

# **Software Architecture Document**

Virtual Room Reservation Assistant

**<version 1.0>**

**DECEMBER 15 , 2021**

**Group:25**

**Author:**

B10704101 王柏鈞

B10815043 鄭鴻翔

B10815052 洪翊方

B10832030 陳政亮

**Submitted in partial fulfillment  
Of the requirements of  
CS3025301 Software Engineering**

<b>1. Introduction</b>	<b>3</b>
1.1 Purpose	3
1.2 Scope	4
1.3 Definition and Acronyms	4
1.3.1 Definition:	4
1.3.2 Acronyms :	4
1.4 References	4
1.5 Overview	4
<b>2. Architectural Representation</b>	<b>5</b>
2.1 Use-case View	5
2.2 Logical View	5
2.3 Process View	5
2.4 Development View	5
2.5 Implementation View	6
<b>3. Architectural Goals and Constraints</b>	<b>6</b>
3.1 Safety	6
3.2 Security	6
3.3 Privacy	6
3.4 Portability	6
3.5 Performance	6
3.6 Reuse	6
3.7 Development Tool	6
<b>4. Use Case View</b>	<b>7</b>
4.1 Use Cases diagram	7
4.1.1 註冊:	7
4.1.2 登入:	8
4.1.3 更新資料:	9
4.1.4 查看會議:	9
4.1.5 預約會議室:	10
4.1.6 取消預約:	11
4.1.7 修改會議資訊:	12
<b>5. Logical View</b>	<b>13</b>
5.1 Overview	13
5.1.1 Subsystems	13
5.1.2 System Layer Package Diagram	13
5.2 Architectural Significant Design Packages	14
5.2.1 Class diagram	14
<b>6. Process View</b>	<b>15</b>
6.1 User activity diagram	15
6.1.1 註冊	15

6.1.2 登入	16
6.1.3 更新	16
6.1.4 查看預約的會議室	17
6.1.5 查看參與的會議室	17
6.1.6 預約會議室	18
6.1.7 取消會議	19
6.1.8 修改會議資訊	20
6.2 System sequence diagram	21
6.2.1 註冊	21
6.2.2 登入	22
6.2.3 更新	22
6.2.4 查看預約或參與的會議室	23
6.2.5 預約會議室	23
6.2.6 取消會議	24
6.2.7 修改會議資訊	24
<b>7. Deployment View</b>	<b>25</b>
<b>8. Implementation View</b>	<b>25</b>
8.1 Overview	25
8.2 Layers	26
<b>9. Data View</b>	<b>27</b>
9.1 Local data view	27
9.1.1 Local Data Entity Diagram	27
9.1.2 Local Data Implementation	27
9.2 SQL view	27
9.2.1 MySql Entity Diagram	27
9.2.2 MySql implementation	28
<b>10. Size and Performance</b>	<b>29</b>
<b>11. Quality</b>	<b>29</b>

# 1. Introduction

## 1.1 Purpose

本文件目的是利用不同種類的架構圖來完整描述Virtual Room Reservation Assistant系統的不同面向，並將專案中的架構及功能以UML圖表表達，使讀者能以較為直觀的方式理解本應用軟體的實作及應用。

## 1.2 Scope

本文件所牽涉的範圍包括三個層次：使用者應用、軟體實作、系統配置。使用者應用包含使用案例、流程；軟體實作包含系統設計架構、資料結構；系統配置包含實際硬體配置。

## 1.3 Definition and Acronyms

### 1.3.1 Definition:

未註冊人：未經過註冊手續的人。

使用者：已經過註冊手續並成功的人。

受邀請人：必須是使用者。受邀請使用會議室的人。

發起人：必須是使用者。成功申請會議室使用的人。

參與人：必須是使用者。受發起人邀請，

使用會議室的人。

資料庫：存放帳密資訊、會議室資訊資料儲存的外部系統，由MySQL實作。

伺服器：外部伺服器，向Azure租借的服務。

### 1.3.2 Acronyms :

API: Application Programming Interface

UI :User Interface

## 1.4 References

Artifact\_Software Architecture Document

EN-Annex\_TS1.2\_Software+Architecture+Document

SAD\_DTCP II\_ver1.4

Sample Software Architecture Document

## 1.5 Overview

此文件參照"4+1 view model"來撰寫，分別包含Use-case , Logical, Process, Deployment, Data View，此外也包含系統大小與性能和品質。

## 2. Architectural Representation

此文件所使用的架構為"4+1 view model", 分別為Use-case、Logical、Process、Development 、Implementation, 藉由分別依據不同層面;使用適當的表示方法, 使不同身分的人了解這份專案的實現方法及目的。

### 2.1 Use-case View

適用讀者:

所有利益相關者, 包括使用者

範圍:

描述系統使用場景、提供給使用者的重要功能, 及簡述實現方法。

### 2.2 Logical View

適用讀者:

應用程式設計者、程式設計師、測試人員

範圍:

描述系統的物件設計;包含重要函式, 屬性, 系統與子系統間的互動關係。

### 2.3 Process View

適用讀者:

程式設計師、使用者

範圍:

描述系統與使用者、系統與資料庫之間的執行流程。系統與使用者之間的關係用activity diagram表達;系統與資料庫之間的關係用sequence diagram表達。

### 2.4 Development View

適用讀者:

資料庫管理員、系統工程師、資源管理者

範圍:

描述系統所使用的硬體配置, 包含外部系統。

## 2.5 Implementation View

適用讀者:

應用程式設計者、程式設計師

範圍:

描述軟體的總體架構，分為不同層次與子系統討論。

## 3. Architectural Goals and Constraints

### 3.1 Safety

MySQL伺服器、mail連線資訊使用lib格式包裝，只有管理員等級的人才能知曉。防止不相干人士干預。

軟體版本控制使用unity collaborate，具有版本回朔功能。

伺服器具有7天資料備份功能。

### 3.2 Security

使用SSL技術加密連線，確保客戶資料不被竊取。

### 3.3 Privacy

只有使用者才能登入並使用功能。此外，使用者只能看到自己有參與的會議及他人的租借日期，確保使用者隱私。

### 3.4 Portability

此軟體支援PC、Android手機，且因Unity可直接輸出APK格式，所以並不需要維護兩份平台的代碼。

### 3.5 Performance

由於使用Unity平台進行開發，所以程式運行速度會比一般的軟體慢，但使用者應無明顯差異。

連線速度由伺服器決定，資料傳遞取速度決於客戶網路。

### 3.6 Reuse

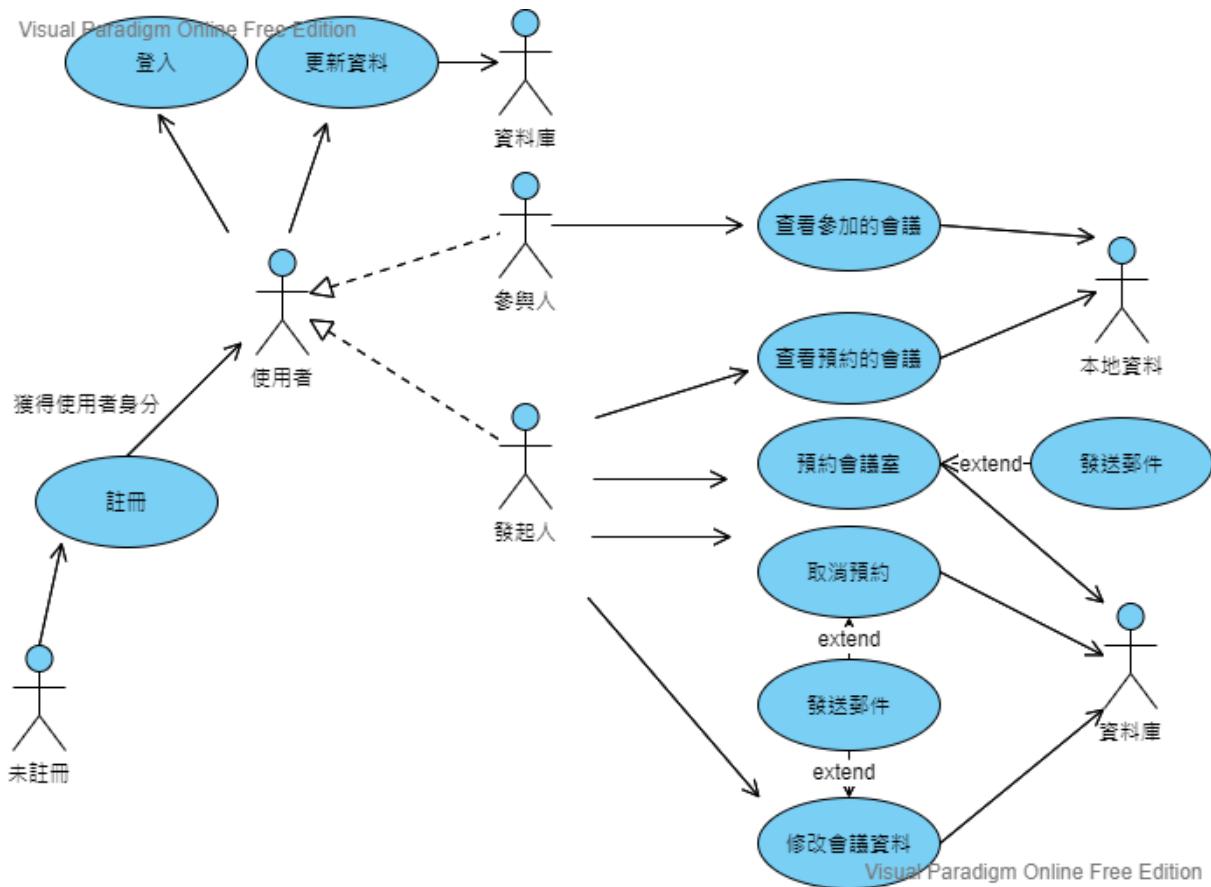
由於功能大多無重疊部分，能重複使用的部分並不高。

### 3.7 Development Tool

開發平台: Unity  
開發語言: C#  
資料庫: MySql  
圖表繪製: Visual Paradigm

## 4. Use Case View

### 4.1 Use Cases diagram



#### 4.1.1 註冊:

Use case	註冊
Scenario	當未註冊使用者嚟往取得使用者身分
Trigger	使用者介面註冊按鈕
Brief description	本例子展示了未註冊者如何透過和系統溝通取得使用者身分。
Actor	未註冊者
Preconditions	使用者在註冊介面
Post condition	回到登入和註冊介面
Flow	<p>1. 傳入資料庫時call檢查信箱function 回傳一個boolean</p> <ul style="list-style-type: none"> <li>• 0-&gt;進入step 2</li> <li>• 1-&gt;回傳("此信箱已被註冊") ，結束連線</li> </ul> <p>2. 傳入資料庫時call檢查帳號function 回傳一個boolean</p> <ul style="list-style-type: none"> <li>• 0-&gt;回傳("帳號建立成功"), 並儲存內容至資料庫，結束連線</li> <li>• 1-&gt;回傳("此帳號已被註冊"), 結束連線</li> </ul>

#### 4.1.2 登入:

Use case	登入
Scenario	使用者透過介面登入系統
Trigger	使用者介面登入按鈕
Brief description	本例子展示了”已註冊使用者”如何登入系統。
Actor	已註冊使用者
Preconditions	使用者在登入介面
Post condition	回到登入介面(0)or進入會議室選擇UI(1)
Flow	<p>1. 傳入資料庫時call檢查帳號function比對資料庫中儲存的帳號，回傳一個boolean</p> <ul style="list-style-type: none"> <li>• 0-&gt;回傳(“此帳號尚未註冊”), 結束連線</li> <li>• 1-&gt;step2</li> </ul> <p>2. call檢查密碼function比對資料庫中帳號對應的密碼，回傳一個boolean。</p> <ul style="list-style-type: none"> <li>• 0-&gt;回傳(“密碼錯誤”), 結束連線</li> <li>• 1-&gt;取得資料庫中會議室情況，回傳至程式，結束連線</li> </ul>

#### 4.1.3 更新資料:

Use case	更新資料
Scenario	使用者透過介面更新資料
Trigger	使用者UI介面更新資料按鈕
Brief description	本例子允許使用者透過更新按鈕獲得資料庫最新的會議資訊。
Actor	已註冊使用者
Preconditions	使用者登入成功
Post condition	資料已更新
Flow	<ol style="list-style-type: none"> <li>1. Call更新function</li>   <li>2. 取得資料庫中會議室情況回傳至程式，結束連線</li> </ol>

#### 4.1.4 查看會議:

Use case	登入
Scenario	使用者透過介面查看與自己相關的會議
Trigger	使用者介面"已預約"或"已參與"按鈕
Brief description	本例子允許使用者查看所有與自己相關的會議資訊
Actor	已註冊使用者
Preconditions	使用者在登入成功
Post condition	進入已預約或已參與介面
Flow	<ol style="list-style-type: none"> <li>1. Call更新function</li>   <li>2. 取得資料庫中會議室情況回傳至程式，結束連線</li> </ol>

#### 4.1.5 預約會議室:

Use case	預約會議室
Scenario	使用者透過介面預約要使用的會議室
Trigger	使用者介面預約會議按鈕
Brief description	本例子允許使用者預約可以借用的會議室，系統根據時段判斷各會議室是否可以借用，並發送郵件給所有受邀請人。
Actor	已註冊使用者
Preconditions	使用者於預約介面
Post condition	顯示預約結果
Flow	<ol style="list-style-type: none"> <li>1. 傳入預約的會議室名稱、信息、時間、受邀請人帳號</li> <li>2. Call 檢查會議室時段function，回傳一個boolean <ul style="list-style-type: none"> <li>• 0-&gt;回傳("該時段已被借用，預約不成立")，結束連線</li> <li>• 1-&gt;step3</li> </ul> </li> <li>3. 針對所有受邀請人Call 檢查帳號function，回傳一個boolean <ul style="list-style-type: none"> <li>• 0-&gt;於輸出存入(("帳號)不是使用者，預約不成立")，繼續檢查</li> <li>• 1-&gt;繼續檢查其他人</li> </ul> </li> <li>4. 針對所有受邀請人call 檢查帳號時段funciton，回傳一個boolean <ul style="list-style-type: none"> <li>• Call 查詢funciton查詢該帳號的使用時間</li> <li>• 0-&gt;於輸出存入(("帳號)已被邀請至其他會議，預約不成立")，繼續檢查</li> <li>• 1-&gt;繼續檢查其他人</li> </ul> </li> <li>5. Call 最終檢查function，拿取step3、step4結果，回傳一個boolean，若</li> </ol>

	<p>結果中有0輸出0, 皆為1輸出1</p> <ul style="list-style-type: none"> <li>● 0-&gt;輸出結果, 結束連線</li> <li>● 1-&gt;將預約的會議室名稱、信息、時間、受邀請人帳號存入資料庫, 回傳("預約成功")與當次預約的內容, 結束連線</li> </ul> <p>6. 預約成功, 執行寄出Email function</p>
--	--

#### 4.1.6 取消預約:

Use case	取消預約
Scenario	使用者需要取消預定錯誤的會議
Trigger	使用者介面取消預約按鈕
Brief description	本例子允許會議發起人取消已預約的會議, 僅限於發起人本人建立的所有會議, 系統會發送郵件給所有被取消會議的會議參與者和受邀請人。
Actor	會議發起人
Preconditions	發起人於會議室資訊介面中
Post condition	顯示執行結果
Flow	<ol style="list-style-type: none"> <li>1. 傳入欲取消之會議室名稱、時間</li> <li>2. Call 刪除 function刪除相關資訊</li> <li>3. 回傳("取消預約成功"), 結束連線</li> <li>4. 取消預約成功, 執行寄出Email function</li> </ol>

#### 4.1.7 修改會議資訊:

Use case	修改會議資訊
Scenario	使用者希望改變自身發起的會議資訊。
Trigger	使用者介面修改按鈕
Brief description	本例子允許會議發起人更改已預約的會議會議資訊(如: 時段、參與人、會議內容與標題)
Actor	會議發起人
Preconditions	發起人於會議室資訊介面中
Post condition	顯示執行結果
Flow	<ol style="list-style-type: none"> <li>1. 傳入修改會議的名稱、時間、內容、受邀請人資訊</li> <li>2. Call 檢查會議室時段function，回傳一個boolean <ul style="list-style-type: none"> <li>• 0-&gt;回傳("時段衝突，修改不成立")，結束連線</li> <li>• 1-&gt;step3</li> </ul> </li> <li>3. Call 修改內容function，修改相關訊息</li> <li>4. 回傳("修改成功")，結束連線</li> <li>5. 修改成功，執行寄出Email function</li> </ol>

