Software Architecture Document
for
Virtual Room Reservation Assistant

Version 1.0 approved

Prepared by

林琛琛 (B10832019)、鄧宥均 (B10832042)、郭垂蓁 (B10832031)、張瑄容 (B10830008)、莊博智 (B10830226)

2021/12/11

Table Of Contents

Revision History

Name	Date	Reason For Change	Version

1. Introduction

1.1 Purpose

這份文件提供了 Virtual Room Reservation Assistant 的全面架構。使用不同的結構圖描述此系統的不同面向，表示系統上的重要決策。

1.2 Scope

此份文件包含了 4+1 view model，分別為 Use-Case View、Logical View、Process View、Deployment View、Implementation View 五個 model 描述 Virtual Room Reservation Assistant 的結構。

1.3 Definition, Acronyms, and Abbreviations

名詞	定義
database	存放資料的地方，使用者可以存取或修改
localhost	一個在電腦網路中用於表示「此電腦」的主機名。
User Interface	使用者和系統交互的介面
google API	和 google 服務溝通的橋樑

1.4 References

- Flask: https://flask.palletsprojects.com/en/2.0.x/tutorial/index.html
- Flask-Login: https://flask-login.readthedocs.io/en/latest/
- SQLite: https://www.sqlite.org/index.html
- Bootstrap: https://getbootstrap.com/

1.5 Overview

這份文件一開始先介紹了 Virtual Room Reservation Assistant 的功能及相關規範，後面則使用五個 model，分別為別為 Use-Case View、Logical View、Process View、Deployment View、Implementation View 五個 model 描述 Virtual Room Reservation Assistant 的結構，最後則探討此系統的特質。

2. Architectural Representation

- Use-case view:描述系統的核心功能，呈現使用者與系統之間的交互的關係。
- Logical view:將整個系統以功能與需求為基準拆解成許多組件。
- Process view:展示非功能性需求，注重並行、分散式處理。
- Deployment view: 將軟體映射到所涉及的硬體的區塊或節點。
- Implementation view: 將系統分成多個子系統，用層級表示子系統間的交互關係。

3. Requirements Specifications

3.1 Performance

若要進入此會議室預約系統，需要使用 PC 或是手機以網頁瀏覽的方式連線到我們的系統，即可以使用該系統所提供的功能，包括：查看目前會議室預約的情況、個人的會議、建立會議等等的服務。

