Banking System
Arrenn Baral, Prakriti Basyal
Team 10
CS 157A Summer 2020
Instructor Mike Wu

Project Description

This project will be a database application that is based in the general banking system. The goal of this application is to provide a convenient way to access user's banking information. Users will be able to login with designated username and password, access bank account, deposit, withdraw, get a report of transaction if a user wishes to. This application will also provide a user-friendly interface that is clear and intuitive. We use bank system in our daily lives. We login into bank systems like Chase, Wells Fargo, Bank of America, and list goes on. Deposits, withdraws, transfers, statements are some of the example of our daily used operations on the bank systems. But we never thought what actually runs behind the screen. Since we are taking database class, we are getting great opportunity to explore system behind the screen. We got the motivation from our daily used operations in the bank system. Let's start with simple structure our structure of the banking system

Word on structure

We will be Apache Tomcat webserver to run our banking system.

We will be implementing our code through Java IDE.

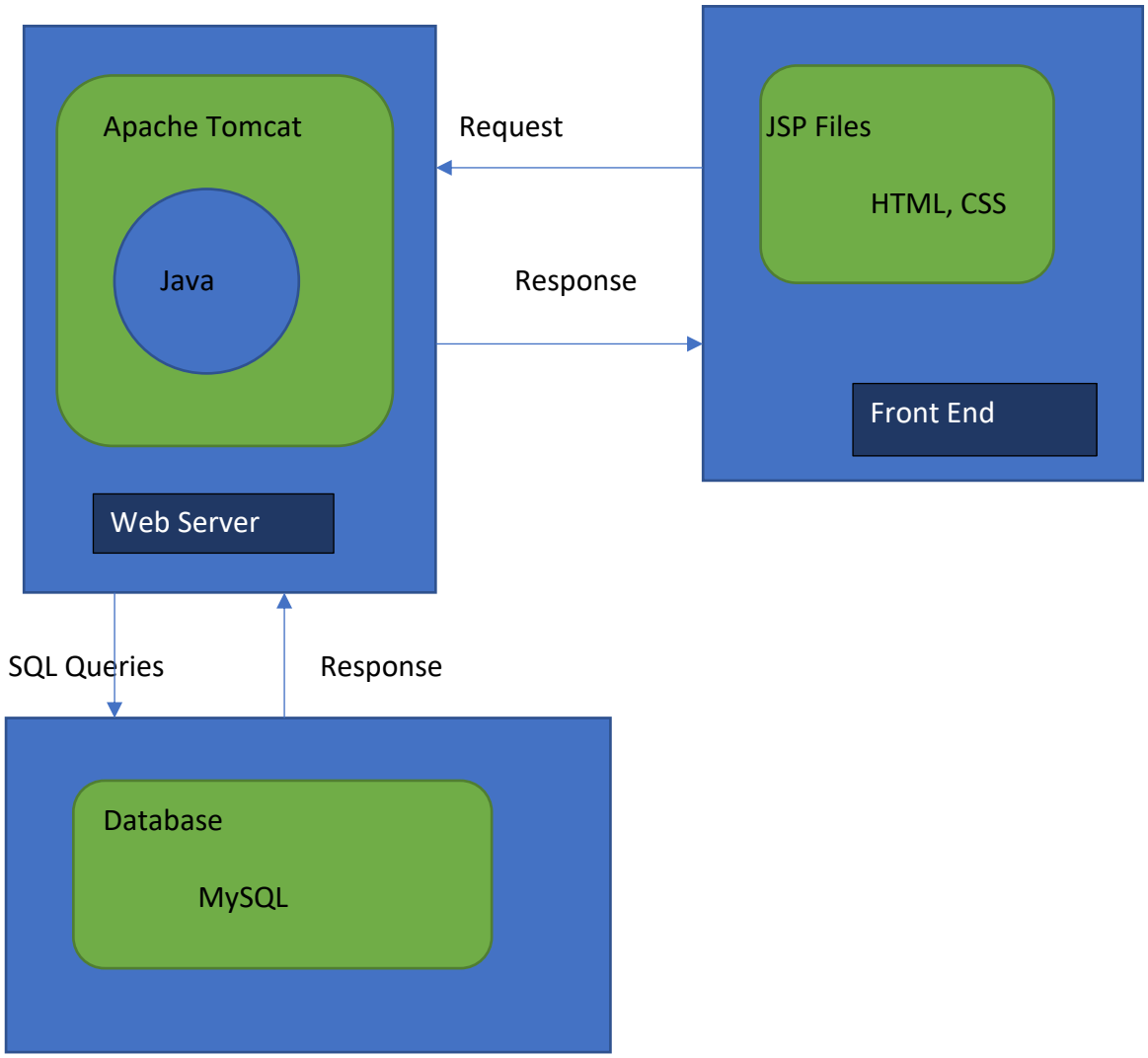
For database system, we are using MySQL but data entries.

We will be using DMLs, and DDLs for operations like deposits, withdrawals, and get statement.

We will be using HTML and CSS for beautify our banking system. It is important to give user-friendly look to users.

System Environment

Structure of the System



Hardware and Software

Apache Tomcat

MacOS

RDMS

MySQL Community Server 8.0.20

Application Languages

Java, HTML, CSS, PHP (maybe), SQL

Functional Requirements

How user will access the system

The system provides functionality to create an account. User will be asked to set up a new account if he/she is new user else simply log in using their existing user credential in the database. If user wishes to setup a new account, the system opens new page for the user where the system asks several information for setting up the account such as First name, Last name, Date of Birth, Email Address, Gender, username, and password. Once the setup process completed, user will be redirected to banking system where they can select different operations. User cannot login if user's credential is not saved in the database. User cannot access without setting up the account. If user tries to access with fake username and password, they will be redirected to login page to use valid username or password. However, once user logs in, system will not ask to sign in again. System will assume user logged in through trusted device. Once logged in, users will be able to use several functions.

Functions

Log in

User shall set up an account to get an access to the banking system.

User's credentials must be on saved in the database.

Withdraw

Once logged in, user shall select the withdraw option to get their money.

Once user chooses this function, the withdrawal sum will be deducted from the total amount.

Deposit

Once logged in, user shall use this function to deposit the money.

Once deposited, the deposited sum will be added to the total amount.

Statement

User shall choose this function to get a statement of the transaction after successful withdrawal and deposits.

Apply for Loan

User shall use this function to apply for a loan.

User gets approved only if thousand dollars is available in his bank account.

Order a cheque book

User can use this function to order a cheque book.

There will be a charge of $1.00 for ordering a check.

Manage/update account information

User shall use this function to edit account information such as address,

and phone number.

Transfer fund

User shall use this function to transfer fund to other.

Receiving person much be setup before transferring.

Search statement by date

If user wants to search statement from specific date, they can choose this function to select the date and search.

Sort the transactions

If user wishes to sort the transactions from higher to lower or lower higher, they can use this function to sort as wished.

Delete account.

If user wishes to terminate the account, they can simply use this function to delete the account. Once deleted, account will not be recovered. It will be permanent.

Display Account Information

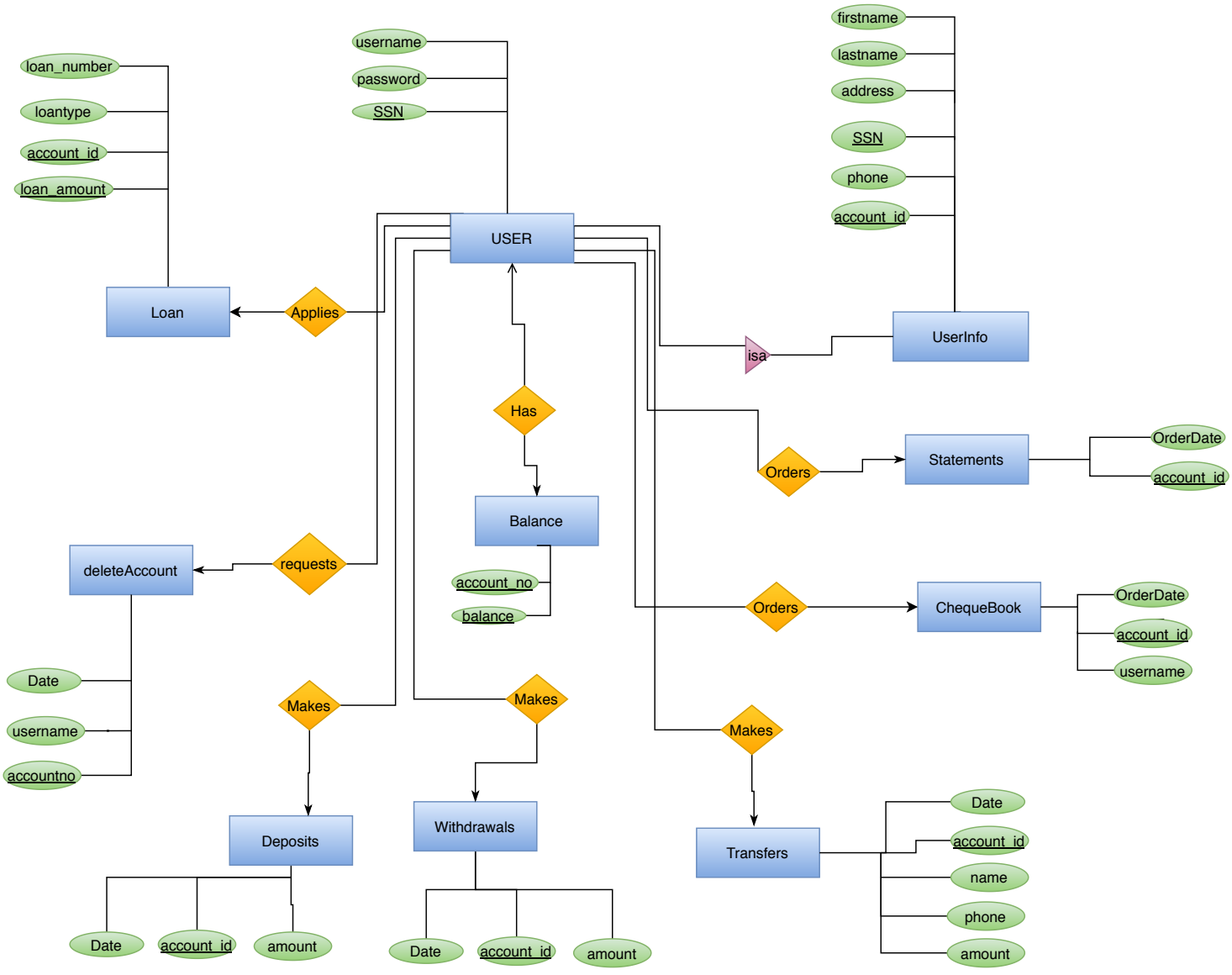
The user will be able to view the account information through this function.

Graphics User Interface (GUI)

For the GUI, we will be using HTML to provide user-friendly environment. If possible, we will allow user to upload a picture of them for the account.

The security of each user account will be protected by a username and password created by the user. The username for each account will not be their email. User name will be the one which they choose while setting up the account. The user account information will be securely stored in the server. In order to login to the system the user must provide an existing username account along with the associated password. The website will be implemented using https providing an encrypted connection between user and server.

ERD Diagram



USERS:

User entity is the service offered by the bank where one may deposit, withdraw or transfer money, check balance, apply loan, check statement in their account information. Every customer has a unique account. The entity set contains the customers username and password to login and the SSN. SSN is the primary key.

USERINFO:

The userInfo entity is the entity that holds the information of the user. This entity represents the unique customer information, which is the customer's firstname, lastname, address, phone, SSN, account_id. The SSN and the account_id is the primary key. It is child of userInfo so it inherits the attributes from the USERS.

BALANCE*:*

The Balance entity is the entity that represents the total amount of the fund in the user's account. Every user has a balance. It has two attributes the account_no and the balance which gives them the information. Account_id is primary key.

LOAN*:*

Loan entity represents lending money to one or more individuals with the required interest rate from the bank. The user can apply for a loan. So the loan entity has the attributes loan number, Loan_type, account_id, loan amount where account_id is primary key.

***DELETEDACCOUNT**:*

This is an entity provided to the user as a service to delete an account. A user can request to delete the account if they no longer are interested in the service. Its attributes are date, username, and account_no where account number is the primary key.

***STATEMENTS**:*

Statement entity represents a record of the financial activities done by the customer which are also given as the written reports. It basically keeps track of the record list of transactions done by the customer over a certain period of time. User can order the statement any time. So, orderdate and the account_id are the attributes where account_id is the primary key.

***CHEQUEBOOK**:*

This entity is another service provided to the customer where user can order the cheque book providing their account_id, and username. OrderDate keeps track when it was ordered.

***DEPOSITS**:*

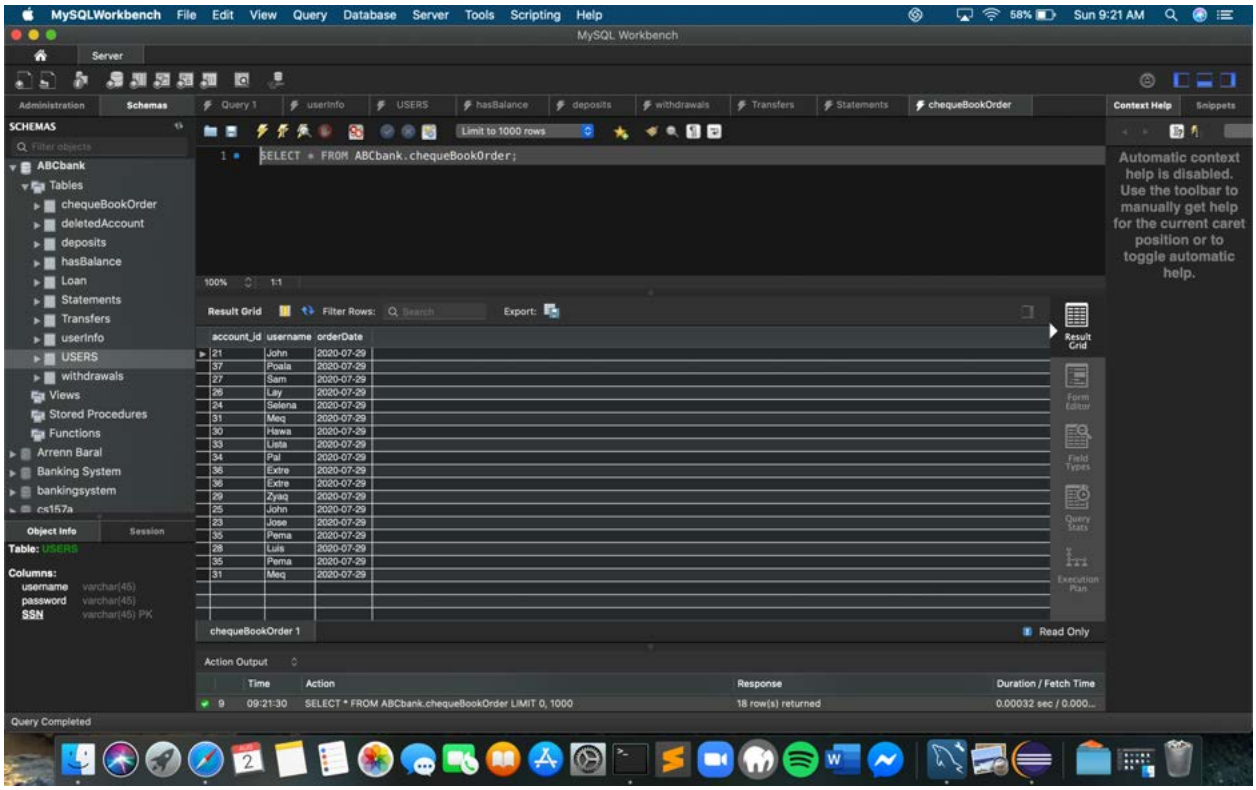
Deposit is entity set which is available to customer to deposit money into their bank account. These deposits can be made through cash, check by the customer. So, for the security of the customer's information. Deposits entity set has the account_id, Date and the amount as an attribute.

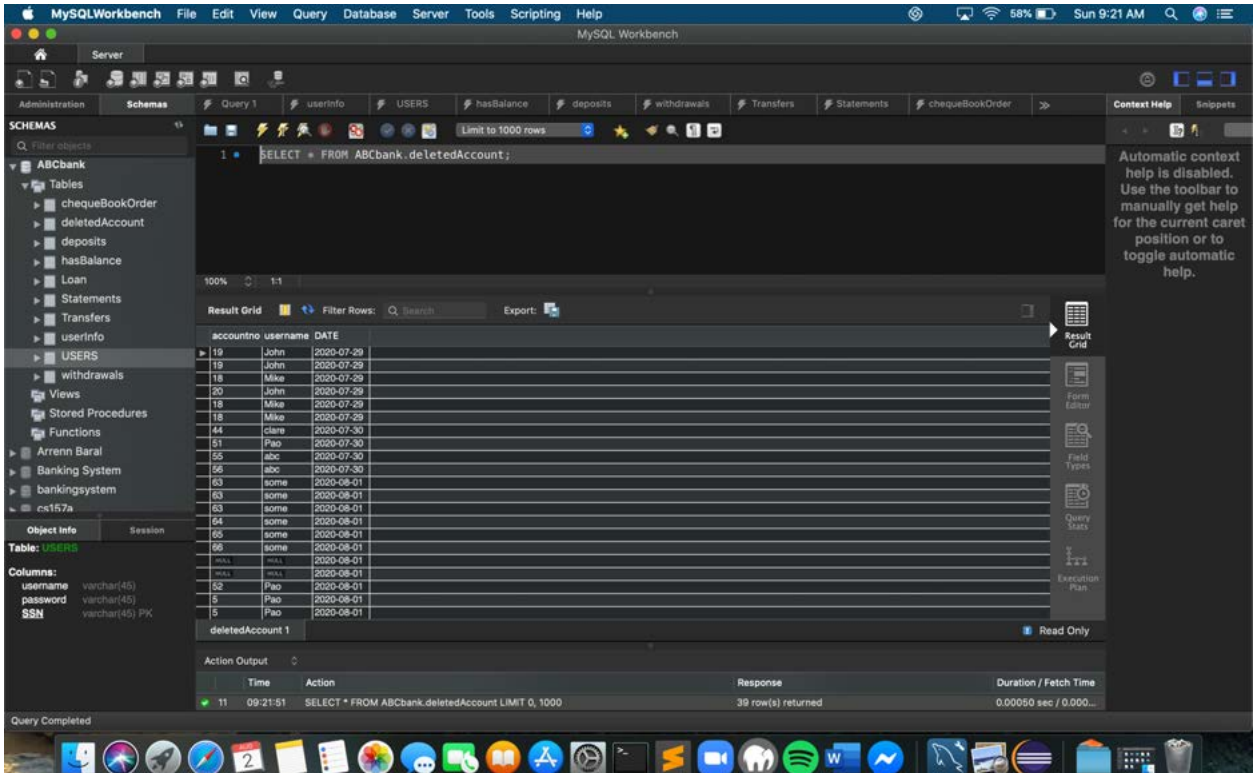
***WITHDRAWALS**:*

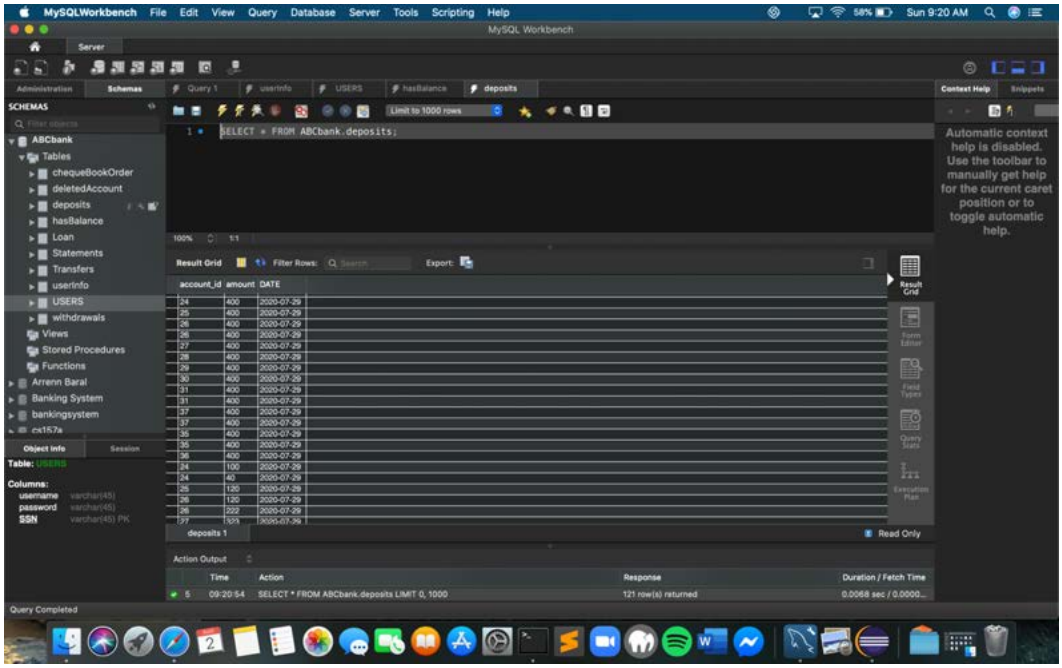
Withdrawal entity set represents giving the customer an option to withdraw money whenever they want. The bank grants access to the customer for the withdrawals except for the unusual activities. This entity set contains withdraw amount, account_id and the DATE to withdraw from their account as an attribute.

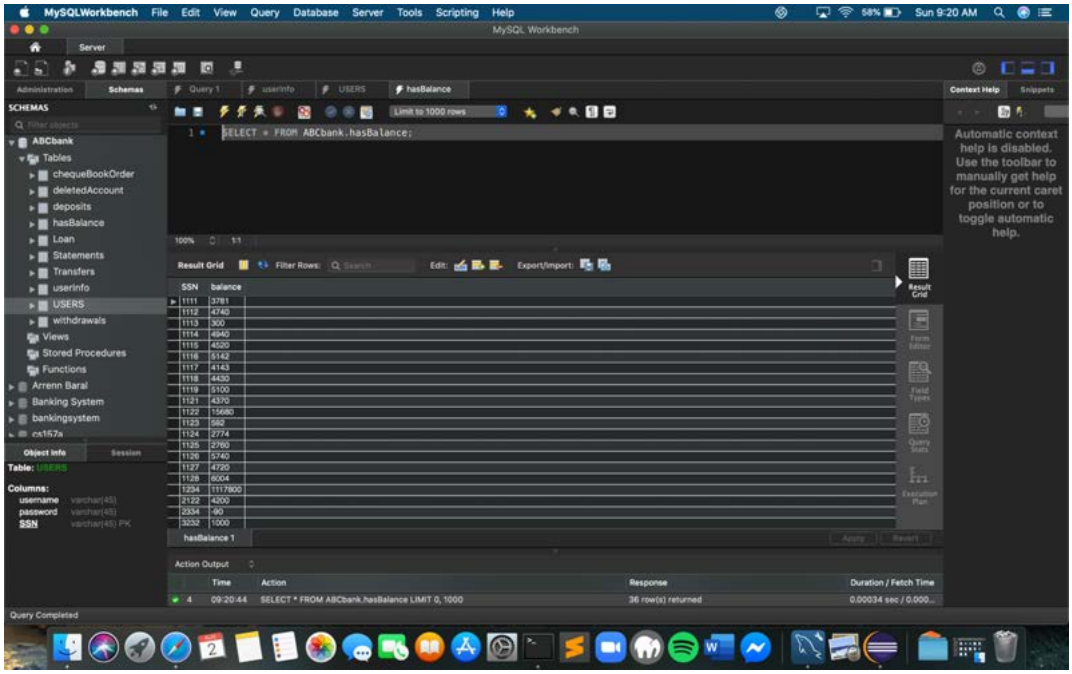
***TRANSFERS**:*

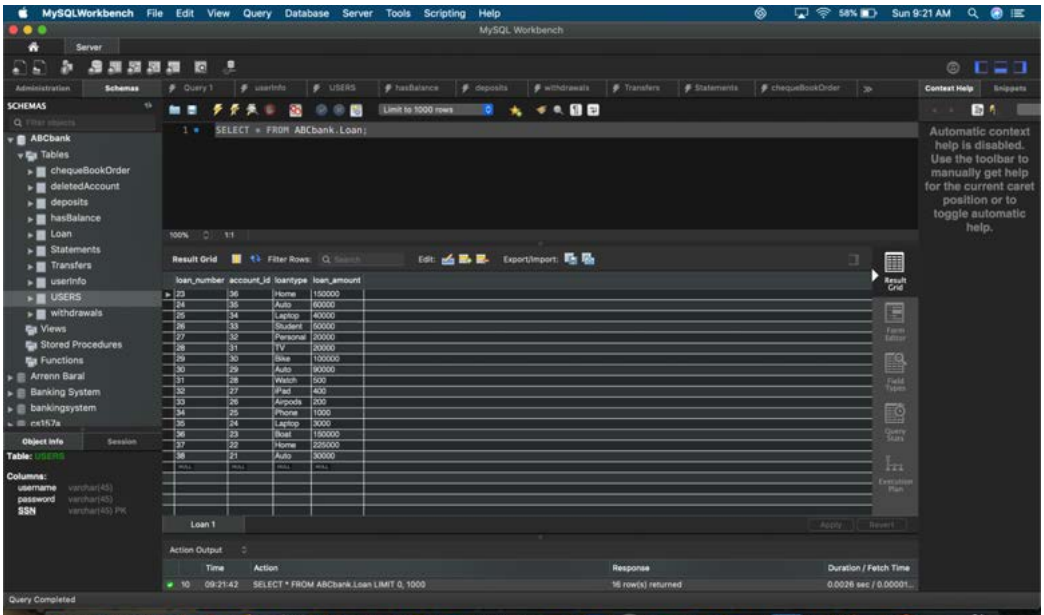
This entity set represents the electronic fund transfer from one person to another to the bank accounts. It is one of the services provided by the bank so that people can transfer, making people's lives easier. This entity set has the amount, date, name, phone number and the account_id so they can transfer directly to their account.

