

Coursework Submission Cover Sheet

Please use Adobe Reader to complete this form. Other applications may cause incompatibility issues.

Student Number	1667866
Module Code	CMT223
Submission Date	08/05/2022
Hours spent on this exercise	415

Special Provision

☐

(Please place an x in the box above if you have provided appropriate evidence of need to the Disability & Dyslexia Service and have requested this adjustment).

Group Submission

For group submissions, *each member of the group must submit a copy of the coversheet*. Please include the student number of the group member tasked with submitting the assignment.

Student number of submitting group member 1667866

By submitting this cover sheet you are confirming that the submission has been checked, and that the submitted files are final and complete.

Declaration

By submitting this cover sheet you are accepting the terms of the following declaration.

I hereby declare that the attached submission (or my contribution to it in the case of group submissions) is all my own work, that it has not previously been submitted for assessment and that I have not knowingly allowed it to be copied by another student. I understand that deceiving or attempting to deceive examiners by passing off the work of another writer, as one's own is plagiarism. I also understand that plagiarising another's work or knowingly allowing another student to plagiarise from my work is against the University regulations and that doing so will result in loss of marks and possible disciplinary proceedings.



FIND MY ROOM

GROUP 4

CMT223 Internet of Things: Systems Design

Hossein Ramezani
1667866



Abdullah Barayan
2012898





Chaitanya Taru
21109423

Mohammed Shaad Matcheswala
21065793

Mohammed Fayaz Ansar Jelani
21058750

Internet of Things Systems Design – Group Answer Sheet

- Your IoT application will be evaluated from eight different aspects, as highlighted below.
- Additionally, you should also nominate two aspects to claim additional marks where you may have put more effort than the rest of the aspects.
- In order to express your areas of focus, please copy and paste  marks and  marks symbols from the advanced feature column into any of the eight columns. Please note that you cannot put both into one aspect/column.
- The demo video is marked for quality, clarity, and demonstration of problem-solving.

 	Applications and Use cases	Architecture	Sensing and Actuation	Networking and Communications	Data Management and Analytics	Privacy and Security	Human Factors and Interaction	Design Strategies and Prototyping
Advance Feature Marks								

- Each student should provide how many hours they spent working on each aspect (Please mark them in percentage as well).
- We are collecting the following information only for later analysis and for the improvement of this coursework.
- The following information will not be used for marking.
- It is a MUST for all students in the group to provide the following information.

Student Name		Applications and Use cases	Architecture	Sensing and Actuation	Networking and Communications	Data Management and Analytics	Privacy and Security	Human Factors and Interaction	Design Strategies and Prototyping	Total
Student 1 Abdullah Barayan 2012898	Number of Hours spent	120	90	10	70	15	10	5	50	370
	Percentage (%)	32.4	24.3	2.7	19	4	2.7	1.4	13.5	
Student 2 [Hossein Ramezani] [1667866]	Number of Hours spent	90	90	20	100	30	20	5	60	415
	Percentage (%)	21.7	21.7	4.8	24	7.2	4.8	1.2	14.5	
Student 3 [Chaitanya Taru] [21109423]	Number of Hours spent	10	60	75	25	10	5	15	10	210
	Percentage (%)	4.8	28.8	36	12	4.8	2.4	7.2	2.8	
Student 4 [Mohammed Shaad Matcheswala] [21065793]	Number of Hours spent	15	65	70	20	15	5	10	15	215
	Percentage (%)	7	30.2	32.5	9.3	7	2.3	4.7	7	
Student 5	Number of Hours spent	5	55	50	20	5	5	60	30	230

[illegible]

The total word limit for this section is 300-1000 words.

Applications and Use cases

	Domain	Smart Building
	Title	Find My Room
	Description	An application helps students look for study space in a university building. The application shows only available study space and its environmental properties such as crowdedness, temperature, brightness and noise level. Thus, students choose a suitable workspace. The number of students in the study area is calculated using the camera by detecting incoming and outgoing students. Other environmental characteristics are obtained through temperature, sound and light sensors in the study space. In addition, there is a dashboard for the building management team to add rooms, manage users, and query and analyse the rooms data, as the data is linked to Google Data Studio.
	Current Marketplace	There is currently a competitive market for IoT-based occupancy and counting solutions. Some solutions place sensors in front of each seat, but this solution may be a little expensive if there are many offices. Others use motion sensors to see if someone is occupying the space Or if the place is empty. In addition, some use cameras to find out how many people are present. These solutions help save energy by knowing the utilised places and make it easier for users to know the available places.
	Explain the Value Proposition	<i>“No more time-wasting”</i> Universities are always interested in creating the appropriate atmosphere for study. Find My Room makes finding a suitable study space easier instead of wasting students' time. It also helps

	the building management manage the building and know the utilized study spaces for energy saving.
The technology used (Both Hardware and Software)	<p>Software:</p> <ol style="list-style-type: none"> 1. Python 3 2. Flask Framework 3. joblib 4. sklearn 5. Google data studio 6. OpenCV 7. Pi camera module 8. Grovepi <p>Hardware:</p> <ol style="list-style-type: none"> 1. Raspberry Pi 2. GrovePi Plus 3. Raspberry Pi camera 4. Light Sensor 5. Temperature and Humidity Sensor 6. Sound Sensor
Link to Gitlab Account/Website URL/Web application	<p>https://gitlab.com/HosseinSR/CMT223-Group04</p> <p>https://iotuni.herokuapp.com/</p>
	<ul style="list-style-type: none"> • Create an architecture / data flow diagram (e.g., mini poster) to visualise and explain your IoT application • Label all components, data types you collect, sensors and actuators use, network protocols and communication technologies used, data analytics developed (i.e., what happened to data), security and privacy techniques you implemented, etc.

