

# **Project Visualization System**

## **Software Design Document (SDD)**

**Version: 1.0**

**Team#3**

<b>Name</b>	<b>ID</b>	<b>E-mail</b>
劉恒育	107590007	t107590007@ntut.org.tw
莊 永	107590005	t107590005@ntut.org.tw
鄭立杰	107590013	t107590013@ntut.org.tw
黃詩涵	107AB0008	t107ab0008@ntut.org.tw

**Department of Computer Science & Information Engineering  
National Taipei University of Technology**

**12/12/2021**

## Table of Contents

<b>Table of Contents .....</b>	<b>1</b>
<b>Section 1 Introduction .....</b>	<b>2</b>
1.1 Scope of the System.....	2
1.2 Purpose of the Document.....	2
1.3 Overview of the Document.....	2
<b>Section 2 System Requirements.....</b>	<b>3</b>
2.1 Functional Requirements .....	3
2.2 Non-Functional Requirements .....	3
2.2.1 效能需求 (Performance Requirements).....	3
2.2.2 資安需求 (Security Requirements).....	3
2.2.3 相容性需求 (Compatiability Requirements) .....	4
2.2.4 測試需求 (Test Requirements) .....	4
<b>Section 3 Design Constraints and Solutions .....</b>	<b>5</b>
3.1 Technical Solution Criteria .....	5
3.2 Alternative Solutions.....	5
3.3 Selected Solution .....	5
<b>Section 4 System Architecture .....</b>	<b>6</b>
4.1 UMS.....	6
4.2 PRMS .....	7
4.3 RCS.....	8
4.4 RVS.....	8
<b>Section 5 Detailed System Design .....</b>	<b>9</b>
5.1 System Design Models .....	9
5.1.1 Use Case Model.....	9
5.1.2 Static Models (Structural Models) .....	10
5.1.3 Dynamic Models (Behavioral Models) .....	14
5.2 Interface Design .....	17
5.2.1 API Design .....	17
5.2.2 Internal Interface Design .....	18
5.2.3 External Interface Design .....	18
5.2.4 User Interface Design .....	19
<b>References.....</b>	<b>23</b>

## Section 1 Introduction

### 1.1 Scope of the System

Project Visualization System 是為了輔助使用者有系統地查看開發過程中的專案狀態。本系統主要分為以下四個子系統：

- UMS：負責使用者註冊、登入驗證以及第三方帳號連結。
- PRMS：負責建立、管理 Project 以及 Project 內的 Repository。使用者可通過此系統建立要管理的專案，並連結多個 Repository，如 GitHub 上的 Repository 之 Issue、Commit、SonarQube 上的報表等，藉此多方面瞭解專案開發的狀態與團隊效率。
- RCS：負責發送 API 從第三方平台(Github、SonarQube 等)中取得 Repository 資訊並儲存。
- RVS：負責將 Repository 資訊整理，並以各種不同方式展示圖表。

### 1.2 Purpose of the Document

本文件目的為提供開發團隊足夠的系統設計描述，詳細描述各子系統的規格、架構、流程，以及各子系統之間如何互動，使參閱本文件的開發人員可以迅速了解要建構的內容和預期的建構方式。

### 1.3 Overview of the Document

- 第二章描述系統需求以 User Story 的格式呈現
- 第三章決定系統實作的策略
- 第四章描述各個子系統的架構
- 第五章決定實作的細節與介面的規格

## Section 2 System Requirements

### 2.1 Functional Requirements

需求編號	優先順序	描述
FR-01	高	提供使用者進行註冊的功能
FR-02	高	提供使用者進行登入的功能
FR-03	中	提供使用者修改使用者資料的功能
FR-04	高	提供管理專案的功能
FR-05	高	提供修改專案內容的功能
FR-06	高	提供專案資料視覺化的功能
FR-07	高	提供檢視程式碼品質的功能
FR-08	高	提供檢視專案開發流程紀錄的功能

#### 2.1.1 User Stories and Acceptance Criteria

編號	描述
US-01	身為使用者，我想要能夠註冊 PVS 系統的帳號
US-02	身為使用者，我想要能夠登入 PVS 系統
US-03	身為使用者，我想要能夠修改註冊的資料
US-04	身為使用者，我想要能夠管理專案
US-05	身為使用者，我想要能夠修改專案的內容
US-06	身為使用者，我想要看到專案資料變成視覺化
US-07	身為使用者，我想要看到程式碼的品質好壞
US-08	身為使用者，我想要看到專案開發的流程紀錄

### 2.2 Non-Functional Requirements

#### 2.2.1 效能需求 (Performance Requirements)

需求編號	優先順序	描述
PR-01	高	系統需提供 100 人同時使用
PR-02	高	系統需承受每秒 1 筆的資料流量

#### 2.2.2 資安需求 (Security Requirements)

需求編號	優先順序	描述
SR-01	高	使用者登入系統時需要驗證帳號密碼
SR-02	高	系統連線到資料庫時需要驗證密碼
SR-03	高	系統內的操作需要經過使用者身分驗證
SR-04	中	使用者的密碼需經過加密才可儲存進資料庫

SR-05	中	資料在傳輸過程中需經過加密
-------	---	---------------

### 2.2.3 相容性需求 (Compatiability Requirements)

需求編號	優先順序	描述
CR-01	高	系統需與主流瀏覽器相容(Ex. Chrome, Edge, FireFox)
CR-02	高	頁面需能夠在 1920 x 1080 解析度下正確顯示

### 2.2.4 測試需求 (Test Requirements)

需求編號	優先順序	描述
TR-01	高	前端含有邏輯處理的程式碼需經過單元測試
TR-02	高	後端含有邏輯處理的程式碼需經過單元測試
TR-03	高	系統需經過手動整合測試
TR-04	中	系統需經過 Selenium 自動化測試
TR-05	高	系統的程式碼品質需經過 SonarQube 檢測

## Section 3 Design Constraints and Solutions

### 3.1 Technical Solution Criteria

- 開發
- 熟悉度
- 熱門度

### 3.2 Alternative Solutions

	Ruby/Sinatra	Java/SpringBoot	Python/Django
熟悉度	沒有成員使用過，皆無相關經驗	團隊成員大多有使用此項目開發的經驗	多數成員使用過。
熱門度	普通，以網頁應用程式為主	高，以後端、Android APP、跨平台應用程式為主	高，以 AI、大數據、資料科學為主
開發	開發上非常自由，由於不側重「慣例」，沒有相同的文件夾結構，所以會比較難去理解現有的 Sinatra 應用。是一種簡約框架，只提供啟動需要的最低設定。	慣例優於設定，能寫出風格與架構一致的程式碼，有很多套件可以使用。提供開發所需要的一切。如果想使用其他的東西，構建框架的開發人員已經為開發者做出了一些關鍵決策，因此可以遵循他們建議使用的模式	慣例優於設定，能寫出風格與架構一致的程式碼，很多套件可以使用。提供開發所需要的一切。如果想使用其他的東西，構建框架的開發人員已經為開發者做出了一些關鍵決策，因此可以遵循他們建議使用的模式。
維運	受限於開發語言，執行效率慢	執行效率良好	受限於開發語言，執行效率慢

