CENG-322
Deliverable 2

Team Name: TBD

Project Name: Smart Library Study Room Management and Comfort System

Group: 2

Members:
Mathew Anderson-Saavedra N01436706
Nicole Chlea Manaoat N01565017
Medi Muamba Nzambi N01320883
Safah Virk N01596470

Table Of Contents

Members Info And Participation

Name	Student ID	Github ID	Signature	Effort
Medi Muamba Nzambi	N01320883	MediMuamba0883	Signed by MediMuamba	90
Mathew Anderson-Saavedra	N01436706	MathewAnderson6706	Signed by Mathew A	100
Nicole Chlea Manaoat	N01565017	NicoleManaoat5017	Signed by NicoleChleaManaoat	90
Safah Virk	N01596470	Safahvirk6470	Signed by Safah Virk	90

Project Scope and Goals

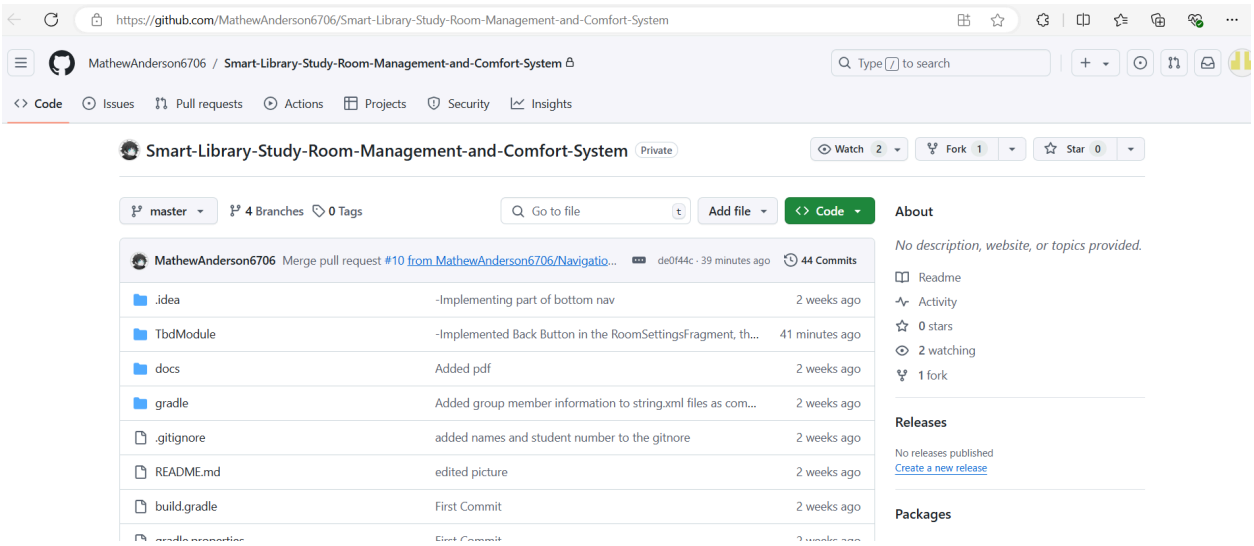
The Scope of this project is to create an easy-to-understand UI, viewing different buildings and accessing different rooms with a variety of different settings to change. The objects in the app will be constantly updated by the database so the user can have the most up-to-date information about the availability of rooms, and different information on the rooms themselves. The goal of this project is to make a well developed and designed app to show off our skills and maybe get this implemented in colleges.