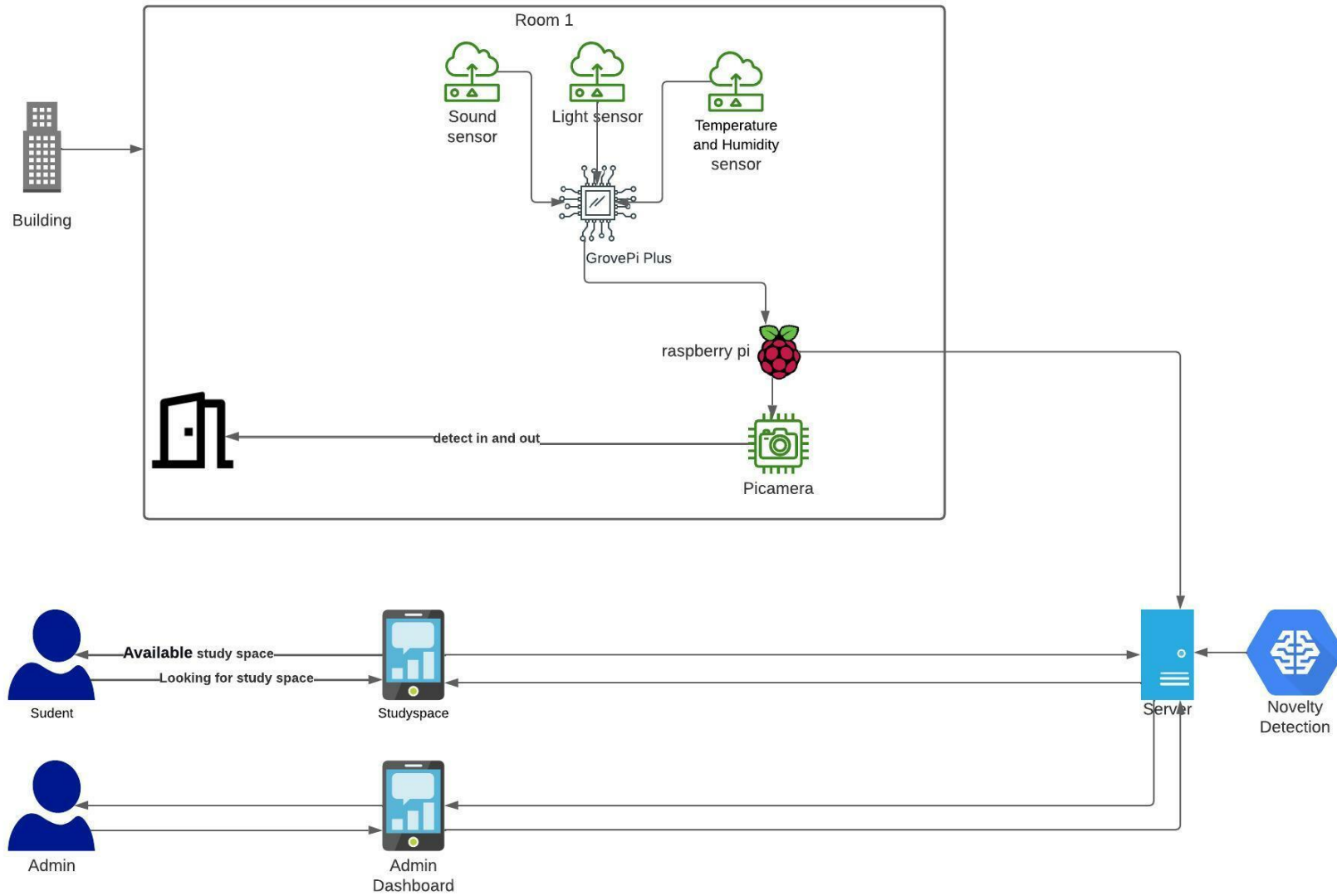


figure 1

- Explain how you applied what you learnt through lectures and labs into practice
- For each category, use 300 - 800 words MAX.

How did you apply what you learnt into practice (300 - 600 words MAX)

Architecture

IoT Device:

The sensors that are connected with the Raspberry pi are temperature and humidity, sound, light sensor v1.2 of Seeed Studio, and Raspberry pi camera v2 sensor of 8MP and are providing the reading based on the information of the surrounding. The values returned by the temperature are in float and the rest of the values are in integer. The readings are processed and sent to the local server. These sensors could be placed at any position where accurate information could be received. All these sensors are connected to Raspberry through Grove Pi. There is also a local server hosted on the raspberry pi through which the readings from the sensor could be forwarded.