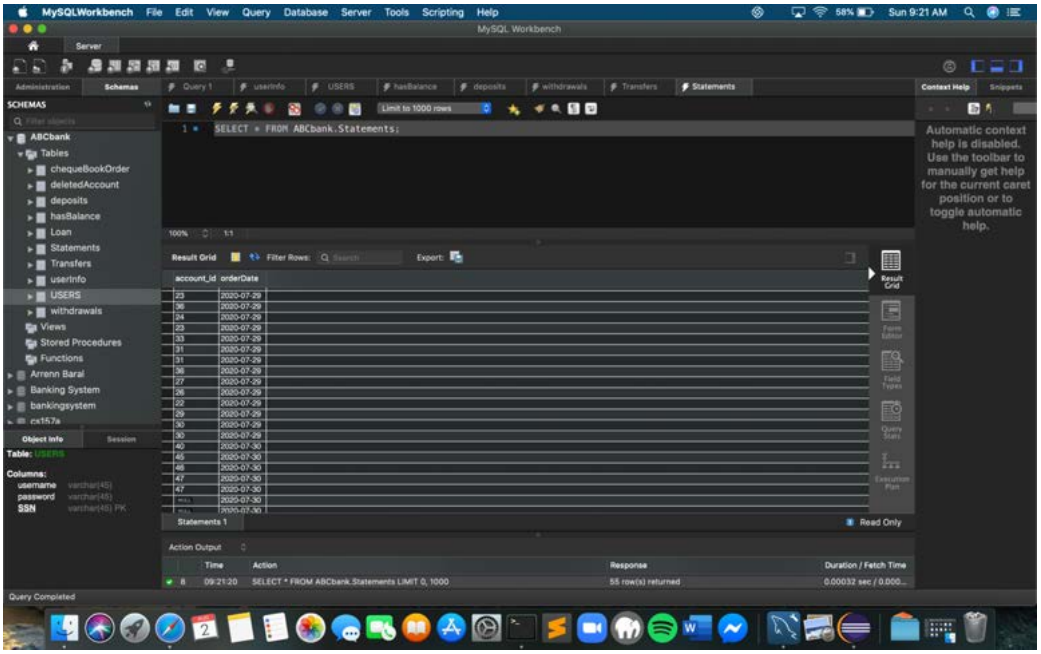


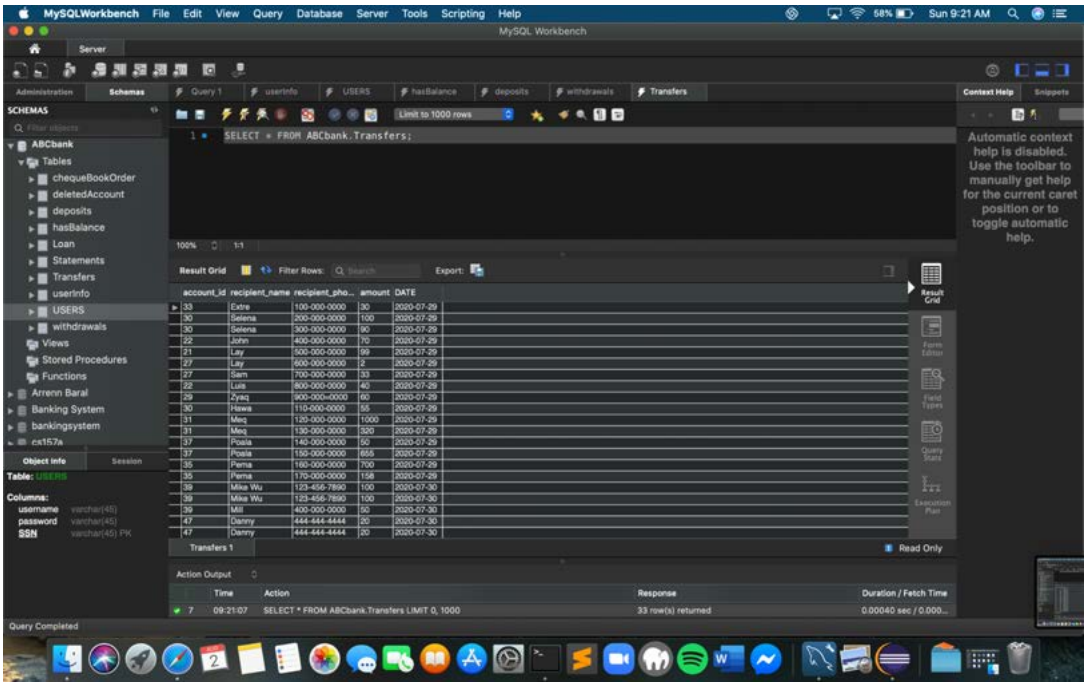


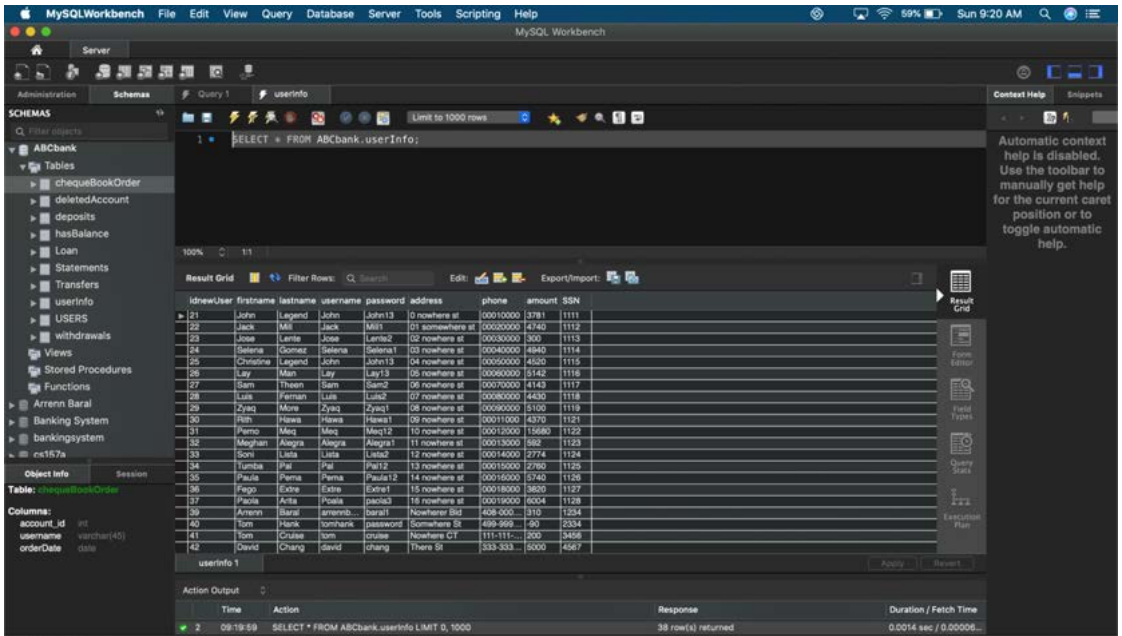

MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
Sun 9:20 AM
MySQL Workbench
SELECT * FROM ABCbank.deposits;
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Result Grid
account_id amount DATE
Action Output
Time Action Response Duration / Fetch Time
09:20:54 SELECT * FROM ABCbank.deposits LIMIT 0, 1000 121 row(s) returned 0.0068 sec / 0.0000...
Query Completed

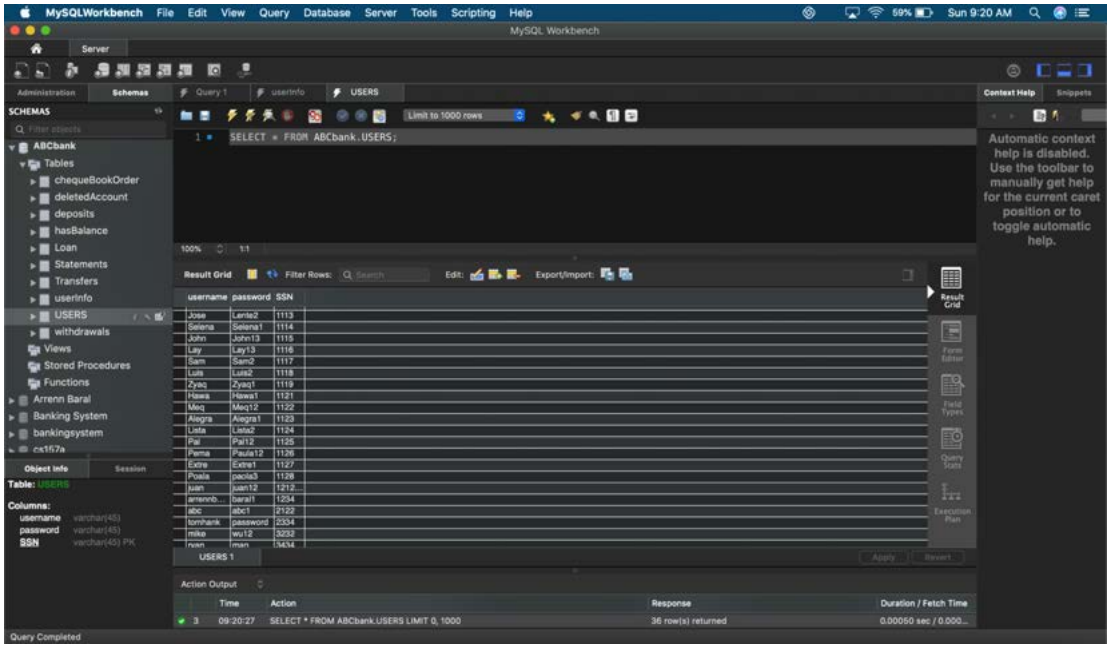

MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
Sun 9:20 AM
MySQL Workbench
SELECT * FROM ABCbank.hasBalance;
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Result Grid
SSN balance
Action Output
Time Action Response Duration / Fetch Time
09:20:44 SELECT * FROM ABCbank.hasBalance LIMIT 0, 1000 36 row(s) returned 0.00034 sec / 0.000...
Query Completed

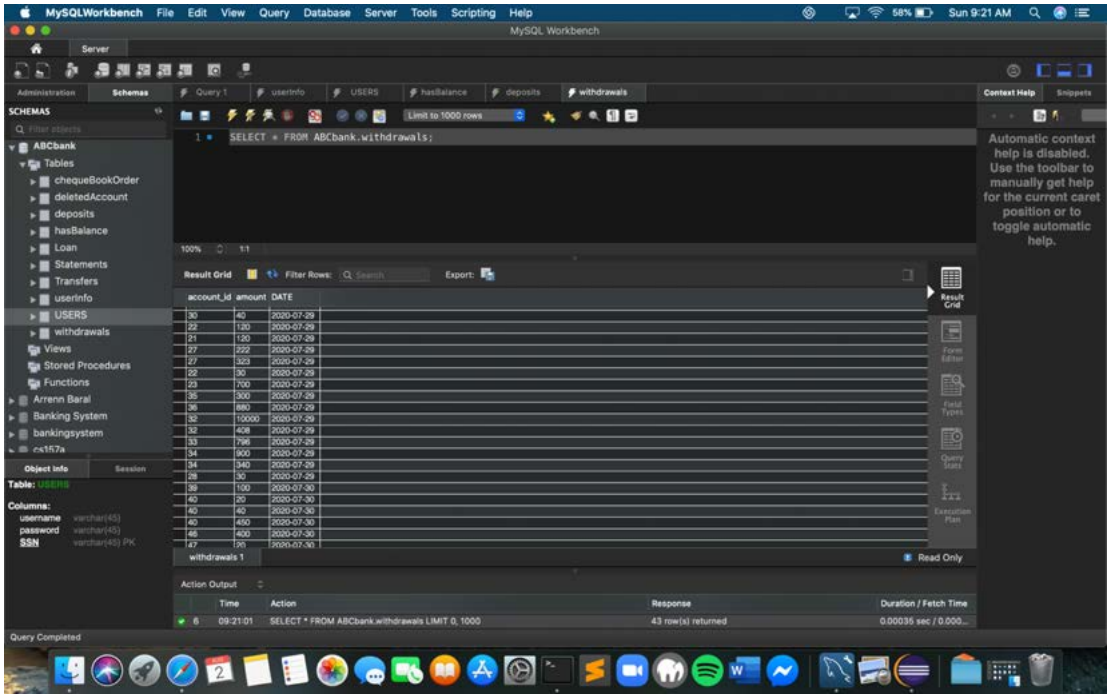

MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
Sun 9:21 AM
MySQL Workbench
SELECT * FROM ABCbank.Loan;
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Result Grid
loan_number account_id loantype loan_amount
Action Output
Time Action Response Duration / Fetch Time
09:21:42 SELECT * FROM ABCbank.Loan LIMIT 0, 1000 16 row(s) returned 0.0026 sec / 0.00001...
Query Completed


MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
Sun 9:21 AM
MySQL Workbench
SELECT * FROM ABCbank.Statements;
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Result Grid
account_id orderDate
Statements 1
Read Only
Action Output
Time Action Response Duration / Fetch Time
09:21:20 SELECT * FROM ABCbank.Statements LIMIT 0, 1000 55 row(s) returned 0.00032 sec / 0.000...
Query Completed

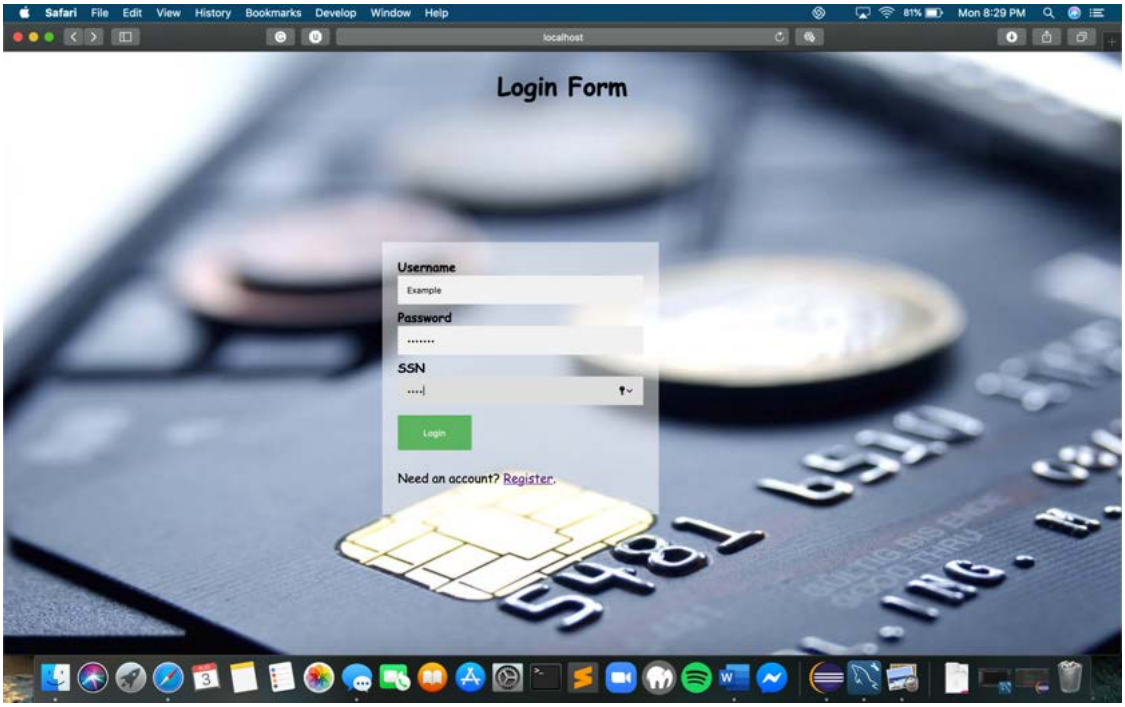

MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
Sun 9:21 AM
MySQL Workbench
SELECT * FROM ABCbank.Transfers;
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Result Grid
account_id recipient_name recipient_pho... amount DATE
Transfers 1
Read Only
Action Output
Time Action Response Duration / Fetch Time
09:21:07 SELECT * FROM ABCbank.Transfers LIMIT 0, 1000 33 row(s) returned 0.00040 sec / 0.000...
Query Completed


MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
Sun 9:20 AM
MySQL Workbench
SELECT * FROM ABCbank.userInfo;
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Result Grid
idnewUser firstname lastname username password address phone amount SSN
userInfo 1
Action Output
Time Action Response Duration / Fetch Time
09:19:59 SELECT * FROM ABCbank.userInfo LIMIT 0, 1000 38 row(s) returned 0.0014 sec / 0.00006...

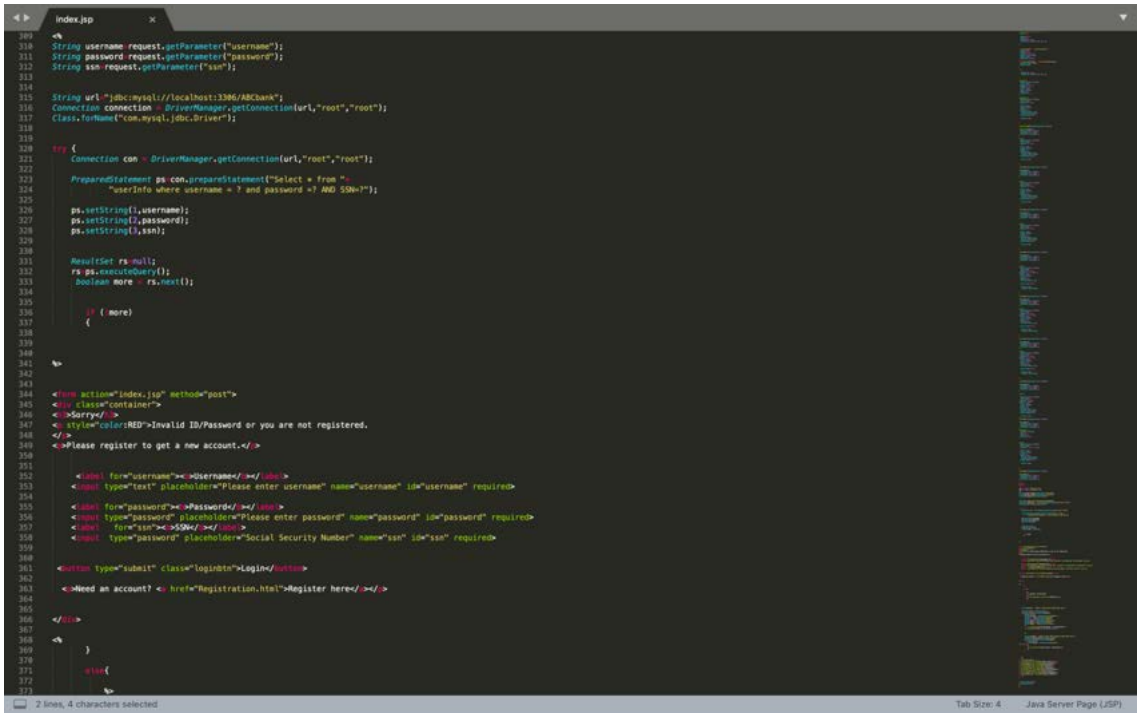

MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
Sun 9:20 AM
MySQL Workbench
SELECT * FROM ABCbank.USERS;
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Query Completed


MySQLWorkbench File Edit View Query Database Server Tools Scripting Help
Sun 9:21 AM
MySQL Workbench
SELECT * FROM ABCbank.withdrawals;
Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.
Query Completed

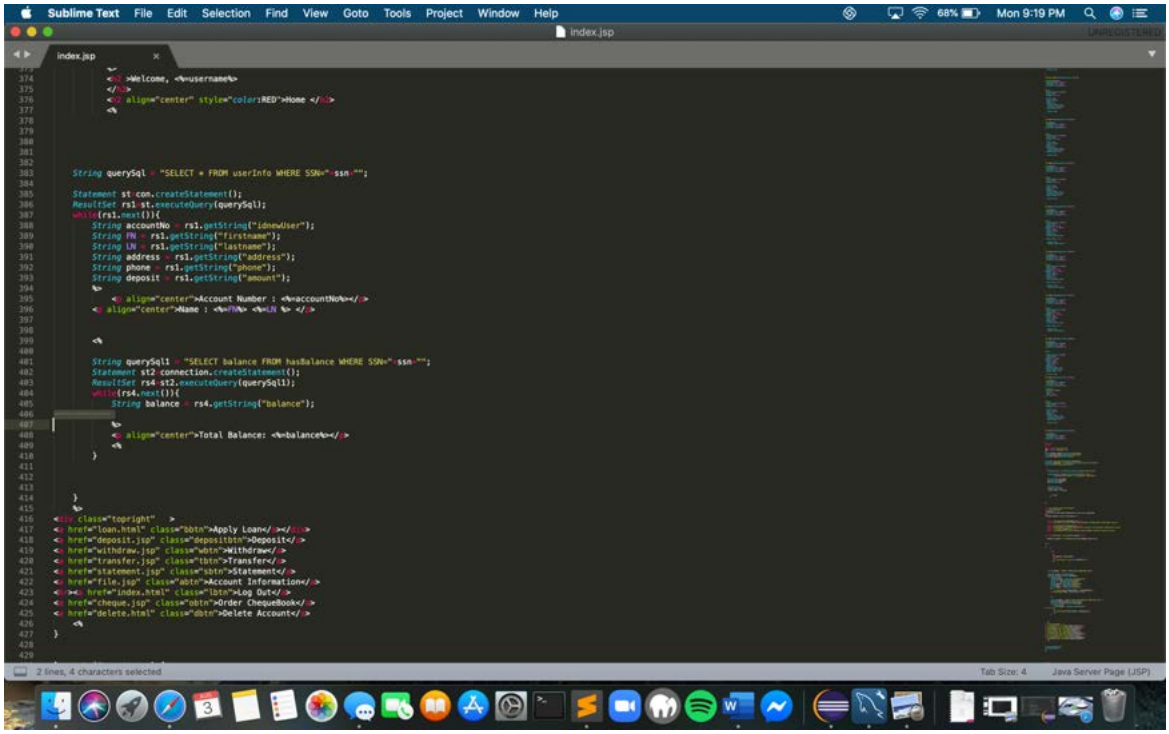
IMPLEMENTATION



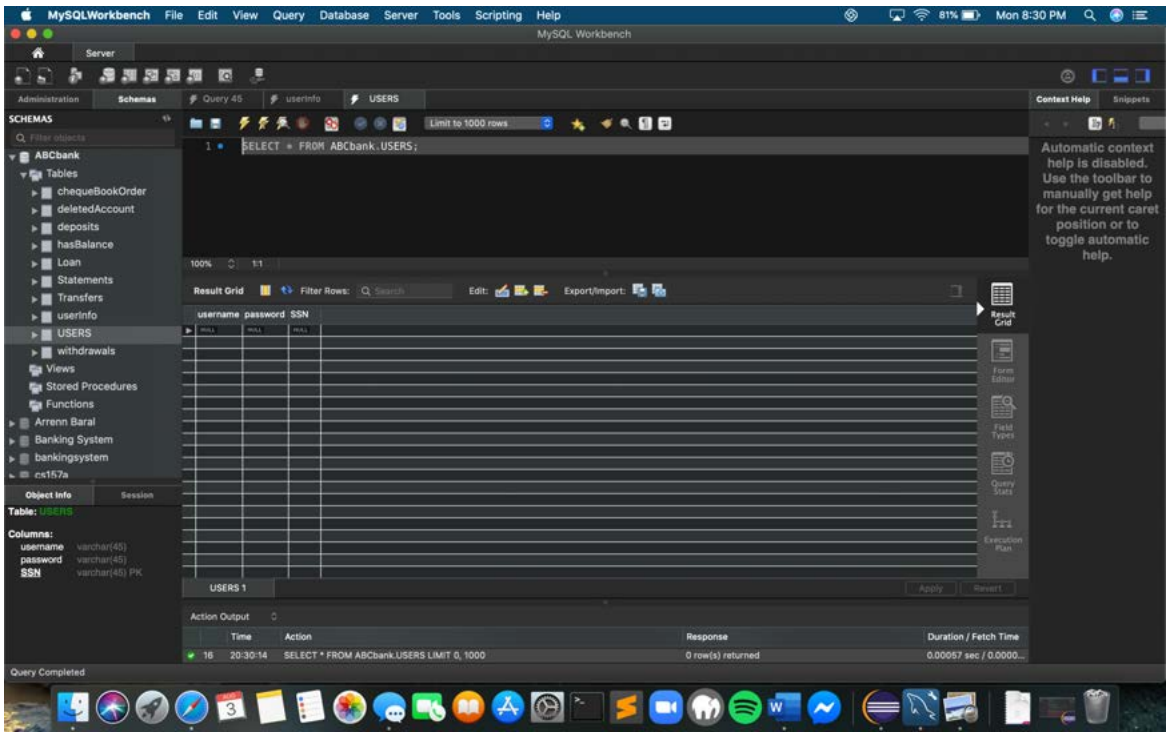
Front-end for login form.
Any new user can access the Banking System. But to access the system, an user must register beforehand. As you can see in the SS, a user with username “example” is logging in.



