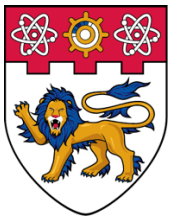

NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

NTUCollab: A Collaboratory Platform for NTU Students

Design Report on Software Maintainability

Version 1.3

Prepared by Team Eagles

Dandapath Soham
Gupta Jay
Kanodia Ritwik
Mundhra Divyesh
Bhatia Ritik
Somani Palak
Bansal Aditya

05 April 2021

VERSION HISTORY

Version #	Implemented By	Revision Date	Approved By	Approval Date	Reason
1.0	Dandapath Soham	03/25/2021	Bansal Aditya	03/27/2021	Initial Design Report
1.1	Bansal Aditya	03/29/2021	Dandapath Soham	03/29/2021	Adds software configuration management tools
1.2	Dandapath Soham	04/01/2021	Bansal Aditya	04/01/2021	Adds the system architecture diagram
1.3	Bansal Aditya	04/04/2021	Dandapath Soham	04/05/2021	Final proofread, review, and formatting

TABLE OF CONTENTS

1 Design Strategies

1.1 The Planning Phase Before Development

Initiating the planning phase before the development, NTUCollab has been designed to be made extensible and scalable as the user base grows. Any improvements required in the future are anticipated and analysed earlier in the lifecycle, to ensure a flexible design of the application. As the application traffic increases it will be scaled up to handle higher user requests. Hence, we targeted Scalability as one of the factors that we would have to investigate in the future.

NTUCollab is an application targeting university students. For our current release, we target primarily the students of the NTU. Once the application becomes popular, we would be extending our application to more users from other universities as well. This would cause our application to be scaled up to handle heavy user traffic. Hence Scalability is one of the factors that we listed that needs to be investigated in the future.

We would also be adding new features such as machine learning models to make more accurate recommendation for the user. Hence we also considered Flexibility as one of the criteria while designing the system architecture. Flexibility and Scalability each promotes the other.

Considering the above constraints, we decided to develop NTUCollab based on Micro-Service architecture. This system architecture allows the application to be broken into smaller independent components. This makes the application more flexible as any change to a component will not break the whole application.

1.2 The Process of Developing

The NTUCollab application is tested using a test-driven development approach. Due to the COVID-19 pandemic, safe distancing measures and contact minimization have been enforced. This has restricted us to approach more user from our target based for testing out our application. However, to follow quality assurance, team members have taken up the role of the tester. They provide continuous feedback on the design and usability of the product which are considered for improving the software application.

Moreover, during the development phase itself, the application continued to evolve, and the design was revised throughout the software development lifecycle. Using various of kinds of test, reviews, and usability analysis the application has been designed to achieve high scalability, usability, and maintainability.

During the code phase the various components of the application, have been developed to be as generic and re-usable as possible to ensure an easy extensibility, and reusable as possible. The use of decoupled and single use application components allows for an easy maintenance of the application.

1.3 Correction by Nature

The feedback and the errors that would be received during the testing needs to be corrected.

1.3.1 Corrective Software Maintenance

The bugs reported from the users during testing are fixed through this approach.

1.3.2 Preventive Software Maintenance

Each of the features will be tested to look for fault tolerance to increase the life of the software application and make it more reliable.

1.4 Enhancement by Nature

The software needs to be updated based on the changes and new features that are added to time to time.

1.4.1 Adaptive Software Maintenance

Due to the recent unpredictable nature of the university, there would be requirement for a change quite often. This approach allows the application to be flexible to update to these new changes.

1.4.2 Perfective Software Maintenance

The software maintenance that arises after the release of the product in the market will be taken care by this approach.

1.5 Maintainability Practices

Formally, the IEEE Standard Glossary of Software Engineering Terminology defines maintainability as: *"The ease with which a software system or component can be modified to correct faults, improve performance or other attributes, or adapt to a changed environment."*

To follow the quality assurance for both the process and the product NTUCollab, we have implemented the following maintainability practices over the course of the project:

1.5.1 Readable Code

It is inevitable the requirements on which the NTUCollab has been built are likely to change in the future. This is because as the application will interact with the real world, new changes and requirements will be necessary.

