系統名稱

設計文件

|  |  |
| --- | --- |
| 專案名稱 | 專案名稱 |
| 撰寫日期 | 撰寫日期 |
| 發展者 | 發展者姓名 |

(紅字均為說明，都可刪除)

(藍字均為範例，都應刪除或可做適度修改，並改為黑色)

**版次變更記錄**

|  |  |  |
| --- | --- | --- |
| 版次 | 變更項目 | 變更日期 |
| 0.1 | 初版 | 2024/11/9 |
| 0.2 | 修正類別圖的stereotype  修正資料庫設計 | 2024/11/16 |
| 0.3 |  |  |
| 0.4 |  |  |
| 0.5 |  |  |
| 0.6 |  |  |
| 1.0 |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

目錄

[版次變更記錄 1](#_Toc178146372)

[1. 系統模型與架構(System Model/System Architecture) 1](#_Toc178146373)

[2. 介面需求與設計(Interface Requirement and Design) 1](#_Toc178146374)

[3. 流程設計(Process Design) 1](#_Toc178146375)

[4. 使用者畫面設計(User Interface Design) 1](#_Toc178146376)

[5. 資料設計(Data Design) 1](#_Toc178146377)

[6. 類別圖設計(Class Diagram) 1](#_Toc178146378)

[7. 實作方案(Implementation Languages and Platforms) 2](#_Toc178146379)

[8. 設計議題(Design Issue) 2](#_Toc178146380)

1. 系統模型與架構(System Model/System Architecture)

描述系統之架構。可用C4 model的Container diagram + Component diagram、UML之Component diagram或單純的Block diagram (方塊圖)表達此系統包含的模組，以及與模組之間的關係。

架構圖應與SRD一致，但可特別強調介面(Interface)與實際佈署之環境。

1. 介面需求與設計(Interface Requirement and Design)

根據架構圖，描述模組間的介面資訊，包含介面名稱、介面提供者、介面使用者、連結方式、輸入資料、輸出資料與介面描述等。

請特別留意介面提供者與介面使用者的意義，以前後端Web系統來說，API介面的介面使用者通常是前端某個模組，介面提供者通常是後端某個模組，連結方式應該是HTTP GET/POST/PUT(PATCH)/DELETE。

若想表達後端重要的共用模組，以Java為例，介面使用者若是後端某個模組，介面提供者應是後端另外一個模組，連結方式可能是Java Interface。

1. 流程設計(Process Design)

可用UML之Activity diagram / State Transition diagram或一般的程式流程圖描述所開發的系統流程，目的是運用流程圖或狀態圖設計整個系統的完整運作。

此部份應是操作概念的細部設計。

1. 使用者畫面設計(User Interface Design)

可用各種畫面設計方式，如發展網頁prototype、繪製PowerPoint投影片等，去設計你的主要系統畫面。此部份應是SRD中使用者介面分析的進階設計版本。

1. 資料設計(Data Design)

設計此系統的資料庫schema、檔案結構、XML schema、JSON (JavaScript Object Notation)物件等。

1. 類別圖設計(Class Diagram)

發展UML之class diagram (可對應C4 model中的Code diagram)，明確設計系統中包含哪些類別，以及這些類別之間的關係。

若遇到非一般物件類別或特殊物件類別，如HTML、JavaScript、JSP、Servlet等，可用stereotype表達，如<<JavaScript>>、<<Servlet>>。

可把重點放在定義程式結構：訂出所有類別名稱、拉出所有類別關係，再加入重要的attibute/operation即可。而特殊的類別職責設計可加上文字說明。

1. 實作方案(Implementation Languages and Platforms)

說明系統之平台(如網站或行動App)。

說明預計採用的程式語言或技術(如後端Java、node.js、PHP等，前端JS+Boostrap、Android、iOS等)。

說明預計採用的框架(Framework，如前端React、Angular、Vue，後端Spring、SpringBoot等)、函式庫(如前端jQuery、d3.js，後端jsoup、iText)、Servless服務(Firebase)等。

說明部署方式 (雲端IaaS、雲端PaaS或自架伺服器但提供外部存取)。

1. 設計議題(Design Issue)

將設計此系統過程中遭遇的議題紀錄下來，包含(1)議題內容、(2)可能解決方案(alternative solutions)、(3)最後解決方案(final solution)與理由(rationale)。