Nowadays, almost everyone has something they don't need. Instead of leaving such unused items undisposed, selling them to those in need is a better choice. In order to build a bridge between the sellers and the buyers, a second-hand goods trading platform is developed to allow the sellers to upload goods and allow the buyers to browse and purchase goods. However, due to some database issues, the platform can not work well. As an outstanding student taking the Database Concepts course, you are invited to design a database for the platform to support its normal operations.

Notes

- For some commercial considerations, the payment function and delivery function of the platform have been simplified.
- Each of the goods can only be bought by one customer.

Tasks

- Design a database for the platform that allows a person to
    - Join the platform with username and password. (Note that a username can only be registered for one person.)
    - Manage the goods
        - Upload goods specified by name, description, image, price and postage;
        - Browse, search, filter and sort goods;
        - Modify goods information;
        - Delete goods uploaded by him/her.
    - Manage the orders
        - (Buyers) Apply to buy goods;
        - (Sellers) Approve the applications;
        - (Buyers) Provide delivery address for an order and pay for the goods;
        - (Sellers) Deliver the goods, and provide the name of the courier services company as well as the courier number;
        - (Buyers) Confirm receiving the goods and finish the order.
    - Manage the comments
        - Comment on goods;
        - Browse comments;
        - Delete comments posted by him/her.
    - Manage the delivery address
        - Add a new delivery address;
        - Browse his/her delivery addresses;
        - Delete his/her delivery addresses.
- Given a semi-finished platform and a physical model of its database,
    - Give the DDL statements for the physical model;
    - Create the database in a PostgreSQL server;
    - Implement the 25 interfaces of the platform to make it work;
    - Optimize the platform in any way (e.g., pagination).

Here is an example for one of the interfaces.

```python
def create_user(username, password):
    """
    功能：在数据库中创建用户，用于实现注册功能。
    分值：2分
    :param username: 用户名称
    :param password: 用户明文密码，建议存储时使用md5加密
    :return: 如果创建成功，返回True；否则返回False。
    """
    # TODO：请实现该函数的功能。
    print(username, password)
    return True
```

Deliverables

- ERD and relational schemas (15')
- Source code (75')
  - DDL statements (5')
  - Database interface implementation for
    - User module: 4 interfaces (2 + 2 + 2 + 3 = 9')
    - Goods module: 6 interfaces (2 + 2 + 5 + 3 + 2 + 3 = 17')
    - Order module: 9 interfaces (2 + 3 + 3 + 2 + 2 + 2 + 2 + 4 + 5 = 25'),
    - Comment module: 3 interfaces (2 + 3 + 2 = 7')
    - Address module: 3 interfaces (2 + 3 + 2 = 7')
  - Any optimization (5')
- Documentation (10')
  - System flowchart
  - Details on the database interface implementation and any optimization
  - Screenshots on
    - Operations on the platform
    - Schemas in PostgreSQL
  - Summary
  - Suggestions w.r.t. the course work and the class

Important Dates

- ERD and relational schemas due: **April 29, 2022** (Friday)
- Source code and documentation due: **June 12, 2022** (Sunday)