Software Testing and Verification

Tasks

Test Design Specification

資工碩一 107598010 陳郁欣

資工碩一 107598013 鄭鴻仁

資工碩一 107598020 林冠璋

資工碩一 107598025 陳巧宜

資工碩一 107598042 洪子軒

Version：2.00 Date: 2019/4/14

Version history

Version	Date	Author	Reviewer	Status	Description
1.00	19.3.26	Whole Team	Whole Team	Draft	Initial draft
2.00	19.4.14	Whole Team	Whole Team	final	Fix use cases

Table of contents

Test design specification identifier

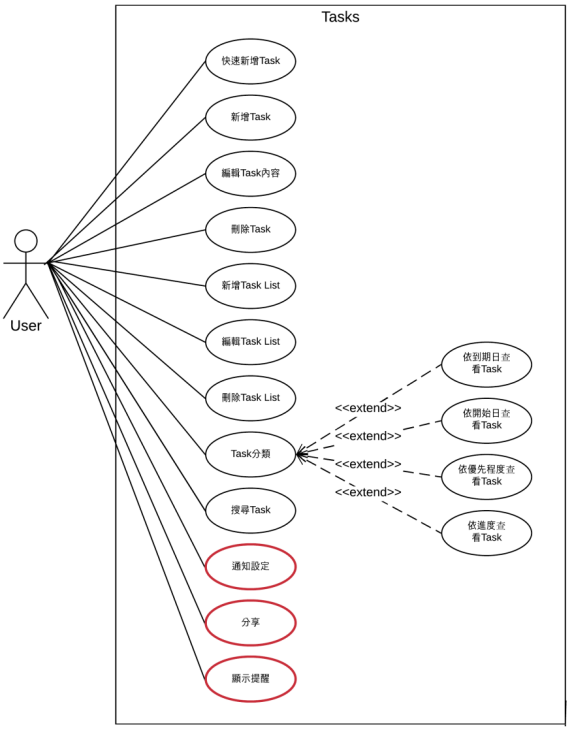
Identifier for document is **TASKS-TDS-001 1.00**.

This is document is associated with test plan **TASKS-STP-001**.

Features to be tested

The following use case diagram from [2] shows the basic

functionality of the program.



[2]

- Each use case will be tested by at least one test case and all test cases should be passed.

Feature	Priority
Create List	High
Delete List	Low
Update List	Low
Create Task(Quick_Add)	High
Create Task(Normal_Add)	High
Delete Task	High
Update Task	High
Search Task	Medium
Classification	Medium

The following combinations will be used:

{Create List、Create Task(Quick or Normal)}, {Create Task(Quick or Normal)、Delete Task}, {Create List、Delete List}, {Create Task(Quick or Normal)、Classisfication}, {Create Task(Quick or Normal)、Search Task}, {Create Task(Quick or Normal)、Update Task}, {Create List Delete List}, {Create List、Update List}

Approach refinements

The four levels of the V-model will be tested separately.

Level	Responsibility	Refinement	Identification
Acceptance testing	Client	Take the software to the client	Client sees, that the software is created according to demands.
System testing	Test group	Create new tests, run them	Manual tests cases produce visible result
Integration testing	Developer	Insert new module	System works visually and compiles, automated tests return clear result
Module testing	Developer	Create new module, test changed module	Module compiles and junit testing passes.

Test cases for phase 1 will be created using common sense and can be analyzed by anyone with experience in computer programming and java.

Results of the automated testing tools, Appium, will not be analyzed beyond bug reports and phase 1.

Testing identification

(TC01)Test Scenario : Quick add task	
User scenario	1. 使用者點擊 Task List。 2. 系統顯示+符號。 3. 點擊+符號，彈出快速新增視窗。 4. 使用者輸入 Title 並且按下“SAVE“按鈕。 5. 成功新增 Task。
Test Input	使用者輸入 Title 名稱。
Test Output	顯示新增的 Task，且 Task 名稱為使用者輸入內容。
Precondition	打開應用程式，Activity 為 MainActivity。
Postcondition	新增一個 Task。

(TC02)Test Scenario : Continuously quick add task	
User scenario	1. 使用者點擊 Task List。 2. 系統顯示+符號。 3. 點擊+符號，彈出快速新增視窗。 4. 使用者輸入 Title 並且按下“SAVE AND CONTINUE“按鈕。 5. 系統顯示 Task Created 訊息。 6. 重複步驟 4，直到不再新增時改按“Save“按鈕。
Test Input	輸入多個 Task 的 Title。
Test Output	顯示新增的三個 Task，名稱為依序輸入的內容。
Precondition	打開應用程式，Activity 為 MainActivity。
Postcondition	新增一個 Task。

