

CSIE4105 Database Systems

創意 T-shirt 交易平台 (Creative T-shirt Trading Platform)

Final Report

系統需求規格書

Software Requirements Specification (SRS)

Version: 4.25

第十一組

姓名	學號	E-mail
梁皓鈞	104360098	danielleungdaniel@gmail.com
蔡一玄	104590024	dreaming34523@gmail.com
洪晟毅	104590048	cyku@cyku.tw
陳志芳	104590027	mmmmm123306@gmail.com
李宇傑	104590452	whitepaper939@gmail.com

**Department of Computer Science & Information Engineering
National Taipei University of Technology**

12/31/2017

目錄 (Table of Contents)

Section 1 簡介 (Introduction)	1
1.1 目的 (Purpose)	1
1.2 系統名稱 (Identification)	1
1.3 概觀 (Overview)	1
1.4 符號描述 (Notation Description)	2
Section 2 系統(System)	3
2.1 系統描述 (System Description)	3
2.1.1 系統架構圖 (System Context Diagram)	4
2.2 操作概念 (Operational Concepts)	5
2.3 設計限制 (Design, Data, and Implementation Constrains)	5
2.4 技術限制 (Technological Limitations)	6
2.5 介面需求 (Interface Requirements)	6
2.5.1 使用者介面需求 (User Interfaces Requirements)	6
2.5.2 外部介面需求 (External Interface Requirements).....	6
2.5.3 內部介面需求 (Internal Interface Requirements)	6
2.6 功能性需求 (Functional Requirements)	6
2.7 非功能性需求 (Non-Functional Requirements)	7
2.7.1 效能需求 (Performance Requirements)	7
2.7.2 測試需求 (Test Requirements)	7
2.7.3 安全需求 (Security Requirements)	7
2.8 其他需求 (Other Requirements)	8
2.8.1 環境需求 (Environmental Requirement)	8
2.8.2 安裝需求 (Installation Requirement)	8
Section 3 資料庫概念設計(Conceptual Design of the Database)	9
3.1 Entity-Relationship (ER) Model	9
Section 4 邏輯資料庫綱要(Logical Database Schema)	10
4.1 Schema of the Database	10
4.2 Constraints and Details of Schema	11
4.3 SQL Statements Used to Construct the Schema	14
4.4 The implementation of tables in target DBMS	18
4.5 Expectation of the possible database operations	19
Section 5 The Use of the System	20
5.1 System Installation Description	20
5.2 The Use of the System	22
Section 6 Functional Dependencies and Database Normalization	26
6.1 Functional Dependencies	26

6.2 Database Normalization	28
6.3 SQL Statements for Constructing the Normalized Table	28
Section 7 Additional queries and views	29
7.1 Database Queries	29
Section 8 Suggestions on Database Tuning	32
8.1 Database Design and Queries	32
Section 9 Conclusions, Future Work and Contribution	33
9.1 Conclusions	33
9.1.1 梁皓鈞	33
9.1.2 蔡一玄	33
9.1.3 洪晟毅	34
9.1.4 陳志芳	34
9.1.5 李宇傑	34
9.2 Future Works	35
9.3 Contribution	36
Glossary	37
References	38
Appendix	39
附件 A	39
附件 B GitHub Repository	40

Section 1 簡介 (Introduction)

1.1 目的 (Purpose)

此專案名稱為：「創意 T-shirt 交易平台 (Creative T-shirt Trading Platform)」，是一個可在線上購買他人所設計的 T-shirt 的平台，任何人亦可自行設計並上傳販售，平台上相同類型商品均為同等價格。會員無需擔心存貨、物流、金流、營運等細節，可專心於創意的設計，售出商品的部分收入將歸屬於其設計者的會員，平台營運方僅會抽取部分利潤作為營運費用，因此任何人均可在無成本的情況下販售自行設計的 T-shirt。

1.2 系統名稱 (Identification)

本專案建置範圍涵蓋下列各系統及其所屬子系統：

- 會員系統 (Membership, **MS**)
- 會員管理系統 (Membership Management System, **MMS**)
- 交易系統 (Trading System, **TS**)
 - ◆ 購物車子系統 (Shopping Cart Subsystem, **SCSS**)
 - ◆ 物流子系統 (Logistics Subsystem, **LSS**)
- 訂單管理系統 (Order Management System, **OMS**)
- 商品瀏覽系統 (Order Management System, **OMS**)
- 商品管理系統 (Goods Management System, **GMS**)
 - ◆ 促銷管理子系統 (Promotional Management Subsystem, **PMSS**)
- 商品評價系統 (Goods Rating System, **GRS**)
- 報告管理系統 (Report Management System, **RMS**)
- 推薦系統 (Recommender System, **RS**)
- 員工管理系統 (Staff Management System, **SMS**)
- 資料庫系統 (Database Management System, **DBMS**)
- 快取資料庫系統 (Cache Database System, **CDBS**)

1.3 概觀 (Overview)

操作系統的角色身分大致可區分為顧客 (Customer Member)、創作者 (Designer Member)、員工 (Staff)、管理員 (Administrator)，其中顧客與創作者皆屬於會員 (Member)，會員成功註冊後會立即擁有此兩種身分。

創作者上傳設計，由顧客挑選並購買，交易正式完成後，顧客再進行評價、留言，即完成一次交易的循環。過程中由員工進行必要審核、監督，以維持交易的公平及系統的正常運作，管理員則是唯一可監督所有員工的身分，作為維護系統的最終手段。

1.4 符號描述 (Notation Description)

其中 xxx 代表流水號，流水號位數與 x 的數量相同。

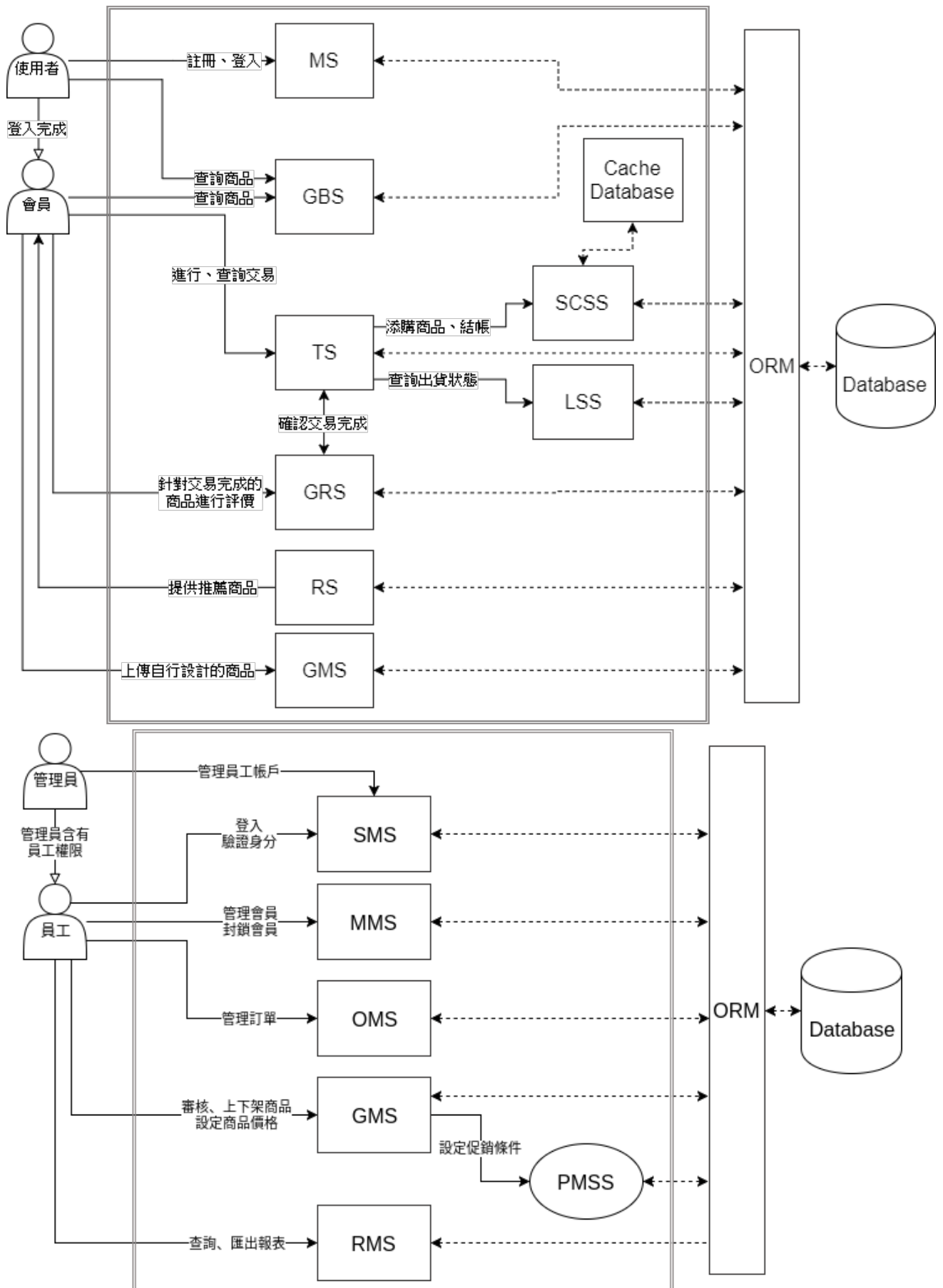
符號	說明
DIC-xxx	設計限制相關需求所使用的標記。
TL-xxx	技術限制相關需求所使用的標記。
IR-UI-xxx	介面需求、使用者介面所使用的標記。
IR-EI-xxx	介面需求、外部介面所使用的標記。
IR-II-xxx	介面需求、內部介面所使用的標記。
FR-xxx	功能性需求所使用的標記。
NF-PR-xxx	非功能性、效能需求所使用的標記。
NF-TR-xxx	非功能性、測試需求所使用的標記。
NF-SR-xxx	非功能性、安全需求所使用的標記。
OR-ENV-xxx	其他、環境需求所使用的標記。
OR-INS-xxx	其他、安裝需求所使用的標記。

Section 2 系統(System)

