Xi'an Jiaotong-Liverpool University
西交利物浦大学

SCHOOL OF ADVANCED TECHNOLOGY

CAN302 Assignment

Student Name	:	Rui Qiu
Student ID	:	1930209

DB Design Report

For this project, which a simple ecommerce admin site is built, following functions are required to be implemented:

- **Login**:
  - view/update my profile
- **Category**:
  - manage products categories: *create* new, *update*, *delete* categories
- **Products**:
  - *view/update/insert* new products/*remove* out stock products
- **Users**:
  - manage users means can alter user's info including *payment methods*, *shipping address*, *order history*
- **Orders**:
  - *view/update* orders

In order to implement these functions, a detailed, granular and normalized database design is preferred. The following sections will describe the design of the database.

For the purpose of have the database in BCNF, there are 22 tables in total. The main tables are **user**, **product**, **address**, **order**. While the other tables are used to store the information of the main tables and used to implement the *many-to-many* relationship by table junctions.

In this report, only the main tables are referred to, and the other tables are not mentioned as they are implied by the relationship between the main tables. The detailed design can be found in attached sql schema, and ER diagrams.

Three functional groups are considered when design the database:

User Management

The users of the system are divided into two groups: **admin** and **customer**. The admin group is the one who can manage the system, while the customer group is the one who can buy products from the system.

The admin group are simple enough, has login credentials, personal information and a list of services such as product, category, user and order management. Which is a lot of functions to be implemented but not too complicated on the database structure.

The customer group, on the other hand, is more complicated as they are the central part of a ecommerce system. Besides the login credentials and personal information, they also have a list of **payment** methods, shipping **address** and **order** history.

For more modularization, **address** itself is a table with *many-to-many* relationship with customer, in which a customer can have multiple addresses and an address can be used by multiple customers. Since the address is standalone, it can also be used by other tables such as the product inventory and shipping address.

Payment details, is an *one-to-many* relationship with customer, as a customer may have multiple payment methods but a payment method can only be used by one customer.

Product Management

Products, as the core of the ecommerce system, are the most complicated part of the database design. During the shooping process, a product will be moving through at least three different parties: the *vendor*, the *logistic company* and the *customer*. Each party sees the product in a different way. To the *vendor* and *customer*, each product has a price, and can be searched, filtered or sorted by categories, keywords, price and tags.

Products are stored in *inventory*, which is like a warehouse, it has an address, and associated management staff. A product can be stored in multiple inventories, and an inventory can store multiple products. This design would simplify the process of moving products from one party to another.

Category

A product would fall under one category, which might be a sub-category of another category. Each category would have a name, description, possibly a parent category and a list of sub-categories. The category table is a *one-to-many* relationship with itself, in which a category can have multiple sub-categories and a sub-category can only have one parent category. The category table is also a *one-to-many* relationship with product, in which a product can be under exactly one category and a category can have multiple products.

However, to simply the user experience, a product would also have a list of **tags**, which in turn can be looked up by associated **keywords**. A product can have multiple tags, and a tag can be associated with multiple products. A keyword can be associated with multiple tags, and a tag can be associated with multiple keywords. By searching one or more keywords, the system can quickly find products that the user is interested in.

Shopping Process

The shopping process can be divided into two parts depending on whether it is completed or not. The first parts involves customer browsing the products and adding products to the shopping cart. The second part involves customer checking out the products in the shopping cart and vendor shipping the products to the customer.

Shopping Cart

One user can have at most one shopping cart at a time, although the content of the shopping cart are changed frequently. Each entry in the shopping cart is a product referenced by its id, and its quantity and price. The shopping cart would also have a total price, which is the sum of the price of all products in the shopping cart. The shopping cart is a *one-to-many* relationship with product, in which a product can be in at most one shopping cart and a shopping cart can have multiple products. A shopping cart has a *one-to-one* relationship with the customer.

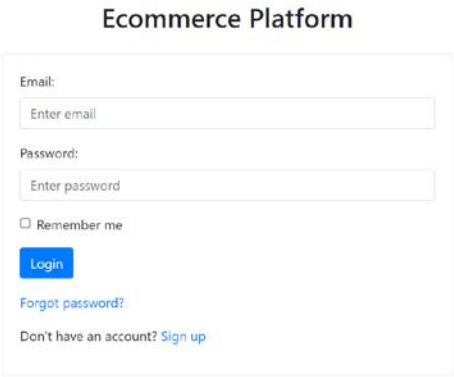
Order

When user checks out the shopping cart, the selected products are evicted from the cart and stored in the **order** table. The order table is similar to the shopping cart table, it would two prices, *amout due* and *amount paied*, with associated payment detail. Also the order has *one-to-many* relationship with the customer that a customer can have multiple orders and an order can only be placed by one customer.

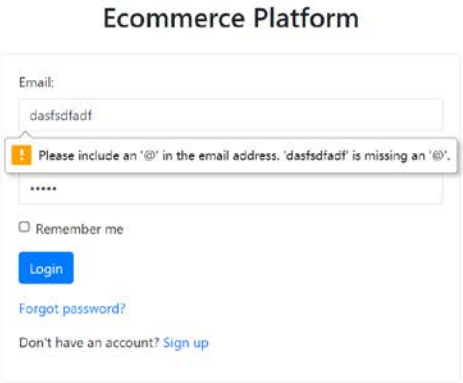
Products within an order need to be shipped to the customer, which contains information of which inventory this shipment is from, and the customer's address that the product will be shipped to. In an order, there might be multiple shipments as different products in the same order might be stored and shipped from different inventories. The shipment would also have details of logistics company and the tracking number.

Appendix

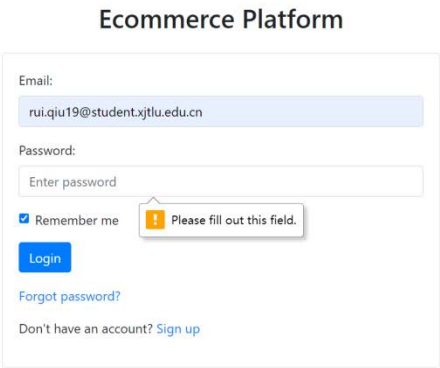
UI Design



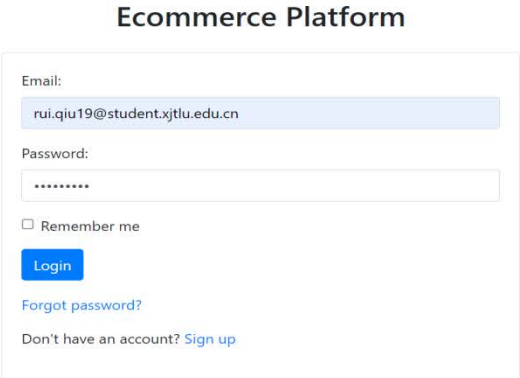
Step1. Login Page



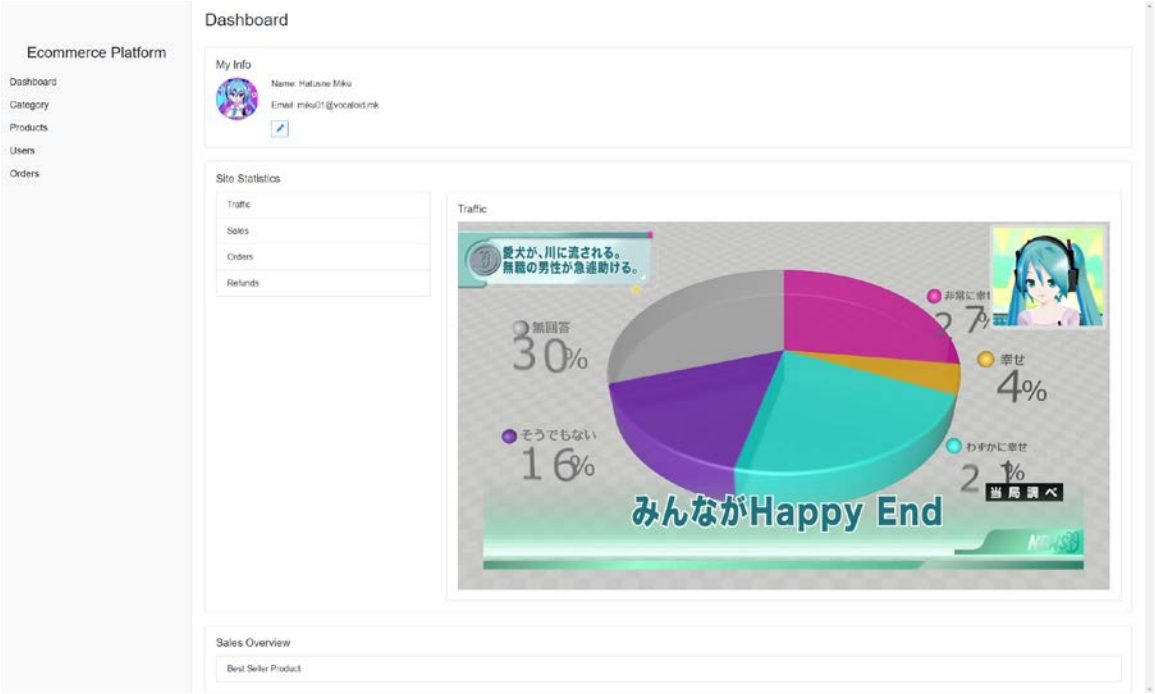
Step2. An incorrect format would not be allowed



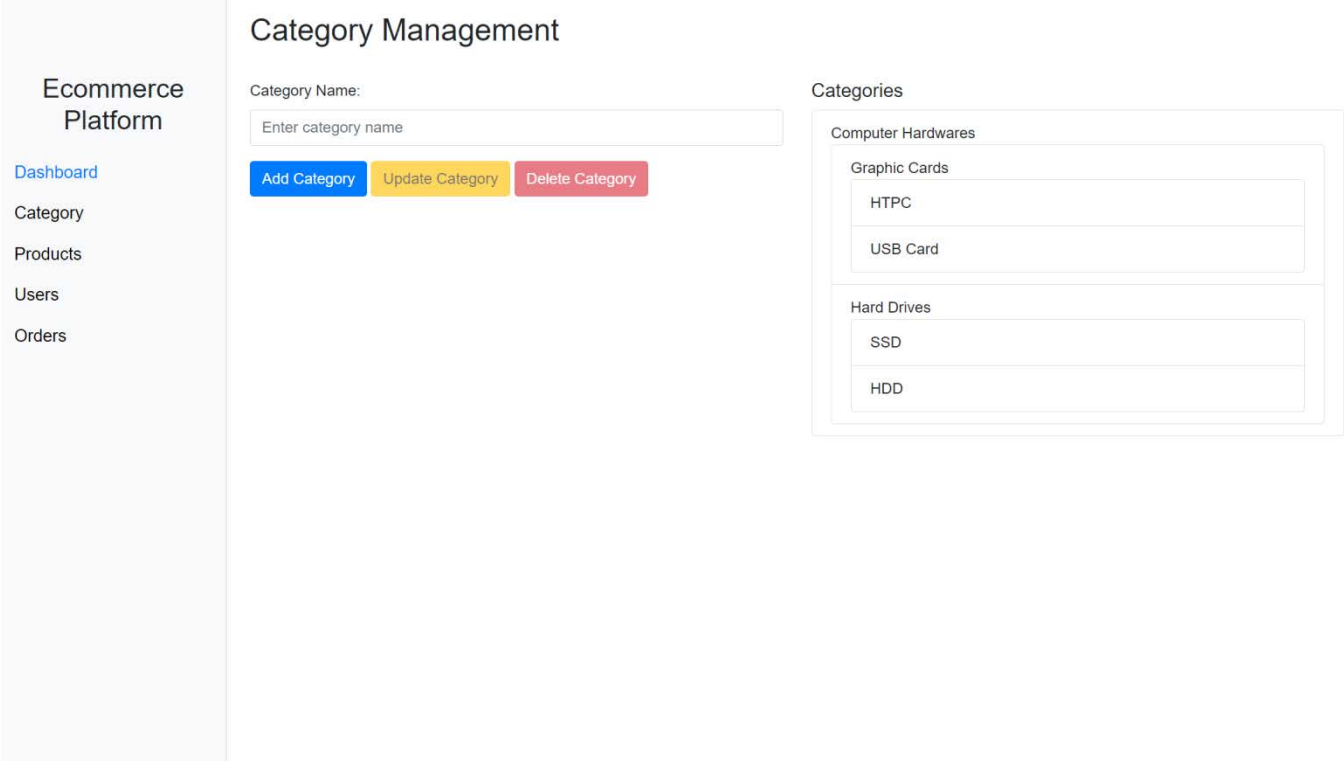
Step3. Must fill out the “Password” field



