
# NTUCollab: A Collaboratory Platform for NTU Students

RELEASE MANAGEMENT PLAN

Version 1.4
Prepared by Team Eagles

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

20 March 2021

# VERSION HISTORY

| Version # | Implemented By | Revision Date | Approved By | Approval Date | Reason |
|---|---|---|---|---|---|
| 1.0 | Bhatia Ritik | 20/03/2021 | Bhatia Ritik | 21/03/2021 | Initial Release Management Plan |
| 1.1 | Kanodia Ritwik | 24/03/2021 | Bhatia Ritik | 26/03/2021 | Introduction, Overview |
| 1.2 | Bhatia Ritik | 27/03/2021 | Bhatia Ritik | 28/03/2021 | Release Approach – Rationale, Content, Schedule etc. |
| 1.3 | Kanodia Ritwik | 30/04/2021 | Bhatia Ritik | 01/04/2021 | Glossary, Acronyms |
| 1.4 | Bhatia Ritik | 02/04/2021 | Bhatia Ritik | 02/04/2021 | Final editing and formatting |

# TABLE OF CONTENTS

# 1. INTRODUCTION

This document lays out the release plan for NTUCollab, which is a collaboration platform developed by Team Eagles. It aims to describe the release strategy for NTUCollab and provide relevant details for all past, current and future releases of the platform, with the version 1.0.0 being the release when the application is first rolled out for users. This document will be updated and used to keep a track of all the releases of the application. It will help all future team members understand the progress of the application over the years and take decisions accordingly. It will also help developers resolve any issues the users of older releases may be facing. The document will also be used to capture and track features for upcoming releases. The main audience of the document will be the internal development and product team. It should not be made public as it may contain sensitive information about future releases. Some information from the document may be made public to provide more information to the users once the release is rolled out. However, it must be done completely under the discretion of the project manager.

# 2. REFERENCED DOCUMENTS

| Document Name | Issuance Date |
|---|---|
| NTUCollab Project Proposal | 02/02/2021 |
| NTUCollab Project Plan | 16/02/2021 |
| NTUCollab Quality Plan | 16/02/2021 |
| NTUCollab System Requirement Specification | 16/02/2021 |

# 3. OVERVIEW

One important aspect of university life is to meet like-minded people who share similar interests and develop lifelong bonds. This helps people feel more inclusive and pursue their interests more openly and to a greater extent. However, meeting such people can be difficult sometimes in a University like NTU, where the size of the community is huge. NTUCollab aims to do away with this by providing a digital platform for people to meet and discuss their passions.

CCAs have always been an important part of the community at NTU. NTUCollab also aims at recommending the right societies and clubs to the students based on their interests. Additionally, it also recommends the modules a student should take up based on their interests and provides all relevant details of the modules, including ratings by students who may have taken it up in the past. This helps the student to plan their academics well and make informed decisions, leading to better results.

NTUCollab was developed by Team Eagles, a group of students in NTU. It was developed using the Flutter Framework in the frontend along with Google firebase in the front end. The version 0.0.0 was developed internally for the purpose of testing and improvement. The version 1.0.0 is planned to be the first release which is rolled out to the users. The Waterfall lifecycle method was used for the development of the application. The updated use case diagram for the system which provides a high-level context of the system is as follows:

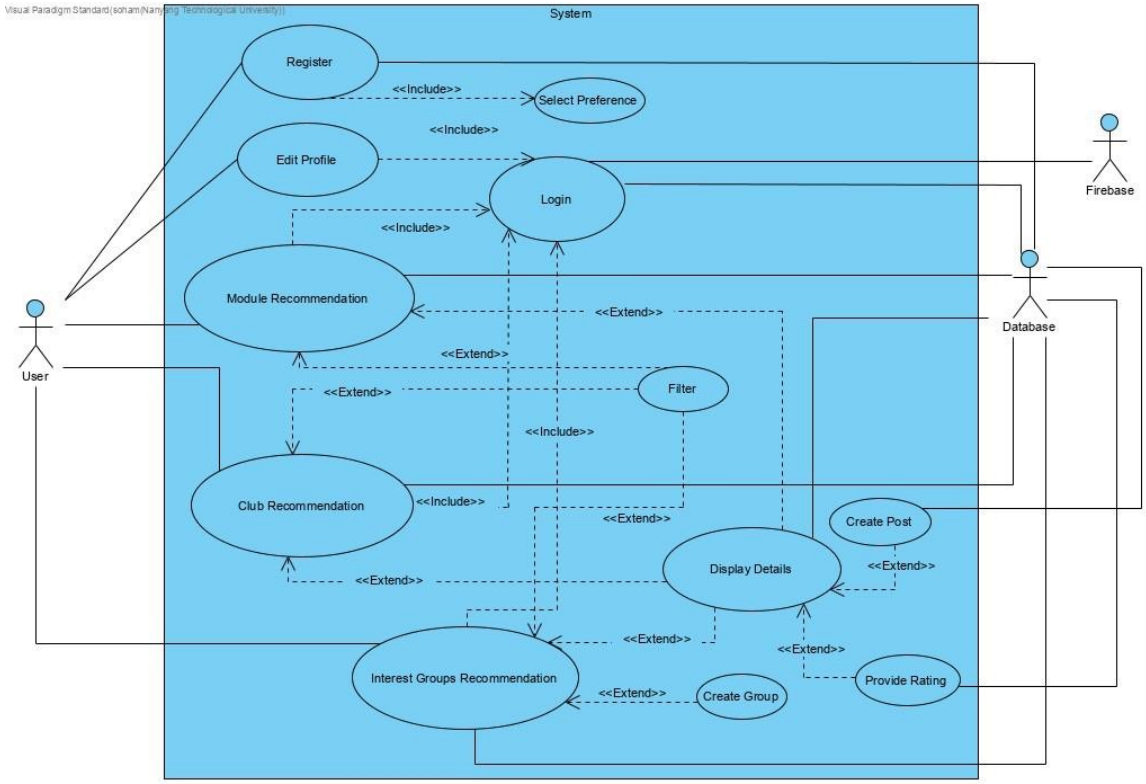


**Figure 1:** Use-Case Diagram

The system architecture as per on 27th March 2021 is as follows,

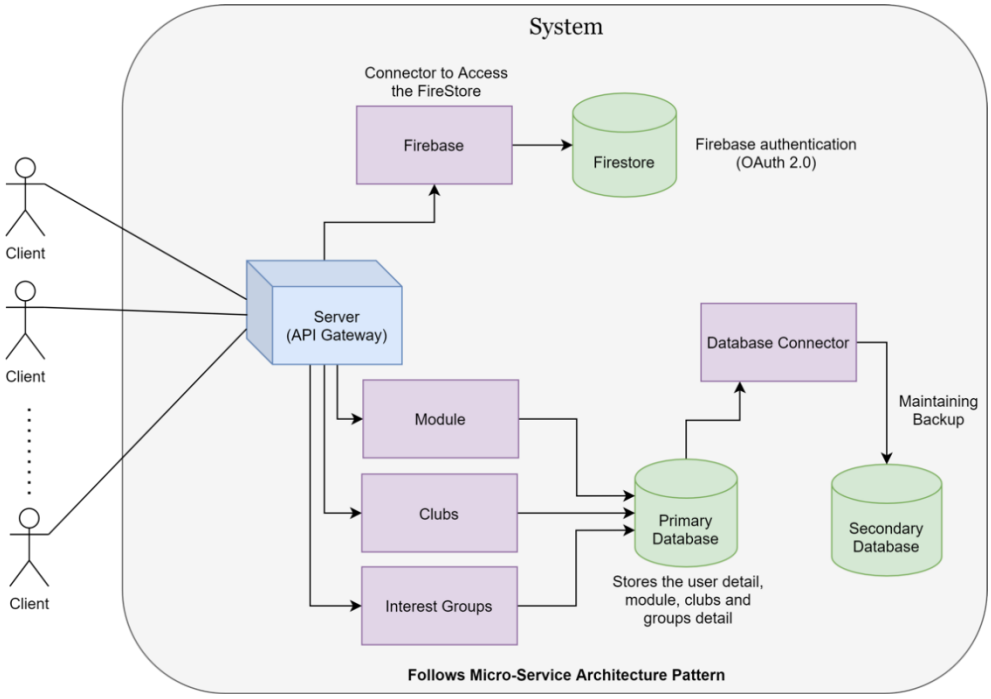


**Figure 2:** System Architecture

# 4. ASSUMPTIONS, CONSTRAINTS, RISKS

**Assumptions:**

Administrative:

1. NTUCollab does not go against the policies of NTU.
2. NTU STARS is updated with latest information about the courses at NTU.
3. Co-curricular activity clubs are willing to list themselves as a club in NTUCollab.

Schedule:

1. The development and release timeline should not be altered considerably.
2. The team members are willing to commit the required amount of time for NTUCollab.
3. All tasks are completed as per the schedule.

