

Loveat²

設計文件

專案名稱	Loveat ²
撰寫日期	2020/01/11
發展者	游傑如、李萱、黃莉庭、李珮慈、 林怡萱、王心妤、謝雅芳

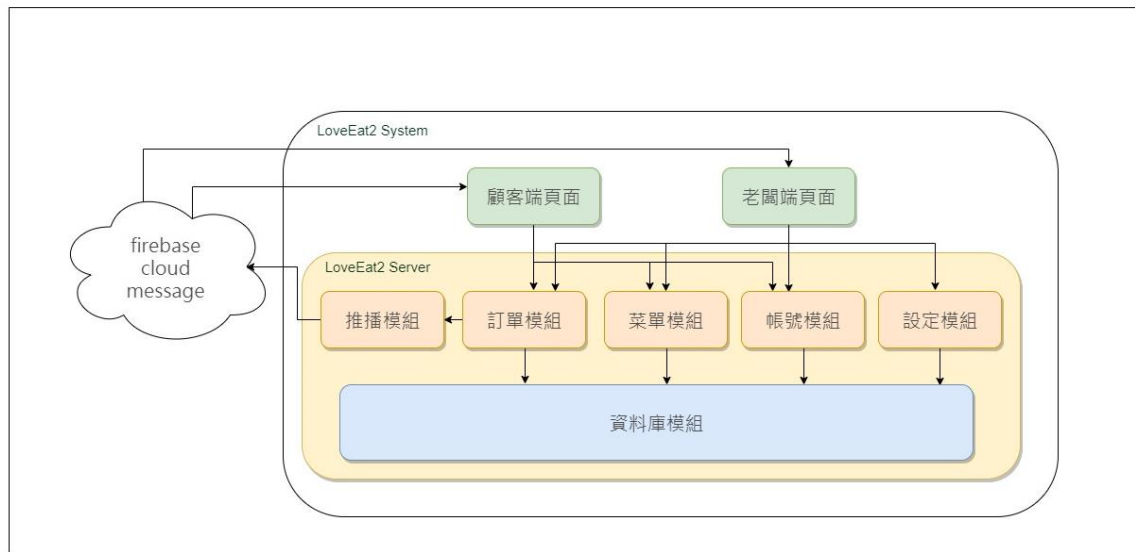
版次變更記錄

版次	變更項目	變更日期
0.1	初版	2019/11/20
0.2	根據老師建議修改並更新介面需求	2019/12/07
0.3	修改顧客訂單流程圖中推播部分	2019/12/08
1.0	更新class diagram、架構圖及介面需求	2019/12/11
2.0	添加黑名單、推播資訊、拒絕訂單 顧客推播功能之設計文件	2020/01/11

目錄

版次變更記錄	1
1. 系統模型與架構(SYSTEM MODEL/SYSTEM ARCHITECTURE).....	1
2. 介面需求與設計(INTERFACE REQUIREMENT AND DESIGN).....	2
3. 流程設計(PROCESS DESIGN)	23
4. 使用者畫面設計(USER INTERFACE DESIGN).....	39
5. 資料設計(DATA DESIGN)	52
6. 類別圖設計(CLASS DIAGRAM)	52
7. 實作技術(IMPLEMENTATION LANGUAGES AND PLATFORMS).....	71
8. 設計議題(DESIGN ISSUE)	71

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



本系統主要可分為設定模組、帳號模組、菜單模組、訂單模組、推播模組及資料庫模組。各模組功能說明如下：

- 設定模組：設定模組負責設定營業時間，後續可再擴充其他設定相關之功能。
- 帳號模組：提供帳號與使用者權限之 api，如 login、register、forgetPassword、updateUserInfo…等等。
- 菜單模組：提供關於菜單之 api(如針對 type、combo、item 新增刪除修改查詢)。
- 訂單模組：提供關於訂單之 api(如 updateState、addOrder、get、add…等等)。
- 推播模組：負責與第三方 API firebase cloud message 進行溝通。
- 資料庫模組：負責統一對資料庫 collection 進行的操作，同時也會處理各 collection 之間的相互關係。

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

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
獲取菜單資訊界面	菜單模組	老闆端/顧客端菜單頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP GET	無	<pre>[{ "type": "飲料", "category": "item", "content": [{ "name": "紅茶", "picture": "/img/29e01ced", "price": 15, "description": "紅茶", "top": false, }] }, { "type": "超值套餐", "category": "combo", "content": [{ "name": "漢堡套餐", "picture": "/img/29e01ced", "price": 15, "top": true, "content": [{ "name": "紅茶", "quantity": 1 }, { "name": "豬肉漢堡", "quantity": 1 }] }] }]</pre>

		<pre>], "description": "超划算" }] }]</pre>
URL		
https://loveat2.appspot.com/api/menu		
對應之介面需求(Interface Requirement Description)		
菜單模組回傳所有菜單資訊		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
登入介面	帳號模組	登入頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	<pre>{ "userName": "test", "password": "testPwd" }</pre>	Status code: ● 200: 登入成功 回傳格式: <pre>{ "state": "active" // 帳號 state }</pre> ● 401: 帳號或密碼錯誤
URL		
https://loveat2.appspot.com/api/user/login		
對應之介面需求(Interface Requirement Description)		
提供帳號驗證並登入之介面		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
註冊介面	帳號模組	註冊頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	{ "userName": "test", "password": "testPwd" "age": 1, "gender": "female", "phone": "0939784547", "email": "test@gmail.com" }	Status code: <ul style="list-style-type: none"> ● 200: 登入成功 ● 400: 資料格式錯誤 ● 409: 此帳號已經註冊過
URL		
https://loveat2.appspot.com/api/user/register		
對應之介面需求(Interface Requirement Description)		
提供帳號註冊之介面		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
忘記密碼申請介面	帳號模組	忘記密碼頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	{ "userName": "test", "email": "test@gmail.com" }	Status code: <ul style="list-style-type: none"> ● 200: 驗證成功 ● 401: 帳號或信箱錯誤
URL		
https://loveat2.appspot.com/api/user/		
對應之介面需求(Interface Requirement Description)		
輸入帳號、信箱，會發送信件到此信箱並給予重設密碼網址，請使用者前往修改。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
重設密碼介面	帳號模組	重設密碼頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	{ "password": "123456789" }	Status code: ● 200: 更新成功 ● 401: 該網頁已過期
URL		
https://loveat2.appspot.com/api/user/password/reset/{token}		
對應之介面需求(Interface Requirement Description)		
輸入新密碼，DB 更新密碼資訊。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
修改密碼介面	帳號模組	編輯個資頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	{ "oldPassword": "123456789", "newPassword": "12345689" }	Status code: ● 200: 修改成功 ● 401: 舊密碼錯誤
URL		
https://loveat2.appspot.com/api/user/password/update		
對應之介面需求(Interface Requirement Description)		
使用者修改密碼，由後台驗證身分是否正確，並對 DB 進行更新。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
更新個資介面	帳號模組	編輯個資頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	{ "age": "10", "gender": "female", "email": "test@gmail.com", "phone": "0918781125", "picture": file }	Status code: ● 200: 更新成功 ● 400: 資料格式錯誤
URL		
https://loveat2.appspot.com/api/user/update		
對應之介面需求(Interface Requirement Description)		
使用者更新個人資料，由後台對 DB 做更新。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
更新推播 token 介面	帳號模組	所有頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	{ "token": "cDps- BVTkgIFjohPztK019:APA91bElmC9yZXsYkmZ-- gDyxdd7igIveAu7lIGSvspaMjfyq-kSsYX5M1lme15Nay" }	Status code: ● 200: 更新成功 ● 401: 未授權
URL		
https://loveat2.appspot.com/api/user/token		
對應之介面需求(Interface Requirement Description)		
使用者接受網頁推播後，須定時更新 token，才可以持續使用推播功能。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
新訂單推播介面	Firebase	老闆端顧客點餐頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
Firebase API	無	{ "id": "5dd0b8043257515818a8c101", "takenAt": "2019-11-17T16:00:00.000Z", "notes": "1 杯紅茶去冰", "content": [{ "name": "紅茶", "quantity": 2 }, { "name": "綠茶", "quantity": 1 }] }
對應之介面需求(Interface Requirement Description)		
老闆接收推播資訊，更新顧客點餐頁面。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
訂單狀態推播介面	Firebase	顧客端訂單狀態頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
Firebase API	無	{ "id": "5dd0b8b93257515818a8c103", "state": "doing" }
對應之介面需求(Interface Requirement Description)		
利用推播，即時更新訂單狀態。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
設定點餐時間介面	設定模組	老闆端設定可點餐 時間頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	<pre>[{ "day": "mon", "startTime": "2019-10-31T00:00", "endTime": "2019-10-31T10:00" }]</pre> <p>*注：day 可能的值為 mon、tue、web、thu、fri、sat、sun</p>	Status code: <ul style="list-style-type: none"> ● 200: 更新成功 ● 400: 資料格式錯誤 ● 403: 權限錯誤
URL		
https://loveat2.appspot.com/api/setting/business-time		
對應之介面需求(Interface Requirement Description)		
更新 DB 裡點餐時間之資訊。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
新增訂單介面	訂單模組	顧客端購物車頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	<pre>{ "takenAt": "2019-10-31T00:00", "userName": "loveEatEat", "notes": "不要加番茄", "total": "500", "content": [{ "id": "5dd0b8b93257515818a8c103", "category": "item", "quantity": 1 }, { "id": "5dd0b8b93257515818a8c103", "category": "combo", "quantity": 1 }] }</pre>	Status code: <ul style="list-style-type: none">● 200: 成功下訂● 401: 使用者尚未登入● 403: 使用者帳號狀態錯誤(被凍結)● 422: 訂單格式錯誤或取餐時間不營業
URL		
https://loveat2.appspot.com/api/order/new		
對應之介面需求(Interface Requirement Description)		
顧客送出訂單後存進資料庫		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
更新訂單狀態介面	訂單模組	老闆端顧客點餐頁面, 老闆端餐點製作清單頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	{ "id": "5dd0b8b93257515818a8c103", "state": "doing", "content": "you can' t eat." }	Status code: ● 200: 更新成功 ● 403: 權限錯誤 ● 404: 此筆訂單不存在
URL		
https://loveat2.appspot.com/api/order/update		
對應之介面需求(Interface Requirement Description)		
<ul style="list-style-type: none"> ● 老闆接受(doin)或拒絕(cancel)訂單。 ● 老闆更新訂單為已完成(finish)或是已取餐(end)狀態。 		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
歷史訂單查詢介面	訂單模組	老闆端歷史點菜單主畫面頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP GET	URI: https://loveat2.appspot.com/api/order?startTime=2019-10-31T00:00&endTime=2019-10-31T30:00 *注：可以改變 URI 中的 startTime、endTime 來限制搜尋範圍	Status code: ● 200: 更新成功 ● 403: 權限錯誤 回傳資料格式: <pre>[{ "orderId": "1", "createdAt": "2019-11-14T12: 00", "note": "冰紅茶去冰", "content": [{ "name": "巧克力吐司", "quantity": "3" }, { "name": "冰紅茶", "quantity": "2" }] }]</pre>
對應之介面需求(Interface Requirement Description)		
老闆取得區段時間內之所有歷史訂單。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
歷史訂單分析介面	訂單模組	老闆端歷史點菜單銷售情況頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP GET	URI: https://loveat2.appspot.com/api/order/analysis?startTime=2019-10-31T00:00&endTime=2019-10-31T30:00 *注：可以改變 URI 中的 startTime、endTime 來限制搜尋範圍	Status code: <ul style="list-style-type: none"> ● 200: 更新成功 ● 403: 權限錯誤 回傳資料格式: <pre>{ "interval": ["0-9", "10-19", "20-29", "30-39", "40-49", "50-59", "60+"], "itemAnalysis": { "乳酪蛋餅": { "female": [10, 10, 10, 10, 10, 10, 10], "male": [10, 10, 10, 10, 10, 10, 10], "sum": [20, 20, 20, 20, 20, 20, 20], "femaleTotal": 70, "maleTotal": 70, "total": 140 }, "紅茶": { "female": [10, 10, 10, 10, 10, 10, 10], "male": [10, 10, 10, 10, 10, 10, 10], "sum": [20, 20, 20, 20, 20, 20, 20] } } }</pre>

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
取得製作中訂單 介面	訂單模組	老闆端餐點製作清單頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP GET	無	Status code: <ul style="list-style-type: none"> ● 200: 獲取成功 ● 403: 權限錯誤 回傳資料格式: <pre>[{ "_id": "5de39e55ca56ec14844700a8", "content": [{ "name": "西西里雞腿堡", "quantity": 1 }], "notes": "漢堡加蛋", "orderId": 1, "phone": "0939783483", "state": "doing", "takenAt": "2019/11/30 20:08", "user_id": "5dd752c0ccce596d624d9a9d" }]</pre>
URL		
https://loveat2.appspot.com/api/order/todo		
對應之介面需求(Interface Requirement Description)		
老闆取得餐點製作清單的相關資訊。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
新增種類名稱介面	菜單模組	老闆端編輯菜單 的編輯種類頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	{ "type": "超值套餐", "category": "combo" }	Status code: ● 200: 更新成功 ● 403: 權限錯誤 ● 409: 該種類已存在
URL		
https://loveat2.appspot.com/api/menu/type/new		
對應之介面需求(Interface Requirement Description)		
新增種類，更新 DB。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
修改種類名稱介面	菜單模組	老闆端編輯菜單 的編輯種類頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	{ "id": "5dd0b8b93257515818a8c103", "type": "漢堡套餐" }	Status code: ● 200: 修改成功 ● 403: 權限錯誤 ● 409: 該種類名稱已存在
URL		
https://loveat2.appspot.com/api/menu/type/update		
對應之介面需求(Interface Requirement Description)		
修改種類，更新 DB。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
刪除種類介面	菜單模組	老闆端編輯菜單 的編輯種類頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	{ " id": "5dd0b8b93257515818a8c103" }	Status code: ● 200: 刪除成功 ● 403: 權限錯誤
URL		
https://loveat2.appspot.com/api/menu/type/delete		
對應之介面需求(Interface Requirement Description)		
刪除種類，更新 DB。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
取得單品資訊介面	菜單模組	顧客端購物車頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	["5dd67f098f0f6afb3ebc1b68"] *傳送 ID 資訊	[{ "_id": "5dd67f098f0f6afb3ebc1b68", "price": 20, "name": "紅茶" }]
URL		
https://loveat2.appspot.com/api/menu/item		
對應之介面需求(Interface Requirement Description)		
利用單品 ID 取得對應之單品資訊。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
取得套餐資訊介面	菜單模組	顧客端購物車頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	["5dd695b9fb8001b28236efb7"] *傳送 ID 資訊	Status code: ● 200: 獲取成功 回傳資料格式: [{ "_id": "5dd695b9fb8001b28236efb7", "price": 60, "name": "A 餐" }]
URL		
https://loveat2.appspot.com/api/menu/combo		
對應之介面需求(Interface Requirement Description)		
利用套餐 ID 取得對應之套餐資訊。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
新增單品介面	菜單模組	老闆端編輯菜單的單品新增頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	{ "type": "5db31a99e8603473845fece5", "name": "豬肉漢堡", "picture": file(binary string), "price": 30, "description": "非組合肉" }	Status code: ● 200: 新增成功 ● 403: 權限錯誤 ● 409: 該單品已存在
URL		
https://loveat2.appspot.com/api/menu/item/new		
對應之介面需求(Interface Requirement Description)		
新增單品，更新 DB。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
修改單品介面	菜單模組	老闆端編輯菜單的單品修改頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	{ "id": "5dd0c3163257515818a8c107", "type": "5db31a99e8603473845fece5", "name": "豬肉漢堡", "picture": file(binary string), "price": 30, "description": "非組合肉" }	Status code: ● 200: 新增成功 ● 403: 權限錯誤 ● 409: 該單品名稱已存在
URL		
https://loveat2.appspot.com/api/menu/item/update		
對應之介面需求(Interface Requirement Description)		
可修改單品，更新 DB。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
刪除單品介面	菜單模組	老闆端編輯菜單頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	{ " id": "5dd0b8b93257515818a8c103" }	Status code: ● 200: 刪除成功 ● 403: 權限錯誤
URL		
https://loveat2.appspot.com/api/menu/item/delete		
對應之介面需求(Interface Requirement Description)		
刪除單品，更新 DB。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
新增套餐介面	菜單模組	老闆端編輯菜單 的套餐新增頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	<pre>{ "type": "5dd0c32a3257515818a8c109", "name": "乳酪蛋餅超值套餐", "picture": file(binary string), "price": 55, "description": "超划算的，快來買!", "content": [{ "id": "5dd0c3c33257515818a8c10b", "quantity": 1 }, { "id": "5dd0c3d63257515818a8c10c", "quantity": 1 }] }</pre>	Status code: <ul style="list-style-type: none">● 200: 新增成功● 403: 權限錯誤● 409: 該套餐已存在
URL		
https://loveat2.appspot.com/api/menu/combo/new		
對應之介面需求(Interface Requirement Description)		
可新增套餐，更新 DB。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
修改套餐介面	菜單模組	老闆端編輯菜單 的套餐修改頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	<pre>{ "id": "5dd0c3163257515818a8c107", "type": "5db31a99e8603473845fece5", "name": "乳酪蛋餅超值套餐", "picture": file(binary string), "price": 55, "description": "超划算的，快來買!", "content": [{ "id": "5dd0c3c33257515818a8c10b", "quantity": 1 }, { "id": "5dd0c3d63257515818a8c10c", "quantity": 1 }] }</pre>	Status code: <ul style="list-style-type: none">● 200: 新增成功● 403: 權限錯誤● 409: 該套餐名稱已存在
URL		
https://loveat2.appspot.com/api/menu/combo/update		
對應之介面需求(Interface Requirement Description)		
可修改套餐，更新 DB。		

