Project Dashboard

Software Design Document (SDD)

Version:1.0

Team#02

姓名	學號	E-mail
余珮綺	112598028	t112598028@ntut.org.tw
周雨柔	112598004	t112598004@ntut.org.tw
張書瑜	112598021	t112598021@ntut.org.tw
陸紀霖	112C72010	t112C72010@ntut.org.tw
吳思正	111C71003	t111C71003@ntut.org.tw
李維	111C71015	t111C71015@ntut.org.tw

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

11/30/2023

Table of Contents

Section 1 Introduction

1.1 Scope of the System

專案管理者或開發者可以使用本系統檢視專案資訊，系統應可以查看及新增專案、查看 Repository 資訊及新增 Repository。

1.2 Purpose of the Document

本文件主要是介紹專案儀表板（Project Dashboard）的研究背景與動機以及 系統的設計、規格、流程，其中系統規格、架構、流程，皆有說明，藉由參閱本文件的開 發人員迅速了解此系統的設計規格與架構。

1.3 Overview of the Document

此文件分為五個部分：

- ➢ Introduction
- ➢ System Requirements
- ➢ Design Constraints and Solutions
- ➢ Subsystem Architecture
- ➢ Detailed of Subsystem and Interface Description

Section 2 System Requirements

2.1 Functional Requirements

專案儀表板（Project Dashboard）是為了協助專案開發，方便觀看專案狀況與成 員的管理。本系統共分為四個子系統：

1. 使用者帳號管理子系統 (User Account Management Subsystem, UAMS)
2. Repository 管理子系統 (Repository Management Subsystem, RMS)
3. 資訊顯示子系統描述 (RIVS, Repository Information Visualization Subsystem Description)
4. 工作管理子系統描述(TMS, Task Management Subsystem Description)

PMS 的 Functional Requirement 如下：

需求編號	優先順序	需求描述
PD-F-01	1	提供帳戶管理及權限辯識功能。
PD-F-02	1	提供 Project 及 Repository 管理功能。
PD-F-03	1	提供 Repository 視覺化資訊功能。
PD-F-04	1	提供Task的管理功能。
UAMS-F-01	1	提供使用者登入，進行身份辯識。
UAMS-F-02	1	使用者可以設定/修改自己的個人資訊。
UAMS-F-03	1	提供帳號管理功能，包括帳號的新增、修改、刪除和查詢。
RMS-F-01	1	提供 Administrator 管理 Repository。
RMS-F-02	1	提供 Administrator 管理 Project
RMS-F-03	1	使用者可以設定自己的 Repository。
RMS-F-04	1	使用者可以設定自己的 Project。
RMS-F-05	1	提供 Repository 管理功能，包括新增、刪除和查詢。
RIVS-F-01	1	使用者可以查看自己的 Repository 相關資料。
RIVS-F-02	1	使用者可以查看自己的 Project 相關資料。
TMS-F-01	1	提供Task的管理需求。
TMS-F-02	1	使用者可以查看自己的Project 相關資料。

2.1.1 User Stories and Acceptance Criteria

編號	**使用者故事**	**驗收準則**
US-01	身為開發者，我想提供帳戶管理功能，讓使用者可以登入系統。	手動測試:以 admin 及 user 身分都可以登入。 通過後端 unit test。
US-02	身為使用者，我想新增 Project，在 Project 中有 Repository 管理功能，讓我可以看到 Repository 的細部資訊。	手動測試:user 可以新增 projct、管理 Repository。 通過後端 unit test。
US-03	身為使用者，我想看到 Repository 的視覺化資訊。	手動測試:user 可以看到 Repository 的視覺化資訊

US-04	身為使用者，我想看到Trello的工作管理看板，並進行操作。	手動測試:user 可以看到Trello的工作管理看板，並進行操作。

2.2 Non-Functional Requirements

需求編號	優先順序	需求描述
PD-N-21	1	繁忙時間的 99%回應時間應少於 2 秒。繁忙時間的平均回 應時間最多為 1 秒。
PD-N-22	1	系統應維持每秒 1 個事件的吞吐量。
PD-N-23	1	系統應支援至少同時讓 100 位用戶每秒生成 0.1 個事件。
PD-N-24	1	報表產生速度小於 10 秒。
PD-V2-N-03	1	資料庫中之資料不會被非法存取、竄改。
PD-V2-N-04	1	在 Windows 系統下之不同 Browser 皆可正常運作。
PD-N-25	1	通過 Use Case 所寫的 Test Case。
PD-V2-N-05	1	通過 User Story 的手動測試。
PD-N-26	1	需提供可恢復資料的方法和備份資料的方法。
PD-N-27	1	處理器 64 位元雙核心時脈 超過 2.0GHz 以上。
PD-N-28	1	儲存空間至少需要 200GB 之硬碟空間。
PD-N-29	1	2 GB 以上記憶體。
PD-N-30	1	安裝 Visual Studio 2019
PD-N-31	1	安裝瀏覽器。
PD-N-32	1	通過 Unit Test 以及手動測試。
PD-N-33	1	四份文件：PEP、SRS、SDD、STD。
PD-N-34	1	由程式開發者提供程式的維護服務。
PD-N-35	1	使用子系統獨立架構，易於維護

Section 3 Design Constraints and Solutions

3.1 Technical Solution Criteria

有關 Solution Criteria 這方面，針對 PMS 擬定了將來會遇到的各種限制，包括：

- 易學性：考慮選擇的應用軟體是否容易上手。
- 可攜性：考慮能否在各個 OS 上執行。
- 安全性：考慮應用軟體設計上的安全保密性。
- 擴充性：考慮後續的擴充是否容易。
- 廠商支持程度：考慮是否有廣泛的使用者。
- 維護性：考慮後續的維護是否容易。

上述的限制問題，基本上與系統所用的軟體是否有關。

3.2 Alternative Solutions

在本系統設計前，要考慮各種可能影響系統架構的因素，包括程式設計架構、資料庫軟體 系統與程式語言等選擇。

(一) 本專案的系統架構，開發者提供了可行的系統架構，分別為單機系統、主從系統與網 頁系統架構，以下列出與上述限制比較表。

	單機系統	主從系統	網頁系統
易學性	高，開發時間較短	低，開發需要較長時間	低，開發需要較長時間
可攜性	低，必需因應每個平台而修改	中，必須額外開放使用	高，使用瀏覽器連上即可
安全性	高，單機資料保護較容易	中，有被盜取資料的風險	中，有被盜取資料的風險
擴充性	彈性小，不能完全符合使用者需求	彈性大，可完全符合使用者需求	彈性大，可完全符合使用者需求
廠商支持程度	低，客戶(老師)不允許	低，客戶(老師)不允許	低，客戶(老師)不允許
維護性	高，由開發者自行維護，安裝簡單。	高，由開發者自行維護。安裝較為複雜	高，由開發者自行維護。 安裝容易

(二) 可用來開發本系統的程語言有：C++、C#、Java

	C++	C#	Java
易學性	低，需自行管理記憶體釋放問題，較難使用	中，團隊中幾乎無人熟悉該語言	中，團隊中熟悉該語言的人不多
可攜性	高，最早成熟的語言，有豐富的資源	高，可支援所有平台開發	高，可支援所有平台開發，且有豐富資源
安全性	中，物件導向語言可	中，可實現資料封裝	中，可實現資料封裝

	實現資料封裝		
擴充性	小，不完全符合開發者需求	大，符合開發者需求	中，不完全符合開發者需求
廠商支持程度	低，開發團隊不支援	高，.NET Core 支援不同平台開發	高，JAVASCRIPT 和 JSP 作為腳本，支援度高
維護性	低，需考量記憶體釋放問題	高，物件導向較易於維護	高，物件導向較易於維護

(三) 系統 User Interface 有三種選擇，分別有 Angular、React、Vue

	Angular	React	Vue
易學性	低，較適合大型網站架構且社群較小	中，較適合大型網站架構但社群資源豐富	高，較適合小型網站架構且社群資源豐富
可攜性	無可攜性需求	無可攜性需求	無可攜性需求
安全性	無安全性需求	無安全性需求	無安全性需求
擴充性	低，套件數量較少	高，易新增功能及頁面，套件數量多	高，易新增功能及頁面，套件數量多
廠商支持程度	無廠商支持程度需求	無廠商支持程度需求	無廠商支持程度需求
維護性	中，頁面元件化，資料流單向綁定	高，有瀏覽器專用開發工具，頁面元件化，資料流雙向綁定	高，有瀏覽器專用開發工具，頁面元件化，資料流雙向綁定

(四) 資料庫有三種選擇，分別是 SQLite、SQL Server、MySQL

	SQLite	SQLServer	MySQL
易學性	高，原來的使用者已有基礎	高，原來的使用者已有基礎	中，原來的使用者已有基礎
可攜性	極高，不用安裝。	中，MSSQL 與.NET 整合，必須使用.NETCore 才能使用到完整的功能	高，為企業用免費的系統，容易安裝
安全性	低，個人使用資料容易外洩	高，MS 會定期提供安全性更新	中，MySQL 安全性目前無重大漏失
擴充性	低，因為為個人使用的資料庫系統	高，資料量大，其 SQLServer 在重負載下表現突出	中，儲存量適於中小型的資料量儲存
廠商支持程度	低，不支援企業應用	高，但須付費	高，免費
維護性	高，不須自行維護	高，不須自行維護	低，須自行維護

3.3 Selected Solution

(一) 系統架構選擇

	單機系統	主從系統	網頁系統
易學性	4	2	2
可攜性	2	3	4
安全性	4	2	3
擴充性	2	4	4
廠商支持程度	0	0	5
維護性	4	4	2
加總	16	15	20