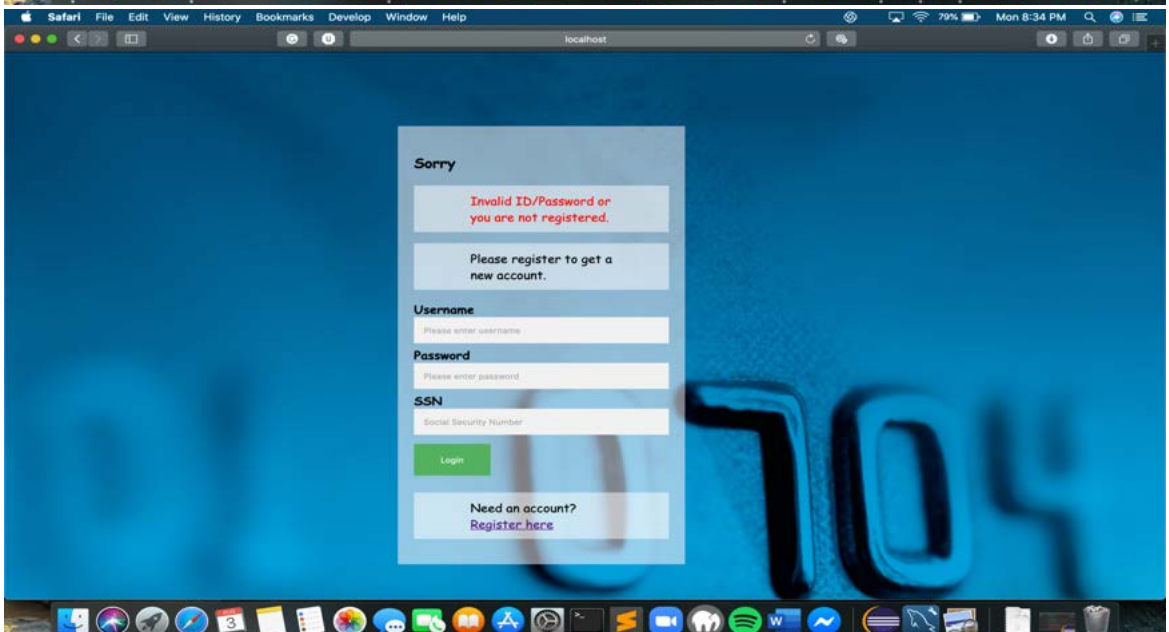
Back-end for login form.
This code will check if there an user is registered in the database system.
If not, it will prompt that it is not registered.



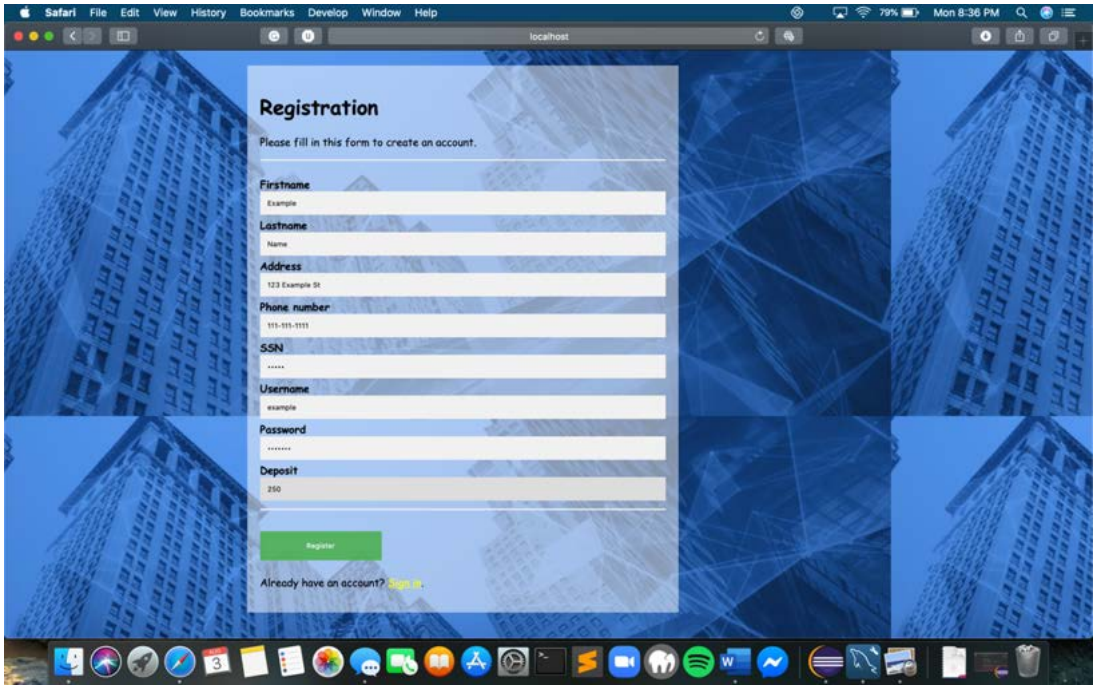
Part of the login form back-end



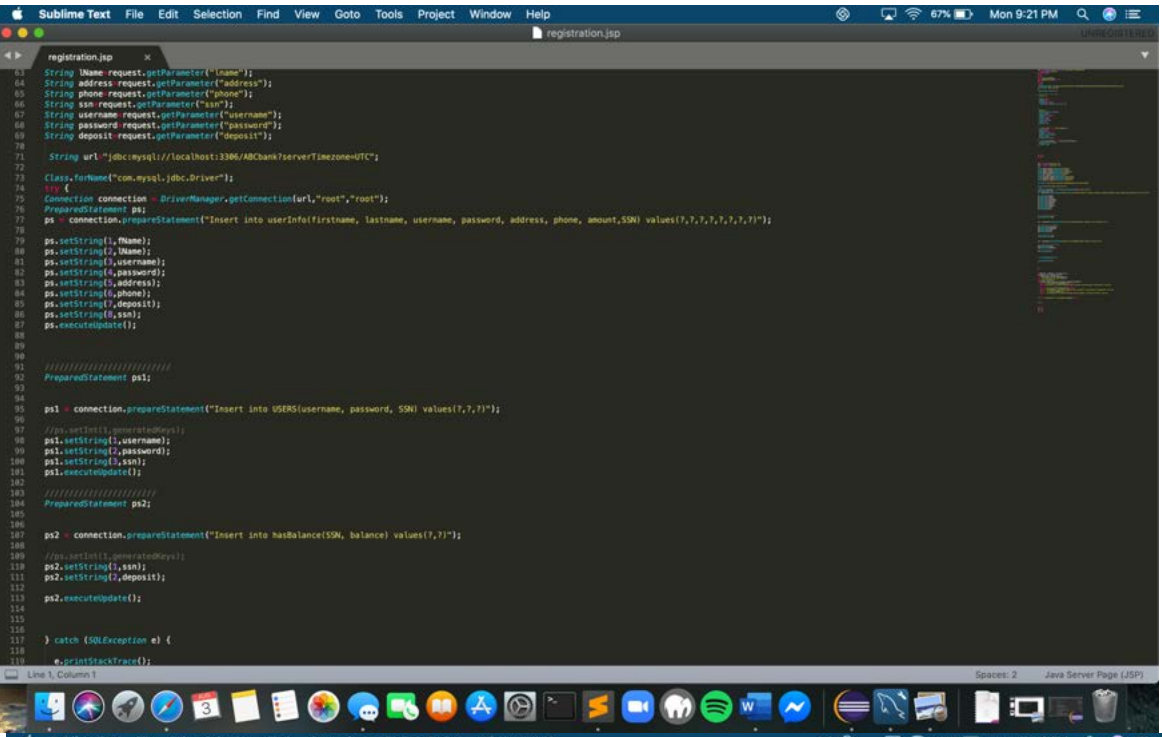
As we can see, there are no any users registered in the database. So it will prompt the not registered message.



Front-end for login form.
So, there are no any registered user in the database system. It prompts if the username or password is mistaken or the user is not registered.

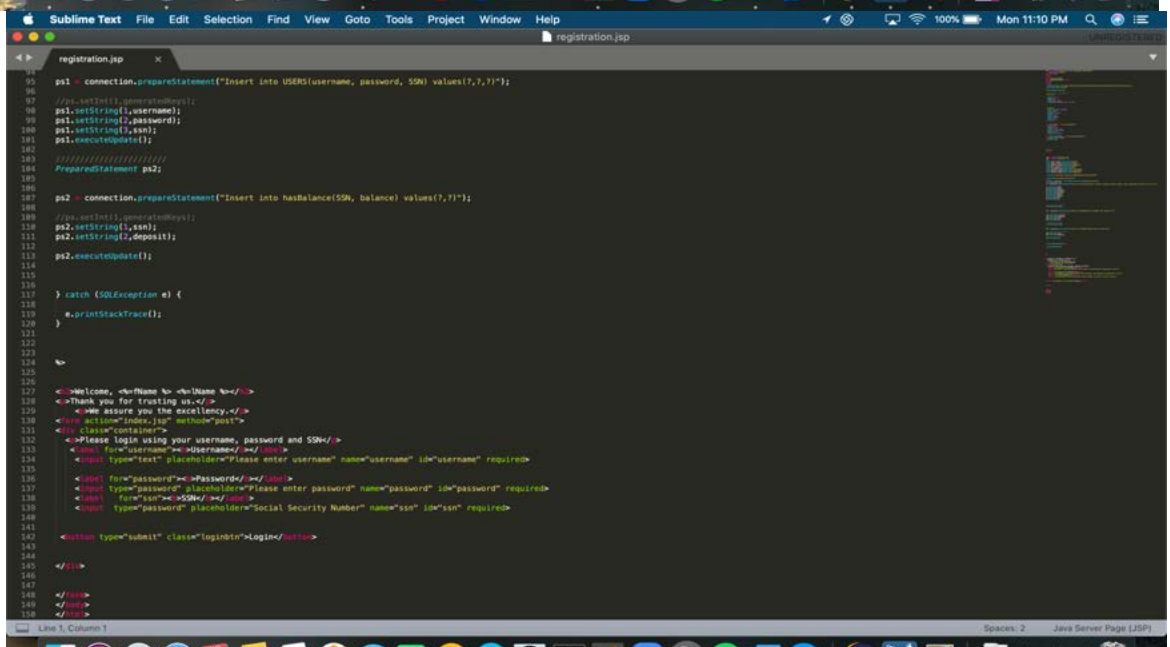


Front-end for Registration
After, the user get the register prompt, user clicks Register Here link, it redirects to here, so the user can create an account to get an access to the system. Here, user provides his/her information and first deposit.



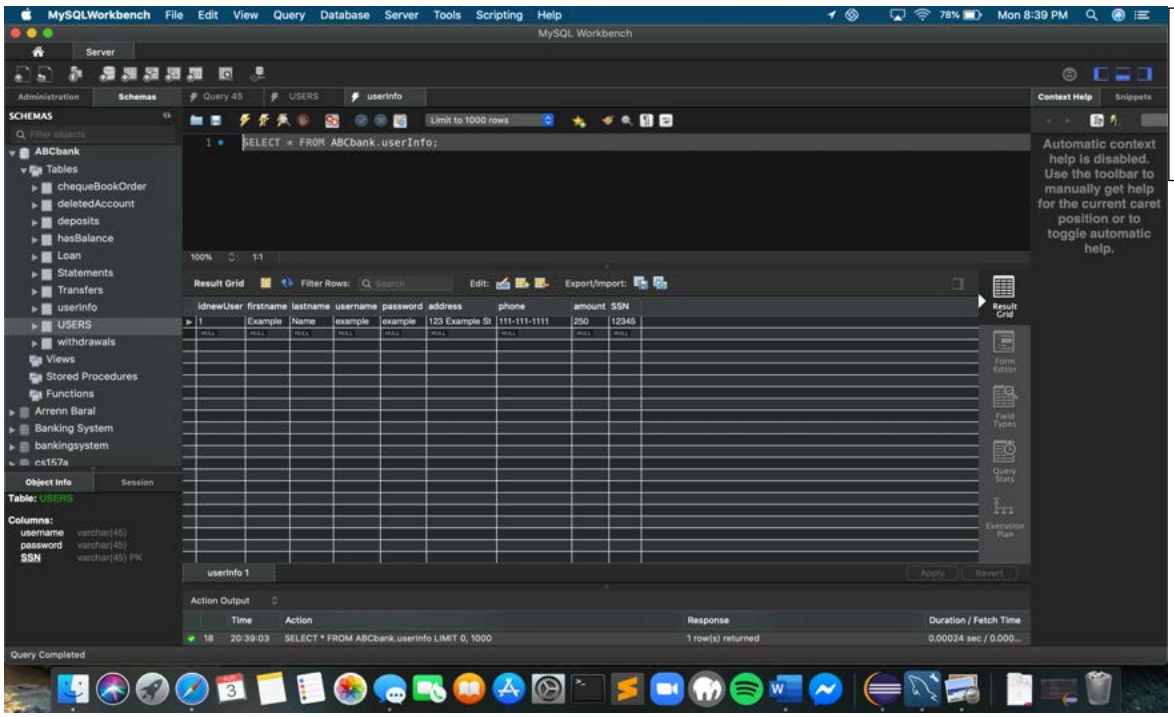
```
String fName=request.getParameter("fname");
String lName=request.getParameter("lname");
String address=request.getParameter("address");
String phone=request.getParameter("phone");
String ssn=request.getParameter("ssn");
String username=request.getParameter("username");
String password=request.getParameter("password");
String deposit=request.getParameter("deposit");

 String url="jdbc:mysql://localhost:3306/ABCbank?serverTimezone=UTC";

Class.forName("com.mysql.jdbc.Driver");
try {
Connection connection = DriverManager.getConnection(url,"root","root");
PreparedStatement ps;
ps = connection.prepareStatement("Insert into userInfo(firstname, lastname, username, password, address, phone, amount,SSN) values(?,?,?,?,?,?,?,?)");

ps.setString(1,fName);
ps.setString(2,lName);
ps.setString(3,username);
ps.setString(4,password);
ps.setString(5,address);
ps.setString(6,phone);
ps.setString(7,deposit);
ps.setString(8,ssn);
ps.executeUpdate();



//////////////////////
PreparedStatement ps1;


ps1 = connection.prepareStatement("Insert into USERS(username, password, SSN) values(?,?,?)");

//ps.setInt(1,generatedKey1);
ps1.setString(1,username);
ps1.setString(2,password);
ps1.setString(3,ssn);
ps1.executeUpdate();

//////////////////////
PreparedStatement ps2;


ps2 = connection.prepareStatement("Insert into hasBalance(SSN, balance) values(?,?)");

//ps.setInt(1,generatedKey1);
ps2.setString(1,ssn);
ps2.setString(2,deposit);

ps2.executeUpdate();



} catch (SQLException e) {

  e.printStackTrace();
```

Back-end for registration
This code will insert all the information to the userInfo, and USERS tables.

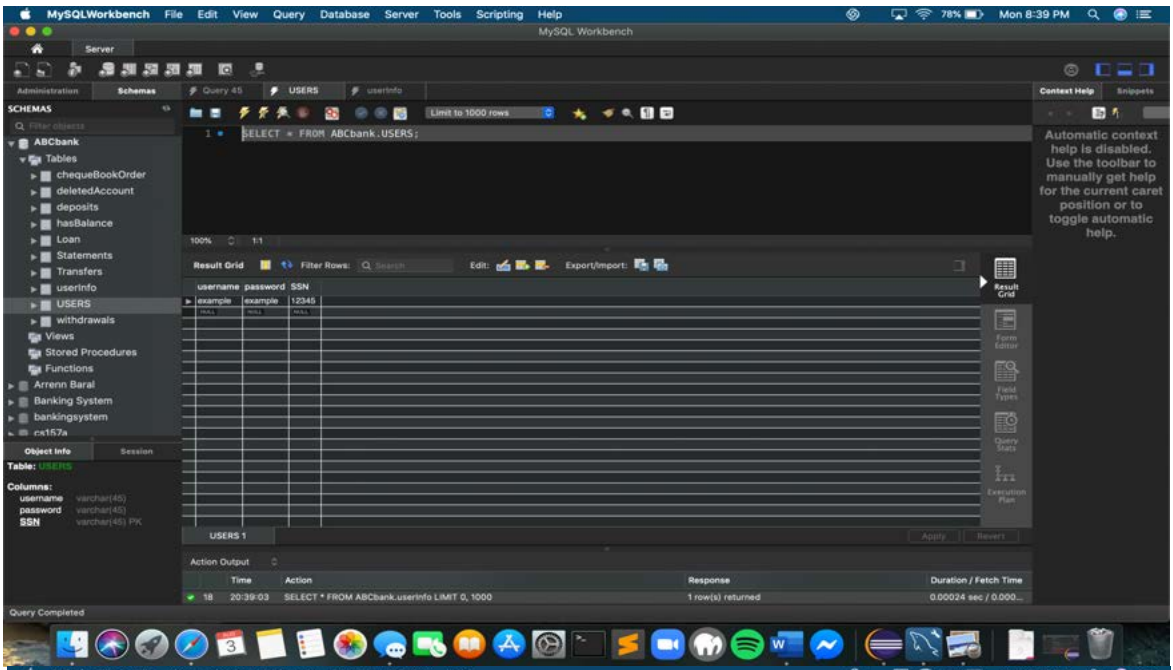


```
ps1 = connection.prepareStatement("Insert into USERS(username, password, SSN) values(?,?,?)");

//ps.setInt(1,generatedKey1);
ps1.setString(1,username);
ps1.setString(2,password);
ps1.setString(3,ssn);
ps1.executeUpdate();

//////////////////////
PreparedStatement ps2;


ps2 = connection.prepareStatement("Insert into hasBalance(SSN, balance) values(?,?)");

//ps.setInt(1,generatedKey1);
ps2.setString(1,ssn);
ps2.setString(2,deposit);

ps2.executeUpdate();



} catch (SQLException e) {

  e.printStackTrace();
 }


%>


<h3>Welcome, <%=fName %> <%=lName %></h3>
<h3>Thank you for trusting us.</h3>
    <h3>We assure you the excellency.</h3>
<form action="index.jsp" method="post">
<div class="container">
  <h3>Please login using your username, password and SSN</h3>
    <label for="username"><b>Username</b></label>
    <input type="text" placeholder="Please enter username" name="username" id="username" required>

    <label for="password"><b>Password</b></label>
    <input type="password" placeholder="Please enter password" name="password" id="password" required>
    <label    for="ssn"><b>SSN</b></label>
    <input   type="password" placeholder="Social Security Number" name="ssn" id="ssn" required>


  <button type="submit" class="loginbtn">Login</button>


</div>


</form>
</body>
</html>
```

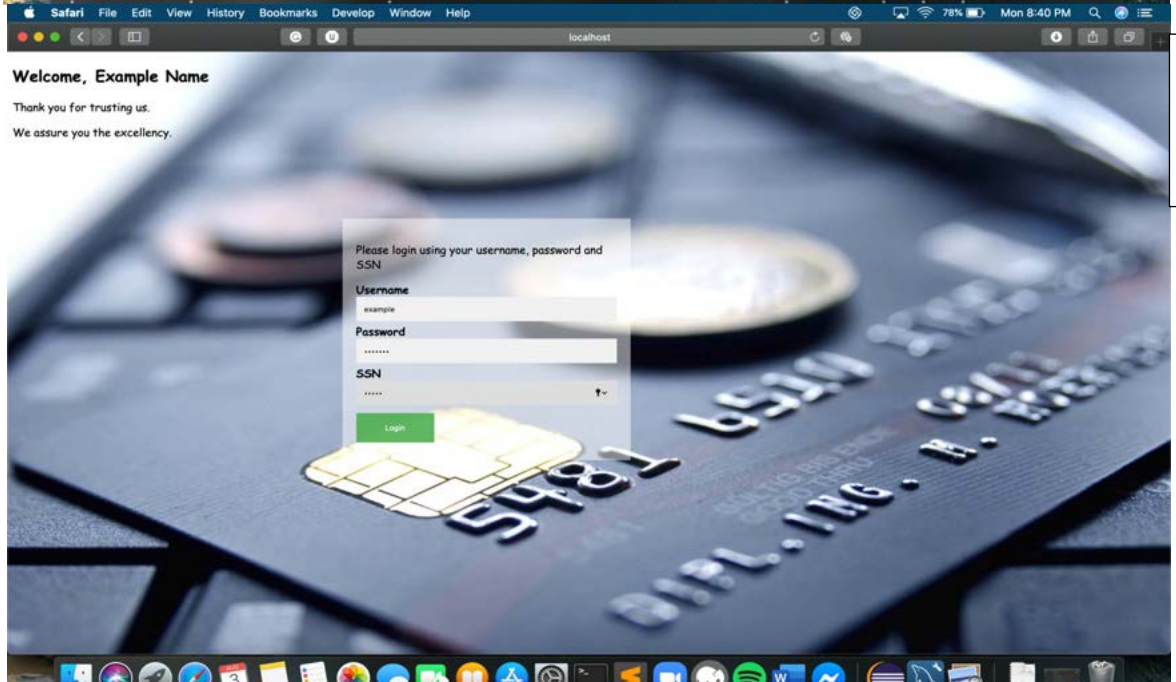
Back-end for registration
Part of the registration



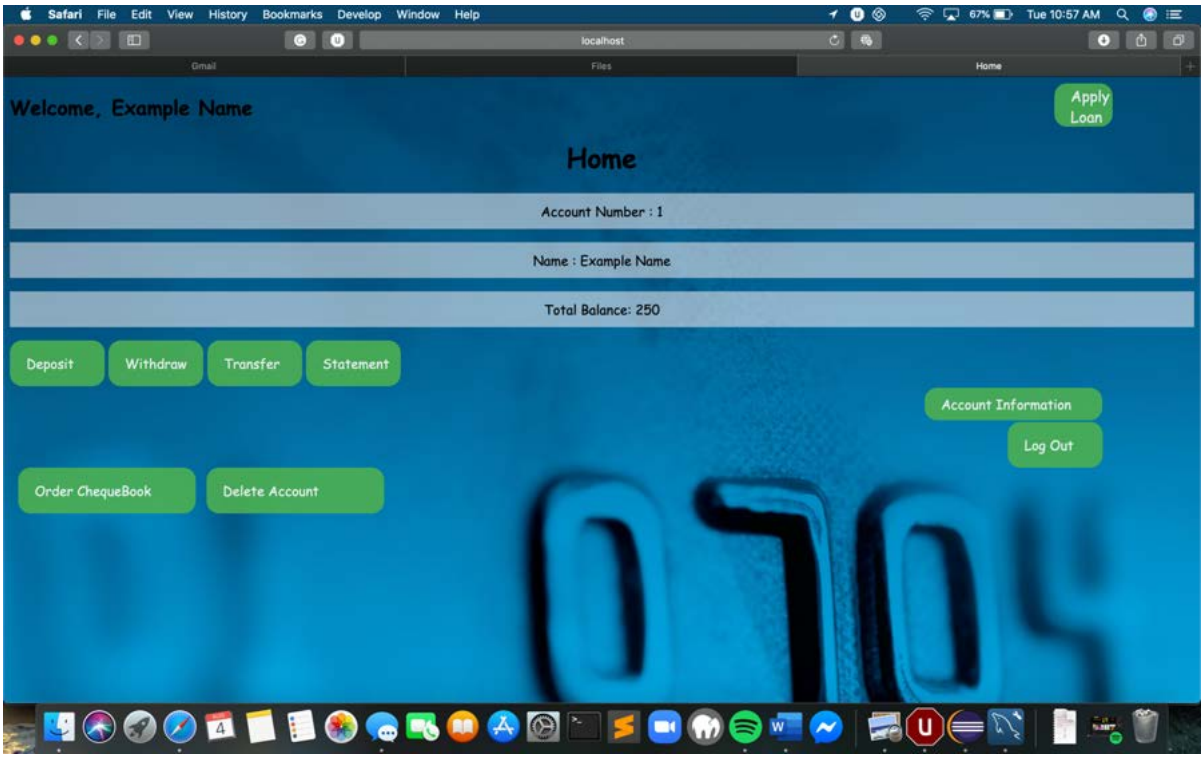
Tables for registration
After registration, it successfully adds the user information in the userInfo.