Budget and Financing:

1. There are no sudden budget cuts or hidden costs.
2. The budget estimation and resources allocation for NTUCollab is precise.

Development:

1. The developers are skilled and able to meet expectations.
2. The developers and customers are able to collaborate on the development activities to get proper outcome.

**Constraints:**

Manpower:

1. The team size is relatively small and consists only of 8 members.

Functionality:

1. The information for modules depends on STARS.
2. The information on Co-curricular activities depends on the respective clubs.

**Risks:**

A detailed description of the risks can be found in NTUCollab Risk Management plan. A summary of the risks, along with its portability and severity can be found in the table below.

| S. No | Risk | Probability | Effects |
|---|---|---|---|
| **1.** | **TECHNOLOGY** | | |
| **1.1.** | The scraper assumes that the data provided from the Stars Planner has a particular structure. Drastic changes in this structure would cause the scraper to not work as per intended | **Low** | **Low** |

| | | | |
|---|---|---|---|
| **1.2.** | The database used in the system cannot process as many transactions per second as expected | **Low** | **Medium** |
| **1.3.** | Server crashes and essential data is lost | **Low** | **High** |
| **1.4.** | Google Login API malfunctions | **Low** | **Medium** |
| **1.5.** | Reused software components might have bugs and defects which may break the code | **Medium** | **High** |
| **2.** | **PEOPLE** | | |
| **2.1.** | The developers are occupied with multiple projects at the same time and may be unavailable to attend possible meetings | **Medium** | **Medium** |
| **2.2.** | Conflicts between the developers and project manager | **Low** | **Medium** |
| **2.3.** | As the project duration is short, a team member falling ill and needing some time off would drastically delay project milestones | **Medium** | **Medium** |
| **2.4.** | Undermined motivation within the team members and developers | **Low** | **Low** |
| **2.5.** | Inadequate domain knowledge within developers | **Low** | **Medium** |
| **3.** | **ORGANIZATIONAL** | | |
| **3.1.** | The course project might be revamped with different components | **Medium** | **Low** |
| **3.2.** | The project management might be restructured with different people responsible for the different parts | **Low** | **Low** |

| | | | |
|---|---|---|---|
| | | | |
| **4.** | **TOOLS** | | |
| **4.1.** | The code generated by mobile SDK might be inefficient for actual deployment | **Low** | **High** |
| **4.2.** | The use of Firebase for data storage might be insufficient for the user data | **Medium** | **Medium** |
| **4.3.** | The deployment servers might not be scalable enough to meet the demands of peak user traffic | **Low** | **High** |
| **5.** | **REQUIREMENTS** | | |
| **5.1.** | Changes in requirements are proposed to improve the quality of the software at late stage of development | **Medium** | **Medium** |
| **5.2.** | New use cases arise at late stage of development, giving rise to new requirements to be incorporated | **Medium** | **Medium** |
| **5.3.** | Customers fail to understand the impact of requirement changes | **Medium** | **Medium** |
| **6.** | **ESTIMATION** | | |
| **6.1.** | The time required to develop the various components of the software is underestimated | **Medium** | **Medium** |
| **6.2.** | Abandoning of planning under pressure | **Low** | **Medium** |
| **6.3.** | The rate of bugs and issues in the system might be underestimated | **Medium** | **High** |
| **6.4.** | The size of the software application with be underestimated | **Low** | **High** |

| | | | |
|---|---|---|---|
| 6.5. | The user base of the application might be under or over-estimated, hence adversely impacting the design decisions | **Low** | **Medium** |

***Table 2: Risk Analysis for NTUCollab***

# 5. RELEASE APPROACH

## 5.1. Rationale

Team Eagles has used the **Waterfall** model as the ideal choice of lifecycle for development of NTUCollab software. The main reason for using the Waterfall model is because it is much more structured than other traditional methods wherein each phase must be completed before the next phase of development. Further, it is relatively easy to accommodate any changes made to the software during the development process. Given the cost constraints, as mentioned in the Project Plan, the Waterfall model will help to decrease costs associated with the development process.

Since the release strategy adopted depends on the type of development lifecycle, the release approach specified here will be in accordance with the Waterfall methodology. Extensive testing coupled with thorough documentation between any two phases of the development process will be carried out, to ensure that NTUCollab remains bug-free, errors, if any, are reproducible, and successive phases of development can rely on documentation to speed up the development process. Such steps will help ensure that any release developed also meets appropriate quality requirements.

