**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](#_Toc482030516)

[1.1 Change History 3](#_Toc482030517)

[1.2 Problem Statement 4](#_Toc482030518)

[1.3 System Context Diagram 5](#_Toc482030519)

[1.4 Summary of System Features 5](#_Toc482030520)

[1.5 Use Case Diagram 6](#_Toc482030521)

[1.6 Use Cases 7](#_Toc482030522)

[1.7 Non-functional Requirements and Constraints 14](#_Toc482030523)

[1.8 Glossary 14](#_Toc482030524)

[1.9 Software Environments 14](#_Toc482030525)

[2 Domain model 15](#_Toc482030526)

[2.1 Domain class diagram showing only concepts 15](#_Toc482030527)

[2.2 Add Associations 16](#_Toc482030528)

[2.3 Add Attributes 17](#_Toc482030529)

[3 Design 18](#_Toc482030530)

[3.1 Logical Architecture 18](#_Toc482030531)

[3.2 Use-Case Realizations with GRASP Patterns 19](#_Toc482030532)

[3.3 Design Class Model 37](#_Toc482030533)

[4 Implementation Class Model 38](#_Toc482030534)

[4.1 Implementation Class Diagram 38](#_Toc482030535)

[4.2 Different 38](#_Toc482030536)

[4.3 Calculate Line of Code 39](#_Toc482030537)

[5 Programming 41](#_Toc482030538)

[5.1 Snapshots of system execution 41](#_Toc482030539)

[5.2 Source Code Listing 49](#_Toc482030540)

[6 Unit Testing Code Listing 71](#_Toc482030541)

[6.1 snapshot of testing result 71](#_Toc482030542)

[6.2Unit Test Code listing 71](#_Toc482030543)

[Measurement 80](#_Toc482030544)

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 |

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 | CRUD DB |
| FEA-07 | View Project Requirement Test |
| FEA-08 | Create traceability matrix |

1.5 Use Case Diagram

C:\Users\leo\Downloads\use case (1).png

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 | 使用者開啟PMS  點選註冊按鈕  輸入使用者資訊  點選確定  系統將使用者資訊寫入資料庫  完成註冊 |
| Extensions | 4a 系統偵測到使用者重覆的狀況  要能跳出警告訊息要求使用者再輸入一次  5a 資料庫存取異常  出現錯誤訊息視窗 |
| Special Requirements | UI友善 |
| Technology and Data Variations List | 3a 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. 開啟PMS系統 2. 點選登入 3. 輸入帳號、密碼 4. 點選確定 5. 系統驗證帳號、密碼 6. 成功登入 |
| Extensions | 5a 如果輸入帳密不正確   1. 跳出警告視窗 |
| Special Requirements | 1. UI友善 |
| Technology and Data Variations List | 3a 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 Management Project

|  |  |
| --- | --- |
| Use Case Section | Comment |
| Use Case Name | Management Project |
| Scope | PMS Application |
| Level | user goal |
| Primary Actor | Manager |
| Stakeholders and Interests | Manager: 想要檢視、建立、修改、刪除專案 |
| Preconditions | 必須是Manager，且要登入系統 |
| Success Guarantee | 能管理專案 |
| Main Success Scenario | A Scenario(建立)   1. 登入並識別身分 2. 建立專案 3. 資料庫新增專案資料 4. 主畫面會新增一筆專案提供使用者檢視   B Scenario(刪除)   1. 刪除專案 2. 刪除此專案下相關的資訊(專案名稱、需求、測試、相關人員) 3. 進入資料庫刪除資料 4. 刷新頁面確保刪除   C Scenario(檢視)   1. 點選存在專案 2. 列出此專案下相關的資訊(專案名稱、需求、測試、相關人員)   D Scenario(修改)   1. 點選存在專案 2. 修改專案下的相關資訊(專案名稱、需求、測試、相關人員) 3. 把修改資料更新資料庫 4. 刷新頁面確保修改 |
| 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  12a專案名稱、描述為空或重覆   1. 跳出警告視窗   1.a:使用者點選:回到修改畫面 |
| Special Requirements | 1. UI友善 |
| Technology and Data Variations List | None |
| Frequency of Occurrence | 經常發生(Manager管理專案的時候) |
| Miscellaneous | None |

1.6.4 Management Requirement

|  |  |
| --- | --- |
| Use Case Section | Comment |
| Use Case Name | Management Requirement |
| Scope | PMS Application |
| Level | user goal |
| Primary Actor | Manager |
| Stakeholders and Interests | Manager: 想要檢視、建立、修改、刪除需求 |
| Preconditions | 必須是Manager，且系統需有專案 |
| Success Guarantee | 能管理需求 |
| Main Success Scenario | A Scenario(建立)   1. 登入並識別身分 2. 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   B Scenario(刪除)   1. 刪除需求 2. 刪除此需求下相關的資訊(測試、相關人員) 3. 進入資料庫刪除資料 4. 刷新頁面確保刪除   C Scenario(檢視)   1. 存在需求 2. 列出此需求下相關的資訊(需求名稱、需求描述、測試、相關人員)   D Scenario(修改)   1. 點選存在需求 2. 修改需求下的相關資訊(所屬專案、需求名稱、需求描述、測試、相關人員) 3. 把修改資料更新資料庫 4. 刷新頁面確保修改 |
| 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  12a需求名稱、描述為空或重覆   1. 跳出警告視窗   1.a:使用者點選確認:回到修改畫面 |
| Special Requirements | 1. UI友善 |
| Technology and Data Variations List | None |
| Frequency of Occurrence | 經常發生(Manager管理需求的時候) |
| Miscellaneous | None |

1.6.5 Management Test

|  |  |
| --- | --- |
| Use Case Section | Comment |
| Use Case Name | Management Test |
| Scope | PMS Application |
| Level | user goal |
| Primary Actor | Manager |
| Stakeholders and Interests | Manager: 想要檢視、建立、修改、刪除測試 |
| Preconditions | 必須是Manager，且系統需有專案和需求 |
| Success Guarantee | 能管理測試 |
| Main Success Scenario | A Scenario(建立)   1. 點開PMS系統 2. 登入並識別身分 3. 主畫面看到現有專案和需求列表 4. 點選建立測試 5. 輸入測試名稱、描述 6. 選擇所屬專案和需求 7. 點選確定 8. 資料庫新增測試資料 9. 主畫面會新增測試在所屬專案和需求下提供使用者檢視   B Scenario(刪除)   1. 重覆A step1~3 2. 點選刪除測試 3. 確定刪除 4. 刪除此測試下相關的資訊(測試名稱、測試描述、相關人員) 5. 進入資料庫刪除資料 6. 刷新頁面確保刪除   C Scenario(檢視)   1. 重覆A step1~3 2. 點選存在測試 3. 列出此測試下相關的資訊(測試名稱、測試描述、相關人員)   D Scenario(修改)   1. step1~3 2. 點選存在的測試 3. 修改測試資訊(所屬專案、所屬需求、測試名稱、測試描述、相關人員) 4. 點選修改確定 5. 把修改資料更新資料庫 6. 刷新頁面確保修改 |
| Extensions | 5a 需求名稱、描述為空或重覆   1. 跳出警告視窗   21a需求名稱、描述為空或重覆   1. 跳出警告視窗 |
| Special Requirements | 1. UI友善 |
| Technology and Data Variations List | None |
| Frequency of Occurrence | 經常發生(Manager管理測試的時候) |
| Miscellaneous | None |

1.6.6 CRUD DB

|  |  |
| --- | --- |
| Use Case Section | Comment |
| Use Case Name | CRUD Database |
| Scope | DB |
| Level | Sub-funtion |
| Primary Actor | PMS |
| Stakeholders and Interests | PMS:管理專案、需求、測試時能確實將資料庫更新 |
| Preconditions | 1. 必需要有網路連線 2. 資料庫要存在 |
| Success Guarantee | 正確的對DB進行操作 |
| Main Success Scenario | 對資料庫能夠正確的執行CRUD指令，資料庫可以產生對應的結果 |
| Extensions | None |
| Special Requirements | 1. UI友善 |
| Technology and Data Variations List | None |
| Frequency of Occurrence | 經常發生 |
| Miscellaneous | DB Server掛掉的處理方法 |

