Hydroponic Growing in Ikea PS Locker with Automatic Flood and Drain Self-priming Syphon

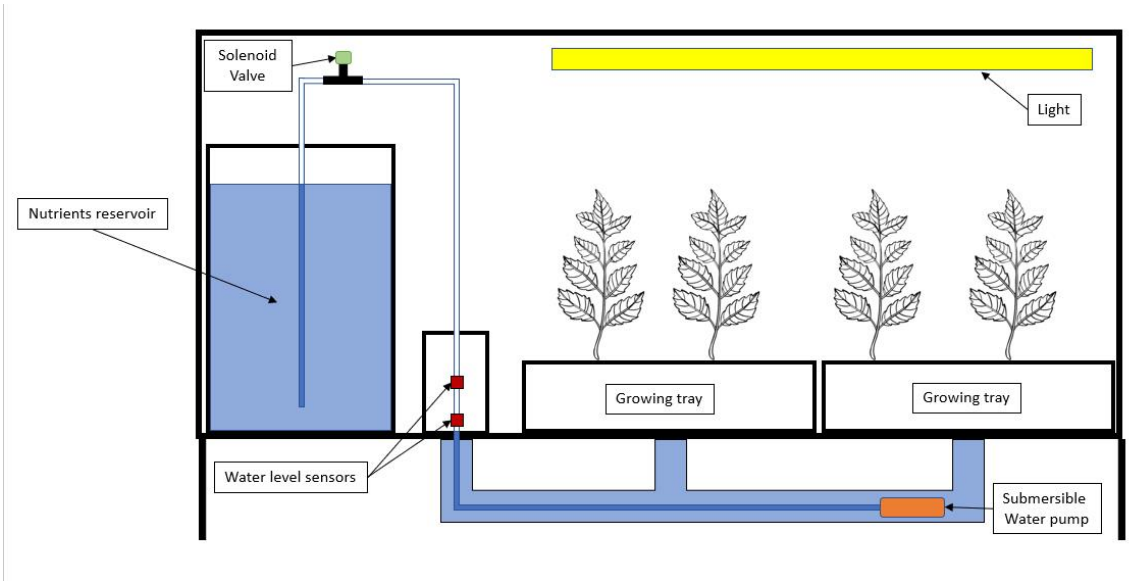
The goal of this project is to modify a standard Ikea PS locker into a device to grow leafy green vegetables hydroponically. Hydroponics is a technic to grow plants without soil. Plants grow directly in an inert substrate while nutrients are provided through a circulating water solution. There are multiple approaches to designing a hydroponic system. For this specific project, a flood and drain system has been chosen. In these systems the plants grow in a tray which is flooded and drained with a nutrient solution periodically a few times per day depending on the growing stage. When the tray is flooded, the inert substrate, in this case clay pebbles, absorbs the nutrient solution which is consumed by the plants.

In order to explain how the device works the system can be divided into 4 sub-systems.

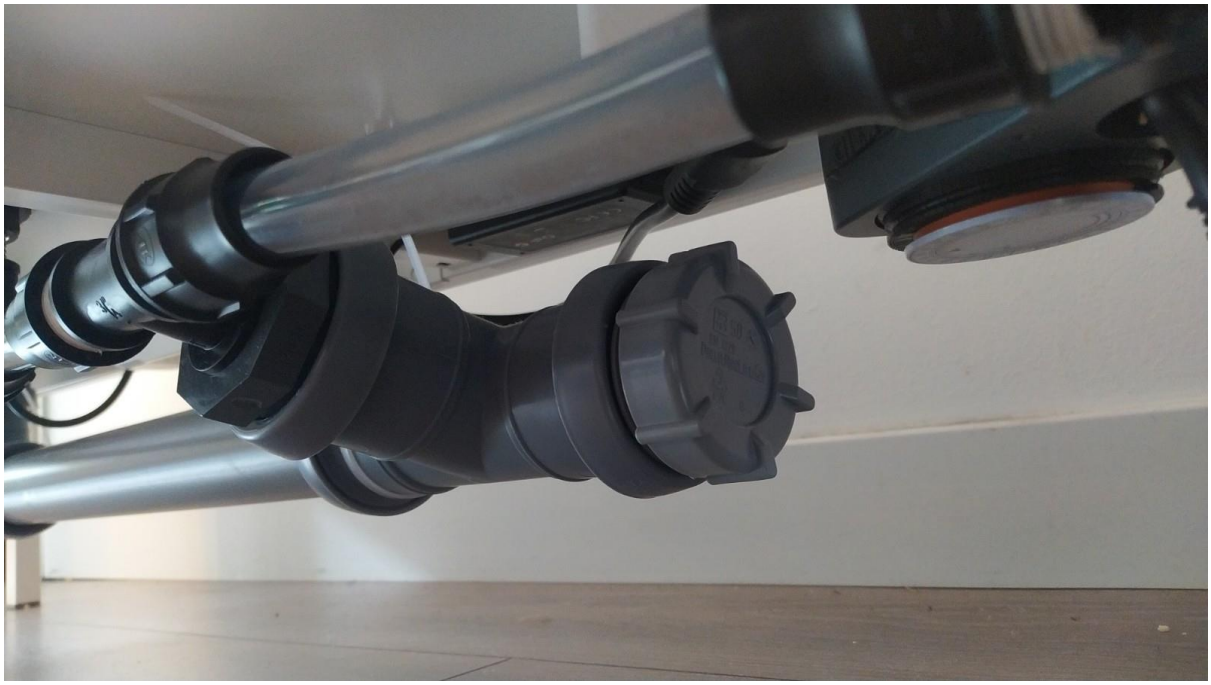
1. Flood and drain
2. Ventilation
3. Lighting
4. Control