2.1 系統描述 (System Description)

系統名稱	縮寫	描述
會員系統	MS	提供使用者註冊會員、登入等服務。包含協助其他系統驗證使用者身份。
會員管理系統	MMS	提供員工、系統管理員進行會員管理之相關功能。
交易系統	TS	提供會員購買商品等交易服務，其下包含購物車及物流子系統。
購物車子系統	SCSS	此系統歸屬於交易系統 (TS)，記錄會員欲購買之所有商品，並提供結帳前最終確認才可正式送出訂單。
物流子系統	LSS	此系統歸屬於交易系統 (TS)，提供各訂單的物流狀態資訊，讓會員結帳後可持續追蹤商品的出貨狀態。
訂單管理系統	OMS	提供員工對所有交易訂單進行狀態管理、記錄查詢等功能。
商品瀏覽系統	GBS	提供給會員瀏覽所有可購買商品，並包含依據篩選條件查詢之功能，可讓會員快速尋覓到所需的商品。
商品管理系統	GMS	提供員工針對商品進行上、下架管理等功能，其下包含促銷管理子系統。
促銷管理子系統	PMSS	此系統歸屬於商品管理系統 (GMS)，針對商品的促銷內容、價格等設定，並可提供資訊給商品管理系統 (GMS)、交易系統 (TS) 以計算訂單價格。
商品評價系統	GRS	在交易完成後，可提供會員針對某項商品交易進行公開評價、留言，以利日後其他會員於交易前作為參考。
報告管理系統	RMS	提供員工查詢、匯出銷售報表等統計數據。
推薦系統	RS	依據過往交易記錄、促銷活動等資訊，主動提供會員可能會中意的商品。
員工管理系統	SMS	提供員工身份驗證、員工帳戶管理等功能
資料庫系統	DBMS	管理、保存所有系統的實體資料，包含會員、商品、訂單資料等。此系統僅能由其他系統呼叫、溝通，不提供任何介面給任何角色。
快取資料庫系統	CDBS	提供部分系統在記憶體中存取資料，而不必存取實體資料。

2.1.1 系統架構圖 (System Context Diagram)



2.2 操作概念 (Operational Concepts)

任何瀏覽此系統的人皆可搜尋商品、查看商品外觀及金額。搜尋時亦可依照自身需求設定篩選條件、使用關鍵字搜尋，並可依據如上架日期等對商品順序進行排序，以順利獲取所需的商品資訊。

如需進一步使用功能，譬如購買商品、刊登自行設計的商品，則必須申請加入系統一般會員，亦可使用第三方帳戶 Google、Facebook 進行註冊、登入。完成註冊認證後的會員，將會立即具有兩種身分：顧客、創作者。

以下將以不同身分的角度進行操作概念的說明：

- 顧客會員角度 (Customer Member)：可以消費購買網站內所有商品，並且可使用購物車系統暫存所有中意的商品，直至最後再進行結帳動作、完成交易獲得訂單。當交易過程結束，包含確認顧客獲得商品之後，將自動開放評價系統的功能，可以讓顧客回饋對商品的滿意度，一切評價及留言將會完全公開，以供其他顧客作為日後參考。
- 創作者會員角度 (Designer Member)：任何會員均可免費刊登自行設計的商品，上傳設計圖，經過員工的確認、審核之後，會立即由系統上架至可購買商品列表之中，供所有顧客進行挑選，此外也提供統計過往交易記錄的功能，讓創作者隨時可觀察到自己商品的銷售狀況。創作者可以獲得已售出商品金額之一定比例的收入，其餘金額將由系統營運方吸收，作為營運及刊登商品的費用。
- 員工角度 (Staff)：員工的操作介面不會對外公開，且不允許外部人員申請，僅能透過管理員新增、刪除員工帳戶，員工能提供客服服務、審核欲上架商品、管理商品資料(更新、刪除)、獲取銷售報表統計數據、監督會員行為等多項功能，各樣操作項目最終目的均為維持系統正常營運。
- 管理員 (Administrator)：唯一能對員工帳戶進行更動的身分，亦含有員工的所有操作權限，雖擁有幾乎所有權限，但基本只會針對員工帳戶進行設定。

2.3 設計限制 (Design, Data, and Implementation Constrains)

需求編號	需求描述
DIC-001	採用 Python/Flask 作為網頁後端語言、框架。
DIC-002	採用 uWSGI 作為 Web Server 與 Flask 應用程式進行溝通。
DIC-003	採用 Nginx 作為 Reverse Proxy Server。
DIC-004	採用 MySQL/MariaDB 作為本系統之主要 DBMS。
DIC-005	採用 Redis 作為快取資料庫。
DIC-006	採用 RESTful 作為 API 設計的風格。

DIC-007	採用 AJAX 作為前端存取 API 之技術，並以 JSON 作為資料傳遞的格式。
---------	---

2.4 技術限制 (Technological Limitations)

需求編號	需求描述
暫無	

2.5 介面需求 (Interface Requirements)

2.5.1 使用者介面需求 (User Interfaces Requirements)

需求編號	需求描述
IR-UI-001	當服務無法正常運作時，應呈現問題回報、通知介面。

2.5.2 外部介面需求 (External Interface Requirements)

需求編號	需求描述
IR-EI-001	採用 Flask-SQLAlchemy 作為與 MySQL 溝通的 ORM 框架
IR-EI-002	採用 Flask-Redis 作為與 Redis 服務連接的介面

2.5.3 內部介面需求 (Internal Interface Requirements)

需求編號	需求描述
暫無	

2.6 功能性需求 (Functional Requirements)

需求編號	需求描述
FR-001	使用者可以瀏覽所有 T-shirt、搜尋。
FR-002	使用者可以註冊成為會員、收會員驗證電子郵件或以 Google 或 Facebook 登入。
FR-003	會員可以以創作者的身份把自己設計的 T-shirt 在平台上販賣。
FR-004	會員可以以顧客的身份在平台上購買其他創作者設計的 T-shirt。
FR-005	會員可以透過系統生成銷售統計數據。
FR-006	會員可以瀏覽、搜尋所有 T-shirt，並可將 T-shirt 加入購物車、結帳，完成交易後可以評價及留言。
FR-007	會員可以查詢購買歷史及追蹤訂單。
FR-008	會員可以更新個人資訊、聯絡資訊、個人實體店面資訊。
FR-009	員工可以管理產品資料（查詢、更新、處理產品訂單、刪除產品、更新訂

	單狀態)。
FR-010	員工可以設定促銷活動 (增加、更新、查詢、刪除、設定時限、買 X 送 Y $\forall X \forall Y \in \mathbb{N} \wedge 0 < X \wedge 0 < Y$)。
FR-011	員工可以產生各種銷售報表及統計數據。
FR-012	員工可以追蹤訂單狀態。
FR-013	管理員可以管理所有使用者資料 (名字、密碼、電郵、地址)。
FR-014	管理員可以管理所有產品的資料。
FR-015	管理員可以管理所有訂單的資料及狀態。
FR-016	管理員可以更新維護整個系統。

2.7 非功能性需求 (Non-Functional Requirements)

2.7.1 效能需求 (Performance Requirements)

需求編號	需求描述
NF-PR-001	MySQL Connection Pool 大小需充足，避免在存取資料庫產生瓶頸。
NF-PR-002	各頁面響應時間應不大於 3 秒

2.7.2 測試需求 (Test Requirements)

需求編號	需求描述
NF-TR-001	所有客戶端介面均須進行 Web UI Testing。
NF-TR-002	所有提供異步呼叫的 API 至少須通過單元測試(Unit Testing)。
NF-TR-003	所有系統均須依照需求規格進行系統測試(System Testing)。
NF-TR-004	所有系統均須通過基本的安全弱點掃描(Vulnerability Scanning)。

2.7.3 安全需求 (Security Requirements)

需求編號	需求描述
NF-SR-001	注意 CSRF，應在各重大請求中添加校驗 token，避免出現非帳戶持有人本意的操作。
NF-SR-002	注意 SQL Injection，因本專案 DBMS 採用 MySQL，應充分檢查所有 SQL 語句，防止遭偽造查詢結果。
NF-SR-003	注意 SSRF，因本專案包含內網服務(如：Redis)，需避免遭到惡意人士利用 SSRF 攻擊內網服務。
NF-SR-004	注意 Race Condition，本專案部分系統可能需在同一請求中對資料庫進行一連串複雜查詢，應妥善使用 Transaction、Locking Reads 機制避免資料產生異常。
NF-SR-005	針對資料庫使用者的權限配置進行控管，應盡可能針對各系統安排不同使

	用者，將各使用者的操作權限壓至最低要求。
--	----------------------

2.8 其他需求 (Other Requirements)

2.8.1 環境需求 (Environmental Requirement)

