IoT based Horticulture Monitoring System

Monika Rabka
School *of* Computing and Digital Media, Communications Technology, London Metropolitan University

Introduction

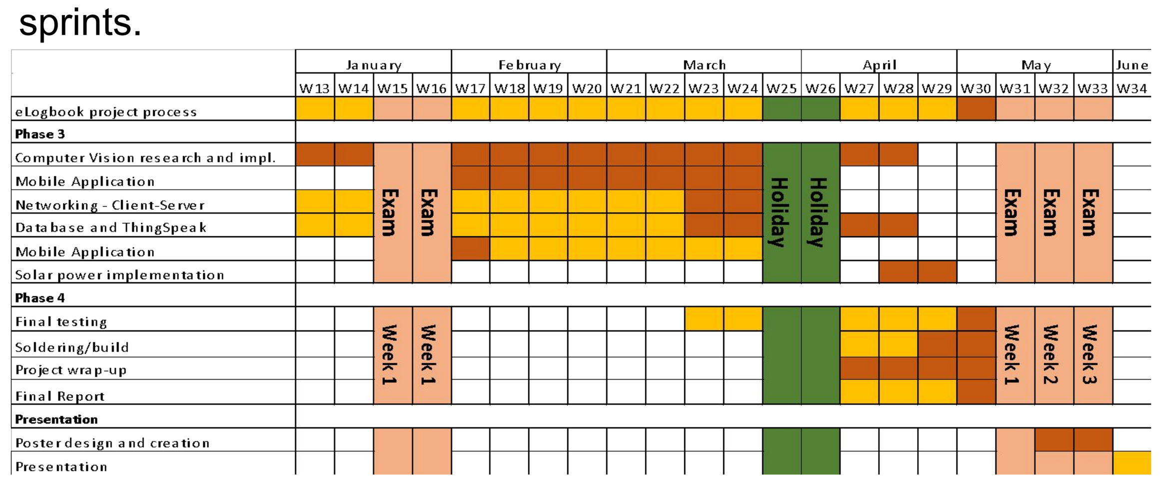
This project's topic is the design and implementation of an Internet of Things (IoT) based embedded system called Real-time Autonomous Horticulture Monitoring System (RAHMS), which monitors various aspects of controlled-environment agriculture (CEA). The RAHMS uses a mobile application for viewing the greenhouse crop data and camera feed of plants, and cloud databases like Firebase and MATLAB ThingSpeak to access and display the sensor data. This project aimed to develop a low-powered and inexpensive monitoring system for small to medium, remote greenhouses that can access the Internet, either in the form of broadband, satellite, or 3G/4G access points. RAHMS project covers various computer systems engineering areas. It covers computer science, software engineering, electrical and electronic engineering, and computer network engineering.

Aims, Objectives and Project Management

This project aimed to design and implement a self-sustained horticultural real-time monitoring system that uses hardware, networking, and software to gather sensor data, process it, potentially use Machine learning, and display it on a mobile application. Some of the objectives were:

- To investigate various sensors that measure the natural world conditions and utilize hardware skills and knowledge of interfacing embedded systems,
- To utilize software and programming skills in developing the database and mobile application,
- To investigate and implement wireless communication between the distributed embedded systems
- To study cloud architecture for scalability of the system.

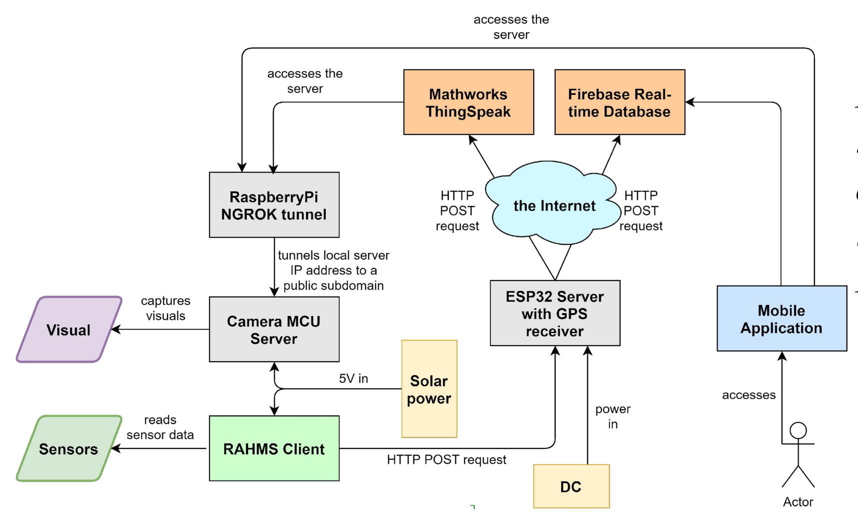
It was important to follow the standard engineering design process loop throughout the project which starts with identifying and researching a need □ developing possible solutions □ making a prototype □ testing and evaluating □ and modifying and retesting the solution. RAHMS followed the industry-standard Agile and Waterfall project management techniques to fully optimise the allocated time by dividing the project into four distinct phases, within which each topic area underwent at least two sprints.