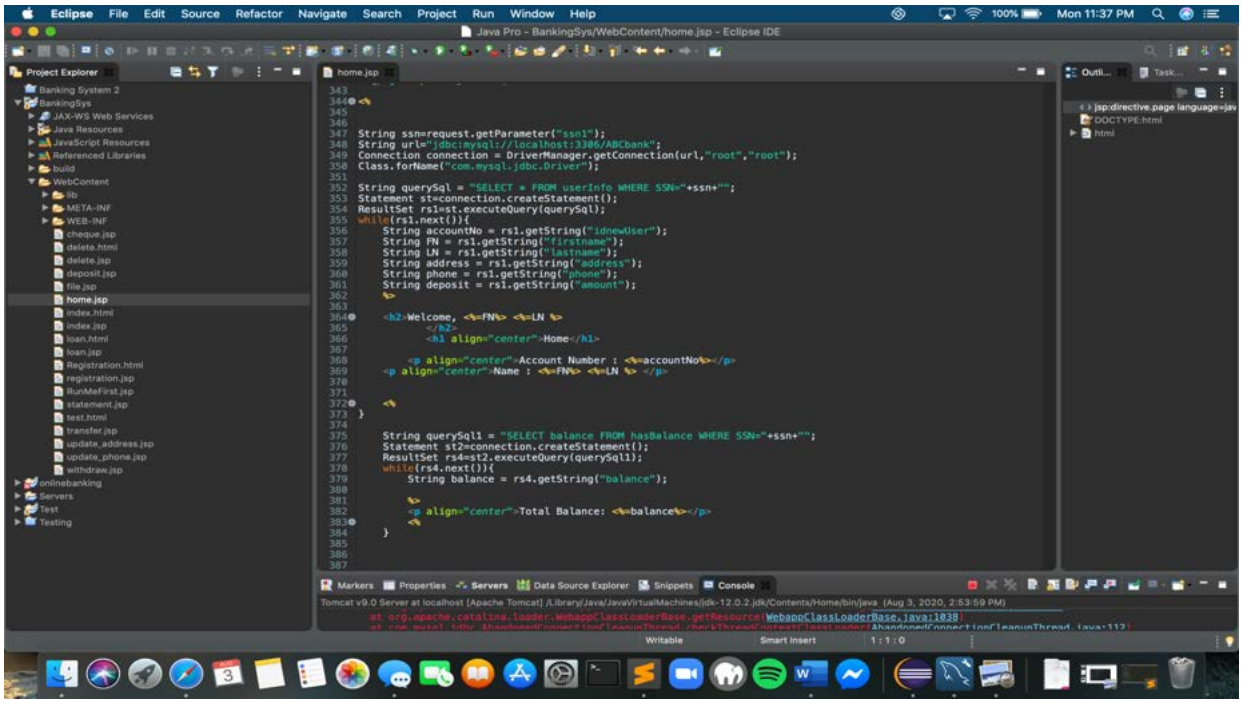
Tables for registration
After registration, it successfully adds the username, password, and SSN in the USERS table.



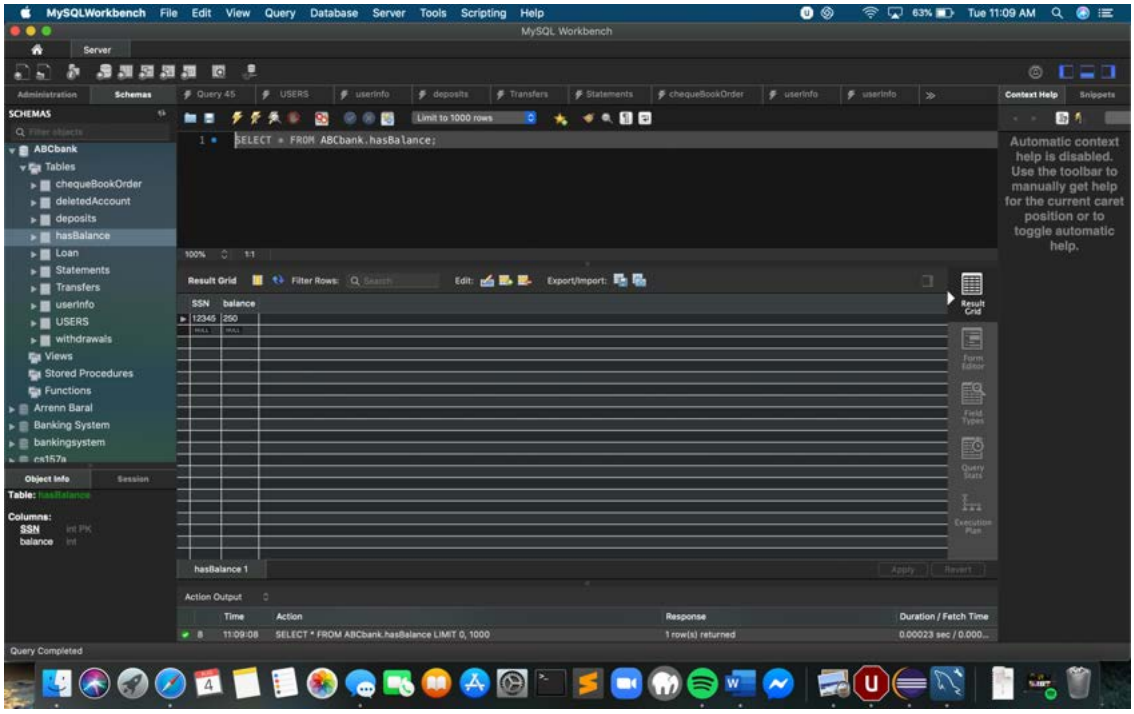
Front-end after the successful registration
User will be asked to use their recent username, password, and SSN to login.
It also welcomes user with the message on the top.



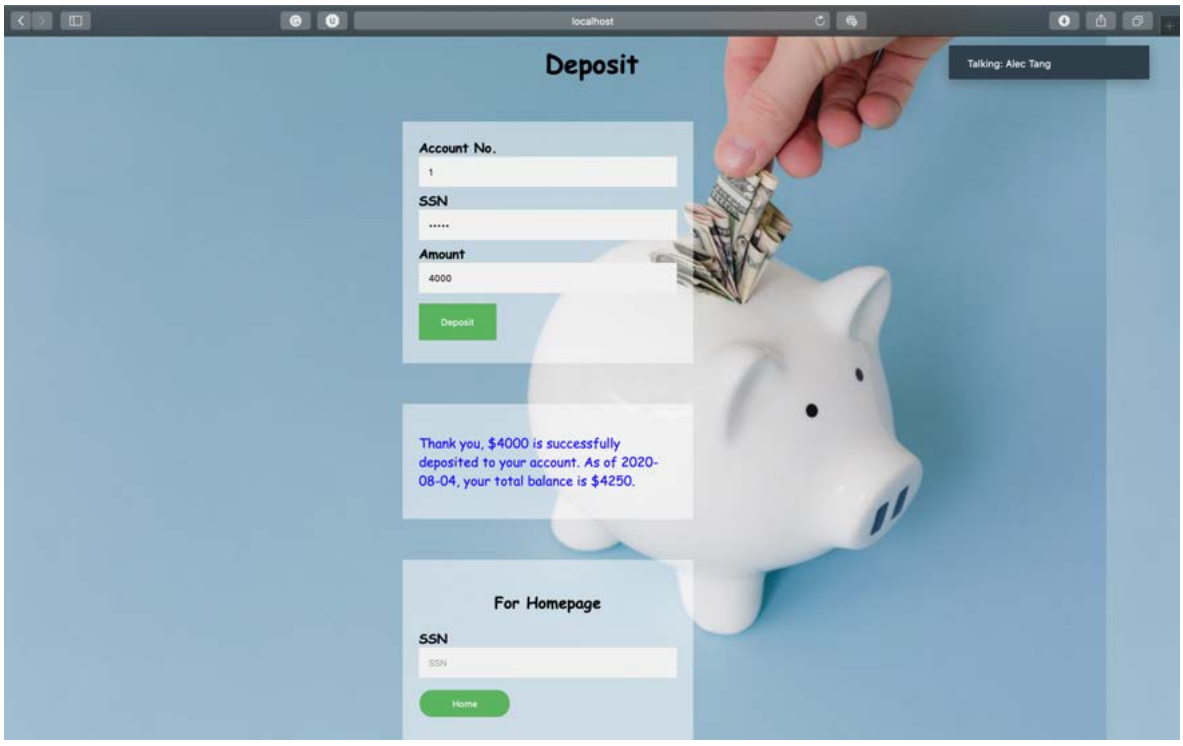
Front-end Homepage
Once logged in, it will redirect registered user to the Home Page of their account.
User sees the account, name, and total balance with other features like deposit, withdraw, transfer, statement, order cheque book, account information, loan apply, delete account, and log out. Total balance is 250 after the first deposit while registration.



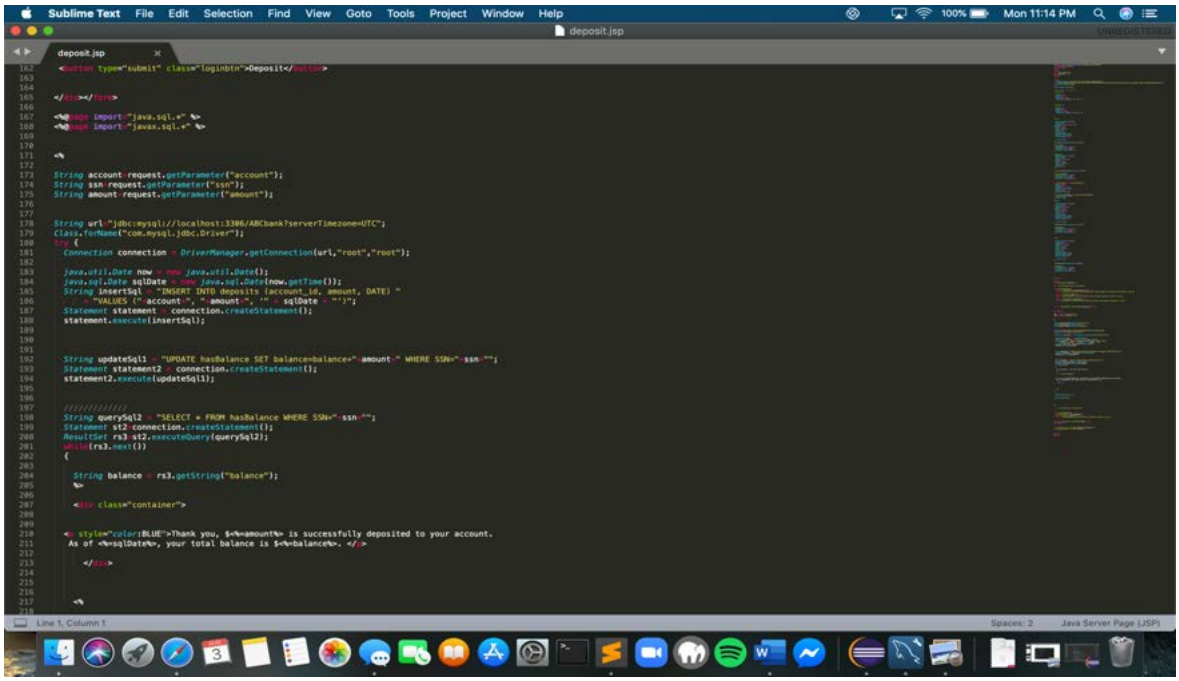
Back-end Homepage
Its selects account_no, firstname, and last_name from userInfo. And its selects balance from the hasBalance table.



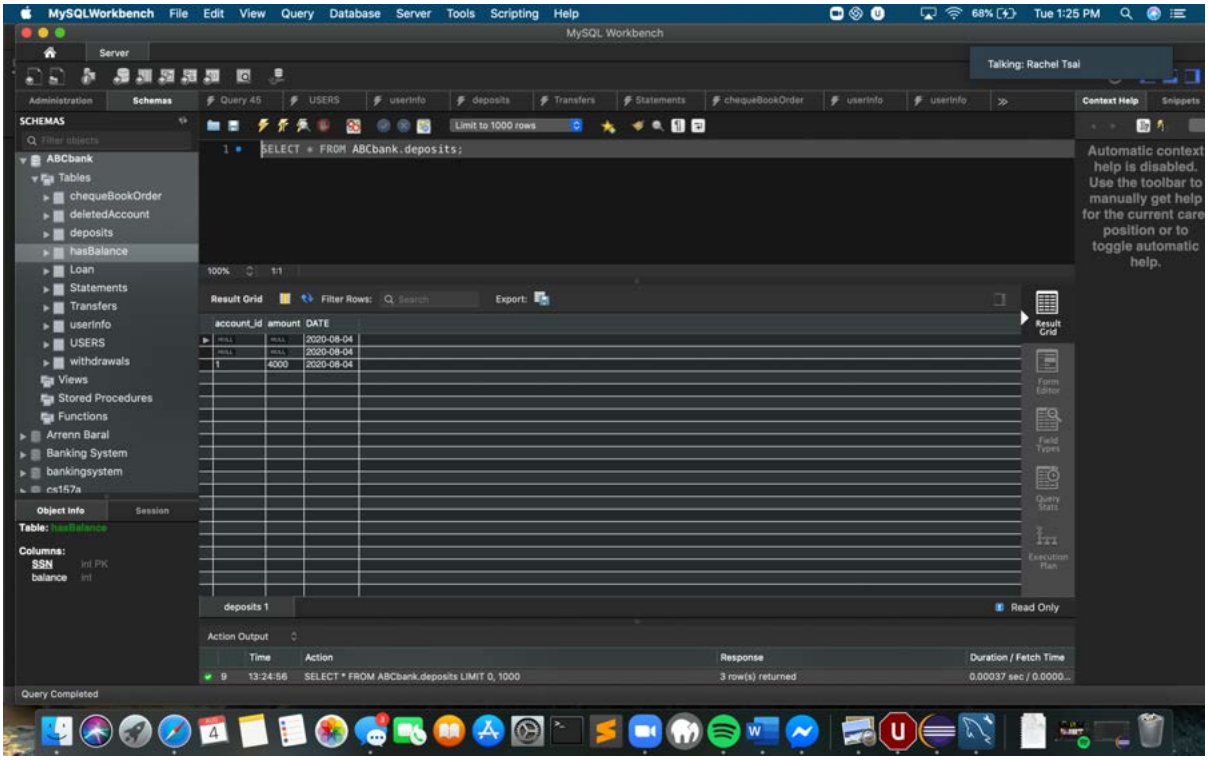
When an user registers, they deposit certain amount that gets stored on the table hasBalance.



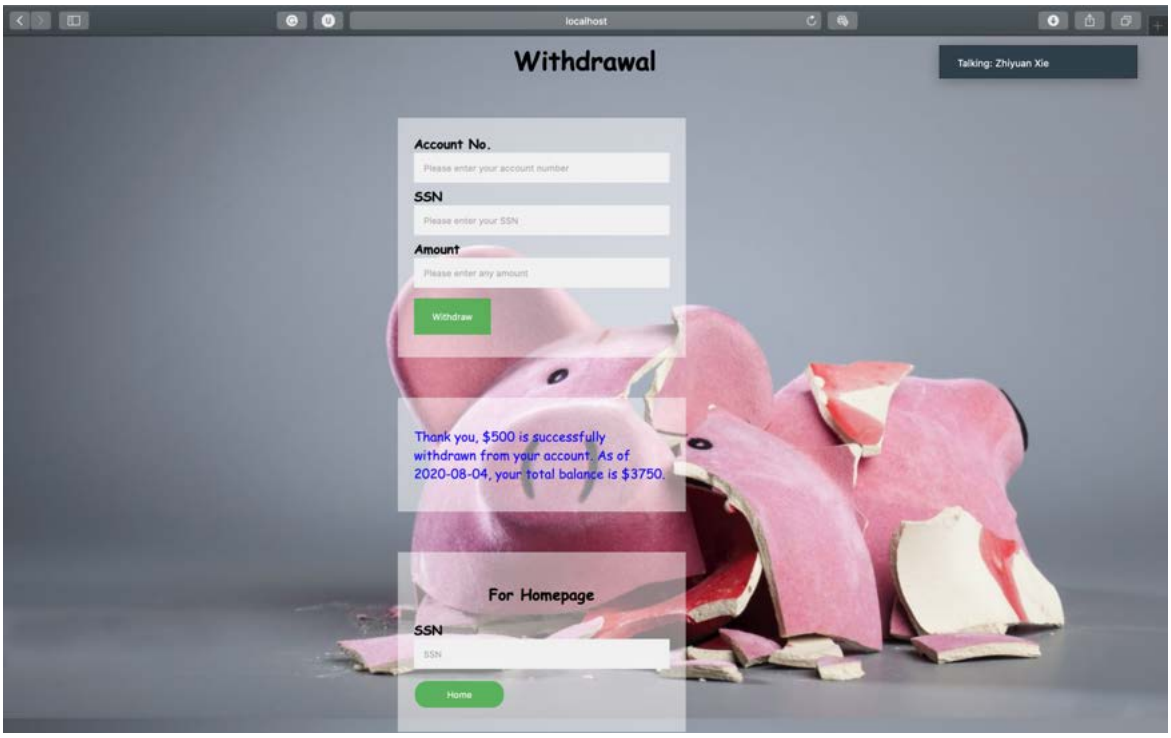
From homepage, if user chooses Deposit button, it will redirect to deposit page where user puts account number, ssn and amount. After the deposit, it gives the message that the amount has been deposited and the total amount.



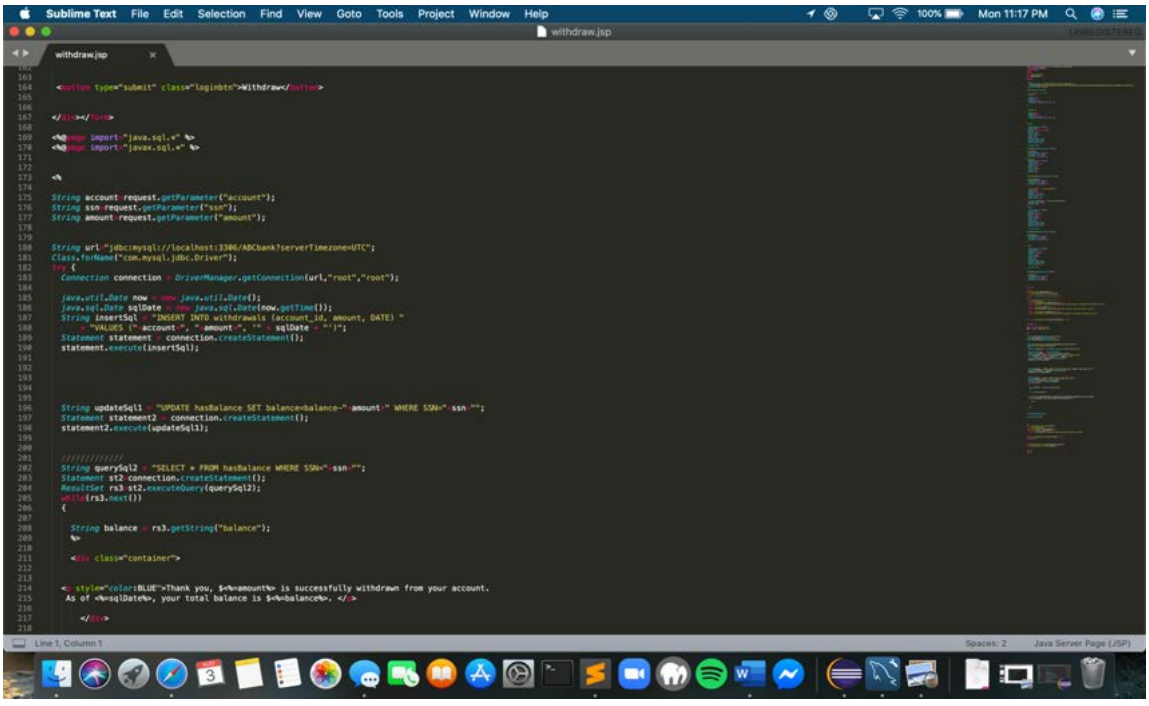
Back-end program
Once the data is submitted, this code will help to store informations on the table deposits table and update hasBalance.



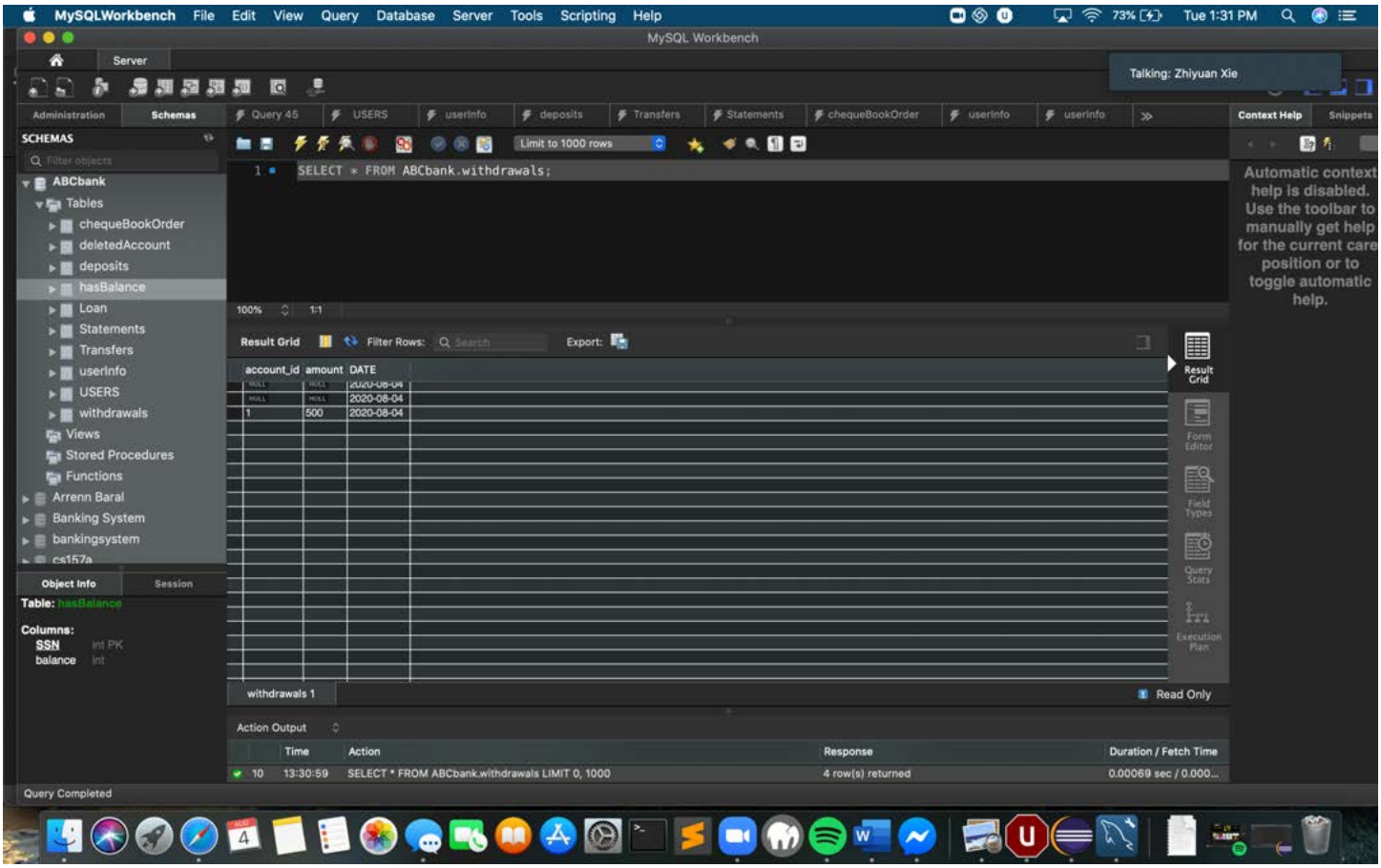
As we can see, the deposit record has been recorded on the table.



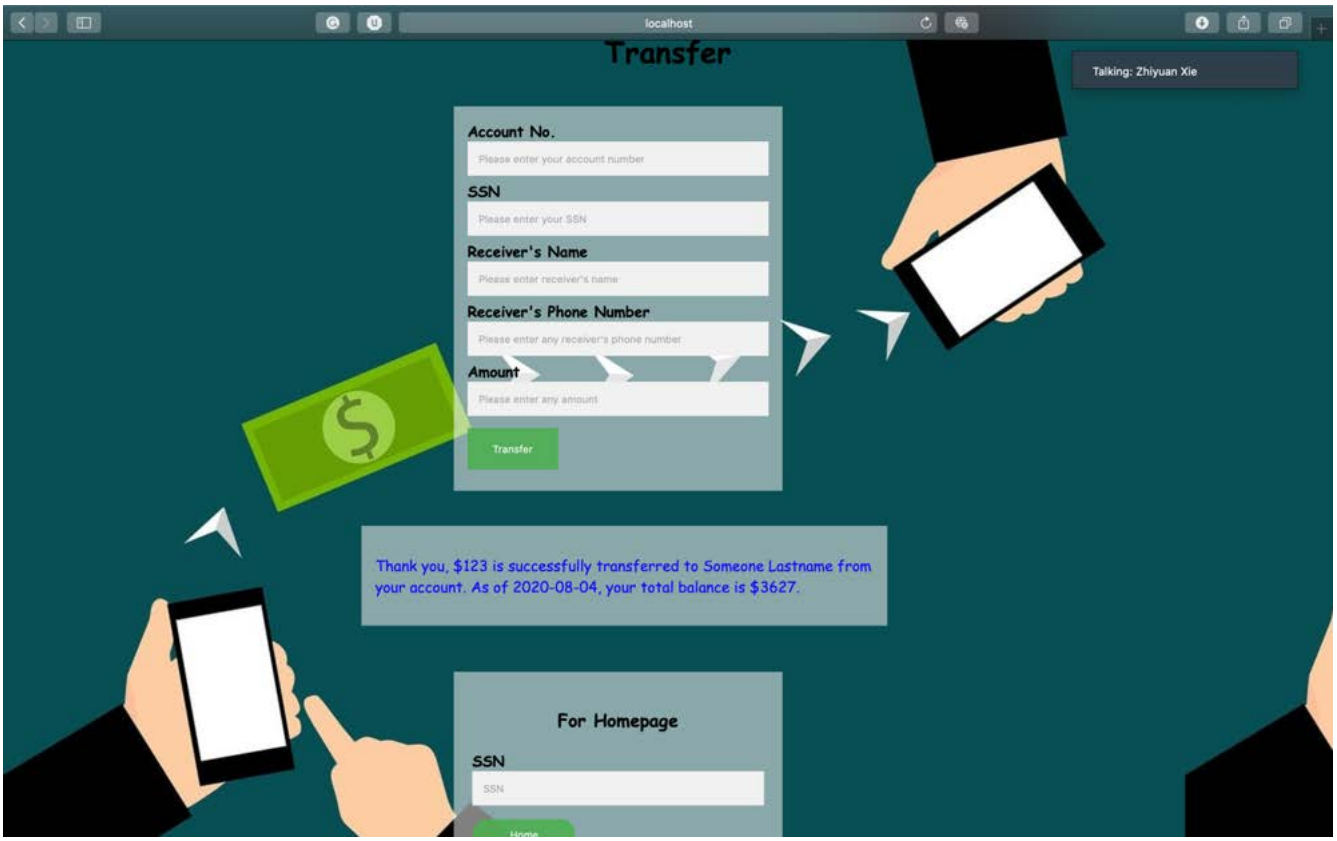
From homepage, if user chooses Withdraw button, it will redirect to withdraw page where user puts account number, ssn and amount. After the withdrawal, it gives the message that the amount has been withdrawn and the total amount.



