LCCS 2022 Computer Science Write Up

1) Investigation and Plan (Approximately 600 words)

Evidence of research on brief

I decided to create a smart home system. I researched my idea online. I read Smart Home's website and watched their YouTube videos.

I found that a smart home uses devices connected via the internet to allow homeowners remote monitoring and/or control of appliances and systems with an app or on a web page.

These types of automation provide added security, convenience, comfort and energy efficiency.

- They remotely control lighting systems turning lights on/off from another room.
- They schedule and monitor thermostats, by setting the thermostat on a timer thus energy efficiency is increased.
- They control access through smart locks and monitor security cameras increasing home security.

Examples of artificial-intelligence-powered smart home assistants are Amazon's Alexa or Google Home. However, they're costly and raise privacy issues as someone could hack the system.

I researched the following solutions – using the internet and a micro:bit IoT kit.

Existing Solutions:

1. Ring video doorbell hardwired - €55

Features: Live HD video, Motion Detection, Night Vision, 2-way audio and Motion Detection.

This allows remote viewing of your front door from your smartphone, this benefits security for residents when not at home.

2. Google Nest Thermostat 3rd Generation - €260

Features: provides remote temperature control using your smartphone and an energy consumption report. Heat switches on when you walk in the room and turns off when you've left, thus reducing energy use.

Possible Solutions:

3. A Plant Watering System controls soil moisture levels.

A pump submerged into a water supply pipes water into the pot, a sensor in the plant pot and an 'if statement' determines if the moisture level read in requires water to be pumped to the plant.

This system provides the optimum amount of water daily, solving the problem of forgetting to water the plants and will optimise plant growth.

4. Voice Activated Lighting switches lights on whenever noise in a room reaches a predefined level.

A noise sensor and lights connected to a micro:bit with an 'if statement' turns the lights on when the required noise level is reached.

This eliminates the need to search a dark, empty room for a light switch, as an increase in noise will turn the lights on.

My system

My room monitoring system's objective is to monitor the room's temperature, dust level, door status and any flame. This information is displayed on a website for the homeowners and allows them to know the status of every room. This provides homeowners with affordable home protection, prompting them to close their doors and reducing home insurance costs.

My system will be useful and marketable to all people, accommodating a wide range of abilities regardless of the user's body size, posture, or mobility. It's easily understood, comfortable to use, tolerant to errors, and necessary information is communicated effectively.

Stakeholders are device manufacturers, service providers - sell product/service, platform providers of software and end-users.

Target end users are people looking to reduce their house insurance costs whilst protecting their homes from environmental damage caused by fire and smoke.

Social implications: Increased home security, monitors dust levels which improves health, financial - reduced insurance costs.

Deeper Research: There is a viable market for my solution. Many insurance companies discount 5-20% for security systems and higher discounts for environmental monitoring, detecting fire, smoke and water damage.

Technologies to be used are LEDs, OLED display, batteries, embedded system, flame, dust and temperature sensors, database, programming and scripting languages. Further research found that embedded systems work well in smart homes. The internet sends data to a program for cleaning and sorting, then to a cloud-based database such as firebase. The website fetches the data and graphs it using JavaScript and my unique firebase credentials.

2) Design (Approximately 600 words)

Outline of project

This system aims to protect homes by prompting owners to close their doors thus reducing home insurance.

The design includes a Micro:bit which when triggered by timed events reads in data, turning on/off LEDs whether the door is opened/closed. Data is sent to Python where it is cleaned, sorted and stored in Firebase. The company website displays this data for the homeowners. It provides information on Smart Homes and a validation checked email subscription.

In reality, the timed events create data at one transaction per minute. This collects a reasonable amount of information, whilst being cost-effective for a company, as it costs money to store data.

For the project, my system would create one transaction every three seconds because:

- I want to see the data flowing through the system
- It's the fastest the micro:bit can transfer data without error
- I don't want to fill up the table with too much data

System Requirements

Software

- MakeCode editor for writing code for the micro:bit
- Python to clean and send data
- Firebase to store data
- HTML, CSS and JavaScript to create a company website and graph data

Hardware

Parts list: micro:bit; micro:bit IoT kit; OLED Display; temperature, dust, and flame sensors; two LEDs; battery pack; USB cables.

Reasoning:

- Python as it is easy to use and is a powerful language for cleaning data
- Firebase is cost-effective, cloud-based, real-time database
- Micro:bit is easy to learn and use allowing me to focus on other modules, including more advanced features like firebase. Although devices like the raspberry-pi are more powerful and internet-enabled.

Plan and Approach

I used an agile plan for my project with iteration used upon problems.

A modular approach was used to decompose the project into 4 independent modules along the data plane (see architecture diagram):

