

# DB Coursework Documentation:

## Online Second-Hand Trading Platform

Yunnan He

### Content

<b>1 System Workflow .....</b>	<b>3</b>
<b>2 Schema in PostgreSQL .....</b>	<b>4</b>
<b>3 function implementation .....</b>	<b>6</b>
3.1 User module .....	6
3.1.1 Check if registered .....	6
3.1.2 Create a user .....	7
3.1.3 Log in .....	7
3.1.4 get_user_info .....	8
3.2 Goods module .....	9
3.2.1 Create goods .....	9
3.2.2 Update goods.....	9
3.2.3 Get goods list.....	10
3.2.4 Get goods for a specific user .....	11
3.2.5 Delete goods.....	11
3.2.6 Get goods detail .....	12
3.3 Order module .....	13
3.3.1 Create an order .....	13
3.3.2 Abandon an order .....	13
3.3.3 Approve an order .....	14
3.3.4 Establish an order .....	14
3.3.5 Deliver goods .....	15
3.3.6 Finish order .....	16
3.3.7 Get order by user and goods .....	17

3.3.8 Get order information for a specific user.....	17
3.3.9 Get order information to a specific user.....	18
3.4 Comment module .....	18
3.4.1 Create a comment .....	18
3.4.2 Get comments for a specific goods .....	18
3.4.3 Delete a comment .....	19
3.5 Address Module .....	20
3.5.1 Create address .....	20
3.5.2 Get address list of a specific user.....	20
3.5.3 Delete address .....	20
<b>4 Optimization .....</b>	<b>22</b>
4.1 Adding constraints on the password when signing in .....	22
4.2 Optimizing on presenting the publisher of comments .....	23
4.3 Enhancing the stability when creating comments .....	24
<b>5 Summary &amp; Reflection .....</b>	<b>25</b>
<b>6 Suggestions w.r.t. the course .....</b>	<b>25</b>

# 1 System Workflow

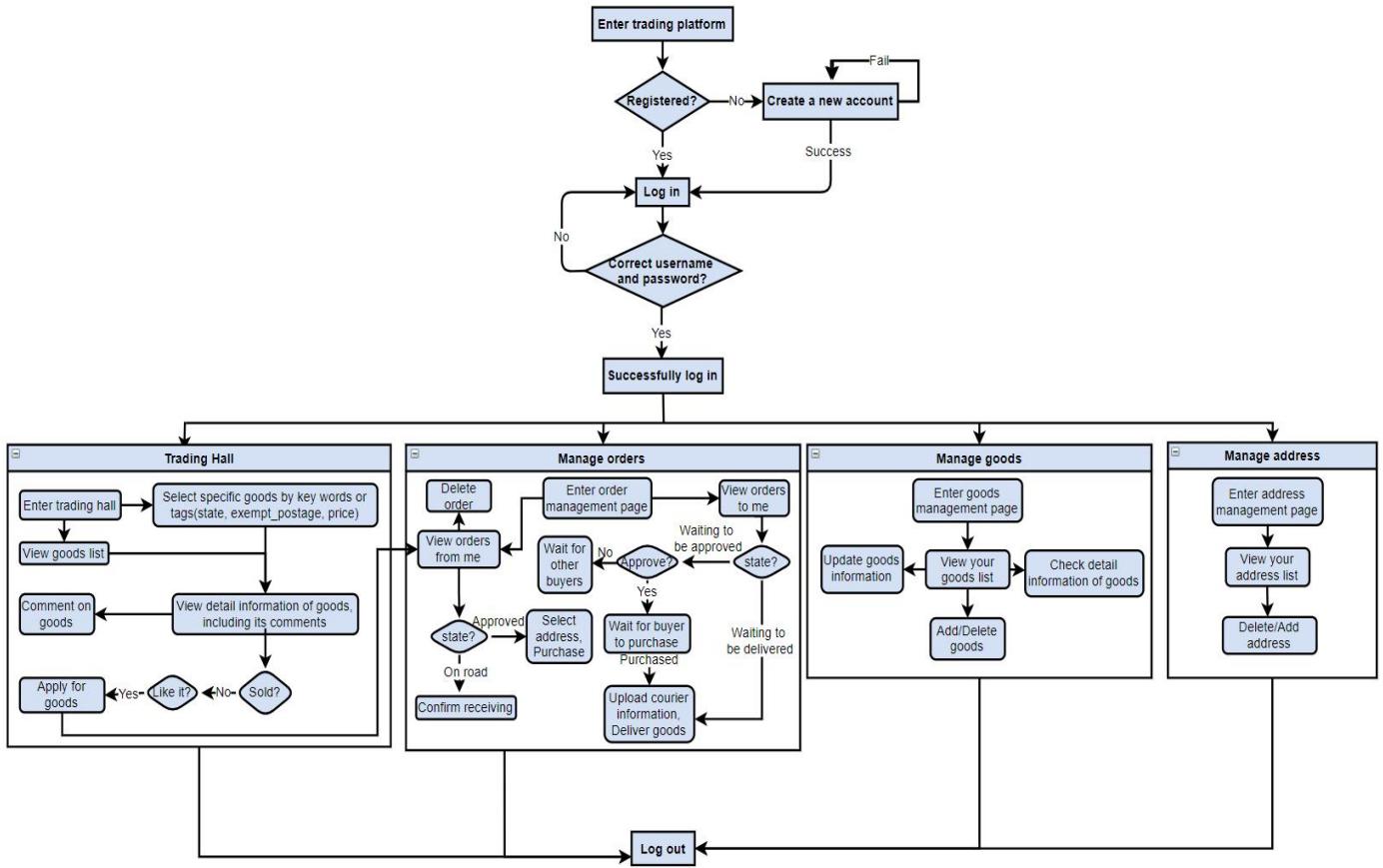


Figure 1: System Flowchart

Users start with creating an account and logging in the system. After logging in, they can check out four function pages: trading hall, goods management, order management and address management. Detailed information will be listed below.