1. Flood and drain

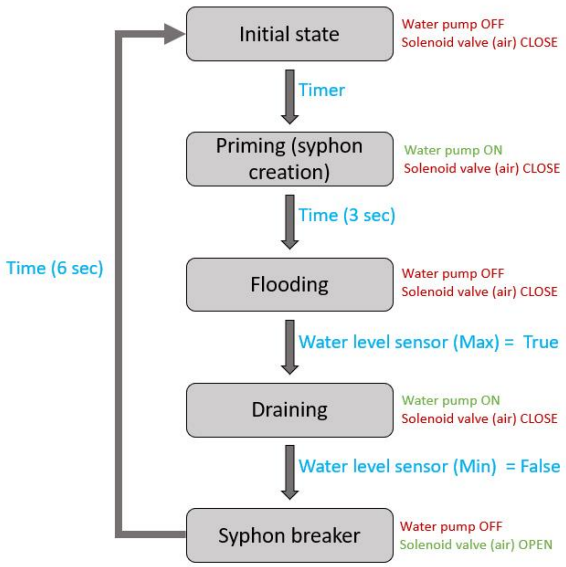
The flood and drain system compromises 2 growing trays, 1 nutrient reservoir, an additional compartment to place the water level sensors, a solenoid valve, a water pump and pipes as shown in the figure below (in the actual device the sensors are placed behind the nutrients reservoir).



Both growing trays are connected to the sensor box with a 50mm pipe below the locker. This way the water level in the sensor box is the same as the growing trays. There is a submersible water pump inside the pipe. The outlet of this pump ends at the bottom of the main reservoir. There is a solenoid valve at the top of the path introducing (or blocking) air in the pipe.



The flood and drain process has 5 different states which are explained below.



Initial state

In the initial state the system is on standby. It is waiting for a specific time interval to trigger the beginning of the flood and drain cycle. The water pump is switched off, the solenoid valve is closed, and the trays are empty. The nutrients reservoir is full as well as the pipe below. It is important to fill this pipe with the solution when the device is used for the first time and after cleaning or when the nutrient solution is renewed.

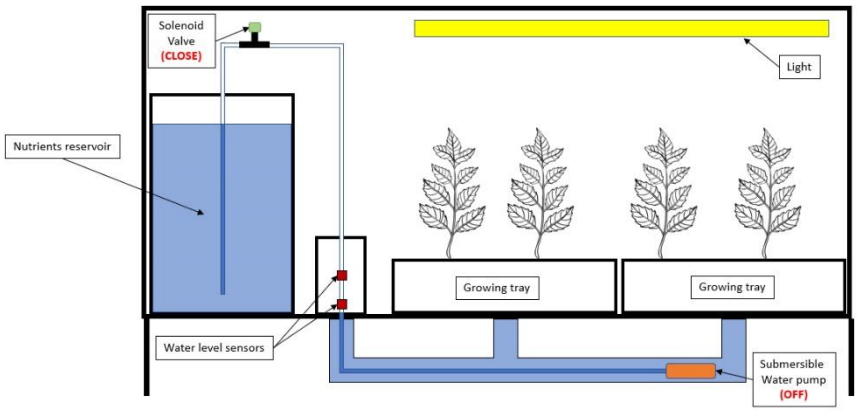


Figure 1. Initial state

Priming

The objective of this stage is to create a syphon. The water pump is activated for only 3 seconds to remove the air in the tube. When the pump is switched off the syphon is created and flooding begins.

