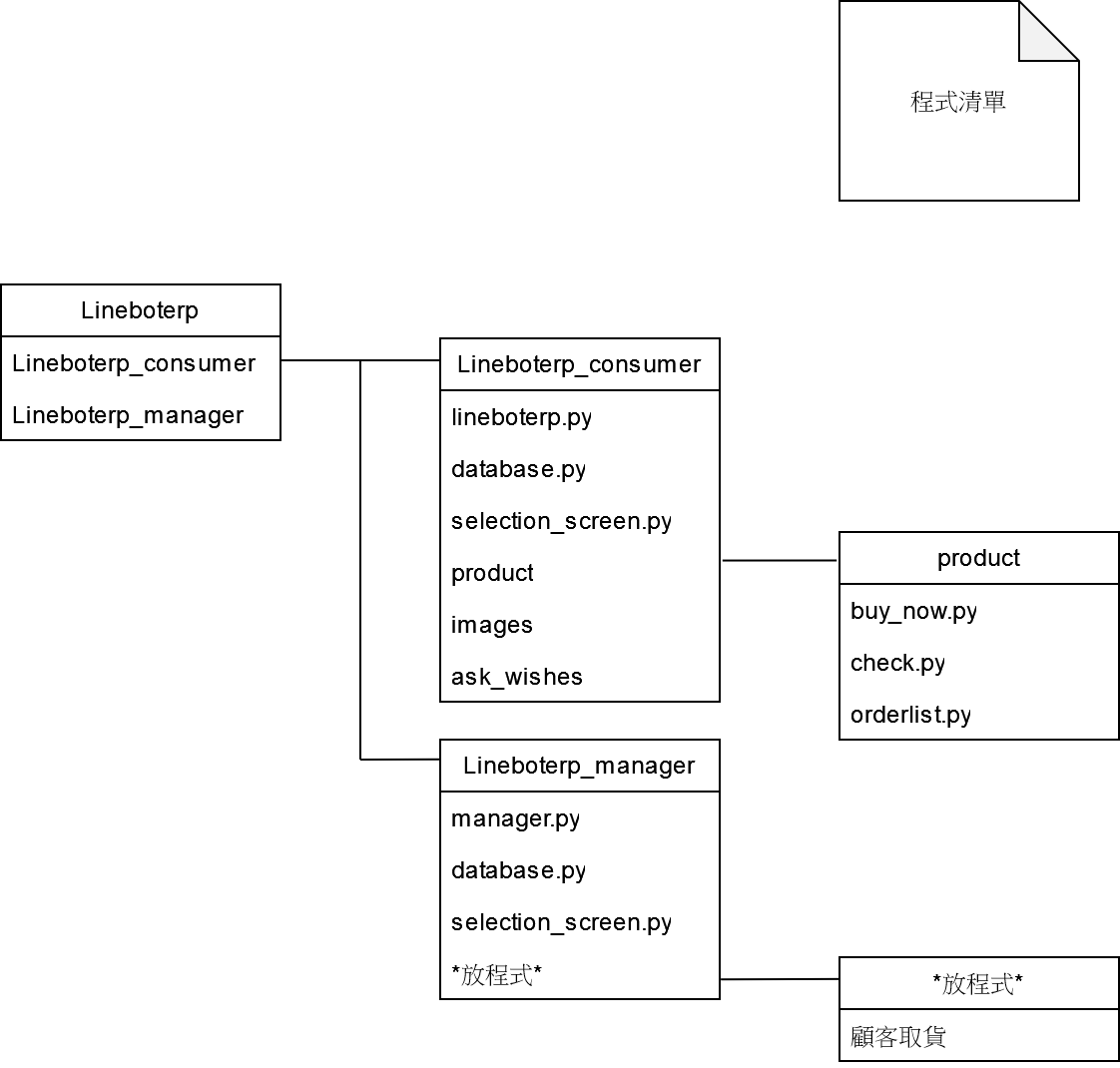
第九章 程式規格



9-1軟體架構與程式清單

|  |  |  |
| --- | --- | --- |
| 資料夾名稱: Lineboterp | | |
| 編號 | 檔案名稱 | 功能 |
| 1-1 | Lineboterp\_consumer | 資料夾-存放消費者的程式 |
| 1-2 | Lineboterp \_manager | 資料夾-存放管理者的程式 |
| 資料夾名稱: Lineboterp \_ consumer | | |
| 編號 | 檔案名稱 | 功能 |
| 2-1 | lineboterp.py | 處理消費者相關的關鍵字 |
| 2-2 | database.py | 新增、修改和查詢資料庫 |
| 2-3 | selection\_screen.py | 文字訊息美編 |
| 2-4 | product | 資料夾-存放主程式 |
| 2-5 | images | 資料夾-存放消費者上傳的圖片 |
| 資料夾名稱:product | | |
| 編號 | 檔案名稱 | 功能 |
| 3-1 | buy\_now.py | 購買商店內的現貨商品 |
| 3-2 | check.py | 檢查使用者狀態 |
| 3-3 | orderlist.py | 個人的訂單列表 |
| 資料夾名稱: Lineboterp \_manager | | |
| 編號 | 檔案名稱 | 功能 |
| 4-1 | manager.py | 處理管理者相關的關鍵字 |
| 4-2 | database.py | 新增、修改和查詢資料庫 |
| 4-3 | selection\_screen.py | 文字訊息美編 |
| 4-4 | \*\* | 資料夾-存放主程式 |
| 4-5 | images | 資料夾-存放報表存放的圖片 |