## 2 Schema in PostgreSQL

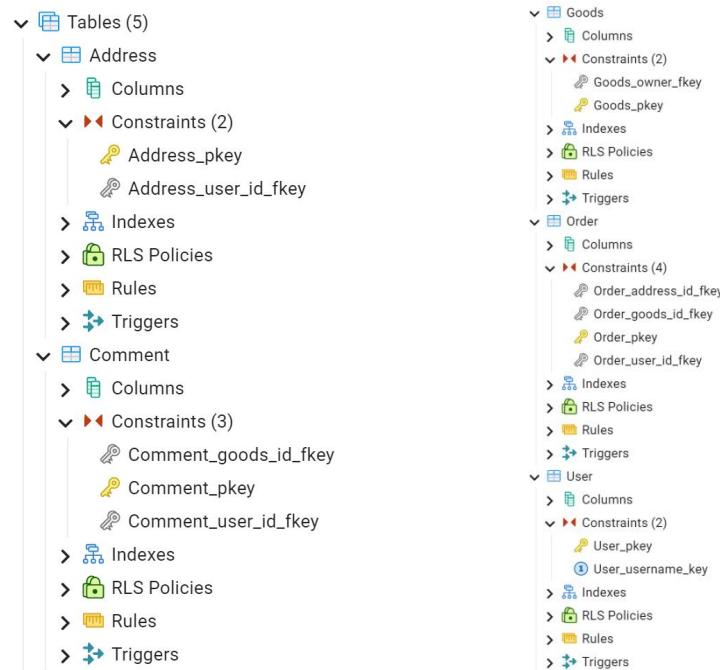


Figure 2: Schema in Database

Detailed information of tables are shown in Figure 3-7.

The screenshot shows the 'Address' table configuration. The 'Columns' tab is selected. The table has five columns: 'id' (integer, primary key, not null), 'user\_id' (integer, not null, default nextval('Ac')), 'name' (character varying, length 255, not null), 'phone' (character varying, length 255, not null), and 'location' (character varying, length 255, not null).

	Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
<input type="checkbox"/>	id	integer			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
<input type="checkbox"/>	user_id	integer			<input checked="" type="checkbox"/>	<input type="checkbox"/>	nextval("Ac")
<input type="checkbox"/>	name	character varying	255		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	phone	character varying	255		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	location	character varying	255		<input checked="" type="checkbox"/>	<input type="checkbox"/>	

Figure 3: Table "Address"

The screenshot shows the 'Comment' table configuration. The 'Columns' tab is selected. The table has five columns: 'id' (integer, primary key, not null, default nextval('Cc')), 'user\_id' (integer, not null, default nextval('Cc')), 'goods\_id' (integer, not null, default nextval('Cc')), 'content' (character varying, length 255, not null), and 'create\_at' (timestamp without time zone, not null, default CURRENT\_).

	Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
<input type="checkbox"/>	id	integer			<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	nextval("Cc")
<input type="checkbox"/>	user_id	integer			<input checked="" type="checkbox"/>	<input type="checkbox"/>	nextval("Cc")
<input type="checkbox"/>	goods_id	integer			<input checked="" type="checkbox"/>	<input type="checkbox"/>	nextval("Cc")
<input type="checkbox"/>	content	character varying	255		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
<input type="checkbox"/>	create_at	timestamp without time zone			<input type="checkbox"/>	<input checked="" type="checkbox"/>	CURRENT_

Figure 4: Table "Comment"

**Goods**

	Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
	id	integer	▾		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	nextval("Go
	name	character varying	▾	255	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
	description	character varying	▾	1024	<input type="checkbox"/>	<input type="checkbox"/>	
	img	character varying	▾	255	<input type="checkbox"/>	<input type="checkbox"/>	
	price	numeric	▾	20	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
	owner	integer	▾		<input checked="" type="checkbox"/>	<input type="checkbox"/>	nextval("Go
	exempt_postage	boolean	▾		<input checked="" type="checkbox"/>	<input type="checkbox"/>	

Figure 5: Table “Goods”

**Order**

	Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
	id	integer	▾		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	nextval("Or
	user_id	integer	▾		<input checked="" type="checkbox"/>	<input type="checkbox"/>	nextval("Or
	goods_id	integer	▾		<input checked="" type="checkbox"/>	<input type="checkbox"/>	nextval("Or
	state	integer	▾		<input checked="" type="checkbox"/>	<input type="checkbox"/>	
	address_id	integer	▾		<input type="checkbox"/>	<input type="checkbox"/>	
	express_code	character varying	▾	255	<input type="checkbox"/>	<input type="checkbox"/>	
	express_company	character varying	▾	255	<input type="checkbox"/>	<input type="checkbox"/>	

