System Requirement Specifications for NTUCollab: Collaboratory platform for NTU Community

Dandapath Soham – Project Manager, Back-end Developer
Gupta Jay – Lead Developer, Release Engineer
Kanodia Ritwik – Front-end Developer, Release Engineer
Mundhra Divyesh – Front-end Lead Developer
Bhatia Ritik – Back-end Lead Developer
Somani Palak – QA Manager, QA Engineer
Bansal Aditya – QA Engineer

Team Eagles
School of Computer Science & Engineering
Nanyang Technological University, Singapore

Submitted to—
Dr Shen Zhiqi
School of Computer Science & Engineering
Nanyang Technological University, Singapore

1. Table of Contents

2. Problem Statement

Social interactions play an important role in an individual's life, especially a university student. Everyone prefers to have acquaintances with whom they share common interests and can talk to freely. The level of social interaction is directly related to the levels of happiness and stress among students. Surveys show that level of stress among college students across the world is at an average of 62% but this trend is much more worrying in Singapore where it is significantly higher at around 85-90 % [1]. One of the major reasons of this high level of perceived stress is the increased focus on academic knowledge and grades, which leaves no time for individuals to pursue their hobbies and interact with people outside their academic bubble. The gravity of the situation is truly realized when one cannot find many applications to address this issue and help university students combat stress and many of its negative consequences. As a result, many students are left to fend off for themselves, without access to an immediate solution that is readily available. Introverted students face an even tougher time trying to find peers with whom they share interests and can bond with.

This is when the need for an application that can effectively address this issue was felt, specifically targeted for the community at Nanyang Technological University. NTUCollab is a cross-platform mobile application, which will empower students of NTU to find groups among their peers based on similar interests and hobbies. Our application will allow users to join groups mainly under three categories of *Modules*, *Clubs,* and *Interests*, based on their preferences. These groups are meant to facilitate easy interaction among peers who want to connect with one another and collaboratively work on projects, competitions, sports etc. With dedicated user profiles using social media login and ability to change preferences and create unique groups, NTUCollab has limitless potential to combat the issues of stress and lack of interaction with others by being an important aspect of every student's daily life. The purpose of NTUCollab is to be a one-stop solution for matching university students and making their social as well as academic life more fruitful and enriching.

3. Overview

3.1 Background

Universities in Singapore provide a very highly competitive environment and the huge focus on academics makes majority of the students miss out on one of the main aspects of collaborating with their peers beyond their own majors. Universities should guarantee holistic development for students and prepare them better for the workplace, which includes a harmonious balance between academics and refining softs skills, beyond academics. The universities in Singapore do provide a very good platform for the development of such skills, however, the competitive environment leads to all focus being made on academics and grades. Students develop amazing hard skills but miss out on cultivating soft skills, a major requirement for being successful in one's career. To further add to the seriousness of the situation, mobile applications provided by NTU like NTULearn [2] and U-Wave [3], which are the most extensively used applications among students, only serve individual needs and do not facilitate an easy form of interaction and group creation among their peers. These applications are academically focused and hence just facilitate discussions regarding courses and schoolwork.

We all know that universities are much more than academics and here an individual grows in all directions in addition to the tremendous gains in academic knowledge. Evidently, there is a need for a mobile application that is widely available and can help students easily meet with like-minded peers and together empower each other to be better at and beyond academics. NTUCollab is the perfect candidate to address this problem as it can act as a dependable platform, meant for the purpose of promoting the interests of students and in general, the university community.

3.2 Overall Description

NTUCollab has great flexibility in terms of the audience that it is available for. Hence, the members of Team Eagles decided that it will be beneficial to extend NTUCollab to include a wider demographic and be available for people of different professions. In essence, NTUCollab will be used by the entire NTU Community which includes students as well as professors, and staff members, to pursue their passion and hobbies while connecting with their peers. To cater to the needs of the entire university community, Team Eagles will focus on building an easily accessible, scalable, maintainable, and highly user-friendly application. To reach out to a wider audience, the project will be developed using the Flutter framework as it allows for rapid, cross-platform development, availability, and scalability, so that the application can be used by users of both Android and iOS platforms. Team Eagles will put in every effort to ensure a well-rounded application that is extremely feature-rich and powerful enough to connect people with just a few taps of the finger, personalized for the needs, interests and preferences of every user.

4. Investigation & Analysis Methodology

4.1 System Investigation

NTUCollab's system follows industry-standard and practices for the entire flow of the application, so that the user can have an enriching experience. When the user opens the application, he / she has to login using his / her Google account credentials. This authentication is ensured to be completely secure as the sign-in process and authentication is managed by Google servers in the backend. Upon successful authentication, the user is led to a page where he / she has to provide personal details like name, course of study, year of study, phone number etc. for profile creation and to register the account of the user with our backend and database service. These details are securely stored in the Cloud Firestore database service, provided by Google Firebase, so that they can be retrieved as and when needed in the future.

The user proceeds to select tags to mark their interest / preferences in each of the three categories, namely *Modules*, *Groups* and *Clubs*. These interests are also stored in the Cloud Firestore service, which is a NoSQL database service, to provide the user with the most up-to-date information. NTUCollab's system then handles the recommendation logic in the backend and presents the user with top ten recommendations in a single page, based on the user's input interests. These recommendations are again from each of the three categories mentioned earlier and are updates as new additions are made to our database. For each category, the user can select a specific recommendation (Module, View or Club) and view details about it like description, ratings (difficulty, time commitment etc.) and other user's posts in the discussion forum. The user can himself / herself post in the discussion forum. Further, if the user does not feel that any recommendation suits him / her, then he / she can create a new group of their interest. This new group will be recommended to other users who have similar interests, and this will in turn help to bring like-minded people together.

Efficient searching of a module, club or group is also available by making use of the *Filter* and *Search* functionalities. These functions efficiently query our database and return the results to the user based on the query input, helping the user save time by not having to scroll through all the recommendations. Our database is indexed, to reduce latency and help return search results as efficiently as possible.

The NTUCollab system also involves interfacing with external systems, namely the **Student Automated Registration System** (STARS), which is the course registration and module viewing system provided by NTU and is used by all students. This interfacing is meant to provide the users with the most accurate and recent information on modules like description, schedule etc. to provide a more well-rounded application to the users.

4.2 Analysis Methodology

4.2.1 Feasibility Study and Requirements Elicitation

To make NTUCollab as feasible and relevant as possible, detailed research, surveys and interviews will be carried out. The User Experience (UX) team will be tasked with reaching out and interacting with the course registration system team in NTU to get accurate details about all the courses offered in NTU, the curriculum and the schedule for each, to maintain consistency of information between STARS and NTUCollab. This information will serve as the meta-data for the module recommendation part of our application. Further, all the teams will individually carry out a series of interviews with approximately 60 students to understand their needs and what they desire with regards to planning undergraduate courses, the structure of each and base this on the years they split.