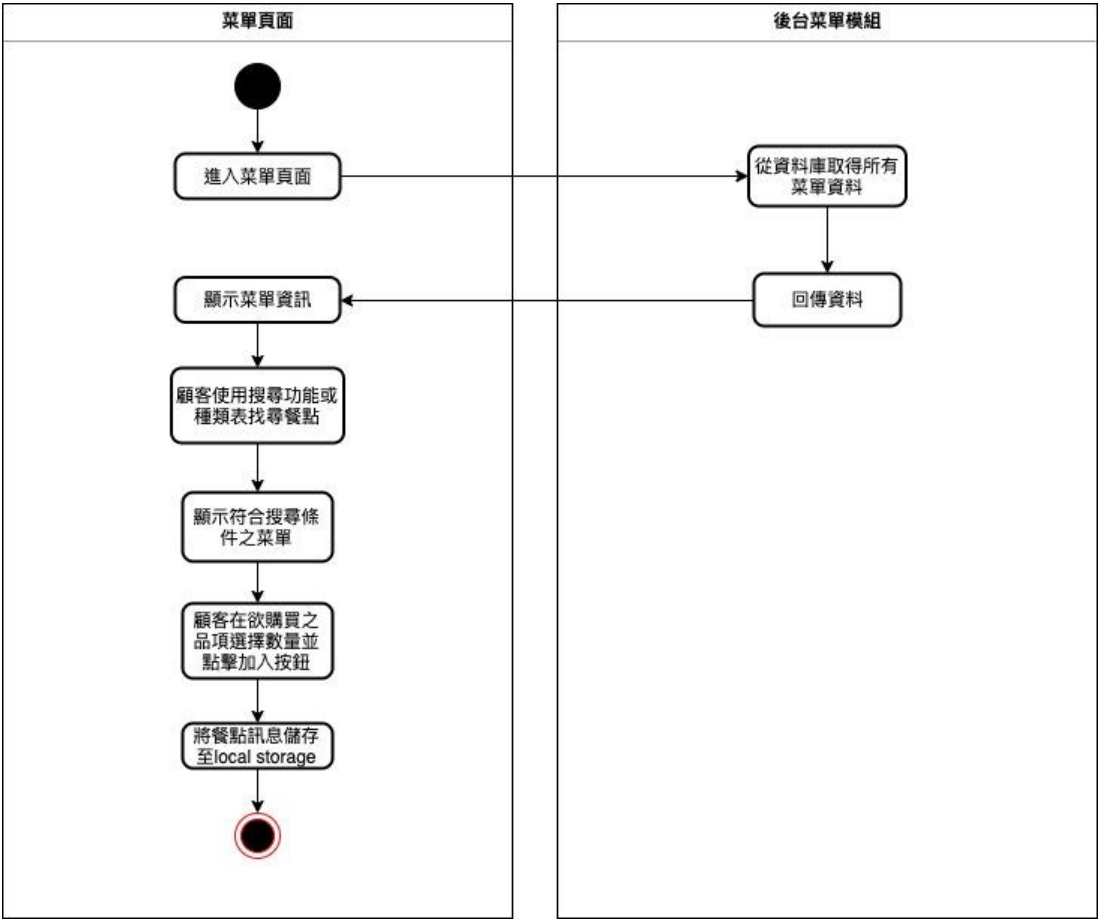
介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
刪除套餐介面	菜單模組	老闆端編輯菜單頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	{ "id": "5dd0b8b93257515818a8c103" }	Status code: <ul style="list-style-type: none">● 200: 刪除成功● 403: 權限錯誤
URL		
https://loveat2.appspot.com/api/menu/combo/delete		
對應之介面需求(Interface Requirement Description)		
刪除套餐，更新 DB。		

介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
資訊推播介面	設定模組	老闆端資訊推播頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	{ "title": "Hello", "content": "Hello" }	Status code: <ul style="list-style-type: none">● 200: 推播成功● 403: 權限錯誤
URL		
https://loveat2.appspot.com/api/setting/news		
對應之介面需求(Interface Requirement Description)		
送出推播內容給顧客。		

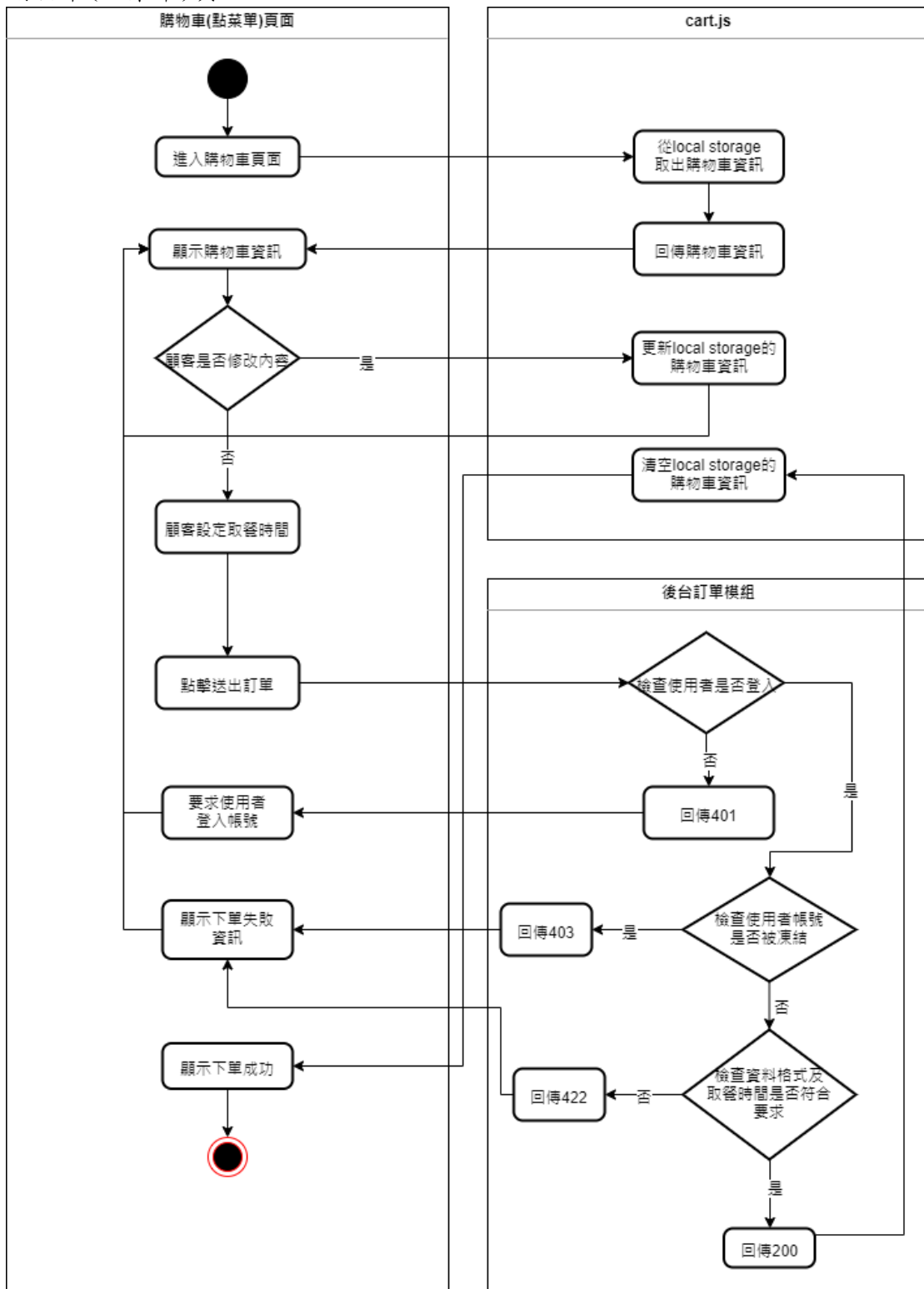
介面名稱 (Interface Name)	介面提供者 (Interface Provider)	介面使用者 (Interface Consumer)
更新帳號狀態介面	帳號模組	老闆端帳號管理頁面
連結方式 (Connection Type)	輸入資料 (Input Data)	輸出資料 (Output Data)
HTTP POST	{ "id": "test1", "state": "active" }	Status code: <ul style="list-style-type: none">● 200: 登入成功● 400: 格式錯誤● 403: 權限錯誤● 404: 使用者不存在
URL		
https://loveat2.appspot.com/api/user/update/state		
對應之介面需求(Interface Requirement Description)		
改變顧客的帳號狀態		

3. 流程設計(Process Design)

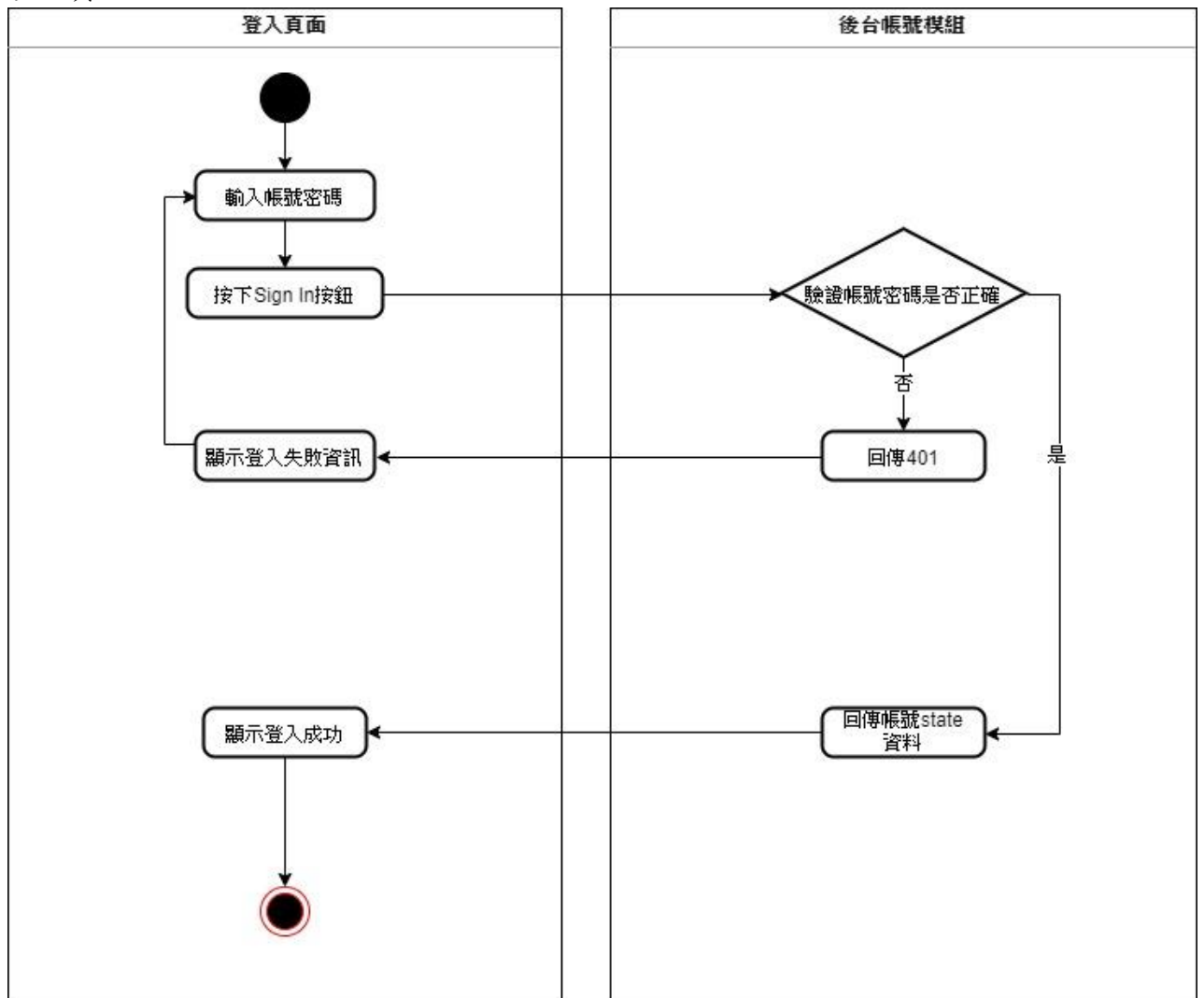
- 菜單頁面：



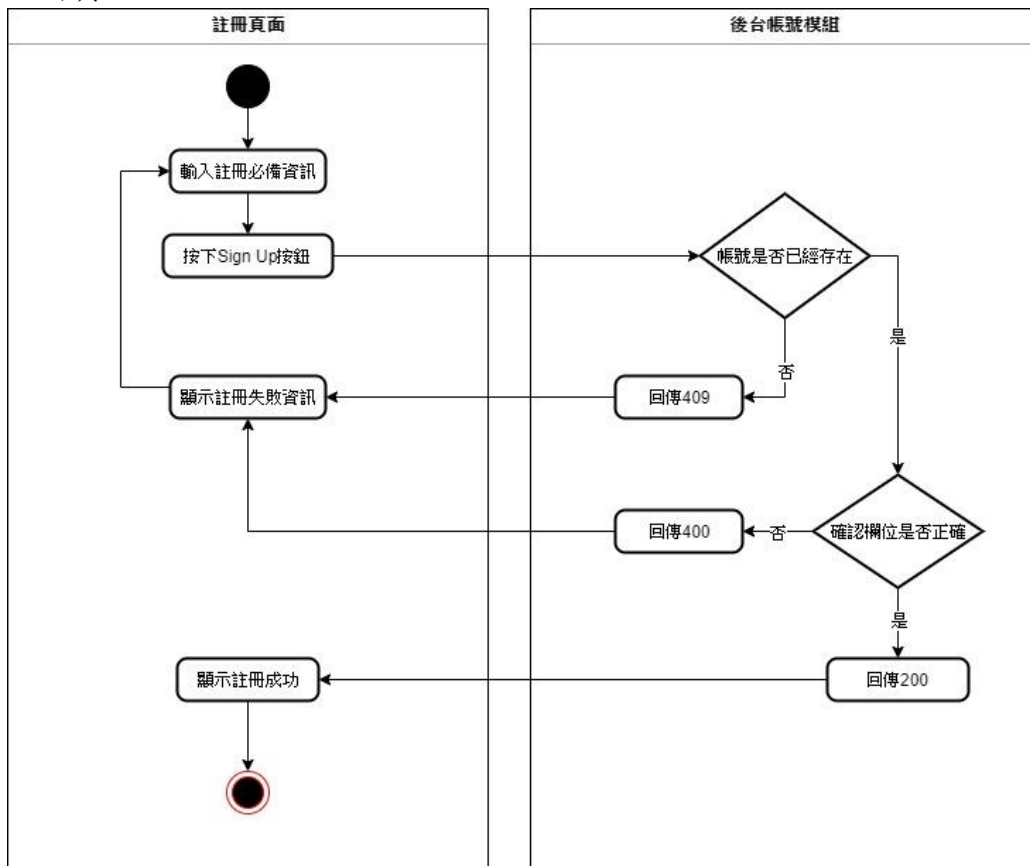
● 購物車(點菜單)頁面:



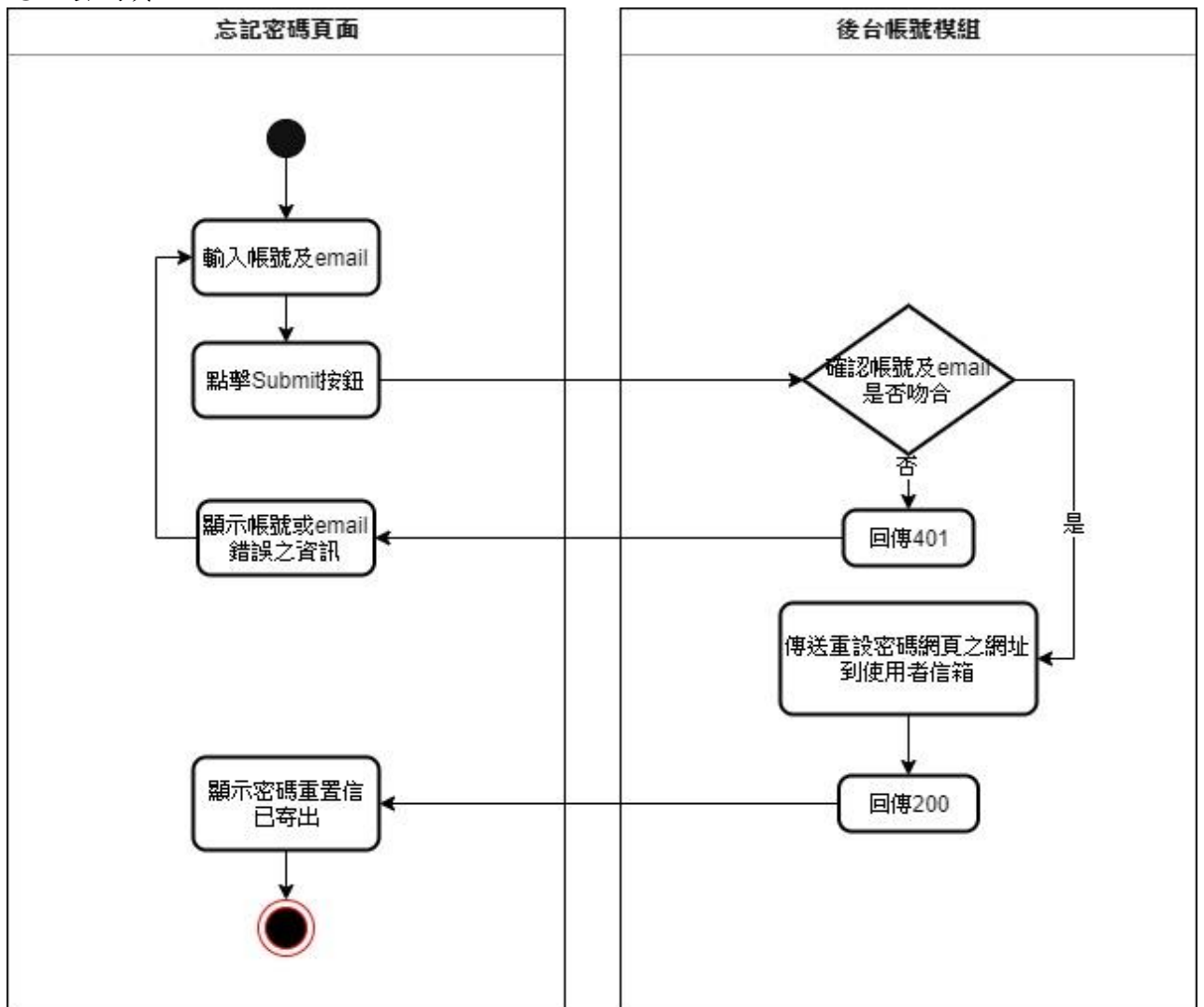
- 登入頁面



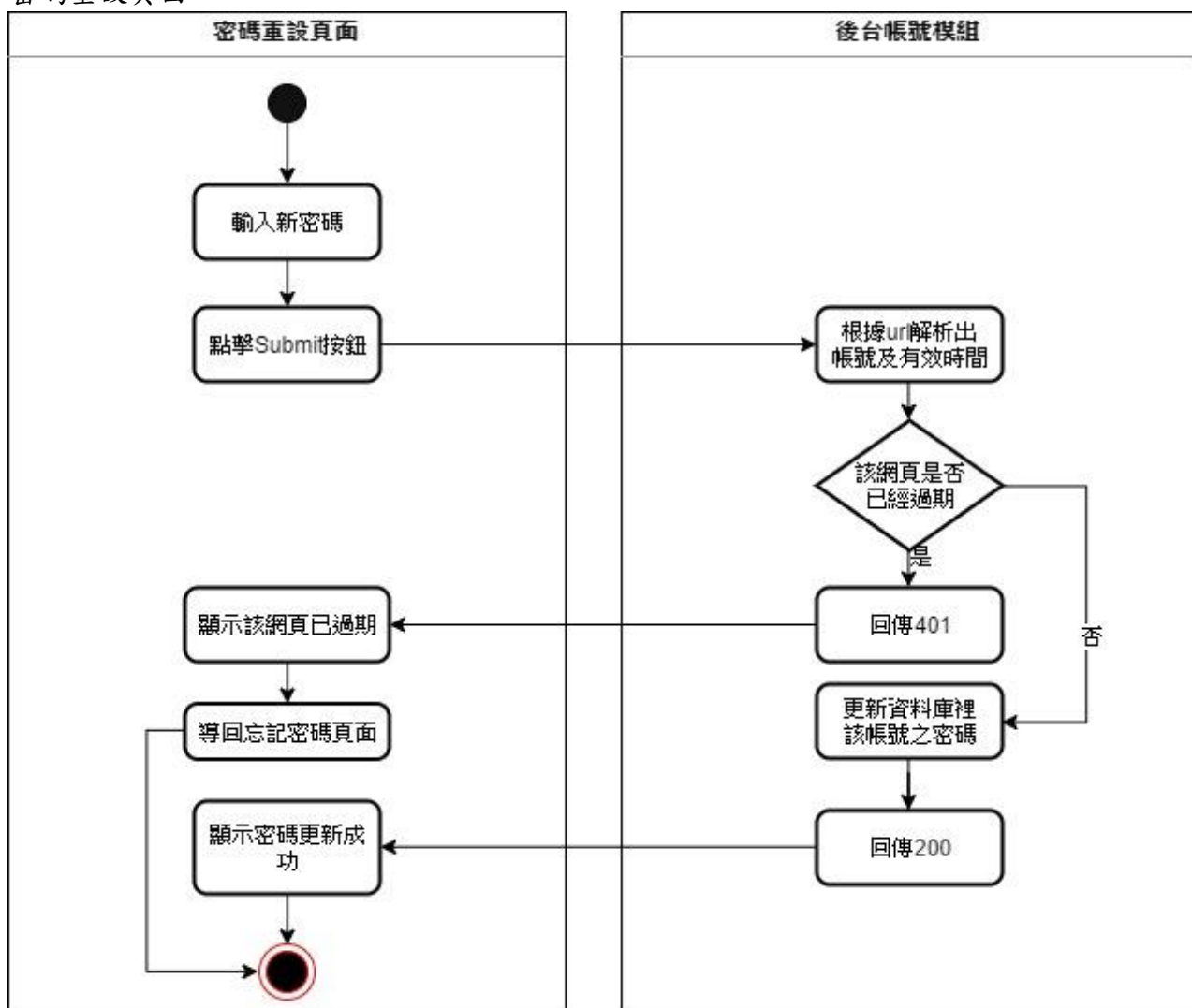
● 註冊頁面



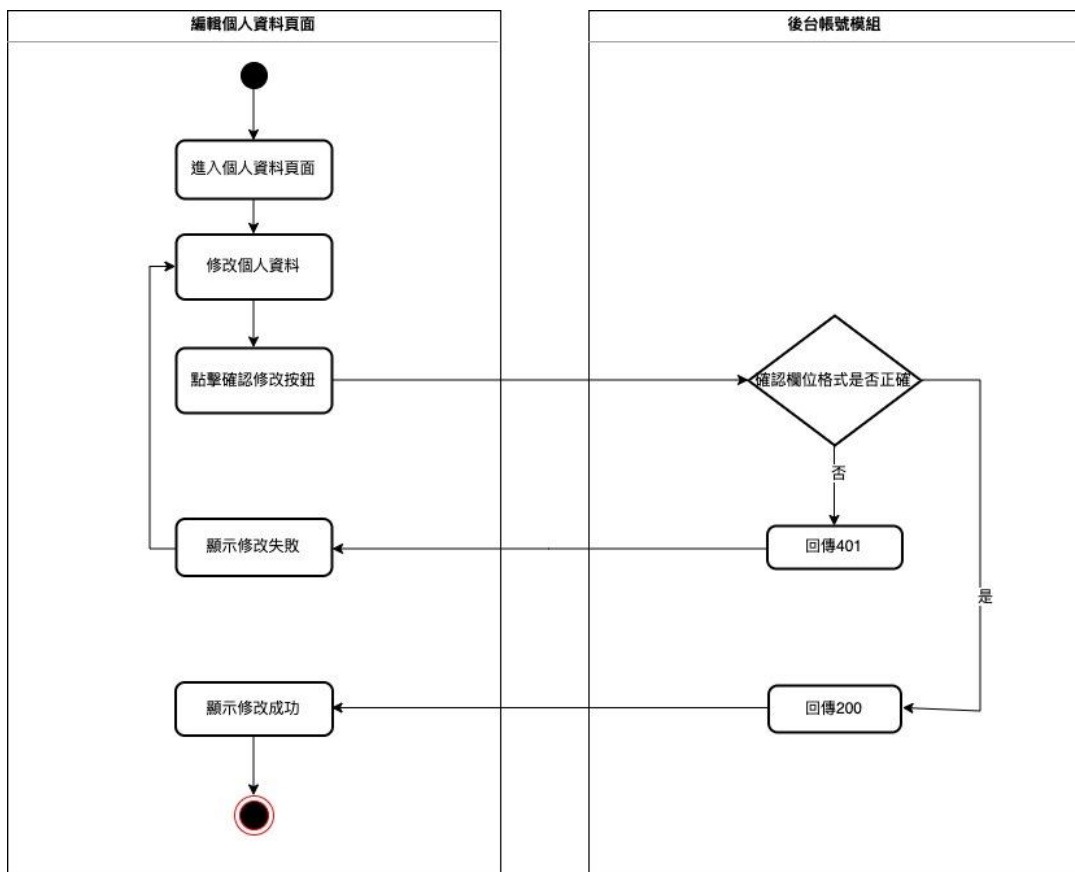
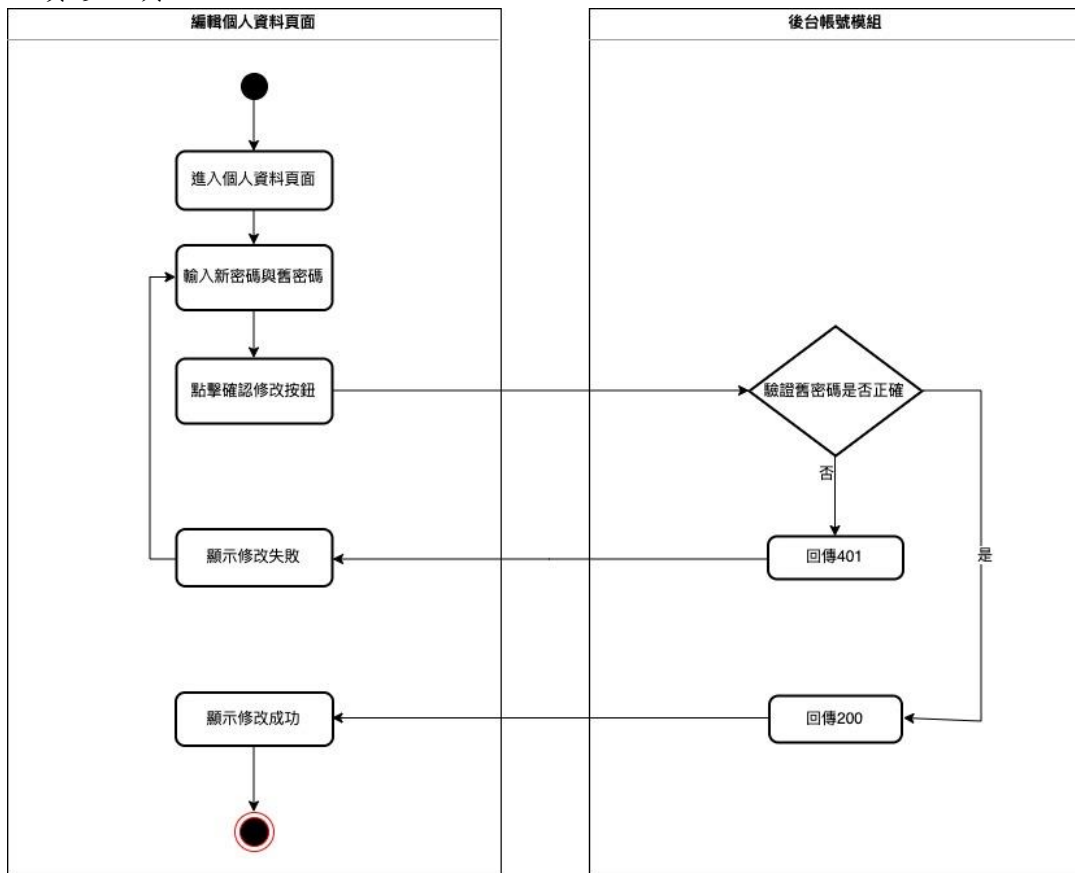
● 忘記密碼頁面



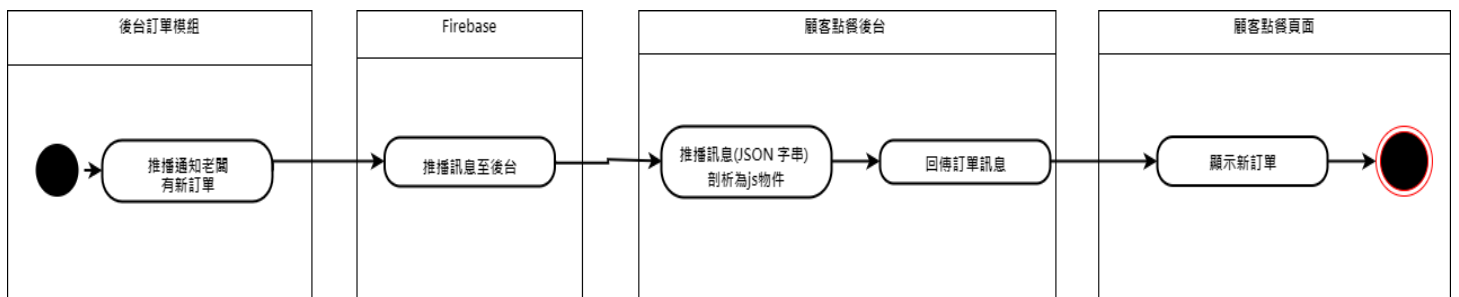
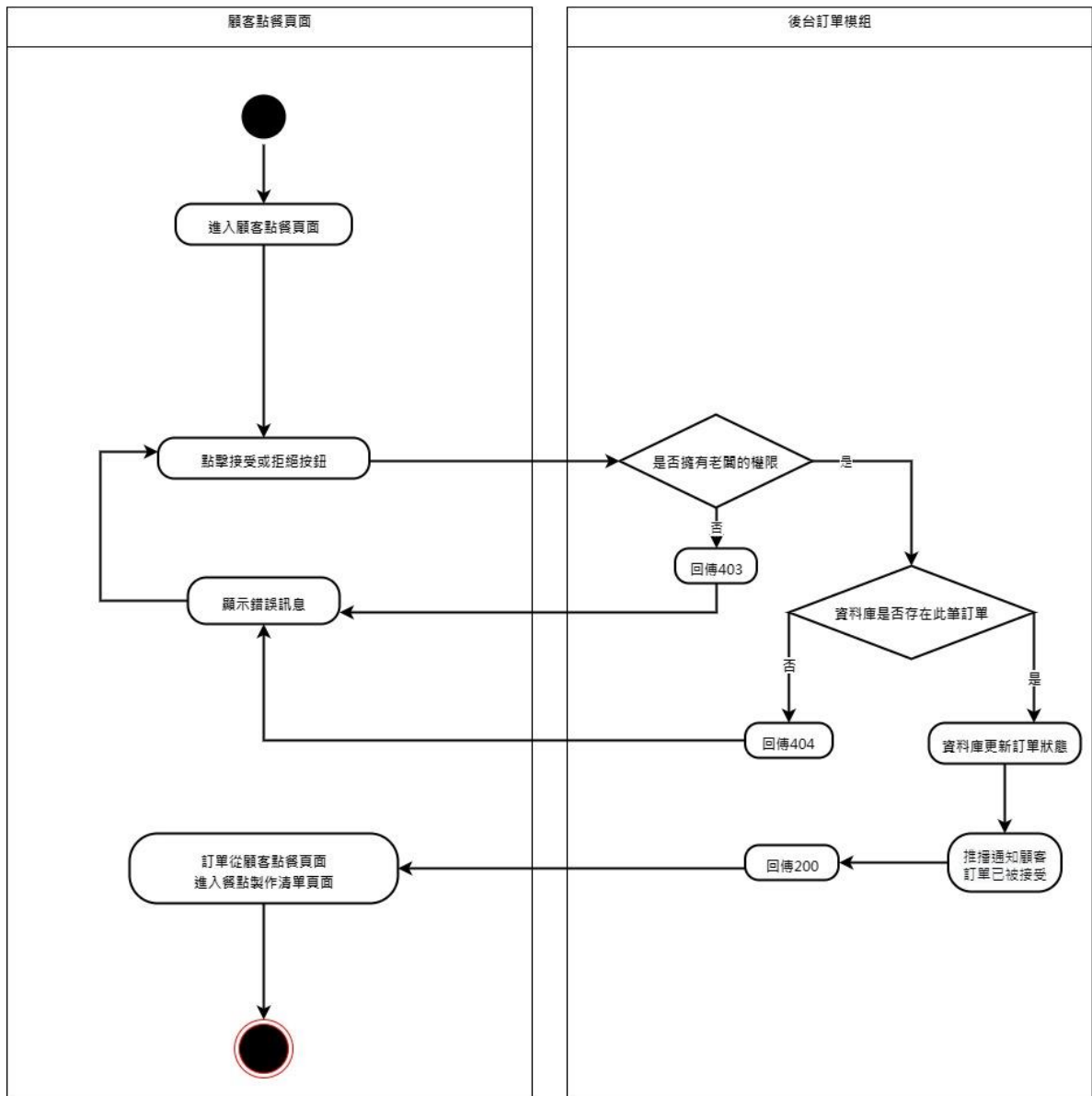
- 密碼重設頁面