3.2 Safety

1. 此系統只會讓會議參與者看見該會議的資訊，非參與者無法存取會議資訊。
2. 只有會議建立者可以編輯會議，他人無法修改。

3.3 Security

1. 用戶必須輸入正確的帳號密碼才能登入存取會議資訊
2. 此系統並不會將使用者的 email 及會議資訊作為非法用途。

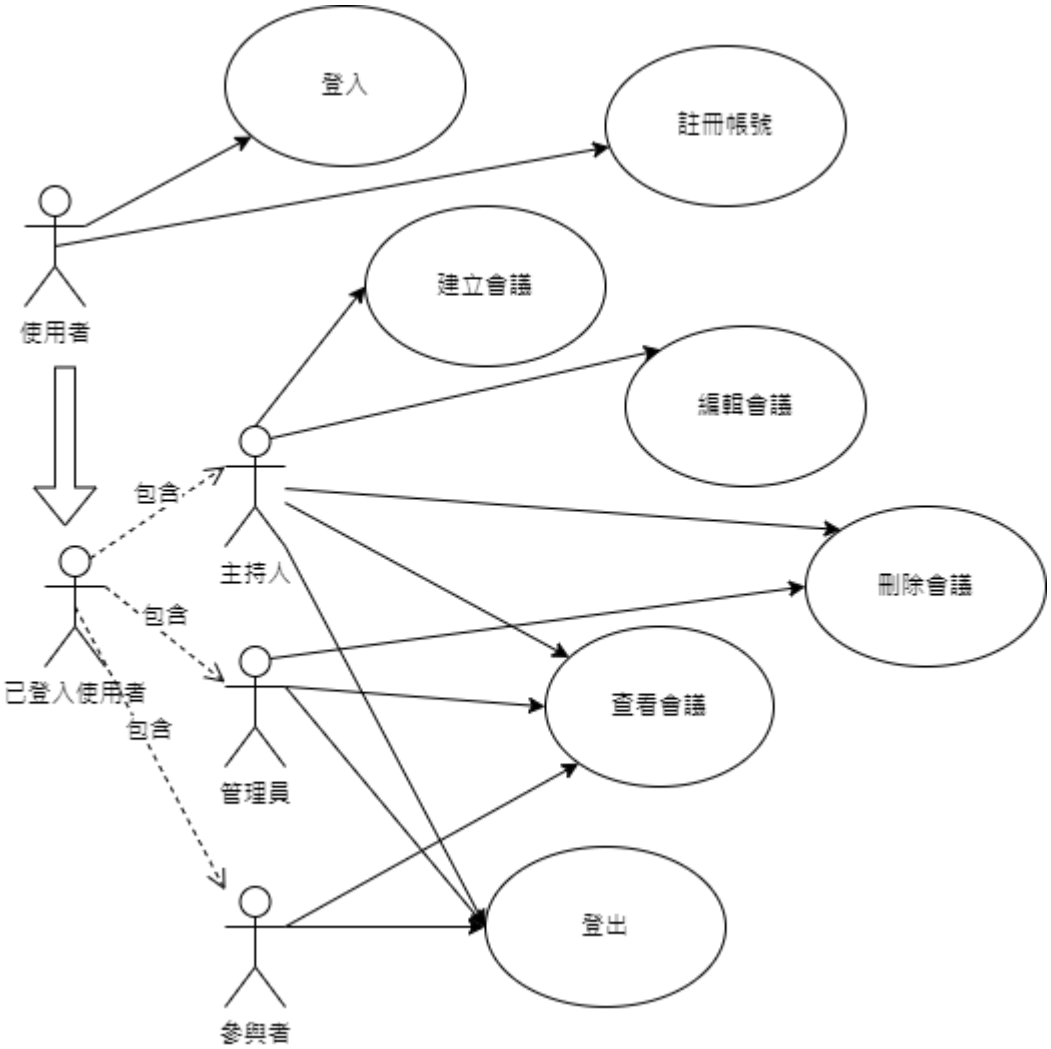
3.4 Privacy

密碼經過加密後儲存在資料庫中

3.5 Use of an Off-the-shelf Product

使用 gmail 作為寄信通知參與者的信箱

4. Use-Case View



4.1 Use-Case Realizations

4.1.1 註冊帳號

填寫 email、使用者名稱及密碼進行註冊。

4.1.2 登入

填寫正確帳號密碼登入系統，以使用此系統服務。

4.1.3 建立會議

填寫會議名稱、參與者、房間及時間建立會議。

4.1.4 編輯會議

修改會議名稱、參與者、房間及時間。

4.1.5 查看會議

查看會議名稱、參與者、房間及時間

4.1.6 刪除會議

將會議從系統上移除

4.1.7 登出

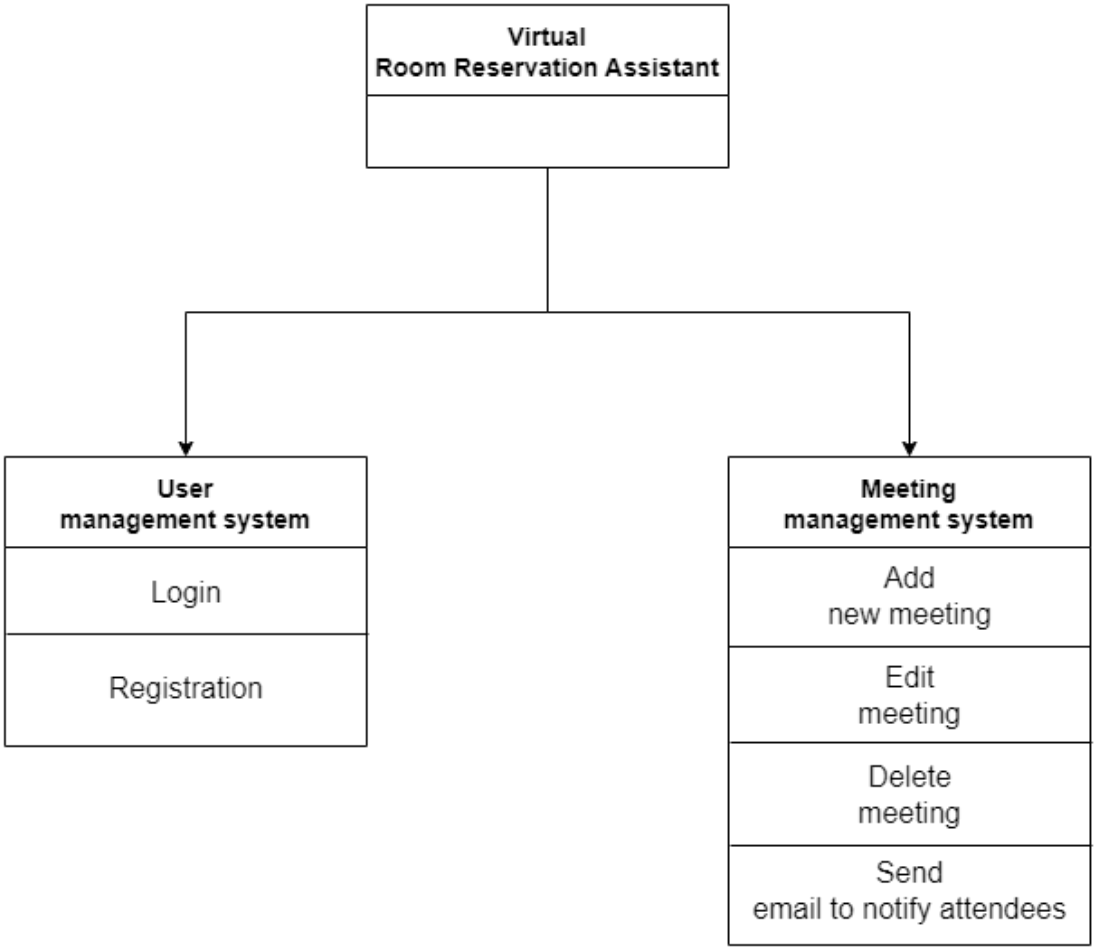
使用者無法使用此系統

5. Logical View

此部分將會將 Virtual Room Reservation Assistant 分解成兩個 subsystem，描述 subsystem 中重要的 class 之功能，以架構圖解析 subsystem 的功能，並在此文件的第 8 部分 Implementation View，詳細介紹系統的 Layer 及其功能。

5.1 Overview

Virtual Room Reservation Assistant 分成兩個 subsystem，第一個是 User management system，第二個是 Meeting management system。