Figure 6: Table “Order”

**User**

	Name	Data type	Length/Precision	Scale	Not NULL?	Primary key?	Default
	id	integer	▾		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	nextval("Us
	username	character varying	▾	255	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
	password	character varying	▾	255	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
	create_at	timestamp without tim...	▾		<input type="checkbox"/>	<input type="checkbox"/>	CURRENT_

Figure 7: Table “User”

## 3 function implementation

### 3.1 User module

#### 3.1.1 Check if registered

Username on this platform must be unique. So we implement this function by comparing the input name with the existing names in the current “User” table row to row. If the name is not the same with any existing name, then the new user can use it as his/her username.

Screenshots:

50 `SELECT * from "User";`

Data Output Messages Notifications

	<code>id</code> [PK] integer	<code>username</code> character varying (255)	<code>password</code> character varying (255)	<code>create_at</code> timestamp without time zone
1	1	Alice	698d51a19d8a121ce581499d7b7016...	2023-05-28 11:49:40.342144
2	2	Bob	bcbe3365e6ac95ea2c0343a2395834...	2023-05-28 11:49:52.012987
3	3	Cindy	310dcbbf4cce62f762a2aa148d556bd	2023-05-28 11:50:07.270942

Figure 8: Current “User” Table

Figure 9: Enter an existing name, registration fails

Figure 10: Enter an existing name, registration succeeds

### 3.1.2 Create a user

To create a user, we find out the max id existed and add 1 to get the new id. We use md5 encryption when storing the password. Screenshots:

	<b>id</b> [PK] integer	<b>username</b> character varying (255)	<b>password</b> character varying (255)	<b>create_at</b> timestamp without time zone
1	1	Alice	698d51a19d8a121ce581499d7b701...	2023-05-28 11:49:40.342144
2	2	Bob	bcbe3365e6ac95ea2c0343a2395834...	2023-05-28 11:49:52.012987
3	3	Cindy	310dcbbf4cce62f762a2aaa148d556bd	2023-05-28 11:50:07.270942
4	4	David	550a141f12de6341fba65b0ad0433500	2023-05-28 12:00:22.467578

Figure 11: Current “User” Table

Figure 12: Register a new account with username Frank. Successfully registered.

1	<code>Select * from "User";</code>
Data Output    Messages    Notifications 	

Figure 13: Check the updated “User” table and there is a new user named Frank

### 3.1.3 Log in

Only when both of the steps finish correctly can the user log in successfully. Screenshots:

Figure 14: Log in with wrong password



Figure 15: Log in with nonexistent username



Figure 16: Log in with correct username and password

### 3.1.4 get\_user\_info

We select the seller's name from the “User” table. Then we select the number of goods sold by the seller from a join table of “Goods” and “Order”. When counting the number of goods sold by the user, we rule out the goods with order state applied or off-sale, since the order can be established by several buyers and the goods can only be sold to one buyer eventually.

Screenshots:



Figure 17: Show information of the seller

## 3.2 Goods module

### 3.2.1 Create goods

Screenshots:

1 SELECT * FROM "Goods";				
Data Output Messages Notifications				
	id [PK] integer	name character varying (255)	description character varying (1024)	img character varying (255)
1	4	数据库大作业答案 (进...	心动不如行动！！！！	/static/images/92a2449b49ac45589b41198e0c017e68.d...
2	3	Concert_ticket	Do you want to go to MAYDAY concerts?!	/static/images/9f7e031aed254b228c635bea4247b500.m...
3	5	Candy	Soooooo sweet!	/static/images/6732d5819921497caf96fb15dcb10f31_64...

Figure 18: Current “Goods” table

Figure 19: Add new goods

Check the table:

1 SELECT * FROM "Goods";				
Data Output Messages Notifications				
	id [PK] integer	name character varying (255)	description character varying (1024)	img character varying (255)
1	4	数据库大作业答案 (进...	心动不如行动！！！！	/static/images/92a2449b49ac45589b41198e0c017e68.d...
2	3	Concert_ticket	Do you want to go to MAYDAY concerts?!	/static/images/9f7e031aed254b228c635bea4247b500.m...
3	5	Candy	Soooooo sweet!	/static/images/6732d5819921497caf96fb15dcb10f31_64...
4	6	T-shirt	Fashionable	/static/images/cart.jpeg

Figure 20: Updated “Goods” table with a new goods T-shirt

### 3.2.2 Update goods

When the user didn't choose a new image, we will not update the image information to the default picture. Instead, we leave the picture as the current one. Screenshots:

Figure 21: Update goods with image changed

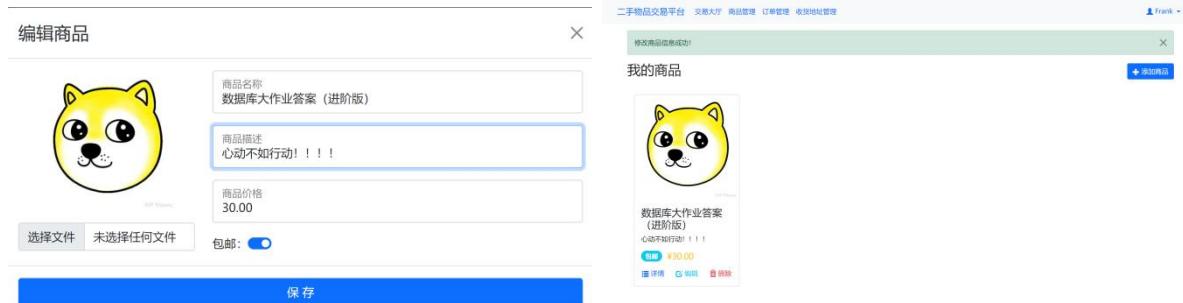


Figure 22: Update goods without image changed

### 3.2.3 Get goods list

We can either select with type or keywords. According to the user's browsing keyword and selected types, we add the corresponding to the query statement. Screenshots:

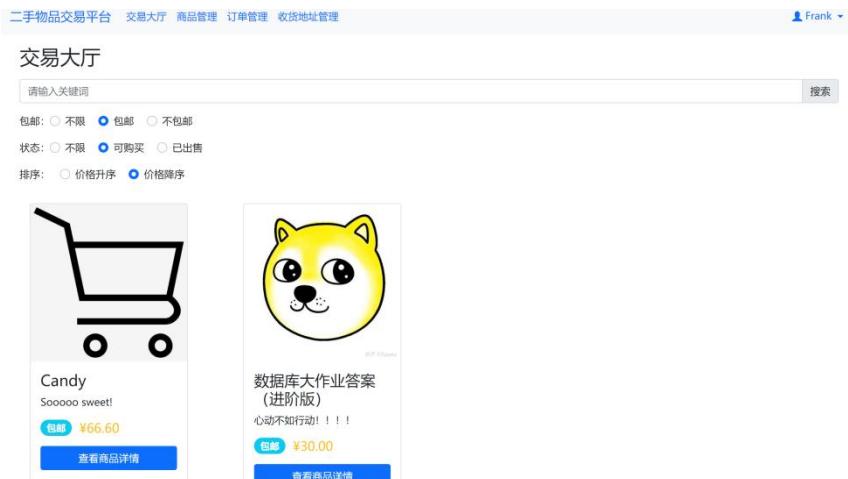


Figure 23: Select goods with types



Figure 24: Select goods with keywords and types

### 3.2.4 Get goods for a specific user

According to the user's id, we select the goods from the where the goods' owner is the user.

Screenshots:



Figure 25: Show goods of a user

From pgAdmin:

Data Output														
	id	[PK] integer	name	character varying (255)	description	character varying (1024)	img	character varying (255)	price	numeric (20,2)	owner	integer	exempt_postage	boolean
1		4	数据库大作业答案	心动不如行动		/static/images/5230f5a5888f4ab79ecc967551141ac1_cart.j...			50.00		5	true		

Figure 26: Corresponding results in pgAdmin

### 3.2.5 Delete goods

We do this by deleting the goods with corresponding goods id and owner. Screenshots:



Figure 27: Delete goods ipad for user Frank

Data Output														
	id	[PK] integer	name	character varying (255)	description	character varying (1024)	img	character varying (255)	price	numeric (20,2)	owner	integer	exempt_postage	boolean
1		4	数据库大作业答案	心动不如行动		/static/images/5230f5a5888f4ab79ecc967551141ac1_cart.j...			50.00		5	true		

Figure 28: Corresponding results in pgAdmin

### 3.2.6 Get goods detail

We select the goods information from the join table of “Goods” and “Order”.

Especially, when trying to determine whether the goods is off-sale or not, we check the order state related to this goods. If all of the order state is none or applied, then the goods is not off-sale. Otherwise, it's off-sale. Screenshots:



Figure 29: Show detail of an unsold goods



Figure 30: Show detail of a sold goods

### 3.3 Order module

#### 3.3.1 Create an order

Screenshots:

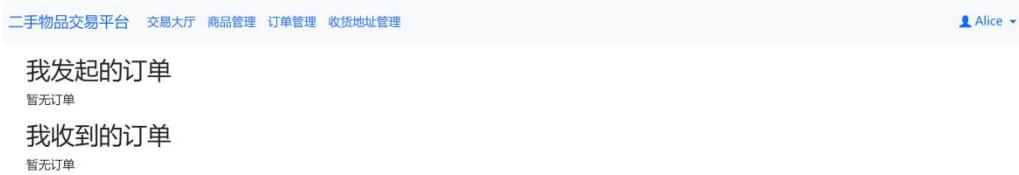


Figure 31: Before create an order



Figure 32: Apply for goods



Figure 33: Successfully create an order

#### 3.3.2 Abandon an order

Screenshots:

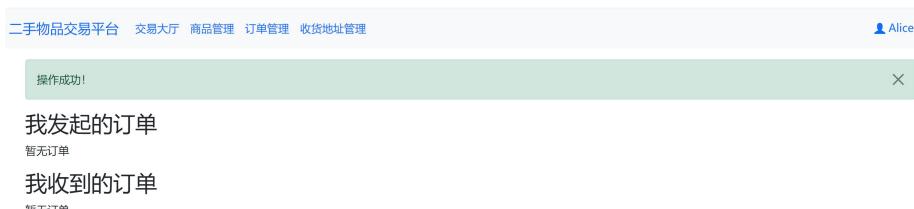


Figure 34: Successfully abandon the order we had just created

### 3.3.3 Approve an order

If owner decides to sell goods to a certain applicant, according to the goods' id and order's id, we set the related order's state to be “APPROVED”. Meanwhile, if there are other applicants applying for the same goods, we will set their orders' state to be “OFF-SALE”. Screenshots:

The screenshot shows the user interface for managing orders. At the top, there are navigation links: 二手物品交易平台, 交易大厅, 商品管理, 订单管理, 收货地址管理, and a user profile for Frank. Below this, there are two sections: "我发起的订单" (My Initiated Orders) and "我收到的订单" (Orders Received). The "我发起的订单" section displays two rows in a table:

商品名称	商品价格	发起人	订单状态	操作
数据库大作业答案 (进阶版)	包邮 ¥30.00	Alice	待同意	同意
数据库大作业答案 (进阶版)	包邮 ¥30.00	Bob	待同意	同意

Figure 35: Before approving an order

The screenshot shows the user interface after one of the orders has been approved. A green success message "操作成功!" (Operation successful!) is displayed at the top. The "我发起的订单" section now shows the status for Alice's order as "待补充信息" (Pending additional information) and for Bob's order as "被买走" (Bought away). The table data is as follows:

商品名称	商品价格	发起人	订单状态	操作
数据库大作业答案 (进阶版)	包邮 ¥30.00	Alice	待补充信息	-
数据库大作业答案 (进阶版)	包邮 ¥30.00	Bob	被买走	-

Figure 36: After approving an order

### 3.3.4 Establish an order

Once the seller approved an application, the buyer can select the address and click to purchase. We update record in the “Order” table, adding the address's id and setting the order state to be“ESTABLISHED”. Screenshots:

The screenshot shows the user interface for managing orders. At the top, there are navigation links: 二手物品交易平台, 交易大厅, 商品管理, 订单管理, 收货地址管理, and a user profile for Alice. Below this, there are two sections: "我发起的订单" (My Initiated Orders) and "我收到的订单" (Orders Received). The "我收到的订单" section displays one row in a table:

商品名称	商品价格	订单状态	操作
数据库大作业答案 (进阶版)	包邮 ¥30.00	待补充信息	地址 Alice  10086 华清   付款 放弃订单

Figure 37: Before establishing an order



Figure 38: After establishing an order

### 3.3.5 Deliver goods

Once the buyer paid the owner can fill in courier information and deliver goods. We use update the information in the related order record, so as to change the order state to be “ON-ROAD”. Screenshots:



Figure 39: Before delivering goods



Figure 40: Fill the courier information

The screenshot shows a user interface for managing orders. At the top, there are navigation links: 二手物品交易平台, 交易大厅, 商品管理, 订单管理, 收货地址管理, and a user profile for Frank. A green success message box displays "操作成功!". Below it, a section titled "我发起的订单" (Orders I Initiated) shows a table with one item: "数据库大作业答案 (进阶版)" at ¥30.00, labeled "被卖出". Another section titled "我收到的订单" (Orders I Received) shows a table with two items: "数据库大作业答案 (进阶版)" at ¥30.00, both labeled "待收货".

Figure 41: Deliver the goods

Buyer's view:

The screenshot shows a user interface for managing orders from Alice's perspective. At the top, there are navigation links: 二手物品交易平台, 交易大厅, 商品管理, 订单管理, 收货地址管理, and a user profile for Alice. A section titled "我发起的订单" (Orders I Initiated) shows a table with one item: "数据库大作业答案 (进阶版)" at ¥30.00, labeled "待收货". Another section titled "我收到的订单" (Orders I Received) shows a table with one item: "数据库大作业答案 (进阶版)" at ¥30.00, labeled "待收货". A modal window titled "快递信息" (Delivery Information) is open, displaying the delivery company as "顺丰" and the tracking number as "PK12306".

Figure 42: Buyer can check the courier information

### 3.3.6 Finish order

The order finishes when the buyer confirms receiving the goods. We update the order state to be “FINISHED”. Screenshots:

The screenshot shows a user interface for managing orders from Alice's perspective. At the top, there are navigation links: 二手物品交易平台, 交易大厅, 商品管理, 订单管理, 收货地址管理, and a user profile for Alice. A section titled "我发起的订单" (Orders I Initiated) shows a table with one item: "数据库大作业答案 (进阶版)" at ¥30.00, labeled "待收货". Another section titled "我收到的订单" (Orders I Received) shows a table with one item: "数据库大作业答案 (进阶版)" at ¥30.00, labeled "待收货".

Figure 43: Before confirming receiving the goods

The screenshot shows a user interface for a second-hand item trading platform. At the top, there are navigation links: 二手物品交易平台, 交易大厅, 商品管理, 订单管理, 收货地址管理. On the right, it says Alice. A green success message box says '操作成功!'. Below it, there are two sections: '我发起的订单' (Orders I Initiated) and '我收到的订单' (Orders I Received). The '我发起的订单' section contains one item: '数据库大作业答案 (进阶版)' with a price of '¥30.00' and a status of '已完成' (Completed). The '我收到的订单' section says '暂无订单' (No orders received).