GitHub Repo Link and Strategy

GitHub Repo Link:

https://github.com/MathewAnderson6706/Smart-Library-Study-Room-Management-and-Comfort-System.git

Verify the link is working



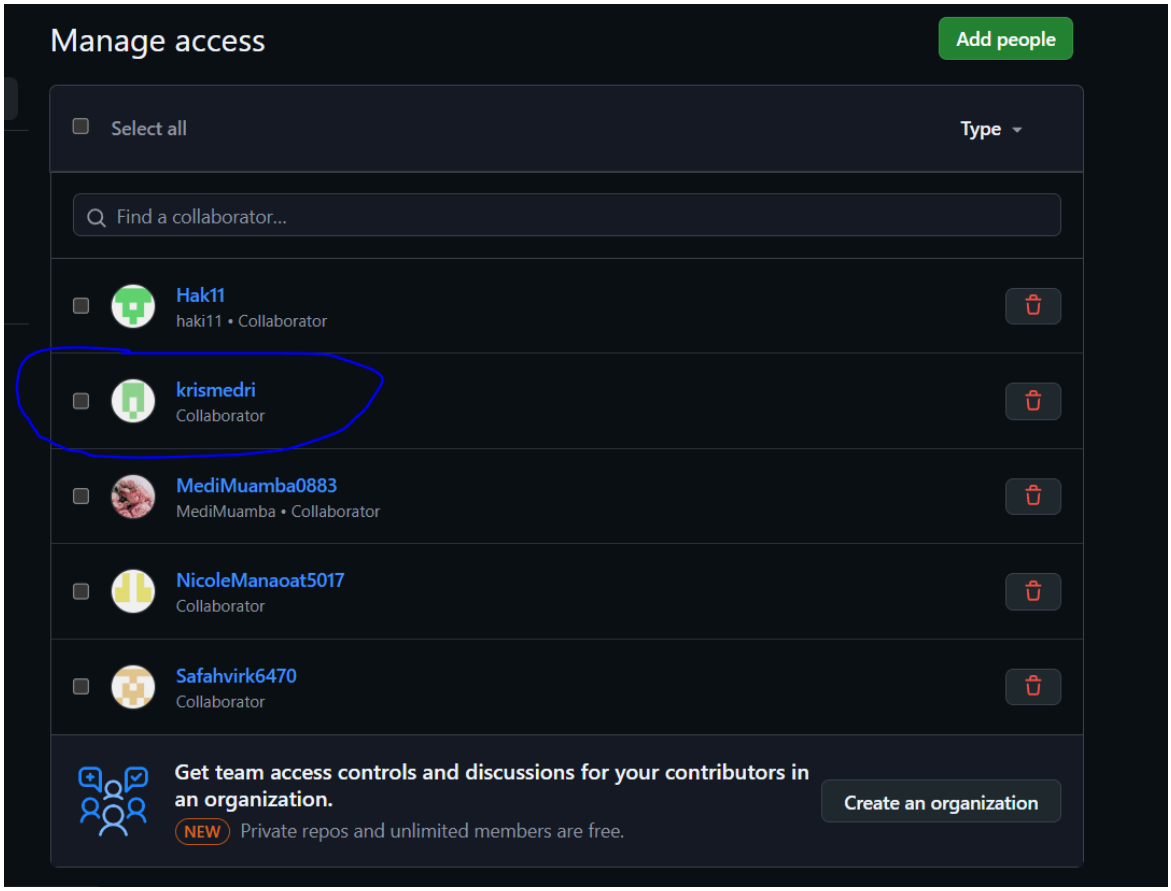
Explain the github strategy

1. Collaboration: This is the first thing to do. Someone of our members created a repository and invited us as collaborators. After this you can verify if the link is working.
2.Cloning the Repository: Each collaborator clones the repository using the repository link to get the existing files and code on our local machines.
3.Branching for Development: We create separate branches to start working on our individual tasks. This allows us to safely develop code without affecting the main codebase on the master/main branch. Each of us works within our own branch.
4.Committing and Pushing Changes: Once we've made progress, we commit our changes and push them to our respective branches. Each branch only contains the changes specific to the work done by the individual team member.
5.Merging Changes to the Master/Main Branch: After development is complete, we need to merge the changes back into the master branch.

- **Method: Pull Request:** We can create a pull request on GitHub. This is a request to merge our work into the master/main branch. A collaborator will review the changes and, if everything looks good, they or you can approve and confirm the merge. Once approved, the changes will be added to the master branch.

6.Getting Updates on the Local Machine: Once the changes are merged into the master/main branch, you may want to pull the latest changes to your local machine. In Android Studio, this can be done by clicking on the Git menu and selecting Pull or Update Project. This ensures your local code is in sync with the latest version on GitHub.