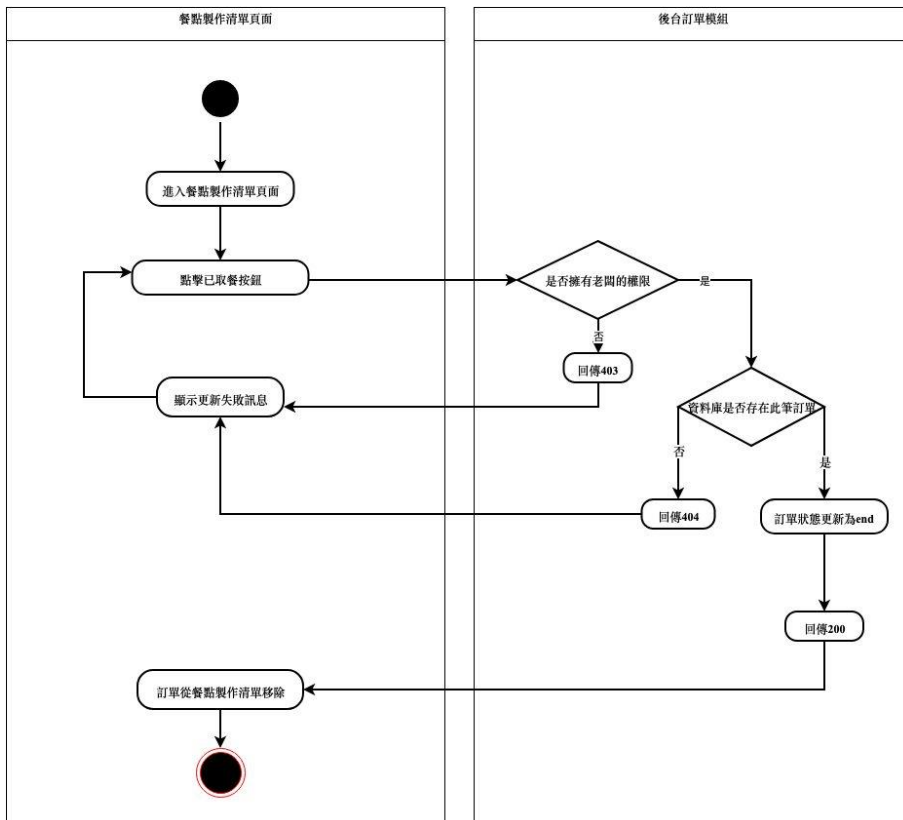
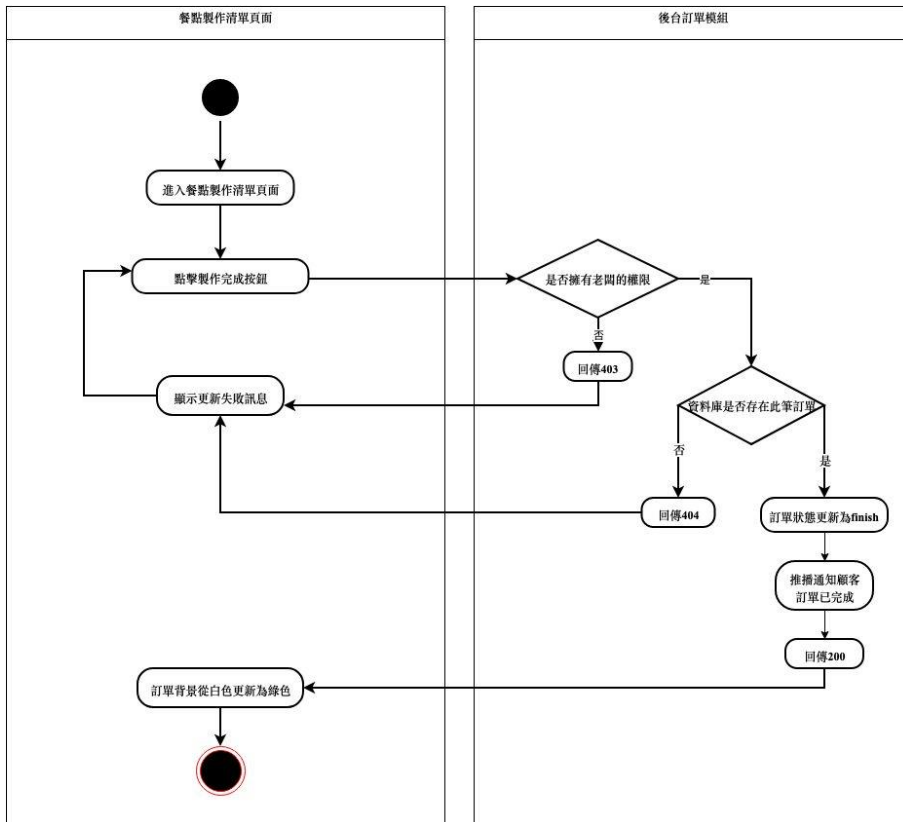
● 個資修改頁面



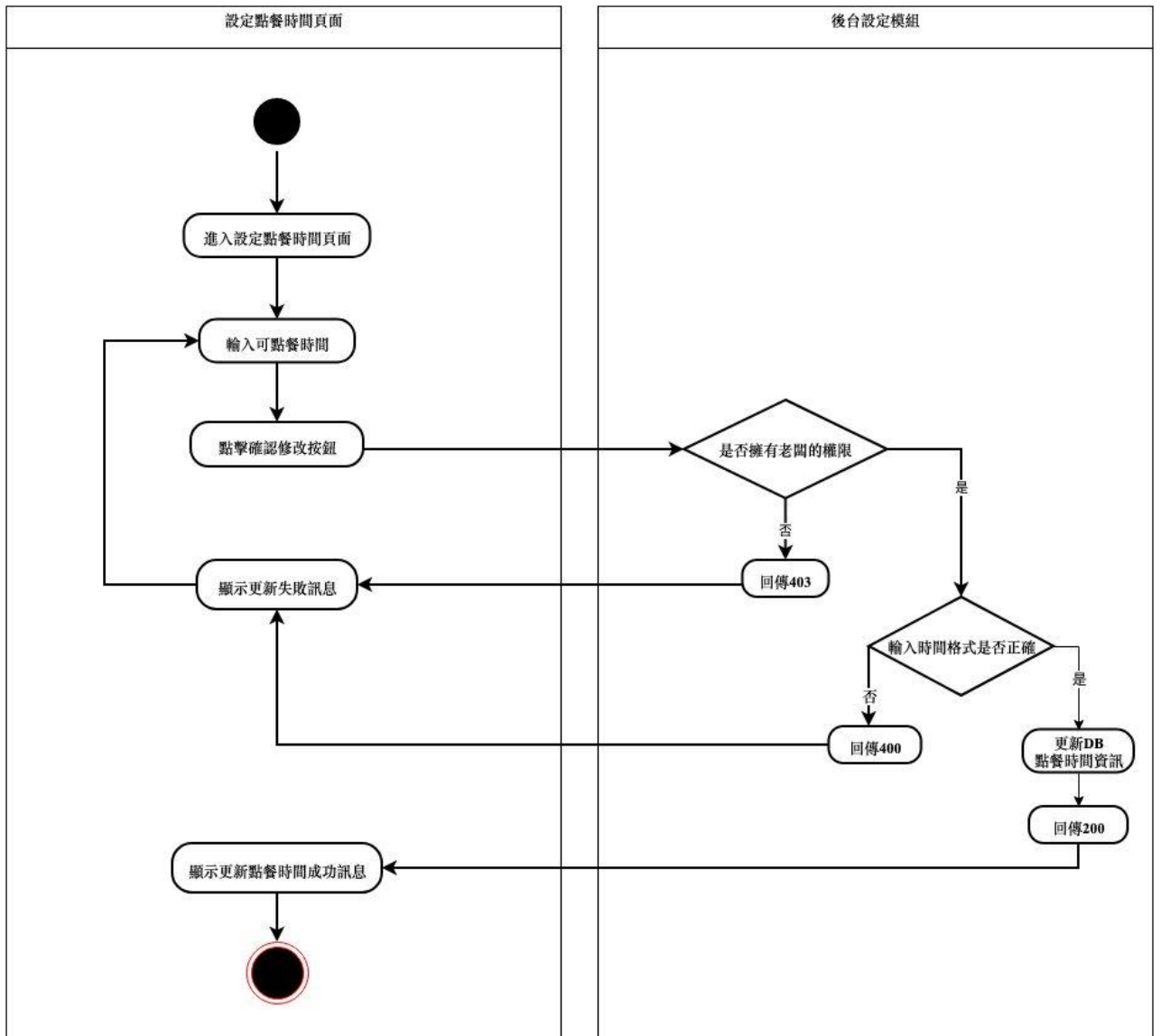
● 顧客點餐頁面



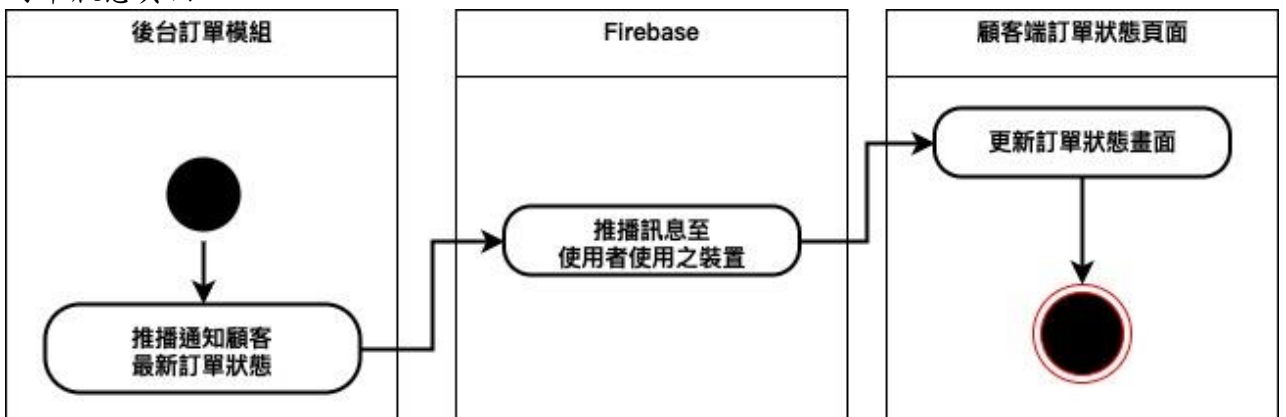
● 餐點製作清單頁面



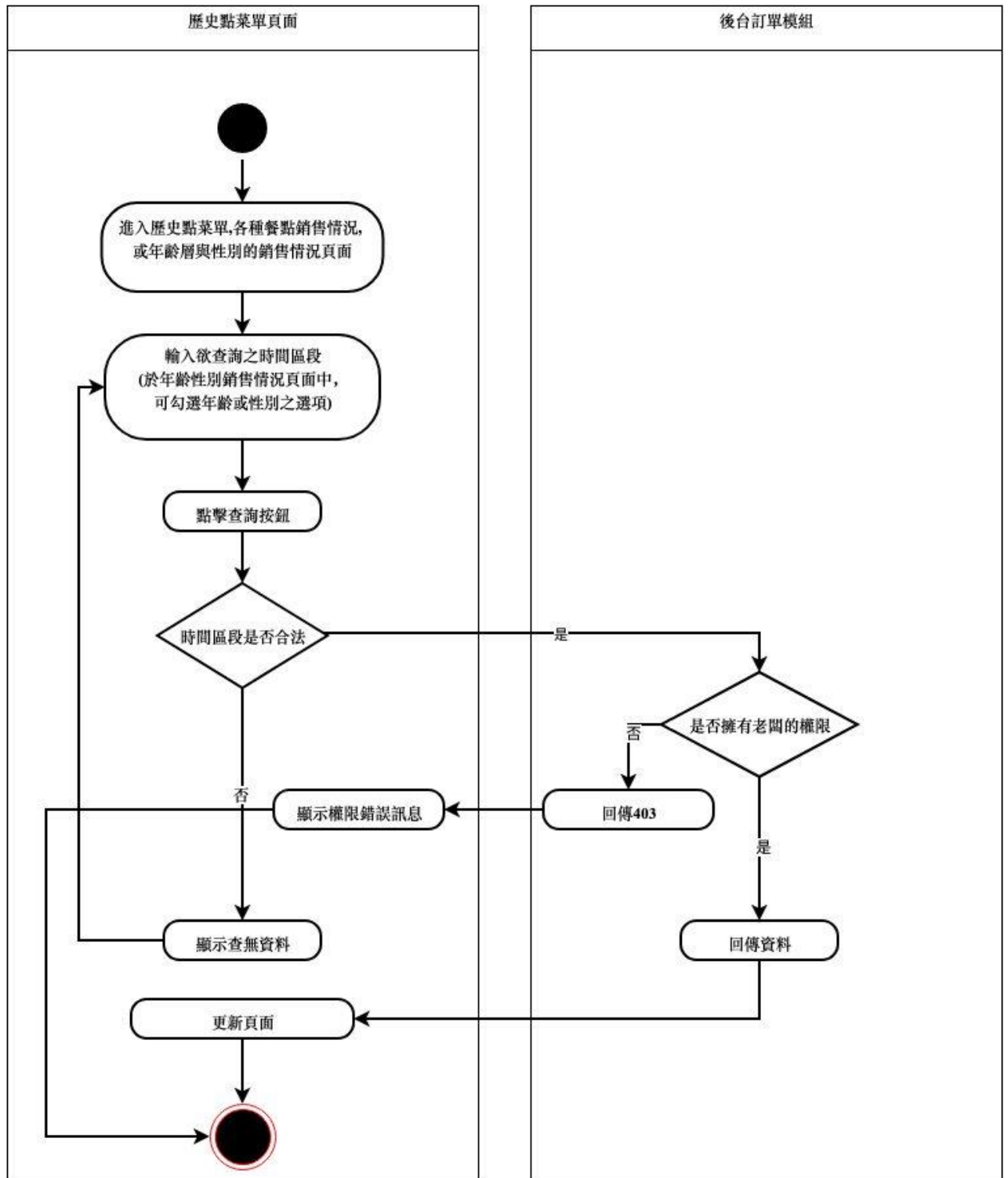
● 編輯點餐時間頁面



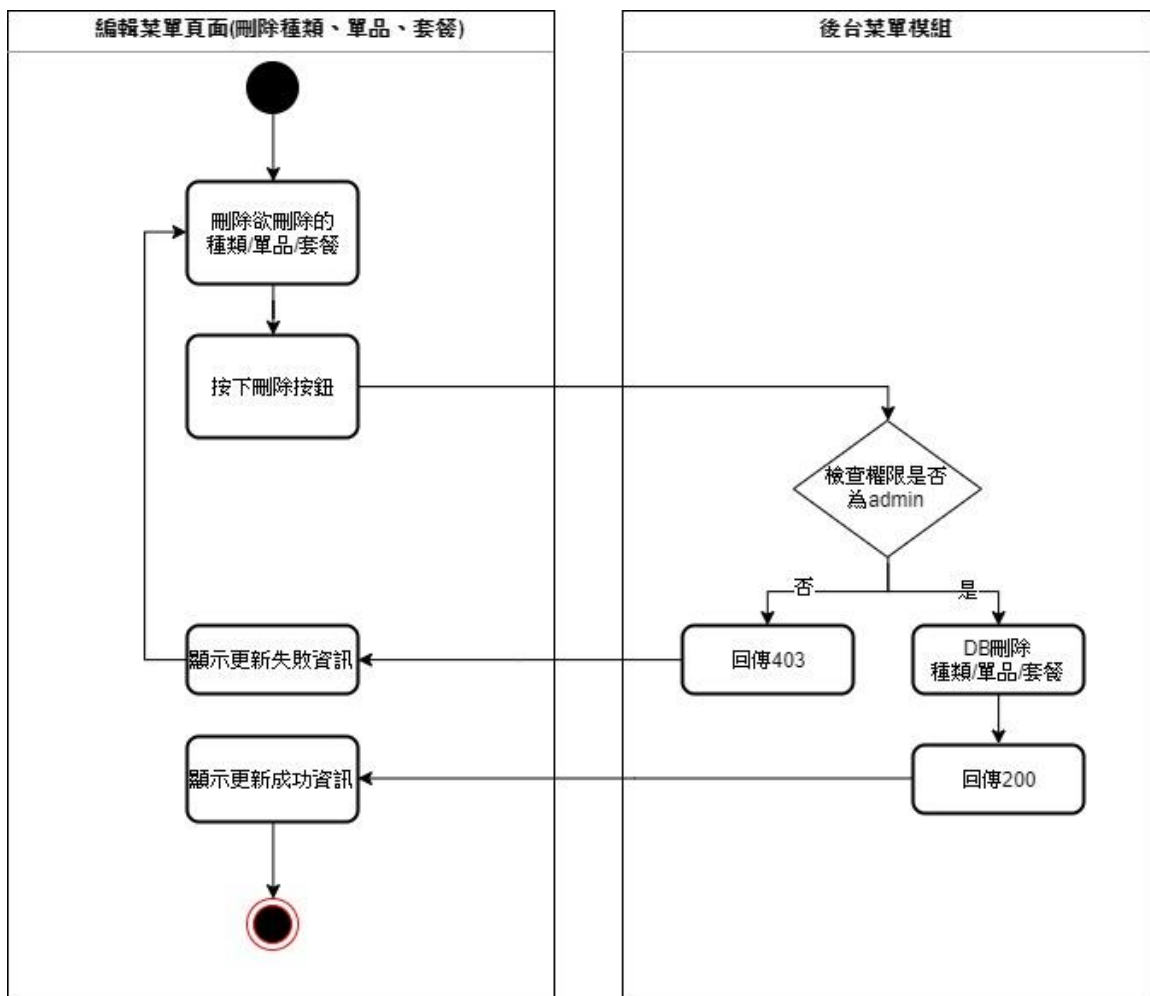
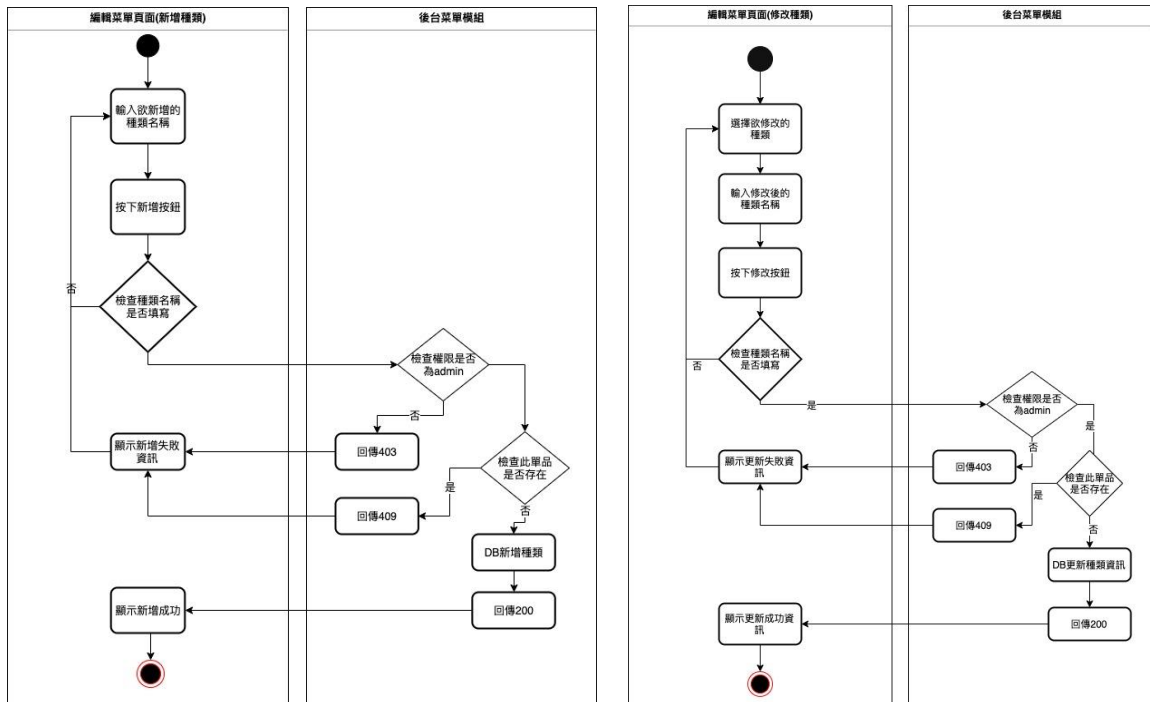
● 訂單狀態頁面



