

CHITKARA

SOFTWARE ENGINEERING 2: DEVELOPING INTERNET-OF-THINGS APPLICATIONS 2023

ONTRACK SUBMISSION

Project Handover

Submitted By:

Saksham BEHAL

2110994801

2023/05/07 17:59

Tutor:

Amit KUMAR

| Outcome | Weight |
|-------------|--------|
| Develop | ◆◆◆◆◆ |
| Connect | ◆◆◆◆◆ |
| Collaborate | ◆◆◆◆◆ |
| Deploy | ◆◆◆◆◆ |
| Document | ◆◆◆◆◆ |
| Justify | ◆◆◆◆◆ |

This was a vital part of the semester as it is the evidence of the completion of our project. It includes all the information ranging from purpose to documentation and beyond. This illustrates our implementation of concepts that we have learnt throughout the course to solve real-world problems.

May 7, 2023





Deakin University
Smart Lecture Hall Management System

Project Handover

05.05.2023

Team Name
Divide And Conquer

Project Team
Saksham Behal(Back-End Developer and team leader), 2110994801
Gaurish Bhatia(Hardware Team Leader and back-end researcher), 2110994762
Mehak(Front-End Developer and team coordinator), 2110994772
Krish(Hardware team member and researcher), 2110994848
Armaan Chetal(Hardware Team Member), 2110994755

Contents

| | |
|---|-----------|
| Purpose | 4 |
| Project Description | 4 |
| High-level architecture of the product:..... | 5 |
| Detailed description of the hardware architecture diagram: | 14 |
| Detailed description of the communications diagram: | 16 |
| Features completed: | 17 |
| Artefacts List : | 21 |
| Planned Work/Open Issues: | 21 |
| Lessons Learned: | 22 |
| Setup Guide..... | 27 |
| User Manual | 29 |
| Hardware: | 39 |
| API Documentation: | 40 |
| API Testing: | 43 |
| Examples of some of the tests: | 44 |
| Meeting minutes: | 45 |
| Sprint Timeline: | 47 |
| GitHub repository link : | 48 |
| Link for the system architecture: | 48 |
| Link for hardware and communication architecture/ diagram: | 48 |
| Google drive link for project demonstration video: | 48 |
| Sprint Timeline Diagram: | 48 |
| Sign-off | 48 |

Table of Figures:

| | |
|--|----|
| Figure 1 Initial plan and system architecture | 5 |
| Figure 2 Final architecture | 6 |
| Figure 3 Student page..... | 7 |
| Figure 4 Teacher page..... | 8 |
| Figure 5 Landing page | 9 |
| Figure 6 Landing page | 10 |
| Figure 7 Screenshot of the whole architecture | 11 |
| Figure 8 Hardware architecture..... | 12 |
| Figure 9 Schematic 1 of hardware..... | 13 |
| Figure 10 Schematic 2 of hardware..... | 14 |
| Figure 11 Communication diagram | 15 |
| Figure 12 Hardware setup..... | 25 |
| Figure 13 Screen display (Hardware implementation) | 26 |
| Figure 14 Screenshot of Landing page | 30 |
| Figure 15 Timetable displayed | 31 |
| Figure 16 Student page (feedback portal) | 32 |
| Figure 17 Screenshot of Admin page | 32 |
| Figure 18 Screenshot of Teacher Page..... | 34 |
| Figure 19 To address the feedbacks | 34 |
| Figure 20 Responsive webpages | 36 |
| Figure 21 Teacher page functionalities | 37 |

Purpose

The purpose of a smart lecture hall management system is to provide a comprehensive solution for academic institutions to manage their lecture halls efficiently and effectively. By solving the major issues of booking lecture halls by teachers and students, disturbance of ongoing classes to check the status of a lecture hall, and students not knowing the venue of their lecture till the last moment, the system aims to provide a hassle-free and streamlined process for managing lecture halls.

The system accomplishes this through the use of both hardware and software components. The hardware includes sensors and displays that provide real-time information about the status of lecture halls, allowing teachers and students to easily check the availability of lecture halls without disrupting ongoing classes. The software component provides a user-friendly interface that allows teachers and students to book lecture halls easily, and also provides up-to-date information about the venue of lectures, eliminating confusion among students.

In addition to addressing the major issues faced by academic institutions in managing lecture halls, the system also provides additional functionalities such as a grievance portal that allows students to raise their concerns and complaints and the ability for teachers to upload the timetable from their mobile phones. These additional functionalities contribute to a more efficient and effective management system, leading to better learning outcomes for students and higher satisfaction rates among stakeholders.

Project Description

- Smart lecture hall system utilizes both hardware and software components to provide a comprehensive solution to the challenges faced by academic institutions in managing their lecture halls.
- The system provides a user-friendly interface that enables teachers to book lecture halls easily and allows students to view timetables and posters for upcoming events.
- The system integrates an IoT system that provides real-time information on the occupancy, temperature, and humidity in the lecture hall. This enables teachers and students to easily check the availability of lecture halls without disrupting ongoing classes. The system also ensures that the lecture halls are not disturbed while the lecture goes on without any hindrance, leading to a better learning experience for students.
- The website design is secured, and the users that sign in are authenticated by the system, providing user security.
- The project aims to facilitate communication between faculty and students by providing features such as timetables, posters, and other information.
- The scheduling feature enables the effective allocation of lecture halls, saving time and resources by providing the feature of booking.
- One of the key objectives of the project is to provide a low-cost solution compared to existing applications that have high subscription charges and fees such as ascTimetables,

Go Schooler, and Prime Table. This ensures that the system is accessible to academic institutions of all sizes.

- The project also includes a grievance/feedback portal, providing students with a platform to raise concerns and complaints related to the lecture hall management system. This contributes to a more efficient and effective management system, leading to better learning outcomes for students.

High-level architecture of the product:

Initial plan and system architecture

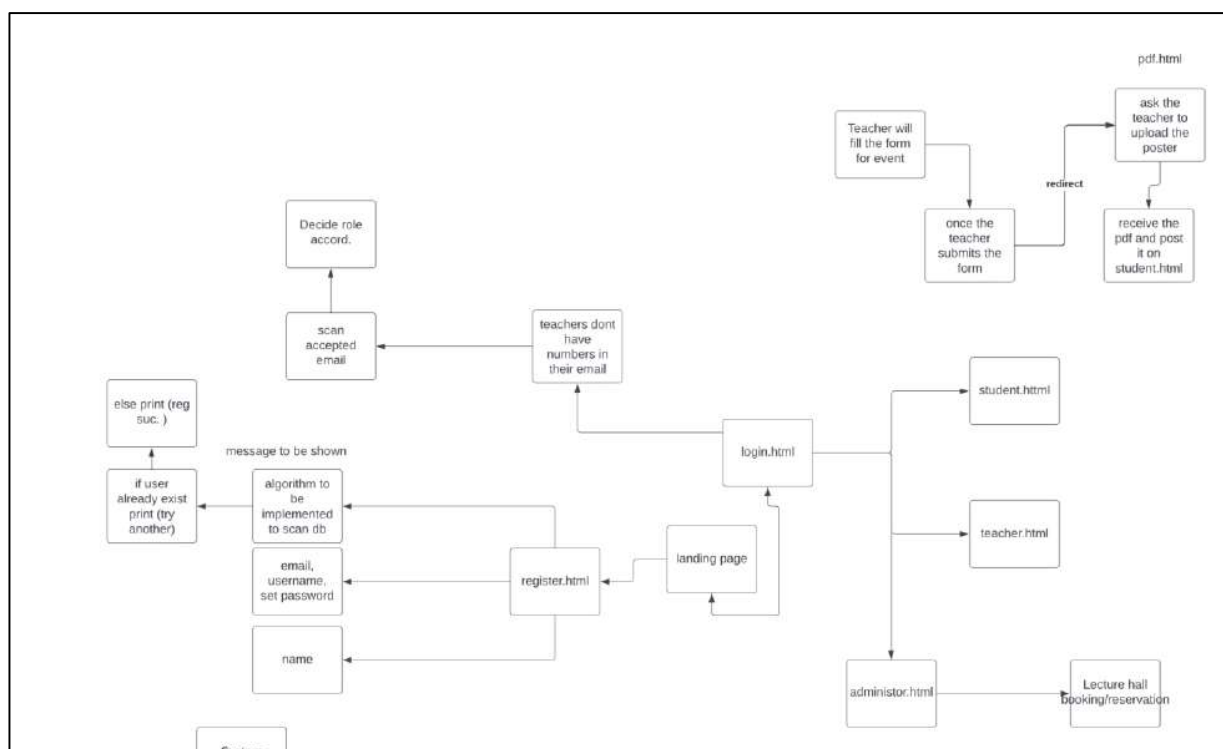
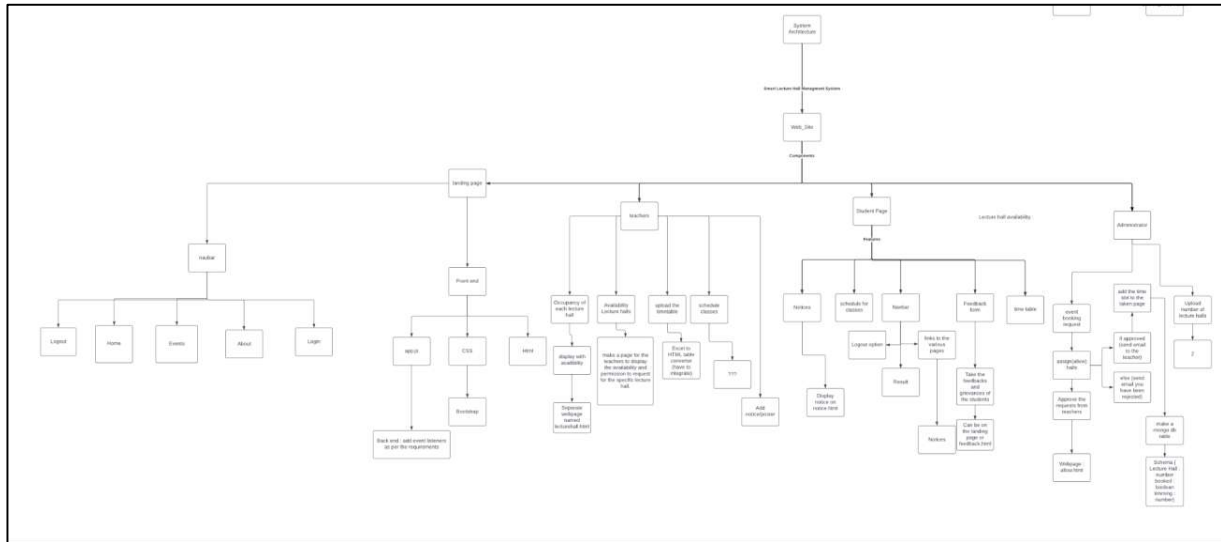


Figure 1 Initial plan and system architecture



Final architecture:

MongoDB schemas

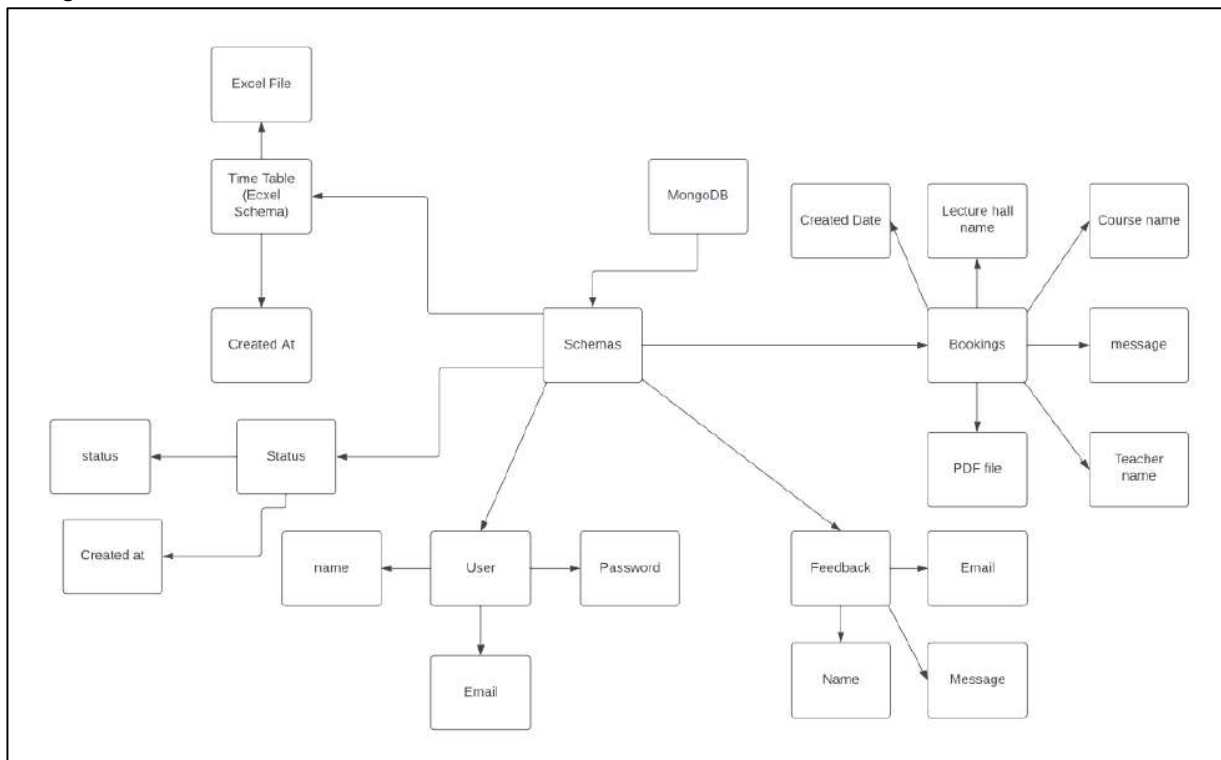


Figure 2 Final architecture

Student Page

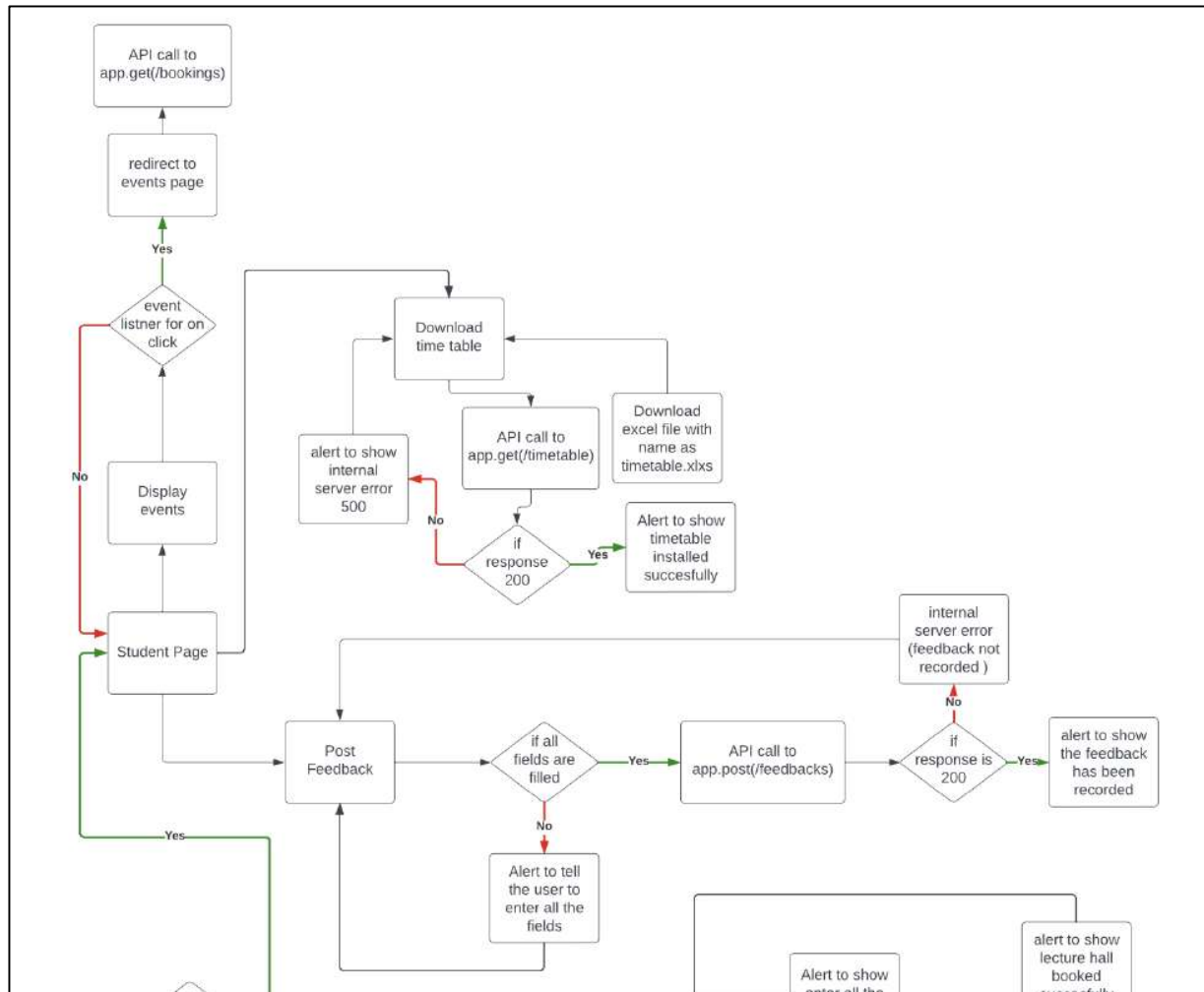


Figure 3 Student page

Teacher Page

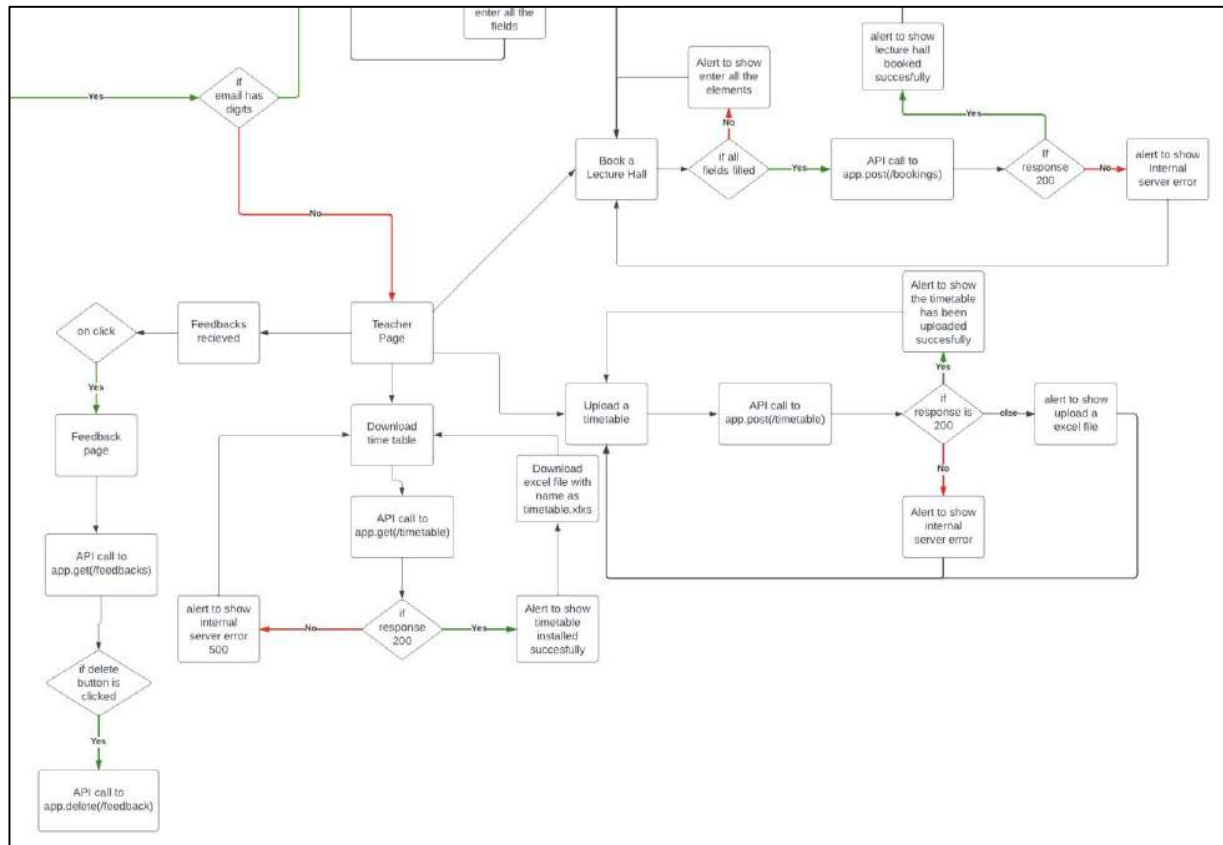


Figure 4 Teacher page

Landing Page

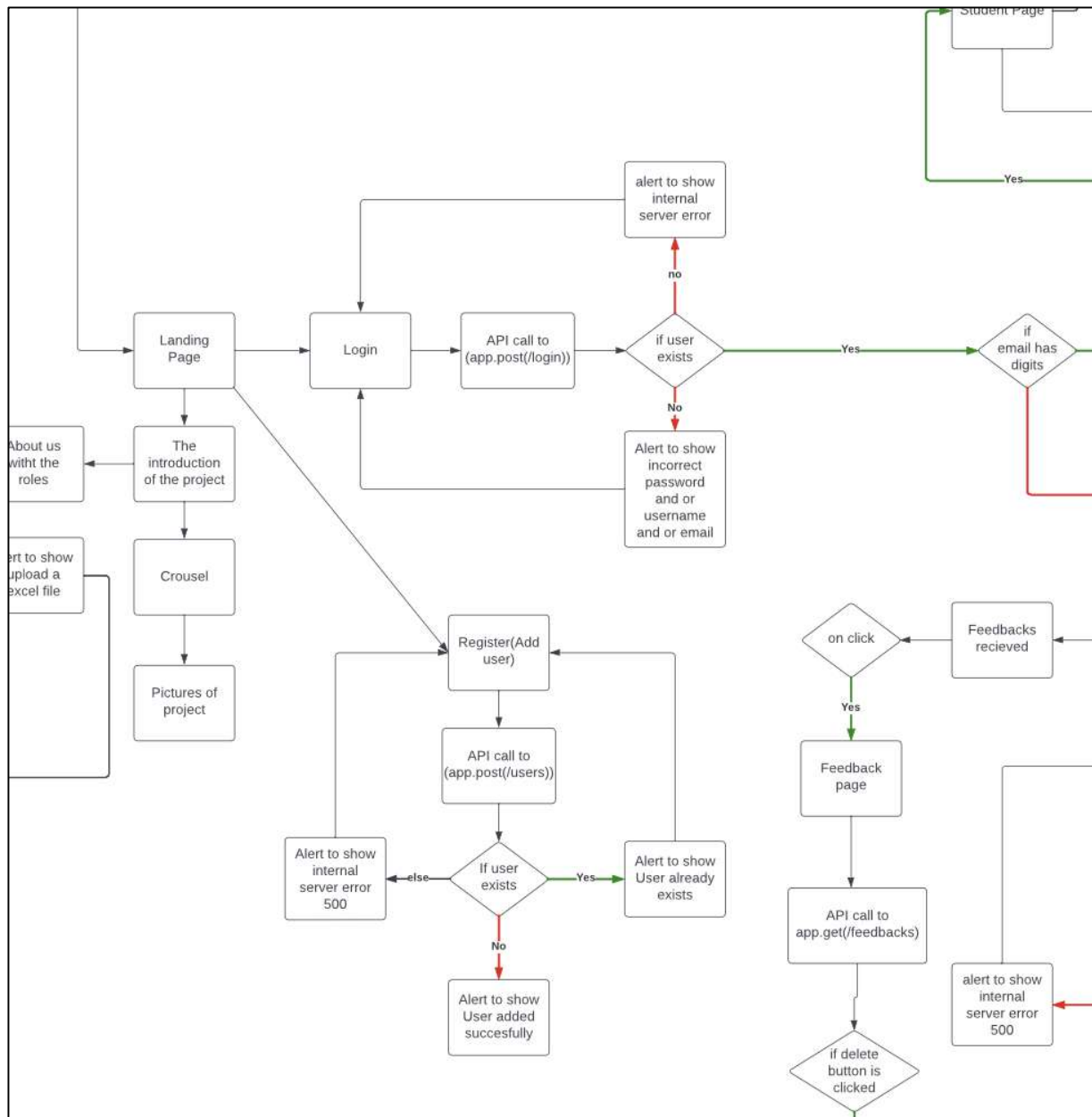


Figure 5 Landing page

Admin Page

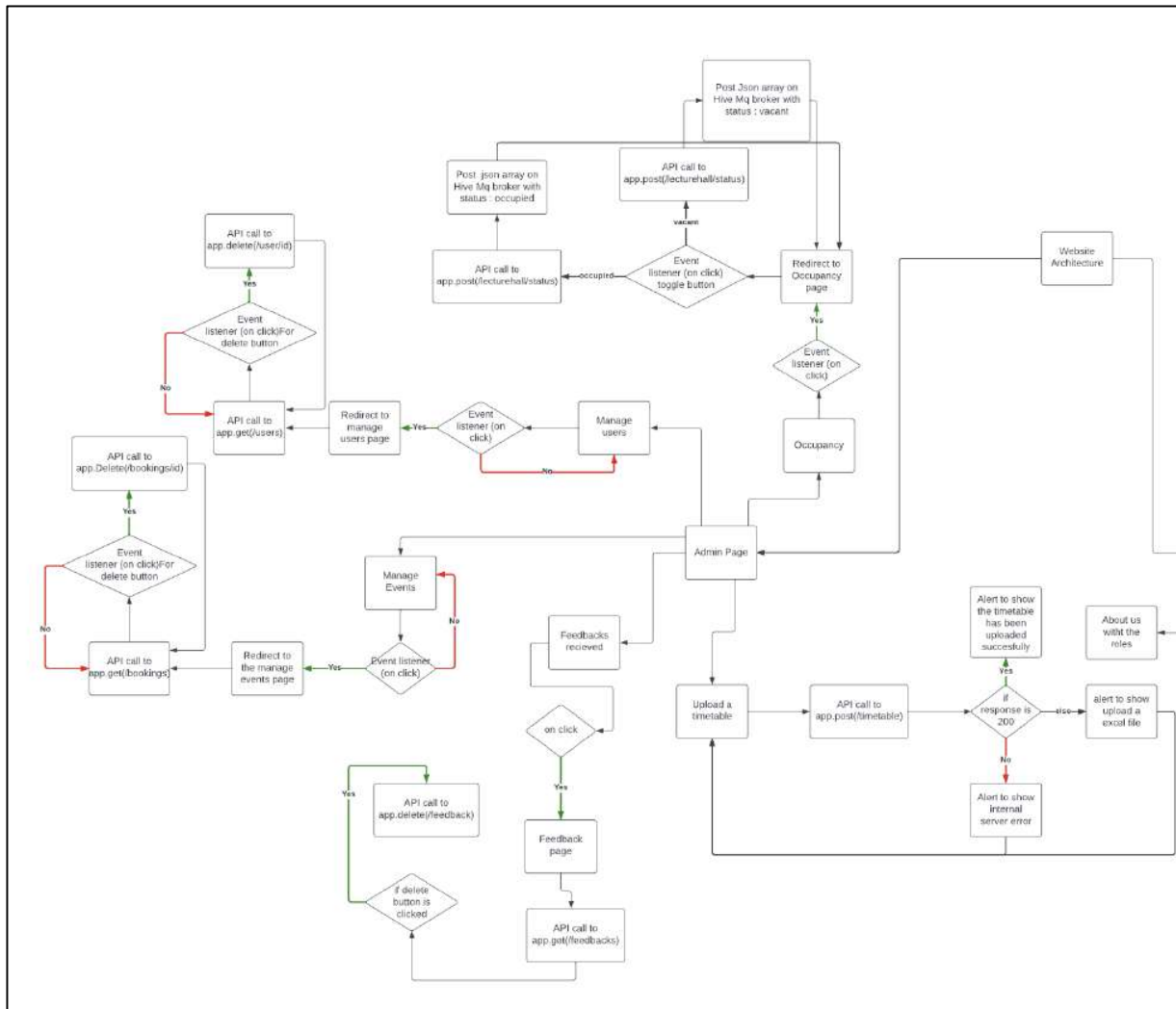


Figure 6 Landing page

Screenshot of the whole architecture

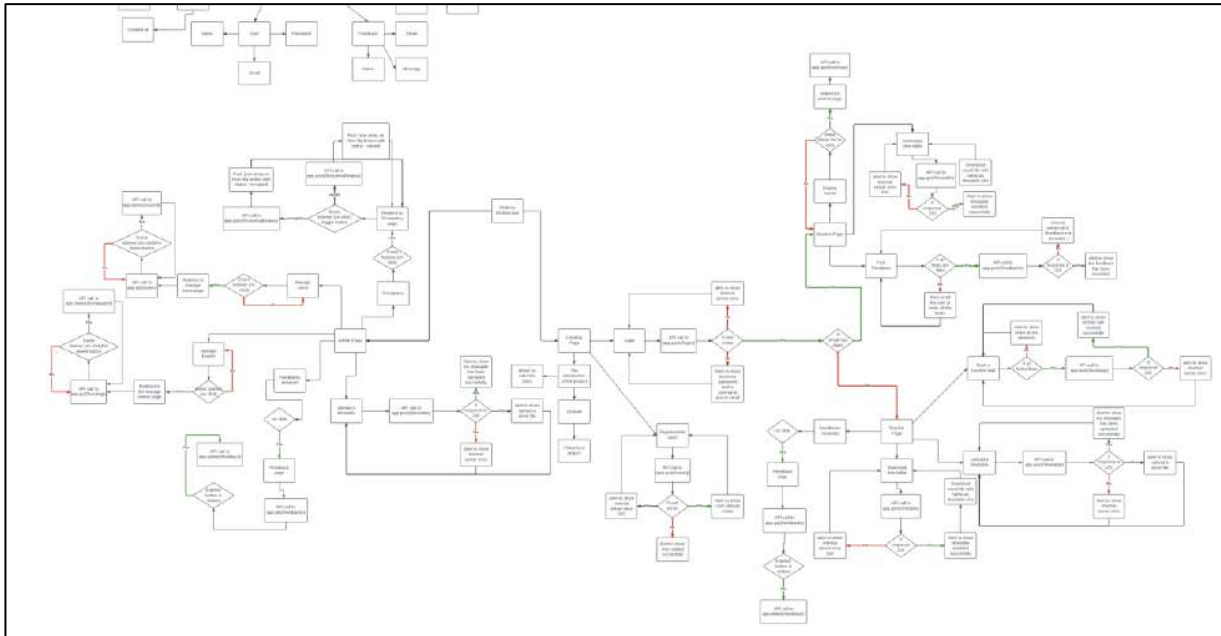


Figure 7 Screenshot of the whole architecture

Hardware Architecture/ Block diagram

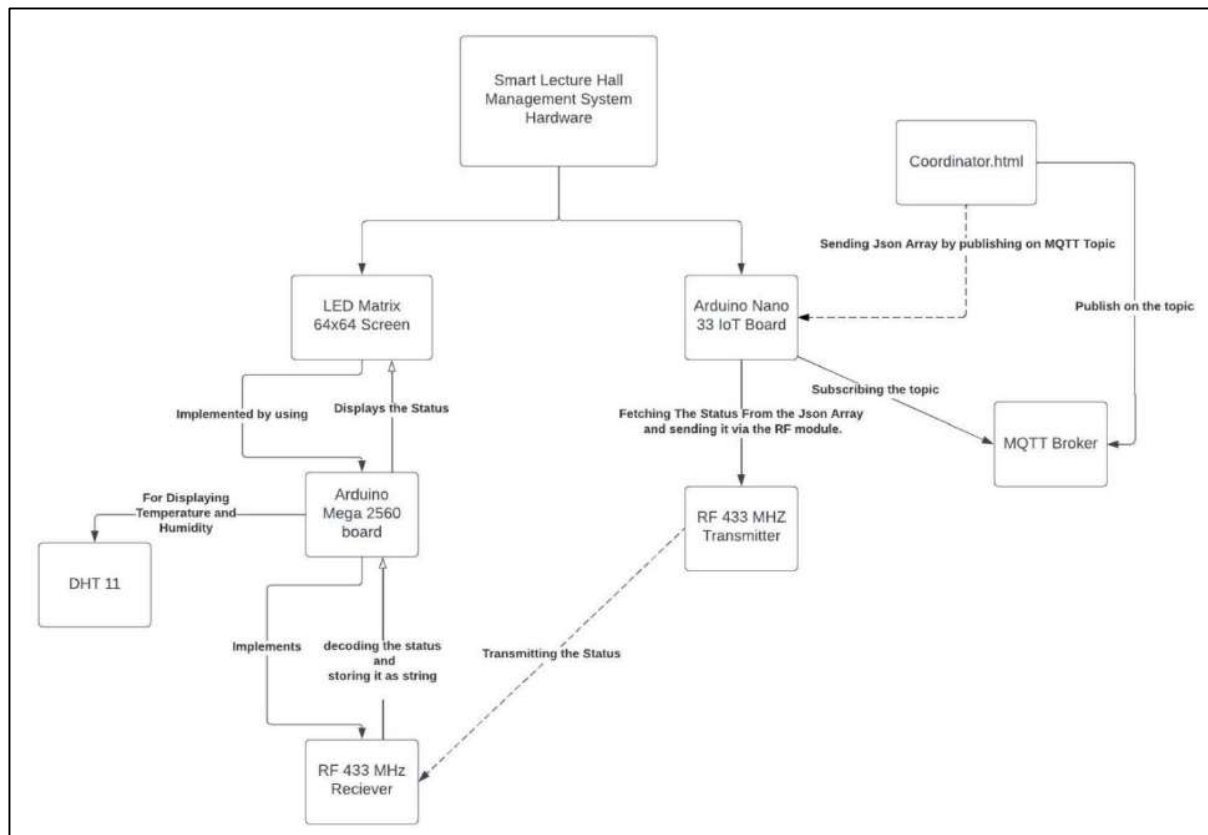


Figure 8 Hardware architecture

Schematic Diagrams of the hardware

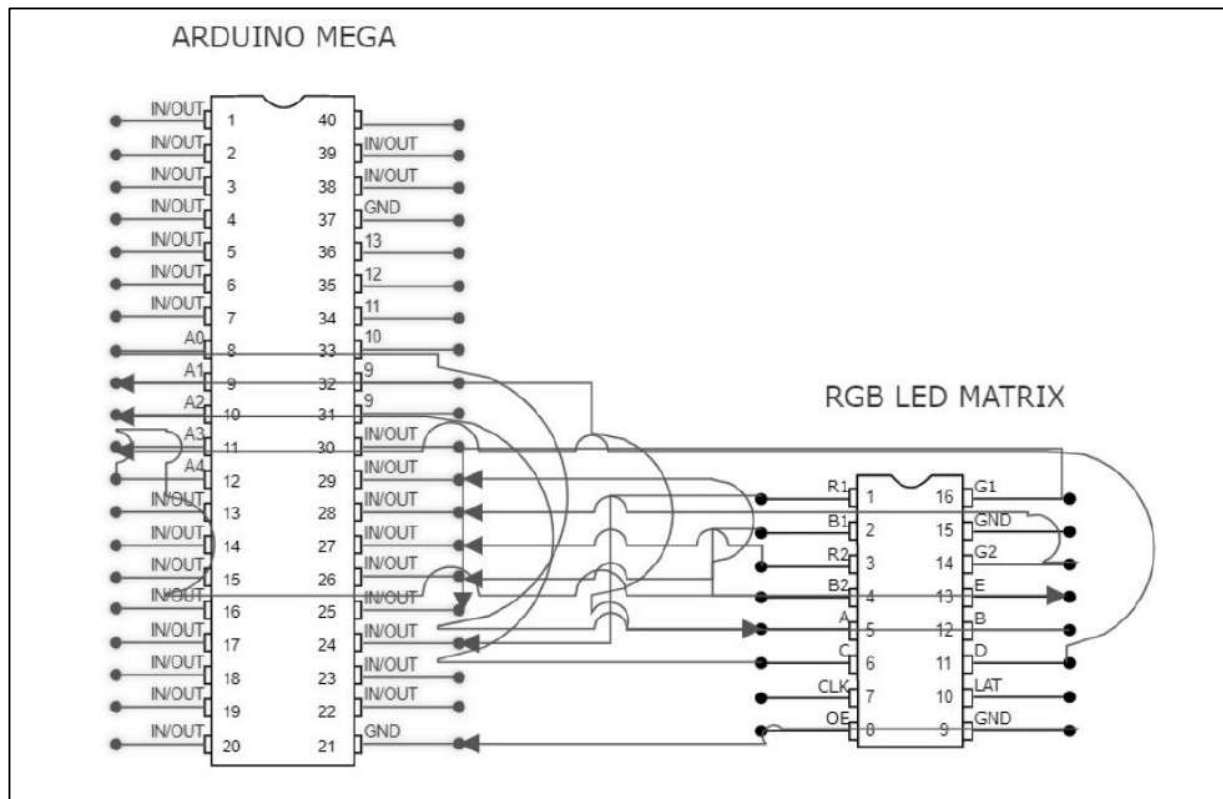


Figure 9 Schematic 1 of hardware

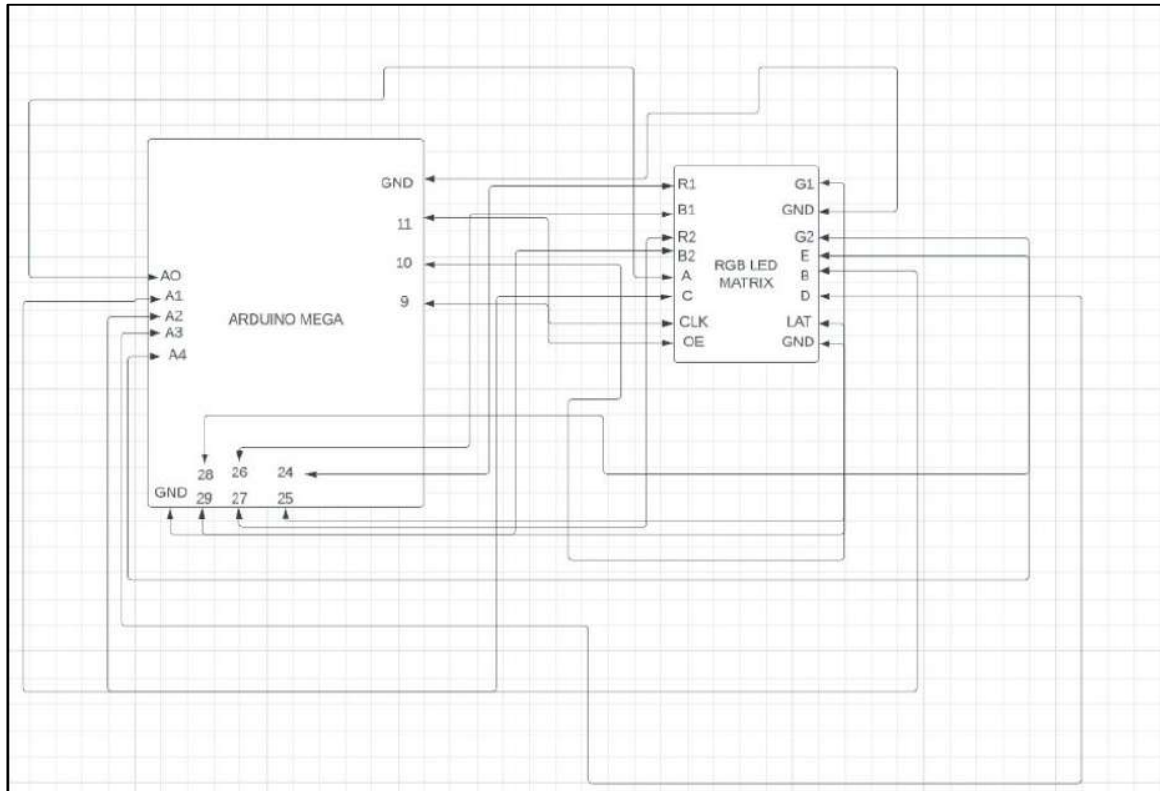


Figure 10 Schematic 2 of hardware

This is the schematic diagram for the connections between the Arduino Mega and RGB LED MATRIX HUB75 INTERFACE. The connections have been clearly displayed in the diagram.

Detailed description of the hardware architecture diagram:

- In the hardware architecture diagram, it could be clearly seen that the smart lecture hall management system contains two main components, the micro controllers are Arduino Mega 2560 and Arduino Nano 33 IoT board, the same has been indicated in the block diagram for the main system.
- The system implements a 64 x 64 LED matrix with 3 mm pitch having 4096 LED dots in total. Hence, this implementation required a lot of flash memory and Digital pins as it could only be interfaced using a HUB 75 interface, which requires 16 I/O pins, which was possible with Arduino Mega with 256Kb of flash memory and 54 digital I/O pins and 16 Analog pins.
- Further, as the Arduino Mega board does not have an integrated Wi-Fi, the Arduino Nano 33 IoT board was taken into account for the communication with the web.

application by the means of MQTT protocol, which works on the PubSub mechanism.

- Hence, in this case, the Arduino Nano 33 IoT board retrieves the status of the occupancy of the lecture hall by receiving a JSON array containing the status from the MQTT topic.
- After receiving the Status of the lecture hall, it passes it to the Arduino Mega board with an RF 433 MHz transmitter, which is received at the receiver module connected to the Arduino Mega and it is displayed onto the screen by implementing the RGBmatrixPanel.h library to display the status of the lecture hall in moving text format.
- Further, the DHT 11 sensor is connected to the Arduino Mega 2560 board to display the real time Temperature and Humidity on the screen for the respective lecture hall.

Hence, this is how the system is implemented to make it easier for the coordinator to change the occupancy of the lecture hall while displaying the temperature and humidity for the same.

Communication Diagram

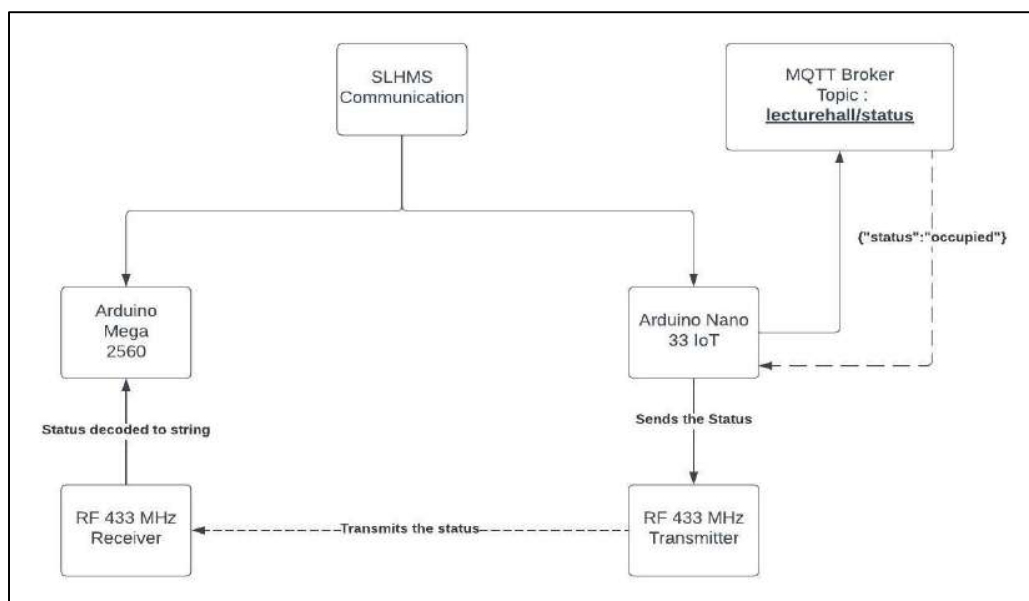


Figure 11 Communication diagram

Detailed description of the communications diagram:

The system implements the master to slave communication by use of the following:



- In the communications part, the main implementation is focussed via the use of the MQTT protocol, in which the Arduino Nano 33 IoT board connects to the MQTT broker by using the inbuilt WiFi Nina module.
- The MQTT broker sends the text for the JSON array containing the value of the status in the form of string, which is then converted to uint8_t by the inbuilt casting and then, it is transmitted to the Arduino Mega board by using the RF 433 MHz Transmitter module in the uint_8t format.
- The same is received by the Arduino Mega 2560 microcontroller by the implementation of the RF 433 MHz Receiver and then converted to String to display it onto the screen.

Hence, this was how communication was established with the Arduino Nano 33 IoT board as the master and the Arduino Mega 2560 as the Slave.

Features completed:


Projects / Smart Lecture Hall Management


SLHM Sprint 4


 

TO DO


IN PROGRESS 3 ISSUES


submitting a progress report.
 SLHM-9


final progress report with user instructions.
 SLHM-12


Submission of the entire project with the complete front end, back end and hardware components.
 SLHM-10


DONE 6 ISSUES ✓


Building schema for the sensor data and integrating the database with the back-end.
 SLHM-22 ✓


Exposing API to app.js and server.js(Event listeners and handlers)
 SLHM-20 ✓

Integration of hardware and MQTT with our system.
 SLHM-18 ✓

Deliver the entire user interface of the teacher.
 SLHM-7 ✓

Establishing the machine to machine communication with the system.
 SLHM-8 ✓

Make and deliver the API documentation along with the M2M communication.
 SLHM-11 ✓



| Feature | Client Sign-Off State | State | Notes |
|-------------------------------------|-----------------------|-------|---|
| Efficient booking of Lecture Halls | Signed off 2023-04-13 | Final | This feature has been implemented to allow teachers to book lecture halls. The list of bookings is readily visible, facilitating efficient management of the lecture halls and bookings |
| Capability to enable updates to the | Signed off 2023-04-25 | Final | Teacher/Coordinator can upload the excel sheet from any device |

| | | | |
|--|-----------------------|-------|--|
| timetable in excel format | | | which is then visible in the tabular form on the student interface. |
| Facilitate the uploading of event updates in the form of PDF files | Signed off 2023-04-18 | Final | We have incorporated a feature that allows for the upload of PDFs, such as posters or announcements, which can be displayed on the website for students to view. By implementing this feature, we seek to enhance the professionalism of the website and provide a platform for seamless dissemination of information to students. |
| Grievance/feedback portal | | Final | We have included a Grievance/Feedback portal as part of our project. This portal provides students and faculty with an avenue to voice their concerns or provide feedback on various aspects of the project. By incorporating this feature, we aim to create an inclusive and responsive environment that encourages open communication and continuous improvement. We are committed to addressing all grievances and feedback in a timely and efficient manner, thereby enhancing the overall user experience of the project. |