5.1.1 User management system

User management system 負責以下功能：

Login	使用者必須先用 email 註冊過後，才能登入系統。
Registration	新用戶必須使用 email 註冊。

5.1.2 Meeting management system

Meeting management system 包含下列功能：

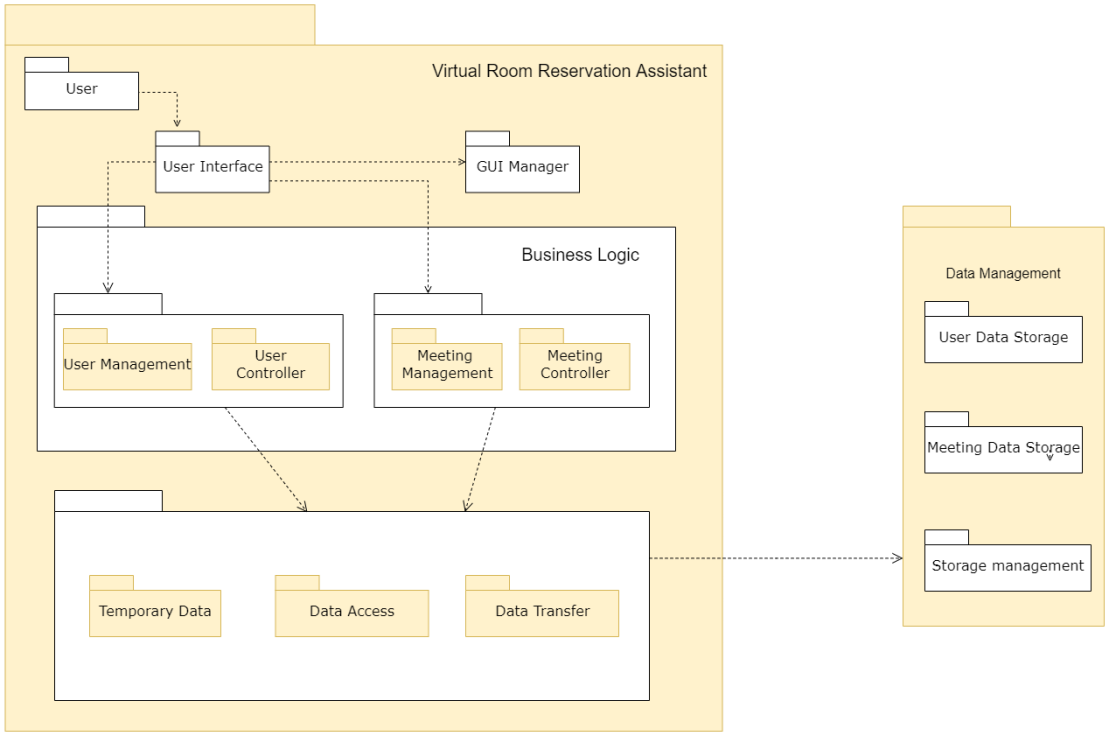
Add new meeting	已登入的使用者可以建立新會議，新會議的時間、房號不能與已建立的會議有衝突，且會議的開始時間不能比結束時間晚。
Edit meeting	會議建立者有權利按下 edit 按鈕編

	輯會議，修改會議後會再通知會議參加者。
Delete meeting	會議建立者、系統管理員有權利刪除會議，刪除會議後會通知會議參加者。
Send email to notify attendees	當會議建立後，或被修改、刪除時，將會寄信給會議參與者。

5.2 Architecturally Significant Design Packages

5.2.1 package diagram

此圖是 package diagram，User 與 UI 互動，UI 由 GUI Manager 管理，並做出適當的響應，當 User 有需要傳輸資料時，像是登入、註冊、建立會議時，會由 Business Logic 檢查邏輯正確性，如果沒有問題後，再向下傳到電腦暫存 Data 的記憶體，最後送到 Data management，存放在 SQLite 資料庫中。