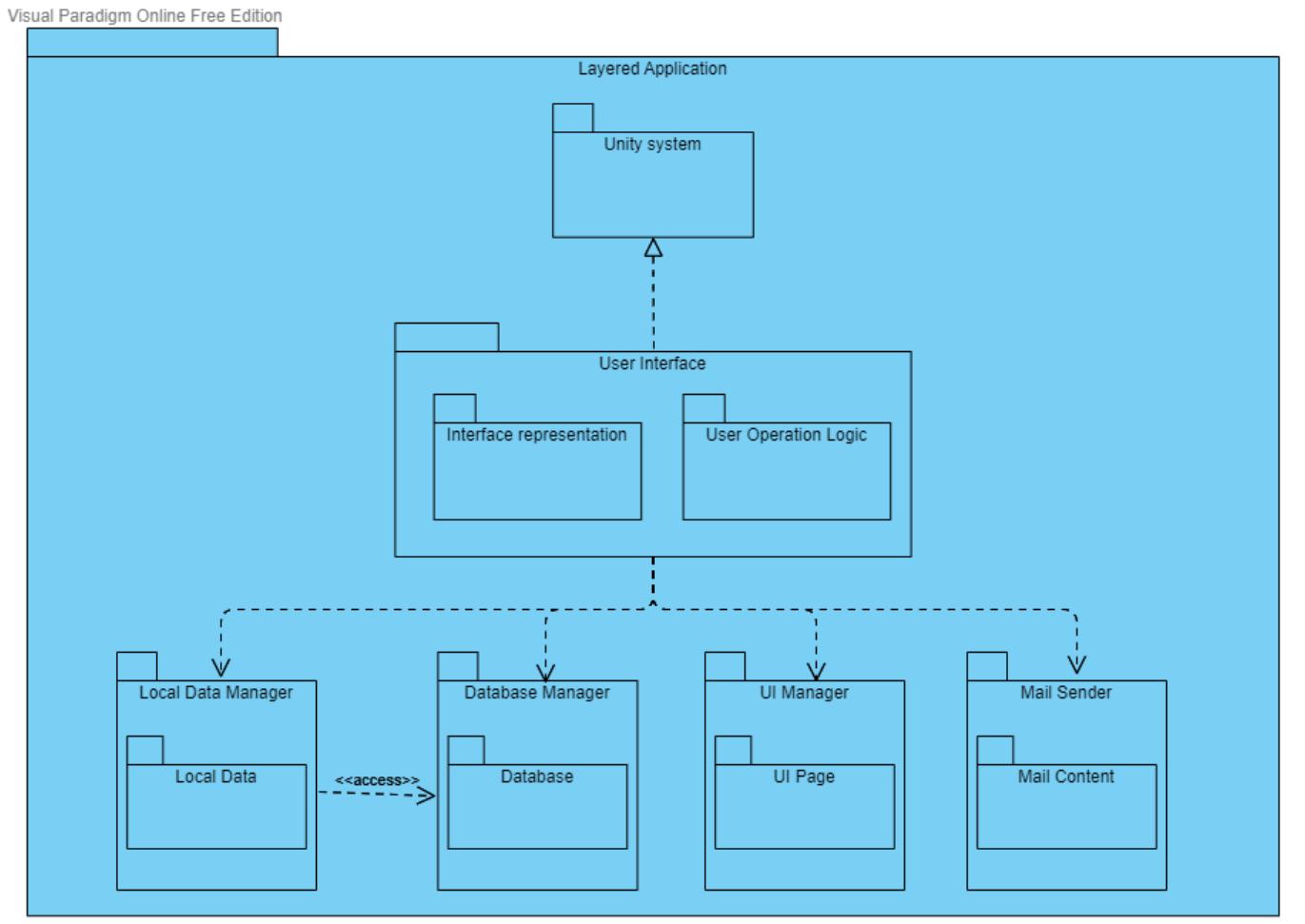
## 5. Logical View

### 5.1 Overview

#### 5.1.1 Subsystems

1. Unity system  
顯示介面、實現使用者操作邏輯
2. UI Manager  
由Unity提供的架構實現，用來管理UI頁面
3. Local Data Manager  
維護系統內部所儲存的資料，以提供資訊給User Interface system顯示。藉由Database Manager所提供的介面來獲取資料庫的資料。
4. Database Manager  
提供獲取、新增、驗證外部資料庫資料的API。

#### 5.1.2 System Layer Package Diagram



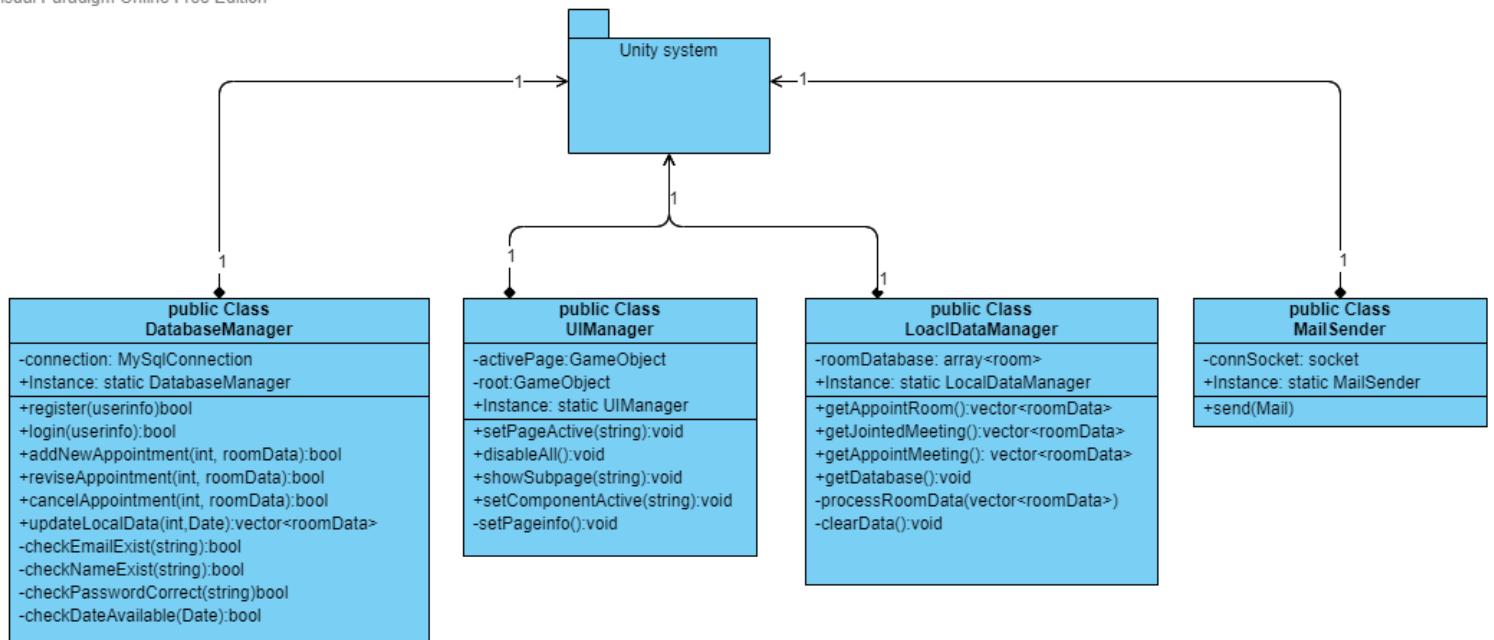
Visual Paradigm Online Free Edition

## 5.2 Architectural Significant Design Packages

### 5.2.1 Class diagram

物件的函式的調用由Unity system的UI event呼叫，因此所有關鍵元件的生命週期都與Unity system相關。每個Class中都會自動創建一份static Instance，使Unity System只會對每個Class維護一份Instance。

Visual Paradigm Online Free Edition



Visual Paradigm Online Free Edition

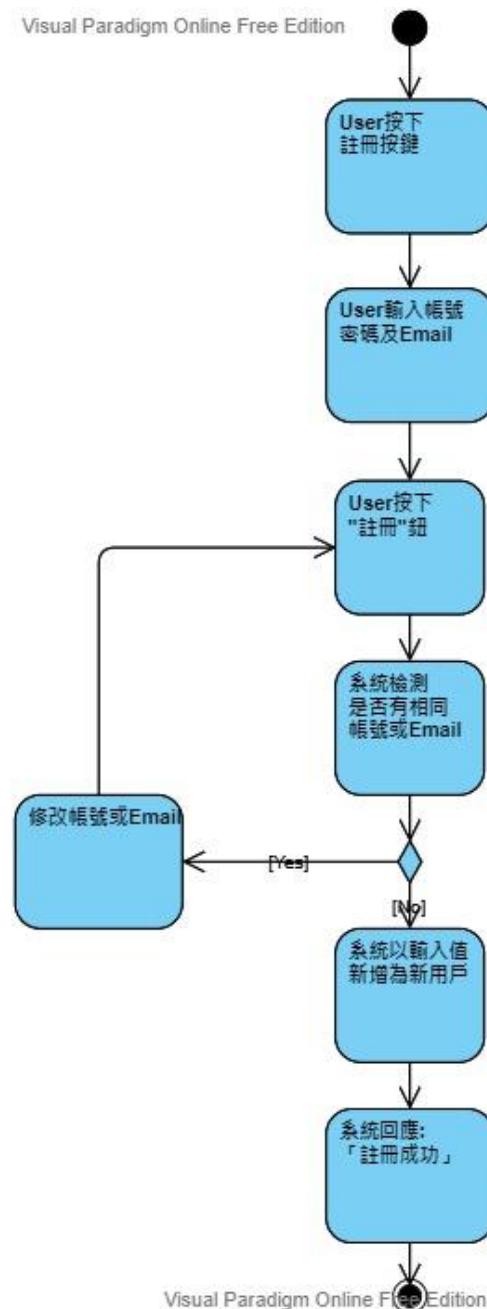
## 6. Process View

此章節會提供Activity Diagram與Sequence Diagram, 前者著重於使用者與使用者介面之間的行為關係;後者著重於使用者介面與內外部資料庫的時序關係, 而不會詳細表達使用者行為。

