CS 6360 Database Design

Project Report

Navaneeth Venugopala Rao (nbv140130)

Nikhil G Rao (ngr140030)

Prajwal Halasahally Keshavar (pxh140930)

Introduction

This system is designed to create convenient and easy-to-use software for oil traders who are trying to buy and sell oil for their clients. Main motivation for this project is to increase accountability and maintainability of oil trader's data.

System Description

The system will be a three-tier architecture system where users will interact with web server. Web servers will utilize database server and return the result.

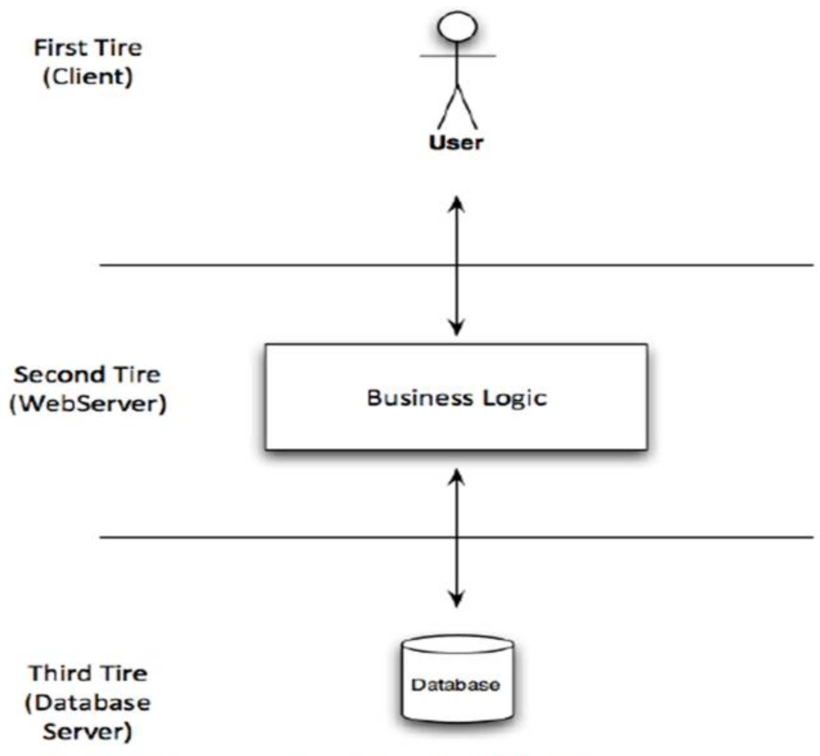


Figure: System Architecture

MVC (Model View Controller)

To make the application more robust, model view controller architecture is used. It is logically split into 3 parts Model, View and Controller.

Models

In this project, models are of two kinds,

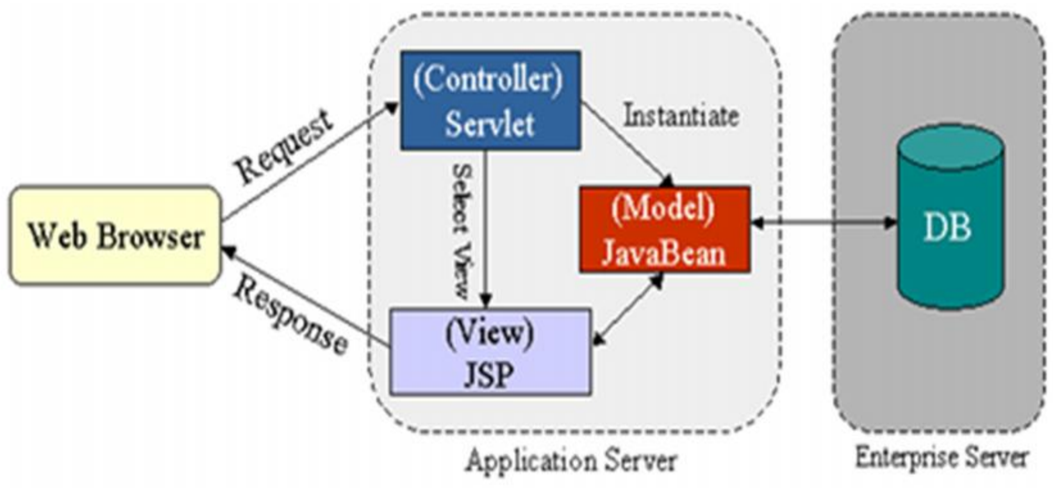
1. **Domain** – These classes represent the row of a table and relation among them.
2. **Service** – These classes encapsulate business logic of the application and manage data retrieval and insertion.