### 3.3 Selected Solution

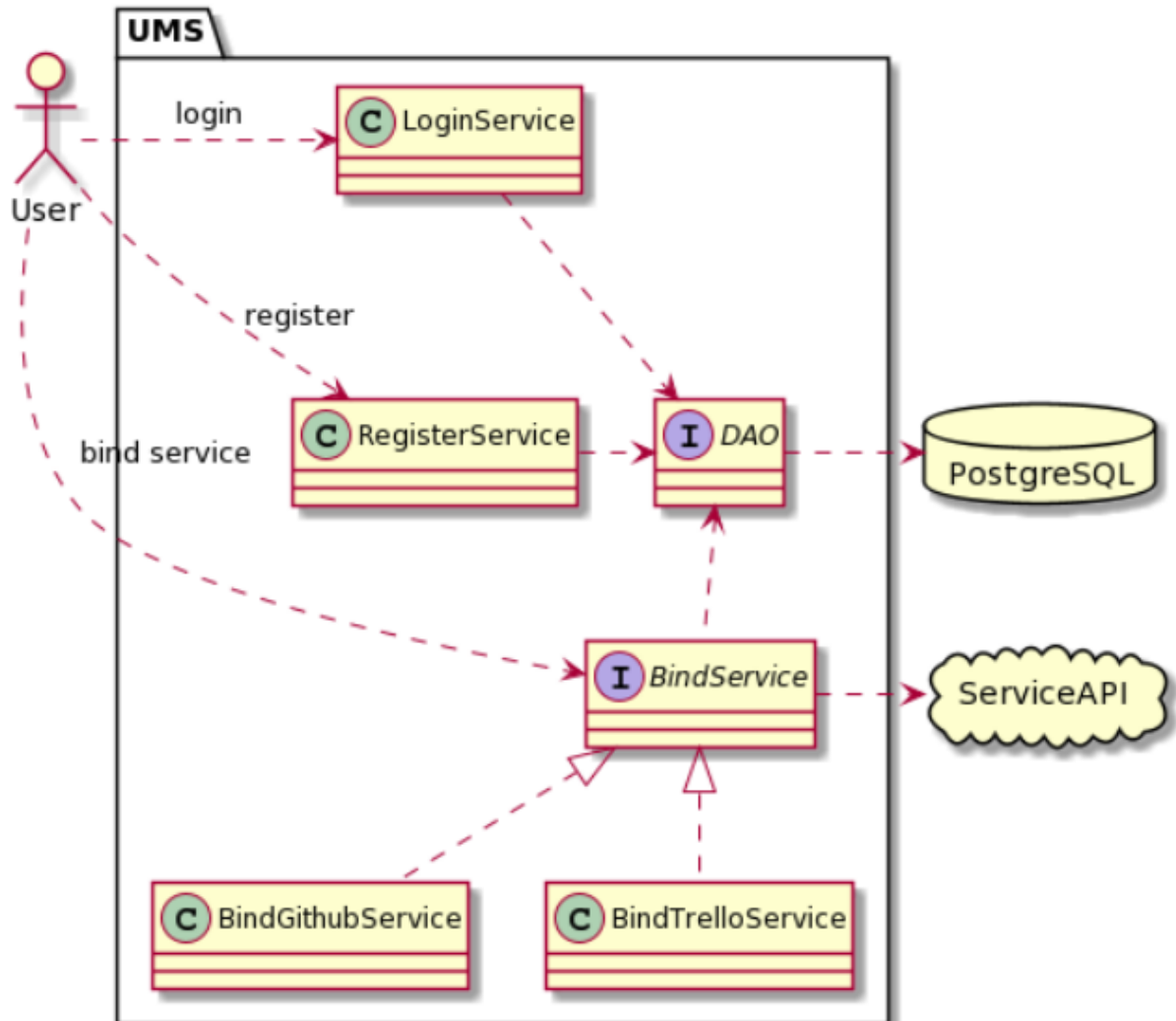
	Ruby/Sinatra	Java/SpringBoot	Python/Django
熟悉度	1	5	2
熱門度	2	5	4
開發	3	5	4
維運	2	4	2
加總	8	19	12