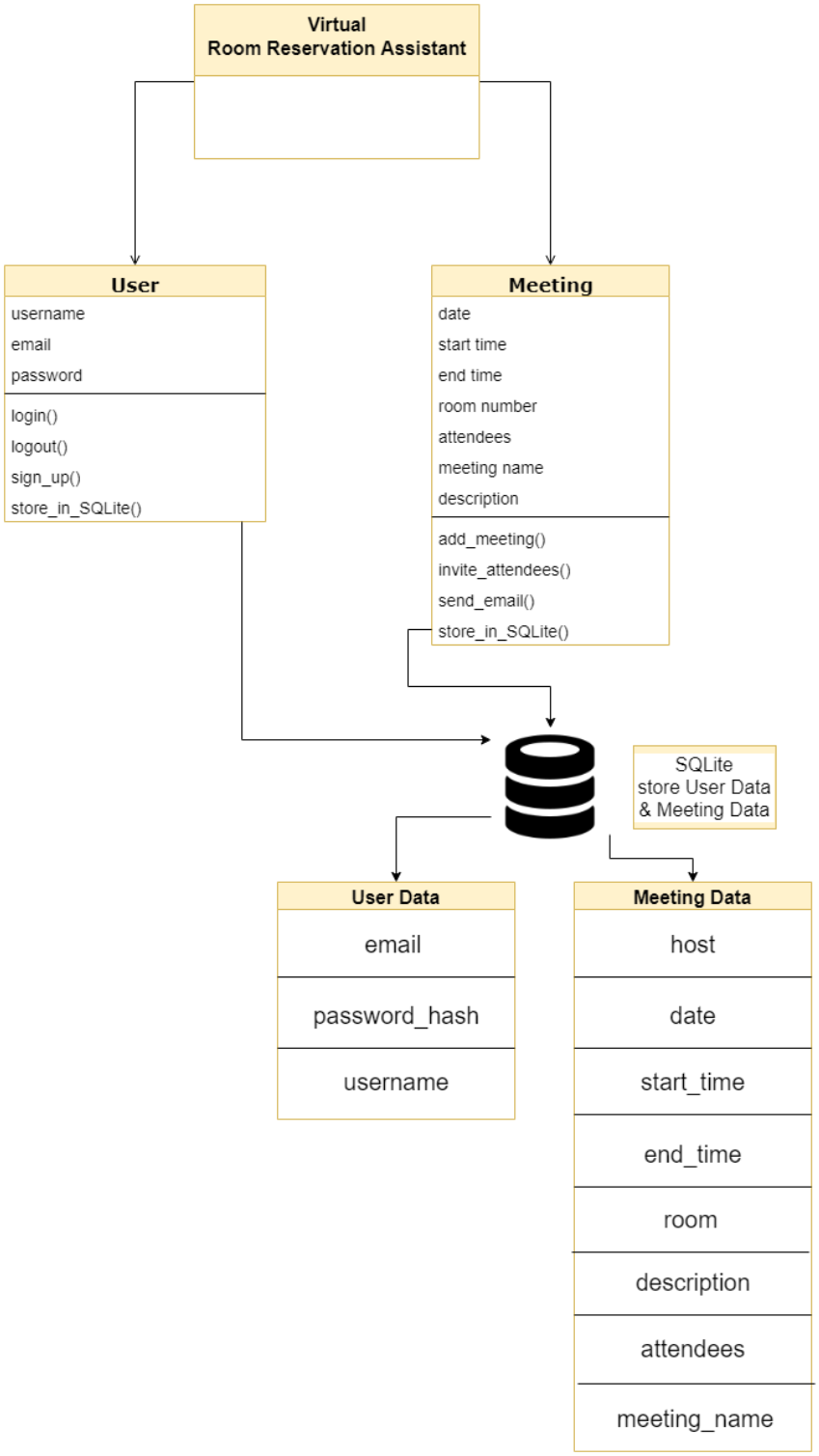


5.2.2 class diagram

此圖是 class diagram，Virtual Room Reservation Assistant 將系統分成兩部分，為 User 和 Meeting，每一個 class 的上半部是參數，下半部是 method，當資料被整理好後就會存入資料庫當中，資料庫會存在兩個 form，一個是 User Data，一個是 Meeting Data，存在於資料庫的變數都如圖中表示。

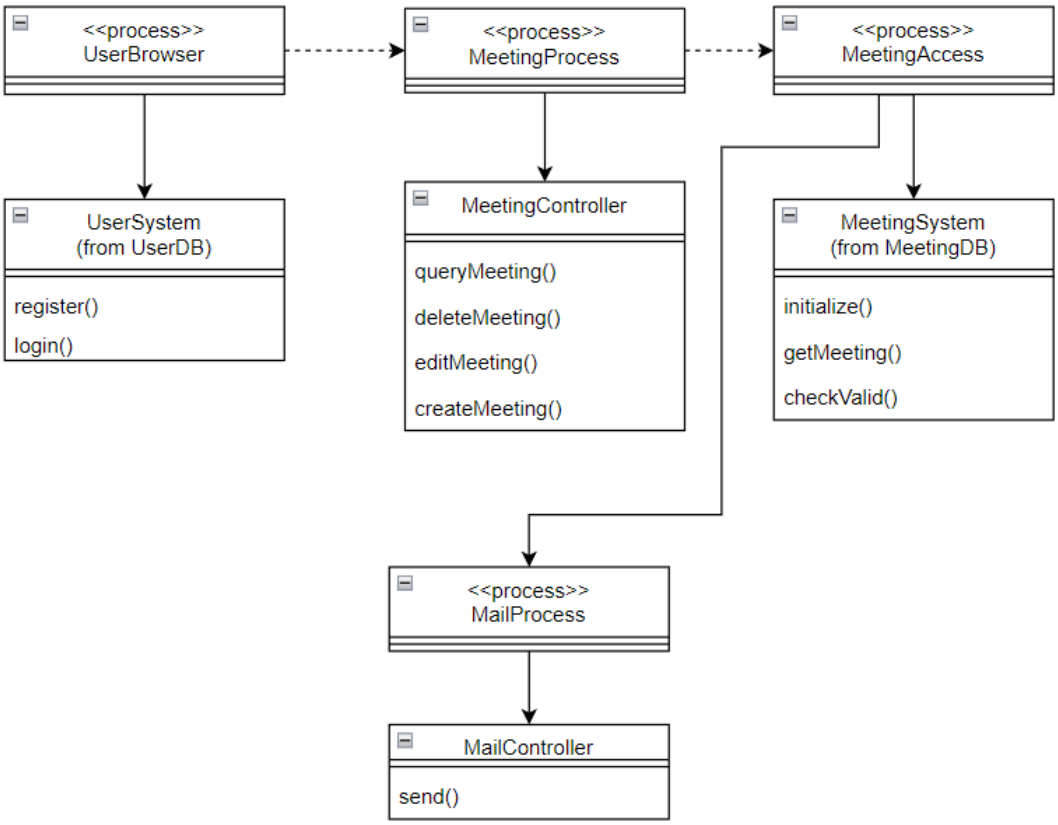
User class 參數	1. username 存使用者名稱 2. email 存使用者信箱 3. password 存使用者登入密碼
User class Method	1. login() 2. logout() 3. sign_up() 4. store_in_SQLite()
Meeting class 參數	1. date 存會議日期 2. start time 存會議起始時間 3. end time 存會議結束時間 4. room number 存會議室代碼 5. attendees 存會議參加者 6. meeting name 存會議名稱 7. description 存會議詳細說明
Meeting class Method	1. add_meeting() 2. invite_attendees() 3. send_email() 4. store_in_SQLite()
User Data Attribute	1. email 存使用者註冊的信箱 2. password_hash 存使用者的密碼，只存放加密過後的密碼

	3. username 存使用者的名稱
Meeting Data Attribute	1. host 會議建立者 2. date 存會議日期 3. start time 存會議起始時間 4. end time 存會議結束時間 5. room 存會議室代碼 6. description 存會議詳細說明 7. attendees 存會議參加者 8. meeting name 存會議名稱