Controllers

Controllers control flow of a user request. For domain specific isolation, three packages are created to handle three different types of user logic.

Views

Views are responsible to display the final output. Here views are written in JSP (Java Server Pages) using JSTL (JSP Standard Template Library).



- **Database Management System**: MYSQL
- **Application Server**: Apache Tomcat
- **Web Scripting Language:** Java
- **Java Framework**: J2EE (Servelts and JSP)
- **Java Script and CSS Framework**: JQuery and Twitter Bootstrap
- **Working Environment**: Windows

User Description
There are 3 types of users in the system-

1. **Client**
   This user is allowed to login into the online system and specify the amount of oil he/she wants to buy or sell. He/she should also specify the whether they want to pay the commission in oil or in cash.

2. **Trader**
   This user is allowed to perform a transaction of behalf of the client. They are also allowed to cancel certain payments and oil transaction.

3. **Manager**
   The users who are assigned this role are allowed to view daily, weekly and monthly transactions.

Data Requirements

DR1:
Each client has a unique client id generated by the system, a name (first and last), a phone number, a cell-phone number, an e-mail address, and an address (including street address, city, state, and zip code).

DR2:
Each client is assigned to one of two different levels based on his or her past transaction volume. Once a client trades more than 30 barrel in any single month, the client is classified as a "Gold" customer and is charged a different commission rate for all subsequent transactions. Otherwise, the client is classified as "Silver".

DR3:
When a client wants to make a transaction, the client logs into online system and specifies the amount of oil he/she wants to buy or sell. If the client wants to sell oil, the system should automatically check if the client has enough oil stored by the company to satisfy the client's request.

DR4:
The client also needs to specify whether he or she wants to pay the commission for the transaction in oil or cash. Based on the client's choices, the system places the order. The system calculates the transaction commission based on the client's classification. If the transaction fee is paid in oil, the system automatically adjusts the amount of oil left in the customer account. On the other hand, if the customer chooses to pay the commission in cash, the system must automatically compute the fee based on current oil prices.

DR5:
The value of the transaction (e.g., the value of the oil bought or sold), the date of the transaction, and the commission paid should be stored separately for each transaction.

DR6:
From time to time, clients will pay money to settle their transaction costs. For each payment transaction, the amount paid, the date, and the information related to the trader who accepted the payment are to be stored.

DR7:
In some cases, traders may want to cancel certain payment and oil transactions. Although the system should allow such cancellations, logs should be stored for such cancelations for auditing purposes.

DR8:
In the final phase of the software, a web based system has to be developed where a trader can issue transactions for a client, can search the client history for specific client based on name, address and etc. In addition, you should provide an interface for the manager that can give aggregate information for daily, weekly and monthly total transactions based on the dates entered by the manager. Finally, clients can also log into system to issue trades for themselves.

Functional Requirements

FR1:
The system shall allow the clients to buy or sell oil.

FR2:
The system verifies the client/trader/manager through a login password.

FR3:
The system should allow the client to pay the commission in cash or in oil.

FR4:
The system should allow the traders to cancel the client’s transaction.

FR5:
The system should allow the trader to search for a client based on name and address.

FR6:
The system should allow the manager to view the daily, weekly and monthly transactions.