Priorities { Scale : 1 = Worst, 5 = Best }

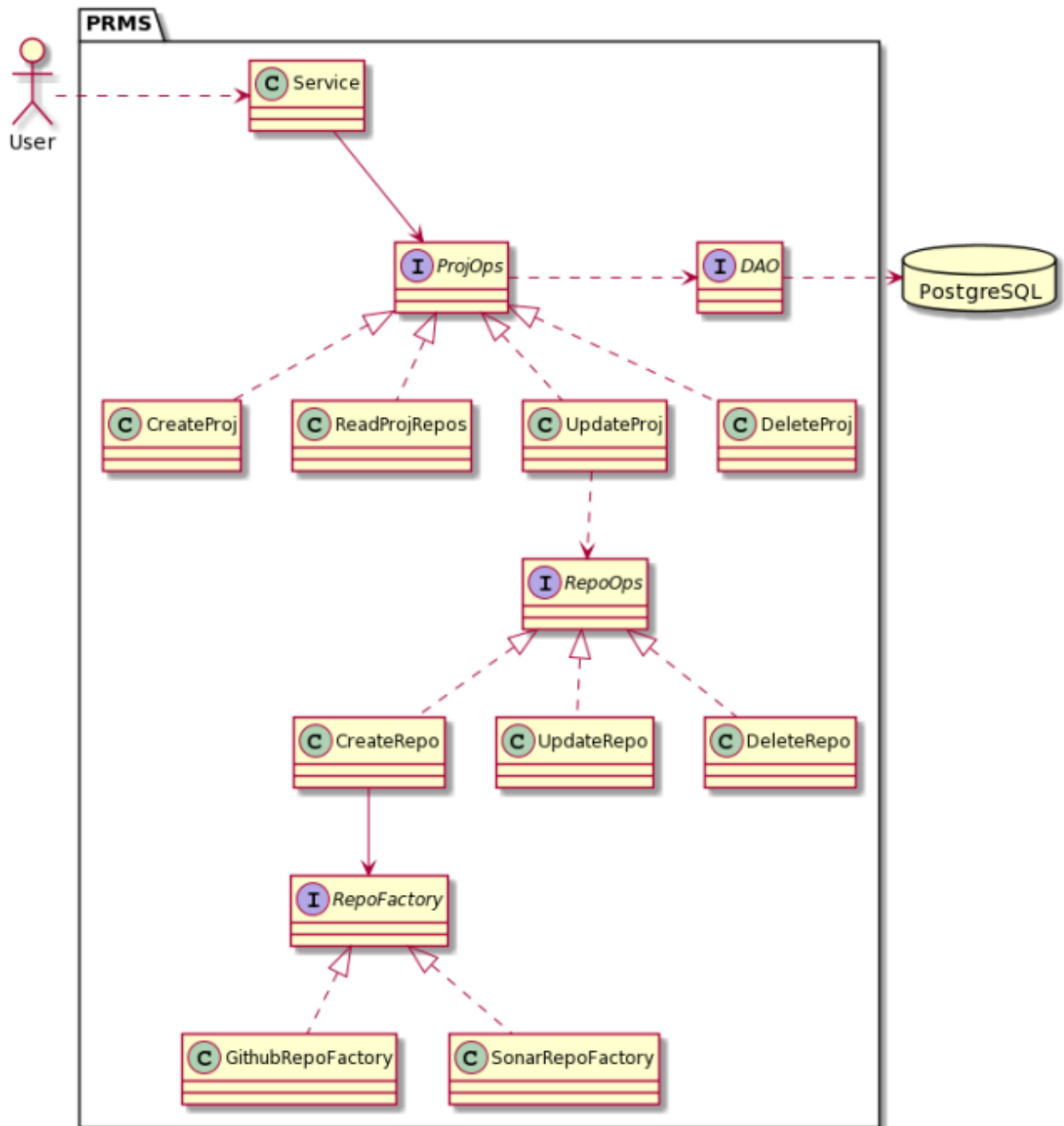
估計的方法是依據團隊成員的經驗法則。

## Section 4 System Architecture

### 4.1 UMS

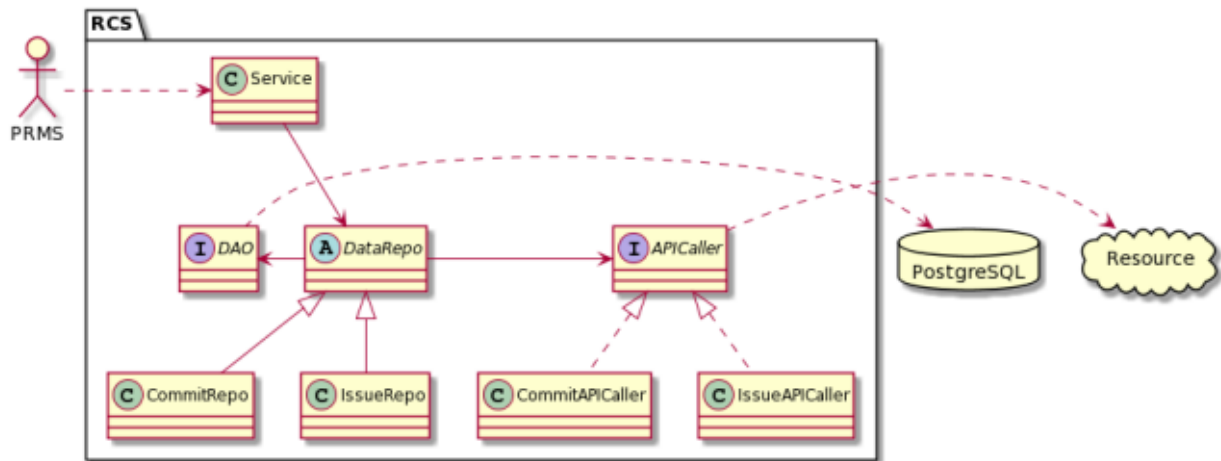


## 4.2 PRMS

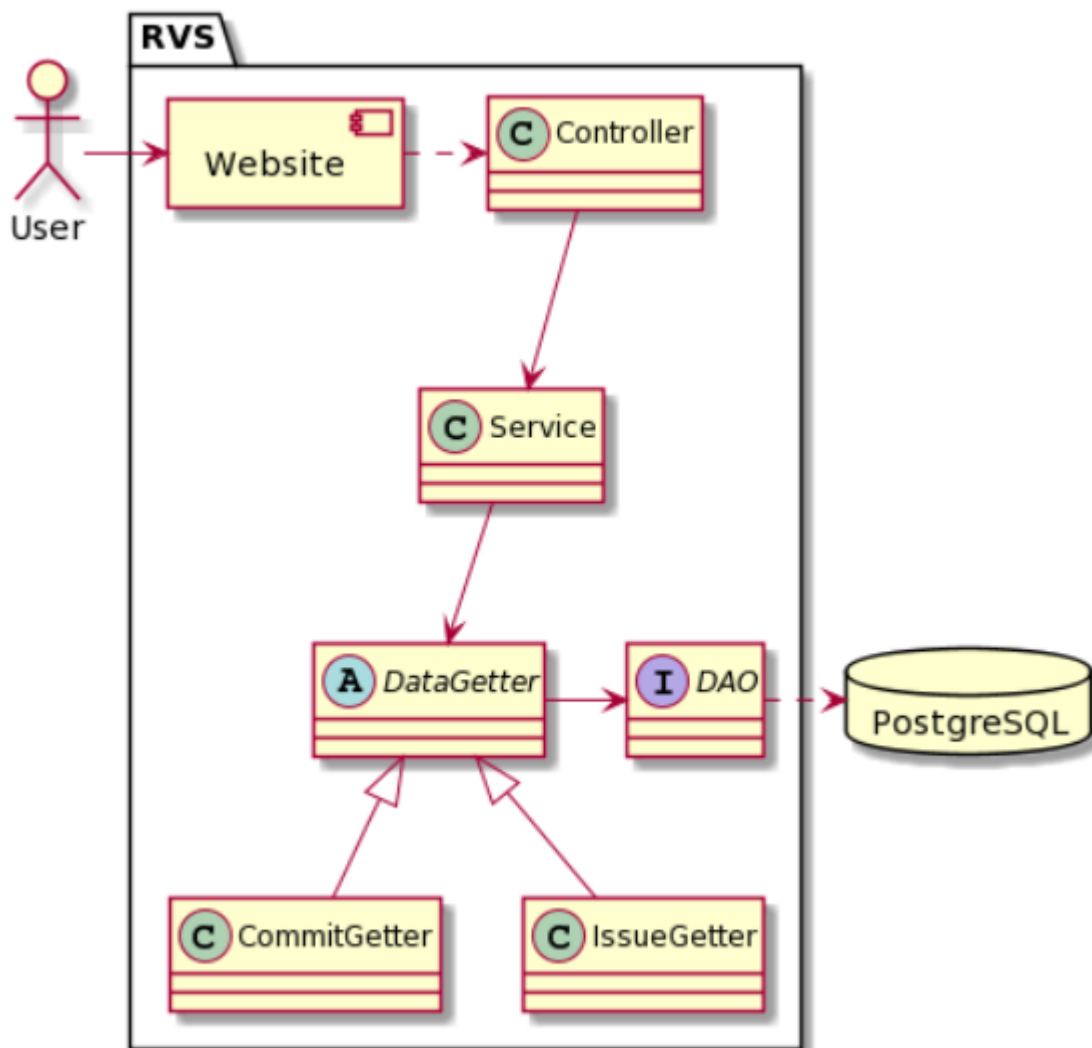