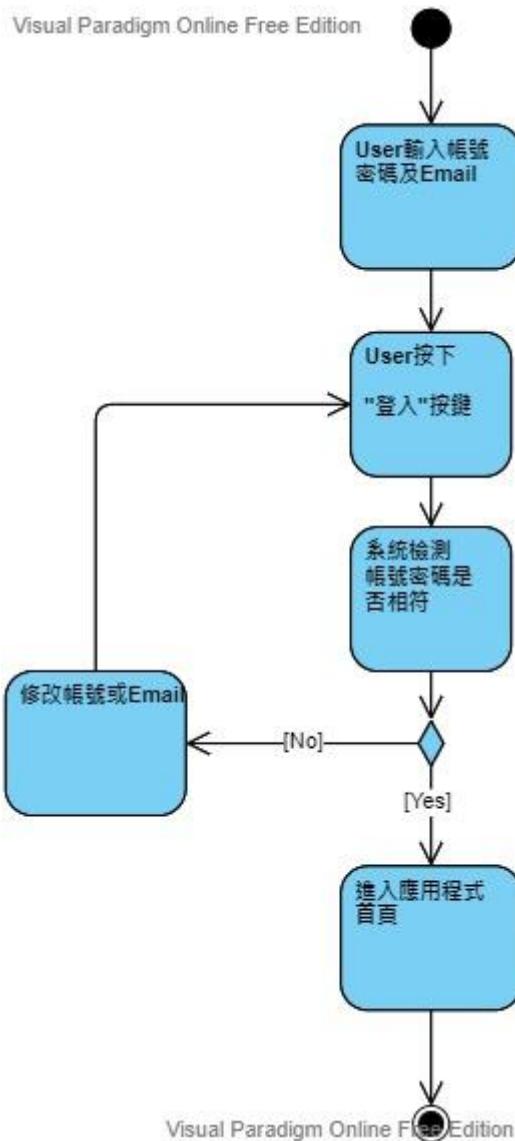
### 6.1 User activity diagram

以下activity diagram顯示了每種Use case的使用者行為流程

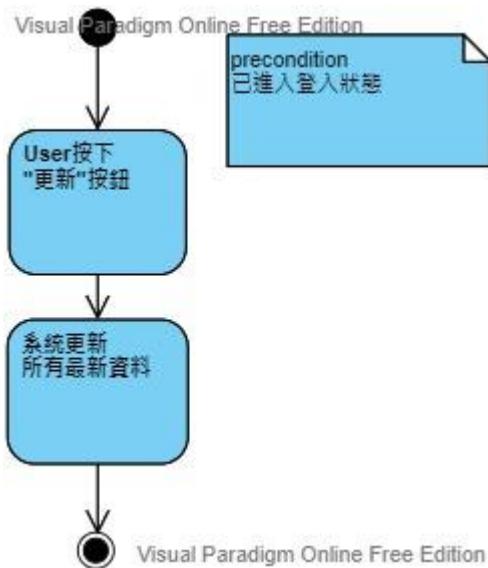
#### 6.1.1 註冊



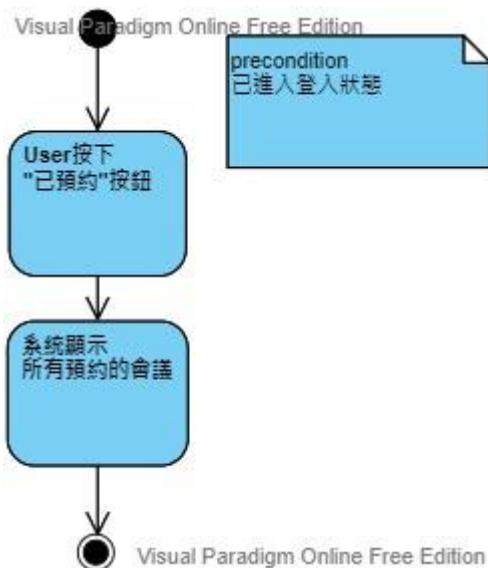
### 6.1.2 登入



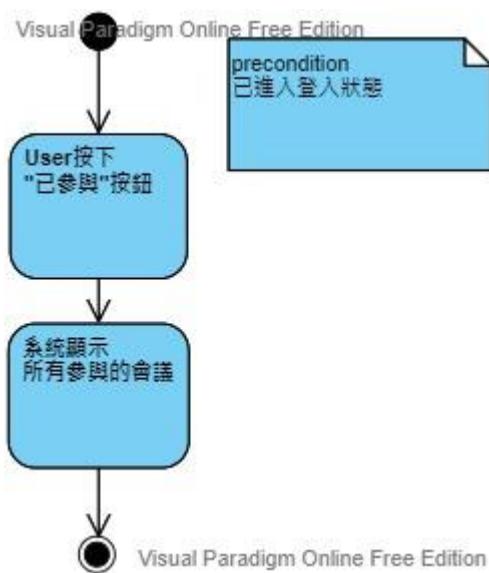
### 6.1.3 更新



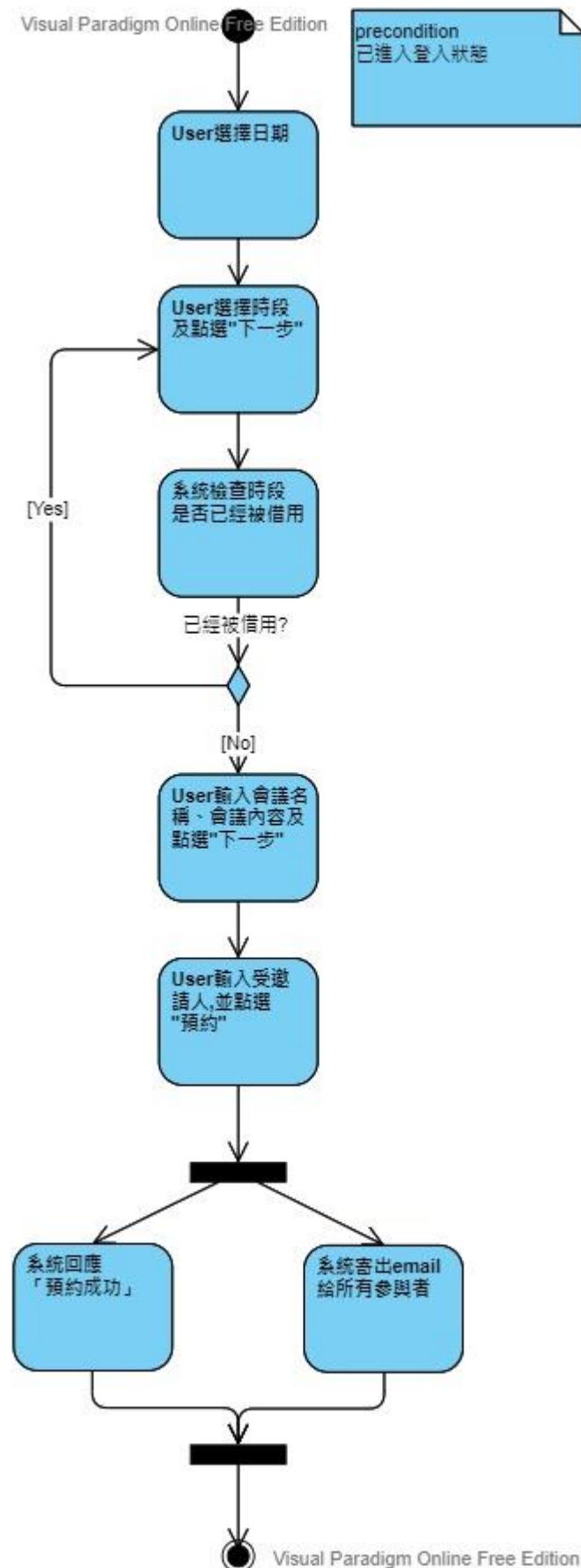
### 6.1.4 查看預約的會議室



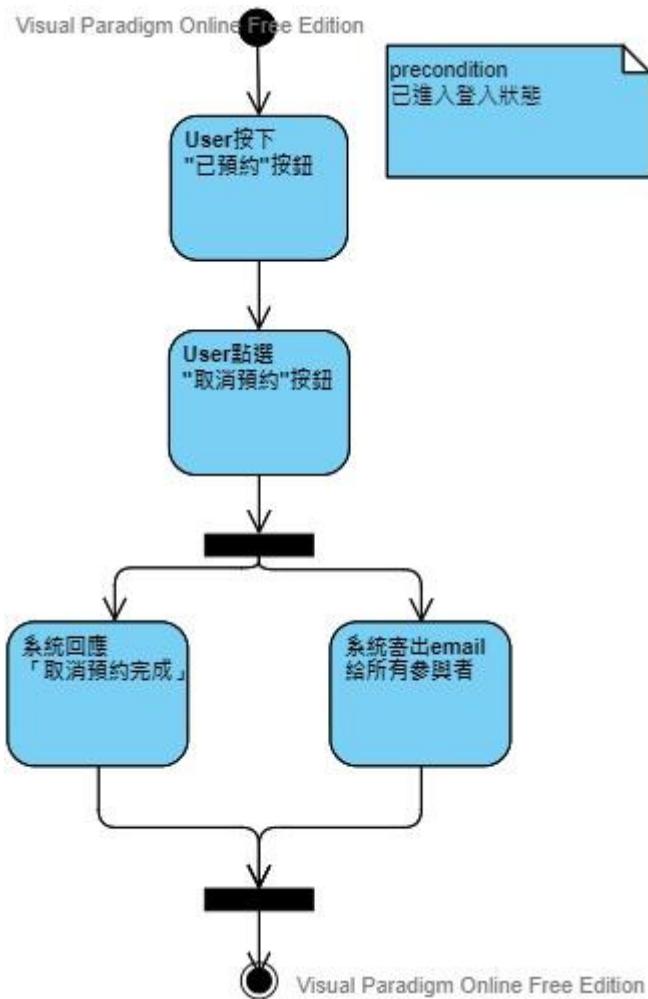
