

國立臺北商業大學

資訊管理系

109 資訊系統專案設計

系統手冊



組 別：第 109408 組

題 目：CatchU 桌助你

指導老師：林宏仁老師

組 長：N1066442 葉芝秀

組 員：10646037 關宇辰 N1066418 周育德

N1066432 林琪容

中 華 民 國 109 年 11 月 25 日

國立臺北商業大學專題課程作品

電子文件上網授權書

本授權書所授權之作品為授權人在 國立臺北商業大學 資訊管理(科/系/所)

109408 組 109 學年度 第 二 學期專題課程之成果報告作品。

專題題目：CatchU 桌助你

指導教授：林宏仁

☐ 本人 ☐ 團體著作人

☐ 同意 提供讀者基於個人非營利性質之線上檢索、閱覽、下載或列印。

茲同意將授權人擁有著作權之上列論文全文(含摘要)無償授權本人就讀學校(國立臺北商業大學)圖書館，不限地域、時間與次數，以微縮、光碟或其他各種數位化方式將上列作品收錄、重製與利用，並得將數位化之上列作品及其電子檔上載資料庫系統。於著作權法合理使用範圍內，讀者得進行線上檢索、閱覽、下載或列印。

專題課程成果報告作品全文上載網路公開之範圍及時間：

本校區域網路 ☐ 立即公開

☐ 自中華民國 年 月 日公開

校外網際網路 ☐ 立即公開

☐ 自中華民國 年 月 日公開

☐ 不同意 提供讀者基於個人非營利性質之線上檢索、閱覽、下載或列印。

授權人
簽 名

中 華 民 國 109 年 11 月 25 日

目錄

第一章 背景與動機.....	15
1-1 簡介.....	15
1-2 問題與機會.....	17
1-3 相關系統探討.....	18
2-1 系統目標.....	19
2-2 預期成果.....	21
第三章 系統規格.....	22
3-1 系統架構.....	22
3-2 系統軟、硬體需求技術平台.....	23
3-3 使用標準與工具.....	24
第四章 專案時程與組織分工.....	25
4-1 專案時程.....	25
第 5 章 需求模型.....	28
5-1 使用者需求.....	28
5-2 使用者個案圖(Use case diagram).....	29
5-3 使用個案描述.....	30
5-4 分析類別圖(Analysis class diagram).....	41
第 6 章設計模型.....	42
6-1 循序圖(Sequential diagram).....	42

6-2 設計類別圖(Design class diagram)	49
第七章 實作模型.....	50
7-1 佈署圖(Deployment diagram)	50
7-2 套件圖(Package diagram)	51
7-4 狀態機(State machine)	53
第八章 資料庫設計.....	54
8-1 資料庫關聯表	54
8-2 表格及其 Meta data	55
第九章 程式.....	61
9-1 元件清單及其規格描述	61
十、測試模型.....	168
10-1 測試計畫	168
10-2 測試個案與測試結果資料	173
第十一章 操作手冊.....	193
第十二章 使用手冊.....	195
第十三章 感想.....	204
第十四章 參考資料.....	208

表目錄

表 1-2-1 SWOT 分析表.....	17
表 1-3-1 相關系統探討比較表.....	18
表 3-2-1 伺服器端規格表	23
表 3-2-2 網站端規格表	23
表 3-2-3 系統硬體需求表	23
表 3-3-1 使用標準與工具表	24
表 4-1-1 專案時程表	25
表 4-2-1 專案分工表	27
表 8-2-1 資料表描述：會員	55
表 8-2-2 資料表描述：員工	55
表 8-2-3 資料表描述：店家資訊.....	56
表 8-2-4 資料表描述：菜單	57
表 8-2-5 資料表描述：餐點類別	57
表 8-2-6 資料表描述：會員訂單明細	58
表 8-2-7 資料表描述：計算時間	59
表 8-2-8 資料表描述：會員結帳明細	59
表 8-2-9 資料表描述：會員儲值	60
表 9-1-1 首頁功能列表	61
表 9-1-2 員工功能列表	61

表 9-1-3 店家資訊功能列表	62
表 9-1-4 菜單資訊功能列表	63
表 9-1-5 類別功能列表	64
表 9-1-6 餐點訂單功能列表	65
表 9-1-7 會員消費明細功能列表	65
表 9-1-8 儲值點數功能列表	66
表 9-1-9 使用者功能列表	66
表 9-1-10 首頁功能列表	67
表 9-1-11 使用者功能列表	67
表 9-1-12 查看店家資訊功能列表	68
表 9-1-13 訂單明細功能列表	68
表 9-1-14 會員資料功能列表	69
表 9-1-15 查看菜單功能列表	70
表 9-1-16 計時及結帳功能列表	70
表 9-1-17 首頁功能列表	71
表 9-1-18 搜尋員工資料列表	71
表 9-1-19 建立員工更新資料列表	72
表 9-1-20 導至尋找員工資料之頁面列表	73
表 9-1-21 顯示員工原本資料於更新頁面列表	74
表 9-1-22 函式呼叫服務列表	75

表 9-1-23 建立店家資料列表	77
表 9-1-24 導至店家資料新增頁面列表	78
表 9-1-25 搜尋店家資料列表	79
表 9-1-26 取得欲刪除店家之編號列表	80
表 9-1-27 導至店家資料刪除頁面列表	81
表 9-1-28 建立店家更新資料列表	81
表 9-1-29 導至搜尋店家編號之頁面列表	83
表 9-1-30 原有店家資料回傳至更新頁面列表	83
表 9-1-31 函式呼叫服務列表	85
表 9-1-32 建立菜單餐點物件列表	88
表 9-1-33 導至餐點資料新增頁面列表	90
表 9-1-34 搜尋餐點資料列表	91
表 9-1-35 取得欲刪除餐點之編號列表	92
表 9-1-36 導至餐點刪除頁面列表	93
表 9-1-37 建立餐點更新資料列表	94
表 9-1-38 導至搜尋餐點編號之頁面列表	95
表 9-1-39 原有餐點資料回傳至更新頁面列表	96
表 9-1-40 函式呼叫服務列表	97
表 9-1-41 建立類別資料列表	101
表 9-1-42 導至類別新增頁面列表	102

表 9-1-43 搜尋類別資料列表	103
表 9-1-44 取得欲刪除類別之編號列表	104
表 9-1-45 導至類別刪除頁面列表	105
表 9-1-46 建立類別更新資料列表	106
表 9-1-47 導至搜尋類別編號之頁面列表	107
表 9-1-48 原有類別資料回傳至更新頁面列表	108
表 9-1-49 函式呼叫服務列表	109
表 9-1-50 搜尋會員訂單之服務列表	112
表 9-1-51 函式呼叫服務列表	113
表 9-1-52 搜尋會員結帳明細列表	114
表 9-1-53 函式呼叫服務列表	115
表 9-1-54 建立會員儲值資料列表	116
表 9-1-55 導至會員儲值新增頁面列表	117
表 9-1-56 函式呼叫服務列表	118
表 9-1-57 顯示登入中員工名稱列表	119
表 9-1-58 將員工登出	119
表 9-1-59 判斷員工是否登入	120
表 9-1-60 搜尋員工帳號密碼服務列表	121
表 9-1-61 檢查登入權限列表	122
表 9-1-62 函式呼叫服務列表	123

表 9-1-63 首頁功能列表	124
表 9-1-64 顯示登入中會員名稱列表	124
表 9-1-65 將會員登出列表	125
表 9-1-66 判斷會員是否登入列表	125
表 9-1-67 搜尋會員帳號密碼服務列表	126
表 9-1-68 檢查登入權限列表	127
表 9-1-69 函式呼叫服務列表	128
表 9-1-70 搜尋店家資料之服務列表	129
表 9-1-71 函式呼叫服務列表	130
表 9-1-72 會員的訂單明細列表	131
表 9-1-73 建立訂單明細列表	132
表 9-1-74 導至訂單明細新增頁面列表	133
表 9-1-75 取得欲刪除訂單明細之編號列表	134
表 9-1-76 導至訂單明細刪除頁面列表	135
表 9-1-77 建立訂單明細更新資料列表	136
表 9-1-78 導至搜尋訂單明細編號之頁面列表	137
表 9-1-79 將原有訂單明細回傳至更新頁面列表	138
表 9-1-80 函式呼叫服務列表	139
表 9-1-81 取得會員的資料列表	143
表 9-1-82 建立會員資料列表	144

表 9-1-83 導至會員資料新增頁面列表	146
表 9-1-84 搜尋會員資料列表	146
表 9-1-85 取得欲刪除會員之帳號列表	147
表 9-1-86 導至會員資料刪除頁面列表	148
表 9-1-87 建立會員更新資料列表	148
表 9-1-88 導至搜尋會員帳號之頁面列表	149
表 9-1-89 將原有會員資料回傳至更新頁面列表	150
表 9-1-90 函式呼叫服務列表	151
表 9-1-91 搜尋菜單列表	154
表 9-1-92 函式呼叫服務列表	155
表 9-1-93 建立計時資料列表	156
表 9-1-94 導至計時資料新增頁面列表	157
表 9-1-95 建立結帳資料列表	158
表 9-1-96 導至結帳資料頁面列表	159
表 9-1-97 函式呼叫服務列表	160
表 10-2-1 員工註冊	173
表 10-2-2 員工登入	173
表 10-2-3 員工	174
表 10-2-4 店家資訊	175
表 10-2-5 菜單	177

表 10-2-6 餐點類別資訊	179
表 10-2-7 會員餐點訂單	181
表 10-2-8 會員消費明細	182
表 10-2-9 儲值點數	182
表 10-2-10 登出	183
表 10-2-11 會員註冊	184
表 10-2-12 會員登入	184
表 10-2-13 店家資訊	185
表 10-2-14 開始遊玩	185
表 10-2-15 點餐	186
表 10-2-16 訂單明細	187
表 10-2-17 結帳	189
表 10-2-18 會員	191
表 10-2-19 登出	192
表 11-1-1 網頁操作手冊表	193
表 12-1 登入及註冊	195
表 12-2 會員及員工資訊	196
表 12-3 店家資訊	197
表 12-4 點餐及菜單	198
表 12-5 訂單明細	199

表 12-6 員工端之查看會員消費明細	200
表 12-7 員工端之餐點類別資訊	200
表 12-8 員工端之儲值點數	201
表 12-9 會員端之開始遊玩	201
表 12-10 會員端之結束遊玩	202

圖目錄

圖 3-1-1 系統架構圖	22
圖 5-2-1 使用者個案圖	29
圖 5-3-1 「登入及註冊」之活動圖	30
圖 5-3-2 「會員資料」之活動圖	31
圖 5-3-3 「員工資料」之活動圖	32
圖 5-3-4 「會員查看店家資訊」之活動圖	33
圖 5-3-5 「員工管理店家資訊」之活動圖	34
圖 5-3-6 「會員點餐」之活動圖	35
圖 5-3-7 「員工管理菜單」之活動圖	35
圖 5-3-8 「會員訂單明細」之活動圖	36
圖 5-3-9 「員工查看會員訂單明細」之活動圖	37
圖 5-3-10 「結帳」之活動圖	38
圖 5-3-11 「開始計算時間」之活動圖	39
圖 5-3-12 「登出」之活動圖	39
圖 5-3-13 「儲值」之活動圖	40
圖 5-4-1 分析類別圖	41
圖 6-1-1 「登入及註冊」之循序圖	42
圖 6-1-2 「會員資料」之循序圖	43
圖 6-1-3 「員工資料」之循序圖	43

圖 6-1-4 「查看店家資訊」之循序圖	44
圖 6-1-5 「員工管理店家資訊之循序圖	44
圖 6-1-6 「點餐」之循序圖	45
圖 6-1-7 「員工管理菜單」之循序圖	45
圖 6-1-8 「查看餐點訂單」之循序圖	46
圖 6-1-9 「查看會員消費明細」之循序圖	46
圖 6-1-10 「開始計算時間」之循序圖	47
圖 6-1-11 「儲值」之循序圖	47
圖 6-1-12 「結帳」之循序圖	48
圖 6-1-13 「登出」之循序圖	48
圖 6-2-1 設計類別圖	49
圖 7-1-1 佈署圖	50
圖 7-2-1 套件圖	51
圖 7-3-1 「會員端」之元件圖	52
圖 7-3-2 「員工端」之元件圖	52
圖 7-4-1 狀態機	53
圖 8-1-1 資料庫關聯表	54

第一章 背景與動機

1-1 簡介

在這人手一機的時代，許多人利用手機做任何事包括食、衣、住、行、育、樂等，然而許多店家發現這項數位化的商機都紛紛將部分作業流程改為線上化的方式，讓顧客可以在任何地點進行點餐或是結帳的動作。

目前在台灣吹起了一波桌遊熱潮，許多民眾都透過台灣綜藝節目《娛樂百分百》中的一個單元《凹嗚狼人殺》影響，喜歡桌遊的人數與日俱增，托節目的福，許多桌遊店開始大受歡迎。

CatchU 是一個能簡化桌遊店家與顧客之間繁瑣程序的系統，結合了計算時間、點餐、結帳及店家管理的服務，我們將桌遊店家的計算時間、點餐、結帳及店家管理線上化，只要顧客及員工都有註冊會員或員工的身份，來光臨桌遊店的顧客就可以進行線上操作，店家也可以進行線上店家管理的操作。

桌遊店通常都以人工的方式計算遊玩的時間，我們便增加計算遊玩時間的功能，會員可直接透過手機計算遊戲時間，此用途不但增添了許多方便性更能減少店家人事成本。會員可以直接在座位上利用手機點餐，不需再走到櫃檯，員工也不必逐桌進行詢問；若是多人一起到店遊玩想要查看菜單也不必輪流翻閱，菜單改為線上化方式，縮短耗時的點餐流程。而我們的付款方式也有別於一般的店家，利用儲值點數的方式進行付款，會員只需到櫃檯進行儲值，在結帳時就可直接利用手機扣除點數，不必帶現金或是信用卡，只需帶一支手機即可出門。

而店家可以在線上設計菜單、收費標準、以及設置店家資訊，也可以得知會員是否結帳並可以看到當日所有會員的消費明細。

我們希望能在讓業者方便之餘也帶動桌遊業的風氣，隨著科技網絡的發展，愈來愈多的桌遊線上化，遊戲已慢慢失去了最初桌遊所帶給人的目的及價值，桌遊意指桌上遊戲，更深層的意義便是人與人的互動與接觸，我們希望透過「CatchU 桌助你」所帶來的方便性，能吸引更多的桌遊玩家面對面的玩遊戲，此即「CatchU 桌助你」所誕生的意義及願景。

1-2 問題與機會

S(優勢)	W(劣勢)
<ol style="list-style-type: none"> 1. 所有流程可以線上操作，減少人多時造成的混亂，避免交易糾紛，也可以提升會員在桌遊店遊玩的體驗。 2. 介面清晰簡單，不會過於複雜，新進員工容易上手。 	<ol style="list-style-type: none"> 1. 假如店家的位置如地下室或信號不好區域，又沒有提供 WIFI 則可能不行使用 CatchU。
O(機會)	T(威脅)
<ol style="list-style-type: none"> 1. 雖然網咖系統與我們相似，在是行動裝置可以使用的靈活性較高。 2. 大部分桌遊店規模不是很大，不需要太複雜的系統，且支付不了那麼高昂的費用，則可以選擇 CatchU。 	<ol style="list-style-type: none"> 1. 許多線上桌遊也很盛行，可能導致實體店桌遊店風氣退去之後，而無法使用我們的系統。 2. 如果桌遊店還有提供餐點及計時以外的服務的話，則可能超出系統的功能範圍。

表 1-2-1 SWOT 分析表

1-3 相關系統探討

	CatchU 桌助你	網咖管理系統
登入	需註冊帳號（手機號碼）	需註冊帳號（手機號碼）
使用方式	用行動裝置進入網頁	使用電腦進入管理系統
線上點餐	有	有
計時服務	有	有
付款方式	儲值點數	儲值點數
餐點付款方式	合併到遊玩時數的金額裡	另外給點家現金或刷卡
得知會員座位	因為人會移動所以無法	電腦座位是固定所以可以
系統規模	較簡易	較複雜

表 1-3-1 相關系統探討比較表

第二章 系統目標與預期成果

2-1 系統目標

為了減少店家的負擔，會員能夠透過線上的方式進行計時、結帳及點餐，店家也能線上制定菜單、收費標準、店家詳細資訊、查看會員消費明細，藉此我們希望能夠增加店家在每項作業上的效率。

本組目標如下：

會員端

一、計算遊戲時間

每位會員到達桌遊店的時間及開始遊玩的時間都不同，我們可以透過線上計時自己本次的遊玩時長，就不必由店家人工計算，可精準計算出使用者遊玩時間，來減少錯誤率發生，更能避免多餘的人力成本。

二、線上點餐

在遊戲過程中，會員不必到櫃檯點餐，可透過行動裝置進行線上點餐的動作，點餐後店家隨即收到訂單詳細資訊，接著出餐給使用者。若是多人一起到店遊玩想要查看菜單也不必輪流翻閱，不但可以省去翻看紙本菜單的時間，更能縮短耗時的點餐流程。

三、手機結帳

當會員結束遊玩，可已透過行動裝置進行線上結帳，不必到櫃台排隊。當次消費餐飲及遊玩時間也會在結帳頁面一併看到。

四、手機付款

會員可到櫃台先儲值金額，結帳時就可直接在手機上扣除儲值金，不必用現金結帳，會員只需帶一支手機也能出門，不但方便且減少許多不必要的時間。

員工端

五、店家管理

店家可以在線上自訂菜單、店家資訊、收費標準，不必再像以前寫在小黑板擺在門口，也可以看到哪些會員還沒結帳，減少人多時的結帳亂象，並可以查看當日的結帳明細，避免交易的糾紛發生。

2-2 預期成果

本組預期成果如下：

會員端

- 一、利用線上的計時的功能，來計時自己本次的遊玩時長。
- 二、透過線上點餐不但可以幫助店家省去多餘的人力成本，還能讓會員不必到櫃檯點餐，直接透過行動裝置進行點餐的動作。
- 三、利用結帳功能，在遊戲結束後，會員不需再至櫃台排隊等候結帳，可直接透過行動裝置結帳。
- 四、透過儲值點數，付款不再侷限於現金或信用卡。

員工端

- 五、店家可以在線上制定菜單、店家詳細資訊、收費標準，所有資訊可以集中，也可以看到訂單及會員消費記錄，避免交易糾紛。

第三章 系統規格

3-1 系統架構

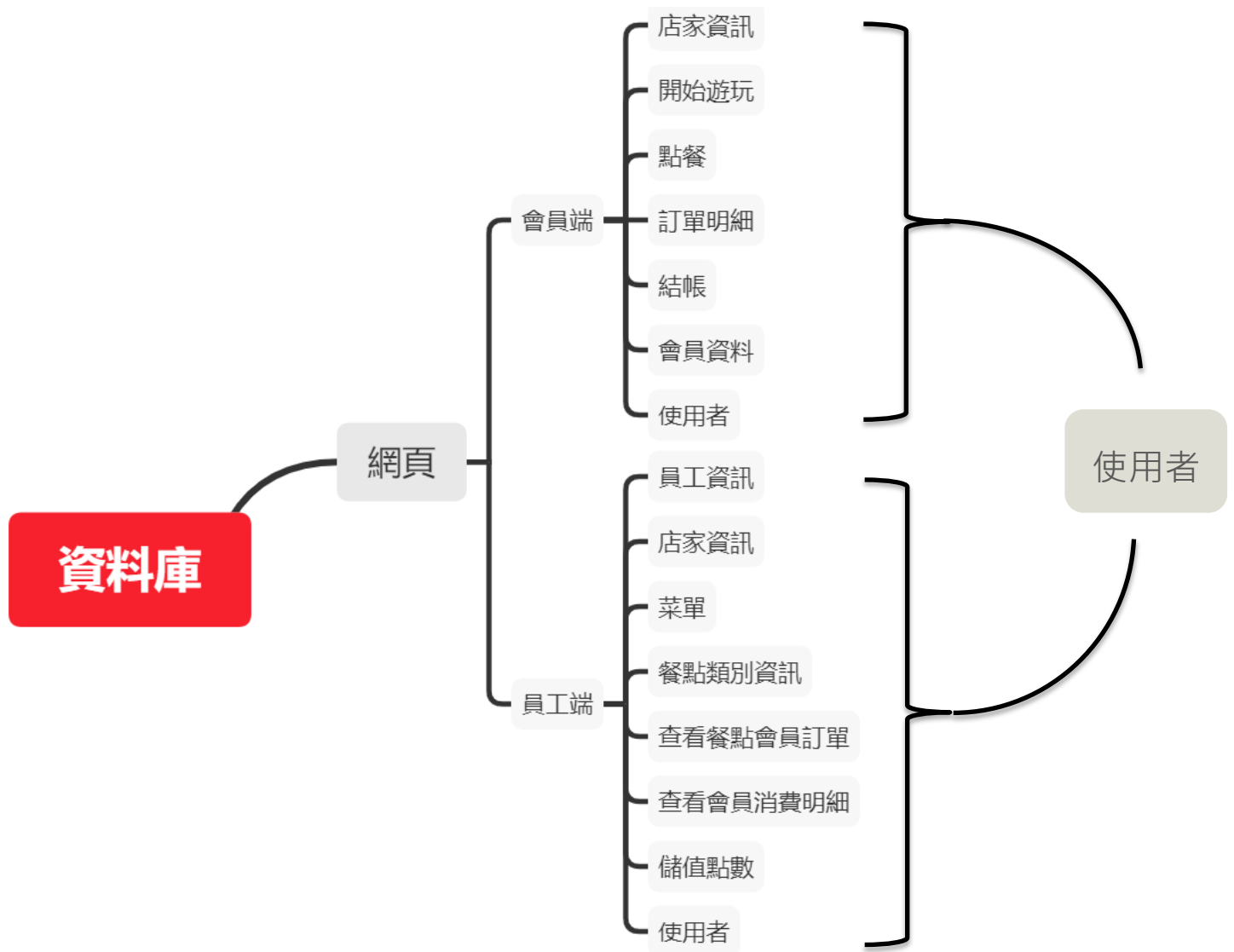


圖 3-1-1 系統架構圖

3-2 系統軟、硬體需求技術平台

表 3-2-1 伺服器端規格表

伺服器端規格	
程式語言	JavaScript
整合式開發環境(IDE)	Visual Studio Code、Visual Basic
開發環境	Windows 10、MacOS 10.14
框架	Node.js
資料庫工具	PostgreSQL
伺服器	Heroku

表 3-2-2 網站端規格表

網站端規格	
開發語言	HTML 5、CSS、jQuery、JavaScript
整合式開發環境(IDE)	Visual Studio Code、Visual Basic、Adobe Xd

表 3-2-3 系統硬體需求表

系統硬體需求表	
作業系統	Android、iOS
使用裝置	智慧型手機、個人電腦
無線傳輸	3G/4G/5G、Wi-Fi、行動網路

3-3 使用標準與工具

表 3-3-1 使用標準與工具表

伺服器端規格	
整合式開發環境(IDE)	Visual Studio Code
資料庫管理工具	Navicat Premium Essentials 12
版本控制工具	Windows 10、MacOS 10.14
文件撰寫工具	
文件製作	Word 2007/2016、Google 文件
簡報製作	PowerPoint 2016
UML 工具	Visual Paradigm for UML
專案管理工具	
溝通工具	Line、Google Meet

第四章 專案時程與組織分工

4-1 專案時程

表 4-1-1 專案時程表

時間 事項	2020 年										
	1 月	2 月	3 月	4 月	5 月	6 月	7 月	8 月	9 月	10 月	11 月
主題訂定											
資料蒐集											
技術學習											
資料庫建置											
資料庫設計											
前端程式撰寫											
會員後端程式撰寫											
員工後端程式撰寫											
前後端連接											
文件製作											

簡報製作											
影片製作											
海報製作											
名牌製作											
Logo 設計											

4-2 專案組織與分工表

4-2-1 專案分工表

表中 ★:主要負責人，☆:次要負責人

負責人 工作分配	10646037 關宇辰	N1066418 周育德	N1066432 林琪容	N1066442 葉芝秀
主題訂定	★	★	★	★
資料蒐集		★	★	★
技術學習		★	★	★
資料庫建置		★	☆	★
資料庫設計		★	☆	★
前端程式撰寫			★	
會員後端程式撰寫				★
員工後端程式撰寫		★		
前後端連接			★	
文件製作		★	☆	★
簡報製作		★	★	★
影片製作		★	☆	☆
海報製作		☆	★	☆
名牌製作	★	★	☆	★
Logo 設計		☆	★	★

第 5 章 需求模型

5-1 使用者需求

功能性需求：

- 1.會員透過網站進行登入/註冊會員身份、查看店家資訊、線上計時、點餐、查看訂單明細、結帳、管理會員資料。
- 2.員工透過網站登入/註冊員工身份、管理員工資料、管理菜單、替會員儲值、管理店家資訊，查看餐點訂單、查看會員消費明細。
- 3.管理端透過資料庫管理達成會員及員工的需求。

非功能性需求：

- 1.作業系統：不限。
- 2.裝置需求：支援 Wi-Fi/3G/4G 網路。

5-2 使用者個案圖(Use case diagram)