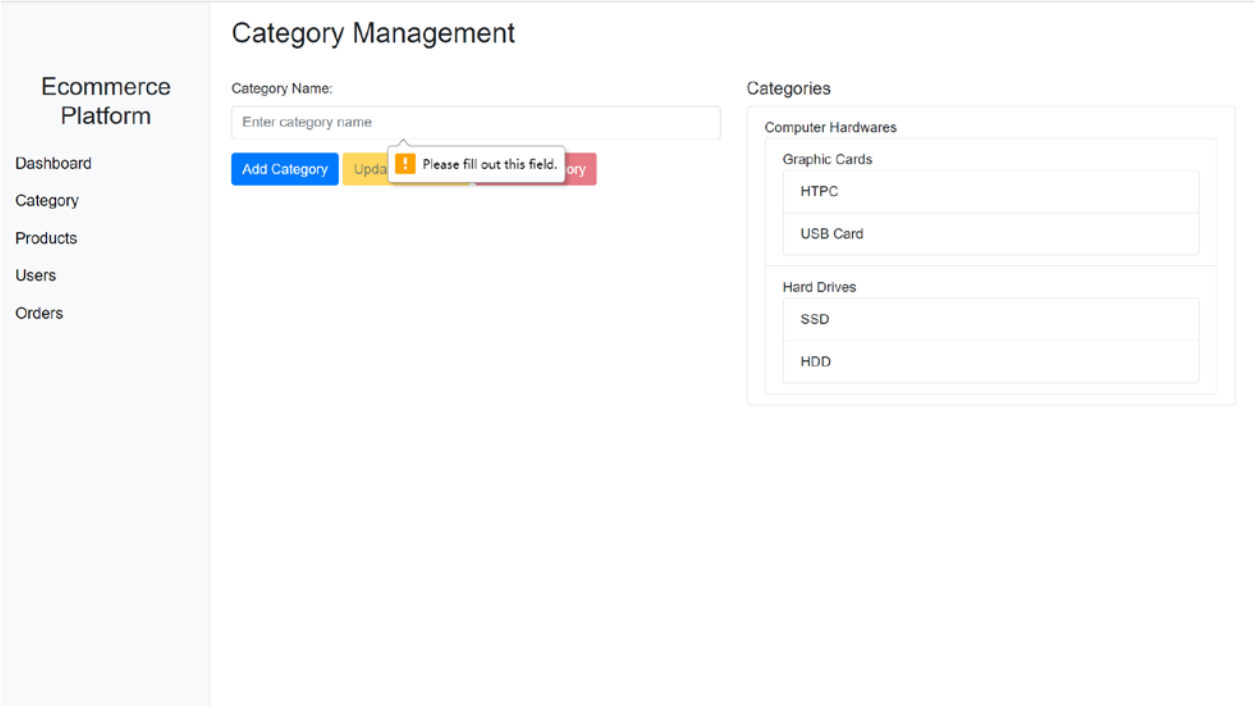
Step4. Correct user information, then click on the “Login” button.



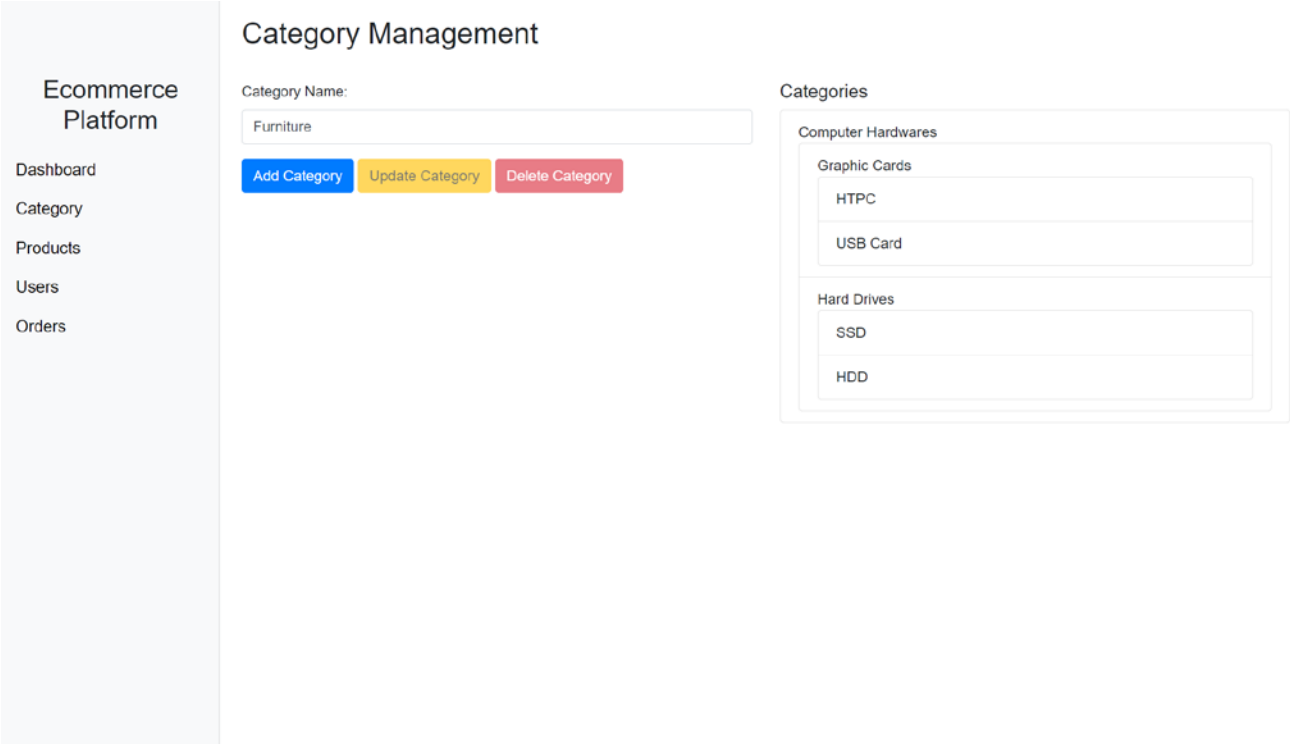
Step5. After clicking “Login”, it shows the Dashboard page. Click the “Pen” button in blue to edit the information.



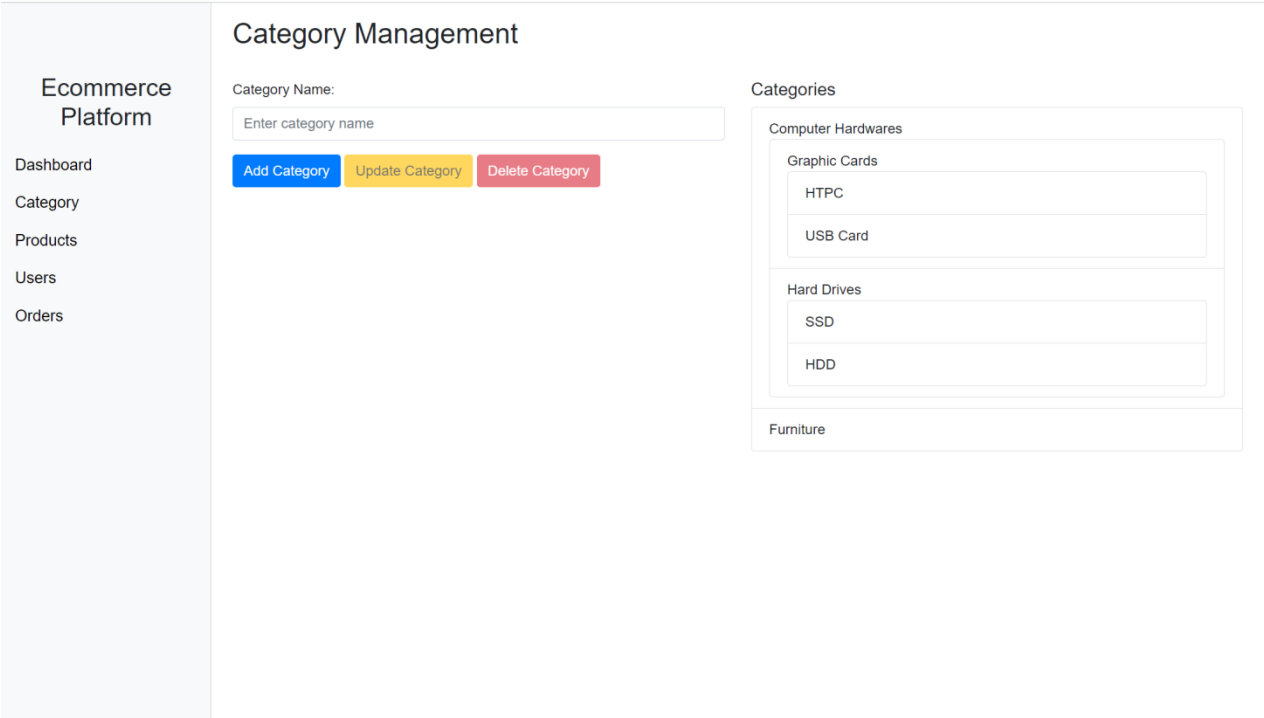
Step6. Click on the “Category” button in the menu on the left.



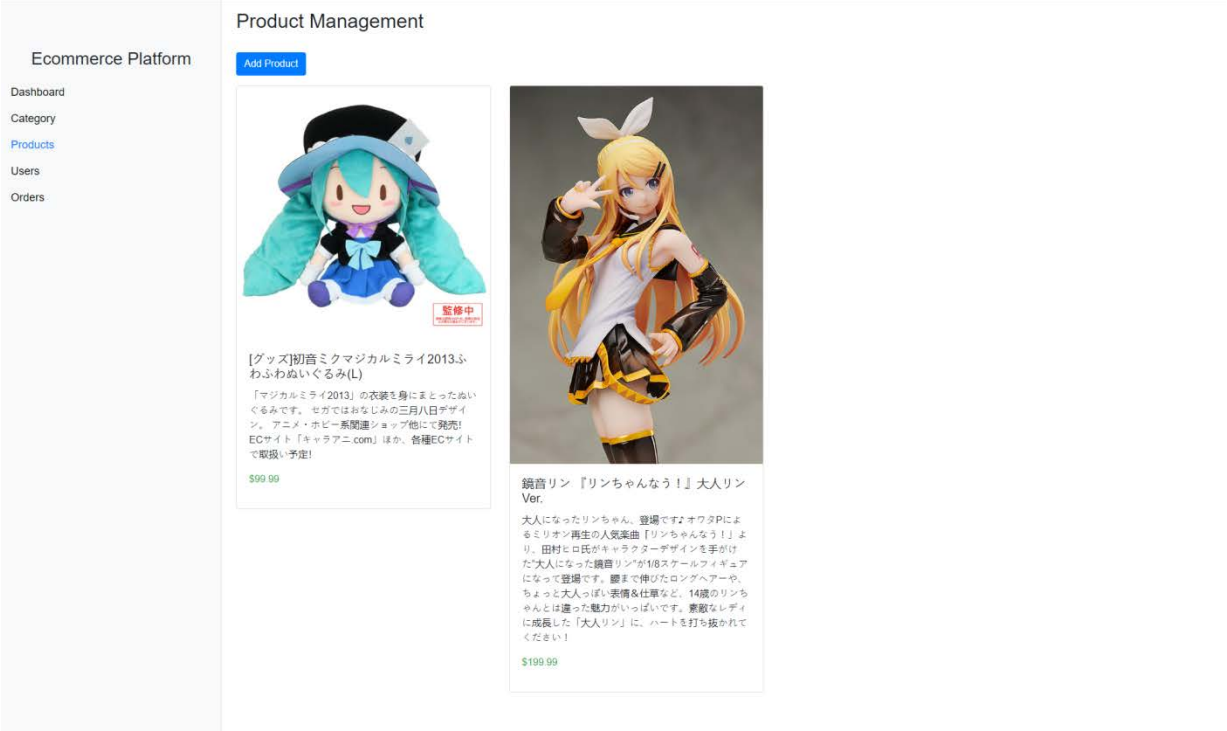
Step7. Must input the content, otherwise the system will indicate that.