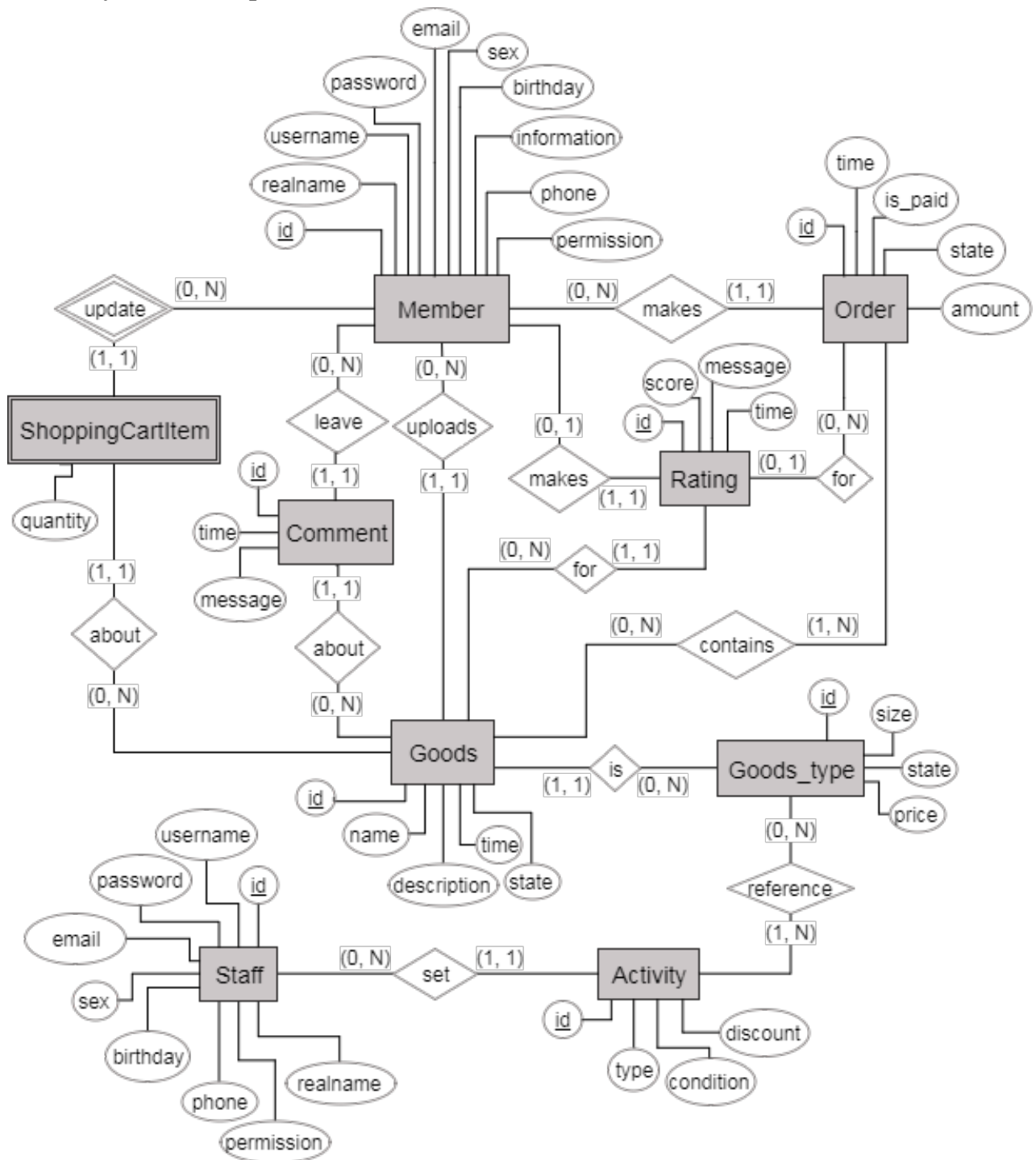
需求編號	需求描述
OR-ENV-001	伺服器使用 Linux Kernel Based 的作業系統。
OR-ENV-002	伺服器使用 CPython 3.6 以上的版本作為 Flask 應用程序的運行環境。

2.8.2 安裝需求 (Installation Requirement)

需求編號	需求描述
OR-INS-001	須建立 uwsgi.service 檔案，以作為控制 uWSGI daemon 的入口點。
OR-INS-002	須事先在資料庫中建立所須資料表及資料庫使用者。
OR-INS-003	Nginx 須支援 ngx_http_uwsgi_module。
OR-INS-004	佈署 Flask 應用程序應當使用 Python Virtual Environment。

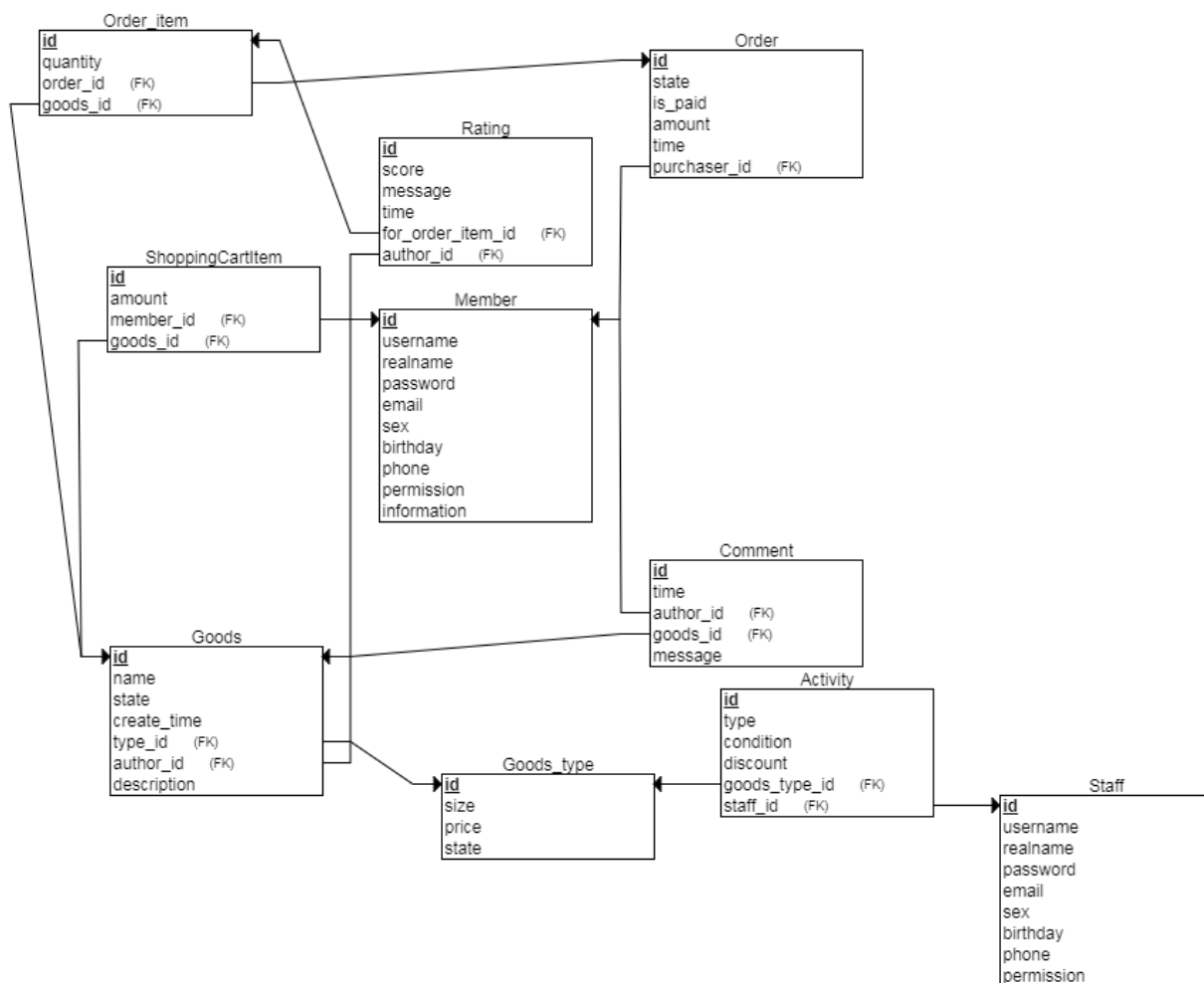
Section 3 資料庫概念設計(Conceptual Design of the Database)

3.1 Entity-Relationship (ER) Model



Section 4 邏輯資料庫綱要(Logical Database Schema)

4.1 Schema of the Database



■ 大圖見附件 A

4.2 Constraints and Details of Schema

Member				
Description: 儲存會員個人資料及相關資訊				
Attribute	Domain Type	Key	Nullable	Description
id	INT	Primary	Not Null	會員編號
username	VARCHAR(20)	Unique	Not Null	會員用戶名稱
realname	VARCHAR(50)		Not Null	會員真實姓名
password	VARCHAR(1024)		Not Null	會員密碼雜湊值
email	VARCHAR(256)		Not Null	會員電子郵件
sex	VARCHAR(10)		Not Null	會員性別：Male、Female、others
birthday	DATE		Not Null	會員生日日期
phone	VARCHAR(15)		Not Null	會員聯絡電話
permission	INT		Not Null	會員上傳作品許可,預設 0
information	VARCHAR(4096)		Not Null	會員簡介及資訊。

GoodsType				
Description: 商品類型				
Attribute	Domain Type	Key	Nullable	Description
id	INT	Primary	Not Null	商品類型編號
size	VARCHAR(5)		Not Null	商品衣服類型：S、M、L、XL、XXL
price	INT		Not Null	商品價格
state	VARCHAR(10)			商品類型狀態

Goods				
Description: 商品				
Attribute	Domain Type	Key	Nullable	Description
id	INT	Primary	Not Null	商品編號
name	VARCHAR(30)		Not Null	商品名稱
state	VARCHAR(10)		Not Null	商品狀態
create_time	TIMESTAMP		Not Null	商品上架日期
type_id	INT	Foreign	Not Null	商品類型編號
author_id	INT	Foreign	Not Null	賣家編號
description	VARCHAR(4096)			商品描述

Order				
Description: 訂單資訊				
Attribute	Domain Type	Key	Nullable	Description
id	INT	Primary	Not Null	訂單編號
state	VARCHAR(16)		Not Null	訂單狀態
is_paid	Boolean		Not Null	是否已經付款
amount	INT		Not Null	訂單總金額
time	TIMESTAMP		Not Null	下訂日期
purchaser_id	INT	Foreign	Not Null	下訂會員編號

Order_item				
Description: 訂單項目				
Attribute	Domain Type	Key	Nullable	Description
id	INT	Primary	Not Null	訂單項目編號
quantity	INT		Not Null	商品數量，預設 1
order_id	INT	Foreign	Not Null	對應訂單編號
goods_id	INT	Foreign	Not Null	對應商品編號

Rating				
Description: 評價記錄				
Attribute	Domain Type	Key	Nullable	Description
id	INT	Primary	Not Null	評價編號
score	INT		Not Null	評分，範圍 1~5
message	VARCHAR(128)			評價留言
time	TIMESTAMP		Not Null	評價日期
for_order_item_id	INT	Foreign	Not Null	評價對應的訂單項目
author_id	INT	Foreign	Not Null	評價的作者的會員編號

Comment				
Description: 商品留言				
Attribute	Domain Type	Key	Nullable	Description
id	INT	Primary	Not Null	留言編號
time	TIMESTAMP		Not Null	留言日期
author_id	INT(16)	Foreign	Not Null	留言的作者的會員編號
goods_id	INT(16)	Foreign	Not Null	對應的商品編號
message	VARCHAR(4096)		Not Null	留言內容