## 5.2. Release Strategy

The Release Strategy employed for NTUCollab will be **Phased Function Rollout**. This involves incremental development and implementation of separate modules / functions of the system, capable of existing independently, and combining all these in the end. This strategy sits very well with the Waterfall software development lifecycle, since the Waterfall model also involves a phased approach wherein, the development can progress to the next stage only if the current phase is completely and thoroughly implemented and tested.

To further facilitate the streamlining of the release process, **Continuous Integration** followed by a **Build-Package-Deploy** process will be employed. Continuous Integration, which involves continuous merging of components and sub-systems along the way versus one big integration move, will improve quality through enforcement of best practices, the benefits of which include anyone having access to the latest build and everyone being able to see what is happening, thereby decreasing time and costs for development. We will use **Git** as the default Version Control Software (VCS) to enable continuous integration.

Prior to a release, relevant members of Team Eagles will have to ensure combining correct versions of all software configuration items, using the appropriate data, into an executable for delivery, to the users as well as to the test team. They must ensure that the entire process of building the software and then consequently releasing it, satisfies the following key requirements:

- *Complete* – self-sufficient
- *Repeatable* – automatic, consistent

- *Informative* – provides feedback
- *Schedulable* – auto-triggered
- *Portable* – independent (of IDE)

The Build-Package-Deploy process will include the following order of steps:

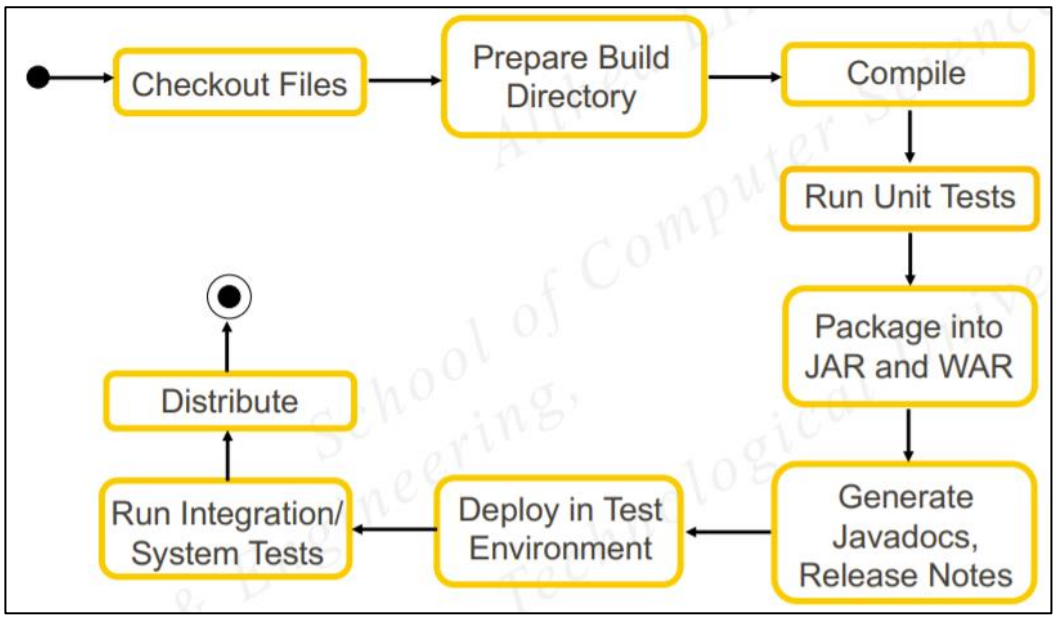


Figure 1: **Build-Package-Deploy Process** (Source - CZ3002 Lecture Notes - Dr Althea Liang)

The lack of an existing application like NTUCollab ensures that this application will indeed be widely accepted and used by the NTU community. Further, the time and resources spent in developing the application is high, indicating the intricacy involved in developing the major features and core functionalities of the application. However, the lack of an existing application also entails that NTUCollab will always have many areas of development, as the product matures, and we start receiving user feedback. Hence, it will surely involve several major and minor releases throughout the lifetime. The different types of releases involved for NTUCollab will be:

- **Major Release**: This includes the addition and updates of the most important functionalities of NTUCollab. These include features which form the basis of the application and without which, the application would not function, such as the recommendation system, tag selection, database migrations etc. A major release will indicate a significant change and improvement in the application and will be an important upgrade for the user.
- **Minor Release**: This includes smaller feature changes and minor additions, which do not impact the existing features of the application at a large scale, but only serve to improve user experience and remove significant bugs that might have been introduced as part of a previous major or minor release.
- **Revision**: This includes the smallest changes as well as minor bug fixes and improvement. The users can make do without having to download the latest revision,

that is, the importance of a revision is lesser than that of major and minor releases. Although recommended, Revision installations will be strictly optional.

