# Introduction

IoTomatoes is an automated system and his goal is to take a care about growing tomatoes. System consists of a web application, client scripts on Raspberry Pi and hardware. The hardware consists of a box with plants, an air humidity sensor, soil humidity sensors, brightness sensor and temperature sensor. Also, there is a cooling/heating system that consists of the ventilation tubes and fan. For heating and cooling there is a Peltier. There is a tire on the plants that serves to irrigate the plants. irrigation comes from canisters with water (and artificial fertilizer) using a water pump. Based on the obtained values, the system decides what action should be taken to create ideal conditions for tomato growing. The whole logic is developed on the client (Raspberry Pi 3), and the rules decide when to execute the specific action. Rules are written in Json Logic, which is defined in the web application. Json Logic allows a generic approach defining rules, so there isn’t a limit to the complexity of rules applied on a plant. This enables us to make custom rules for a variety of plants.

# Components

## Raspberry Pi 3

Raspberry Pi 3 is microcontroller based on Linux operation system (Raspbian). This microcontroller is a central place for all sensor values. With RPi is possible to manage all sensors. All logic is developed in Python programming language.

## Sensors

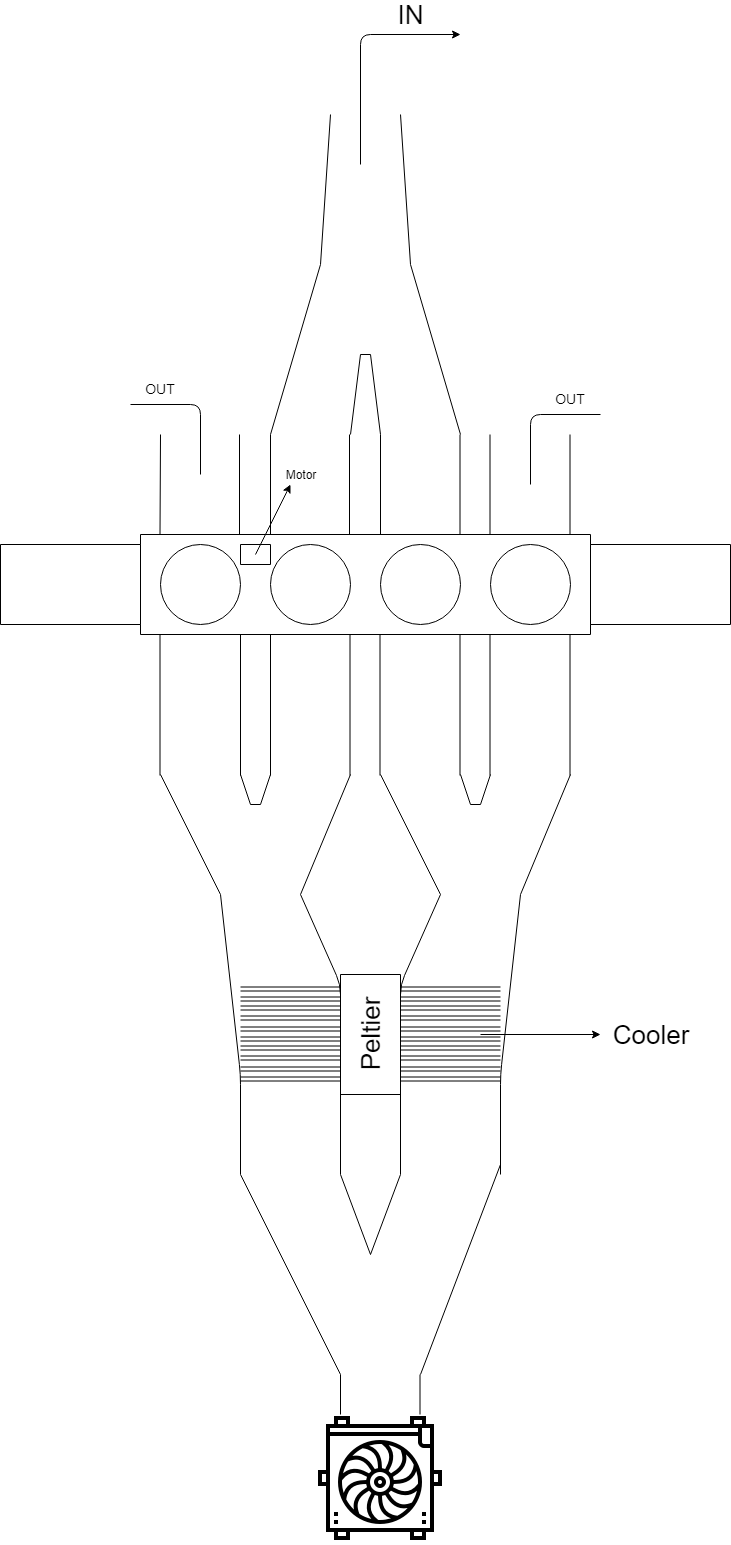
Used sensors are Soil Humidity sensors, Temperature sensor Air Humidity senor and brightness sensor. Soil Humidity sensor is sensor which gives us an information about soil humidity. This value is a key value to decide when to start water. So, gives us an information when it is humidity is too low in the soil. Air Humidity and Temperature sensor give us an information about air and is there a need for a heating or cooling. Based on that information, system will start heating or cooling. For light, we used a lamp, which simulated daily light.