Staff				
Description: 員工資料				
Attribute	Domain Type	Key	Nullable	Description
id	INT	Primary	Not Null	員工編號
username	VARCHAR(20)	Unique	Not Null	員工帳戶名稱
realname	VARCHAR(50)		Not Null	員工真實姓名
password	VARCHAR(1024)		Not Null	員工密碼雜湊值
email	VARCHAR(256)		Not Null	員工電子郵件
birthday	Date		Not Null	員工生日日期
sex	VARCHAR(10)		Not Null	員工性別 $\forall \text{sex} \in \{'M', 'F'\}$
phone	VARCHAR(15)		Not Null	員工聯絡電話
permission	INT		Not Null	員工權限，預設 0

Activity				
Description: 活動				
Attribute	Domain Type	Key	Nullable	Description
id	INT	Primary	Not Null	活動編號
type	VARCHAR(10)		Not Null	活動名稱
condition	VARCHAR(1024)		Not Null	活動折扣條件 儲存 Serialize Python Object
discount	DECIMAL(3, 2)		Not Null	活動折扣
goods_type_id	INT	Foreign	Not Null	活動對應的商品類型
staff_id	INT	Foreign	Not Null	上架活動的員工編號

ShoppingCartItem				
Description: 儲存會員購物車上的商品				
Attribute	Domain Type	Key	Nullable	Description
amount	INT		Not Null	商品數量
member_id	INT	Foreign	Not Null	對應會員編號
goods_id	INT	Foreign	Not Null	對應商品編號

4.3 SQL Statements Used to Construct the Schema

```
SET foreign_key_checks = 0;
```

```
DROP TABLE IF EXISTS `db_course`.`Member`;
```

```
CREATE TABLE `db_course`.`Member` (  
  `id` INT PRIMARY KEY AUTO_INCREMENT,  
  `username` VARCHAR(20) UNIQUE NOT NULL,  
  `realname` VARCHAR(50) NOT NULL,  
  `password` VARCHAR(1024) NOT NULL,  
  `email` VARCHAR(256) NOT NULL,  
  `sex` VARCHAR(10) NOT NULL,  
  `birthday` DATE NOT NULL,  
  `phone` VARCHAR(15) NULL,  
  `permission` INT NOT NULL DEFAULT 0,  
  `information` TEXT NOT NULL  
) ENGINE=InnoDB;
```

```
DROP TABLE IF EXISTS `db_course`.`GoodsType`;
```

```
CREATE TABLE `db_course`.`GoodsType` (  
  `id` INT PRIMARY KEY AUTO_INCREMENT,  
  `size` VARCHAR(5) NOT NULL,  
  `price` INT NOT NULL,  
  `state` VARCHAR(10) NOT NULL  
) ENGINE=InnoDB;
```

```
DROP TABLE IF EXISTS `db_course`.`Goods`;
```

```
CREATE TABLE `db_course`.`Goods` (  
  `id` INT PRIMARY KEY AUTO_INCREMENT,  
  `name` VARCHAR(30) NOT NULL,  
  `state` VARCHAR(10) NOT NULL,  
  `create_time` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,  
  `type_id` INT NOT NULL,  
  `author_id` INT NOT NULL,  
  `description` TEXT NOT NULL,  
  FOREIGN KEY (`type_id`)  
    REFERENCES `db_course`.`GoodsType` (`id`),  
  FOREIGN KEY (`author_id`)  
    REFERENCES `db_course`.`Member` (`id`)
```

```
) ENGINE=InnoDB;
```

```
DROP TABLE IF EXISTS `db_course`.`Order`;
```

```
CREATE TABLE `db_course`.`Order` (  
  `id` INT PRIMARY KEY AUTO_INCREMENT,  
  `state` VARCHAR(16) NOT NULL,  
  `is_paid` TINYINT(1) NOT NULL DEFAULT 0,  
  `amount` INT NOT NULL,  
  `time` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,  
  `purchaser_id` INT NOT NULL,  
  FOREIGN KEY (`purchaser_id`)  
    REFERENCES `db_course`.`Member` (`id`)  
    ON DELETE RESTRICT  
    ON UPDATE CASCADE  
) ENGINE=InnoDB;
```

```
DROP TABLE IF EXISTS `db_course`.`OrderItem`;
```

```
CREATE TABLE `db_course`.`OrderItem` (  
  `id` INT PRIMARY KEY AUTO_INCREMENT,  
  `quantity` INT NOT NULL DEFAULT 1,  
  `order_id` INT NOT NULL,  
  `goods_id` INT NOT NULL,  
  FOREIGN KEY (`order_id`)  
    REFERENCES `db_course`.`Order` (`id`),  
  FOREIGN KEY (`goods_id`)  
    REFERENCES `db_course`.`Goods` (`id`)  
) ENGINE=InnoDB;
```

```
DROP TABLE IF EXISTS `db_course`.`Rating`;
```

```
CREATE TABLE `db_course`.`Rating` (  
  `id` INT PRIMARY KEY AUTO_INCREMENT,  
  `score` INT NOT NULL,  
  `message` VARCHAR(128) NOT NULL,  
  `time` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,  
  `for_order_item_id` INT NOT NULL,  
  `author_id` INT NOT NULL,  
  FOREIGN KEY (`for_order_item_id`)  
    REFERENCES `db_course`.`OrderItem` (`id`),  
)
```

```

        FOREIGN KEY (`author_id`)
            REFERENCES `db_course`.`Member` (`id`)
    ) ENGINE=InnoDB;

```

```

DROP TABLE IF EXISTS `db_course`.`Comment`;

```

```

CREATE TABLE `db_course`.`Comment` (
    `id` INT PRIMARY KEY AUTO_INCREMENT,
    `time` TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
    `author_id` INT NOT NULL,
    `goods_id` INT NOT NULL,
    `message` TEXT NOT NULL,
    FOREIGN KEY (`author_id`)
        REFERENCES `db_course`.`Member` (`id`),
    FOREIGN KEY (`goods_id`)
        REFERENCES `db_course`.`Goods` (`id`)
) ENGINE=InnoDB;

```

```

DROP TABLE IF EXISTS `db_course`.`Staff`;

```

```

CREATE TABLE `db_course`.`Staff` (
    `id` INT PRIMARY KEY AUTO_INCREMENT,
    `username` VARCHAR(20) UNIQUE NOT NULL,
    `realname` VARCHAR(50) NOT NULL,
    `password` VARCHAR(1024) NOT NULL,
    `email` VARCHAR(256) NOT NULL,
    `birthday` DATE NOT NULL,
    `sex` VARCHAR(10) NOT NULL,
    `phone` VARCHAR(15) NOT NULL,
    `permission` INT NOT NULL DEFAULT 0
);

```

```

DROP TABLE IF EXISTS `db_course`.`Activity`;

```

```

CREATE TABLE `db_course`.`Activity` (
    `id` INT PRIMARY KEY AUTO_INCREMENT,
    `type` VARCHAR(10) NOT NULL,
    `condition` VARCHAR(1024) NOT NULL,
    `discount` DECIMAL(3, 2) NOT NULL DEFAULT 1,
    `goods_type_id` INT NOT NULL,
    `staff_id` INT NOT NULL,

```

```

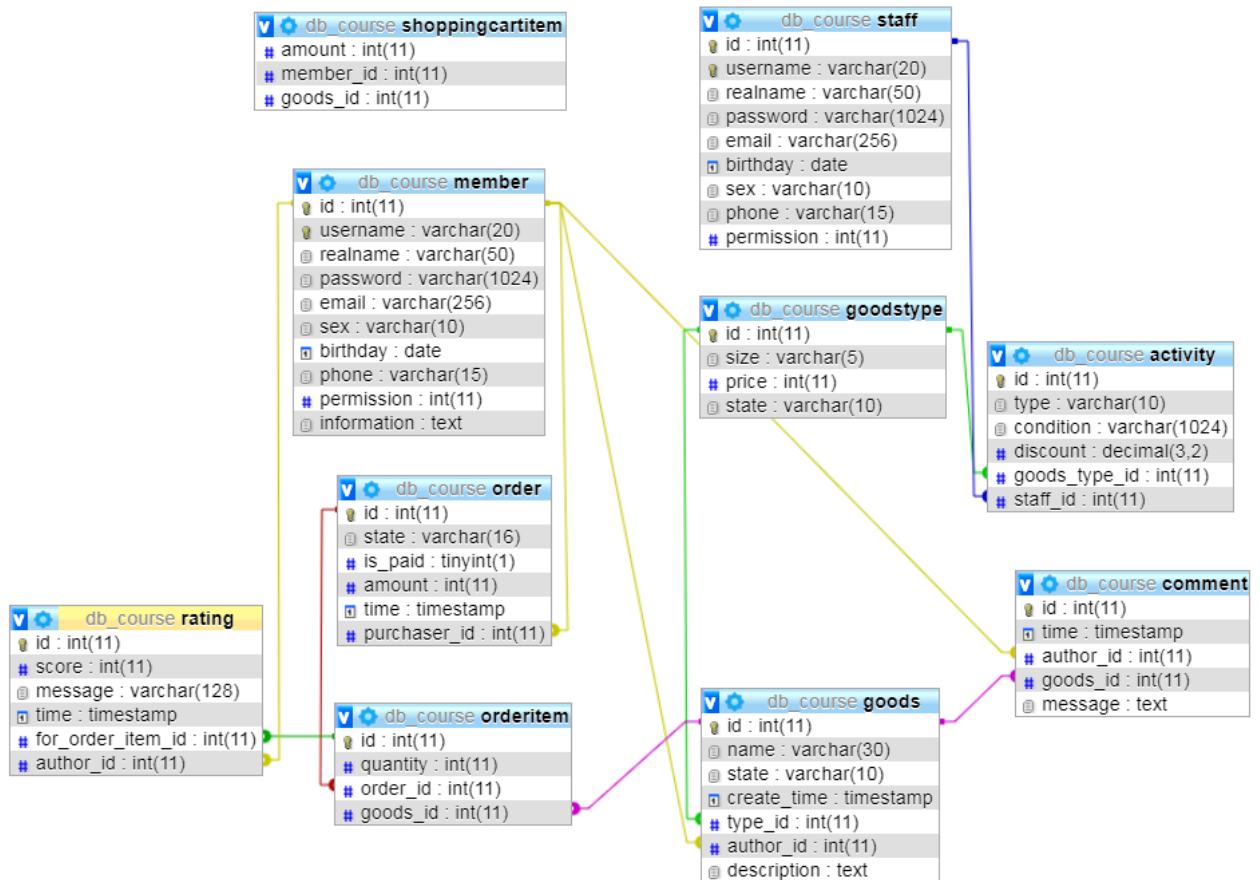
FOREIGN KEY (`goods_type_id`)
    REFERENCES `db_course`.`GoodsType` (`id`),
FOREIGN KEY (`staff_id`)
    REFERENCES `db_course`.`Staff` (`id`)
);

DROP TABLE IF EXISTS `db_course`.`ShoppingCartItem`;
CREATE TABLE `db_course`.`ShoppingCartItem` (
    `amount` INT NOT NULL,
    `member_id` INT NOT NULL,
    `goods_id` INT NOT NULL,
    FOREIGN KEY (`member_id`)
        REFERENCES `test`.`Member` (`id`),
    FOREIGN KEY (`goods_id`)
        REFERENCES `test`.`Goods` (`id`)
);

SET foreign_key_checks = 1;