Design of the Database Schema

ER Design
(Attached at the end)

Mapping to Relational model

Relations

Address (stname, city, state, zipcode);

Belongsto (clientid, category);

Client (id, userid, fname, lname, emailed, cellnum, phnum, oilbalance, cashbalance);

Clienttype (category, comrate);

Client_trader_transaction_history (clientid, traderid, transactionid, date, costoftransaction, comcharge, settledflag);

Inventory (productid, oilrepo, priceperbarell, cashrepo);

Livesin (clientid, stname);

Login (userid, password, roleid);

Role (id, desc);

Trader (id, name, phnum, userid);

Transaction (id, quantity, buysell, costoftransaction, comtype);

Foreign Key

Belongsto.Category refers to **Clienttype.Category**

Client.Userid refers to **Login.Userid**

Client_trader_transaction_history.Traderid refers to **Trader.Id**

Client_trader_transaction_history.Transactionid refers to **Transaction. Id**

Client_trader_transaction_history.client_id refers to **Client. Id**

Livesin.Stname refers to **Address.stname**

Login.Roleid refers to **Role.Id**

Trader.Userid refers to **Login.Userid**

Decomposition to normalized form

The designed schema is in a normalized form.

Example: Type of Client (gold, silver) is stored in client type table rather than client table. If we had stored in client table itself and if only one tuple is present for a particular client type and when we delete this client we would loose information about that particular category itself.

Functional Dependencies

Address

Stname, Zipcode - > City, State, Zipcode

Belongsto

Clientid - > Category

Client

Id - > Userid, fname, lname, emailed, cellnum, phnum, oilbalance, cashbalance

Clienttype

Category - > Comrate

Client_Trader_Transaction_History

Clientid, Traderid, Transactionid - > date, cost_of_transaction, comcharge, settledflag

Inventory

Productid - > Oilrepo, priceperbarell, cashrepo

Login

Userid - > Password, roleid

Livesin

Clientid - > Stname

Role

Id - > Desc

Trader

Id	- >	Name, phnum, userid

Transaction

Id	- >	Quantity, buysell, costoftransaction, comtype

Queries using prepared statements to prevent SQL Injection attacks:

We have written queries as part of PreparedStatement. A SQL statement is precompiled and stored in a PreparedStatement object. This object can then be used to efficiently execute this statement multiple times. One of the main feature of a PreparedStatement object is that, unlike a Statement object, it is given a SQL statement when it is created. The advantage of using such a mechanism is that in most cases, this SQL statement is sent to the DBMS right away, where it is compiled. As a result, the PreparedStatement object comprises not just a SQL statement, but a SQL statement that has been precompiled. This means that when the PreparedStatement is executed, the DBMS can just run the PreparedStatement SQL statement without having to compile it first and this beneficial in all cases.

The SQL statements used in a PreparedStatement is precompiled on the driver and from that point on, the parameters are sent to the driver as literal values and not executable portions of SQL; and hence no SQL can be injected using a parameter.

Further, we are able to handle user input as the content of a parameter and also, we are not using SQL command by joining strings together which makes vulnerable even when we use PreparedStatement.