GitHub invitation for Hardware Professor



Sprint Dashboard

Develop Room Availabilty

Task	Owner	Status	Due date	Priority	Timeline
Implement Building selecti...		Done	~~Oct 8~~	Medium	~~Oct 7 - 8~~
Connect Buildings to Roo...		Done	~~Oct 8~~	High	~~Oct 7 - 8~~
Add Green/Red images to ...		Working on it	Oct 8	Low	Oct 7 - 8
Implement 4 fragments w...		Done	~~Oct 8~~	High	~~Oct 6 - 8~~
Add basic details to the ro...		Working on it	Oct 8	Low	Oct 8
+ Add task					
			Oct 8		Oct 6 - 8

Database

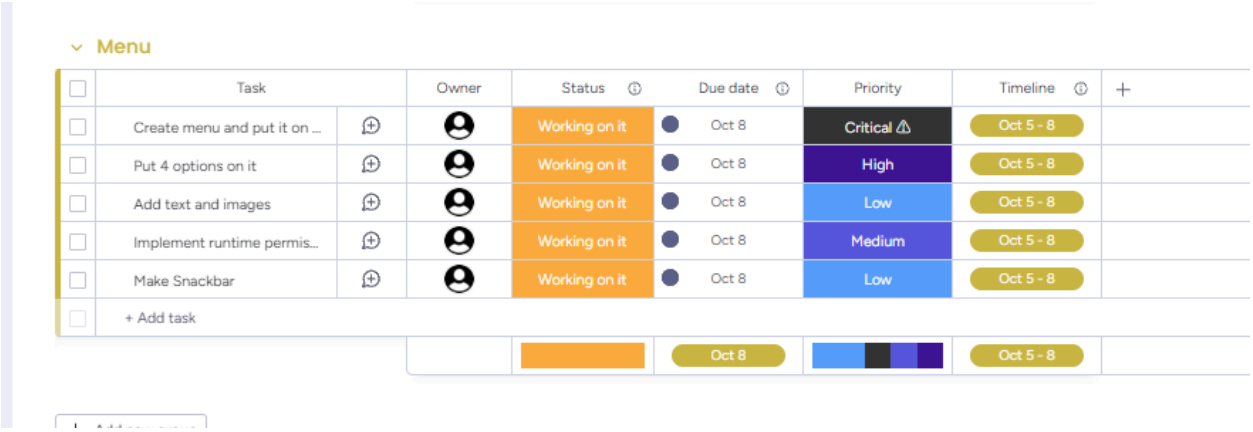
Task	Owner	Status	Due date	Priority	Timeline
Create Database		Done	~~Oct 8~~	Critical ⚠	~~Oct 4 - 8~~
Implement Temperature/ai...		Done	~~Oct 8~~	Medium	~~Oct 4 - 8~~
Implement Light		Working on it	Oct 8	Medium	Oct 4 - 8
Implement Timer		Working on it	Oct 8	Medium	Oct 4 - 8
Implent Check In/Check O...		Working on it	Oct 8	Medium	Oct 4 - 8
Implement Door Opener/C...		Working on it	Oct 8	Medium	Oct 4 - 8
+ Add task					
			Oct 8		Oct 4 - 8

User Authentication

Task	Owner	Status	Due date	Priority	Timeline
Allow occupied rooms to b...		Done	~~Oct 8~~	Low	~~Oct 7 - 8~~
Create a passcode for test...		Done	~~Oct 8~~	Low	~~Oct 7 - 8~~
Allow user to type in a cod...		Done	~~Oct 8~~	Low	~~Oct 7 - 8~~
Transfer User to room setti...		Done	~~Oct 8~~	Low	~~Oct 7 - 8~~
Implement changing passc...		Not Started	Oct 15	Low	Oct 12 - 15
+ Add task					
			Oct 8 - 15		Oct 7 - 15