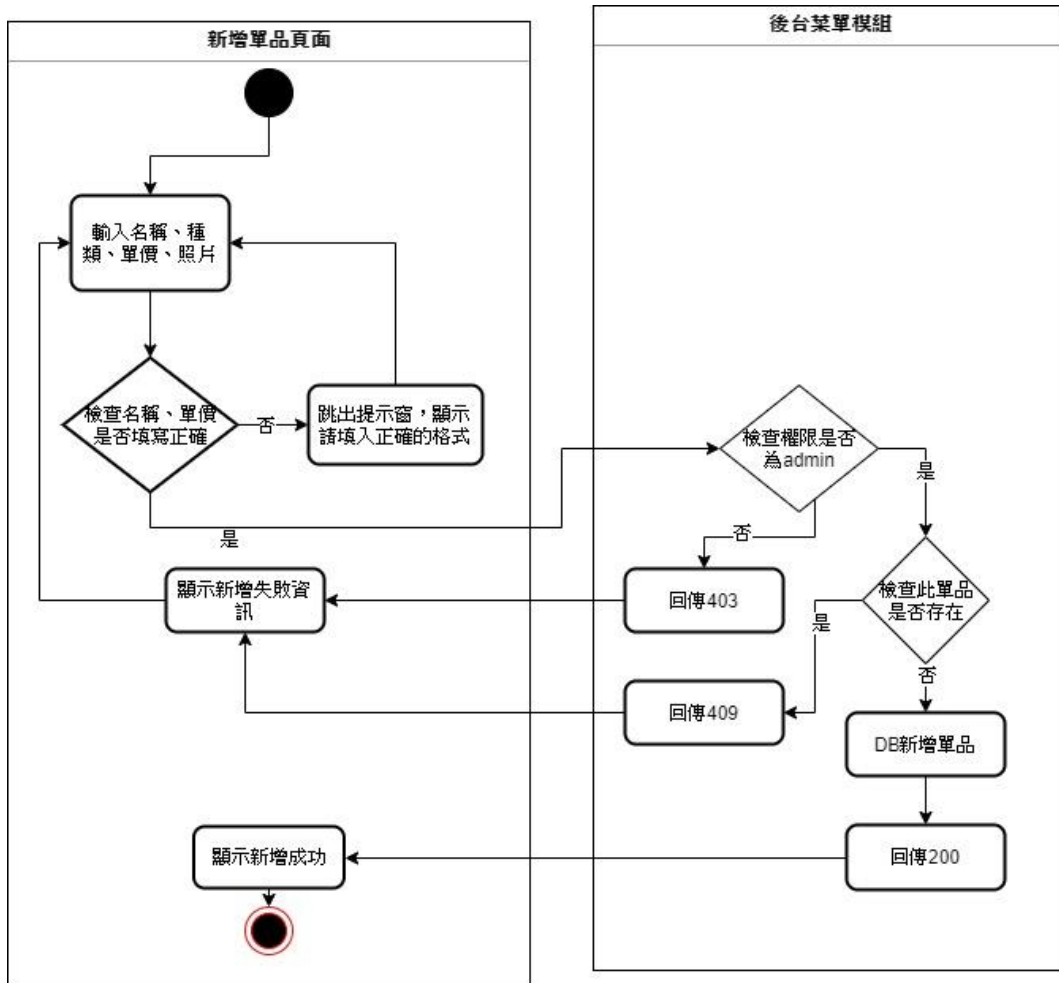
● 歷史訂單頁面



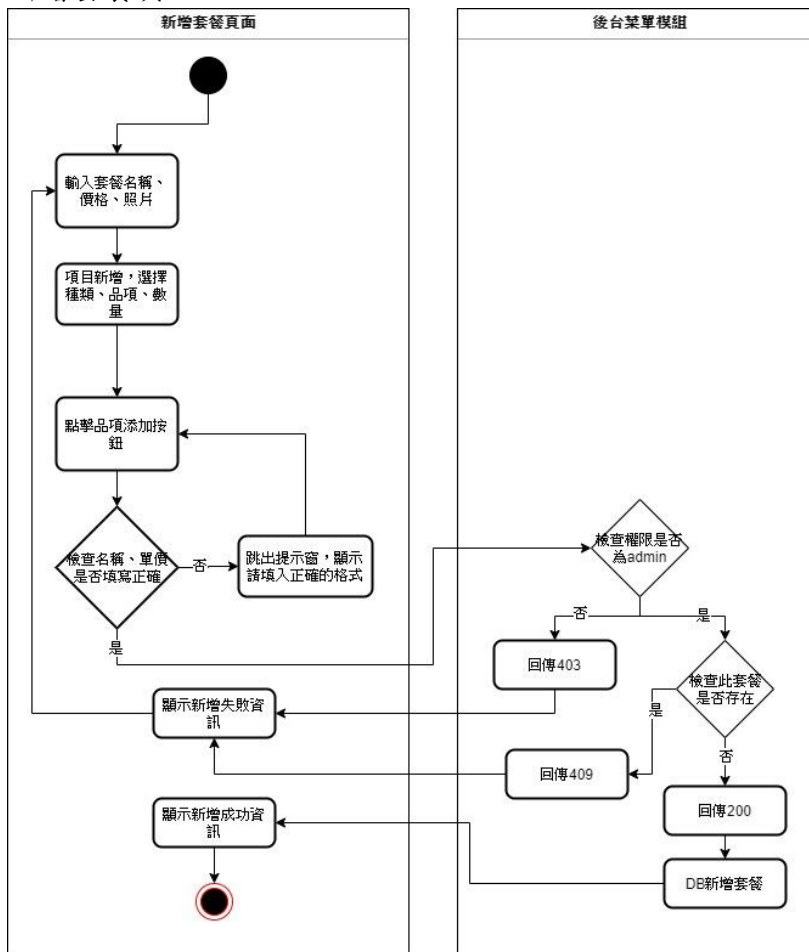
● 編輯菜單頁面



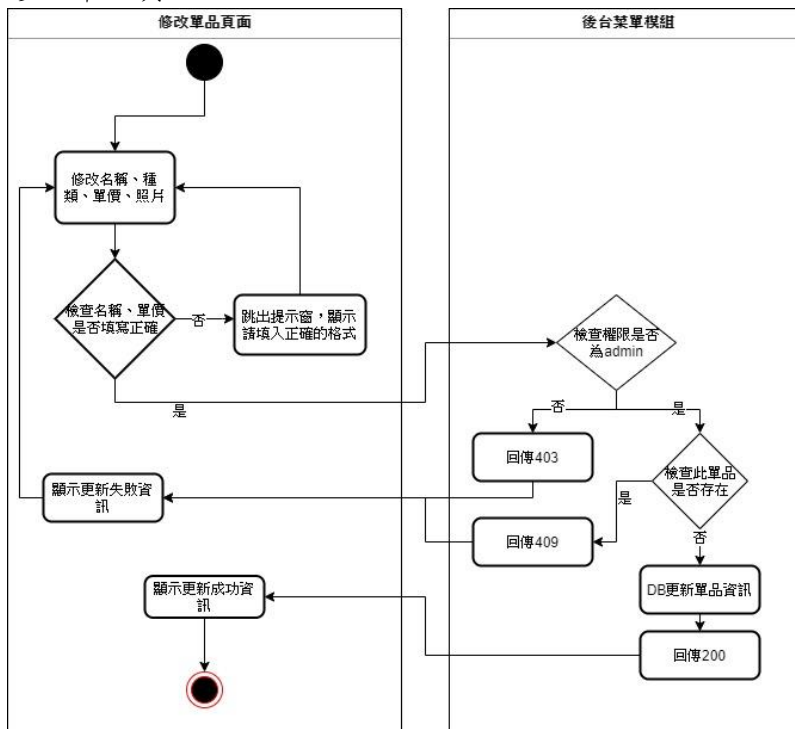
● 新增單品頁面



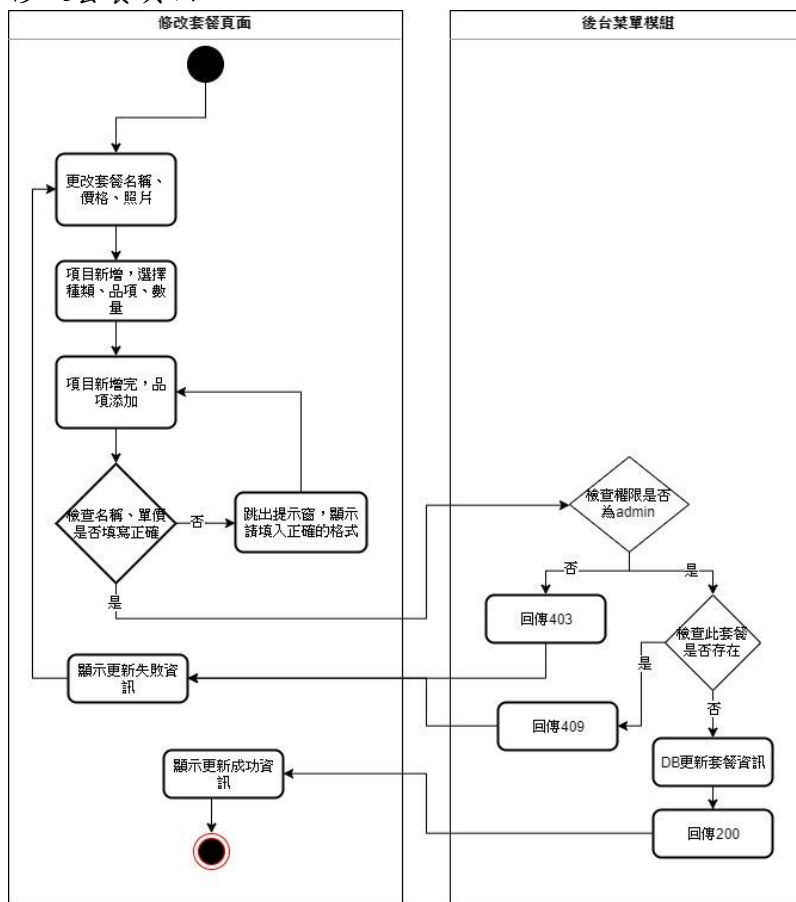
● 新增套餐頁面



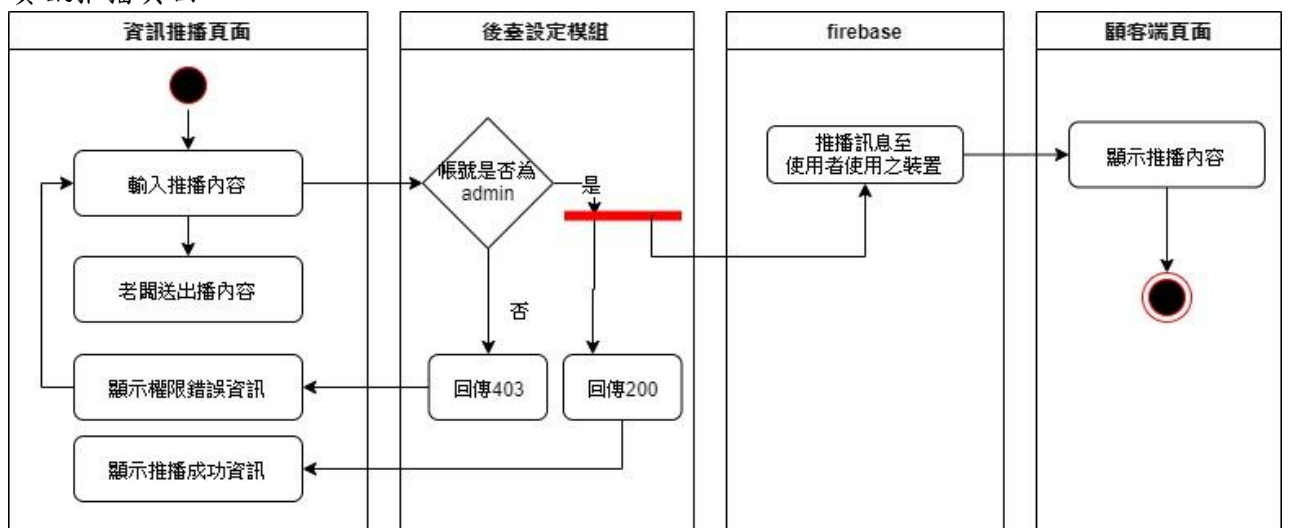
● 修改單品頁面



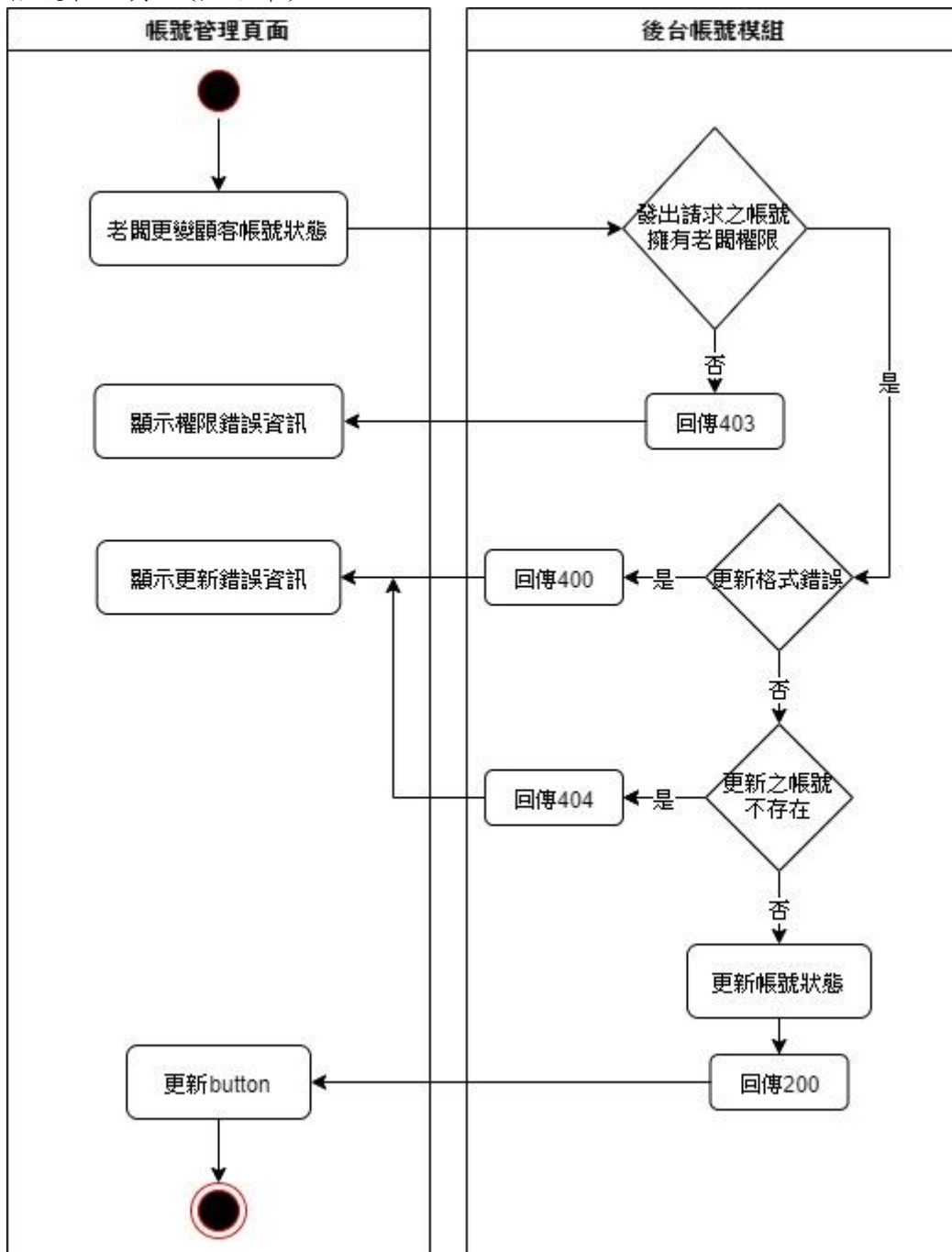
● 修改套餐頁面



● 資訊推播頁面



- 帳號管理頁面(黑名單):