```

4.4 The implementation of tables in target DBMS



4.5 Expectation of the possible database operations

Table	Expected Operations	Expected Frequency	Amount of Data	System Loading
rating	使用者評分	5	100	500I/D
order	記錄訂單	100		100I/D
order	修改	50	1000	50000Q/D 50U/D
orderitem	個別 item	10		100I/D
member	新增會員	5		5I/D
member	使用者驗證	40	200	8000Q/Day
member	更改會員資料	50	120	6000Q/D 50U/D
shoppingcartitem	使用者放在購物車的暫存	20	200	4000/Day
staff	新增員工	10		10I/D
staff	更改員工資料	15	120	1800Q/D 15U/D
goodstype	新增商品類型	5		5I/D
goodstype	修改商品類型	5	50	250Q/D 5U/D
activity	新增活動	5		5I/D
activity	修改活動	5	100	500Q/D 5U/D
goods	新增商品	25		25I/D
goods	修改商品	10	300	3000Q/D 10I/D
comment	新增留言	15		15I/D
comment	修改	7	100	700Q/D 7I/D

Section 5 The Use of the System


5.1 System Installation Description

連上 MariaDB 官方網站

<https://downloads.mariadb.org/mariadb/repositories/#mirror=ossplanet>

依照網站步驟，選擇系統平台的類型

MariaDB Foundation


MariaDB
FOUNDATION

MariaDB is free and open source software

The MariaDB database server is published as free and open source software under the General Public License version 2. You can download and use it as much as you want free of charge. *All use of the binaries from mariadb.org is at your own risk as stated in the GPLv2.* While we do our best to make the world's best database software, the MariaDB Foundation does not provide any guarantees and cannot be held liable for any issues you may encounter.

The MariaDB Foundation does not provide any help or support services if you run into troubles

Downloads


Setting up MariaDB Repositories

To generate the entries select an item from each of the boxes below. Once an item is selected in each box, your customized repository configuration will appear below.

1. Choose a Distro

- openSUSE
- Arch Linux
- Mageia
- Fedora
- CentOS
- RedHat
- Mint
- Ubuntu
- Debian

MariaDB Foundation


MariaDB
FOUNDATION

MariaDB is free and open source software

The MariaDB database server is published as free and open source software under the General Public License version 2. You can download and use it as much as you want free of charge. *All use of the binaries from mariadb.org is at your own risk as stated in the GPLv2.* While we do our best to make the world's best database software, the MariaDB Foundation does not provide any guarantees and cannot be held liable for any issues you may encounter.

Downloads

Setting up MariaDB Repositories

To generate the entries select an item from each of the boxes below. Once an item is selected in each box, your customized repository configuration will appear below.

1. Choose a Distro

- openSUSE
- Arch Linux
- Mageia
- Fedora
- CentOS
- RedHat
- Mint
- Ubuntu
- Debian

2. Choose a Release

- CentOS 7 (x86_64)
- CentOS 6 (x86_64)
- CentOS 6 (x86)

3. Choose a Version

- 10.2 [Stable]
- 10.3 [Beta]
- 10.1 [Stable]
- 10.0 [Stable]
- 5.5 [Stable]

20

選取完畢，等候頁面讀取。再依據提供的說明設定所需檔案、輸入指令，即可完成安裝。

MariaDB Foundation



MariaDB is free and open source software

The MariaDB database server is published as free and open source software under the General Public License version 2. You can download and use it as much as you want free of charge. *All use of the binaries from mariadb.org is at your own risk as stated in the GPLv2.* While we do our best to make the world's best database software, the MariaDB Foundation does not provide any guarantees and cannot be held liable for any issues you may encounter.

The MariaDB Foundation does not provide any help or support services if you run into troubles while using MariaDB. Support and guarantees are available on commercial terms from multiple [MariaDB vendors](#). There are also many resources you can use to [learn MariaDB](#) and support yourself or get peer support online.

Supported and certified

Downloads Setting up MariaDB Repositories

To generate the entries select an item from each of the boxes below. Once an item is selected in each box, your customized repository configuration will appear below.

1. Choose a Distro

- openSUSE
- Arch Linux
- Mageia
- Fedora
- **CentOS**
- RedHat
- Mint
- Ubuntu
- Debian

2. Choose a Release

- **CentOS 7 (x86_64)**
- CentOS 6 (x86_64)
- CentOS 6 (x86)

3. Choose a Version

- **10.2 [Stable]**
- 10.3 [Beta]
- 10.1 [Stable]
- 10.0 [Stable]
- 5.5 [Stable]

Here is your custom MariaDB YUM repository entry for CentOS. Copy and paste it into a file under `/etc/yum.repos.d/` (we suggest naming the file `MariaDB.repo` or something similar).

```
# MariaDB 10.2 CentOS repository list - created 2018-01-02 12:46 UTC
# http://downloads.mariadb.org/mariadb/repositories/
[mariadb]
name = MariaDB
baseurl = http://yum.mariadb.org/10.2/centos7-amd64
gpgkey=https://yum.mariadb.org/RPM-GPG-KEY-MariaDB
gpgcheck=1
```

After the file is in place, install MariaDB with:

```
sudo yum install MariaDB-server MariaDB-client
```

If you haven't already accepted the MariaDB GPG key, you will be prompted to do so. See "[Installing MariaDB with yum](#)" for detailed information.

Please see [Installing OQGraph](#) for details on additional install steps needed for that storage engine.

Here is your custom MariaDB YUM repository entry for CentOS. Copy and paste it into a file under `/etc/yum.repos.d/` (we suggest naming the file `MariaDB.repo` or something similar).

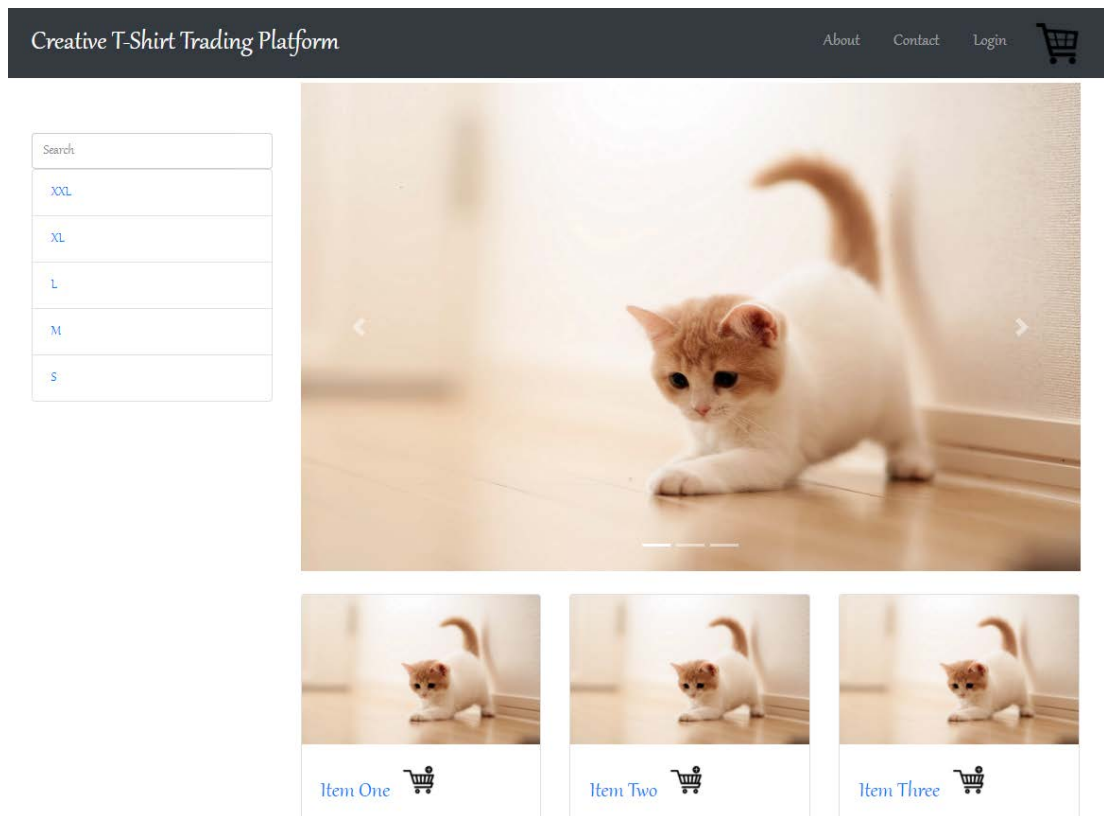
```
# MariaDB 10.2 CentOS repository list - created 2018-01-02 12:46 UTC
# http://downloads.mariadb.org/mariadb/repositories/
[mariadb]
name = MariaDB
baseurl = http://yum.mariadb.org/10.2/centos7-amd64
gpgkey=https://yum.mariadb.org/RPM-GPG-KEY-MariaDB
gpgcheck=1
```