## Water Pumps

Water pumps are also connected to the Raspberry Pi and they are located within canister, on the water. When the value of Soil Humidity sensor is low, Json Logic will return the “true” for watering and pumps and water will start.

## Peltier and Cooling/Hearing system

Peliter is a small device which cools or heats on the particular voltage. It looks like a small tile. One side of this tile is cooling and other side is heating at the same time. On both sides are bonded coolers (something like small radiators) through which heat/cold passes. System for heating/cooling is made by ventilation tube. One edge of the tube consists a fan and on the other edge comes the warm/cold air. On the middle of system is a plastic panel which contains four holes. The tube passes through each hole. Above the panel, there is a Peliter with coolers. Figure 1 shows the system. The motor drives the lower part of the panel, which blocks or leaks air, and based on the value of the sensor, decides whether to heat or cool. On this basis, the engine is driven and passes through the first and third or second and fourth holes, where through second hole passes warm air, and through third cold air. For example, if the bottom panel leaks air to the second and fourth holes, then the warm air passes up and enters the box, and the cold through the fourth hole comes out. If the air passes through the first and third holes, it would mean that the warm air through first hole comes out, and cold air comes through the third hole and enters the box.



Slika 1 Cooling/Heating system

# Communication

For communication is used REST. Microcontroller communicates with sensors with digital signal and then the data is sent to the server through HTTP protocol. Also, Raspberry Pi use REST services for GET call to retrieve the ruleset from the server (ruleset for specific farm, because can be many farms). This “ruleset communication” is running in specific thread as a background task and updated ruleset is retrieved every 15 minutes. Server saves the data into the database and refresh the frontend in that moment when the data came.

# Software (backend & frontend)

Backend (server side) is developed in Microsoft .NET Core (C#). There are services on which Raspberry Pi and frontend application are connected. Frontend application is developed in VueJS Javascript framework. Clean architecture was used.

# Client

Raspberry Pi is a client of this system and consists a few different scripts. Main script is a script which connect all classes (classes contains information about sensors and methods to manipulate with these sensors) and script which communicates with server and decide about all actions which have to be performed. Raspberry Pi and all hardware (sensors, lamp, ventilator…) is powered by computer power supply. Some sensors (like Soil Humidity sensor) send only analog signals, so there was a need for conversion from analog to digital (AD converter). Reason is that because Raspberry Pi have only digital inputs.

# Improvements

System is generic but there is some things which can be improved. First thing is cooling/heating system. This system is made very good, but motor is a too weak. All construction is very sensitive and this can be optimized as well. Irrigation tires should be above the plants and water should be scattered on plants.