1.6.7 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 | 1. UI友善 |
| Technology and Data Variations List | None |
| Frequency of Occurrence | 經常發生 |
| Miscellaneous | None |

1.6.8 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. 點擊確定 5. 產生追溯報表(Traceability Matrix) 6. 使用者看到報表 |
| Extensions | None |
| Special Requirements | 1. 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 |

1.7 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#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 |

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 |  |  |  |  |
|  |  |  |  |  |

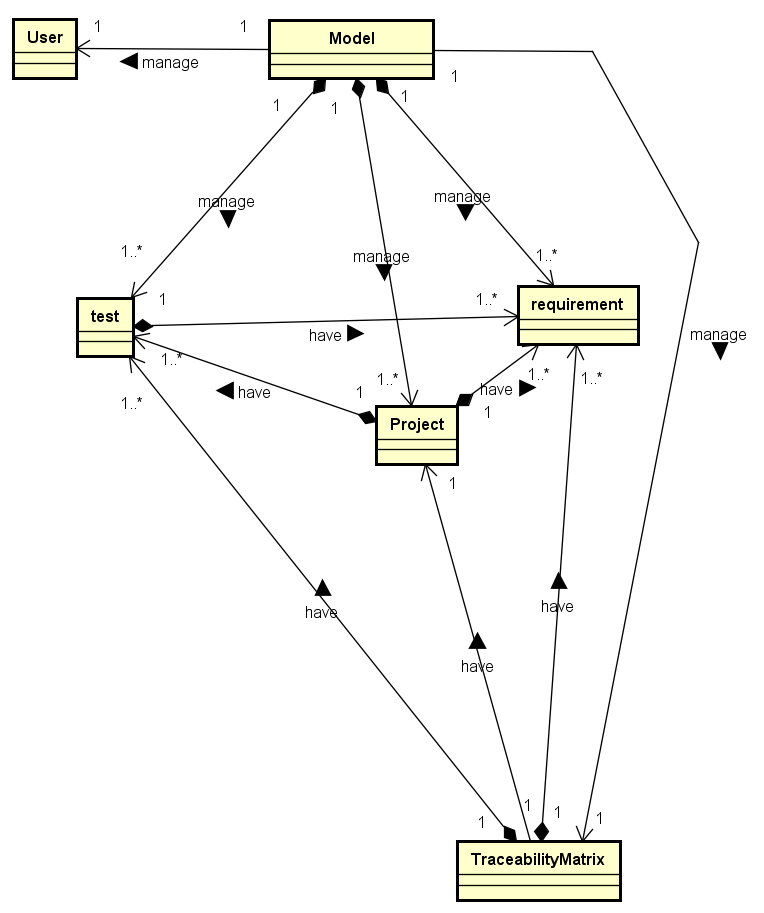
* Attributes：屬於某個Class的屬性
* Vague：模糊不清，可忽視
* Operation：操作步驟中的ㄧ部份
* Roles：為某角色、操作介面
* Implementation Construction：建構系統的架構

本專案適合的Class

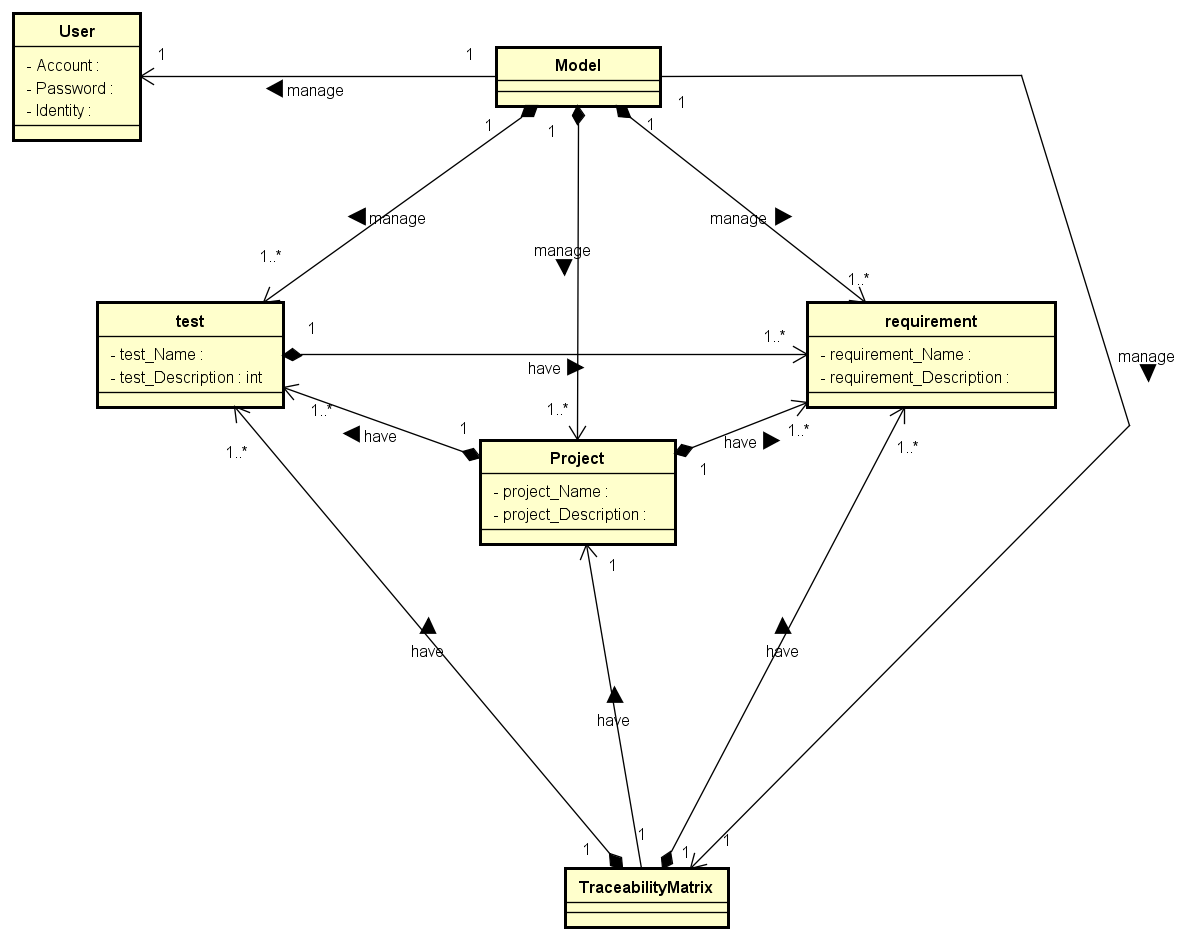
|  |  |
| --- | --- |
| Traceability Matrix | User |
| Project | Requirement |
| Test | Model |