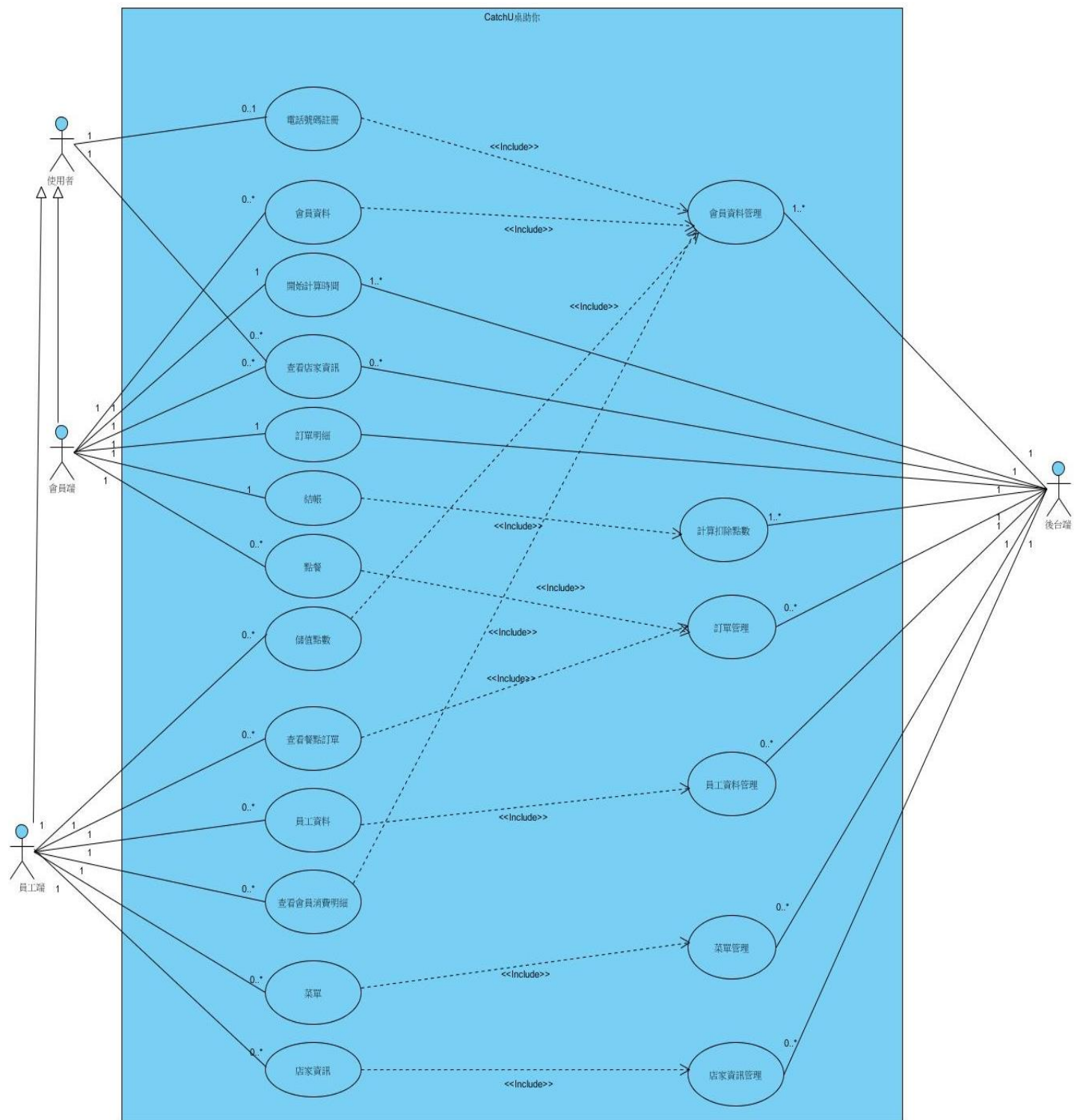


圖 5-2-1 使用者個案圖

5-3 使用個案描述

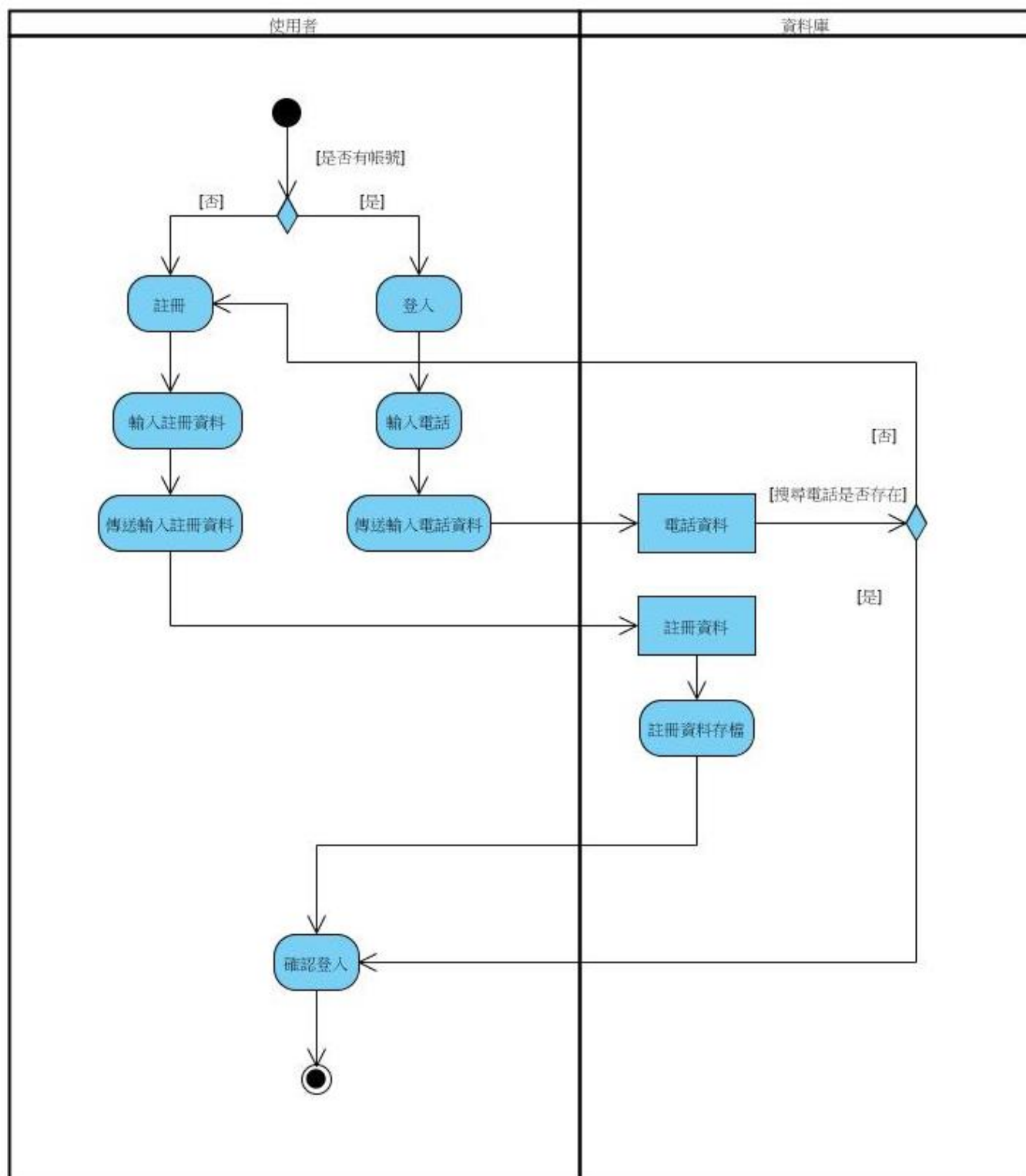


圖 5-3-1 「登入及註冊」之活動圖

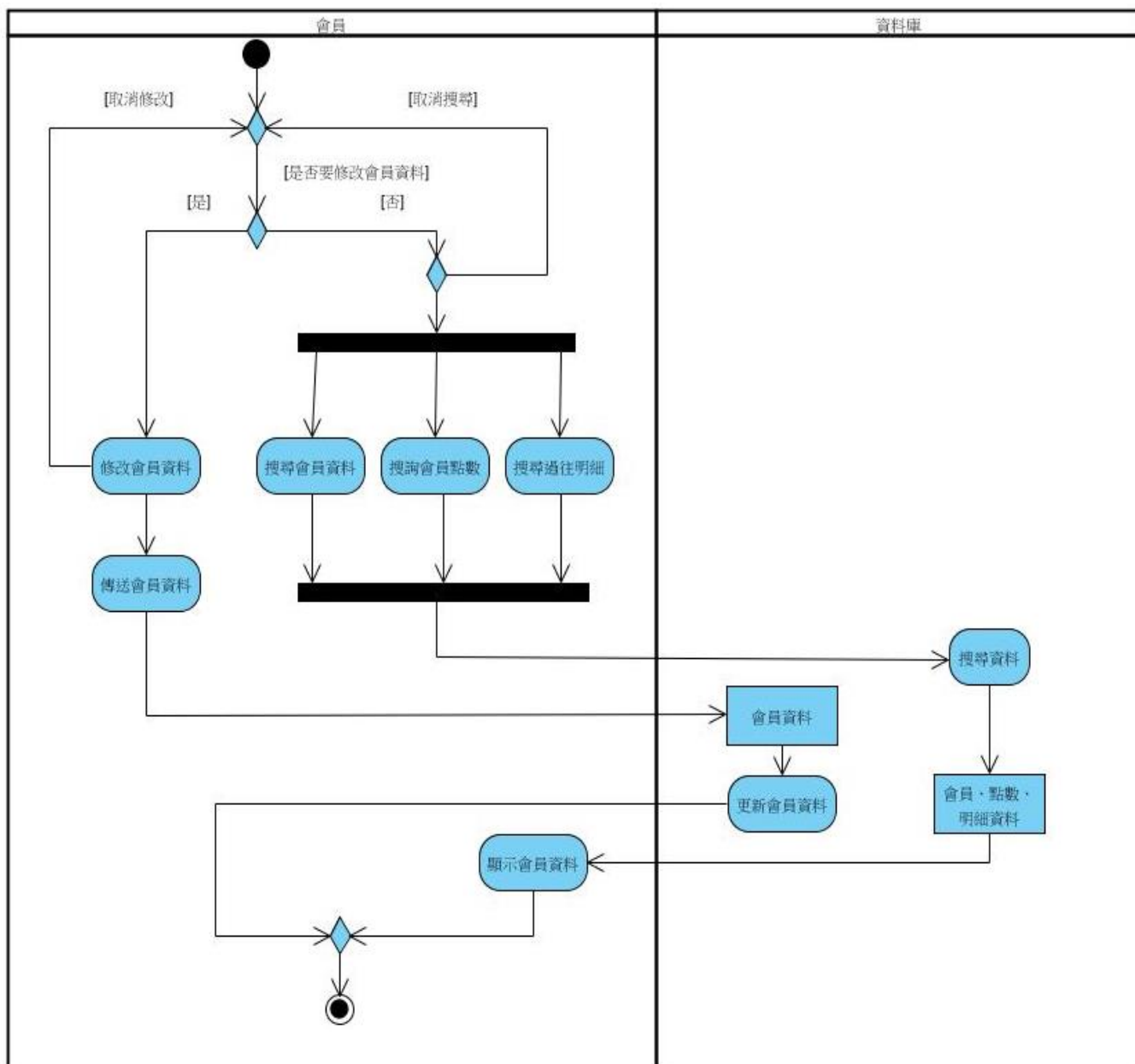


圖 5-3-2 「會員資料」之活動圖

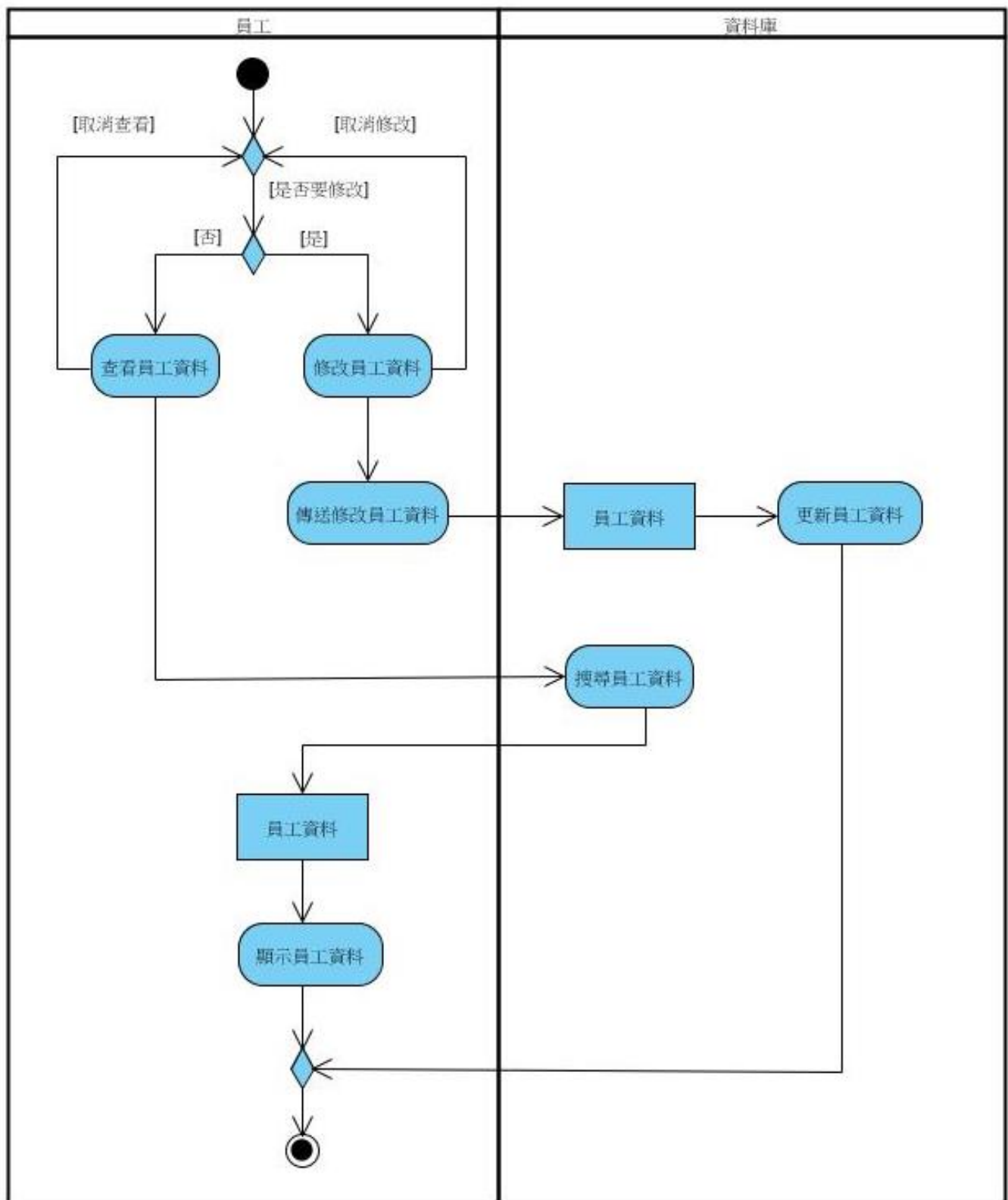


圖 5-3-3 「員工資料」之活動圖

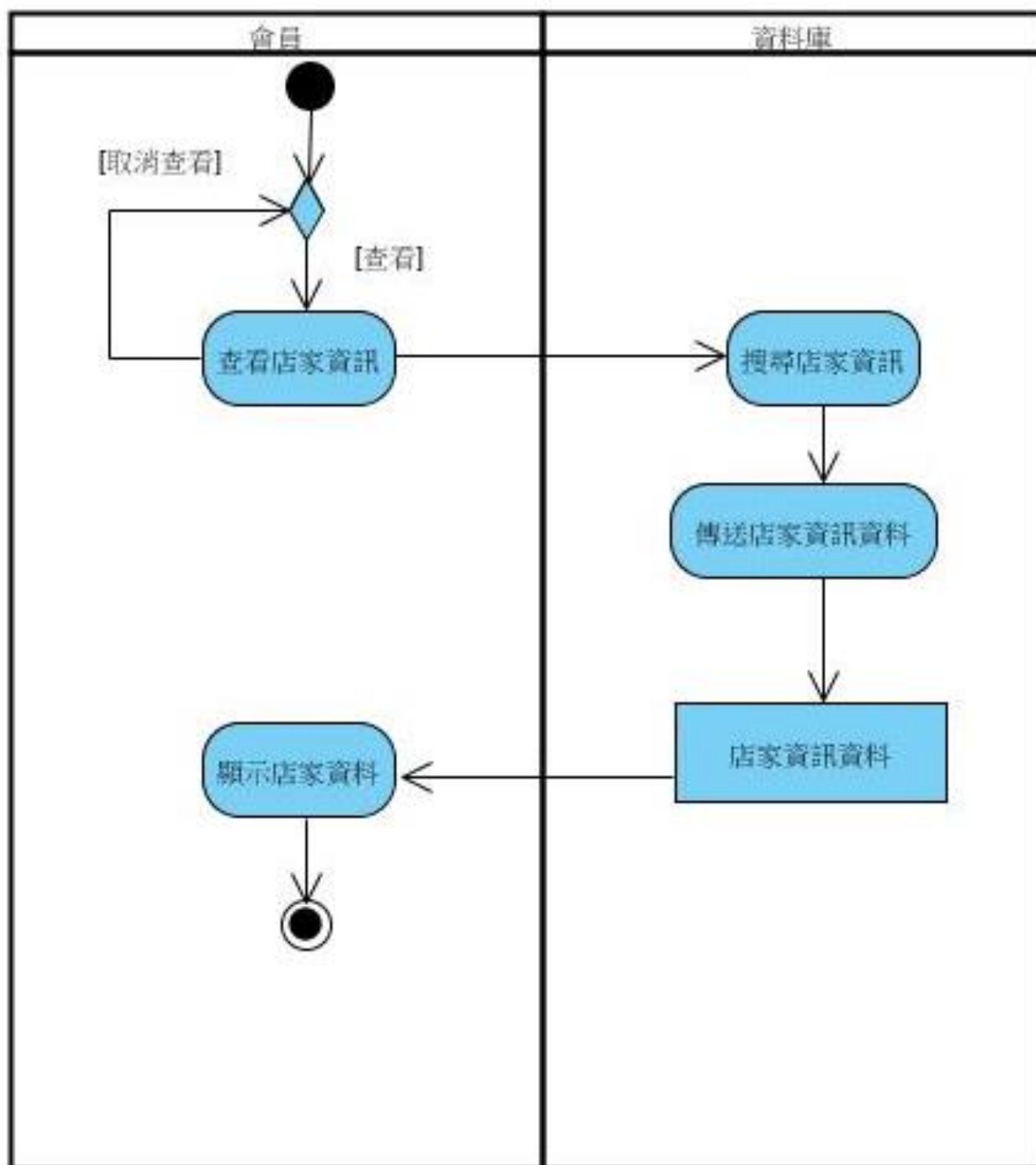


圖 5-3-4 「會員查看店家資訊」之活動圖

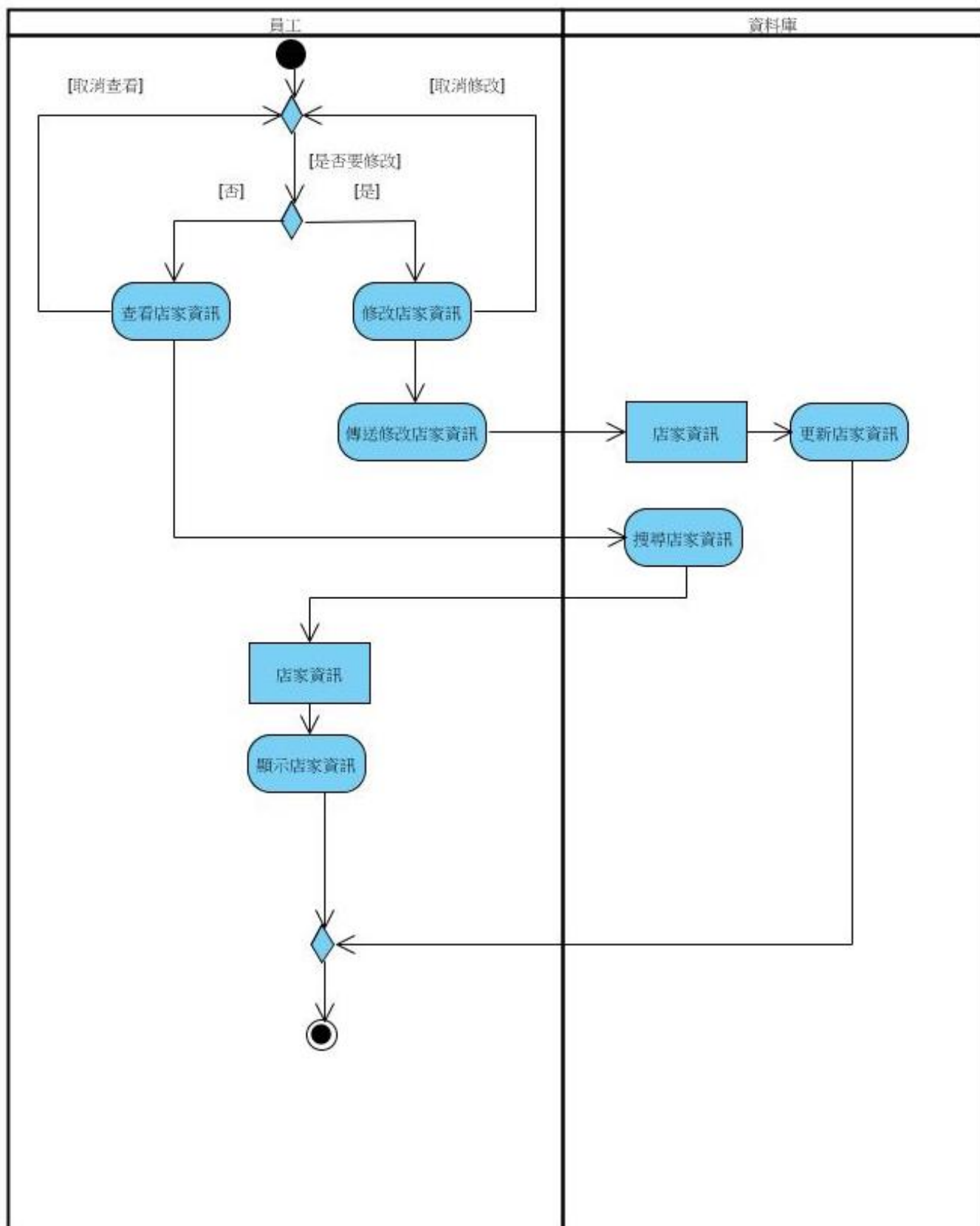


圖 5-3-5 「員工管理店家資訊」之活動圖

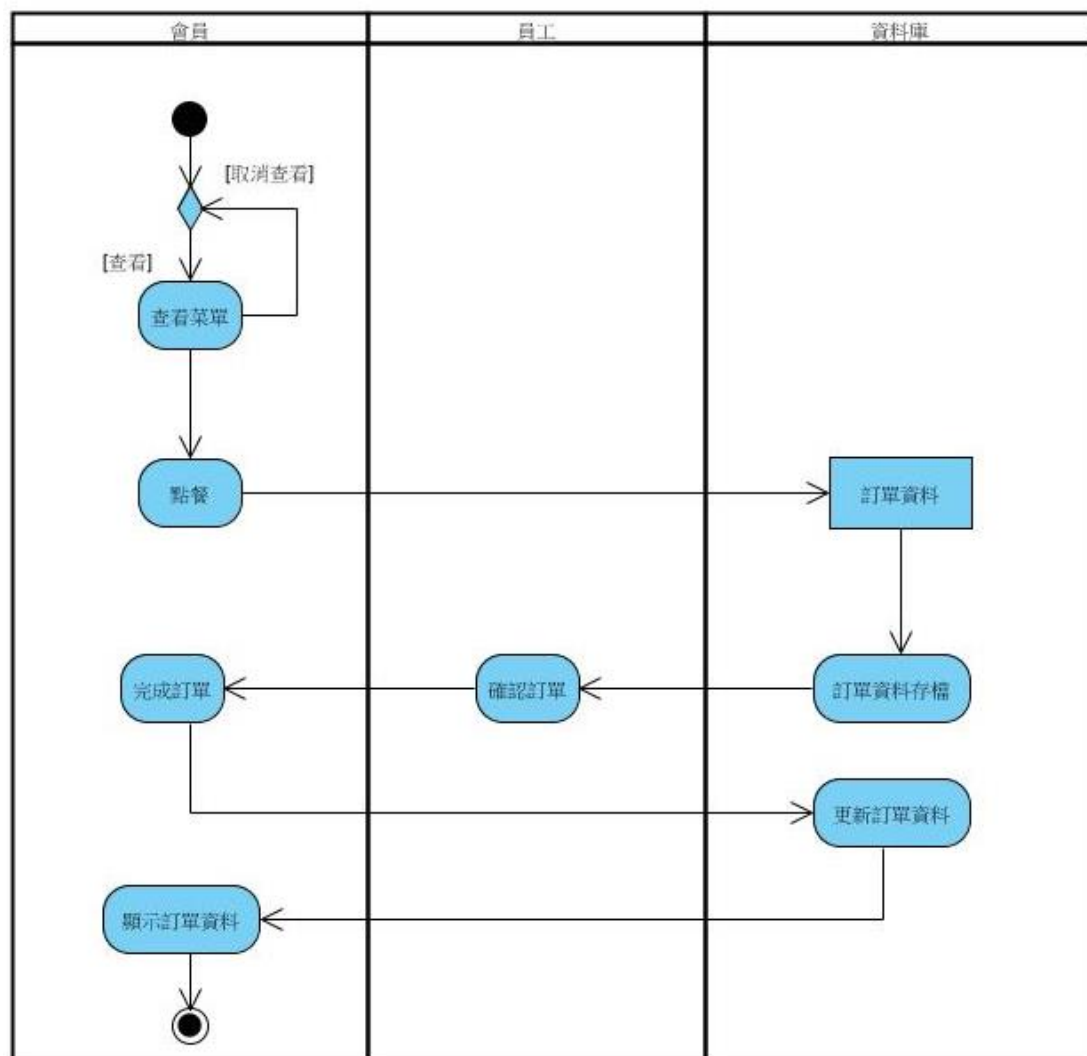


圖 5-3-6 「會員點餐」之活動圖

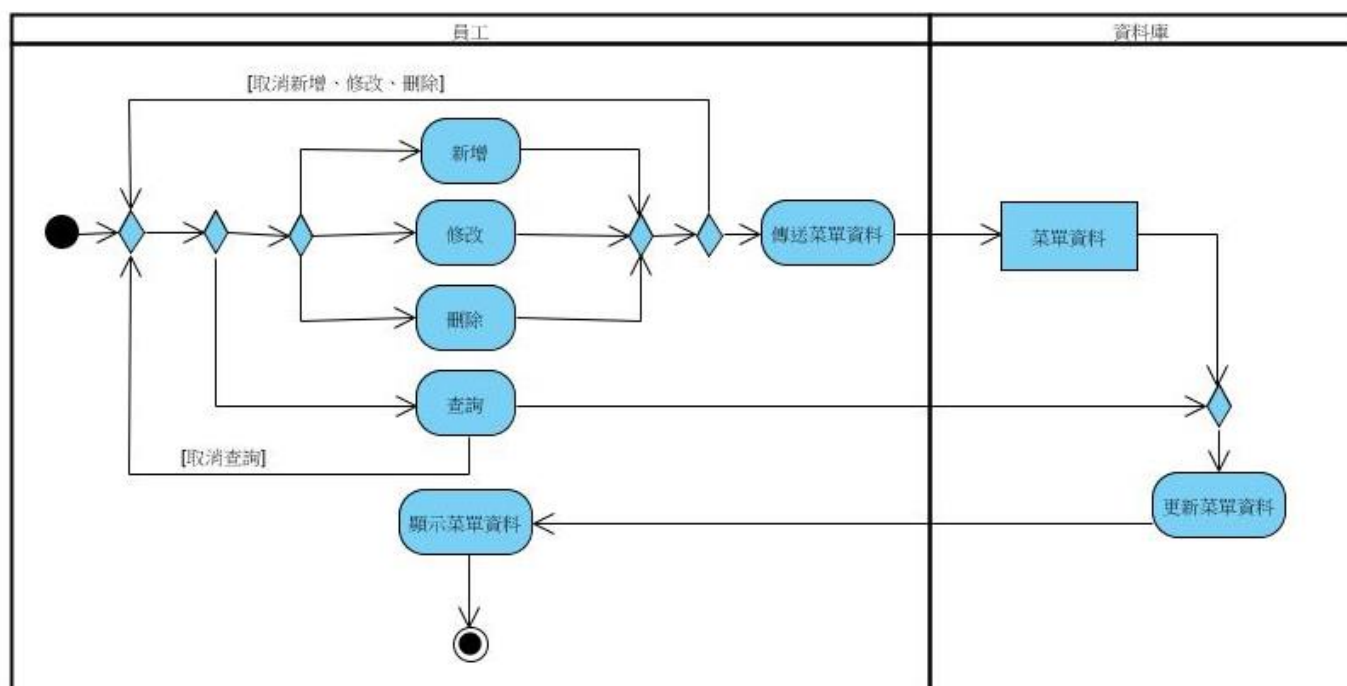


圖 5-3-7 「員工管理菜單」之活動圖