The Requirements Elicitation will also include contacting several popular co-curricular clubs in NTU and interviewing their Presidents, to know more about what each club does, the number of students that sign up for it every year, their schedule of activities and so forth. Additional metadata will be procured through official informative meetings with the **Student Services Centre** (SSC) and the **Student Affairs Office** (SAO) in NTU, as they oversee the activity of every club. The data so obtained will be an integral part of the club recommendation system of NTUCollab.

Finally, the group recommendation system will be based on the information obtained through open surveys and feedback forms that will be filled out by the general NTU population, to have a massive database containing anonymous interests and preferences of people and form the initial groups and discussion forums using the same.

4.2.2 System Analysis and Requirements Specification

Scope

1. Use user's Google account for secure authentication and link it with the corresponding NTUCollab account
2. Build secure, reliable, and efficient communication between all components of NTUCollab, namely the frontend, backend, and the database
3. Provide the most relevant and up-to-date recommendations to the user based on the interests specified earlier
4. Allow the user to know more about each recommendation such as details, ratings such as time commitment, difficulty etc.
5. Enable the user to provide their rating and post their comments for a particular group in any of the categories and store it in the database
6. Enable users to filter results based on specific parameters or search for a specific module, club, or group, to save time and repetitive work

Limitations

1. NTUCollab's system could be prone to crashes in case there are several users concurrently using the application. Appropriate load testing might be needed to overcome the issue.
2. A user will not be able to enter the application and use its functionalities if the initial login fails, in the unlikely event when the third-party server (that is, Google's server) is down and cannot perform the authentication
3. If the database communication is not working properly, any changes will not reflect when the user logs in on a new session. This can lead to repetitive work as the user will have to perform the same actions, such as creating group, updating preferences etc. for it to accurately reflect in the database
4. The database is susceptible to race conditions when two users simultaneously attempt to perform similar actions that update the same section of the database. Appropriate alternatives like using semaphores, locks etc. will be needed to overcome this issue

4.2.3 Object Oriented Design Using UML

A detailed object-oriented design will be developed using UML as the language for the graphical representation. The system is primarily a recommendation system which at its core provides recommendation for various categories (as mentioned earlier) based on the tags that are initially selected by the user. To improve the model's result, the past selection of the groups will also be considered. The system fetches data from a MySQL server which stores the entire data for all the groups. The system is secured with OAuth verification linked with the Google account of the user. All industry-standard object-oriented practices such as modularization, the MVC (Model-View-Controller) structure for the code, low coupling using concepts of inheritance etc. will be employed to maintain high quality of the code and so that it is easier to add, remove and modify certain sections of the code in the future for new functionalities and update of existing ones. The system will broadly consist of objects, each for a different functionality, that communicate with each other for achieving a particular requirement.

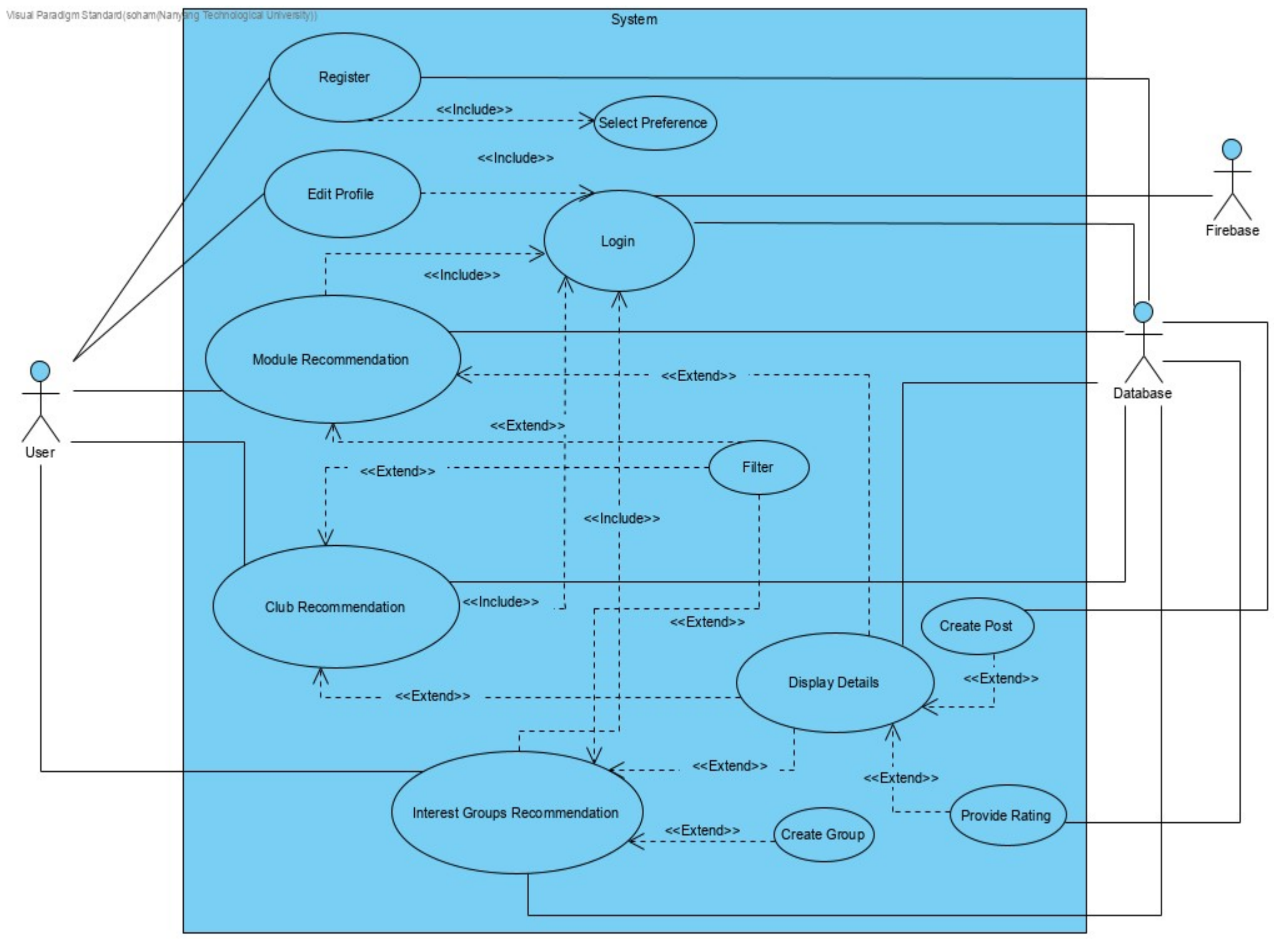
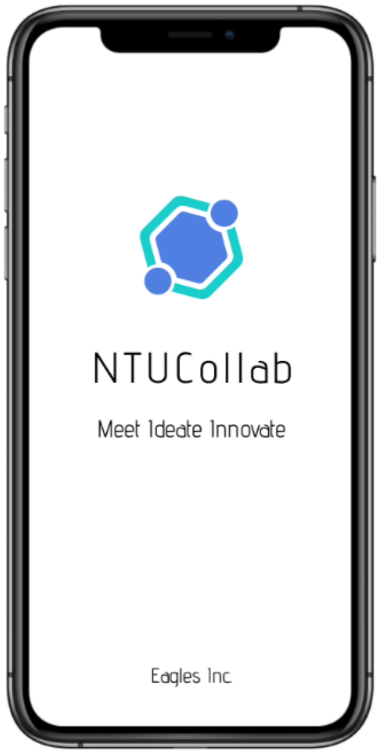
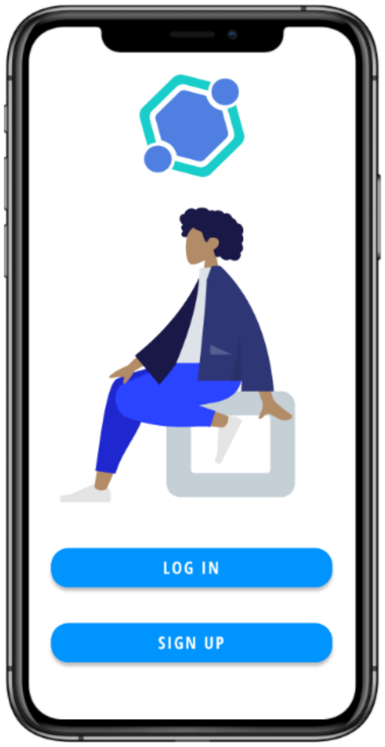
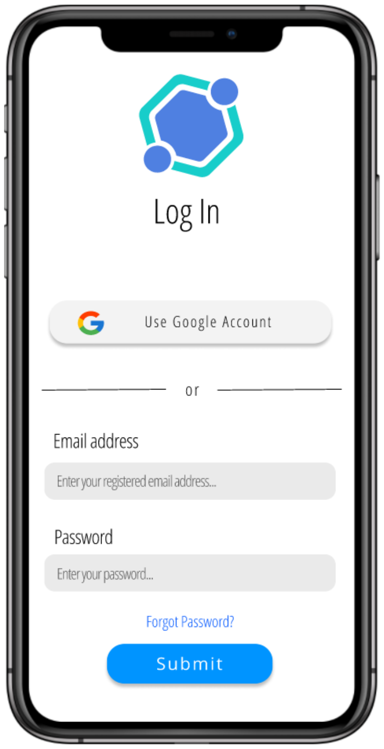