Priorities {Scale: 1= Most bad(differcult), 5= Most good(easy), 0= Not necessary }

(二) 開發語言選擇

	C++	C#	JAVA
易學性	2	3	4
可攜性	4	4	4
安全性	4	4	4
擴充性	2	5	4
廠商支持程度	2	5	5
維護性	2	5	5
加總	16	28	26

Priorities {Scale: 1= Most bad(differcult), 5= Most good(easy), 0= Not necessary }

(三) User Interface 框架選擇

	Angular	React	Vue
易學性	1	3	5
可攜性	0	0	0
安全性	0	0	0
擴充性	1	5	5
廠商支持程度	0	0	0
維護性	3	5	5
加總	5	13	15

Priorities {Scale: 1= Most bad(differcult), 5= Most good(easy), 130= Not necessary }

(四) 資料庫選擇

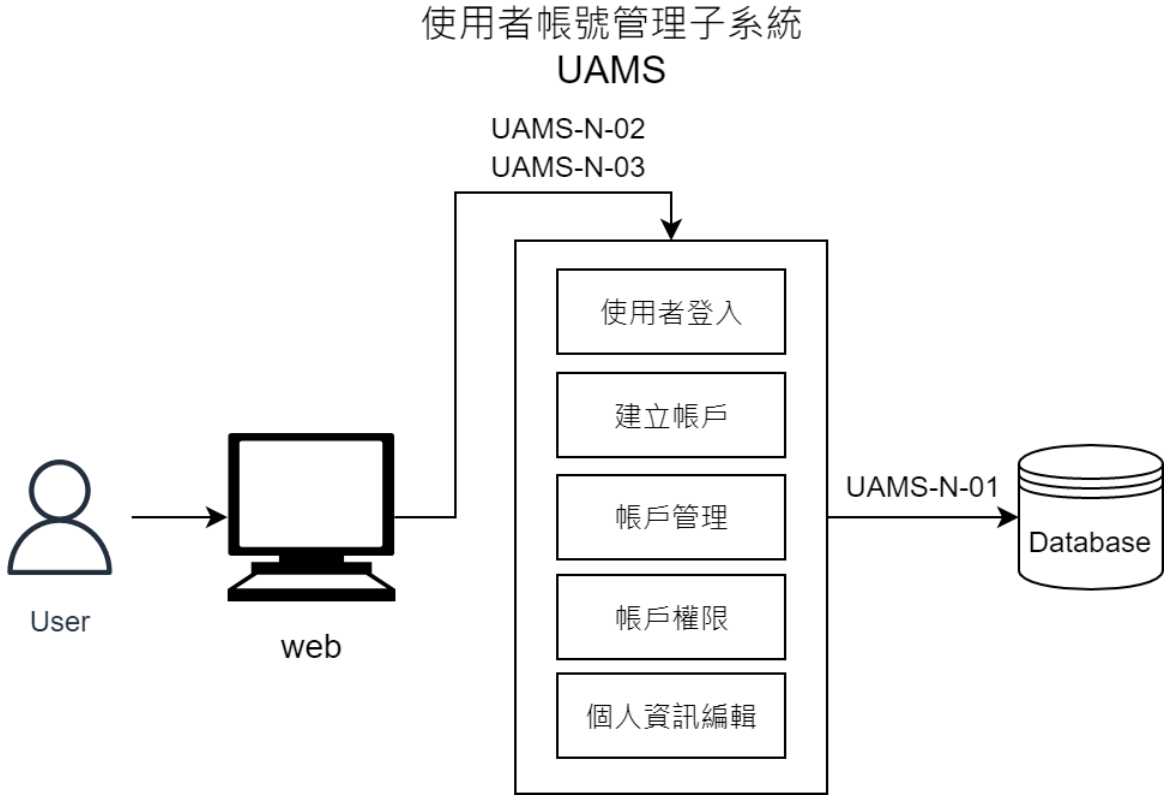
	SQLite	SQLServer	MySQL
易學性	5	3	3
可攜性	5	1	1
安全性	1	3	3
擴充性	5	3	3

廠商支持程度	1	3	3
維護性	5	1	1
加總	22	14	14

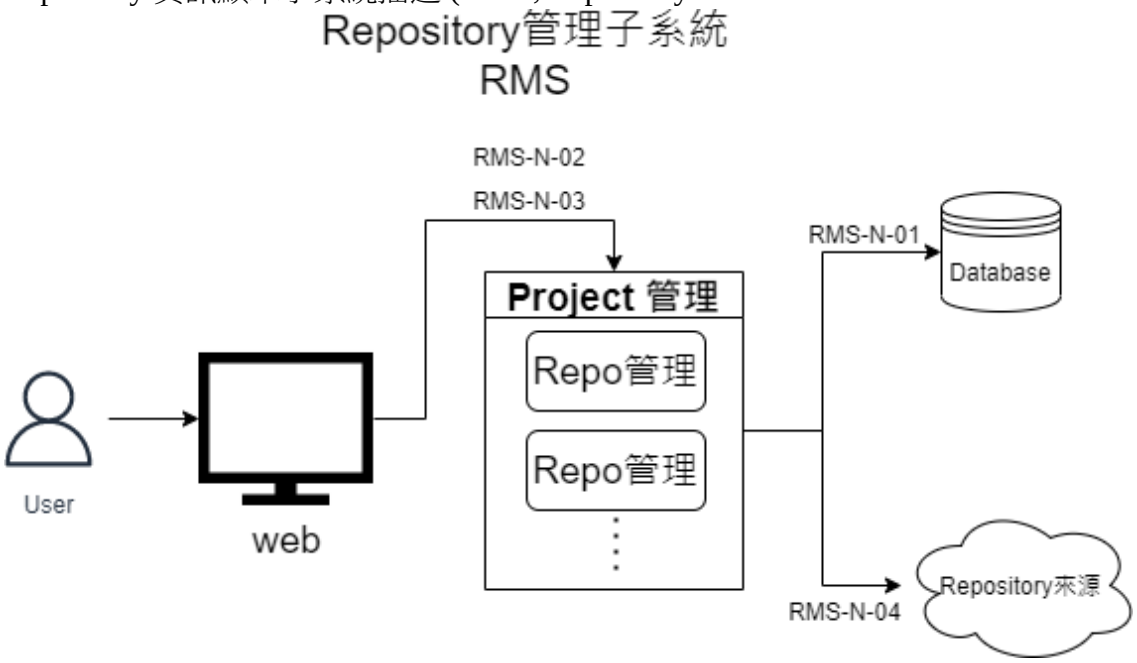
Priorities {Scale: 1= Most bad(differcult), 5= Most good(easy), 130= Not necessary }

Section 4 System Architecture

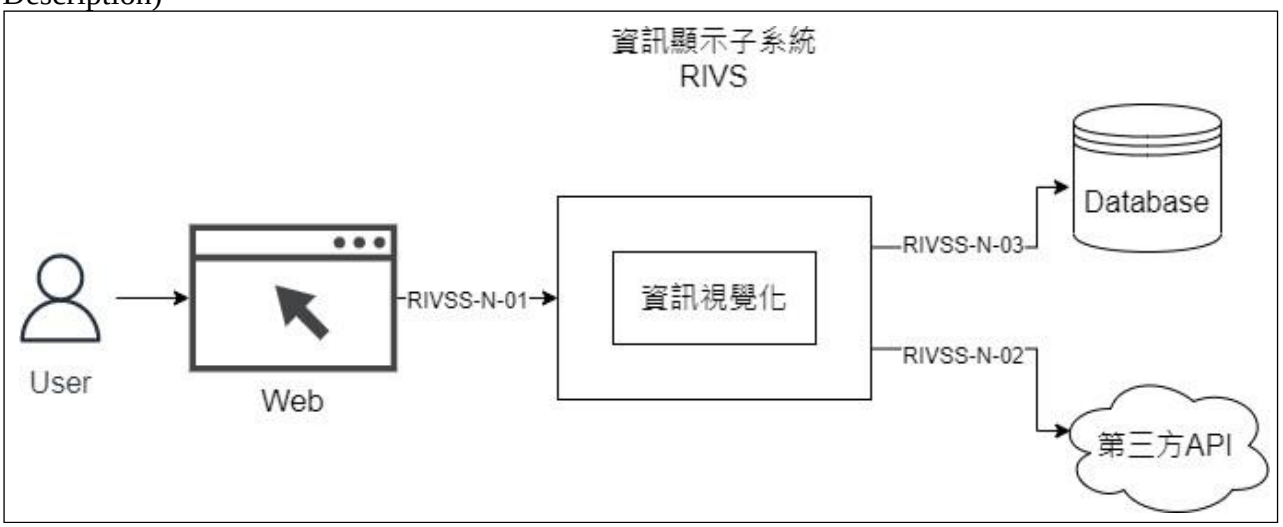
使用者帳號子系統描述 (User Account Management Subsystem Description)