Documentation

Task	Owner	Status	Due date	Priority	Timeline
Create document and for...		Done	~~Oct 4~~	Medium	~~Oct 4~~
Task 10-13, 21		Done	~~Oct 6~~	Medium	~~Oct 6~~
Task 27 & 28		Working on it	Oct 8	Medium	Oct 8
Task 22,29,30,31		Working on it	Oct 8	Medium	Oct 8
Task 9, 16-20		Working on it	Oct 8	Medium	Oct 8
+ Add task					
			Oct 4 - 8		Oct 4 - 8



DoD Criteria

For the tasks that have been marked completed, these were the steps taken to meet the DoD criteria. Once we believe a task is done, we do testing to see if it passes. That includes normal testing, unit testing, edge case testing, etc. Once completed, we get the code in our branch reviewed by ourselves and others. Once all the lights are green, we merge it will the main branch.

List of items you included in the DoD

-Complete Code
-Testing(normal testing, edge case testing, unit testing)
-Code Review
-Merge

Business Model Canvas

BUSINESS MODEL CANVAS

Key Partners

- **Universities/College:** Partnerships with educational institutions to integrate the app into their campus systems and make it a part of student services.
- **Hardware suppliers:** Reliable vendors to provide the required sensors for human presence, distance, temperature, and air quality tracking.
- **Cloud Service Providers (e.g., Firebase):** To store and manage real-time room data.
- **App distribution partners:** Google Play Store and IT departments to ensure smooth deployment and updates of the app.

Key Activities

- **App Development:** Ensuring the app's user interface is seamless and its sensor data integration is smooth.
- **Sensor Integration:** Installation and calibration of sensors in study rooms to monitor room occupancy and environmental conditions.
- **Database Management**: Maintaining real-time data flow between rooms and the app, including sensor information and user activity
- **Customer support:** Providing updates and troubleshooting in-app issues.
- **Security and authentication:** Implementing and maintaining a secure authentication system for room access and controls.

Key Resources

- **App development team:** Skilled software developers who can ensure the app functions seamlessly.
- **Sensor hardware:** ability to work with reliable and accurate sensors for human presence, temperature, air quality, and distance.
- **Cloud infrastructure:** Secure, scalable cloud hosting services to store and process real-time data.
- **Data analytics:** Tools for analyzing usage data to improve room availability forecasts and environmental controls.
- **University/College partnerships:** Collaboration to promote and implement the system across campuses.

Value Propositions

- **Convenience:** Easy access and control via a mobile app.
- **Real-Time Room Availability:** Students can easily find vacant study rooms.
- **Environmental Comfort:** Ability to monitor and adjust room temperature, lighting, and air quality.
- **Security:** Secure access to room controls.

Customer Relationship

- **Self-Service**: Users can choose study rooms independently through the app.
- **User Training/Onboarding:** Providing a simple tutorial when first using the app, explaining how to navigate through buildings, check room status, and use room control features.
- **Technical Support:** A help desk or IT support for more complex problems related to app usage or hardware malfunctions.

Channel

- **University/College Website/Portal:** Promoting the app through official platforms such as student portals, email newsletters, and campus orientation programs to ensure that students are aware of its availability.
- **App Stores (Google Play):** Making the app available for download from Google Play Store for easy access to students.
- **Social Media:** For marketing, engaging with users and ads.

Customer Segments

- **Students:** Primary users who need to find available study rooms and manage room conditions.
- **Faculty and Staff:** They could use the app for meetings or workspace management in the same way as students.
- **Tutors:** Looking for a quiet environment for one-on-one tutoring sessions.
- **Exam Preparers:** Those studying for major exams.