### 6.1.5 查看參與的會議室



### 6.1.6 預約會議室



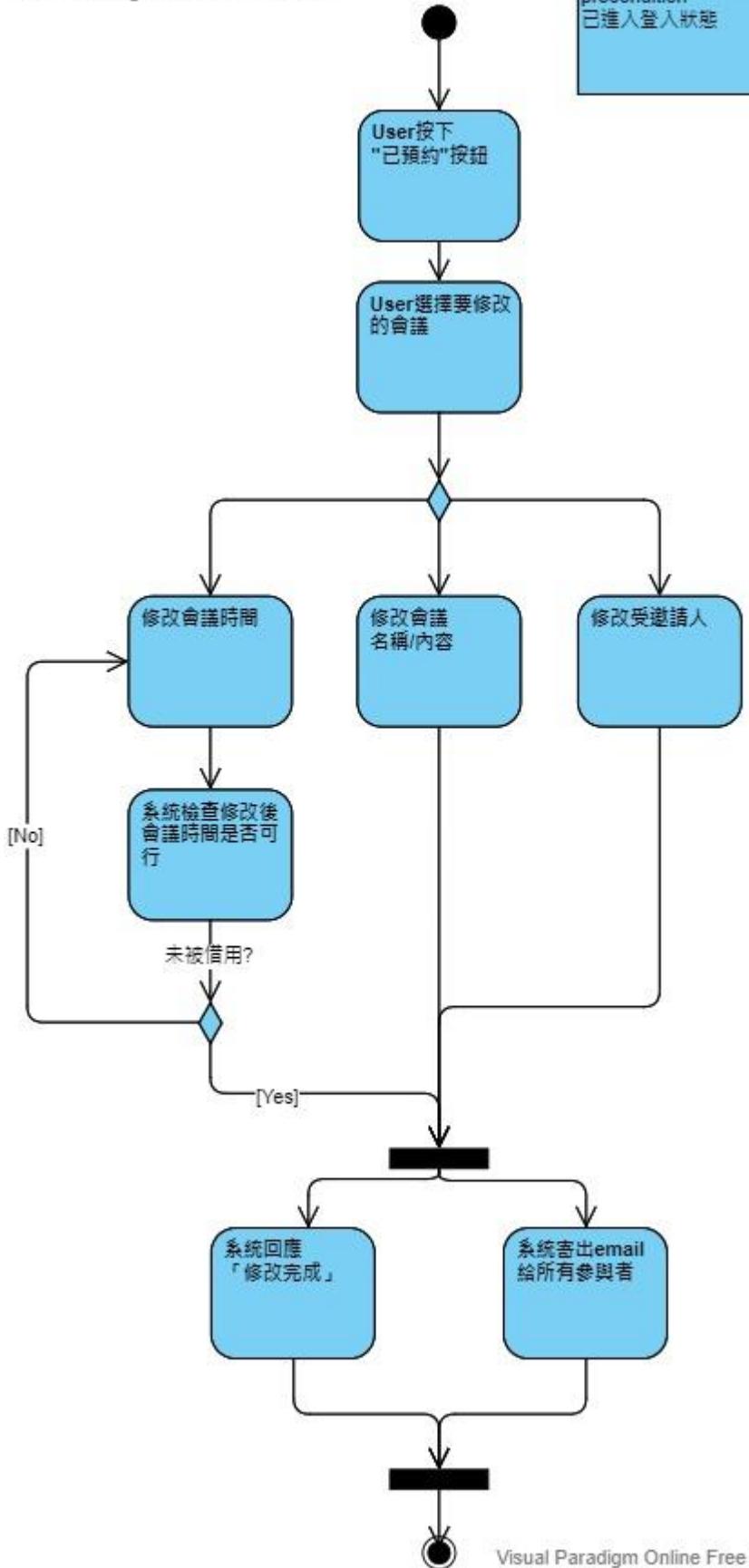
### 6.1.7 取消會議



### 6.1.8 修改會議資訊

Visual Paradigm Online Free Edition

precondition  
已進入登入狀態

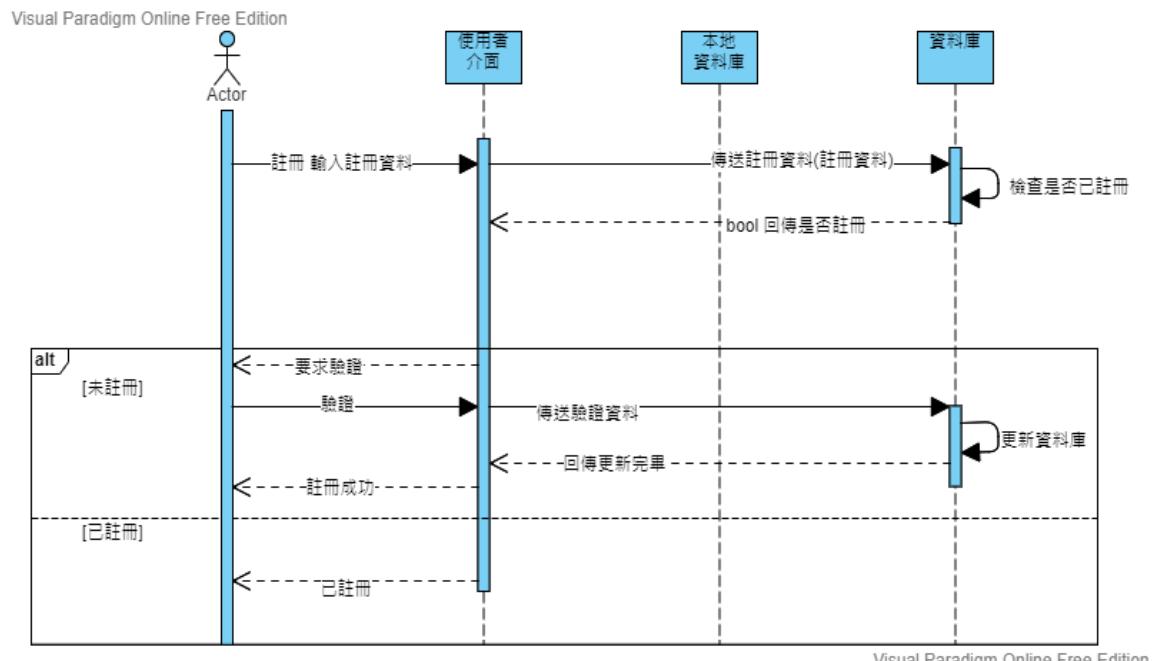


Visual Paradigm Online Free Edition

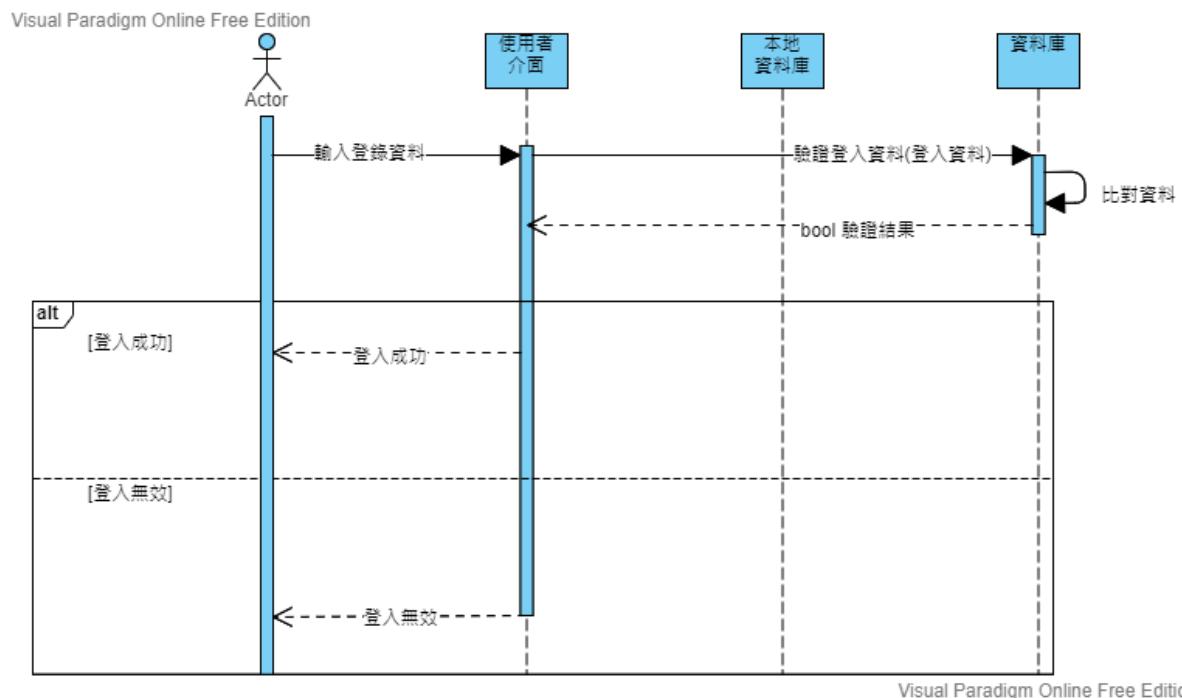
## 6.2 System sequence diagram

以下sequence diagram對每種使用者行為顯示內外資料庫與使用者界面之間的互動時序關係

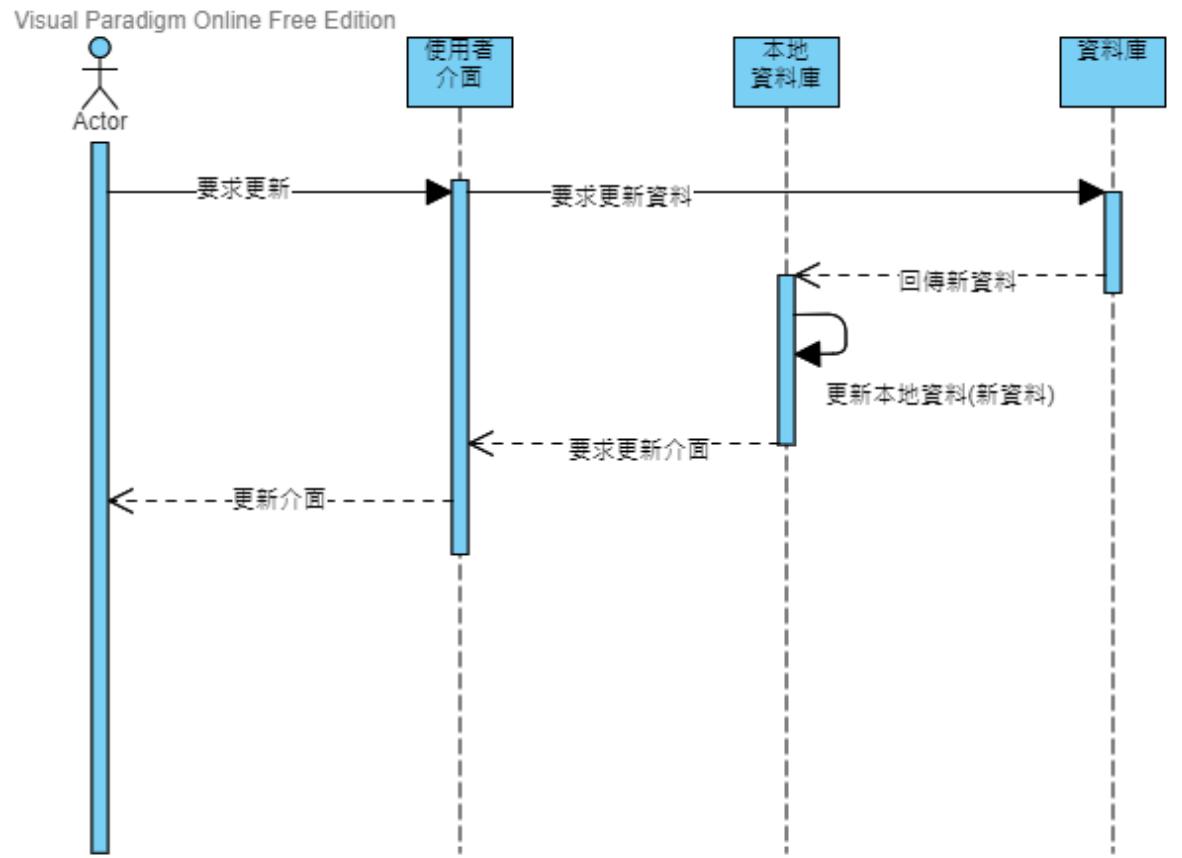