6. Process View

6.1 Processes



6.1.1 UserBrowser

使用者透過瀏覽器與本系統互動，以進一步取得會議室預約系統的服務。

6.1.2 UserSystem

負責管理註冊使用者的資料，以及檢驗登入、註冊請求的合法性。

6.1.3 MeetingProcess

負責會議相關的功能如:新增、刪除會議、處理來自 UserBrowser 的需求並且存取 MeetingDB 的會議資訊。

6.1.4 MeetingController

針對 MeetingProcess 需要的操作，進行相對應的處理。

6.1.5 MeetingAccess

負責會議資料的取得、更新以及初始化，處理來自 MeetingProcess 的要求。

6.1.6 MeetingSystem

針對 MeetingAccess 需要的操作，進行相對應的處理。

6.1.7 MailProcess

負責當有會議新增、編輯與刪除時，發送 email 通知會議的所有參加者。

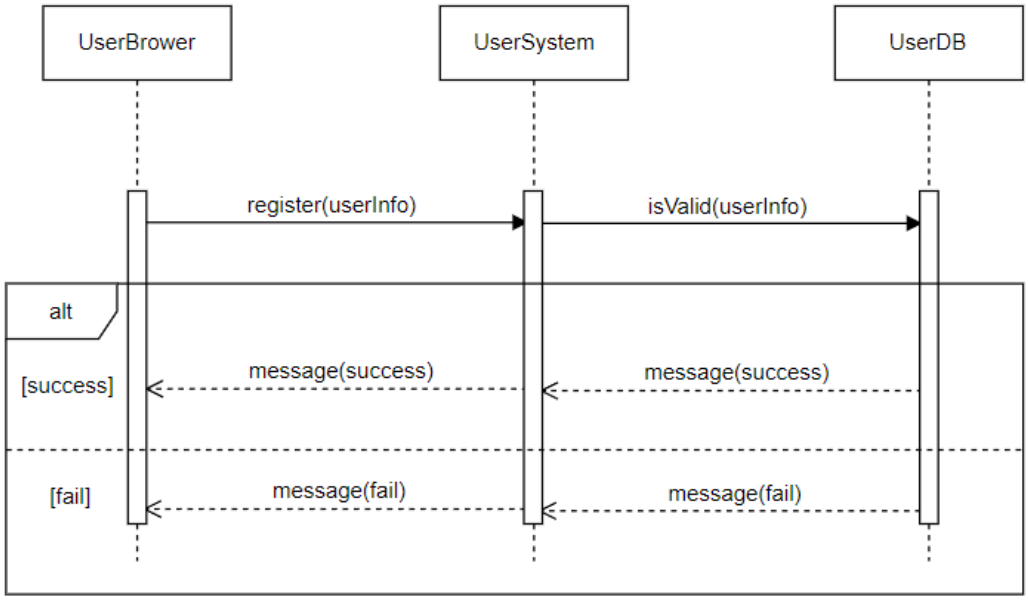
6.1.8 MailController

針對 MailProcess 需要的操作，進行相對應的處理。

6.2 System sequence diagrams

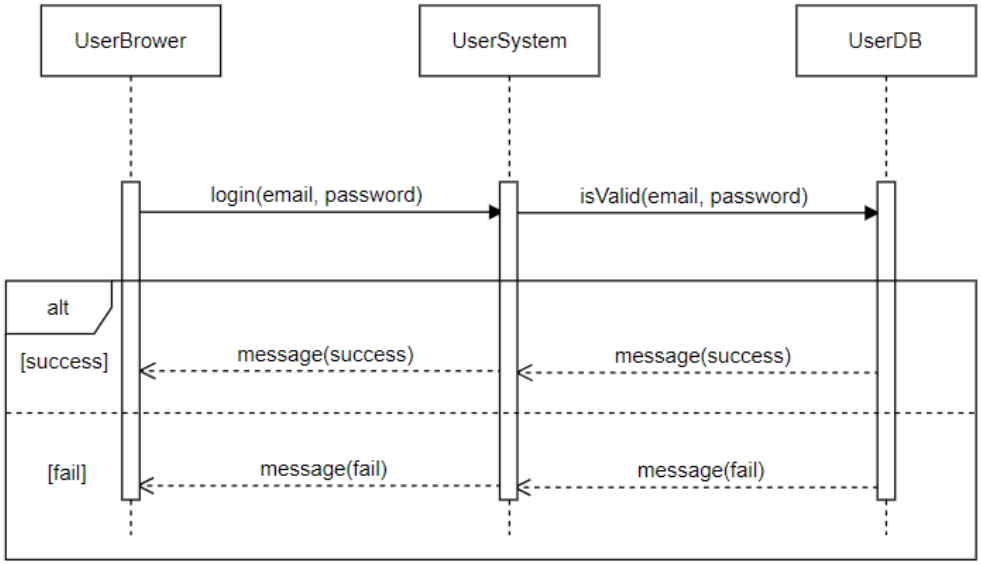
6.2.1 User Register

當使用者透過瀏覽器送出註冊請求後，本系統會檢查該筆使用者資料是否合法。若合法，則更新資料庫並回傳成功訊息「已成功註冊本系統會員」；若不合法如:重複註冊，則不受理此請求並回傳錯誤訊息「該 Email 已被註冊」。