Local Server:

For communication purposes, there are two servers, the first one is the local server on the raspberry pi, and the other is the cloud server. Each time the sensors generate any value, this is sent to the local server (gateway) where all the values are combined in a dictionary format and then sends the webhooks to the cloud server based on the URL which is provided. Apart from sending sensor values to the cloud server the gateway also processes the data and scales the value in terms of "high", "medium", and "low" making it in human-readable form.

Cloud server:

The flask server is hosted on Heroku. This server is a passive component and a three-layer architecture that has a web application, database, admin-flask, and website. The server contains a database used for storing the user information, logs containing the webhooks coming from the local server, and a table for keeping the room value. Google data studio is implemented on the database which provides an analysis report on the information inside it. For the admin dashboard, admin-flask is being used in the cloud server to manage records in the databases. Moreover, there is a machine learning model being used for identifying abnormal conditions, and in case of one mail will be sent to the appropriate person. Furthermore, the cloud server also hosts a website and the URL is <https://iotuni.herokuapp.com/>. This web application displayed the temperature, noise, humidity, light, and occupancy of the room which are stored in the database. Also, users could register themselves and with the same registered credentials, the user could log in and choose the desired room based.

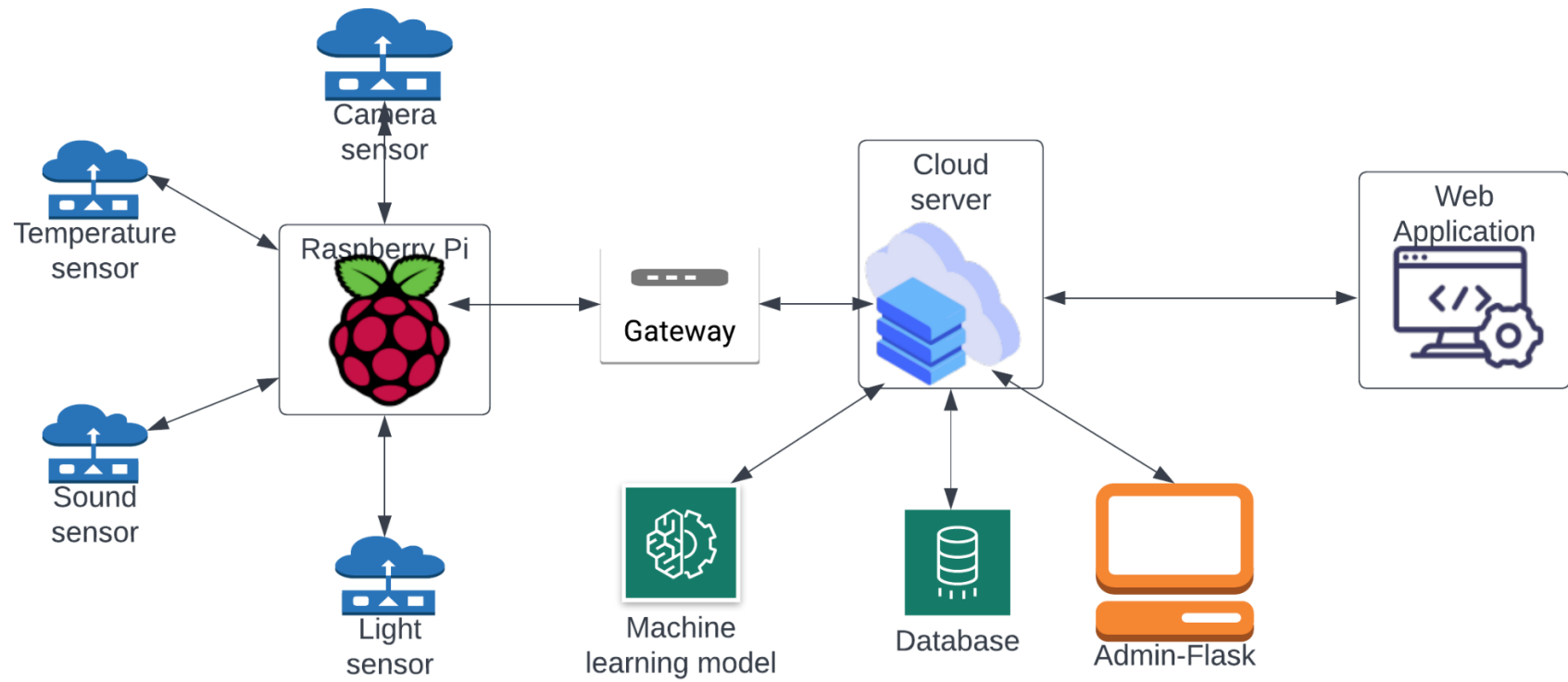


figure 2

Sensing and Actuation

Temperature-humidity, sound, and light:

The project focuses on measuring the temperature, humidity, light, sound, and occupancy of the room. Starting with detecting overall temperature, the temperature sensor placed on the raspberry pi will keep detecting values and return the appropriate value to the application. Moreover, sound and light sensors are connected to the raspberry pi. The light sensor and the sound sensor will keep detecting the values simultaneously. If the value from the temperature-humidity sensor is NOT NaN (Not a Number) then the program will pass the previous value further to the application. The temperature sensor will read the value slower than that of sound and light sensors. Thus we take values periodically in sequence to nullify the effect of delays and reading differences between different sensors. As the temperature and humidity sensor sometimes fails to read values and create NaN entries. Hence, only when the value from the temperature sensor is not NaN, we will consider the value. The temperature sensor reads the value in celsius which is the standard reading therefore it is passed as it is. Similarly, the humidity sensor provides values in 0-100 range and there is no need for processing, which is directly passed to the application. Sound analog reading is converted to decibels and gets processed to low, medium, or high depending upon the values. For light sensors, analog readings are in the range of 0 to 1023, which are then converted into low, medium, and high depending upon the values[2].

Camera:

The camera is used as the core sensor in this project. The camera will be placed at some height near the entrance of the room, which will be running continuously and taking the live feed. Once someone passes from the frame of the camera, the model is designed to capture the motion and actuates the program to detect whether someone is entering the room or exiting it. Using this information we can calculate how many people are there inside a specific room.

It is not ideal to run complex neural network algorithms on raspberry pi for object detection. This object detection is based on the Local Binary Pattern [1]. This method perfectly fits the low computational powered raspberry pi device. Once the pi camera reads a frame, the frame then gets processed in a grayscale image and all the pixels will get binarised. Binarization of the image is a method where for some threshold value, all the pixel values below the threshold will be mapped to 0 and all the pixels above the threshold will be mapped to 255. This way we get pure black and white frame. Once we get this, any moving object will provide opposite values as to the background and we can detect an object. Managing the previous frame as a reference frame, movement can be detected based on the delta between them. Using two reference lines placed on the camera stream camera can determine whether someone is entering or exiting the room. Finally, all the values (count, temperature, humidity, sound, light) will be sent to the server through webhooks.

Networking and Communications

The data fetched from the IoT devices need to be transferred to the servers where the processing will be done. In our project webhooks send the sensor data to the gateway and again the values are processed and then forwarded to the server. Webhooks are nothing but automated messages from a device whenever it generates some instance. That is after certain readings when data is appropriate webhooks will send data to the server. Webhook is nothing but sending a request to the intended URL. Based on this request data will be entered into the database. For each entry, the database will save the timestamp for unique and timely records. The whole communication is organised using the UDP messaging technique. Raspberry pi will send a request to the server and the message transfer will be activated from the server-side as a response. This system also works with the website. The website is designed for users to check the metrics of the room. Based on the personal choices the user can select from multiple rooms.

Password encryption using bcrypt:

The passwords in the database are encrypted so that even if the information is lost or stolen, it would be hard to decrypt and for encrypting the password Bcrypt is used. Bcrypt is a password hashing function which uses 128-bit salt (which could be any random string), 192-bit hash value and a password string of up to 72 bytes and the output which is generated is an encrypted string.

HTTPS:

The user is communicating with the server through the website and the request which is sent is using the TCP protocol. The user can register and login into their accounts. The credentials are transformed into the form of HTTPS get requests and the information regarding the rooms fetched from the database is provided in terms of HTTPS post requests.

Heroku is a platform as a server(PaaS) in which the development environment and deployment are in the cloud and the resources could be delivered from the cloud application to sophisticated or cloud-enabled enterprise applications. The website is deployed on Heroku, making the website directly deployable from tools like git, GitHub or continuous integration. The website could be managed online and could also increase its performance using a web-based Heroku dashboard.