- **Internal Release (Promotions)**: Throughout the development lifecycle of NTUCollab as well as for internal **beta-testing**, successive versions of the application would be made available within Team Eagles. Such internal releases / promotions are absolutely essential before a major / minor release or a revision is made available to the users, to remove any bugs or errors discovered during testing by members not involved with the actual development.

The preparation for any type of release in Git (our choice of Version Control System) will be through branches and will involve several steps like tagging releases, distribution files etc. This process will remain consistent for all releases.
The detailed steps involved are as follows:

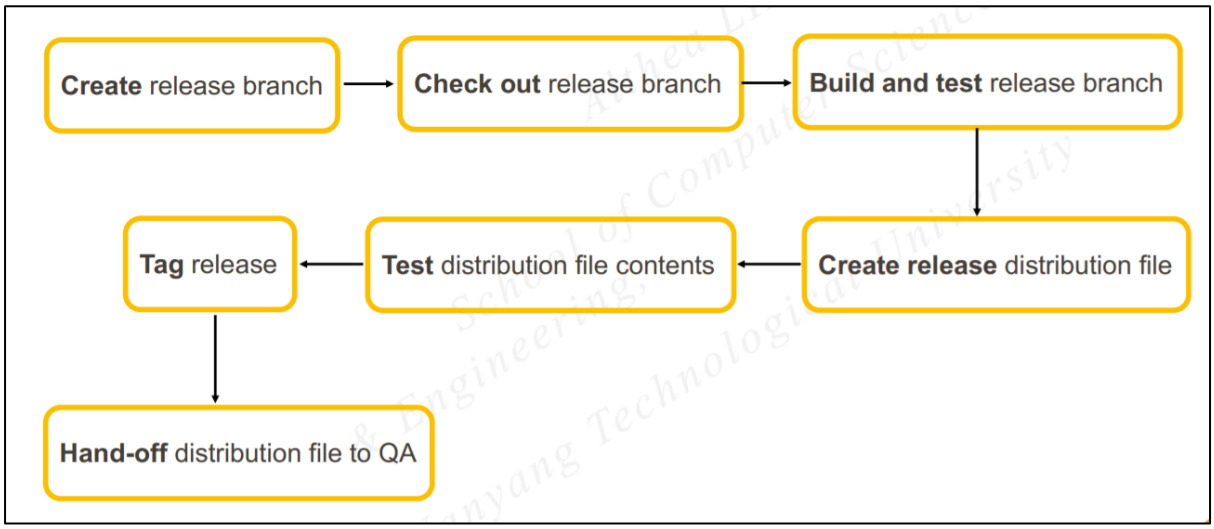


Figure 2: **Release Preparation for NTUCollab in Git** (Source - CZ3002 Lecture Notes - Dr Althea Liang)

### 5.2.1. Release Content

Based on NTUCollab's System Requirements Specification and Use Case Descriptions, following are the intended functionalities of the application:

1. **User Registration and Login**
   This use case intends to allow the user to login and register for a personal profile on NTUCollab. An independent profile is meant to tune the recommendation system to provide user – specific recommendations, based on his / her preferences.

2. **Select Preferences by selecting relevant tags**
   This use case allows the user to select tags of interest from each of modules, clubs and groups. The selected preferences are stored in the database, for use by the recommendation engine.

3. **Module, Club and Interest Group Recommendation**
   This use case includes displaying the recommendations to the user, provided by the in - house recommendation engine.

4. **Edit user profile**
   This use case allows the user to edit his / her profile, specifically interest tags to start receiving new recommendations. This is incorporated as the interests of a user can evolve over time.

5. **Discussion Forum**
   This allows the users to interact with each other and aims to form a closely knit community within the application itself. This will help bring the NTU community closer as well as increase interactions leading to the exchange of ideas and advice.

6. **Create new interest group**
   As not all interest groups will be available in the database, the user will be provided an option to create his/ her own interest group to let other like – minded users join their groups as well. Since the application can perfectly function without this feature, it will be made available in a later release.

7. **Filter and Search Recommendations**
   This use case is a helper feature to allow the users to search for relevant categories effectively and efficiently.

8. **Display details of relevant modules, clubs and groups**
   This use case aims to provide more detailed insights about each module, club and group such as ratings and descriptions.