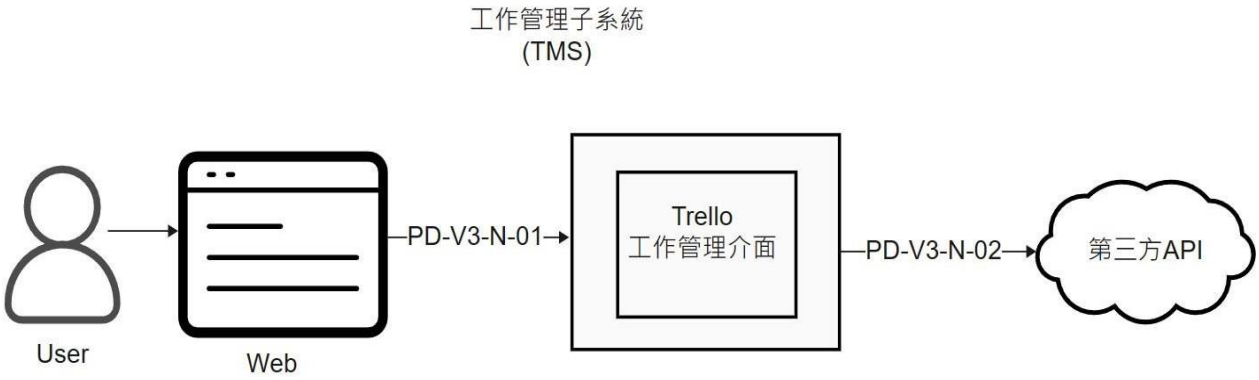
Repository 資訊顯示子系統描述 (RIVS, Repository Information Visualization



Repository 資訊顯示子系統描述 (RIVS, Repository Information Visualization Subsystem Description)



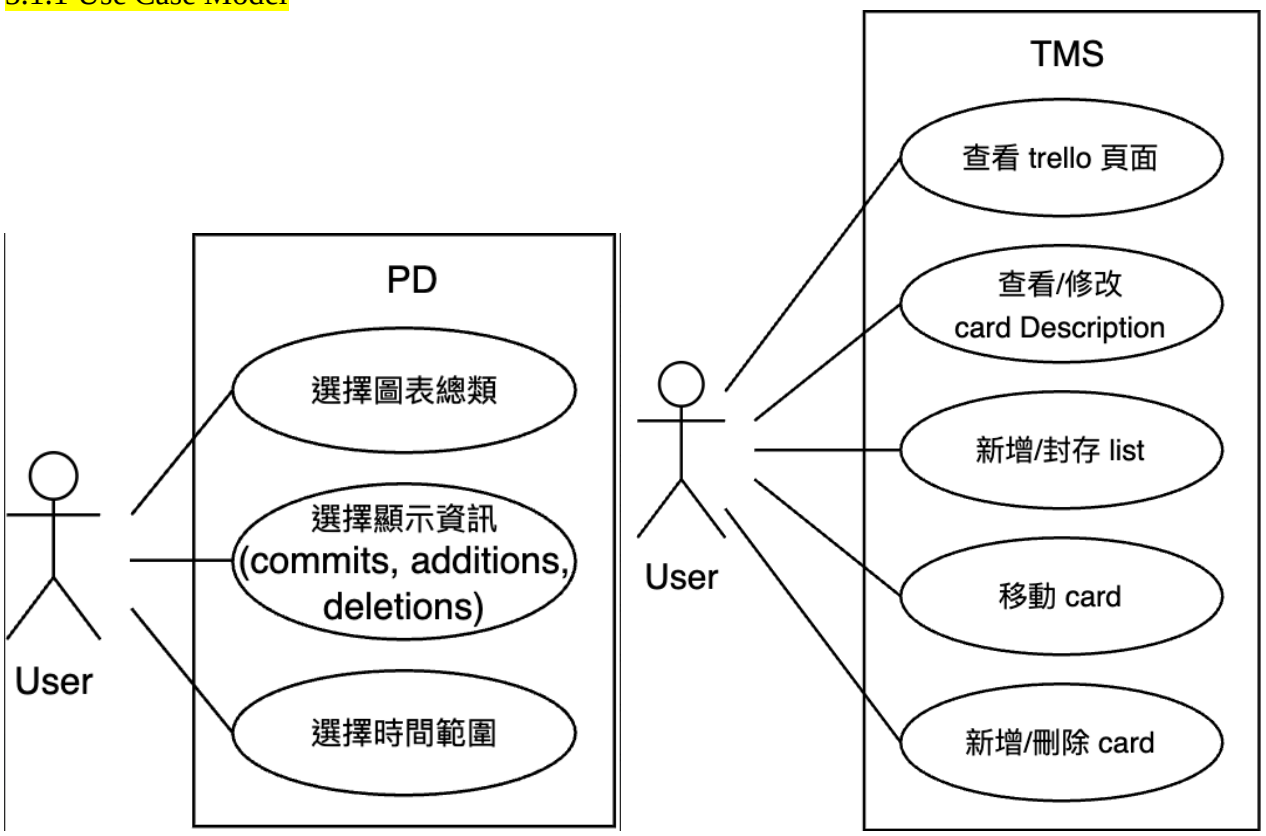
工作管理子系統描述(TMS, Task Management Subsystem Description)



Section 5 Detailed System Design

5.1 System Design Models

5.1.1 Use Case Model



Use Case UC 01	選擇圖表種類
Scope	Project Dashboard
Level	User-goal
Primary Actor	Project Manager
Stakeholders and Interests	Project Manager：想要選擇查看 project 中某個 repository 資訊的方式
Preconditions	Project Manager 需先登入系統
Success Guarantee	讓 Project Manager 可正常地選擇查看 repository 資訊的方式
Main Success Scenario	1. 系統顯示使用者所擁有的 project 2. 使用者選擇某個他所擁有的 project 3. 系統顯示被選專案中所有的 repository 4. 使用者選擇某個 repository 5. 使用者選擇圖表種類 6. 系統以使用者選擇的圖表顯示 repository 的相關資訊

Extensions	*a. 出現錯誤: 1. 顯示相對應的回應提示使用者
	6a. 使用者想查看其他種類的圖表: 1. 回到第 5 步 2. 重新選擇圖表種類
Special Requirements	None
Technology and Data Variations List	None
Frequency of Occurrence	often
Open Issues	None

Use Case UC 02	選擇顯示資訊(commits、additions、deletions)
Scope	Project Dashboard
Level	User-goal
Primary Actor	Project Manager
Stakeholders and Interests	Project Manager：想要選擇查看 project 中某個 repository 資訊的種類
Preconditions	Project Manager 需先登入系統
Success Guarantee	讓 Project Manager 可正常地選擇查看 repository 資訊的種類
Main Success Scenario	1. 系統顯示使用者所擁有的 project 2. 使用者選擇某個他所擁有的 project 3. 系統顯示被選專案中所有的 repository 4. 使用者選擇某個 repository 5. 使用者選擇資訊種類 6. 系統以使用者選擇的資訊顯示 repository 的圖表
Extensions	*a. 出現錯誤: 1. 顯示相對應的回應提示使用者 6a. 使用者想查看其他種類的資訊: 1. 回到第 5 步 2. 重新選擇資訊種類
Special Requirements	None
Technology and Data Variations List	None
Frequency of Occurrence	often
Open Issues	None

Use Case UC 03	選擇時間範圍
Scope	Project Dashboard
Level	User-goal

Primary Actor	Project Manager
Stakeholders and Interests	Project Manager：想要選擇查看 project 中某個 repository 特定時段的資訊
Preconditions	Project Manager 需先登入系統
Success Guarantee	讓 Project Manager 可正常地選擇查看 repository 特定時段的資訊
Main Success Scenario	1. 系統顯示使用者所擁有的 project 2. 使用者選擇某個他所擁有的 project 3. 系統顯示被選專案中所有的 repository 4. 使用者選擇某個 repository 5. 使用者選擇時段 6. 系統以使用者選擇的時段顯示 repository 的圖表
Extensions	*a. 出現錯誤: 1. 顯示相對應的回應提示使用者 6a. 使用者想查看其他時段的圖表: 1. 回到第 5 步 2. 重新選擇時段
Special Requirements	None
Technology and Data Variations List	None
Frequency of Occurrence	often
Open Issues	None

Use Case UC 04	查看 Trello Board 頁面
Scope	Task Management System
Level	User-goal
Primary Actor	Project Manager
Stakeholders and Interests	Project Manager：查看 Trello Board 頁面
Preconditions	Project Manager 需先登入系統
Success Guarantee	讓 Project Manager 可正常地查看 Trello Board頁面
Main Success Scenario	1. 系統顯示使用者所建立的project

	2. 使用者選擇某個他所擁有的 project 3. 系統顯示被選專案中所有的list 4. 系統顯示所有list中的每個card
Extensions	*a. 出現錯誤: 1. 顯示相對應的回應提示使用者
Special Requirements	None
Technology and Data Variations List	None
Frequency of Occurrence	often
Open Issues	None

Use Case UC 05	查看 Trello card 頁面
Scope	Task Management System
Level	User-goal
Primary Actor	Project Manager
Stakeholders and Interests	Project Manager：查看 Trello card 頁面
Preconditions	Project Manager 需先登入系統
Success Guarantee	讓 Project Manager 可正常地查看Trello card 頁面
Main Success Scenario	1. 系統顯示使用者所擁有的 project 2. 使用者選擇某個他所擁有的 project 3. 系統顯示被選專案中所有的list 4. 系統顯示所有list中的每個card 5. 選擇欲顯示的card 6. 顯示專案在所選 card 中所有的 task information
Extensions	*a. 出現錯誤: 1. 顯示相對應的回應提示使用者
Special Requirements	None
Technology and Data Variations List	None
Frequency of Occurrence	often
Open Issues	None

Use Case UC 06	新增 Trello card
Scope	Task Management System
Level	User-goal
Primary Actor	Project Manager
Stakeholders and Interests	Project Manager：新增 Trello card
Preconditions	Project Manager 需先登入系統