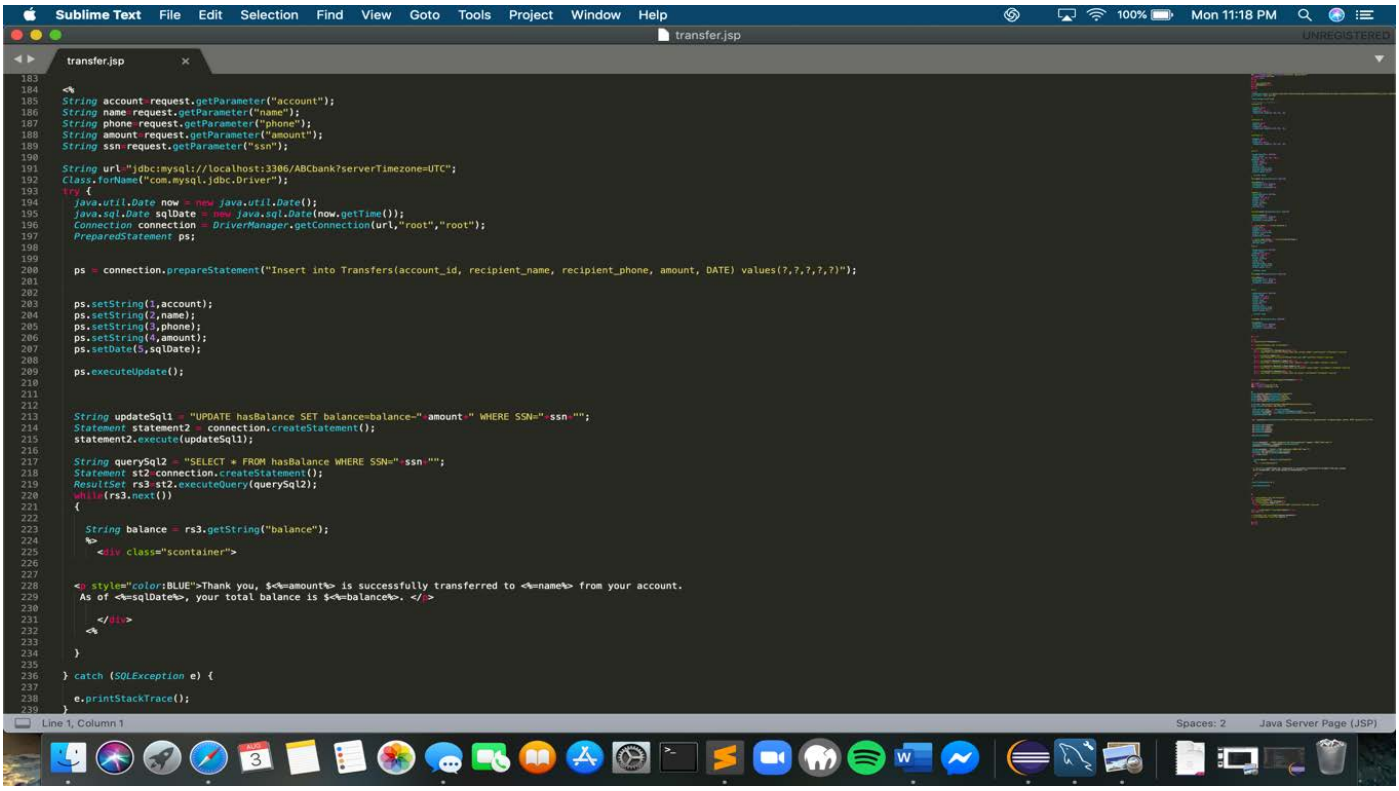
Back-end program
Once the data is submitted, this code will help to store informations on the table withdrawal table and update hasBalance.



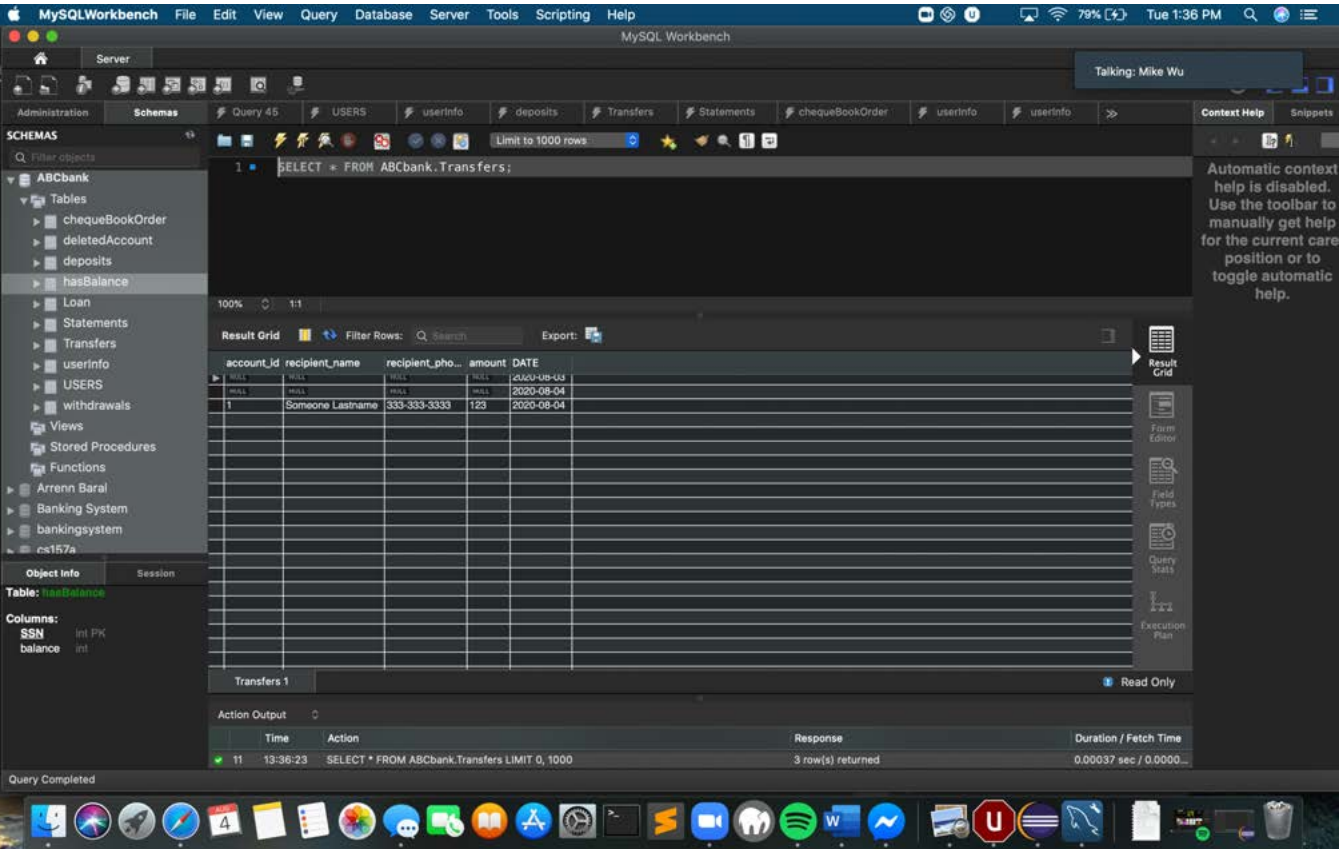
As we can see, the withdraw record has been recorded on the table.



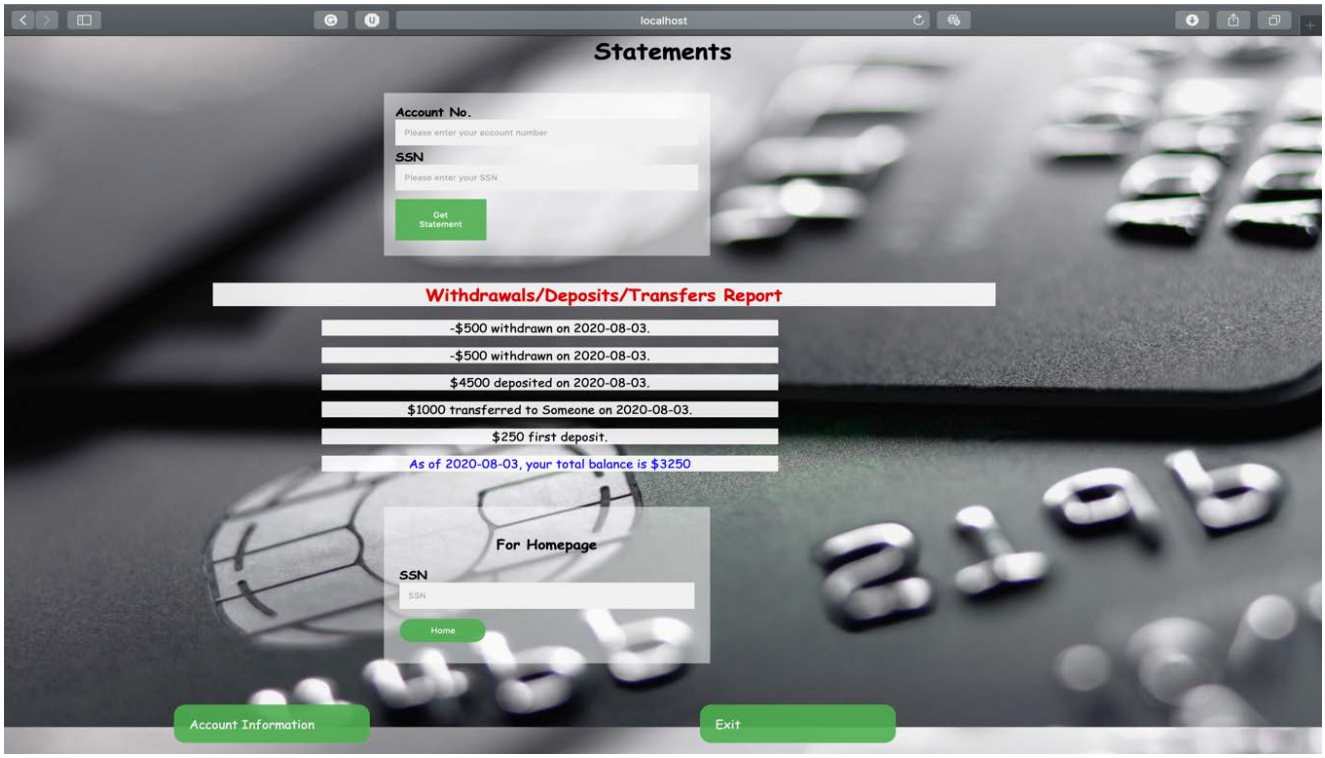
From homepage, if user chooses Transfer button, it will redirect to transfer page where user puts account number, ssn, receiver's name, phone, and amount. After the transfer, it gives the message that the amount has been transferred to the receiver and the total amount.



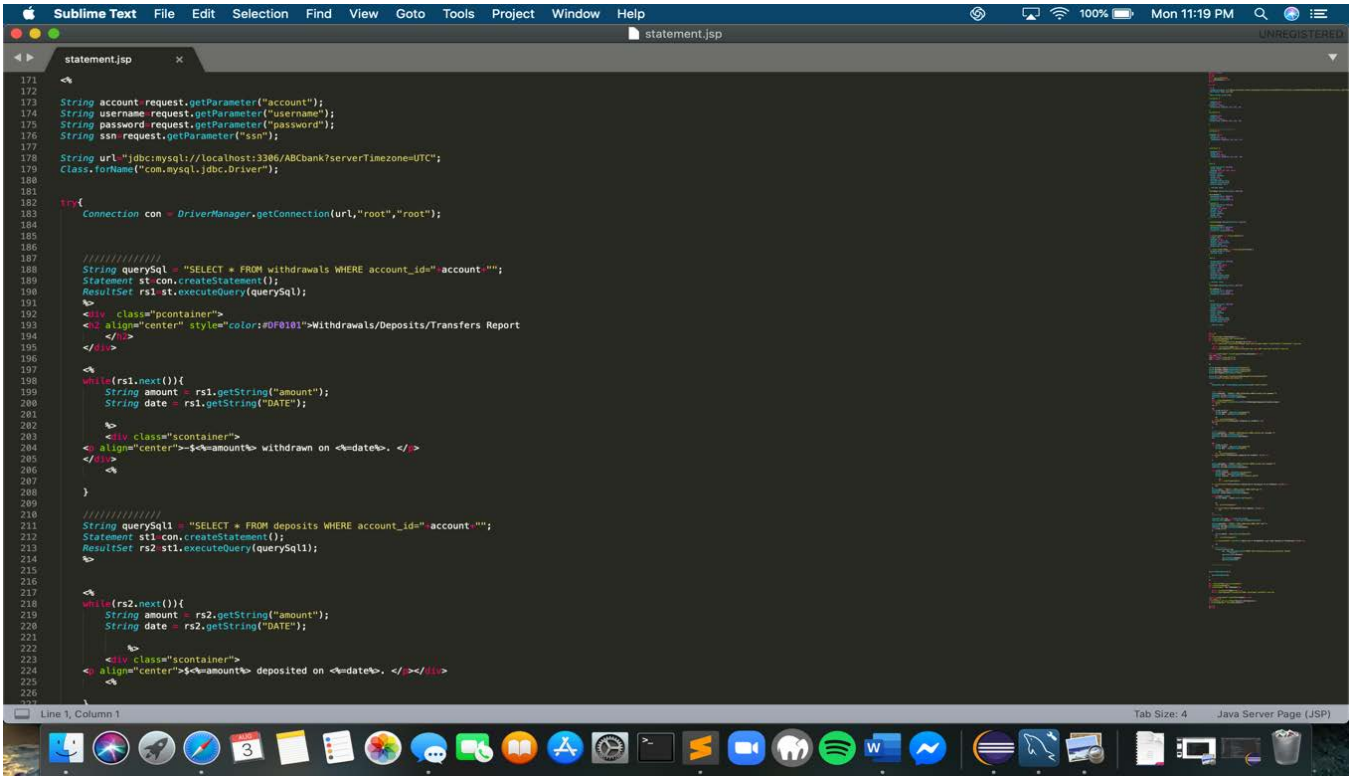
Back-end program
Once the data is submitted, this code will help to store informations on the table transfer table and update hasBalance.



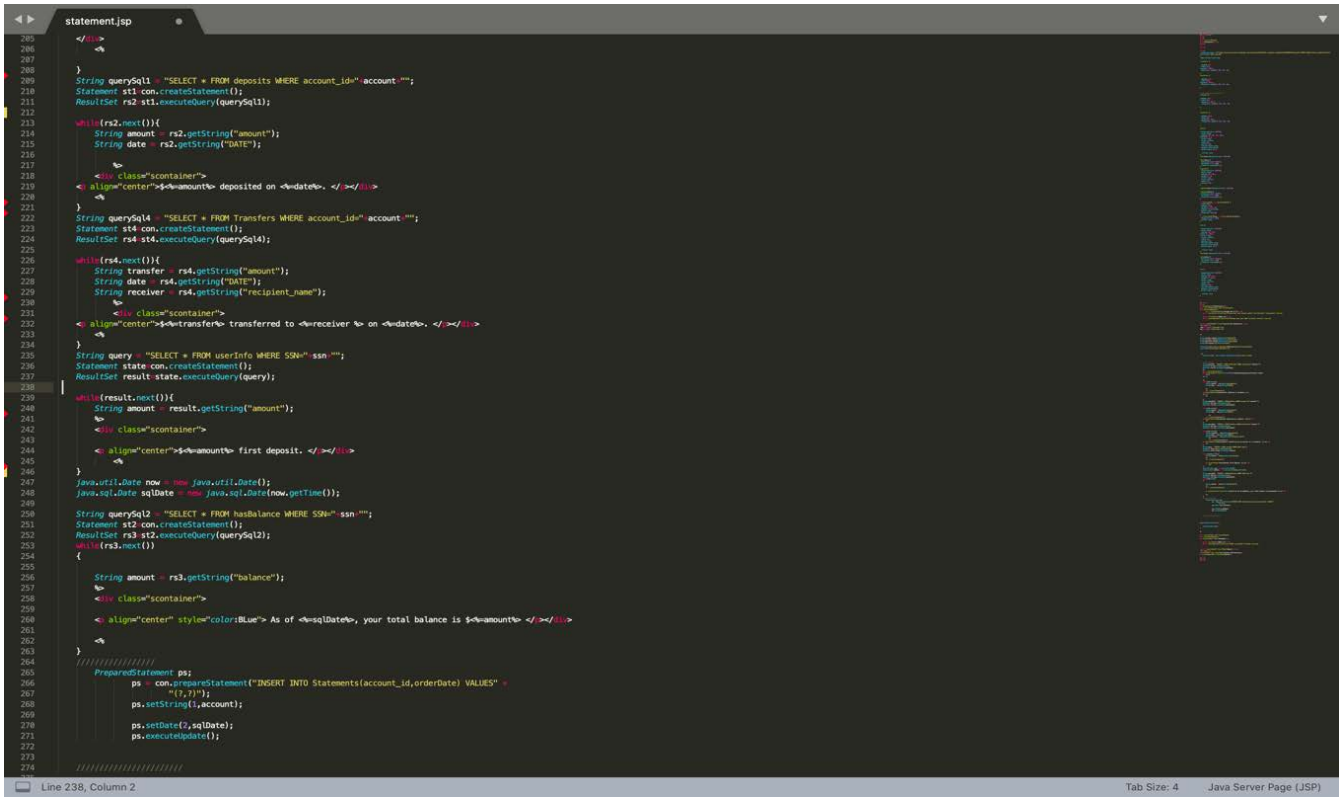
As we can see, the Transfer record has been recorded on the table.



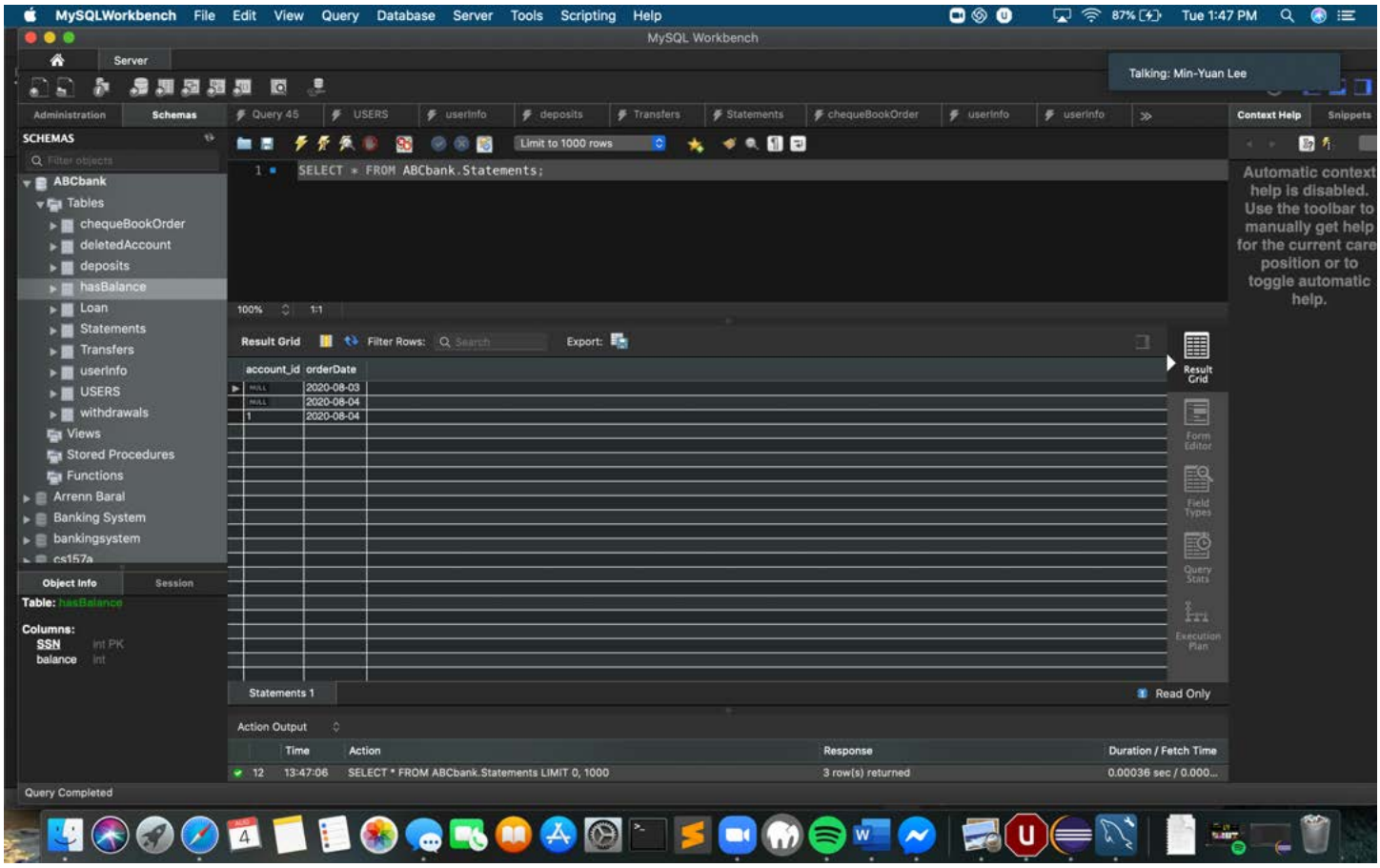
From homepage, if user chooses Statement button, it will redirect to statement page where user puts account number, and ssn. Once the data is received, it gives the records of recent transactions.



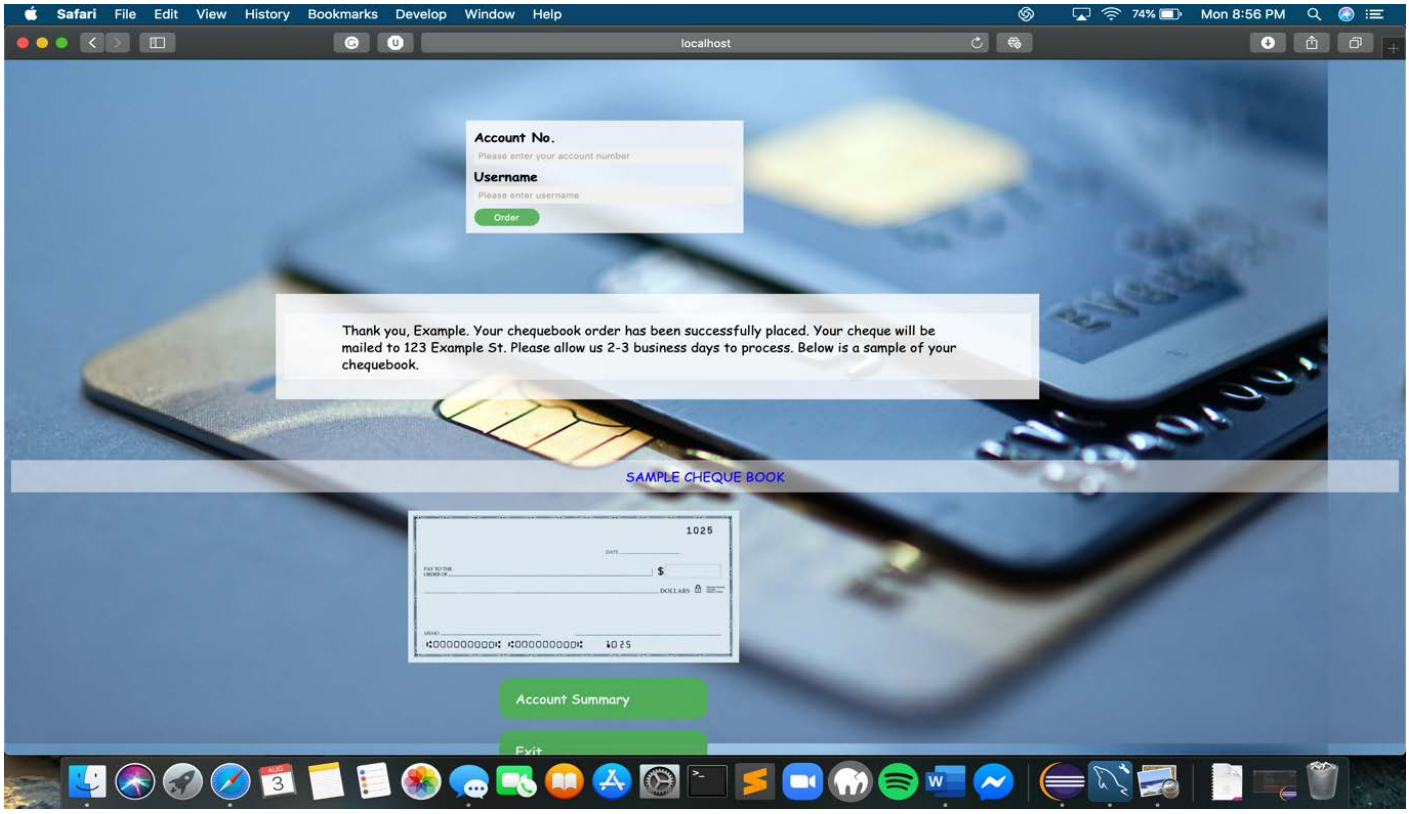
Back-end program
Once the data is submitted, this code will help to store informations on the table statements.