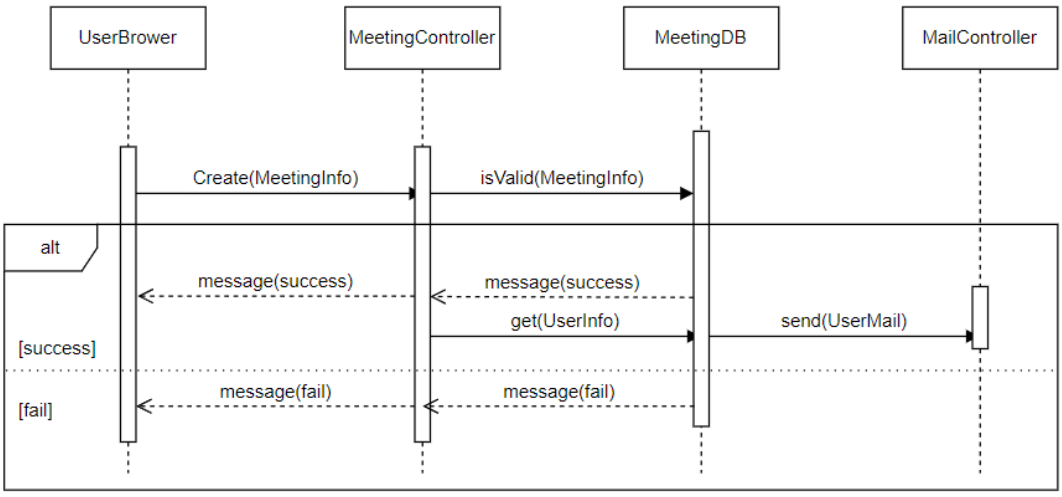
6.2.2 User Login

當使用者透過瀏覽器送出登入請求後，本系統會檢查該筆使用者資料是否合法。若合法，則更新當前狀態並回傳成功訊息「成功登入系統」；若不合法如:密碼錯誤、使用者不存在，則不受理此請求並回傳錯誤訊息「錯誤的 Email 或 Password」。



6.2.3 Create Meeting

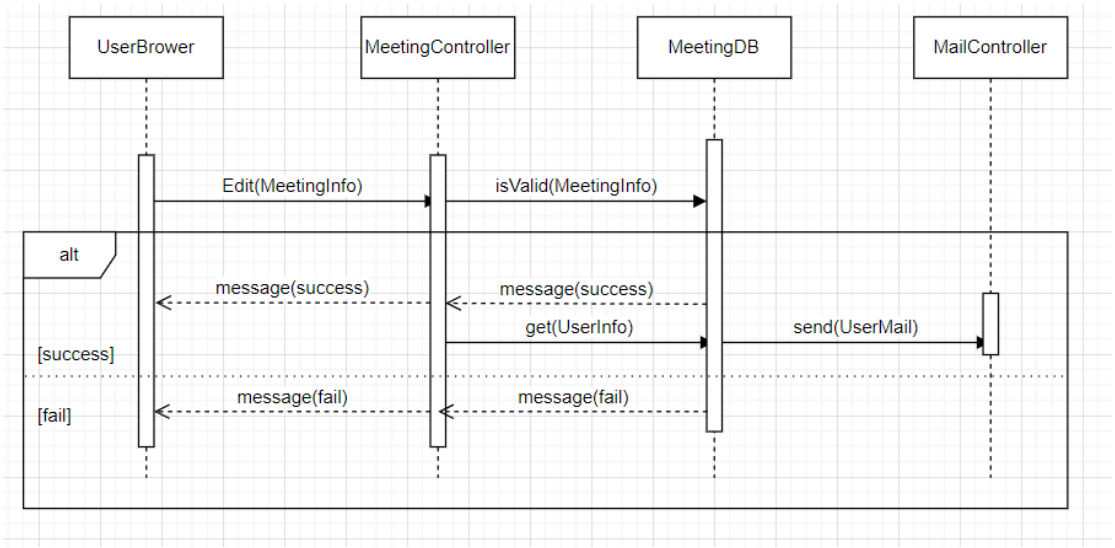
當使用者透過瀏覽器送出新增會議請求後，本系統會檢查該筆會議資料是否合法。若合法，則更新資料庫、發送通知信件並回傳成功訊息「新增會議成功」；若不合法如:新增會議與原有會議時間衝突，則不受理此請求並回傳錯誤訊息，如:「預約時間錯誤」。



6.2.4 Edit Meeting

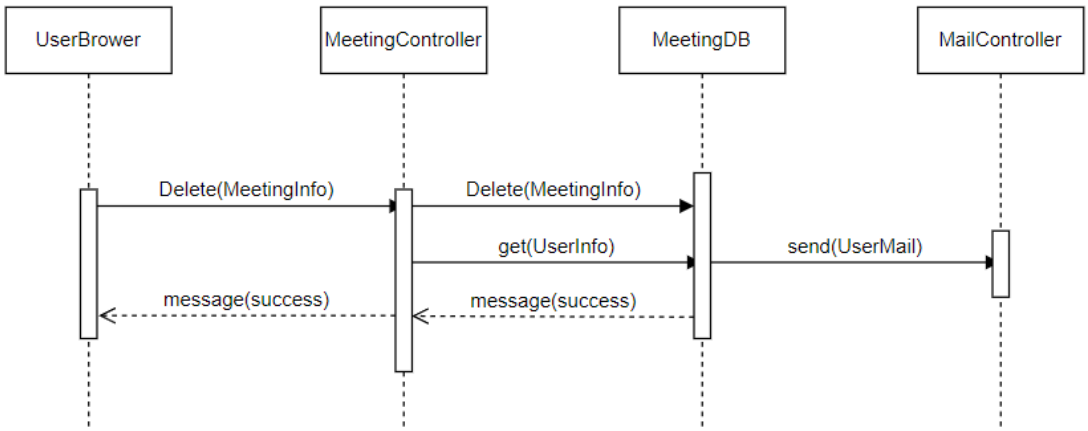
當使用者透過瀏覽器送出編輯會議請求後，本系統會檢查該筆會議資料是否合法。若合法，則更新資料庫、發送通知信件並回傳成功訊息「編輯會議成功」；若不合法如:使用者不存在，則不受理此請求並回傳錯誤訊息