Success Guarantee	讓 Project Manager 可正常地新增Trello card
Main Success Scenario	1. 系統顯示使用者所擁有的 project 2. 使用者選擇某個他所擁有的 project 3. 系統顯示被選專案中所有的list 4. 系統顯示所有list中的每個card 5. 在有需求的 List中新增一個card
Extensions	*a. 出現錯誤: 1. 顯示相對應的回應提示使用者
Special Requirements	None
Technology and Data Variations List	None
Frequency of Occurrence	often
Open Issues	None

Use Case UC 07	刪除 Trello card
Scope	Task Management System
Level	User-goal
Primary Actor	Project Manager
Stakeholders and Interests	Project Manager：刪除 Trello card
Preconditions	Project Manager 需先登入系統
Success Guarantee	讓 Project Manager 可正常地刪除Trello card
Main Success Scenario	1. 系統顯示使用者所擁有的 project 2. 使用者選擇某個他所擁有的 project 3. 系統顯示被選專案中所有的list 4. 系統顯示所有list中的每個card 5. 選擇欲刪除的card 6. 顯示專案在所選 card 中所有的 task information 7. 刪除card
Extensions	*a. 出現錯誤: 1. 顯示相對應的回應提示使用者
Special Requirements	None
Technology and Data Variations List	None
Frequency of Occurrence	often
Open Issues	None

Use Case UC 08	移動 Trello card 所在 List
Scope	Task Management System
Level	User-goal
Primary Actor	Project Manager

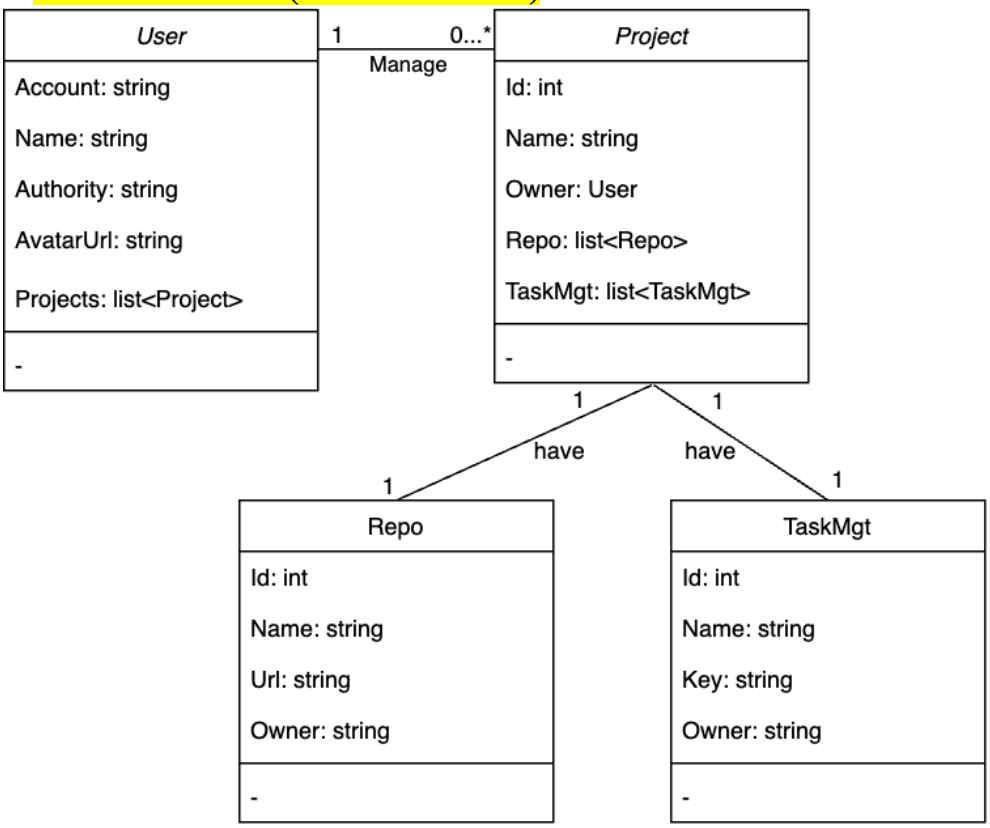
Stakeholders and Interests	Project Manager：移動 Trello card 所在 List
Preconditions	Project Manager 需先登入系統
Success Guarantee	讓 Project Manager 可正常地刪除Trello card
Main Success Scenario	1. 系統顯示使用者所擁有的 project 2. 使用者選擇某個他所擁有的 project 3. 系統顯示被選專案中所有的list 4. 系統顯示所有list中的每個card 5. 選擇欲移動的card 6. 顯示專案在所選 card 中所有的 task information 7. 選擇card所要移動到的目的List
Extensions	*a. 出現錯誤: 1. 顯示相對應的回應提示使用者
Special Requirements	None
Technology and Data Variations List	None
Frequency of Occurrence	often
Open Issues	None

Use Case UC 09	新增 Trello List
Scope	Task Management System
Level	User-goal
Primary Actor	Project Manager
Stakeholders and Interests	Project Manager：新增 Trello List
Preconditions	Project Manager 需先登入系統
Success Guarantee	讓 Project Manager 可正常地新增Trello List
Main Success Scenario	1. 系統顯示使用者所擁有的 project 2. 使用者選擇某個他所擁有的 project 3. 系統顯示被選專案中所有的list 4. 新增一個Trello List
Extensions	*a. 出現錯誤: 1. 顯示相對應的回應提示使用者
Special Requirements	None
Technology and Data Variations List	None
Frequency of Occurrence	often
Open Issues	None

Use Case UC 10	封存 Trello List
Scope	Task Management System

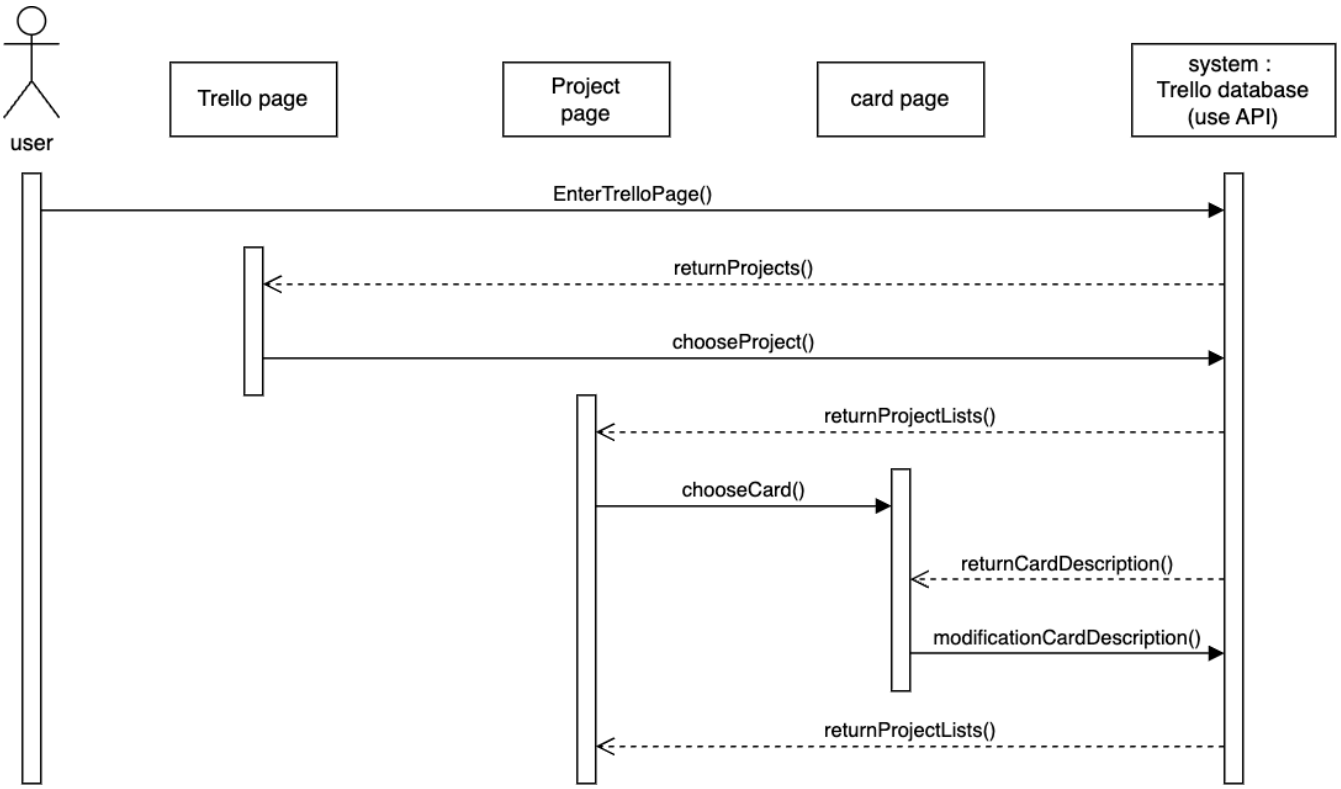
Level	User-goal
Primary Actor	Project Manager
Stakeholders and Interests	Project Manager：封存 List
Preconditions	Project Manager 需先登入系統
Success Guarantee	讓 Project Manager 可正常地封存Trello card
Main Success Scenario	8. 系統顯示使用者所擁有的 project 9. 使用者選擇某個他所擁有的 project 10.系統顯示被選專案中所有的list 11.系統顯示所有list中的每個card 12.封存目標List
Extensions	*a. 出現錯誤: 1. 顯示相對應的回應提示使用者
Special Requirements	None
Technology and Data Variations List	None
Frequency of Occurrence	often
Open Issues	None