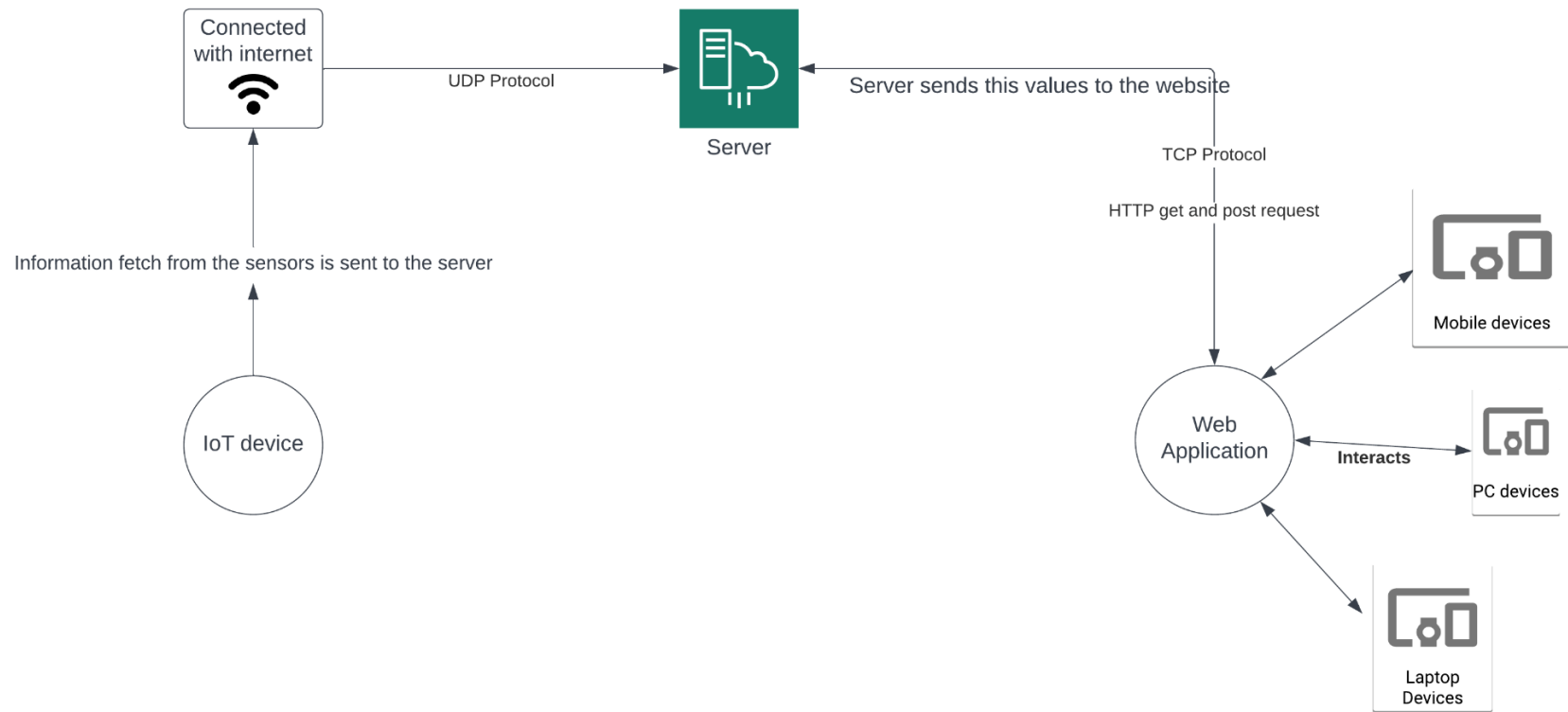


figure 3

Database:

The database used for our website is PostgreSQL which contains various features used for helping developers to build applications, protect data integrity, build fault tolerance environments and help to manage the data irrespective of its size. In our case, the database is cloud-based which is hosted on Heroku and it contains a URL for accessing it. The database is used to store three types of queries, one for user's registration, the other for storing the logs provided from the webhooks and lastly for storing the admins' credentials. For accessing (querying) the database, a user requires a server certificate, client certificate and client private key making sure that it is secure and only the authentic user could access it. The client key and certificates are generated by executing an OpenSSL command on the server's bash terminal and for generating the server certificate a python script is executed on the terminal with the URL of the database.

Adding flexibility to the database:

Connecting to the database is a bit complex for the third part (managing team, administrative team, etc.) and google data studio makes it flexible for them to connect. This could also help the third party to access the data and also generate an analysis report based on the data, making this process less complicated.

Admin Rights:

The database should only be accessed by an authorised person and for accessing the database the administrator rights are mandatory and all the data could be managed properly by the admin. For example, any improper user information, or any inaccurate logs regarding the rooms or room number values could be added, updated or deleted.

Flask-admin reduces the task of building and creating an admin infrastructure on top of the existing model. It creates a user interface to manage the web server's data.

For connecting the database from flask-admin a URL of the database is required and a secret key could be provided through a python file. The admin has full access to the database and could modify, update or delete any table which is required.

However, it's important to note that a full authentication system for the admins is not yet implemented, however, the registration system has been streamlined to easily integrate a full authentication system for admins.

ML model for novelty detection:

A machine learning model has been implemented on the website to differentiate between an ideal condition and an abnormal condition. The model that is used is one-class SVM which is an unsupervised algorithm that learns decisions for novelty detection and was trained on the dataset containing temperature values and if the reading showed any abnormality the mail is sent to the staff member.

Privacy and Security

Privacy preservation during data collection:

The data gathered from light, sound(noise level), temperature and humidity sensors provide only environmental readings whereas the Pi camera module processes information based on capturing the people's movements. People's faces are not recorded as the device would be installed with the Pi camera having a top view. In top view only the human's movements are captured and that information is processed to identify whether an individual has entered or exited a room. So, during the data collection process none of the sensitive information pertaining to individuals present inside a room is gathered. Hence preserving privacy of individuals.

Secure communication:

The first layer of encryption and privacy solutions has been implemented on the IoT side of the project. Instead of the IoT device sending data directly to the cloud server, a proposed solution was to create a local flask application that handles the communication from the Raspberry Pi and processes the data, then sends all relevant information directly to the cloud server utilizing webhooks with the added benefit of the SSL/TLS protocol to encrypt the HTTPS communications. Thus, securing the shared data from manipulation during transformation.