「預約時間錯誤」。



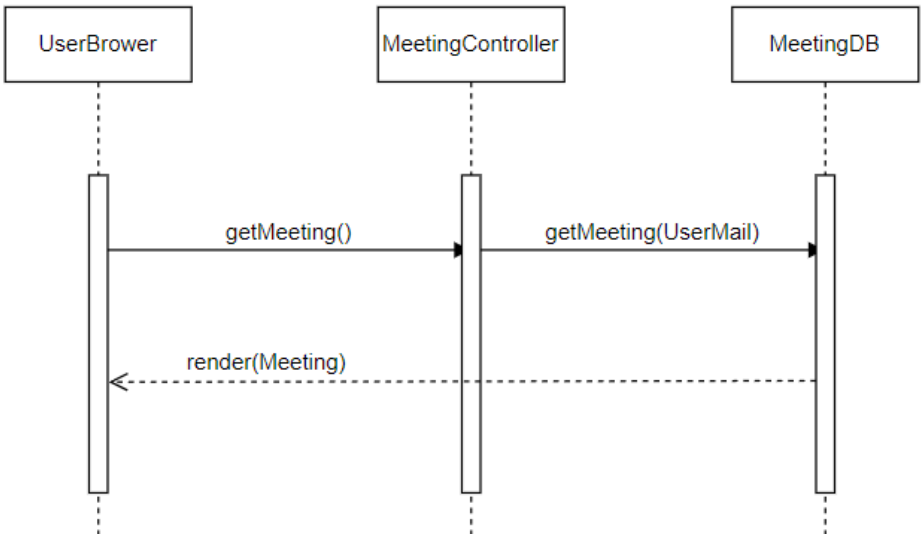
6.2.5 Delete Meeting

當使用者透過瀏覽器送出新增會議請求後，本系統會發送信件通知所有與會者，並且刪除資料庫中該筆會議資料，回傳成功訊息「刪除會議成功」。



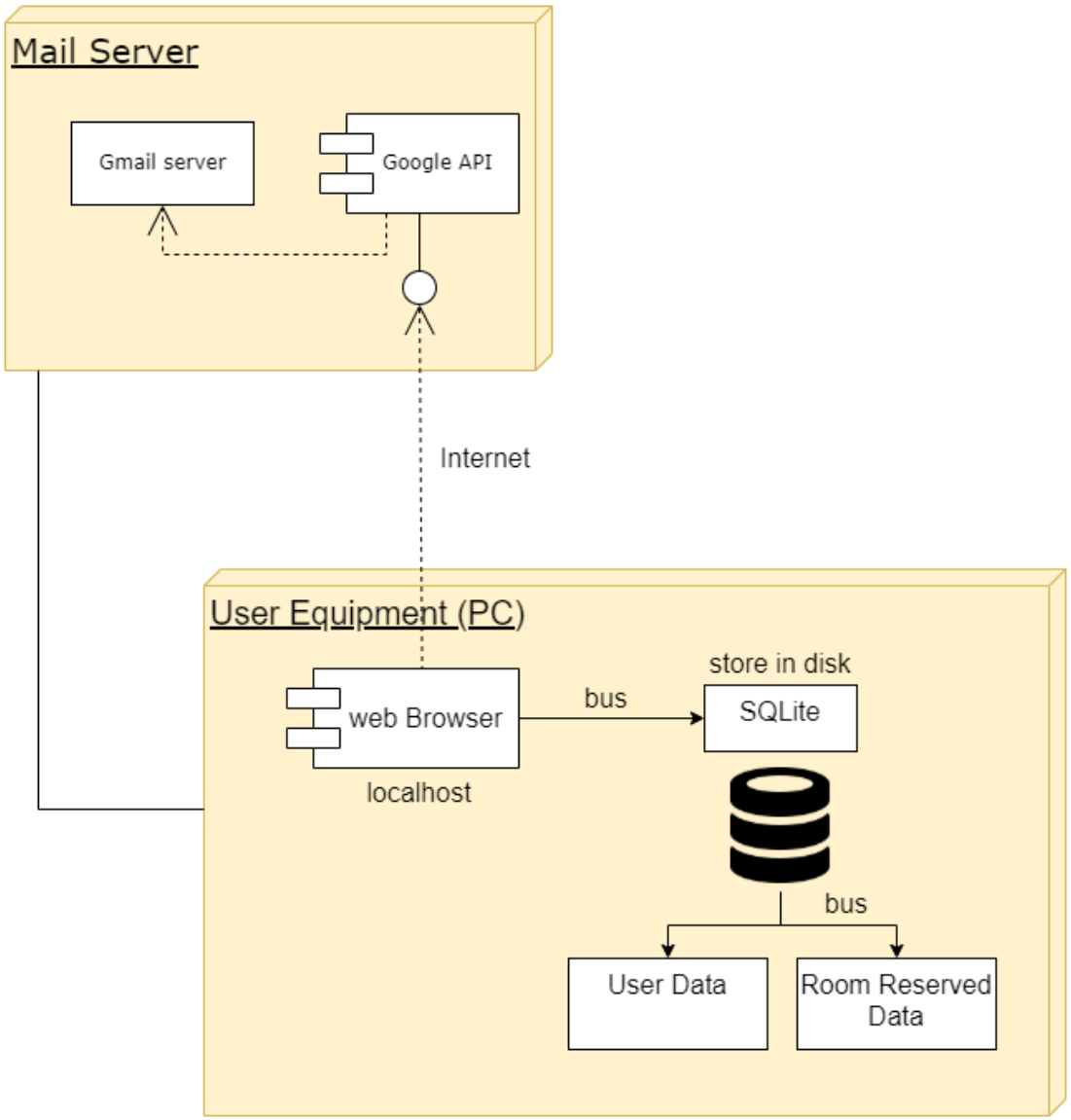
6.2.6 My Meeting

當使用者透過瀏覽器瀏覽「My Meeting Page」頁面，本系統會取得當下登入使用者的資料，從資料庫取得使用者參加的所有會議，並顯示在頁面上。



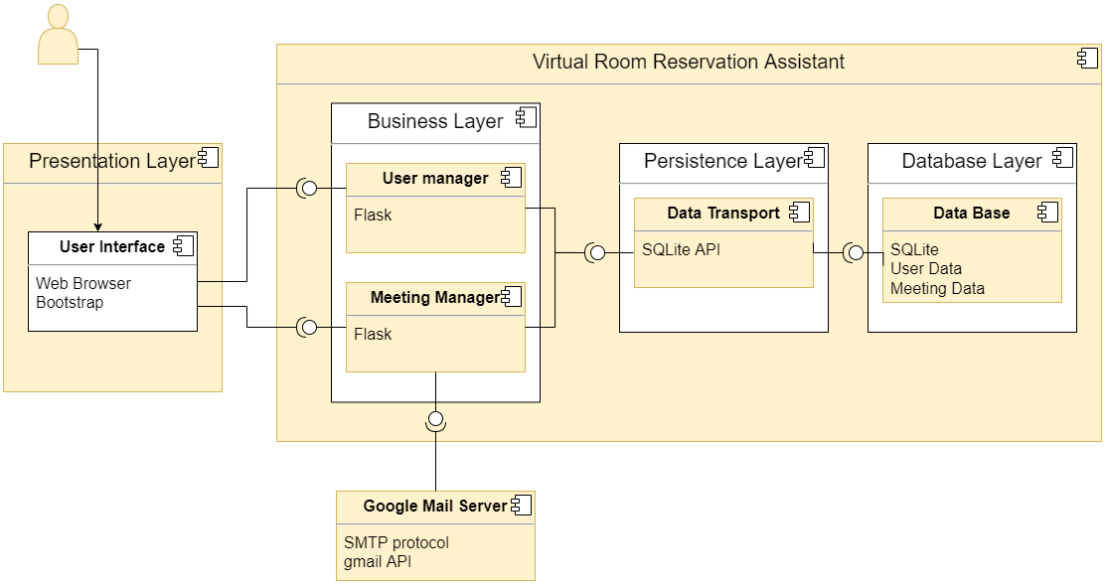
7. Deployment View

此部分描述系統會用到的所有硬體設備，以及硬體設備的功能和彼此之間溝通的方式。主要運行系統的是使用者的個人電腦（PC），server 的電腦以 http://0.0.0.0:5000/為 IP 執行 User Interface，後端的連接是在 server 電腦的硬碟裡架設資料庫，前端後端利用匯流排 bus 連接，使用者可以在瀏覽器中直接打上 server 的 IP 連上伺服器，資料庫統一由 server 管理。由於會議需要寄 email 提醒其他參加者，因此我們也會藉由 internet 連接 google API，再連接上 google 的 mail server，藉由 SMTP protocol 傳送信件給會議參與者。下圖為 server 架構。



8. Implementation View

以 Model-View-Controller 架構延伸，為系統分層，將分成 Presentation Layer，Business Layer，Persistence Layer，Database Layer。使用分層架構的優勢是更改特定一層的內容時不需要修改整個系統，每一層不互相影響，分層也有助於後續的測試，能較快找到問題。



8.1 Overview

系統分成 Presentation Layer(view)，Business Layer(controller)，Persistence Layer(model)，Database Layer。Presentation Layer 也是 Model-View-Controller 架構中的 view，負責與使用者互動；Business Layer 是 controller，負責系統中的邏輯檢查；Persistence Layer 讓軟體去存取資料庫中的資料；Database Layer 在此系統是 SQLite，負責存資料。

8.2 Layers

Presentation Layer 內的 user interface	系統使用 web browser 作為 user interface，我們的頁面使用 bootstrap 套件，讓工具、按鍵更為美觀。