註: 因為在設計上會採用Model作為所有資料的操作控管，因此會有Model這個Class。

2.2 Add Associations

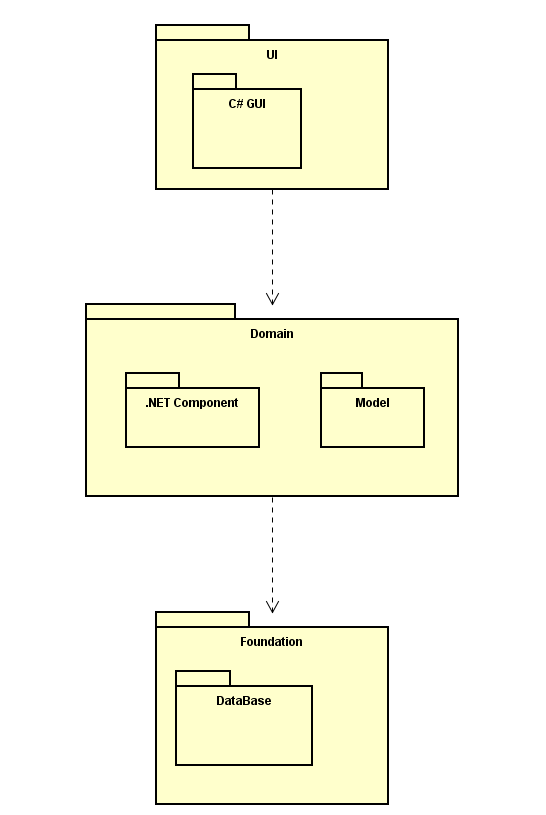


|  |
| --- |
| Relationships |
| Model管理某個User |
| Model管理某個Traceability Matrix |
| Model管理多個Project |
| Model管理多個Requirement |
| Model管理多個Test |
| Project擁有多個Requirement |
| Project擁有多個Test |
| Test擁有多個Requirement |
| Traceability Matrix擁有某個Project的資訊 |
| Traceability Matrix擁有某個Project的多個Requirement的資訊 |
| Traceability Matrix擁有某個Project的多個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

|  |
| --- |
| Manage Project |
|  |
| Manage Requirement |
|  |
| Manage Test |
|  |
| Traceability Matrix |
|  |

3.2.2 Operation Contract

|  |  |
| --- | --- |
| Contract ID | Operation Name |
| CO-01 | Login |
| CO-02 | addProject |
| CO-03 | editProject |
| CO-04 | deleteProject |
| CO-05 | selectProjectToShowRequirements |
| CO-06 | addRequirement |
| CO-07 | editRequirement |
| CO-08 | deleteRequirement |
| CO-09 | selectProjectToShowTests |
| CO-10 | addTest |
| CO-11 | editTest |
| CO-12 | deleteTest |
| CO-13 | selectProjectToShowTraceabilityMatrix |

3.2.2.1 Login

|  |  |
| --- | --- |
| Operation | Login(account: string, password: string) |
| Cross References | Use Case: Manage Project |
| Preconditions | PMS opened |
| Postconditions | Login PMS |

3.2.2.2 addProject

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

3.2.2.3 editProject

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

3.2.2.4 deleteProject

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

3.2.2.5 selectProjectToShowRequirements

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

3.2.2.6 addRequirement

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

3.2.2.7 editRequirement

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

3.2.2.8 deleteRequirement

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

3.2.2.9 selectProjectToShowTests

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

3.2.2.10 addTest

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

3.2.2.11 editTest

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

3.2.2.12 deleteTest

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

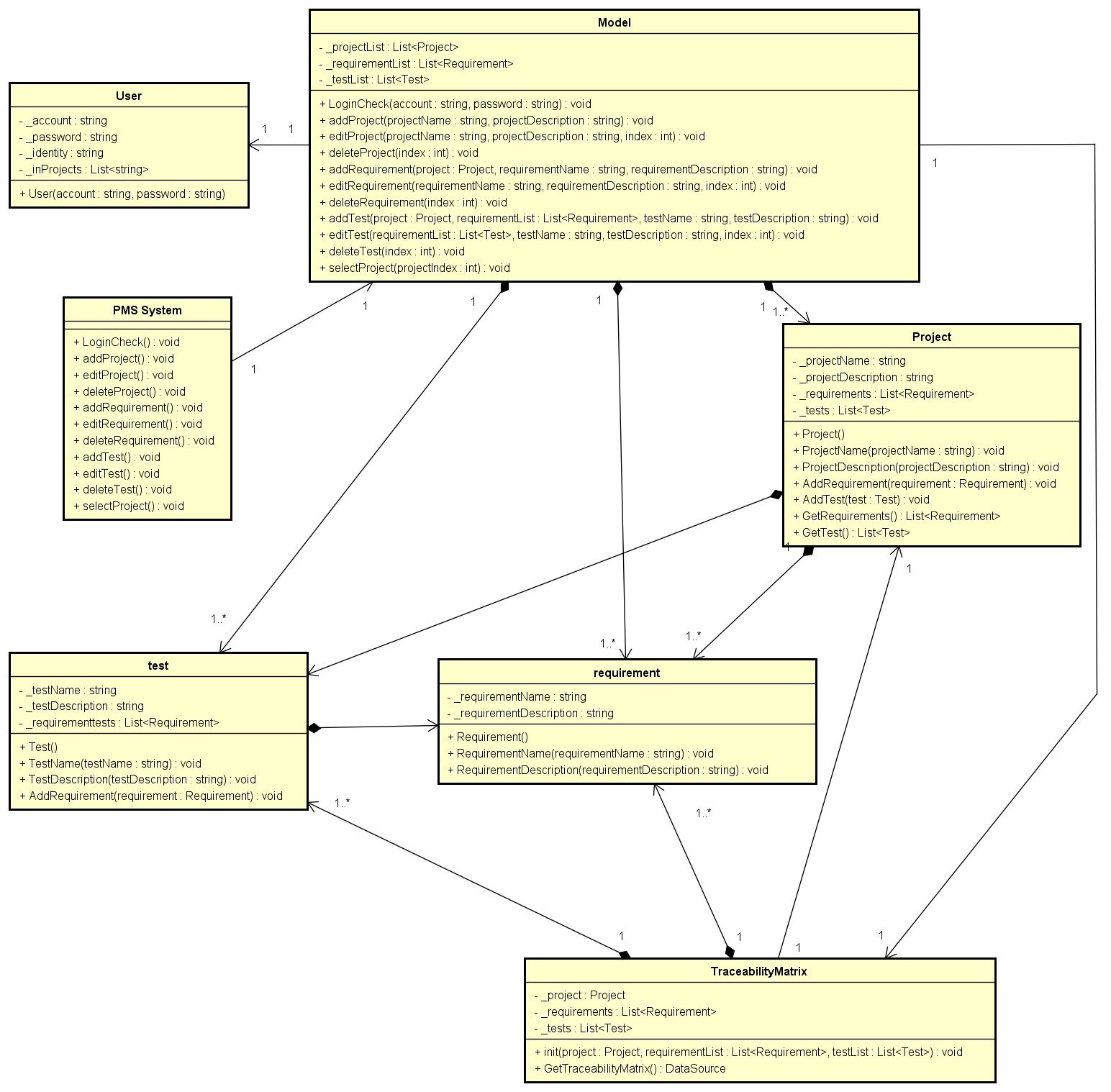
3.2.2.13 selectProjectToShowTraceabilityMatrix

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

3.2.3 Operation Sequence Diagram

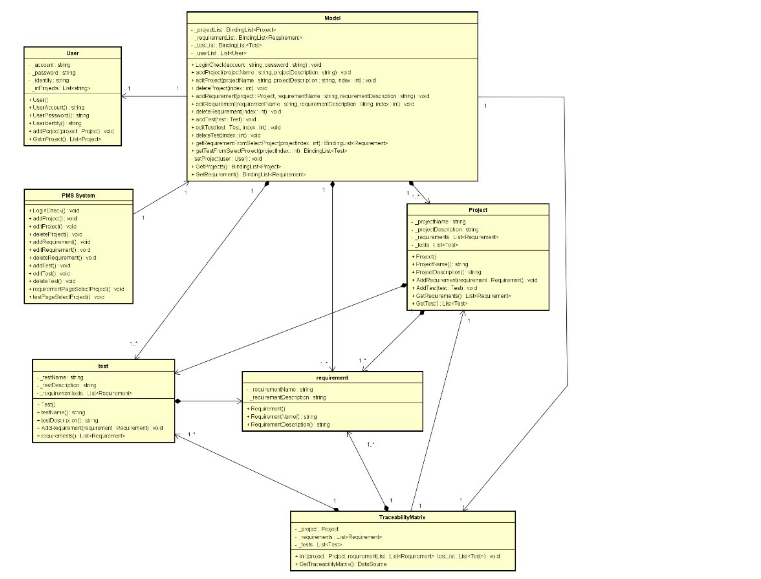
|  |
| --- |
| Log in |
|  |
| Add Project |
|  |
| Edit Project |
|  |
| Delete Project |
|  |
| Add Requirement |
|  |
| Edit Requirement |
|  |
| Delete Requirement |
|  |
| Add Test |
|  |
| Edit Test |
|  |
| Delete Test |
|  |
| selectProjectToShowRequirements |
|  |
| selectProjectToShowTests |
|  |
| selectProjectToShowTraceabilityMatrix |
|  |