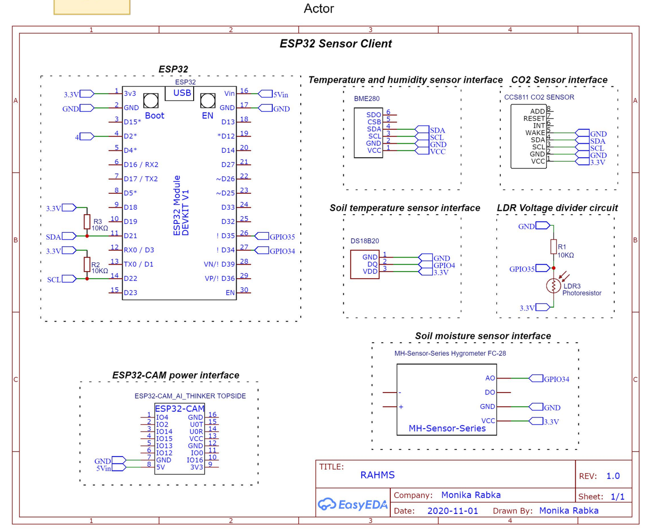
Second Gantt chart of the Project Plan with changes to original in brown.

Method

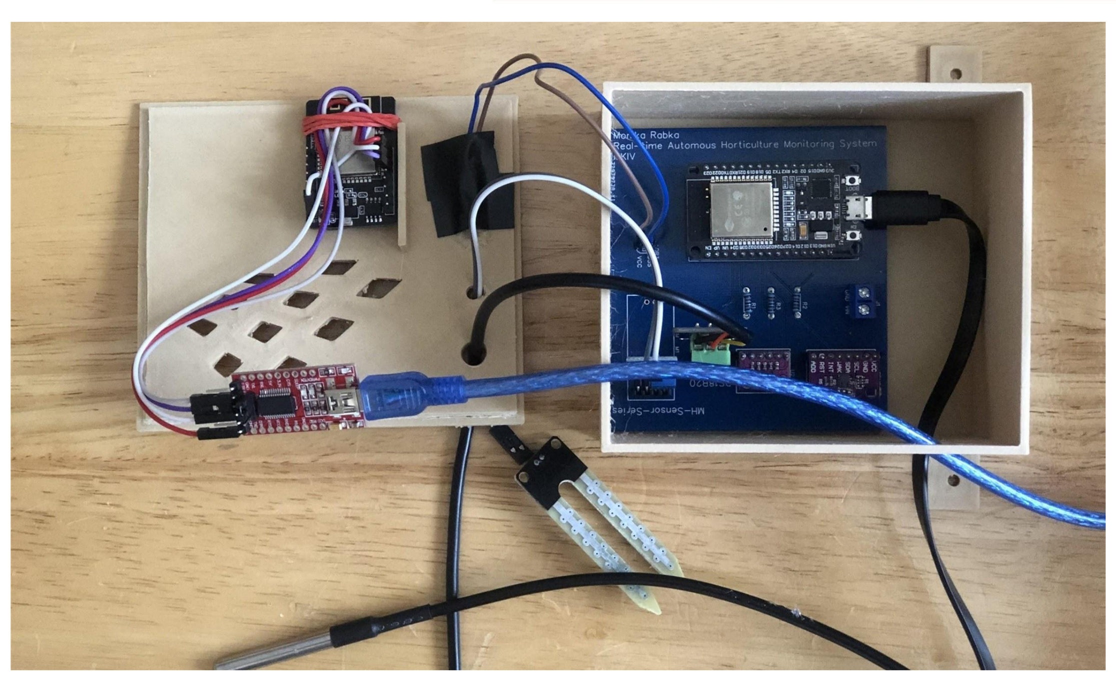
The design of the RAHMS is based on a client-server networking architecture model. The sensor ESP32 board is the client which collects and sends the data to an ESP32 server, then which transfers it over the Internet to cloud databases from where this data is then processed. The camera microcontroller is a standalone server from where the photos can be accessed from. The RAHMS Client and ESP32-CAM module are powered together with a solar power bank and the server is powered separately with 5V. The sensor data received by the server is sent to ThingSpeak to be visualised on graphs and to Firebase. Firebase is linked to the Android app where the data is also displayed along with the photo from the ESP32-CAM module. Tunnelling service NGROK is used to expose the local server on the ESP32-CAM module to the public network for the photos to be accessed by ThingSpeak and Android application.