### 6.2.1 註冊



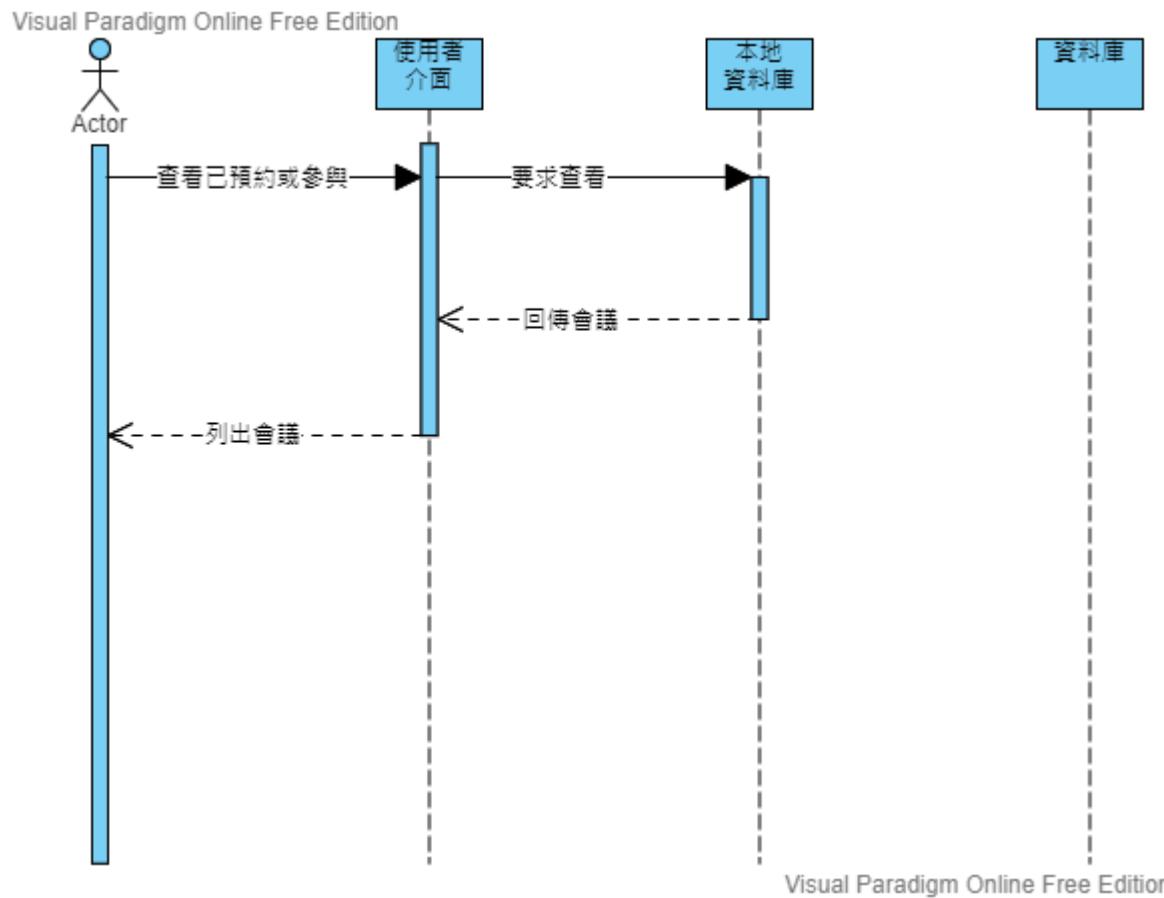
## 6.2.2 登入



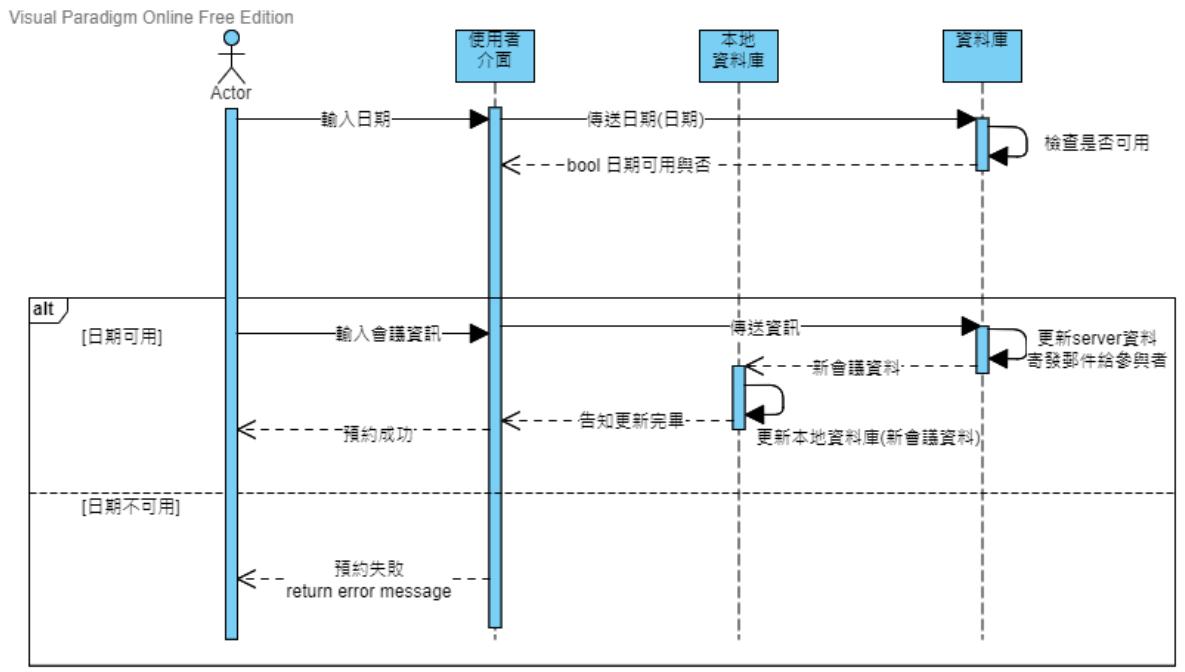
## 6.2.3 更新



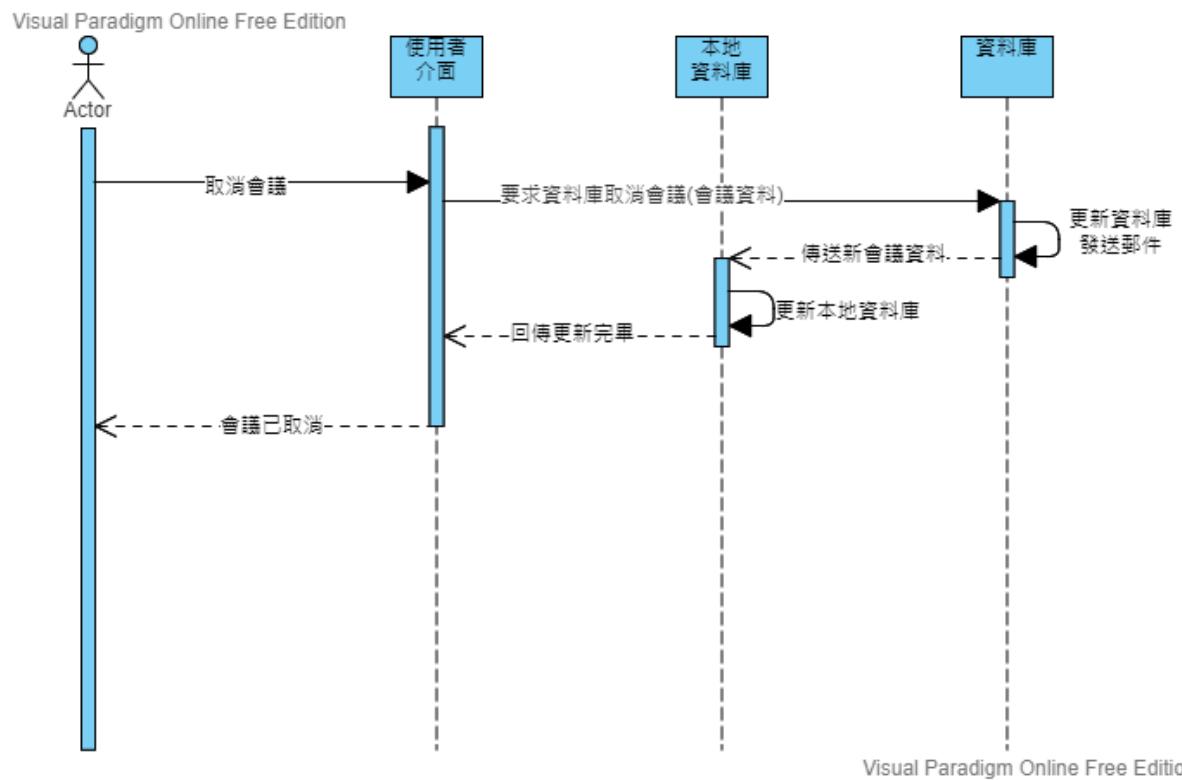
#### 6.2.4 查看預約或參與的會議室



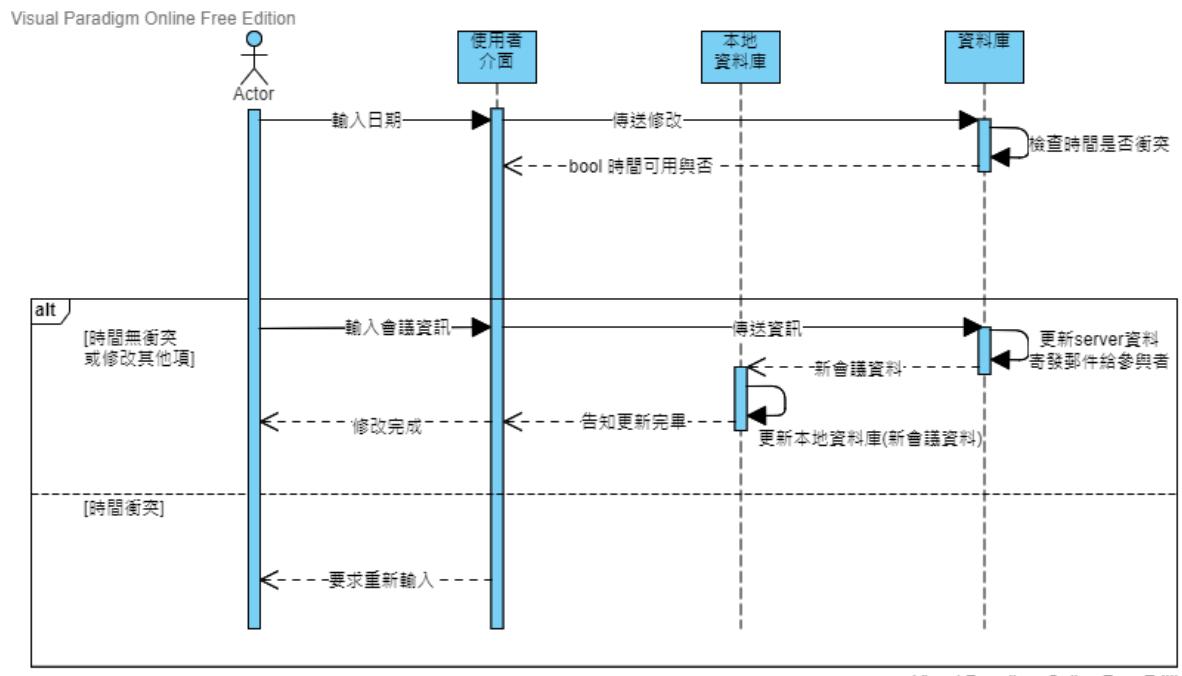
#### 6.2.5 預約會議室



### 6.2.6 取消會議

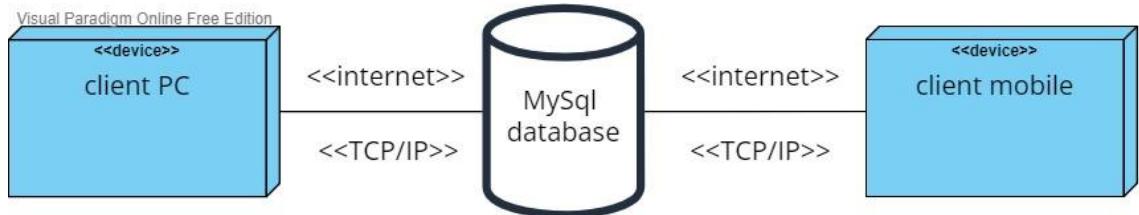


### 6.2.7 修改會議資訊



## 7. Deployment View

此預約系統的硬體只需要客戶端的裝置，利用裝置上安裝的此預約程式來進行相關操作，然後以網路TCP/IP協定來傳輸request到mysql