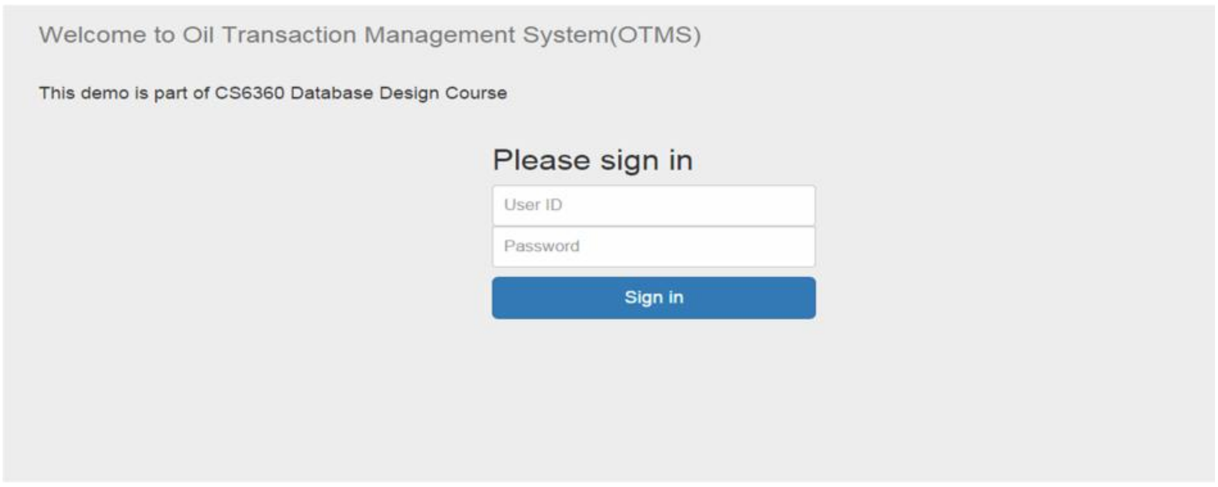
To give you an example,

```
PreparedStatement pstmt = con.prepareStatement("update
belongsto set category=\"gold\" where clientid=(?)");

pstmt.setInt(1, 5)

Resultset rs = pstmt.execute();
```

Demonstration: Screen Shots

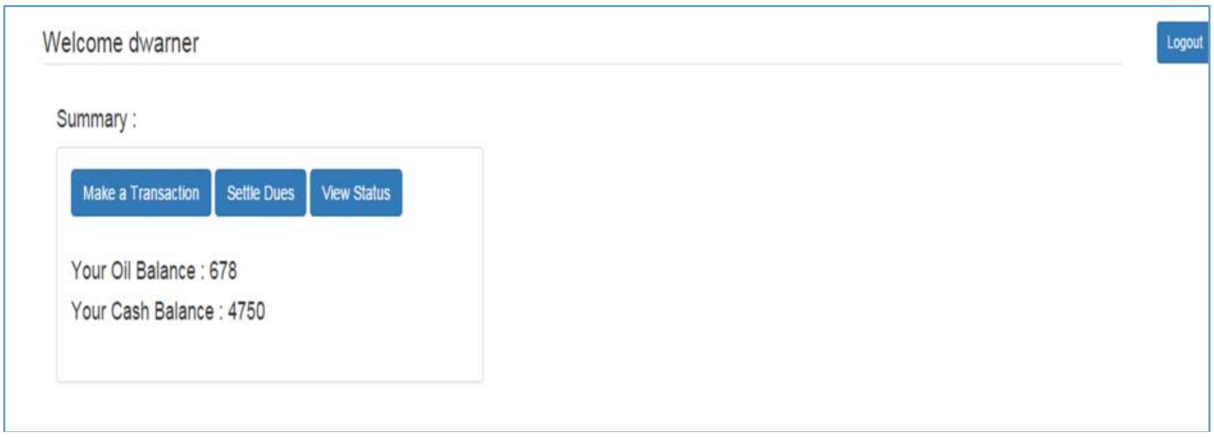


Welcome to Oil Transaction Management System(OTMS)