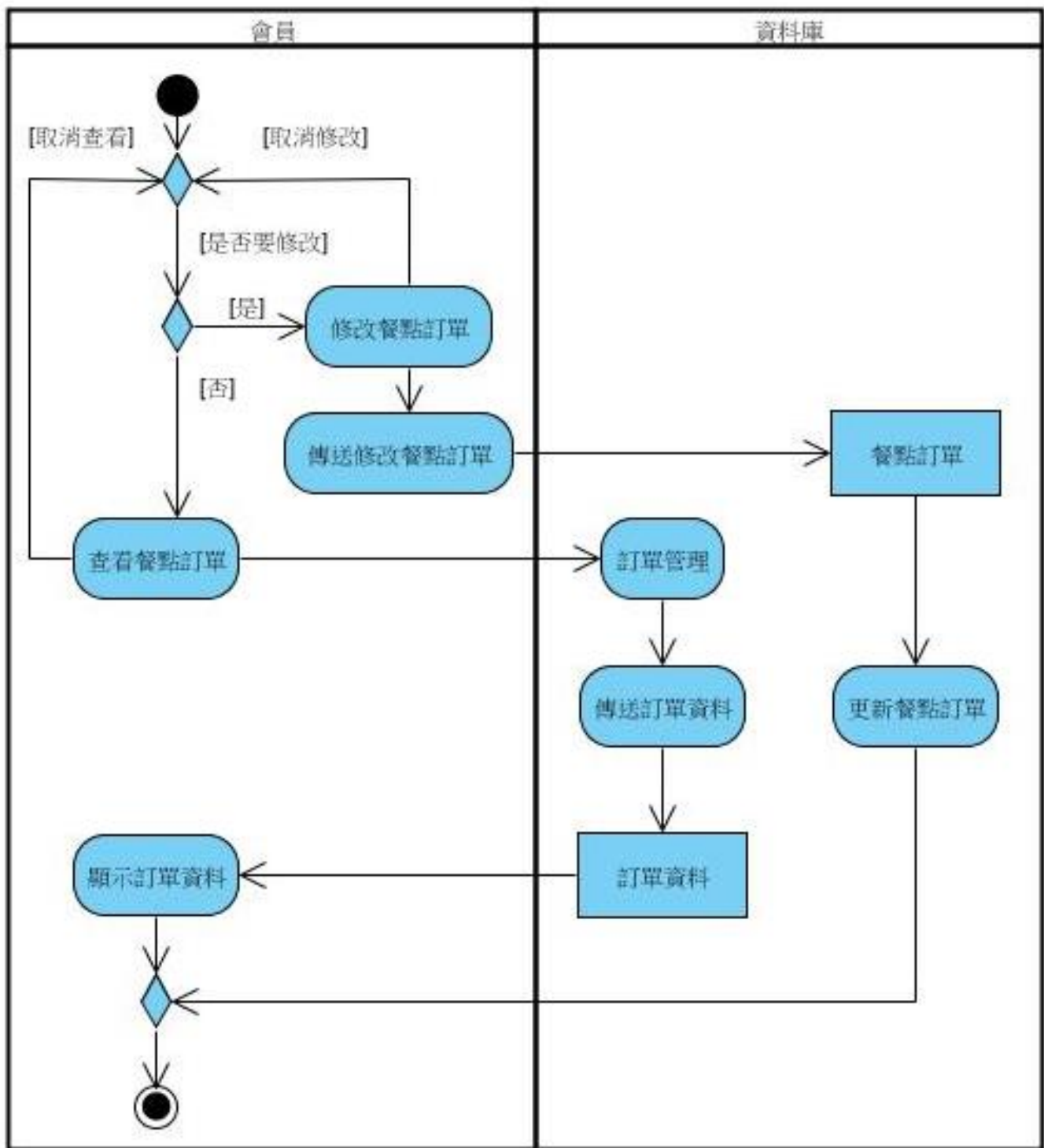


圖 5-3-8 「會員訂單明細」之活動圖

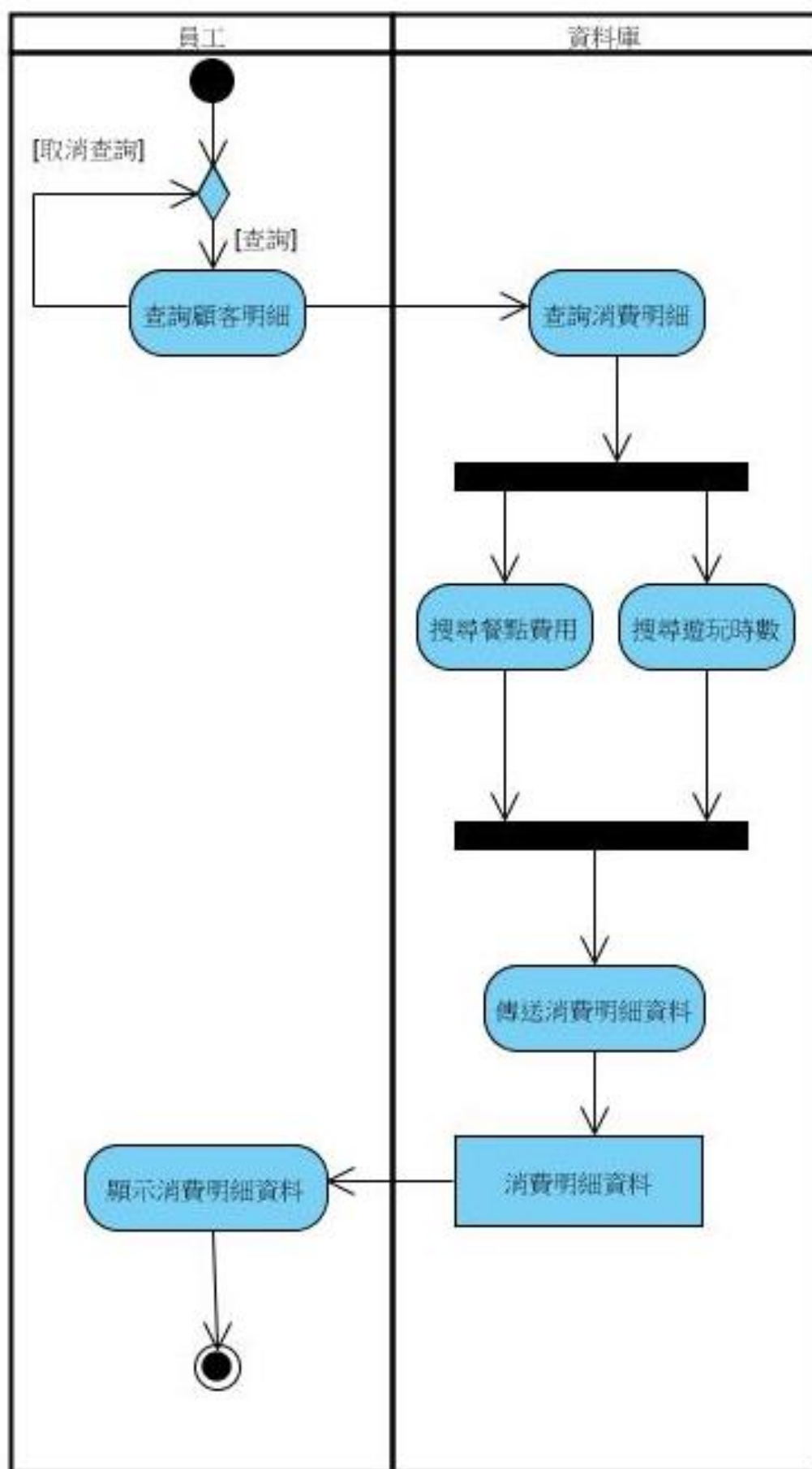


圖 5-3-9 「員工查看會員訂單明細」之活動圖

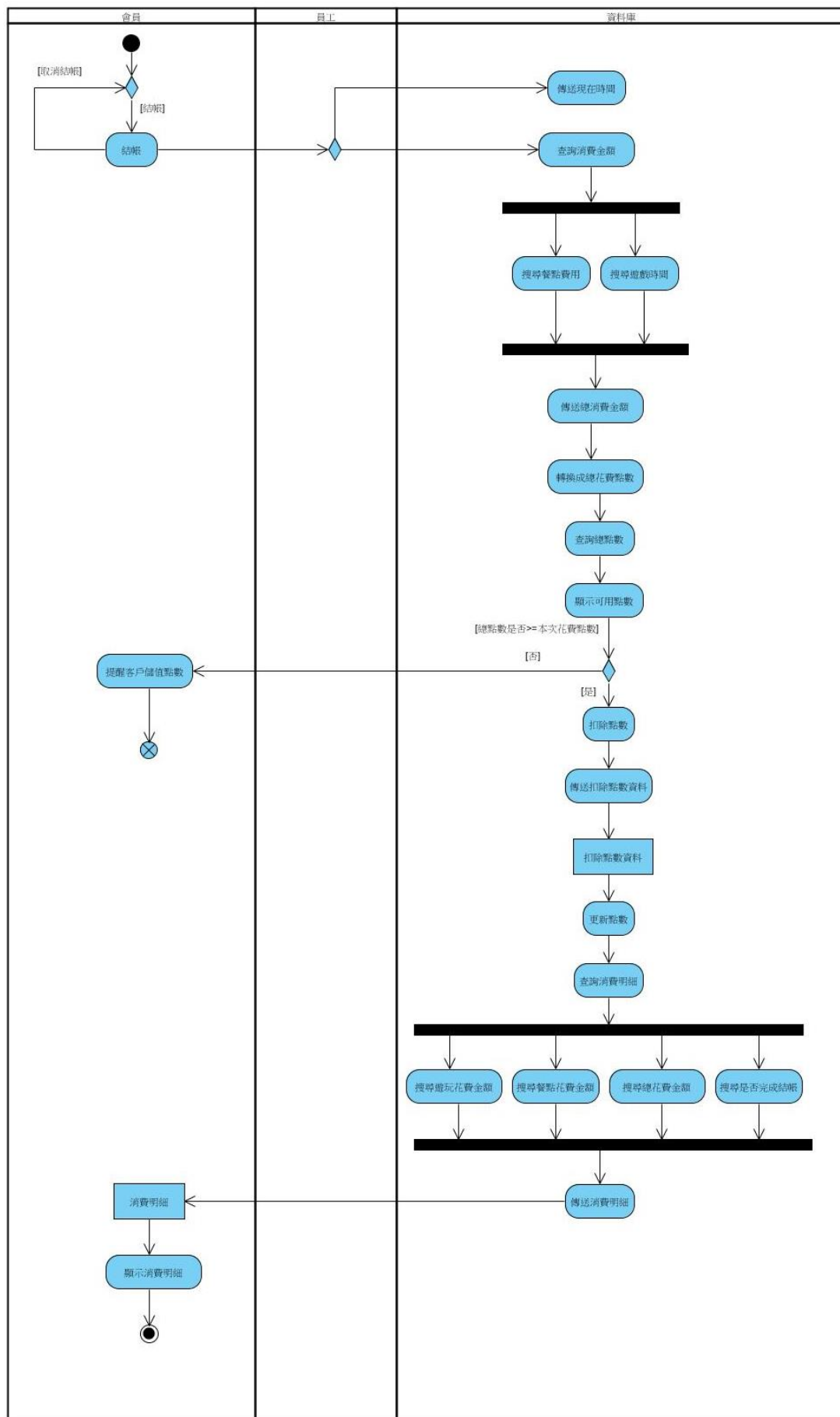


圖 5-3-10 「結帳」之活動圖

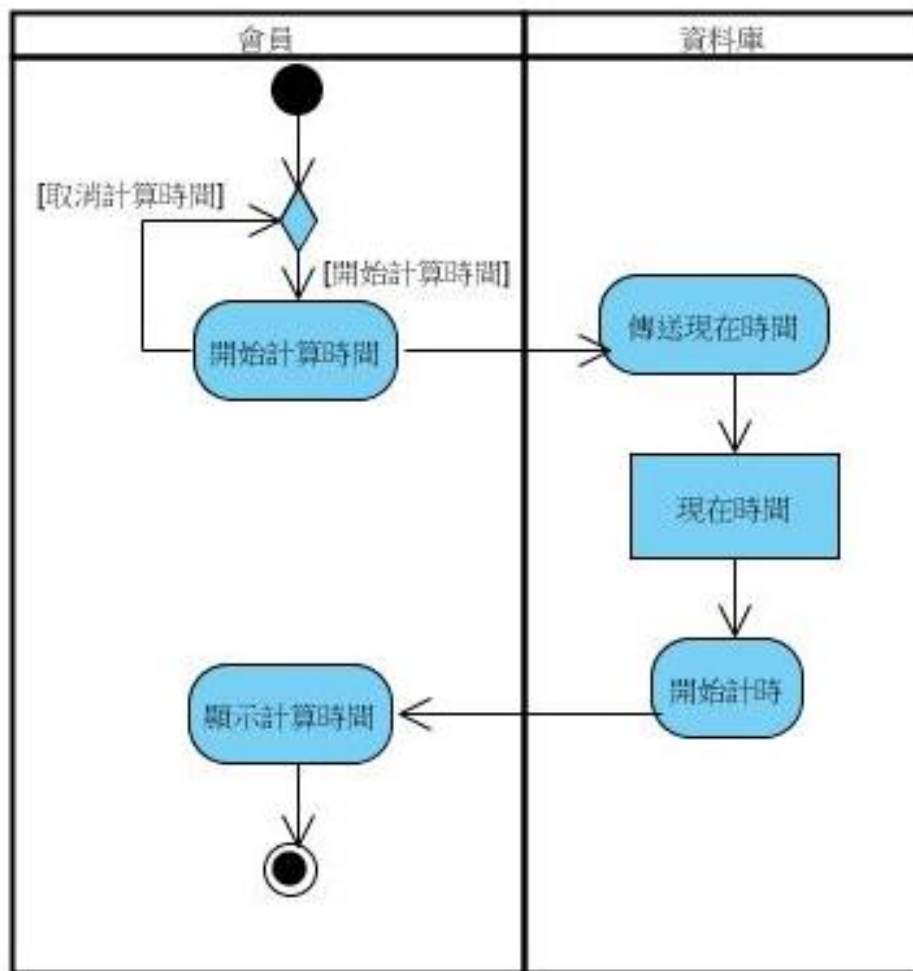


圖 5-3-11 「開始計算時間」之活動圖

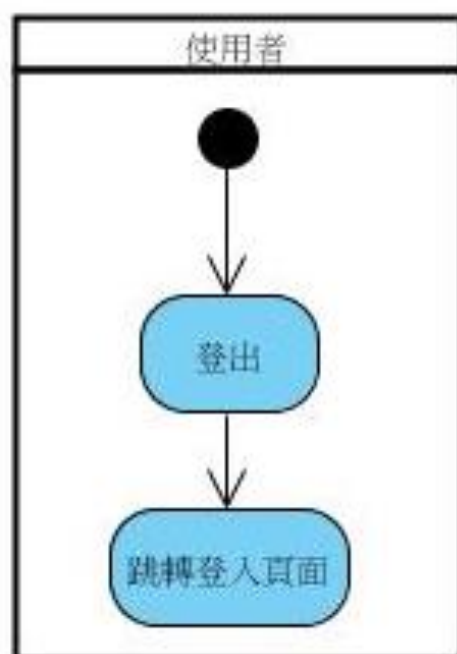


圖 5-3-12 「登出」之活動圖

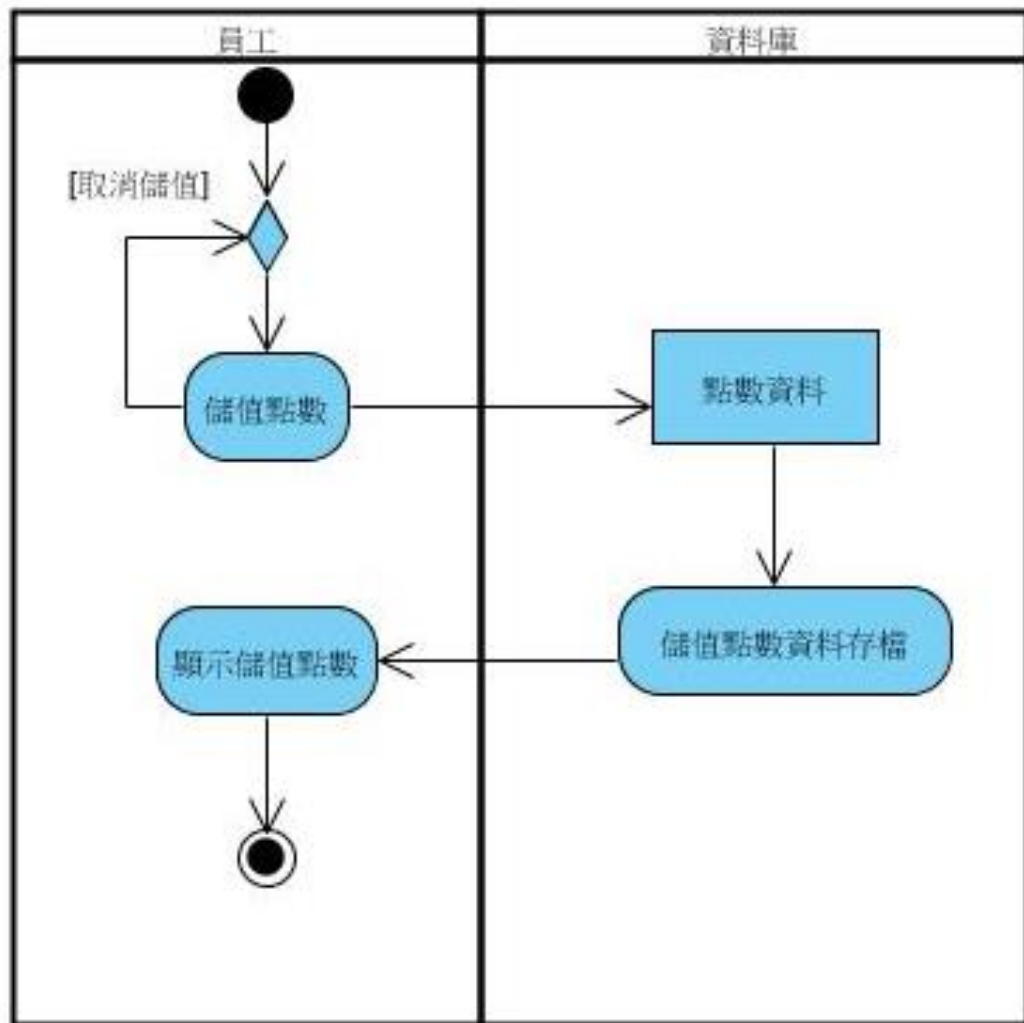


圖 5-3-13 「儲值」之活動圖

5-4 分析類別圖(Analysis class diagram)

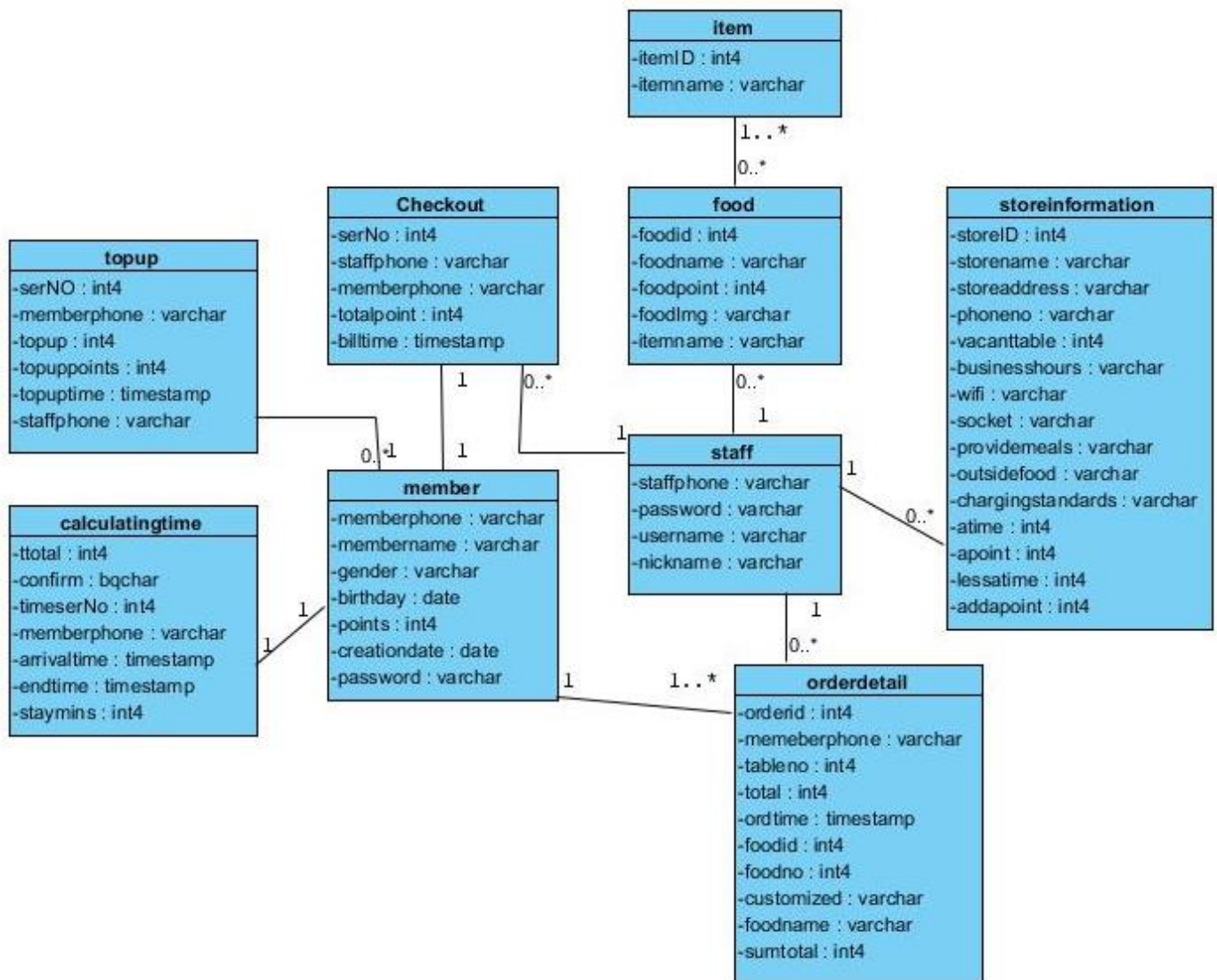


圖 5-4-1 分析類別圖

第 6 章設計模型

6-1 循序圖(Sequential diagram)