On the basis of the above, following are the releases and the content of each:

| Release Version | Release Type | Release Content |
|---|---|---|
| 0.0.0 | Internal (within Team Eagles) | Use Cases 1 to 4 (as mentioned above) to be ready for testing within Team Eagles. These use cases are the bare essential for NTUCollab and hence need to be implemented and tested before a user-facing release is made |
| 0.0.1 | Internal (within Team Eagles) | Use Cases 6 and 7. Since these are secondary features, they nevertheless add great value to the application and hence should be implemented and tested before |

| | | |
|---|---|---|
| | | the first major release is made available to the users |
| 1.0.0 | Major (to the users) | This will include all the functionalities that were implemented and tested by internal release version 0.0.1<br><br>With this release, NTUCollab will be made available on all major application stores like PlayStore and AppStore. After a few weeks, Team Eagles will begin collecting important information like adoption rate of the application, number of downloads and user feedback. These would help to improve the application through future releases.<br><br>Concurrently, a forum where users can identify and report bugs in a release will be opened. This will help in fast and easy error-identification and increase test coverage that includes test cases. |
| 1.1.0 | Minor (to the users) | This will include tested implementations of Use case 8. Relevant notification mechanisms will be employed to let the users know of the availability of this important but optional feature.<br><br>Again, relevant statistics like user outreach about this release, number of new downloads etc. will be made so that Team Eagles can make necessary amendments for future releases. |
| 1.1.X | Revisions (includes Patches) | As mentioned earlier, bugs and feedback collected earlier will be continuously worked on and maintained. The updated versions of software will be available through successive revisions, that users can download as per convenience. |

### 5.2.2. Release Schedule

The Release Schedule is an important aspect for NTUCollab as the frequency of releases can determine the user satisfaction. If the frequency of releases is too low users might be dissatisfied with the product as a whole as they might assume it's not maintained and does not have regular updates to add new features. However, too frequent releases can irritate users and they might choose to skip releases, thereby making them use older version of the application.

The following is an approximate schedule for the different releases specified above:

| Release Version | Date of Release |
|---|---|
| 0.0.0 | 30th March, 2021 |
| 0.0.1 | 20th April, 2021 |
| 1.0.0 | 27th April, 2021 |
| 1.1.0 | 20th May, 2021 |
| 1.1.X | Recurring monthly revisions and patches |

The above timeline has been chosen to ensure timely completion of NTUCollab's first major release as well as at least a week for testing and bug identification. Well – spaced out minor releases and patches will help ensure users do not get irritated by frequent updates and at the same time are assured that continuous maintenance and improvement of the application is taking place.

### 5.2.3. Release Impacts

With every release, there are several components that are impacted by it. This includes the system, business processes, user interfaces and the overall release strategy. However, each release has a purpose and there are several goals and objectives that are met by it.

Following are the impacts of each release on each of the different domains:

| Release | Business Process Impact | System Impact | Goals and Objectives |
|---|---|---|---|
| 0.0.0 | Complete essential features of NTUCollab as a product | Updated source code ready in main branch. | Complement implementation and testing of essential |

| | | | |
|---|---|---|---|
| | | | features of the application. |
| 0.0.1 | Add product value through new features | Release branch tagged and ready for deployment in Git VCS. | NTUCollab ready for deployment to actual users |
| 1.0.0 | Product rolled out to users | Main branch now ready for new changes to be made. | First major release to users |
| 1.1.0 | Improve business value through addition of important feature. | Minor release branch tagged, and release deployed to users | First minor release to users, with optional but helpful feature of displaying important information about each module, club, and group. |
| 1.1.X | Improve user experience through bug squashes | Parallel updates of main and release trunks in Git VCS | Remove bugs, deploy patches, and introduce minor feature improvements continuously. |

In addition to the above impacts, since Team Eagles has employed a Build – Package – Deploy Process, the following machines are involved:

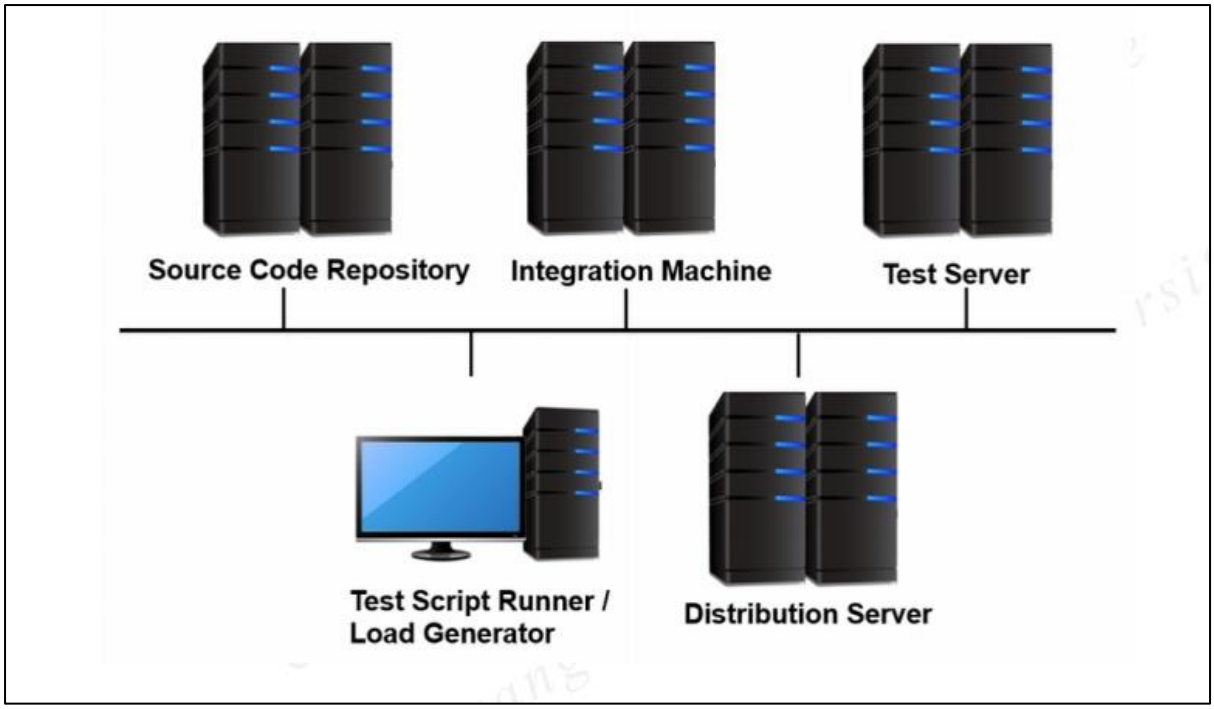


Figure 3: **Build-Package-Deploy Configuration** (Source - CZ3002 Lecture Notes - Dr Althea Liang)

Each of the above machines will be involved in any particular release. The source code repository will contain all the source code, integration machines will be responsible for integrating different sub systems together, test servers and test scripts runners will carry out effective testing prior to any release and the distribution server will be actually involved in distributing both internal and external releases.

Appropriate identification of the impact of each release is essential as it helps to improve the quality of any future release and provides a good baseline. If any release (major / minor / release) encounters an issue during the process, it will be easier to avoid the same mistake in a future release. To aid this process, Team Eagles will ensure that all releases and related processes (business impact, system changes etc.) are properly documented and made available within the team, as a reference point for the future. Appropriate steps and guidelines shall also be specified for all issues encountered in the process of each release. All such steps will help mitigate the impact of errors and prolong user satisfaction.

### 5.2.4. Release Notification

When a release version is generated, it is imperative to let the stakeholders and users of NTUCollab know about its availability. This can be achieved through a detailed and well-laid out release notification mechanism. Notifying the users, a of release ensures that the users know that the application is being upgraded and improved continuously and that every new release will ensure a better user experience.

The release notification mechanism involves the use of the following media:

- **Push notifications** on the phone from the relevant application store that NTUCollab will be available on (AppStore for iOS, PlayStore for Android)

- **In-app notifications** so that users are reminded every time they use the application
- Email and text message reminders for users who have subscribed for the same
- Detailed information upon user request through customer hotlines
- **Firebase Cloud Messaging SDK** for notifications over cloud to relevant users. This uses the following architecture to achieve **high throughput** and **maximum reach**:

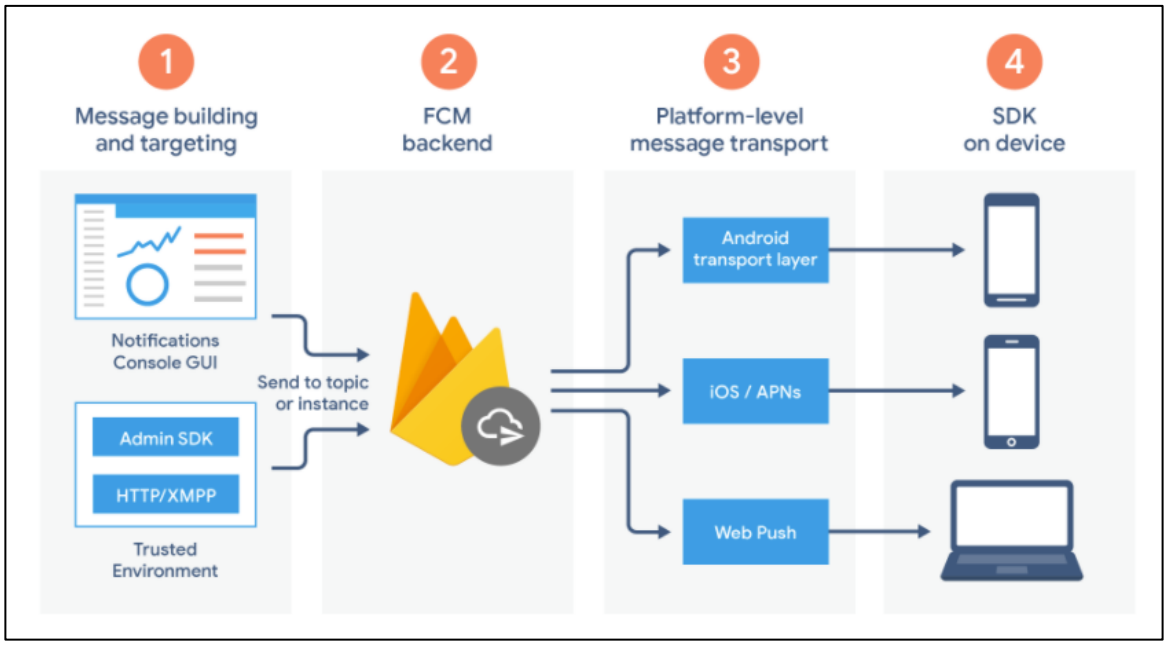


Figure 4: **Firebase Cloud Messaging Architecture** (Source: https://firebase.google.com/docs/cloud-messaging/fcm-architecture)

Such extensive methods of notification will ensure that users stay up-to-date and are do not miss out any release, thereby ensuring a better user experience.

Apart from the above methods that are intended for the users of NTUCollab, it is imperative to notify the entire team about the approval of a new release and the changes it involves. This is to ensure that all the members of Team Eagles are aware of the new version, even if they did not directly work on it, to minimize time to pick up work from where it was left off. The primary mode of notification for team members will be official Email IDs, team meetings and briefings as well as text messages on the registered mobile phone numbers.

Further information regarding the information provided, the timeframe etc. will be provided as below:

| **Stakeholder** | **Information included in the notification** | **Timeframe for receipt of notification** |
|---|---|---|
| Users | Changes made, which includes features updated, features added, bug fixes made, how the new | One day prior to the release of version, followed by recurring reminders in the application |

| | | |
|---|---|---|
| | changes will positively affect the user | (if appropriate permissions are provided by the user) |
| NTUCollab Team (notification of internal promotions) | All the information to bring the entire team up to speed, with important information about architecture changes, availability of documentations, team members involved in the development of that release as well as user-side changes made in the application | Immediately after the approval of the new release |

# 6. GLOSSARY

| Term | Definition |
|---|---|
| Use case diagram | Simple representation if user's interaction with the system. |
| Waterfall lifecycle method | A linear lifecycle method used for the development of software systems. |
| Release | A particular version of the application being made available for the users |
| Modules | A subject taken up as part of course work at NTU. |
| Firebase Cloud Messaging | A backend notification service |
| Flutter | A UI tool kit provided by google |
| Build-Package-Deploy | A standard procedure involving building the application, packaging it into one unit and deploying it for users |

# 7. ACRONYMS

| Acronym | Definition |
|---|---|
| CCA | Co-curricular Activities |
| IDE | Integrated Development Environment |
| NTU | Nanyang Technological University |
| SDK | Software Development Kit |
| STAR | Student Automated Registration System |
| VCS | Version Control System |

# 8. APPENDICES

NTUCollab Project Proposal: https://172.21.149.196/svn/3002/B2/Eagles/Lab-1/Project_Proposal.pdf

NTUCollab Project Plan: https://172.21.149.196/svn/3002/B2/Eagles/Lab-3/Documents/Project_Plan.pdf

NTUCollab Quality Plan: https://172.21.149.196/svn/3002/B2/Eagles/Lab-2/Quality_Plan.pdf

NTUCollab System Requirement Specification: https://172.21.149.196/svn/3002/B2/Eagles/Lab-2/System_Requirement_Specifications.pdf