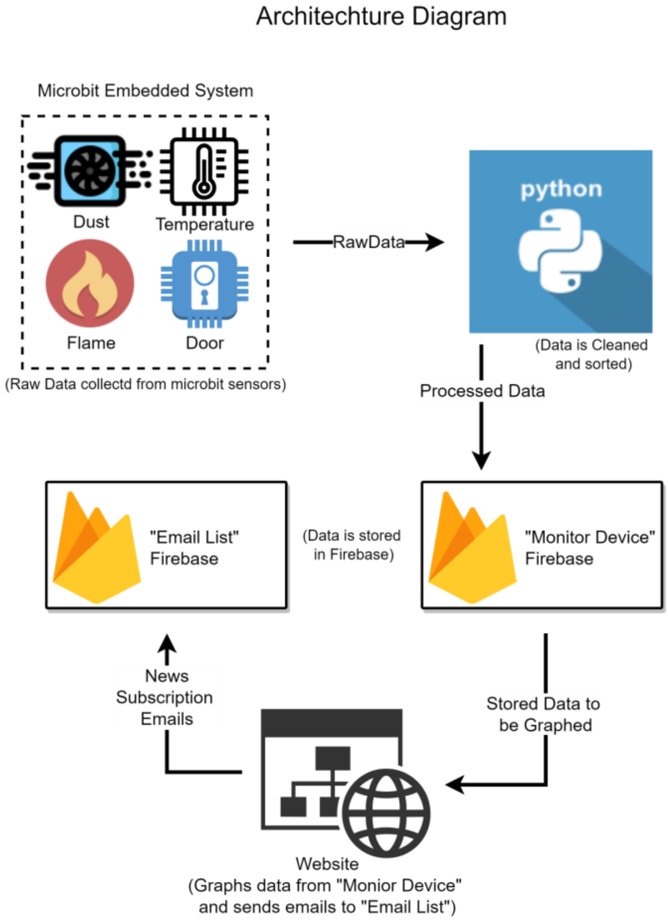
- embedded system capturing data
- python program and database
- website receiving and graphing data
- website displaying information and storing emails

I addressed each module one at a time. I decomposed each module into a set of smaller tasks. I took an agile approach to completing the tasks because this allowed me to iterate on the designs and be more flexible.

How it meets the brief

- I created a home automation embedded system with timed and triggered events capturing data to be sorted, stored and graphed (see Architecture Diagram).
- I then created a website with a home page, product page and five others, containing a user-friendly navigation system, a clean, crisp, consistent design with well-formatted, informative and concise information (see Webpage Wireframe).
- Increasing/decreasing the navigation bar size has improved user accessibility.
- An email subscription and contact form capture data, with implemented validation checks and GDPR compliance (see Architecture Diagram).
- The website was created using a combination of headings, paragraphs, lists, tables, links, images, videos, text areas, buttons, drop-down lists, calendars and similar.
- Data collected will be relevant, used for its intended purpose, securely stored in firebase and won't be passed on as per GDPR

System Architecture/Data flow Diagram


Architechture Diagram
Microbit Embedded System
Dust
Temperature
Flame
Door
(Raw Data collectd from microbit sensors)
RawData
python
(Data is Cleaned and sorted)
Processed Data
"Email List" Firebase
(Data is stored in Firebase)
"Monitor Device" Firebase
News Subscription Emails
Stored Data to be Graphed
Website
(Graphs data from "Monior Device" and sends emails to "Email List")

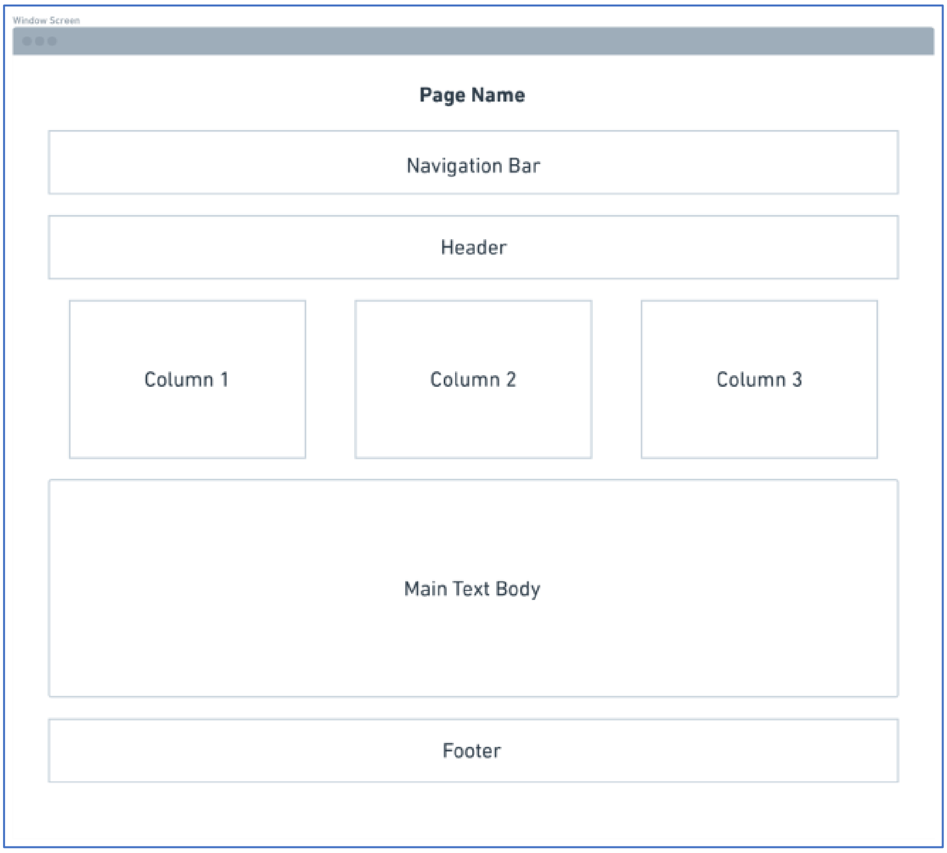
Step 1: Every 3 seconds the Micro:bit Embedded System collects raw data from the sensors. It sends this data to Python via serial port.

Step 2: The Python programme receives, cleans and sorts the data before sending it in processed form to Firebase to be stored.

Step 3: The website graphs the stored data from the 'monitorDevice' section.

Step 4: The website captures data and stores it in the 'emailList' section.

