OilComp System

Technical Manual

December 2017

Contents

1 Database Design

The database for OilComp System was implemented in MySQL relationship Database Management System (version 5.7.20). The Entity Relationship diagram was properly designed according to the business rules provided in the begining of the project, and it may be found in A. Following in this Section, we introduce the relational schema generated from Entity Relationship Diagram, presenting the meaning of each attribute and we further we discuss about some additional objects implemented in the DBMS for this schema.

CUSTOMERS(Cid, UFname, ULname, UPhone, UCellPhone, Uemail, UUserName, UHashedPassword, CStreet, CCity, CState, CZipCode, CLevel, CQBarrels, CCredit)

TRADERS(Tid, UFname, ULname, UPhone, UCellPhone, Uemail, UUserName, UHashedPassword)

MANAGERS(Mid, UFname, ULname, UPhone, UCellPhone, Uemail, UUserName, UHashedPassword)

TRANSACTIONS(Xid, XBuyFlag, XQtbarrels, XCommissionValue, XCommissionPaymentType, Xvalue, XDate, Tid, Cid)

PAYMENTS(Tid, Cid, PAmtPaid, PDate)

AUDITING(Xid, XBuyFlag, XQTbarrels, XCommissionValue, XCommissionPmtType, Xvalue, xDate, Tid, Cid, CancelDate, TidCancel)

PRICE(PrBarrelPrice, PrDate, silverCommissionRate, goldCommissionRate)

Please find below a brief description of each relation and their respective attributes:

- CUSTOMERS: This relation stores data of the customers of the oil company.
  - Cid (int): This is the Customer ID. Although it is transparent for the final users of the system, this is the Primary Key of CUSTOMERS table. This an auto-increment attribute;
  - UFname (varchar(15)): First name of the user;
  - ULname (varchar(15)): Last name of the user;
  - UPhone (varchar(10)): Phone number;
  - UCellPhone (varchar(10)): MobileCell phone number;
  - Uemail (varchar(30)): email address of the user;
  - UUserName (varchar(20)): usernamecredential used to login in OilComp system. This is an unique attribute;
  - UHashedPassword (varchar(100)): hashed password associated to the username. This is an unique attribute;
  - CStreet (varchar(25)): Address (street) of the customer;
  - CCity (varchar(25)): Address (City) of the customer;
  - CState (varchar(10)): Address (State) of the customer;
  - CZipCode (int): Address (Zip Code) of the customer;
  - CLevel (char(1)): level of the customer (G for gold and S for Silver);
  - CQBarrels (decimal(10,2)): quantity of barrels customer still have stored;
  - CCredit (decimal(10,2)): customer's balance (in USD).
- TRADERS: This relation stores data related to the traders who work for the oil company. Note that most of the attributes of this relation (UFname, ULname, UPhone, UCellPhone, Uemail, UUserName, UHashedPassword) are exactly the same as those already mentioned in CUSTOMERS relation.
  - Tid (int): This is the Trader ID. Tid is transparent for the final users, but it is the Primary Key of TRADERS table. This is also an auto-increment attribute;
- MANAGERS: Similarly to TRADERS relation, MANAGERS stores data of the managers who work for the oil company, and most of the attributes of this relation (UFname, ULname, UPhone, UCellPhone, Uemail, UUserName, UHashedPassword) are exactly the same as those already mentioned in CUSTOMERS relation.
  - Mid (int): This is the Manager ID. Mid is transparent for the final users, but it is the Primary Key of MANAGERS table. This is also an auto-increment attribute;
- TRANSACTIONS: This relation stores information about the transactions performed by customers (under trader's support) while buying and selling oil.

  - Xid (int): This is the Transaction ID. Although this attribute is transparent for the final users, it is the Primary Key of this relation. This is also an auto-increment attribute;
  - XBuyFlag (char(1)): It indicates if the transaction is a buy or sell ('B' or 'S' respectively);
  - XQtbarrels (int): Number of barrels bought or sold in the transaction;
  - XCommissionValue (decimal(10,2)): Amount paid (in USD) as commission for a given transaction;
  - XCommissionPaymentType (char(1)): It describes how the customer paid for the commission ('O' for oil and 'C' for cash);
  - Xvalue (decimal(10,2)): The value (in USD) of the transaction (The the value of the oil bought or sold);
  - XDate (date): Date that transaction occurred;
  - Tid (int): Trader ID of the trader who supported the transaction. Usually customers may buy or sell oil by themselves. However, transactions can also may be performed through a trader behalf of a customer (in these cases Tid of the trader is stored in the record);
  - Cid (int): Id of the customer who performed a given transaction.