| | | | |
|--|-----------------------|-------|---|
| Dedicated portal for receiving and addressing grievances and feedback from users | Signed off 2023-04-24 | Final | We have implemented a dedicated portal within our full-stack project that is specifically designed for receiving and addressing grievances and feedback from users. This portal provides a structured and streamlined approach for users to submit their concerns or suggestions, which will be promptly reviewed and resolved by our team. By incorporating this feature, we aim to demonstrate our commitment to providing excellent user experience and to continuously improving the functionality and usability of our project. We welcome and encourage all feedback and grievances, as they serve as valuable inputs for enhancing the overall quality of our project. |
| Ability to Delete Users by Admin | Signed off 2023-04-27 | Final | We have included a feature that enables Admin users to delete other users within our project. This feature provides an additional layer of security and control, allowing Admin users to manage the user base more effectively. By incorporating this feature, we aim to ensure that the project is safe and secure for all users, while also providing greater flexibility and customization options for Admin users. We have implemented this feature in a way that is intuitive and easy to use, ensuring that Admin users can quickly and efficiently manage |

| | | | |
|---|-----------------------|-------|---|
| | | | user accounts as needed. |
| Ability to Delete and manage the events. | Signed off 2023-04-09 | Final | We have included a feature in our project that enables authorized users to delete and manage events. This feature provides greater flexibility and control over the events that are displayed on the website, allowing for easy modification or removal of events that are no longer relevant. By incorporating this feature, we aim to ensure that the events displayed on the website are up-to-date and relevant, thereby enhancing the overall user experience. Our system has been designed in such a way that authorized users can efficiently manage events with ease, thereby streamlining the event management process. |
| To toggle and display the occupancy of the lecture hall as per the requirements(For Admin). There is display of temperature and humidity in lecture hall. | Signed off 2023-04-24 | Final | We have implemented a feature within our project that allows for the displaying of lecture hall occupancy as per the requirements. This feature provides greater flexibility and customization options for users, enabling them to view the occupancy of the lecture hall to . By incorporating this feature, we aim to ensure that the lecture halls are utilized efficiently and effectively, while also providing users with the ability to adapt to changing circumstances or requirements. Our system has been designed in such a way that viewing the occupancy of lecture halls avoids disturbance during the lecture. In addition to this |

