**專案管理系統 HW5**

**Team member:**

105598003 劉彥麟

105598048 陳政皓

105598074 呂昭陞

目錄

[1 Requirement Document 3](#_Toc485575148)

[1.1 Change History 3](#_Toc485575149)

[1.2 Problem Statement 5](#_Toc485575150)

[1.3 System Context Diagram 6](#_Toc485575151)

[1.4 Summary of System Features 6](#_Toc485575152)

[1.5 Use Case Diagram 7](#_Toc485575153)

[1.6 Use Cases 8](#_Toc485575154)

[1.7 Non-functional Requirements and Constraints 17](#_Toc485575155)

[1.8 Glossary 17](#_Toc485575156)

[1.9 Software Environments 17](#_Toc485575157)

[2 Domain model 18](#_Toc485575158)

[2.1 Domain class diagram showing only concepts 18](#_Toc485575159)

[2.2 Add Associations 19](#_Toc485575160)

[2.3 Add Attributes 20](#_Toc485575161)

[3 Design 21](#_Toc485575162)

[3.1 Logical Architecture 21](#_Toc485575163)

[3.2 Use-Case Realizations with GRASP Patterns 22](#_Toc485575164)

[3.3 Design Class Model 36](#_Toc485575165)

[4 Implementation Class Model 38](#_Toc485575166)

[4.1 Implementation Class Diagram 38](#_Toc485575167)

[4.2 Different 39](#_Toc485575168)

[4.3 Calculate Line of Code 40](#_Toc485575169)

[5 Programming 41](#_Toc485575170)

[5.1 Snapshots of system execution 41](#_Toc485575171)

[5.2 Source Code Listing 50](#_Toc485575172)

[6 Unit Testing Code Listing 67](#_Toc485575173)

[6.1 snapshot of testing result 67](#_Toc485575174)

[6.2Unit Test Code listing 67](#_Toc485575175)

[Measurement 86](#_Toc485575176)

1 Requirement Document

1.1 Change History

|  |  |  |
| --- | --- | --- |
| Revision | Description | Date |
| Iteration I | | |
| 1 | Problem statement.  The Development language.  Measurement. | Feb 24, 2017 |
| 2 | System Context Diagram Summary of system features Use case diagram Use cases Non-functional Requirements and Constraints Glossary Measurement. | Mar 14,2017 Mar 15,2017 Mar 16,2017 Mar 17,2017 |
| 3 | Domain class diagram showing only concepts  Add Associations  Add Attributes | Mar 30,2017 |
| 4 | Add Associations  Add Attributes  Logical Architecture  Use-Case Realizations with GRASP Patterns  Design Class Model | April 22,2017 |
| Iteration II | | |
| 1 | Use Case Diagram  Use Cases  Domain model | May 25,2017 |
| 2 | Use-Case Realizations with GRASP Patterns  Design Class Model  Implementation Class Diagram  Calculate Line of Code  Programming  Unit Testing Code Listing | June 7,2017  June 15,2017  June 16,2017  June 17,2017  June 18,2017 |

1.2 Problem Statement

本軟體主要是為了讓專案管理者和開發人員可以追蹤專案進度與需求而設計。應用於各軟體開發公司或實驗室。

現今軟體的規模較複雜，軟體的需求時常改變，因此在管理與追蹤需求上會有難度，沒有一套系統可以用來追蹤與更新這些需求完成與否，導致需要花額外的心力和成本去關注專案的進度。

本軟體提供簡單的介面可以管理專案需求與測試，讓使用者可以清楚明白兩者間的關係，並且能有系統的管理專案

本軟體主要以圖形介面呈現，使用者一開始需要把專案的需求與測試項目新增至本軟體，軟體會依據使用者新增的內容，產生需求與測試的關係圖，讓使用者知道需求與測試的關係和完成狀態。

1.3 System Context Diagram

C:\Users\leo\Downloads\use case (2).png

* Manager:

Login: 登入

Management Project: 管理專案

Management Requirement: 管理需求

Management Test: 管理測試

View Project、Requirement、Test: 檢視專案、需求、測試

產生報表traceability matrix

* User:

Register、Login: 註冊、登入

View Project、Requirement、Test: 檢視專案、需求、測試

產生報表traceability matrix

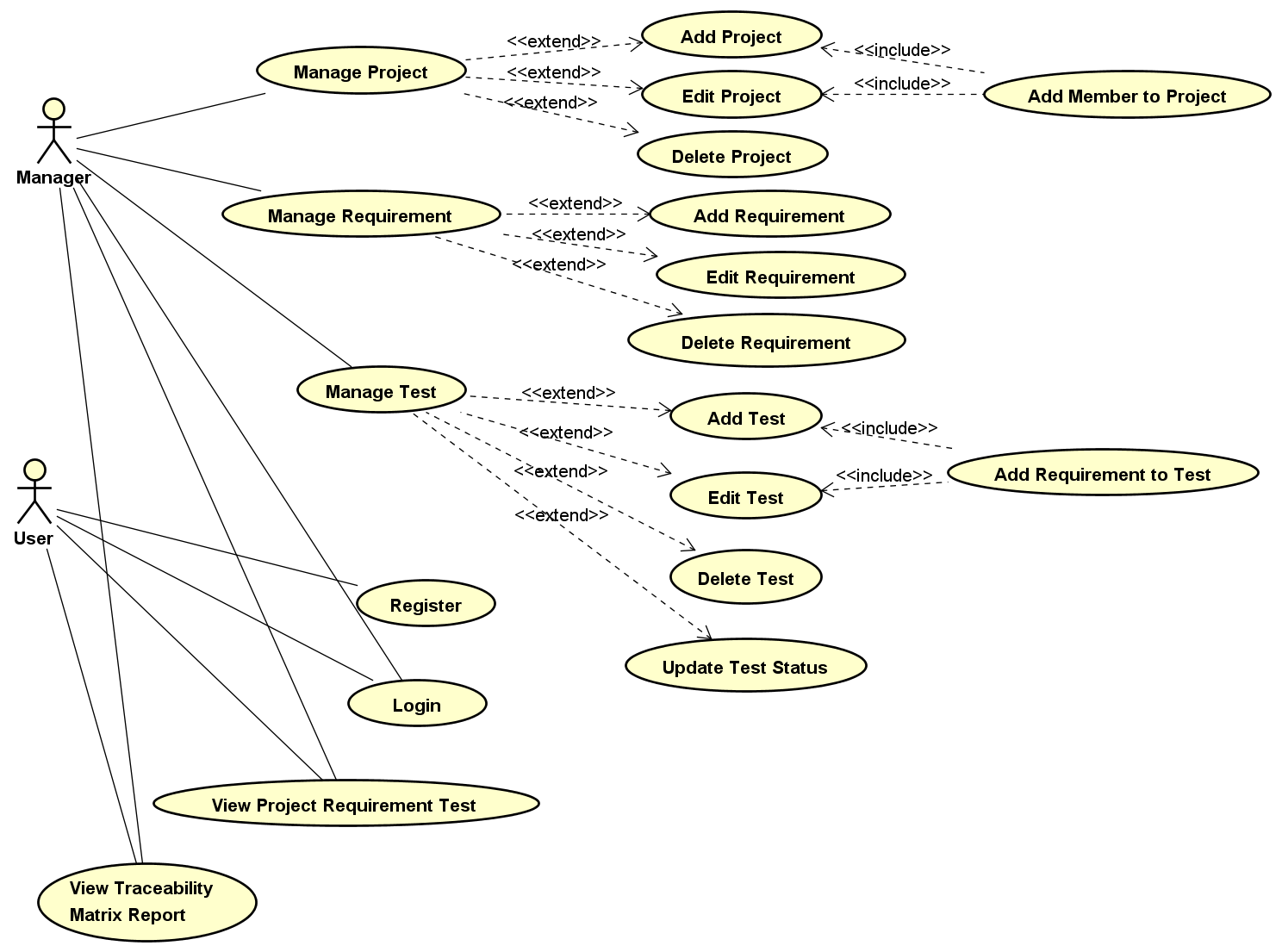
* DB:

CRUD: 新增、查詢、修改、刪除

1.4 Summary of System Features

|  |  |
| --- | --- |
| Feature ID | Description |
| FEA-01 | Register |
| FEA-02 | Login |
| FEA-03 | Management Project |
| FEA-04 | Management Requirement |
| FEA-05 | Management Test |
| FEA-06 | View Project Requirement Test |
| FEA-07 | Create traceability matrix |

1.5 Use Case Diagram



1.6 Use Cases

1.6.1Register

|  |  |
| --- | --- |
| Use Case Section | Comment |
| Use Case Name | Register Account |
| Scope | PMS Application |
| Level | user goal |
| Primary Actor | User |
| Stakeholders and Interests | User: 想要進入系統，必須先註冊帳號 |
| Preconditions | 你必須要是Team的member |
| Success Guarantee | 成功創建使用者帳號 |
| Main Success Scenario | 1. User點選註冊按鈕 2. User輸入使用者資訊 3. 系統將使用者資訊寫入資料庫 4. User完成註冊 |
| Extensions | 2a 系統偵測到使用者重覆的狀況   1. 要能跳出警告訊息要求使用者再輸入一次   3a 資料庫存取異常   1. 出現錯誤訊息視窗 |
| Special Requirements | UI友善 |
| Technology and Data Variations List | 2a 密碼格式不可以包含符號 |
| Frequency of Occurrence | 偶爾發生(第一次使用系統需要註冊時) |
| Miscellaneous | 非公司員使用這個系統，但註冊了，應該怎麼處理 |

1.6.2 Login

|  |  |
| --- | --- |
| Use Case Section | Comment |
| Use Case Name | Login Account |
| Scope | PMS Application |
| Level | user goal |
| Primary Actor | User、Manager |
| Stakeholders and Interests | User: 想要進入系統，必須先登入帳號  Manager: 想要進入系統，必須先登入帳號 |
| Preconditions | 你必須要是Team的member or leader |
| Success Guarantee | 成功登入 |
| Main Success Scenario | 1. 點選登入 2. 輸入帳號、密碼 3. 系統驗證帳號、密碼 4. 成功登入 |
| Extensions | 2a 如果輸入帳密不正確   1. 跳出警告視窗 |
| Special Requirements | UI友善 |
| Technology and Data Variations List | 2a 密碼格式不可以包含符號 |
| Frequency of Occurrence | 偶爾發生(使用系統需要登入時) |
| Miscellaneous | None |

1.6.3 Add Project

|  |  |
| --- | --- |
| Use Case Section | Comment |
| Use Case Name | Add Project |
| Scope | PMS Application |
| Level | user goal |
| Primary Actor | Manager |
| Stakeholders and Interests | Manager: 想要建立專案 |
| Preconditions | 必須是Manager，且要登入系統 |
| Success Guarantee | 能新增專案 |
| Main Success Scenario | 1. Manager登入並識別身分 2. Manager建立專案 3. Manager選擇成員進專案 4. 資料庫新增專案資料 5. 主畫面會新增一筆專案提供使用者檢視 |
| Extensions | 2a 專案名稱、描述為空或重覆   1. 跳出警告視窗提示錯誤   1.a:使用者點選確認:回到新增畫面 |
| Special Requirements | UI友善 |
| Technology and Data Variations List | None |
| Frequency of Occurrence | 經常發生(Manager新建專案的時候) |
| Miscellaneous | None |

1.6.4 Edit Project

|  |  |
| --- | --- |
| Use Case Section | Comment |
| Use Case Name | Edit Project |
| Scope | PMS Application |
| Level | user goal |
| Primary Actor | Manager |
| Stakeholders and Interests | Manager: 想要編輯專案 |
| Preconditions | 必須是Manager，且要登入系統 |
| Success Guarantee | 能編輯專案 |
| Main Success Scenario | 1. Manager登入並識別身分 2. Manager編輯專案 3. Manager編輯專案底下的成員 4. 資料庫更新專案資料 5. 主畫面會刷新專案資料提供使用者檢視 |
| Extensions | 2a 專案名稱、描述為空或重覆   1. 跳出警告視窗提示錯誤   1.a:使用者點選確認:回到編輯畫面 |
| Special Requirements | UI友善 |
| Technology and Data Variations List | None |
| Frequency of Occurrence | 經常發生(Manager編輯專案的時候) |
| Miscellaneous | None |

1.6.5 Delete Project

|  |  |
| --- | --- |
| Use Case Section | Comment |
| Use Case Name | Edit Project |
| Scope | PMS Application |
| Level | user goal |
| Primary Actor | Manager |
| Stakeholders and Interests | Manager: 想要刪除專案 |
| Preconditions | 必須是Manager，且要登入系統 |
| Success Guarantee | 能刪除專案 |
| Main Success Scenario | 1. Manager登入並識別身分 2. Manager刪除專案 3. 資料庫刪除專案資料 4. 主畫面會刷新專案資料確保專案刪除 |
| Extensions | None |
| Special Requirements | UI友善 |
| Technology and Data Variations List | None |
| Frequency of Occurrence | 經常發生(Manager刪除專案的時候) |
| Miscellaneous | None |

1.6.6 Add Requirement

|  |  |
| --- | --- |
| Use Case Section | Comment |
| Use Case Name | Add Requirement |
| Scope | PMS Application |
| Level | user goal |
| Primary Actor | Manager |
| Stakeholders and Interests | Manager: 想要建立需求 |
| Preconditions | 必須是Manager，且系統需有專案 |
| Success Guarantee | 能建立需求 |
| Main Success Scenario | 1. Manager登入並識別身分 2. Manager建立需求 3. 資料庫新增需求資料 4. 主畫面會新增需求在所屬專案下提供使用者檢視 |
| Extensions | 2a 需求名稱、描述為空或重覆  1. 跳出警告視窗提示錯誤  1.a:使用者點選確認:回到新增畫面 |
| Special Requirements | UI友善 |
| Technology and Data Variations List | None |
| Frequency of Occurrence | 經常發生(Manager新增需求的時候) |
| Miscellaneous | None |