This demo is part of CS6360 Database Design Course

Please sign in

Login Page



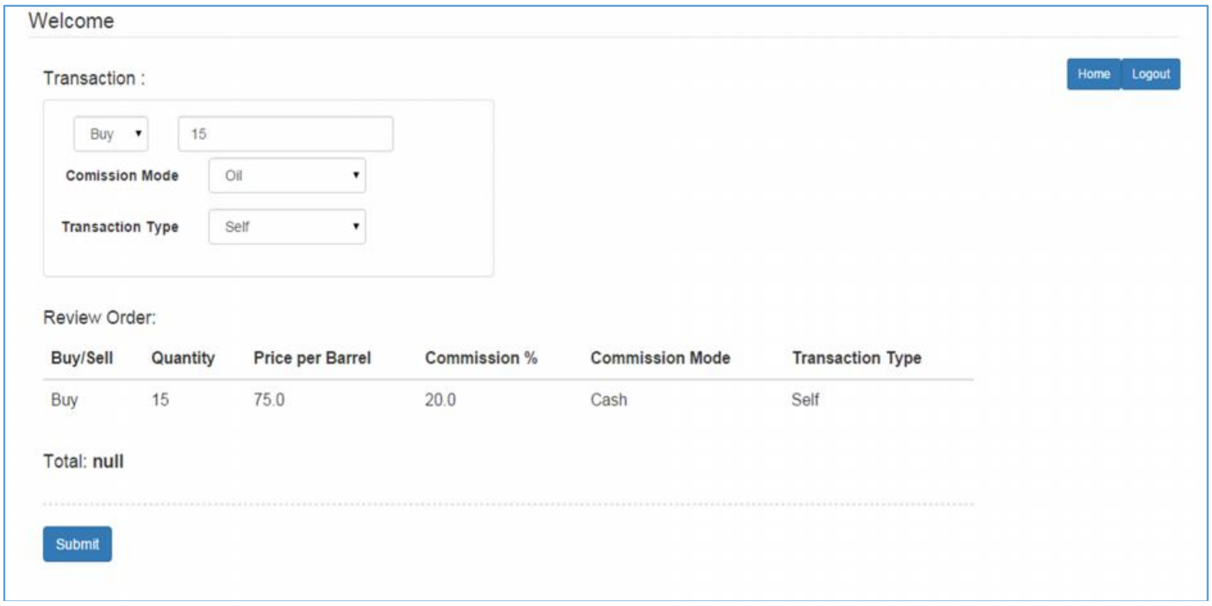
Welcome dwarner

Summary :

Your Oil Balance : 678

Your Cash Balance : 4750

Client Home page



Welcome

Transaction :

Buy 15

Comission Mode: Oil

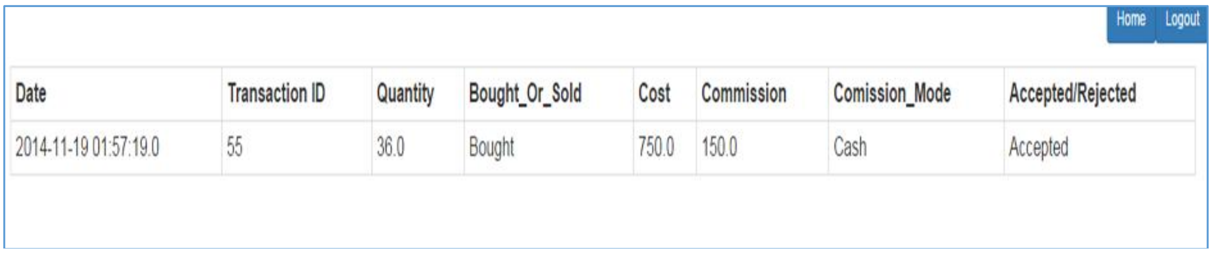
Transaction Type: Self

Review Order:

Buy/Sell	Quantity	Price per Barrel	Commission %	Commission Mode	Transaction Type
Buy	15	75.0	20.0	Cash	Self