In such a scenario, readable source code facilitates the reading and understanding of the abstraction phases and as a result, facilitates the evolution of the codebase. Readable code saves future developers' time and effort. And is easy to understand ("write programs for people").

Hence Code readability is a crucial part of NTUCollab, to ensure high maintainability.

1.5.2 Version Control

Version Control also called as Version Management, or Revision Control, allows to effectively track and control changes to a collection of related entities. It enables to track and control changes to source code. For NTUCollab, version control is a very important tool within an overall life cycle management strategy for information solutions.

For the development of NTUCollab we have used the Git version control system, hosted on the web-based application GitHub. All the development code has been hosted on GitHub and all the changes, branches and various version of the application have tracked, and controlled on the platform.

Moreover, for tracking and controlling all the documentation and various documents pertaining to various stages in the SDLC of NTUCollab, SVN has been used.

1.5.3 Standardized Documents

All the documents prepared for documentation of NTUCollab have been developed using standardised Industry templates, guidelines, and frameworks, for instance the IEEE practices. This ensures that all the prepared documentation items are standardised and can be tracked and maintained effectively through version control.

1.5.4 Modularity

NTUCollab has been developed using a microservice based application architecture. Hence, the majority of the components are decoupled from each other and have a singular use to give maximum modularity. This allows for easy reuse and replacement of components in the application. Further, modularity in the application design will enable us to effectively maintain NTUCollab in the future as the new changes and updates are rolled out in the application.

2 Architectural Design Patterns

NTUCollab is using the micro-service architecture design pattern. Each of the smaller components are running independently accessed as REST API. The smaller components include:

2.1 Databases

We are using a google cloud's Firestore product to store all our data. This also enables us to use Firbase authentication thus allowing users to sign in with their google account.

Google Firestore allows to easily develop rich applications using a fully managed, scalable, and serverless document database. The serverless document database will allow to seamlessly scale NTUCollab application to meet any demand, with minimal maintenance. Through Firestore there is direct connectivity to the database, built-in live synchronization, and offline mode. Moreover, fully customizable security and data validation rules ensure that the data is always protected. Firestore further provides seamless integration with Firebase and Google Cloud services like Cloud Functions and BigQuery.

2.2 Recommendation algorithm

Our current recommendation algorithm uses the tag defined by the users and those that are present in our database. It calculates the overlapping between the two sets and finds the top-most recommendation based on the score. We plan to use machine learning algorithm in or next versions to provide even more robust and appropriate recommendation for the user.

Using machine learning there can be two kinds of recommendation algorithms which could be employed for providing the recommendations. Firstly, content-based filtering methods, make recommendations based on user preferences for product features. Collaborative filtering mimics user-to-user recommendations. It predicts user preferences as a linear, weighted combination of other user preferences.

2.3 API server gateway

The API server gateway passes the information to the other components. The API server gateway is the main communication block of the architecture. It directs URLs from the user end to the appropriate smaller component which runs the service.