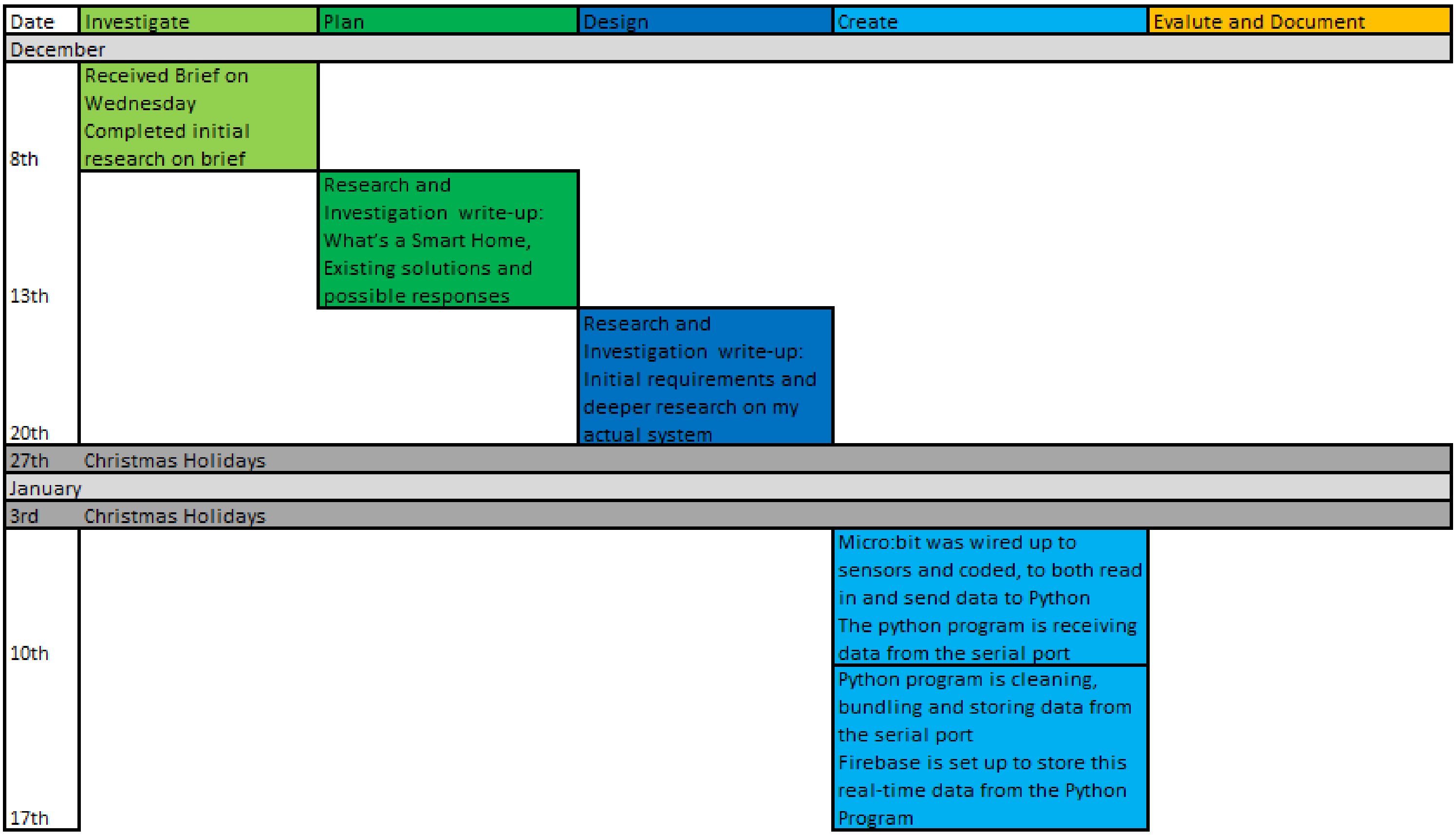
Website Wireframe



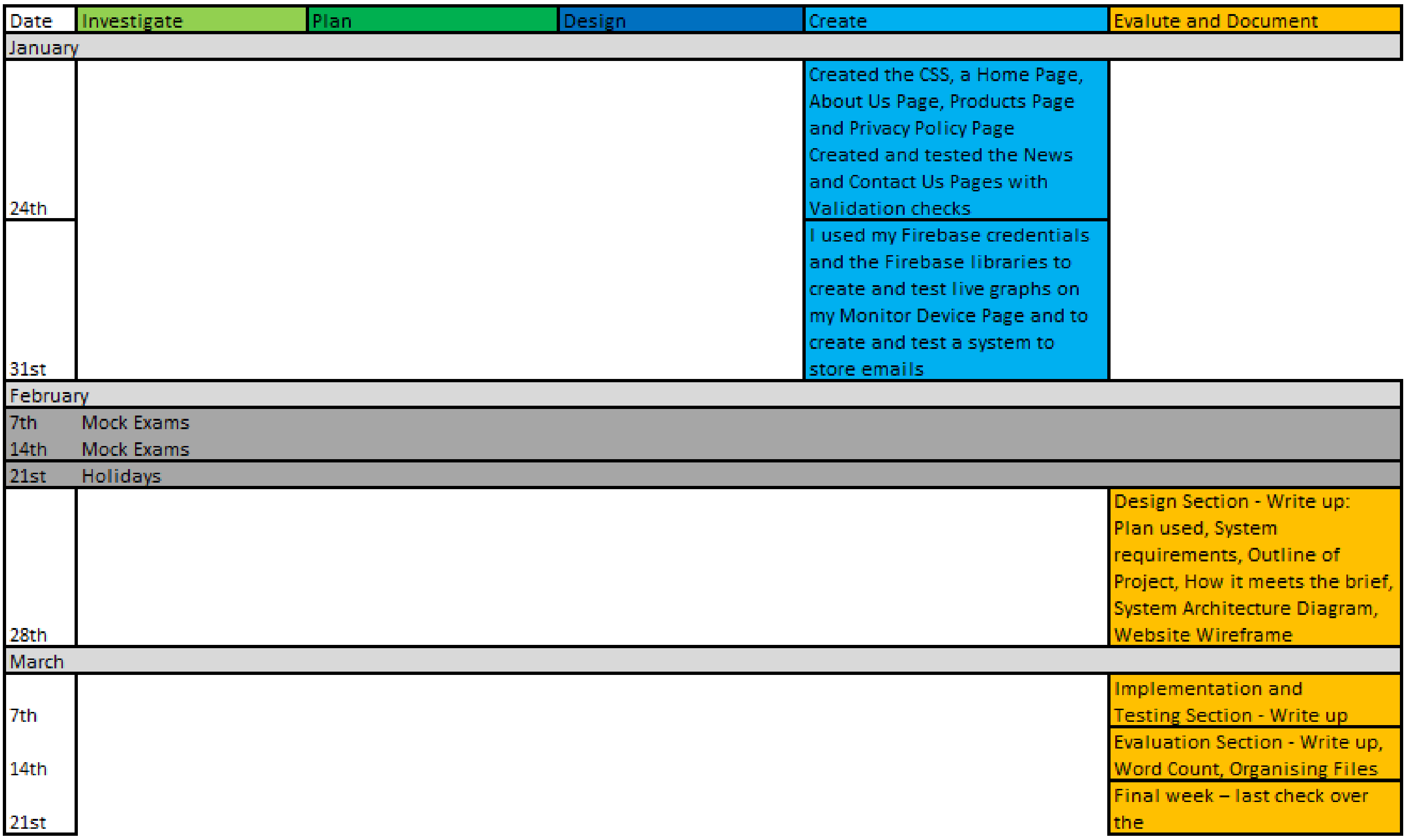
This website wireframe illustrates my webpages design. It creates a user-friendly interface for each page of the website allowing for easy navigation and a clean, crisp and well-formatted display of the information.

3) Implementation and Testing (Approximately 1000 words)

Project Timeline



Date	Investigate	Plan	Design	Create	Evalute and Document
December					
8th	Received Brief on Wednesday Completed initial research on brief				
13th		Research and Investigation write-up: What's a Smart Home, Existing solutions and possible responses			
20th			Research and Investigation write-up: Initial requirements and deeper research on my actual system		
27th	Christmas Holidays				
January					
3rd	Christmas Holidays				
10th				Micro:bit was wired up to sensors and coded, to both read in and send data to Python The python program is receiving data from the serial port	
17th				Python program is cleaning, bundling and storing data from the serial port Firebase is set up to store this real-time data from the Python Program	



Date	Investigate	Plan	Design	Create	Evalute and Document
January					