Cost Structure

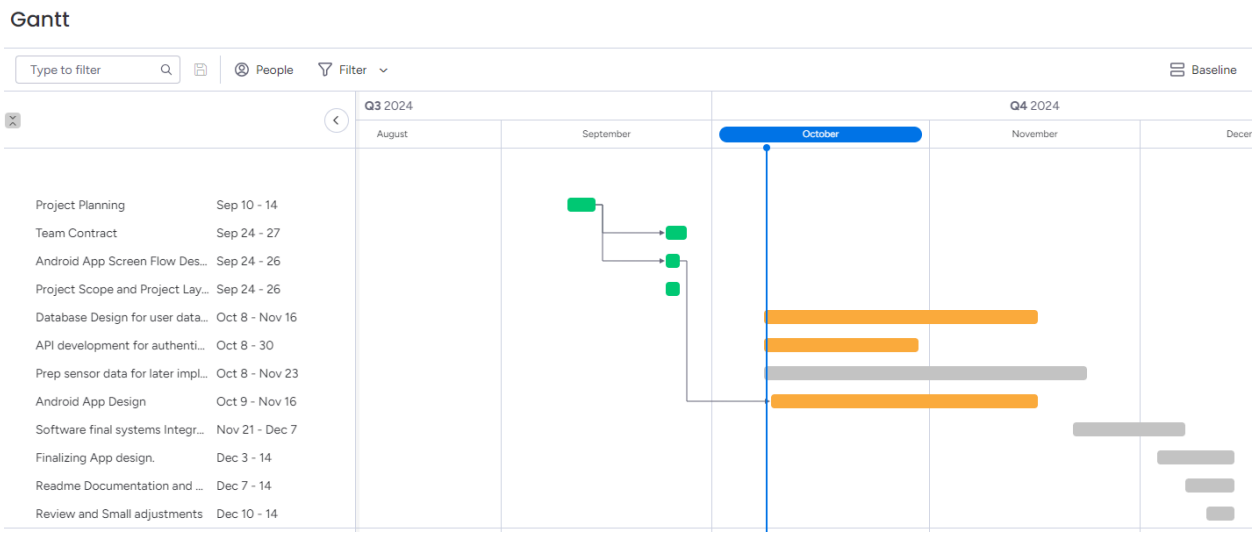
- **App Development & Maintenance:** Ongoing development costs to improve features, fix bugs, and ensure compatibility with new Android versions.
- **Sensor Hardware Costs:** Upfront and maintenance costs for installing sensors in various rooms on campus.
- **Cloud Infrastructure:** Costs for storing, processing, and securing real-time data.
- **Marketing and Promotion:** Expenses for promoting the app to students.
- **Customer Support:** Cost of maintaining a helpdesk for users.

Revenue Stream

Contracts/Service Agreements for University/College

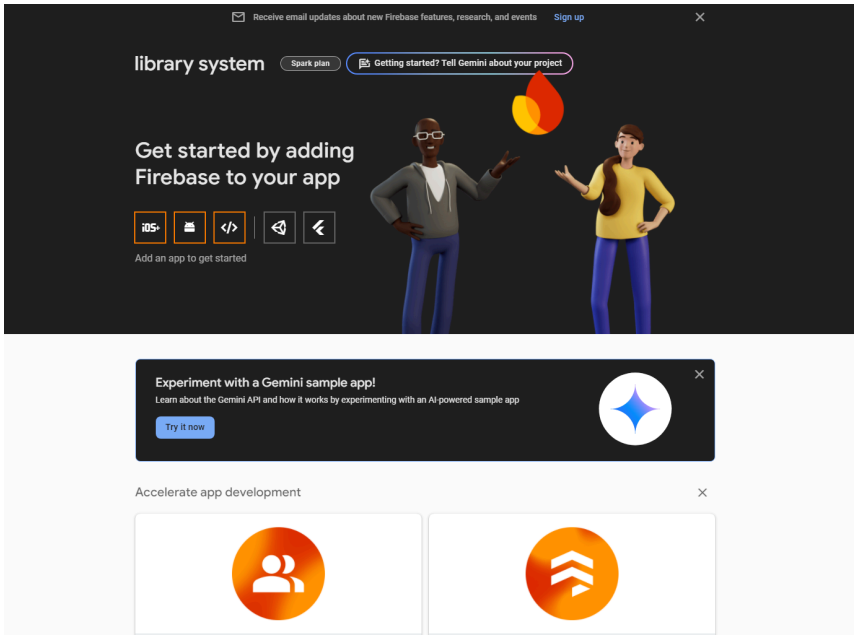
- **Service Contract:** We can negotiate a service contract with the university/college, where they fund the development, deployment, and maintenance of the app as part of the campus infrastructure. They would pay for this service as they benefit from improved resource management and student satisfaction.
- **Ongoing Maintenance Fees:** Charge them a yearly or semesterly fee to cover software maintenance, sensor calibration, hardware repairs, and future upgrades.
- **Initial Setup Fee:** They could pay us a one-time fee for the setup and installation of sensors, as well as the initial app deployment on campus.