Artefacts List :

| Artefact Name | Artefact Type | Revision Number | Notes |
|---|---------------|-----------------|---|
| Final Source code | SRC code | 2023-04-29 | Will be able to run on any platform with appropriate dependencies installed (Mentioned in the setup guide) |
| Photos for the web page | Assets | 2023-04-30 | Provided in the github code itself. |
| Final Demonstration video of the project | Video | 2023-04-27 | The Drive Link or youtube video has been enclosed in the document itself |
| Final System architecture and flow chart. | Screenshots | 2023-04-28 | The screenshots and the explanation is enclosed in this document. |
| Final Schematics of the hardware used | Screenshots | 2023-04-26 | Schematics of the circuit are enclosed in this document itself |
| Hardware setup | Hardware | 2023-04-30 | The hardware setup working with display of occupancy temperature, humidity on the screen with the use of arduino mega, RF module, DHT-11 sensor |

Planned Work/Open Issues:

| Planned Feature | State | Sprint | Notes |
|---|--|--------|---|
| Login page | In Progress sprint 1 Completed in Sprint 2 | 1 | The back-end implementation of the login page was pending which was done in the sprint 2 |
| Integration of MQTT with hardware | In progress sprint 2 In progress sprint 3 Completed sprint 4 | 2 | The integration of MQTT was planned in the sprint 2 and sprint 3 and was done in sprint 4 |
| Time table and poster upload, grievance portal, | In progress sprint 3 Completed sprint 4 | 3 | These features were planned to be completed by sprint 3 but were completed in sprint 4 |

| | | | |
|--|--|---|--|
| booking of lecture halls | | | |
| Displaying of occupancy in lecture halls and temperature, humidity | In progress sprint 3 Completed sprint 4 | 3 | These features were planned to be completed by sprint 3 but were completed in sprint 4 |

Lessons Learned:

Lessons Learned in back end:
MongoDB Connection: Ensure that the application's MongoDB connection is configured and handled correctly. Take care of any potential connection issues and put in place the proper error-handling procedures.

Schema Design: Pay close attention to schema design and establish the necessary schemas for various data entities. Entity relationships should be taken into account when designing schemas.

Model construction: In order to communicate with the MongoDB collections, create models based on the established schemas. Implement data validation criteria and conduct CRUD (Create, Read, Update, Delete) actions using the models.

API Routes: Create and use the proper API routes to accommodate different client requests. Create routes using Express.js for various endpoints, such as authentication, data retrieval, data manipulation, etc.

Integrate with external services: such as MQTT brokers for real-time communication or other APIs for more functionality. Implement the logic required to connect to the MQTT broker, subscribe, and publish messages.

Testing: Use unit tests and integration tests to make sure the backend code is trustworthy and accurate. To automate the testing procedure, use testing frameworks such as Mocha or Jest.

Though we were unable to deploy the website as the services are paid and the free services fail to provide the appropriate build for our server to work. Instead, we were able to research about docker and how to containerise the application using it.

Containerization using Docker: Understand and apply Docker to containerize the application. It is simpler to deploy and manage applications across many environments thanks to Docker, which offers a consistent and isolated environment for the programme to execute in. Specify the environment and dependencies needed to run the application in a Dockerfile.

Create Docker images that contain the application and its dependencies depending on the Dockerfile.

We can combine the application container with a MongoDB container by using Docker Compose to define and manage multi-container applications.

Advantages of Docker: Recognise the advantages of utilizing Docker for development and deployment:

Better portability: Docker containers may be regularly deployed across many environments, which minimizes compatibility problems.

Modularization and Code Organisation: To make the code more readable and maintainable, organize it into independent modules or files.

Create logical modules in the code, such as routes, models, controllers, and middleware.

For better organization, divide the code pertaining to various entities or functionalities into various folders.

Improve the modularity of your code by managing dependencies with modules and import/export statements (CommonJS or ES modules).

Testing and Test-Driven Development (TDD): Adopt a testing strategy to make sure the application is reliable and robust.

Utilizing testing frameworks like Mocha, Chai, or Jest, create unit tests for specific functions, routes, and models.

When testing, think about simulating database interactions or external dependencies using a mocking library (like Sinon).

Implement integration tests to validate how the application's various components interact with one another.

To enable regular testing and early problem detection, incorporate test automation and continuous integration (CI) pipelines.

Middleware Usage: To handle routine activities and promote code reuse, use Express.js' middleware methods.

Use middleware, such as body-parser, to quickly parse request bodies.

Investigate other middleware choices, such as passport for authentication, multer for file upload management, and express-session for user session management.

To manage authentication, error handling, logging, and other overarching issues, create bespoke middleware functions.

Code Documentation :

To enhance maintainability and collaboration, keep your code documentation clean and succinct. The objective of functions or code blocks can be explained in comments, which can also be used to describe APIs.

If you want to generate API documentation automatically, think about utilizing a tool like JSDoc.

For every API endpoint, describe the formats for the requests and responses, how errors should be handled, and any headers or authentication that are necessary.

Lessons Learned in Hardware Implementation:

- **Implementation of the LED matrix 64x64 screen:**

The Hardware team first decided to use the conventional LCD screen for displaying the occupancy status of the lecture hall, which was then discarded due to the fact the team wanted to keep the overall implementation to be pocket friendly. So, the hardware team decided to go on with the 64 x 64 LED matrix screen to be implemented. The screen works on the HUB75 interface with 16 connections as shown in the schematic diagram. Moreover, it required a power input of 5v 4A. Also, it required a microcontroller with 256 kb of flash memory, so the hardware team tried to implement it with an ESP32 Dev module but the screen was showing fluctuations. So, the hardware team tried to implement it with the arduino mega 2560 board, which was able to solve the issue. The Hardware team leader implemented the code for the screen which used the RGBmatrixPanel.h library for display of contents.

- **Establishing communication by the use of Arduino Mega and ESP8266 Wi-Fi module:**

As the Arduino Mega 2560 microcontroller does not have an inbuilt wifi module, it could not connect to the internet for enabling the communication via MQTT. So, for that reason an external module such as ESP8266 was suitable for that reason. The hardware team tried communication with the ESP8266 module for connecting the board with the Wi-Fi by using the AT commands but the connection was quite uncertain as it was connecting and disconnecting again and again due to speed fluctuations.

- **Establishing communication by using two HC05 Bluetooth modules:**

After the uncertainty in connections caused due to the ESP8266 module, the back end team was now concerned about the communication. After this drawback, the hardware team leader came up with the idea of using the Arduino Nano 33 IoT board as a master to connect and subscribe to the MQTT topic to retrieve the status of the occupancy of the lecture hall. As it uses the WiFi Nina module, which shows stable connection, the Arduino Nano 33 IoT was finalized as the board for connection with the MQTT broker. Further, the hardware team leader tried the implementation for the communication between the Arduino Mega 2560 as the slave and the Arduino Nano 33 IoT as the Master. So, he configured two HC05 as the master and Slave By using the AT commands to transfer the status for the occupancy, which was completed at the end, but the system showed a lot of latency.

- **Establishing the communication with the use of 433 MHz RF module by using the RadioHead Library as the means for implementation:**

After the failure of using the HC05 bluetooth due to latency, the hardware team leader came up with the Idea of using the RF 433 MHz module, in which the transmitter was connected to the nano board and the receiver module was connected to the Mega 2560 board. The Implementation was made possible by the use of the RadioHead library for the Nano 33 IoT board and the Arduino Mega board. The Arduino Nano Connected to the MQTT broker and subscribed the mentioned topic , on the publish of message, it fetched the status and transmitted it to the arduino Mega in the int8_t format, which was then converted to string and displayed onto the LED Matrix 64x64 screen.

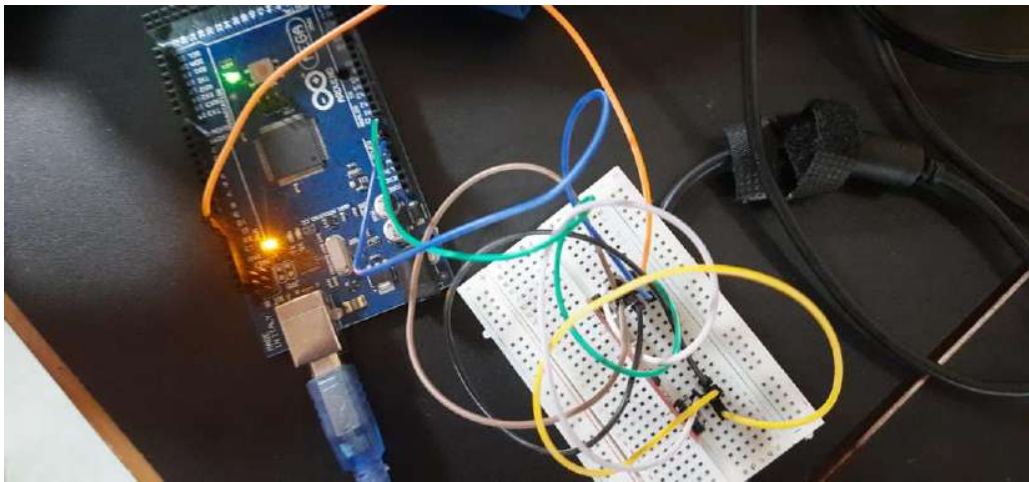


Figure 12 Hardware setup



Figure 13 Screen display (Hardware implementation)

This was how the status of occupancy along with the temperature and humidity was displayed on the LED matrix display.

Setup Guide

Prerequisites:

Install Node.js by downloading the most recent stable version for your operating system from the official Node.js website (<https://nodejs.org>). Observe the installation guidelines that are supplied.

Install MongoDB by downloading the most recent stable version of your operating system from the official MongoDB website (<https://www.mongodb.com>). Observe the installation guidelines that are supplied.

Git: If you haven't already, install Git. Follow the installation instructions after downloading it from the official Git website (<https://git-scm.com>).

Step 1: Clone the project

- Start a command prompt or terminal.
- Go to the directory where you wish to create your project by navigating there.
- Clone the project repository by using the following command:
- Git clone https://github.com/SakshamBehalexe/Project_IOT_SLHMS.git

Step 2: Set up the Backend (Node.js and Express.js)

- Open a terminal or command prompt.
- Navigate to the project's backend directory:
- Initialise the node packet manager in the directory by running the following command in the terminal:
- npm init
- Install the dependencies by running the following command:
- sudo npm install express mongoose body-parser express-session passport passport-local multer MQTT cors
- Start the back end server : npm start
- The API will be running on <http://localhost:5004>

Step 3: Set up the server

- Redirect to the web directory initialise npm by running the following command : npm init
- Install the dependencies required to run the server by running the following command in the terminal.
- sudo npm install express body-parser express-session passport cors
- Start the server : npm start
- The server should be running on <http://localhost:3000>

Step 4: Set up the MQTT server(API)

- Redirect to the MQTT directory and initialize the npm by running the following command:
npm init
- Install the dependencies required to run the server by running the following command in the terminal.
- npm install mongoose MQTT express body-parser cors
- The MQTT API should be running on <http://localhost:5008>

Step 5: Give power to the hardware:

- Connect the provided adapter with the power supply input provided at the back of the unit and provide the required power.