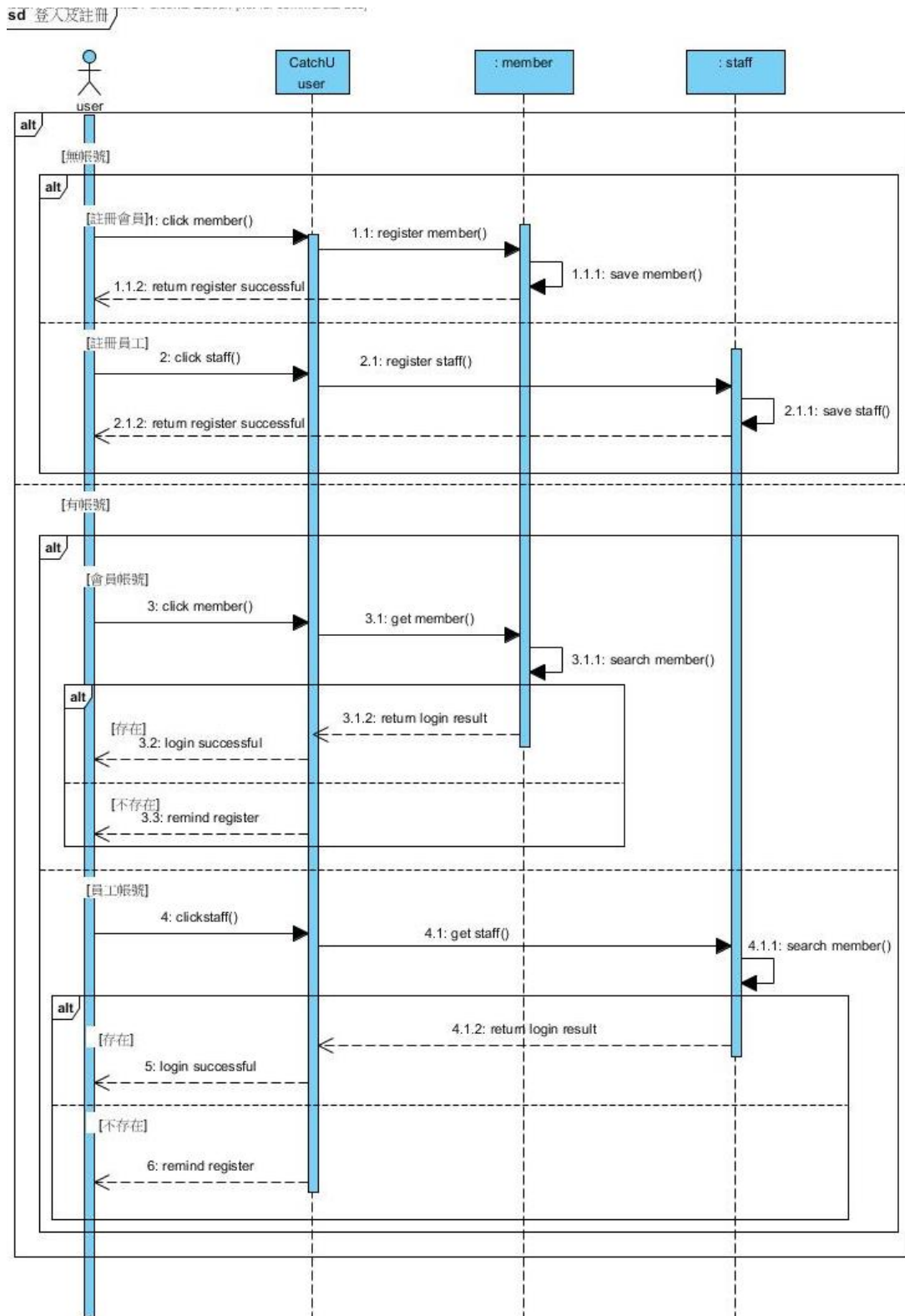


圖 6-1-1 「登入及註冊」之循序圖

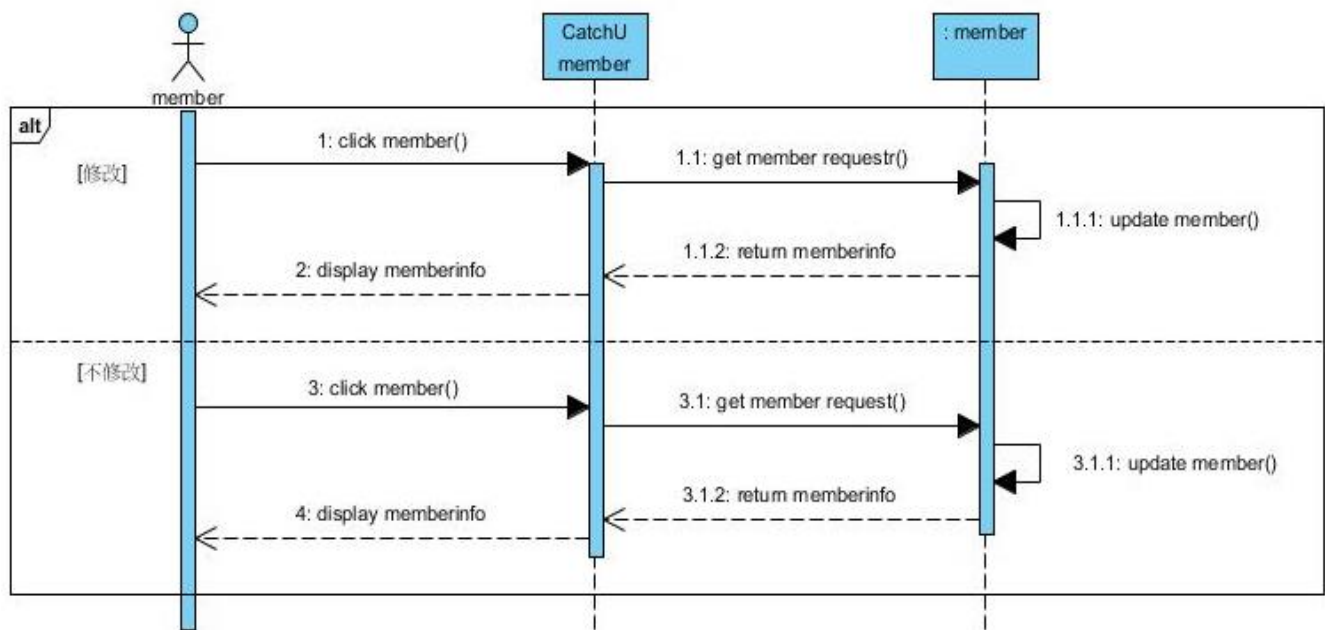


圖 6-1-2 「會員資料」之循序圖

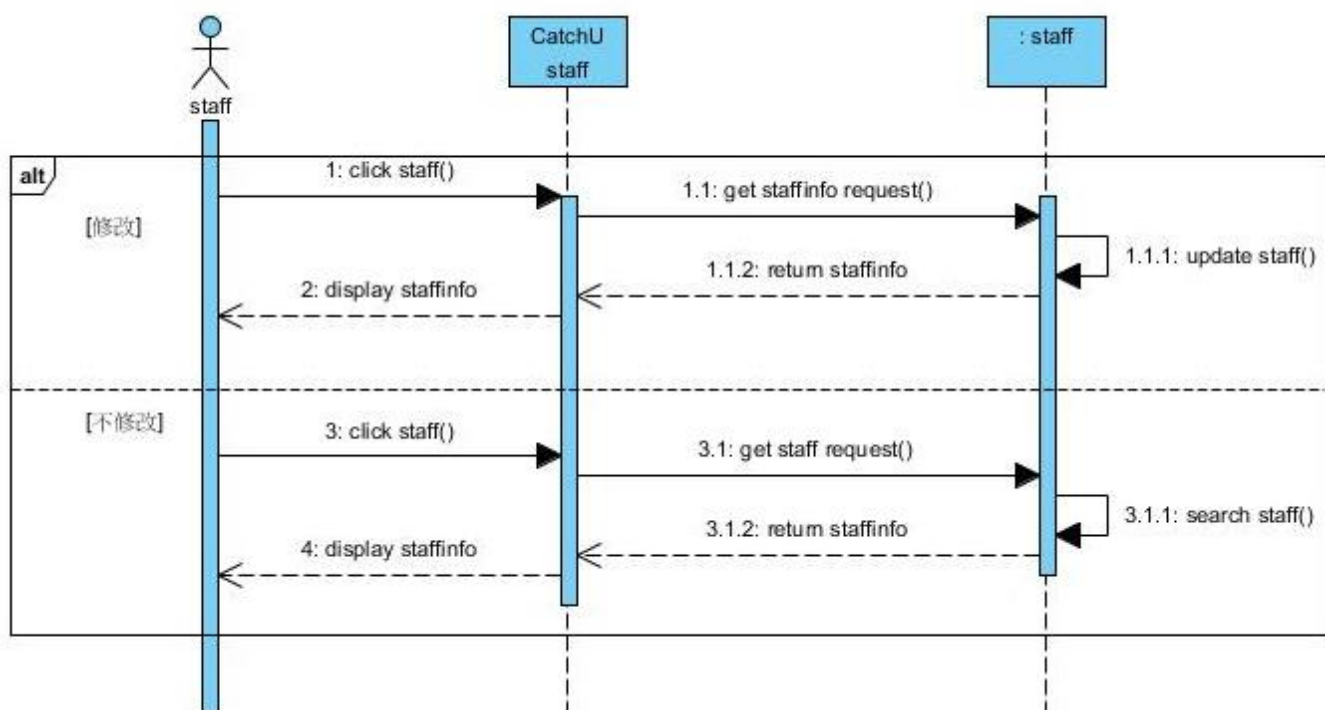


圖 6-1-3 「員工資料」之循序圖

sd 店家資訊

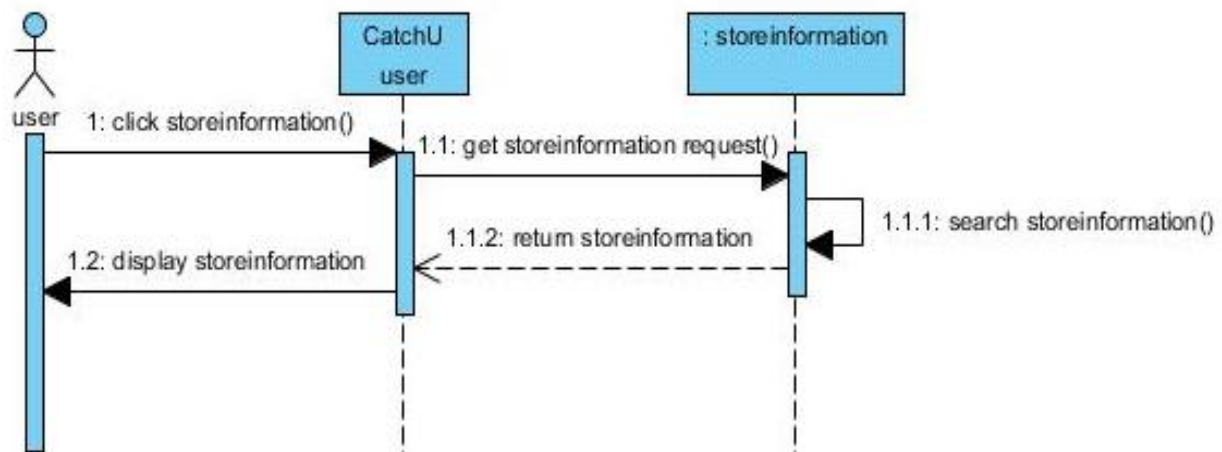


圖 6-1-4 「查看店家資訊」之循序圖

sd 員工店家資訊管理

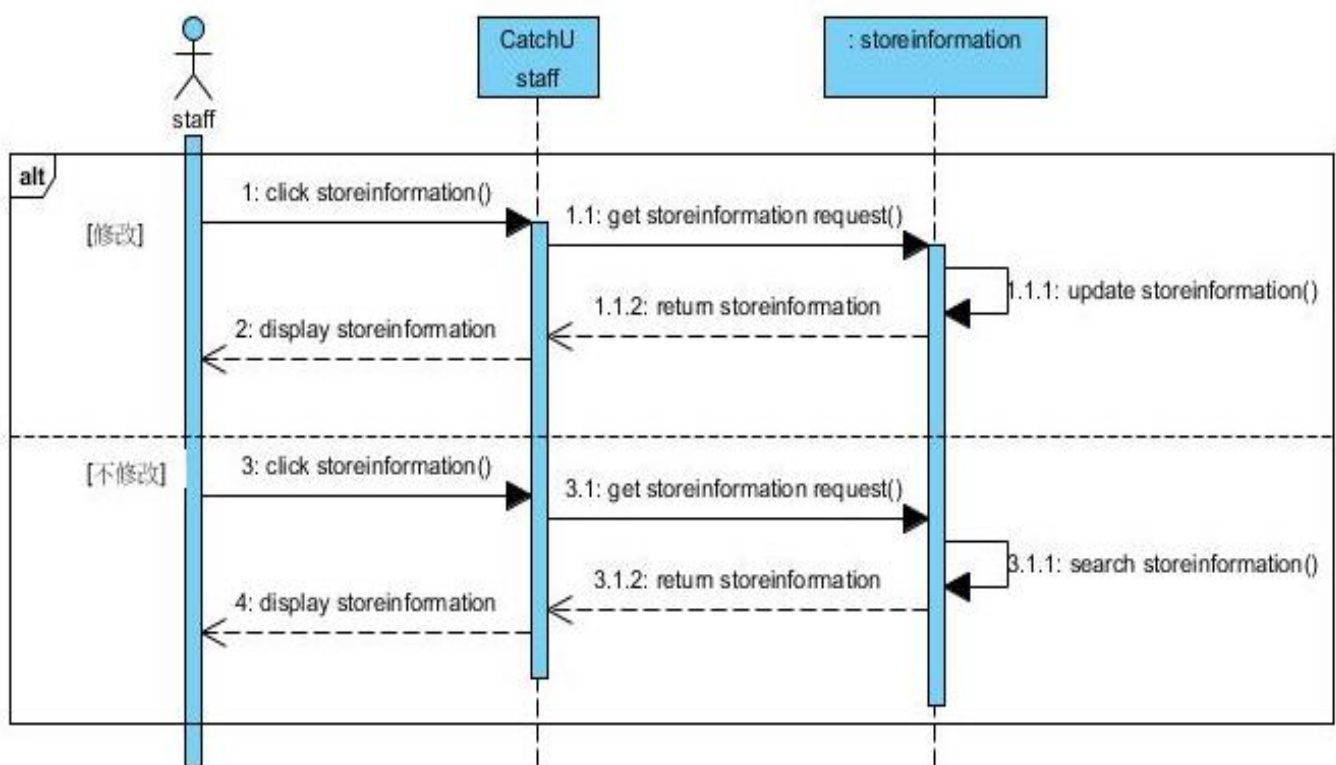


圖 6-1-5 「員工管理店家資訊」之循序圖

sd 點餐

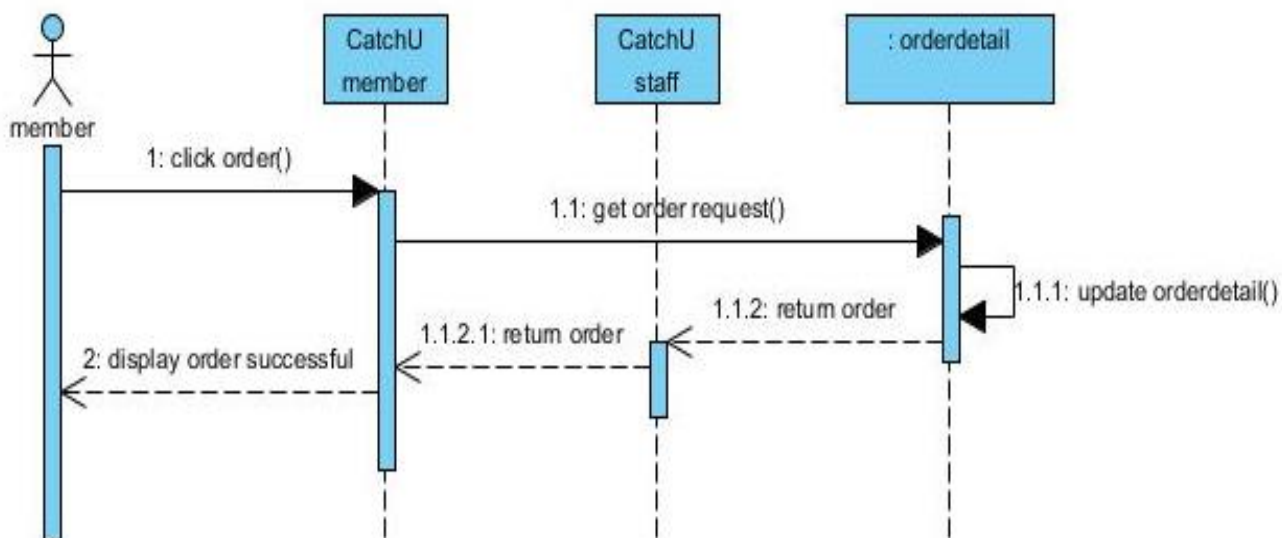


圖 6-1-6 「點餐」之循序圖

sd 員工菜單管理

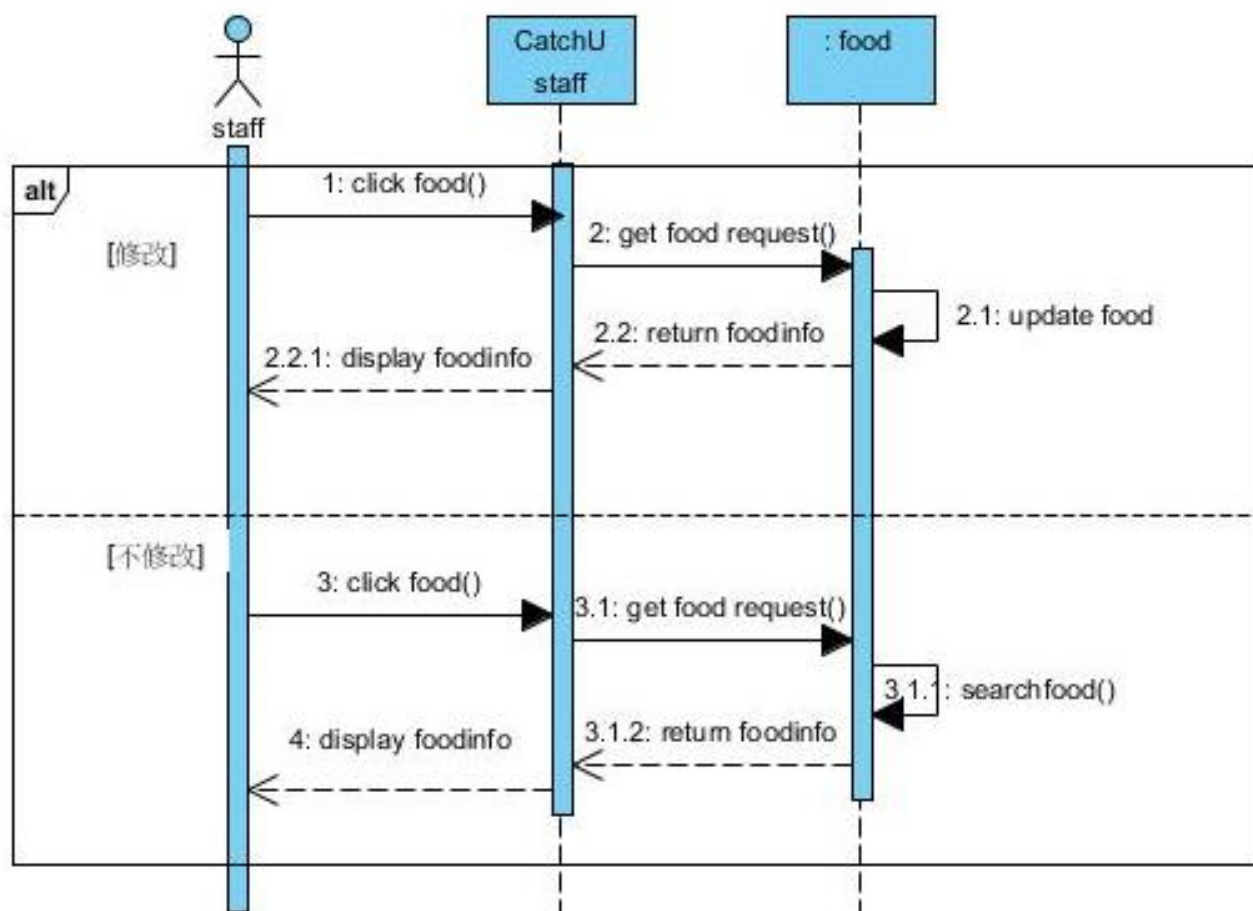


圖 6-1-7 「員工管理菜單」之循序圖

sd 查看餐點訂單

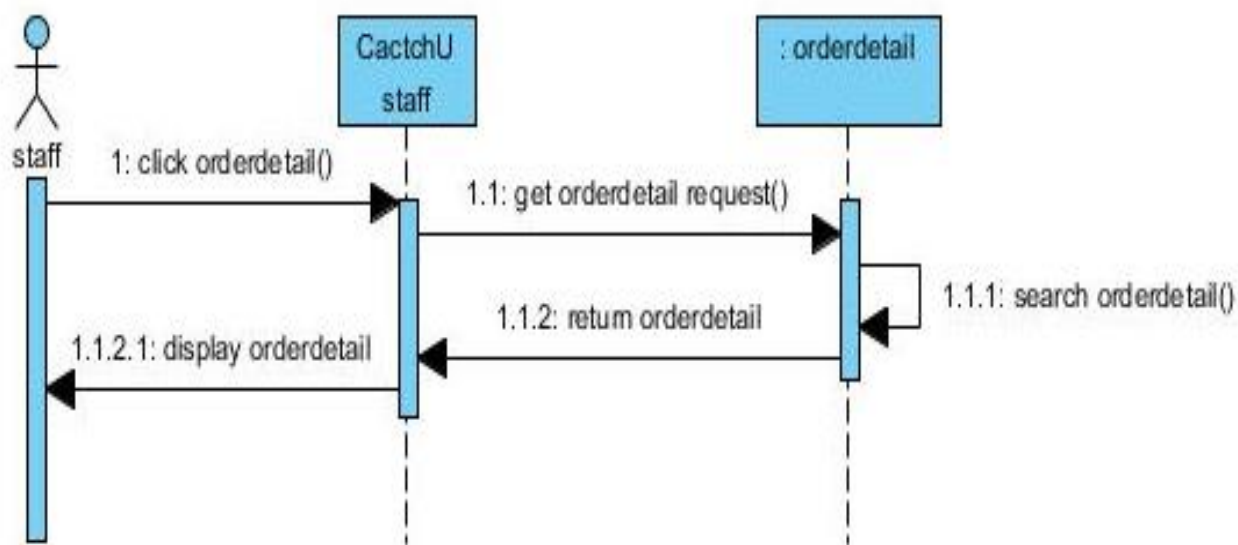


圖 6-1-8 「查看餐點訂單」之循序圖

sd 查看會員消費明細

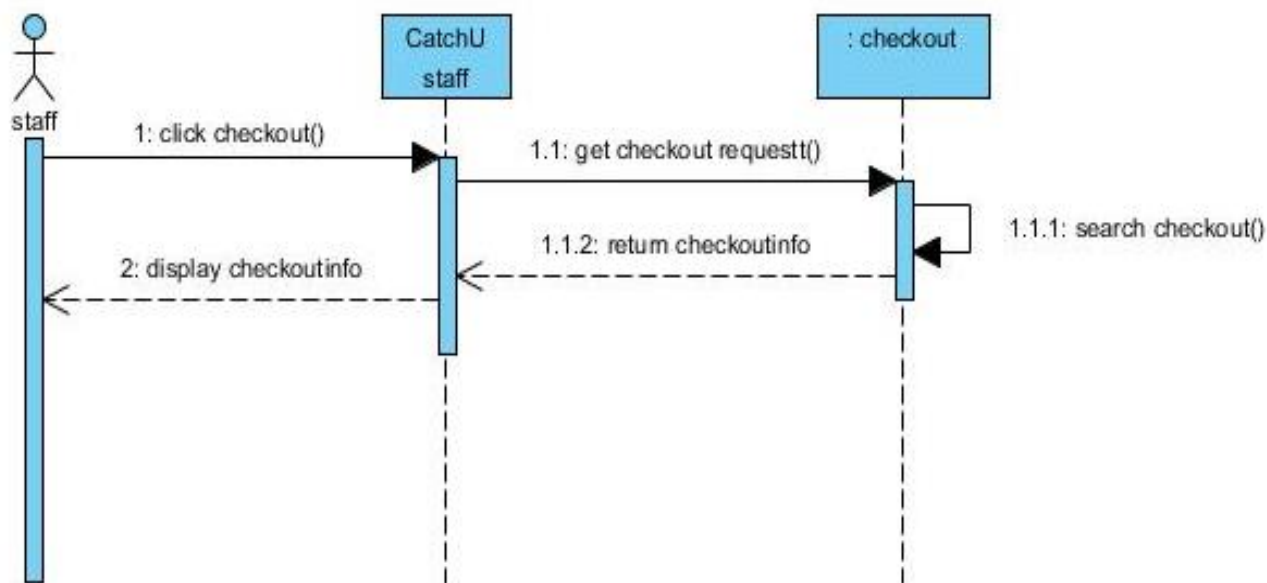


圖 6-1-9 「查看會員消費明細」之循序圖

sd 開始計算時間

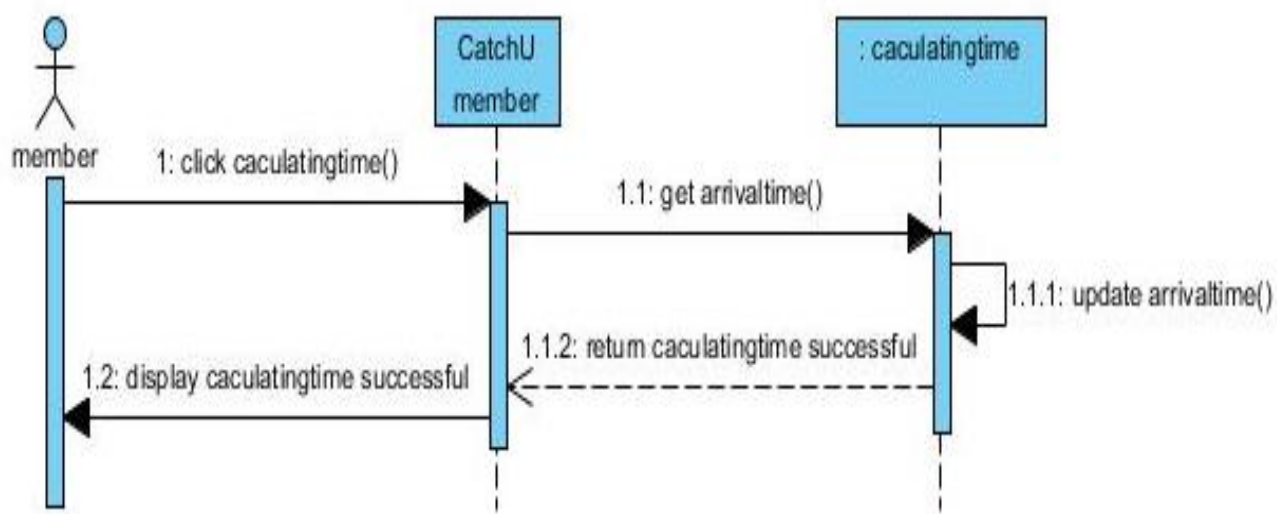


圖 6-1-10 「開始計算時間」之循序圖

sd 儲值

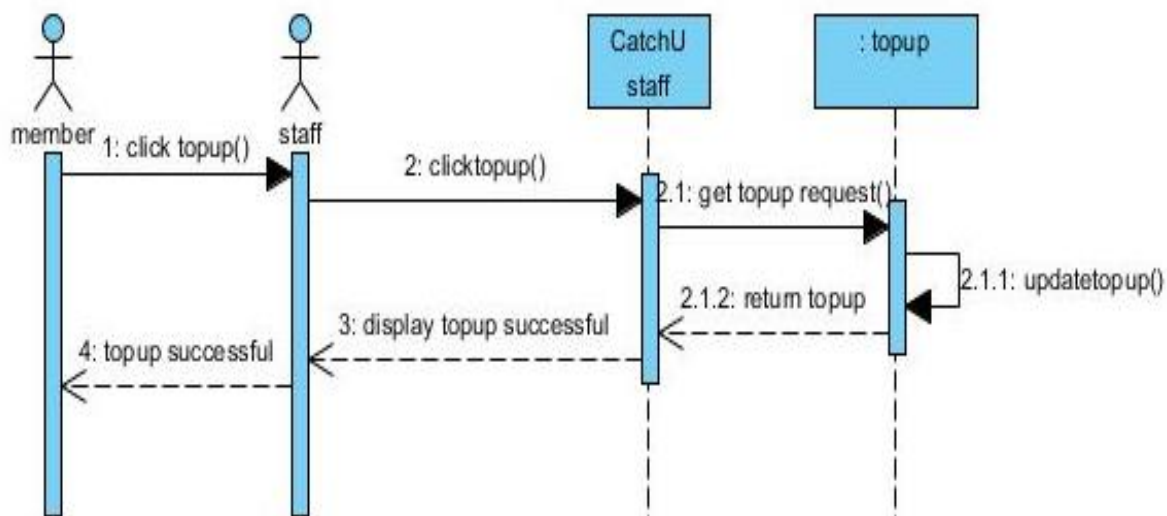


圖 6-1-11 「儲值」之循序圖

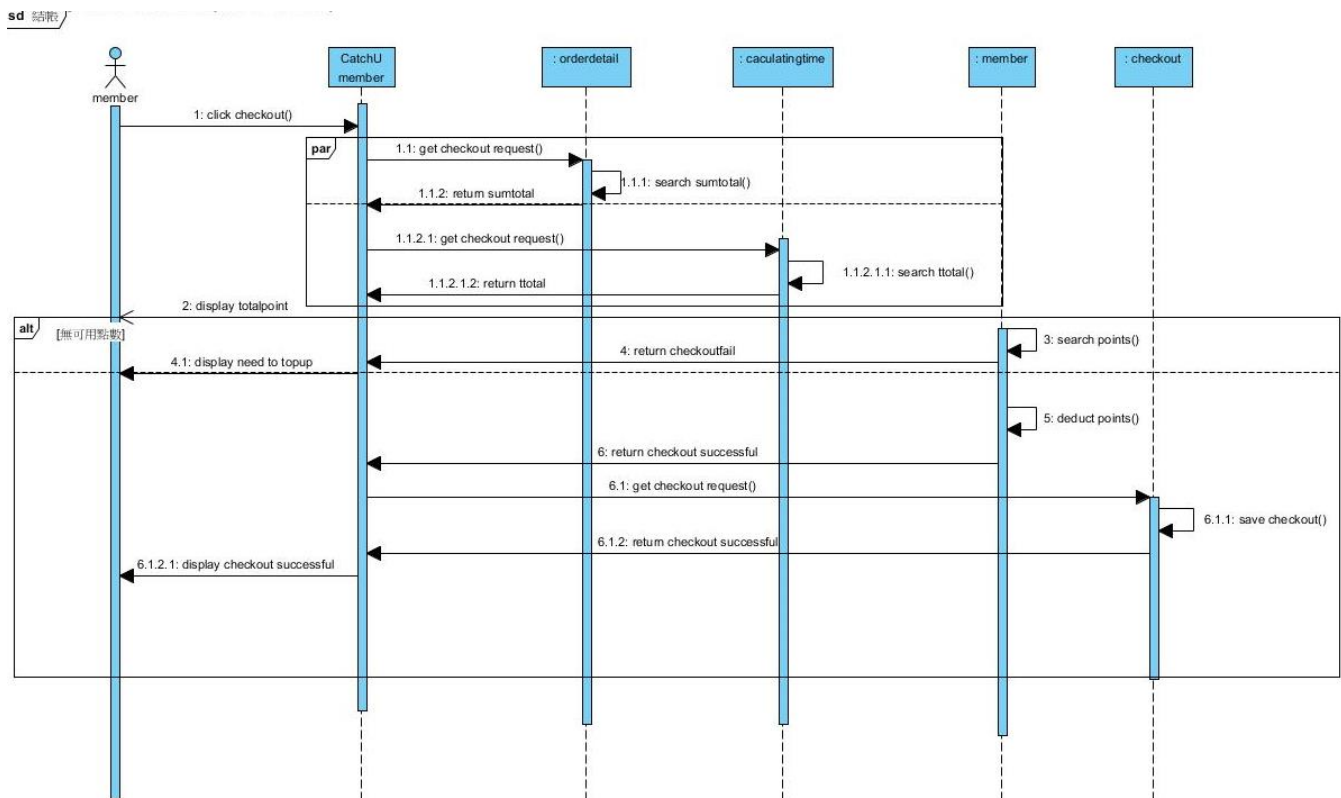


圖 6-1-12 「結帳」之循序圖

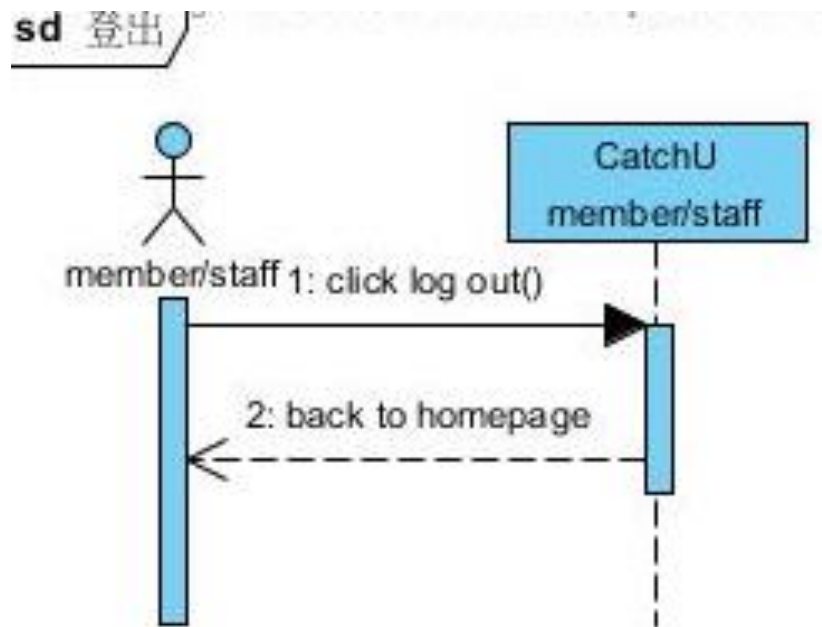


圖 6-1-13 「登出」之循序圖

6-2 設計類別圖(Design class diagram)

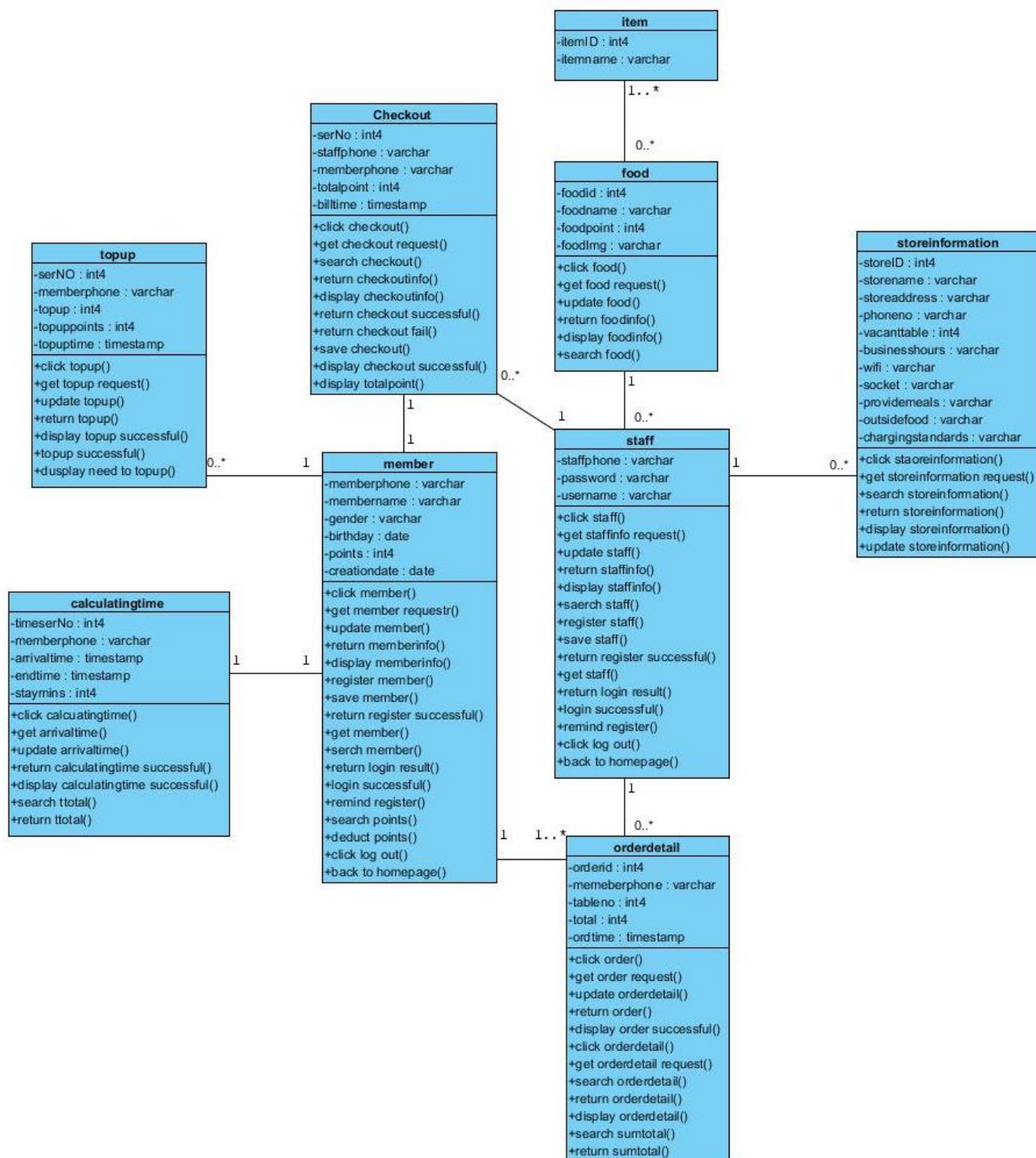


圖 6-2-1 設計類別圖

第七章 實作模型

7-1 佈署圖(Deployment diagram)

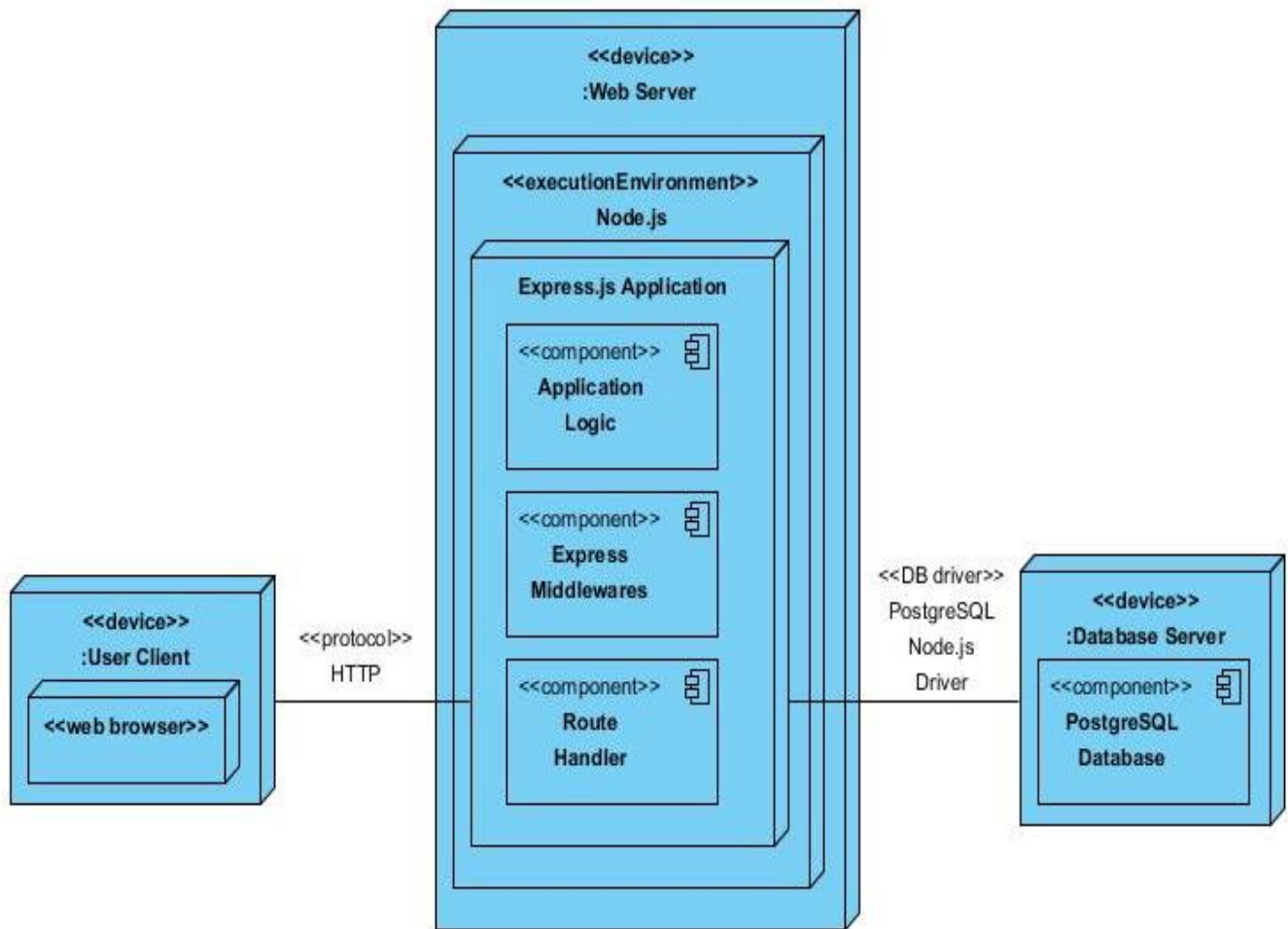


圖 7-1-1 佈署圖

7-2 套件圖(Package diagram)

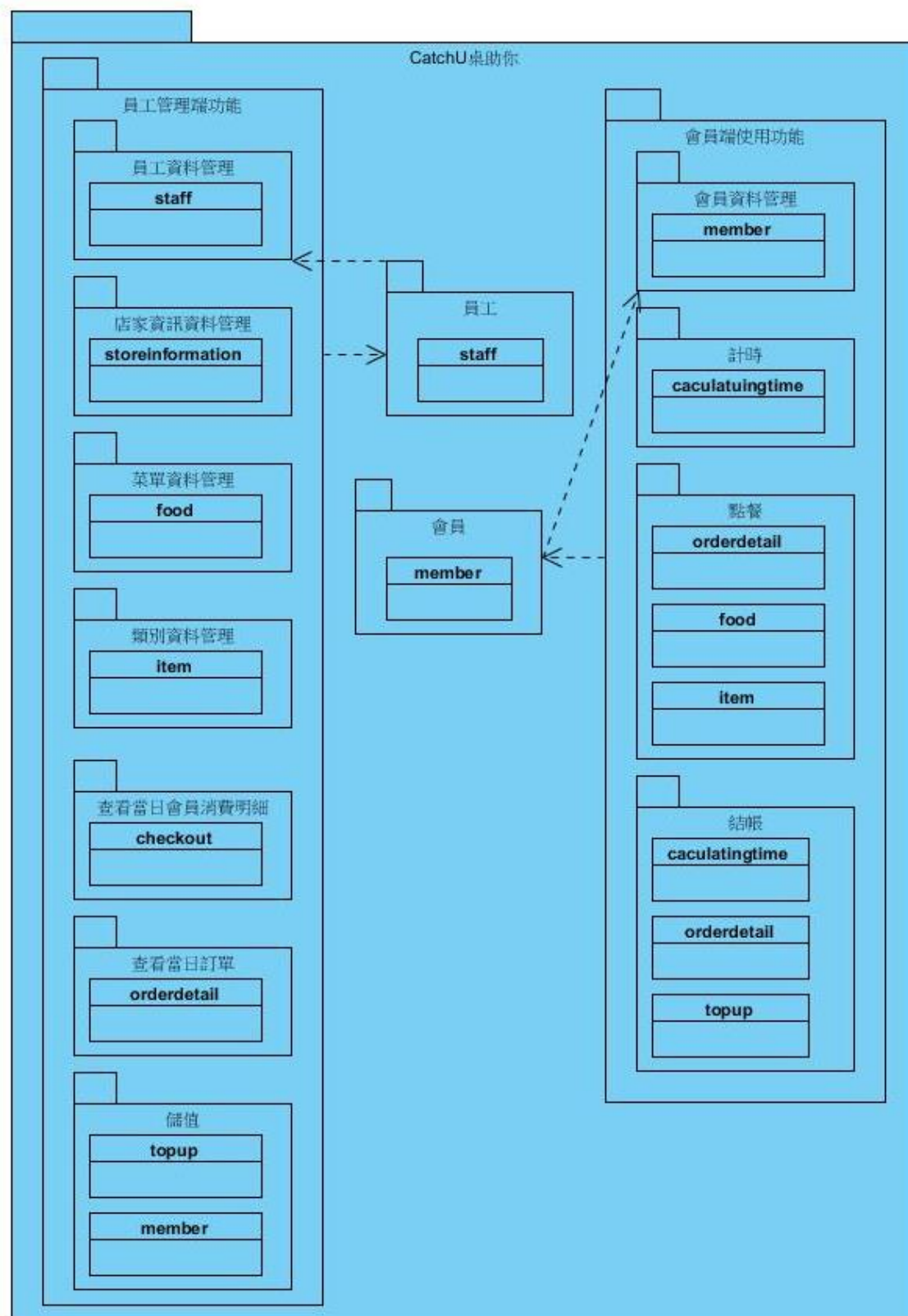


圖 7-2-1 套件圖

7-3 元件圖(Component diagram)

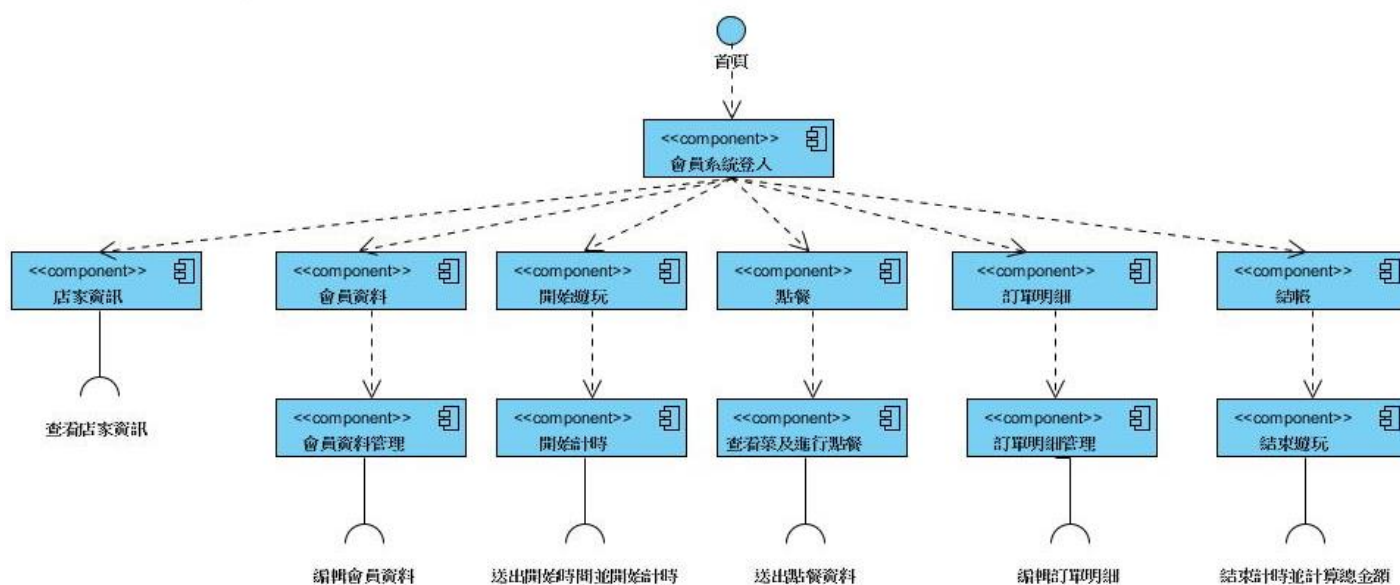


圖 7-3-1 「會員端」之元件圖

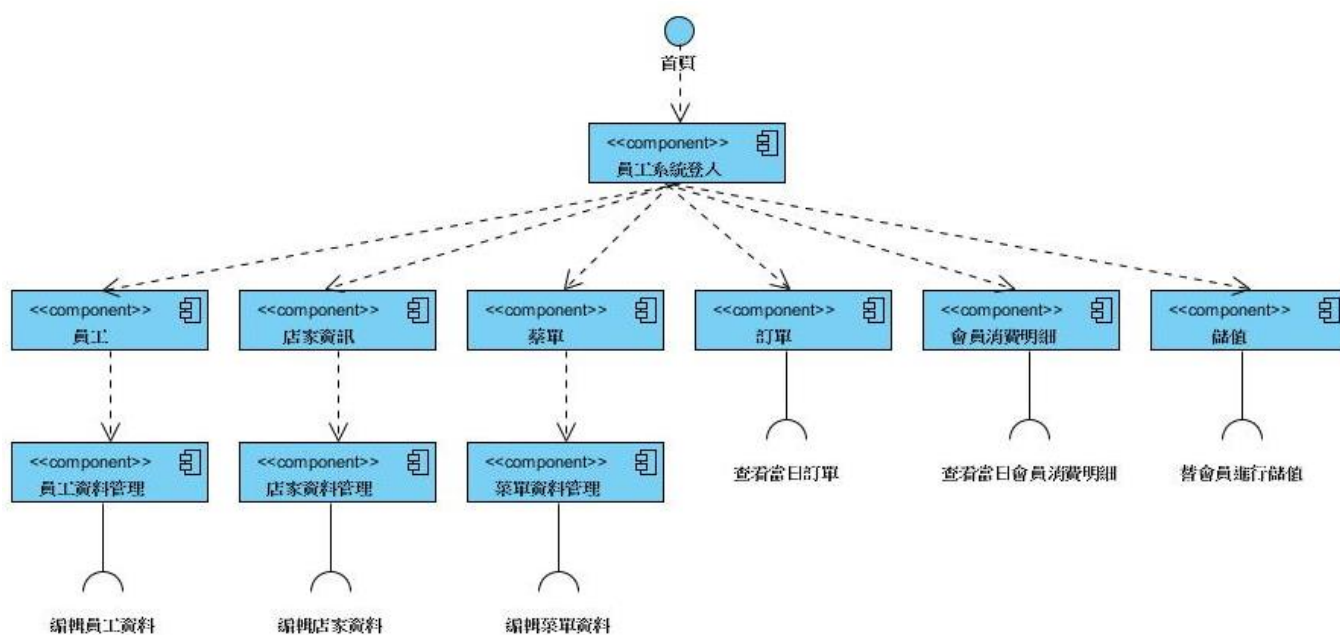


圖 7-3-2 「員工端」之元件圖

7-4 狀態機(State machine)