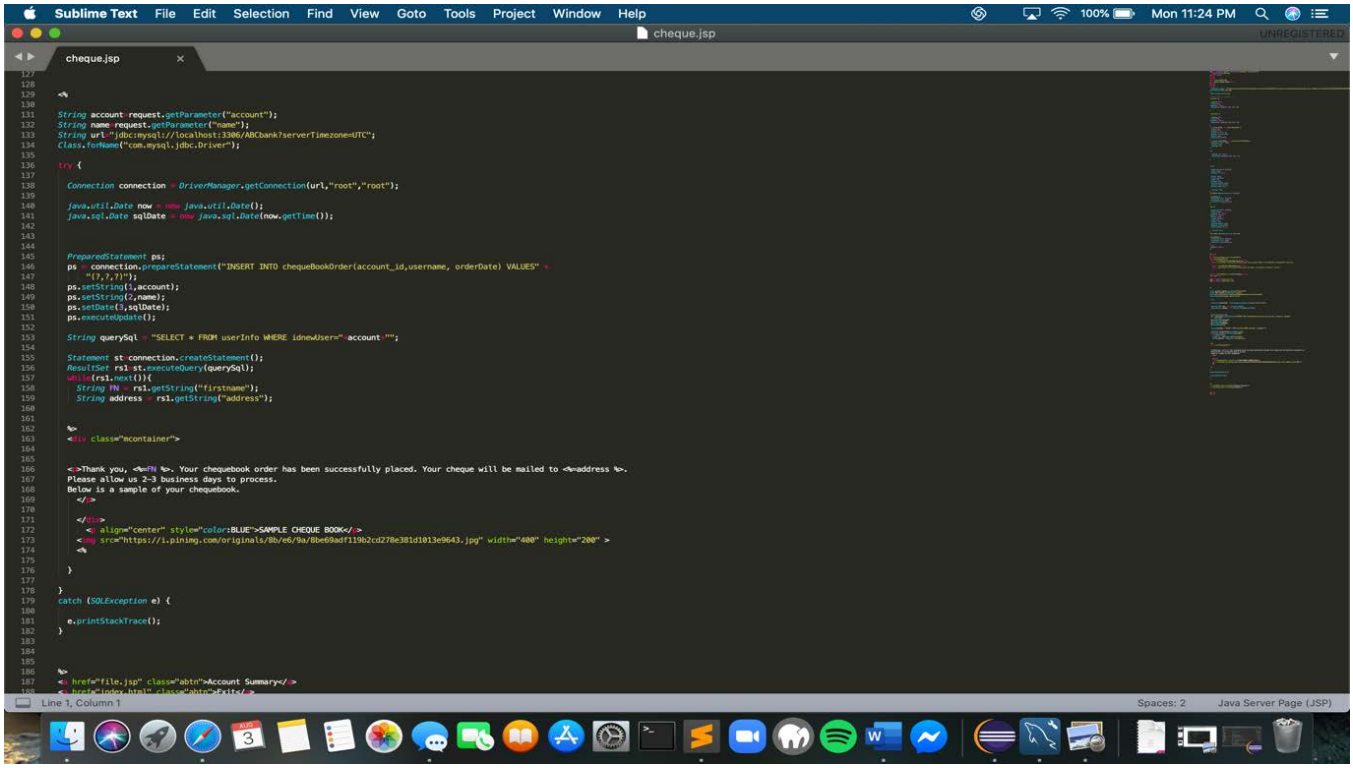
Part of back-end statement feature.



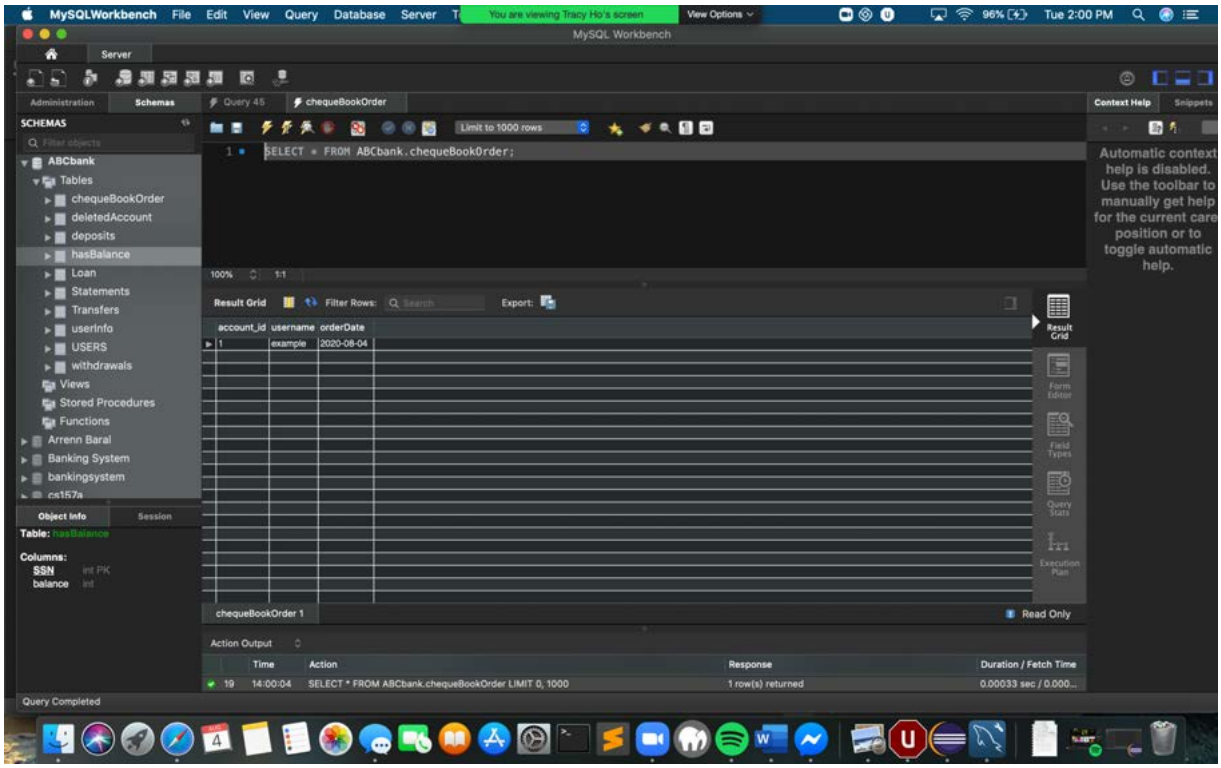
As we can see, the statement record has been recorded on the table.



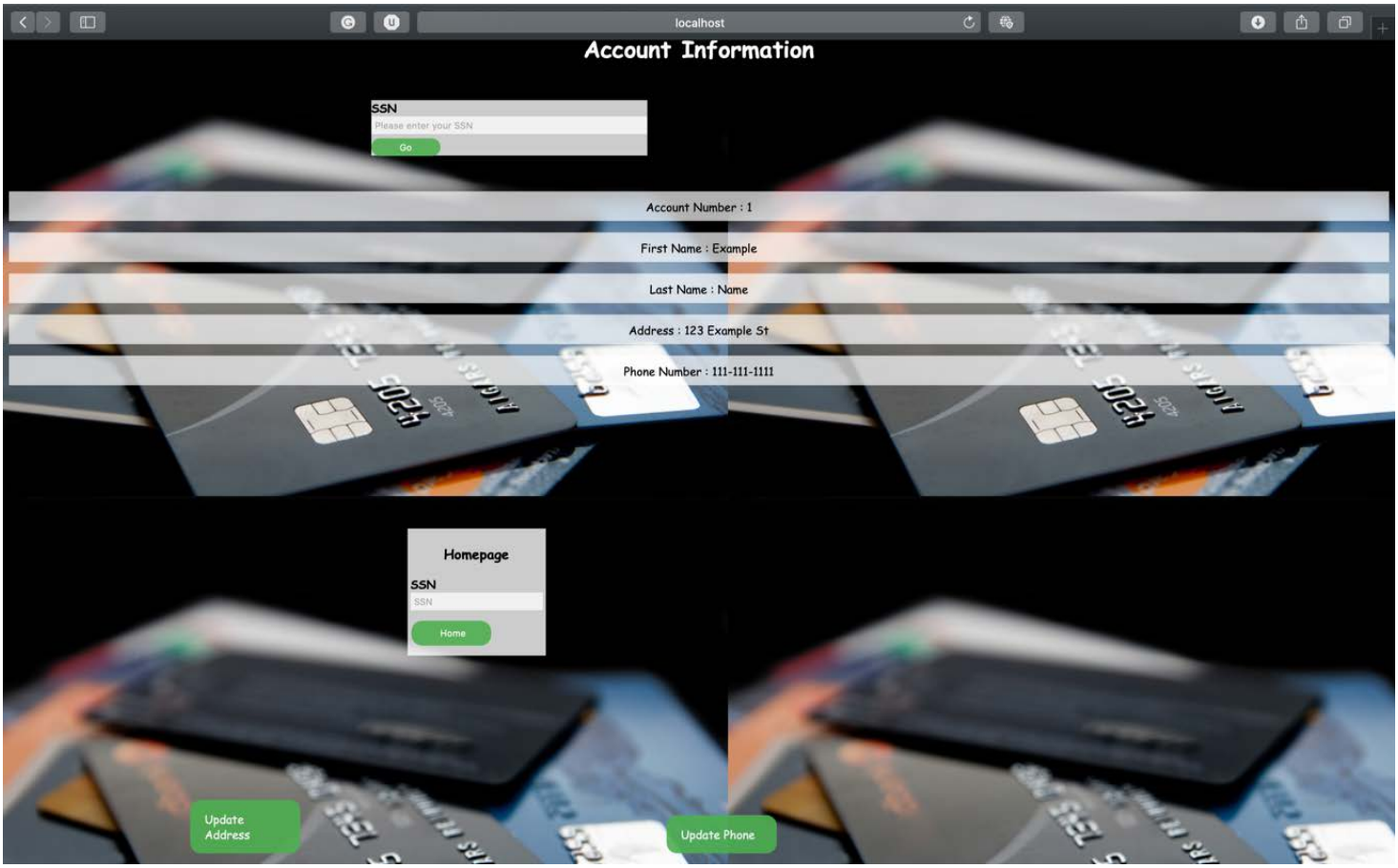
From homepage, if user chooses Order Checkbook button, it will redirect to checkbookorder page where user puts account number, and ssn. Once the data is received, it gives the message that the order is placed and it will be mailed the stored address.



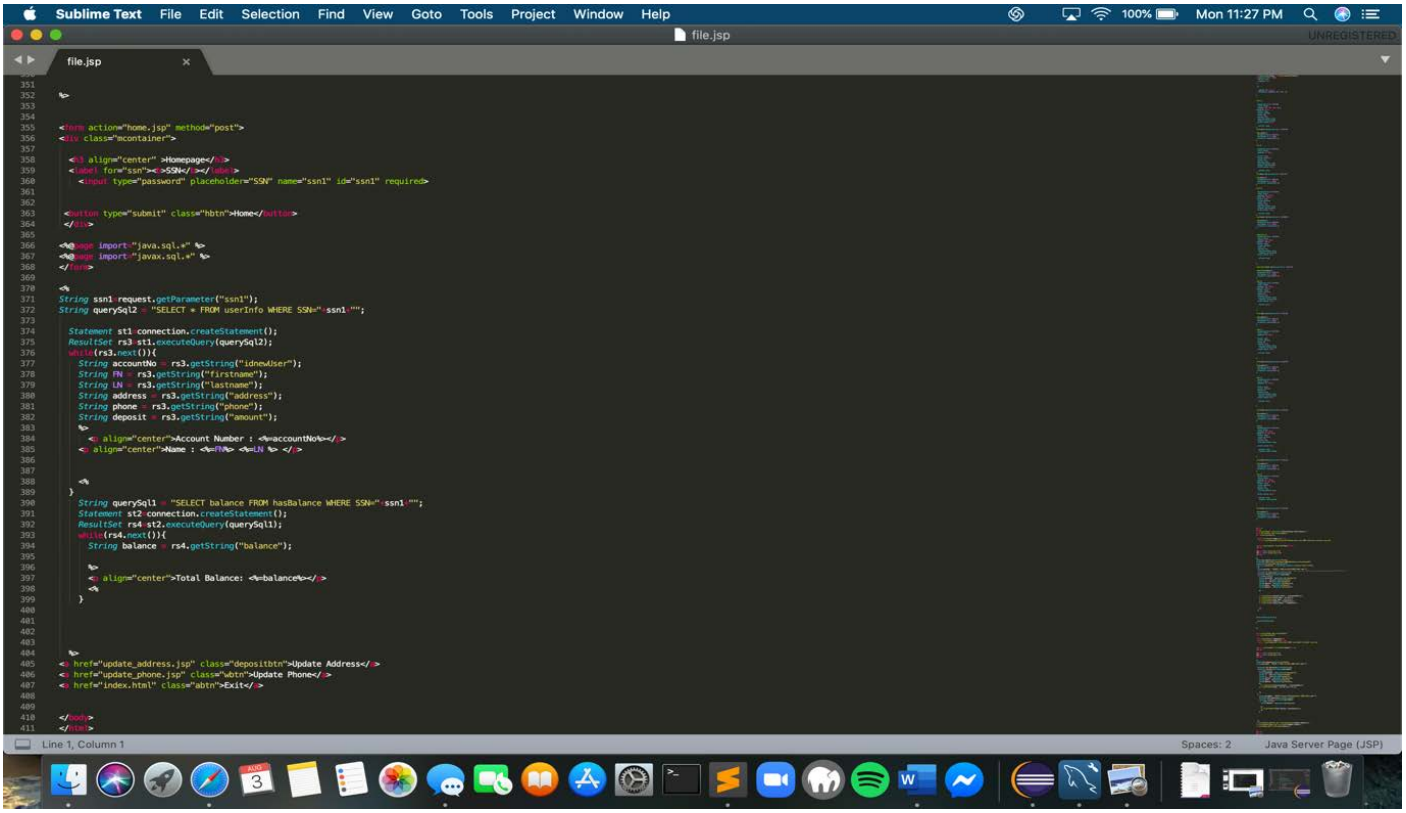
Back-end program
Once the data is submitted, this code will help to store informations on the table chequeBookOrder.



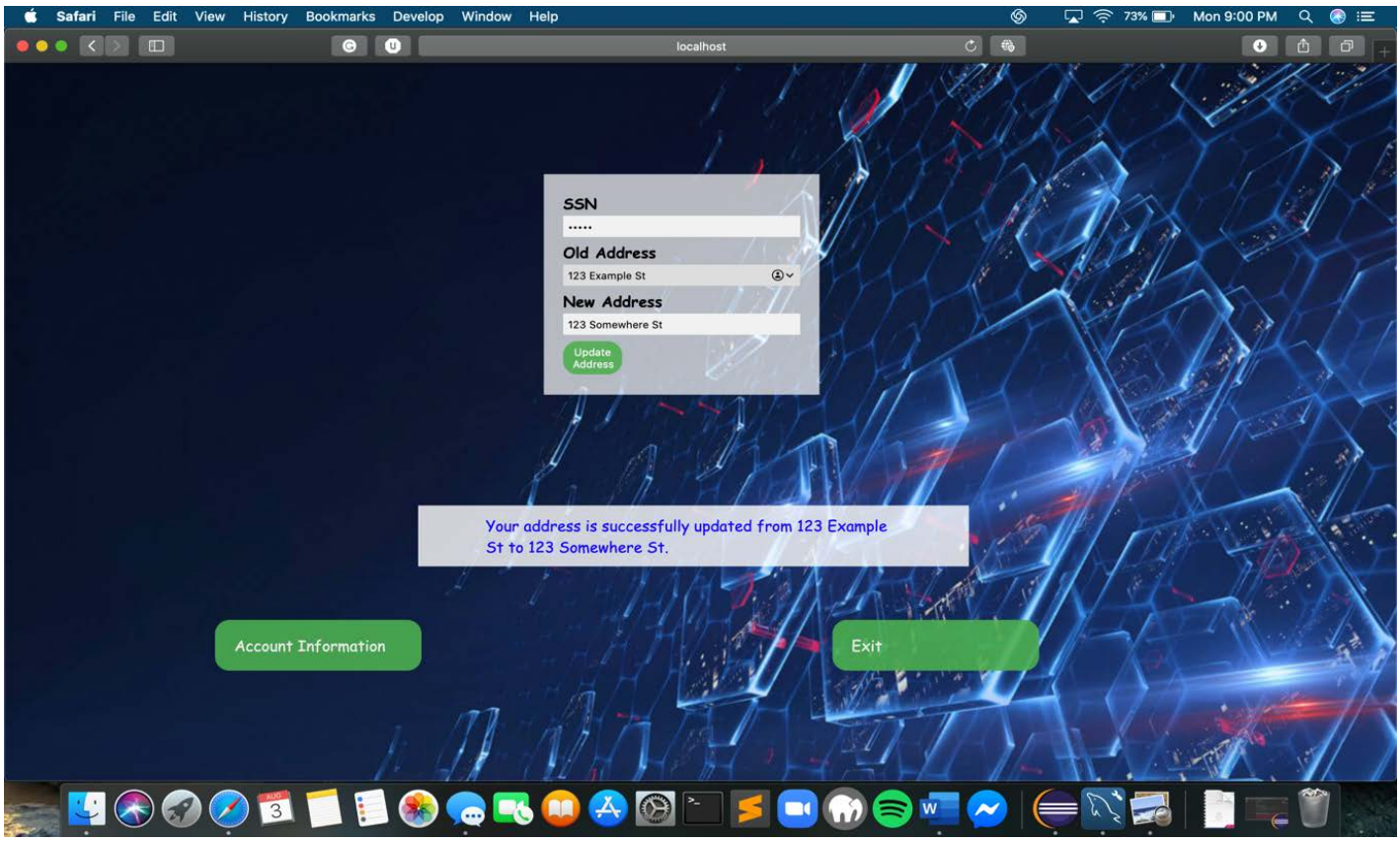
As we can see, the check book order record has been recorded on the table.



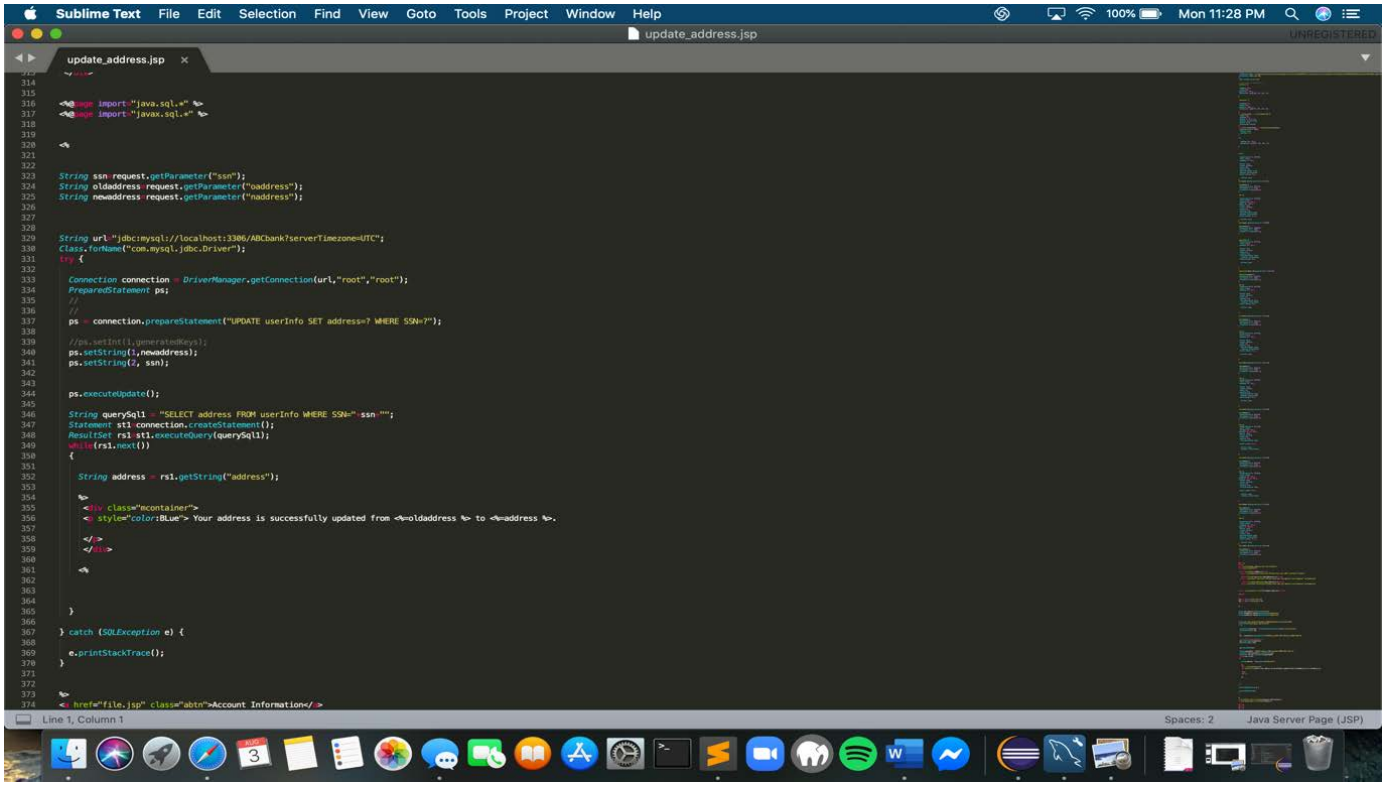
From homepage, if user chooses Account Information button, it will redirect to account information page where user puts ssn. Once the data is received, it displays the account information of the user.



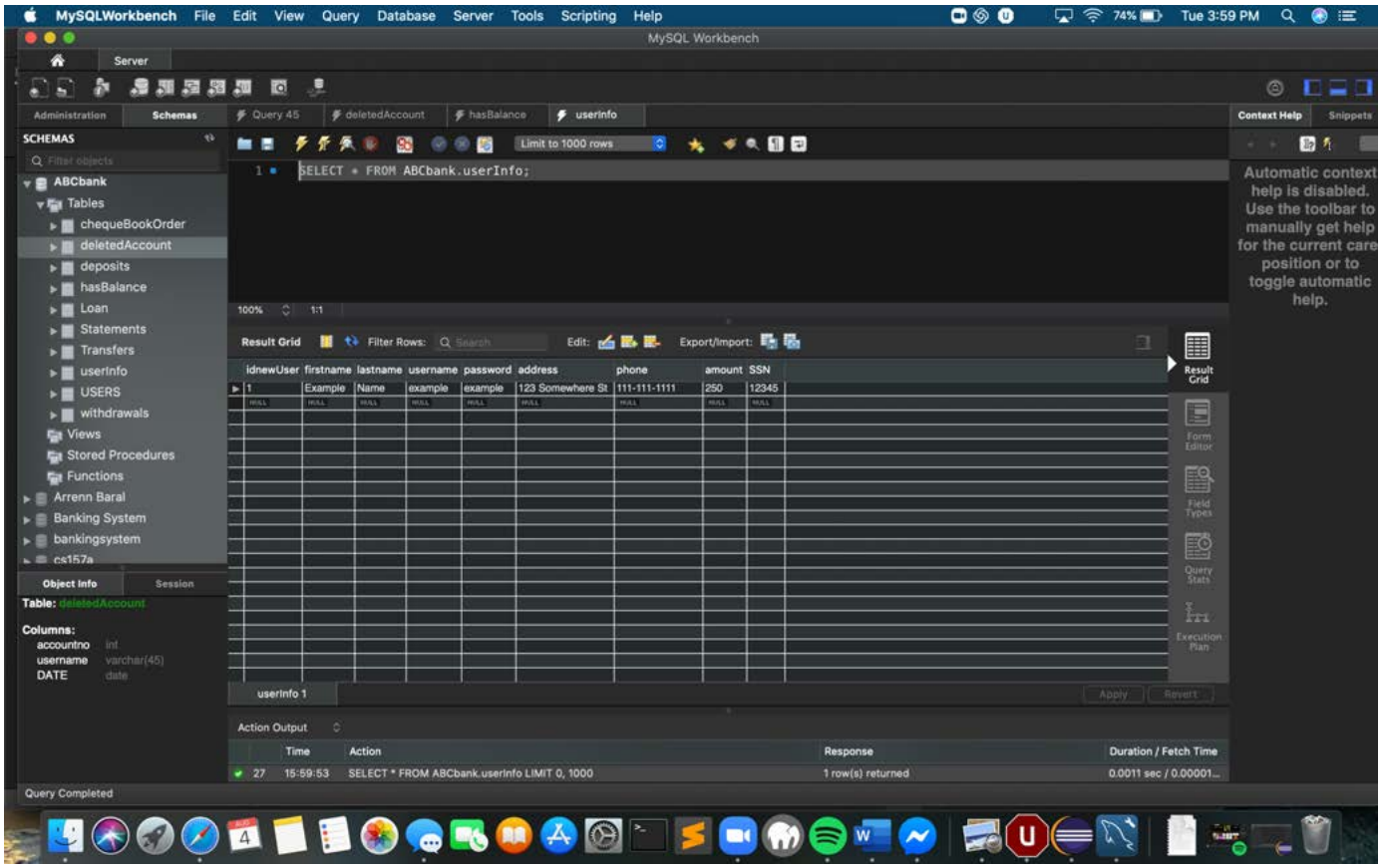
Back-end program
Once the data is submitted, this code will help to display information of user with SSN given.