Step 6: Upload your credentials onto the board:

- Connect the micro USB cable provided to the port.
- Provide your SSID and Password for the connection of the hardware with the Wifi network by uploading the code provided in the resources with your SSID and password.
- Wait for the serial monitor to display the message for successful connection with the network and the broker.

Step 7: View the contents on the screen:

- Use the toggle button on the Web application to change the occupancy status on the screen along with the lecture hall name, temperature and humidity.

User Manual

For the User manual enclosed are the screenshots of the front end and the explanation of all the features and how to use them. Further, the description for the usage of the provided hardware is enclosed.

Starting with the Landing Page:

Our landing it consists of the introduction to the website and provides the user with features like registration and login which the user can fill accordingly.

Add User

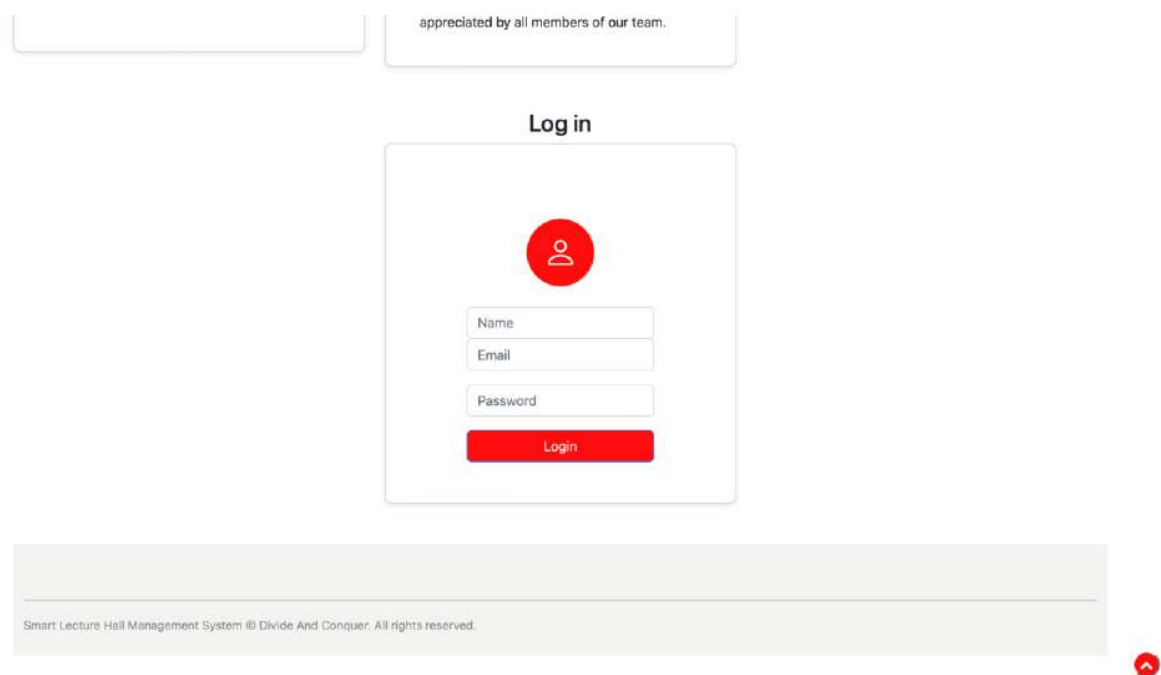


Figure 14 Screenshot of Landing page

According to our observations we have noticed that in our colleges teachers have no digits while on the other hand students have digits in their email address.

So the user will be redirected to the Teacher/Student according to their email address or the input received.

Student Page :

It introduces the following features :

Feedback form once filled and submitted both the teachers and the admin can access it and perform the required actions.

Download Time table once clicked on this button the user will be given access to the latest uploaded timetable and the corresponding file will be downloaded to their system

Display the time table : Even better if the user doesn't have space in their device or does not want to install an excel sheet we have the feature to display the timetable on the html page itself using javascript.

Display the Latest Events : When clicked on this button it will redirect the user to a new page and the call to api will be made to get the information of the 3 most recently uploaded data files and the bookings. The user will also have the option to download the pdf of corresponding pdf.

S.No First Name [Required] Last Name [Required] Uni-Roll no.

| | | | |
|----|-------------|-----------------|------------|
| 1 | Aaryan | Bansal | 2110994751 |
| 2 | Armaan | Chetal | 2110994755 |
| 3 | Arshpratap | Singh Somal | 2110994756 |
| 4 | Bhumika | Chauhan | 2110994757 |
| 5 | Charanpreet | Singh | 2110994758 |
| 6 | Garima | Arora | 2110994761 |
| 7 | Gaurish | Bhatia | 2110994762 |
| 8 | Gunjan | Sharma | 2110994763 |
| 9 | Harshdeep | Singh | 2110994764 |
| 10 | Jaiveer | Singh | 2110994765 |
| 11 | Kanika | Kukreja | 2110994766 |
| 12 | Kanishk | Jain | 2110994767 |
| 13 | Kashish | Bansal | 2110994768 |
| 14 | Khemraj | Verma | 2110994769 |
| 15 | Manav | Singh | 2110994770 |
| 16 | Mayank | Sharma | 2110994771 |
| 17 | Mehak | . | 2110994772 |
| 18 | Mistry | Samarth Dilpesh | 2110994773 |
| 19 | Mohnish | Sharma | 2110994774 |
| 20 | Naman | Garg | 2110994775 |
| 21 | Namit | Mittal | 2110994776 |
| 22 | Osheen | Sharma | 2110994778 |
| 23 | Pari | . | 2110994779 |
| 24 | Payas | Paul | 2110994780 |



Figure 15 Timetable displayed

Download Latest
Timetable

Download

[Click to Display the latest events](#)

Display

We value your



We value your Feedback!

Give your honest feedback and share if you have any grievances

Lecture Hall Bookings

| LH name | Teacher | Date | Description | Course name | PDF Download |
|------------|--------------|-----------|--|-------------|------------------------------|
| sport | saksham | 2/5/2023 | testing purpose | sit 202 | Download PDF |
| sportorium | Dr. Amit Sir | 2/5/2023 | this is for demonstration of the system | sit 202 | Download PDF |
| 5 | amit sir | 26/4/2023 | this is to demonstrate the working of this feature | sit 209 | Download PDF |

[Back to Student Page](#)

Teacher page:

It introduces the following features :



Download Time table once clicked on this button the user will be given access to the latest uploaded timetable and the corresponding file will be downloaded to their system

Display the Latest Events : When clicked on this button it will redirect the user to a new page and the call to api will be made to get the information of the 3 most recently uploaded data files and the bookings. The user will also have the option to download the pdf of corresponding pdf.

Upload the timetable: This feature prompts the user to upload the excel sheet in which timetable is included using multer it is stored in the mongo db and can be accessed using api any time.

Feedback manager : Located in the navbar when clicked on this button it redirects to another webpage for resolving the feedbacks. This can be used to see the information and details regarding the feedback of the system and the user can also click the delete button to delete the corresponding feedback entry once its resolved or if the user dosen't want to entertain it.

Book lecture hall : To use this feature the user is advised to fill all the fields regarding the event and upload the relevant pdf file. Once requested all the data will be saved to the corresponding db collection which can be accessed anytime till the api is running and the and the admin can further regulate or manage the events by deleting or checking the information.



Smart Lecture Hall Management System

Timetable Events Book Feedbacks
LH Received

Upload a file for timetable

Choose File no file selected

Download Latest Timetable

Download

Book a Lecture Hall

Lecture Hall no.

Teacher Name

Course

Explanation

Choose File

no file selected

Send request

Figure 18 Screenshot of Teacher Page

| Feedback Data | | | |
|---------------|----------------------|-------------------------|----------|
| Name | Email | Message | Resolved |
| John Doe | john.doe@example.com | This is a test feedback | Delete |
| John Doe | john.doe@example.com | This is a test feedback | Delete |
| John Doe | john.doe@example.com | This is a test feedback | Delete |
| John Doe | john.doe@example.com | This is a test feedback | Delete |
| John Doe | john.doe@example.com | This is a test feedback | Delete |
| John Doe | john.doe@example.com | This is a test feedback | Delete |

Figure 19 To address the feedbacks

Administrator page :

It introduces the following features :

Download Time table once clicked on this button the user will be given access to the latest uploaded timetable and the corresponding file will be downloaded to their system

Display the Latest Events : When clicked on this button it will redirect the user to a new page and the call to api will be made to get the information of the 3 most recently uploaded data files and the bookings. The user will also have the option to download the pdf of corresponding pdf.

Upload the timetable: This feature prompts the user to upload the excel sheet in which timetable is included using multer it is stored in the mongo db and can be accessed using api any time.

Feedback manager : Located in the navbar when clicked on this button it redirects to another webpage for resolving the feedbacks. This can be used to see the information and details regarding the feedback of the system and the user can also click the delete button to delete the corresponding feedback entry once its resolved or if the user dosen't want to entertain it.

Manage events: This feature redirects the user to another page and gives the option to either delete or inspect the information of the all the events by extracting the data from the mongo db by giving a call to the corresponding api endpoint and displaying it on the page.

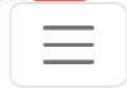
Manage Users: This feature allows the administrator to display the information of the users on a different webpage and it has a delete option to manage the users which is if the user data is incorrect or the user no longer is in the college the admin can delete their record from the corresponding collections in the data base.

Occupancy manager: This feature allows the admin to change the occupancy of the lecture hall by pressing the toggle button which will make an api call to post the changed status of the occupancy either to vacant or unoccupied on the MQTT broker (hive MQ in our case) which will be accessed by the hardware to display Vacant or occupied on the RGB LED matrix (each lecture hall will have their own screen to display the information).

S.No First Name [Required] Last Name [Required] Uni-Roll no.

| | | | |
|----|-------------|-----------------|------------|
| 1 | Aaryan | Bansal | 2110994751 |
| 2 | Armaan | Chetal | 2110994755 |
| 3 | Arshpratap | Singh Somal | 2110994756 |
| 4 | Bhumika | Chauhan | 2110994757 |
| 5 | Charanpreet | Singh | 2110994758 |
| 6 | Garima | Arora | 2110994761 |
| 7 | Gaurish | Bhatia | 2110994762 |
| 8 | Gunjan | Sharma | 2110994763 |
| 9 | Harshdeep | Singh | 2110994764 |
| 10 | Jaiveer | Singh | 2110994765 |
| 11 | Kanika | Kukreja | 2110994766 |
| 12 | Kanishk | Jain | 2110994767 |
| 13 | Kashish | Bansal | 2110994768 |
| 14 | Khemraj | Verma | 2110994769 |
| 15 | Manav | Singh | 2110994770 |
| 16 | Mayank | Sharma | 2110994771 |
| 17 | Mehak | . | 2110994772 |
| 18 | Mistry | Samarth Dilpesh | 2110994773 |
| 19 | Mohnish | Sharma | 2110994774 |
| 20 | Naman | Garg | 2110994775 |
| 21 | Namit | Mittal | 2110994776 |
| 22 | Osheen | Sharma | 2110994778 |
| 23 | Pari | . | 2110994779 |
| 24 | Payas | Paul | 2110994780 |





Smart Lecture Hall Management System

TimeTable

Events

Feedback Received

Manage Events

Manage Users

Occupancy

[Click to Display the latest timetable](#)

Display

Download

[Click to Display the latest events](#)



Figure 20 Responsive webpages

User List

| Name | Email | Password | |
|---------|------------------------|----------|--------|
| qwe23 | qwt@example.com | qwt1 | Delete |
| s | sakshambehla@gmail.com | s | Delete |
| Saksham | test@gmail.com | s | Delete |

DEAKIN

UNIVERSITY

CHITKARA

UNIVERSITY

Smart Lecture Hall Management System

TimeTable

Events

Feedback Received

Manage Events

Manage Users

Occupancy

Click to Display the latest timetable

DisplayDownload

Click to Display the latest events

Display

Smart Lecture Hall Management System © Divide And Conquer. All rights reserved.