3.3 Design Class Model



4 Implementation Class Model

4.1 Implementation Class Diagram



4.2 Different

|  |  |  |  |
| --- | --- | --- | --- |
| Class | Method | Design | Imp. |
| Model | LoginCheck() | V | V |
|  | addProject() | V | V |
|  | editProject() | V | V |
|  | deleteProject() | V | V |
|  | addRequirement() | V | V |
|  | edditRequirement() | V | V |
|  | deleteRequirement() | V | V |
|  | addTest() | V | V |
|  | edditTest() | V | V |
|  | deleteTest() | V | V |
|  | selectProject() | V | X |
|  | getTestFromSelectProject() | X | V |
|  | getRequirementFromSelectProject() | X | V |
|  | setProject() | X | V |
|  | GetProjects() | X | V |
|  | GetRequirement() | X | V |
| Project | Project() | V | V |
|  | ProjectName() | V | V |
|  | ProjectDescription() | V | V |
|  | AddRequirement() | V | V |
|  | AddTest() | V | V |
|  | GetRequirements() | V | V |
|  | GetTest() | V | V |
| requirement | Requirement() | V | V |
|  | RequirementName() | V | V |
|  | RequirementDescription() | V | V |
| test | Test() | V | V |
|  | TestName() | V | V |
|  | TestDescription() | V | V |
|  | AddRequirement() | V | V |
| User | User() | V | V |
|  | UserAccount() | X | V |
|  | UserPassword() | X | V |
|  | UserIdentity() | X | V |
|  | addProject() | X | V |
|  | GetInProjects() | X | V |
| TraceabilityMatrix | init() | V | X |
|  | GetTraceabilityMatrix() | V | X |

Summary of implementation class/method changed

Iteration 1:

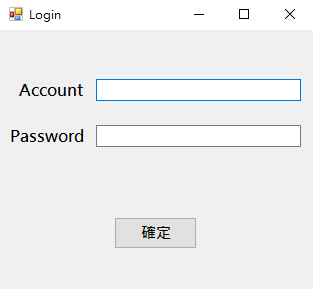
|  |  |  |  |
| --- | --- | --- | --- |
|  | Number of added | Number of removed | Number of modified |
| Class | 6 | 0 | 0 |
| Method | 38 | 0 | 0 |

4.3 Calculate Line of Code

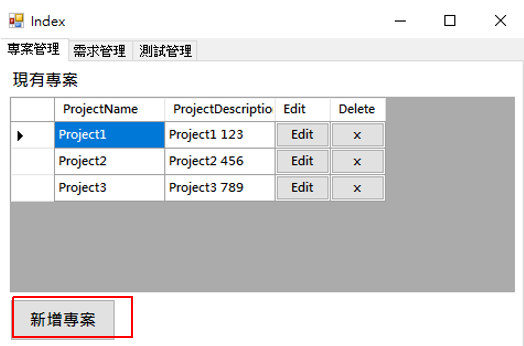
|  |  |  |  |
| --- | --- | --- | --- |
| No | Class Name | Number of methods | Line of Code in Class |
| 1 | Model | 16 | 155 |
| 2 | Project | 7 | 66 |
| 3 | requirement | 3 | 36 |
| 4 | test | 4 | 55 |
| 5 | User | 6 | 64 |
| 6 | TraceabilityMatrix | 2 | 15 |
| 7 | PMS System | 27 | 388 |
| Total | | 65 | 779 |