1.6.7 Edit Requirement

|  |  |
| --- | --- |
| Use Case Section | Comment |
| Use Case Name | Edit Requirement |
| Scope | PMS Application |
| Level | user goal |
| Primary Actor | Manager |
| Stakeholders and Interests | Manager: 想要編輯需求 |
| Preconditions | 必須是Manager，且系統需有專案 |
| Success Guarantee | 能編輯需求 |
| Main Success Scenario | 1. Manager登入並識別身分 2. Manager編輯需求 3. 資料庫更新需求資料 4. 主畫面會在所屬專案下更新需求資料提供使用者檢視 |
| Extensions | 2a 需求名稱、描述為空或重覆  1. 跳出警告視窗提示錯誤  1.a:使用者點選確認:回到編輯畫面 |
| Special Requirements | UI友善 |
| Technology and Data Variations List | None |
| Frequency of Occurrence | 經常發生(Manager編輯需求的時候) |
| Miscellaneous | None |

1.6.8 Delete Requirement

|  |  |
| --- | --- |
| Use Case Section | Comment |
| Use Case Name | Delete Requirement |
| Scope | PMS Application |
| Level | user goal |
| Primary Actor | Manager |
| Stakeholders and Interests | Manager: 想要刪除需求 |
| Preconditions | 必須是Manager，且系統需有專案 |
| Success Guarantee | 能刪除需求 |
| Main Success Scenario | 1. Manager登入並識別身分 2. Manager刪除需求 3. 資料庫刪除需求資料 4. 主畫面會在所屬專案下刪除需求資料並更新畫面 |
| Extensions | none |
| Special Requirements | UI友善 |
| Technology and Data Variations List | None |
| Frequency of Occurrence | 經常發生(Manager刪除需求的時候) |
| Miscellaneous | None |

1.6.9 Add Test

|  |  |
| --- | --- |
| Use Case Section | Comment |
| Use Case Name | Add Test |
| Scope | PMS Application |
| Level | user goal |
| Primary Actor | Manager |
| Stakeholders and Interests | Manager: 想要建立測試 |
| Preconditions | 必須是Manager，且系統需有專案和需求 |
| Success Guarantee | 能新增測試 |
| Main Success Scenario | 1. Manager登入並識別身分 2. Manager新增測試 3. Manager新增測試狀態 4. 資料庫新增測試資料 5. 主畫面會在所屬專案下新增測試資料並更新畫面 |
| Extensions | 2a 測試名稱、描述為空或重覆  1. 跳出警告視窗提示錯誤  1.a:使用者點選確認:回到新增畫面 |
| Special Requirements | UI友善 |
| Technology and Data Variations List | None |
| Frequency of Occurrence | 經常發生(Manager新增測試的時候) |
| Miscellaneous | None |

1.6.10 Edit Test

|  |  |
| --- | --- |
| Use Case Section | Comment |
| Use Case Name | Edit Test |
| Scope | PMS Application |
| Level | user goal |
| Primary Actor | Manager |
| Stakeholders and Interests | Manager: 想要編輯測試 |
| Preconditions | 必須是Manager，且系統需有專案和需求 |
| Success Guarantee | 能編輯測試 |
| Main Success Scenario | 1. Manager登入並識別身分 2. Manager編輯測試 3. Manager編輯測試狀態 4. 資料庫更新測試資料 5. 主畫面會在所屬專案下更新測試資料並更新畫面 |
| Extensions | 2a 測試名稱、描述為空或重覆  1. 跳出警告視窗提示錯誤  1.a:使用者點選確認:回到編輯畫面 |
| Special Requirements | UI友善 |
| Technology and Data Variations List | None |
| Frequency of Occurrence | 經常發生(Manager編輯測試的時候) |
| Miscellaneous | None |

1.6.11 Delete Test

|  |  |
| --- | --- |
| Use Case Section | Comment |
| Use Case Name | Delete Test |
| Scope | PMS Application |
| Level | user goal |
| Primary Actor | Manager |
| Stakeholders and Interests | Manager: 想要刪除測試 |
| Preconditions | 必須是Manager，且系統需有專案和需求 |
| Success Guarantee | 能刪除測試 |
| Main Success Scenario | 1. Manager登入並識別身分 2. Manager刪除測試 3. 資料庫刪除測試資料 4. 主畫面會在所屬專案下刪除測試資料並更新畫面 |
| Extensions | None |
| Special Requirements | UI友善 |
| Technology and Data Variations List | None |
| Frequency of Occurrence | 經常發生(Manager新增測試的時候) |
| Miscellaneous | None |

1.6.12 View Project Requirement Test

|  |  |
| --- | --- |
| Use Case Section | Comment |
| Use Case Name | View Project、Requirement、Test |
| Scope | PMS Application |
| Level | user goal |
| Primary Actor | Manager、User |
| Stakeholders and Interests | Manager: 想要看到目前管理的專案，他底下的需求與測試項目、內容  User: 想要看到目前參與的專案，他底下的需求與測試項目、內容 |
| Preconditions | 必須是Manager或User，且要登入系統 |
| Success Guarantee | Manager: 看到目前管理的專案，他底下的需求與測試項目、內容  User: 看到目前參與的專案，他底下的需求與測試項目、內容 |
| Main Success Scenario | 1. 點選PMS 2. 登入並驗證 3. 看到自己所屬的專案、需求、測試列表 |
| Extensions | None |
| Special Requirements | UI友善 |
| Technology and Data Variations List | None |
| Frequency of Occurrence | 經常發生 |
| Miscellaneous | None |

1.6.13 Create traceability matrix

|  |  |
| --- | --- |
| Use Case Section | Comment |
| Use Case Name | Create traceability matrix |
| Scope | PMS Application |
| Level | user goal |
| Primary Actor | Manager、User |
| Stakeholders and Interests | Manager: 想要看到目前管理的專案，他底下的需求與測試狀況  User: 想要看到目前參與的專案，他底下的需求與測試狀況 |
| Preconditions | 必須是Manager或User，且要登入系統，至少有1個所屬專案 |
| Success Guarantee | 能看到traceability Matrix報表 |
| Main Success Scenario | 1. 點開PMS系統 2. 登入並識別身分 3. 選擇想要看到的專案報表 4. 產生追溯報表(Traceability Matrix) 5. 使用者看到報表 |
| Extensions | None |
| Special Requirements | UI友善 |
| Technology and Data Variations List | None |
| Frequency of Occurrence | 經常發生(Manager、User需要持續追蹤報表) |
| Miscellaneous | None |

1.6.14 Update Test Status

|  |  |
| --- | --- |
| Use Case Section | Comment |
| Use Case Name | Update Test Status |
| Scope | PMS Application |
| Level | user goal |
| Primary Actor | Manager、User |
| Stakeholders and Interests | Manager: 想要修改測試的完成狀態  User:想要修改測試的完成狀態 |
| Preconditions | 必須是Manager或User，且要登入系統，至少有1個所屬專案 |
| Success Guarantee | 能修改測試的完成狀態 |
| Main Success Scenario | 1. 點開PMS系統 2. 登入並識別身分 3. 點選完成的測試 4. 選擇已完成測試的需求 5. 成功更新測試狀態 |
| Extensions | None |
| Special Requirements | UI友善 |
| Technology and Data Variations List | None |
| Frequency of Occurrence | 經常發生(Manager、User會持續對需求做測試) |
| Miscellaneous | None |

1.7 Non-functional Requirements and Constraints

|  |  |  |
| --- | --- | --- |
| NFR ID | Category | Description |
| NFR‐01 | Usability | 提供友善的介面讓使用者管理專案、需求、測試 |
| NFR‐02 | Performance | DB操作時間可以在短時間內完成 |
| NFR‐03 | Performance | 更新資料後，能在短時間內刷新頁面 |
| NFR‐04 | Reliability | 確保從資料庫讀出的資料格式的正確性 |
| NFR‐05 | Reliability | 確保SQL安全性，防止SQL injection |

1.8 Glossary

|  |  |
| --- | --- |
| Item | Definition or Description |
| PMS | Project Management System(專案管理系統) |
| Traceability matrix | 雙向追溯矩陣(需求與測試狀態關係圖) |
| Manager | 專案管理者 |
| User | 專案下的員工 |
| CRUD | 資料庫新增、查詢、修改、刪除 |

1.9 Software Environments

本專案採用C#、SQL Server開發

2 Domain model

2.1 Domain class diagram showing only concepts

先從Use Case找出各個可能的Classes Identified

|  |  |  |  |
| --- | --- | --- | --- |
| Project | Requirement | Test | User |
| Manager | Traceability Matrix | Account | Member |
| Error Message | Internet | Requirement Description | Requirement Name |
| Project Description | Project Name | Test Description | Test Name |
| Account | Password | identity | PMS |
| Test Status | TestManager | ProjectManager | RequirementManager |
| UserManager |  |  |  |

根據上方找出的各個可能的Class，整理出Bad Classes

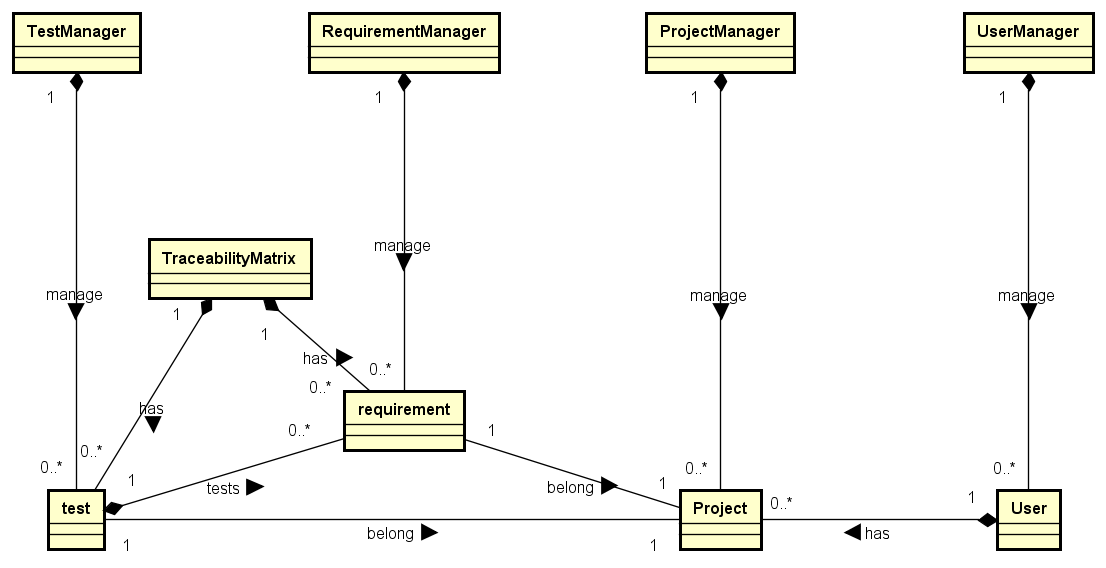
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Attributes | Vague | Operation | Roles | Implementation Construction |
| Password | Internet | Error Message | PMS |  |
| Account |  |  | Member |  |
| Project Name |  |  | Manager |  |
| Project Description |  |  |  |  |
| Requirement Name |  |  |  |  |
| Requirement Description |  |  |  |  |
| Test Name |  |  |  |  |
| Test Description |  |  |  |  |
| identity |  |  |  |  |
| Test Status |  |  |  |  |

* Attributes：屬於某個Class的屬性
* Vague：模糊不清，可忽視
* Operation：操作步驟中的ㄧ部份
* Roles：為某角色、操作介面
* Implementation Construction：建構系統的架構

本專案適合的Class

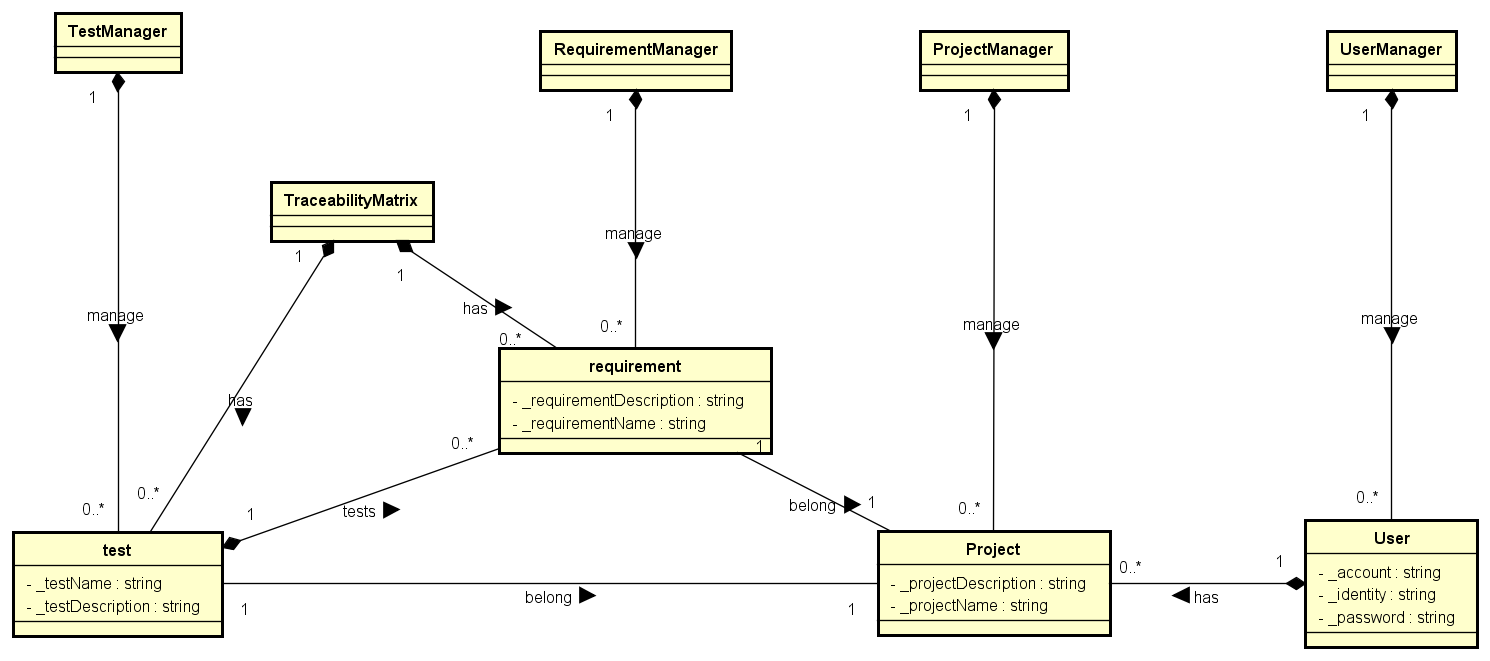
|  |  |
| --- | --- |
| Traceability Matrix | User |
| Project | Requirement |
| Test | RequirementManager |
| TestManager | ProjectManager |
| UserManager |  |