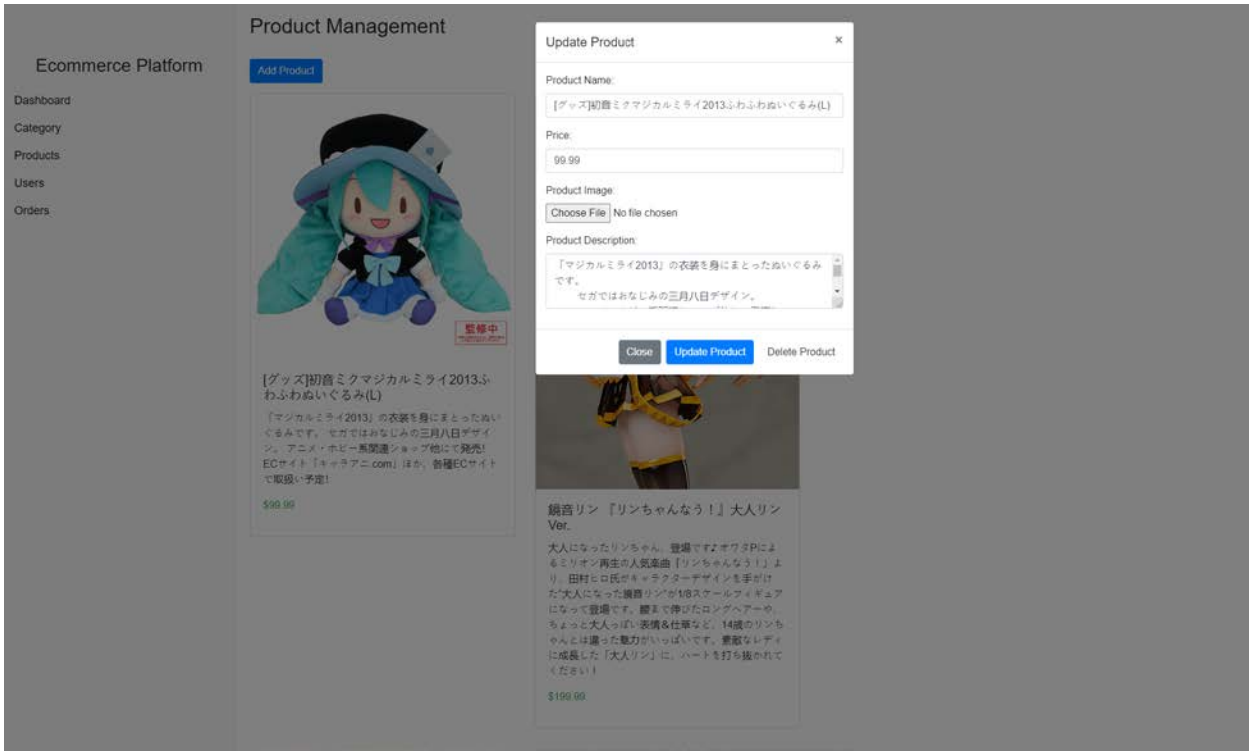
Step8. Input the category(e.g. Furniture), then click on the “Add Category” button.



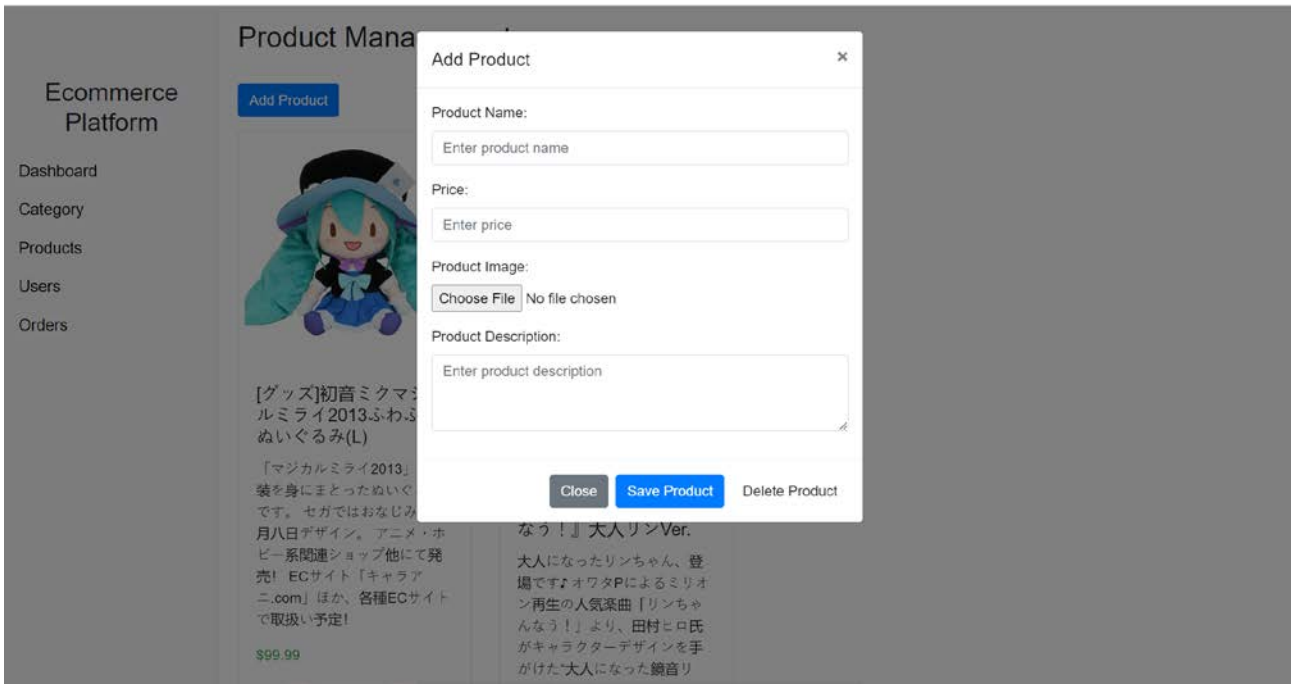
Step9. The system shows category on the right side. Click "Delete Category" to deleted it. Click "Update Category" to update it.



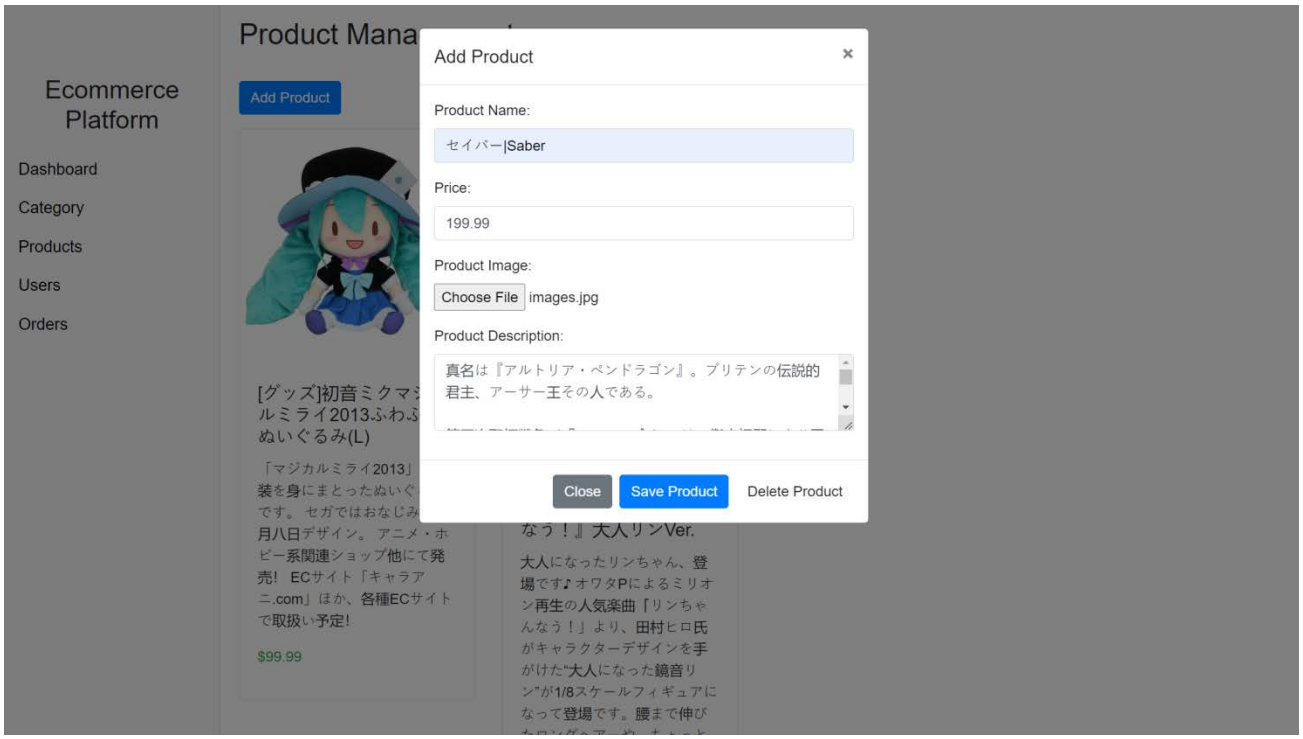
Step10. Click "Products" in the menu, it shows the "Product Management" page.



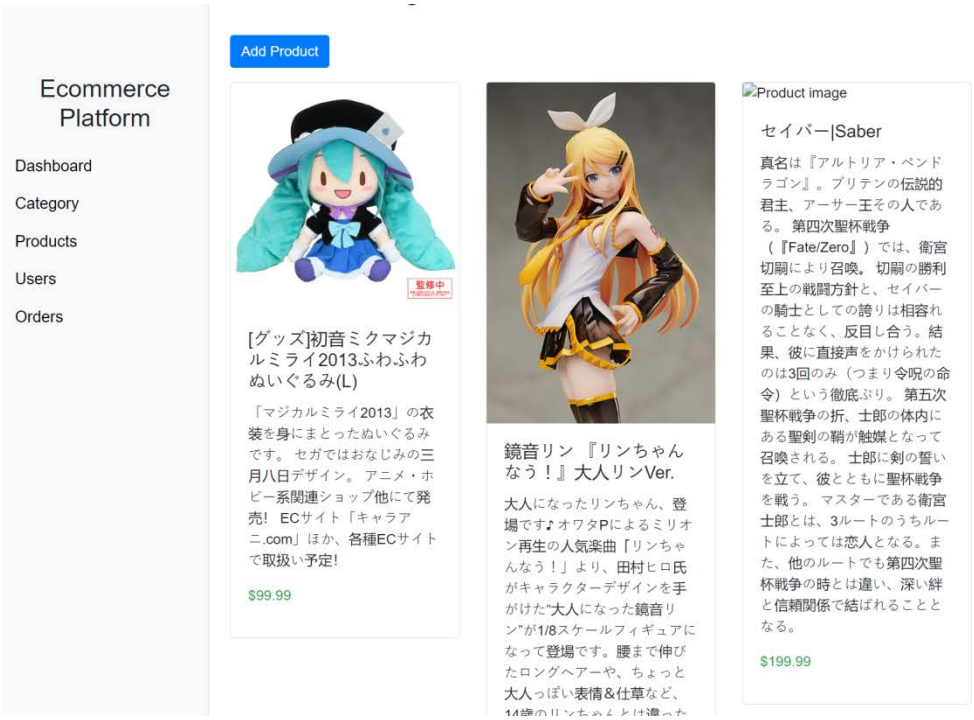
Step11. Click the certain product, alter the information of product, then click “Update Product”, successfully edit, or click “Close”, stop altering.