Figure 44: After confirming receiving the goods

### 3.3.7 Get order by user and goods

When viewing the goods detailed information page, user can check whether he/she has applied to this goods through this function. Screenshots:

This screenshot shows a product detail page for a 'Concert\_ticket'. The product image is a black background with white text 'May is 怪兽 信 传说 沙冠 你 Day 石头 and You'. The title is 'Concert\_ticket', the price is '¥800.00', and the description is '商品描述: Do you want to go to MAYDAY concerts?!'. It also mentions '由Cindy (已卖出0件商品) 发布, 0人想要.' (Posted by Cindy, 0 items sold, 0 people want it). There is a yellow button labeled '♥ 想要' (Want it).

Figure 45: Before applying for goods

This screenshot shows the same product detail page for a 'Concert\_ticket' as in Figure 45, but with a different button. The yellow button now says '♥ 已想要. 点击取消' (Already want it. Click to cancel). This indicates that the user has applied for the item.

Figure 46: After applying for goods

### 3.3.8 Get order information for a specific user

Screenshots:

This screenshot shows a user's order history. At the top, it says '我发起的订单' (Orders I Initiated). Below is a table of orders:

商品名称	商品价格	订单状态	操作
数据库大作业答案 (进阶版)	¥30.00	被买走	-
Concert_ticket	¥800.00	待同意	放弃申请
Candy	¥66.00	待同意	放弃申请

Below the table, it says '我收到的订单' (Orders I Received) and '暂无订单' (No orders received).

Figure 47: User can find orders sent by him/her

### 3.3.9 Get order information to a specific user

Screenshots:

我发起的订单				
商品名称	商品价格	订单状态	操作	
作业库大作业答案 (进阶版)	¥30.00	已卖出	被卖出	-
Concert_ticket	¥800.00	待同意	待同意	放弃申请
Candy	¥66.00	待同意	待同意	放弃申请

我收到的订单				
商品名称	商品价格	发起人	订单状态	操作
T-shirt	¥100.00	David	待同意	同意

Figure 48: User can find orders sent to him/her

### 3.4 Comment module

#### 3.4.1 Create a comment

Screenshots:

Concert\_ticket  
¥800.00

商品描述: Do you want to go to MAYDAY concerts?!

由Cindy (已卖出0件商品) 发布, 1人想要.

商品评论 (0)

Wow, sounds amazing!!!

时间升序

暂无评论

Figure 49: User tries to comment on goods

添加评论成功!

Concert\_ticket  
¥800.00

商品描述: Do you want to go to MAYDAY concerts?!

由Cindy (已卖出0件商品) 发布, 1人想要.

商品评论 (1)

请文明发言

Frank  
Wow, sounds amazing!!!  
发表于: 2023-06-24 17:14:20

时间升序

Figure 50: User comments on goods successfully

#### 3.4.2 Get comments for a specific goods

The user can choose to view the comment whether by time ascending order or descending order.

Especially, if one comment is posted by the owner of the goods or the user who has bought the goods, their identities will be highlighted. This function will be further discussed in the

optimization part. Screenshots:

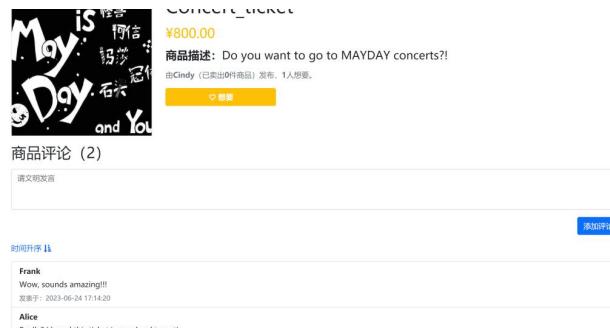


Figure 51: View comments in time ascending order

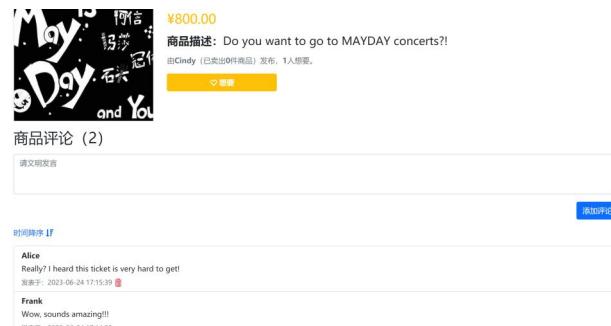


Figure 52: View comments in time descending order

### 3.4.3 Delete a comment

Screenshots:

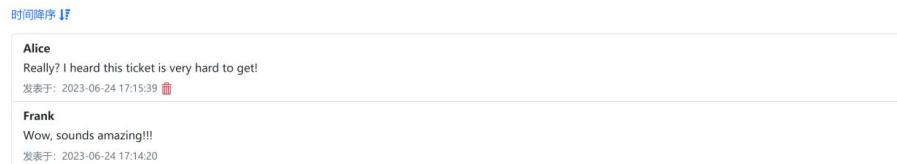


Figure 53: Current comments

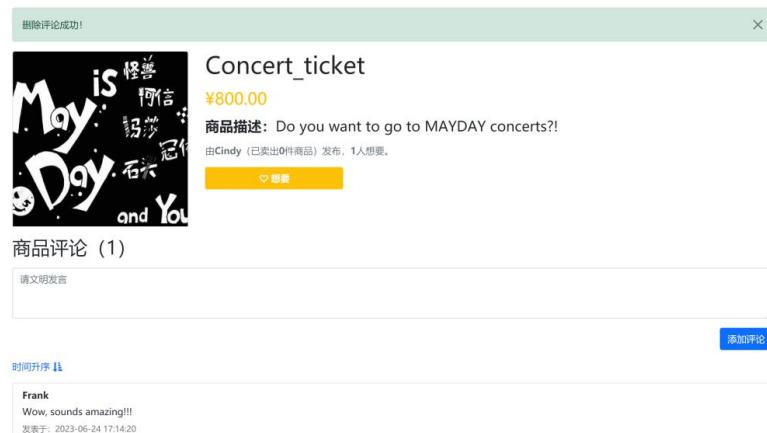


Figure 53: Successfully delete a comment

## 3.5 Address Module

### 3.5.1 Create address

Screenshots:

我的收货地址列表			
收件人	手机号	地址	操作
Alice	10086	华清大学紫荆学生公寓	

Figure 54: Before creating a new address

我的收货地址列表			
收件人	手机号	地址	操作
Alice	10086	华清大学紫荆学生公寓	
Alice	10086	北京大学博雅塔	

Figure 55: After creating a new address

### 3.5.2 Get address list of a specific user

Screenshots:

我的收货地址列表			
收件人	手机号	地址	操作
Alice	10086	华清大学紫荆学生公寓	
Alice	10086	北京大学博雅塔	

Figure 56: View address list

### 3.5.3 Delete address

Screenshots:

我的收货地址列表			
收件人	手机号	地址	操作
Alice	10086	华清大学紫荆学生公寓	
Alice	10086	北京大学博雅塔	

Figure 57: Before deleting address



Figure 58: After deleting an address

## 4 Optimization

### 4.1 Adding constraints on the password when signing in

Although we use md5 when storing the passwords of users, the safety of an account cannot be guaranteed if the password is too short, which means it's easy to guess.

As a result, we add a constrain that the length of the password shouldn't be less than 8 characters when creating a new account.

We achieve this by adding a function in db\_api.py to check whether the length of the password is less than 8. Then we change the code structure of signup( ) function in main.py. When detecting that the password is less than 8, it will return an error message: “密码不能少于 8 位!”

```
def check_password(password):
    if len(password) < 8:
        return True
    else:
        return False
```

Figure 59: Related codes in db\_api.py

```
@app.route('/signup', methods=['GET', 'POST'])
def signup():
    error = None
    if request.method == 'POST':
        username = request.form.get('username', '')
        password = request.form.get('password', '')
        if db_api.check_username_used(username):
            error = '该用户名已被注册!'
        elif db_api.check_password(password):
            error = '密码不能少于8位!'
        else:
            if db_api.create_user(username, password):
                return redirect(url_for('login') + '?success=注册成功! 请登录。')
            else:
                error = '注册失败!'
    return render_template('signup.html', error=error)
```

Figure 60: Related codes in main.py

The figure consists of two side-by-side screenshots of a web application. Both screenshots show a registration form titled "注册二手物品交易平台". The left screenshot shows a successful registration attempt. The right screenshot shows an error message for a password length of 3 characters.

**Left Screenshot (Successful Registration):**

- Username: Jack (Valid)
- Password: ... (Valid)
- Confirm Password: ... (Valid)
- Feedback: 已有账号? 去登录。
- Register Button: 注册 (Enabled)

**Right Screenshot (Error Message):**

- Username: (Empty, Error: 请输入用户名!)
- Password: (Empty, Error: 请输入密码!)
- Confirm Password: (Empty, Error: 请再次输入密码!)
- Error Message: 错误! 密码不能少于8位! (Error: Password must be at least 8 characters long!)
- Feedback: 已有账号? 去登录。
- Register Button: 注册 (Disabled)

Figure 61: When signing up with the length of password less than 8, an error message will show up

## 4.2 Optimizing on presenting the publisher of comments

Users may want get a objective comments when browsing the information of goods. As a result, it's quite useful to identify the seller's comments as well as the comments of the users who has bought the goods.

We achieve this by changing the structure of the query statement in the corresponding function in db\_api.py. We compare the user\_id with the seller's and the buyer's id to see if the identity of publisher needs to be highlighted.

```
# TODO: 请实现该函数的功能。
conn = get_db_conn()
cursor = conn.cursor()

# 获取卖家id
cursor.execute('SELECT owner FROM "Goods" WHERE id={0}'.format(goods_id))
seller = cursor.fetchone()

# 获取已经成功购买的用户id
cursor.execute('SELECT user_id From "Order" WHERE goods_id={0} AND state=5'.format(goods_id))
buyer = cursor.fetchone()

# 获取商品评论
query = 'SELECT * FROM "Comment" LEFT OUTER JOIN "User" ON ("Comment".user_id="User".id) WHERE goods_id={0} '\
    .format(goods_id)
if order == 'asc':
    query += ' ORDER BY "Comment".create_at ASC'
else:
    query += ' ORDER BY "Comment".create_at DESC'
cursor.execute(query)
comment_list = []
rows = cursor.fetchall()
for row in rows:
    if seller[0] == row[1]:
        content = "*此为卖家发言* " + str(row[3])
    elif buyer is not None:
        if buyer[0] == row[1]:
            content = "*此为已购买的买家发言* " + str(row[3])
        else:
            content = str(row[3])
    info = {
        'id': row[0],
        'user_id': row[1],
        'username': row[6],
        'content': content,
        'create_at': str(row[4])[:19] # 19是将时间显示精度限定在整数秒
    }
    comment_list.append(info)
conn.close()
print(goods_id, order)
return comment_list
```