24th				Created the CSS, a Home Page, About Us Page, Products Page and Privacy Policy Page Created and tested the News and Contact Us Pages with Validation checks	
31st				I used my Firebase credentials and the Firebase libraries to create and test live graphs on my Monitor Device Page and to create and test a system to store emails	
February					
7th	Mock Exams				
14th	Mock Exams				
21st	Holidays				
28th					Design Section - Write up: Plan used, System requirements, Outline of Project, How it meets the brief, System Architecture Diagram, Website Wireframe
March					
7th					Implementation and Testing Section - Write up
14th					Evaluation Section - Write up, Word Count, Organising Files
21st					Final week – last check over the

Challenges

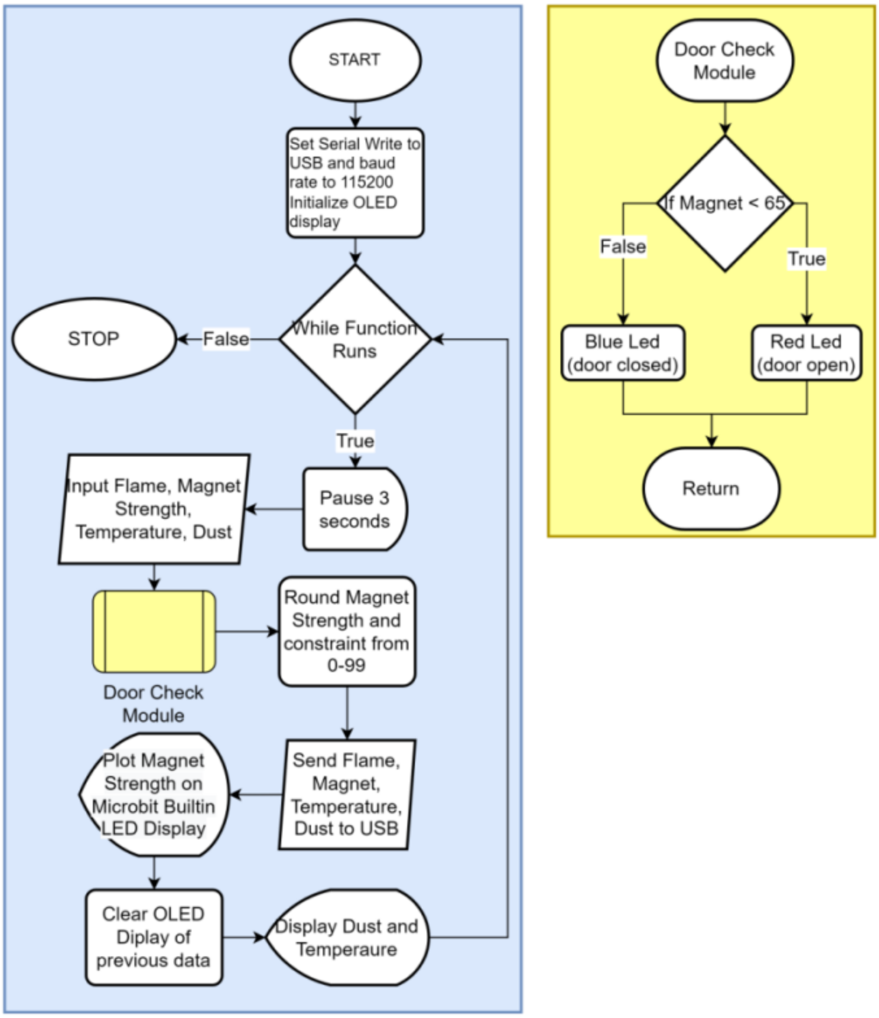
- Mock Exams created a challenge as I couldn't get a significant amount of work done during these two weeks. However, I used this time to reflect on my project which helped me during my Evaluation Section of the write-up.
- Working on other school projects at the same time as this project was challenging as I needed to divide my time up appropriately to balance out the workload of each project