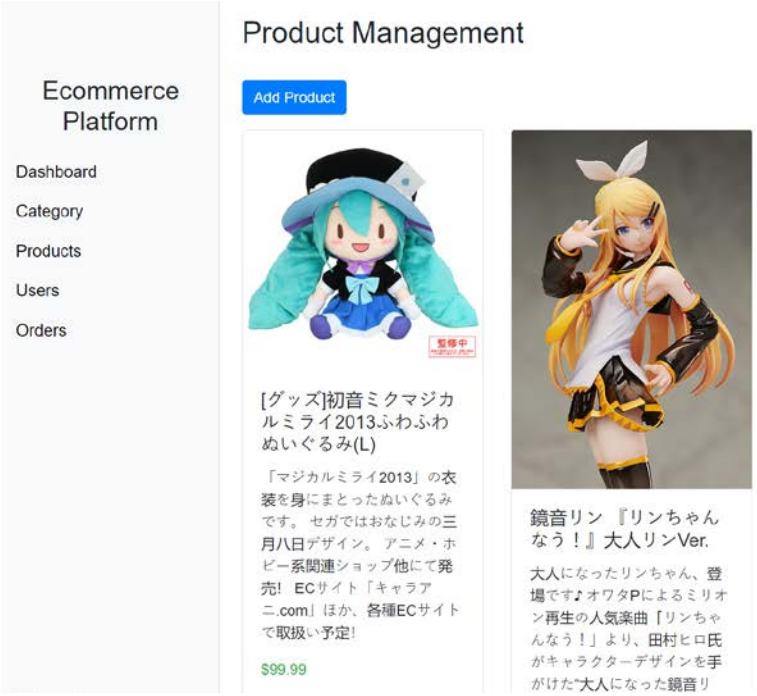
Step12. Click “Add Product”



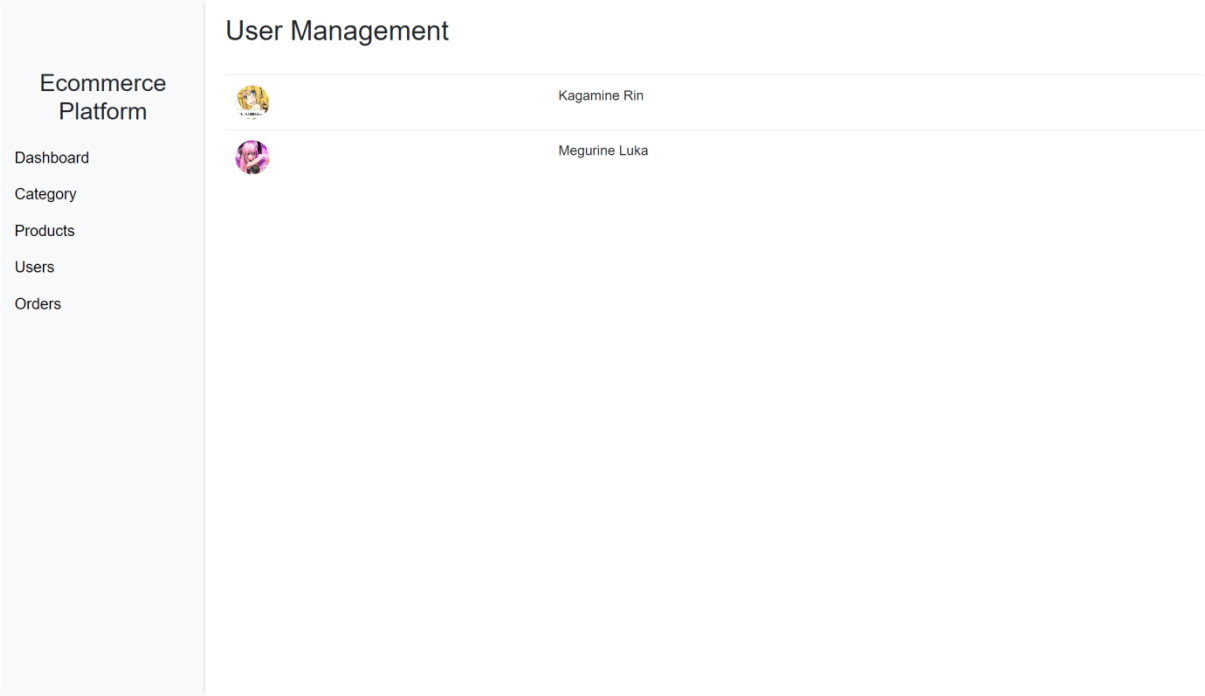
Step13. Add the information of the product, click “Save Product”



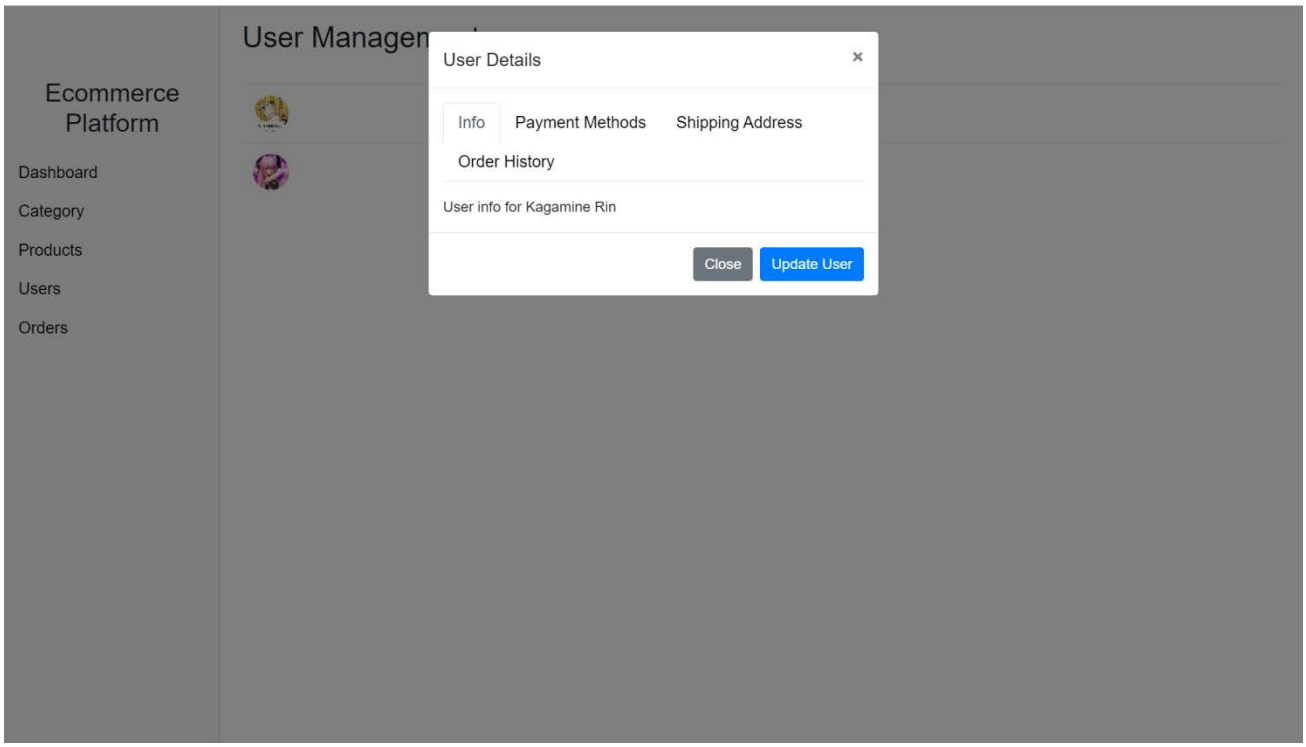
Step14. The new product is showed on the “Products” page



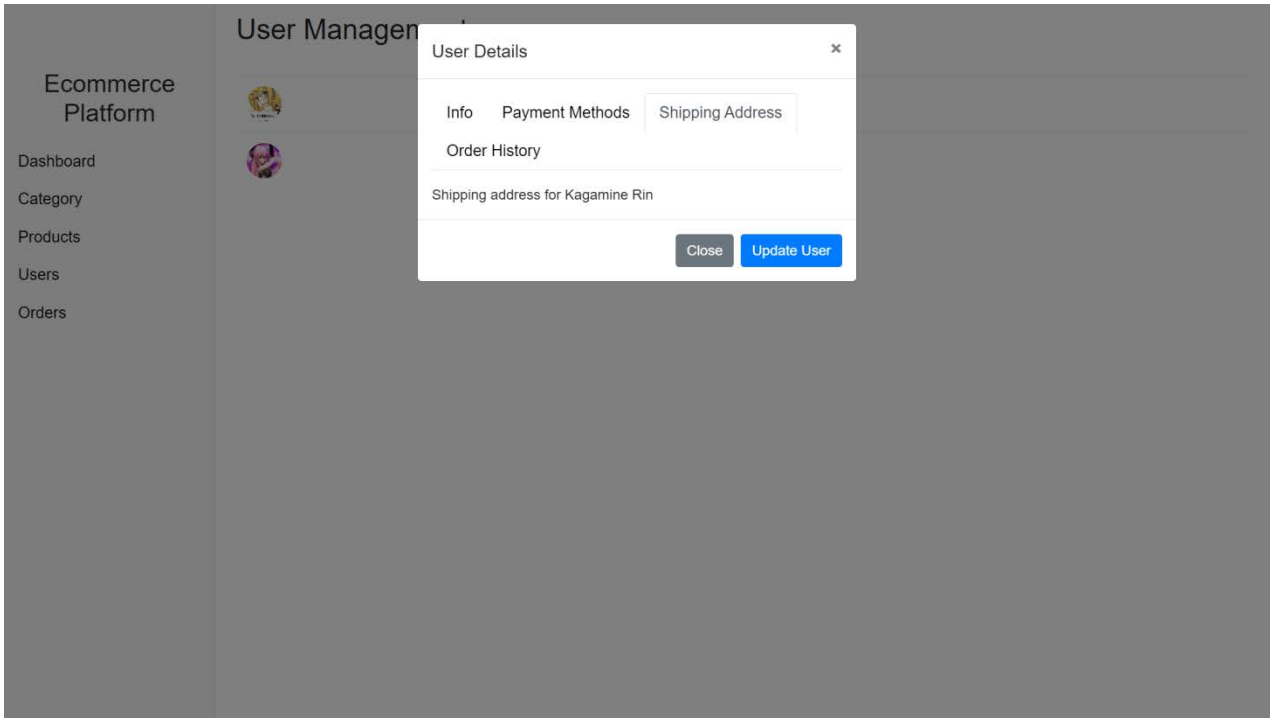
Step15. Click the new product, choose “Delete Product”, the certain product will be deleted.



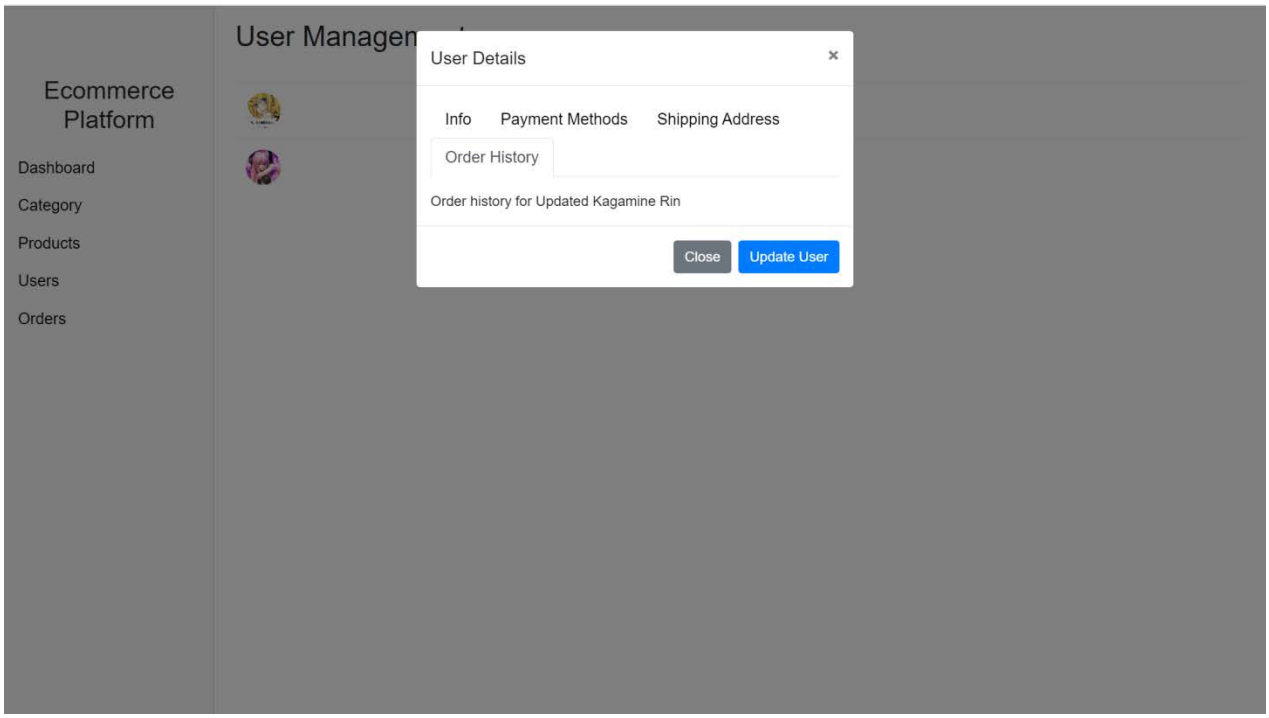
Step16. Click on the “Users” button on the left, it shows the “User Management” page