database之中提取所需的資料，database也以相同的方式將結果傳回客戶裝置。

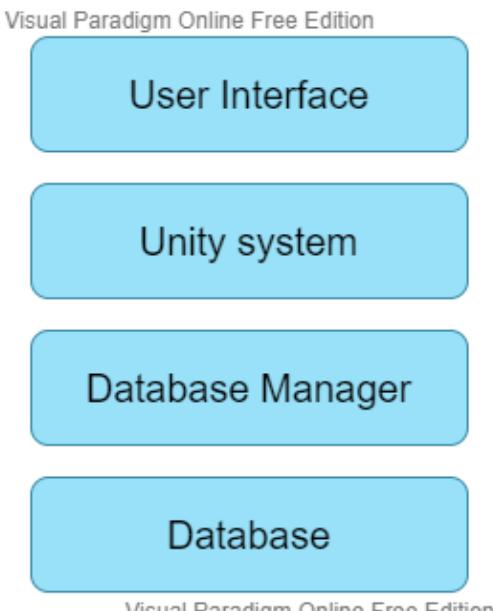


## 8. Implementation View

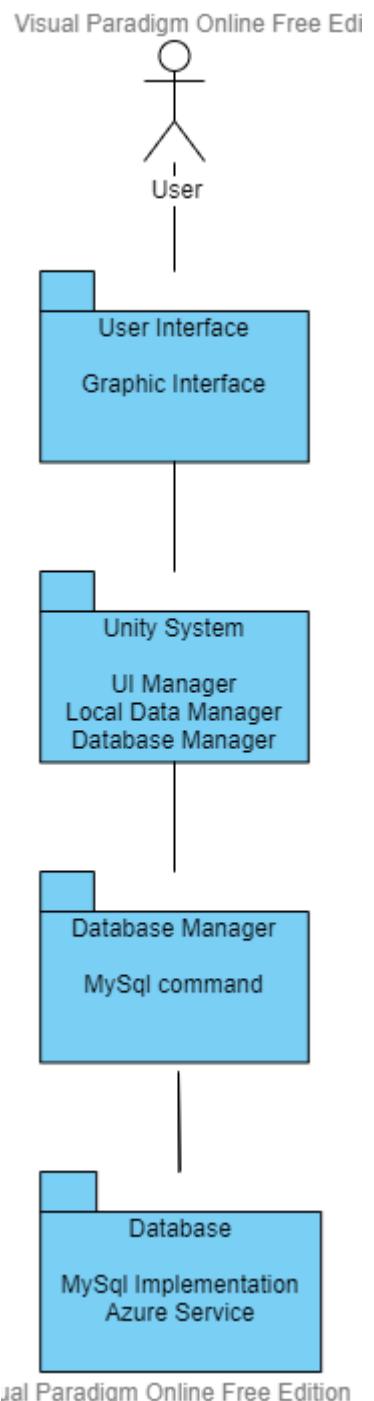
### 8.1 Overview

本系統主要分為以下四大層次。

1. User Interface: 使用者輸入資料、獲取資訊。
2. Unity system: 為主要運作系統，負責管理介面、本地資料，及檢驗輸入資料正確性，詳細實作資訊於[5.1.2](#)。
3. Database Manager: 外部資料庫的接口。負責獲取、傳送資料。
4. Database: 由MySQL實作，Azure提供伺服器服務。