System Architecture Design

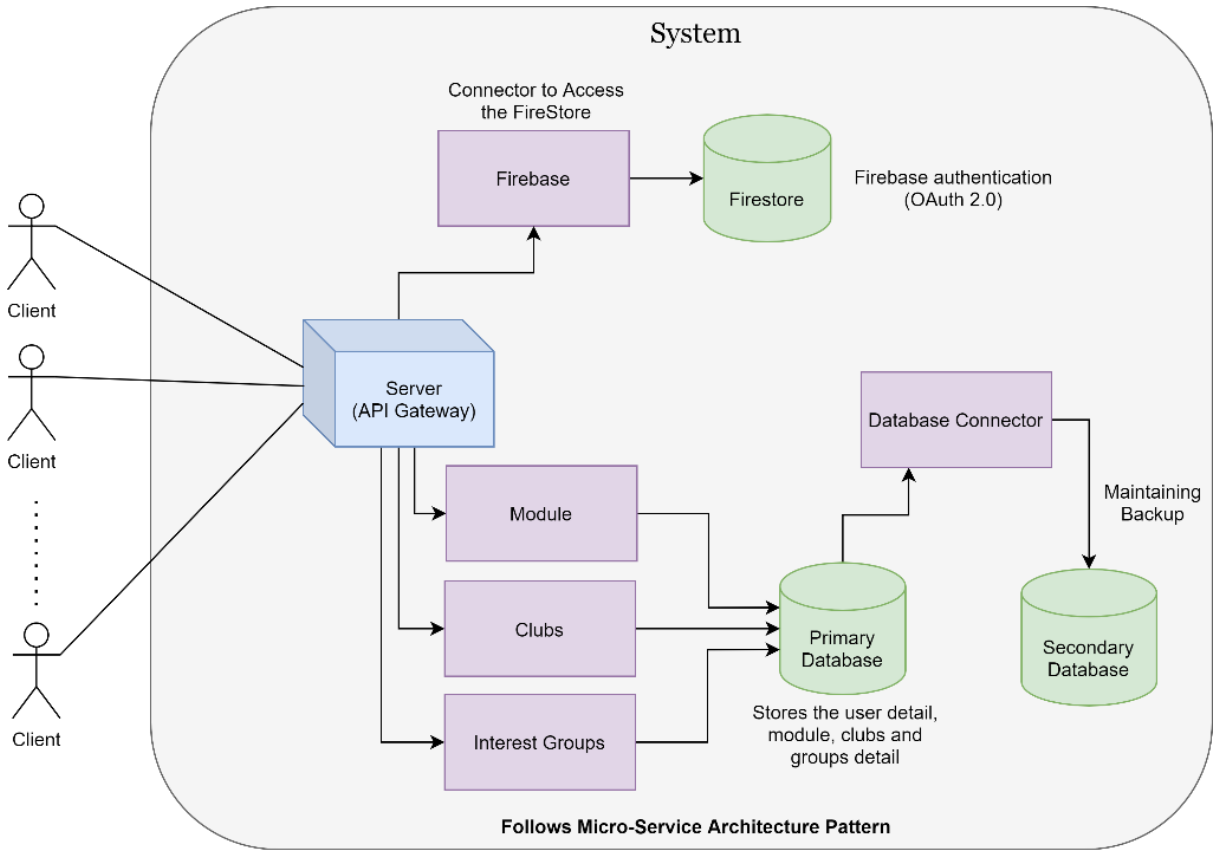

Figure 1 System Architecture Design

3 Software Configuration Management Tools

The following are the software that are used to track the changes made by the developers and for version control:

3.1 MediaWiki

MediaWiki is a free and open-source software used for maintaining the documents which can be edited by anyone having the appropriate permissions. It provides many functions that allows the user to display the information in various formats making the document more user-friendly. MediaWiki allows multiple collaborators to edit the pages at the same time.

3.2 GitHub

GitHub is a provider of Internet hosting for software development and version control using Git. It offers the distributed version control and source code management functionality of Git, plus its own features such as CI/CD pipeline which makes deploying the changes into the market fast and easier. The developers can contribute to the project independently and later merge their code for integration of the different components.

3.3 OneDrive

The OneDrive service is a cloud storage platform that is used by us to store all the initial documents that are needed during the development process. This allows user to share, store files within a group easily. It also allows multiple editors to edit the files which makes collaboration easier. Furthermore, the editing of the documents can be done using the popular Microsoft Word Document, which allows the version control for the document.

3.4 SVN

All the documentation documents have been tracked and recorded using SVN. This allows for easy maintainability of all the documentation items. Further different versions of the documents are also tracked, and the history is maintained in the SVN for easy roll back in case the need arises.

3.5 Microsoft Teams

All the meeting documents and any other project item is stored in the MS Teams. This allows for easy sharing and reference for all the team members. This provides a single source of truth for the whole application and prevents any inconsistencies in different versions of the same document.