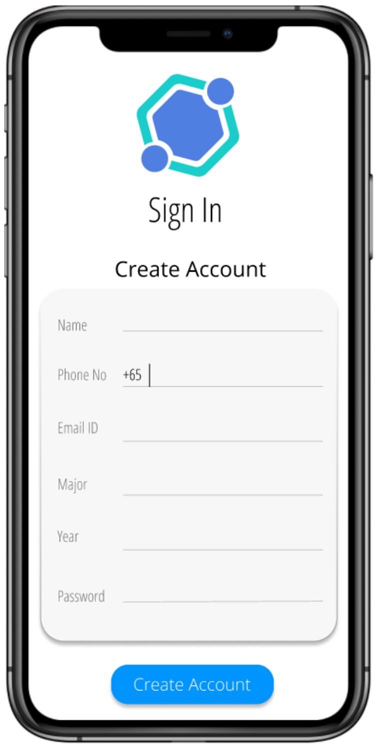
Figure 1: Use-Case Diagram

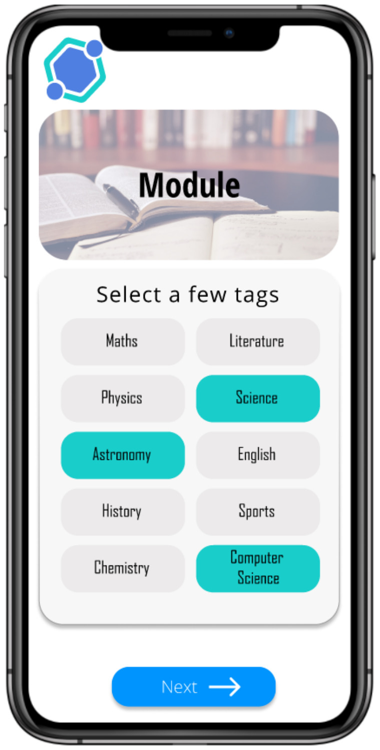
4.2.4 Prototyping

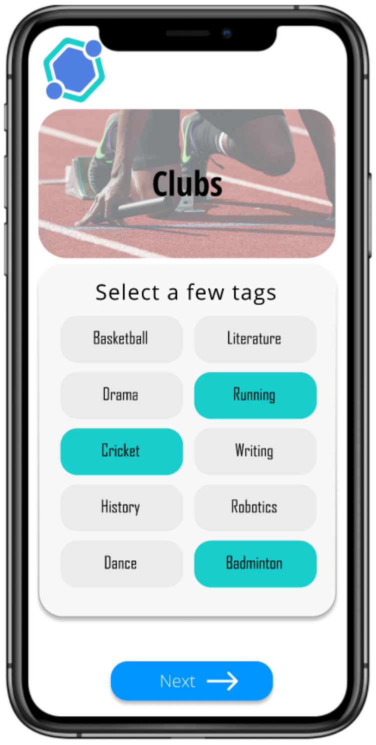


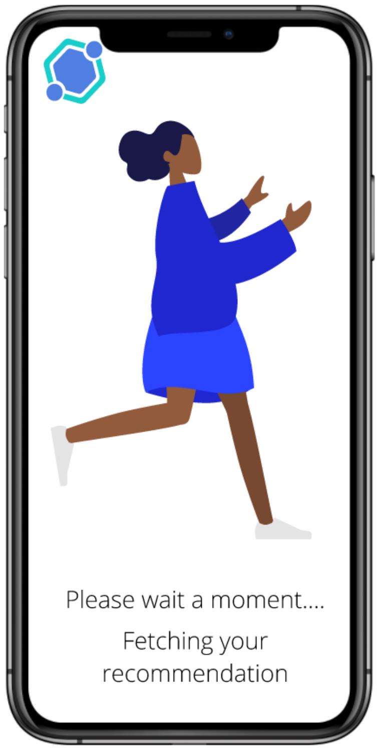


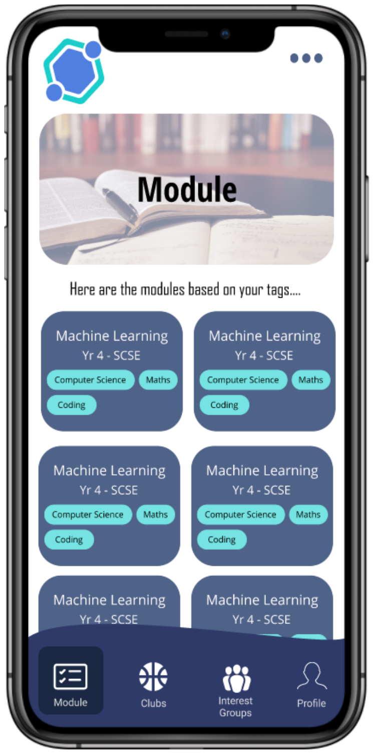


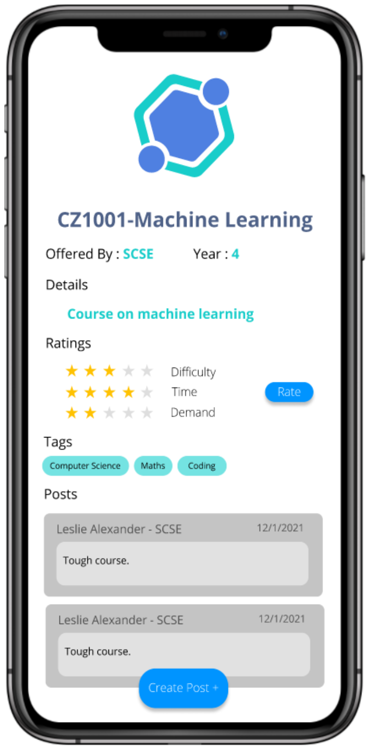











CZ1001-Machine Learning

Offered By : SCSE Year : 4

Details

Course on machine learning

Ratings

★★★☆☆	Difficulty
★★★★☆	Time
★★☆☆☆	Demand

Tags

Computer Science Maths Coding

Posts

Leslie Alexander - SCSE 12/1/2021

Tough course.

Leslie Alexander - SCSE 12/1/2021

Tough course.

5. Constraints

5.1 Scalability

NTUCollab is a cross-platform application that runs on both the Android operating system and iOS. The front-end of the application is made via Flutter, an open-source UI software development kit created by Google, which is installed locally on smartphones. The backend of the app is powered by Firebase, a fully managed cloud backend infrastructure.

As Firebase is a cloud service, it is built for scalability and performance. The APIs are designed to scale linearly with the size of the data being synchronized. Our backend can service up to 10,000 writes per second and simultaneously host 1,000,000 concurrent connections. The backend scaling is automatic and scales up to 1,000,000 writes per second to cater to extreme loads.

5.2 Data and Function Mapping

All data used in the application such as user authentication, recommendation system data for interest groups, modules, and clubs, as well as chat messages used in all three recommender systems are stored in Cloud Firestore, a cloud-hosted NoSQL database. Any module, club, or interest group information can be added remotely from the cloud system. Furthermore, once a user is logged in to NTUCollab, all permissible data can be readily accessed from any device. A source code is change is required if there is a change in the data model such as addition/deletion/updating of new data properties or the way data is displayed in the front-end.

NTUCollab does not run on a mainframe system. Instead, the backend is deployed on the cloud, with a cluster of low-cost servers, operated by a third-party vendor, Google, spread out globally across different data centres for data redundancy. Any new function added to the cloud system is deployed across all users remotely, whereas any change in a function pertaining to the business logic or front-end requires a change and recompilation of the source-code of the main application.

5.3 Proprietary Hardware and Software