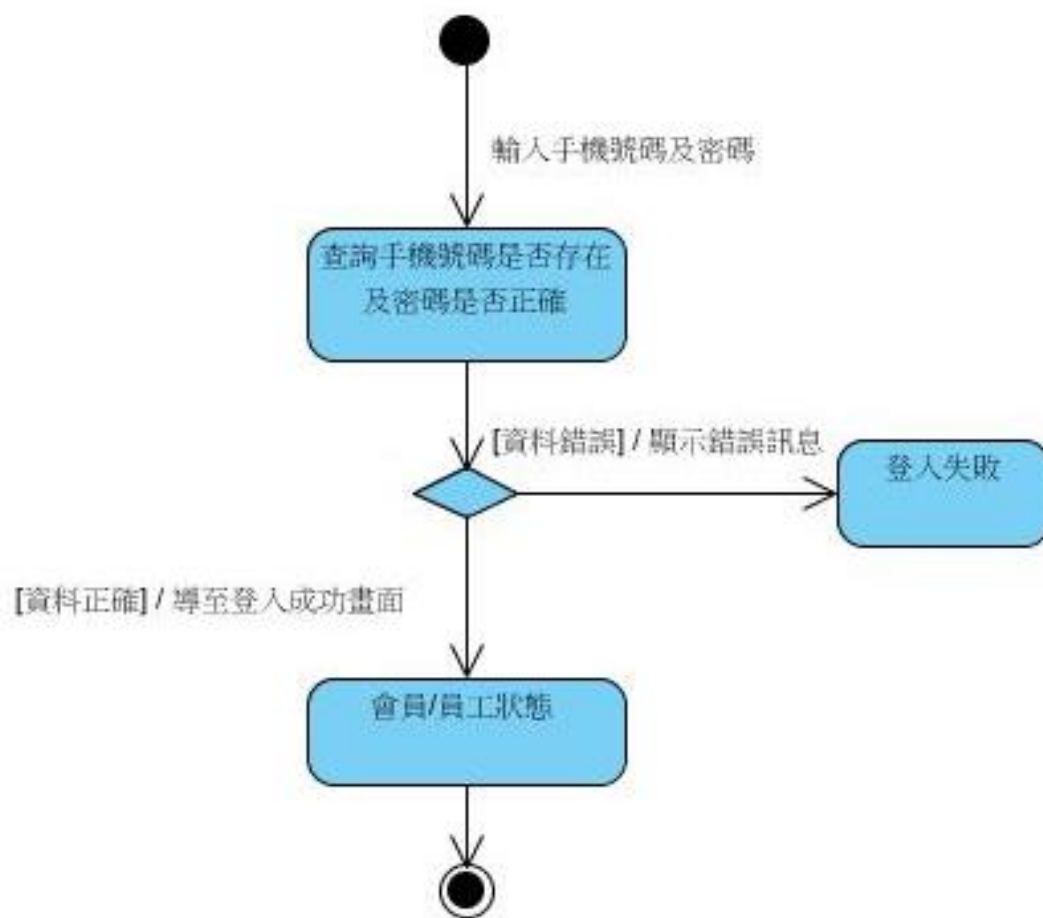


圖 7-4-1 狀態機

第八章 資料庫設計

8-1 資料庫關聯表

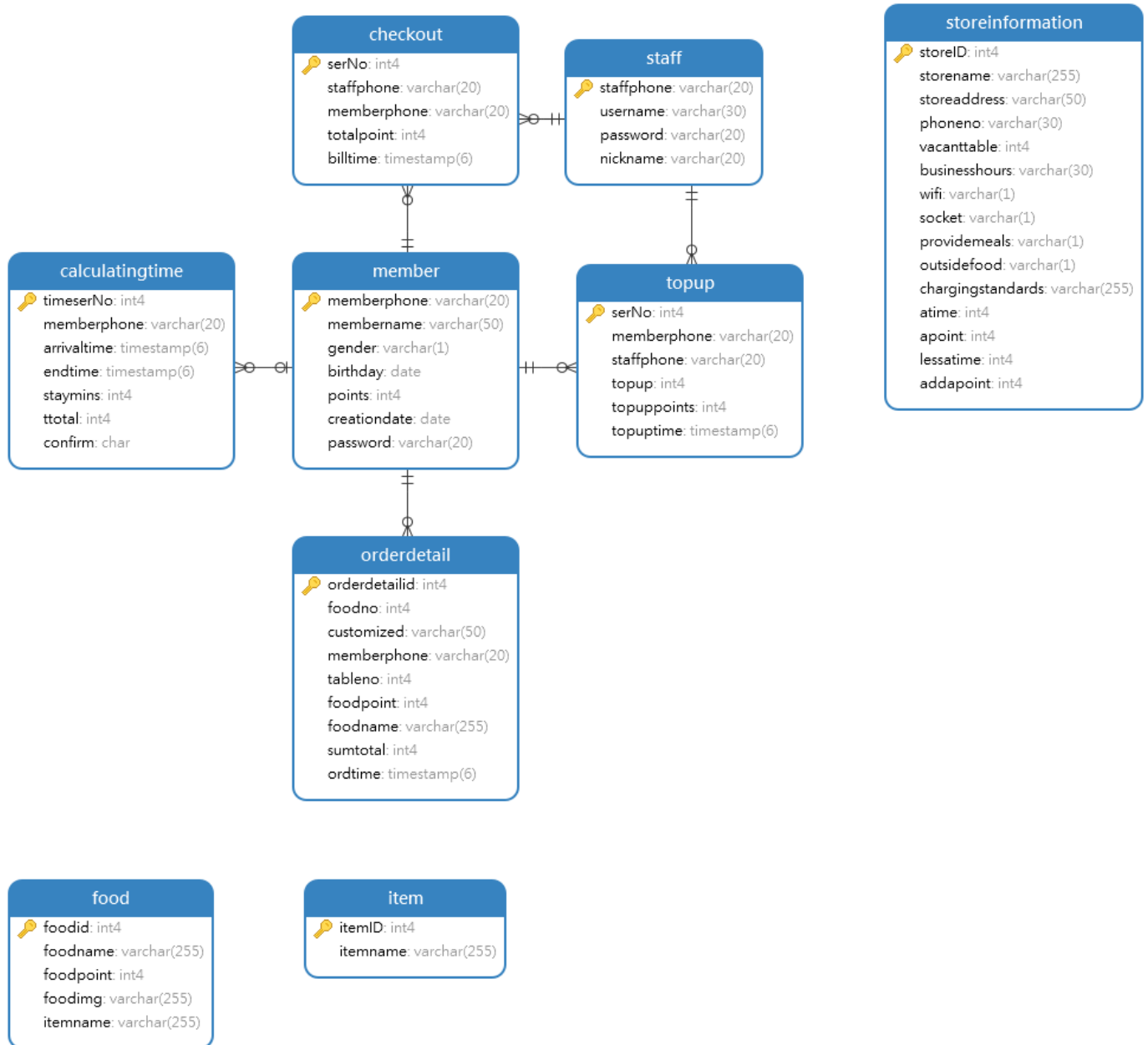


圖 8-1-1 資料庫關聯表

8-2 表格及其 Meta data

表 8-2-1 資料表描述：會員

資料表名稱: member					
欄位名稱	中文名稱	資料類型	長度	允許空值	備註
memberphone	會員手機號碼	varchar	20	否	PK
membername	會員姓名	varchar	50	否	
gender	性別	varchar	1	是	男(M)，女(F)
birthday	出生日期	date	0	是	
points	剩餘點數	int4	32	是	
creationdate	建立日期	date	0	是	
password	密碼	varchar	20	是	

表 8-2-2 資料表描述：員工

資料表名稱：staff					
欄位名稱	中文名稱	資料類型	長度	允許空值	備註
staffphone	員工電話	varchar	20	否	PK
username	員工姓名	varchar	30	否	
password	密碼	varchar	20	否	
nickname	員工綽號	varchar	20	是	

表 8-2-3 資料表描述：店家資訊

資料表名稱：storeinformation					
欄位名稱	中文名稱	資料類型	長度	允許空值	備註
storeID	店家編號	int4	32	否	PK（自動遞增）
storename	店家名稱	varchar	255	否	
storeaddress	店家地址	varchar	50	否	
phoneno	店家電話	varchar	30	否	
vacanttable	空桌數	int4	32	否	
businesshours	營業時間	varchar	30	否	
wifi	提供 Wi-Fi	varchar	1	否	
socket	提供插座	varchar	1	否	
providemeals	提供餐點	varchar	1	否	
outsidefood	攜帶外食	varchar	1	否	
chargingstandards	收費標準	varchar	255	否	
atime	1 小時	int4	32	否	
apoint	1 小時點數	int4	32	否	
lessatime	小於 1 小時	int4	32	否	
addapoint	額外點數	int4	32	否	

表 8-2-4 資料表描述：菜單

資料表名稱：food					
欄位名稱	中文名稱	資料類型	長度	允許空值	備註
foodid	餐點編號	int4	32	否	PK (自動遞增)
foodname	餐點名稱	varchar	255	否	
foodpoint	餐點價格	int4	32	否	
foodimg	餐點圖片	varchar	255	是	
itemname	餐點類別	varchar	255	否	

表 8-2-5 資料表描述：餐點類別

資料表名稱：item					
欄位名稱	中文名稱	資料類型	長度	允許空值	備註
itemID	類別編號	int4	32	否	PK (自動遞增)
itemname	類別名稱	varchar	255	否	

表 8-2-6 資料表描述：會員訂單明細

資料表名稱：orderdetail					
欄位名稱	中文名稱	資料類型	長度	允許空值	備註
orderdetailid	訂單編號	int4	32	否	PK (自動遞增)
foodid	餐點編號	int4	32	否	FK
foodno	餐點數量	int4	32	否	
customized	備註	varchar	50	是	
memberphone	會員電話	varchar	20	否	
tableno	桌號	int4	32	否	
total	合計	int4	32	是	
ordtime	訂單建立時間	timestamp	6	是	
foodname	餐點名稱	varchar	255	否	
sumtotal	餐點總金額	int4	32	是	

表 8-2-7 資料表描述：計算時間

資料表名稱：calculatingtime					
欄位名稱	中文名稱	資料類型	長度	允許空值	備註
timeserNo	到達及結束時間 流水號	int4	32	是	PK（自動遞增）
memberphone	會員電話	varchar	20	否	FK
arrivaltime	到達時間	timestamp	6	否	
endtime	結束時間	timestamp	6	否	
staymins	停留時間	int4	32	否	
ttotal	遊玩花費時間	int4	32	否	
confirm	是否結帳	bpchar	1	是	

表 8-2-8 資料表描述：會員結帳明細

資料表名稱：checkout					
欄位名稱	中文名稱	資料類型	長度	允許空值	備註
serNo	結帳編號	int4	32	否	PK（自動遞增）
memberphone	會員電話	varchar	20	否	FK
totalpoint	消費點數	int4	32	否	
billtime	結帳時間	timestamp	6	否	

表 8-2-9 資料表描述：會員儲值

資料表名稱：topup					
欄位名稱	中文名稱	資料類型	長度	允許空值	備註
serNo	儲值編號	int4	32	否	PK（自動遞增）
memberphone	會員電話	varchar	20	否	FK
staffphone	員工電話	varchar	20	否	FK
topup	儲值金額	int4	32	否	
topuppoints	儲值點數	int4	32	否	
topuptime	儲值時間	timestamp	6	否	

第九章 程式

9-1 元件清單及其規格描述

員工端

表 9-1-1 首頁功能列表

檔案路徑與名稱：routes/		
編號	函式名稱	功能
1-1-1	index.js	顯示各功能之按鈕

表 9-1-2 員工功能列表

檔案路徑：routes/		
編號	檔案名稱	功能
1-2-1	staffone.js	呼叫搜尋員工資料服務，並顯示登入中之員工資料
1-2-2	Staffupdate.js	建立員工更新資料物件，並呼叫更新服務
1-2-3	staffupdateno.js	導至尋找員工資料之頁面
1-2-4	Staffupdateform.js	顯示員工原本資料於更新頁面欄位
檔案路徑：routes/utility/		
1-2-5	staff.js	回應上述函式呼叫服務

表 9-1-3 店家資訊功能列表

檔案路徑：routes/		
編號	檔案名稱	功能
1-3-1	storeadd.js	建立店家資料物件並呼叫新增服務
1-3-2	storeaddform.js	導至店家資料新增頁面
1-3-3	storelist.js	呼叫搜尋店家資料之服務，並回傳及顯示店家資料
1-3-4	storeremove.js	取得欲刪除店家之編號並呼叫刪除服務
1-3-5	storeremoveform.js	導至店家資料刪除頁面
1-3-6	storeupdate.js	建立店家更新資料物件並呼叫更新服務
1-3-7	storeupdateno.js	導至搜尋店家編號之頁面
1-3-8	storeupdateform.js	將原有店家資料回傳至更新頁面之欄位
檔案路徑：routes/utility/		
1-3-9	store	回應上述函式呼叫服務

表 9-1-4 菜單資訊功能列表

檔案路徑與名稱：routes		
編號	檔案名稱	功能
1-4-1	food_add.js	建立菜單餐點物件並呼叫新增服務
1-4-2	food_add_form.js	導至餐點資料新增頁面
1-4-3	food_list.js	呼叫搜尋餐點資料之服務，並回傳及顯示餐點資料
1-4-4	food_remove.js	取得欲刪除餐點之編號並呼叫刪除服務
1-4-5	food_remove_form.js	導至餐點刪除頁面
1-4-6	foodupdate.js	建立餐點更新資料物件並呼叫更新服務
1-4-7	foodupdateneno.js	導至搜尋餐點編號之頁面
1-4-8	foodupdateform.js	將原有餐點資料回傳至更新頁面之欄位
檔案路徑：routes/utility/		
1-4-9	food.js	回應上述函式呼叫服務

表 9-1-5 類別功能列表

檔案路徑：routes/		
編號	檔案名稱	功能
1-5-1	itemadd.js	建立類別資料物件並呼叫新增服務
1-5-2	itemaddform.js	導至類別新增頁面
1-5-3	itemlist.js	呼叫搜尋類別資料之服務，並回傳及顯示類別資料
1-5-4	itemremove.js	取得欲刪除類別之編號並呼叫刪除服務
1-5-5	itemremoveform.js	導至類別刪除頁面
1-5-6	itemupdate.js	建立類別更新資料物件並呼叫更新服務
1-5-7	itemupdateno.js	導至搜尋類別編號之頁面
1-5-8	itemupdateform.js	將原有類別資料回傳至更新頁面之欄位
檔案路徑：routes/utility/		
1-5-9	item.js	回應上述函式呼叫服務

表 9-1-6 餐點訂單功能列表

檔案路徑：routes/		
編號	檔案名稱	功能
1-6-1	orderlist.js	呼叫搜尋會員訂單之服務，並回傳及顯示當日會員訂單
檔案路徑：routes/utility/		
1-6-2	order.js	回應上述函式呼叫服務

表 9-1-7 會員消費明細功能列表

檔案路徑：routes/		
編號	檔案名稱	功能
1-7-1	checkoutlist.js	呼叫搜尋會員結帳明細之服務，並回傳及顯示會員當日結帳明細
檔案路徑：routes/utility/		
1-7-2	checkout.js	回應上述函式呼叫服務

表 9-1-8 儲值點數功能列表

檔案路徑：routes/		
編號	檔案名稱	功能
1-8-1	topupadd.js	建立會員儲值資料物件並呼叫新增服務
1-8-2	topupaddform.js	導至會員儲值新增頁面
檔案路徑與名稱：routes/utility/topup.js		
1-8-3	topup	回應上述函式呼叫服務

表 9-1-9 使用者功能列表

檔案路徑：routes/		
編號	檔案名稱	功能
1-9-1	usershow.js	顯示登入中員工名稱
1-9-2	userlogout.js	將員工登出
1-9-3	userloginform.js	判斷員工是否登入
1-9-4	userlogin.js	呼叫搜尋員工帳號密碼服務，並判斷是否存在
1-9-5	checkAuth.js	檢查登入權限
檔案路徑：routes/utility/		
1-9-6	user.js	回應上述函式呼叫服務

會員端

表 9-1-10 首頁功能列表

檔案路徑與名稱：routes/		
編號	函式名稱	功能
1-10-1	index.js	顯示各功能之按鈕

表 9-1-11 使用者功能列表

檔案路徑：routes/		
編號	檔案名稱	功能
1-11-1	user_show.js	顯示登入中會員名稱
1-11-2	user_logout.js	將會員登出
1-11-3	user_login_form.js	判斷會員是否登入
1-11-4	user_login.js	呼叫搜尋會員帳號密碼服務，並判斷是否存在
1-11-5	checkAuth.js	檢查登入權限
檔案路徑：routes/utility/		
1-11-6	user.js	回應上述函式呼叫服務

表 9-1-12 查看店家資訊功能列表

檔案路徑：routes/		
編號	檔案名稱	功能
1-12-1	storelist.js	呼叫搜尋店家資料之服務，並回傳及顯示店家資料
檔案路徑：routes/utility/		
1-12-2	store.js	回應上述函式呼叫服務

表 9-1-13 訂單明細功能列表

檔案路徑：routes/		
編號	檔案名稱	功能
1-13-1	orderdetail_one.js	取得某一位會員的訂單明細
1-13-2	orderdetail_add.js	建立訂單明細物件並呼叫新增服務
1-13-3	orderdetail_add_form.js	導至訂單明細新增頁面
1-13-4	orderdetail_remove.js	取得欲刪除訂單明細之編號並呼叫刪除服務
1-13-5	orderdetail_remove_form.js	導至訂單明細刪除頁面
1-13-6	orderdetail_update.js	建立訂單明細更新資料物件並呼叫更新服務
1-13-7	orderdetail_update_no.js	導至搜尋訂單明細編號之頁面

1-13-8	orderdetail_update_form.js	將原有訂單明細回傳至更新頁面之欄位
檔案路徑：routes/utility/		
1-13-9	orderdetail.js	回應上述函式呼叫服務

表 9-1-14 會員資料功能列表

檔案路徑：routes/		
編號	檔案名稱	功能
1-14-1	member_one.js	取得某一位會員的資料
1-14-2	member_add.js	建立會員資料物件並呼叫新增服務
1-14-3	member_add_form.js	導至會員資料新增頁面
1-14-4	member_list.js	呼叫搜尋會員資料之服務，並回傳及顯示會員資料
1-14-5	member_remove.js	取得欲刪除會員之帳號並呼叫刪除服務
1-14-6	member_remove_form.js	導至會員資料刪除頁面
1-14-7	member_update.js	建立會員更新資料物件並呼叫更新服務
1-14-8	member_update_no.js	導至搜尋會員帳號之頁面
1-14-9	member_update_form.js	將原有會員資料回傳至更新頁面之欄位

檔案路徑：routes/utility/		
1-14-10	member.js	回應上述函式呼叫服務

表 9-1-15 查看菜單功能列表

檔案路徑：routes/		
編號	檔案名稱	功能
1-15-1	food_list.js	呼叫搜尋菜單之服務，並回傳及顯示菜單
檔案路徑：routes/utility/		
1-15-2	food.js	回應上述函式呼叫服務

表 9-1-16 計時及結帳功能列表

檔案路徑：routes/		
編號	檔案名稱	功能
1-16-1	caltime_add.js	建立計時資料物件並呼叫新增服務
1-16-2	caltime_add_form.js	導至計時資料新增頁面
1-16-3	caltime_addend.js	建立結帳資料物件並呼叫新增服務
1-16-4	caltime_addend_form.js	導至結帳資料頁面
檔案路徑：routes/utility/		
1-16-5	caltime.js	回應上述函式呼叫服務

員工端

表 9-1-17 首頁功能列表

編號	1-1-1	檔案名稱	index.js
功能	顯示各功能之按鈕		
<pre>var express = require('express'); var router = express.Router(); /* GET home page. */ router.get('/', function(req, res, next) { res.render('index', { title: 'Express' }); }); module.exports = router;</pre>			

表 9-1-18 搜尋員工資料列表

編號	1-2-1	檔案名稱	staffone.js
功能	呼叫搜尋員工資料服務，並顯示登入中之員工資料		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 var moment = require('moment'); const staff = require('./utility/staff'); //接收 GET 請求 router.get('/', function(req, res, next) { var staffphone = req.session.staffphone; staff.one(staffphone).then(data => { if (data==null){</pre>			

```

        res.render('error'); //導向錯誤頁面
    }else if(data==-1){
        res.render('notFound'); //導向找不到頁面
    }else{
        res.render('stafflist', {item:data}); //將資料傳給顯示頁面
    }
    })
});

module.exports = router;

```

表 9-1-19 建立員工更新資料列表

編號	1-2-2	檔案名稱	staffupdate.js
功能	建立員工更新資料物件，並呼叫更新服務		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const staff = require('./utility/staff'); //接收 POST 請求 router.post('/', function(req, res, next) { var username = req.body.username; //取得員工姓名 var newData={ username:username, //員工姓名 staffphone: req.body.staffphone, //取得員工電話</pre>			


```

        nickname: req.body.nickname,           //取得暱稱

        password: req.body.password           //取得密碼
    }

    staff.update(newData).then(d => {
        if (d>=0){
            res.render('updateSuccess', {results:d}); //傳至成功頁面
        }else{
            res.render('updateFail'); //導向錯誤頁面
        }
    })
});

//匯出
module.exports = router;

```

表 9-1-20 導至尋找員工資料之頁面列表

編號	1-2-3	檔案名稱	staffupdateno.js
功能	導至尋找員工資料之頁面		
<pre>var express = require('express'); var router = express.Router(); /* GET home page. */ router.get('/', function(req, res, next) { res.render('staffupdateno'); }); //匯出 module.exports = router;</pre>			

表 9-1-21 顯示員工原本資料於更新頁面列表

編號	1-2-4	檔案名稱	staffupdateform.js
功能	顯示員工原本資料於更新頁面欄位		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 var moment = require('moment'); const staff = require('./utility/staff'); //接收 GET 請求 router.get('/', function(req, res, next) { var no = req.session.staffphone; staff.query(no).then(d => { if (d!=null && d!=-1){ var data = { username: d.username, staffphone: d.staffphone, nickname: d.nickname, password: d.password, } res.render('staffupdateform', {item:data}); //將資料傳給更新頁面 }else{ res.render('notFound'); //導向找不到頁面 } }) }); //匯出 module.exports = router;</pre>			

表 9-1-22 函式呼叫服務列表

編號	1-2-5	檔案名稱	staff.js
功能	回應上述函式呼叫服務		
<pre>'use strict'; //引用操作資料庫的物件 const sql = require('./asyncDB'); //----- //執行資料庫動作的函式-取出單一員工 //----- var one = async function(staffphone){ var result={ }; await sql('SELECT * FROM staff WHERE staffphone = \$1', [staffphone]) .then((data) => { if(data.rows.length > 0){ result = data.rows[0]; }else{ result = -1; } }, (error) => { result = null; }); return result; } //----- //執行資料庫動作的函式-取出單一員工</pre>			

```

//-----
var query = async function(staffphone){
    var result={ };

    await sql('SELECT * FROM staff WHERE "staffphone" = $1', [staffphone])
        .then((data) => {
            if(data.rows.length > 0){
                result = data.rows[0];
            }else{
                result = -1;
            }
        }, (error) => {
            result = null;
        });

    return result;
}

//-----
// 更新員工資料

//-----
var update = async function(newData){
    var results;

    await sql('UPDATE staff SET "username"=$2, "nickname"=$3, "password"=$4
WHERE "staffphone" = $1', [newData.staffphone, newData.username,
newData.nickname, newData.password])
        .then((data) => {
            results = data.rowCount;
        }, (error) => {
            results = -1;
        });

    return results;
}
module.exports = {one, query, update}

```

表 9-1-23 建立店家資料列表

編號	1-3-1	檔案名稱	storeadd.js
功能	建立店家資料物件並呼叫新增服務		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const store = require('./utility/store'); //接收POST請求 router.post('/', function(req, res, next) { var storename = req.body.storename; var storeaddress = req.body.storeaddress; var phoneno = req.body.phoneno; var vacanttable = Number(req.body.vacanttable); var businesshours = req.body.businesshours; var wifi = req.body.wifi; var socket = req.body.socket; var providemeals = req.body.providemeals; var outsidefood = req.body.outsidefood; var chargingstandards = req.body.chargingstandards; var atime = Number(req.body.atime); var apoint = Number(req.body.apoint); var lessatime = Number(req.body.lessatime); var addapoint = Number(req.body.addapoint); // 建立一個新資料物件 var newData={ storename:storename, storeaddress:storeaddress, phoneno:phoneno, vacanttable:vacanttable, businesshours:businesshours, wifi:wifi,</pre>			

```

        socket:socket,
        providemeals:providemeals,
        outsidefood:outsidefood,
        chargingstandards:chargingstandards,
        atime:atime,
        apoint:apoint,
        lessatime:lessatime,
        addapoint:addapoint
    }

    store.add(newData).then(d => {

        res.render('addSuccess'); //傳至成功頁面

        if (d==0){
        }else{

            res.render('addFail');    //導向錯誤頁面

        }

    })
});

module.exports = router;

```

表 9-1-24 導至店家資料新增頁面列表

編號	1-3-2	檔案名稱	storeaddform.js
功能	導至店家資料新增頁面		
<pre>var express = require('express'); var router = express.Router(); //接收GET請求 router.get('/', function(req, res, next) { res.render('storeaddform'); }); module.exports = router;</pre>			

表 9-1-25 搜尋店家資料列表

編號	1-3-3	檔案名稱	storelist.js
功能	呼叫搜尋店家資料之服務，並回傳並顯示店家資料		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const store = require('./utility/store'); //接收GET請求 router.get('/', function(req, res, next) { store.list().then(data => { if(data==null){ res.render('error'); //導向錯誤頁面 }else if(data.length > 0){ res.render('storelist', {items:data}); //將資料傳給顯示頁面 }else{ res.render('notFound'); //導向找不到頁面 } }) }); module.exports = router;</pre>			

表 9-1-26 取得欲刪除店家之編號列表

編號	1-3-4	檔案名稱	storeremove.js
功能	取得欲刪除店家之編號並呼叫刪除服務		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const store = require('./utility/store'); //接收POST請求 router.post('/', function(req, res, next) { var storeID = req.body.storeID; //取得店家編號 store.remove(storeID).then(d => { if(d>=0){ res.render('removeSuccess', {results:d}); //傳至成功頁面 }else{ res.render('removeFail'); //導向錯誤頁面 } }) }); module.exports = router;</pre>			

表 9-1-27 導至店家資料刪除頁面列表

編號	1-3-5	函式名稱	storeremoveform.js
功能	導至店家資料刪除頁面		
<pre>var express = require('express'); const store = require('./utility/store'); var router = express.Router(); router.get('/', function(req, res, next) { store.getDropdownData().then(d => { if (d!=[]){ res.render('storeremoveform', {result:d}); //轉至新增頁面 }else{ res.render('removeFail'); //導向錯誤頁面 } }); }); module.exports = router;</pre>			

表 9-1-28 建立店家更新資料列表

編號	1-3-6	檔案名稱	storeupdate.js
功能	建立店家更新資料物件並呼叫更新服務		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const store = require('./utility/store'); //接收POST請求</pre>			

```

router.post('/', function(req, res, next) {
    var storeID = req.body.storeID;    //取得店家編號

    var newData={
        storeID:storeID,                //店家編號

        storename: req.body.storename,    //取得店家名稱

        storeaddress: req.body.storeaddress,    //取得店家地址

        phoneno: req.body.phoneno,        //取得電話號碼

        vacanttable: Number(req.body.vacanttable), //取得空桌數

        businesshours: req.body.businesshours, //取得營業時間

        wifi: req.body.wifi,              //取得提供wifi

        socket: req.body.socket,          //取得提供插座

        providemeals: req.body.providemeals, //取得提供餐點

        outsidefood: req.body.outsidefood, //取得可帶外食

        chargingstandards: req.body.chargingstandards, //取得收費標準

        atime: Number(req.body.atime),
        apoint: Number(req.body.apoint),
        lessatime: Number(req.body.lessatime),
        addapoint: Number(req.body.addapoint)
    }

    store.update(newData).then(d => {
        if (d>=0){
            res.render('updateSuccess', {results:d}); //傳至成功頁面
        }else{

```

```

        res.render('updateFail');    //導向錯誤頁面
    }
    })
});

//匯出
module.exports = router;

```

表 9-1-29 導至搜尋店家編號之頁面列表

編號	1-3-7	檔案名稱	storeupdateno.js
功能	導至搜尋店家編號之頁面		
<pre>var express = require('express'); var router = express.Router(); /* GET home page. */ router.get('/', function(req, res, next) { res.render('staffupdateno'); }); //匯出 module.exports = router;</pre>			

表 9-1-30 原有店家資料回傳至更新頁面列表

編號	1-3-8	檔案名稱	storeupdateform.js
功能	將原有店家資料回傳至更新頁面之欄位		
<pre>var express = require('express'); var router = express.Router();</pre>			

//增加引用函式

```
var moment = require('moment');  
const store = require('./utility/store');
```

//接收GET請求

```
router.get('/', function(req, res, next) {  
  var no = req.query.storeID;  
  
  store.query(no).then(d => {  
    if (d!=null && d!=-1){  
      var data = {  
        storeID: d.storeID,  
        storename: d.storename,  
        storeaddress: d.storeaddress,  
        phoneno: d.phoneno,  
        vacanttable: d.vacanttable,  
        businesshours: d.businesshours,  
        wifi: d.wifi,  
        socket: d.socket,  
        providemeals: d.providemeals,  
        outsidefood: d.outsidefood,  
        chargingstandards: d.chargingstandards,  
        atime: d.atime,  
        apoint: d.apoint,  
        lessatime: d.lessatime,  
        addapoint: d.addapoint,  
      }  
  
      res.render('storeupdateform', {item:data}); //將資料傳給更新頁面  
    }else{  
      res.render('notFound'); //導向找不到頁面  
    }  
  })  
});
```