4. 使用者畫面設計(User Interface Design)

● 瀏覽菜單：

- 已登入/未登入顧客之首頁為菜單頁面。此頁面會顯示各餐點之資訊，餐點項目之右邊可以選擇數量，並且按加入的按鈕即可以將想要的餐點加入點菜單(購物車)中。
- 在畫面的右邊有 search 框可以快速尋找餐點項目名稱，search 下方有種類表，提供顧客可以快速找尋某一類的所有餐點，當顧客點擊種類表其中一項，則在左邊的菜單會顯示相對應種類的項目。
- 點餐時段，顧客僅可以將取餐時間選在老闆設定好的時間內，在規定的時間外，系統會回傳該時段不接受訂餐的資訊。
- 銷售量前三名的單品及套餐，會在該名稱後面有 hot 之符號表示，代表該餐點為人氣商品。



● 瀏覽購物車：

- 在此頁面，顧客可以確認自己的點餐清單，並且對各個項目做數量的調整，也可在備註欄做客製化點餐。如果顧客不想要該項目，則可以點擊該項目後的 Delete 按鈕，即可刪除該項目；如果顧客要刪除所有餐點項目，點擊 Clear 按鈕即可。
- 在訂單送出前，需選擇預定取餐時間(某年某月某日某時某分)，否則訂單無法送出。

- 按下 Send 按鈕後，系統會判斷該顧客是否為登入狀態。
 - 若是已登入，則系統會顯示顧客下單成功或失敗。如果下單成功，畫面會自動跳轉到訂單狀態頁面；如果下單失敗，則停留在點菜單(購物車)頁面，並跳出下訂失敗之資訊。
 - 若是未登入，則跳出對話框，提醒使用者須先登入才能下單。

Ton80
菜單
點菜單(購物車)
訂單狀態

登出 你好 customer

名稱	數量	價格	備註	總共	其他
紅茶	1	99		100	Delete
紅茶	1	100		100	Delete
紅茶	1	100		100	Delete
紅茶	1	100		100	Delete
紅茶	1	100		100	Delete
紅茶	1	100		100	Delete

價格:490元

取餐時間: 2019/11/15 上午 11:20

Clear Send

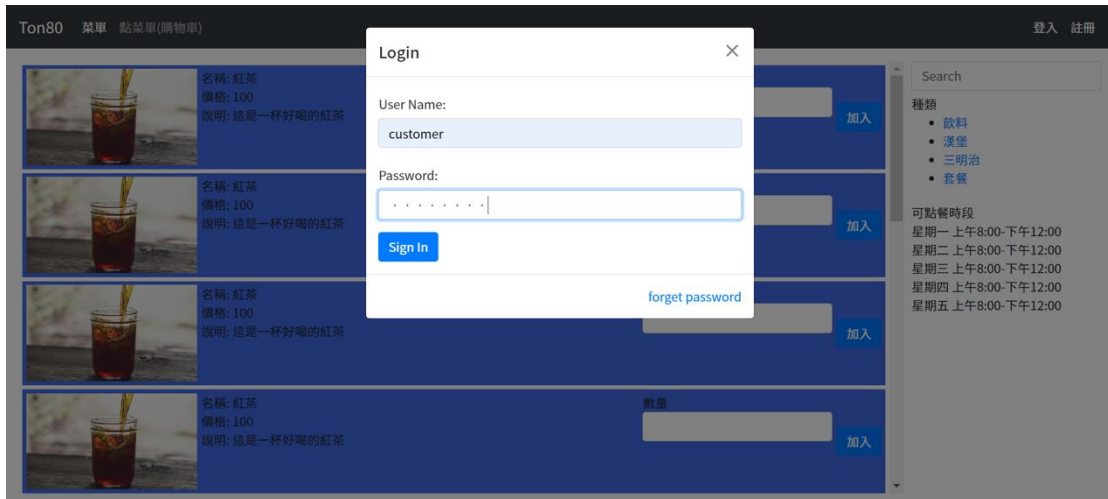
● 瀏覽訂單狀態：

- 此畫面須先登入後才可以使用。
- 此頁面提供顧客觀看訂單細項及訂單狀態。
- 此頁面會顯示訂單的目前狀態，訂單狀態可分為以下 4 種:
 - 若老闆尚未回覆，則在此頁面的該筆訂單底色會呈現黃色，且取餐序號為?，目前訂單狀態為尚未回覆。
 - 若老闆接收訂單且正在製作中，則該筆訂單底色會呈現藍色，且顯示取餐序號，目前訂單狀態為製作中。
 - 若老闆接收訂單且製作完成，則該筆訂單底色會呈現綠色，且顯示取餐序號，目前訂單狀態為製作完成。
 - 若老闆拒絕訂單，則該筆訂單底色會呈現紅色，且顯示取餐序號為 X，目前訂單狀態為取消。

取餐序號	指定取餐時間	數量及項目	目前訂單狀態
1	2019/10/10 09:00	1 * 巧克力吐司 2 * 冰紅茶	製作完成
30	2019/10/10 09:00	1 * 巧克力吐司 3 * 豬肉漢堡 2 * 冰紅茶	製作中
X	2019/10/10 09:00	1 * 巧克力吐司 3 * 豬肉漢堡 2 * 冰紅茶	取消
?	2019/10/10 09:00	1 * 巧克力吐司 3 * 豬肉漢堡 2 * 冰紅茶	未回應
?	2019/10/10 09:00	1 * 巧克力吐司 3 * 豬肉漢堡 2 * 冰紅茶	未回應

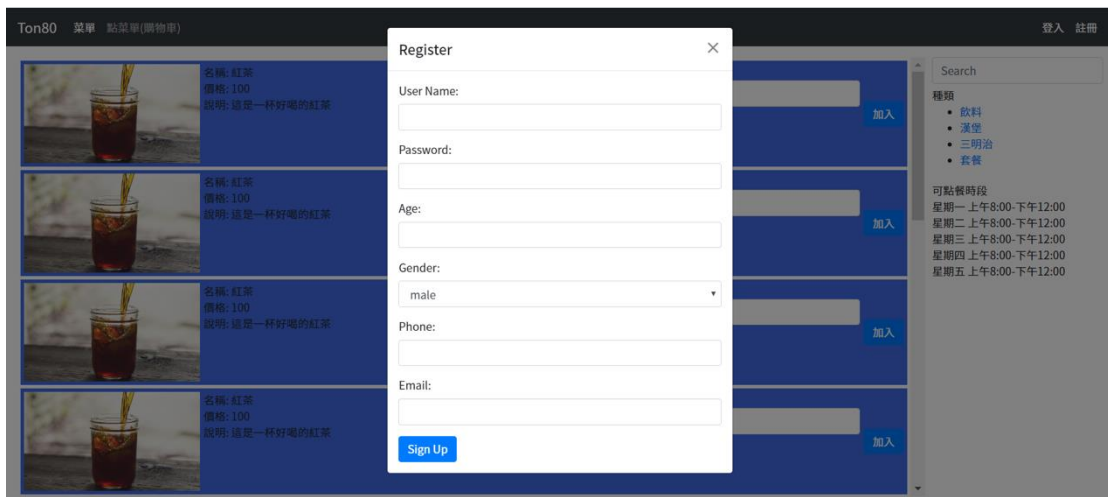
● 登入：

- 在未登入時，頁面右上角會顯示登入及註冊，點擊登入會跳出登入表單，輸入帳號密碼並按下 SIGN IN 登入，若是忘記密碼，可以點擊 forgot Password? 系統會要求使用者輸入帳號，之後系統會將密碼修改頁面之網址，傳送至使用者註冊時填寫之信箱。
- 若是沒有帳號，則按 SIGN UP 進入註冊頁面。
- 登入後，頁面右上角會顯示登出及你好，帳號名。



● 註冊：

使用者可點擊註冊按鈕，畫面會顯示註冊表單，填寫完畢點擊 Sign Up 即傳送註冊訊息給伺服器。若是伺服器驗證該帳號已經註冊過，畫面會跳出對話框說明該帳號已經存在；若該帳號尚未註冊過，則畫面會顯示註冊成功之訊息。



● 忘記密碼:

- 使用者若忘記密碼可點擊 forgot password，進入忘記密碼頁面。
- 使用者輸入註冊時輸入之信箱及帳號，點擊 submit 傳送忘記密碼訊息給伺服器。
若是伺服器驗證，該帳號設定之信箱與使用者輸入之信箱相同，系統會將重設密碼之網址寄給使用者之信箱;若是不同，頁面會顯示信箱或帳號錯誤。

Ton80 菜單 點菜單(購物車)

登入 註冊

Email

請輸入註冊時使用之email

User name

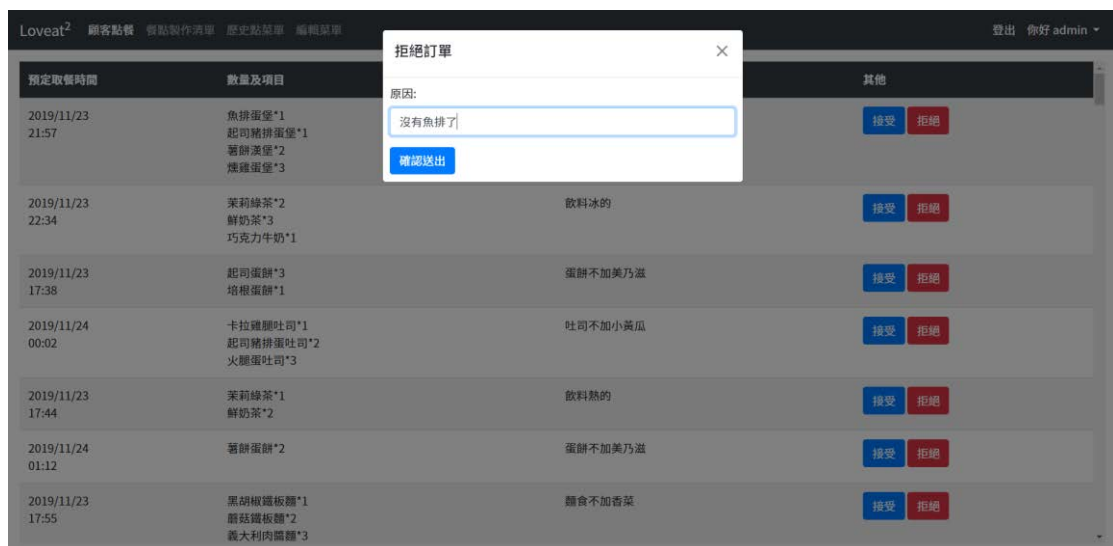
請輸入帳號

Submit

● 顧客點餐:

- 此頁面為老闆一開始看到的首頁。
- 在此頁面，老闆可以觀看每張訂單的預定取餐時間、訂單內容(項目名稱及其數量)、訂單備註，老闆可以決定是否接受/拒絕此訂單。
 - 若是按接受按鈕，則該訂單會進入餐點製作清單，同時系統將產生取餐序號並推播通知顧客老闆已接收該訂單。
 - 若是按拒絕按鈕，會出現拒絕理由填寫視窗，老闆可以在此輸入理由(也可不輸入內容)。送出後，該訂單從顧客點餐頁面移除，同時系統將推播通知顧客老闆拒絕該訂單及拒絕理由。

預定取餐時間	數量及項目	備註	其他
2019/11/23 21:57	魚排漢堡*1 起司豬排漢堡*1 薯餅漢堡*2 燻雞漢堡*3	漢堡不加小黃瓜	<button>接受</button> <button>拒絕</button>
2019/11/23 22:34	茉莉綠茶*2 鮮奶茶*3 巧克力牛奶*1	飲料冰的	<button>接受</button> <button>拒絕</button>
2019/11/23 17:38	起司蛋餅*3 培根蛋餅*1	蛋餅不加美乃滋	<button>接受</button> <button>拒絕</button>
2019/11/24 00:02	卡拉雞腿吐司*1 起司豬排蛋吐司*2 火腿蛋吐司*3	吐司不加小黃瓜	<button>接受</button> <button>拒絕</button>
2019/11/23 17:44	茉莉綠茶*1 鮮奶茶*2	飲料熱的	<button>接受</button> <button>拒絕</button>
2019/11/24 01:12	薯餅蛋餅*2	蛋餅不加美乃滋	<button>接受</button> <button>拒絕</button>
2019/11/23 17:55	黑胡椒鐵板雞*1 蘑菇鐵板雞*2 義大利肉醬麵*3	麵食不加青菜	<button>接受</button> <button>拒絕</button>



● 餐點製作清單：

在此頁面，老闆可以觀看訂單資訊，及各訂單目前之狀態。

- 此頁面會顯示每張訂單的取餐序號、預定取餐時間、數量及項目、備註、顧客資料及更新訂單狀態用之按鈕。
- 可以點擊表格標題中的取餐序號的欄位 或預定取餐時間來做訂單的排序(數字小到大)。
- 若是餐點製作完成，按下製作完成按鈕，系統會將餐點製作清單中該訂單的顏色從白色變成綠色，且製作完成的按鈕變成已取餐餐。
- 若是顧客已取走餐點，老闆按下已取餐按鈕，系統會將餐點製作清單中該訂單移到歷史點菜單。
- 若是顧客超過預定取餐時間還未取餐，老闆可以撥打顧客資料中的電話詢問。

取餐序號	預定取餐時間	數量及項目	備註	顧客資料	其他
67	2019/11/23 21:16	1 * 茉莉綠茶 2 * 鮮奶茶 3 * 巧克力牛奶	飲料去冰	0978748856 詳細資料	已取餐
68	2019/11/23 17:56	2 * 香烤肉排總匯 3 * 洋葱燒肉總匯 1 * 起司豬排總匯	總匯不加番茄醬	0938620681 詳細資料	製作完成
69	2019/11/23 22:26	1 * 巧克力牛奶	飲料去冰	0932053442 詳細資料	製作完成
73	2019/11/23 22:40	2 * 火腿蛋吐司	吐司不加美乃滋	091603966 詳細資料	製作完成
74	2019/11/23 23:30	2 * 火腿蛋吐司	吐司不加美乃滋	0968092483 詳細資料	製作完成
76	2019/11/23 18:53	2 * 火腿蛋吐司	吐司不加美乃滋	0957384513 詳細資料	製作完成
78	2019/11/23 19:53	1 * 香蕉牛奶 2 * 魚排漢堡	飲料熱的	0925456983 詳細資料	製作完成

● 歷史點菜單：

- 在此頁面，老闆可以查詢某個時間段的歷史點菜單、各種餐點的銷售情況、年齡層與性別的銷售情況。
- 在歷史點菜單中，可以觀看在該時間區段的訂單。顯示欄位包含序號、下訂單時間、項目內容及數量、及訂單備註。
- 在各種餐點的銷售情況中，可以觀看所有項目在該時間區段所賣出的數量。

Ton80 顧客點餐 餐點製作清單 歷史點菜單 編輯菜單				登出 你好 admin	
開始時間 2019/11/12 -- --:-- 結束時間 2019/11/15 -- --:-- 查詢		歷史點菜單	餐點銷售情況	年齡層與性別的銷售情況	
序號	下訂單時間	數量及項目		備註	
1	2019/10/10 09:00	1 * 巧克力吐司 2 * 冰紅茶		紅茶要去冰	
1	2019/10/10 09:00	1 * 巧克力吐司 2 * 冰紅茶		紅茶要去冰	
1	2019/10/10 09:00	1 * 巧克力吐司 2 * 冰紅茶		紅茶要去冰	
1	2019/10/10 09:00	1 * 巧克力吐司 2 * 冰紅茶		紅茶要去冰	
1	2019/10/10 09:00	1 * 巧克力吐司 2 * 冰紅茶		紅茶要去冰	
1	2019/10/10	1 * 巧克力吐司		紅茶要去冰	