9-2 其他附屬之各種元件

|  |  |  |  |
| --- | --- | --- | --- |
| 編號 | 2-1 | 程式名稱 | lineboterp.py |
| 目的 | 當使用者按下按鈕發送關鍵字後，系統做出回應 | | |
| 部分程式碼 | | | |
| #-------------------團購商品及2種商品列表--------------  elif '團購商品' in msg:       line\_bot\_api.reply\_message(event.reply\_token, Order\_preorder\_selectionscreen())  elif '【預購商品】列表' in msg:       list\_page[user\_id+'預購min'] = 0       list\_page[user\_id+'預購max'] = 9       product\_show = product\_preorder\_list()       line\_bot\_api.reply\_message(event.reply\_token,product\_show)  elif '【現購商品】列表' in msg:       list\_page[user\_id+'現購min'] = 0       list\_page[user\_id+'現購max'] = 9       buynow\_show = product\_buynow\_list()       line\_bot\_api.reply\_message(event.reply\_token,buynow\_show)  #-------------------查詢、訂單、購物車------------------  elif '訂單/購物車查詢' in msg:       line\_bot\_api.reply\_message(event.reply\_token, Order\_cart\_selectionscreen())  elif '訂單查詢' in msg:       line\_bot\_api.reply\_message(event.reply\_token, Notpickedup\_preordered\_history\_selectionscreen())  elif '未取訂單列表' in msg:       ordernottaken = ordernottaken\_list()       line\_bot\_api.reply\_message(event.reply\_token, ordernottaken)  elif '預購訂單列表' in msg:       line\_bot\_api.reply\_message(event.reply\_token, orderpreorder\_list())  elif '歷史訂單列表' in msg:       history = orderhastaken\_list()       line\_bot\_api.reply\_message(event.reply\_token, history) | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 編號 | 2-2 | 程式名稱 | database.py |
| 目的 | 當主程式要新增、修改和查詢資料庫時，會呼叫相對應的功能 | | |
| 部分程式碼 | | | |
| #-------------------查詢預購商品列表-------------------  def preorder\_list():    query = """            SELECT 商品ID, 商品名稱, 現預購商品, 商品圖片, 商品簡介, 商品單位, 售出單價, 售出單價2, 預購數量限制\_倍數,預購截止時間            FROM Product\_information            WHERE 現預購商品='預購';"""  category ='select' #重試類別select/notselect    preorder\_result,result2 = retry(category,query)    if preorder\_result != []:      listbuynow = preorder\_result    else:      listbuynow = "找不到符合條件的資料。"    return listbuynow  #-------------------預購訂單查詢(100筆)----------------  def orderpreorderlist():    userid = lineboterp.user\_id    query = f"""      select 訂單編號,總額,訂單成立時間      from `Order\_information`      where 會員\_LINE\_ID = '{userid}' and (訂單狀態未取已取 like '預購%') and 訂單狀態未取已取 <> '預購已取'      order by 訂單成立時間 desc      limit 100 offset 0;      """#下一頁加100改offset(目前暫無考慮)    category ='select' #重試類別select/notselect    preorder\_result,result2 = retry(category,query)    if preorder\_result == []:      preorder\_result = '找不到符合條件的資料。'    return preorder\_result  #-------------------查詢預購商品列表-------------------  def preorder\_list():    query = """            SELECT 商品ID, 商品名稱, 現預購商品, 商品圖片, 商品簡介,商品單位, 售出單價, 售出單價2, 預購數量限制\_倍數,  預購截止時間            FROM Product\_information            WHERE 現預購商品='預購';"""    category ='select' #重試類別select/notselect    preorder\_result,result2 = retry(category,query)    if preorder\_result != []:      listbuynow = preorder\_result    else:      listbuynow = "找不到符合條件的資料。"    return listbuynow | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 編號 | 2-3 | 程式名稱 | selection\_screen.py |
| 目的 | 讓系統回傳給使用者的訊息能用Flexmessage格式美化 | | |
| 部分程式碼 | | | |
| #-------------------預購商品的選擇畫面------------------  def Order\_preorder\_selectionscreen():      Order\_preorder\_screen = []      Order = {              "type": "bubble",              "hero": {                  "type": "image",                  "url": "https://i.imgur.com/vLCC99Q.jpg",                  "size": "full",                  "aspectRatio": "20:13",                  "aspectMode": "cover"              },              "body": {                  "type": "box",                  "layout": "vertical",                  "contents": [                  {                      "type": "text",                      "text": "現購商品",                      "weight": "bold",                      "size": "xl",                      "align": "center"                  },                  {                      "type": "text",                      "text": "※商品已在店面。",                      "wrap": True,                      "color": "#3b5a5f",                      "size": "md",                      "flex": 5,                      "margin": "md",                      "weight": "bold"                  },                  {                      "type": "text",                      "text": "※隨買快速領取。",                      "wrap": True,                      "color": "#3b5a5f",                      "size": "md",                      "flex": 5,                      "margin": "sm",                      "weight": "bold"                  },                  {                      "type": "text",                      "text": "※方便又便宜。",                      "wrap": True,                      "color": "#3b5a5f",                      "size": "md",                      "flex": 5,                      "margin": "sm",                      "weight": "bold"                  }                  ]              },              "footer": {                  "type": "box",                  "layout": "vertical",                  "spacing": "md",                  "contents": [                  {                      "type": "button",                      "height": "sm",                      "action": {                      "type": "message",                      "label": "現購商品列表",                      "text": "【現購商品】列表"                      },                      "color": "#1a9879",                      "style": "primary"                  }                  ],                  "flex": 0              }              } | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 編號 | 3-1 | 程式名稱 | product\_preorder.py |
| 目的 | 讓消費者購買預購商品 | | |
| 部分程式碼 | | | |
| #-------------------預購商品---------------------------  def Order\_preorder(errormsg):      user\_id = lineboterp.user\_id      user\_state = lineboterp.user\_state      product\_id = lineboterp.product[user\_id+'product\_id']      product = lineboterp.product[user\_id+'product']      storage\_multiple = lineboterp.storage      product\_order\_preorder = lineboterp.product\_order\_preorder      product\_order\_preorder[user\_id] = '預購'      #Quick Reply 按鈕數量範圍      quantity\_option = []      multiple = multiplesearch(product\_id)      storage\_multiple[user\_id+'multiple'] = multiple      unit = unitsearch(product\_id)      for i in range(10):          if unit != '無':              if (i+1) % multiple == 0:                  quantity\_option.append(QuickReplyButton(action=MessageAction(label=str(i+1)+unit, text=str(i+1))))          else:              if (i+1) % multiple == 0:                  quantity\_option.append(QuickReplyButton(action=MessageAction(label=str(i+1), text=str(i+1))))  #-----------------------------------------------------        user\_state[user\_id] = 'preorder'#從user\_state轉換預購狀態      # 建立畫面及Quick Reply 按鈕      quickreply = QuickReply(items=quantity\_option)      Order\_preorder\_text = Order\_buynow\_preorder\_screen(product\_order\_preorder[user\_id],product\_id,product,quickreply,errormsg)      return Order\_preorder\_text | | | |

第十章

10-1 測試計畫

以使用者的角度測試各個功能是否能正常運作，而測試方向主要為以下所列出的測試清單。此外，由於功能種類繁多，所以目的相近者只擇一測試以及測試重點功能：

消費者：

1. 下訂商品
2. 檢視商品種類：確認消費者可以選取特訂一種商品。
3. 執行購物行為：確認消費者找到想要的商品後可以選擇直接下訂或放進購物車。
4. 操作訂單流程：確認消費者如果選擇直接下訂，填寫購買數量和電話後能完成訂單。
5. 翻頁:確認商品超過一頁顯示數量後，能藉由翻頁瀏覽更多商品
6. 新增用戶資料:確認如果使用者是第一次購買商品，可以新增電話
7. 記憶用戶資料:確認如果使用者不是第一次購買商品，系統會自動帶入他的資料。
8. 訂單查詢-查詢未取訂單
9. 檢視訂單種類：確認使用者能選擇查看未取訂單、預購訂單和歷史訂單
10. 查詢訂單內容：查看近期有的訂單
11. 訂單詳細資訊：查看單一一筆訂單的詳細內容