2.2 Add Associations



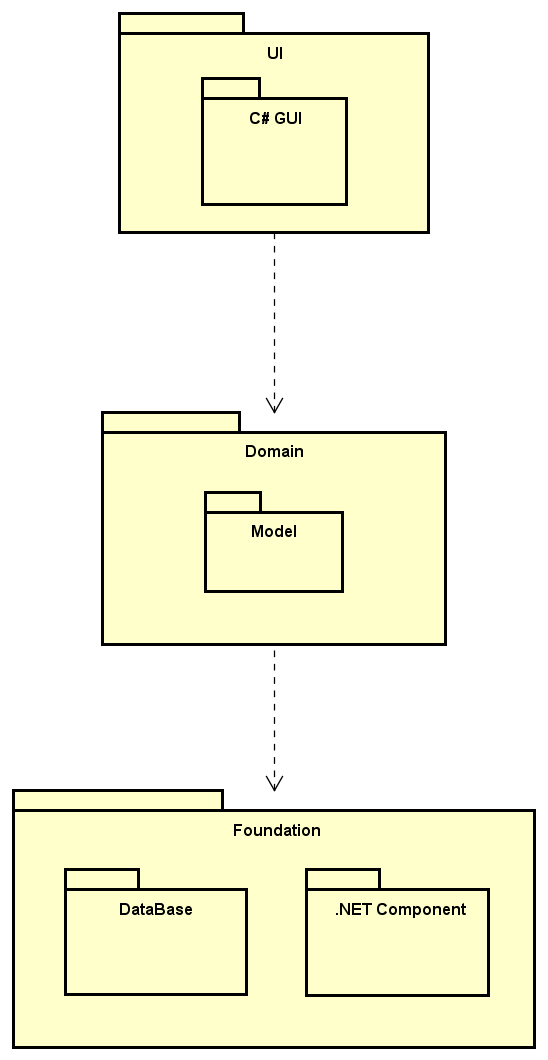
|  |
| --- |
| Relationships |
| UserManager管理多個User |
| ProjectManager管理多個Project |
| RequirementManager管理多個Requirement |
| TestManager管理多個Test |
| User擁有多個Project |
| Requirement屬於某個Project |
| Test屬於某個Project |
| Test測試多個Requirement |
| Traceability Matrix擁有某個多個Requirement的資訊 |
| Traceability Matrix擁有某個多個Test的資訊 |

2.3 Add Attributes



3 Design

3.1 Logical Architecture



3.2 Use-Case Realizations with GRASP Patterns

3.2.1 system sequence diagram

|  |
| --- |
| Register |
|  |
| Login |
|  |
| Manage Project |
|  |
| Manage Requirement |
|  |
| Manage Test |
|  |
| View Project Requirement Test |
|  |
| Create traceability matrix |
|  |

3.2.2 Operation Contract

|  |  |
| --- | --- |
| Contract ID | Operation Name |
| CO-01 | Register |
| CO-02 | Login(User) |
| CO-03 | Login(Manager) |
| CO-04 | addProject |
| CO-05 | editProject |
| CO-06 | selectMember |
| CO-07 | deleteProject |
| CO-08 | addRequirement |
| CO-09 | editRequirement |
| CO-10 | deleteRequirement |
| CO-11 | addTest |
| CO-12 | editTest |
| CO-13 | selectRequirement |
| CO-14 | deleteTest |
| CO-15 | updateTestStatus |
| CO-16 | selectProjectToShowRequirements(User) |
| CO-17 | selectProjectToShowRequirements(Manager) |
| CO-18 | selectProjectToShowTests (User) |
| CO-19 | selectProjectToShowTests (Manager) |
| CO-20 | selectProjectToShowTraceabilityMatrix(User) |
| CO-21 | selectProjectToShowTraceabilityMatrix(Manager) |

3.2.2.1 Register

|  |  |
| --- | --- |
| Operation | Register(account: string, password: string) |
| Cross References | Use Case: Register |
| Preconditions | PMS opened |
| Postconditions | Register success |

3.2.2.2 Login(User)

|  |  |
| --- | --- |
| Operation | Login(account: string, password: string) |
| Cross References | Use Case: Login |
| Preconditions | PMS opened |
| Postconditions | Login success and see default view |

3.2.2.3 Login(Manager)

|  |  |
| --- | --- |
| Operation | Login(account: string, password: string) |
| Cross References | Use Case: Login |
| Preconditions | PMS opened |
| Postconditions | Login success and see default view |

3.2.2.4 addProject

|  |  |
| --- | --- |
| Operation | addProject (projectName: string, projectDescription: string) |
| Cross References | Use Case: Manage Project |
| Preconditions | Login PMS |
| Postconditions | An Project Item was created in List |

3.2.2.5 editProject

|  |  |
| --- | --- |
| Operation | editProject (projectName: string, projectDescription: string) |
| Cross References | Use Case: Manage Project |
| Preconditions | At least exist one Project |
| Postconditions | An Project Item was edited |

3.2.2.6 selectMember

|  |  |
| --- | --- |
| Operation | selectMember () |
| Cross References | Use Case: Manage Project |
| Preconditions | At least exist one register user |
| Postconditions | Show selected user list |

3.2.2.7 deleteProject

|  |  |
| --- | --- |
| Operation | deleteProject (projectIndex: int) |
| Cross References | Use Case: Manage Project |
| Preconditions | At least exist one Project |
| Postconditions | An Project Item was deleted |

3.2.2.8 addRequirement

|  |  |
| --- | --- |
| Operation | addRequirement (requirementName: string, requirementDescription: string, project: Project) |
| Cross References | Use Case: Manage Requirement |
| Preconditions | At least exist one Project |
| Postconditions | An Requirement Item was created in List |

3.2.2.9 editRequirement

|  |  |
| --- | --- |
| Operation | editRequirement (requirementName: string, requirementDescription: string) |
| Cross References | Use Case: Manage Requirement |
| Preconditions | At least exist one Requirement |
| Postconditions | An Requirement Item was edited |

3.2.2.10 deleteRequirement

|  |  |
| --- | --- |
| Operation | deleteRequirement (requirementIndex: int) |
| Cross References | Use Case: Manage Requirement |
| Preconditions | At least exist one Requirement |
| Postconditions | An Requirement Item was deleted |

3.2.2.11 addTest

|  |  |
| --- | --- |
| Operation | addTest (requirementList: List, testName: string, testDescription: string, project: Project) |
| Cross References | Use Case: Manage Test |
| Preconditions | At least exist one Project |
| Postconditions | An Test Item was created in List |

3.2.2.12 editTest

|  |  |
| --- | --- |
| Operation | editTest (testName: string, testDescription: string) |
| Cross References | Use Case: Manage Test |
| Preconditions | At least exist one Test |
| Postconditions | An Test Item was edited |

3.2.2.13 selectRequirement

|  |  |
| --- | --- |
| Operation | selectRequirement() |
| Cross References | Use Case: Manage Test |
| Preconditions | At least exist one Requirement in project |
| Postconditions | Show selected requirement list |

3.2.2.14 deleteTest

|  |  |
| --- | --- |
| Operation | deleteTest (testIndex: int) |
| Cross References | Use Case: Manage Test |
| Preconditions | At least exist one Test |
| Postconditions | An Test Item was deleted |

3.2.2.15 updateTestStatus

|  |  |
| --- | --- |
| Operation | updateTestStatus (test: Test, requirement: Requirement) |
| Cross References | Use Case: Manage Test |
| Preconditions | At least exist one Test |
| Postconditions | An test Item requirements update complete status |

3.2.2.16 selectProjectToShowRequirements(User)

|  |  |
| --- | --- |
| Operation | selectProjectToShowRequirements (project: Project) |
| Cross References | Use Case: Manage Requirement |
| Preconditions | At least exist one Project |
| Postconditions | Show Requirement List |

3.2.2.17 selectProjectToShowRequirements(Manager)

|  |  |
| --- | --- |
| Operation | selectProjectToShowRequirements (project: Project) |
| Cross References | Use Case: Manage Requirement |
| Preconditions | At least exist one Project |
| Postconditions | Show Requirement List |

3.2.2.18 selectProjectToShowTests(User)

|  |  |
| --- | --- |
| Operation | selectProjectToShowTests (project: Project) |
| Cross References | Use Case: Manage Test |
| Preconditions | At least exist one Project |
| Postconditions | Show Test List |

3.2.2.19 selectProjectToShowTests(Manager)

|  |  |
| --- | --- |
| Operation | selectProjectToShowTests (project: Project) |
| Cross References | Use Case: Manage Test |
| Preconditions | At least exist one Project |
| Postconditions | Show Test List |

3.2.2.20 selectProjectToShowTraceabilityMatrix(User)

|  |  |
| --- | --- |
| Operation | selectProjectToShowTraceabilityMatrix (project: Project) |
| Cross References | Use Case: Create Traceability Matrix |
| Preconditions | At least exist one Project |
| Postconditions | Show Traceability Matrix |

3.2.2.21 selectProjectToShowTraceabilityMatrix(Manager)

|  |  |
| --- | --- |
| Operation | selectProjectToShowTraceabilityMatrix (project: Project) |
| Cross References | Use Case: Create Traceability Matrix |
| Preconditions | At least exist one Project |
| Postconditions | Show Traceability Matrix |

3.2.3 Operation Sequence Diagram

|  |
| --- |
| Register |
|  |

|  |
| --- |
| Login(User) |
|  |

|  |
| --- |
| Login(Manager) |
|  |
| addProject |
|  |

|  |
| --- |
| editProject |
|  |

|  |
| --- |
| selectMember |
|  |

|  |
| --- |
| deleteProject |
|  |

|  |
| --- |
| addRequirement |
|  |

|  |
| --- |
| editRequirement |
|  |

|  |
| --- |
| deleteRequirement |
|  |

|  |
| --- |
| addTest |
|  |

|  |
| --- |
| editTest |
|  |

|  |
| --- |
| selectRequirement |
|  |

|  |
| --- |
| deleteTest |
|  |

|  |
| --- |
| updateTestStatus |
|  |

|  |
| --- |
| selectProjectToShowRequirements(User) |
|  |

|  |
| --- |
| selectProjectToShowRequirements(Manager) |
|  |

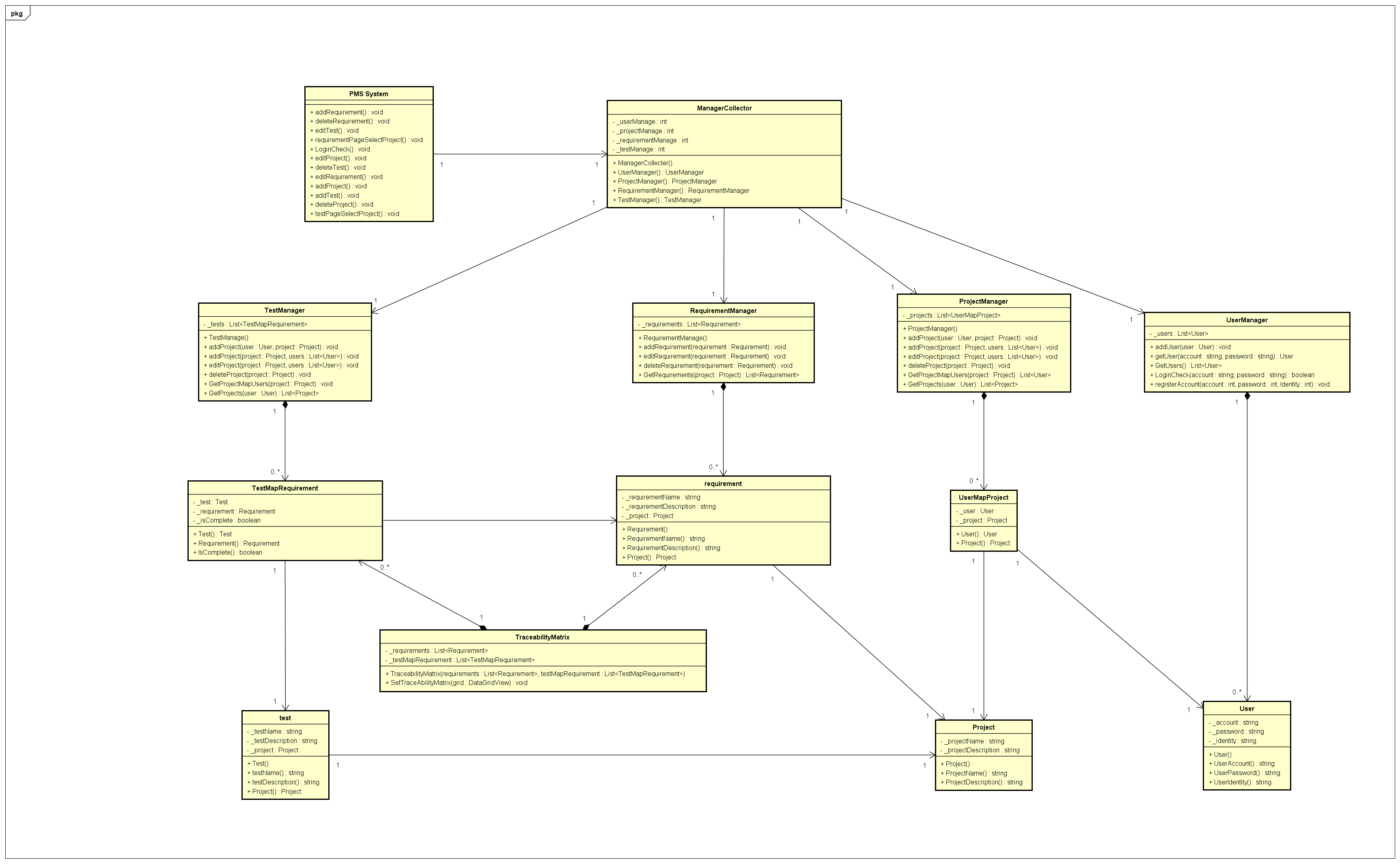
|  |
| --- |
| selectProjectToShowTests (User) |
|  |