Heroku deployed server:

Since the back end application is a flask based server, its URL and data are automatically HTTPS/SSL encrypted, furthermore, the cloud server is hosted on Heroku, and its features are maintained by Heroku; this is beneficial as Heroku provides multiple security solutions with a robust dyno system to detect any irregularities. Furthermore, Heroku consoles can be accessed anywhere and do not need a specialised system allowing for swift and agile maintenance of the server.

Postgresql database:

Heroku provides a postgresql database encrypted at rest with AES-256, block-level storage encryption. Keys are managed by Amazon, and individual volume keys are stable for the lifetime of the volume. All backup files that are stored in an encrypted S3 bucket.

Motivation for focussing on user experience aspect:

The motivation of a user to use "Find My Room" is to find a suitable room in their university to study, conduct meetings, etc., depending on their taste of how a room should be. For example, a user might like to study in dim light and another user might like a less-crowded room. So everyone has their own taste for a room where they want to spend their time in the university. "Find My Room" would make this easy by finding a suitable room according to the user's preferences such as

- Brightness
- Noise level
- Temperature
- Humidity
- Number of Occupants

Designing model focusing on user experience:

The focus of the user experience is not only on the services offered by the "Find My Room" via visual user interface but also on how quick and accurate the physical device can gather necessary readings and send it to the server for processing. Following are the two main aspects of the "Find My Room" system which make the user experience great.

- **Responsiveness:** A web application is created using responsive web design with the help of HTML, CSS and Bootstrap for providing a smooth user interface.
- **Optimization:** To make the physical device read and send data quickly to the server, algorithms that run in the Raspberry Pi are optimised to be computationally lightweight in nature, as Raspberry Pi would take time for processing heavy computational tasks. This helps in reducing latency as much as possible especially since computer vision techniques (heavy computational tasks) are used in the processing of frames from live video feed captured using a Pi camera module to determine the number of individuals present inside a room (calculated using a number of people who have entered and exited a room).

Issues that might affect user experience:

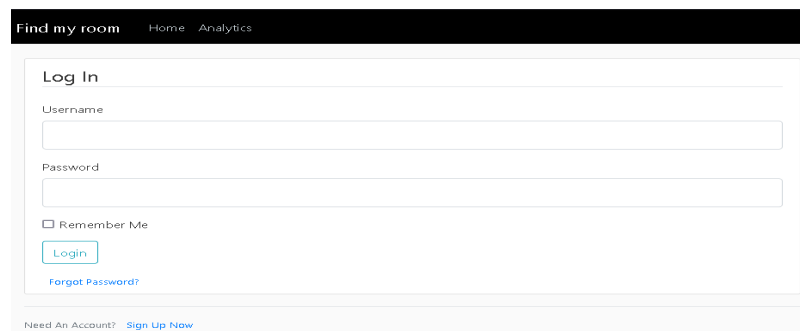
Though the web application and the IoT device run smoothly, without a proper network connection between the IoT device and the server, there is no way a user can have a good experience with the application. Being responsive is the main aspect of "Find My Room" but due to networking issues, there can be potential latency in receiving data from the device which in turn might delay the process of finding the right room suitable to the user or might even pick up the wrong room because of inconsistency with the actual readings and the values present in the server.

Design Strategies and Prototyping

The User Interface:

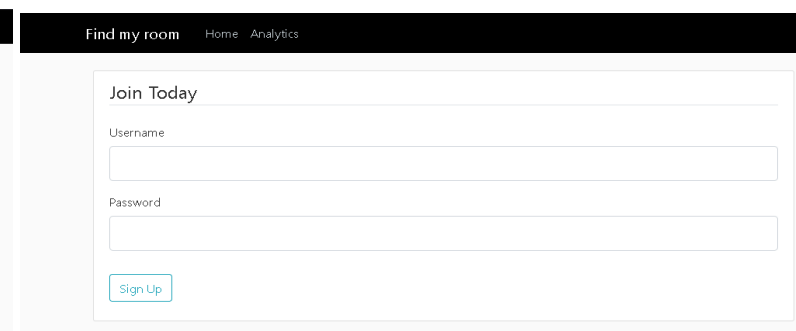
To achieve the primary use case of the project, a clear and specific front end is needed. The web application provides an HTML-CSS-based system that allows the user to register, log in, log out, navigate the rooms and view their status. Furthermore, an admin dashboard is provided for administrative purposes such as viewing and modifying the database. Lastly, the backend database is connected to Google's data studio, which serves as a scalable demonstration of data analytical solutions on the administrative side of things.

The home page shows the available rooms and their properties—the login and Sign up pages.



The login form is titled "Log In" and is located within a dark header bar that also contains the navigation links "Find my room", "Home", and "Analytics". The form includes a "Username" input field, a "Password" input field, a "Remember Me" checkbox, a "Login" button, and a "Forgot Password?" link. At the bottom of the form, there is a link "Need An Account? Sign Up Now".

fig 4



The sign up form is titled "Join Today" and is located within a dark header bar that also contains the navigation links "Find my room", "Home", and "Analytics". The form includes a "Username" input field, a "Password" input field, and a "Sign Up" button.

fig 5

The analytics pages show the analysis data for the rooms, where the data can be filtered based on the room and date range.