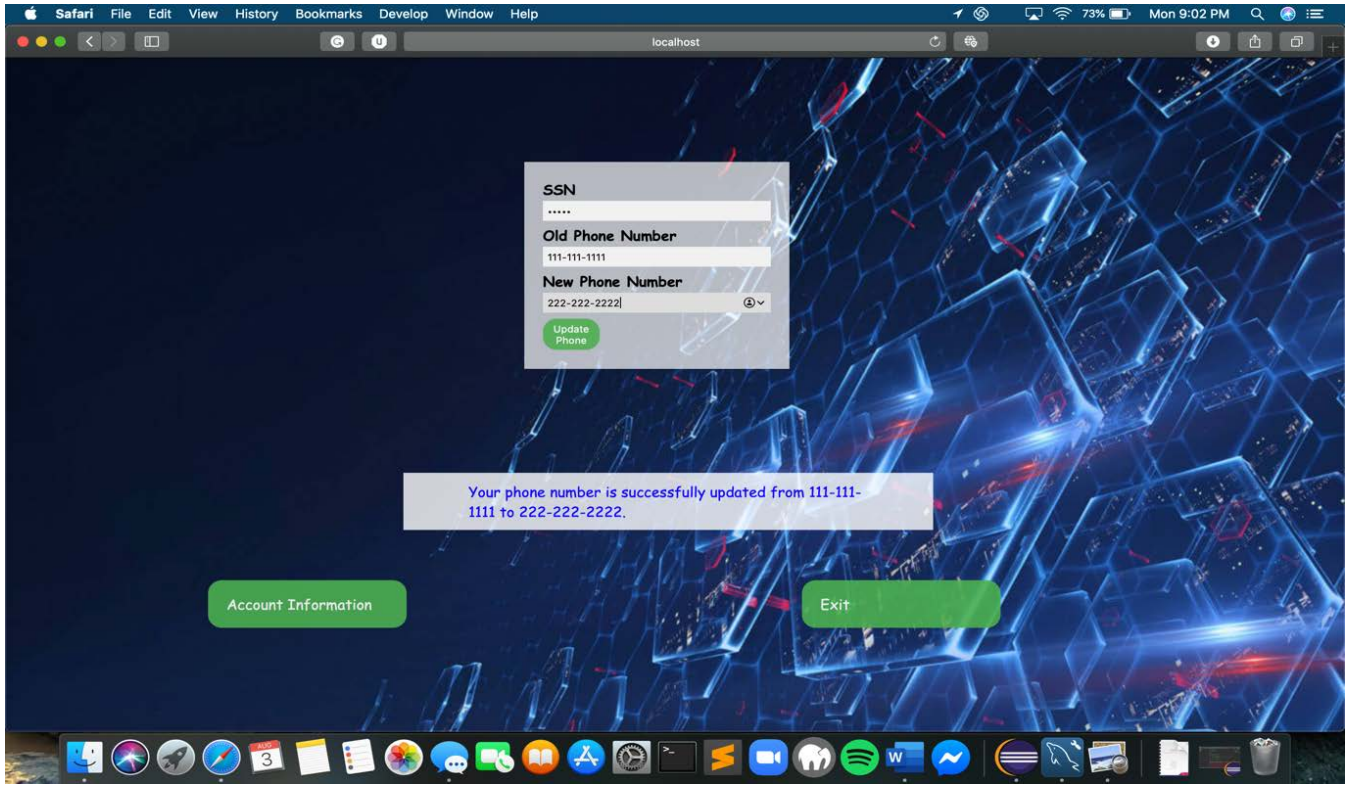
Front end for update address
If user wishes to update their address, then they can click on update address button from account information. They will be asked to put ssn, old address and new address. Message will be prompted to the user about the changed address.



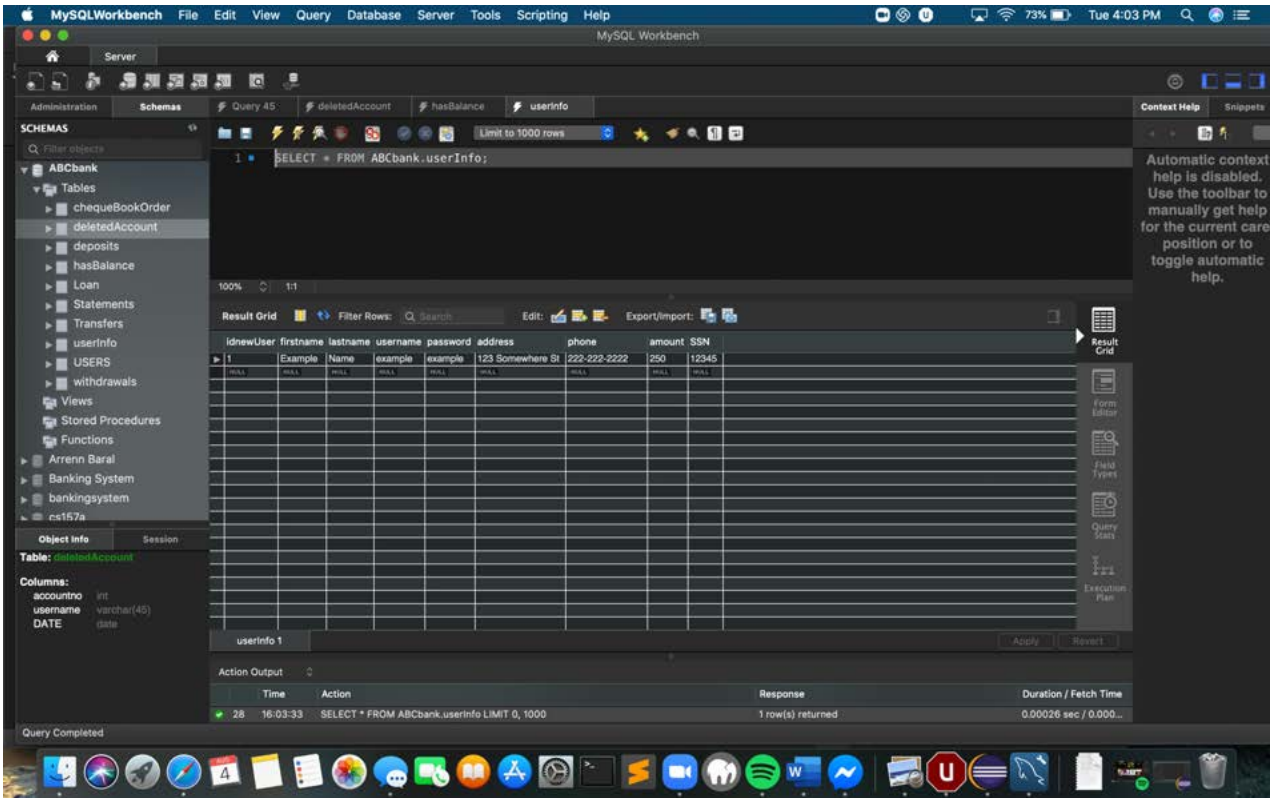
Back-end program
Once the data is submitted, this code will help to display information of user with SSN given with updated address.



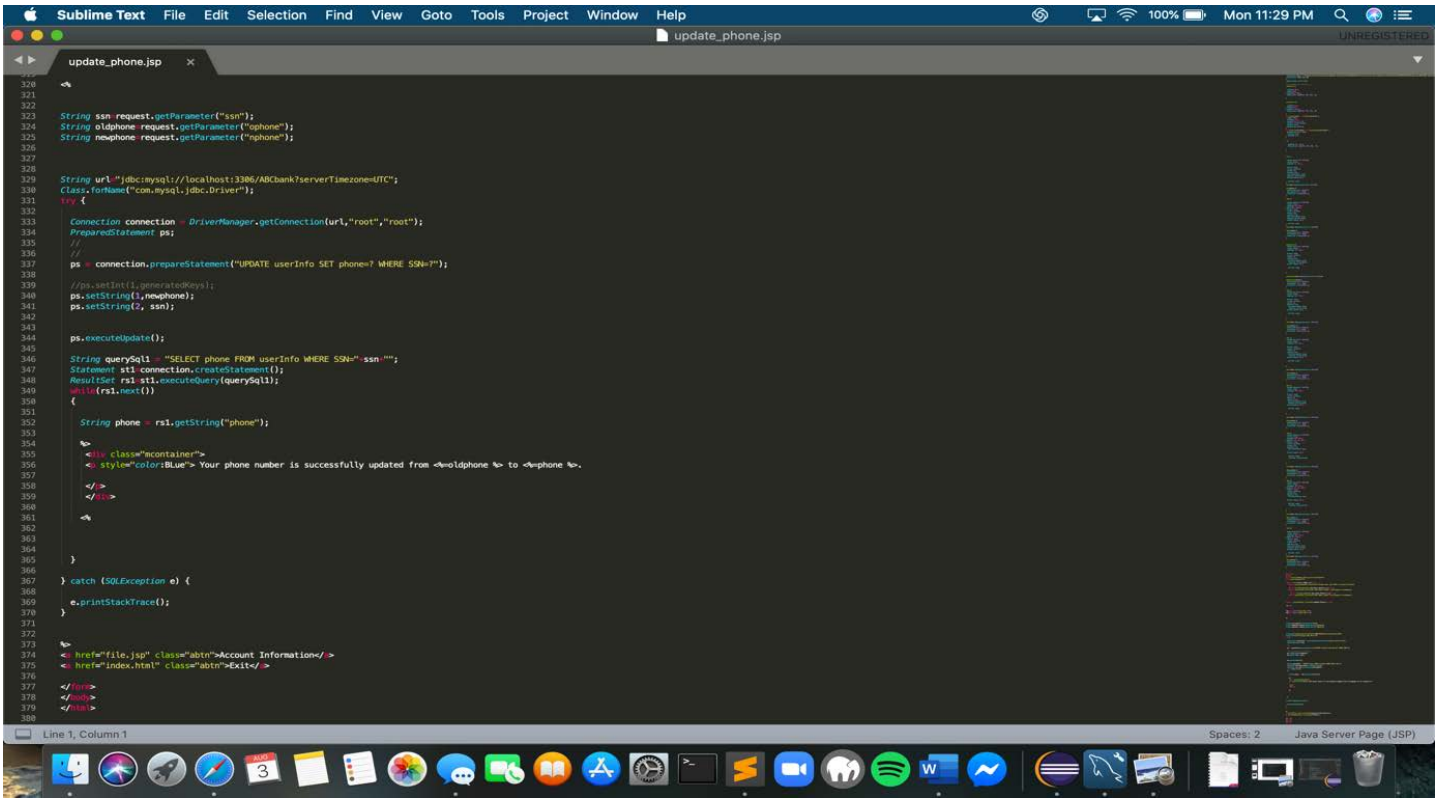
As we can see, the updated address record has been updated on the table.



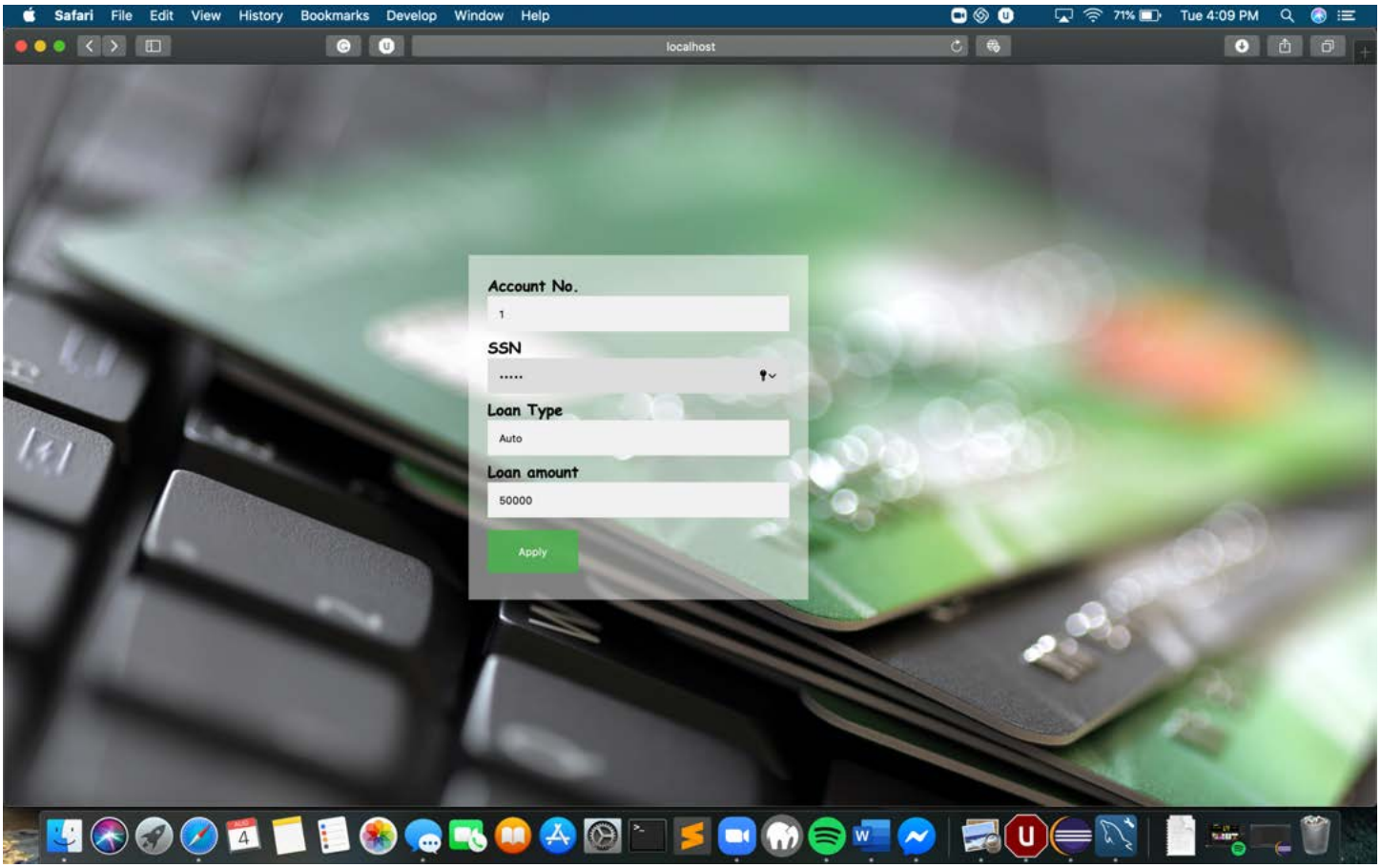
Front end for update phone
If user wishes to update their phone, then they can click on update phone button from account information. They will be asked to put ssn, old phone and new phone. Message will be prompted to the user about the changed phone.



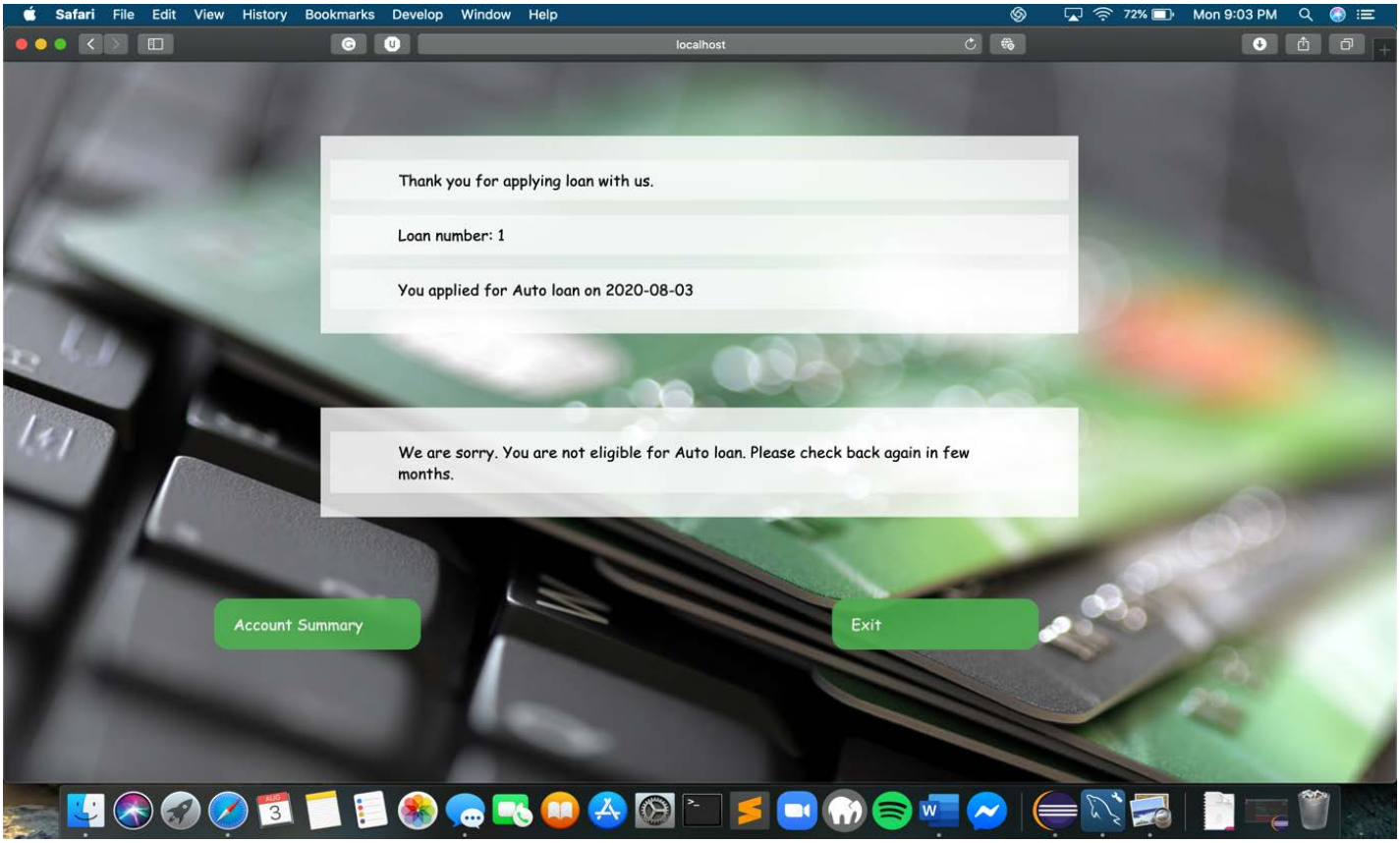
As we can see, the updated phone record has been updated on the table.



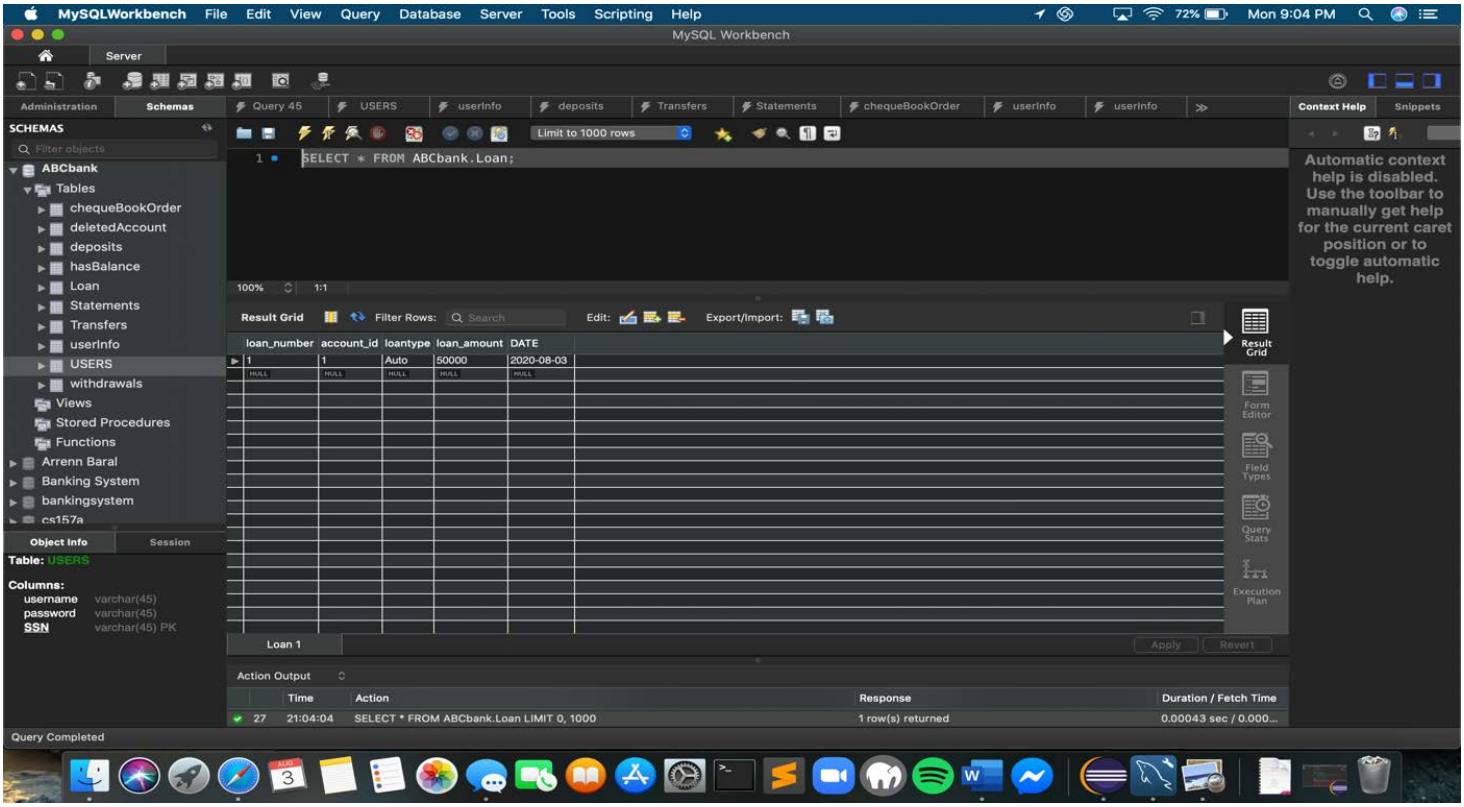
Back-end program
Once the data is submitted, this code will help to display information of user with SSN given with updated phone.



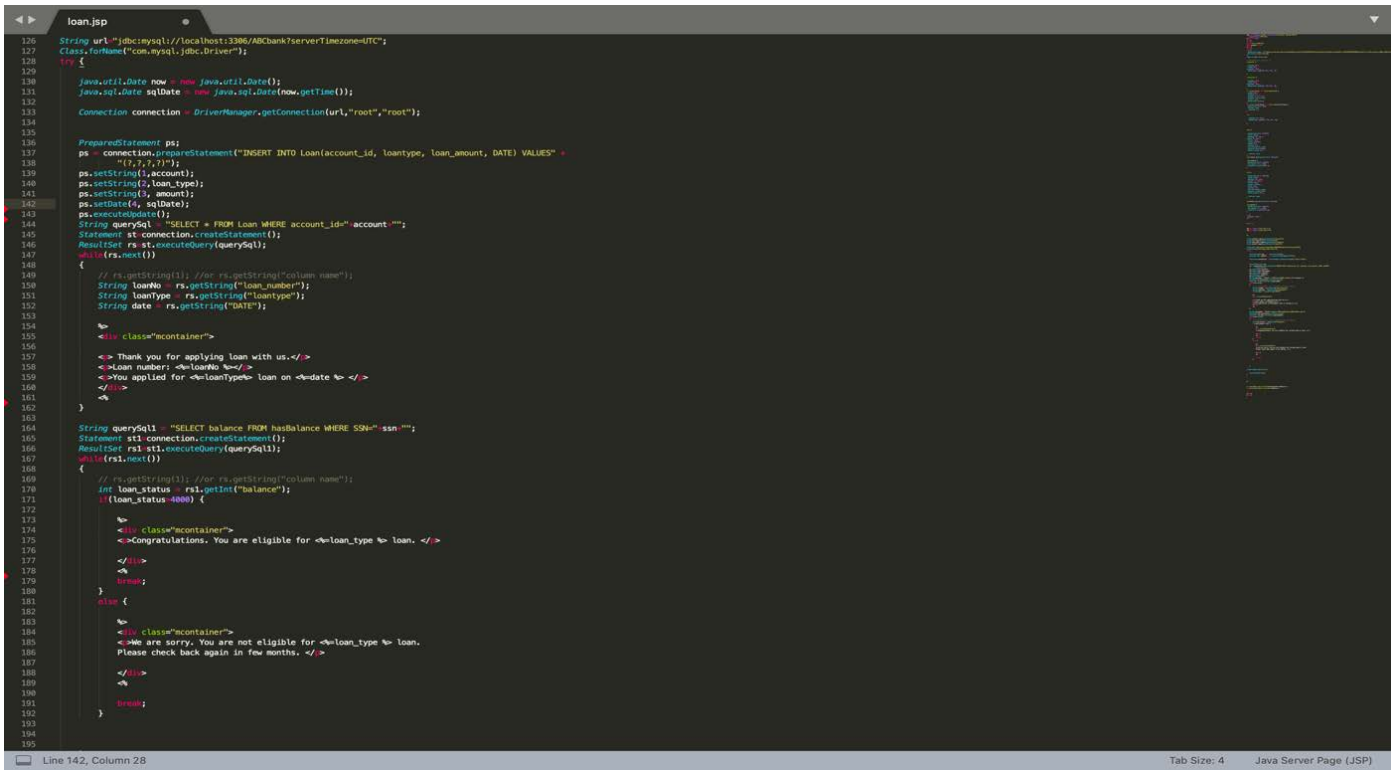
From homepage, if user chooses Apply Loan button from the top right, it will redirect to loan page where user puts account number, and ssn.