5 Programming

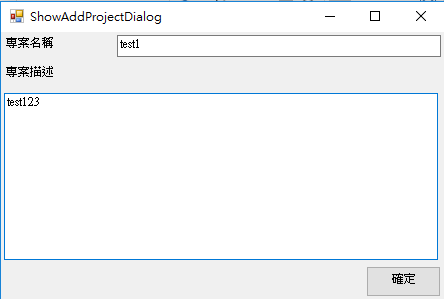
5.1 Snapshots of system execution



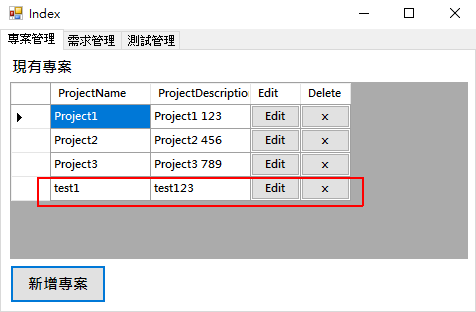
登入畫面



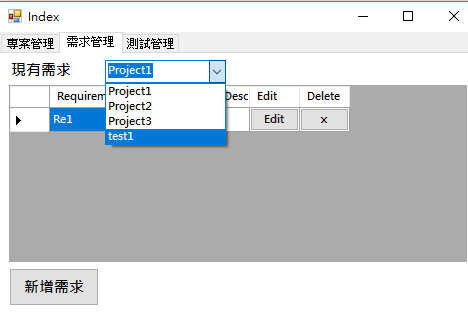
主畫面，系統會依照使用者的不同顯示使用者所參與的專案



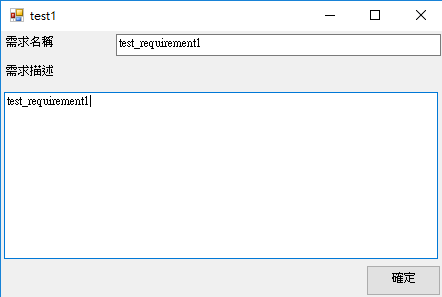
點選新增專案，進入新增專案頁面



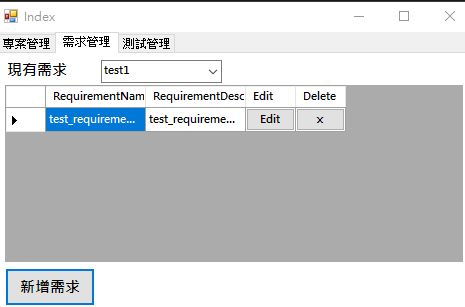
成功新增專案



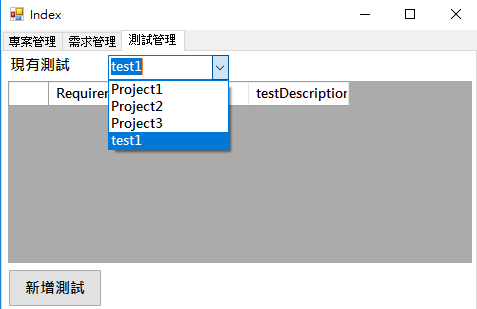
需求管理頁面，上方可選擇目前所參與的專案



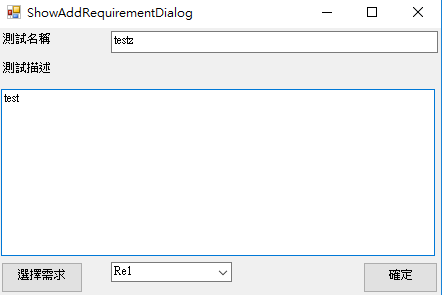
點選新增需求，進入新增需求頁面



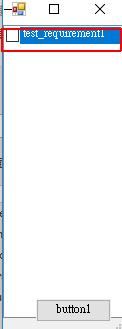
成功新增需求



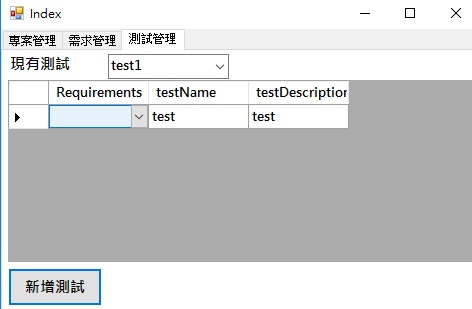
測試管理頁面，上方可選擇目前所參與的測試



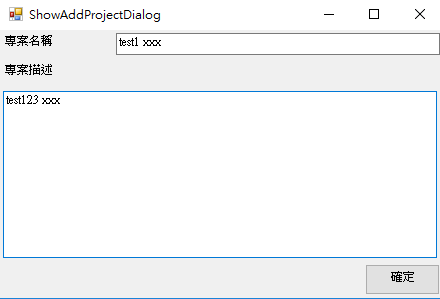
點選新增測試，進入新增測試頁面



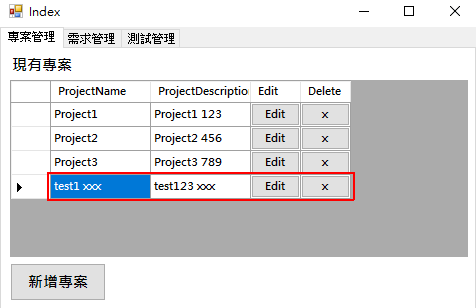
點選選擇需求，進入選擇需求頁面



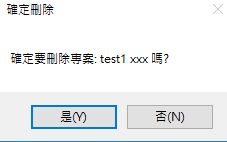
正確新增測試



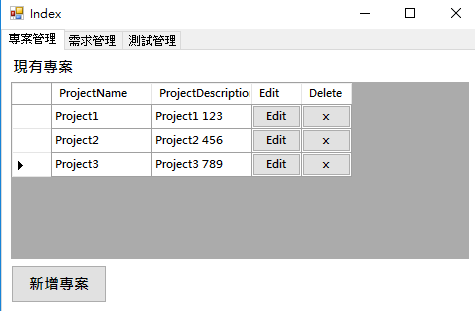
點選專案的Edit按鈕，進入編輯頁面



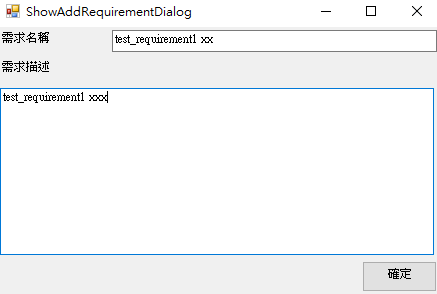
成功編輯專案



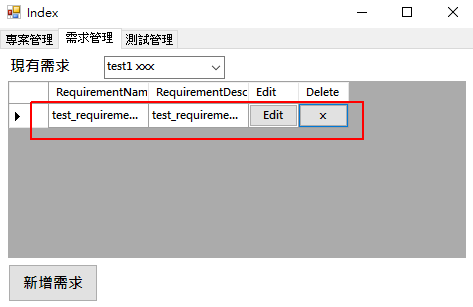
點選x按鈕，確認是否刪除專案



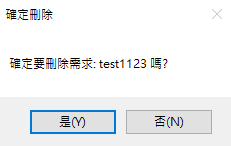
成功刪除專案



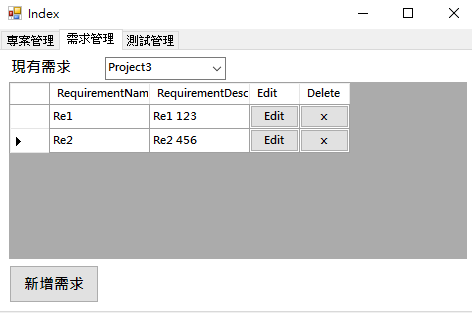
點選Edit按鈕，進入編輯需求頁面



成功編輯需求



點選x按鈕，確認是否刪除需求



成功刪除需求

5.2 Source Code Listing

Model.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Text;

using System.Windows.Forms;

namespace OOAD\_RMS

{

  public class Model

  {

  BindingList<Project> \_projectList;

  BindingList<Requirement> \_requirementList;

  BindingList<Test> \_test List;

  List<User> \_user List;

  Button c = new Button();

  public Model()

  {

  \_user List = new List<User>();

  Project project1 = new Project();

  project1.ProjectName = "Project1";

  project1.ProjectDescription = "Project1 123";

  Requirement requirement1 = new Requirement();

  requirement1.RequirementName = "Re1";

  requirement1.RequirementDescription = "Re1 123";

  project1.AddRequirement(requirement1);

  Test test1 = new Test();

  test1.testName = "Te1";

  test1.testDescription = "Te1 123";

  project1.AddTest(test1);

  Test test2 = new Test();

  test2.testName = "Te2";

  test2.testDescription = "Te2 123";

  project1.AddTest(test2);

  Project project2 = new Project();

  project2.ProjectName = "Project2";

  project2.ProjectDescription = "Project2 456";

  Project project3 = new Project();

  project3.ProjectName = "Project3";

  project3.ProjectDescription = "Project3 789";

  Requirement requirement2 = new Requirement();

  requirement2.RequirementName = "Re1";

  requirement2.RequirementDescription = "Re1 123";

  project3.AddRequirement(requirement2);

  Requirement requirement3 = new Requirement();

  requirement3.RequirementName = "Re2";

  requirement3.RequirementDescription = "Re2 456";

  project3.AddRequirement(requirement3);

  Test test3 = new Test();

  test3.testName = "Te3";

  test3.testDescription = "Te3 123";

  test3.AddRequirement(requirement3);

  project3.AddTest(test3);

  User user1 = new User();

  user1.UserAccount = "admin";

  user1.UserPassword = "admin";

  user1.UserIdentity = "Manager";

  user1.addProject(project1);

  user1.addProject(project2);

  user1.addProject(project3);

  \_userList.Add(user1);

  }

  public void addProject(string projectName, string projectDescription)

  {

  Project project = new Project();

  project.ProjectName = projectName;

  project.ProjectDescription = projectDescription;

  \_projectList.Add(project);

  }

  public BindingList<Requirement> getRequirementFromSelectProject(int projectIndex)

  {

  if (projectIndex > -1)

  \_requirementList = new BindingList<Requirement>(\_projectList[projectIndex].GetRequirements());

  return \_requirementList;

  }

  public BindingList<Test> getTestFromSelectProject(int projectIndex)

  {

  if (projectIndex > -1)

  \_testList = new BindingList<Test>(\_projectList[projectIndex].GetTests());

  return \_testList;

  }

  public void addRequirement(string requirementName, string requirementDescription)

  {

  Requirement requirement = new Requirement();

  requirement.RequirementName = requirementName;

  requirement.RequirementDescription = requirementDescription;

  \_requirementList.Add(requirement);

  }

  public void addTest(Test test)

  {

  \_testList.Add(test);

  }

  public bool checkUser(string account, string password)

  {

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

  if (user.Count == 1)

  {

  setProject(user[0]);

  return true;

  }

  else

  return false;

  }

  private void setProject(User user)

  {

  \_projectList = new BindingList<Project>(user.GetInProjects());

  }

  public BindingList<Project> GetProjects()

  {

  return \_projectList;

  }

  public BindingList<Requirement> GetRequirement()

  {

  return \_requirementList;

  }

  }

}

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;

  Button c;

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

  private List<Test> \_tests = new List<Test>();

  //example set,get projectName

  //public void setProjectName(string name) {

  // \_projectName = name;