### 4.3 RCS



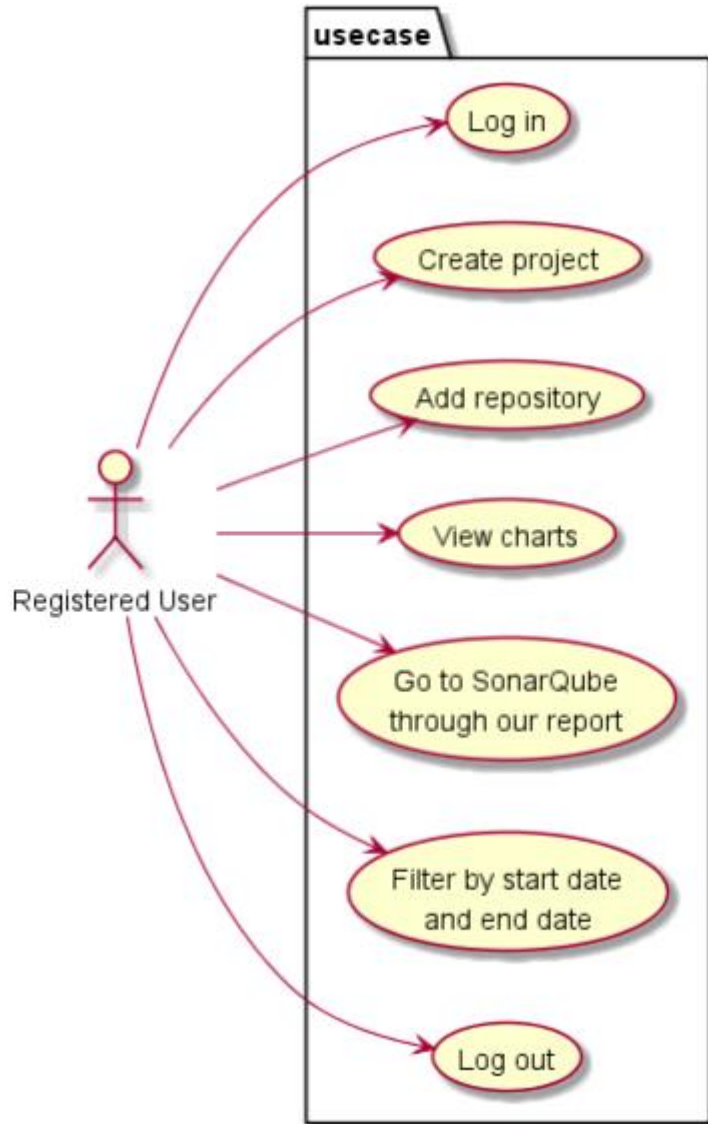
### 4.4 RVS



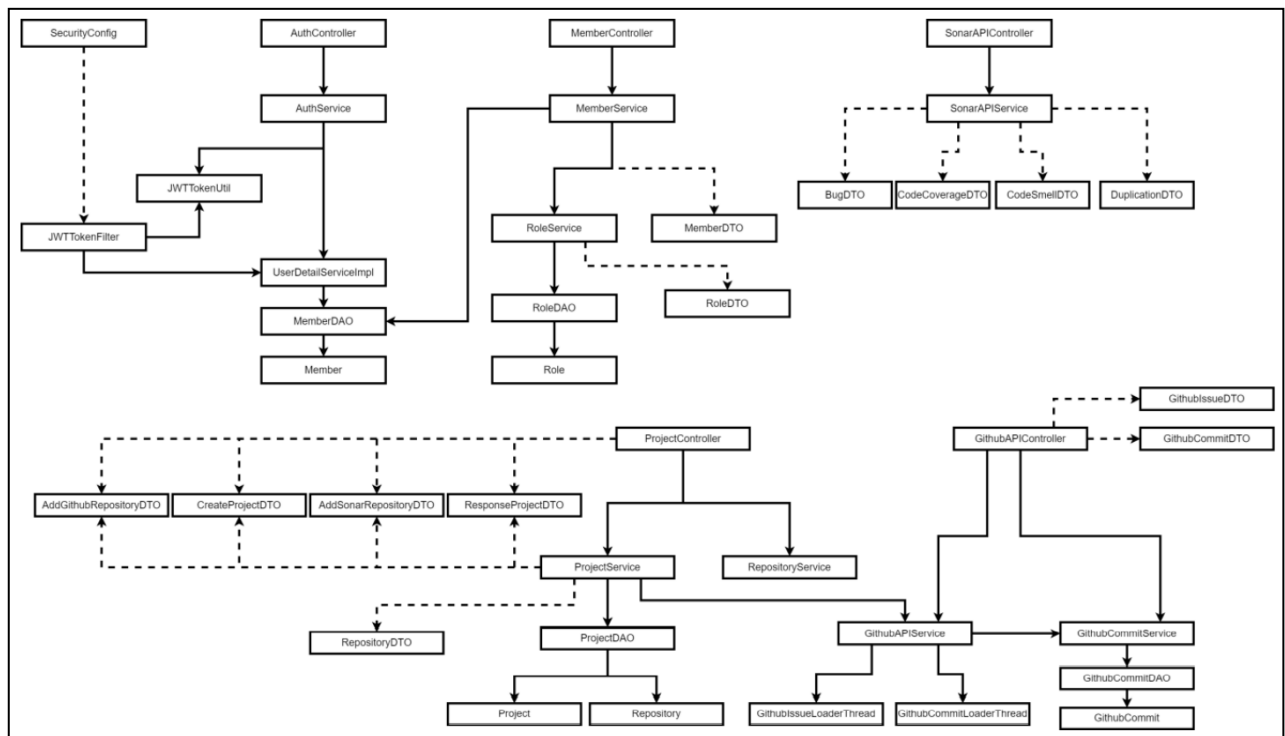
## Section 5 Detailed System Design

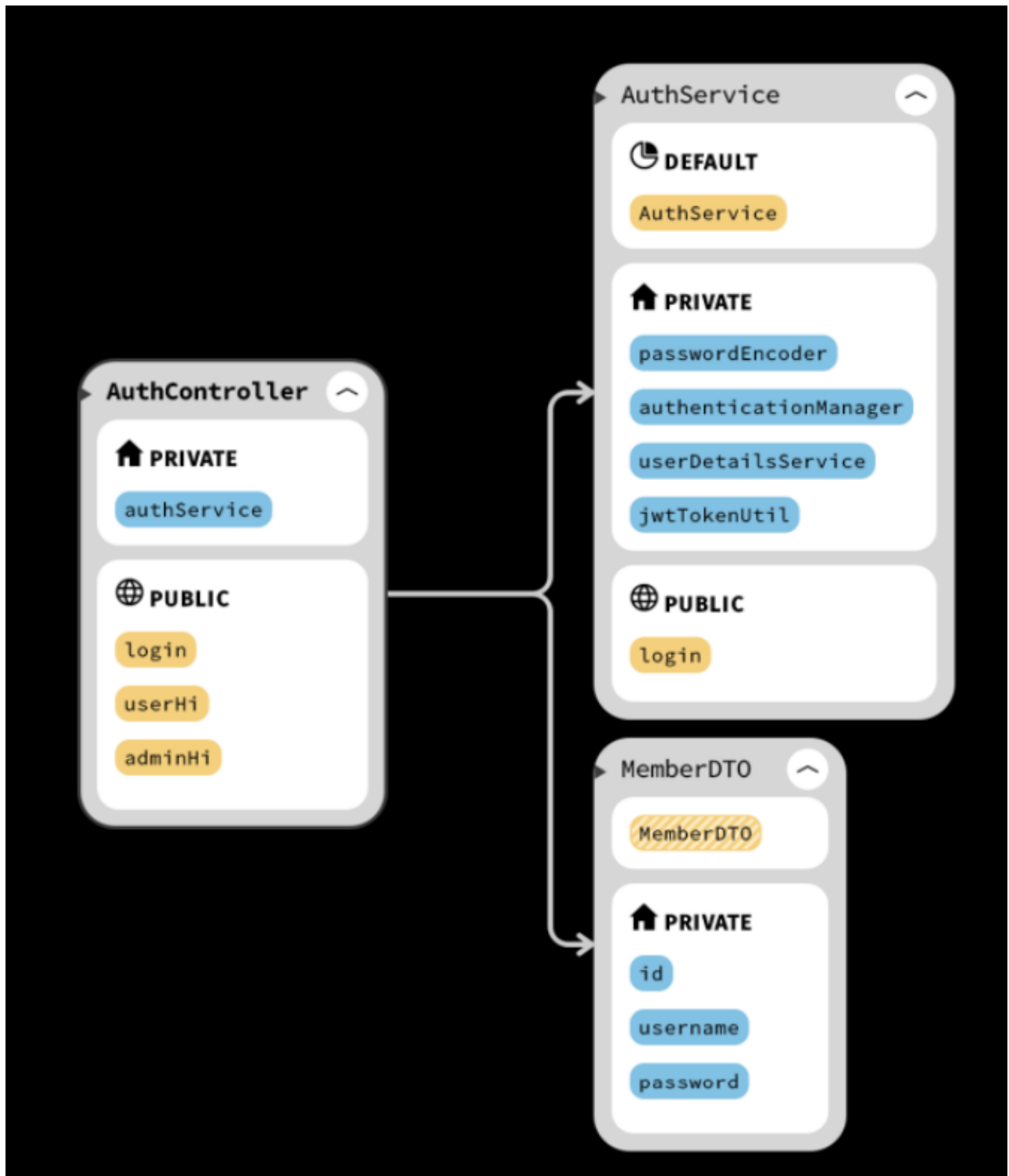