(TC03)Test Scenario : Create a task	
User scenario	1. 使用者點擊右下角+號 Button。 2. 系統彈出視窗。 3. 使用者輸入 Task 內容並且按下確認。 4. 系統成功新增 Task。
Test Input	輸入 Task 內容。
Test Output	Task Summary Page 顯示輸入的內容。
Precondition	打開應用程式，Activity 為 MainActivity。
Postcondition	新增一個 Task。

(TC04)Test Scenario : Update title of task	
User scenario	1. 使用者點擊要更新的 Task。 2. 系統顯示 Task Summary 頁面。 3. 使用者點選編輯按鈕。 4. 系統顯示 Task 編輯頁面。 5. 使用者輸入要更新的欄位。 6. 使用者點擊 Save 按鈕。
Test Input	使用者針對選擇的 Task，要更新的內容。
Test Output	點選修改的 Task，系統把此 Task 修改成使用者輸入的更新內容。
Precondition	1. 打開應用程式，Activity 為 MainActivity。 2. List 裡面的 Task 不為空。
Postcondition	成功更新 Task 的 Title。

(TC05)Test Scenario : Delete task	
User scenario	1. 使用者點選 List，系統顯示此 List 裡的 Task。 2. 使用者點選要刪除的 Task，系統彈出視窗。 3. 使用者點選右上角第一個設定按鈕選擇刪除並且按下確定。 4. Task 被完整刪除。
Test Input	新增一個 Task 。
Test Output	系統顯示的 Task 裡，沒有被刪除的 Task。
Precondition	1. 打開應用程式，Activity 為 MainActivity。 2. List 裡面的 Task 不為空。
Postcondition	選擇的 Task 被完整刪除。

(TC06)Test Scenario : add List	
User scenario	1. 使用者點選右上角 menu 按鈕，系統顯示 menu 選項視窗。 2. 使用者點選 Displayed Lists，系統顯示 Displayed Lists 頁面。 3. 使用者點選右上角+按鈕，系統顯示 list name 輸入視窗。 4. 使用者輸入 List 名稱，並且按下確認按鈕。 5. 系統顯示新增的 list 名稱及 list 顏色。 6. 使用者點選儲存按鈕。 7. 系統成功新增 List。
Test Input	使用者輸入 List 名稱。
Test Output	系統新增新的 List，名稱為使用者輸入的內容。
Precondition	打開應用程式，Activity 為 MainActivity。
Postcondition	List 成功新增。

(TC07)Test Scenario : Delete list	
User scenario	1. 使用者點選右上角 menu 按鈕，系統顯示 menu 選項視窗。 2. 使用者點選 Displayed Lists，系統顯示 Displayed Lists 頁面。 3. 使用者點選要刪除的 List 右邊設定按鈕，系統顯示 list 名稱及顏色的預覽視窗。 4. 使用者點選 Delete List，系統顯示確認刪除對話框。 5. 使用者點選確認刪除按鈕。 6. 系統成功刪除 List。
Test Input	新增一個 List。
Test Output	系統顯示的 List，沒有被刪除的 List。
Precondition	1. 打開應用程式，Activity 為 MainActivity。 2. 應用程式 List 數量不為 0。
Postcondition	系統 List 的數量比原本少 1。

(TC08)Test Scenario : Refresh list name	
User scenario	1. 使用者點選右上角 menu 按鈕，系統顯示 menu 選項視窗。 2. 使用者點選 Displayed Lists，系統顯示 Displayed Lists 頁面。 3. 使用者點選要更新的 List 右邊設定按鈕，系統顯示 List 名稱及顏色的預覽視窗。 4. 使用者點選 Name，系統顯示 List name 修改視窗。 5. 使用者輸入 List 的新名稱，並且按下確定。 6. 系統顯示 List 名稱及顏色的預覽視窗。 7. 使用者點選 List Color，系統顯示 Color 選者視窗。 8. 使用者點選視窗下方的顏色組合，系統顯示顏色組合的選項。 9. 使用者點選顏色，系統顯示 List 名稱及顏色的預覽視窗。 10. 使用者點選儲存按鈕。 11. 系統成功更新 List 名稱及顏色。
Test Input	使用者輸入 List 要更新的名稱。
Test Output	對於更新的 List，系統顯示更新後的名稱。

Precondition	1. 打開應用程式，Activity 為 MainActivity。 2. 應用程式 List 數量不為 0。
Postcondition	List 名稱確實被修改。