Embedded System

My room monitoring system uses a Micro:bit to record metrics about the room and send them to a Python program to be processed.

To begin, I familiarised myself with the Micro:bit and the MakeCode editor. I tested the Micro:bit and each sensor to make sure they functioned correctly. I started by writing up the code to read in the room temperature from the temperature sensor.

To test my code, I displayed the metrics onto the OLED screen. Once that sensor was functioning, I followed the same steps to implement the flame, dust, and magnetic field sensors.


START
Set Serial Write to USB and baud rate to 115200 Initialize OLED display
While Function Runs
False
STOP
True
Pause 3 seconds
Input Flame, Magnet Strength, Temperature, Dust
Door Check Module
Round Magnet Strength and constraint from 0-99
Send Flame, Magnet, Temperature, Dust to USB
Plot Magnet Strength on Microbit Builtin LED Display
Clear OLED Diplay of previous data
Display Dust and Temperaure
Door Check Module
If Magnet < 65
False
True
Blue Led (door closed)
Red Led (door open)
Return

(Flowchart of the Micro:bit control plane on the right)

Step 1: The Micro:bit sets up a serial connection with the computer running the Python program and it initialises the OLED display.

Step 2: The programme performs the following every three seconds:

- Reads in dust, temperature, door status and presence of a flame
- Displays temperature and dust values on the OLED screen
- Sends the four values to the Python program through the serial connection

Python Program

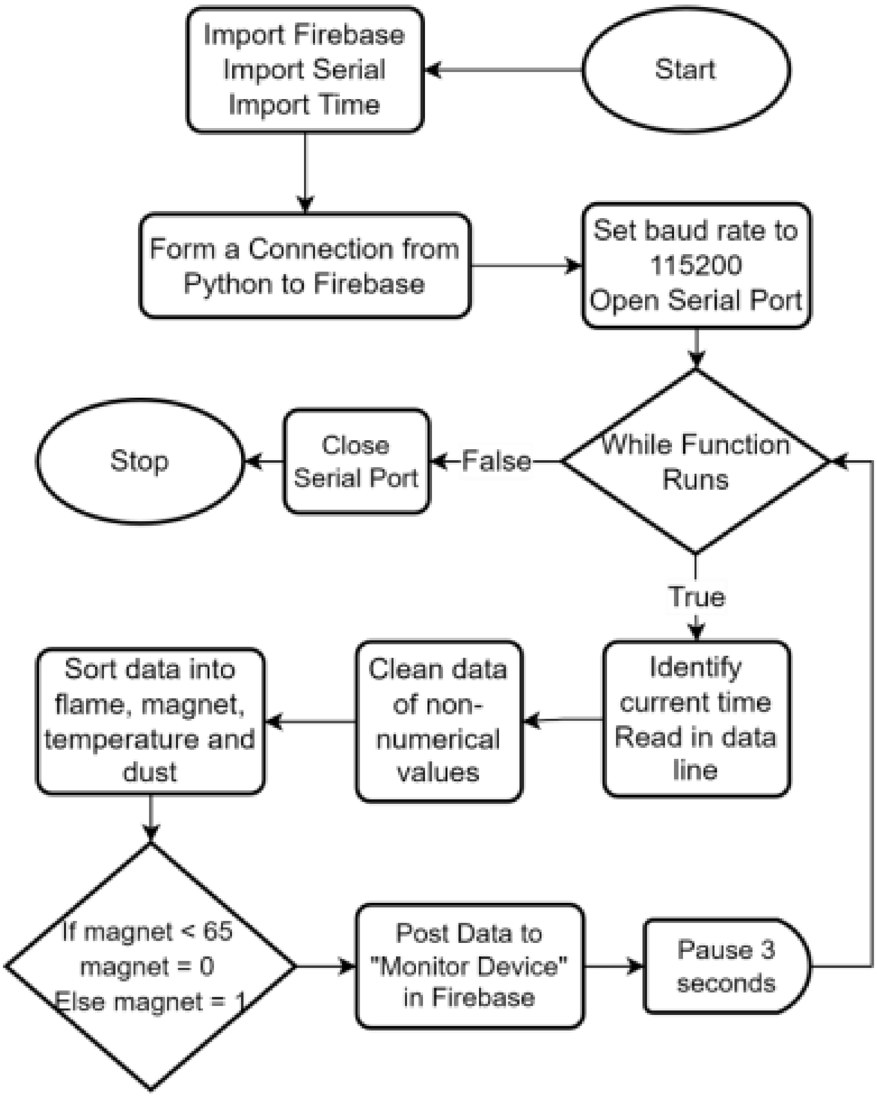
Data from the Micro:bit is sent to this Python program, where it is cleaned and bundled up into timestamped records for storage in Firebase.