- PAYMENTS: This relations stores the money paid by customers to settle their transaction costs.
  - Tid (int): Trader ID of the trader who accepted the payment;
  - Cid (int): Customer ID of the customer who performed the payment;
  - PAmtPaid (decimal(10,2)): Amount (in USD) payed in the payment event;
  - PDate (date): Date that the payment occurred;

- AUDITING: This is an appendix relation, in the sense that it has no relationships with no other relation in the schema. Every time a given transaction is canceled by a trader, the record related to this transaction is dropped from TRANSACTION table and it is automatically inserted into AUDITING table. So, metadata of AUDITING relation is pretty similar to TRANSACTION, and the only additional attributes for AUDITING are described below:
  - CancelDate (date): Data that the transaction was canceled;
  - TidCancel (int): Id of the trader who canceled the transaction.

- PRICE: As AUDITING table, this is an appendix relation which does not have a relationship with any other relation in schema. PRICE table stores the price of the barrel of oil per day. Besides it stores the rate of commission for silver and gold customer levels.
  - PrBarrelPrice decimal(10,2): Price of barrel of oil in a given date;
  - PrDate (date): Date in which each prices are valid. This is an unique attribute;

- silverCommissionRate (decimal(4,3)): Commission rate applied to customers belonging to silver customer level;
- goldCommissionRate (decimal(4,3)): Commission rate applied to customers belonging to gold customer level;

Besides the table described above, the schema implemented for the OilComp System is composed by some additional objects created in MySQL DBMS:

- A hash index on CUSTOMERS table (UUserName attribute), so information related to city and zip code can be easily retrieved when shipping oil to clients;
- A hash index on CUSTOMERS table (CZipCode attribute), so inquiry performed in History Transactions page (shown in Figure 7) can be performed easier;
- A hash index on CUSTOMERS table (UUserName attribute), so information related to city and zip code can be easily retrieved when shipping oil to clients;
- Procedure checkCustomerLevel() which updates the level (Gold or Silver) or each customer according to their historically past transactions;
- Event uptCustLevel which is scheduled to run in daily basis at midnight to call checkCustomerLevel() and update customer level as previously described.

2 Software Architecture and Functionalities

In this Section we briefly describe the architecture of the OilComp system, describing the flow of screens and their functionalities.

- Login: Traders, Managers and Customers log in through the following page, as described below.