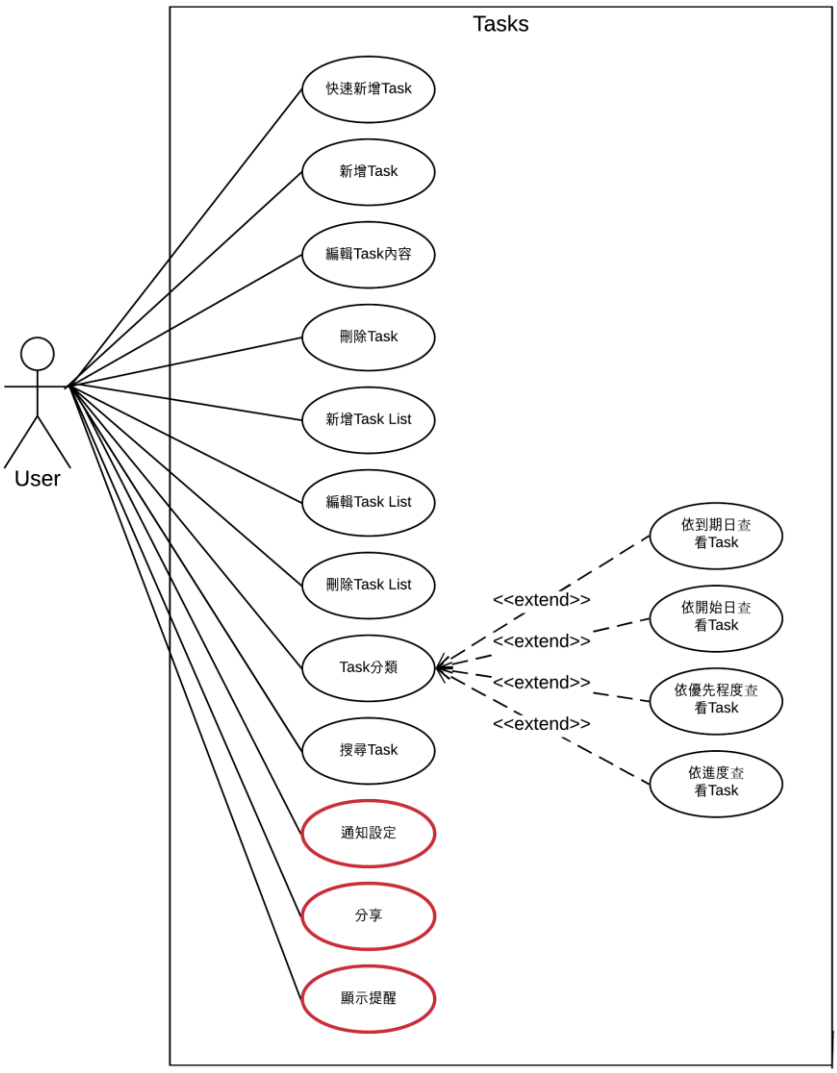
(TC09)Test Scenario : Edit task	
User scenario	1. 使用者點擊 List，系統顯示此 List 所包含的 Task。 2. 使用者點擊要修改的 Task，系統彈出視窗並顯示此 Task 之內容。 3. 使用者修改 Task 內容並且確認。 4. 系統成功修改 Task 內容。
Test Input	使用者輸入要編輯的 Task 內容。
Test Output	對於更新過的 Task，系統顯示修改後的內容。
Precondition	1. 打開應用程式，Activity 為 MainActivity。 2. 應用程式 Task 數量不為 0。
Postcondition	Task 成功被修改。

(TC10)Test Scenario : Search task	
User scenario	1. 使用者點選 TabList 最右邊的搜尋按鈕，系統彈出搜尋視窗。 2. 使用者輸入要搜尋之 Task 名稱並且按下確認。 3. 系統回傳搜尋之 Task。
Test Input	輸入要尋找的 Task 名稱。
Test Output	系統顯示尋找到的 Task。 系統顯示 My Task、系統無顯示任何 Task
Precondition	打開應用程式，Activity 為 MainActivity。
Postcondition	系統顯示搜尋之 Task。

(TC11)Test Scenario : Classification	
User scenario	1. 使用者點擊右下角新增按鈕。 2. 輸入 Task 分類屬性跟內容，並且按下儲存。 3. 使用者點選不同的分類，系統根據不同的分類顯示相對應的 Task。
Test Input	新增 Task 設定 Start、Due、Priority、Progress 屬性(不用每個屬性都要有值)。
Test Output	在相對應的分類可以看到依屬性新增的 Task。
Precondition	打開應用程式，Activity 為 MainActivity。
Postcondition	系統顯示依據的分類。

Feature pass/fail criteria

Each use case will be tested by at least one test case and all test cases should be passed.



References

[1] IEEE Standard for Software Test Documentation, IEEE Std 829, 09/1998