After the file is in place, install MariaDB with:

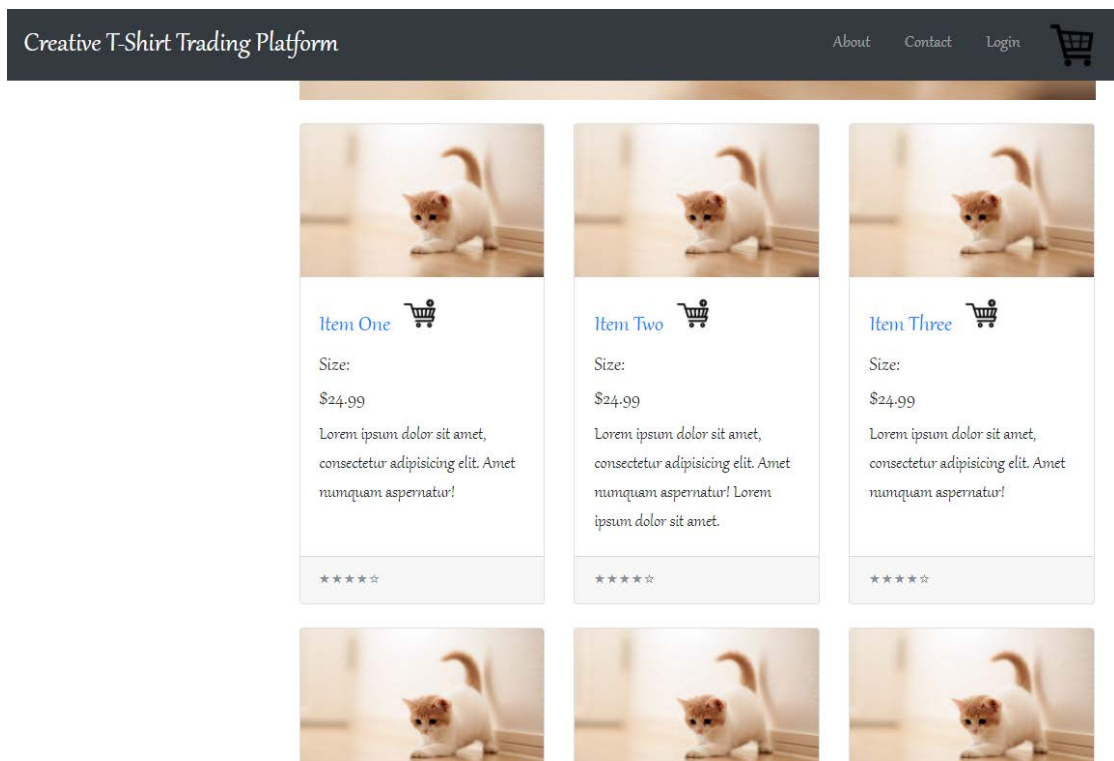
```
sudo yum install MariaDB-server MariaDB-client
```


5.2 The Use of the System

1. 首頁 (上半部)



2. 首頁 (下半部)



3. 登入及註冊頁面

Login

Username *

Password *

[Forgot Password?](#)

LOGIN

Sign Up

Name *

Username *

Password *

Email *


Phone *

REGISTER

4. 購物車頁面

Shop Cart

XXL
XL
L
M
S




Item One

Size:

\$24.99

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Amet nuncquam aspernatur!

★★★★☆




Item Two

Size:

\$24.99

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Amet nuncquam aspernatur! Lorem ipsum dolor sit amet.

★★★★☆




Item Three

Size:

\$24.99

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Amet nuncquam aspernatur!

★★★★☆




Item Four

Size:

\$24.99

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Amet nuncquam

★★★★☆




Item Five

Size:

\$24.99

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Amet nuncquam

★★★★☆



Item Six

Size:

\$24.99

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Amet nuncquam

★★★★☆

5. 會員詳細資料

Creative T-Shirt Trading Platform

會員資料 訂單記錄 登出 

會員資料

帳戶名稱

*真實姓名

*E-Mail

*性別 ☒ 男性 ☐ 女性 ☐ 其他

*生日

*電話

新密碼

確認新密碼

*舊密碼

更新資料

6. 訂單列表

Creative T-Shirt Trading Platform

會員資料 訂單記錄 登出 

首頁 / 訂單記錄

訂單編號	下單時間	費用合計	狀態
1	2018-01-01 07:01:32	399	尚未付款

Copyright © ?\$!?!%!#~ 2017

7. 選購畫面

Creative T-Shirt Trading Platform

會員資料 訂單記錄 登出

首頁 / 商品列表 / 訂購

✓全選	商品名稱	尺寸	價格	數量	
<input type="checkbox"/>	test1	M	399	<input type="text" value="1"/>	×
<input checked="" type="checkbox"/>	test2	M	399	<input type="text" value="1"/>	×
合計		<input type="text" value="399"/>			

結帳

Copyright © ?\$!?\$!%!#~ 2017

8. 確認送出訂單

Creative T-Shirt Trading Platform

會員資料 訂單記錄 登出

首頁 / 商品列表 / 訂購 / 確認訂單

商品名稱	尺寸	價格	數量
test2	M	399	<input type="text" value="1"/>
合計		<input type="text" value="399"/>	

確認無誤，送出訂單

Copyright © ?\$!?\$!%!#~ 2017

9. 訂單詳細資訊

Creative T-Shirt Trading Platform

會員資料 訂單記錄 登出

首頁 / 訂單記錄 / 訂單詳細 (編號:)

下單時間：2018-01-01 16:04:46

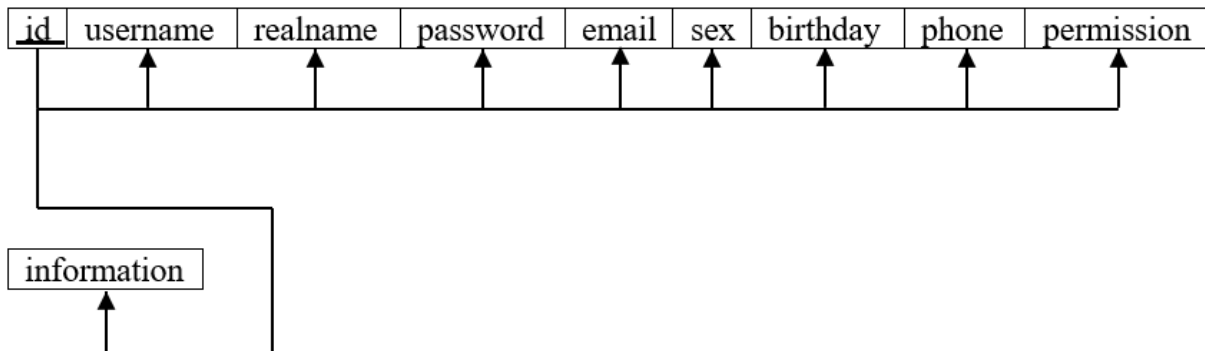
商品名稱	尺寸	價格	數量
test2	M	399	1
合計		<input type="text" value="399"/>	

Copyright © ?\$!?\$!%!#~ 2017

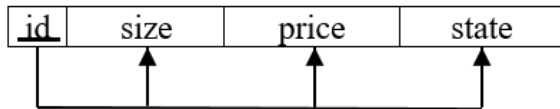
Section 6 Functional Dependencies and Database Normalization

6.1 Functional Dependencies

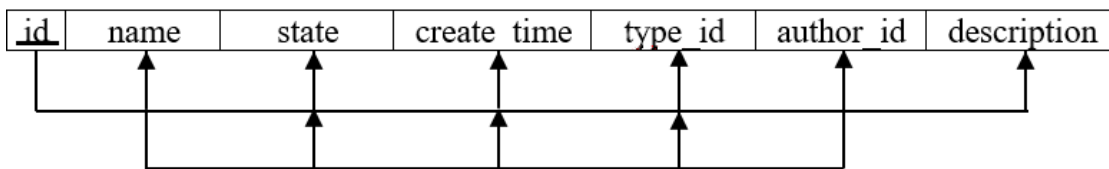
Member



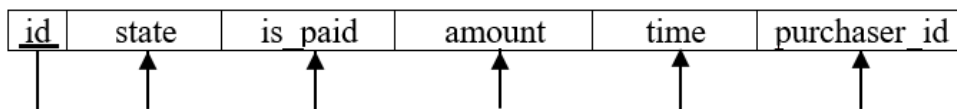
GoodsType



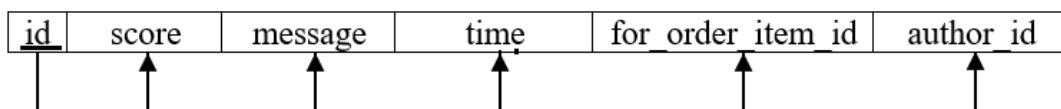
Goods



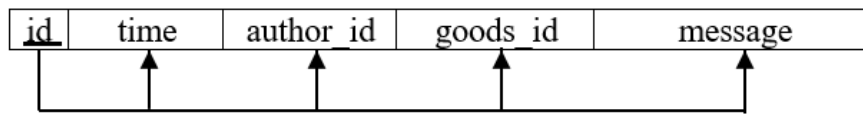
Order



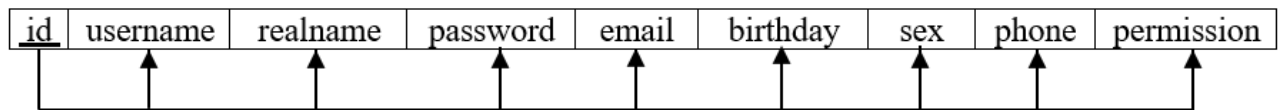
Rating



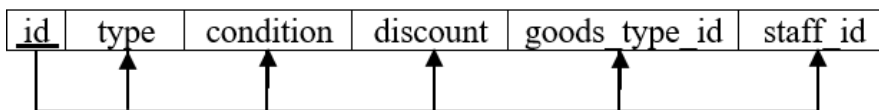
Comment



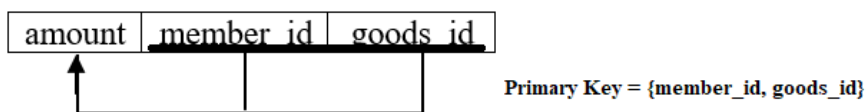
Staff



Activity



ShoppingCartItem



6.2 Database Normalization

大部份的 Relations 都是在第三個 Normal Form (NF3)。唯有 Goods 有 transitive attributes，屬於在第二個 Normal Form(NF2)。但由於在實際的資料檢索上沒有必要把他們的資料以拆開的方式讀取，以及在定義上{name, author_id}應該是其中一個 key 而實際運用上由於有 id 的存在而不需要進行分解(decomposition)。因此保持 NF2 對於運算速度上比較快而且比較實用。