//匯出

```
module.exports = router;
```

表 9-1-31 函式呼叫服務列表

編號	1-3-9	檔案名稱	store.js
功能	回應上述函式呼叫服務		

```
'use strict';

//引用操作資料庫的物件
const sql = require('./asyncDB');

//-----

//執行資料庫動作的函式-傳回所有店家資訊
//-----
var list = async function(){
    var result=[];

    //console.log("查詢店家資訊");

    await sql('SELECT * FROM storeinformation')
        .then((data) => {
            result = data.rows;
            console.log(result)  ;
        }, (error) => {
            result = null;

            //console.log("除去錯誤")  ;

        });

    return result;
}
//-----

// 新增店家資訊
//-----
var add = async function(newData){
    var result;
```

```
console.log(newData)
```

```
    await sql('INSERT INTO storeinformation ("storename", "storeaddress", "phoneno",  
"vacanttable", "businesshours", "wifi", "socket", "providemeals", "outsidefood",  
"chargingstandards", "atime", "apoint", "lessatime", "addapoint") VALUES ($1, $2, $3,  
$4, $5, $6, $7, $8, $9, $10, $11, $12, $13, $14)', [newData.storename,  
newData.storeaddress, newData.phoneno, newData.vacanttable, newData.businesshours,  
newData.wifi, newData.socket, newData.providemeals, newData.outsidefood,  
newData.chargingstandards, newData.atime, newData.apoint, newData.lessatime,  
newData.addapoint])
```

```
        .then((data) => {  
            result = 0;  
        }, (error) => {  
            result = -1;  
        });
```

```
    return result;  
}
```

```
//-----
```

```
// 刪除員工
```

```
//-----
```

```
var remove = async function(storeID){  
    var result;
```

```
    await sql('DELETE FROM storeinformation WHERE "storeID"= $1', [storeID])  
        .then((data) => {  
            result = data.rowCount;  
        }, (error) => {  
            result = -1;  
        });
```

```
    return result;  
}
```

```
//-----
```

```
//執行資料庫動作的函式-取出單一店家資訊
```

```
//-----
```

```
var query = async function(storeID){  
    var result={};
```

```

    await sql('SELECT * FROM storeinformation WHERE "storeID" = $1', [storeID])
      .then((data) => {
        if(data.rows.length > 0){
          result = data.rows[0];
        }else{
          result = -1;
        }
      }, (error) => {
        result = null;
      });

    return result;
  }
//-----

// 取出型態資料

//-----
var getDropDownData = async function(){
  //儲存下拉式選單資料

  var storeinformation;

  //取回prototype資料

  await sql('SELECT * FROM storeinformation ORDER BY "storeID"')
    .then((data) => {
      storeinformation = data.rows;
    }, (error) => {
      result = [];
    });

  //設定回傳資料

  var result = {};
  result.storeinformation = storeinformation;

  //回傳

  return result;
}

```

```

}

//-----

// 更新店家資料

//-----

var update = async function(newData){
    var results;

    await sql('UPDATE storeinformation SET "storename"=$2, "storeaddress"=$3,
"phoneno"=$4, "vacanttable"=$5, "businesshours"=$6, "wifi"=$7, "socket"=$8,
"providemeals"=$9, "outsidefood"=$10, "chargingstandards"=$11, "atime"=$12,
"apoint"=$13, "lessatime"=$14, "addapoint"=$15 WHERE "storeID" = $1',
[newData.storeID, newData.storename, newData.storeaddress, newData.phoneno,
newData.vacanttable, newData.businesshours, newData.wifi, newData.socket,
newData.providemeals, newData.outsidefood, newData.chargingstandards,
newData.atime, newData.apoint, newData.lessatime, newData.addapoint])
    .then((data) => {
        results = data.rowCount;
    }, (error) => {
        results = -1;
    });

    return results;
}

module.exports = {list, add, remove, query, getDropdownData, update}

```

表 9-1-32 建立菜單餐點物件列表

編號	1-4-1	檔案名稱	food_add.js
功能	建立菜單餐點物件並呼叫新增服務		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const food = require('./utility/food');</pre>			


```

//-----
// 引用multer外掛
//-----
const multer = require('multer');

// 宣告上傳存放空間及檔名更改
var storage = multer.diskStorage({
  destination: function (req, file, cb) {
    cb(null, 'public/imgs');
  },

  filename: function (req, file, cb) {
    cb(null, Date.now()+"--"+file.originalname);
  }
});

// 產生multer的上傳物件

var maxSize=1024*1024; //設定最大可接受圖片大小(1M)

var upload = multer({
  storage:storage
});
//-----

//接收POST請求
router.post('/',upload.single('foodimg'), function(req, res, next) {

  var itemname = req.body.itemname;
  var foodname = req.body.foodname;
  var foodpoint = Number(req.body.foodpoint);
  var foodimg;

  // 如果有選擇圖片
  if (typeof(req.file) != 'undefined'){

```

```

        fooding=req.file.filename;    //取得上傳照片名稱
    }

    // 建立一個新資料物件
    var newData={

        itemname:itemname,
        foodname:foodname,
        foodpoint:foodpoint,
        fooding:fooding
    }

    food.add(newData).then(d => {
        if (d==0){
            res.render('addSuccess'); //傳至成功頁面
        }else{
            res.render('addFail');    //導向錯誤頁面
        }
    })
});

module.exports = router;

```

表 9-1-33 導至餐點資料新增頁面列表

編號	1-4-2	檔案名稱	food_add_form.js
功能	導至餐點資料新增頁面		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式</pre>			

```

const food = require('./utility/food');

//接收GET請求
router.get('/', function(req, res, next) {
    food.getDropdownData().then(d => {
        if (d!=[]){
            res.render('food_add_form', {result:d}); //轉至新增頁面
        }else{
            res.render('addFail'); //導向錯誤頁面
        }
    });
});

module.exports = router;

```

表 9-1-34 搜尋餐點資料列表

編號	1-4-3	檔案名稱	food_list.js
功能	呼叫搜尋餐點資料之服務，並回傳並顯示餐點資料		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const food = require('./utility/food'); //接收GET請求 router.get('/', function(req, res, next) { food.list().then(data => { if(data==null){ res.render('error'); //導向錯誤頁面 }else if(data.length > 0){ console.log(data);</pre>			

```

        res.render('food_list', {items:data}); //將資料傳給顯示頁面
    }else{
        res.render('foodnotFound'); //導向找不到頁面
    }
    })
});

module.exports = router;

```

表 9-1-35 取得欲刪除餐點之編號列表

編號	1-4-4	檔案名稱	food_remove.js
功能	取得欲刪除餐點之編號並呼叫刪除服務		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const food = require('./utility/food'); //接收 POST 請求 router.post('/', function(req, res, next) { var foodid = req.body.foodid; //取得餐點編號 food.remove(foodid).then(d => { if(d>=0){ res.render('removeSuccess', {results:d}); //傳至成功頁面 }else{ res.render('removefoodFail'); //導向錯誤頁面 } }) }); module.exports = router;</pre>			

表 9-1-36 導至餐點刪除頁面列表

編號	1-4-5	檔案名稱	food_remove_form.js
功能	導至餐點刪除頁面		
<pre>var express = require('express'); var router = express.Router(); const food = require('./utility/food'); /* //接收GET請求 router.get('/', function(req, res, next) { res.render('food_remove_form'); }); module.exports = router; */ //接收GET請求 router.get('/', function(req, res, next) { food.getfoodnameData().then(d => { if (d!=[]){ res.render('food_remove_form', {result:d}); }else{ res.render('removefoodFail'); //導向錯誤頁面 } }); }); module.exports = router;</pre>			

表 9-1-37 建立餐點更新資料列表

編號	1-4-6	檔案名稱	foodupdate.js
功能	建立餐點更新資料物件並呼叫更新服務		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const food = require('./utility/food'); //接收POST請求 router.post('/', function(req, res, next) { var foodid = req.body.foodid; //取得餐點編號 var newData={ foodid:foodid, //餐點編號 itemname: req.body.itemname, //取得類別編號 foodname: req.body.foodname, //取得餐點名稱 foodpoint: req.body.foodpoint, //取得餐點點數 } food.update(newData).then(d => { if (d>=0){ res.render('updatesuccesss', {result:d}); //傳至成功頁面 }else{ res.render('updateFail'); //導向錯誤頁面 } }) }); //匯出 module.exports = router;</pre>			

表 9-1-38 導至搜尋餐點編號之頁面列表

編號	1-4-7	檔案名稱	foodupdateno.js
功能	導至搜尋餐點編號之頁面		
<pre>var express = require('express'); var router = express.Router(); const food = require('./utility/food'); /* GET home page. */ /* router.get('/', function(req, res, next) { res.render('foodupdateno'); }); //匯出 module.exports = router; */ //接收GET請求 router.get('/', function(req, res, next) { food.getfoodnameData().then(d => { if (d!=[]){ res.render('foodupdateno', {result:d}); //轉至新增頁面 }else{ res.render('notFound'); //導向錯誤頁面 } }); }); module.exports = router;</pre>			

表 9-1-39 原有餐點資料回傳至更新頁面列表

編號	1-4-8	檔案名稱	foodupdateform.js
功能	將原有餐點資料回傳至更新頁面之欄位		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 var moment = require('moment'); const food = require('./utility/food'); //接收GET請求 router.get('/', function(req, res, next) { var no = req.query.foodid; food.query(no).then(d => { if (d!=null && d!=-1){ res.render('foodupdateform', {result:d}); //將資料傳給更新頁面 }else{ res.render('notFound'); //導向找不到頁面 } }) }); //匯出 module.exports = router;</pre>			

表 9-1-40 函式呼叫服務列表

編號	1-4-9	檔案名稱	food.js
功能	回應上述函式呼叫服務		
<pre>'use strict'; //引用操作資料庫的物件 const sql = require('./asyncDB'); //----- //執行資料庫動作的函式-傳回所有餐點資料 //----- var list = async function(){ var result=[]; console.log("查詢餐點"); await sql('SELECT * FROM food') .then((data) => { result = data.rows; console.log(result) ; }, (error) => { result = null; //console.log("除去錯誤") ; }); return result; } //----- // 取出型態資料 //----- var getDropdownData = async function(){</pre>			

```

//儲存下拉式選單資料

var item;

//取回prototype資料

await sql('SELECT * FROM item ORDER BY "itemID"')
  .then((data) => {
    item = data.rows;
  }, (error) => {
    result = [];
  });

//設定回傳資料

var result = {};
result.item = item;

//回傳

return result;
}

//-----

// 取出型態資料

//-----
var getfoodnameData = async function(){

  //儲存下拉式選單資料

  var food;

  //取回prototype資料

  await sql('SELECT * FROM food ORDER BY "foodid"')
    .then((data) => {
      food = data.rows;
    }, (error) => {
      result = [];
    });
}

```

```

//設定回傳資料

var result = { };
result.food = food;


//回傳

return result;
}
//-----

// 新增餐點

//-----
var add = async function(newData){
    var result;
    console.log(newData)
    await sql('INSERT INTO food ( "itemname", "foodname", "foodpoint", "foodimg")
VALUES ($1, $2, $3, $4)', [newData.itemname, newData.foodname, newData.foodpoint,
newData.foodimg])
        .then((data) => {
            result = 0;
        }, (error) => {
            result = -1;
        });

    return result;
}
//-----

// 刪除餐點

//-----
var remove = async function(foodid){
    var result;

    await sql('DELETE FROM food WHERE "foodid" = $1', [foodid])
        .then((data) => {
            result = data.rowCount;
        }, (error) => {
            result = -1;

```

```

    });

    return result;
}
//-----

//執行資料庫動作的函式-取出單一餐點
//-----
var query = async function(foodid){
    var result={};
    await sql('SELECT * FROM food WHERE "foodid" = $1', [foodid])
        .then((data) => {
            if(data.rows.length > 0){
                result.food = data.rows[0];
            }else{
                result.food = -1;
            }
        }, (error) => {
            result.food = null;
        });

    //取回prototype資料

    await sql('SELECT * FROM item ORDER BY "itemID"')
        .then((data) => {
            result.item = data.rows;
        }, (error) => {
            result.item = [];
        });

    console.log(result)

    return result;
}
//-----

// 更新餐點資料
//-----

```

```

var update = async function(newData){
    var results;
    console.log(newData)
    await sql('UPDATE food SET "itemname"=$2, "foodname"=$3, "foodpoint"=$4
WHERE "foodid" = $1', [newData.foodid, newData.itemname, newData.foodname,
newData.foodpoint])
        .then((data) => {
            results = data.rowCount;
        }, (error) => {
            results = -1;
        });

    return results;
}
module.exports = {getDropdownData, list, add, remove, query, update,
getfoodnameData}food_remove.js

```

表 9-1-41 建立類別資料列表

編號	1-5-1	檔案名稱	itemadd.js
功能	建立類別資料物件並呼叫新增服務		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const item = require('./utility/item'); //接收POST請求 router.post('/', function(req, res, next) { var itemname = req.body.itemname; //取得類別名稱 // 建立一個新資料物件 var newData={</pre>			

```

        itemname:itemname,
    }
    console.log(newData)
    item.add(newData).then(d => {
        if (d==0){
            res.render('addSuccess'); //傳至成功頁面
        }else{
            res.render('addFail');    //導向錯誤頁面
        }
    })
});

module.exports = router;

```

表 9-1-42 導至類別新增頁面列表

編號	1-5-2	檔案名稱	itemaddform.js
功能	導至類別新增頁面		
<pre>var express = require('express'); var router = express.Router(); //接收GET請求 router.get('/', function(req, res, next) { res.render('itemaddform'); }); module.exports = router;</pre>			

表 9-1-43 搜尋類別資料列表

編號	1-5-3	檔案名稱	itemlist.js
功能	呼叫搜尋類別資料之服務，並回傳並顯示類別資料		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const item = require('./utility/item'); //接收GET請求 router.get('/', function(req, res, next) { item.list().then(data => { if(data==null){ res.render('error'); //導向錯誤頁面 }else if(data.length > 0){ console.log(data); res.render('itemlist', {items:data}); //將資料傳給顯示頁面 }else{ res.render('itemnotFound'); //導向找不到頁面 } }) }); module.exports = router;</pre>			

表 9-1-44 取得欲刪除類別之編號列表

編號	1-5-4	函式名稱	itemremove
功能	取得欲刪除類別之編號並呼叫刪除服務		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const item = require('./utility/item'); //接收POST請求 router.post('/', function(req, res, next) { var itemID = req.body.itemID; //取得類別編號 item.remove(itemID).then(d => { if(d>=0){ res.render('removeSuccess', {results:d}); //傳至成功頁面 }else{ res.render('removeFail'); //導向錯誤頁面 } }) }); module.exports = router;</pre>			

表 9-1-45 導至類別刪除頁面列表

編號	1-5-5	檔案名稱	itemmoveform.js
功能	導至類別刪除頁面		
<pre>var express = require('express'); const item = require('./utility/item'); var router = express.Router(); /* //接收GET請求 router.get('/', function(req, res, next) { res.render('itemremoveform'); }); module.exports = router; */ //接收GET請求 router.get('/', function(req, res, next) { item.getDropdownData().then(d => { if (d!=[]){ res.render('itemremoveform', {result:d}); //轉至新增頁面 }else{ res.render('removeFail'); //導向錯誤頁面 } }); }); module.exports = router;</pre>			

表 9-1-46 建立類別更新資料列表

編號	1-5-6	檔案名稱	itemupdate.js
功能	建立類別更新資料物件並呼叫更新服務		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const item = require('./utility/item'); //接收POST請求 router.post('/', function(req, res, next) { var itemID = req.body.itemID; //取得類別編號 var newData={ itemID:itemID, //類別編號 itemname: req.body.itemname, //取得類別名稱 } item.update(newData).then(d => { if (d>=0){ res.render('updateSuccess', {results:d}); //傳至成功頁面 }else{ res.render('updateFail'); //導向錯誤頁面 } }) }); //匯出 module.exports = router;</pre>			

表 9-1-47 導至搜尋類別編號之頁面列表

編號	1-5-7	檔案名稱	itemupdateno.js
功能	導至搜尋類別編號之頁面		
<pre>var express = require('express'); var router = express.Router(); const item = require('./utility/item'); /* GET home page. */ /* router.get('/', function(req, res, next) { res.render('itemupdateno'); }); //匯出 module.exports = router; */ //接收GET請求 router.get('/', function(req, res, next) { item.getDropdownData().then(d => { if (d!=[]){ res.render('itemupdateno', {result:d}); //轉至新增頁面 }else{ res.render('notFound'); //導向錯誤頁面 } }); }); module.exports = router;</pre>			

表 9-1-48 原有類別資料回傳至更新頁面列表

編號	1-5-8	檔案名稱	itemupdateform.js
功能	將原有類別資料回傳至更新頁面之欄位		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 var moment = require('moment'); const item = require('./utility/item'); //接收GET請求 router.get('/', function(req, res, next) { var no = req.query.itemID; item.query(no).then(d => { if (d!=null && d!=-1){ var data = { itemID: d.itemID, itemname: d.itemname, } res.render('itemupdateform', {item:data}); //將資料傳給更新頁面 }else{ res.render('notFound'); //導向找不到頁面 } }) }); //匯出 module.exports = router;</pre>			

表 9-1-49 函式呼叫服務列表

編號	1-5-9	檔案名稱	item.js
功能	回應上述函式呼叫服務		
<pre>'use strict'; //引用操作資料庫的物件 const sql = require('./asyncDB'); //----- //執行資料庫動作的函式-傳回所有類別資料 //----- var list = async function(){ var result=[]; console.log("查詢類別"); await sql('SELECT * FROM item ORDER BY "itemID" ') .then((data) => { result = data.rows; console.log(result) ; }, (error) => { result = null; //console.log("除去錯誤") ; }); return result; } //----- // 取出型態資料 //----- var getDropdownData = async function(){</pre>			

```

//儲存下拉式選單資料

var item;

//取回prototype資料

await sql('SELECT * FROM item ORDER BY "itemID"')
  .then((data) => {
    item = data.rows;
  }, (error) => {
    result = [];
  });

//設定回傳資料

var result = {};
result.item = item;

//回傳

return result;
}
//-----

// 新增類別

//-----
var add = async function(newData){
  var result;

  await sql('INSERT INTO item ("itemname") VALUES ($1)', [newData.itemname])
    .then((data) => {
      result = 0;
    }, (error) => {
      result = -1;
    });

  return result;
}
//-----

```

```

// 刪除類別
//-----
var remove = async function(itemID){
    var result;

    await sql('DELETE FROM item WHERE "itemID" = $1', [itemID])
        .then((data) => {
            result = data.rowCount;
        }, (error) => {
            result = -1;
        });

    return result;
}
//-----

//執行資料庫動作的函式-取出單一類別
//-----
var query = async function(itemID){
    var result={ };

    await sql('SELECT * FROM item WHERE "itemID" = $1', [itemID])
        .then((data) => {
            if(data.rows.length > 0){
                result = data.rows[0];
            }else{
                result = -1;
            }
        }, (error) => {
            result = null;
        });

    return result;
}
//-----

// 更新類別資料
//-----

```

```

var update = async function(newData){
    var results;
    console.log(newData)
    await sql('UPDATE item SET "itemname"=$2 WHERE "itemID" = $1',
[newData.itemID, newData.itemname])
        .then((data) => {
            results = data.rowCount;
        }, (error) => {
            results = -1;
        });

    return results;
}
module.exports = {list, add, remove, query, update, getDropdownData}

```

表 9-1-50 搜尋會員訂單之服務列表

編號	1-6-1	檔案名稱	orderlist.js
功能	呼叫搜尋會員訂單之服務，並回傳顯示當日會員訂單		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const order = require('./utility/order'); var moment = require('moment'); //接收GET請求 router.get('/', function(req, res, next) { order.list().then(data => { if(data==null){ res.render('error'); //導向錯誤頁面 }else if(data.length > 0){ for(var i=0; i<data.length; i++){ data[i].ordtime=moment(data[i].ordtime).format("YYYY-MM-DD</pre>			


```

hh:mm:ss")
    }
    console.log(data);

    res.render('orderlist', {items:data}); //將資料傳給顯示頁面

  }else{

    res.render('ordernotFound'); //導向找不到頁面

  }
})
});

module.exports = router;

```

表 9-1-51 函式呼叫服務列表

編號	1-6-2	檔案名稱	order.js
功能	回應上述函式呼叫服務		
<pre>'use strict'; //引用操作資料庫的物件 const sql = require('./asyncDB'); //----- //執行資料庫動作的函式-傳回所有訂單資料 //----- var list = async function(){ var result=[]; console.log("查詢訂單"); await sql('SELECT * FROM orderdetail WHERE date("ordtime") = date(now()) ORDER BY "ordtime" DESC') .then((data) => { result = data.rows;</pre>			

```

        console.log(result);
    }, (error) => {
        result = null;

        //console.log("除去錯誤");

    });

    return result;
}

module.exports = {list}

```

表 9-1-52 搜尋會員結帳明細列表

編號	1-7-1	檔案名稱	checkoutlist.js
功能	呼叫搜尋會員結帳明細之服務，並回傳顯示會員當日結帳明細		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const checkout = require('./utility/checkout'); var moment = require('moment'); //接收GET請求 router.get('/', function(req, res, next) { checkout.list().then(data => { if(data==null){ res.render('error'); //導向錯誤頁面 }else if(data.length > 0){ for(var i=0; i<data.length; i++){ data[i].billtime=moment(data[i].billtime).format("YYYY-MM-DD hh:mm:ss") } console.log(data);</pre>			

```

        res.render('checkoutlist', {items:data}); //將資料傳給顯示頁面
    }else{
        res.render('checknotFound'); //導向找不到頁面
    }
    })
});

module.exports = router;

```

表 9-1-53 函式呼叫服務列表

編號	1-7-2	檔案名稱	checkout.js
功能	回應上述函式呼叫服務		
<pre>'use strict'; //引用操作資料庫的物件 const sql = require('./asyncDB'); //----- //執行資料庫動作的函式-傳回所有結帳明細 //----- var list = async function(){ var result=[]; console.log("查詢結帳明細"); await sql('SELECT * FROM checkout WHERE date("billtime") = date(now()) ORDER BY "billtime" DESC') .then((data) => { result = data.rows; console.log(result); }, (error) => { result = null;</pre>			

```

        //console.log("除去錯誤") ；

    });

    return result;
}

module.exports = {list}

```

表 9-1-54 建立會員儲值資料列表

編號	1-8-1	檔案名稱	topupadd.js
功能	建立會員儲值資料物件並呼叫新增服務		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const topupp = require('./utility/topupp'); //接收POST請求 router.post('/', function(req, res, next) { var memberphone = req.body.memberphone; var staffphone = req.session.staffphone; var topup = Number(req.body.topup); var topuppoints = Number(req.body.topuppoints); console.log(staffphone) // 建立一個新資料物件 var newData={ memberphone:memberphone, staffphone:staffphone, topup:topup, topuppoints:topuppoints,</pre>			

```

    }
    console.log(newData)
    topup.add(newData).then(d => {
        if (d>0){
            res.render('topupSuccess'); //傳至成功頁面
        }else{
            res.render('topupFail');    //導向錯誤頁面
        }
    })
});

module.exports = router;

```

表 9-1-55 導至會員儲值新增頁面列表

編號	1-8-2	檔案名稱	topupaddform.js
功能	導至會員儲值新增頁面		
<pre>var express = require('express'); var router = express.Router(); //接收GET請求 router.get('/', function(req, res, next) { res.render('topupaddform'); }); module.exports = router;</pre>			

表 9-1-56 函式呼叫服務列表

編號	1-8-3	檔案名稱	topup.js
功能	回應上述函式呼叫服務		
<pre>'use strict'; //引用操作資料庫的物件 const sql = require('./asyncDB'); const session = require('express-session'); var add = async function(newData){ var result; const current = new Date(); await sql('INSERT INTO topup ("memberphone" , "staffphone" , "topup", "topuppoints", "topuptime") VALUES (\$1, \$2, \$3, \$4, \$5)', [newData.memberphone, newData.staffphone, newData.topup, newData.topuppoints, current]) .then((data) => { result = 0; }, (error) => { return -1; }); await sql('UPDATE member SET "points" = "points" + \$2 WHERE "memberphone"=\$1', [newData.memberphone, newData.topuppoints]) .then((data) => { result = data.rowCount; }, (error) => { result = -1; }); return result; } module.exports = {add}</pre>			

表 9-1-57 顯示登入中員工名稱列表

編號	1-9-1	檔案名稱	usershow.js
功能	顯示登入中員工名稱		
<pre>var express = require('express'); var router = express.Router(); //接收GET請求 router.get('/', function(req, res, next) { var name = req.session.name; if(name==null name==undefined){ name = '尚未登入'; } res.render('usershow', { name: name }); }); module.exports = router;</pre>			

表 9-1-58 將員工登出

編號	1-9-2	檔案名稱	userlogout.js
功能	將員工登出		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const user = require('./utility/user');</pre>			

```
//接收POST請求

router.get('/', function(req, res, next) {
    req.session.staffphone = null;
    req.session.name = null;

    res.render('usershow', {name:'已登出'}); //傳至登出
});

module.exports = router;
```

表 9-1-59 判斷員工是否登入

編號	1-9-3	檔案名稱	userloginform.js
功能	判斷員工是否登入		
<pre>var express = require('express'); var router = express.Router(); //接收GET請求 router.get('/', function(req, res, next) { if(req.session.staffphone != null){ res.render('alreadylogin'); }else{ res.render('userloginform'); } }); module.exports = router;</pre>			

表 9-1-60 搜尋員工帳號密碼服務列表

編號	1-9-4	檔案名稱	userlogin.js
功能	呼叫搜尋員工帳號密碼服務，並判斷是否存在		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const user = require('./utility/user'); //接收POST請求 router.post('/', function(req, res, next) { var id = req.body.id; //取得帳號 var password = req.body.password; //取得密碼 user.login(id, password).then(d => { if (d==null){ req.session.staffphone = null; req.session.name = null; res.render('loginFail'); //傳至登入失敗 }else{ req.session.staffphone = d.staffphone; req.session.name = d.username; res.render('usershow', { name:d.username}); //導向使用者 console.log('已登入'); console.log(d.staffphone) console.log(d.username) } }) }); module.exports = router;</pre>			

表 9-1-61 檢查登入權限列表

編號	1-9-5	檔案名稱	checkAuth.js
功能	檢查登入權限		
<pre>var express = require('express'); var router = express.Router(); //處理GET, POST, PUT, DELETE等所有請求 router.all('/', function(req, res, next) { //檢查是否有session註記 var id = null; try{ id = req.session.staffphone; }catch(err){ id = null; } if(id===null id===undefined){ res.render('unAuth'); //導向無權限畫面 }else{ next(); //執行在app.use()中，串接在checkAuth之後的函式 } }); module.exports = router;</pre>			

表 9-1-62 函式呼叫服務列表

編號	1-9-6	檔案名稱	user.js
功能	回應上述函式呼叫服務		
<pre>'use strict'; //引用操作資料庫的物件 const sql = require('./asyncDB'); //----- // 使用者登入 //----- var login = async function(id, password){ var result; //取得員工資料 await sql('SELECT * FROM staff WHERE "staffphone"=\$1 and "password"=\$2', [id, password]) .then((data) => { if(data.rows.length > 0){ result = data.rows[0]; }else{ result = null; } }, (error) => { result = null; }); //回傳物件 return result; } //匯出 module.exports = {login};</pre>			

會員端

表 9-1-63 首頁功能列表

編號	1-10-1	函式名稱	index.js
功能	顯示各功能之按鈕		
<pre>var express = require('express'); var router = express.Router(); /* GET home page. */ router.get('/', function(req, res, next) { res.render('index', { title: 'Express' }); }); module.exports = router;</pre>			

表 9-1-64 顯示登入中會員名稱列表

編號	1-11-1	函式名稱	user_show.js
功能	顯示登入中會員名稱		
<pre>var express = require('express'); var router = express.Router(); //接收 GET 請求 router.get('/', function(req, res, next) { var name = req.session.name; if(name==null name==undefined){ name = '尚未登入'; } res.render('user_show', { name: name }); }); module.exports = router;</pre>			

表 9-1-65 將會員登出列表

編號	1-11-2	函式名稱	user_logout.js
功能	將會員登出		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const user = require('./utility/user'); //接收 POST 請求 router.get('/', function(req, res, next) { req.session.memberphone = null; req.session.name = null; res.render('user_show', {name:'已登出'}); //傳至登出 }); module.exports = router;</pre>			

表 9-1-66 判斷會員是否登入列表

編號	1-11-3	函式名稱	user_login_form.js
功能	判斷會員是否登入		
<pre>var express = require('express'); var router = express.Router(); //接收 GET 請求 router.get('/', function(req, res, next) { if(req.session.memberphone != null){ res.render('login_already'); }else{</pre>			

```

        res.render('user_login_form');
    }
});

module.exports = router;

```

表 9-1-67 搜尋會員帳號密碼服務列表

編號	1-11-4	函式名稱	user_login.js
功能	呼叫搜尋會員帳號密碼服務，並判斷是否存在		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const user = require('./utility/user'); //接收 POST 請求 router.post('/', function(req, res, next) { var id = req.body.id; //取得帳號 var password = req.body.password; //取得密碼 user.login(id, password).then(d => { if (d==null){ req.session.memberphone = null; req.session.rname = null; res.render('loginFail'); //傳至登入失敗 }else{ req.session.memberphone = d.memberphone; req.session.name = d.membername; res.render('user_show', {name:d.membername}); //導向使用者 } }) }); module.exports = router;</pre>			

表 9-1-68 檢查登入權限列表

編號	1-11-5	函式名稱	checkAuth.js
功能	檢查登入權限		
<pre>var express = require('express'); var router = express.Router(); //處理 GET, POST, PUT, DELETE 等所有請求 router.all('/', function(req, res, next) { //檢查是否有 session 註記 var id = null; try{ id = req.session.memberphone; }catch(err){ id = null; } if(id===null id===undefined){ res.render('unAuth'); //導向無權限畫面 }else{ next(); //執行在 app.use()中，串接在 checkAuth 之後的函式 } }); module.exports = router;</pre>			

表 9-1-69 函式呼叫服務列表

編號	1-11-6	函式名稱	user.js
功能	回應上述函式呼叫服務		
<pre>'use strict'; //引用操作資料庫的物件 const sql = require('./asyncDB'); //----- // 使用者登入 //----- var login = async function(id, password){ var result; //取得員工資料 await sql('SELECT * FROM member WHERE memberphone=\$1 and password=\$2', [id, password]) .then((data) => { if(data.rows.length > 0){ result = data.rows[0]; }else{ result = null; } }, (error) => { result = null; }); //回傳物件 return result; } //匯出 module.exports = {login};</pre>			

表 9-1-70 搜尋店家資料之服務列表

編號	1-12-1	函式名稱	storelist.js
功能	呼叫搜尋店家資料之服務，並回傳及顯示店家資料		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const store = require('./utility/store'); //接收 GET 請求 router.get('/', function(req, res, next) { store.list().then(data => { if(data==null){ res.render('error'); //導向錯誤頁面 }else if(data.length > 0){ res.render('store_list', {items:data}); //將資料傳給顯示頁面 }else{ res.render('storenotFound'); //導向找不到頁面 } }) }); module.exports = router;</pre>			