  //}

  //public string getProjectName() {

  // return \_projectName;

  //}

  public string ProjectName {

  get {

  return \_projectName;

  }

  set {

  \_projectName = value;

  }

  }

  public string ProjectDescription

  {

  get

  {

  return \_projectDescription;

  }

  set

  {

  \_projectDescription = value;

  }

  }

  public void AddRequirement(Requirement requirement)

  {

  \_requirements.Add(requirement);

  }

  public void AddTest(Test test)

  {

  \_tests.Add(test);

  }

  public List<Requirement> GetRequirements()

  {

  return \_requirements;

  }

  public List<Test> GetTests()

  {

  return \_tests;

  }

  }

}

Requirement.cs

using System;

using System.Collections.Generic;

using System.Text;

namespace OOAD\_RMS

{

  public class Requirement

  {

  private string \_requirementName;

  private string \_requirementDescription;

  public string RequirementName

  {

  get

  {

  return \_requirementName;

  }

  set

  {

  \_requirementName = value;

  }

  }

  public string RequirementDescription

  {

  get

  {

  return \_requirementDescription;

  }

  set

  {

  \_requirementDescription = value;

  }

  }

  }

}

Test.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Text;

namespace OOAD\_RMS

{

  public class Test

  {

  private string \_testName;

  private string \_testDescription;

  private BindingList<Requirement> \_requirements = new BindingList<Requirement>();

  public string testName

  {

  get

  {

  return \_testName;

  }

  set

  {

  \_testName = value;

  }

  }

  public string testDescription

  {

  get

  {

  return \_testDescription;

  }

  set

  {

  \_testDescription = value;

  }

  }

  public void AddRequirement(Requirement requirement)

  {

  \_requirements.Add(requirement);

  }

  public BindingList<Requirement> requirements

  {

  get

  {

  return \_requirements;

  }

  set

  {

  \_requirements = value;

  }

  }

  }

}

TraceabilityMatrix.cs

using System;

using System.Collections.Generic;

using System.Text;

namespace OOAD\_RMS

{

  public class TraceabilityMatrix

  {

  private Project \_project;

  private List<Requirement> \_requirements;

  private List<Test> \_tests;

  }

}

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;

  private List<Project> \_inProjects = new List<Project>();

  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;

  }

  }

  public void addProject(Project project)

  {

  \_inProjects.Add(project);

  }

  public List<Project> GetInProjects()

  {

  return \_inProjects;

  }

  }

}

Index.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Text;

using System.Windows.Forms;

namespace OOAD\_RMS

{

  public partial class Index : Form

  {

  Model \_model;

  BindingList<Test> testList;

  public Index(Model model)

  {

  InitializeComponent();

  \_model = model;

  BindingSource projectSource = new BindingSource(\_model.GetProjects(), null);

  \_projectComboBox.DataSource = projectSource;

  \_projectComboBox.DisplayMember = "ProjectName";

  \_projectComboBox.SelectedIndex = 0;

  \_projectComboBoxTest.DataSource = projectSource;

  \_projectComboBoxTest.DisplayMember = "ProjectName";

  \_projectComboBoxTest.SelectedIndex = 0;

  DataGridViewButtonColumn editProjectBtn = new DataGridViewButtonColumn();

  \_projectGridView.Columns.Add(editProjectBtn);

  \_projectGridView.DataSource = projectSource;

  editProjectBtn.Text = "Edit";

  editProjectBtn.Name = "editBtn";

  editProjectBtn.UseColumnTextForButtonValue = true;

  editProjectBtn.HeaderText = "Edit";

  editProjectBtn.Width = 50;

  editProjectBtn.DisplayIndex = 2;

  DataGridViewButtonColumn deleteProjectBtn = new DataGridViewButtonColumn();

  \_projectGridView.Columns.Add(deleteProjectBtn);

  \_projectGridView.DataSource = projectSource;

  deleteProjectBtn.Text = "x";

  deleteProjectBtn.Name = "deleteBtn";

  deleteProjectBtn.UseColumnTextForButtonValue = true;

  deleteProjectBtn.HeaderText = "Delete";

  deleteProjectBtn.Width = 50;

  deleteProjectBtn.DisplayIndex = 3;

  DataGridViewButtonColumn editRequirementBtn = new DataGridViewButtonColumn();

  \_requirementGridView.Columns.Add(editRequirementBtn);

  editRequirementBtn.Text = "Edit";

  editRequirementBtn.Name = "editBtn";

  editRequirementBtn.UseColumnTextForButtonValue = true;

  editRequirementBtn.HeaderText = "Edit";

  editRequirementBtn.Width = 50;

  editRequirementBtn.DisplayIndex = 2;

  RequirementsColumn.DisplayMember = "RequirementName";

  DataGridViewButtonColumn deleteRequirementBtn = new DataGridViewButtonColumn();

  \_requirementGridView.Columns.Add(deleteRequirementBtn);

  deleteRequirementBtn.Text = "x";

  deleteRequirementBtn.Name = "deleteBtn";

  deleteRequirementBtn.UseColumnTextForButtonValue = true;

  deleteRequirementBtn.HeaderText = "Delete";

  deleteRequirementBtn.Width = 50;

  deleteRequirementBtn.DisplayIndex = 3;

  }

  private void ClickAddProjectBtn(object sender, EventArgs e)

  {

  ShowAddProjectDialog showAddProjectDialog = new ShowAddProjectDialog();

  if (showAddProjectDialog.ShowDialog() == DialogResult.OK) {

  string projectName = showAddProjectDialog.GetProjectName();

  string projectDescription = showAddProjectDialog.GetProjectDescription();

  \_model.addProject(projectName, projectDescription);

  }

  }

  private void ClickAddRequirementBtn(object sender, EventArgs e)

  {

  ShowAddRequirementDialog showAddRequirementDialog = new ShowAddRequirementDialog();

  showAddRequirementDialog.Text = \_projectComboBox.Text;

  if (showAddRequirementDialog.ShowDialog() == DialogResult.OK)

  {

  string requirementName = showAddRequirementDialog.GetRequirementName();

  string requirementDescription = showAddRequirementDialog.GetRequirementDescription();

  \_model.addRequirement(requirementName, requirementDescription);

  }

  }

  private void ClickAddTestBtn(object sender, EventArgs e)

  {

  ShowAddTestDialog showAddTestDialog = new ShowAddTestDialog(\_model);

  showAddTestDialog.ShowDialog();

  }

  private void ComboBoxSelectedIndexChanged(object sender, EventArgs e)

  {

  BindingSource requirementSource = new BindingSource(\_model.getRequirementFromSelectProject(\_projectComboBox.SelectedIndex), null);

  \_requirementGridView.DataSource = requirementSource;

  }

  private void TestComboBoxSelectedIndexChanged(object sender, EventArgs e)

  {

  testList = \_model.getTestFromSelectProject(\_projectComboBoxTest.SelectedIndex);

  BindingSource testSource = new BindingSource(testList, null);

  \_testGridView.DataSource = testSource;

  }

  private void SelectProjectGridViewEvent(object sender, DataGridViewCellEventArgs e)

  {

  int selectedRow = e.RowIndex;

  String getProjectNameFromDataGridView;

  String getProjectDescriptionFromDataGridView;

  DialogResult result;

  if (e.ColumnIndex == 0 && selectedRow>-1)

  {

  getProjectNameFromDataGridView = \_projectGridView.Rows[selectedRow].Cells[2].Value.ToString();

  getProjectDescriptionFromDataGridView = \_projectGridView.Rows[selectedRow].Cells[3].Value.ToString();

  ShowAddProjectDialog showAddProjectDialog = new ShowAddProjectDialog();

  showAddProjectDialog.EditProjectName(getProjectNameFromDataGridView);

  showAddProjectDialog.EditProjectDescription(getProjectDescriptionFromDataGridView);

  if (showAddProjectDialog.ShowDialog() == DialogResult.OK)