|  |
| --- |
| selectProjectToShowTests (Manager) |
|  |

|  |
| --- |
| selectProjectToShowTraceabilityMatrix(User) |
|  |

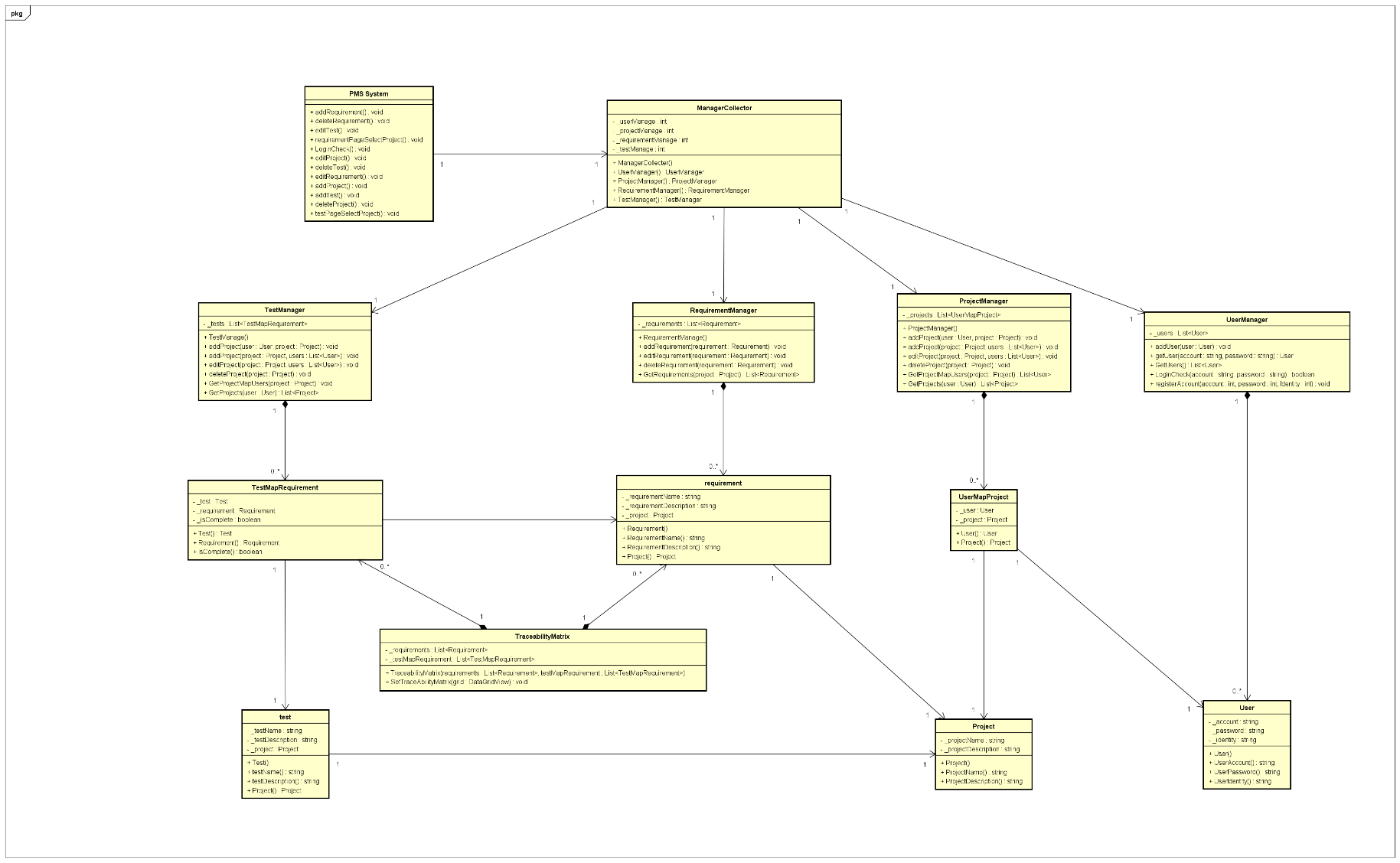
|  |
| --- |
| selectProjectToShowTraceabilityMatrix(Manager) |
|  |

3.3 Design Class Model



4 Implementation Class Model

4.1 Implementation Class Diagram



4.2 Different

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Method | Design | Imp. |
| ManagerCollecter | ManagerCollecter | X | V |
|  | UserManage | X | V |
|  | ProjectManage | X | V |
|  | RequirementManage | X | V |
|  | TestManage | X | V |
| ProjectManager | addProject | V | V |
|  | editProject | V | V |
|  | deleteProject | V | V |
|  | GetProjectMapUsers | V | V |
|  | GetProjects | V | V |
| RequirementManager | addRequirement | V | V |
|  | editRequirement | V | V |
|  | deleteRequirement | V | V |
|  | GetRequirements | V | V |
| TestManager | addTest | V | V |
|  | editTest | V | V |
|  | deleteTest | V | V |
|  | getTestMapRequirementIsComplete | V | V |
|  | updateTestMapRequirementIsComplete | V | V |
|  | GetTestMapRequirements | V | V |
|  | GetTests | V | V |
|  | GetTestMapRequirement | V | V |
| UserManager | addUser | V | V |
|  | getUser | V | V |
|  | GetUsers | V | V |
|  | LoginCheck | V | V |
|  | registerAccount | V | V |
| Project | ProjectName | V | V |
|  | ProjectDescription | V | V |
| requirement | RequirementName | V | V |
|  | RequirementDescription | V | V |
|  | Project Project | V | V |
| test | TestName | V | V |
|  | TestDescription | V | V |
|  | Project Project | V | V |
| TestMapRequirement | TestMapRequirement | V | V |
|  | Test Test | V | V |
|  | Requirement Requirement | V | V |
|  | IsComplete | V | V |
| TraceabilityMatrix | TraceabilityMatrix | V | V |
|  | SetTraceAbilityMatrix | V | V |
| User | User | V | V |
|  | UserAccount | V | V |
|  | UserPassword | V | V |
|  | UserIdentity | V | V |
| UserMapProject | UserMapProject | V | V |
|  | User User | V | V |
|  | Project Project | V | V |

Summary of implementation class/method changed

Iteration 1:

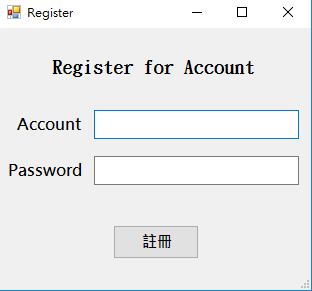
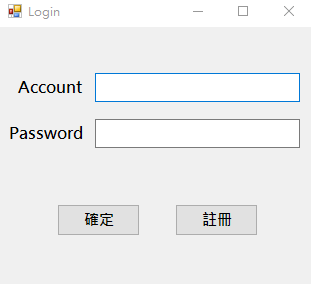
|  |  |  |  |
| --- | --- | --- | --- |
|  | Number of added | Number of removed | Number of modified |
| Class | 6 | 0 | 0 |
| Method | 38 | 0 | 0 |

* 1. Calculate Line of Code

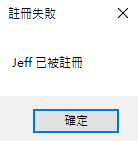
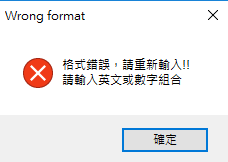
|  |  |  |  |
| --- | --- | --- | --- |
| No | Class Name | Number of methods | Line of Code in Class |
| 1 | ManagerCollector | 5 | 111 |
| 2 | ProjectManager | 7 | 53 |
| 3 | RequirementManager | 4 | 31 |
| 4 | TestManager | 9 | 63 |
| 5 | UserManager | 6 | 59 |
| 6 | Project | 2 | 34 |
| 7 | Requirement | 3 | 50 |
| 8 | Test | 3 | 51 |
| 9 | TestMapRequirement | 4 | 51 |
| 10 | TraceabilityMatrix | 2 | 62 |
| 11 | User | 4 | 53 |
| 12 | UserMapProject | 3 | 39 |
| 13 | PMS | 49 | 865 |
| Total | | 101 | 1522 |

5 Programming

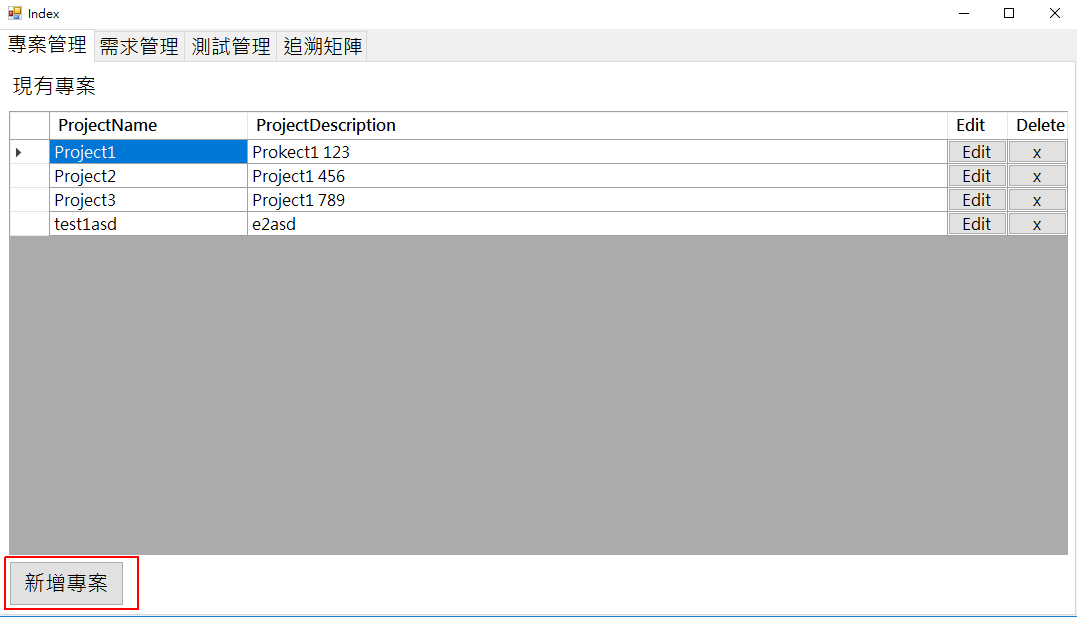
5.1 Snapshots of system execution



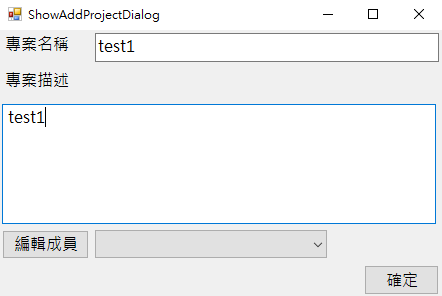
登入/註冊畫面



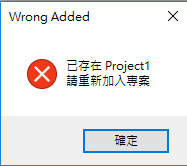
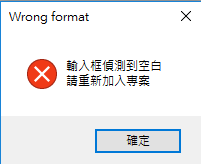
登入/註冊輸入例外檢查 帳號已被註冊例外檢查