● 餐點銷售情況：

- 老闆可以在此頁面觀看所有項目在該時間區段所賣出的數量。
- 點擊切換圖表按鈕，資料將會改以直方圖進行呈現。
- 圖表為直方圖：
- 直方圖：橫軸為所有項目名稱，縱軸為該項目賣出數量。

Ton80

顧客點餐

餐點製作清單

歷史點菜單

編輯菜單

登入

你好 admin

開始時間

2019/11/12 -- --:--

結束時間

2019/11/15 -- --:--

查詢

歷史點菜單

餐點銷售情況

年齡層與性別的銷售情況

切換詳細資料

項目	數量
紅茶	100
綠茶	99
燻雞三明治	97
乳酪三明治	88
乳酪蛋餅	80
豬肉漢堡	50
不好喝的紅茶	0



● 年齡層與性別的銷售情況

- 老闆在此頁面可以觀看所有項目在不同性別與年齡層間賣出的數量，性別分為男女，年齡層以 10 歲做一個區間。

老闆可以勾選性別、年齡欄位，選擇**詳細資料**呈現之欄位。

點擊切換圖表按鈕，資料將會改以圓餅圖呈現。

圖表皆為圓餅圖，分類以性別做分類：

- 女生各年齡層顧客人數圖：女性各年齡層顧客人數占女性總顧客人數。
- 男生各年齡層顧客人數圖：男性各年齡層顧客人數占男性總顧客人數。
- 男女各年齡層總顧客人數圖：女性或男性各年齡層占總顧客人數。

Ton80 顧客點餐 餐點製作清單 歷史點菜單 編輯菜單 登出 你好 admin

開始時間: 2019/11/12 -- --:--

結束時間: 2019/11/15 -- --:--

查詢

歷史點菜單 餐點銷售情況 年齡層與性別的銷售情況

性別 年齡 切換詳細資料

品項	0-9	10-19	20-29	30-39	40-49	50-59	60+
紅茶	50	40	40	40	40	40	40
綠茶	50	40	40	40	40	40	40
燻雞三明治	50	40	40	40	40	40	40
乳酪三明治	50	40	40	40	40	40	40
乳酪蛋糕	50	40	40	40	40	40	40
豬肉漢堡	50	40	40	40	40	40	40
不好喝的红茶	50	40	40	40	40	40	40



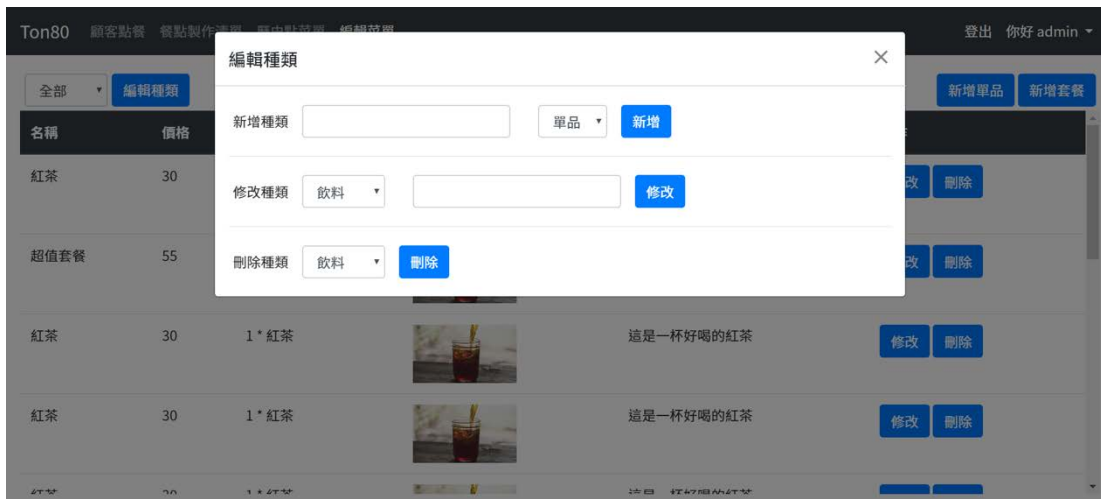
● 編輯菜單：

- 在編輯菜單的頁面中，老闆可以點擊編輯種類按鈕，畫面會彈出編輯種類畫面。
 - 新增種類：老闆手動輸入欲新增之種類名稱，並選擇是單品或是套餐，點擊新增按鈕即新增完畢。
 - 修改種類：老闆於下拉式選單選擇欲修改之種類，並於右方欄位輸入修改後之種類名稱，點擊修改按鈕即修改完畢。
 - 刪除種類：老闆於下拉式選單選擇欲刪除之種類，點擊刪除按鈕即刪除完畢。而原本屬於該種類之品項，將被移至”未分類”該種類。
- 老闆可以透過編輯種類左邊的下拉選單選擇欲觀看的餐點種類，隨即下方的表格會顯示該種類的的所有餐點。

Ton80 顧客點餐 餐點製作清單 歷史點菜單 編輯菜單 登出 你好 admin

全部 編輯種類 新增單品 新增套餐

名稱	價格	內容	照片	說明	操作
紅茶	30	1 * 紅茶		這是一杯好喝的紅茶	修改 刪除
超值套餐	55	1 * 紅茶 1 * 乳酪蛋糕		超划算的，快來買	修改 刪除
紅茶	30	1 * 紅茶		這是一杯好喝的紅茶	修改 刪除
紅茶	30	1 * 紅茶		這是一杯好喝的紅茶	修改 刪除



- 新增單品及套餐：

如果老闆要新增餐點項目，點擊新增單品按鈕進入新增單品的頁面。

■ 如果老闆要新增餐點項目，點擊新增單品按鈕會進入新增單品的頁面。

1. 老闆可以設定餐點名稱、餐點種類、單價、圖片及說明。除了照片及說明，其他欄位皆須填寫完整。
2. 若老闆要清空所有欄位的值，點擊“清空”按鈕即可。
3. 輸入完畢後，老闆可以點選“品項增加”按鈕，便完成新增一筆資料的流程。
4. 在輸入欄位的過程中，頁面下方有項目預覽可以看該項目上架在菜單上的模樣。



■ 如果老闆要新增套餐，點擊新增套餐按鈕會進入新增套餐的頁面。

1. 老闆可以設定套餐名稱、餐點種類、單價、圖片及說明。除了照片及說明，其他欄位皆須填寫完整。
2. 項目新增為老闆點選餐點種類，再從該種類中挑選餐點，輸入該餐點在該套餐的數量，點擊“添加”按鈕，則該筆項目會出現在已添加項目中。老闆

可以在該套餐中添加多個餐點項目。

3. 若老闆要清空所有欄位的值，點擊“清空”按鈕即可。

4. 輸入完畢後，老闆可以點選“品項增加”按鈕，便完成新增一筆資料的流程。

5. 在輸入欄位的過程中，頁面下方有套餐預覽可以看該項目上架在菜單上的模樣。

新豐街早餐店

127.0.0.1:5000/menu/add/combo?

Ton80 顧客點餐 餐點製作清單 歷史點菜單 編輯菜單 登出 你好 admin

套餐新增

名稱

種類

單價

照片 未選擇任何檔案

說明

品項添加

選擇種類	飲料	名稱	數量	其他
品項選擇	紅茶	紅茶	1	<input type="button" value="刪除"/>

數量選擇

套餐預覽

名稱	單價	品項
超值套餐	55	1 * 紅茶

- 修改單品及套餐：

如果老闆要修改菜單上的細節，點擊該餐點項目後的修改按鈕，若該項目為一般項目，則進入一般項目修改頁面；若該項目為套餐，則進入套餐修改頁面。

修改頁面與新增頁面在畫面呈現方式及操作方法完全相同，差別僅在修改頁面會預先填上該商品之資訊。項目修改完畢後點擊確認修改按鈕即修改完畢。

如果老闆要刪除整筆餐點，點擊該項目的刪除項目按鈕即可刪除。

Ton80

顧客點餐

餐點製作清單

歷史點菜單

編輯菜單

登出

你好 admin

單品修改

名稱

紅茶

種類

飲料

單價

100

圖片

選擇檔案 | 未選擇任何檔案

說明

這是一杯好喝的紅茶

清空

確認修改



名稱: 紅茶
價格: 100
說明: 這是一杯好喝的紅茶

Ton80

顧客點餐

餐點製作清單

歷史點菜單

編輯菜單

登出

你好 admin

套餐修改

名稱

超值套餐

種類

套餐

單價

55

圖片

選擇檔案 | 未選擇任何檔案

說明

超划算的，快來買

品項添加

選擇種類

飲料

品項選擇

紅茶

數量選擇


1

添加

清空

確認修改

名稱	數量	其他
紅茶	1	<div>刪除</div>
乳酪蛋糕	1	<div>刪除</div>



名稱: 超值套餐
價格: 55
品項:
1 * 紅茶

● 設定點餐時間：

- 老闆可以透過點擊右上角個人選單之設定點餐時間進入設定點餐畫面。
- 老闆可以直接手動輸入點餐時間，格式為上午/下午 XX:XX 上午/下午 XX:XX，若當日不提供點餐，則欄位留空，修改完畢按 Save 按鈕即完成修改。

Ton80 顧客點餐 餐點製作清單 歷史點菜單 編輯菜單 登出 你好 admin ▾

星期一	上午 08:00	下午 12:00
星期二	上午 08:00	下午 12:00
星期三	上午 08:00	下午 12:00
星期四	上午 08:00	下午 12:00
星期五	上午 08:00	下午 12:00
星期六	-- --:--	-- --:--
星期日	-- --:--	-- --:--


save

● 顧客/老闆個人資料：

- 顧客/老闆可以透過點擊右上角個人選單之個人資料進入個資修改畫面。
- 顧客/老闆可以觀看，並修改自己的個人資料(除了帳號不能修改外)，使用者可以直接修改欄位的值，待修改完畢後，按下右下角的 Save 按鈕，即資料修改完畢且資料更新。
- 老闆觀看顧客個人資料頁面時，頁面只顯示該顧客資料，不會顯示修改按鈕。
- 密碼修改較為特殊，點擊修改按鈕會有彈出式畫面，請使用者輸入原本的密碼，及新的密碼，以確保是使用者本人進行。按下 OK 按鈕後，會像伺服器發出修改密碼請求，若是輸入之原本密碼錯誤，會跳出密碼輸入錯誤之訊息;若是正確，則跳出密碼更新成功之訊息。

觀看自己之個人資料畫面。

Ton80 菜單 點菜單(購物車) 訂單狀態 登出 你好 customer ▾



User Name:

customer

Password: 修改

Age:

25

Gender:

male ▾

Phone:

0905783383

Email:

customer@gmail.com

Save

老闆觀看顧客個人資料畫面。

Ton80 顧客點餐 餐點製作清單 歷史點菜單 編輯菜單

登出 你好 admin



User Name:

customer

Age:

25

Gender:

male

Phone:

0905783383


Email:

customer@gmail.com

修改密碼畫面。

Ton80 菜單 點菜單(購物車) 訂單狀態

登出 你好 customer



修改密碼

×

請輸入原本密碼:

請輸入新密碼:

再輸入一次新密碼:

Cancel OK

User Name:

customer

Age:

25

Gender:

male

Phone:

0905783383

Email:

customer@gmail.com

Save

- 新品推播：
在此頁面，老闆可以輸入欲推播的內容給顧客。

Loveat² 顧客點餐 餐點製作清單 歷史點菜單 編輯菜單

登出 你好 admin

Title

Content

Submit

個人資料

管理頁面

設定點餐時間

推播資訊

帳號管理

- 帳號管理(黑名單)：

- 此頁面老闆可以觀看顧客的帳戶。
- 老闆可以選擇將顧客加入或移除黑名單。點擊加入按鈕，則該帳號被加入黑名單，無法使用訂餐功能;點擊移除按鈕，則該顧客帳號移出黑名單，恢復訂餐功能。
- 加入黑名單後，該名顧客的欄位白色變成灰色。
- 可利用搜索來找尋目標帳號。

Loveat ² 顧客點餐 餐點製作清單 歷史點菜單 編輯菜單				登出 你好 admin ▾
輸入帳號搜尋		搜尋帳號		個人資料 管理頁面 設定點餐時間 推播資訊 帳號管理
帳號	電話	信箱	黑名單	
peipeili	0920198409	suara1201fxt@gmail.com	加入	
river40333	0939783383	river40333@gmail.com	移除	
admin	0920198409	admin@gmail.com	加入	
test0	0989564301	test0@gmail.com	加入	
test1	0900000000	testforupdate@gmail.com	加入	
test2	0962409524	test2@gmail.com	加入	
test3	0989129961	test3@gmail.com	加入	
test4	0978748856	test4@gmail.com	加入	

5. 資料設計(Data Design)

1. user collection: 儲存使用者資訊

1. 欄位

- _id: ObjectID，由 mongoDB 自動生成
- userName: String，使用者帳號
- password: String，使用者密碼，會先經過 hash 才存進資料庫
- gender: String，使用者性別，值可能為 female、male
- phone: String，使用者手機
- email: String，使用者 email
- birth: Date，使用者生日，會以使用者輸入之年紀預估其生日
- role: String，使用者權限，值可能為 admin、customer
- avatar: String，照片編號，為 UUID，可對應到 image collection 中的照片
- token: String，推播用之 token
- state: String，使用者狀態，值可能為 activate、frozen

2. 範例

```
_id: ObjectId("5dd6a8f302c91829ef248162")
userName: "peipeili"
password: "pbkdf2:sha256:150000$rIu3w2YP$9202d7107ec3c49e853ced4ad26b4a50938c4cc4..."
gender: "female"
phone: "0920198409"
email: "suara1201fxt@gmail.com"
birth: 2018-03-27T00:58:51.538+00:00
role: "customer"
avatar: "89ad9537-ded1-4f2c-a335-32ffbf4d5d39"
token: "fRTJcXIjOhko5059xYLgtW:APA91bEAuZHdz3q6VJbJVbqg66AcjqCRjcZnG5r1qdg3-zw..."
state: "activate"
```

```
_id: ObjectId("5dd752c0ccce596d624d9a9d")
userName: "river40333"
password: "pbkdf2:sha256:150000$6C5oQK2B$f23b6e6605060e290773ae9da29477cb13ef6992..."
gender: "female"
phone: "0939783383"
email: "river40333@gmail.com"
birth: 1999-12-11T19:40:37.488+00:00
role: "customer"
avatar: "d4fbb0eb-1824-4cd9-bd5f-036eec452423"
token: "dry720IFjpLEm9F1DFLa-5:APA91bFcb24XkSgFgpysHwka7ldy2vBE1c0M2zwxXmgvR8r..."
state: "frozen"
```

2. type collection: 儲存使用者自定義之類別

1. 欄位

- `_id`: ObjectId, 由 MongoDB 自動生成
- `name`: String, 使用者自定義之類別名稱
- `category`: String, 類別, 值可能為 `item`、`combo`

2. 範例

```
_id: ObjectId("5db31a99e8603473845fece7")
name: "飲料"
category: "item"
```

```
_id: ObjectId("5db31aace8603473845fece8")
name: "超值套餐"
category: "combo"
```

3. image collection: 儲存照片

1. 欄位

- `_id`: ObjectId, 由 MongoDB 自動生成
- `uuid`: String, UUID 編號, 將照片新增至資料庫時由伺服器生成
- `picture`: Binary, 照片

2. 範例

```
_id: ObjectId("5dd0b7b090e5d25818d64c6b")
uuid: "a0b819e4-e443-469c-abfc-76aae9dd7ecf"
picture: Binary('iVBORw0KGgoAAAANSUHEugAABNYAAPSCAYAAAC3Q1cnAAAMTWIDQ1B1JQ0MgUHJvZm1sZQAASImVvwdck0cbv3dkkrACEZAR9hJ1...')
```

4. order collection: 儲存訂單資訊

1. 欄位

- `_id`: ObjectId, 由 mongoDB 自動生成
- `orderId`: String, 訂單 ID, 由伺服器生成
- `createdAt`: Date, 訂單成立時間, 伺服器收到訂單時生成
- `takenAt`: Date, 預定取餐時間
- `userName`: String, 下訂該筆訂單之帳號
- `state`: String, 訂單狀態, 值可能為 unknown、doing、finish、end、cancel
- `notes`: String, 訂單備註, 紀錄使用者客製化訂單之需求
- `total`: Int32, 訂單總額
- `content`: Array, 訂單內容, 該 array 儲存之 Object 結構如下
 - `id`: ObjectId, 商品之 id
 - `name`: String, 商品名稱
 - `category`: String, 商品種類, 值可能為 combo、item
 - `quantity`: Int32, 商品數量

2. 範例

```
_id: ObjectId("5dd8f94ff5a90a5568400a57")
orderId: "2"
takenAt: 2019-11-23T18:08:07.908+00:00
state: "doing"
userName: "test86"
content: Array
  0: Object
    _id: ObjectId("5dd688d30e5cb17b1d4d29fd")
    type: "item"
    quantity: 1
    name: "西西里雞腿堡"
  1: Object
    _id: ObjectId("5dd688d30e5cb17b1d4d29ff")
    type: "item"
    quantity: 2
    name: "起司豬排蛋堡"
notes: "漢堡加蛋"
total: 165
createdAt: 2019-11-23T17:18:07.908+00:00
```