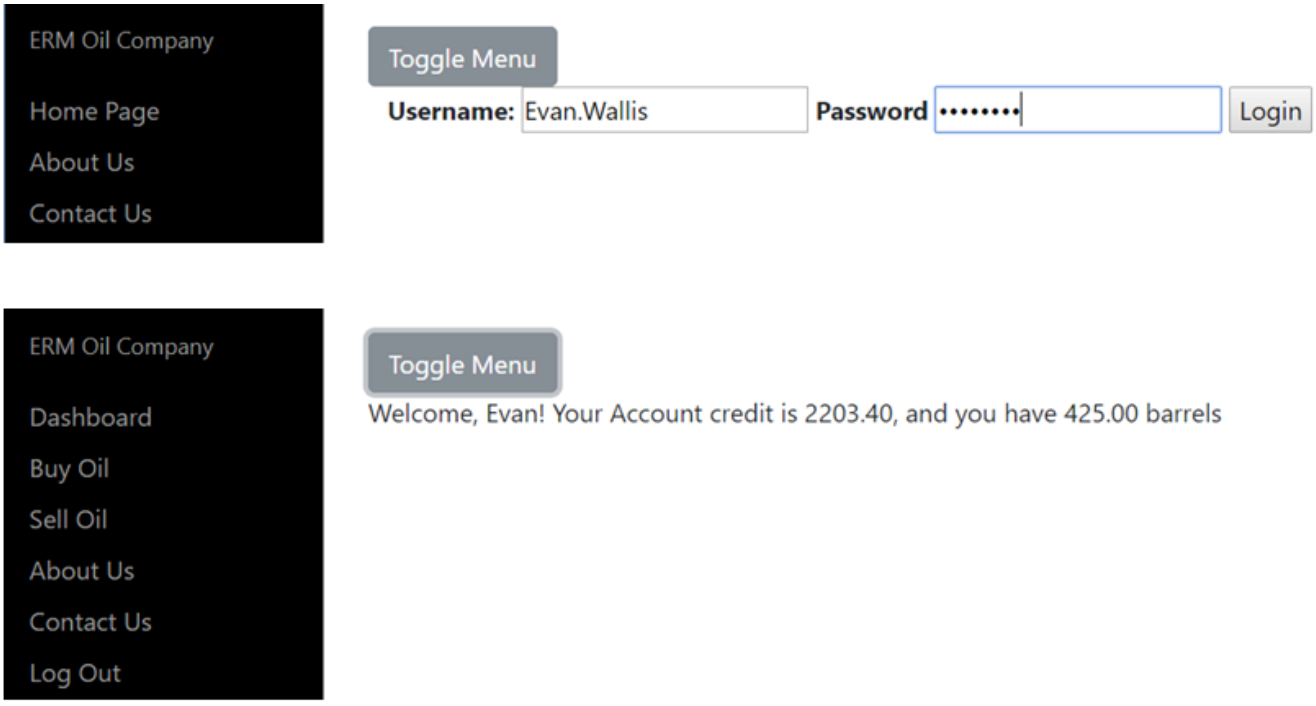

Figure 1: Login page.

- Buy: Customers and Traders can navigate and buy oil through Buy page. The input data for Customers are the number of barrels and the commission payment type, which may vary (Oil or Cash). Besides number of barrels and commission payment type, Traders should input the username who wants to perform the bought.

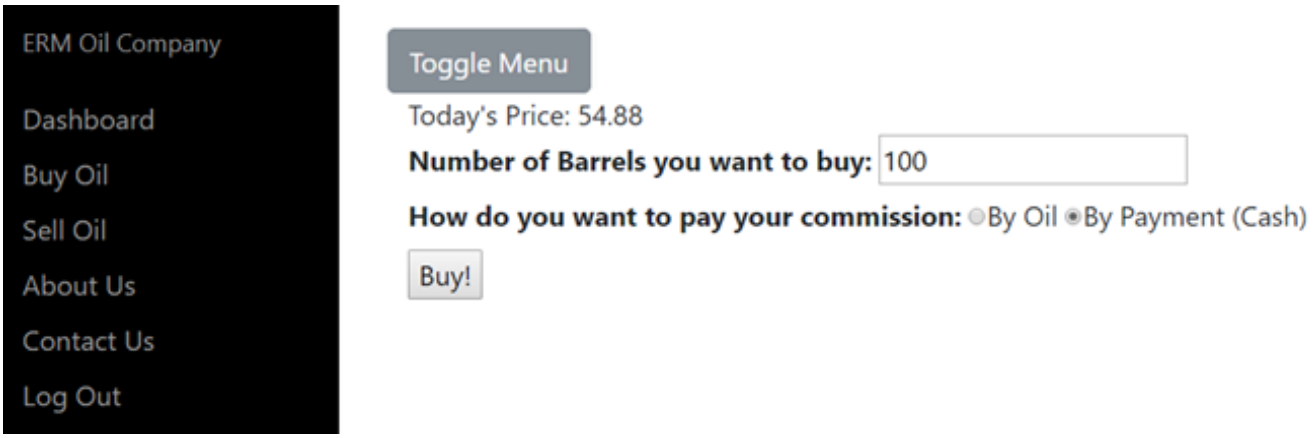

Figure 2: Page for customers buying oil.