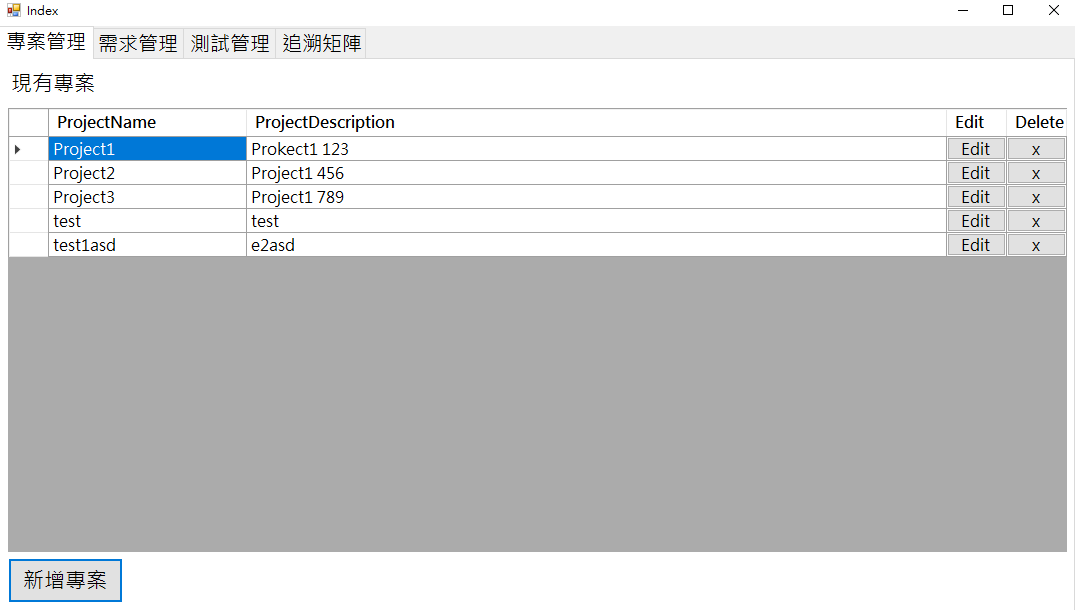
主畫面，系統會依照使用者的不同顯示使用者所參與的專案



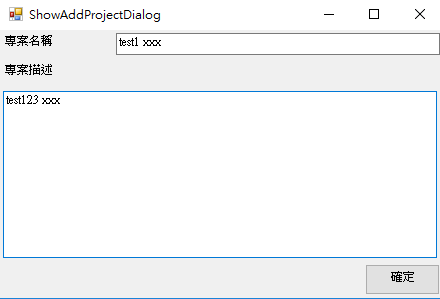
點選新增專案，進入新增專案頁面

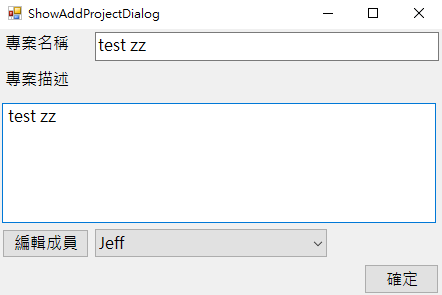
Project名稱重複例外檢查/Project名稱與描述空白例外檢查



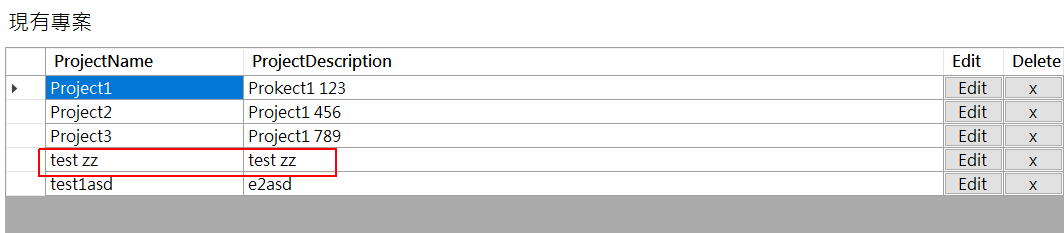
成功新增專案



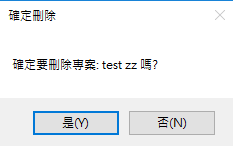
點選專案的Edit按鈕，進入編輯頁面



編輯專案



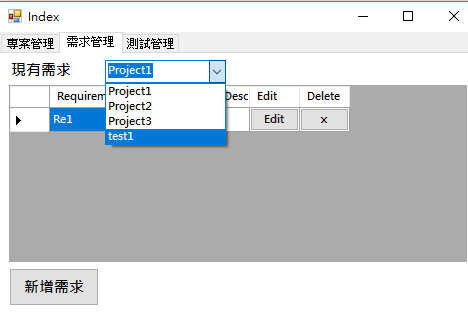
成功編輯專案



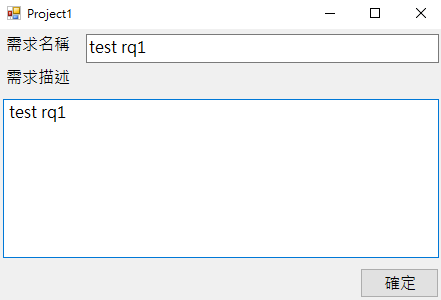
刪除專案



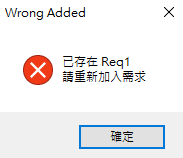
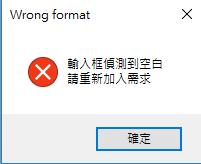
成功刪除test zz專案



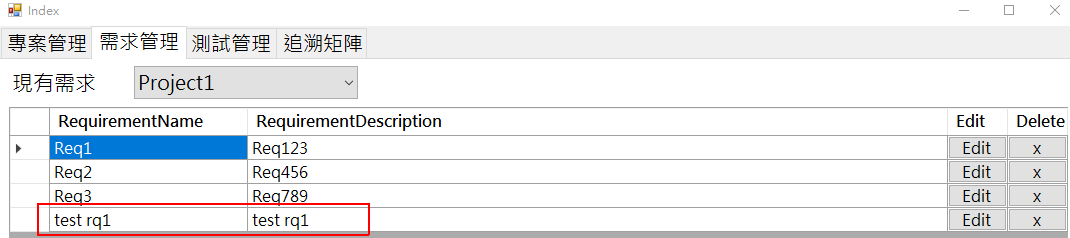
需求管理頁面，上方可選擇目前所參與的專案



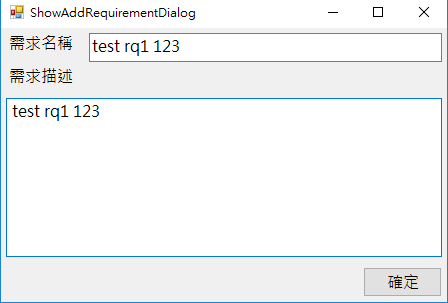
點選新增需求，進入新增需求頁面



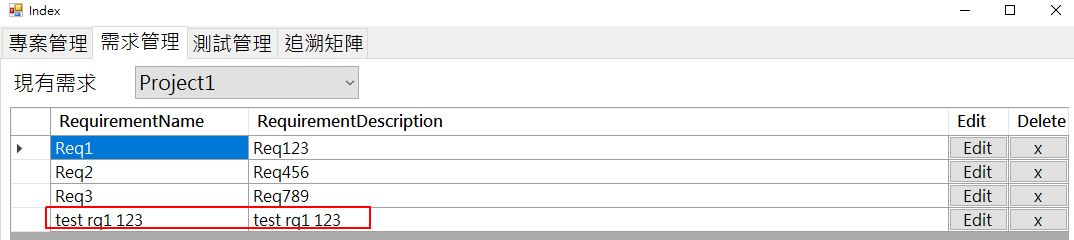
需求名稱與描述空白例外檢查/需求名稱重複例外檢查



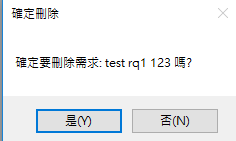
成功新增需求



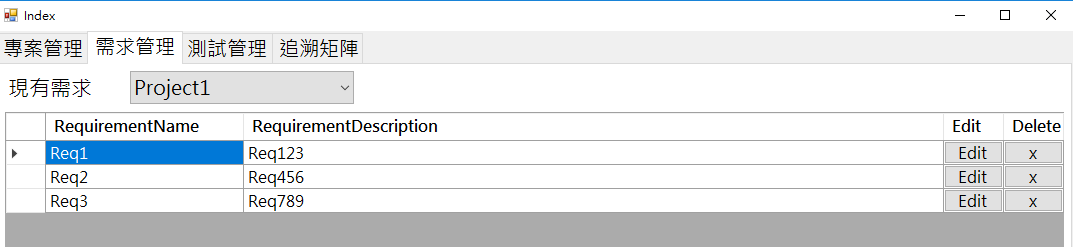
編輯需求



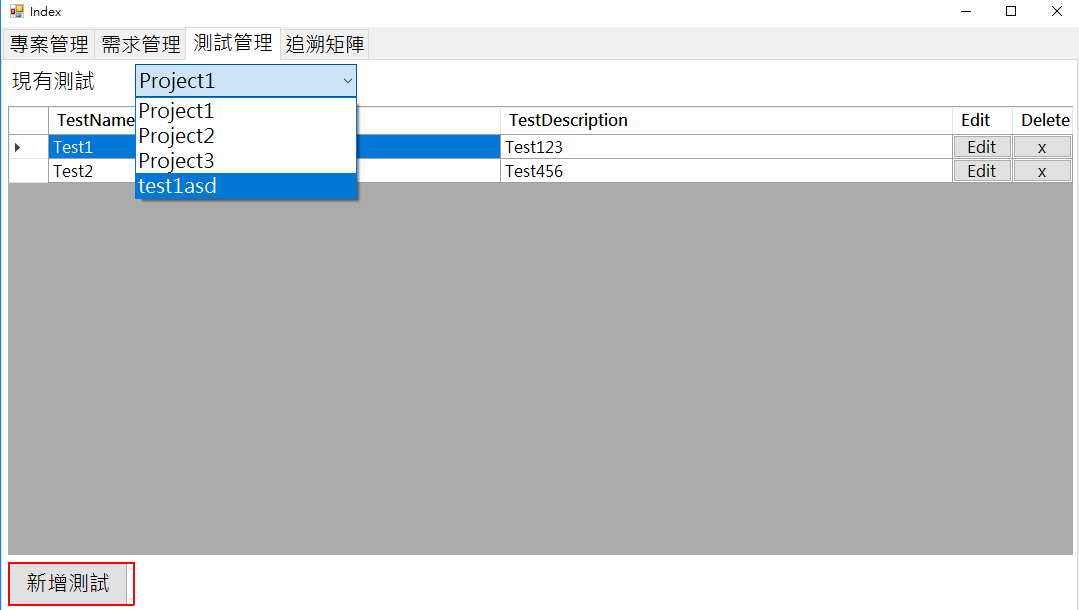
成功編輯需求



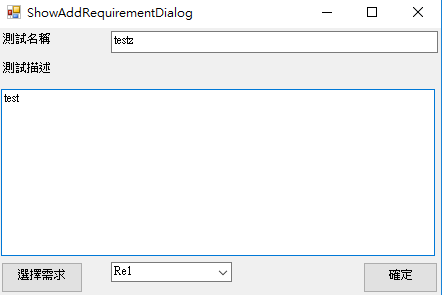
刪除需求



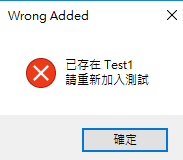
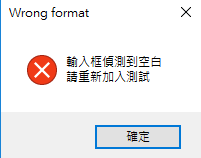
成功刪除 test rq1 123需求



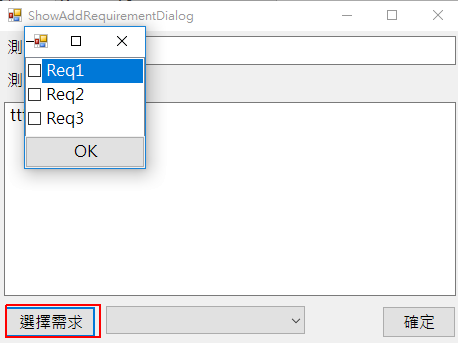
測試管理頁面，上方可選擇目前所參與的測試



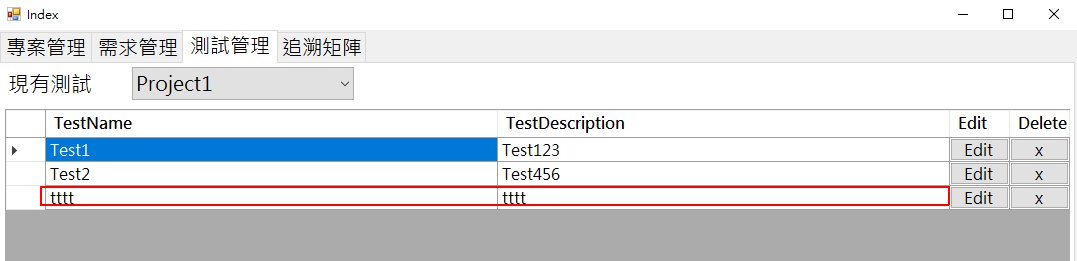
點選新增測試，進入新增測試頁面



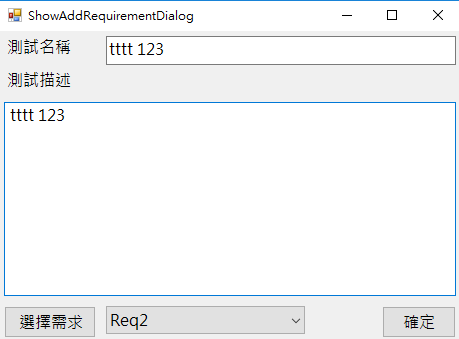
測試名稱與描述空白例外檢查/測試重複例外檢查



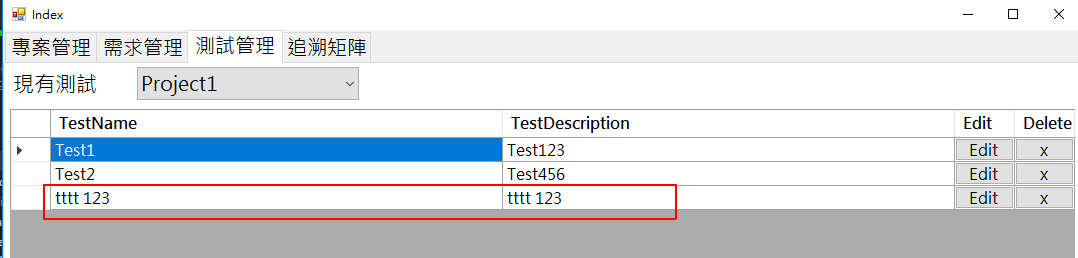
點選選擇需求，進入選擇需求頁面



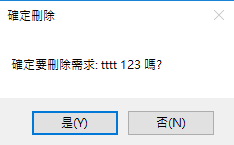
正確新增測試



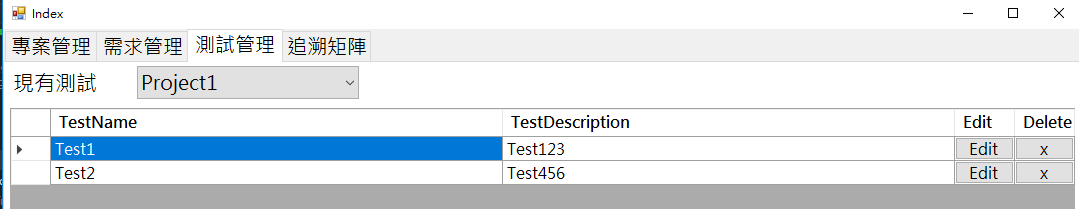
編輯測試



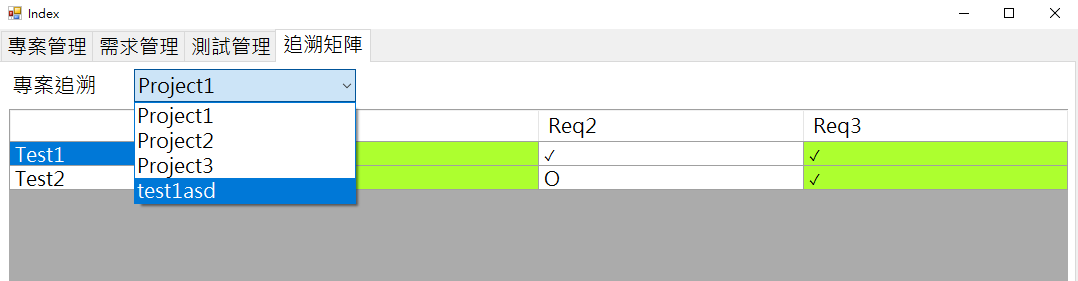
成功編輯測試



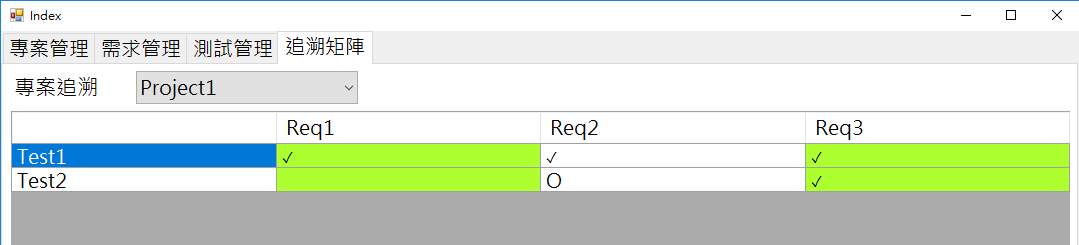
點選x按鈕，確認是否刪除需求



成功刪除需求



上方可選擇專案，顯示對應的追溯矩陣



顯示對應的需求與測試，若完成需求底下的所有測試，則顯示綠色

* 1. Source Code Listing

**ManagerCollecter.cs**

using System;