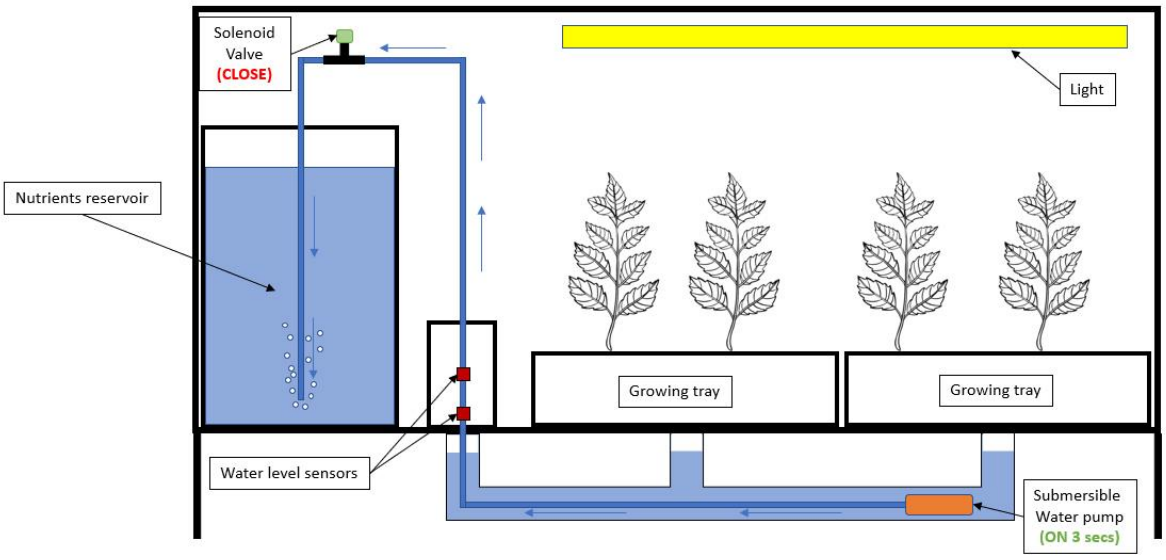


Figure 2. Priming

Flooding

Once the syphon is created the water flows from the main reservoir through the pump to the growing trays as long as the water level in the growing trays is lower than the level in the main reservoir. When the water reaches the water level sensor flooding ends and draining starts.