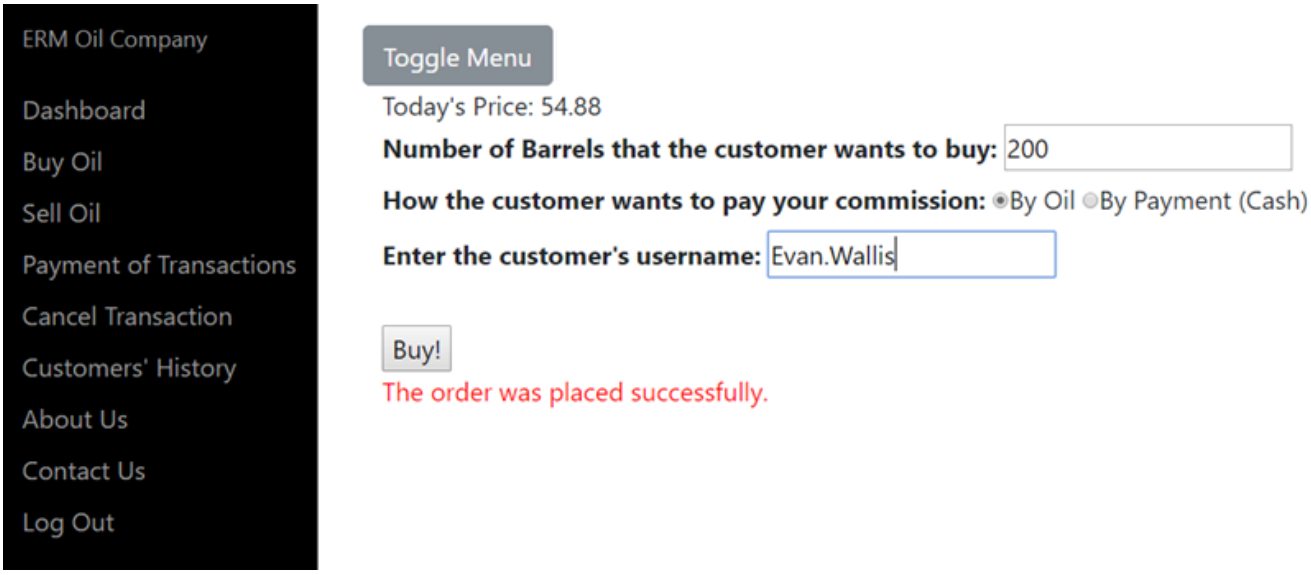

Figure 3: Page for trader buying oil behalf of customers.

- Sell: Customers and Traders can navigate and sell oil through Sell page. Similarly to Buy page, the input data for Customers are the number of barrels and the commission payment type, which may vary (Oil or Cash). Besides number of barrels and commission payment type, Traders should input the username who wants to perform the bought. Note that, if a customer tries to sell a has not enough oil stored by the company to satisfy the request, the transaction will not proceed.

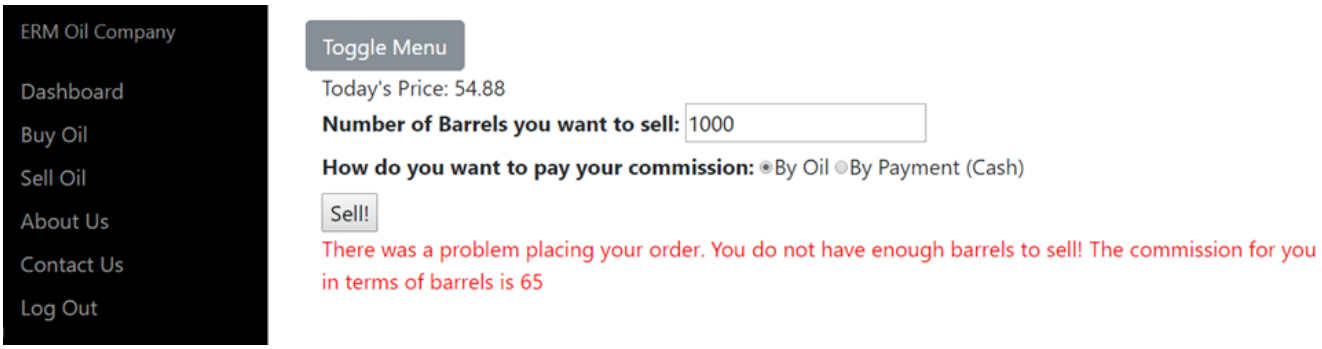

Figure 4: Page for customers selling oil.

- Payment: This page may be accessed only by Traders, who can perform payments behalf of customers. The inputs for this page are the username of the customer who wants to perform the payment and the amount to be payed.