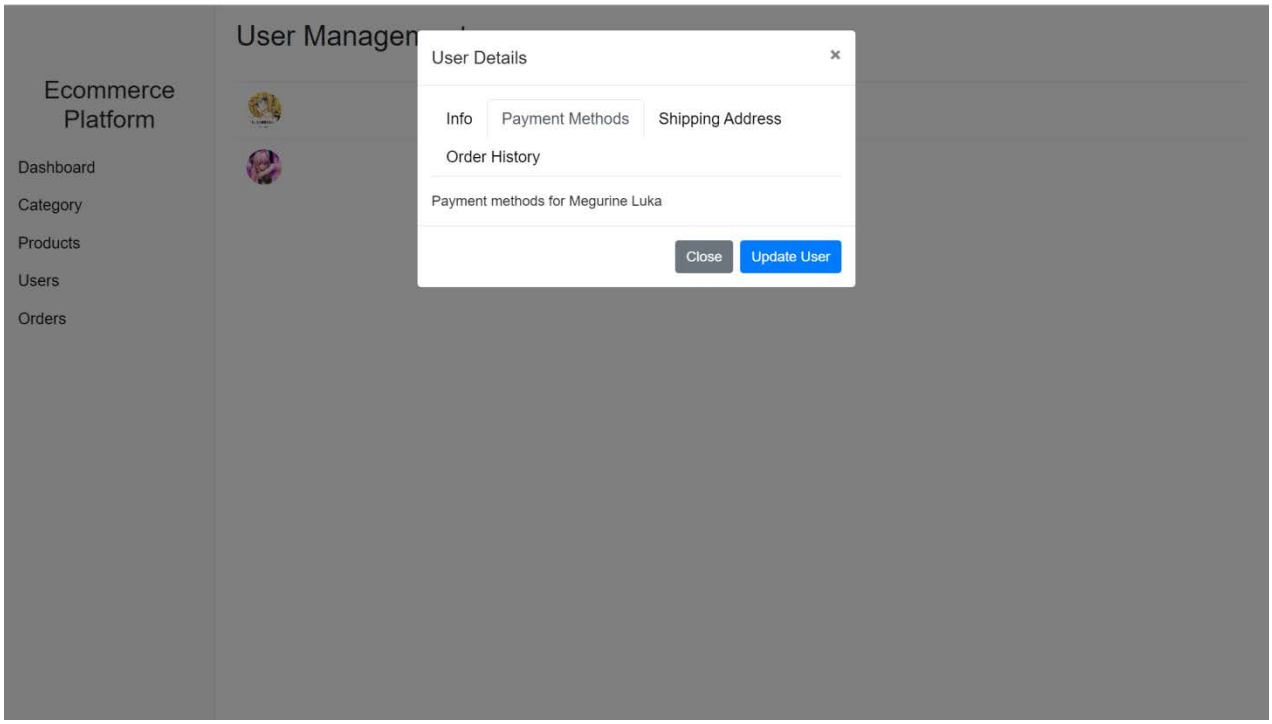
Step17. Click the certain user, it shows “User Details”. Click “Close” to stop altering



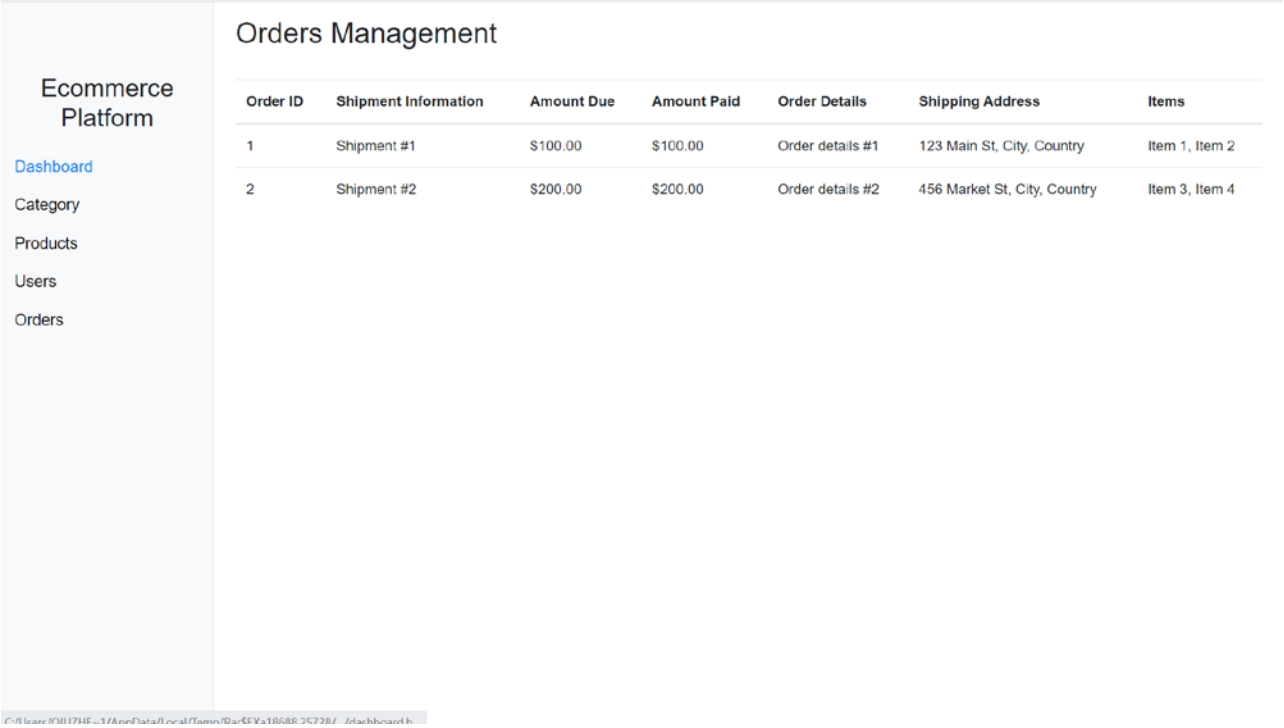
Step18. Update Shipping Address, click “Update User”, the shipping address is successfully updated



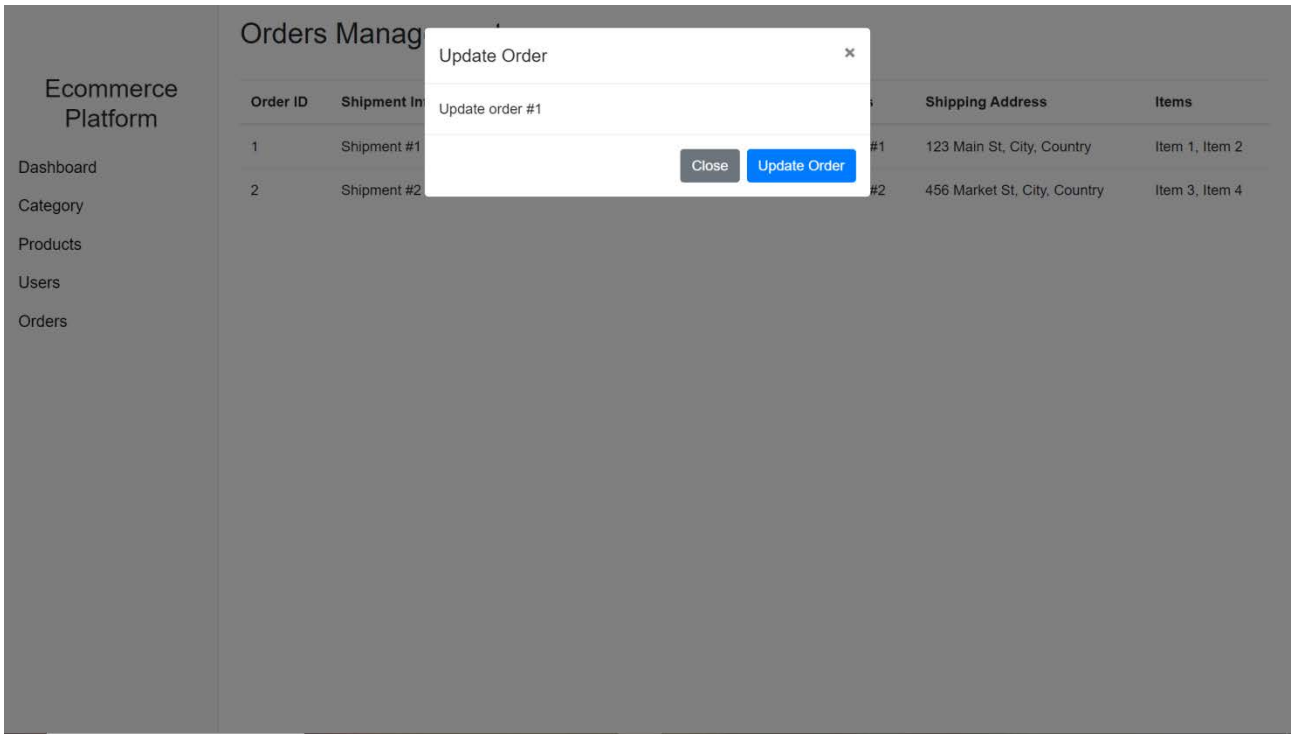
Step19. Update Order History, click “Update User”, the Order History is successfully updated



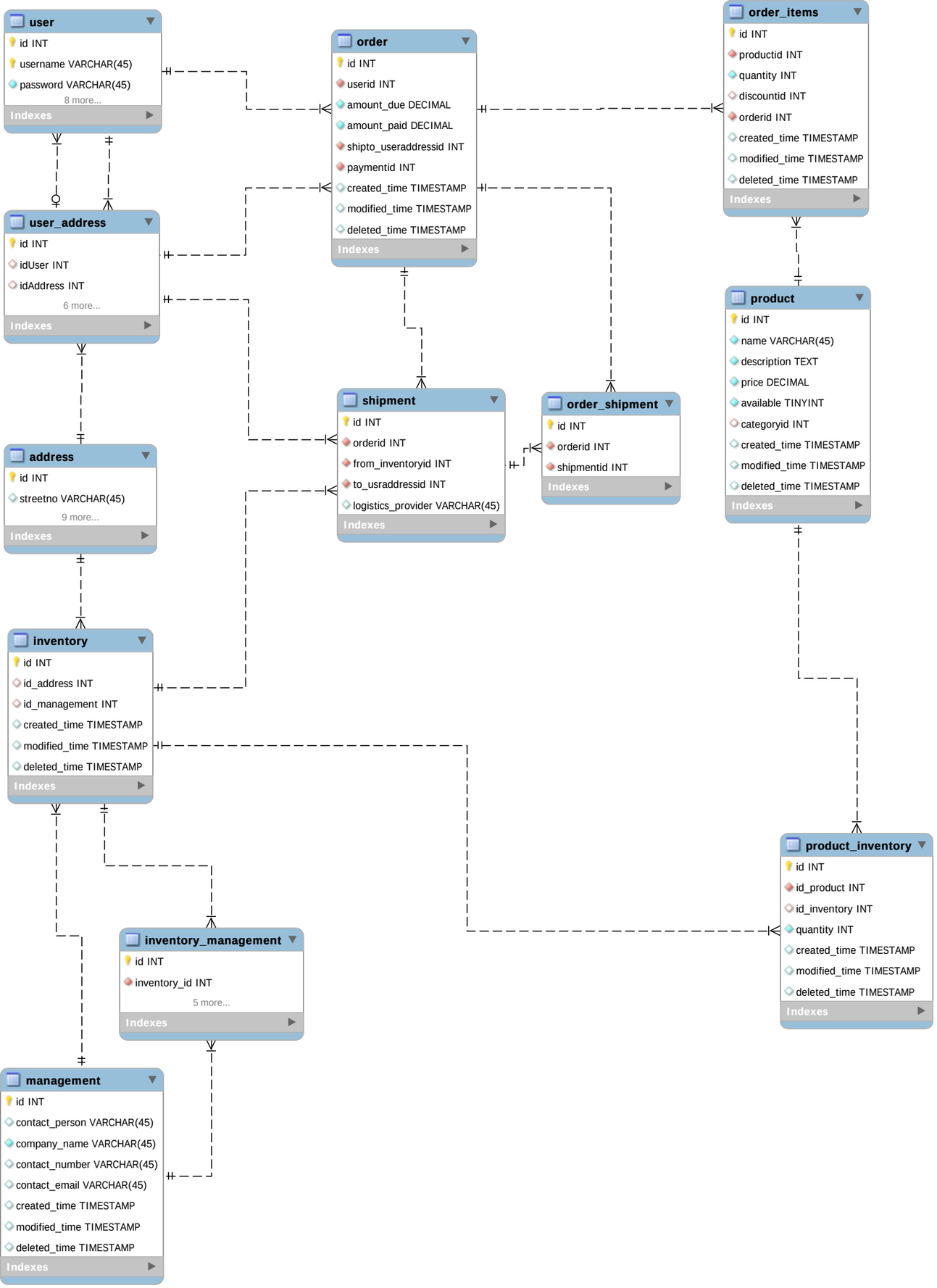
Step20. Update Payment Methods, click “Update User” to alter it successfully