5.1.2 Static Models (Structural Models)

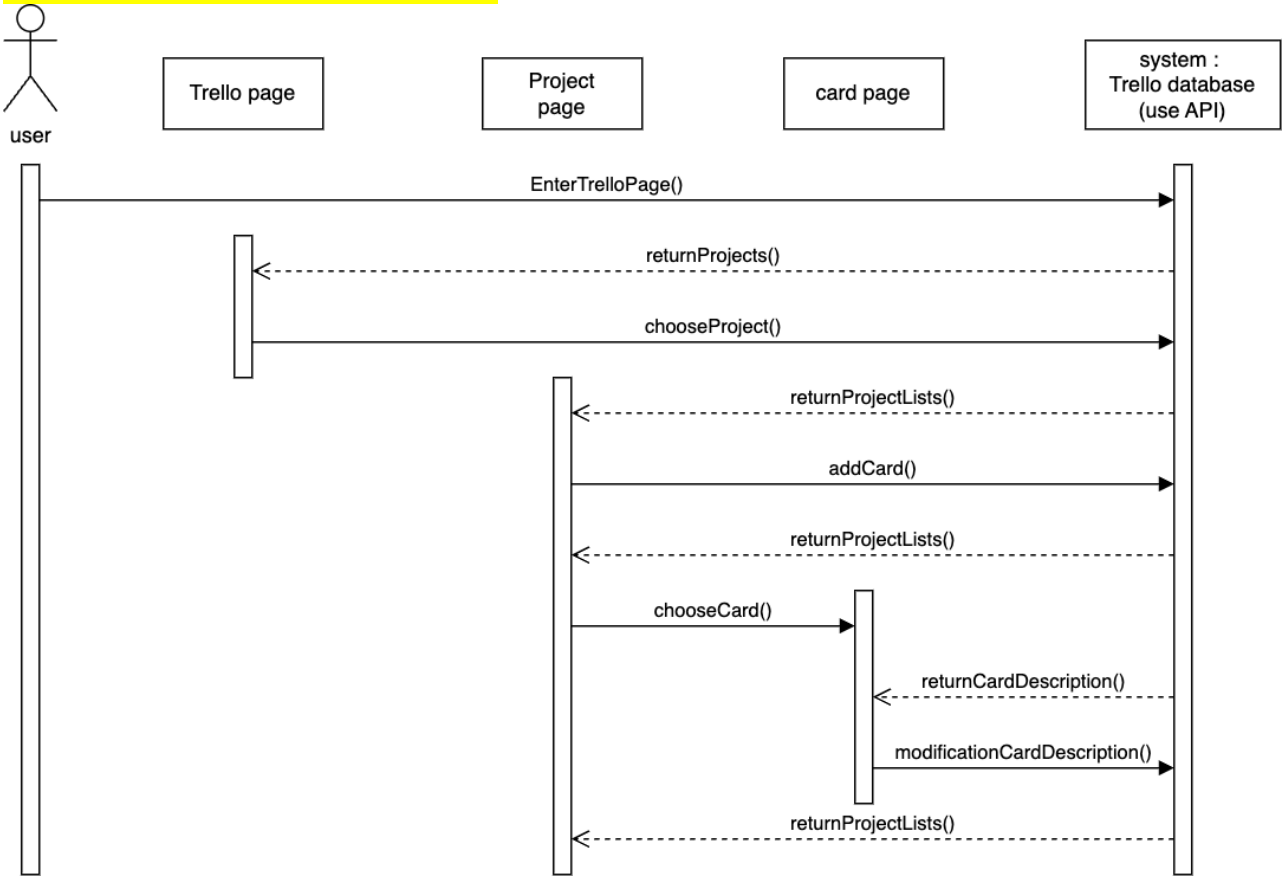


5.1.3 Dynamic Models (Behavioral Models)

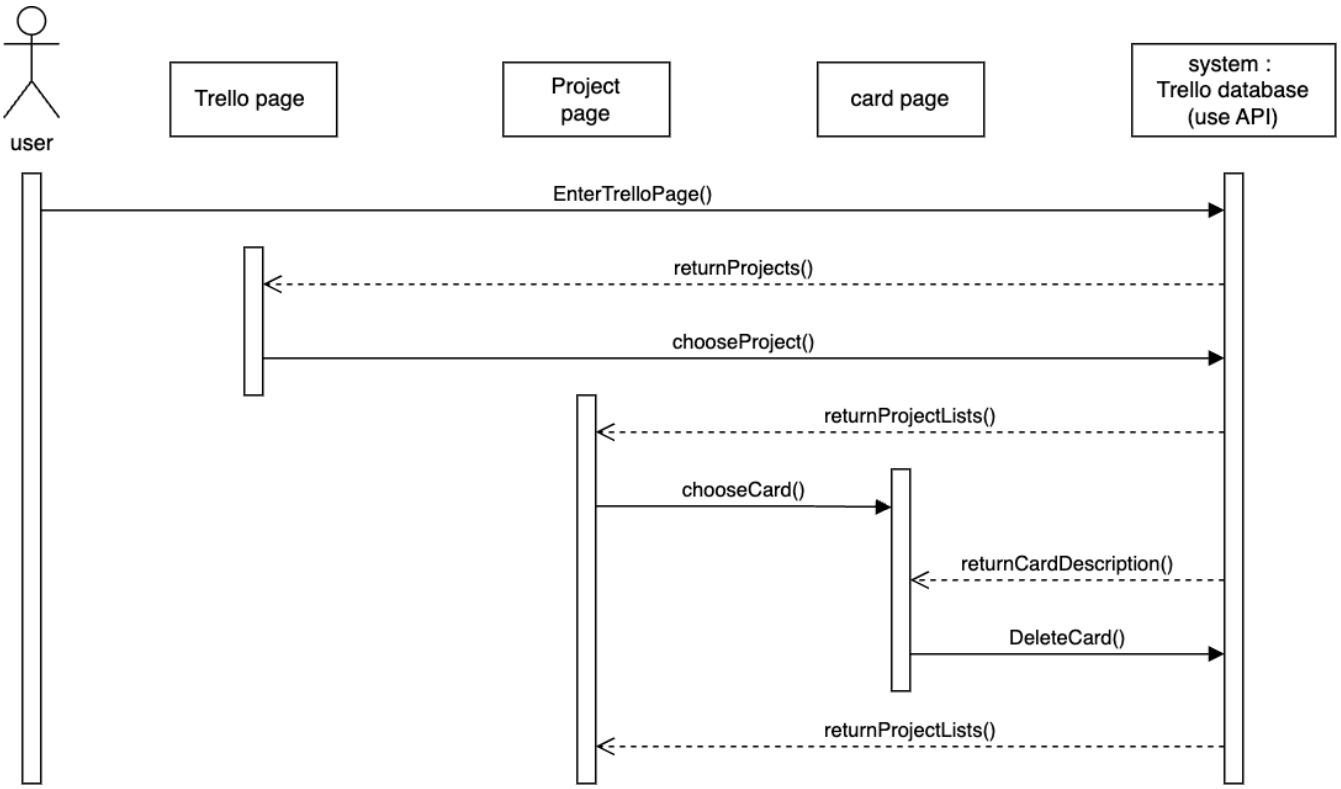
查看Trello Board頁面中指定card的詳細資訊(description)