### 5.1 System Design Models

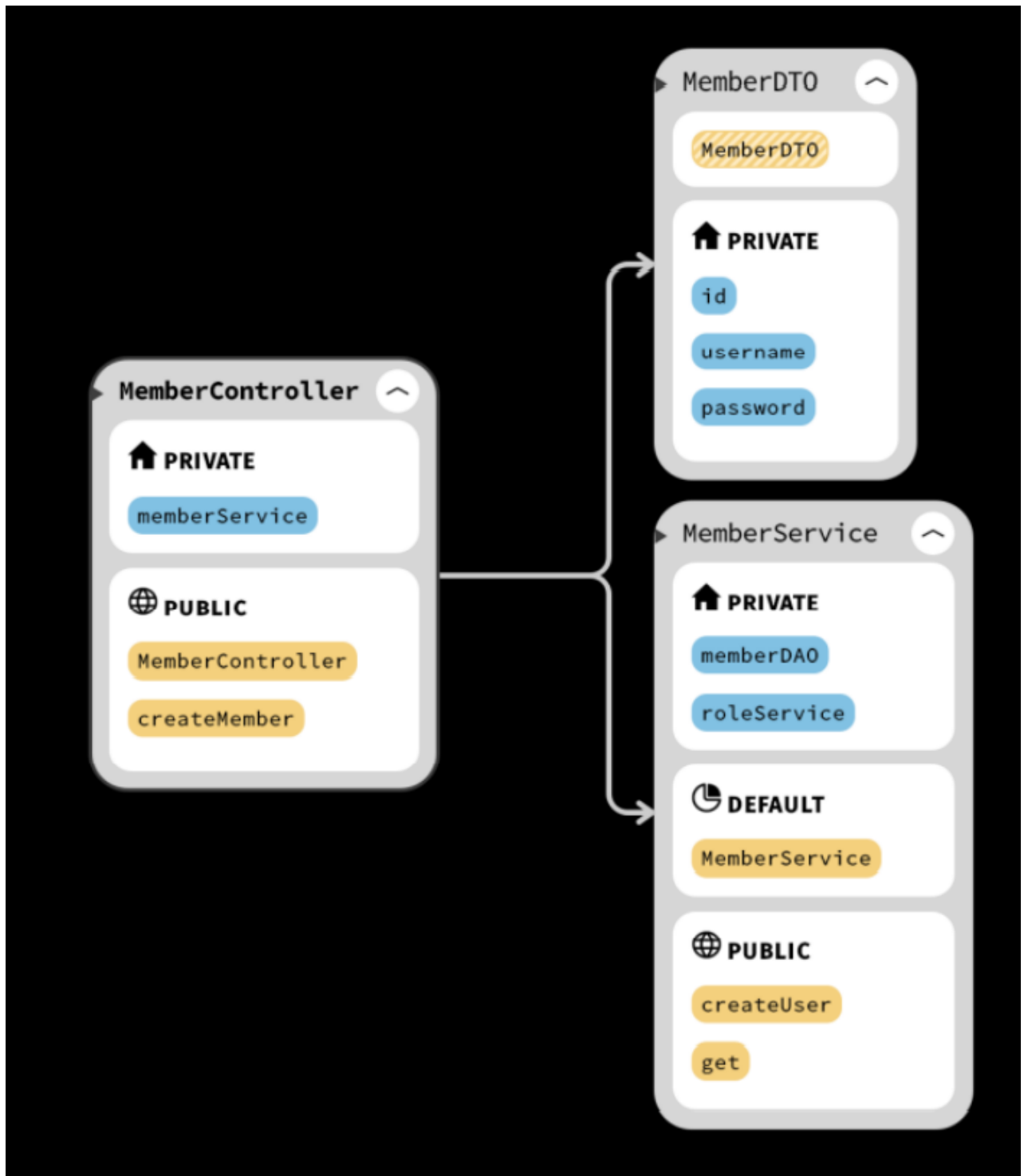
#### 5.1.1 Use Case Model

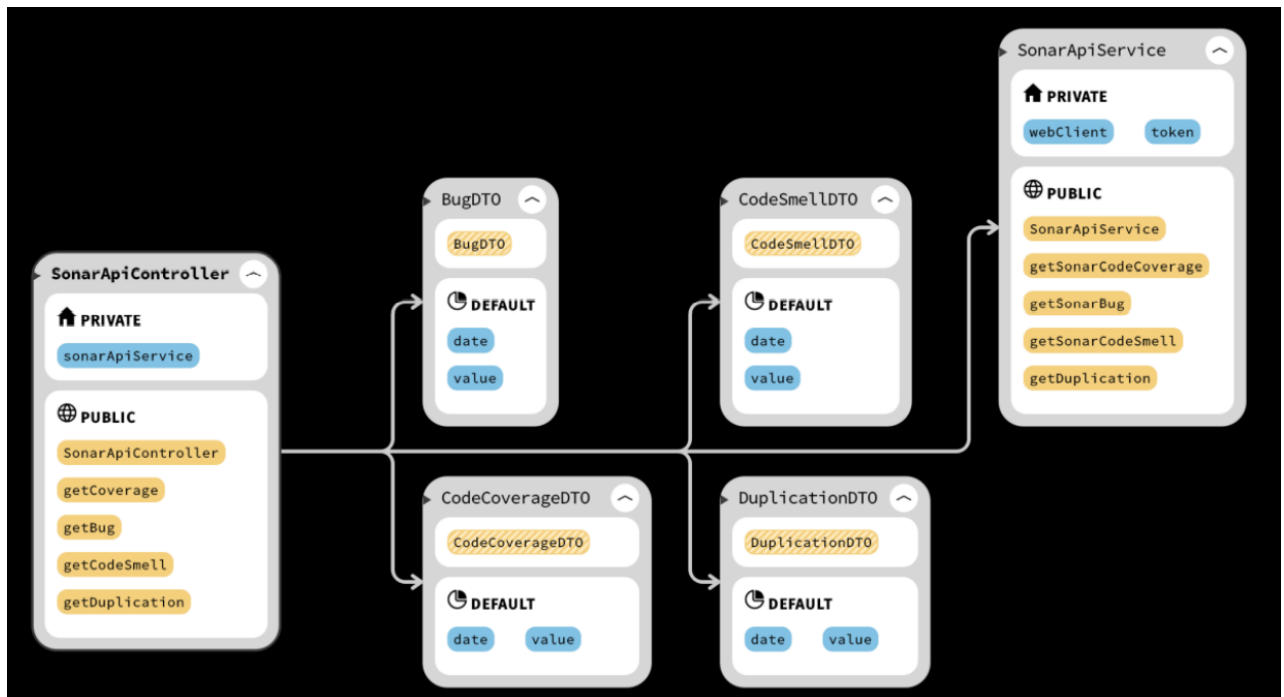
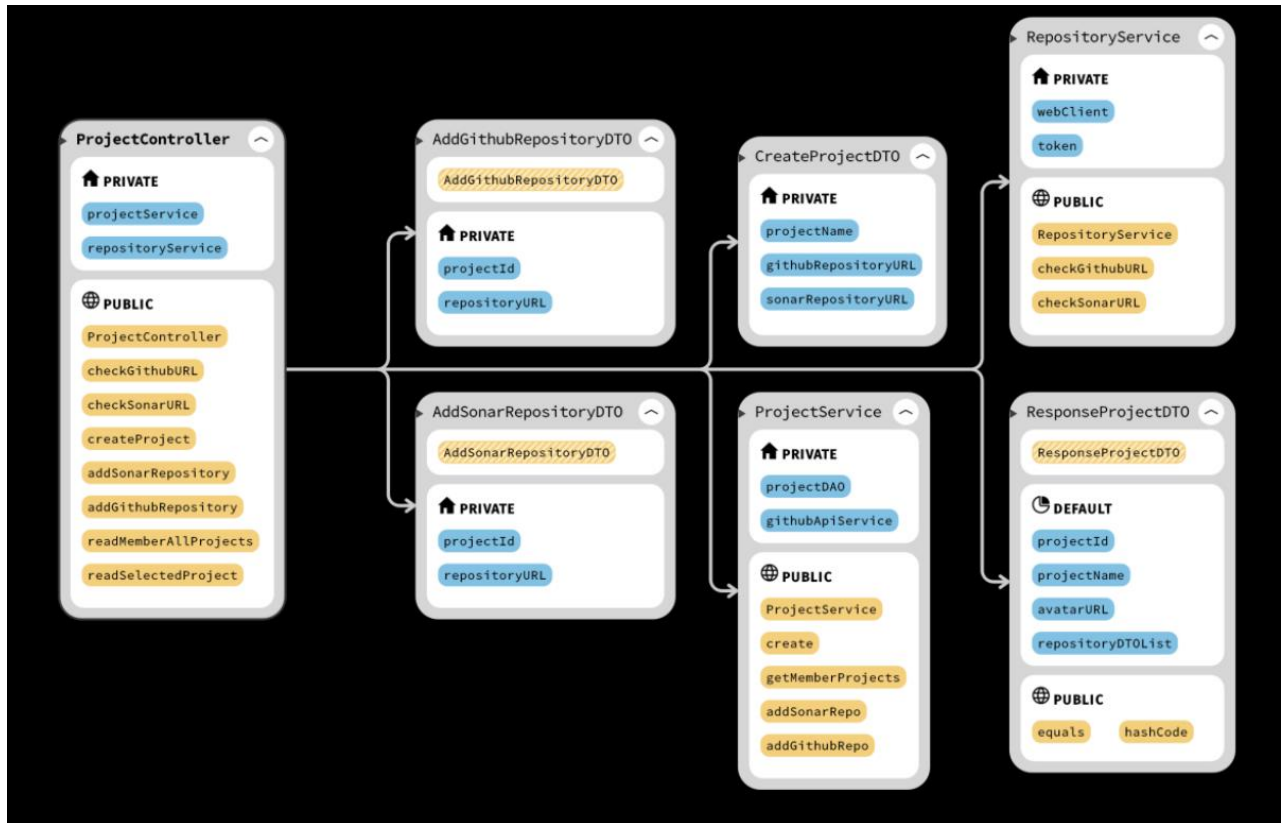


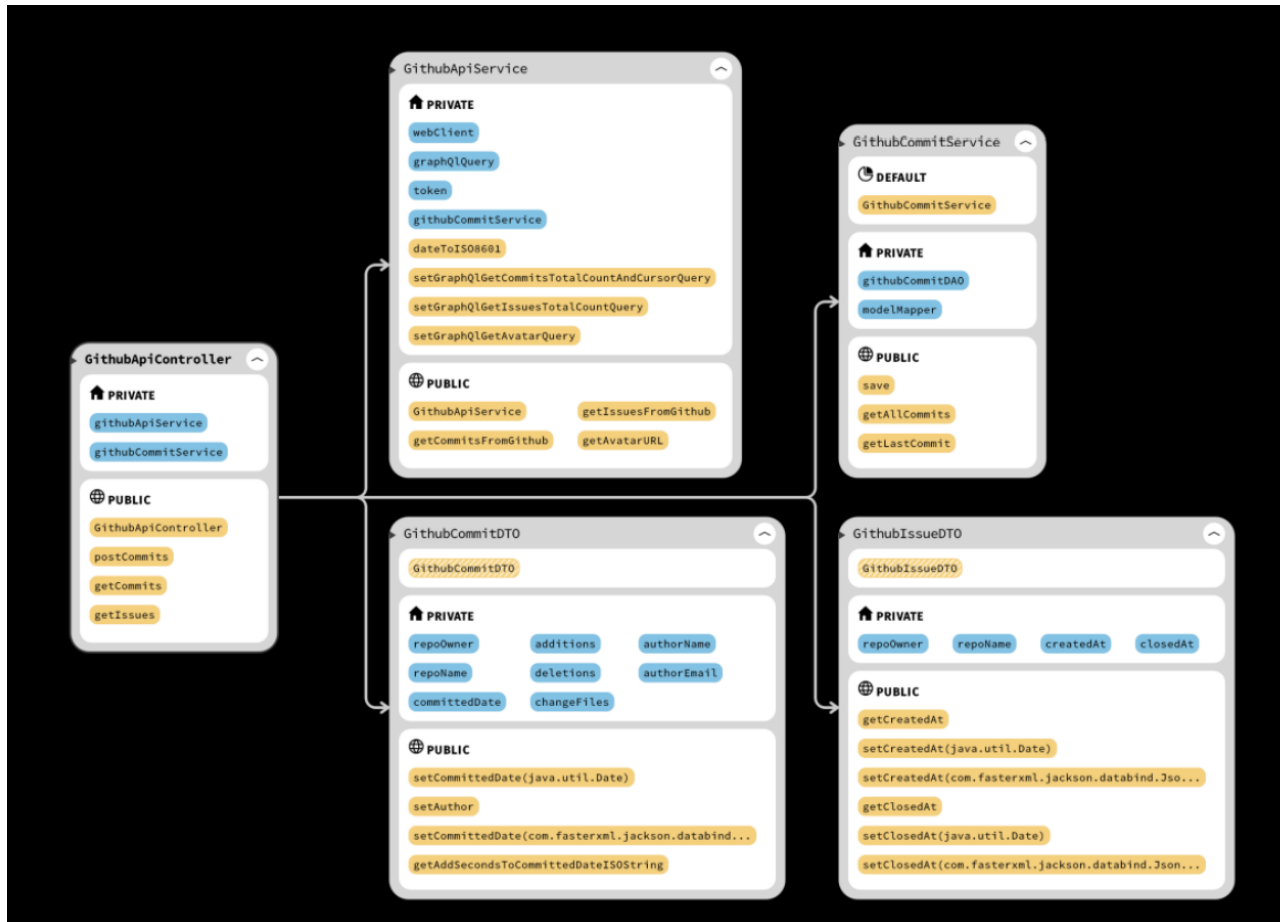
### 5.1.2 Static Models (Structural Models)