I started off by setting up my IDE (Thonny) and python environment by installing the Firebase libraries onto Thonny. I experimented with reading in data from the Micro:bit by setting up the Micro:bit to send data through the Serial Port and for Python to receive this data through the serial port. When I could read in raw data reliably, I started to clean the data. This was done by printing the raw data into the console and cleaning the data line by line.

Once the data was being cleaned consistently, I created a Firebase account using my google account. I selected the closest region to Ireland for lower latency and I loaded the Firebase credentials into Python and setup the Firebase client.

I organised the data into a record and then posted it to Firebase. To test, I first sent mock data and checked the Firebase console to see if it had arrived. Then I used real Micro:bit data and check the console for its arrival.

(Flowchart of the data-cleaning program control plane below)



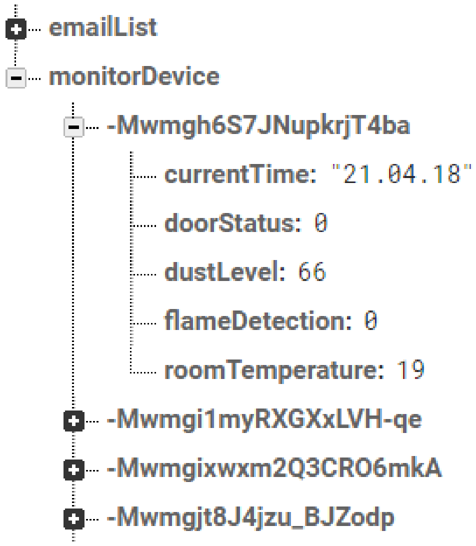
Step 1: The python programme imports classes from Firebase and PySerial.

Step 2: It forms a connection with Firebase using its hardcoded credentials,

Step 3: Sets the baud rate (speed of communication over a data channel) and opens the Serial Port.

Step 4: The programme loops over the following actions:

- Finds the current time and reads in a line of data from Micro:bit.
- Cleans and bundles the data into time-stamped records.
- Sends the record to Firebase via a HTTP POST request.
- Waits for three seconds



```
emailList
monitorDevice
    -Mwmgh6S7JNupkrjT4ba
        currentTime: "21.04.18"
        doorStatus: 0
        dustLevel: 66
        flameDetection: 0
        roomTemperature: 19
    -Mwmgi1myRXGXxLVH-qe
    -Mwmgixwxm2Q3CRO6mkA
    -Mwmgjt8J4jzu_BJZodp
```