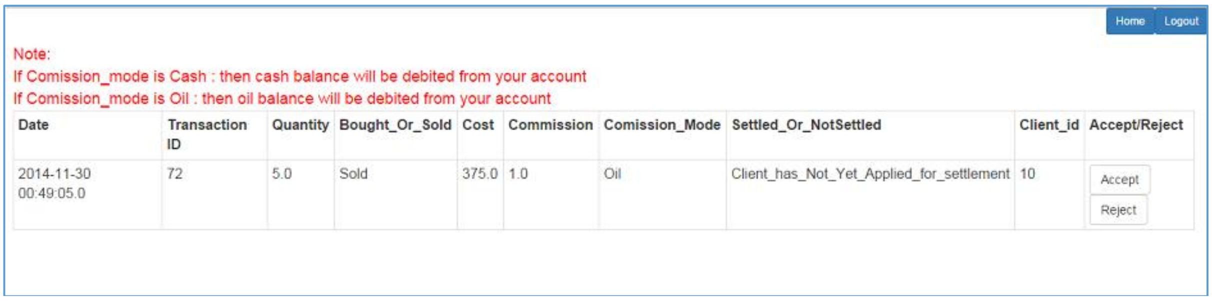
Total: **null**

Client Transaction Page



Date	Transaction ID	Quantity	Bought_Or_Sold	Cost	Commission	Comission_Mode	Accepted/Rejected
2014-11-19 01:57:19.0	55	36.0	Bought	750.0	150.0	Cash	Accepted

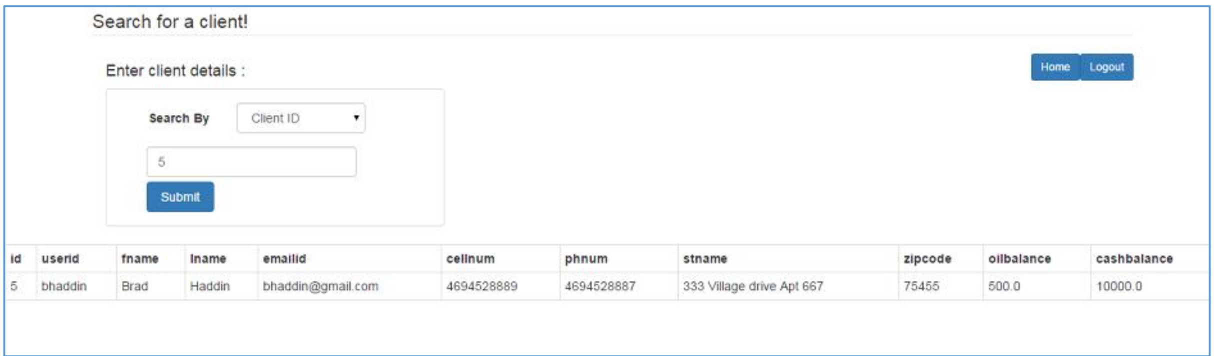
Client View Status Page



Note:
If Comission_mode is Cash : then cash balance will be debited from your account
If Comission_mode is Oil : then oil balance will be debited from your account

Date	Transaction ID	Quantity	Bought_Or_Sold	Cost	Commission	Comission_Mode	Settled_Or_NotSettled	Client_id	Accept/Reject
2014-11-30 00:49:05.0	72	5.0	Sold	375.0	1.0	Oil	Client_has_Not_Yet_Applied_for_settlement	10	Accept Reject

Trader Accept/Reject Page



Search for a client!

Enter client details :