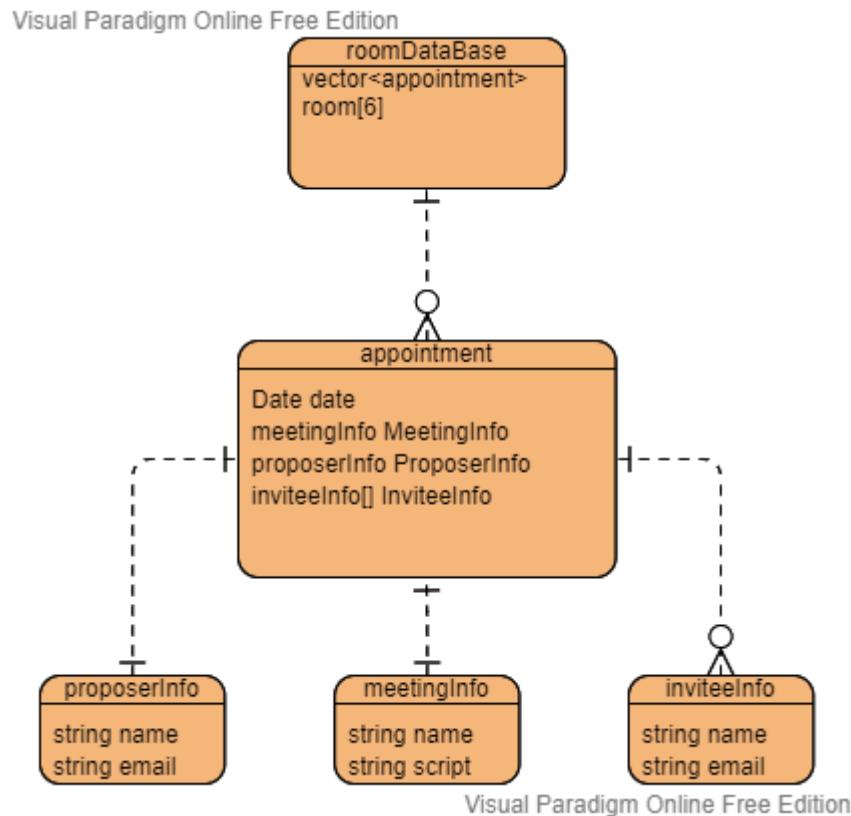
## 8.2 Layers



## 9. Data View

### 9.1 Local data view

#### 9.1.1 Local Data Entity Diagram



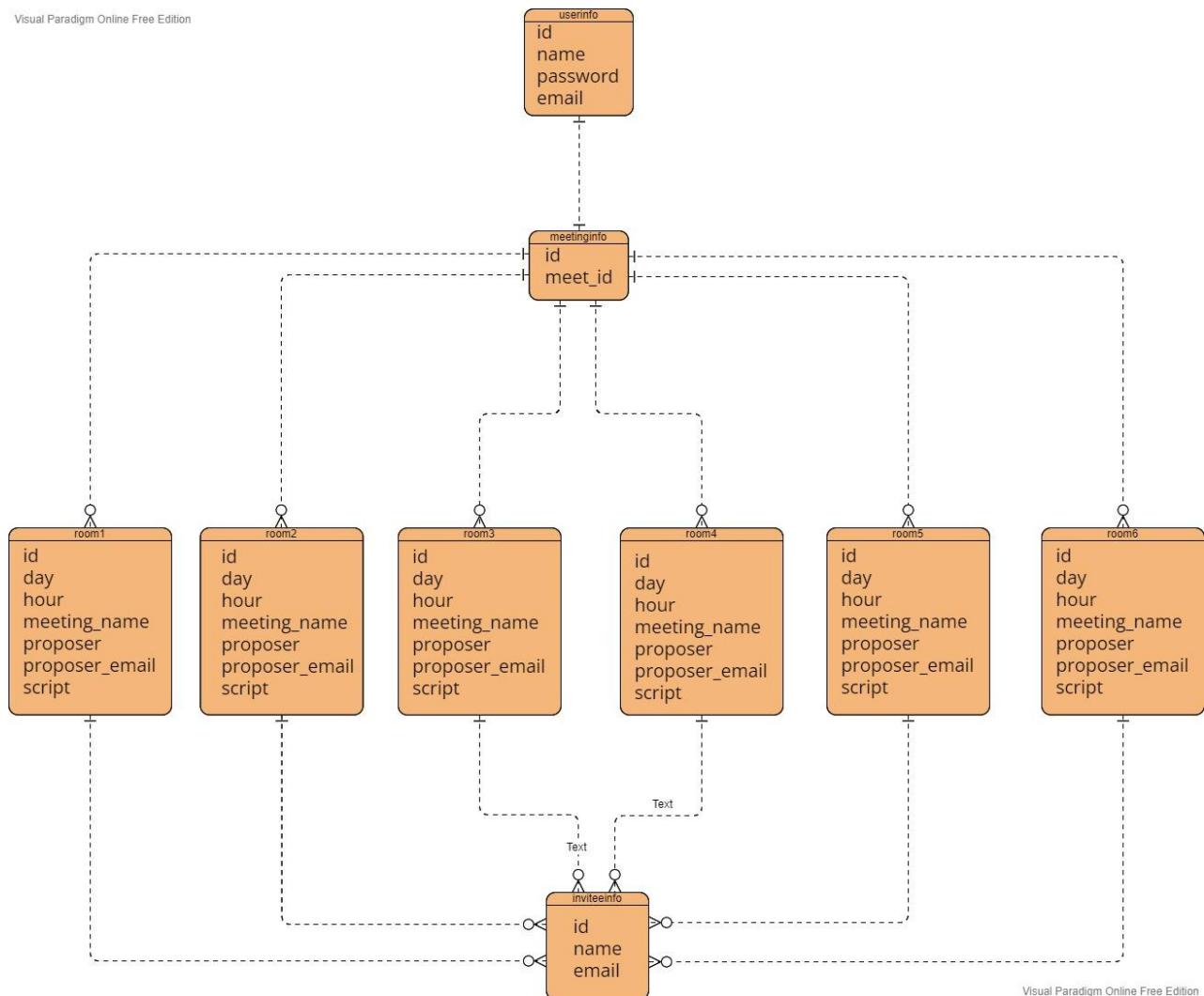
#### 9.1.2 Local Data Implementation

上述entity diagram用以描述Local Database的實作資料格式。

1. roomDatabase: 六間會議室的預約資料，為宣告於 LocalDataManager的實例。
2. appointment: 預約的資料格式。
3. meetingInfo: 會議資訊的資料格式。
4. proposerInfo: 發起人的資料格式
5. inviteeInfo: 受邀請人的資料格式

## 9.2 SQL view

### 9.2.1 MySql Entity Diagram



### 9.2.2 MySql implementation

根據上述的entity diagram 來實作mysql database的table, 以下所有的table都位於同一database之中。

1. **userinfo**：為儲存使用者相關資料的表格。
2. **meetinginfo**：為儲存使用者的有參與的會議表格，一個使用者只能有一個所屬的會議資訊表格。
3. **room1**:會議室1的表格，儲存了所有此會議室的預約資訊。
4. **room2**:會議室2的表格，儲存了所有此會議室的預約資訊。
5. **room3**:會議室3的表格，儲存了所有此會議室的預約資訊。
6. **room4**:會議室4的表格，儲存了所有此會議室的預約資訊。
7. **room5**:會議室5的表格，儲存了所有此會議室的預約資訊。
8. **room6**:會議室6的表格，儲存了所有此會議室的預約資訊。
9. **inviteeinfo**:儲存受邀請至此會議的參與者信息，一個預約可以有多個參與者信息表格。

## 10. Size and Performance

1. 本系統目標客群為中小型公司所以使用者數量預設為300人以內。  
系統對於會議室的分配應以建立時間為標準。
2. 連結與資料讀取速度由資料庫管理者決定，但延遲不大於10秒。
3. 資料庫傳送完所有信件的時間應小於1分鐘。
4. 系統使用客戶端與伺服器溝通來進行工作，同時會在使用者端建立一份本地資料庫儲存會議資料，因此客戶端需有本系統應用程式大小加上資料庫的儲存空間(實際大小尚未計算)。
5. 建立時間以資料庫端系統時間作為判斷。

## 11. Quality

1. 使用者最少需要能夠運行unity的系統需求，需求如下表

OS	Windows	Mac OS	Linux
OS version	Windows 7 (SP1+) , Windows 10 and Windows 11	High Sierra 10.13+	Ubuntu 20.04, Ubuntu 18.04, and CentOS 7
CPU	x86, x64 architecture with SSE2 instruction set support.	Apple Silicon, x64 architecture with SSE2.	x64 architecture with SSE2 instruction set support.
G API	DX10, DX11, DX12 capable.	Metal capable Intel and AMD GPUs	OpenGL 3.2+, Vulkan capable.

2. 使用者介面設計應易讀且方便使用大多數操作解以按鈕觸發。
3. 伺服器端應隨時保持運行全年無休。
4. 本系統操作僅限PC、Andriod平台。
5. 介面提供開發者資訊並從客戶端接收使用回饋。
6. 非會議參與者僅能查看會議室是否可使用。