using System.Collections.Generic;

using System.Data;

using System.Linq;

using System.Text;

namespace OOAD\_RMS

{

public class ManagerCollecter

{

UserManager \_userManage;

ProjectManager \_projectManage;

RequirementManager \_requirementManage;

TestManager \_testManage;

public ManagerCollecter()

{

\_userManage = new UserManager();

\_projectManage = new ProjectManager();

\_requirementManage = new RequirementManager();

\_testManage = new TestManager();

DataTable projectTable = DBManager.GetProjects();

List<Project> projects = new List<Project>();

foreach (DataRow projectRow in projectTable.Rows)

{

Project project = new Project();

project.ProjectName = projectRow["ProjectName"].ToString();

project.ProjectDescription = projectRow["ProjectDescription"].ToString();

projects.Add(project);

DataTable requirementTable = DBManager.GetRequirementByProjectId((int)projectRow["Id"]);

foreach (DataRow reqRow in requirementTable.Rows)

{

Requirement requirement = new Requirement();

requirement.RequirementName = reqRow["RequirementName"].ToString();

requirement.RequirementDescription = reqRow["RequirementDescription"].ToString();

requirement.Project = project;

\_requirementManage.addRequirement(requirement);

}

DataTable testTable = DBManager.GetTestByProjectId((int)projectRow["Id"]);

foreach (DataRow testRow in testTable.Rows)

{

Test test = new Test();

test.TestName = testRow["TestName"].ToString();

test.TestDescription = testRow["TestDescription"].ToString();

test.Project = project;

List<Requirement> requirementsMap = new List<Requirement>();

DataTable testMapRequirementTable = DBManager.GetTestMapRequirementByTestId((int)testRow["Id"]);

foreach (DataRow testMapReq in testMapRequirementTable.Rows)

{

Requirement reqRelateTest = \_requirementManage.GetRequirements(project).Find(c => (c.RequirementName == testMapReq["RequirementName"].ToString()));

requirementsMap.Add(reqRelateTest);

\_testManage.addTest(test, reqRelateTest, (testMapReq["IsCompleted"].ToString() == "True") ? true : false);

}

}

}

DataTable userTable = DBManager.GetUsers();

foreach (DataRow userDataRow in userTable.Rows)

{

User user = new User();

user.UserAccount = userDataRow["Account"].ToString();

user.UserPassword = userDataRow["Password"].ToString();

user.UserIdentity = userDataRow["Title"].ToString();

DataTable userInProjectTable = DBManager.GetProjectByUserAccount(user.UserAccount);

foreach (DataRow projectRow in userInProjectTable.Rows)

{

Project project = projects.Find(c => (c.ProjectName == projectRow["ProjectName"].ToString() && c.ProjectDescription == projectRow["ProjectDescription"].ToString()));

\_projectManage.addProject(user, project);

}

\_userManage.addUser(user);

}

}

public UserManager UserManage

{

get {

return \_userManage;

}

}

public ProjectManager ProjectManage

{

get

{

return \_projectManage;

}

}

public RequirementManager RequirementManage

{

get

{

return \_requirementManage;

}

}

public TestManager TestManage

{

get

{

return \_testManage;

}

}

}

}

**ProjectManager.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace OOAD\_RMS

{

public class ProjectManager

{

List<UserMapProject> \_projects;

public ProjectManager()

{

\_projects = new List<UserMapProject>();

}

public void addProject(User user, Project project)

{

\_projects.Add(new UserMapProject(user, project));

}

public void addProject(Project project, List<User> users)

{

foreach (User user in users)

{

\_projects.Add(new UserMapProject(user, project));

}

}

public void editProject(Project project, List<User> users)

{

\_projects.RemoveAll(c => c.Project == project);

foreach (User user in users)

{

\_projects.Add(new UserMapProject(user, project));

}

}

public void deleteProject(Project project)

{

\_projects.RemoveAll(c => c.Project == project);

}

public List<User> GetProjectMapUsers(Project project)

{

return \_projects.FindAll(c => c.Project == project).Select(s => s.User).ToList();

}

public List<Project> GetProjects(User user)

{

return \_projects.OrderBy(y => y.Project.ProjectName).ToList().FindAll(c => c.User == user).Select(t => t.Project).ToList();

}

}

}

**RequirementManager.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace OOAD\_RMS

{

public class RequirementManager

{

List<Requirement> \_requirements = new List<Requirement>();

public void addRequirement(Requirement requirement)

{

\_requirements.Add(requirement);

}

public void editRequirement(Requirement requirement)

{

}

public void deleteRequirement(Requirement requirement)

{

\_requirements.Remove(requirement);

}

public List<Requirement> GetRequirements(Project project)

{

return \_requirements.OrderBy(y => y.RequirementName).ToList().FindAll(c => c.Project == project);

}

}

}

**TestManager.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace OOAD\_RMS

{

public class TestManager

{

List<TestMapRequirement> \_tests;

public TestManager()

{

\_tests = new List<TestMapRequirement>();

}

public void addTest(Test test, Requirement requirement, bool isComplete)

{

\_tests.Add(new TestMapRequirement(test, requirement, isComplete));

}

public void addTest(Test test, List<Requirement> requirements)

{

foreach (Requirement req in requirements)

{

\_tests.Add(new TestMapRequirement(test, req, false));

}

}

public void editTest(Test test, List<Requirement> requirements)

{

\_tests.RemoveAll(c => c.Test == test);

foreach (Requirement req in requirements)

{

\_tests.Add(new TestMapRequirement(test, req, false));

}

}

public void deleteTest(Test test)

{

\_tests.RemoveAll(c => c.Test == test);

}

public bool getTestMapRequirementIsComplete(Test test, Requirement req)

{

return \_tests.Find(c => c.Test == test && c.Requirement == req).IsComplete;

}

public void updateTestMapRequirementIsComplete(Test test, Requirement req, bool isComplete)

{

\_tests.Find(c => c.Test == test && c.Requirement == req).IsComplete = isComplete;

}

public List<Requirement> GetTestMapRequirements(Test test, Project project)

{

return \_tests.FindAll(c => c.Requirement.Project == project && c.Test.Project == project && c.Test == test).Select(s => s.Requirement).ToList();

}

public List<Test> GetTests(Project project)

{

return \_tests.Select(s => s.Test).GroupBy(g => g).Select(s => s.Key).OrderBy(y => y.TestName).ToList().FindAll(c => c.Project == project);

}

public List<TestMapRequirement> GetTestMapRequirement()

{

return \_tests;

}

}

}

**UserManager.cs**

using System;

using System.Collections.Generic;

using System.Data;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace OOAD\_RMS

{

public class UserManager

{

List<User> \_users;

public UserManager()

{

\_users = new List<User>();

}

public void addUser(User user)

{

\_users.Add(user);

}

public User getUser(string account, string password)

{

return \_users.Find(x => (x.UserAccount == account) && (x.UserPassword == password));

}

public List<User> GetUsers()

{

return \_users;

}

public bool LoginCheck(string account, string password)

{

List<User> user = \_users.FindAll(x => (x.UserAccount == account) && (x.UserPassword == password));

if (user.Count == 1)

{

DBManager.CurrentUser = user[0];

//setProject(user[0]);

return true;

}

else

return false;

}

public void registerAccount(string account, string password, string Identity)

{

DataTable userTable = DBManager.GetUsersIsRegister(account);

string NotExist = userTable.Rows[0]["Result"].ToString();

if (NotExist == "TRUE")

{

DBManager.AddUser(account, password, Identity);

MessageBox.Show(account + " 註冊成功\n回到登入畫面", "註冊成功");

}

else

MessageBox.Show(account + " 已被註冊", "註冊失敗");

}

}

}

**Project.cs**

using System;

using System.Collections.Generic;

using System.Text;

using System.Windows.Forms;

namespace OOAD\_RMS

{

public class Project

{

private string \_projectName;

private string \_projectDescription;

public string ProjectName {

get {

return \_projectName;

}

set {

\_projectName = value;

}

}

public string ProjectDescription

{

get

{

return \_projectDescription;

}

set

{

\_projectDescription = value;

}

}

}

}

**Requirement.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace OOAD\_RMS

{

public class Requirement

{

private Project \_project;

private string \_requirementName;

private string \_requirementDescription;

public string RequirementName

{

get

{

return \_requirementName;

}

set

{

\_requirementName = value;

}

}

public string RequirementDescription

{

get

{

return \_requirementDescription;

}

set

{

\_requirementDescription = value;

}

}

public Project Project

{

get

{

return \_project;

}

set

{

\_project = value;

}

}

}

}

**Test.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Text;

using System.Linq;

namespace OOAD\_RMS

{

public class Test

{

private Project \_project;

private string \_testName;

private string \_testDescription;

public string TestName

{

get

{

return \_testName;

}

set

{

\_testName = value;

}

}

public string TestDescription

{

get

{

return \_testDescription;

}

set

{

\_testDescription = value;

}

}

public Project Project

{

get

{

return \_project;

}

set

{

\_project = value;

}

}

}

}

**TestMapRequirement.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace OOAD\_RMS

{

public class TestMapRequirement

{

Test \_test;

Requirement \_requirement;

bool \_isComplete;

public TestMapRequirement(Test test, Requirement req, bool isComplete)

{

\_test = test;

\_requirement = req;

\_isComplete = isComplete;

}

public Test Test

{

get {

return \_test;

}

set {

\_test = value;

}

}

public Requirement Requirement

{

get {

return \_requirement;

}

set {

\_requirement = value;

}

}

public bool IsComplete

{

get {

return \_isComplete;

}

set {

\_isComplete = value;

}

}

}

}

**TraceabilityMatrix.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace OOAD\_RMS

{

public class TraceabilityMatrix

{

private List<Requirement> \_requirements;

private List<TestMapRequirement> \_testMapRequirement;

public TraceabilityMatrix(List<Requirement> requirements,List<TestMapRequirement> testMapRequirement)

{

\_requirements = requirements;

\_testMapRequirement = testMapRequirement;

}

public void SetTraceAbilityMatrix(DataGridView grid)

{

grid.Rows.Clear();

grid.Columns.Clear();

//x軸 (需求)

DataGridViewTextBoxColumn firstRequirementColumn = new DataGridViewTextBoxColumn();

firstRequirementColumn.HeaderText = "";

grid.Columns.Add(firstRequirementColumn);

foreach (Requirement re in \_requirements)

{

DataGridViewTextBoxColumn requirementColumn = new DataGridViewTextBoxColumn();

if (\_testMapRequirement.FindAll(f => f.Requirement == re).All(c => c.IsComplete == true))

requirementColumn.DefaultCellStyle.BackColor = System.Drawing.Color.GreenYellow;

grid.Columns.Add(requirementColumn);

requirementColumn.HeaderText = re.RequirementName;

}

//y軸 (測試)

List<Test> tests = \_testMapRequirement.Select(t => t.Test).GroupBy(g => g).Select(s => s.Key).ToList().OrderBy(y => y.TestName).ToList();

foreach (Test te in tests)

{

grid.Rows.Add(te.TestName);

}

//掃每個格子

for (int i = 0; i < tests.Count; i++)

{

for (int j = 0; j < \_requirements.Count; j++)

{

List<Requirement> requirementInTest = \_testMapRequirement.FindAll(f => f.Test == tests[i]).Select(s => s.Requirement).ToList();

if (requirementInTest.Contains(\_requirements[j]))

{

if (\_testMapRequirement.Find(f => f.Requirement == \_requirements[j] && f.Test == tests[i]).IsComplete)

grid.Rows[i].Cells[j + 1].Value = "✓";

else

grid.Rows[i].Cells[j + 1].Value = "O";

}

}

}

}

}

}

**User.cs**

using System;

using System.Collections.Generic;

using System.Text;

namespace OOAD\_RMS

{

public class User

{

private string \_account;

private string \_password;

private string \_identity;

public User()

{

}

public string UserAccount

{

get

{

return \_account;

}

set

{

\_account = value;

}

}

public string UserPassword

{

get

{

return \_password;

}

set

{

\_password = value;

}

}

public string UserIdentity

{

get

{

return \_identity;

}

set

{

\_identity = value;