Business Layer 內的 User manager	若使用者需要申請帳號、登入、登出時，我們會在 User manager 檢查邏輯問題，包含使用者是否有權限登入；註冊帳號時 password 與 confirm password 必須相同；使用者是否已經用相同信箱註冊過了。
Business Layer 內的 meeting manager	與會議相關的邏輯都會在這部分檢

	查，如：會議起始時間不得晚於結束時間；兩個會議若在相同時間舉行，使用的地點不得衝突；使用者是否為會議建立者，有權力可以編輯、刪除會議。
Persistence Layer	當使用者的行為 trigger 需要讀取或存資料時，Persistence Layer 負責讀取和存 Database 的資料。
Database Layer	Database 中有兩個表格，分別存 User 的資料和 meeting 的資料。
Google Mail Server	這部分使用 Google API 和 SMTP protocol 傳送資料，因為是使用外部的系統，因此獨立於 Virtual Room Reservation Assistant module。

9. Size and Performance

9.1 Size

- 本會議室預約系統約為 800KB，具體大小視系統版本、資料庫大小而有不同。
- 本系統共包含五間會議室，可預約時段自上午九點至下午六點。
- 本系統的註冊使用者數量上限為三百人。
- 單一會議的參加者上限為三十人。
- 單一使用者的參加會議數目上限為一百場，若超過此上限， 則最早的會議將無法顯示。

9.2 Performance

- 會議相關操作如:新增、編輯、刪除等功能，應在點擊按鈕後五秒內完成。
- 使用者登入、註冊帳號應在點擊按鈕後五秒內完成。
- 通知信件應在一分鐘內送達使用者信箱。
- 本系統效能會受裝置的硬體、網路狀況影響。

10. Quality

10.1 Extensibility

本會議室預約系統使用 Python 的 Flask 框架進行開發，透過 Model-View-Controller(MVC) 的設計理念將程式切割為不同的邏輯區塊，以此維持程式的簡潔性與增進程式的可擴充彈性(Extensibility)，使得未來能夠靈活的升級功能、修復錯誤。

10.2 Reliability

本系統每次更動程式與資料庫前皆事先經過審慎評估，以確保程式在正常的環境下得以如常運行，不會發生資料遺失、操作異常等非預期事件。

10.3 Portability

如同 10.1 所述，由於程式有妥善的架構與分層，只要符合系統限制，本系統能夠在不同的環境下提供服務。

10.4 Security

為確保使用者資料的安全性，本系統在使用者註冊時即經過加密才存入資料庫。此外，會議的編輯、刪除皆須由會議建立者或者系統管理員操作，保證使用者資料與會議資料無法輕易洩漏、竄改。