  {

  \_projectGridView.Rows[selectedRow].Cells[2].Value = showAddProjectDialog.GetProjectName();

  \_projectGridView.Rows[selectedRow].Cells[3].Value= showAddProjectDialog.GetProjectDescription();

  }

  }

  if (e.ColumnIndex == 1 && selectedRow > -1)

  {

  getProjectNameFromDataGridView = \_projectGridView.Rows[selectedRow].Cells[2].Value.ToString();

  result=MessageBox.Show("確定要刪除專案: "+ getProjectNameFromDataGridView+" 嗎?", "確定刪除",MessageBoxButtons.YesNo);

  if (result == DialogResult.Yes) {

  \_projectGridView.Rows.RemoveAt(selectedRow);

  }

  }

  }

  private void SelectRequirementGridViewEvent(object sender, DataGridViewCellEventArgs e)

  {

  int selectedRow = e.RowIndex;

  String getRequirementNameFromDataGridView;

  String getRequirementDescriptionFromDataGridView;

  DialogResult result;

  if (e.ColumnIndex == 0 && selectedRow > -1)

  {

  getRequirementNameFromDataGridView = \_requirementGridView.Rows[selectedRow].Cells[2].Value.ToString();

  getRequirementDescriptionFromDataGridView = \_requirementGridView.Rows[selectedRow].Cells[3].Value.ToString();

  ShowAddRequirementDialog requirementDialog = new ShowAddRequirementDialog();

  requirementDialog.EditRequirementName(getRequirementNameFromDataGridView);

  requirementDialog.EditRequirementDescription(getRequirementDescriptionFromDataGridView);

  if (requirementDialog.ShowDialog() == DialogResult.OK)

  {

  \_requirementGridView.Rows[selectedRow].Cells[3].Value = requirementDialog.GetRequirementDescription();

  \_requirementGridView.Rows[selectedRow].Cells[2].Value = requirementDialog.GetRequirementName();

  }

  }

  else if (e.ColumnIndex == 1 && selectedRow > -1)

  {

  getRequirementNameFromDataGridView = \_requirementGridView.Rows[selectedRow].Cells[2].Value.ToString();

  getRequirementDescriptionFromDataGridView = \_requirementGridView.Rows[selectedRow].Cells[3].Value.ToString();

  result = MessageBox.Show("確定要刪除需求: " + getRequirementNameFromDataGridView + " 嗎?", "確定刪除", MessageBoxButtons.YesNo);

  if (result == DialogResult.Yes)

  {

  \_requirementGridView.Rows.RemoveAt(selectedRow);

  }

  }

  }

  private void CompleteGridViewDataBinding(object sender, DataGridViewBindingCompleteEventArgs e)

  {

  for (int i = 0; i < \_testGridView.Rows.Count; i++)

  {

  DataGridViewComboBoxCell combo = (DataGridViewComboBoxCell)\_testGridView.Rows[i].Cells[0];

  BindingSource requirementsSource = new BindingSource(testList[i].requirements, null);

  combo.DataSource = requirementsSource;

  }

  }

  }

}

RequirementCheckList.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Text;

using System.Windows.Forms;

namespace OOAD\_RMS

{

  public partial class RequirementCheckList : Form

  {

  Model \_model;

  BindingList<Requirement> \_checkedRequirements = new BindingList<Requirement>();

  public RequirementCheckList(Model model)

  {

  \_model = model;

  InitializeComponent();

  BindingSource bs = new BindingSource(\_model.GetRequirement(), null);

  ((ListBox)\_requirementCheckedListBox).DataSource = bs;

  ((ListBox)\_requirementCheckedListBox).DisplayMember = "RequirementName";

  }

  private void \_selectReOk\_Click(object sender, EventArgs e)

  {

  for (int i = 0; i < \_requirementCheckedListBox.Items.Count; i++)

  {

  if (\_requirementCheckedListBox.GetItemCheckState(i) == CheckState.Checked)

  {

  \_checkedRequirements.Add((Requirement)\_requirementCheckedListBox.Items[i]);

  }

  }

  }

  public BindingList<Requirement> getRequirements()

  {

  foreach (Requirement re in \_checkedRequirements)

  Console.WriteLine("RequirementName: " + re.RequirementName);

  return \_checkedRequirements;

  }

  }

}

Login.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Text;

using System.Windows.Forms;

namespace OOAD\_RMS

{

  public partial class Login : Form

  {

  Model \_model;

  public Login(Model model)

  {

  \_model = model;

  InitializeComponent();

  \_passwordTextBox.PasswordChar = '\*';

  }

  private void LoginCheck(object sender, EventArgs e)

  {

  string account = \_accountTextBox.Text;

  string password = \_passwordTextBox.Text;

  if (\_model.checkUser(account, password)) {

  Hide();

  Index indexForm = new Index(\_model);

  indexForm.ShowDialog();

  Close();

  }

  }

  }

}

ShowAddProjectDialog.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Text;

using System.Windows.Forms;

namespace OOAD\_RMS

{

  public partial class ShowAddProjectDialog : Form

  {

  public ShowAddProjectDialog()

  {

  InitializeComponent();

  }

  public void EditProjectName(String projectName) {

  \_projectNameTxt.Text = projectName;

  }

  public void EditProjectDescription(String projectDescription) {

  \_projectDescriptionTxt.Text = projectDescription;

  }

  public void EditRequirement(string requirementName)

  {

  }

  public string GetProjectName()

  {

  return \_projectNameTxt.Text;

  }

  public string GetProjectDescription()

  {

  return \_projectDescriptionTxt.Text;

  }

  }

}

ShowAddRequirement.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Text;

using System.Windows.Forms;

namespace OOAD\_RMS

{

  public partial class ShowAddRequirementDialog : Form

  {

  public ShowAddRequirementDialog()

  {

  InitializeComponent();

  }

  public void EditRequirementName(string requirementName)

  {

  \_requirementNameTxt.Text = requirementName;

  }

  public void EditRequirementDescription(string requirementDescription)

  {

  \_requirementDescriptionTxt.Text = requirementDescription;

  }

  public string GetRequirementName()

  {

  return \_requirementNameTxt.Text;

  }

  public string GetRequirementDescription()

  {

  return \_requirementDescriptionTxt.Text;

  }

  }

}

ShowAddTestDialog.cs

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Text;

using System.Windows.Forms;

namespace OOAD\_RMS

{

  public partial class ShowAddTestDialog : Form

  {

  Model \_model;

  Test \_test;

  public ShowAddTestDialog(Model model)

  {

  \_model = model;

  \_test = new Test();

  InitializeComponent();

  \_testRequirementComboBox.DisplayMember = "RequirementName";

  }

  private void \_editRequirementList\_Click(object sender, EventArgs e)

  {

  RequirementCheckList requirementCheckList = new RequirementCheckList(\_model);

  if (requirementCheckList.ShowDialog() == DialogResult.OK)

  {

  \_test.requirements = requirementCheckList.getRequirements();

  BindingSource testSource = new BindingSource(requirementCheckList.getRequirements(), null);

  \_testRequirementComboBox.DataSource = testSource;

  }

  }

  private void \_okBtn\_Click(object sender, EventArgs e)

  {

  \_test.testName = \_testNameTxt.Text;

  \_test.testDescription = \_testDescriptionTxt.Text;

  \_model.addTest(\_test);

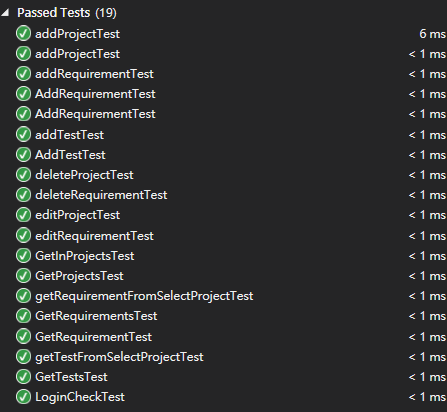
  }

  }

}

6 Unit Testing Code Listing

6.1 snapshot of testing result



6.2Unit Test Code listing

**ModelTest.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 ModelTests

{

Model \_model;