}

}

}

}

**UserMapProject.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace OOAD\_RMS

{

public class UserMapProject

{

User \_user;

Project \_project;

public UserMapProject(User user, Project project)

{

\_user = user;

\_project = project;

}

public User User

{

get {

return \_user;

}

set {

\_user = value;

}

}

public Project Project

{

get {

return \_project;

}

set {

\_project = value;

}

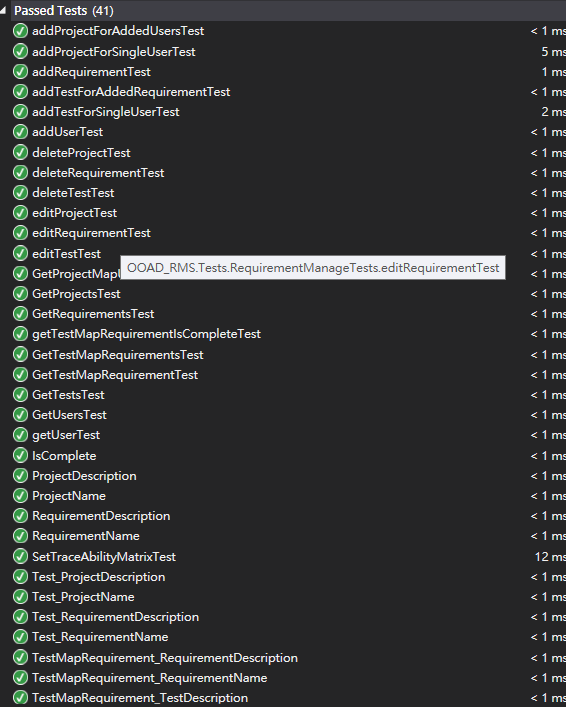
}

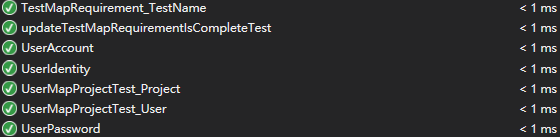
}

}

6 Unit Testing Code Listing

6.1 snapshot of testing result





6.2Unit Test Code listing

**ManagerCollecterTests.cs**

using Microsoft.VisualStudio.TestTools.UnitTesting;

using OOAD\_RMS;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace OOAD\_RMS.Tests

{

[TestClass()]

public class ProjectManageTests

{

ProjectManager \_pm;

List<User> \_users;

Project \_project;

User \_admin;

User \_member;

[TestInitialize()]

public void ProjectManageTestsInit()

{

\_pm = new ProjectManager();

\_project = new Project();

\_project.ProjectName = "pn";

\_project.ProjectDescription = "pd";

\_admin = new User();

\_admin.UserAccount = "admin";

\_admin.UserIdentity = "admin";

\_admin.UserPassword = "admin";

\_member = new User();

\_member.UserAccount = "Jeff";

\_member.UserIdentity = "member";

\_member.UserPassword = "123";

\_users = new List<User>();

\_users.Add(\_admin);

\_users.Add(\_member);

}

[TestMethod()]

public void addProjectForSingleUserTest() //使用者新增專案

{

List<Project> projectList = new List<Project>();

\_pm.addProject(\_admin, \_project);

projectList = \_pm.GetProjects(\_admin);

Assert.AreEqual("pn", projectList.Find(x => x.ProjectName == "pn").ProjectName);

Assert.AreEqual("pd", projectList.Find(x => x.ProjectName == "pn").ProjectDescription);

}

[TestMethod()]

public void addProjectForAddedUsersTest() //project底下新增的成員

{

List<User> userList = new List<User>();

\_pm.addProject(\_project, \_users);

userList = \_pm.GetProjectMapUsers(\_project);

Assert.AreEqual("admin", userList.Find(x => x.UserAccount == "admin").UserAccount);

Assert.AreEqual("Jeff", userList.Find(x => x.UserAccount == "Jeff").UserAccount);

}

[TestMethod()]

public void editProjectTest()

{

List<Project> projectList;

\_project.ProjectName = "editpn";

\_project.ProjectDescription = "editpd";

\_pm.editProject(\_project, \_users);

projectList = \_pm.GetProjects(\_admin);

Assert.AreEqual("editpn", projectList[0].ProjectName);

Assert.AreEqual("editpd", projectList[0].ProjectDescription);

}

[TestMethod()]

public void deleteProjectTest()

{

List<Project> projectList = new List<Project>();

\_pm.deleteProject(\_project);

projectList = \_pm.GetProjects(\_admin);

Assert.AreEqual(0, projectList.Count);

}

[TestMethod()]

public void GetProjectMapUsersTest()

{

List<User> userList = new List<User>();

\_pm.addProject(\_project, \_users);

userList = \_pm.GetProjectMapUsers(\_project);

Assert.AreEqual("admin", userList.Find(x => x.UserAccount == "admin").UserAccount);

Assert.AreEqual("Jeff", userList.Find(x => x.UserAccount == "Jeff").UserAccount);

}

[TestMethod()]

public void GetProjectsTest()

{

List<Project> projectList = new List<Project>();

\_pm.addProject(\_admin, \_project);

projectList = \_pm.GetProjects(\_admin);

Assert.AreEqual("pn", projectList.Find(x => x.ProjectName == "pn").ProjectName);

Assert.AreEqual("pd", projectList.Find(x => x.ProjectName == "pn").ProjectDescription);

}

}

}

**ProjectManageTests.cs**

using Microsoft.VisualStudio.TestTools.UnitTesting;

using OOAD\_RMS;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace OOAD\_RMS.Tests

{

[TestClass()]

public class ProjectManageTests

{

ProjectManager \_pm;

List<User> \_users;

Project \_project;

User \_admin;

User \_member;

[TestInitialize()]

public void ProjectManageTestsInit()

{

\_pm = new ProjectManager();

\_project = new Project();

\_project.ProjectName = "pn";

\_project.ProjectDescription = "pd";

\_admin = new User();

\_admin.UserAccount = "admin";

\_admin.UserIdentity = "admin";

\_admin.UserPassword = "admin";

\_member = new User();

\_member.UserAccount = "Jeff";

\_member.UserIdentity = "member";

\_member.UserPassword = "123";

\_users = new List<User>();

\_users.Add(\_admin);

\_users.Add(\_member);

}

[TestMethod()]

public void addProjectForSingleUserTest() //使用者新增專案

{

List<Project> projectList = new List<Project>();

\_pm.addProject(\_admin, \_project);

projectList = \_pm.GetProjects(\_admin);

Assert.AreEqual("pn", projectList.Find(x => x.ProjectName == "pn").ProjectName);

Assert.AreEqual("pd", projectList.Find(x => x.ProjectName == "pn").ProjectDescription);

}

[TestMethod()]

public void addProjectForAddedUsersTest() //project底下新增的成員

{

List<User> userList = new List<User>();

\_pm.addProject(\_project, \_users);

userList = \_pm.GetProjectMapUsers(\_project);

Assert.AreEqual("admin", userList.Find(x => x.UserAccount == "admin").UserAccount);

Assert.AreEqual("Jeff", userList.Find(x => x.UserAccount == "Jeff").UserAccount);

}

[TestMethod()]

public void editProjectTest()

{

List<Project> projectList;

\_project.ProjectName = "editpn";

\_project.ProjectDescription = "editpd";

\_pm.editProject(\_project, \_users);

projectList = \_pm.GetProjects(\_admin);

Assert.AreEqual("editpn", projectList[0].ProjectName);

Assert.AreEqual("editpd", projectList[0].ProjectDescription);

}

[TestMethod()]

public void deleteProjectTest()

{

List<Project> projectList = new List<Project>();

\_pm.deleteProject(\_project);

projectList = \_pm.GetProjects(\_admin);

Assert.AreEqual(0, projectList.Count);

}

[TestMethod()]

public void GetProjectMapUsersTest()

{

List<User> userList = new List<User>();

\_pm.addProject(\_project, \_users);

userList = \_pm.GetProjectMapUsers(\_project);

Assert.AreEqual("admin", userList.Find(x => x.UserAccount == "admin").UserAccount);

Assert.AreEqual("Jeff", userList.Find(x => x.UserAccount == "Jeff").UserAccount);

}

[TestMethod()]

public void GetProjectsTest()

{

List<Project> projectList = new List<Project>();

\_pm.addProject(\_admin, \_project);

projectList = \_pm.GetProjects(\_admin);

Assert.AreEqual("pn", projectList.Find(x => x.ProjectName == "pn").ProjectName);

Assert.AreEqual("pd", projectList.Find(x => x.ProjectName == "pn").ProjectDescription);

}

}

}

**RequirementManageTests.cs**

using Microsoft.VisualStudio.TestTools.UnitTesting;

using OOAD\_RMS;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace OOAD\_RMS.Tests

{

[TestClass()]

public class RequirementManageTests

{

RequirementManager \_rm;

Project \_project;

Requirement \_requirement;

[TestInitialize()]

public void RequirementManageTestsInit()

{

\_rm = new RequirementManager();

\_project = new Project();

\_project.ProjectName = "pn";

\_project.ProjectDescription = "pd";

\_requirement = new Requirement();

\_requirement.RequirementDescription = "rd";

\_requirement.RequirementName = "rn";

\_requirement.Project = \_project;

}

[TestMethod()]

public void addRequirementTest()

{

List<Requirement> requirementList;

\_rm.addRequirement(\_requirement);

requirementList = \_rm.GetRequirements(\_project);

Assert.AreEqual("rn", requirementList.Find(x => x.RequirementName == "rn").RequirementName);

}

[TestMethod()]

public void editRequirementTest()

{

List<Requirement> requirementList;

\_requirement.RequirementName = "editrn";

\_requirement.RequirementDescription = "editrd";

\_rm.editRequirement(\_requirement);

\_rm.addRequirement(\_requirement);

requirementList = \_rm.GetRequirements(\_project);

Assert.AreEqual("editrn", requirementList.Find(x => x.RequirementName == "editrn").RequirementName);

Assert.AreEqual("editrd", requirementList.Find(x => x.RequirementName == "editrn").RequirementDescription);

}

[TestMethod()]

public void deleteRequirementTest()

{

List<Requirement> requirementList;

\_rm.deleteRequirement(\_requirement);

requirementList = \_rm.GetRequirements(\_project);

Assert.AreEqual(0, requirementList.Count);

}

[TestMethod()]

public void GetRequirementsTest()

{

List<Requirement> requirementList;

\_rm.addRequirement(\_requirement);

requirementList = \_rm.GetRequirements(\_project);

Assert.AreEqual("rn", requirementList.Find(x => x.RequirementName == "rn").RequirementName);

}

}

}

**TestManageTests.cs**

using Microsoft.VisualStudio.TestTools.UnitTesting;

using OOAD\_RMS;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace OOAD\_RMS.Tests

{

[TestClass()]

public class RequirementManageTests

{

RequirementManager \_rm;

Project \_project;

Requirement \_requirement;

[TestInitialize()]

public void RequirementManageTestsInit()

{

\_rm = new RequirementManager();

\_project = new Project();

\_project.ProjectName = "pn";

\_project.ProjectDescription = "pd";

\_requirement = new Requirement();

\_requirement.RequirementDescription = "rd";

\_requirement.RequirementName = "rn";

\_requirement.Project = \_project;

}

[TestMethod()]

public void addRequirementTest()

{

List<Requirement> requirementList;

\_rm.addRequirement(\_requirement);

requirementList = \_rm.GetRequirements(\_project);

Assert.AreEqual("rn", requirementList.Find(x => x.RequirementName == "rn").RequirementName);

}

[TestMethod()]

public void editRequirementTest()

{

List<Requirement> requirementList;

\_requirement.RequirementName = "editrn";

\_requirement.RequirementDescription = "editrd";

\_rm.editRequirement(\_requirement);

\_rm.addRequirement(\_requirement);

requirementList = \_rm.GetRequirements(\_project);

Assert.AreEqual("editrn", requirementList.Find(x => x.RequirementName == "editrn").RequirementName);

Assert.AreEqual("editrd", requirementList.Find(x => x.RequirementName == "editrn").RequirementDescription);

}

[TestMethod()]

public void deleteRequirementTest()

{

List<Requirement> requirementList;

\_rm.deleteRequirement(\_requirement);

requirementList = \_rm.GetRequirements(\_project);

Assert.AreEqual(0, requirementList.Count);

}

[TestMethod()]

public void GetRequirementsTest()

{

List<Requirement> requirementList;

\_rm.addRequirement(\_requirement);

requirementList = \_rm.GetRequirements(\_project);

Assert.AreEqual("rn", requirementList.Find(x => x.RequirementName == "rn").RequirementName);

}

}

}

**UserManagerTests.cs**

using Microsoft.VisualStudio.TestTools.UnitTesting;

using OOAD\_RMS;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace OOAD\_RMS.Tests

{

[TestClass()]

public class UserManagerTests

{

UserManager \_um;

User \_admin;

User \_member;

[TestInitialize()]

public void UserManagerTestsInit()

{

\_um = new UserManager();

\_admin = new User();

\_admin.UserAccount = "admin";

\_admin.UserIdentity = "admin";

\_admin.UserPassword = "admin";

\_member = new User();

\_member.UserAccount = "Jeff";

\_member.UserIdentity = "member";

\_member.UserPassword = "123";

}

[TestMethod()]

public void addUserTest()

{

\_um.addUser(\_admin);

User user = \_um.getUser("admin", "admin");

Assert.AreEqual("admin", user.UserAccount);

}

[TestMethod()]

public void getUserTest()

{

\_um.addUser(\_admin);

User user = \_um.getUser("admin", "admin");

Assert.AreEqual("admin", user.UserAccount);

}

[TestMethod()]

public void GetUsersTest()

{

List<User> userList;

\_um.addUser(\_admin);

\_um.addUser(\_member);

userList = \_um.GetUsers();

Assert.AreEqual("admin", userList[0].UserAccount);

Assert.AreEqual("Jeff", userList[1].UserAccount);