Gantt Chart



27. Screenshot showing you have created the DB on the cloud, with some details and type of DB used.

FireBase database



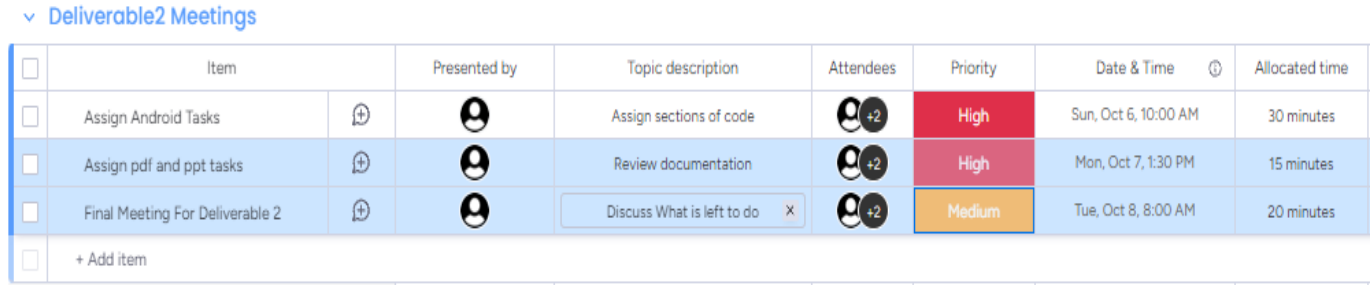
28. Explain how you are planning to use the DB.

Will be using the database to read if the room is occupied or vacant, reading temperature, how many people are in the room, check in/out system, see if door is open/closed, changing code. A lot of our functionality will rely on the database.

Coding work progress & additional features/functionality added since deliverable 1

Based on the feedback from the prof, instead of just using bottom nav, we decided to also use nav drawer. The feedback was told that it didn't make sense to have the buildings in the bottom nav as what if there were like 10 buildings. What we decided to do was implement a nav drawer to contain all the buildings. With each building containing rooms, we implemented image buttons to represent the rooms, with some being occupied and vacant. The occupied ones will need a code to access it, made a dummy code just for testing purposes. Once the code is entered, it will bring us into a different fragment container, containing all the room settings that can be viewed and changed. This fragment container has the bottom navigation, where we can swipe between different room settings. All the room settings have basic images/buttons/texts with no functionality. Also implemented a back arrow to go from the room settings fragment back to the buildings fragment. Made a database, menu, and app settings.

A Table Records of the Daily Stand-Ups



Deliverable2 Meetings

Item	Presented by	Topic description	Attendees	Priority	Date & Time	Allocated time
Assign Android Tasks		Assign sections of code	+2	High	Sun, Oct 6, 10:00 AM	30 minutes
Assign pdf and ppt tasks		Review documentation	+2	High	Mon, Oct 7, 1:30 PM	15 minutes
Final Meeting For Deliverable 2		Discuss What is left to do	+2	Medium	Tue, Oct 8, 8:00 AM	20 minutes
+ Add item						

Document of Design Principle

```
private void navigateToRoomSettings() {
    RoomSettingsFragment roomSettingsFragment = new RoomSettingsFragment();

    getActivity().getSupportFragmentManager()
            .beginTransaction()
            .replace(R.id.tbdFlFragment, roomSettingsFragment)
            .addToBackStack(null)
            .commit();
}
```

Made a function that will take me to the RoomSettingsFragment. This follows one of the design principles of not repeating yourself. If you find yourself writing a block of code more than twice, turn it into a method.