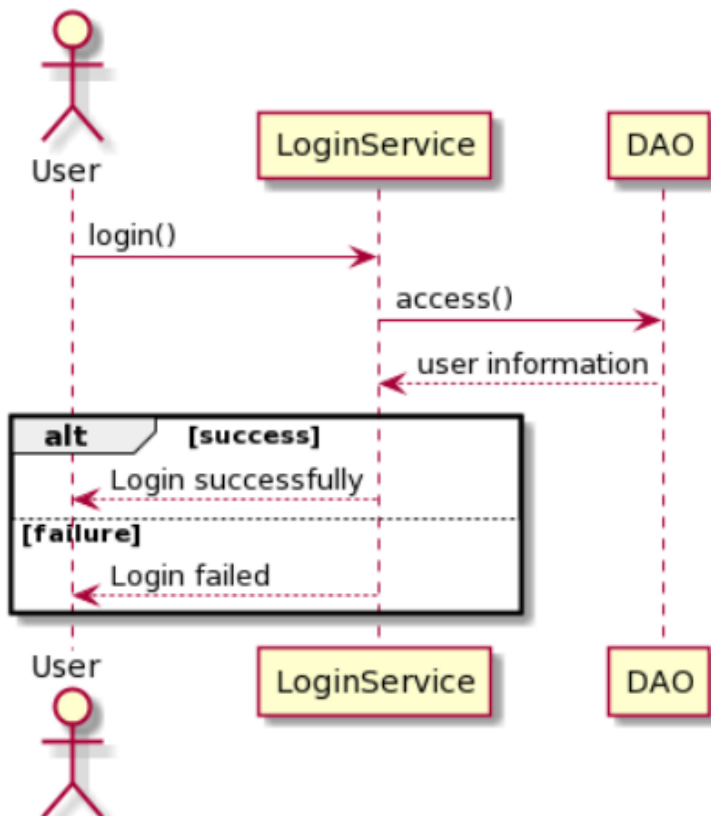


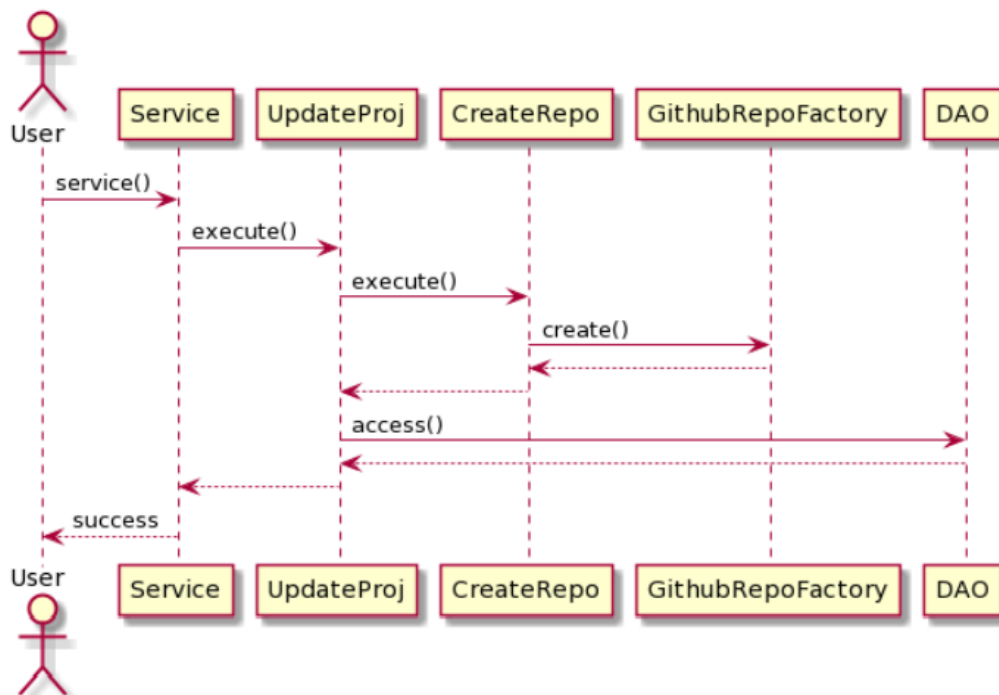
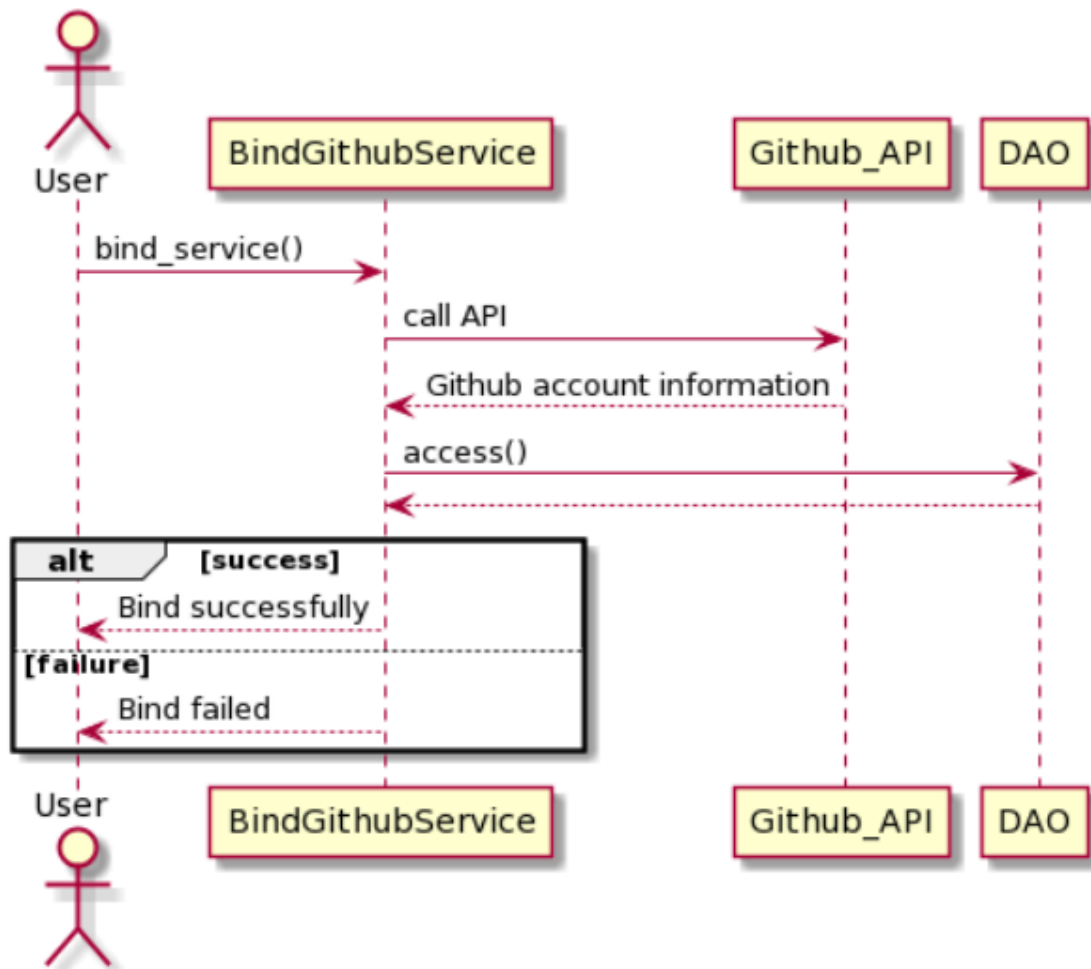




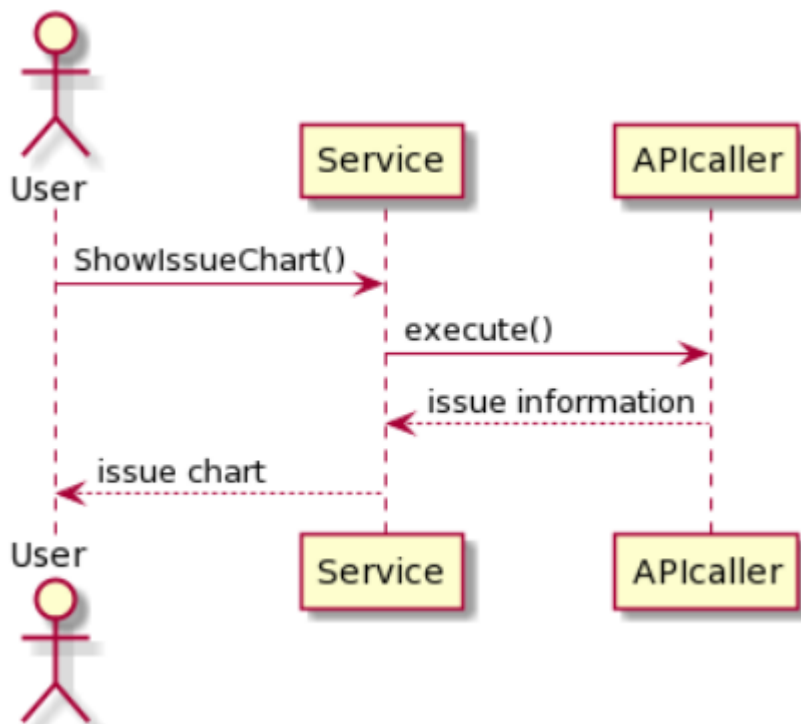
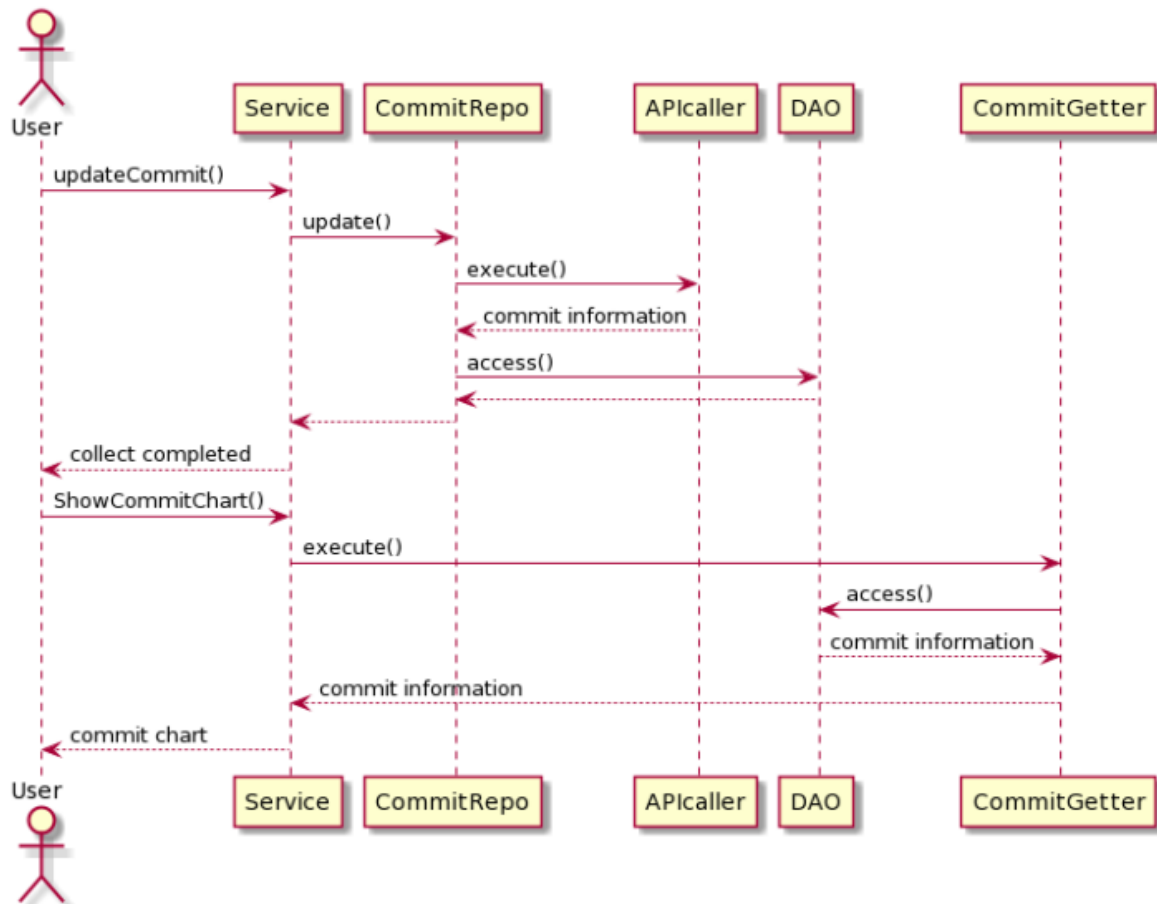