6.3 SQL Statements for Constructing the Normalized Table

沒有使用到任何 SQL 進行 Normalization，全部都直接根據正確的 ER model 所製作，因此大部份 Relations 都直接以 NF3 設計。

Section 7 Additional queries and views

7.1 Database Queries

依據頁數及每頁數量列出商品 (依據建立時間遞減)

```
SELECT *
FROM Goods
ORDER BY Goods.create_time DESC
LIMIT @page, @count
```

依據關鍵字、商品類型、頁數及每頁數量篩選列出商品

```
SELECT Goods.*
FROM Goods
JOIN GoodsType ON Goods.type_id = GoodsType.id
WHERE Goods.name LIKE '%@name%'
      AND GoodsType.size = @good_type
ORDER BY Goods.create_time DESC
LIMIT @page, @count
```

列出商品詳細資料

```
SELECT *
FROM Goods
WHERE Goods.id = @goods_id
```

會員登入

```
SELECT *
FROM Member
WHERE Member.username = @username
```

取得會員詳細資料

```
SELECT *
FROM Member
WHERE Member.id = @user_id
```

會員註冊

```
INSERT INTO Member
(username, realname, password, email, sex, birthday, phone)
VALUES
(@username, @realname, @password, @email, @sex, @birthday, @phone)
```


更新會員個人資料

```
UPDATE Member SET  
(realname=@realname, password=@password, email=@email, sex=@sex, birthda  
y=@birthday, phone=@phone)
```

列出會員的所有訂單

```
SELECT `Order`.*  
FROM `Order`  
JOIN Member ON `Order`.purchaser_id = Member.id  
WHERE Member.id = @user_id
```

列出會員的訂單的詳細資料

```
SELECT `Order`.*  
FROM `Order`  
JOIN Member ON `Order`.purchaser_id = Member.id  
WHERE Member.id = @user_id AND `Order`.id = @order_id
```

新增訂單

```
INSERT INTO `Order`  
(amount, purchaser_id, is_paid)  
VALUES  
(@amount, @purchaser_id, FALSE)
```

新增訂單的項目

```
INSERT INTO `OrderItem`  
(quantity, goods_id, order_id)  
VALUES  
(@quantity, @goods_id, @order_id)
```

新增購物車項目

```
INSERT INTO `ShoppingCartItem`  
(goods_id, member_id)  
VALUES  
(@goods_id, @member_id)
```

取得各用戶的過往訂單數量

```
SELECT Member.id, Member.username, COUNT(`Order`.id)
FROM   `Order`
JOIN   Member ON `Order`.purchaser_id = Member.id
GROUP BY `Order`.purchaser_id
ORDER BY `Order`.purchaser_id ASC
```

取得購買次數前十的商品

```
SELECT Goods.*
FROM Goods
WHERE Goods.id IN (
    SELECT O.good_id
    FROM `OrderItem` AS O
    GROUP BY O.good_id
    ORDER BY COUNT(O.id) DESC
)
LIMIT 10
```

建立取得購買次數前十的商品的 View

```
CREATE VIEW TopTenGoodsView AS
SELECT Goods.*
FROM Goods
WHERE Goods.id IN (
    SELECT O.good_id
    FROM `OrderItem` AS O
    GROUP BY O.good_id
    ORDER BY COUNT(O.id) DESC
)
LIMIT 10
```

建立依據建立時間遞減排序的商品列表的 View

```
CREATE VIEW GoodsDescView AS
SELECT *
FROM   Goods
ORDER BY Goods.create_time DESC
```

Section 8 Suggestions on Database Tuning

8.1 Database Design and Queries

這次資料庫是以購物平台為主。購物平台主要以速度，效率以及支援熱銷活動的能力。鑑於這方面的系統效能需求，我們比較偏向不需要高 Normal Form 的準則。根據課堂教授所提到，Normal Form 理論上到了第 3 等級已經足夠。然而當然愈高在理論層面愈好。但我們在現實情況，在這一次資料庫中我們的表格運算上我們著重的在於效能上，因此我們在設計上沒有使用很高的 Normal Form 準則。

在設計上，為了以高效處理，我們盡可能排除了由多個 Attribute 所組成的 Primary Key 的做法，改以定義 id 作為 Primary Key。這樣的做法不單減少了多個 Attribute 運算才能找到單一筆資料的運算量，實際上在運用中大部份的資料都可以直接在同一個表格中找到。

我們在設計上，選擇了以 NF3 為準則，其中一個 Relations 雖然是 NF2，但並沒有影響。因為在購物平台運算上，該 Relations 本身就是直接整筆資料進行讀取，沒有需要分解成多個 Relations 而達到 NF3 的需求。這樣效能才會達到我們所期望的。

我們的 ER Model 在一開始設計時已經透過與老師多次討論而達到一個相對比較好設計而且合理的模型，因此在我們把 ER Model 轉換成表格時，直接達到 NF3，不需要作額外的 SQL 來做 Normalization 的動作。

在 Transaction 的部份，我們沒有特別為資料庫出錯時要進行及時處理的部份，這部份雖然不在這一次專案的要求以及需求中，但我們可以以後在這方面進行思考以及實作。

Section 9 Conclusions, Future Work and Contribution

9.1 Conclusions

9.1.1 梁皓鈞

資料庫系統對我來說是一個較為有挑戰的一門課，因為我本身實作經驗不多，在這門課中，不但要學習原理如 ER Model，資料庫設計背後的一些思維以及設計方式，還要把整個系統設計出來，工作量的確是有點大。但對於資料庫系統本身這門課而言，實作是必需的。因此我認為在這門課，我學到很多。這門課實作部份以及理論都可以讓我們將來在工作上所使用。

本次專題其實在一開始是最複雜的，因為在完全不知道一個規格書是怎樣開始的情況下，根本不知道簡介以及系統是怎樣的東西。幸好我們在多次請教教授以及請教授幫我們檢查思維上有沒有少了什麼，我們順利完成了專題。很多實作部份我仍然在學習中，幸好有組員在這方面比較有經驗，我可以一直請教他們，在過程中不斷學習。

一開始我以為資料庫系統只是在談有關於表格的事情。沒想到原來還有 ER, EER 之類的設計方式。後來還有一些 Schema 定義，Functional Dependencies, Normalization 這些方法以及理論。讓我感覺到原來資料庫完全不像是我們平時聽起來的東西，在理論層面，資料庫可以很深，甚至範圍包括到實際上硬體儲存以及資料結構。

在作業簿部份很多題目都十分有意義而且有點小困難，需要花大量時間進行思考，翻本參考書才能完成。雖然有點小困難，但作業中一些題目的確讓我們清楚了解到一些理論層面的原理。例如 ER Model 的設計，SQL 語法。這些不只是對於我們考試是一種複習的方式，更加對我們將來出去工作在資料庫設計的流程以及熟練度都有所提升。

9.1.2 蔡一玄

我覺得課程中安排這種實作型的專題是必須的，因為常常不少人會覺得理論課程很無趣，可能也不太了解所學的知識到底有沒有用途，很容易就對一門課產生一些較負面的印象。實作專題的好處在於，我們能夠將理論課程學習到的知識發揮在實際的作品上，也比較接近到公司工作或是接案子的情況。

這次專題考驗的是網站製作與資料庫整合的能力，剛好我二年級下學期課程中的專題使用到網頁前端的部分，暑假時我也自己練習網頁後端與資料庫連接，因此大部分的實作方法對我來說不會太難。不過，這次專題所使用到的資料庫數量遠多於之前的作品，因此在實作前必須謹慎規劃資料庫中實體之間的關係，以及欄位、鍵值的設定。而這方面規劃的方法，像是 ER Model、Database Schema 這些我之前從未學習過，但經過老師課堂上的教學，加上作業的練習與觀看課本習題，我漸漸有能力自己畫出整個資料庫的架構。

製作本次專題的過程中，我覺得一開始的時候是較困難的，俗話說：「萬事起頭難」，大部分的人一開始都會比較沒有方向，不知道要如何下手。不過老師將這次的專題分成四個階段，如此一來，我們能夠按照順序完成整個專題，對整個架構也比較有頭緒。最後，我覺得資料庫這個領域在我們生活中可以說是不可或缺，在學校、醫院等機構的使用更是頻繁，由此可知這方面的人才需求量絕對不會太少，因此我們若把資料庫學好，

對以後找工作應該會有幫助。

9.1.3 洪晟毅

雖然本身對小程序撰寫已有不少次數，但從未有完整從零開始撰寫需求規格書的經驗，一直以來總是在實作時遇到問題才解決，透過這一學期的資料庫課程的規劃體驗，感受到自己缺少的部分就是經驗。在整個系統設計的過程中，最初原本所構想的主意總以為是完善的，直到在後續進一步的細部規劃後，許多細節的問題才逐漸浮出，原先所設想的設計也因此需要修改。此課程的安排真的是很棒的經驗，可以嘗試統整自己曾學過的工具並練習其應用，雖然此次專案未能完美地成功運用工具，但留下了很深刻的失敗的體驗，可以期待下次在其他專案上的表現，利用此次經驗，避免重蹈覆轍。