5. item collection: 儲存單品資訊

1. 欄位

- `_id`: ObjectId, 由 mongoDB 自動生成
- `type`: ObjectId, 商品類別, 可以對應到 `type collection`
- `name`: String, 商品名稱
- `picture`: String, 照片編號, 為 UUID, 可以對應到 `image collection` 中的照片
- `price`: Int32, 商品價錢
- `description`: String, 商品說明
- `sell`: Int32, 商品銷售量

2. 範例

```
_id: ObjectId("5dd67f098f0f6afb3ebc1b69")
type: ObjectId("5de35bb170a5892250dfdc6c")
name: "奶茶"
picture: "6a9843d0-7789-4a7b-ab5b-3c79ccad4310"
price: 20
description: "奶精, 小心會拉肚子"
sell: 6
```

```
_id: ObjectId("5dd6899dd268a673de091edc")
type: ObjectId("5dd678b95f19051c7c4f0bb3")
name: "培根蛋吐司"
picture: "6558a154-fc fb-4baf-927d-fb40dc69586a"
price: 30
description: ""
sell: 10
```

6. combo collection: 儲存套餐資訊

1. 欄位

- `_id`: ObjectId, 由 mongoDB 自動生成
- `type`: ObjectId, 套餐類別, 可以對應到 `type` collection
- `name`: String, 套餐名稱
- `picture`: String, 照片編號, 為 UUID, 可以對應到 `image` collection 中的照片
- `price`: Int32, 套餐價錢
- `description`: String, 套餐說明
- `content`: Array, 套餐包含之單品, 該 array 儲存之 Object 結構如下
 - `id`: ObjectId, 商品 id
 - `quantity`: Int32, 商品數量
 - `name`: String, 商品名稱
- `sell`: Int32, 商品銷售量

2. 範例

```
_id: ObjectId("5dda5c6a24e95f5c72b8c444")
type: ObjectId("5de35d6370a5892250dfdc6f")
name: "超值套餐"
picture: "aa4cf83b-7ae0-4b89-b3fd-ade750b94799"
price: 75
description: ""
content: Array
  0: Object
    id: ObjectId("5dd6899ad268a673de091ed8")
    quantity: 1
    name: "卡拉雞腿吐司"
  1: Object
    id: ObjectId("5dd6a6e0b1ee120dfb88775c")
    quantity: 1
    name: "薯餅"
  2: Object
    id: ObjectId("5dd67f098f0f6afb3ebc1b68")
    quantity: 1
    name: "紅茶"
sell: 1
```

7. businessTime collection: 儲存營業時間資訊

1. 欄位

- `_id`: ObjectID, 由 mongoDB 自動生成
- `mon`: Object, 星期一營業時間, Object 結構如下
 - `start`: String, 開始營業時間(24 小時制)
 - `end`: String, 結束營業時間(24 小時制)
- `tue`: Object, 星期二營業時間, Object 結構如下
 - `start`: String, 開始營業時間(24 小時制)
 - `end`: String, 結束營業時間(24 小時制)
- `wed`: Object, 星期三營業時間, Object 結構如下
 - `start`: String, 開始營業時間(24 小時制)
 - `end`: String, 結束營業時間(24 小時制)
- `thu`: Object, 星期四營業時間, Object 結構如下
 - `start`: String, 開始營業時間(24 小時制)
 - `end`: String, 結束營業時間(24 小時制)
- `fri`: Object, 星期五營業時間, Object 結構如下
 - `start`: String, 開始營業時間(24 小時制)
 - `end`: String, 結束營業時間(24 小時制)
- `sat`: Object, 星期六營業時間, Object 結構如下
 - `start`: String, 開始營業時間(24 小時制)
 - `end`: String, 結束營業時間(24 小時制)
- `sun`: Object, 星期日營業時間, Object 結構如下
 - `start`: String, 開始營業時間(24 小時制)
 - `end`: String, 結束營業時間(24 小時制)

2. 範例

```
_id: ObjectId("5dd13dc899b86faf7d945987")
mon: Object
  start: "06:00"
  end: "13:00"
tue: Object
  start: "06:00"
  end: "13:00"
web: Object
  start: "06:00"
  end: "13:00"
thu: Object
  start: "06:00"
  end: "13:00"
fri: Object
  start: "06:00"
  end: "13:00"
sat: Object
  start: "06:00"
  end: "13:00"
sun: Object
  start: "06:00"
  end: "13:00"
```

[illegible]

深黃色：推播模組

淺藍色： 訂單模組

深藍色：顧客頁面

藍綠色：前台通用頁面

粉紅色：老闆頁面

橘紅色：資料庫模組

灰色：使用者模組

Class Name	db
Description	處理對資料庫進行連結之操作
Relationship with other classes	user、menu(model)、order、business_time、image
Traceability with other use case	無

Class Name	user
Description	處理對資料庫內 user collection 的操作
Relationship with other classes	user_web_route、user_api_route、db
Traceability with other use case	註冊、登入、修改個資、找回密碼

Class Name	menu(model)
Description	處理對資料庫內 type, item, combo, collection 的操作
Relationship with other classes	menu_web_route、menu_api_route、db
Traceability with other use case	瀏覽菜單、編輯菜單、上架單品、設定套餐、編輯種類

Class Name	order
Description	處理對資料庫內 order collection 的操作
Relationship with other classes	order_web_route、order_api_route、db
Traceability with other use case	送出訂單、查看訂單狀態、回覆顧客點菜單請求、管理製作清單、查詢歷史點菜單

Class Name	busiess_time
Description	處理對資料庫內 businessTime collection 的操作
Relationship with other classes	another_web_route、menu_web_route、setting_api_route、order_api_route、db
Traceability with other use case	瀏覽菜單、送出訂單、設置營業時間

Class Name	image
Description	處理對資料庫內 image ，collection 的操作
Relationship with other classes	root_route
Traceability with other use case	瀏覽菜單、編輯菜單、上架單品、設定套餐

Class Name	push
Description	連接 firebase，並進行推播
Relationship with other classes	order_api_route
Traceability with other use case	查看訂單狀態、回覆顧客點餐請求、管理製作清單

Class Name	order_api_route
Description	負責跟 order 有關之 api
Relationship with other classes	push、business_time、order、main
Traceability with other use case	送出訂單、查看訂單狀態、回覆顧客點餐請求、管理製作清單、查詢歷史點菜單

Class Name	setting_api_route
Description	負責跟設定有關之 api
Relationship with other classes	business_time、main
Traceability with other use case	設置營業時間

Class Name	user_api_route
Description	負責跟 user 有關之 api
Relationship with other classes	user、main
Traceability with other use case	登入、註冊、修改個資、找回密碼

Class Name	menu_api_route
Description	負責跟 menu 有關之 api
Relationship with other classes	type、item、combo、main
Traceability with other use case	瀏覽菜單、編輯菜單、編輯種類、上架單品、設定套餐

Class Name	order_web_route
Description	負責跟 order 有關之頁面與購物車（點菜單）頁面
Relationship with other classes	order、main、order_state、order_todo、new_order、order_history、cart.html
Traceability with other use case	送出訂單、查看訂單狀態、回覆顧客點餐請求、管理製作清單、查詢歷史點菜單

Class Name	menu_web_route
Description	負責跟 menu 有關之頁面
Relationship with other classes	type、item、combo、main、menu、edit_menu、new_edit_combo、new_edit_item
Traceability with other use case	瀏覽菜單、編輯菜單、編輯種類、上架單品、設定套餐

Class Name	setting_web_route
Description	負責跟設定有關之頁面
Relationship with other classes	business_time、main、edit_business_time
Traceability with other use case	設置營業時間

Class Name	user_web_route
Description	負責跟 user 有關之頁面
Relationship with other classes	user、main、user_info、forget_password、reset_password
Traceability with other use case	修改個資、找回密碼

Class Name	root_route
Description	負責根路徑的處理
Relationship with other classes	main
Traceability with other use case	查看訂單

Class Name	main
Description	程式進入點
Relationship with other classes	auth、user_web_route、setting_web_route、menu_web_route、order_web_route、user_api_route、setting_api_route、menu_api_route、order_api_route
Traceability with other use case	登入、註冊、修改個資、找回密碼、瀏覽菜單、送出訂單、查看訂單狀態、設置營業時間、編輯菜單、編輯種類、上架單品，設定套餐、回覆顧客點餐請求、管理製作清單、查詢歷史點菜單

Class Name	service_woker
Description	處理背景推播
Relationship with other classes	layout
Traceability with other use case	無

Class Name	auth
Description	權限管理
Relationship with other classes	main、user
Traceability with other use case	修改個資、送出訂單、查看訂單狀態、設置營業時間、編輯菜單、編輯種類、上架單品，設定套餐、回覆顧客點餐請求、管理製作清單、查詢歷史點菜單

Class Name	manage_user
Description	帳號管理(黑名單)
Relationship with other classes	user_web_route, layout, fetch
Traceability with other use case	顧客狀態更新

Class Name	reset_password
Description	重設密碼
Relationship with other classes	user_web_route, layout, fetch
Traceability with other use case	找回密碼

Class Name	forget_password
Description	送出重設密碼請求
Relationship with other classes	user_web_route, layout, fetch
Traceability with other use case	找回密碼

Class Name	user_info
Description	更新個人資訊
Relationship with other classes	user_web_route, layout, fetch
Traceability with other use case	修改個資

Class Name	cart(html)
Description	將餐點加入購物車並顯示已加入的餐點
Relationship with other classes	order_web_route, layout, fetch, cart.js
Traceability with other use case	瀏覽菜單

Class Name	edit_bussiness_time
Description	編輯顧客點餐時間

Relationship with other classes	setting_web_route, layout, fetch
Traceability with other use case	設置營業時間

Class Name	new_edit_item
Description	新增和編輯單品
Relationship with other classes	menu_web_route, layout, fetch
Traceability with other use case	上架單品、編輯菜單

Class Name	new_edit_combo
Description	新增和編輯套餐
Relationship with other classes	menu_web_route, layout, fetch
Traceability with other use case	設定套餐、編輯菜單

Class Name	edit_menu
Description	刪除種類、單品、套餐，新增、編輯種類
Relationship with other classes	menu_web_route, layout, fetch
Traceability with other use case	編輯菜單、編輯種類

Class Name	menu
Description	取得所有餐點資訊
Relationship with other classes	menu_web_route, layout, fetch, cart.js
Traceability with other use case	瀏覽菜單

Class Name	order_history
Description	查看顧客歷史點餐紀錄
Relationship with other classes	order_web_route, layout, fetch
Traceability with other use case	查詢歷史點菜單

Class Name	new_order
Description	更新顧客點餐請求
Relationship with other classes	order_web_route, layout, fetch
Traceability with other use case	回覆顧客點餐請求

Class Name	order_todo
Description	接受或拒絕顧客點餐
Relationship with other classes	order_web_route, layout, fetch
Traceability with other use case	回覆顧客點餐請求

Class Name	order_state
Description	設定已接受餐點之狀態
Relationship with other classes	order_web_route, layout
Traceability with other use case	管理製作清單

Class Name	News_push
Description	推播訊息給顧客
Relationship with other classes	setting_web_route, layout、fetch
Traceability with other use case	推播新資訊給顧客

Class Name	layout
Description	顯示網頁頁面
Relationship with other classes	reset_password (python), forget_password, user_info, cart, edit_bussiness_time, new_edit_item, new_edit_combo, edit_menu, menu, order_history, new_order, order_todo, order_state, navbar, login, register, serveice_workout
Traceability with other use case	註冊、登入、修改個資、找回密碼、顯示菜單、送出訂單、查看訂單狀態、設置營業時間、編輯菜單、編輯種

	類、上架單品、設定套餐、回覆顧客點餐請求、管理製作清單、查詢歷史點菜單
--	-------------------------------------

Class Name	cart (javascript)
Description	對 local storage 儲存之訂單進行操作
Relationship with other classes	cart, menu
Traceability with other use case	瀏覽菜單、送出訂單

Class Name	navbar
Description	提供導覽列
Relationship with other classes	layout
Traceability with other use case	註冊、登入、修改個資、找回密碼、顯示菜單、送出訂單、查看訂單狀態、設置營業時間、編輯菜單、編輯種類、上架單品、設定套餐、回覆顧客點餐請求、管理製作清單、查詢歷史點菜單

Class Name	login
Description	登入帳號
Relationship with other classes	layout
Traceability with other use case	登入

Class Name	register
Description	註冊帳號
Relationship with other classes	layout
Traceability with other use case	註冊

Class Name	fetch
Description	獲取後台資料
Relationship with other classes	reset_password, forget_password, user_info, cart, edit_bussiness_time, new_edit_item, new_edit_combo, edit_menu, menu, order_history, new_order, order_todo
Traceability with other use case	找回密碼、修改個資、瀏覽菜單、設置營業時間、上架單品、編輯菜單、設定套餐、編輯種類、瀏覽菜單、查詢歷史點菜單、回覆顧客點餐請求、回覆顧客點餐請求

7. 實作方案(Implementation Languages and Platforms)

- 前端：純 html、css、JavaScript 撰寫，額外搭配 chart.js 進行圖表繪製，且為了讓使用者在電腦或移動端平台皆方便使用，系統以 RWD 響應式網頁的方式呈現。
- 後端：以 python 為主要語言搭配 flask 框架，除了其輕量的 web 架構方便開發，還有大量的第三方模組可以進行擴充。
- 資料庫：非關聯式 mongoDB，方便我們儲存巢狀式的資料。
- 部署平台：選用 GCP (Google Cloud Platform) 上的 Serverless 服務 GAE (Google App Engine) 進行部署，方便自動化整合

8. 設計議題(Design Issue)

- 議題 1：實作訂單接受通知功能
 - 可能方案：
 - 1) e-mail 通知顧客訂單相關訊息如老闆已接受顧客的訂單。
 - 2) 顧客於送出點菜單後停在等候頁面等待老闆確認訂單。
 - 3) 顧客跳回主畫面，並可自由操作系統，接受訂單等相關訊息藉由推播功能通知。
 - 最後選擇方案：以推播功能通知顧客。
 - 原因：較貼合顧客使用情況，避免停留頁面的等待時間過長造成使用觀感差，也讓顧客不須額外跳轉信箱頁面接收訊息，操作更簡易與方便。
- 議題 2：忘記密碼如何處理
 - 最後選擇方案：於註冊頁面新增 e-mail 欄位，往後若忘記密碼，點取忘記密碼的按鈕，輸入帳號後，會將更改密碼的網站連結寄給當初註冊的信箱以供使用者更改密碼。
 - 原因：最符合群眾對於忘記密碼後的使用流程印象。

- 議題 3：為何選用 Google App Engine 作為開發平台？
 - 原因：因為其具備高頻寬低延遲、第一年免費使用的 300 元美金額度與提供推播功能所需的 SSL 等優點，因此選擇 Google App Engine 作為部署平台。
- 議題 4：網頁端與伺服器端資料傳輸
 - 可能選擇方案：
 - 1) jQuery 的 Ajax
 - 2) Fetch
 - 最後選擇方案：Fetch
 - 原因：因為 JavaScript 原生的 Fetch 不須引入龐大的 jQuery 程式庫，可以減少性能的損耗。
- 議題 5：顧客點選菜單內容加入購物車實作構想
 - 最後選擇方案：將菜單頁面與購物車頁面對 localStorage 的操作，封裝成模組，提供對儲存於 localStorage 之訂單進行新增、刪除及修改之介面。
 - 原因：方便管理 localStorage 的前綴 ID，日後維護較容易。
- 議題 6：資料庫系統選擇
 - 可能方案：
 - 1) MySQL
 - 2) MongoDB
 - 最後選擇方案：MongoDB。
 - 原因：因為 MongoDB 會將資料以類似 JSON 格式之文件進行儲存，其格式較為自由且支援 array 及 object 型態，對於需要巢狀結構的訂單及套餐資料，在開發上較為方便。
- 議題 7：Web 應用框架選擇
 - 可能方案：
 - 1) Flask
 - 2) Django
 - 最後選擇方案：Flask。
 - 原因：Flask 相較之下較輕量，且模組化的設計可以輕鬆適應開發人員需求。

- 議題 8：歷史訂單分析資料使用的 API 回傳格式，為何不用一個物件回傳，使其較易讀也較直覺？
 - 原因：目前使用的回傳格式，可以使其更具資料彈性，可彈性更改年齡間隔，使顯示之分析資料更為詳盡，並減輕前端計算負擔，前端可直接取用後端傳送之資訊繪出表格。
 - 可能遭遇問題：資料遺漏或缺失，例如：女性有年齡 60 以上的客戶，男性則無。
 - 解決方案：後端自動補零。

- 議題 9：前端肥大
 - 可能遭遇問題：程式碼較難維護。
 - 解決方案：將目前的前端架構分得更細。

- 議題 10：API 中 ID 之生成方式

- 討論：MongoDB 於新增資料時便會自動配置一組 ID。若需要系統生成獨一無二的 ID（例如：品項所需之圖片 ID），則使用 python 的 uuid 庫，並選擇以隨機數字來產生 ID。