表 9-1-71 函式呼叫服務列表

編號	1-12-2	函式名稱	store.js
功能	回應上述函式呼叫服務		
<pre>'use strict'; //引用操作資料庫的物件 const sql = require('./asyncDB'); //----- //執行資料庫動作的函式-傳回所有店家資訊 //----- var list = async function(){ var result=[]; //console.log("查詢店家資訊"); await sql('SELECT * FROM storeinformation') .then((data) => { result = data.rows; console.log(result) ; }, (error) => { result = null; //console.log("除去錯誤") ; }); return result; } module.exports = {list}</pre>			

表 9-1-72 會員的訂單明細列表

編號	1-13-1	函式名稱	orderdetail_one.js
功能	取得某一位會員的訂單明細		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 var moment = require('moment'); const orderdetail = require('./utility/orderdetail'); //接收 GET 請求 router.get('/', function(req, res, next) { var memberphone = req.session.memberphone; //console.log(memberphone) orderdetail.list(memberphone).then(data => { //console.log(data) if(data==null){ res.render('error'); //導向錯誤頁面 }else if(data.length > 0){ res.render('orderdetail_list', {items:data}); //將資料傳給顯示頁面 }else{ res.render('notFound'); //導向找不到頁面 } }) }); module.exports = router;</pre>			

表 9-1-73 建立訂單明細列表

編號	1-13-2	函式名稱	orderdetail_add.js
功能	建立訂單明細物件並呼叫新增服務		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const orderdetail = require('./utility/orderdetail'); //接收 POST 請求 router.post('/', function(req, res, next) { console.log('222222222222222') var memberphone=req.session.memberphone; var k = req.body.foodname; k = k.split(','); var foodname= k[0]; var foodpoint= k[1]; var foodno= req.body.foodno; var customized= req.body.customized; var tableno=req.body.tableno; // 建立一個新資料物件 var newData={ memberphone: memberphone, foodname: foodname, foodpoint: foodpoint, foodno: foodno, customized: customized, tableno: tableno } console.log('3333333333333333333') console.log(newData) orderdetail.add(newData).then(d => { if (d==0){</pre>			

```

        res.render('addSuccess'); //傳至成功頁面
    }else{
        res.render('addFail');    //導向錯誤頁面
    }
    })
});

module.exports = router;

```

表 9-1-74 導至訂單明細新增頁面列表

編號	1-13-3	函式名稱	orderdetail_add_form.js
功能	導至訂單明細新增頁面		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const orderdetail = require('./utility/orderdetail'); //const e = require('express'); //接收 GET 請求 router.get('/', function(req, res, next) { console.log("11111111111111") orderdetail.getDropdownData().then(d => { if (d!=[]){ console.log(d) res.render('food_list', {result:d}); //轉至新增頁面 }else{ res.render('addFail'); //導向錯誤頁面 } }); }); module.exports = router;</pre>			

表 9-1-75 取得欲刪除訂單明細之編號列表

編號	1-13-4	函式名稱	orderdetail_remove.js
功能	取得欲刪除訂單明細之編號並呼叫刪除服務		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const orderdetail = require('./utility/orderdetail'); //接收 POST 請求 router.post('/', function(req, res, next) { //var memberphone = req.session.memberphone; //取得產品編號 var orderdetailid = req.body.orderdetailid; orderdetail.remove(orderdetailid).then(d => { if(d>=0){ res.render('removeSuccess', {result:d}); //傳至成功頁面 }else{ res.render('removeFail'); //導向錯誤頁面 } }) }); module.exports = router;</pre>			

表 9-1-76 導至訂單明細刪除頁面列表

編號	1-13-5	函式名稱	orderdetail_remove_form.js
功能	導至訂單明細刪除頁面		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const orderdetail = require('./utility/orderdetail'); //接收 GET 請求 router.get('/', function(req, res, next) { var memberphone = req.session.memberphone; //取得產品編號 var orderdetailid = req.body.orderdetailid; orderdetail.getDropdown(memberphone, orderdetailid).then(d => { if (d!=[]){ res.render('orderdetail_remove_form', {result:d}); //轉至新增頁面 }else{ res.render('addFail'); //導向錯誤頁面 } }); }); module.exports = router;</pre>			

表 9-1-77 建立訂單明細更新資料列表

編號	1-13-6	函式名稱	orderdetail_update.js
功能	建立訂單明細更新資料物件並呼叫更新服務		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const orderdetail = require('./utility/orderdetail'); //接收 POST 請求 router.post('/', function(req, res, next) { var orderdetailid = req.body.orderdetailid; var newData={ orderdetailid:orderdetailid, memberphone: req.body.memberphone, foodname: req.body.foodname, foodpoint: req.body.foodpoint, foodno: req.body.foodno, customized: req.body.customized, tableno: req.body.tableno, //ordtime: req.body.ordtime, } console.log(newData) orderdetail.update(newData).then(d => { if (d>=0){ res.render('updateSuccess', {result:d}); //傳至成功頁面 }else{ res.render('updateFail'); //導向錯誤頁面 } }) }); //匯出 module.exports = router;</pre>			

表 9-1-78 導至搜尋訂單明細編號之頁面列表

編號	1-13-7	函式名稱	orderdetail_update_no.js
功能	導至搜尋訂單明細編號之頁面		
<pre>var express = require('express'); const orderdetail = require('./utility/orderdetail'); var router = express.Router(); //接收 GET 請求 router.get('/', function(req, res, next) { var memberphone = req.session.memberphone; //取得產品編號 var orderdetailid = req.body.orderdetailid; orderdetail.getDropdown(memberphone, orderdetailid).then(d => { if (d!=[]){ res.render('orderdetail_update_no', {result:d}); //轉至新增頁面 }else{ res.render('notFound'); //導向錯誤頁面 } }); }); //匯出 module.exports = router;</pre>			

表 9-1-79 將原有訂單明細回傳至更新頁面列表

編號	1-13-8	函式名稱	orderdetail_update_form.js
功能	將原有訂單明細回傳至更新頁面之欄位		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 var moment = require('moment'); const orderdetail = require('./utility/orderdetail'); //接收 GET 請求 router.get('/', function(req, res, next) { var no = req.query.orderdetailid; var foodname = req.body.foodname; var foodpoint = req.body.foodpoint; orderdetail.query(no, foodname, foodpoint).then(d => { if (d!=null && d!=-1){ res.render('orderdetail_update_form', {result:d}); //將資料傳給更新頁面 }else{ res.render('notFound'); //導向找不到頁面 } }) }); //匯出 module.exports = router;</pre>			

表 9-1-80 函式呼叫服務列表

編號	1-13-9	函式名稱	orderdetail.js
功能	回應上述函式呼叫服務		
<pre>'use strict'; //引用操作資料庫的物件 const sql = require('./asyncDB'); const session = require('express-session'); //----- //執行資料庫動作的函式-取出單一員工 //----- var list = async function(memberphone){ var result={ }; await sql('SELECT * FROM orderdetail WHERE memberphone = \$1', [memberphone]) .then((data) => { //console.log(data.rows.length) console.log(data.rows) if(data.rows.length > 0){ result = data.rows; }else{ result = -1; } }, (error) => { result = null; }); return result; } //-----</pre>			

```

// 取出型態資料
//-----
var getDropdownData = async function(){
    //儲存下拉式選單資料

    //取回 prototype 資料
    await sql('SELECT * FROM food')
        .then((data) => {
            result.food = data.rows;
        }, (error) => {
            result.food = [];
        });

    //設定回傳資料
    var result = {};
    result.food = food;

    //回傳
    return result;
}

//-----
// 取出型態資料
//-----
var getDropdown = async function(memberphone){
    //儲存下拉式選單資料

    var orderdetail;

    await sql('SELECT * FROM orderdetail WHERE "memberphone" = $1 ORDER BY
orderdetailid', [memberphone])
        .then((data) => {
            orderdetail = data.rows;

```

```

    }, (error) => {
        result = [];
        //console.log(result)
    });

//設定回傳資料

var result = {};
result.orderdetail = orderdetail;

//回傳

return result;
}

//-----

//執行資料庫動作的函式-新增會員資料

//-----

var add = async function(newData){
    var result;

    const current = new Date();

    await sql('INSERT INTO orderdetail (foodname, foodpoint, foodno, customized,
memberphone, tableno, ordtime)  VALUES ($1, $2, $3, $4, $5, $6, $7)',
[newData.foodname, newData.foodpoint, newData.foodno, newData.customized,
newData.memberphone, newData.tableno, current])
        .then((data) => {
            result = 0;
        }, (error) => {
            result = -1;
        });

    return result;
}

//-----

```

```

// 刪除會員資料

//-----
var remove = async function(orderdetailid){
    var result;

    await sql('DELETE FROM orderdetail WHERE "orderdetailid" = $1', [orderdetailid])
        .then((data) => {
            result = data.rowCount;
        }, (error) => {
            result = -1;
        });

    return result;
}

//-----
//執行資料庫動作的函式-取得一個會員資料

//-----
var query = async function(orderdetailid){
    var result={ };

    await sql('SELECT * FROM orderdetail WHERE "orderdetailid" = $1',
[orderdetailid])
        .then((data) => {
            if(data.rows.length > 0){
                result.orderdetail = data.rows[0];
            }else{
                result.orderdetail = -1;
            }
        }, (error) => {
            result.orderdetail = null;
        });

    await sql('SELECT * FROM food ORDER BY foodname')
        .then((data) => {
            result.food = data.rows;
        }, (error) => {

```

```

        result.food = [];
    });

    //回傳
    return result;
}

//-----
// 更新會員資料
//-----
var update = async function(newData){
    var results;

    const current = new Date();

    await sql('UPDATE orderdetail SET "foodname"=$2, "foodpoint"=$3, "foodno"=$4,
"customized"=$5, "memberphone"=$6, "tableno"=$7, "ordtime"=$8 WHERE
"orderdetailid" = $1', [newData.orderdetailid, newData.foodname, newData.foodpoint,
newData.foodno, newData.customized, newData.memberphone, newData.tableno,
current])
        .then((data) => {
            results = data.rowCount;
        }, (error) => {
            results = -1;
        });

    return results;
}
module.exports = {list, add, getDropdownData, remove, query, update, getDropdown}

```

表 9-1-81 取得會員的資料列表

編號	1-14-1	函式名稱	member_one.js
功能	取得某一位會員的資料		
<pre>var express = require('express'); var router = express.Router();</pre>			

```

//增加引用函式

var moment = require('moment');
const member = require('./utility/member');

//接收 GET 請求
router.get('/', function(req, res, next) {
    var memberphone = req.session.memberphone;

    member.one(memberphone).then(data => {
        if(data==null){
            res.render('error'); //導向錯誤頁面
        }else if(data==-1){
            res.render('notFound'); //導向找不到頁面
        }else{
            res.render('member_list', {items:data}); //將資料傳給顯示頁面
        }
    })
});

module.exports = router;

```

表 9-1-82 建立會員資料列表

編號	1-14-2	函式名稱	member_add.js
功能	建立會員資料物件並呼叫新增服務		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const member = require('./utility/member');</pre>			

//接收 POST 請求

```
router.post('/', function(req, res, next) {
```

```
    var memberphone=req.body.memberphone;           //會員手機號碼
```

```
    var password=req.body.password;                   //密碼
```

```
    var membername= req.body.membername;              //會員名稱
```

```
    var gender= req.body.gender;                       //性別
```

```
    var birthday= req.body.birthday;                   //生日
```

```
// 建立一個新資料物件
```

```
var newData={
```

```
    memberphone: memberphone,           //會員手機號碼
```

```
    password: password,                  //密碼
```

```
    membername: membername,              //會員名稱
```

```
    gender:gender,                       //性別
```

```
    birthday:birthday,                   //生日
```

```
}
```

```
console.log(newData)
```

```
member.add(newData).then(d => {
```

```
    if (d==0){
```

```
        res.render('registered_success'); //傳至成功頁面
```

```
    }else{
```

```
        res.render('registered_fail');    //導向錯誤頁面
```

```
    }
```

```
    })
```

```
});
```

```
module.exports = router;
```

表 9-1-83 導至會員資料新增頁面列表

編號	1-14-3	函式名稱	member_add_form.js
功能	導至會員資料新增頁面		
<pre>var express = require('express'); var router = express.Router(); //接收 GET 請求 router.get('/', function(req, res, next) { res.render('member_add_form'); }); module.exports = router;</pre>			

表 9-1-84 搜尋會員資料列表

編號	1-14-4	函式名稱	member_list.js
功能	呼叫搜尋會員資料之服務，並回傳及顯示會員資料		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const member = require('./utility/member'); //接收 GET 請求 router.get('/', function(req, res, next) { member.list().then(data => { if(data==null){ res.render('error'); //導向錯誤頁面 }else if(data.length > 0){ console.log(data); } }); });</pre>			

```

        res.render('member_list', {items:data}); //將資料傳給顯示頁面
    }else{
        res.render('notFound'); //導向找不到頁面
    }
    })
});

module.exports = router;

```

表 9-1-85 取得欲刪除會員之帳號列表

編號	1-14-5	函式名稱	member_remove.js
功能	取得欲刪除會員之帳號並呼叫刪除服務		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const member = require('./utility/member'); //接收 POST 請求 router.post('/', function(req, res, next) { var memberphone = req.body.memberphone; //取得產品編號 member.remove(memberphone).then(d => { if(d>=0){ res.render('member_remove_Success', {results:d}); //傳至成功頁面 }else{ res.render('member_remove_Fail'); //導向錯誤頁面 } }) }); module.exports = router;</pre>			

表 9-1-86 導至會員資料刪除頁面列表

編號	1-14-6	函式名稱	member_remove_form.js
功能	導至會員資料刪除頁面		
<pre>var express = require('express'); var router = express.Router(); //接收 GET 請求 router.get('/', function(req, res, next) { res.render('member_remove_form'); }); module.exports = router;</pre>			

表 9-1-87 建立會員更新資料列表

編號	1-14-7	函式名稱	member_update.js
功能	建立會員更新資料物件並呼叫更新服務		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const member = require('./utility/member'); //接收 POST 請求 router.post('/', function(req, res, next) { var memberphone = req.body.memberphone; //取得會員手機號碼 var newData={ memberphone: memberphone, //會員手機號碼</pre>			

```

        membername: req.body.membername,          //會員名稱

        password: req.body.password,              //密碼

        gender: req.body.gender,                  //性別

        birthday: req.body.birthday,              //生日

        //creationdate: req.body.creationdate      //建立日期

    }
    console.log(newData)
    member.update(newData).then(d => {
        if (d>=0){
            res.render('member_update_Success', {results:d}); //傳至成功頁面
        }else{
            res.render('member_update_Fail');          //導向錯誤頁面
        }
    })
});

//匯出
module.exports = router;

```

表 9-1-88 導至搜尋會員帳號之頁面列表

編號	1-14-8	函式名稱	member_update_no.js
功能	導至搜尋會員帳號之頁面		
<pre>var express = require('express'); var router = express.Router(); /* GET home page. */ router.get('/', function(req, res, next) { res.render('member_update_no'); });</pre>			

//匯出

module.exports = router;

表 9-1-89 將原有會員資料回傳至更新頁面列表

編號	1-14-9	函式名稱	member_update_form.js
功能	將原有會員資料回傳至更新頁面之欄位		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 var moment = require('moment'); const member = require('./utility/member'); //接收 GET 請求 router.get('/', function(req, res, next) { var no = req.query.memberphone; member.query(no).then(d => { if (d!=null && d!=-1){ var data = { memberphone: d.memberphone, membername: d.membername, password: d.password, gender: d.gender, birthday: moment(d.birthday).format("YYYY-MM-DD"), //creationdate: moment(d.creationdate).format("YYYY-MM-DD"), } res.render('member_update_form', {item:data}); //將資料傳給更新頁面 }else{ res.render('notFound'); //導向找不到頁面 } }) }); //匯出 module.exports = router;</pre>			

表 9-1-90 函式呼叫服務列表

編號	1-14-10	函式名稱	member.js
功能	回應上述函式呼叫服務		
<pre>'use strict'; //引用操作資料庫的物件 const sql = require('./asyncDB'); //----- //執行資料庫動作的函式-傳回所有會員資料 //----- var list = async function(){ var result=""; //console.log("查看會員資訊"); await sql('SELECT * FROM member') .then((data) => { result = data.rows; //console.log(result) ; }, (error) => { result = null; //console.log("除去錯誤") ; }); return result; } //----- //執行資料庫動作的函式-取出單一會員 //----- var one = async function(memberphone){</pre>			

```

var result={ };

await sql('SELECT * FROM member WHERE memberphone = $1', [memberphone])
  .then((data) => {
    if(data.rows.length > 0){
      result = data.rows[0];
    }else{
      result = -1;
    }
  }, (error) => {
    result = null;
  });

return result;
}

//-----
//執行資料庫動作的函式-新增會員資料
//-----
var add = async function(newData){
  var result;

  console.log(newData)
  console.log(newData.memberphone)
  console.log(newData.password)
  console.log(newData.membername)
  console.log(newData.gender)
  console.log(newData.birthday)
  //console.log(newData.creationdate)

  const current = new Date();

  await sql('INSERT INTO member (memberphone, password, membername, gender,
birthday, creationdate) VALUES ($1, $2, $3, $4, $5, $6)', [newData.memberphone,
newData.password, newData.membername, newData.gender, newData.birthday, current])
    .then((data) => {
      result = 0;
    }, (error) => {
      result = -1;
    });

```



```

    });

    return result;
}

//-----
// 刪除會員資料
//-----
var remove = async function(memberphone){
    var result;

    await sql('DELETE FROM member WHERE memberphone = $1', [memberphone])
        .then((data) => {
            result = data.rowCount;
        }, (error) => {
            result = -1;
        });

    return result;
}

//-----
//執行資料庫動作的函式-取得一個會員資料
//-----
var query = async function(memberphone){
    var result={ };

    await sql('SELECT * FROM member WHERE memberphone = $1', [memberphone])
        .then((data) => {
            if(data.rows.length > 0){
                result = data.rows[0];
            }else{
                result = -1;
            }
        }, (error) => {
            result = null;
        });

    return result;
}

```

```

}

//-----

// 更新會員資料

//-----

var update = async function(newData){
    var results;

    const current = new Date();

    await sql('UPDATE member SET membername=$2, password=$3, gender=$4,
    birthday=$5, creationdate=$6 WHERE memberphone = $1', [newData.memberphone,
    newData.membername, newData.password, newData.gender, newData.birthday, current])
        .then((data) => {
            results = data.rowCount;
        }, (error) => {
            results = -1;
        });

    return results;
}

module.exports = {list, one, add, remove, query, update}

```

表 9-1-91 搜尋菜單列表

編號	1-15-1	函式名稱	food_list.js
功能	呼叫搜尋菜單之服務，並回傳及顯示菜單		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const food = require('./utility/food'); //接收 GET 請求</pre>			

```

router.get('/', function(req, res, next) {
  food.list().then(data => {
    if(data==null){

      res.render('error'); //導向錯誤頁面

    }else if(data.length > 0){
      console.log('*****')
      console.log(data);

      res.render('food_list', {result:data}); //將資料傳給顯示頁面

    }else{

      res.render('notFound'); //導向找不到頁面

    }
  })
});

module.exports = router;

```

表 9-1-92 函式呼叫服務列表

編號	1-15-2	函式名稱	food.js
功能	回應上述函式呼叫服務		
<pre>'use strict'; //引用操作資料庫的物件 const sql = require('./asyncDB'); //----- //執行資料庫動作的函式-傳回所有會員資料 //----- var list = async function(){ var result=""; //console.log("查看菜單");</pre>			

```

    await sql('SELECT * FROM food')
      .then((data) => {
        result = data.rows;
        console.log(result)
        //console.log(result)  ;
      }, (error) => {
        result = null;

        //console.log("除去錯誤")  ;

      });

    return result;
  }

module.exports = {list}

```

表 9-1-93 建立計時資料列表

編號	1-16-1	函式名稱	caltime_add.js
功能	建立計時資料物件並呼叫新增服務		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const caltime = require('./utility/caltime'); //const { now } = require('moment'); //接收 POST 請求 router.post('/', function(req, res, next) { var memberphone = req.session.memberphone; //取得到達時間</pre>			

```

// 建立一個新資料物件

//var timestamp = new Date().getTime();
//var Timestamp = (new Date()).valueOf();

var newData={
    memberphone:memberphone
}
console.log(newData)
caltime.add(newData).then(d => {
    if (d==0){
        res.render('caltime_success'); //傳至成功頁面
    }else{
        res.render('caltime_fail');    //導向錯誤頁面
    }
})
});

module.exports = router;

```

表 9-1-94 導至計時資料新增頁面列表

編號	1-16-2	函式名稱	caltime_add_form.js
功能	導至計時資料新增頁面		
<pre>var express = require('express'); var router = express.Router(); //接收 GET 請求 router.get('/', function(req, res, next) { res.render('caltime_add_form'); }); module.exports = router;</pre>			

表 9-1-95 建立結帳資料列表

編號	1-16-3	函式名稱	caltime_addend.js
功能	建立結帳資料物件並呼叫新增服務		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 const caltime = require('./utility/caltime'); //接收 POST 請求 router.post('/', function(req, res, next) { var memberphone = req.session.memberphone; //取得到達時間 var newData={ memberphone:memberphone, tttotal: req.body.tttotal, sumtotal: req.body.sumtotal, totalpoint: req.body.totalpoint, confoirm: req.body.confoirm, billtime: req.body.billtime, } caltime.query(newData).then(d => { if (d.rowCount>0){ res.render('caltime_end_success', {result:d}); //傳至成功頁面 }else{ res.render('caltime_end_fail'); //導向錯誤頁面 } }) }); module.exports = router;</pre>			

表 9-1-96 導至結帳資料頁面列表

編號	1-16-4	函式名稱	caltime_addend_form.js
功能	導至結帳資料頁面		
<pre>var express = require('express'); var router = express.Router(); //增加引用函式 var moment = require('moment'); const caltime = require('./utility/caltime'); //接收 GET 請求 router.get('/', function(req, res, next) { var memberphone = req.session.memberphone; caltime.addend(memberphone).then(d => { //console.log(d) if (d!=null && d!=-1){ res.render('caltime_addend_form'); //將資料傳給更新頁面 }else{ res.render('notFound'); //導向找不到頁面 } }) }); //匯出 module.exports = router;</pre>			

表 9-1-97 函式呼叫服務列表

編號	1-16-5	函式名稱	caltime.js
功能	回應上述函式呼叫服務		
<pre>'use strict'; //引用操作資料庫的物件 const sql = require('./asyncDB'); const { Client } = require('pg'); const session = require('express-session'); //----- // 新增開始時間 //----- var add = async function(newData){ var result; const current = new Date(); await sql('INSERT INTO calculatingtime ("memberphone", "arrivaltime") VALUES (\$1, \$2)', [newData.memberphone, current]) .then((data) => { result = 0; }, (error) => { result = -1; }); return result; } //----- // 新增結束時間 //----- var addend = async function(memberphone){ var results;</pre>			


```

const current = new Date();

await sql('UPDATE calculatingtime SET endtime = $2  where "timeserNo" =(select
max("timeserNo") from calculatingtime where memberphone =$1)', [memberphone,
current])
    .then((data) => {
        if(data.rowCount > 0){
            results = 1;
        }else{
            results = -1;
        }
    }, (error) => {
        results = null;
    });
/*
await sql('UPDATE checkout SET billtime = $2  where "serNo" =(select
max("serNo") from checkout where memberphone =$1)', [memberphone, current])
    .then((data) => {
        if(data.rowCount > 0){
            results = 1;
        }else{
            results = -1;
        }
    }, (error) => {
        results = null;
    });
*/
return results;
}

//-----
//執行資料庫動作的函式-計算總花費時間
//-----
var query = async function(data){
    let results={ };
    let result={ };
    let minutes;
    let mp=data.memberphone;
    let arrivaltime = data.arrivaltime;

```

```

let t =data.ttotal;
let glo_staymins;

let st =data.sumtotal;

let tp =data.totalpoint;

let glo_points;
let glo_confirm = data.confirm;

const current = new Date();

await sql('select * from calculatingtime WHERE memberphone= $1 and
date(arrivaltime) = current_date and date(endtime) = current_date', [data.memberphone])
    .then((data) => {
        result = data.rows[0];

        var dt1 = new Date(result.arrivaltime);
        var dt2 = new Date(result.endtime);

        var diff =(dt2.getTime() - dt1.getTime()) / 60000;
        minutes = Math.abs(Math.round(diff));
        glo_staymins =minutes;

    }, (error) => {
        result = -1;
    });

await sql('update calculatingtime set staymins= $1 WHERE memberphone= $2 and
date(arrivaltime) = current_date and date(endtime) = current_date', [minutes, mp])
    .then((data) => {
        result = data.rows;
    }, (error) => {
        result = -1;
    });

//-----
// 計算遊玩所花費的金額
//-----

```

```

    await sql('select * from calculatingtime WHERE memberphone= $1 and
date(arrivaltime) = current_date and date(endtime) = current_date', [mp])
      .then((data) => {
        results.tttotal = data.rows[0].tttotal;
      }, (error) => {
        result = -1;
      });

    await sql('select * from storeinformation')
      .then((data) => {
        result = data.rows[0];

        var atime = result.atime;
        var apoint = result.apoint;
        var lessatime = result.lessatime;
        var addapoint = result.addapoint;

        var h = parseInt(glo_staymins/atime)
        var r =  glo_staymins % atime

        if(r>=lessatime){
          h++;
          r=0;
          t = h * apoint
        }else{
          t = h * apoint + addapoint
        }

        console.log(t)

        results.tttotal =t;

      }, (error) => {
        result = null;
      });

    await sql('update calculatingtime set tttotal= $1 WHERE memberphone= $2 and
date(arrivaltime) = current_date and date(endtime) = current_date', [t, mp])
      .then((data) => {

```

```

        result = data.rowCount;
    }, (error) => {
        result = -1;
    });

//-----

// 計算餐點費用

//-----

    await sql('select memberphone, sum(foodpoint) as sumtotal from orderdetail where
date(ordtime) = current_date group by orderdetail.memberphone')
    .then((data) => {
        console.log(data.rows[0])
        if(data.rowCount > 0){
            st = data.rows[0].sumtotal;
        }else{
            st = 0;
        }
        results.sumtotal=st;
    }, (error) => {
        result = -1;
    });

    await sql('update orderdetail set sumtotal= $1 WHERE memberphone= $2 and
date(ordtime) = current_date', [st, mp])
    .then((data) => {
        result = data.rowCount;
    }, (error) => {
        result = -1;
    });

//-----

// 判斷是否點餐和計算總花費金額及轉換成點數

//-----

    await sql('select * from checkout ')
    .then((data) => {
        result = data.rows[0];

        if(st > 0){

```

```

        tp = parseInt(t) + parseInt(st);
    }else{
        tp = t;
        result = -1;
    }
    results.totalpoint=tp;

}, (error) => {
    result = null;
    console.log(tp)
});

await sql('INSERT INTO checkout (memberphone, totalpoint, billtime) VALUES
($1, $2, $3)', [mp, tp, current])
    .then((data) => {
        result = 0;
    }, (error) => {
        result = -1;
    });

await sql('update checkout set totalpoint= $1 WHERE memberphone= $2', [tp, mp])
    .then((data) => {
        result = data.rowCount;
    }, (error) => {
        result = -1;
    });

//-----
// 判斷點數是否足夠及計算剩餘點數
//-----

await sql('select * from member WHERE memberphone= $1', [mp])
    .then((data) => {
        result = data.rows[0];
        glo_points = result.points;
    }, (error) => {
        result = -1;
    });

await sql('select * from calculatingtime WHERE memberphone= $1 and

```

```

date(endtime) = current_date', [mp])
    .then((data) => {
        result = data.rows[0].glo_confirm;
    }, (error) => {
        result = -1;
    });

    await sql('select * from checkout WHERE memberphone= $1 and date(billtime) =
current_date', [mp])
    .then((data) => {
        result = data.rows[0];

        if(glo_points > tp){
            glo_points = glo_points - tp;
            glo_confirm = '是';
        }else{
            glo_confirm = '否';
        }
        results.glo_confirm = glo_confirm;
    }, (error) => {
        result = null;
    });

    await sql('update member set points= $1 WHERE memberphone= $2', [glo_points,
mp])
    .then((data) => {
        results.rowCount = data.rowCount;
    }, (error) => {
        result = -1;
    });

    await sql('update calculatingtime set confirm= $1 WHERE memberphone= $2 and
date(endtime) = current_date', [glo_confirm, mp])
    .then((data) => {
        result = data.rowCount;
        console.log(result)
        console.log(results)
    });

```

```
    }, (error) => {  
        result = -1;  
  
    });  
    return results;  
}
```

```
module.exports = {add, addend, query};
```

十、測試模型

10-1 測試計畫

本組的測試計畫會分為員工端網頁及會員端網頁兩部分，以下為主要功能測試：

一、員工端網頁測試計畫

1.員工註冊

1-1.點選「我是員工」，點選「註冊」並輸入註冊所需資料，最後點選「註冊」按鈕提交註冊資料。

2.員工登入

2-1.點選「我是員工」，點選「登入」並輸入手機號碼及密碼，最後點選「登入」按鈕。

3.員工

3-1.查看員工資訊：點選「員工」即可看到員工的基本資訊。

3-2.修改員工資訊：點選「員工」，再點選「修改員工資訊」，轉至修改頁面輸入資料再點選「修改」按鈕提交修改內容。

4.店家資訊

4-1.查看店家資訊：點選「店家資訊」，即可查看店家基本資訊。

4-2.新增店家資訊：點選「店家資訊」，再點選「新增店家資訊」，轉至新增頁面輸入需要新增的店家資料，最後點選「新增」按鈕即可提交新增資料。

4-3.修改店家資訊：點選「店家資訊」，再點選「修改店家資訊」，轉至修改頁面於下拉選單選取欲修改的店家資訊，輸入資料再點選「修改」按鈕提交修改內容。

4-4.刪除店家資訊：點選「店家資訊」，再點選「刪除店家資訊」，轉至刪除頁面於下拉選單選取欲刪除的店家資訊，最後點選「刪除」按鈕。

5.菜單

5-1.查看餐點資訊：點選「菜單」，即可查看餐點資訊。

5-2.新增餐點資訊：點選「菜單」，再點選「新增餐點資訊」，轉至新增頁面輸入需要新增的餐點資訊，最後點選「新增」按鈕即可提交新增資料。

5-3.修改餐點資訊：點選「菜單」，再點選「修改餐點資訊」，轉至修改頁面於下拉選單選取欲修改的餐點資訊，輸入資料再點選「更新」按鈕提交修改內容。

5-4.刪除餐點資訊：點選「菜單」，再點選「刪除餐點資訊」，轉至刪除頁面於下拉選單選取欲刪除的餐點資訊，最後點選「刪除」按鈕。

6.餐點類別資訊

6-1.查看餐點類別資訊：點選「餐點類別資訊」，即可查看餐點類別資訊。

6-2.新增餐點類別資訊：點選「餐點類別資訊」，再點選「新增餐點類別資訊」，轉至新增頁面輸入需要新增的餐點類別資訊，最後點選「新增」按鈕即可提交新增資料。

6-3.修改餐點類別資訊：點選「餐點類別資訊」，再點選「修改餐點類別資訊」，轉至修改頁面於下拉選單選取欲修改的餐點類別資訊，輸入資料再點選「更新」按鈕提交修改內容。