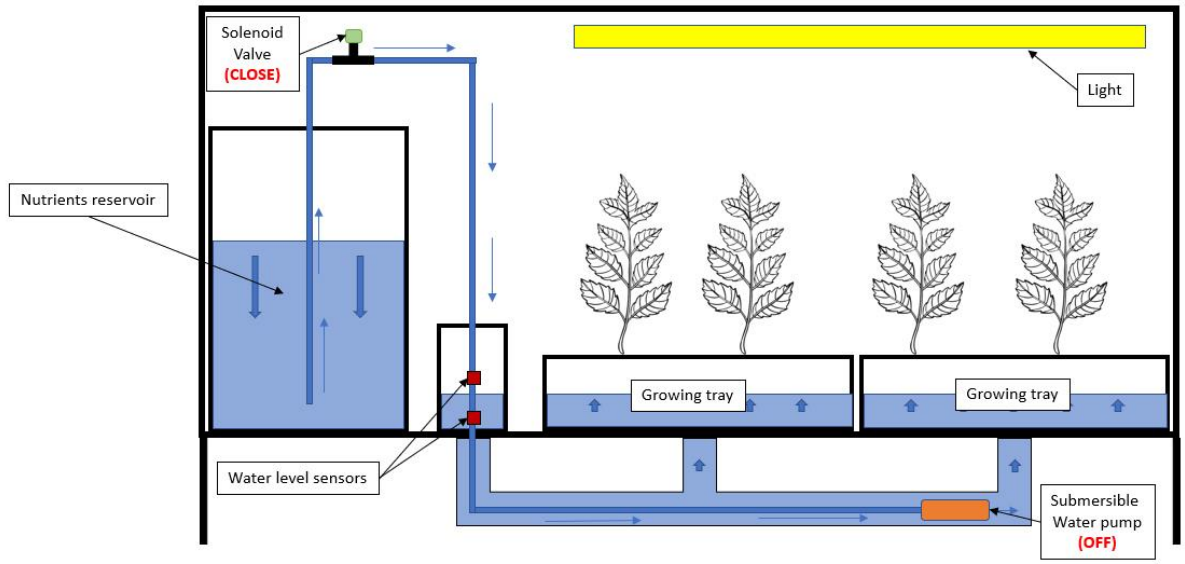


Figure 3. Flooding

Draining

To remove the water from the growing trays back to the main reservoir the water pump is turned on until the sensor at the bottom detects no water.

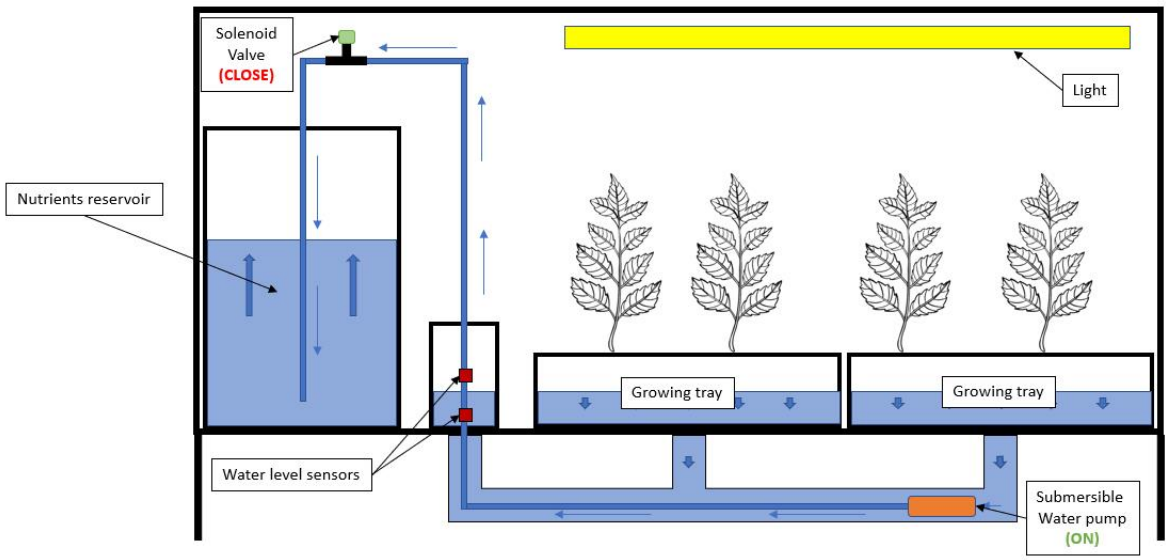


Figure 4. Draining

Syphon breaker

When the water pump is switched off at the end of the draining phase the water tends to flow back to the growing trays since the syphon is still active. To finish the flood and drain process the syphon is broken by opening the solenoid valve. Air is introduced in the tube and, therefore, the system returns to the initial state.