Figure 21 Teacher page functionalities

List of Bookings

| LH | Teacher Name | Course | Explanation | Actions |
|------------|--------------|---------|--|------------------------|
| sport | saksham | sit 202 | testing purpose | Delete |
| sportorium | Dr. Amit Sir | sit 202 | this is for demonstration of the system | Delete |
| 5 | amit sir | sit 209 | this is to demonstrate the working of this feature | Delete |

[Back to Student Page](#)

[Go back to teacher page](#)

[Go back to admin page](#)

Lecture Hall 5 Lecture Hall 2

Toggle Status (occupied) ☒



Success

Status updated successfully

OK

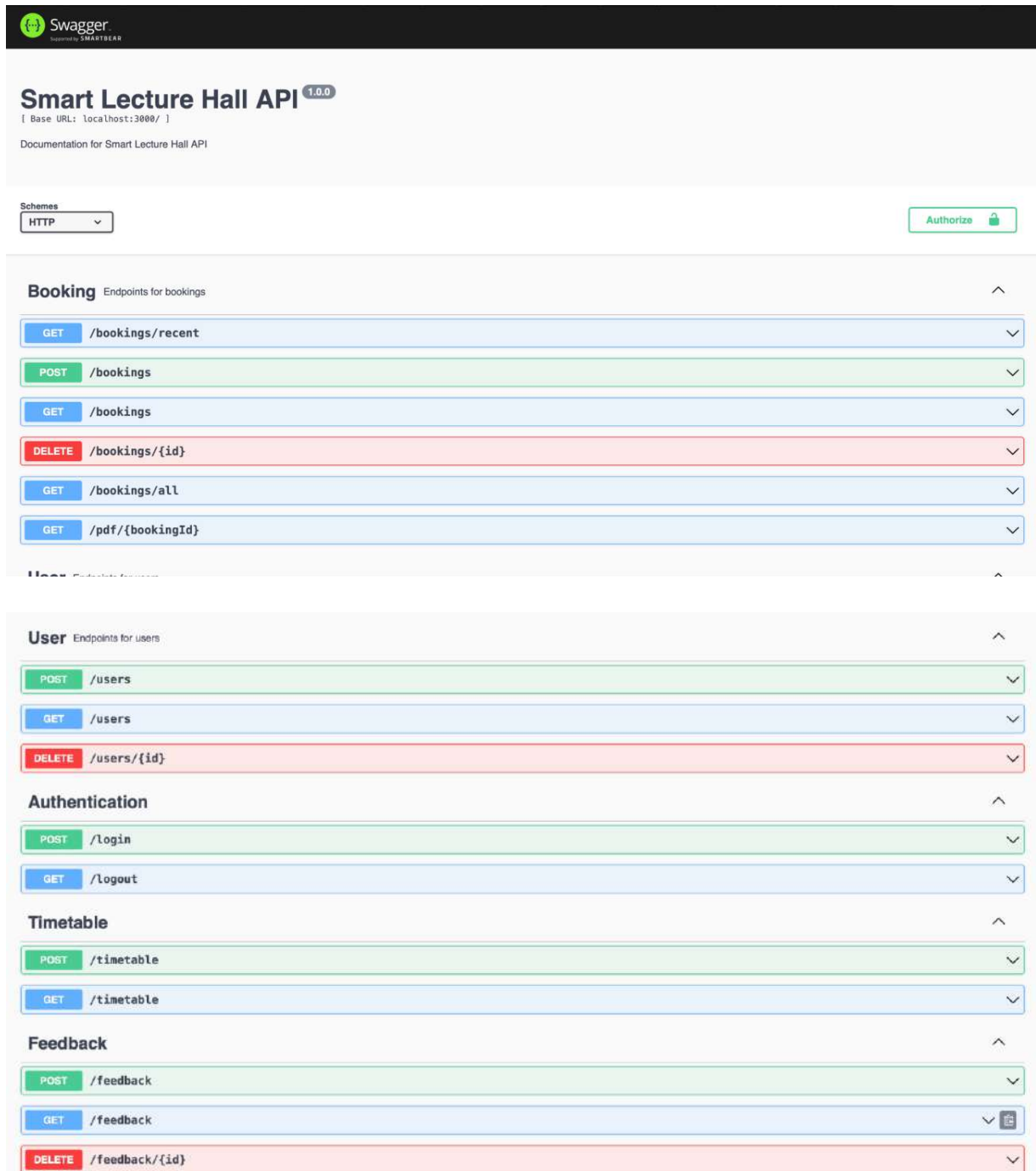
Hardware:

- **Connect the hardware unit with power:** use the power cord provided to give power to the hardware unit to supply the power to the components.
- **Connect the micro USB cable to the slot at the back of the unit** and upload the code provided with the use of Arduino IDE to provide the SSID and Password for the connection of the hardware to the network.
- **Make sure that you have a fast and working internet connection** with a minimum speed of 1 MBPS.
- **Upon successful connection**, you will be able to see the successfully connected message on the serial monitor.
- Now, you will be able to see the lecture hall number, the occupancy status(which is synced with the toggle button) on the screen along with the temperature and the humidity of the lecture hall.

This was all about the setup for the hardware.

API Documentation:

For the Documentation of API we have used swagger screenshots of which are as follows and the code for it is enclosed in our project github repository itself.



The image is a screenshot of the Swagger UI for the 'Smart Lecture Hall API' (version 1.0.0). The base URL is 'localhost:3000/'. The documentation is for the 'Smart Lecture Hall API'. The 'Schemes' dropdown is set to 'HTTP'. There is an 'Authorize' button. The API is organized into several sections: 'Booking', 'User', 'Authentication', 'Timetable', and 'Feedback'. Each section contains a list of endpoints with their respective HTTP methods (GET, POST, DELETE) and paths. The endpoints are color-coded: green for POST, blue for GET, and red for DELETE. Each endpoint has a dropdown arrow on the right side.

Swagger
powered by SMARTLEARN

Smart Lecture Hall API 1.0.0
[Base URL: localhost:3000/]
Documentation for Smart Lecture Hall API

Schemes: HTTP Authorize

Booking Endpoints for bookings

- GET /bookings/recent
- POST /bookings
- GET /bookings
- DELETE /bookings/{id}
- GET /bookings/all
- GET /pdf/{bookingId}