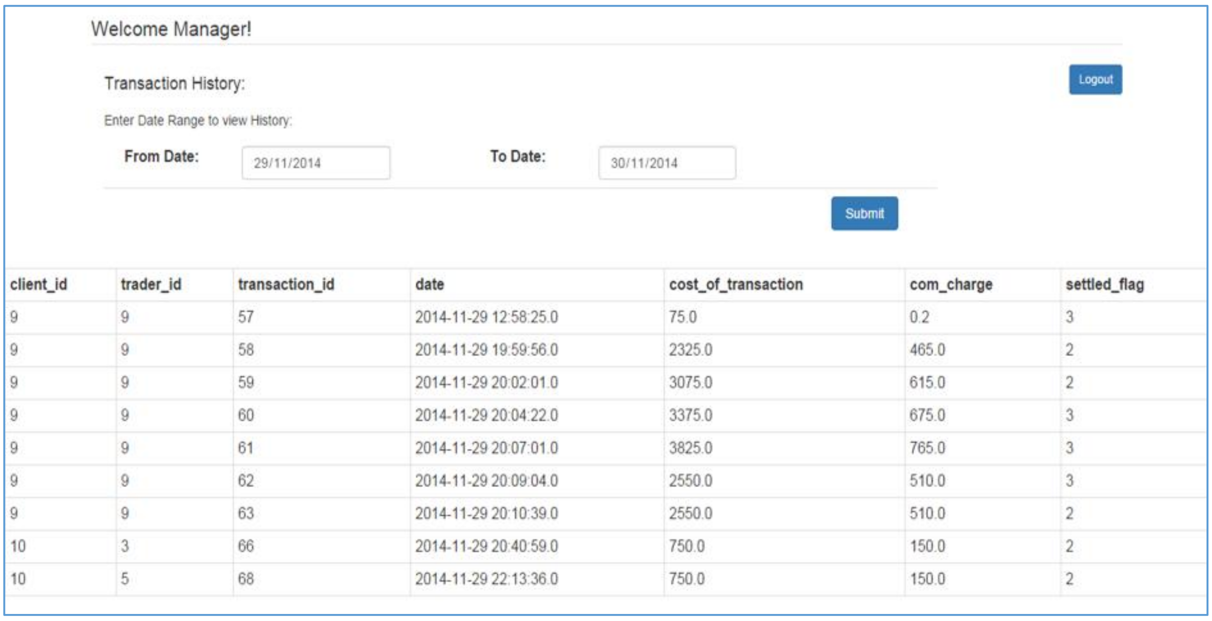
Search By: Client ID

5

Submit

id	userid	fname	lname	emailid	cellnum	phnum	stname	zipcode	oilbalance	cashbalance
5	bhaddin	Brad	Haddin	bhaddin@gmail.com	4694528889	4694528887	333 Village drive Apt 667	75455	500.0	10000.0

Trader’s Search Client Page



Welcome Manager!

Transaction History:

Enter Date Range to view History:

From Date: 29/11/2014 To Date: 30/11/2014

client_id	trader_id	transaction_id	date	cost_of_transaction	com_charge	settled_flag
9	9	57	2014-11-29 12:58:25.0	75.0	0.2	3
9	9	58	2014-11-29 19:59:56.0	2325.0	465.0	2
9	9	59	2014-11-29 20:02:01.0	3075.0	615.0	2
9	9	60	2014-11-29 20:04:22.0	3375.0	675.0	3
9	9	61	2014-11-29 20:07:01.0	3825.0	765.0	3
9	9	62	2014-11-29 20:09:04.0	2550.0	510.0	3
9	9	63	2014-11-29 20:10:39.0	2550.0	510.0	2
10	3	66	2014-11-29 20:40:59.0	750.0	150.0	2
10	5	68	2014-11-29 22:13:36.0	750.0	150.0	2