This is a cross-platform application built for both Android and iOS operating systems. It supports Android Jelly Bean, API level 16, 4.1.x or newer, and iOS 13.4 or newer. Furthermore, the mobile hardware compatibility is as follows – iOS devices (iPhone SE, iPhone 6S or newer) and all ARM based Android devices. The device is required to have internet connectivity, either Wi-Fi and/or mobile data (4G broadband cellular connectivity or later).

The application can be downloaded and installed via both Google Play Store for Android devices and Apple App Store for iOS devices. No further software installation is required.

5.4 Batch Updates vs (close) Real-Time Updates

NTUCollab requires registration of all users for authentication purposes. All registered user data is synchronized via Cloud Firestore database system on a close to real-time basis. Other data from recommendation systems such as interest group, module, and club data are displayed and updated in the application in real-time as well with the help of event listeners.

Regardless of any manual backups, the system is configured to perform a daily backup of the database, assigned to a specific hour each day. The data is transferred to our Google Cloud Storage bucket will be timestamped in ISO 8601 standard with the specific naming convention via a scheduled job.

Any change in the source-code is available in the form of an application update from our marketplace partners – Google Play Store and Apple App Store. All such updates are staged, and any update reaches only a percentage of users, which is increased gradually. This is done to get early crash reports and monitor user feedback on new releases.

5.5 Project Schedule

There is a four-month timeframe to implement a production system for NTUCollab: a Collaboratory platform for NTU students, starting from January 2021 to April 2021 in time for the incoming freshmen in the university from August 2021 onwards.

Firstly, project requirements are solicited along with a clear definition of the product scope. Secondly, to further iron out the requirements, a use-case model and description is generated in the design phase. Thirdly, all development activities are started only after the completion of the first two phases by the development team. Finally, the release team engages in publishing a release plan, followed by QA team to perform necessary testing.

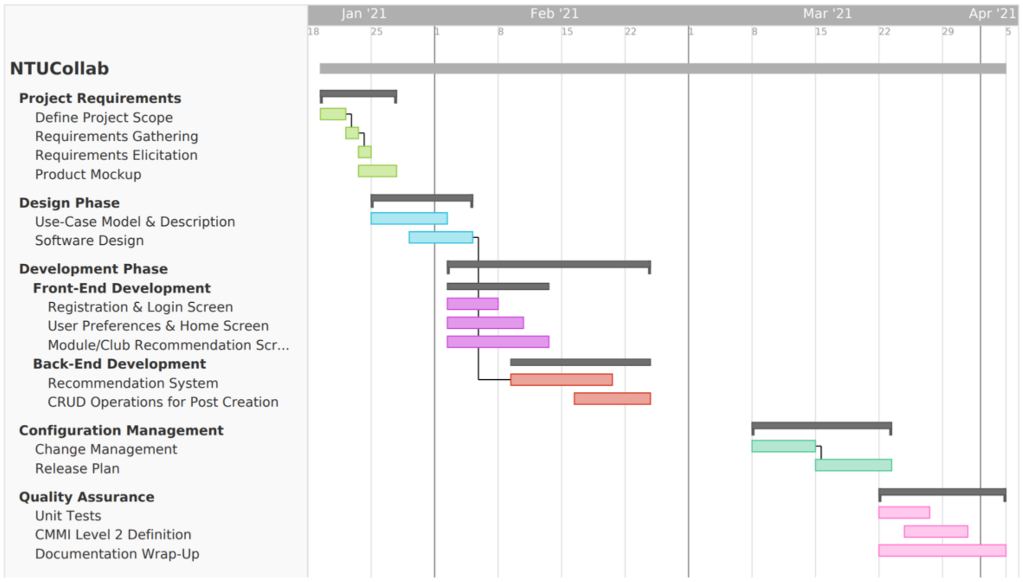


Figure 1: Gantt chart for the project. The solid bars indicate the portions of the tasks that we have accomplished within the timelines.