}

}

}

**ProjectTests.cs**using Microsoft.VisualStudio.TestTools.UnitTesting;

using OOAD\_RMS;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace OOAD\_RMS.Tests

{

[TestClass()]

public class ProjectTests

{

Project \_project;

[TestInitialize()]

public void ProjectInit()

{

\_project = new Project();

}

[TestMethod()]

public void ProjectName()

{

\_project.ProjectName = "test";

Assert.AreEqual("test", \_project.ProjectName);

}

[TestMethod()]

public void ProjectDescription()

{

\_project.ProjectDescription = "test123";

Assert.AreEqual("test123", \_project.ProjectDescription);

}

}

}

**RequirementTest.cs**using Microsoft.VisualStudio.TestTools.UnitTesting;

using OOAD\_RMS;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace OOAD\_RMS.Tests

{

[TestClass()]

public class RequirementTest

{

Requirement \_requirement;

[TestInitialize()]

public void RequirementInit()

{

\_requirement = new Requirement();

}

[TestMethod()]

public void RequirementName()

{

\_requirement.RequirementName = "test";

Assert.AreEqual("test", \_requirement.RequirementName);

}

[TestMethod()]

public void RequirementDescription()

{

\_requirement.RequirementDescription = "test123";

Assert.AreEqual("test123", \_requirement.RequirementDescription);

}

}

}

**TestMapRequirementTests.cs**using Microsoft.VisualStudio.TestTools.UnitTesting;

using OOAD\_RMS;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace OOAD\_RMS.Tests

{

[TestClass()]

public class RequirementTest

{

Requirement \_requirement;

[TestInitialize()]

public void RequirementInit()

{

\_requirement = new Requirement();

}

[TestMethod()]

public void RequirementName()

{

\_requirement.RequirementName = "test";

Assert.AreEqual("test", \_requirement.RequirementName);

}

[TestMethod()]

public void RequirementDescription()

{

\_requirement.RequirementDescription = "test123";

Assert.AreEqual("test123", \_requirement.RequirementDescription);

}

}

}

**TestTest.cs**

using Microsoft.VisualStudio.TestTools.UnitTesting;

using OOAD\_RMS;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace OOAD\_RMS.Tests

{

[TestClass()]

public class RequirementTest

{

Requirement \_requirement;

[TestInitialize()]

public void RequirementInit()

{

\_requirement = new Requirement();

}

[TestMethod()]

public void RequirementName()

{

\_requirement.RequirementName = "test";

Assert.AreEqual("test", \_requirement.RequirementName);

}

[TestMethod()]

public void RequirementDescription()

{

\_requirement.RequirementDescription = "test123";

Assert.AreEqual("test123", \_requirement.RequirementDescription);

}

}

}

**TraceabilityMatrixTests.cs**

using Microsoft.VisualStudio.TestTools.UnitTesting;

using OOAD\_RMS;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace OOAD\_RMS.Tests

{

[TestClass()]

public class TraceabilityMatrixTests

{

DataGridView \_grid;

Requirement \_rq1,\_rq2,\_rq3;

List<TestMapRequirement> \_tmp;

List<Requirement> \_rq;

//TestMapRequirement \_tmp1,\_tmp2,\_tmp3;

TraceabilityMatrix \_tm;

Test \_test1ForRQ1, \_test2ForRQ1, \_test1ForRQ2, \_test2ForRQ2, \_test3ForRQ2, \_test1ForRQ3;

Project \_pro1;

[TestInitialize()]

public void TraceabilityMatrixTestsInit()

{

\_pro1 = new Project();

\_pro1.ProjectName = "pn";

\_pro1.ProjectDescription = "pd";

\_rq1 = new Requirement();

\_rq1.RequirementDescription = "rd1";

\_rq1.RequirementName = "rn1";

\_rq1.Project = \_pro1;

\_rq2 = new Requirement();

\_rq2.RequirementDescription = "rd2";

\_rq2.RequirementName = "rn2";

\_rq2.Project = \_pro1;

\_rq3 = new Requirement();

\_rq3.RequirementDescription = "rd3";

\_rq3.RequirementName = "rn3";

\_rq3.Project = \_pro1;

\_rq = new List<Requirement>();

\_rq.Add(\_rq1);

\_rq.Add(\_rq2);

\_rq.Add(\_rq3);

\_test1ForRQ1 = new Test();

\_test1ForRQ1.TestName = "t1RQ1tn";

\_test1ForRQ1.TestDescription = "t1RQ1td";

\_test1ForRQ1.Project = \_pro1;

\_test2ForRQ1 = new Test();

\_test2ForRQ1.TestName = "t2RQ1tn";

\_test2ForRQ1.TestDescription = "t2RQ1td";

\_test2ForRQ1.Project = \_pro1;

\_test1ForRQ2 = new Test();

\_test1ForRQ2.TestName = "t1RQ2tn";

\_test1ForRQ2.TestDescription = "t1RQ2td";

\_test1ForRQ2.Project = \_pro1;

\_test2ForRQ2 = new Test();

\_test2ForRQ2.TestName = "t2RQ2tn";

\_test2ForRQ2.TestDescription = "t2RQ2td";

\_test2ForRQ2.Project = \_pro1;

\_test3ForRQ2 = new Test();

\_test3ForRQ2.TestName = "t3RQ2tn";

\_test3ForRQ2.TestDescription = "t3RQ2td";

\_test3ForRQ2.Project = \_pro1;

\_test1ForRQ3 = new Test();

\_test1ForRQ3.TestName = "t1RQ3tn";

\_test1ForRQ3.TestDescription = "t1RQ3td";

\_test1ForRQ3.Project = \_pro1;

\_tmp = new List<TestMapRequirement>();

\_tmp.Add(new TestMapRequirement(\_test1ForRQ1,\_rq1,false));

\_tmp.Add(new TestMapRequirement(\_test2ForRQ1, \_rq1, false));

\_tmp.Add(new TestMapRequirement(\_test1ForRQ2, \_rq2, false));

\_tmp.Add(new TestMapRequirement(\_test2ForRQ2, \_rq2, false));

\_tmp.Add(new TestMapRequirement(\_test3ForRQ2, \_rq2, false));

\_tmp.Add(new TestMapRequirement(\_test1ForRQ3, \_rq3, false));

\_tm = new TraceabilityMatrix(\_rq, \_tmp);

\_grid = new DataGridView();

}

[TestMethod()]

public void SetTraceAbilityMatrixTest()

{

\_tmp[0].IsComplete = true;

\_tm.SetTraceAbilityMatrix(\_grid);

Console.WriteLine(\_grid.Rows[0].Cells[1].Value);

Assert.AreEqual("✓", \_grid.Rows[0].Cells[1].Value);

Assert.AreEqual("O", \_grid.Rows[1].Cells[2].Value);

Assert.AreEqual("O", \_grid.Rows[2].Cells[3].Value);

Assert.AreEqual("O", \_grid.Rows[3].Cells[1].Value);

Assert.AreEqual("O", \_grid.Rows[4].Cells[2].Value);

Assert.AreEqual("O", \_grid.Rows[5].Cells[2].Value);

}

}

}

**UserMapProjectTests.cs**

using Microsoft.VisualStudio.TestTools.UnitTesting;

using OOAD\_RMS;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace OOAD\_RMS.Tests

{

[TestClass()]

public class UserMapProjectTests

{

User \_user;

Project \_project;

Tuple<User, Project> \_ump;

[TestInitialize()]

public void UserMapProject()

{

\_user = new User();

\_user.UserAccount = "admin";

\_user.UserPassword = "admin";

\_user.UserIdentity = "member";

\_project = new Project();

\_project.ProjectName = "pn";

\_project.ProjectDescription = "pd";

\_ump = new Tuple<User, Project>(\_user, \_project);

}

[TestMethod()]

public void UserMapProjectTest\_User()

{

Assert.AreEqual("admin", \_ump.Item1.UserAccount);

Assert.AreEqual("admin", \_ump.Item1.UserPassword);

Assert.AreEqual("member", \_ump.Item1.UserIdentity);

}

[TestMethod()]

public void UserMapProjectTest\_Project()

{

Assert.AreEqual("pn", \_ump.Item2.ProjectName);

Assert.AreEqual("pd", \_ump.Item2.ProjectDescription);

}

}

}

**UserTest.cs**

using Microsoft.VisualStudio.TestTools.UnitTesting;

using OOAD\_RMS;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace OOAD\_RMS.Tests

{

[TestClass()]

public class UserMapProjectTests

{

User \_user;

Project \_project;

Tuple<User, Project> \_ump;

[TestInitialize()]

public void UserMapProject()

{

\_user = new User();

\_user.UserAccount = "admin";

\_user.UserPassword = "admin";

\_user.UserIdentity = "member";

\_project = new Project();

\_project.ProjectName = "pn";

\_project.ProjectDescription = "pd";

\_ump = new Tuple<User, Project>(\_user, \_project);

}

[TestMethod()]

public void UserMapProjectTest\_User()

{

Assert.AreEqual("admin", \_ump.Item1.UserAccount);

Assert.AreEqual("admin", \_ump.Item1.UserPassword);

Assert.AreEqual("member", \_ump.Item1.UserIdentity);

}

[TestMethod()]

public void UserMapProjectTest\_Project()

{

Assert.AreEqual("pn", \_ump.Item2.ProjectName);

Assert.AreEqual("pd", \_ump.Item2.ProjectDescription);

}

}

}

Measurement

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 105598003劉彥麟 | | 105598048陳政皓 | | 105598074呂昭陞 | |
| #HW1 | | | | | |
| 2017/02/24 14:30 ~ 16:30 | 2HR | 2017/02/24 14:30 ~ 16:30 | 2HR | 2017/02/24 14:30 ~ 16:30 | 2HR |
| Total | 2HR | Total | 2HR | Total | 2HR |
| #HW2 | | | | | |
| 2017/03/14 15:30 ~ 18:30 | 3HR | 2017/03/14 15:30 ~ 18:30 | 3HR | 2017/03/14 15:30 ~ 18:30 | 3HR |
| 2017/03/15 14:00 ~ 15:15 | 1.2HR | 2017/03/15 14:00 ~ 15:15 | 1.2HR | 2017/03/15 14:00 ~ 15:15 | 1.2HR |
| 2017/03/16 14:30 ~ 16:30 | 2HR | 2017/03/16 14:30 ~ 16:30 | 2HR | 2017/03/16 14:30 ~ 16:30 | 2HR |
| 2017/03/17 1030 ~ 11:30 | 1HR | 2017/03/17 10:30 ~ 11:30 | 1HR | 2017/03/17 10:30 ~ 11:30 | 1HR |
| Total | 7.2HR | Total | 7.2HR | Total | 7.2HR |
| #HW3 | | | | | |
| 2017/03/30 10:00 ~ 12:00 | 2HR | 2017/03/30 10:00 ~ 12:00 | 2HR | 2017/03/30 10:00 ~ 12:00 | 2HR |
| Total | 2HR | Total | 2HR | Total | 2HR |
| #HW4 | | | | | |
| 2017/04/21 10:00 ~ 11:00 | 1HR | 2017/04/21 10:00 ~ 11:00 | 1HR | 2017/04/21 10:00 ~ 11:00 | 1HR |
| 2017/04/21 14:00 ~ 17:00 | 3HR | 2017/04/21 14:00 ~ 17:00 | 3HR | 2017/04/21 14:00 ~ 17:00 | 3HR |
| 2017/04/22 14:00 ~ 18:00 | 4HR | 2017/04/22 14:00 ~ 18:00 | 4HR | 2017/04/22 14:00 ~ 18:00 | 4HR |
| Total | 8HR | Total | 8HR | Total | 8HR |
| #HW5 | | | | | |
| 2017/05/04  11:00~12:00 | 1HR | 2017/05/04  11:00~12:00 | 1HR | 2017/05/04  11:00~12:00 | 1HR |
| 2017/05/04  20:00~21:00 | 1HR | 2017/05/04  20:00~21:00 | 1HR | 2017/05/04  20:00~21:00 | 1HR |
| 2017/05/05  10:00~12:00 | 2HR | 2017/05/05  10:00~12:00 | 2HR | 2017/05/05  10:00~12:00 | 2HR |
| 2017/05/06  13:00~18:00 | 5HR | 2017/05/06  13:00~18:00 | 5HR | 2017/05/06  13:00~18:00 | 5HR |
| 2017/05/07  14:00~18:00 | 4HR | 2017/05/07  14:00~18:00 | 4HR | 2017/05/07  14:00~18:00 | 4HR |
| Total | 13HR | Total | 13HR | Total | 13HR |
| #HW6 | | | | | |
| 2017/05/25  10:30~12:00 | 1.5HR | 2017/05/25  10:30~12:00 | 1.5HR | 2017/05/25  10:30~12:00 | 1.5HR |
| 2017/05/25  16:30~17:30 | 1HR | 2017/05/25  16:30~17:30 | 1HR | 2017/05/25  16:30~17:30 | 1HR |
| Total | 2.5HR | Total | 2.5HR | Total | 2.5HR |
| #HW7 | | | | | |
| 2017/06/07  19:00~21:00 | 2HR | 2017/06/07  19:00~21:00 | 2HR | 2017/06/07  19:00~21:00 | 2HR |
| 2017/06/15  09:30~12:00  14:00~16:00 | 4.5HR | 2017/06/15  09:30~12:00  14:00~16:00 | 4.5HR | 2017/06/15  09:30~12:00  14:00~16:00 | 4.5HR |
| 2017/06/16  14:00~19:00 | 5HR | 2017/06/16  14:00~19:00 | 5HR | 2017/06/16  14:00~19:00 | 5HR |
| 2017/06/17 14:00~19:30 | 5.5HR | 2017/06/17 14:00~19:30 | 5.5HR | 2017/06/17 14:00~19:30 | 5.5HR |
| 2017/06/18 14:00~20:00 | 6HR | 2017/06/18 14:00~20:00 | 6HR | 2017/06/18 14:00~20:00 | 6HR |
| Total | 23HR | Total | 23HR | Total | 23HR |