Detailed block diagram of the RAHMS



Circuit diagram of the sensor board circuit with the ESP32-CAM module for the RAHMS Client

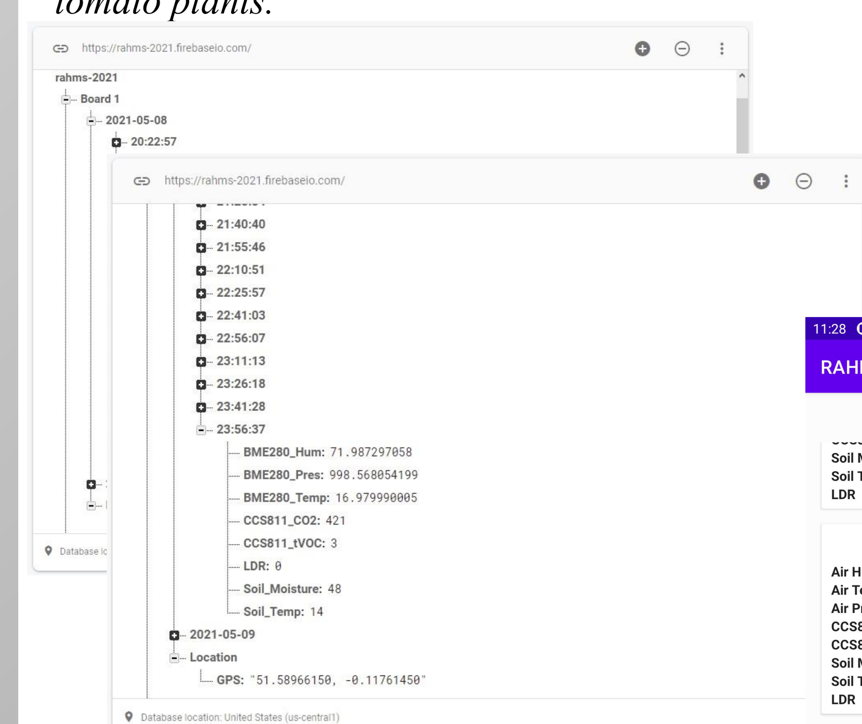


RAHMS Client manufactured PCB with the ESP32-CAM module inside the 3D printed box.

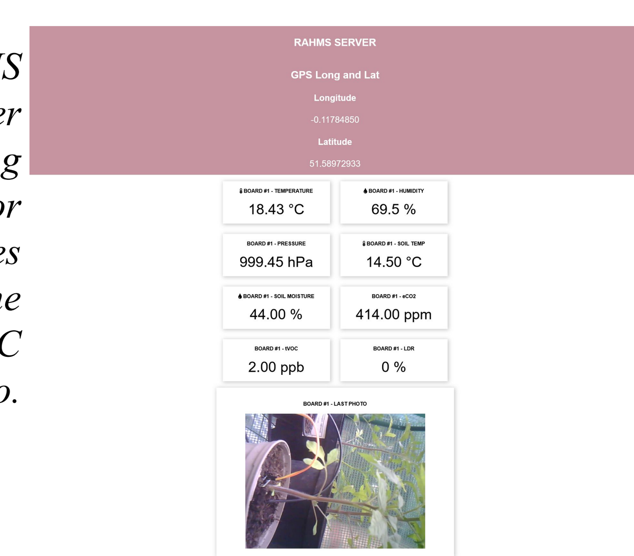
Results

In the final test, the RAHMS Client was powered with a solar power bank and put inside a greenhouse for 24 hours. The results were positive as the system and all its subsystems worked as intended.

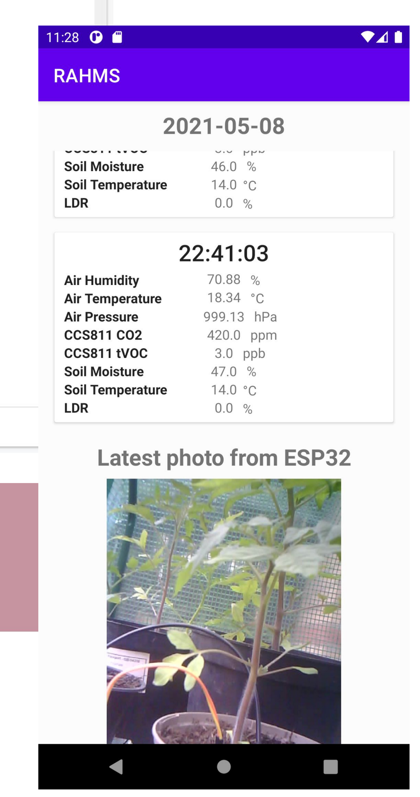
RAHMS Client enclosed in the 3D printed case in the greenhouse with tomato plants.