管理者：

1. 商品管理-新增上架-新廠商：
2. 檢視商品管理動作：確認管理者能選擇[查詢/修改/下架]、[停售及截止商品列表]和[新增上架]
3. 選擇廠商屬性: 確認管理者能選擇新、舊廠商
4. 功能沒反應
5. 庫存管理-新增及快速進貨商品-快速進貨：管理者能填入商品的庫存資料
6. 檢視庫存管理動作：確認管理者能選擇[新增及快速進貨商品]、[查詢商品庫存]和[進貨商品狀態查詢]中的[新增及快速進貨商品]查詢
7. 選擇功能-快速進貨：確認管理者能選擇庫存類別[新增]和[快速進貨]
8. 選擇查詢商品方式-依類別：確認管理者能選擇查詢商品方式[依類別]和[依廠商]
9. 選擇商品類別-冷凍食品：確認管理者能從各商品種類中擇一
10. 進貨流程：確認管理者能進入進貨流程
11. 顧客取貨-電話後三碼：

確認管理者能透過[顧客電話]或[訂單編號]中的[顧客電話後三碼]找到訂單並完成購買

1. 報表管理-成本：

確認管理者能選擇[成本]、[利潤]和[熱門商品]中的[成本]，然後管理者能選擇該報表的時間範圍，最終顯示符合選擇的報表

10-2 測試個案與測試結果

|  |  |
| --- | --- |
| 功能名稱 | 下訂現購商品 |
| 測試流程 | 1. 點選團購商品 2. 點選現購商品 3. 點選”下一頁” 4. 選擇一件商品 5. 立即購買該商品 6. 手機鍵盤輸入數量 7. 手機鍵盤輸入電話，完成下訂。 8. 再做一次下定流程，確認系統有記憶使用者資料 |
| 預期成果 | 正常跑完流程並且無系統當機 |
| 執行成果 | 正常跑完流程並且無系統當機 |
| 後續處理 | 因正常執行，無後續處理 |

|  |  |
| --- | --- |
| 功能名稱 | 訂單查詢-查詢未取訂單 |
| 測試流程 | 1. 點選訂單/購物車查詢 2. 點選訂單查詢 3. 點選未取訂單 4. 點選查詢未取訂單 5. 顯示最近100筆訂單 6. 點選單一一筆訂單 7. 顯示訂單詳細內容 |
| 預期成果 | 正常跑完流程並且無系統當機 |
| 執行成果 | 正常跑完流程並且無系統當機 |
| 後續處理 | 無 |

|  |  |
| --- | --- |
| 功能名稱 | 商品管理-新增上架-新廠商 |
| 測試流程 | 1. 點選商品管理 2. 點選新增商品 3. 點選新廠商 4. 進入填寫流程 |
| 預期成果 | 正常跑完流程並且無系統當機 |
| 執行成果 |  |
| 後續處理 | 無 |

|  |  |
| --- | --- |
| 功能名稱 | 庫存管理-新增及快速進貨商品-二次進貨 |
| 測試流程 | 1. 點選庫存管理 2. 點選新增進貨/二次進貨 3. 點選二次進貨 4. 點選依類別 5. 選擇任一類別 6. 點選任一商品 7. 進入進貨流程 |
| 預期成果 | 正常跑完流程並且無系統當機 |
| 執行成果 | 如果進貨資料和過往相同，能正常跑完流程並且無系統當機。但當二次進貨的商品價格發生了變動，會因為仍是相同商品ID的緣故所以無法顯示兩種商品價格。 |
| 後續處理 | 屬於執行邏輯上的問題 |

|  |  |
| --- | --- |
| 功能名稱 | 顧客取貨 |
| 測試流程 | 1. 點選顧客取貨 2. 點選輸入電話後三碼 |
| 預期成果 | 正常跑完流程並且無系統當機 |
| 執行成果 | 正常跑完流程並且無系統當機 |
| 後續處理 | 無 |

|  |  |
| --- | --- |
| 功能名稱 | 報表管理-報表-成本-月報表 |
| 測試流程 | 1. 點選報表管理 2. 點選報表 3. 點選成本 4. 點選成本月報表 5. 選擇年分 6. 選擇月份 7. 顯示報表 |
| 預期成果 | 正常跑完流程並且無系統當機 |
| 執行成果 | 正常跑完流程並且無系統當機 |
| 後續處理 | 無 |