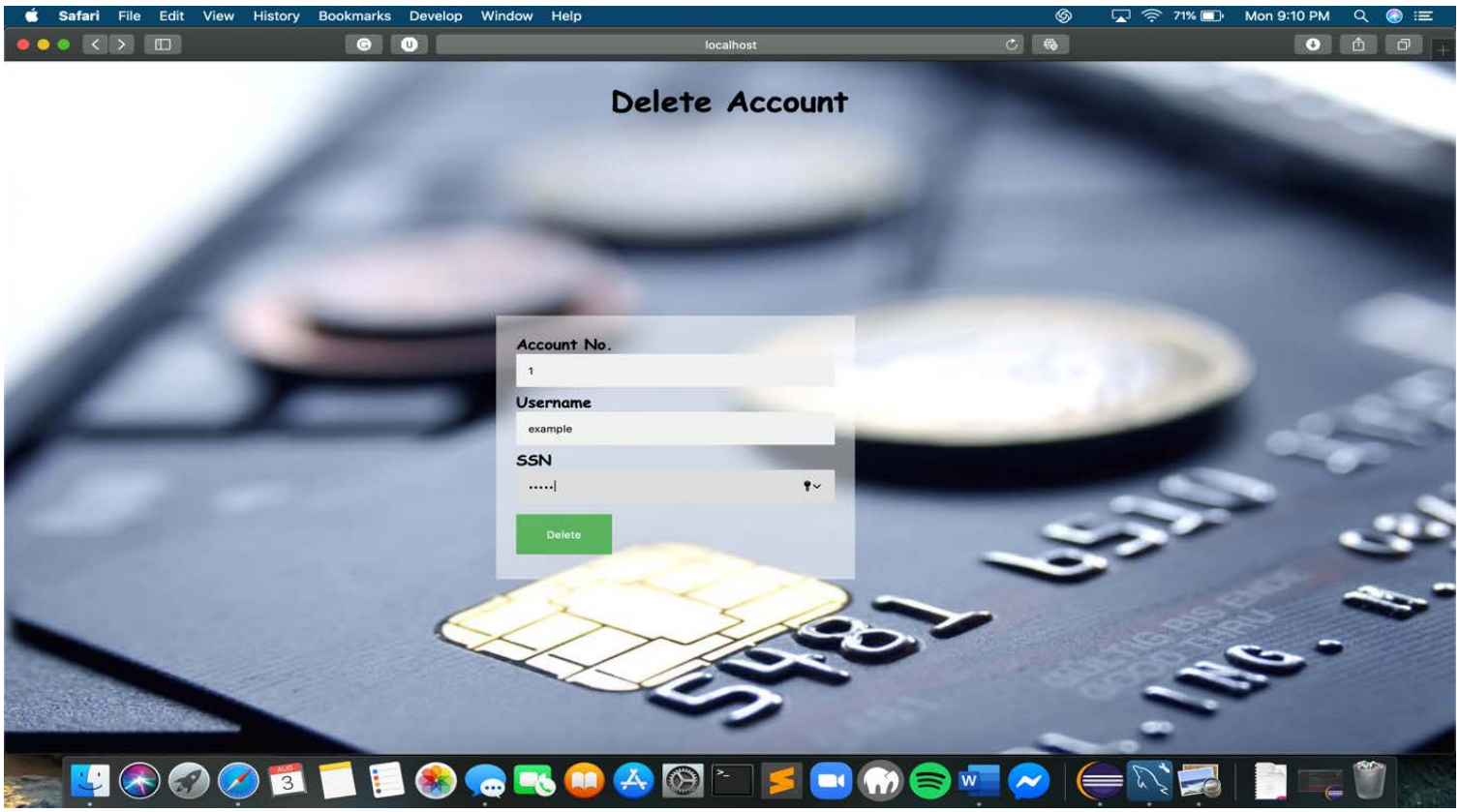
Once they apply for loan, it will give the instance result. In the back end program, I have the requirement that if a user has more than 4000 balance, they get accepted for the loan.



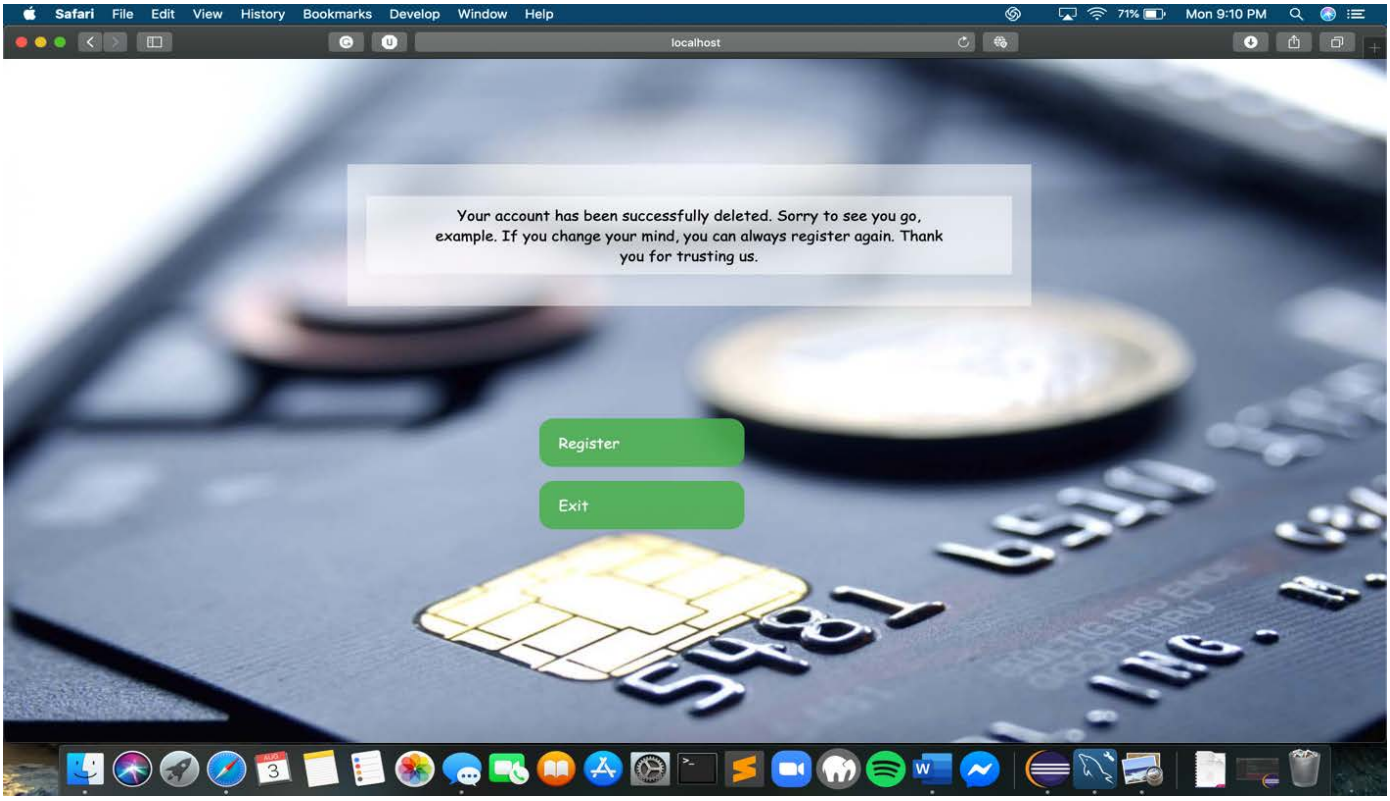
As we can see, the loan record has been recorded on the table.



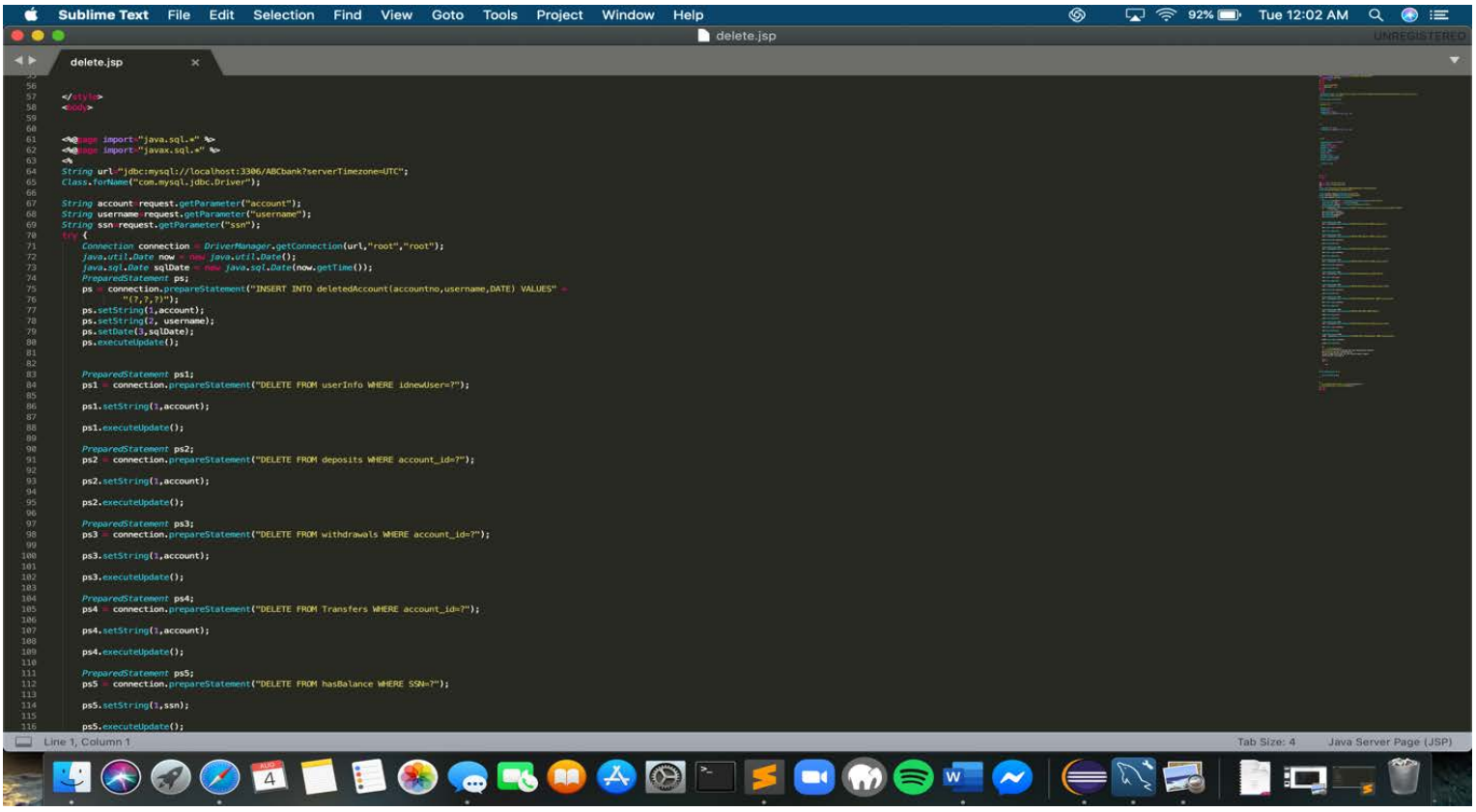
Back-end program
Once the data is submitted, this code will help to record loan information from user with SSN given, and display the result.

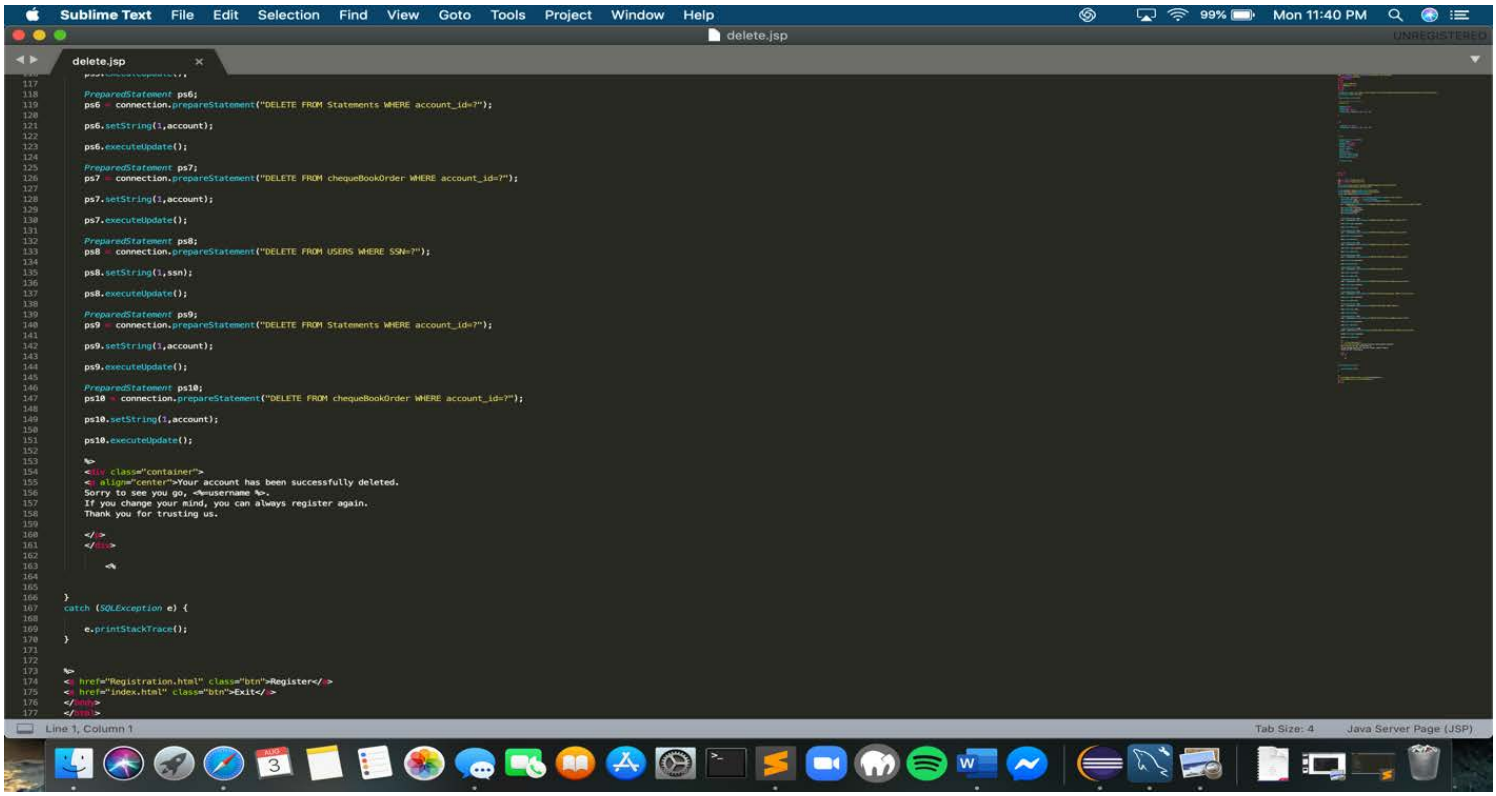


From homepage, if user chooses delete account button from the bottom left, it will redirect to delete account page where user puts account number, ssn and username..

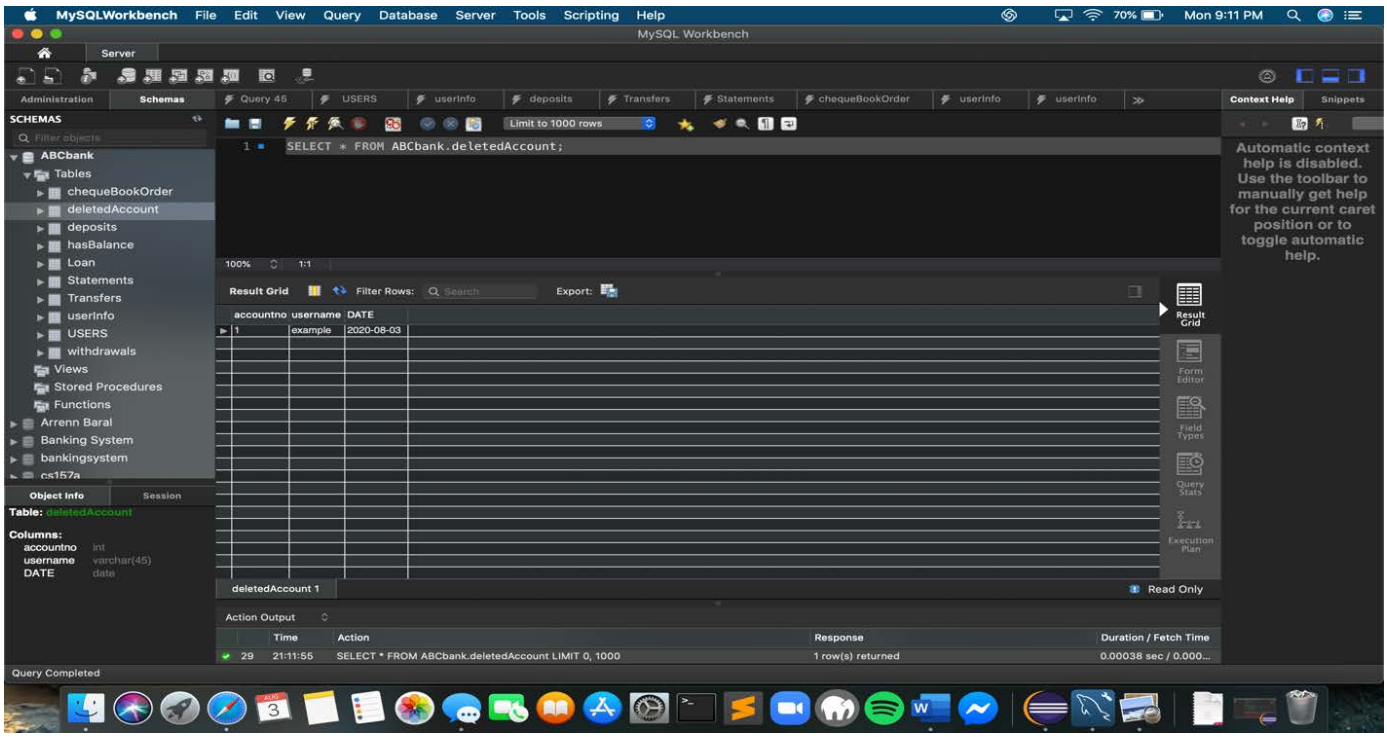


Once the data is received, it gives the message that the account has been deleted. Also, given the option to register again.

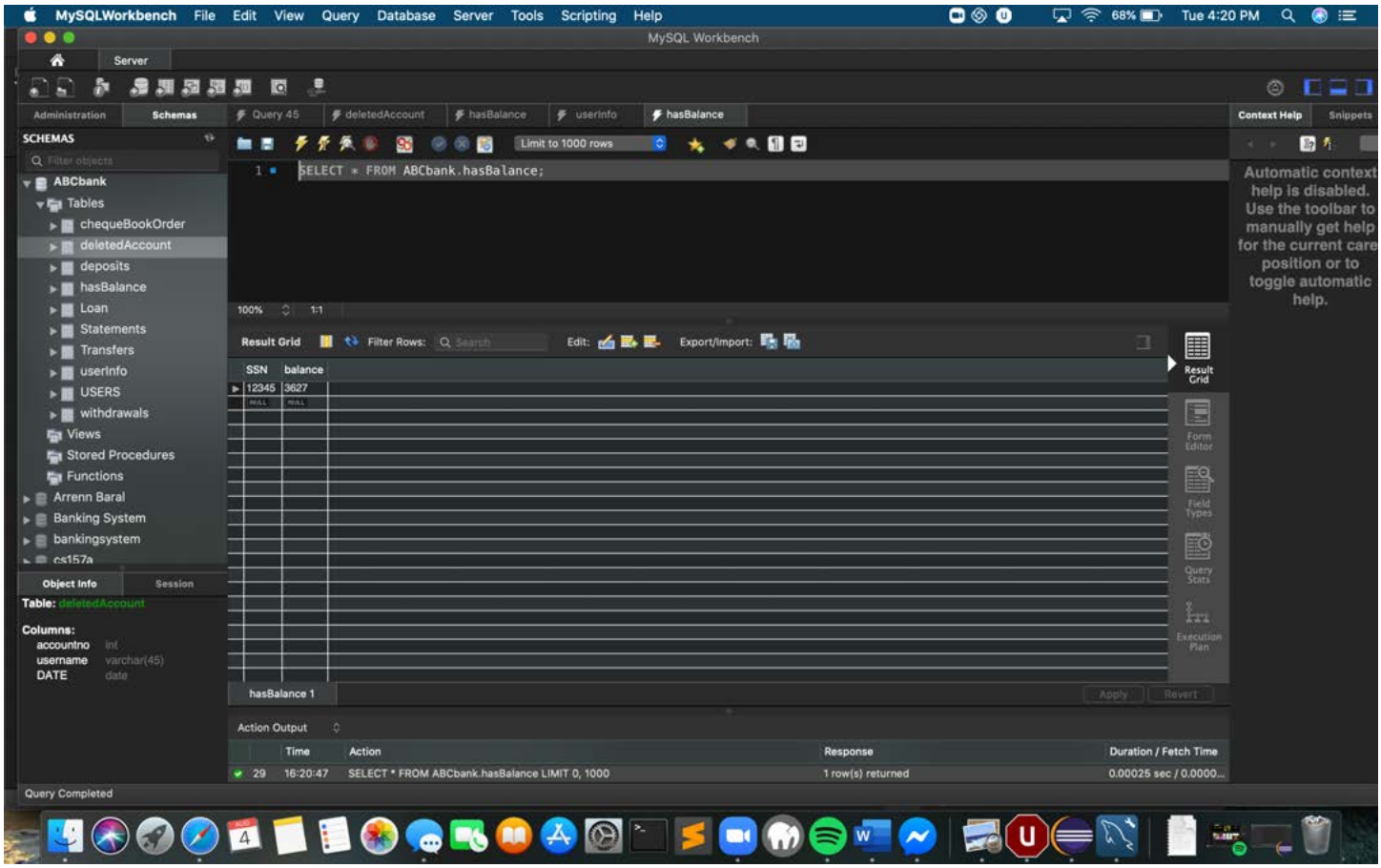




Back end
This chunk of code will help to delete user's information from all the tables where the given SSN is



As we can see, the deleted account record has been recorded on the table.



So, after all the transaction, the total balance of this user is 3627. After the delete account option, every record will be cleared out. They cannot recover.

Project Conclusion

Arrenn Baral

Throughout the summer semester, I got an opportunity to get exposed to database management system which was completely new topic. We learned from the basics of database management system to the complex one. Learning DB management is one of the important skills in the tech industry nowadays.

We started from the design of the project and ended with great implementation at the end of the summer session. We learned to use Github, Node JS, CSS and HTML to implement our design. These are very powerful tool to work on project. It was worth learning. This is my first web-based application which I feel pretty great about it.

One of the great lessons from this project is working with team member. Later in the future, we have to work with the team and this project helped to us to learn with the team. Collaboration is one of the hard things to implement, but it can be if worked together with teaming mindset.

Prakriti Basyal

Implementing and designing the banking database application of the project I learned multiple ways to handle the software system. Through our web application project, I get to learn more about the Mysql codes, HTML, CSS, and its implementations. This is pretty new to me to develop a web application, so I got a hand on experience dealing with ide and the server as well as developing the application. I have a more sense of databases now through this project. Collaborating with my teammate was one of the crucial things to finish the project. Time management was needed so I learned that it could help me in the future to manage my time for successful results. I had a little knowledge of using GitHub but through this project, I get to explore more about Github.

Finally, in this short amount of time this project, I have gained more knowledge and also have more ideas about the database management systems and the other tools to complete the project. It was challenging to learn all at once but learning this is going to be helpful for my future.

Future Implementation

For the future improvements, we are planning on extra work on design and convenience to provide great user interface and experience. Right now, user needs to put their social security number and account number to make any kind of transactions or features. To reduce that redundancy, we will be implementing dynamic programming where user can update and see right away. Also, features like:

Recover account

Right now, user cannot recover their account once deleted. We will add this feature in future.

Better home page

User might experience redundancy with our system right now. So, we will design in a way that user experience easy interface.

Activity log

User will be able to see activity log from the certain date.

Joint Account

If user wishes to get a joint account like husband and wife, then they can request for join account.

Overall, in the future, our banking system will sure have great user experience. We try to work from user's perspectives. We need to think from user point of view to make our system to the easy and convenient. Since we chose banking system as our project and it deals with money, we need to win users trust and provide transparency with tiny details. It was a great opportunity to get to know the back work of banking system. Now we know how bank deals with the system and makes our life easy.