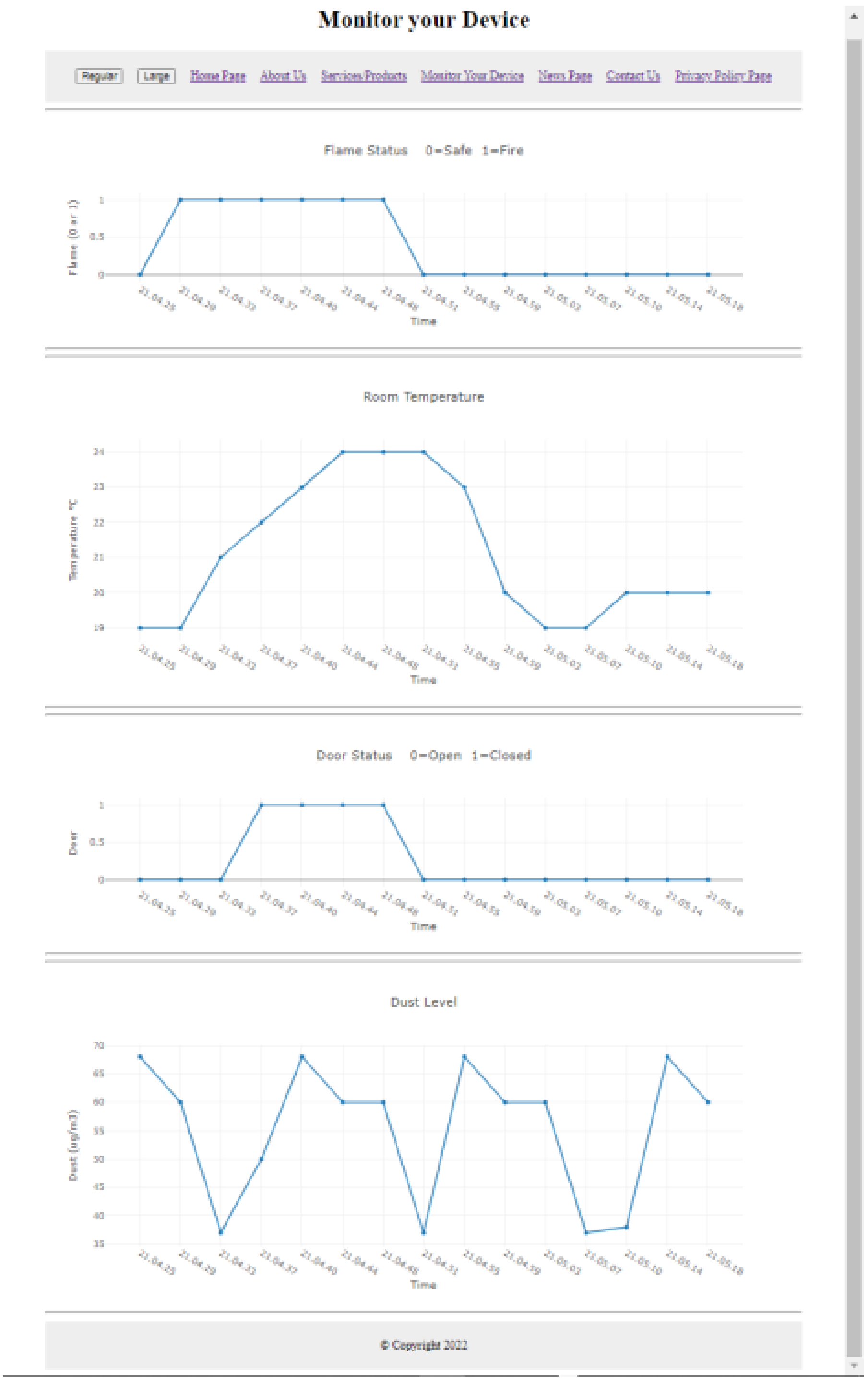
Monitor Your Device Page

This webpage is responsible for showing the user the data recorded by the micro:bit.

To begin, I learned about the Firebase Software Development Kit) by reading their documentation and going through their samples.

I had to import the Firebase library into the Javascript file. I loaded the credentials and setup a Firebase client. I used the client to retrieve the most recent data from Firebase. To test this code, I added mock data to the database and I logged the data to the browser console with *console.log()*.

I decided to use Plotly because it is covered in our school textbook. I re-familiarised myself with Plotly by going through the example in my Golden Key 'Computer Science for Leaving Certificate' book and referencing the Plotly website. I created Plotly graphs for each set of sensor data.



Website Design

Our website is used to display the product supplied, allow users to access the Monitor Device page and display other information regarding smart homes.

I drafted the content for the website in a text document. I used HTML and CSS to build up the website to resemble the wireframe (Design Page).

I added features to make the website accessible to a larger audience. For instance:

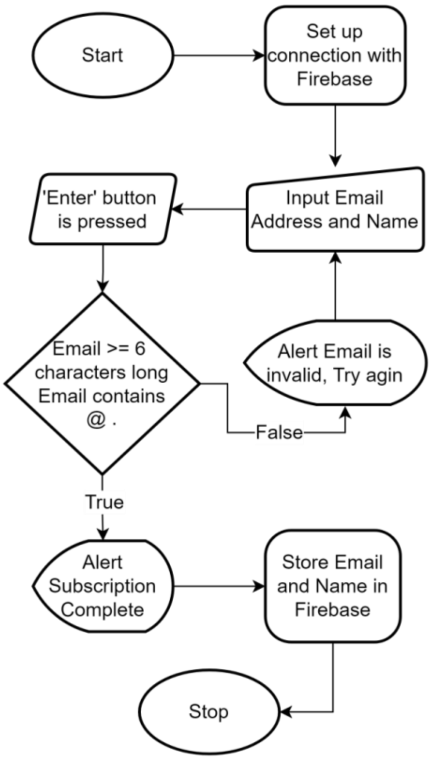
- I used a clean and simple design
- I grouped data into text boxes
- I used percentages in CSS so it scales well on different display sizes
- I added a button to change the font size. The button calls a Javascript function to update the text size

To give a concise explanation of home automation systems, I embedded an informative YouTube video.

Email Subscription

Customers can subscribe to our news feed by entering their email I used my existing knowledge of Firebase to import the credentials and store the emails in Firebase.

(Email Subscription Flowchart on the right)


Start
Set up connection with Firebase
Input Email Address and Name
'Enter' button is pressed
Email >= 6 characters long Email contains @ .
False
Alert Email is invalid, Try agin
True
Alert Subscription Complete
Store Email and Name in Firebase
Stop

It’s important that customers can give feedback and get in contact with us. On the right is the contact us form with implemented validation checks and a flowchart explaining the validation process.

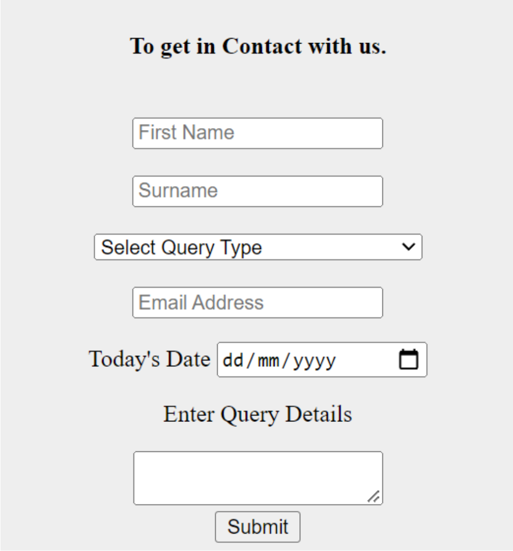
Contact Us Flowchart

Step 1: When the webpage is loaded, the user can input the relevant details into the required fields.

Step 2: The function runs through the inputs you entered once the "Submit" button is pressed.

Step 3: It checks the inputted data is valid and all fields of the form have been filled out.

Step 4: It then runs through each input one by one and when it finds a problem with the data entered, it will send an alert to the user who can change that input and try again.



To get in Contact with us.