6. Operational Requirements

6.1 System User Support

System users must have access to 'Support' section in the application. The support section must include FAQ (Frequently Asked Questions) to address common user concerns. The FAQ section must be duly updated in case of any new recurring question or issues. Moreover, a community support forum should also be created for the users to be easily able to find answers to specific and recent issues, which might not be present in the FAQ. The questions this forum should also be then answered by the official support team at the earliest.
In addition to the FAQ, the support section must include an online chat feature as well as a telephone number of the help desk. The chat feature as well as the help desk must be available 24x7 to offer any technical assistance the system user may need such as help in resetting their account password, unable to join a group chat, etc. Further, in case of any such assistance, it must be appropriately recorded with the solution, for further follow-ups.

6.2 Application Services and Technical Support

Users can report any bugs they come across the application or recommend enhancements through feedback. The users can use the above community forum, or the report bugs feature in the support section to report new bugs. Moreover, the users must be asked if they allow crash reports to be sent to the developers for further investigations into system issues or any potential exceptions. Developers must have access to the source code to address either these or any bugs or enhancements they come across on their own. Any issue with a necessary code fix will be duly recorded and a ticket will be created for the bug fix.
In order to ensure high availability and uptime, Network Administrator and Data base Administrator support is required. Database redundancy and network best practices must be put in place to allow adequate system performance and efficiency. Further, some users can be selected as beta testers for the new features, which will allow to find unidentified bugs or issues before major releases.

6.3 Security Features

Since the application contains a group chat feature, system security must be implemented and maintained. The group chat messages must be end-to-end encrypted, i.e., when the user sends a message, it must be encrypted before storing it in the database. The message must then be fetched from the database and decrypted before showing it to other members of the group. There must be no intermediate party who can access the messages.

In case of on-to-one messaging, as soon as both the parties have received the required message and data, it must be removed from the remote servers. Hence, the application will not hold any permanent chat data, but only act as an interim storage till the time the chat message has not been delivered. Further, the application can only be downloaded from authorised application distribution stores, i.e., the Google Play Store and the Apple App Store. The app must not be made available through any other website, to ensure complete compliance of security standards.

An app passcode protection must also be implemented, to enable the user to unlock the user with a passcode. The user must also be given the option to secure the app using the device provided protection. In case of iPhones, Touch ID or Face ID can be enabled to lock the application. An option to archive the chats would be provided to transfer the chats to an archive section.

6.4 Administration Features

There must be authorized system administrator(s) who have the right to delete any groups or block any users from using the application, should the need arise. In case the users are not authorised to access NTU data, they must be blocked from the application and will require separate assistance to reenable the application. In case the user,
Further, measures to prevent scrapping of the application data, must be put in place to prevent robots or crawlers from misusing the system. The administrators will also monitor the presence of any fake accounts. The accounts must be verified with an official NTU email address and only then allowed registration into the application.
The administration must also monitor the chats on the chat forums of interest groups, clubs and module groups and ensure complete adherence to the code of conduct and standards of communication among the members. Any members violating the code of conduct and policies will be blocked from using the application.

6.5 System Hardware and Data

A computer operations centre must be set up. It must service and maintain the system hardware and set up a failover for unforeseen or scheduled primary system down time. It must also carry out scheduled data back up to avoid data loss and release system patches should the need be. The hardware must be serviced every quarter and any quality checks must be performed to ensure quality standards of the equipment. Disaster recovery protocols and drills must be in place in case of any such system wide or regional failures. The cloud provider must have adequate service level agreements to ensure proper uptime of the application. Moreover, complete procedures and guidelines must be followed for data storage in compliance with regional guidelines and data regulations.

6.6 Audit Trail

System audit trail must be an inherit part of NTUCollab. All transactions in the system will capture the action that was taken, the time and by whom. The audit trail includes both the user access and modification as well as administration access. Proper Identity Access (IAM) Policies must be in place to ensure complete track of the access and modification history to check any unintended or unauthorised access of the application. Regular monthly audit log checks must be taken, and the audit trail must be verified on a fortnightly basis.

7. Functional Requirements

NTUCollab is a "Collaborative-platform" that shall help peers establish connection and plan out interactive sessions among the whole NTU Community.

7.1 System Functionality to be Performed

7.1.1 User Registration

7.1.1.1 When user opens the app for the first time, user must be directed to input their particulars.

7.1.1.1.1 Particulars must include the following

7.1.1.1.1.1 Name

7.1.1.1.1.2 NTU Email ID

7.1.1.1.1.3 Course and Year of study

7.1.1.1.1.4 Personal Description

7.1.1.1.1.5 Selection of Interest Tags

7.1.2 User Login

7.1.2.1 When user open the app after registration user must enter their email address and password to log in to the app

7.1.2.1.1 The system must verify if the credentials are valid

7.1.2.1.1.1 If the credentials are valid, the user must be logged into the application and directed to the home page

7.1.2.1.1.2 If the credential are invalid, the user must be asked to re-enter their details up to a maximum of three times, after which their account must be locked and require admin assistance

7.1.2.2 There must be an option for forgot password

7.1.2.2.1 The user will be asked to enter their registered email address and a password recovery link will be generated.

7.1.2.2.2 If the email address does not exist in the database, a message "No Account found", must be displayed and the user must be re-directed to the registration page

7.1.3 Display of Groups

7.1.3.1 The system must display recommended groups under 3 categories, "Modules", "Clubs" and "Interest" groups.

7.1.3.1.1 The groups under "Modules" category must show the details regarding different courses provided in NTU.

7.1.3.1.1.1 Module information such as course code, name, description, preference tags, members must be displayed in the group description.

7.1.3.1.1.2 NTUCollab must be integrated with the STARS system in NTU, which will tell us details regarding course prerequisites.

7.1.3.1.1.3 Members must be allowed to post and chat in the user discussion forum for these group.

7.1.3.1.1.4 Show rating for each module under 3 sections, "Time Commitment", "Difficulty" and "Demand".

7.1.3.1.2 The groups under "Clubs" category will show the details regarding different Extra-Curricular clubs in NTU.

7.1.3.1.2.1 Club information such as club name, description, preference tags, purpose must be displayed in the group description.

7.1.3.1.2.2 Members must be allowed to post and chat in the user discussion forum for these groups.

7.1.3.1.2.3 Show rating for each club under 3 sections, "Time Commitment", "Entry Difficulty" and "Learning".

7.1.3.1.3 The groups under "Interest" category will show the details regarding different Interest groups formed by individuals in NTU itself.

7.1.3.1.3.1 Group information such as group name, description, preference tags, purpose must be displayed in the description.