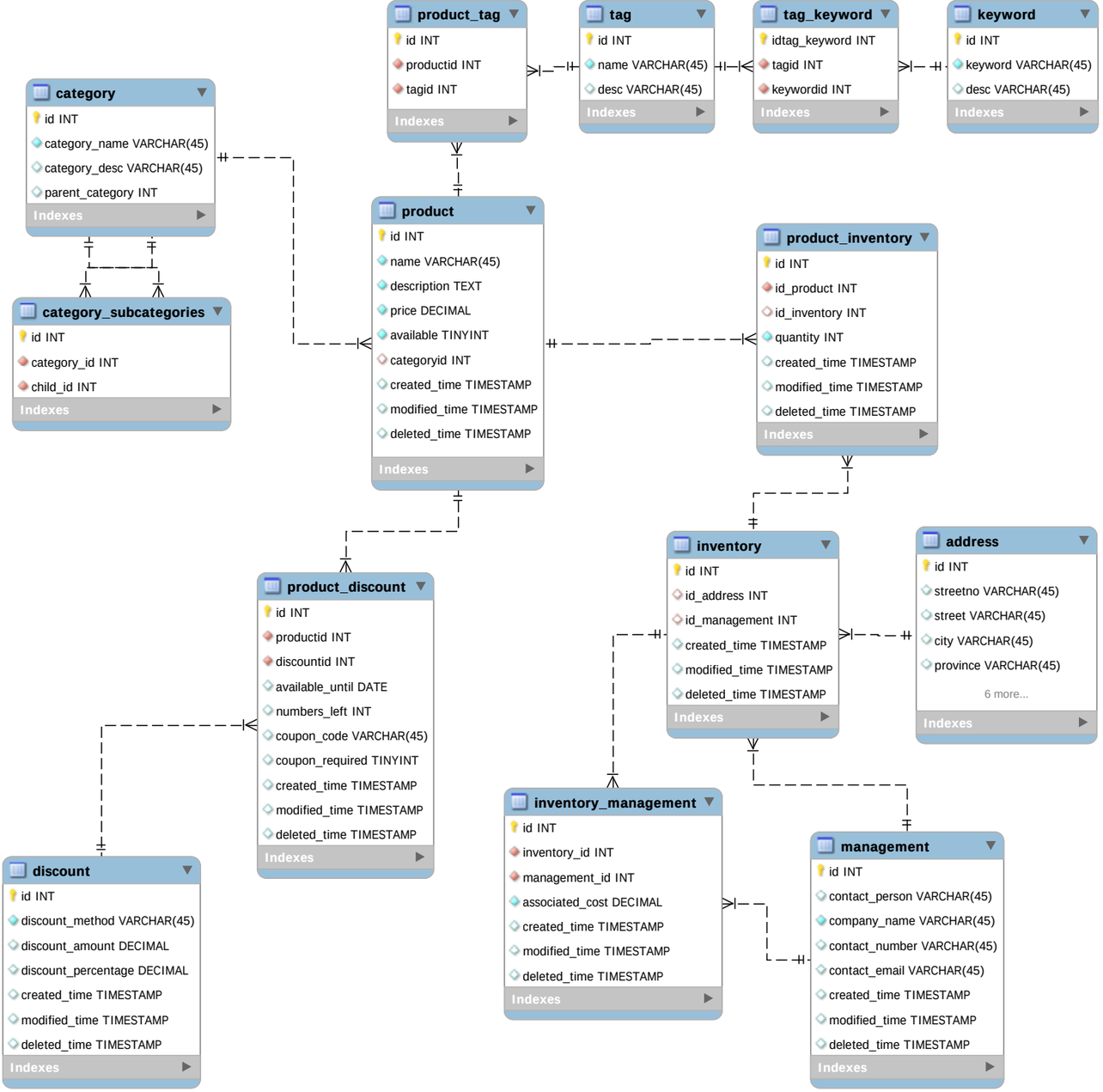


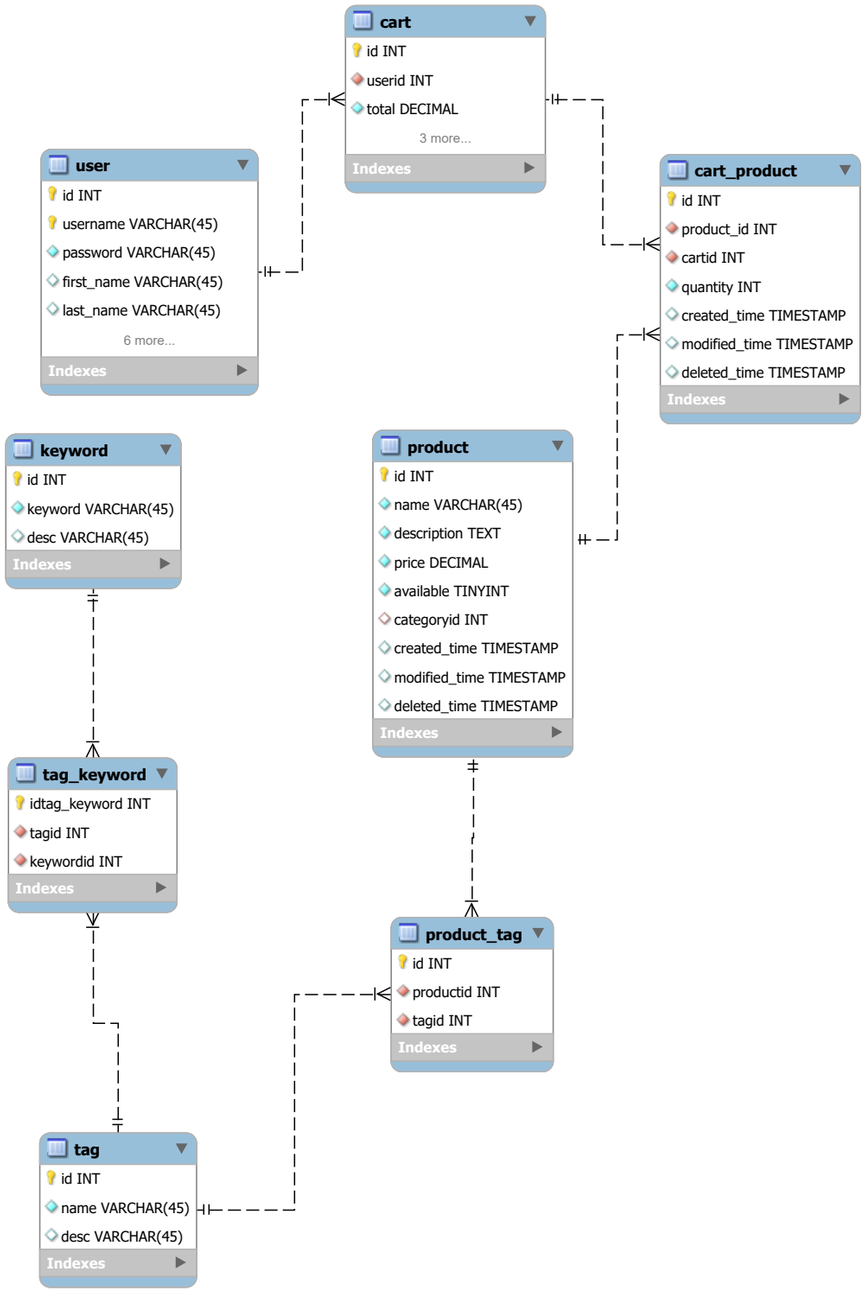
Step21. Click “Orders” button in the menu, it shows “Orders Management” page.



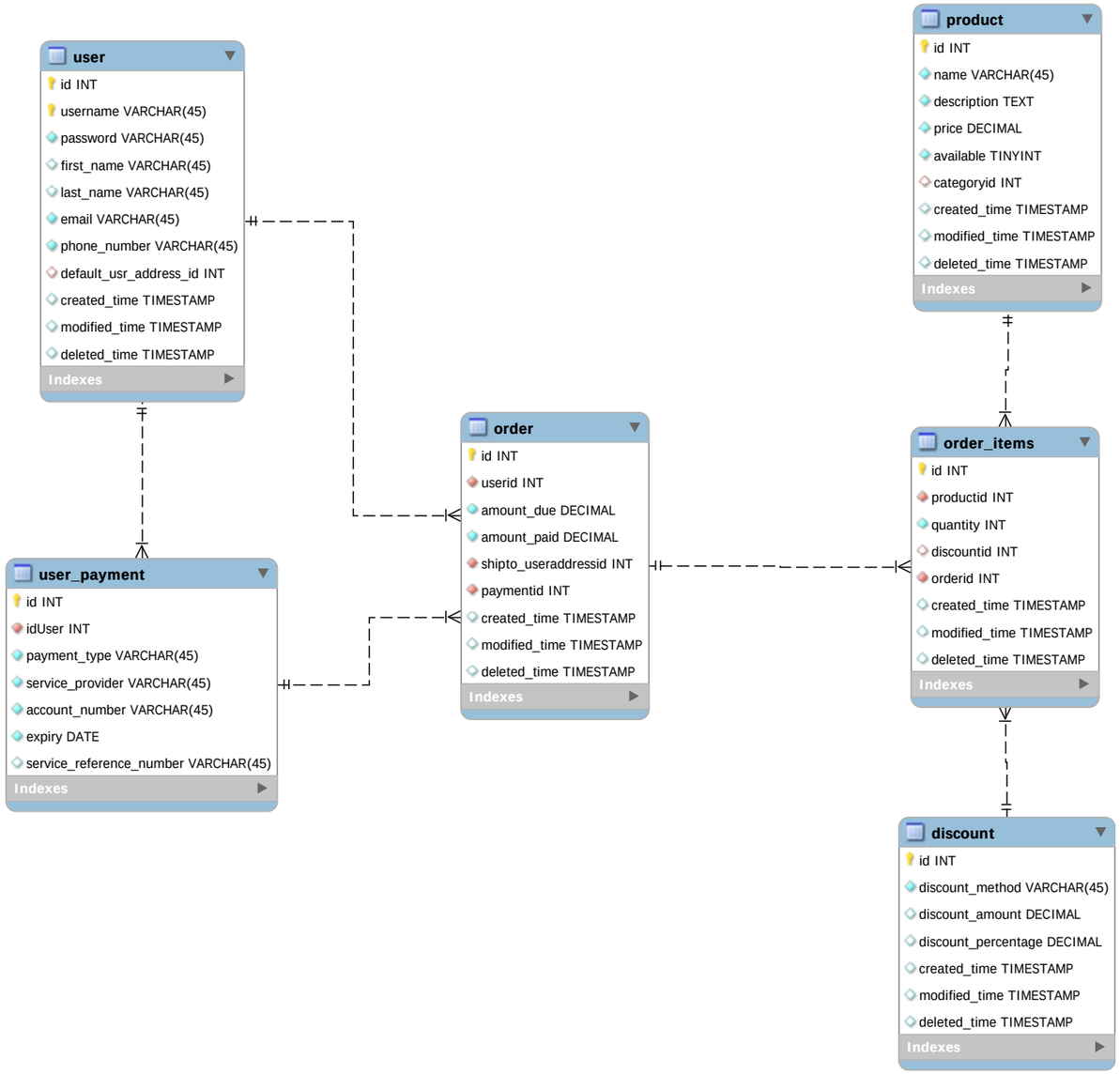
Step22. Click on a certain Shipment order. Click “Update Order” to successfully updated it. Click “Close” to stop edition.