First Name

Surname

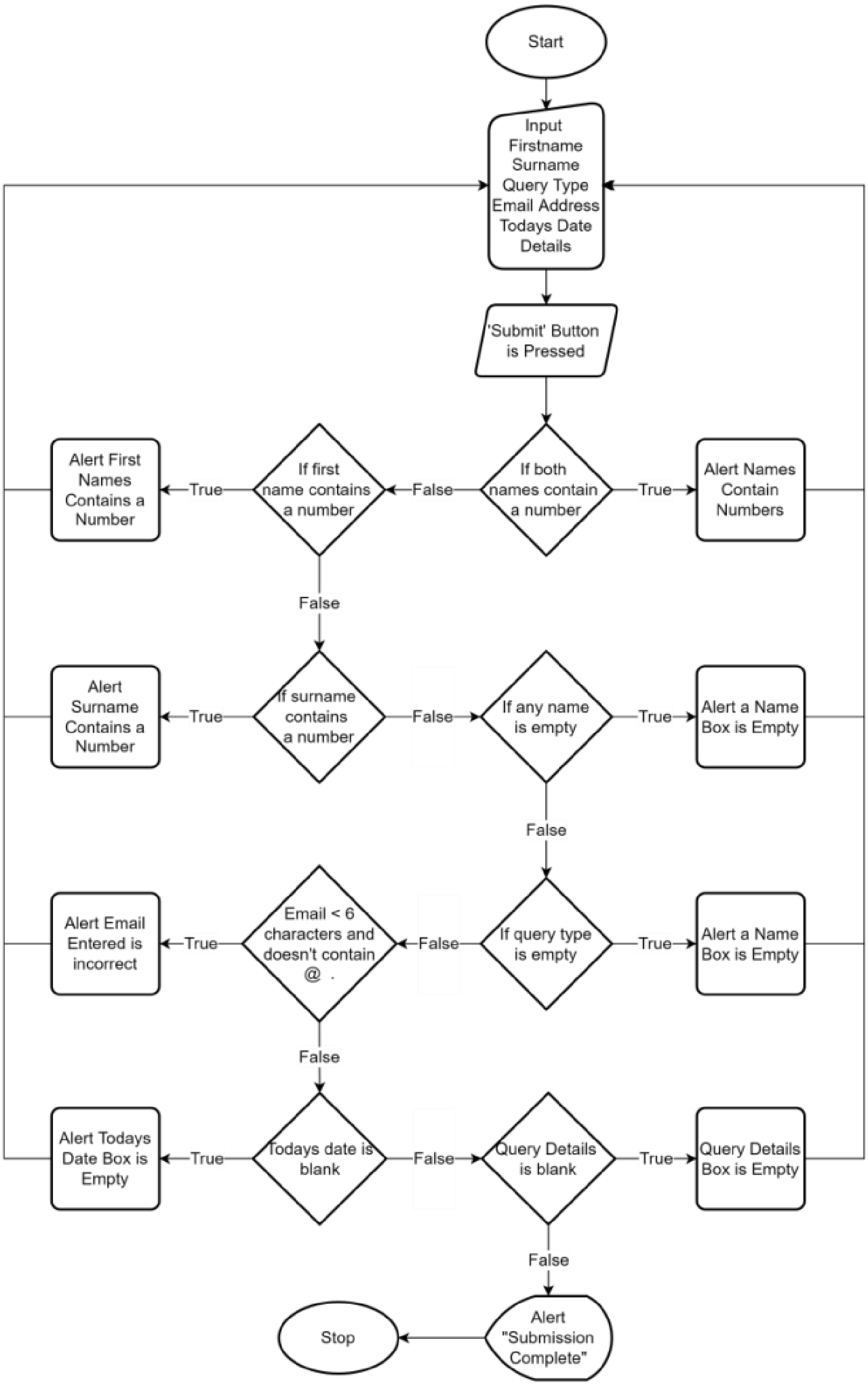
Select Query Type

Email Address

Today's Date dd/mm/yyyy

Enter Query Details

Submit


Start
Input
Firstname
Surname
Query Type
Email Address
Todays Date
Details
'Submit' Button is Pressed
If both names contain a number
True
Alert Names Contain Numbers
False
If first name contains a number
True
Alert First Names Contains a Number
False
If surname contains a number
True
Alert Surname Contains a Number
False
If any name is empty
True
Alert a Name Box is Empty
False
If query type is empty
True
Alert a Name Box is Empty
False
Email < 6 characters and doesn't contain @ .
True
Alert Email Entered is incorrect
False
Todays date is blank
True
Alert Todays Date Box is Empty
False
Query Details is blank
True
Query Details Box is Empty
False
Alert "Submission Complete"
Stop

Conclusion

Finally, I tested the system end-to-end. I verified that:

- The embedded system captures and passes data to the Python programme
- Python cleans and bundles the data into timestamped records
- Monitor Device page reads and graphs from data Firebase
- Email subscription stores data in Firebase
- The website displays all pages correctly

Here is the system working end-to-end with real-time data:

https://www.youtube.com/watch?v=_6h3kbOzw-0

4) Evaluation (Approximately 300 words)

For this project, I endeavoured to create a smart home device to improve home safety and provide users with real-time metrics on their home. I believe this system accomplished this goal: it can measure temperature, dust, magnetic strength and presence of a flame; display data on a dashboard; the website is full of useful information.

I'm happy with the software aspect of this project, especially the data processing and storage system. It efficiently processes and stores data that can be served to the user via the graph on the website.

The hardware fulfilled the goal of the project. It was quite easy to setup the micro:bit and the sensors were easy to use. But there is scope for improvement. For instance the Mirco:bit is reliant on a computer to run the Python program to process the data.

If I was to work on this project again I would make the following amendments:

- Use a Wi-Fi module to connect the embedded system wirelessly to the computer. A wireless connection is similar to what is used in the existing smart home products available on the market.

- I used the embedded system micro:bit as it is relatively easy to learn how to use it. However, I would like to experiment further with using Raspberry-Pi. This would allow me to run my Python script locally instead of on my computer, thus removing the need for a computer and a wireless connection. The Raspberry-Pi would clean, organise and upload the data to Firebase, which would reduce any delays in data transfer.