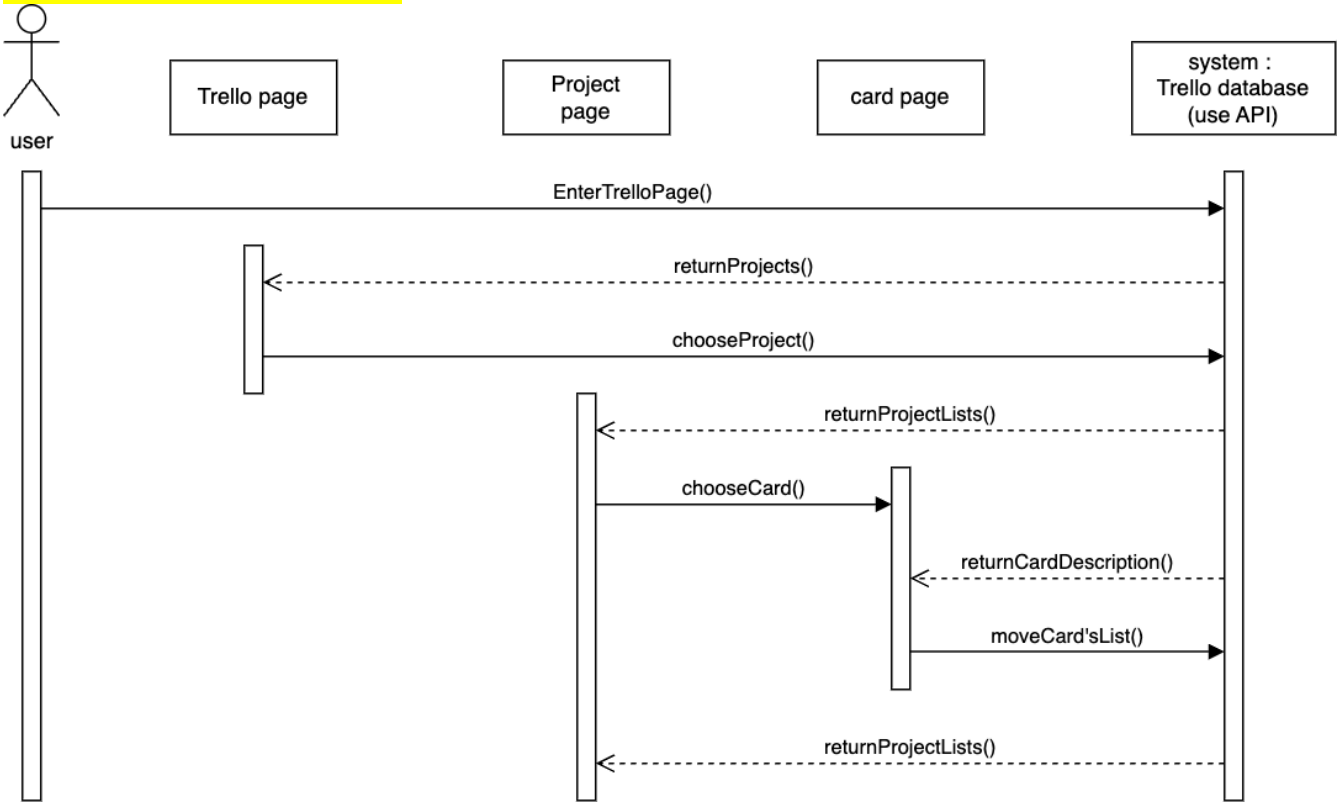
Trello頁面中在指定list中新增card



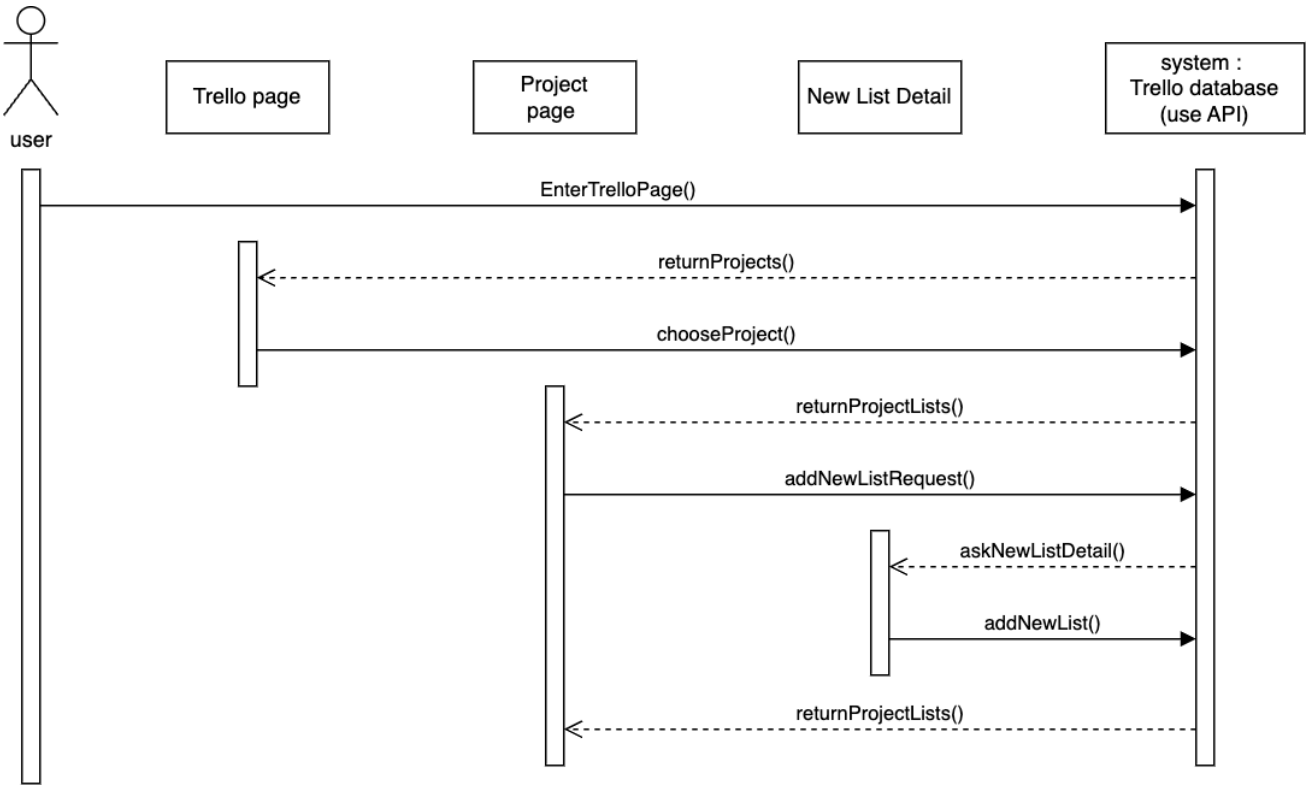
刪除Trello頁面中指定的card



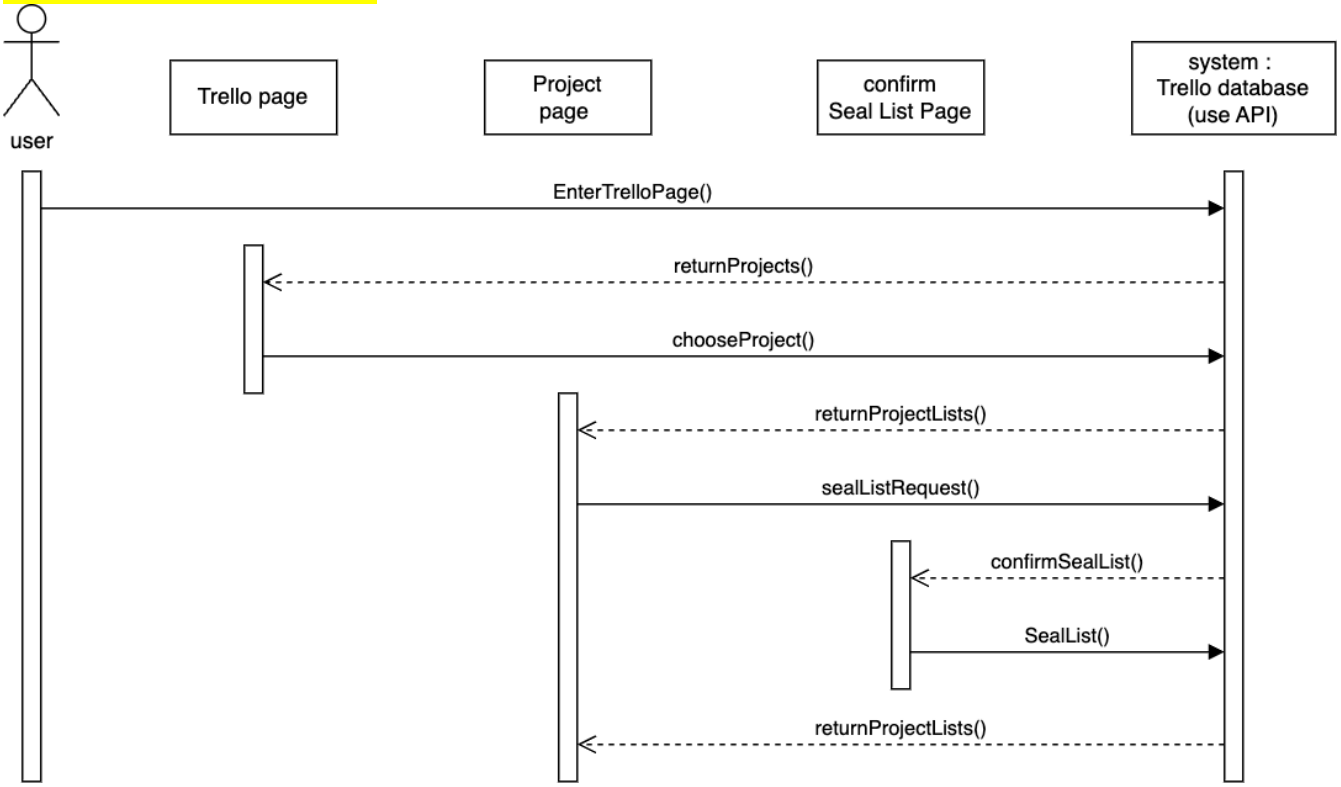
移動Trello頁面中指定的card



新增list於Trello頁面中



封存Trello頁面中指定的list



5.2 Interface Design

5.2.1 API Design

POST /Trello/createRepo – 創建 Trello repository

```
Request:
{
    DomainURL: "{ DomainURL }",
    APIToken: "{ APIToken }",
    Account: "{ Account }",
    BoardId: "{ BoardId }",
    ProjectId: "{ ProjectId }"
}
Response:
{
    Success: true,
    Message: "Added Success"
}
```