user
id INT
username VARCHAR(45)
password VARCHAR(45)
8 more...
Indexes
order
id INT
userid INT
amount_due DECIMAL
amount_paid DECIMAL
shipto_useraddressid INT
paymentid INT
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes
order_items
id INT
productid INT
quantity INT
discountid INT
orderid INT
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes
user_address
id INT
idUser INT
idAddress INT
6 more...
Indexes
product
id INT
name VARCHAR(45)
description TEXT
price DECIMAL
available TINYINT
categoryid INT
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes
shipment
id INT
orderid INT
from_inventoryid INT
to_usraddressid INT
logistics_provider VARCHAR(45)
Indexes
order_shipment
id INT
orderid INT
shipmentid INT
Indexes
address
id INT
streetno VARCHAR(45)
9 more...
Indexes
inventory
id INT
id_address INT
id_management INT
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes
product_inventory
id INT
id_product INT
id_inventory INT
quantity INT
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes
inventory_management
id INT
inventory_id INT
5 more...
Indexes
management
id INT
contact_person VARCHAR(45)
company_name VARCHAR(45)
contact_number VARCHAR(45)
contact_email VARCHAR(45)
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes


product_tag
id INT
productid INT
tagid INT
Indexes
tag
id INT
name VARCHAR(45)
desc VARCHAR(45)
Indexes
tag_keyword
idtag_keyword INT
tagid INT
keywordid INT
Indexes
keyword
id INT
keyword VARCHAR(45)
desc VARCHAR(45)
Indexes
category
id INT
category_name VARCHAR(45)
category_desc VARCHAR(45)
parent_category INT
Indexes
category_subcategories
id INT
category_id INT
child_id INT
Indexes
product
id INT
name VARCHAR(45)
description TEXT
price DECIMAL
available TINYINT
categoryid INT
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes
product_inventory
id INT
id_product INT
id_inventory INT
quantity INT
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes
inventory
id INT
id_address INT
id_management INT
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes
address
id INT
streetno VARCHAR(45)
street VARCHAR(45)
city VARCHAR(45)
province VARCHAR(45)
6 more...
Indexes
product_discount
id INT
productid INT
discountid INT
available_until DATE
numbers_left INT
coupon_code VARCHAR(45)
coupon_required TINYINT
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes
discount
id INT
discount_method VARCHAR(45)
discount_amount DECIMAL
discount_percentage DECIMAL
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes
inventory_management
id INT
inventory_id INT
management_id INT
associated_cost DECIMAL
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes
management
id INT
contact_person VARCHAR(45)
company_name VARCHAR(45)
contact_number VARCHAR(45)
contact_email VARCHAR(45)
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes


cart
id INT
userid INT
total DECIMAL
3 more...
Indexes
user
id INT
username VARCHAR(45)
password VARCHAR(45)
first_name VARCHAR(45)
last_name VARCHAR(45)
6 more...
Indexes
cart_product
id INT
product_id INT
cartid INT
quantity INT
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes
keyword
id INT
keyword VARCHAR(45)
desc VARCHAR(45)
Indexes
product
id INT
name VARCHAR(45)
description TEXT
price DECIMAL
available TINYINT
categoryid INT
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes
tag_keyword
idtag_keyword INT
tagid INT
keywordid INT
Indexes
product_tag
id INT
productid INT
tagid INT
Indexes
tag
id INT
name VARCHAR(45)
desc VARCHAR(45)
Indexes


user_payment
id INT
idUser INT
payment_type VARCHAR(45)
service_provider VARCHAR(45)
account_number VARCHAR(45)
expiry DATE
service_reference_number VARCHAR(45)
Indexes
user
id INT
username VARCHAR(45)
password VARCHAR(45)
first_name VARCHAR(45)
last_name VARCHAR(45)
email VARCHAR(45)
phone_number VARCHAR(45)
default_usr_address_id INT
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes
user_address
id INT
idUser INT
idAddress INT
contact_name VARCHAR(45)
contact_phone VARCHAR(45)
contact_email VARCHAR(45)
cretated_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes
address
id INT
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
streetno VARCHAR(45)
street VARCHAR(45)
city VARCHAR(45)
province VARCHAR(45)
country VARCHAR(45)
postal_code VARCHAR(45)
apartment_no VARCHAR(45)
Indexes


user
id INT
username VARCHAR(45)
password VARCHAR(45)
first_name VARCHAR(45)
last_name VARCHAR(45)
email VARCHAR(45)
phone_number VARCHAR(45)
default_usr_address_id INT
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes
product
id INT
name VARCHAR(45)
description TEXT
price DECIMAL
available TINYINT
categoryid INT
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes
order
id INT
userid INT
amount_due DECIMAL
amount_paid DECIMAL
shipto_useraddressid INT
paymentid INT
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes
order_items
id INT
productid INT
quantity INT
discountid INT
orderid INT
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes
user_payment
id INT
idUser INT
payment_type VARCHAR(45)
service_provider VARCHAR(45)
account_number VARCHAR(45)
expiry DATE
service_reference_number VARCHAR(45)
Indexes
discount
id INT
discount_method VARCHAR(45)
discount_amount DECIMAL
discount_percentage DECIMAL
created_time TIMESTAMP
modified_time TIMESTAMP
deleted_time TIMESTAMP
Indexes