7.1.3.1.3.2 Members must be allowed to post and chat in the user discussion forum for these groups.

7.1.3.1.3.3 Show rating for each module under a "General Rating" section.

7.1.4 Recommendation System

7.1.4.1 The application must use appropriate matching algorithms where we show only the groups which have majority of the preference tags matched with the user's interests.

7.1.4.2 The user must have an option to join the group or discard our recommendation for a particular group.

7.1.4.3 The user must have an option to edit his/her interest tags at any point while using the application. Based on these changes, the application must suggest recommended groups dynamically.

7.1.5 User Discussion Forum

7.1.5.1 Each group must have a forum, where group members will be able to chat and post on a common platform.

7.1.5.2 The user must be able to attach images, documents, and links with their posts as and when required.

7.1.6 Rating system

7.1.6.1 Each group under the 3 categories of “Modules”, “Clubs” and “Interest Groups” will have an option for their members to rate it.

7.1.6.1.1 Groups under “Modules” category must have 3 fields to rate it.

7.1.6.1.1.1 Fields to be rated are “Time Commitment”, “Difficulty” and “Demand” for each of the module group.

7.1.6.1.1.2 Whenever there is an addition to these ratings by a member, a rolling average must be computed for each of these 3 fields and updated to be shown under group details.

7.1.6.1.2 Groups under “Clubs” category must have 3 fields to rate it.

7.1.6.1.2.1 Fields to be rated are “Time Commitment”, “Entry Difficulty” and “Learning” for each of the club group.

7.1.6.1.2.2 Whenever there is an addition to these ratings by a member, a rolling average must be computed for each of these 3 fields and updated to be shown under group details.

7.1.6.1.3 Groups under “Interest” category must have 1 field to rate it.

7.1.6.1.3.1 Field to be rated is “General Rating” for each of the interest group.

7.1.6.1.3.2 Whenever there is an addition to these ratings by a member, a rolling average must be computed for the rated field and updated to be shown under group details.

7.1.7 Edit Profile

7.1.7.1 The user must be allowed to change profile-specific information like preferences, interests etc.

7.1.7.1.1 On clicking the profile tab, the user must be able to add or remove their preferences

7.1.7.1.2 The user must also be allowed to modify their existing preferences by clicking on the corresponding preference tags

7.1.8 Create Post

7.1.8.1 The user must be allowed the create a post in the group discussion forum

7.1.8.1.1 The user must be prompted if they are sure to post on the message on the group

7.1.8.2 The user must be able to remove the post within 2 minutes of posting to remove any inadvertent posts

7.1.9 Search

7.1.9.1 The user must be allowed to search for a particular interest group, module, or club in the application

7.1.9.1.1 The user must be able to enter the name of the tag or group and fuzzy search in the application

7.1.9.2 If no matching results are found the user must be informed that no such groups, clubs or modules found

7.1.10 Create group

7.1.10.1 The user must be allowed to create an interest group of their choice

7.1.10.1.1 The user must be asked the name, description, and tags of the particular interest group they are forming

7.1.10.1.2 If a group with the same name or tags already exists, the user should be informed about the pre-existing group and directed to the group page

7.1.11 Filter

7.1.11.1 The user must be able to filter the displayed groups, clubs, or modules

7.1.11.1.1 The filtering options must be as follows

7.1.11.1.1.1 Time Commitment

7.1.11.1.1.2 Rating

7.1.11.1.1.3 Demand

7.1.11.1.2 If the filter query is empty the recommendations of the groups, clubs or modules must be in the default order

7.2 Interface with Other Systems

7.2.1 The system must be integrated with the STARS system provided by NTU to gather module information.

The system must get information from STARS such as Module prerequisites, AU's, lecture duration, seminar/tutorial duration, lab duration and frequency, which will be displayed for the groups under the "Module" category.

7.2.2 The system must be integrated with login APIs

The system must allow the user to login using other online accounts like google, Facebook or GitHub. The system must call the corresponding API of the login functionality provider and direct the user to the login page of that online account.

8. Input Requirements

8.1 Student/Staff Email ID Provided By NTU

Each member in the NTU community is given a unique email ID upon admission to the university. While creating account for NTUCollab, any user should know this email ID in order to use the application. Since NTUCollab caters to the need of individuals in NTU only, this ensures that no outsider has access to the application. Accounts may be disabled as and when people leave the university. Any users without an NTU account must not be able to register with the application.

The check for a valid email address would be done at the time of registration. After entering the email address a verification link will be sent to the NTU email address. On clicking the verification link the account will be successfully registered and only then, the user will be able to access the functionality of the application.

8.2 Interest Tags

While creating account for NTUCollab, each user will be asked to select certain tags in the application based on their interests and hobbies. The application will then use its recommendation system to suggest users to join groups in accordance with their selected tags. These tags can be edited based on the user's changing preferences and an option to edit them will be displayed on the screen, each time the application is being used.

The user will also be allowed to create their own interest groups and invite members to a particular interest group. Moreover, an in-built group chat and forum functionality will allow the users to discuss and learn more about the particular interest groups.

8.3 Rating for Groups

Members of any group can rate the group based on certain specific parameters. The groups under the "Module" category will be rated based on "Difficulty Level", "Time Commitment" and "Demand". Groups under the category "Clubs" will be rated on "Entry Difficulty", "Time Commitment" and "Learning" while the "Interest" groups will just have a "General" rating. A new user can get a quick look and summary of a particular group by having a look at these ratings and then they can decide if they want to be a part of a group. These ratings can also be used by the user to filter the recommendations they get on their home screen. The users can filter the recommendations, based on time commitment, demand as well the overall rating of the group to get relevant results.

9. Process Requirements

9.1 Database Transaction

A database transaction symbolizes a unit of work performed within a database management system (or similar system) against a database and treated in a coherent and reliable way independent of other transactions.

A transaction generally represents any change in a database. Transactions in a database environment have two main purposes:

- To provide reliable units of work that allow correct recovery from failures and keep a database consistent even in cases of system failure, when execution stops (completely or partially) and many operations upon a database remain uncompleted, with unclear status.
- To provide isolation between programs accessing a database concurrently. If this isolation is not provided, the programs' outcomes are possibly erroneous.

Also, a database transaction, by definition, must be atomic (it must either complete in its entirety or have no effect whatsoever), consistent (it must conform to existing constraints in the database), isolated (it must not affect other transactions) and durable (it must get written to persistent storage).

For our system, the system must be able to send, receive and trigger transactions to the Firestore database system.

9.2 Data Integrity

Data integrity is the overall accuracy, completeness, and consistency of data. Data integrity also refers to the safety of data in regard to regulatory compliance — such as GDPR compliance — and security.

When the integrity of data is secure, the information stored in a database will remain complete, accurate, and reliable no matter how long it is stored or how often it is accessed. Data integrity also ensures that the data is safe from any outside forces.

9.2.1 Data Integrity Risks

There is an assortment of factors that can affect the integrity of the data stored in a database.

A few examples include:

1. **Human error:** When individuals enter information incorrectly, duplicate or delete data, do not follow the appropriate protocol, or make mistakes during the implementation of procedures meant to safeguard information, data integrity is put in jeopardy.