POST / Trello / boardInfo – 取得 Trello Board 資料

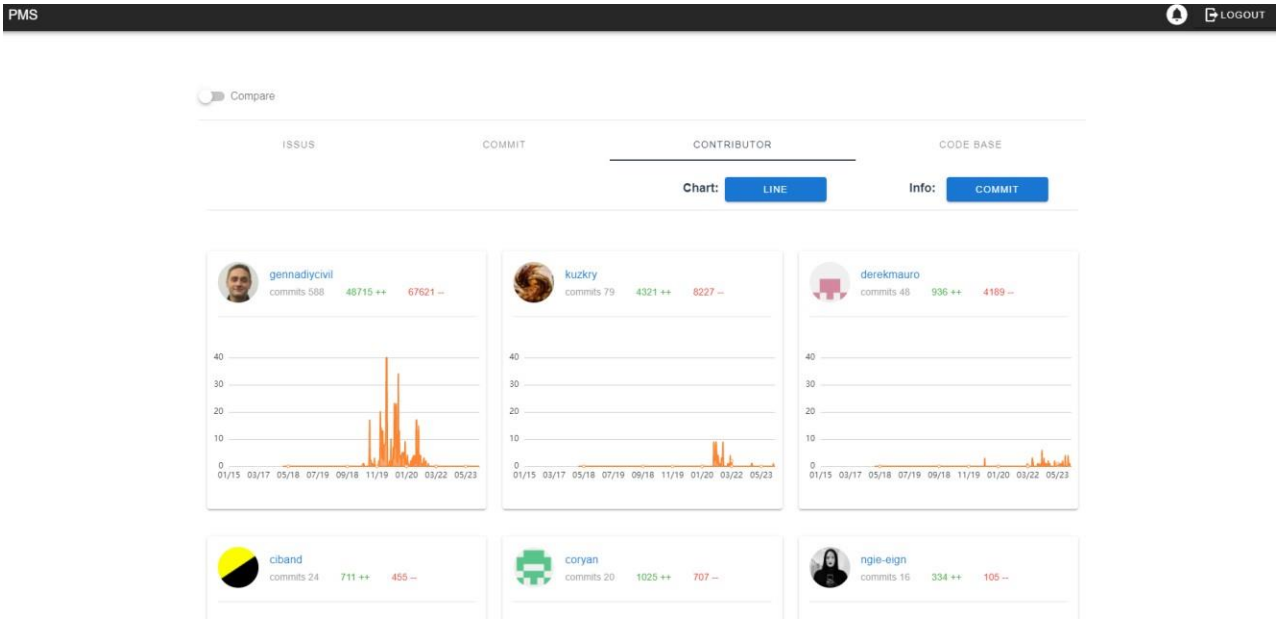
```
Request:
{
    DomainURL: "{ DomainURL }",
    APIToken: "{ APIToken }",
    Account: "{ Account }",
    BoardId: "{ BoardId }",
    ProjectId: "{ ProjectId }"
}
Response:
{
    Success: true,
    Data: [{
        Id: "{ Id }",
        Name: "{ Name }"
    }]
}
```

POST / Trello / issue – 取得 Trello issue 資料

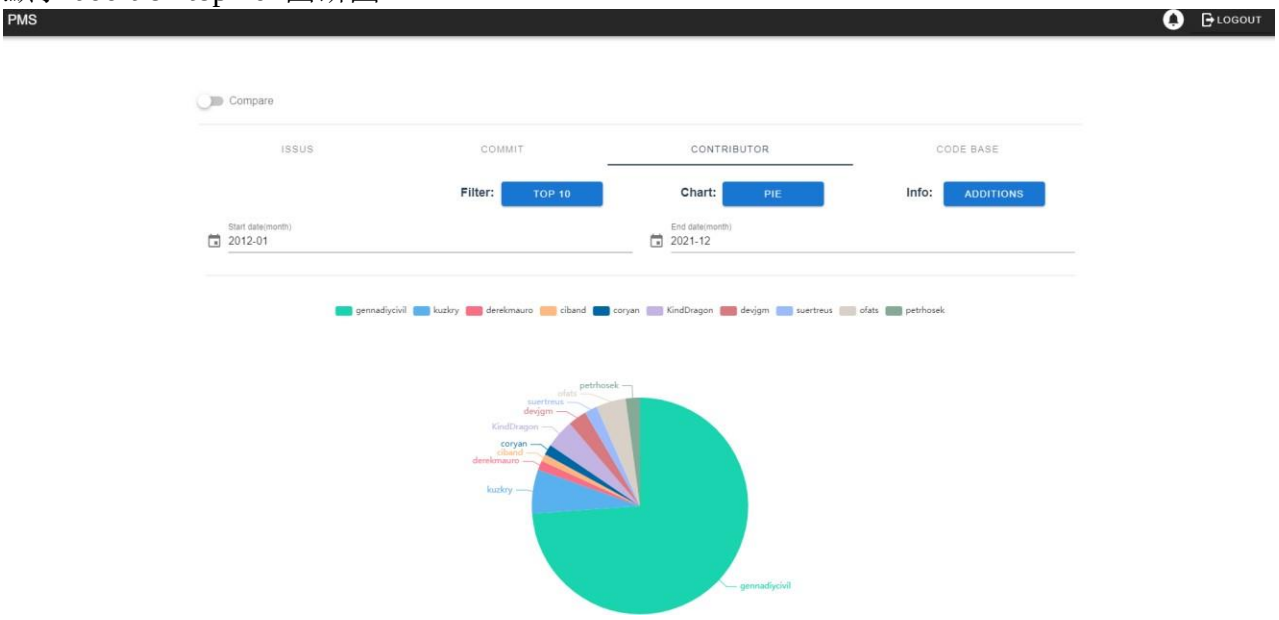
```
Request:
{
    RepoId: "{ RepoId }"
}
Response:
{
    [{
        Summary: "{ Summary}",
        Status: "{ Status }",
        Priority: "{ Priority }",
        Key: "{ Key }",
        Resolution: "{ Resolution }",
        Created: "{ Created }",
        Type: "{ Type }",
        Updated: "{ Updated }",
        Label: ["{ Label }"]
    }]
}
```

