Firstly, database and its tables were created according to the task. Then those were populated with recommended data. After that we wrote accountTransactions procedure which takes an account ID to get its transactions. Also, we have deposit and withdraw procedures which both take account ID and amount. If the user exists then we make a transaction of deposit or withdrawal and update corresponding balance in accounts table.

To connect to the database in Python we are using such construction:

```python
def __init__(self,
             host="localhost",
             port="3306",
             database="banks_portal",
             user='root',
             password='1337'):

    self.host = host
    self.port = port
    self.database = database
    self.user = user
    self.password = password
    self.connection = None
    self.cursor = None
    self.connect()
def connect(self):
    try:
        self.connection = mysql.connector.connect(
            host=self.host,
            port=self.port,
            database=self.database,
            user=self.user,
            password=self.password)

        if self.connection.is_connected():
            return
    except Error as e:
        print("Error while connecting to MySQL", e)
```

where we specify these credentials.

Next, let’s talk about methods:

For instance, the getAllAccounts method fetches all account records from the database and returns them in the form of a list of tuples. Similarly, the getAllTransactions method retrieves all transaction records from the database and returns them in the form of a list of tuples.

The deposit and withdraw methods execute transactions on a specified account. They invoke stored procedures called 'deposit' and 'withdraw', respectively, and provide the account ID and amount to deposit or withdraw as arguments. The methods then print the outcome produced by the stored procedure and commit the transaction.

The addAccount method adds a new account record to the database by using an SQL query with the INSERT INTO statement and the VALUES keyword. It then outputs the count of inserted records and commits the transaction.

The accountTransactions method retrieves all transaction records for a specified account by calling a stored procedure called 'accountTransactions' with the account ID as an argument. It adds the results to a list and returns the list.

Lastly, the deleteAccount method eliminates an account record from both the accounts and transactions tables with the use of an SQL query that includes the DELETE FROM statement. It then commits the transaction after deleting the record.

As for portalServer.py, we have routing there. Mostly, in get requests, we get some information from the user to perform a certain action, and then this information is used when processing the post request.