2. **Transfer errors:** When data cannot successfully transfer from one location in a database to another, a transfer error has occurred. Transfer errors happen when a piece of data is present in the destination table, but not in the source table in a relational database.

3. **Bugs and viruses:** Spyware, malware, and viruses are pieces of software that can invade a computer and alter, delete, or steal data.

4. **Compromised hardware:** Sudden computer or server crashes, and problems with how a computer or other device functions, are examples of significant failures and may be indications that the hardware is compromised. Compromised hardware may render data incorrectly or incompletely, limit or eliminate access to data, or make information hard to use.

9.2.2 Minimizing/Eliminating Data Integrity Risks

Below are few ways by which we aim to minimize data integrity risks:

1. Limiting access to data and changing permissions to restrict changes to information by unauthorized parties.
2. Validating data to make sure it is correct both when it is gathered and used
3. Backing up data
4. Using logs to keep track of when data is added, modified, or deleted.
5. Keeping all systems update with the latest anti-virus software to detect and remove malware and viruses

Additionally, in our system, transaction that are completed must be committed to the database. Unfinished or timed-out transactions must be rolled back and handled. Intermediate data security must be maintained.

9.3 Data Validation

Data validation is an essential part of any data handling task whether one is in the field collecting information, analysing data, or preparing to present data to stakeholders.

Hence validating the accuracy, clarity, and details of data is necessary to mitigate any project defects.

In our system, data error from the user's end and from the back-end database-processing end must be gracefully handled. There must be data validation and error-handling routines as part of the system. Lastly, login credentials must be validated before allowing access to the system.

9.4 Performance

The System shall have a responsive look and feel at all times. Long running operations or requests shall avoid blocking the UI. In cases where requests may take some considerable time (more than 5 seconds) to display the results, a user-friendly loader message or progress bar shall be presented to the user, preferably with an indication on how much time the operation still needs to complete. The responsive look and feel shall be valid after a long-term System usage when there are a significant number of records inserted into the database.
The system must also resolve locking issues and handle concurrent use of the system on a 24x7 basis. Messages must be displayed, and appropriate feedback must be provided to enrich user experience.

9.5 Data Repository

A data repository can be defined as a place that holds data, makes data available to use, and organizes data in a logical manner. A data repository may also be defined as an appropriate, subject-specific location where researchers can submit their data. Data repositories may have specific requirements concerning subject or research domain; data re-use and access; file format and data structure; and the types of metadata that can be used.

In our system, the online registration system must maintain the existing Firestore database instance as the main repository of data. The purpose of our data repository is to keep a certain population of data isolated so that it can be mined for greater insight or business intelligence or to be used for a specific reporting need that may arise in the future.

10. Output Requirements

10.1 Transaction Summary and Confirmation

Each database transaction or sequence of actions taken by the user must be summarised and confirmed where needed. For example, when the user wants to exit a group, a confirmation pop up must be displayed before carrying out the action/transaction. The user will also be allowed to undo actions wherever possible.

10.2 Exception Reports

System exception reports must be consolidated to record to special or popular user groups whose participation may exceed the maximum allowed. It must also record any groups which do not conform to the standards and rules.

10.3 Registration Reports and Summaries

The appointed system administrators must be able to extract the data into summarized and meaningful information. This data however does not include the group chat messages.

All records must be archived but accessible on demand. The group chat messages may be decrypted by intermediaries, in case a serious investigation is required.

11. Hardware Requirements

11.1 Network

A stable and high-speed wireless connection is required for stable functioning of the application. We recommend a minimum speed of 10 Mbps to ensure smooth functioning of the application.
The bandwidth used by our application will be optimized for the best experience based on the participant's network. It will automatically adjust for 3G, 4G or Wi-Fi environments.

11.2 Client Computers

The mobile hardware compatibility is as follows – iOS devices (iPhone SE, iPhone 6S or newer) and all ARM based Android devices.

Processor and RAM requirements:

	Minimum	**Recommended**
Processor	Single-core 1 GHZ or higher	Dual-core 2 GHZ or higher
Ram	N/A	2 GB

Note: CPU and RAM usage may vary significantly based on the operating system and application usage.

Memory usage is based on the volume of group chats, images, and other media on the NTUCollab Collaboratory platform. As a default, we recommend a minimum of 75 megabytes of available storage on the mobile device.

11.3 Production Support Systems

NTUCollab uses Google's Firebase (backend cloud infrastructure) and Firestore (cloud database system) as the primary production support system.

Google Firebase is a Google-backed application development software that enables developers to develop iOS, Android and Web apps. Firebase provides tools for tracking analytics, reporting, and fixing app crashes, creating marketing and product experiment. Cloud Firestore is a flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud. Furthermore, cloud Firestore is a cloud-hosted, NoSQL database that your iOS, Android, and web apps can access directly via native SDKs. Cloud Firestore is also available in native Node.js, Java, Python, Unity, C++, and Go SDKs, in addition to REST and RPC APIs.

Additionally, below are some of the services we plan on leveraging by using firebase and Firestore:

- **Analytics** – Google Analytics for Firebase offers free, unlimited reporting on as many as 500 separates events. Analytics presents data about user behaviour in iOS and Android apps, enabling better decision-making about improving performance and app marketing.

- **Authentication** – Firebase Authentication makes it easy for developers to build secure authentication systems and enhances the sign-in and onboarding experience for users. This feature offers a complete identity solution, supporting email and password accounts, phone auth, as well as Google, Facebook, GitHub, Twitter login and more.

- **Cloud messaging** – Firebase Cloud Messaging (FCM) is a cross-platform messaging tool that lets companies reliably receive and deliver messages on iOS, Android and the web at no cost.

- **Realtime database** – the Firebase Realtime Database is a cloud-hosted NoSQL database that enables data to be stored and synced between users in real time. The data is synced across all clients in real time and is still available when an app goes offline.

- **Crashlytics** – Firebase Crashlytics is a real-time crash reporter that helps developers track, prioritize and fix stability issues that reduce the quality of their apps. With Crashlytics, developers spend less time organizing and troubleshooting crashes and more time building features for their apps.

- **Performance** – Firebase Performance Monitoring service gives us (developers) insight into the performance characteristics of their iOS and Android apps to help them determine where and when the performance of their apps can be improved.

- **Test lab** – Firebase Test Lab is a cloud-based app-testing infrastructure. With one operation, developers can test their iOS or Android apps across a variety of devices and device configurations. We can also see the results, including videos, screenshots, and logs, in the Firebase console.

- **Expressive querying** - In Cloud Firestore, we plan on using queries to retrieve individual, specific documents or to retrieve all the documents in a collection that match our query parameters. Our queries can include multiple, chained filters and combine filtering and sorting. They are also indexed by default, so query performance is proportional to the size of our result set, not our data set.