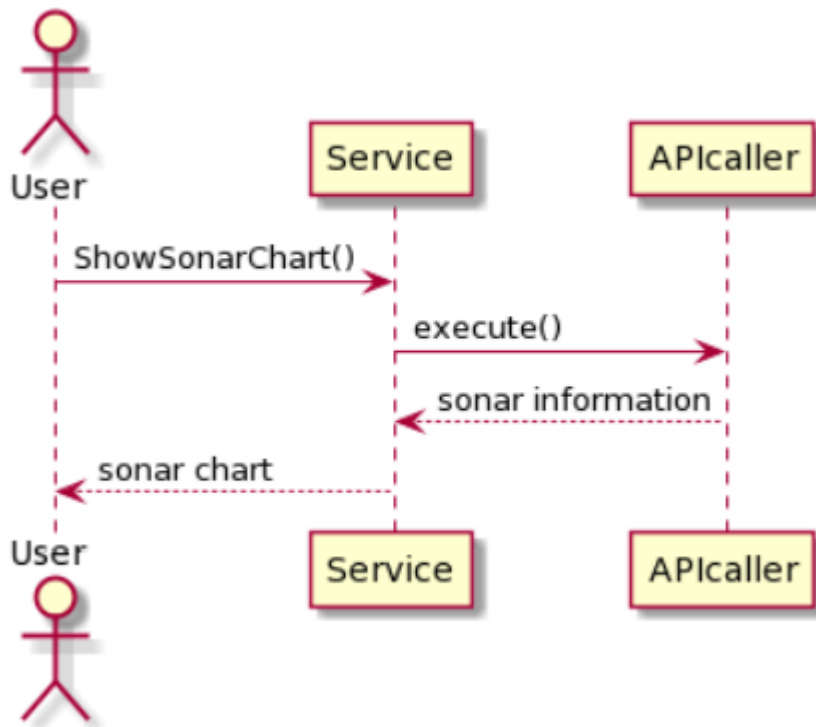
### 5.1.3 Dynamic Models (Behavioral Models)











## 5.2 Interface Design

### 5.2.1 API Design

- /member  
提供使用者進行註冊
- /auth/login  
提供使用者進行登入
- /project  
提供使用者建立一個專案
- /project/{projectId}/repository/github  
提供使用者將 Github 加入至專案
- /project/{projectId}/repository/sonar  
提供使用者將 Sonar 加入至專案
- /project/delete/repository/github  
提供使用者刪除專案內的 Github
- /project/delete/repository/sonar  
提供使用者刪除專案內的 Sonar
- (Post)/github/commits/{repoOwner}/{repoName}  
依照使用者填入的連結去獲取 Github 上 Repository 最新的 Commit 資訊
- (Get)/github/commits/{repoOwner}/{repoName}  
獲取資料庫內 Repository 的 Commit 資訊
- /github/issues/{repoOwner}/{repoName}

依照使用者填入的連結去獲取 Github 上 Repository 最新的 Issue 資訊

- /repository/github/check  
驗證使用者填入的 Github 連結是否正確
- /repository/sonar/check  
驗證使用者填入的 Sonar 連結是否正確
- /project/{memberId}  
提供使用者檢視某會員 ID 下所有的專案
- /project/{memberId}/{projectId}  
提供使用者檢視某會員 ID 下的某個專案
- /sonar/{component}/coverage  
提供使用者檢視某個 Componet 的測試覆蓋率
- /sonar/{component}/bug  
提供使用者檢視 SonarQube 檢測出來某個 Componet 的 Bug
- /sonar/{component}/code\_smell  
提供使用者檢視 SonarQube 檢測出來某個 Componet 的程式碼品質
- /sonar/{component}/duplication  
提供使用者檢視 SonarQube 檢測出來某個 Componet 的程式碼重複程度

#### 5.2.2 Internal Interface Design

- PRMS：可向 UMS 存取使用者帳戶資訊，以便取得使用者管理的 Project、Repository。
- RCS：可向 UMS 存取使用者帳戶 Token，以便發送 API 取得外部資料。
- RCS：可向 PRMS 存取使用者追蹤的 Project 及 Repository 相關資料。
- RVS：可向 RCS 存取使用者追蹤的 Repository 的相關資料，以便將資訊可視化呈現。

#### 5.2.3 External Interface Design

- UMS：需發送 API 從第三方平台取得 Token，並向 PostgreSQL 存取使用者資訊。
- PRMS：需向 PostgreSQL 存取 Project 及 Repository 資訊。
- RCS：需發送 API 從第三方平台取得資料，並將資料存取至 PostgreSQL。
- RVS：需向 PostgreSQL 存取 Repository 資訊。