6-4.刪除餐點類別資訊：點選「餐點類別資訊」，再點選「刪除餐點資訊」，轉至刪除頁面於下拉選單選取欲刪除的餐點類別資訊，最後點選「刪除」按鈕。

7.會員餐點訂單

7-1.查看當日會員餐點訂單明細：點選「查看會員餐點訂單」，即可查看當日會員餐點訂單明細。

8.會員消費明細

8-1.查看當日會員消費明細：點選「查看會員消費明細」，即可查看當日會員消費明細。

9.儲值點數

9-1.替會員儲值點數：點選「儲值點數」，轉至儲值頁面定輸入會員的電話號碼及儲值金額，最後點選「儲值」按鈕提交儲值資料。

10.登出

10-1.登出：點選「登出」，顯示已登出。

二、會員端網頁測試計畫

1.會員註冊

1-1.點選「我是會員」，點選「註冊」並輸入註冊所需資料，最後點選「註冊」按鈕提交註冊資料。

2.會員登入

2-1.點選「我是會員」，點選「登入」並輸入手機號碼及密碼，最後點選「登入」按鈕。

3.店家資訊

3-1.查看店家資訊：點選「店家資訊」，即可查看店家基本資訊。

4.開始遊玩

4-1.計算開始遊玩時間：點選「開始遊玩」，最後點選「計時開始」按鈕提交現在時間。

5.點餐

5-1.查看餐點資訊：點選「點餐」，即可查看餐點資訊。

5-2.新增餐點：點選「點餐」轉至新增頁面輸入需要新增的餐點資訊，最後點選「新增」按鈕即可提交新增資料。

6.訂單明細

6-1.查看餐點的訂單明細：點選「訂單明細」，即可查看餐點的訂單明細。

6-2.修改餐點的訂單明細：點選「訂單明細」，再點選「修改餐點」，轉至修改頁面於下拉選單選取欲修改的訂單，點選「查詢」按鈕，輸入資料再點選「更新」按鈕提交修改內容。

6-3.刪除餐點資訊：點選「訂單明細」，再點選「刪除餐點」，轉至刪除頁面於下拉選單選取欲刪除的餐點資訊，最後點選「刪除」按鈕。

7.結帳

7-1. 計算結束遊玩時間和總共花費的金額及扣除點數：點選「結束遊玩」，最後點選「結帳」按鈕提交現在時間和開始計算費用及扣除點數。

8.會員

8-1.查看會員資訊：點選「會員資料」即可看到會員的基本資訊。

8-2.修改會員資訊：點選「會員資料」，再點選「修改會員」，轉至修改頁面輸入手機號碼再點選「查詢」按鈕，輸入資料再點選「更新」按鈕提交修改內容。

8-3.刪除會員資訊：點選「會員資料」，再點選「刪除會員」，轉至刪除頁面輸入欲刪除的帳號，最後點選「刪除」按鈕。

9.登出

9-1.登出：點選「登出」，顯示已登出。

10-2 測試個案與測試結果資料

一、 員工端網頁測試個案與測試結果資料

表 10-2-1 員工註冊

1.員工註冊	
功能名稱	1-1.員工註冊
測試流程	1.至首頁點選「我是員工」。 2.轉至員工頁面，點選「註冊」並輸入註冊所需資料。 3.最後點選「註冊」提交註冊資料。
預期成果	1.使用者註冊成功後獲得員工帳號，資料庫成功新增員工資料至員工資料表，跳出「註冊成功」之訊息。
執行成果	測試成功，執行成果如預期結果。

表 10-2-2 員工登入

2.員工登入	
功能名稱	2-1.員工登入
測試流程	1.至首頁點選「我是員工」。 2.轉至員工頁面，點選「登入」並輸入登入手機號碼及密碼。

	3.最後點選「登入」提交登入資料	
預期成果	1.員工登入成功後，會跳出「目前員工」之訊息，即可使用員工端所有管理功能。	1.手機號碼或密碼輸入錯誤，皆會顯示「登入失敗」之提醒，若未登入就點選員工管理功能則會跳出「尚未登入，無使用權限」之提醒。
執行成果	測試成功，執行成果如預期結果。	

表 10-2-3 員工

3.員工	
功能名稱	3-1.查看員工資訊
測試流程	1.員工身份登入。 2.點選「員工」。
預期成果	1.自動從資料庫之員工資料表抓取登入中的員工基本資訊，即可看到員工基本資訊。
執行成果	測試成功，執行成果如預期結果。
功能名稱	3-2.修改員工資訊

測試流程	1.員工身份登入 2.點選「員工」 3.再點選「修改員工資訊」輸入需要修改的資料。 4.最後點選「修改」按鈕。	
預期成果	1.成功修改該資料庫之員工資料表的員工基本資訊，跳出「修改成功」之提醒。	1.修改員工資訊失敗，跳出修改失敗之提醒。
執行成果	測試成功，執行成果如預期結果。	

表 10-2-4 店家資訊

4.店家資訊	
功能名稱	4-1.查看店家資訊
測試流程	1.員工身份登入。 2.點選「店家資訊」。
預期成果	1.從資料庫之店家資料表抓取店家資料，即可看到店家基本資訊。
執行成果	測試成功，執行成果如預期結果。
功能名稱	4-2. 新增店家資訊
測試流程	1.員工身份登入

	2.點選「店家資訊」 3.再點選「新增店家資訊」輸入需要新增的資訊。 4.最後點選「新增」按鈕。	
預期成果	1.成功新增店家資訊到資料庫之店家資料表，跳出「新增成功」之提醒。	1.新增店家資訊失敗，跳出新增失敗之提醒。
執行成果	測試成功，執行成果如預期結果。	
功能名稱	4-3.修改店家資訊	
測試流程	1.員工身份登入 2.點選「店家資訊」 3.再點選「修改店家資訊」。 4.選取欲修改的店家資訊，輸入需要修改的資訊。 5.最後點選「修改」按鈕。	
預期成果	1.成功修改店家資訊至資料庫之店家資料表，跳出「修改成功」之提醒。	1.修改店家資訊失敗，跳出修改資料失敗之提醒。
執行成果	測試成功，執行成果如預期結果。	
功能名稱	4-4.刪除店家資訊	

測試流程	1.員工身份登入 2.點選「店家資訊」 3.再點選「刪除店家資訊」。 4.選取欲刪除的店家資訊。 5.最後點選「刪除」按鈕。	
預期成果	1. 成功刪除資料庫之店家資料表的店家資訊，跳出「刪除成功」之提醒。	1.刪除店家資訊失敗，跳出刪除資料失敗之提醒。
執行成果	測試成功，執行成果如預期結果。	

表 10-2-5 菜單

5.菜單	
功能名稱	5-1.查看餐點資訊
測試流程	1.員工身份登入。 2.點選「菜單」。
預期成果	1.從資料庫之菜單資料表抓取餐點資訊，即可看到餐點資訊。
執行成果	測試成功，執行成果如預期結果。
功能名稱	5-2. 新增餐點資訊

測試流程	1.員工身份登入 2.點選「餐點」 3.再點選「新增餐點資訊」輸入需要新增的餐點。 4.最後點選「新增」按鈕。	
預期成果	1.成功新增餐點資訊到資料庫之菜單資料表，跳出「新增成功」之提醒。	1.新增餐點資訊失敗，跳出新增失敗之提醒。
執行成果	測試成功，執行成果如預期結果。	
功能名稱	5-3.修改餐點資訊	
測試流程	1.員工身份登入 2.點選「菜單」 3.再點選「修改餐點資訊」。 4.選取欲修改的餐點資訊，輸入需要修改的餐點。 5.最後點選「修改」按鈕。	
預期成果	1.成功修改資料庫之菜單資料表的餐點資訊，跳出「修改成功」之提醒。	1.修改餐點資訊失敗，跳出修改資料失敗之提醒。
執行成果	測試成功，執行成果如預期結果。	

功能名稱	5-4.刪除餐點資訊	
測試流程	1.員工身份登入 2.點選「菜單」 3.再點選「刪除餐點資訊」。 4.選取欲刪除的餐點。 5.最後點選「刪除」按鈕。	
預期成果	1. 成功刪除資料庫之菜單資料表的餐點資訊，跳出「刪除成功」之提醒。	1.刪除餐點資訊失敗，跳出刪除資料失敗之提醒。
執行成果	測試成功，執行成果如預期結果。	

表 10-2-6 餐點類別資訊

6.餐點類別資訊		
功能名稱	6-1.查看餐點類別資訊	
測試流程	1.員工身份登入。 2.點選「餐點類別資訊」。	
預期成果	1.從資料庫之餐點類別資料表抓取餐點類別資訊，即可看到餐點類別資訊。	

執行成果	測試成功，執行成果如預期結果。	
功能名稱	6-2. 新增餐點類別資訊	
測試流程	1.員工身份登入 2.點選「餐點類別資訊」 3.再點選「新增餐點類別資訊」輸入需要新增的餐點類別。 4.最後點選「新增」按鈕。	
預期成果	1.成功新增餐點類別資訊到資料庫之餐點類別資料表，跳出「新增成功」之提醒。	1.新增餐點類別資訊失敗，跳出新增失敗之提醒。
執行成果	測試成功，執行成果如預期結果。	
功能名稱	6-3.修改餐點類別資訊	
測試流程	1.員工身份登入 2.點選「餐點類別資訊」 3.再點選「修改餐點類別資訊」。 4.選取欲修改的餐點類別資訊，輸入需要修改的餐點類別。 5.最後點選「修改」按鈕。	
預期成果	1.成功修改資料庫之餐點類別資料表的餐點類別資訊，跳出「修	1.修改餐點類別資訊失敗，跳出修改資料失敗之提醒。

	改成功」之提醒。	
執行成果	測試成功，執行成果如預期結果。	
功能名稱	6-4.刪除餐點類別資訊	
測試流程	1.員工身份登入 2.點選「餐點類別資訊」 3.再點選「刪除餐點類別資訊」。 4.選取欲刪除的餐點類別。 5.最後點選「刪除」按鈕。	
預期成果	1.成功刪除資料庫之餐點類別資料表的餐點類別資訊，跳出「刪除成功」之提醒。	1.刪除餐點類別資訊失敗，跳出刪除資料失敗之提醒。
執行成果	測試成功，執行成果如預期結果。	

表 10-2-7 會員餐點訂單

7.會員餐點訂單		
功能名稱	7-1.查看會員餐點訂單明細	
測試流程	1.員工身份登入。 2.點選「會員餐點訂單」。	

預期成果	1.從資料庫之訂單明細資料表抓取當日會員餐點訂單，即可看到當日的訂單。	1.當日無會員點餐，則顯示「今日還沒有會員點餐」之提醒。
執行成果	測試成功，執行成果如預期結果。	

表 10-2-8 會員消費明細

8.會員消費明細		
功能名稱	8-1.查看會員消費明細	
測試流程	1.員工身份登入。 2.點選「會員消費明細」。	
預期成果	1.從資料庫之消費明細資料表抓取當日會員消費明細，即可看到當日的會員消費明細。	1.當日無會員結帳，則顯示「今日還沒有會員結帳」之提醒。
執行成果	測試成功，執行成果如預期結果。	

表 10-2-9 儲值點數

9.儲值點數	
功能名稱	9-1.儲值點數

測試流程	1.員工身份登入。 2.點選「儲值點數」。 3.輸入欲儲值的會員電話號碼。 4.輸入儲值金額，點選「儲值」。	
預期成果	1.成功新增會員儲值紀錄至資料庫之儲值資料表，並更新會員資料表之剩餘點數，跳出「儲值成功」之提醒。	1.儲值失敗，跳出「儲值失敗之提醒」。
執行成果	測試成功，執行成果如預期結果。	

表 10-2-10 登出

10.登出	
功能名稱	10-1.登出
測試流程	1.員工身份登入。 2.點選「登出」。
預期成果	1.清除網頁之登入資訊，並導至首頁。
執行成果	測試成功，執行成果如預期結果。

二、 會員端網頁測試個案與測試結果資料

表 10-2-11 會員註冊

1.會員註冊	
功能名稱	1-1.會員註冊
測試流程	1.至首頁點選「我是會員」。 2.轉至會員頁面，點選「註冊」並輸入註冊所需資料。 3.最後點選「註冊」提交註冊資料。
預期成果	1.使用者註冊成功後獲得會員帳號，資料庫成功新增會員資料至會員資料表，跳出「註冊成功」之訊息。
執行成果	測試成功，執行成果如預期結果。

表 10-2-12 會員登入

2.會員登入	
功能名稱	2-1.會員登入
測試流程	1.至首頁點選「我是會員」。 2.轉至會員頁面，點選「登入」並輸入登入手機號碼及密碼。 3.最後點選「登入」提交登入資料

預期成果	1.會員登入成功後，會跳出「目前員工」之訊息，即可以使用會員端所有管理功能。	1.手機號碼或密碼輸入錯誤，皆會顯示「登入失敗」之提醒，若未登入就點選會員管理功能則會跳出「尚未登入，無使用權限」之提醒。
執行成果	測試成功，執行成果如預期結果。	

表 10-2-13 店家資訊

3.店家資訊	
功能名稱	3-1.查看店家資訊
測試流程	1.會員身份登入。 2.點選「店家資訊」。
預期成果	1.從資料庫之店家資料表抓取店家資料，即可看到店家基本資訊。
執行成果	測試成功，執行成果如預期結果。

表 10-2-14 開始遊玩

4.開始遊玩	
功能名稱	4-1.計算開始遊玩時間

測試流程	1.會員身份登入。 2.點選「開始遊玩」。 3. 最後點選「計時開始」按鈕提交現在時間。
預期成果	1.將現在時間傳到資料庫的到達時間欄位。
執行成果	測試成功，執行成果如預期結果。

表 10-2-15 點餐

5.點餐	
功能名稱	5-1.查看餐點
測試流程	1.會員身份登入。 2.點選「點餐」。
預期成果	1.從資料庫之菜單資料表抓取餐點資訊，即可看到餐點資訊。
執行成果	測試成功，執行成果如預期結果。
功能名稱	5-2. 新增餐點
測試流程	1.會員身份登入。 2.點選「點餐」。 3.輸入需要新增的餐點，再點選「新增」。

預期成果	1.成功新增餐點資訊到資料庫之 訂單明細資料表，跳出「新增成 功」之提醒。	1.新增餐點資訊失敗，跳出新增 失敗之提醒。
執行成果	測試成功，執行成果如預期結果。	

表 10-2-16 訂單明細

6. 訂單明細	
功能名稱	6-1. 查看餐點的訂單明細
測試流程	1.會員身份登入。 2.點選「訂單明細」。
預期成果	1.從資料庫之訂單明細資料表抓取餐點的訂單明細，即可看到餐點 的訂單明細。
執行成果	測試成功，執行成果如預期結果。
功能名稱	6-2. 修改餐點的訂單明細
測試流程	1.會員身份登入 2.點選「訂單明細」 3.再點選「修改餐點」。 4.選取欲修改餐點的編號，點選「查詢」按鈕。

	<p>5.選取欲修改的餐點資訊，輸入需要修改的餐點。</p> <p>6.最後點選「更新」按鈕。</p>	
預期成果	1.成功修改資料庫之訂單明細資料表的餐點資訊，跳出「修改成功」之提醒。	1.修改餐點資訊失敗，跳出修改資料失敗之提醒。
執行成果	測試成功，執行成果如預期結果。	
功能名稱	6-4.刪除餐點類別資訊	
測試流程	<p>1. 會員身份登入</p> <p>2.點選「訂單明細」</p> <p>3.再點選「刪除餐點」。</p> <p>4.選取欲刪除的餐點的編號。</p> <p>5.最後點選「刪除」按鈕。</p>	
預期成果	1.成功刪除資料庫之訂單明細資料表的餐點資訊，跳出「刪除成功」之提醒。	1.刪除餐點資訊失敗，跳出刪除資料失敗之提醒。
執行成果	測試成功，執行成果如預期結果。	

表 10-2-17 結帳

7.結帳		
功能名稱	7-1.計算結束遊玩時間和總共花費的金額及扣除點數	
測試流程	1.會員身份登入。 2.點選「結束遊玩」。 3. 點選「結帳」按鈕。	
預期成果	1. 從資料庫之計算時間資料表 抓取當日到達時間和結束時 間相減，即可算出停留時間。 2. 從資料庫之店家資料表抓取 店家計算遊玩時間的費用，再 抓取計算時間資料表的停留 時間，將停留時間乘以店家計 算遊玩時間的費用，即可算出 遊玩所花費的金額。 3. 從資料庫之訂單明細資料表 抓取會員當天所點的餐點費 用做總和，即可算出當天會員 的總餐點費用。	1.結帳失敗，跳出結帳失敗之提 醒。

	<p>4. 從資料庫之計算時間資料表 抓取會員當日遊玩所花費的金額，再從資料庫之點單明細資料表抓取會員當日點餐所花費的金額，將會員當日遊玩所花費的金額和會員當日點餐所花費的金額相加，計算出會員當日花費的總金額。</p> <p>5. 從資料庫之會員資料表查看會員的點數夠不夠扣除當日花費的總金額，如果點數不夠，網頁會跳出警告視窗，請會員先行至櫃檯儲值，如果點數夠，則會直接扣除會員點數，並會顯示會員當日遊玩所花費的金額和點餐所花費的金額和總花費的金額及是否完成結帳給會員看。</p>	
執行成果	測試成功，執行成果如預期結果。	

表 10-2-18 會員

8.會員		
功能名稱	8-1.查看會員資訊	
測試流程	1.會員身份登入。 2.點選「會員資料」。	
預期成果	1.從資料庫之會員資料表抓會員資料，即可看到會員的基本資訊。	
執行成果	測試成功，執行成果如預期結果。	
功能名稱	8-2.修改會員資訊	
測試流程	1.會員身份登入 2.點選「會員資料」 3.再點選「修改會員」。 4.輸入欲修改的帳號，點選「查詢」按鈕。 5.選取欲修改的會員資訊，輸入需要修改的會員資料。 6.最後點選「更新」按鈕。	
預期成果	1.成功修改資料庫之會員資料表的會員資訊，跳出「更新成功」之提醒。	1.修改會員資料失敗，跳出修改資料失敗之提醒。

執行成果	測試成功，執行成果如預期結果。	
功能名稱	8-3.刪除餐點類別資訊	
測試流程	2. 會員身份登入 2.點選「會員資料」 3.再點選「刪除會員」。 4.輸入欲刪除的會員。 5.最後點選「刪除」按鈕。	
預期成果	1.成功刪除資料庫之會員資料表的會員，跳出「刪除成功」之提醒。	1.刪除會員資料失敗，跳出刪除資料失敗之提醒。
執行成果	測試成功，執行成果如預期結果。	

表 10-2-19 登出

9.登出	
功能名稱	9-1.登出
測試流程	1.會員身份登入。 2.點選「登出」。
預期成果	1.清除網頁之登入資訊，並導至首頁。
執行成果	測試成功，執行成果如預期結果。

第十一章 操作手冊

表 11-1-1 網頁操作手冊表

網頁	
1. 在電腦中開啟任一瀏覽器；若用手機操作可開啟任一瀏覽器或直接開啟掃描器掃描 QR CODE	
	
2. 輸入網址或直接掃描 QR CODE 即可進入網頁：	
會員端 https://catchumember.herokuapp.com/	
員工端 https://catchustaff.herokuapp.com/	
會員端	員工端
	



第十二章 使用手冊

一、 網頁使用手冊

表 12-1 登入及註冊

登入及註冊
開啟「CatchU」進入頁面後，非會員可註冊新帳號，會員可直接登入。

表 12-2 會員及員工資訊

會員資訊

進行登入後，會員將鼠標移至功能列點選會員資訊，即可對會員資料進行查看及更改。

chumember.herokuapp.com

桌助你CATCHU

MENU

餐點資料

刪除會員

修改會員

會員帳號	會員名稱	性別	生日	建立日期
11111	陳小潔	F	Tue Sep 08 1998 00:00:00 GMT+0000 (Coordinated Universal Time)	Tue Nov 24 2020 00:00:00 GMT+0000 (Coordinated Universal Time)

員工資訊

進行登入後，員工將鼠標移至功能列點選員工資訊，即可對員工資料進行查看及更改。

catchstaff.herokuapp.com/staff/one

桌助你CATCHU

返回首頁

員工資訊

修改員工資料

電話號碼	0982222222
員工名稱	ddd
暱稱	nickk

表 12-3 店家資訊

會員端

進行登入後，會員將鼠標移至功能列點選店家資訊，即可查看店家資訊。

員工端

進行登入後，員工將鼠標移至功能列點選店家資訊，即可對店家資料進行修改。

chumember.herokuapp.com

桌助你CATCHU

MENU

店家資訊

店家名稱	店家地址	電話號碼	空桌數量	營業時間	wifi	插座	提供餐點
夢桌遊一號店	台北市松江路69巷	123455	5	14:00-23:00	Y	Y	Y

Catchu 店家資訊

catchustaff.herokuapp.com/store/list

桌助你CATCHU

返回首頁

新增店家資訊

刪除店家資訊

修改店家資訊

店家編號	店家名稱	店家地址	聯絡電話	空桌數	營業時間	Wi-Fi	提供插座	提供餐點	外食	收費標準
1	夢桌遊一號店	台北市松江路69巷	123455	5	14:00-23:00	Y	Y	Y	N	一小时150

表 12-4 點餐及菜單



會員端	員工端
<p>進行登入後，會員將鼠標移至功能列點選點餐，即可進行點餐及查看菜單。</p>	<p>進行登入後，員工將鼠標移至功能列點選菜單，即可對菜單資料進行查看及更改。</p>
	

表 12-5 訂單明細

會員端	員工端
進行登入後，會員將鼠標移至功能列點選訂單明細，即可對訂單明細進行查看及更改。	進行登入後，員工將鼠標移至功能列點選查看會員餐點訂單明細，即可查看會員今日的餐點訂單明細。
	

表 12-6 員工端之查看會員消費明細

查看會員消費明細

進行登入後，員工將鼠標移至功能列點選查看會員消費明細，即可查看會員今日的消費明細。



The screenshot shows a web browser window with the URL `catchustaff.herokuapp.com/checkout/list`. The page header includes the logo '桌助你 CATCHU' and a link '返回首頁'. The main content area is titled '會員消費明細' and features a table with the following data:

流水號	會員電話	消費點數	結帳時間
1010	11111	50	2020-11-24 10:22:48
1009	11111	50	2020-11-24 10:21:16
1012	0922222222	170	2020-11-24 02:29:24

表 12-7 員工端之餐點類別資訊

餐點類別資訊							
進行登入後，員工將鼠標移至功能列點選查看會員消費明細，即可查看會員今日的消費明細。							
 <table border="1"> <thead> <tr> <th>類別編號</th><th>類別名稱</th></tr> </thead> <tbody> <tr> <td>1013</td><td>熟食</td></tr> <tr> <td>1014</td><td>手沖咖啡</td></tr> </tbody> </table>		類別編號	類別名稱	1013	熟食	1014	手沖咖啡
類別編號	類別名稱						
1013	熟食						
1014	手沖咖啡						

表 12-8 員工端之儲值點數

儲值點數
<p>進行登入後，員工將鼠標移至功能列點選儲值，即可幫會員進行點數儲值。</p>


表 12-9 會員端之開始遊玩

開始遊玩	
進行登入後，會員將鼠標移至功能列點選開始遊玩，進入開始遊玩頁面後，點選計時開始，就會跳轉至開始計頁面。	
	

表 12-10 會員端之結束遊玩

結束遊玩

進行登入後，會員將鼠標移至功能列點選結束遊玩，進入結束遊玩頁面後，點選結帳，就會跳轉至結帳完成頁面。

A screenshot of a web browser showing the '結束遊玩' (End Play) page. The browser's address bar displays 'chumember.herokuapp.com'. The page header features the '桌助你 CATCHU' logo and a 'MENU' button. The main content area is a dark rectangle with the text '結束遊玩' in white, followed by a '結帳' (Checkout) button. At the bottom of this rectangle, it says '© 2020 CatchU|桌助你 | Design by 109408 組'. A small white button with an upward arrow is in the bottom right corner.

會員點數足夠扣除本次花費點數

A screenshot of a web browser showing the '結帳完成!' (Checkout Complete!) page. The browser's address bar displays 'chumember.herokuapp.com'. The page header features the '桌助你 CATCHU' logo and a 'MENU' button. The main content area is a dark rectangle with the text '結帳完成!' in white. Below this text is a table with four columns: '遊玩花費金額', '點餐花費金額', '總花費金額', and '是否結帳完成'. The table contains one row of data: 50, 120, 170, and 是. At the bottom of the rectangle, it says '© 2020 CatchU|桌助你 | Design by 109408 組'. A small white button with an upward arrow is in the bottom right corner.

會員點數不足扣除本次花費點數，會先跳出警告視窗提醒會員點數不足，須先至櫃台進行儲值，再來點選結束遊玩。



第十三章 感想

N1066418 周育德

歷經了快一年的時間，專題終於結束，從一開始羨慕其他學校只有一學期的專題時間到現在終於也可以放下重擔，記得一開始在構思題目的時候心情非常忐忑，因為想著不知道如何開始，程式的能力是否可以運用在專題上面，但非常幸運可以遇到我們的專題指導老師林宏仁老師，構思完題目後老師都會提供我們不同的方向，甚至不同的程式讓我們去練習，從懵懵懂懂到後面拼拼湊湊慢慢拼湊出一個有邏輯支撐的網頁系統，不枉費我們一直以來每週專題開會並且固定留在學校的艱辛過程，而且由於組員人數較少，所以分擔的任務會比較多，UML diagram、後端、SQL 都有負責到，所以感覺非常充實有成就感，最後謝謝組員和指導老師的努力和耐心，努力撐到了最後。

N1066442 葉芝秀

經過了這將近一年的專題時間，我在做專題的期間更深入的學習及了解在大學期間所學的，不只有網頁前後端的撰寫，也有資料庫的建置、設計及關連，還有 UML 的設計等等；從一開始完全不清楚要如何寫出一個網頁，到現在我能夠看的懂部分的程式碼及能寫出後端的功能以及能夠建置一個資料庫等等，雖然花了很多的時間及心思去學習，但是最後看到自己把好幾個網頁功能寫出來並且有辦法執行成功，不只讓我增添了許多成就感，也增進了我的程式能力。

我非常的感謝我們專題的指導老師，宏仁老師，因為知道我們這組的程式能力不是很深厚，所以特地讓我們利用假日時間到夜間部隨班學習，好讓我們有點基礎能完成這次的專題；每當我在寫程式的過程中遇到了問題，老師不只會教導我解決的方法，也會教我若下次有類似的問題甚至是其它問題，自己要如何找出是程式哪一部份出現了錯誤或者是如何自己解決等等，讓我多了很多可以自行思考的空間，使我的程式能力比以前進步非常地多。

N1066432 林琪容

鄰近專題快接近尾聲，心中的緊張與壓力也慢慢消失，這一年來經歷了大大小小的挫折也曾有想放棄的念頭，但在林宏仁老師與同學的鼓勵下仍是堅持完成原本陌生而不知從何開始著手的程式，在此要先感謝林宏仁老師的指導與鼓勵，當我們撰寫程式時遇到錯誤或卡頓時老師總能細心指導我們並一起找到錯誤。在這不長不短的一年中兩位組員的流失與放棄導致人數不足而剩餘的三人扛起整個專題，大家從一開始的絕望與無助到慢慢跟上節奏，真的非常感謝剩餘的組員們，即便大家的任務更多更重仍毫無怨言，也不枉費暑假一周兩次的專題討論，加上宏仁老師的課程指導才完成我們的專題製作，真心感謝。

10646037 關宇辰

將近了一年的籌備策劃，專題終於要在今日落下序幕。從一開始的主題構想到後來的系統架設，測試，運行，這一路走來實在不易。一開始構想主題的時候，饒了很多圈，始終沒找到滿意的。從那一刻起，才發現專題策劃並不是那麼好做的。很慶幸林宏仁老師能作為我們的指導老師，時刻地給予我們協助，並指導我們從何開始。經過反復的測試與修改，系統架設也慢慢的成型，我們也漸漸地看到了一絲曙光。當然，我們深知這並不表示我們已經取得成功。憑藉着組員們的努力，我們一步一步地完成前端，UML 系統架構，SQL，以及後端。最後，感謝組員的無私付出和努力 and 指導老師的細心栽培。

第十四章 參考資料

1. 網頁前後端教學。 <https://www.w3schools.com/js/>
2. bootstrap 前端元件套件。 <https://getbootstrap.com/>
3. 會員計算時間教學
 - <https://codertw.com/%E5%89%8D%E7%AB%AF%E9%96%8B%E7%99%BC/269606/>
 - https://blog.csdn.net/qq_32584661/article/details/81632920
 - <https://www.cnblogs.com/kissdodog/p/5419913.html>
4. 網頁連接資料庫教學。 <https://www.fooish.com/sql/in.html>

文件附錄一——初評之評審建議與修正情形

評審建議事項	修正情形
1. 簡報流程、時間控制，宜再練習修正。	已加強訓練簡報流程及時間控制。
2. 文件請修正：封面的題目；表格的標題應放在表的上方，圖標題在圖下方。	已針對此問題作修正。
3. 是否有從桌遊店、去桌遊店的實際使用者角度去思考所有功能&流程？市場上確實有這些功能需求嗎？	已實際探訪桌遊店家並了解其功能、需求。
4. 組員分工宜再調整，每個人在專案發展期間，都要有相同程度的投入。	已針對此問題作修正。
5. 建議該專題的名稱修改，以符合專題的內容。	已修正專題名稱並更改主軸內容。
6. 專案規劃之工作宜加強。	已針對此問題作修正。
7. 進度延遲太多。	已針對此問題作修正。
8. 程式整合情況應加強。	已針對此問題作修正。
9. 專題只看結果，因此需有實務之產出。	已針對此問題作修正。
10. 若是想要改善的是指店內的計時和遊戲選購的部分，可以更著重這邊的內容	已在文件中加強敘述儲值點數(含計時)與點餐功能。

說明。	
11. LineBOT 的整合在這個系統中顯得是突兀的，因為這跟行銷有關，但專題的配置比重這塊明顯很少，卻又一直放在重點的說明上。	經小組討論評估已將 LineBOT 整合部分移除，並著重於系統功能上。
12. 使用者的訪談建議可以增加對象，不是只問說是不是沒有很貼切的產品，而是要能夠利用訪談得出真正的使用者需求是什麼，在進行開發。	已針對此問題對店家進行三度探訪，店家表示因桌遊店近期才逐漸盛行故系統引進也是零散無整合的，希望能有輔助整合類型的系統來提升店家效率，而消費者則表示桌遊業者在營業高峰時期並無法對客人提出全面服務，希望能透過多功能系統的方式減少等待店家服務的時間。
13. 感覺四邊(桌遊店 ERP，LineBot 推廣行銷，揪團，計時收費)都沒兜上邊，可以著重一個切入點，讓整個作品的完整度提升。	已針對此問題作修正，將重心置於系統整合功能上，並移除揪團、LineBOT 行銷整合。

14. 因為很像是每個人自己提出自己做的部分，然後再拼湊起來，這樣對於專題來說，很不好，團隊應該是一起構思，分配功能，然後在一起克服難題。	經過組員們每周的合作討論後漸漸團結一心，也感謝老師的支持與鼓勵。
15. 現有店家已有的 POS 如何差異化？	我們系統跟 POS 系統不相同，POS 是銷售實體商品紀錄庫存量的增減，我們系統主要注重於提升會員在桌遊店遊玩的體驗，並提供桌遊店家管理上的方便。
16. UI/UX 再優化。	已針對此問題作修正。
17. 建議該專題的名稱修改，以符合專題的內容。	已將專題名稱修改為”CatchU 桌助你”，取捉住你（心）之諧音以表達我們的桌遊輔助系統功能深入店家與消費者的心。
18. 揪團為主題，市場有更多相似的軟件。	經由討論已將揪團功能移除。
19. Github 上的程式很少，五月才開始上傳 Code，Commit 有衝突未解，都是文件的 commit，沒有整合。	已針對此問題作修正。
20. 功能太少，無法 DEMO，前後端都無法演示，功能規劃要更完整。	已針對此問題作修正。