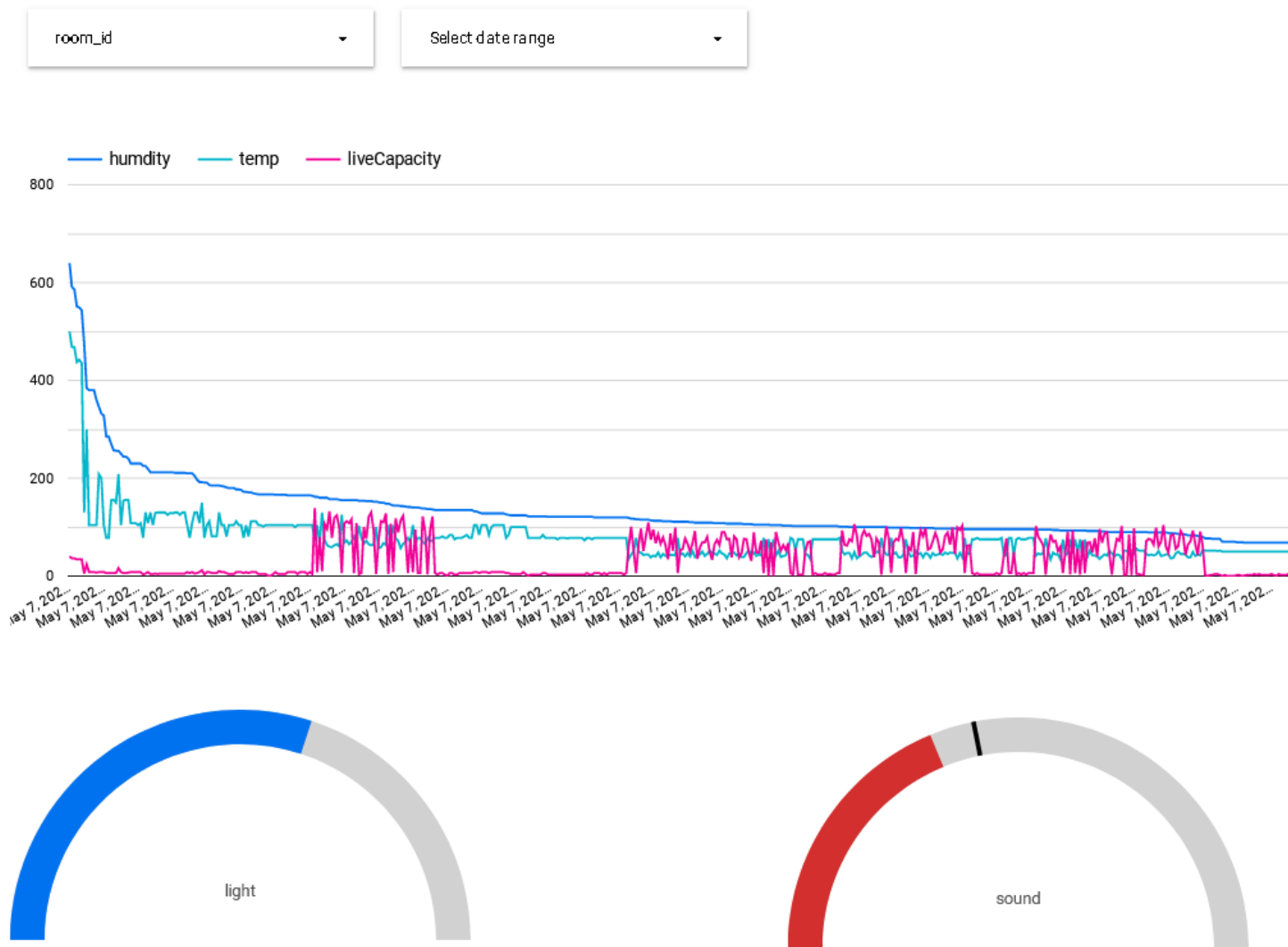
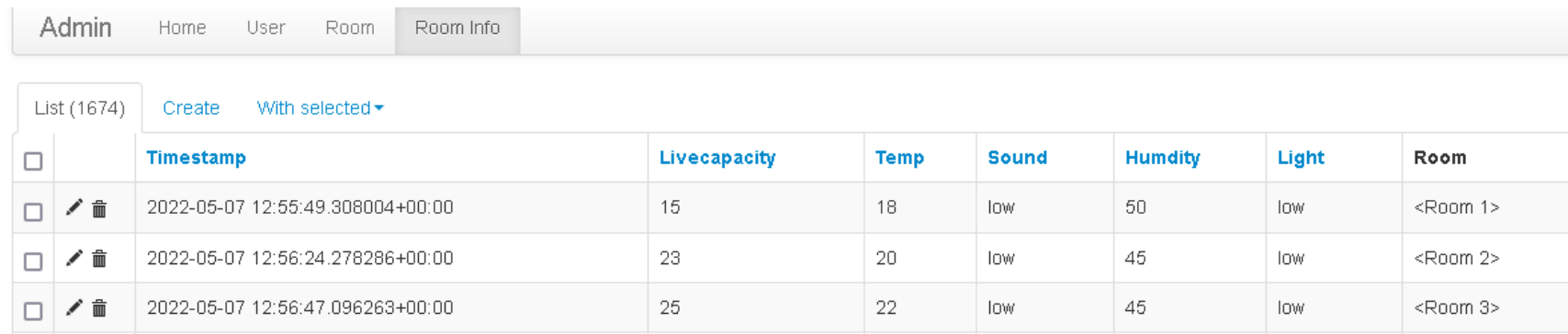