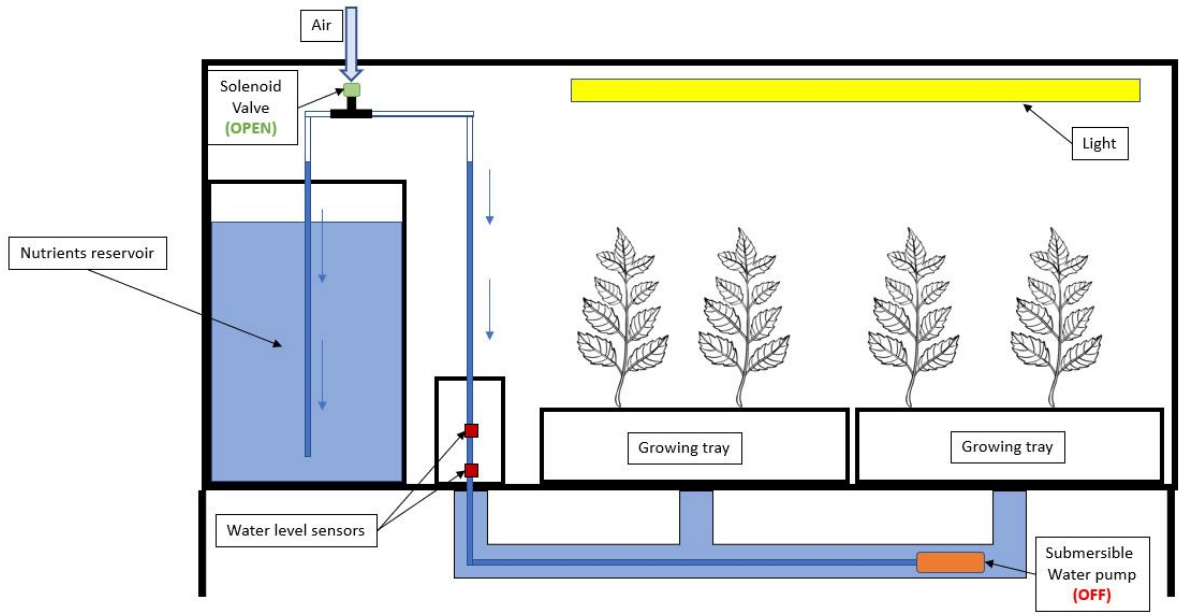


Figure 5. Syphon breaker

2. Ventilation

Air circulation is needed to remove the heat produced by the light and to replace the air inside the locker. There is a 120mm 12V computer fan installed in the back in a high position.

3. Lighting

The light is a 40W LED specially designed for growing. The main advantage of this light is its short thickness.

4 Control

Electric components and sensors are controlled by an Arduino UNO. This board basically controls the flood and drain system and the fan. The fan works only when the light is turned on. The light works independently of the rest of the system with a mechanical timer.

This is the code for Arduino:

```
//define pins
#define water_level_max 5
#define water_level_min 6
#define relay_pump 7
#define relay_solenoid 8
#define lightSensor 10
#define relay_fan 9

int priming_time = 3000;  // Priming time
int hours_interval = 5; //hours between flood_drain cycle
const unsigned long interval_flood_drain = 1000 * 60 * 60 * hours_interval;
//flood-drain interval
unsigned long flood_drain_Millis = 0;



void setup() {
  pinMode(water_level_max, INPUT);
  pinMode(water_level_min, INPUT);
  pinMode(lightSensor, INPUT);
  pinMode(relay_pump, OUTPUT);
  pinMode(relay_fan, OUTPUT);
  pinMode(relay_solenoid, OUTPUT);

  digitalWrite(relay_pump, HIGH);
  digitalWrite(relay_solenoid, HIGH);
  digitalWrite(relay_fan, HIGH);

}

void loop()
{

  unsigned long currentMillis = millis();
  int lightSensorReading = digitalRead(lightSensor);
  if (currentMillis - flood_drain_Millis >= interval_flood_drain)
  {
    flood_drain_Millis = currentMillis;
    if (lightSensorReading == 0) //if light is on
    {
    flood_drain();
    }
    else
    {
      fanControl();
    }
  }
  else
  {
    fanControl();
  }
}

void flood_drain()
{
    unsigned long delay_drain = 40000; //40 seconds

    //PRIMING
    digitalWrite(relay_pump, LOW); //pump ON
```

```
  delay(priming_time);
  digitalWrite(relay_pump, HIGH); //pump OFF

  //FLOODING
  int Liquid_level_max = digitalRead(water_level_max);

  while (Liquid_level_max == 0)
  {
    Liquid_level_max = digitalRead(water_level_max);
  }

  //DRAINING
  int Liquid_level_min = digitalRead(water_level_min);

  while (Liquid_level_min == 1)
  {
    digitalWrite(relay_pump, LOW); //pump ON
    Liquid_level_min = digitalRead(water_level_min);
  }

  delay(delay_drain);
  digitalWrite(relay_pump, HIGH);
  delay(1000);

  //SIPHON BRAKER
  digitalWrite(relay_solenoid, LOW);
  delay(13000); //valve open for 13 seconds
  digitalWrite(relay_solenoid, HIGH);

  return;
  }


void fanControl()
{
  int lightSensorReading = digitalRead(lightSensor);
  if (lightSensorReading == 0) //light is ON
  {
   digitalWrite(relay_fan, LOW); //switch ON fan
   return;
  }
  else
  {
   digitalWrite(relay_fan, HIGH); //switch OFF fan
   return;
  }

}
```