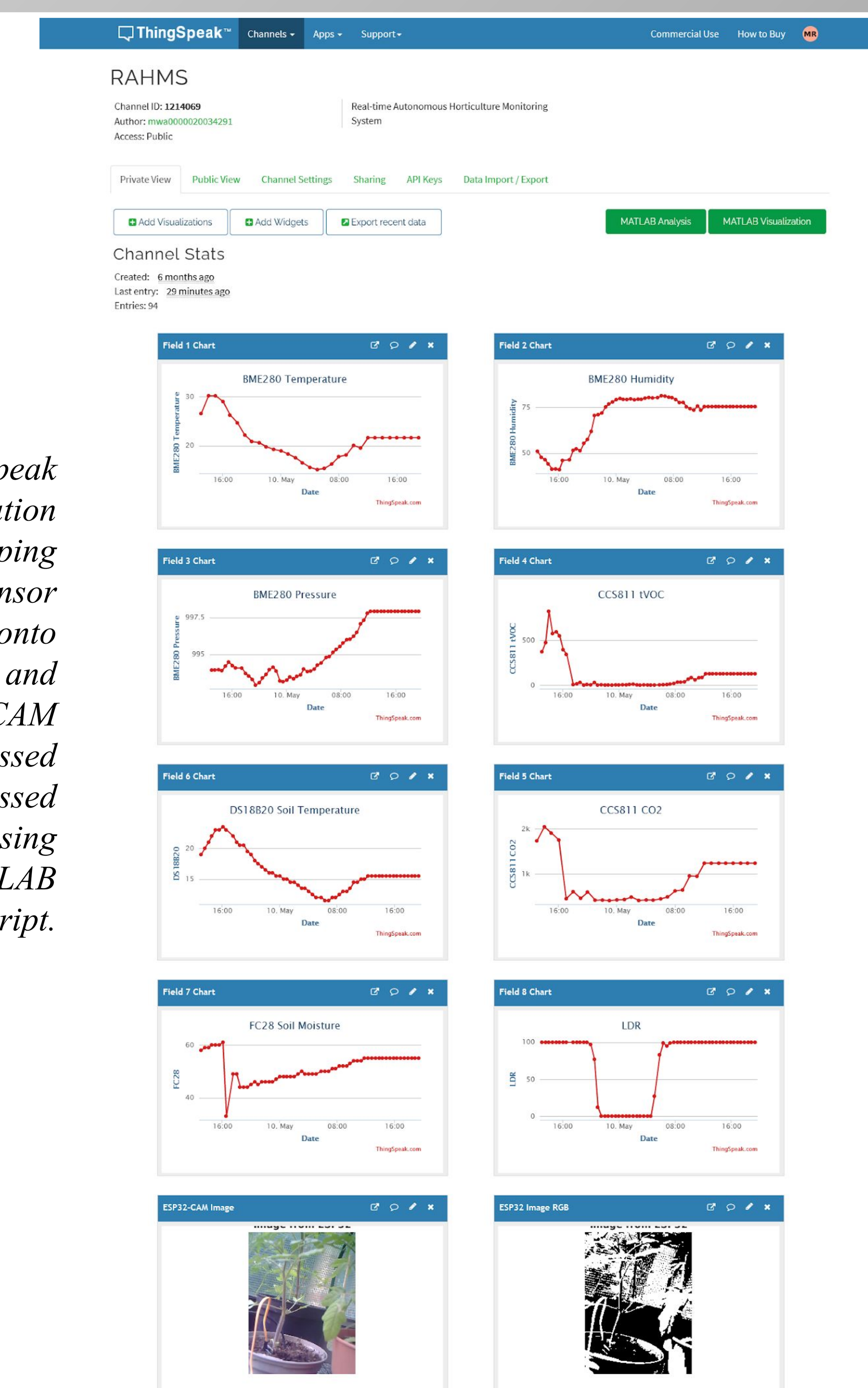
Firebase node structure receiving the date/time named JSON objects.



RAHMS Server displaying the sensor values and the ESP32-CAM photo.



RAHMS Android application during the final test of the project in the greenhouse.



ThingSpeak visualisation fields mapping the sensor values onto charts and ESP32-CAM unprocessed and processed photos using MATLAB script.

Conclusion & Future Work

The final product have met the aims and objectives and the end product deliverables coincide with the deliverables set out at the beginning of the project. Due to the diversity of the project, including software development and engineering, cloud and databases, Computer Vision and Image Processing, the experience and transferrable skills gained will be helpful for any future work outside of the university.

For future work, LTE sim MCU, Over-the-air (OTA), security implementation, actuators for greenhouse watering and fans, improvements to the app, and a wider sensor range could be implemented to make the system better.

Acknowledgements

I would like to thank my supervisor, Dr Shukla, for all the support and guidance given for this project and other tutors throughout my degree who helped me understand even the most challenging topics.