[TestInitialize()]

public void ModelInitialize()

{

\_model = new Model();

Project project1 = new Project();

project1.ProjectName = "Project1";

project1.ProjectDescription = "Project1 123";

Requirement requirement1 = new Requirement();

requirement1.RequirementName = "Re1";

requirement1.RequirementDescription = "Re1 123";

project1.AddRequirement(requirement1);

Test test1 = new Test();

test1.testName = "Te1";

test1.testDescription = "Te1 123";

project1.AddTest(test1);

Test test2 = new Test();

test2.testName = "Te2";

test2.testDescription = "Te2 123";

project1.AddTest(test2);

Project project2 = new Project();

project2.ProjectName = "Project2";

project2.ProjectDescription = "Project2 456";

Project project3 = new Project();

project3.ProjectName = "Project3";

project3.ProjectDescription = "Project3 789";

Requirement requirement2 = new Requirement();

requirement2.RequirementName = "Re1";

requirement2.RequirementDescription = "Re1 123";

project3.AddRequirement(requirement2);

Requirement requirement3 = new Requirement();

requirement3.RequirementName = "Re2";

requirement3.RequirementDescription = "Re2 456";

project3.AddRequirement(requirement3);

Test test3 = new Test();

test3.testName = "Te3";

test3.testDescription = "Te3 123";

test3.AddRequirement(requirement3);

project3.AddTest(test3);

User user1 = new User();

user1.UserAccount = "admin";

user1.UserPassword = "admin";

user1.UserIdentity = "Manager";

user1.addProject(project1);

user1.addProject(project2);

user1.addProject(project3);

\_model.setProject(user1);

}

[TestMethod()]

public void addProjectTest()

{

\_model.addProject("test\_projectName", "test\_projectDescription");

Assert.AreEqual("test\_projectName",\_model.GetProjects()[3].ProjectName);

}

[TestMethod()]

public void editProjectTest()

{

\_model.editProject("test\_projectName1", "test\_projectDescription1",2);

Assert.AreEqual("test\_projectName1", \_model.GetProjects()[2].ProjectName);

Assert.AreEqual("test\_projectDescription1", \_model.GetProjects()[2].ProjectDescription);

}

[TestMethod()]

public void deleteProjectTest()

{

\_model.deleteProject(1);

Assert.AreEqual("Project3", \_model.GetProjects()[1].ProjectName);

}

[TestMethod()]

public void getRequirementFromSelectProjectTest()

{

Assert.AreEqual("Re1", \_model.getRequirementFromSelectProject(0)[0].RequirementName);

}

[TestMethod()]

public void getTestFromSelectProjectTest()

{

Assert.AreEqual("Te1", \_model.getTestFromSelectProject(0)[0].testName);

}

[TestMethod()]

public void addRequirementTest()

{

\_model.getRequirementFromSelectProject(0);

\_model.addRequirement("test\_reName", "test\_reDescription");

Assert.AreEqual("test\_reName", \_model.GetRequirement()[1].RequirementName);

Assert.AreEqual("test\_reDescription", \_model.GetRequirement()[1].RequirementDescription);

}

[TestMethod()]

public void editRequirementTest()

{

\_model.getRequirementFromSelectProject(0);

\_model.editRequirement("test\_reName", "test\_reDescription", 0);

Assert.AreEqual("test\_reName", \_model.GetRequirement()[0].RequirementName);

Assert.AreEqual("test\_reDescription", \_model.GetRequirement()[0].RequirementDescription);

}

[TestMethod()]

public void deleteRequirementTest()

{

\_model.getRequirementFromSelectProject(0);

\_model.getTestFromSelectProject(0);

\_model.deleteRequirement(0);

Assert.AreEqual(0, \_model.GetRequirement().Count);

}

[TestMethod()]

public void addTestTest()

{

Test test1 = new Test();

test1.testName = "t\_name";

test1.testDescription = "t\_description";

\_model.getTestFromSelectProject(0);

\_model.addTest(test1);

Assert.AreEqual("t\_name", \_model.getTestFromSelectProject(0)[2].testName);

Assert.AreEqual("t\_description", \_model.getTestFromSelectProject(0)[2].testDescription);

}

[TestMethod()]

public void LoginCheckTest()

{

Assert.IsTrue(\_model.LoginCheck("admin", "admin"));

}

[TestMethod()]

public void GetProjectsTest()

{

Assert.AreEqual(3, \_model.GetProjects().Count);

}

[TestMethod()]

public void GetRequirementTest()

{

\_model.getRequirementFromSelectProject(0);

Assert.AreEqual(1, \_model.GetRequirement().Count);

}

}

}

**ProjectTest.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;

private string \_projectName;

private string \_projectDescription;

private List<Requirement> \_requirements;

private List<Test> \_tests;

[TestInitialize()]

public void ProjectInit()

{

\_project = new Project();

\_requirements = new List<Requirement>();

\_tests = new List<Test>();

}

[TestMethod()]

public void AddRequirementTest()

{

Requirement re1 = new Requirement();

re1.RequirementName = "re1\_name";

re1.RequirementDescription = "re1\_description";

\_project.AddRequirement(re1);

Assert.AreEqual("re1\_name", \_project.GetRequirements()[0].RequirementName);

Assert.AreEqual("re1\_description", \_project.GetRequirements()[0].RequirementDescription);

}

[TestMethod()]

public void AddTestTest()

{

Test test1 = new Test();

test1.testName = "test\_name";

test1.testDescription = "test\_description";

\_project.AddTest(test1);

Assert.AreEqual("test\_name",\_project.GetTests()[0].testName);

Assert.AreEqual("test\_description", \_project.GetTests()[0].testDescription);

}

[TestMethod()]

public void GetRequirementsTest()

{

Requirement re1 = new Requirement();

re1.RequirementName = "re1\_name";

re1.RequirementDescription = "re1\_description";

\_project.AddRequirement(re1);

Assert.AreEqual(1, \_project.GetRequirements().Count);

}

[TestMethod()]

public void GetTestsTest()

{

Test test1 = new Test();

test1.testName = "test\_name";

test1.testDescription = "test\_description";

\_project.AddTest(test1);

Assert.AreEqual(1, \_project.GetTests().Count);

}

}

}

**TestTests.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 TestTests

{

List<Requirement> \_requirements;

Test \_test;

[TestInitialize()]

public void TestInit()

{

\_test = new Test();

\_requirements = new List<Requirement>();

}

[TestMethod()]

public void AddRequirementTest()

{

Requirement re1 = new Requirement();

re1.RequirementName = "re1\_name";

re1.RequirementDescription = "re\_description";

\_test.AddRequirement(re1);

Assert.AreEqual("re1\_name", \_test.requirements[0].RequirementName);

Assert.AreEqual("re\_description", \_test.requirements[0].RequirementDescription);

}

}

}

**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 UserTests

{

List<Project> \_inProjects;

User \_user;

[TestInitialize()]

public void TestInit()

{

\_user = new User();

\_inProjects = new List<Project>();

}

[TestMethod()]

public void addProjectTest()

{

Project pro = new Project();

pro.ProjectName = "pro\_name";

pro.ProjectDescription = "pro\_description";

\_user.addProject(pro);

Assert.AreEqual("pro\_name", \_user.GetInProjects()[0].ProjectName);

Assert.AreEqual("pro\_description", \_user.GetInProjects()[0].ProjectDescription);

}

[TestMethod()]

public void GetInProjectsTest()

{

Project pro = new Project();

pro.ProjectName = "pro\_name";

pro.ProjectDescription = "pro\_description";

\_user.addProject(pro);

Assert.AreEqual(1, \_user.GetInProjects().Count);

}

}

}

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 |