Figure 62: Related codes in db\_api.py



Figure 63: We can see the identity of the buyer and the seller

### 4.3 Enhancing the stability when creating comments

When users create with single quotation, the quotation mark may cause trouble the SQL query executor when the filed property is VARCHAR, .

To deal with this problem, we replace all the single quotation marks with two single quotation marks as in the following codes. In PostgreSQL's query statement, two single quotation marks will be eventually identified as one mark and stored successfully, allowing users to use quotations freely without causing confusion in storing the information.

```
# TODO: 请实现该函数的功能。
conn = get_db_conn()
cursor = conn.cursor()
success = True
try:
    cursor.execute('SELECT max(id) FROM "Comment"')
    max_id = cursor.fetchone()
    if max_id[0] is None:
        n = 1
    else:
        n = max_id[0] + 1
    content = content.replace("'", "''") # sql语句中单引号嵌套单引号
    query = 'INSERT INTO "Comment" VALUES({0},{1},{2},'{3}')'.format(n, user_id, goods_id, content)
    cursor.execute(query)
    conn.commit()
except Exception as err:
    print(err)
    success = False
conn.close()
return success
```

Figure 64: Related code in db\_api.py

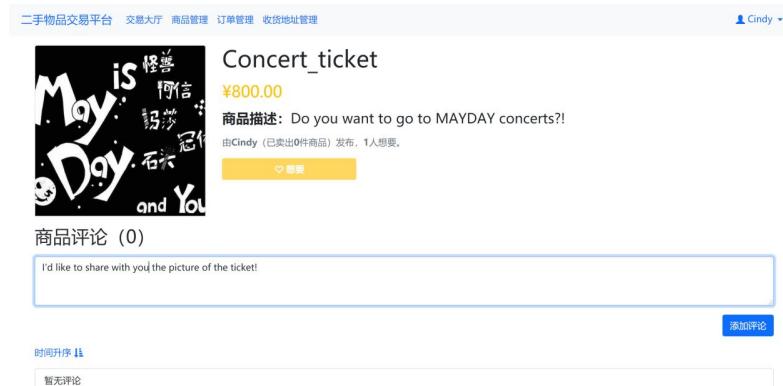


Figure 65: Create comment with single quotation

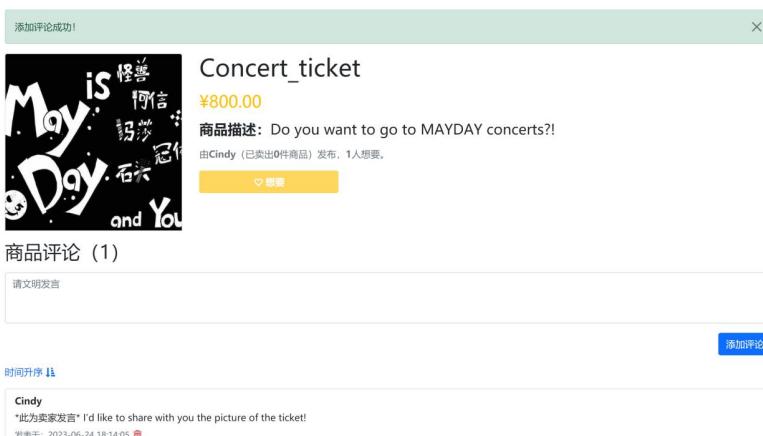


Figure 66: Create comment with single quotation successfully

## 5 Summary & Reflection

To sum up, the trading platform is designed and built based on users' need and we try to provide better experience for users.

Through the project of second-hand goods trading platform, I practiced the knowledge I learnt in class in a more systematic way rather than writing single statement separately as we did in daily SQL exercise. The project hugely improved my ability to establish a database based on the ERD , to use SQL statement, as well as to build connection between Python and PostgreSQL connection.

Optimizing the project let me learn more about back-end and front-end knowledge. And I learned that behind every little improvement is countless efforts and time spent. Apparently, the platform can still be optimized in multiple ways and more functions could be added, such as responding to other comments, setting default address etc. So I still have a long way to go in further study.

I did gain a sense of satisfaction when finishing the project, and I'm longing for professor and TA's feedback and advice. Many thanks to the professor and TAs for teaching and instructing me both in and out of course!

## 6 Suggestions w.r.t. the course

The content of the lectures throughout the whole semester is really useful, and the TAs are quite really helpful whenever I reach out for help. And I do like that the discussion session, which helps me review what I've learnt in class.

As for suggestions, I'd like to suggest that we can explain the part of database connectivity more explicitly, which was quite confusing for me at the beginning. But the course work really helps me to understand that part more.