db_init.sql

```sql
-- MySQL Script generated by MySQL Workbench
-- Thu Apr 13 15:24:38 2023
-- Model: New Model    Version: 1.0
-- MySQL Workbench Forward Engineering

SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
SET @OLD_SQL_MODE=@@SQL_MODE,
SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_DIVISION

-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------
DROP SCHEMA IF EXISTS `mydb` ;

-- -----------------------------------------------------
-- Schema mydb
-- -----------------------------------------------------
CREATE SCHEMA IF NOT EXISTS `mydb` DEFAULT CHARACTER SET utf8 ;
USE `mydb` ;

-- -----------------------------------------------------
-- Table `mydb`.`address`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`address` ;

CREATE TABLE IF NOT EXISTS `mydb`.`address` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `streetno` VARCHAR(45) NULL,
  `street` VARCHAR(45) NULL,
  `city` VARCHAR(45) NULL,
  `province` VARCHAR(45) NULL,
  `country` VARCHAR(45) NULL,
  `postal_code` VARCHAR(45) NULL,
  `apartment_no` VARCHAR(45) NULL,
  `created_time` TIMESTAMP NULL DEFAULT CURRENT_TIMESTAMP,
  `modified_time` TIMESTAMP NULL,
  `deleted_time` TIMESTAMP NULL,
  PRIMARY KEY (`id`));


-- -----------------------------------------------------
-- Table `mydb`.`user_address`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`user_address` ;

CREATE TABLE IF NOT EXISTS `mydb`.`user_address` (
  `id` INT NULL AUTO_INCREMENT,
  `idUser` INT NULL,
  `idAddress` INT NULL,
  `contact_name` VARCHAR(45) NOT NULL,
  `contact_phone` VARCHAR(45) NOT NULL,
  `contact_email` VARCHAR(45) NULL,
  `cretated_time` TIMESTAMP NULL DEFAULT CURRENT_TIMESTAMP,
  `modified_time` TIMESTAMP NULL,
  `deleted_time` TIMESTAMP NULL,
```

```sql
  PRIMARY KEY (`id`),
  INDEX `fk_userid_idx` (`idUser` ASC) VISIBLE,
  INDEX `fk_useraddr_addrid_idx` (`idAddress` ASC) VISIBLE,
  CONSTRAINT `fk_useraddr_userid`
    FOREIGN KEY (`idUser`)
    REFERENCES `mydb`.`user` (`id`)
    ON DELETE NO ACTION
    ON UPDATE RESTRICT,
  CONSTRAINT `fk_useraddr_addrid`
    FOREIGN KEY (`idAddress`)
    REFERENCES `mydb`.`address` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`user`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`user` ;

CREATE TABLE IF NOT EXISTS `mydb`.`user` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `username` VARCHAR(45) NOT NULL,
  `password` VARCHAR(45) NOT NULL,
  `first_name` VARCHAR(45) NULL,
  `last_name` VARCHAR(45) NULL,
  `email` VARCHAR(45) NOT NULL,
  `phone_number` VARCHAR(45) NOT NULL,
  `default_usr_address_id` INT NULL DEFAULT -1,
  `created_time` TIMESTAMP NULL DEFAULT CURRENT_TIMESTAMP,
  `modified_time` TIMESTAMP NULL,
  `deleted_time` TIMESTAMP NULL,
  PRIMARY KEY (`id`, `username`),
  UNIQUE INDEX `username_UNIQUE` (`username` ASC) VISIBLE,
  INDEX `fk_user_defaultAddr_idx` (`default_usr_address_id` ASC) VISIBLE,
  CONSTRAINT `fk_user_defaultAddr`
    FOREIGN KEY (`default_usr_address_id`)
    REFERENCES `mydb`.`user_address` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`user_payment`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`user_payment` ;

CREATE TABLE IF NOT EXISTS `mydb`.`user_payment` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `idUser` INT NOT NULL,
  `payment_type` VARCHAR(45) NOT NULL,
  `service_provider` VARCHAR(45) NOT NULL,
  `account_number` VARCHAR(45) NOT NULL,
  `expiry` DATE NOT NULL,
  `service_reference_number` VARCHAR(45) NULL,
  PRIMARY KEY (`id`),
  INDEX `fk_payment_userid_idx` (`idUser` ASC) VISIBLE,
  CONSTRAINT `fk_payment_userid`
```

```sql
    FOREIGN KEY (`idUser`)
    REFERENCES `mydb`.`user` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`category`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`category` ;

CREATE TABLE IF NOT EXISTS `mydb`.`category` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `category_name` VARCHAR(45) NOT NULL,
  `category_desc` VARCHAR(45) NULL,
  `parent_category` INT NULL,
  PRIMARY KEY (`id`));


-- -----------------------------------------------------
-- Table `mydb`.`product`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`product` ;

CREATE TABLE IF NOT EXISTS `mydb`.`product` (
  `id` INT NOT NULL,
  `name` VARCHAR(45) NOT NULL,
  `description` TEXT NOT NULL,
  `price` DECIMAL NOT NULL,
  `available` TINYINT NOT NULL,
  `categoryid` INT NULL,
  `created_time` TIMESTAMP NULL DEFAULT CURRENT_TIMESTAMP,
  `modified_time` TIMESTAMP NULL,
  `deleted_time` TIMESTAMP NULL,
  PRIMARY KEY (`id`),
  INDEX `fk_product_cateid_idx` (`categoryid` ASC) VISIBLE,
  CONSTRAINT `fk_product_cateid`
    FOREIGN KEY (`categoryid`)
    REFERENCES `mydb`.`category` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`management`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`management` ;

CREATE TABLE IF NOT EXISTS `mydb`.`management` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `contact_person` VARCHAR(45) NULL,
  `company_name` VARCHAR(45) NOT NULL,
  `contact_number` VARCHAR(45) NULL,
  `contact_email` VARCHAR(45) NULL,
  `created_time` TIMESTAMP NULL DEFAULT CURRENT_TIMESTAMP,
  `modified_time` TIMESTAMP NULL,
  `deleted_time` TIMESTAMP NULL,
  PRIMARY KEY (`id`));
```

```sql
-- -----------------------------------------------------
-- Table `mydb`.`inventory`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`inventory` ;

CREATE TABLE IF NOT EXISTS `mydb`.`inventory` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `id_address` INT NULL,
  `id_management` INT NULL,
  `created_time` TIMESTAMP NULL DEFAULT CURRENT_TIMESTAMP,
  `modified_time` TIMESTAMP NULL,
  `deleted_time` TIMESTAMP NULL,
  PRIMARY KEY (`id`),
  INDEX `fk_inv_addr_idx` (`id_address` ASC) VISIBLE,
  INDEX `fk_inv_man_idx` (`id_management` ASC) VISIBLE,
  CONSTRAINT `fk_inv_addr`
    FOREIGN KEY (`id_address`)
    REFERENCES `mydb`.`address` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_inv_man`
    FOREIGN KEY (`id_management`)
    REFERENCES `mydb`.`management` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);


-- -----------------------------------------------------
-- Table `mydb`.`product_inventory`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`product_inventory` ;

CREATE TABLE IF NOT EXISTS `mydb`.`product_inventory` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `id_product` INT NOT NULL,
  `id_inventory` INT NULL,
  `quantity` INT NOT NULL,
  `created_time` TIMESTAMP NULL DEFAULT CURRENT_TIMESTAMP,
  `modified_time` TIMESTAMP NULL,
  `deleted_time` TIMESTAMP NULL,
  PRIMARY KEY (`id`),
  UNIQUE INDEX `id_UNIQUE` (`id` ASC) VISIBLE,
  INDEX `fk_prodinv_prodid_idx` (`id_product` ASC) VISIBLE,
  INDEX `fk_prodinv_invid_idx` (`id_inventory` ASC) VISIBLE,
  CONSTRAINT `fk_prodinv_prodid`
    FOREIGN KEY (`id_product`)
    REFERENCES `mydb`.`product` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_prodinv_invid`
    FOREIGN KEY (`id_inventory`)
    REFERENCES `mydb`.`inventory` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);


-- -----------------------------------------------------
-- Table `mydb`.`inventory_management`
```

```sql
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`inventory_management` ;

CREATE TABLE IF NOT EXISTS `mydb`.`inventory_management` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `inventory_id` INT NOT NULL,
  `management_id` INT NOT NULL,
  `associated_cost` DECIMAL NOT NULL,
  `created_time` TIMESTAMP NULL DEFAULT CURRENT_TIMESTAMP,
  `modified_time` TIMESTAMP NULL,
  `deleted_time` TIMESTAMP NULL,
  PRIMARY KEY (`id`),
  UNIQUE INDEX `inventory_id_UNIQUE` (`inventory_id` ASC) VISIBLE,
  INDEX `fk_inventman_man_idx` (`management_id` ASC) VISIBLE,
  CONSTRAINT `fk_inventman_inv`
    FOREIGN KEY (`inventory_id`)
    REFERENCES `mydb`.`inventory` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_inventman_man`
    FOREIGN KEY (`management_id`)
    REFERENCES `mydb`.`management` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);


-- -----------------------------------------------------
-- Table `mydb`.`discount`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`discount` ;

CREATE TABLE IF NOT EXISTS `mydb`.`discount` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `discount_method` VARCHAR(45) NOT NULL,
  `discount_amount` DECIMAL NULL,
  `discount_percentage` DECIMAL NULL,
  `created_time` TIMESTAMP NULL DEFAULT CURRENT_TIMESTAMP,
  `modified_time` TIMESTAMP NULL,
  `deleted_time` TIMESTAMP NULL,
  PRIMARY KEY (`id`));


-- -----------------------------------------------------
-- Table `mydb`.`cart`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`cart` ;

CREATE TABLE IF NOT EXISTS `mydb`.`cart` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `userid` INT NOT NULL,
  `total` DECIMAL NOT NULL,
  `created_time` TIMESTAMP NULL DEFAULT CURRENT_TIMESTAMP,
  `modified_time` TIMESTAMP NULL,
  `deleted_time` TIMESTAMP NULL,
  PRIMARY KEY (`id`),
  INDEX `fk_cart_uid_idx` (`userid` ASC) VISIBLE,
  CONSTRAINT `fk_cart_uid`
    FOREIGN KEY (`userid`)
    REFERENCES `mydb`.`user` (`id`)
    ON DELETE NO ACTION
```

```sql
    ON UPDATE NO ACTION);


-- -----------------------------------------------------
-- Table `mydb`.`cart_product`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`cart_product` ;

CREATE TABLE IF NOT EXISTS `mydb`.`cart_product` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `product_id` INT NOT NULL,
  `cartid` INT NOT NULL,
  `quantity` INT NOT NULL,
  `created_time` TIMESTAMP NULL DEFAULT CURRENT_TIMESTAMP,
  `modified_time` TIMESTAMP NULL,
  `deleted_time` TIMESTAMP NULL,
  PRIMARY KEY (`id`),
  INDEX `fk_cartprod_prodid_idx` (`product_id` ASC) VISIBLE,
  INDEX `fk_cartprod_cartid_idx` (`cartid` ASC) VISIBLE,
  CONSTRAINT `fk_cartprod_prodid`
    FOREIGN KEY (`product_id`)
    REFERENCES `mydb`.`product` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_cartprod_cartid`
    FOREIGN KEY (`cartid`)
    REFERENCES `mydb`.`cart` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);


-- -----------------------------------------------------
-- Table `mydb`.`order`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`order` ;

CREATE TABLE IF NOT EXISTS `mydb`.`order` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `userid` INT NOT NULL,
  `amount_due` DECIMAL NOT NULL,
  `amount_paid` DECIMAL NOT NULL,
  `shipto_useraddressid` INT NOT NULL,
  `paymentid` INT NOT NULL,
  `created_time` TIMESTAMP NULL DEFAULT CURRENT_TIMESTAMP,
  `modified_time` TIMESTAMP NULL,
  `deleted_time` TIMESTAMP NULL,
  PRIMARY KEY (`id`),
  INDEX `fk_order_useraddrid_idx` (`shipto_useraddressid` ASC) VISIBLE,
  INDEX `fk_order_userid_idx` (`userid` ASC) VISIBLE,
  INDEX `fk_order_paymentid_idx` (`paymentid` ASC, `userid` ASC) VISIBLE,
  CONSTRAINT `fk_order_userid`
    FOREIGN KEY (`userid`)
    REFERENCES `mydb`.`user` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_order_useraddrid`
    FOREIGN KEY (`shipto_useraddressid`)
    REFERENCES `mydb`.`user_address` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
```

```sql
  CONSTRAINT `fk_order_paymentid`
    FOREIGN KEY (`paymentid` , `userid`)
    REFERENCES `mydb`.`user_payment` (`id` , `idUser`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);


-- -----------------------------------------------------
-- Table `mydb`.`order_items`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`order_items` ;

CREATE TABLE IF NOT EXISTS `mydb`.`order_items` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `productid` INT NOT NULL,
  `quantity` INT NOT NULL DEFAULT 1,
  `discountid` INT NULL DEFAULT -1,
  `orderid` INT NOT NULL,
  `created_time` TIMESTAMP NULL DEFAULT CURRENT_TIMESTAMP,
  `modified_time` TIMESTAMP NULL,
  `deleted_time` TIMESTAMP NULL,
  PRIMARY KEY (`id`),
  INDEX `fk_orderitem_prodid_idx` (`productid` ASC) VISIBLE,
  INDEX `fk_orderitem_discountid_idx` (`discountid` ASC) VISIBLE,
  INDEX `fk_orderitem_orderid_idx` (`orderid` ASC) VISIBLE,
  CONSTRAINT `fk_orderitem_prodid`
    FOREIGN KEY (`productid`)
    REFERENCES `mydb`.`product` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_orderitem_discountid`
    FOREIGN KEY (`discountid`)
    REFERENCES `mydb`.`discount` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_orderitem_orderid`
    FOREIGN KEY (`orderid`)
    REFERENCES `mydb`.`order` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);


-- -----------------------------------------------------
-- Table `mydb`.`category_subcategories`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`category_subcategories` ;

CREATE TABLE IF NOT EXISTS `mydb`.`category_subcategories` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `category_id` INT NOT NULL,
  `child_id` INT NOT NULL,
  PRIMARY KEY (`id`),
  INDEX `fk_categoryhier_cateid_idx` (`category_id` ASC) VISIBLE,
  INDEX `fk_categoryhier_childid_idx` (`child_id` ASC) VISIBLE,
  CONSTRAINT `fk_categoryhier_cateid`
    FOREIGN KEY (`category_id`)
    REFERENCES `mydb`.`category` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_categoryhier_childid`
```

```sql
    FOREIGN KEY (`child_id`)
    REFERENCES `mydb`.`category` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION);


-- -----------------------------------------------------
-- Table `mydb`.`tag`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`tag` ;

CREATE TABLE IF NOT EXISTS `mydb`.`tag` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `name` VARCHAR(45) NOT NULL,
  `desc` VARCHAR(45) NULL,
  PRIMARY KEY (`id`));


-- -----------------------------------------------------
-- Table `mydb`.`keyword`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`keyword` ;

CREATE TABLE IF NOT EXISTS `mydb`.`keyword` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `keyword` VARCHAR(45) NOT NULL,
  `desc` VARCHAR(45) NULL,
  PRIMARY KEY (`id`))
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`tag_keyword`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`tag_keyword` ;

CREATE TABLE IF NOT EXISTS `mydb`.`tag_keyword` (
  `idtag_keyword` INT NOT NULL AUTO_INCREMENT,
  `tagid` INT NOT NULL,
  `keywordid` INT NOT NULL,
  PRIMARY KEY (`idtag_keyword`),
  INDEX `fk_tagkey_tagid_idx` (`tagid` ASC) VISIBLE,
  INDEX `fk_tagkey_keyid_idx` (`keywordid` ASC) VISIBLE,
  CONSTRAINT `fk_tagkey_tagid`
    FOREIGN KEY (`tagid`)
    REFERENCES `mydb`.`tag` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_tagkey_keyid`
    FOREIGN KEY (`keywordid`)
    REFERENCES `mydb`.`keyword` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`product_tag`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`product_tag` ;
```

```sql
CREATE TABLE IF NOT EXISTS `mydb`.`product_tag` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `productid` INT NOT NULL,
  `tagid` INT NOT NULL,
  PRIMARY KEY (`id`),
  INDEX `fk_prodtag_prodid_idx` (`productid` ASC) VISIBLE,
  INDEX `fk_prodtag_tagid_idx` (`tagid` ASC) VISIBLE,
  CONSTRAINT `fk_prodtag_prodid`
    FOREIGN KEY (`productid`)
    REFERENCES `mydb`.`product` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_prodtag_tagid`
    FOREIGN KEY (`tagid`)
    REFERENCES `mydb`.`tag` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`shipment`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`shipment` ;

CREATE TABLE IF NOT EXISTS `mydb`.`shipment` (
  `id` INT NOT NULL AUTO_INCREMENT,
  `orderid` INT NOT NULL,
  `from_inventoryid` INT NOT NULL,
  `to_usraddressid` INT NOT NULL,
  `logistics_provider` VARCHAR(45) NULL,
  PRIMARY KEY (`id`),
  INDEX `fk_ship_orderid_idx` (`orderid` ASC) VISIBLE,
  INDEX `fk_ship_frominvid_idx` (`from_inventoryid` ASC) VISIBLE,
  INDEX `fk_ship_tousraddrid_idx` (`to_usraddressid` ASC) VISIBLE,
  CONSTRAINT `fk_ship_orderid`
    FOREIGN KEY (`orderid`)
    REFERENCES `mydb`.`order` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_ship_frominvid`
    FOREIGN KEY (`from_inventoryid`)
    REFERENCES `mydb`.`inventory` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_ship_tousraddrid`
    FOREIGN KEY (`to_usraddressid`)
    REFERENCES `mydb`.`user_address` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


-- -----------------------------------------------------
-- Table `mydb`.`order_shipment`
-- -----------------------------------------------------
DROP TABLE IF EXISTS `mydb`.`order_shipment` ;

CREATE TABLE IF NOT EXISTS `mydb`.`order_shipment` (
```

```sql
  `id` INT NOT NULL AUTO_INCREMENT,
  `orderid` INT NOT NULL,
  `shipmentid` INT NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE INDEX `shipmentid_UNIQUE` (`shipmentid` ASC) VISIBLE,
  INDEX `fk_ordership_orderid_idx` (`orderid` ASC) VISIBLE,
  CONSTRAINT `fk_ordership_orderid`
    FOREIGN KEY (`orderid`)
    REFERENCES `mydb`.`order` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION,
  CONSTRAINT `fk_ordership_shipid`
    FOREIGN KEY (`shipmentid`)
    REFERENCES `mydb`.`shipment` (`id`)
    ON DELETE NO ACTION
    ON UPDATE NO ACTION)
ENGINE = InnoDB;


SET SQL_MODE=@OLD_SQL_MODE;
SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;
```