5.2.2 User Interface Design

顯示 commit 折線圖

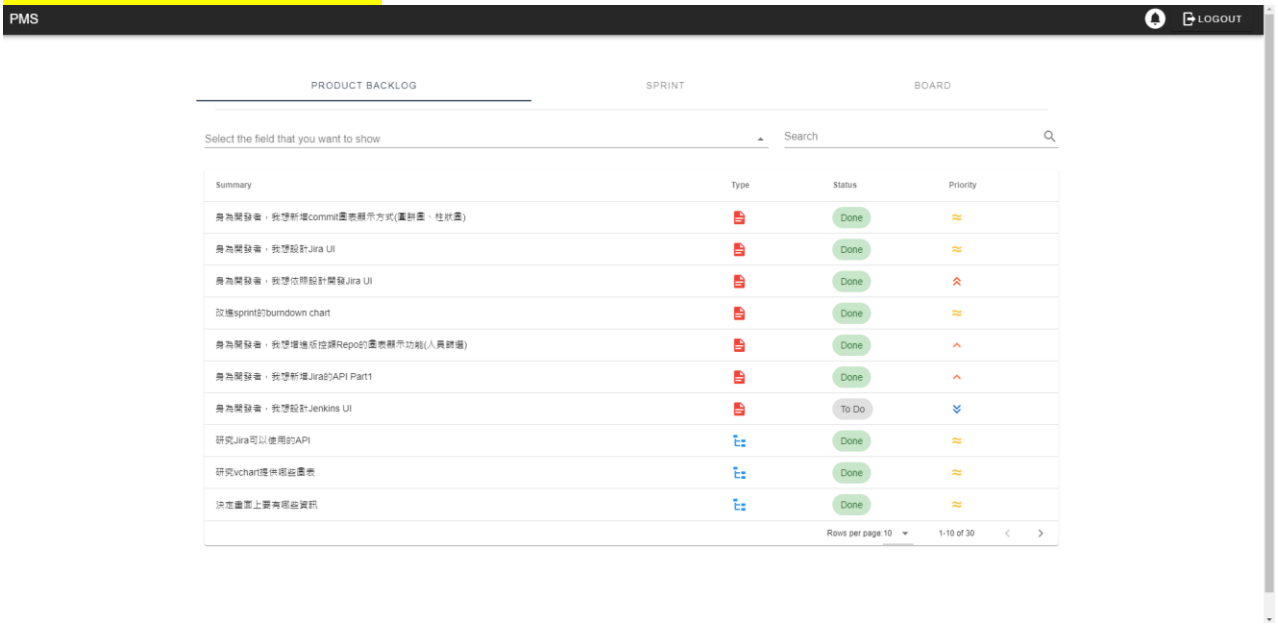


顯示 addition top 10 圓餅圖

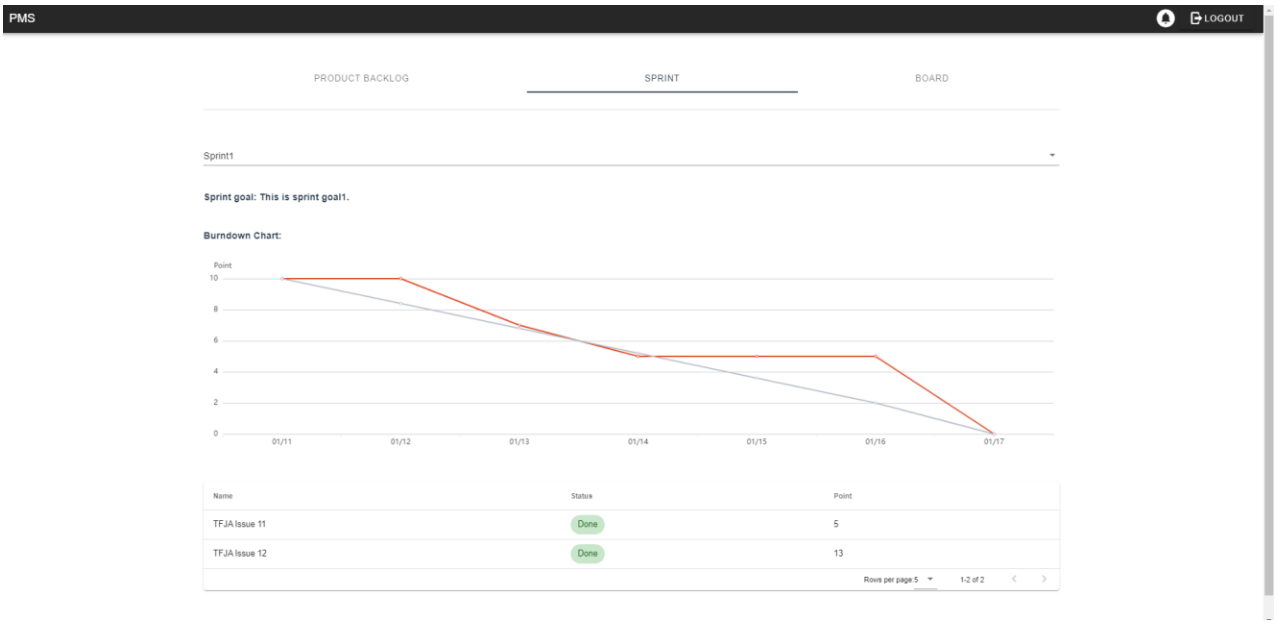


顯示 deletion top 10 長條圖

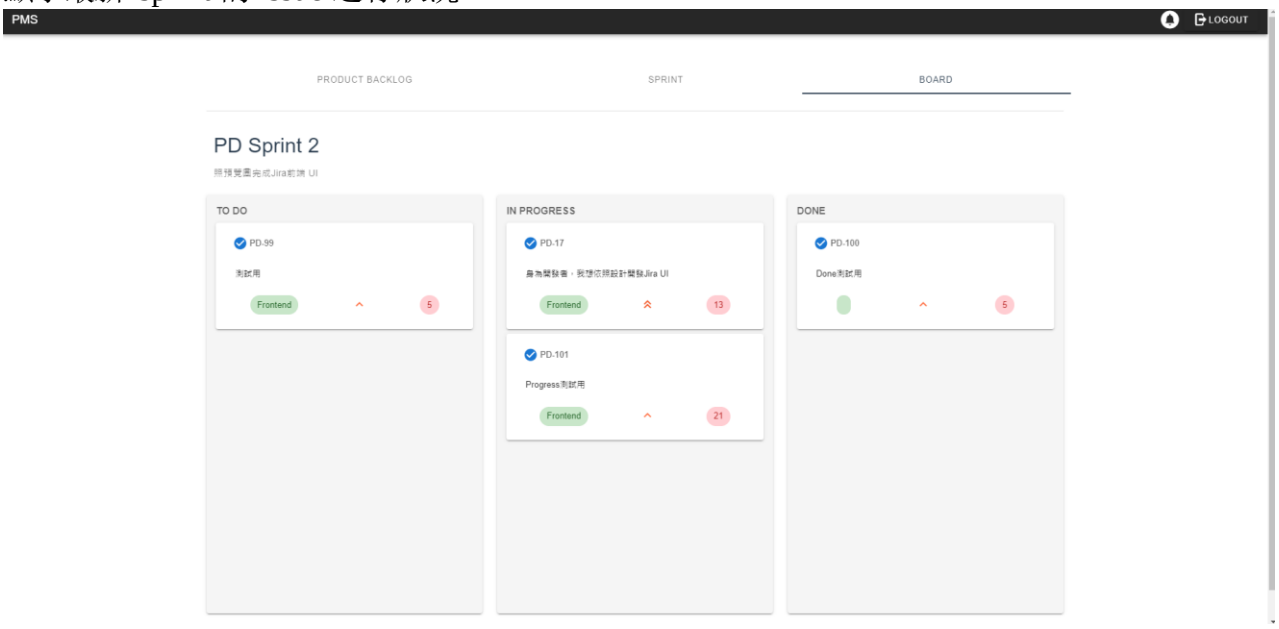
顯示 Trello Board 全部 issue



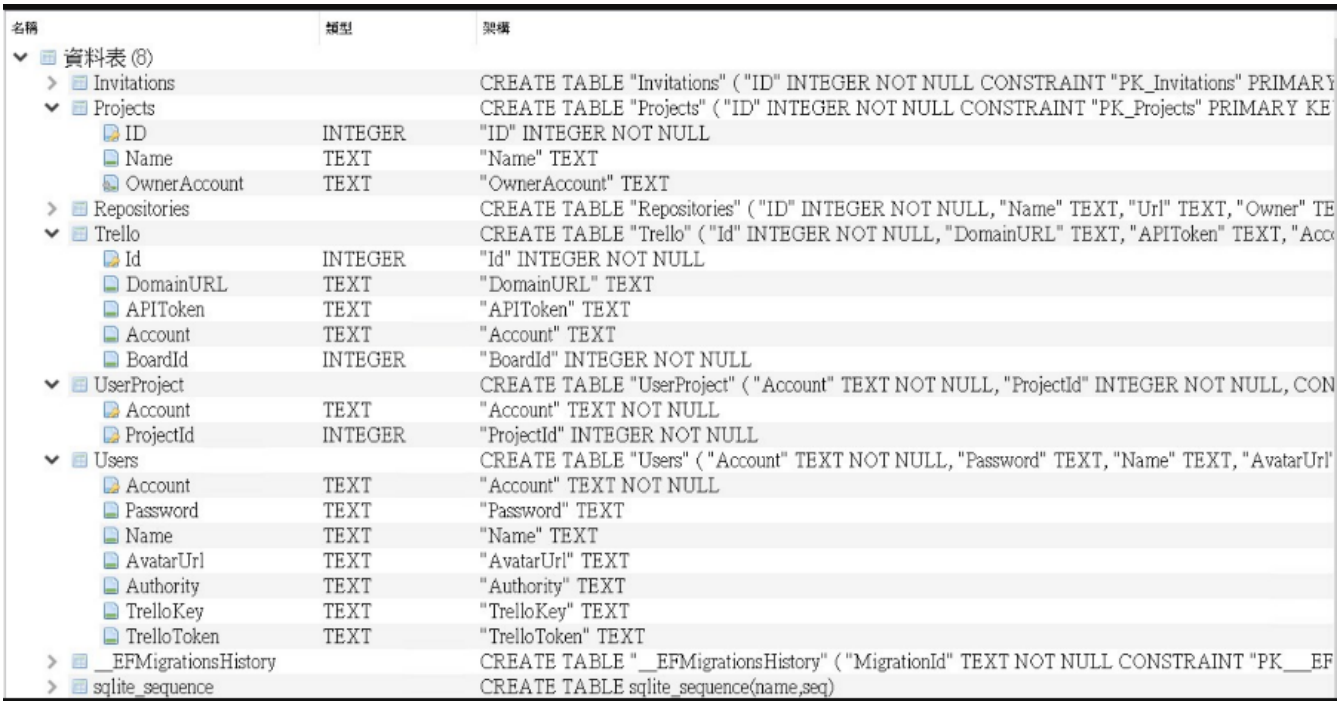
顯示特定 sprint 的 goal, chart and issue



顯示最新 sprint 的 issue 進行狀況



5.3 Database Design



名稱	類型	架構
資料表 (8)		
Invitations		CREATE TABLE "Invitations" ("ID" INTEGER NOT NULL CONSTRAINT "PK_Invitations" PRIMARY
Projects		CREATE TABLE "Projects" ("ID" INTEGER NOT NULL CONSTRAINT "PK_Projects" PRIMARY KE
ID	INTEGER	"ID" INTEGER NOT NULL
Name	TEXT	"Name" TEXT
OwnerAccount	TEXT	"OwnerAccount" TEXT
Repositories		CREATE TABLE "Repositories" ("ID" INTEGER NOT NULL, "Name" TEXT, "Url" TEXT, "Owner" TE
Trello		CREATE TABLE "Trello" ("Id" INTEGER NOT NULL, "DomainURL" TEXT, "APIToken" TEXT, "Acc
Id	INTEGER	"Id" INTEGER NOT NULL
DomainURL	TEXT	"DomainURL" TEXT
APIToken	TEXT	"APIToken" TEXT
Account	TEXT	"Account" TEXT
BoardId	INTEGER	"BoardId" INTEGER NOT NULL
UserProject		CREATE TABLE "UserProject" ("Account" TEXT NOT NULL, "ProjectId" INTEGER NOT NULL, CON
Account	TEXT	"Account" TEXT NOT NULL
ProjectId	INTEGER	"ProjectId" INTEGER NOT NULL
Users		CREATE TABLE "Users" ("Account" TEXT NOT NULL, "Password" TEXT, "Name" TEXT, "AvatarUrl"
Account	TEXT	"Account" TEXT NOT NULL
Password	TEXT	"Password" TEXT
Name	TEXT	"Name" TEXT
AvatarUrl	TEXT	"AvatarUrl" TEXT
Authority	TEXT	"Authority" TEXT
TrelloKey	TEXT	"TrelloKey" TEXT
TrelloToken	TEXT	"TrelloToken" TEXT
__EFMigrationsHistory		CREATE TABLE "__EFMigrationsHistory" ("MigrationId" TEXT NOT NULL CONSTRAINT "PK__EF
sqlite_sequence		CREATE TABLE sqlite_sequence(name,seq)

5.4 Traceability Matrix – Requirements vs Components

	UAMS	RMS	RIVS	TMS
UAMS-F-01	●			
UAMS-F-02	●			
UAMS-F-03	●			
RMS-F-01		●		
RMS-F-02		●		
RMS-F-03		●		
RMS-F-04		●		
RMS-F-05		●		
RIVS-F-01			●	
RIVS-F-02			●	
TMS-F-01				●
TMS-F-02				●

Glossary

References

[1] Erich Gamma et al., *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, 1995.

[2]

Appendices

A. Tracebility Matrix (Use Cases v.s. Classes or User Stories v.s. Classes)

B. Tracebility Matrix (Classes v.s. Classes)