- **Flexibility** - The Cloud Firestore data model supports flexible, hierarchical data structures. We plan to store our data in documents, organized into collections. Our documents can thus contain complex nested objects in addition to subcollections.

- **Offline support** - Cloud Firestore also caches data that our app is actively using, so the app can write, read, listen to, and query data even if the device is offline. When the device comes back online, Cloud Firestore synchronizes any local changes back to Cloud Firestore.

- **Designed to scale** - Cloud Firestore brings the best of Google Cloud's powerful infrastructure: automatic multi-region data replication, strong consistency guarantees, atomic batch operations, and real transaction support. Cloud Firestore thus allows to handle the toughest database workloads from the world's biggest apps.

Furthermore, all cloud systems comprising of hardware, virtualization, and storage components are fully managed and maintained by Google. No infrastructure management is required. The data centres hosting our application’s data also comprise of back-up tapes, redundant drives, and are spread across multiple regions for low-latency delivery.

Below is how we plan on implementing cloud Firestore into our application:

1. Integrating the Cloud Firestore SDKs
2. **Securing our data** - Using Cloud Firestore Security Rules or Identity and Access Management (IAM) to secure our data for mobile and server development, respectively.
3. **Adding data** - Creating documents and collections in our database.
4. **Getting data** - Creating queries or using real-time listeners to retrieve data from the database.

12. Software Requirements

12.1 Client Operating Systems

12.1.1 Android – Jelly Bean, API level 16, 4.1.x or newer

Android is a mobile operating system based on a modified version of the Linux kernel and other open-source software, designed primarily for touchscreen mobile devices such as smartphones and tablet.

12.1.2 Apple – iOS 13.4 or newer

iOS is a mobile operating system created and developed by Apple Inc. exclusively for its hardware. It is the operating system that powers many of the company's mobile devices, including the iPhone.

12.2 Client Application

The front-end of the application is built using Flutter, a UI development framework. Flutter SDK is Google's UI toolkit for crafting beautiful, natively compiled applications for mobile, web, and desktop from a single codebase.

Flutter uses Dart as the programming language which compiles into self-contained, native-executable code, due to which, it does not require any client application support apart from the application installation itself.

Furthermore, Dart is a client-optimized language for fast apps on any platform. It is optimized for building user interfaces with features such as the spread operator for expanding collections, and collection if for customizing UI for each platform. Dart code can be AOT-compiled into machine code (native instruction sets). Apps built with Flutter, are then deployed to app stores as AOT-compiled Dart code.

12.3 Network System

All data regardless of sensitivity, is sent to the Firebase backend system over the HTTPS protocol. Since some of the data sent over the web is sensitive, an encrypted network protocol such as WPA2/AES is highly encouraged to prevent any data leakage.

WPA2 is a type of encryption used to secure the vast majority of Wi-Fi networks. A WPA2 network provides unique encryption keys for each wireless client that connects to it.

There is no network diagnostic tool built-in the application to warn users of any potential network pitfalls.

The following network protocol used to establish communication with the backend service. Please make sure that there is no explicit blocking of port 443 in the mobile system.

12.3.1 TCP/IP (Port 443)

The Transmission Control Protocol (TCP) is a communications standard that enables application programs and computing devices to exchange messages over a network. It is designed to send packets across the internet and ensure the successful delivery of data and messages over networks. TCP organizes data so that it can be transmitted between a server and a client. It guarantees the integrity of the data being communicated over a network.

Additionally, the Internet Protocol (IP) is the method for sending data from one device to another across the internet. Every device has an IP address that uniquely identifies it and enables it to communicate with and exchange data with other devices connected to the internet. IP is responsible for defining how applications and devices exchange packets of data with each other.

12.3.2 HTTPS

Hypertext transfer protocol secure (HTTPS) is the secure version of HTTP, which is the primary protocol used to send data between a web browser and a website. HTTPS is encrypted in order to increase security of data transfer. This is particularly important when users transmit sensitive data such as passwords within our application. HTTPS thus prevents websites from having their information broadcast in a way that is easily viewed by anyone snooping on the network.

Please allow the following URLs to be accessed through the system firewall and ensure that there is no explicit blocking.

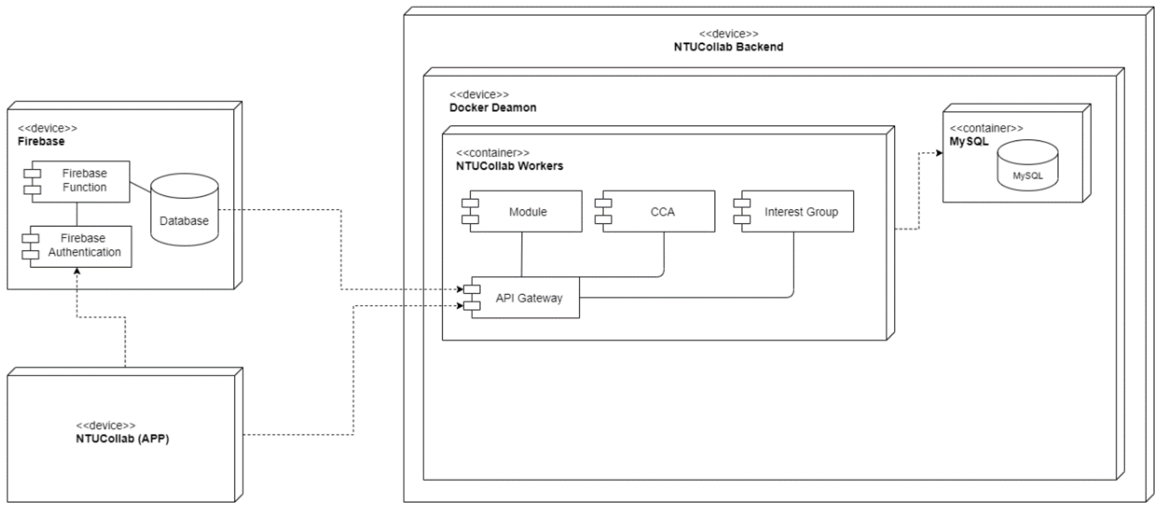
- https://*.firebaseio.com (backend system)
- https://*.ntu.edu.sg (NTU STARS integration)

12.4 Licenses

Valid licenses are required to run software from third-party vendors.

Application Development Tools	Android Studio	Apache License 2.0
	GitHub	MIT
	Figma	Organization License
	Flutter	BSD License 2.0
Backend and Database System	Google Firebase	Blaze Plan
Application Marketplace	Google Play Store	Google Play Developer Distribution Agreement
	Apple App Store	Apple Developer Enterprise Program

13. Deployment Requirements


<<device>>
Firebase
Firebase Function
Firebase Authentication
Database
<<device>>
NTUCollab (APP)
<<device>>
NTUCollab Backend
<<device>>
Docker Deamon
<<container>>
NTUCollab Workers
Module
CCA
Interest Group
API Gateway
<<container>>
MySQL
MySQL