite(UVPin, HIGH); //UV
      for (int i = 0; i < 10; i++) {
        if (millis() - IR >= 2000) {
          irsend.sendNEC(0xF7609F, 32);  //BLUE
          IR = millis();
        }
      }
    }
    if (hour == x + 6 || hour == x + 7) {
      digitalWrite(UVPin, HIGH); //UV
      for (int i = 0; i < 10; i++) {
        if (millis() - IR >= 2000) {
          irsend.sendNEC(0xF720DF, 32);  //RED
          IR = millis();
        }
      }
    }
    if (hour == x + 8 || hour == x + 9 || hour == x + 10 || hour == x + 11) {
      digitalWrite(UVPin, HIGH); //UV
      for (int i = 0; i < 10; i++) {
        if (millis() - IR >= 2000) {
          irsend.sendNEC(0xF7609F, 32);  //BLUE
          IR = millis();
        }
      }
    }
    if (hour == x + 12) {
      digitalWrite(ledPin, LOW);
      digitalWrite(UVPin, LOW); //UV
      on = false;
    }
    cycle + 1;
  }
}
if (cycle > 30) {
  digitalWrite(harvestLed, HIGH);
}
val1 = digitalRead(resetPin);
if (val1 == 1) {
  digitalWrite(harvestLed, LOW);
  cycle = 0;
}
```

```
  //VERSION 1 of the watering system, there are 2 main watering points in the day, in the morning at
0900 and in the afternoon at 1800.
  //With the sensor data we try to approximate the amount of water the plant needs at the watering
moment.

  //For the upper level
  if (hour == morning && water == true && valHigh == 0 || hour == evening && water == true && valHigh
== 0) {
    if (upperLayer >= LOWLEVEL && upperLayer < MIDLEVEL) {
      digitalWrite(rPin1, HIGH);
      currentTimeA = millis();
      if (millis() - currentTimeA == small) {
        digitalWrite(rPin1, LOW);
        water = false;
      }
    }
    if (upperLayer >= MIDLEVEL && upperLayer < HIGHLEVEL) {
      digitalWrite(rPin1, HIGH);
      currentTimeB = millis();
      if (millis() - currentTimeB == medium) {
        digitalWrite(rPin1, LOW);
        water = false;
      }
    }
    if (upperLayer >= HIGHLEVEL && upperLayer < 101) {
      digitalWrite(rPin1, HIGH);
      currentTimeC = millis();
      if (millis() - currentTimeC == large) {
        digitalWrite(rPin1, LOW);
        water = false;
      }
    }
  }

  //For the lower level
  if (hour == morning && waterL == true && valLow == 0 || hour == evening && waterL == true &&
valLow == 0) {
    if (lowerLayer >= LOWLEVEL && lowerLayer < MIDLEVEL) {
      digitalWrite(rPin2, HIGH);
      Stime = true;
      if (Stime == true) {
        currentTimeD = millis();
        Stime = false;
      }
      currentTimeD = millis();
      if (millis() - currentTimeD == small) {
        digitalWrite(rPin2, LOW);
        waterL = false;
      }
    }
    if (lowerLayer >= MIDLEVEL && lowerLayer < HIGHLEVEL) {
      digitalWrite(rPin2, HIGH);
      currentTimeE = millis();
      if (millis() - currentTimeE == medium) {
        digitalWrite(rPin2, LOW);
        waterL = false;
      }
    }
    if (lowerLayer >= HIGHLEVEL && lowerLayer < 101) {
      digitalWrite(rPin2, HIGH);
      currentTimeF = millis();
      if (millis() - currentTimeF == large) {
        digitalWrite(rPin2, LOW);
        waterL = false;
      }
    }
  }

  if (hour == (morning - 1)  || hour == (evening - 1)) {
    water = true;

  }
}
```