fig 6

The admin dashboard is where the admin can manage rooms and users.



The screenshot shows the Admin dashboard with the 'Room Info' tab selected. The dashboard has a navigation bar with links: Admin, Home, User, Room, and Room Info. Below the navigation bar, there is a 'List (1674)' button, a 'Create' button, and a 'With selected' dropdown menu. The main content is a table with the following columns: Timestamp, Livecapacity, Temp, Sound, Humidity, Light, and Room. The table contains three rows of data.







		Timestamp	Livecapacity	Temp	Sound	Humidity	Light	Room
<input type="checkbox"/>	 	2022-05-07 12:55:49.308004+00:00	15	18	low	50	low	<Room 1>
<input type="checkbox"/>	 	2022-05-07 12:56:24.278286+00:00	23	20	low	45	low	<Room 2>
<input type="checkbox"/>	 	2022-05-07 12:56:47.096263+00:00	25	22	low	45	low	<Room 3>

fig 7

The Hardware:

To keep the project within economic and financial range, all the used hardware was simple, easy to get and relatively cheap. For the main unit, a Raspberry PI with Raspbian OS as the default operating system with the following sensors for a proof of concept:

1. Raspberry Pi camera
2. Light Sensor
3. Temperature and Humidity Sensor
4. Sound Sensor

The Backend:

Since the aim was to create an end-to-end IoT system, the team has decided to create a fully-featured web application as the main communication layer between the IoT device and the web. As such, and after extensive research, we opted for the FLASK python library to create a robust and stable back end. It is deployed to Heroku and connected to an SSL encrypted POSTGRESQL database.

The connectivity:

The raspberry pi is connected directly to the WI-FI and communicates directly to the server via python webhooks; the server then updates the databases, and clients can see the changes. Postgresql database is hosted on the server.

References


1. Farouk Khalifa, A., Badr, E., & Elmahdy, H. (2019). A survey on human detection surveillance systems for Raspberry Pi. *Image And Vision Computing*, 85, 1-13. <https://doi.org/10.1016/j.imavis.2019.02.010>
2. Tayef, S., Rahman, M., & Sakib, M. (2021). Design and Implementation of IoT based Smart Home Automation System. 2021 24Th International Conference On Computer And Information Technology (ICCIT). <https://doi.org/10.1109/iccit54785.2021.9689809>

Group 4

General Results:

Average	100%
Completion	100%
Start Time	2/21/2022, 4:37:31 PM
Duration	300s

Interactivity Results


Name	Score	Duration	Weighting	Completed
Applications and Use Cases Quiz	100%	287	1	

Group 4

General Results:

Average	94%
Completion	100%
Start Time	3/27/2022, 10:39:42 PM
Duration	1765s

Interactivity Results


Name	Score	Duration	Weighting	Completed
Architectures Quiz	94%	1763	1	

Group 4

General Results:

Average	96.67%
Completion	100%
Start Time	2/21/2022, 10:43:19 PM
Duration	362s

Interactivity Results


Name	Score	Duration	Weighting	Completed
Sensing and Actuation Quiz	97%	346	1	

Group 4

General Results:

Average	93.33%
Completion	100%
Start Time	2/23/2022, 12:13:23 PM
Duration	418s

Interactivity Results


Name	Score	Duration	Weighting	Completed
Networking and Communications Quiz	93%	414	1	

Group 4

General Results:

Average	93.33%
Completion	100%
Start Time	3/27/2022, 4:08:24 PM
Duration	149s

Interactivity Results


Name	Score	Duration	Weighting	Completed
Data Management and Analytics Quiz	93%	129	1	

Group 4

General Results:

Average	100%
Completion	100%
Start Time	3/25/2022, 1:25:26 PM
Duration	2780s

Interactivity Results


Name	Score	Duration	Weighting	Completed
Privacy and Security Quiz	100%	2778	1	

Group 4

General Results:

Average	96.67%
Completion	100%
Start Time	3/27/2022, 4:08:24 PM
Duration	2173s

Interactivity Results

Name	Score	Duration	Weighting	Completed
Human Factors and Interaction	97%	2172	1	

Group 4

General Results:

Average	93.55%
Completion	100%
Start Time	3/27/2022, 10:26:58 PM
Duration	441s

Interactivity Results

Name	Score	Duration	Weighting	Completed
Design Strategies and Prototyping Quiz	94%	438	1	