## 5.2.4 User Interface Design

使用者登入頁面



LOGIN


REGISTER

PVSS




Start Month and Year  
December 2020


End Month and Year  
December 2021

Projects

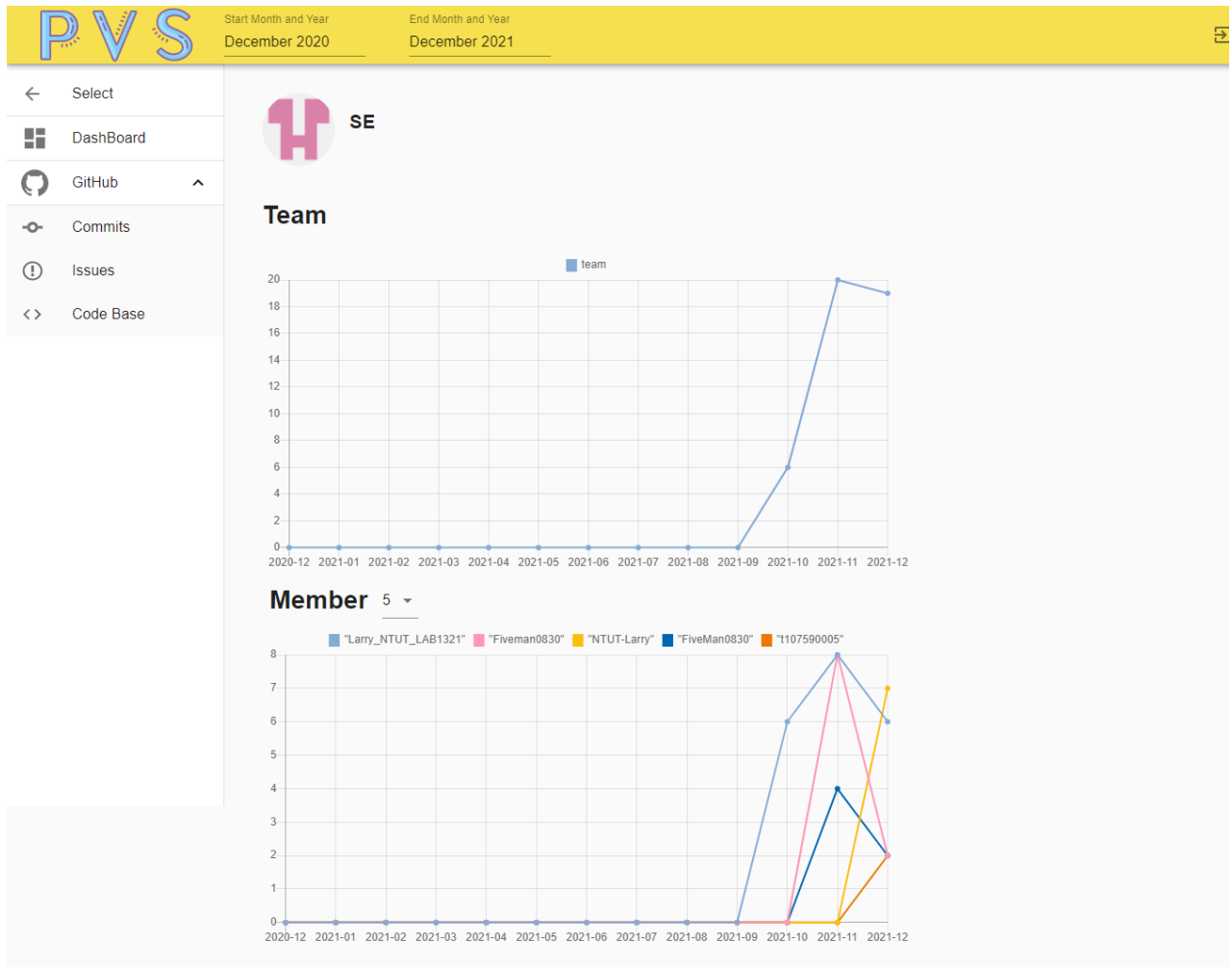


SE

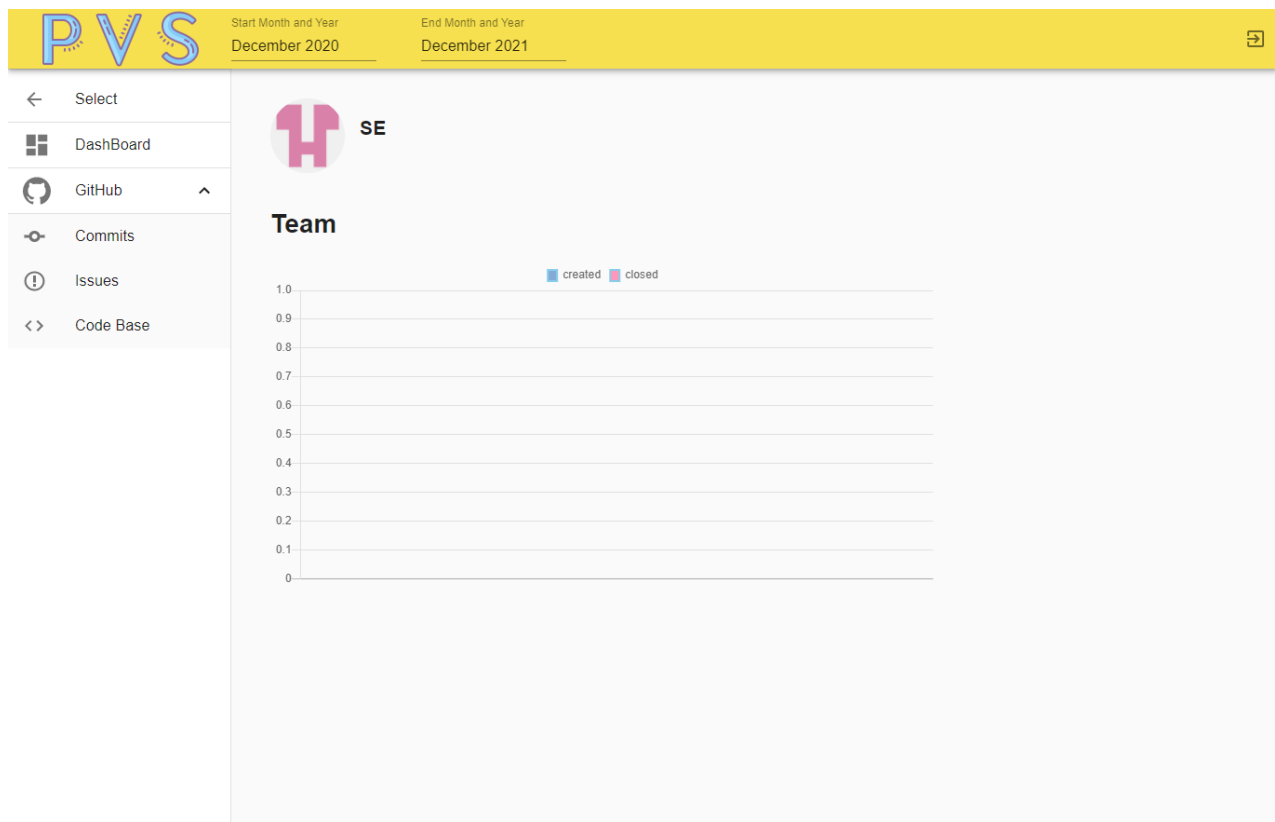
  



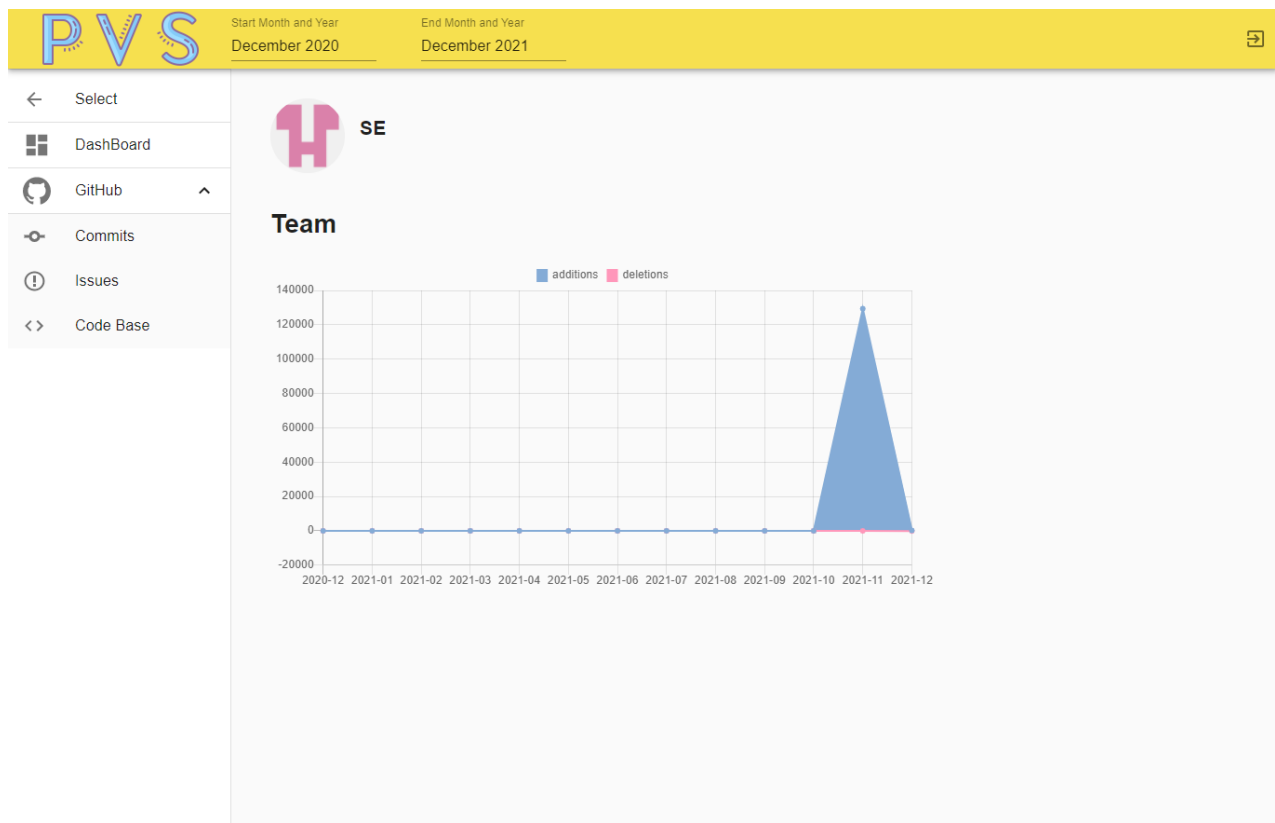
## 專案內 Repository Commit 視覺化頁面



## 專案內 Repository Issue 視覺化頁面



## 專案內 Repository Code Base 視覺化頁面



## References

- [1] 詹昆宸、張景博、張雅雯、吳昱成、杜奕萱，”Project Visualization System Software Design Document”