Manager Report Page

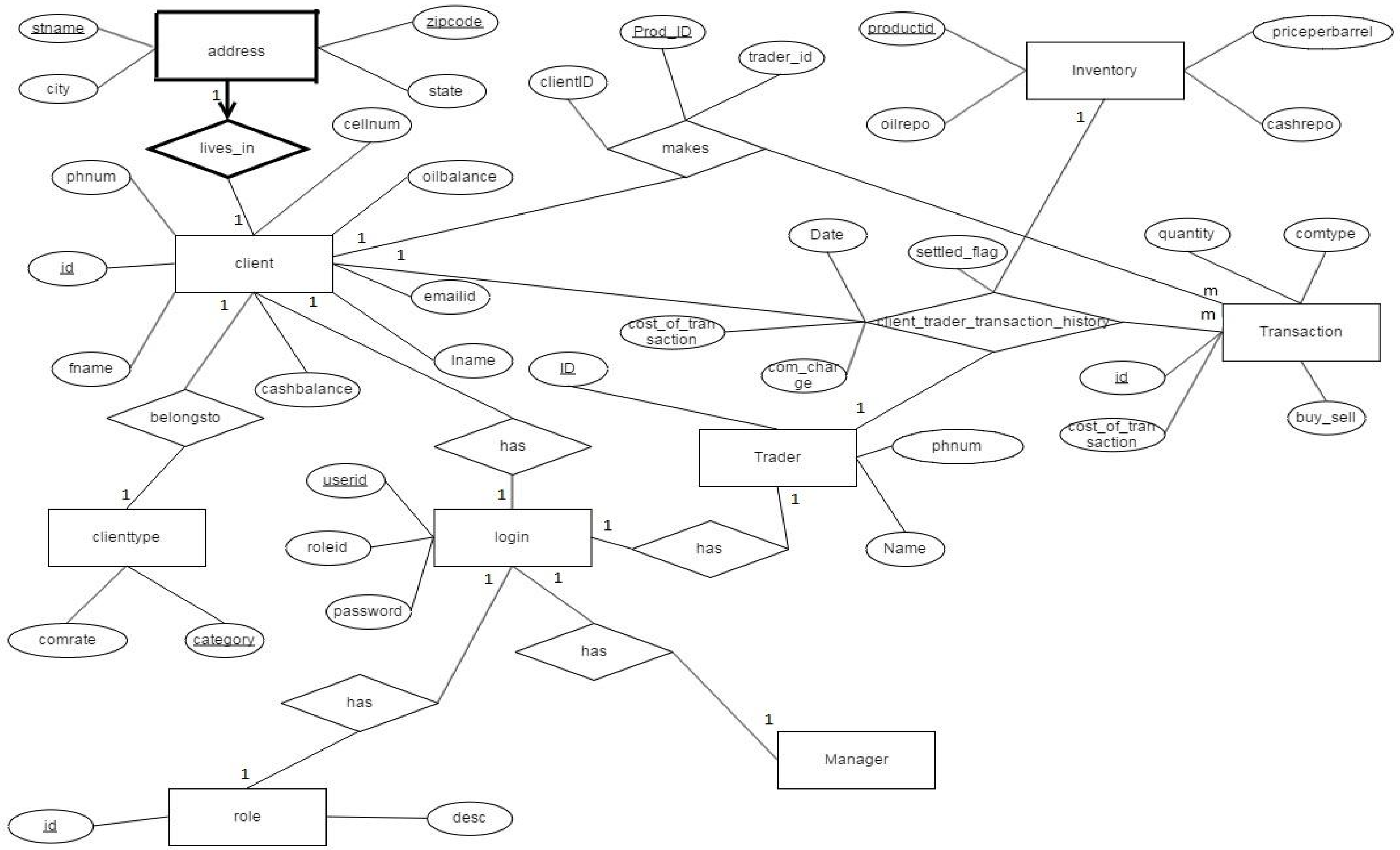
Accomplishments:

- Client, traders and Managers can log into the system.
- Client can issue a transaction – Buy or Sell and View Status of his or her previous transaction.
- Client can apply to settle his or dues from time to time.
- Traders can accept or reject a transaction.
- Trader can search a client’s address and other details by having client’s id or name.
- Managers can view the transactions history for all customers from wide time range.

References and Inspiration:

- https://github.com/fahadshaon/sirius (Fahad’s Database project Sirius)
- Overview of Prepared Statements (https://docs.oracle.com/javase/tutorial/jdbc/basics/prepared.html)
- Interface PreparedStatement https://docs.oracle.com/javase/7/docs/api/java/sql/PreparedStatement.html

ER Diagram



stname
zipcode
address
city
state
1
lives_in
cellnum
phnum
oilbalance
1
client
1
1
id
emailid
fname
1
1
lname
cashbalance
belongsto
has
1
1
clienttype
userid
login
roleid
1
password
comrate
category
1
1
has
has
1
role
id
desc
Manager
1
Prod_ID
trader_id
clientID
makes
productid
Inventory
priceperbarrel
oilrepo
cashrepo
1
Date
settled_flag
quantity
comtype
m
m
cost_of_tran saction
client_trader_transaction_history
Transaction
ID
com_char ge
id
1
cost_of_tran saction
buy_sell
Trader
phnum
1
has
Name