User Endpoints for users

- POST /users
- GET /users
- DELETE /users/{id}

Authentication

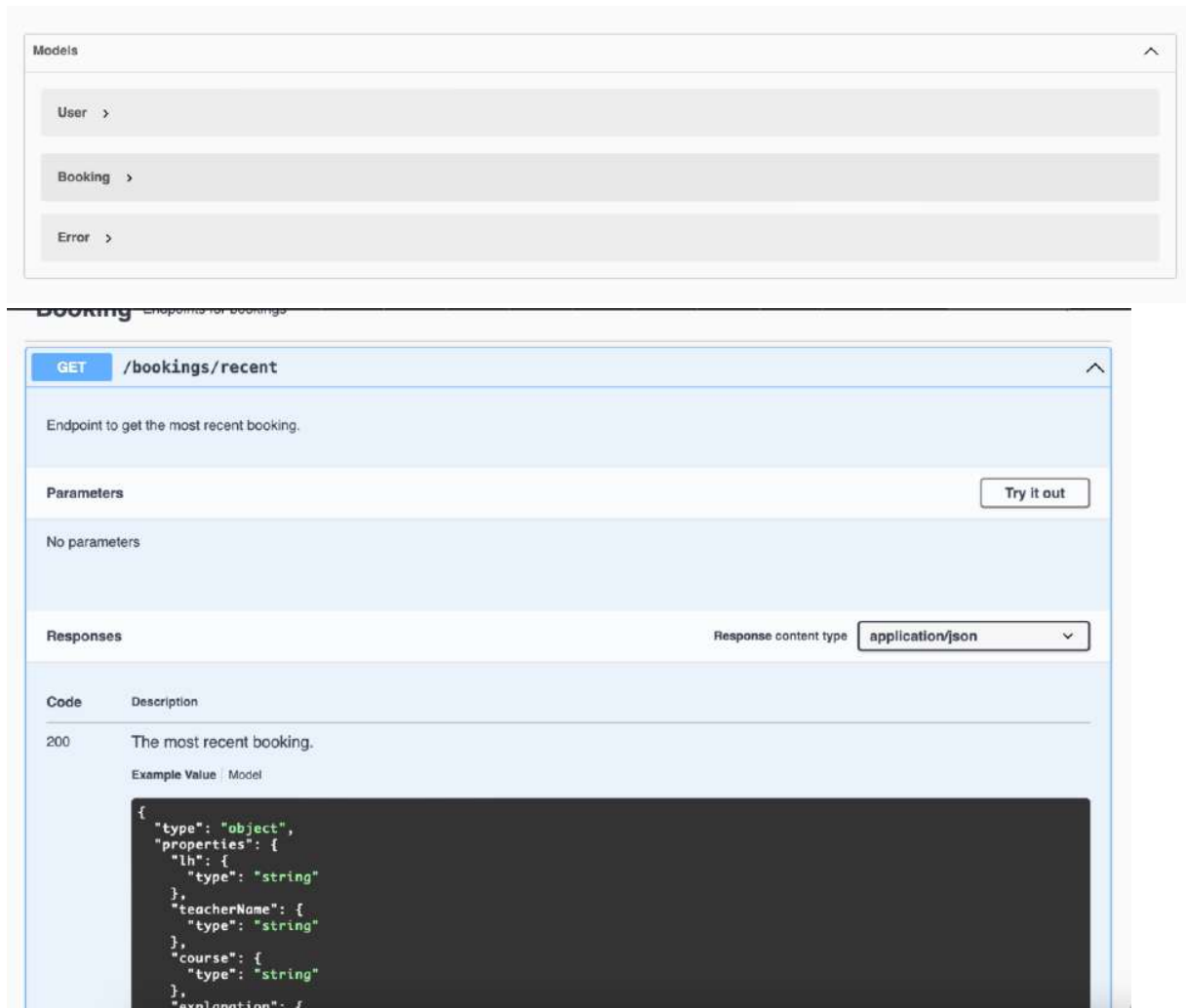
- POST /login
- GET /logout

Timetable

- POST /timetable
- GET /timetable

Feedback

- POST /feedback
- GET /feedback
- DELETE /feedback/{id}



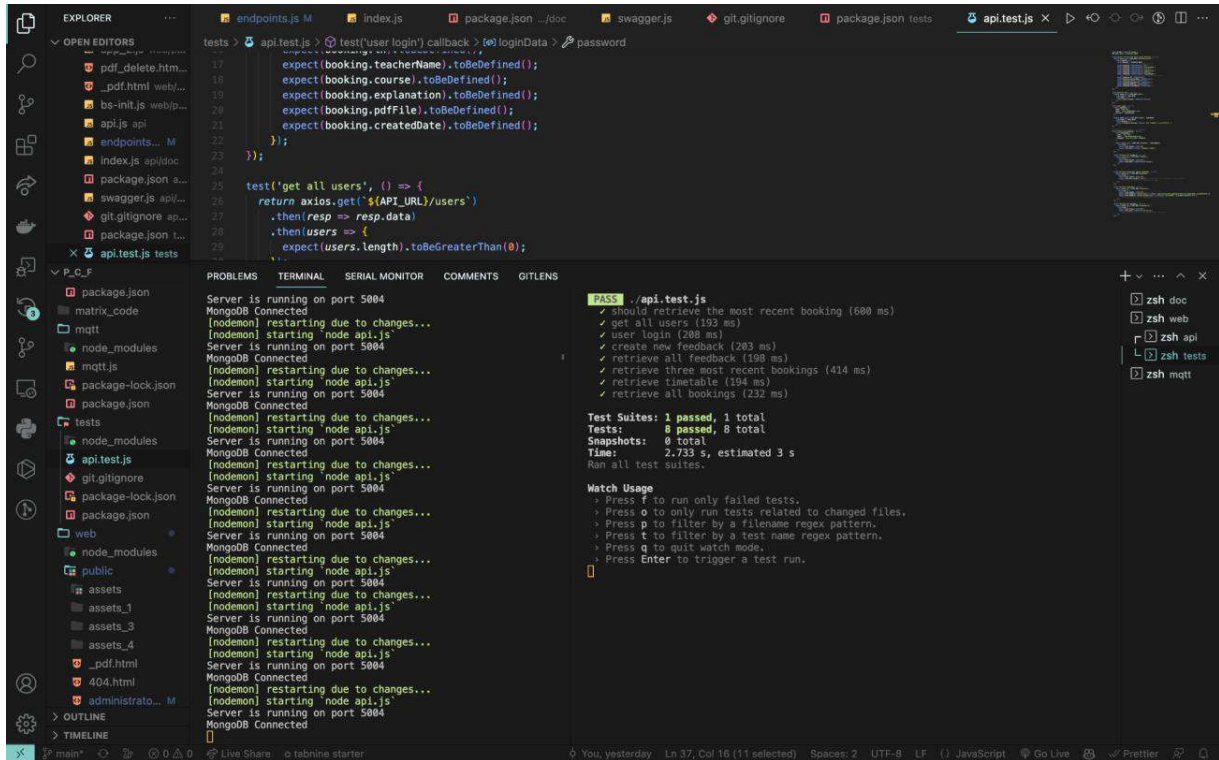
Screenshot of one of the endpoints with the annotations as an example how we have used swagger:

```
app.get('/bookings/recent', async (req, res) => {
  try {
    // #swagger.tags = ['Booking']
    // #swagger.description = 'Endpoint to get the most recent booking.'
    /* #swagger.responses[200] = {
      schema: { $ref: "#/definitions/Booking" },
      description: 'The most recent booking.'
    } */
    const booking = await Booking.findOne({}).sort({ createdAt: -1 }).exec();
    res.json(booking);
  } catch (err) {
    console.log(err);
    /* #swagger.responses[500] = {
      description: 'Error retrieving booking from database.'
    } */
    res.status(500).send('Error retrieving booking from database.');
```

API Testing:

The code for the Testing is in the repository mentioned at the end of the document (project git repo)

The screenshots showing its working are as follows.



Examples of some of the tests:

```
3
4 test('should retrieve the most recent booking', () => {
5   return axios.get(`${API_URL}/bookings/recent`)
6     .then(response => {
7       const booking = response.data;
8       // Perform assertions on the booking object
9       expect(booking).toHaveProperty('lh');
10      expect(booking).toHaveProperty('teacherName');
11      expect(booking).toHaveProperty('course');
12      expect(booking).toHaveProperty('explanation');
13      expect(booking).toHaveProperty('pdfFile');
14      expect(booking).toHaveProperty('createdDate');
15      // Additional assertions on specific properties/values
16      expect(booking.lh).toBeDefined();
17      expect(booking.teacherName).toBeDefined();
18      expect(booking.course).toBeDefined();
19      expect(booking.explanation).toBeDefined();
20      expect(booking.pdfFile).toBeDefined();
21      expect(booking.createdDate).toBeDefined();
22    });
23 });
24
25 test('get all users', () => {
26   return axios.get(`${API_URL}/users`)
27     .then(resp => resp.data)
28     .then(users => {
29       expect(users.length).toBeGreaterThan(0);
30     });
31 });
32
33 test('user login', () => {
34   const loginData = {
35     name: 'John Doe',
36     email: 'john.doe@example.com',
37     password: 'password123'
38   };
39
```

Meeting minutes:

- During Sprint 1, the team focused on implementing the front end, conducting research, initializing the back end, and creating an API to handle mongo. The team completed the front-end implementation, kept the sprint management up to date, and created a MongoDB cluster while conducting research on data management.

The team planned to work on the user interface for teachers, complete the application code, implement the API, and define various endpoints to handle mongo DB during the next sprint. Proposed scope amendments included placing more emphasis on time management, focusing strongly on the fundamentals and system architecture, and ensuring flexible communication with the client for any necessary changes.

To achieve these goals, Team Divide and Conquer aimed to improve team coordination and ensure effective communication among team members.

- During Sprint 2, the team focused on completing the front-end development, implementing the IoT authentication system for users, designing the back-end API architecture, developing the user interface for teachers and students, connecting the MongoDB database to the back-end API, and integrating a hardware display screen. In the previous meeting held on March 20, 2023, the team had decided to complete the Initial-page, add API endpoints for user management, and manage the sprint and backlog on JIRA.

The team made several decisions during this sprint, including completing the user interface, implementing the back-end API endpoints for IoT authentication, and researching the most efficient ways to manage the database. The Sprint Increment Report was prepared by Dr. Amit Pandey, and the team consisted of Mehak (2110994772), the supervisor, Armaan Chetal (2110999455), the team leader, and Saksham Behal (2110994801). Gaurish Bhatia (2110994762) and Krish (2110994848) reported the progress of the sprint, which focused on front-end implementation for the user interface of teachers and students and the API for IoT authentication.

The outcome of Sprint 2 included the completion of the front-end development for teachers and students, implementation of the back-end API using Postman, updating the sprint management on the JIRA cluster, data management on the MongoDB database, finalizing the system architecture, and achieving the goals set in the previous meeting without creating new backlogs. In the upcoming activities, the team planned to work on machine-to-machine communication using the MQTT protocol (hivemq), integrate and finalize the hardware, update data based on sensor readings, add event listeners and handlers in app.js for animations, and update the progress report and backlog management.

- During sprint 3, the team focused on integrating hardware and MQTT with the system, animating the front-end using basic JavaScript, exposing the API to the script file for the

application and server.js, finalizing and ordering the necessary hardware components, building the schema for sensor data, integrating the database with the back-end, and collectively working on documenting the system. The back-end team completed the integration of hardware into the system using the Message Queuing Telemetry Transport Protocol, while the front-end team introduced basic animation on the webpage with JavaScript. The team ordered the required parts, including the screen and microcontroller, and agreed to continuously update the system architecture flow chart and documentation.

This sprint focused on designing the webpage and implementing features such as PDF converter, connecting MQTT topics and back-end, exposing the MQTT API and api.js to the server side of the code, adding animation to the client-side of the code, finalizing the hardware components required to implement the smart lecture hall management system, and integrating the hardware with the system using the MQTT protocol.

The upcoming sprint will focus on completing the set goals, without creating any backlogs. Activities will include integrating hardware components with the system, implementing animation in web pages, finalizing and ordering hardware components, implementing the schema for sensor data, integrating the back-end with the MongoDB database, using MQTT to establish machine-to-machine communication, updating system architecture with more features, and implementing previous ones.

Proposed amendments to the project scope include effectively managing time to complete the project on the proposed timeline, improving coordination among team members, focusing on fundamentals and system architecture, assessing the feasibility of proposed features, evaluating and testing individual features, and completing documentation, testing, and deployment. The team will also focus on implementing hardware components, adding insights to documentation, managing each sprint accordingly, communicating effectively, and delivering set goals on time.

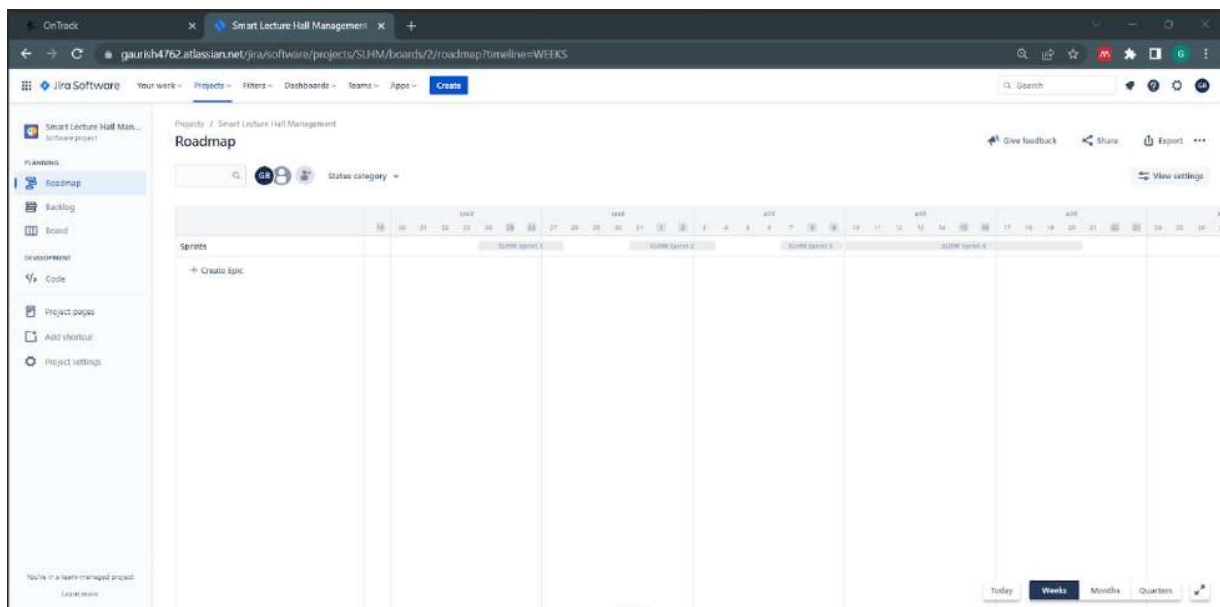
- During Sprint 4, the team consisted of Saksham Behal (211099801), Team Leader Gaurish Bhatia (2110994762), Supervisor Mehak (2110994772), Armaan Chetal (2110994755), and Krish (2110994848). The main agenda for this sprint was the integration of hardware and MQTT, integration of front and back end, documentation of API endpoints, and presentation slide preparation. During the meeting, the team discussed completion of hardware using the Arduino mega and 64*64 LED matrix screen, integration of front and back end, completion of coordinator front end page, and integration of the front and back end. Decisions were made to introduce basic animation in the web page, integrate the front end with the back end by the end of the sprint, and research more about the screen and the microcontroller for proper implementation of the display.

The action items for this sprint included integration of hardware and MQTT with the system, building the Schema for the sensor data and integrating the database with the back end, and building the Schema for the sensor data. The most important details in this text are the results of the recent sprint, which focused on completing the hardware and integrating it with MQTT along with the integration of back end with front end. The outcome of the sprint was the completion of

the hardware implementation in the system, integration of front and back end, and completion of the front-end part with animation.

Proposed amendments to the scope included assessment of the completion time for the project, effective management of the time to complete the project on the proposed timeline, and evaluation and testing of the individual features of both front and back end. The team will continue to focus on the implementation of the hardware components and its integration with MQTT, continuously add more insights in the documentation, and add more features. The team will also continue to deliver the set goals on time, manage each sprint accordingly, and effectively communicate with each team member. Finally, the team will stop adding unreasonable/time-consuming attributes to the project.

Sprint Timeline:



GitHub repository link :

https://github.com/SakshamBehalexe/Project_IOT_SLHMS

Link for the system architecture:

(initial) <http://surl.li/qwawe>

(Final) <http://surl.li/qwavs>

Link for hardware and communication architecture/ diagram:

https://lucid.app/lucidchart/3056f22c-603f-49b4-a8d3-3c6c3675a466/edit?invitationId=inv_5c82ec45-c188-45fb-9790-8885068ac048&page=0WrVs00e2Vy_#

https://lucid.app/lucidchart/3056f22c-603f-49b4-a8d3-3c6c3675a466/edit?invitationId=inv_5c82ec45-c188-45fb-9790-8885068ac048&page=UEMVFhz1W1lz#

Google drive link for project demonstration video:

https://drive.google.com/drive/folders/14kqAvagsjbUyxZSVqLw0RkfXzMXMLF_b?usp=sharing

Sprint Timeline Diagram:

https://drive.google.com/drive/folders/1s3H1RdiutCC_41tjuCXXg76Q-XqZBP6C?usp=sharing

Sign-off

We, the team members of Divide and Conquer (Saksham Behal, Gaurish Bhatia, Mehak, Krish, Armaan Chetal), have included all relevant material which is agreed to be included in this handover. If an artefact is not included, it is stipulated in the Planned Work section, or artefacts list.

Date: 06/05/2023

Signed: Saksham Behal, Gaurish Bhatia, Mehak, Krish, Armaan Chetal