9.1.4 陳志芳

這次的資料庫系統中，我在組中主要是負責文書整理，雖然只有做文書整理，但是由於跟寫程式的同學一起作業，所以也從中學到了一些後端的連結及應用方式，雖然現在並沒辦法實際自己將前後端連結起來，不過相信在望後自主練習的情況下能彌補這個缺失。

這堂課是我目前上過最累人、也最充實的課，同時有作業、考試、實作及專題一起完成，一開始覺得非常累人，畢竟以前的課並不會同時擁有這四項在一起，不過在每次寫作業、實作作業中都慢慢知道語法運用、資料庫操作及設計的方法，實際上相對其他課來說或許是重了些，但是會促進自己的學習速度，也增強自己找資料及運用的功力，因為之前的程式課並不會包含那麼多的東西在一起，許多東西也是老師在上課有提到，但是到了三年級，老師期許我們自己找方法、增強自己的能力，而不是當今天從學校走出去沒辦法具備自己處理事情、解決問題的能力。

在每次老師上課的過程中，一步步教導我們語法如何運用及一些做資料庫之前須要先擁有那些東西；擁有了資料庫大致上之後，先畫出 ER model 或 EER model 接著將他們轉成 schema 畫出關聯性之後再接著做 tables 跟 tuples 的輸入，接著再用語法將想要的資料丟進或取出做操作。

9.1.5 李宇傑

這學期資料庫，前半大多著重在資料庫的設計上，後半則開始 SQL 的實作，而我自己針對每章的課程在試著實作後，能確實地感覺到進步。但資料庫的需要做的事情較其他課程較多，相對的也比較花時間。雖然結果來說是好的，但不免覺得可惜，覺得自己要是可以想的再快一點，做的再快一點就能節省時間了，而資料庫 project，我個人覺得自己練習到很多以前自己所不熟悉的事情，從一開始畫 schema，到開始寫 SQL，在花了時間後都得到了不錯的練習成果。同組的組員的實行力更是讓我感到佩服與訝異，在幾次的 report 間。比較了自己寫得和其他人完成的東西後，都會讓自己覺得可以再做得更好，資料庫實作的部分算是我覺得學到最多東西的部分，由於自己對 python 並不是這麼熟悉，因此可以觀察並學習其他組員的成果是很好的學習方式。總結來說，這學期 project

從討論到實作的部分，都是很好的過程，希望報告能有個好的結束。

9.2 Future Works

這一次基本上我們完成了所有我們所提出以及課堂所要求的部份。不過在實際上如果要在商業中使用，必需要考慮更加多現實情況，例如交易出錯之類的特殊問題。這方面碰不是本資料庫系統課程中的要求，但在這課程之後，為了學習更多有關資料庫在實際應用的注意事項，我們可以從這方面進行學習以及改善。

9.3 Contribution

姓名	貢獻比例	分工內容
梁皓鈞	25%	參與討論，草稿專案整體(簡介、系統)，於各 Phase 與教授確認專案的文件細節及分類，草稿 ER Model，估計資料庫流量，檢查 Functional Dependencies 及確定 Normal Form，總結與心得
蔡一玄	25%	參與討論，草稿專案整體(簡介、系統)，草稿 ER Model 及 Schema，資料庫前端製作，整合整個資料庫系統專案平台，總結與心得
洪晟毅	26%	參與討論，草稿專案整體(簡介、系統)，整合專案整體，草稿 ER Model 及 Schema，與教授確認系統所有需求，整合 ER Model 及 Schema，資料庫後端製作，整合整個資料庫系統專案平台，總結與心得，課堂報告準備
陳志芳	18%	參與討論，草稿專案整體(簡介、系統)，草稿 ER Model，文書格式整理，總結與心得
李宇傑	6%	參與討論，總結與心得
總比例	100%	

Glossary

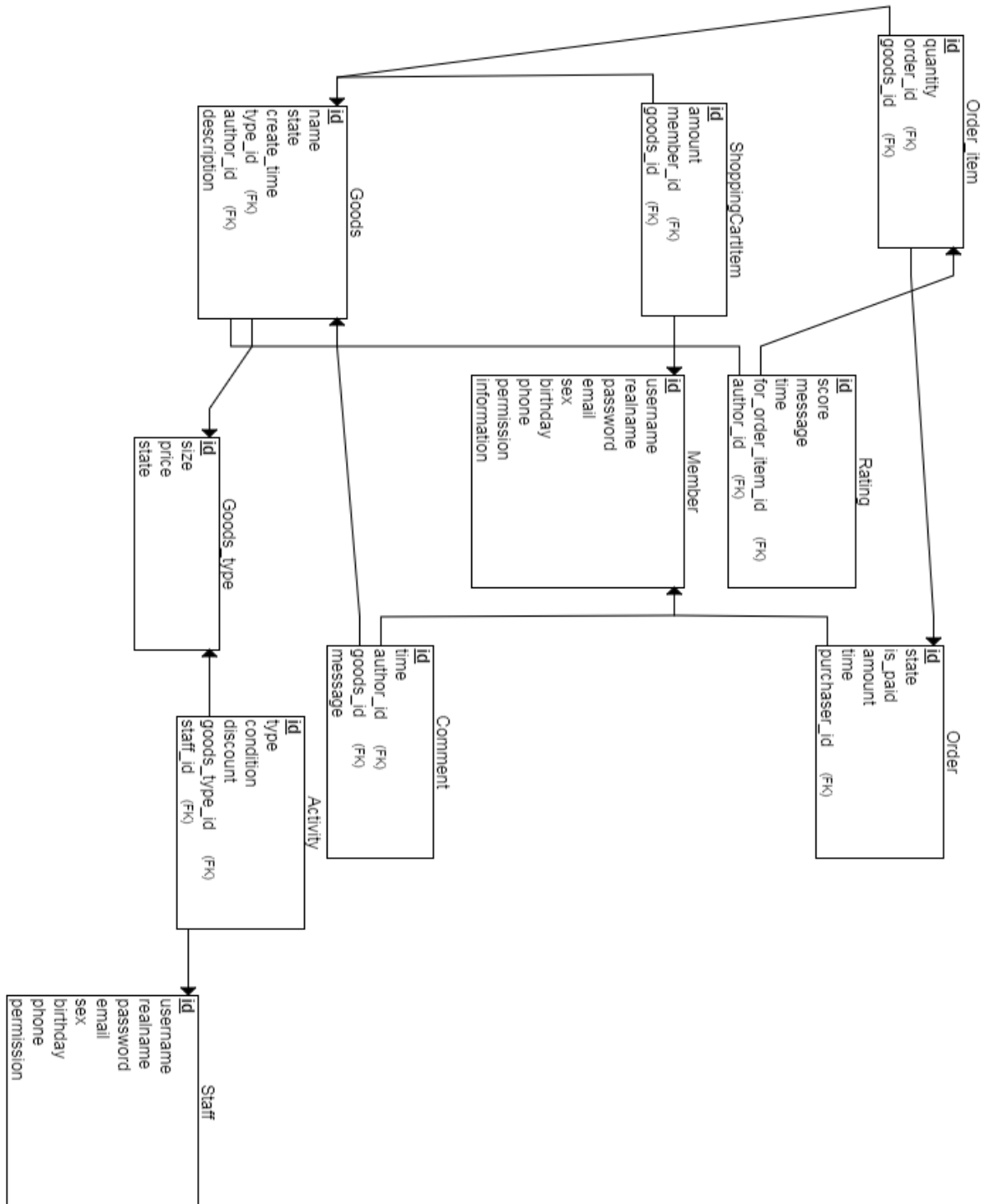
詞彙	說明
Python	物件導向、直譯式的程式語言。
CPython	為 Python 官方的主要直譯器，由 C 語言撰寫而成。
Flask	由 Python 語言撰寫的輕量級網頁應用程式框架。
WSGI	針對 Python 所定義，作為 Web Server 及 Web Application 之間溝通的通用界面協議，是基於 CGI 的延伸。
uWSGI	可視為一個輕量化的 Web Server，支援 WSGI 協議。
NGINX	可作為 Web Server、Reverse Proxy Server 等服務的軟體。
Redis	基於記憶體體的 Key-Value 儲存資料庫。
RESTful	形容符合 REST 軟體設計風格的架構。
AJAX	位於瀏覽器端之非同步傳輸的網頁開發技術。
JSON	一種極輕量的資料交換格式。
ORM	將資料庫資料映射為物件導向語言中的物件的技術。
SQLAlchemy	在 Python 語言下的 ORM 工具，其中 Flask-SQLAlchemy 為應用於 Flask 上的擴展工具。
CSRF	資訊安全的專有名詞，一種挾持用戶進行非本意操作的攻擊手法。
SQL Injection	資訊安全的專有名詞，透過在輸入的字串中注入惡意的 SQL 指令進行攻擊的手法。
SSRF	資訊安全的專有名詞，透過服務端發起的請求進行攻擊的手法。
Race Condition	資訊安全的專有名詞，同時進行兩個以上相同或不同動作時，可能產生意外狀況導致服務無法正常運行。

References

- MariaDB Documentation <https://mariadb.com/kb/en/library/documentation/>
- Redis Documentation <https://redis.io/documentation>
- Flask Documentation <http://flask.pocoo.org/docs/0.12/>
- Flask-SQLAlchemy Documentation <http://flask-sqlalchemy.pocoo.org/2.3/>
- Flask-Login Documentation <https://flask-login.readthedocs.io/en/latest/>
- Flask-Bcrypt Documentation <https://flask-bcrypt.readthedocs.io/en/latest/>
- Flask-Redis Documentation <https://github.com/andymccurdy/redis-py>
- WTForms Documentation <http://wtforms.readthedocs.io/en/latest/>
- Jinja2 Documentation <http://jinja.pocoo.org/>
- uWSGI Documentation <http://uwsgi-docs.readthedocs.io/en/latest/>

Appendix

附件 A



附件 B GitHub Repository

[https://github.com/CykuTW/DB T-shirt trading system](https://github.com/CykuTW/DB_T-shirt_trading_system)

/doc 資料夾中是整個專案至今所有的文件以及簡報

/readme.md 中是有關專案的使用方法

整個專案沒有執行程式，直接按照 readme.md 使用即可