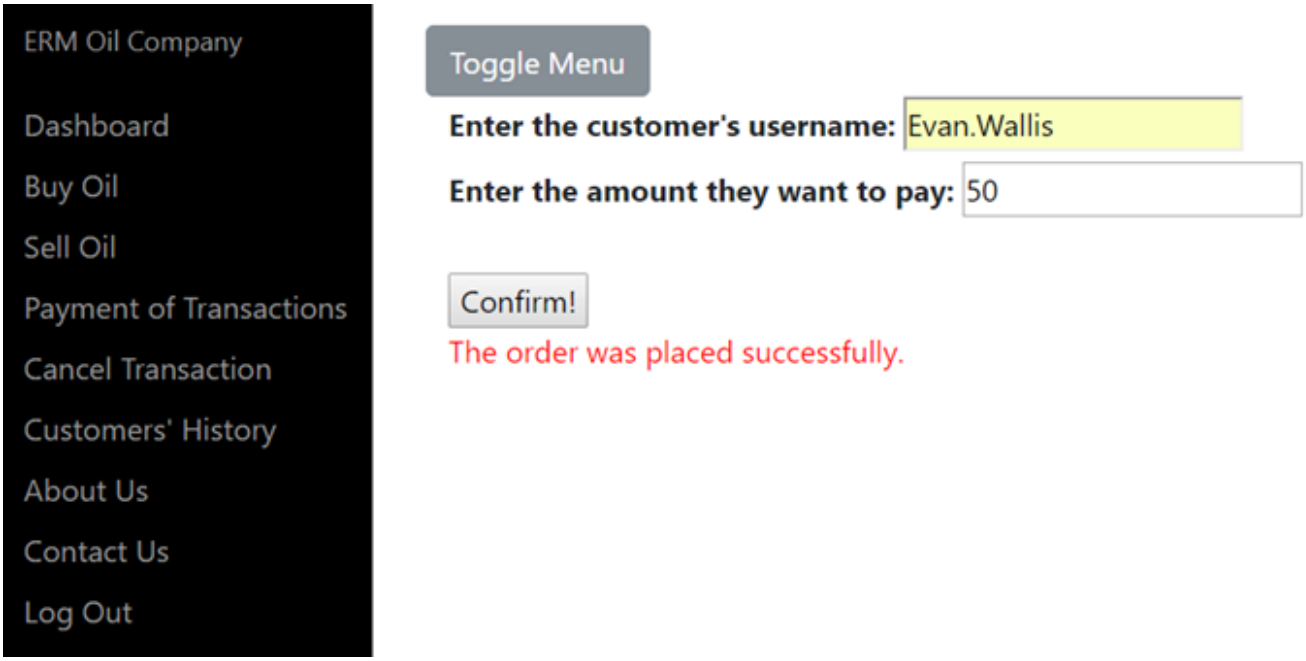

Figure 5: Page for traders performing payments behalf of customers.

- Cancel: Like Payment page, this may be accessed only by Traders. Through this page, they cancel transactions performed by customers. The only input for this page is the transaction Id to be canceled.

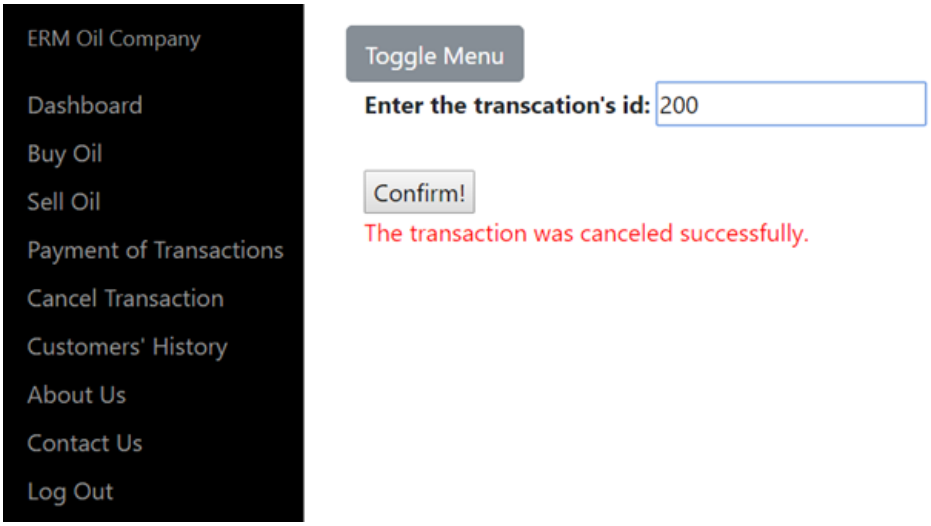

Figure 6: Page for traders canceling transactions.

- Customers' History: This is a inquiry page where traders can input the customers' zip code or username the perform the search. The output shows the customers found for the given input and their respective transactions history.

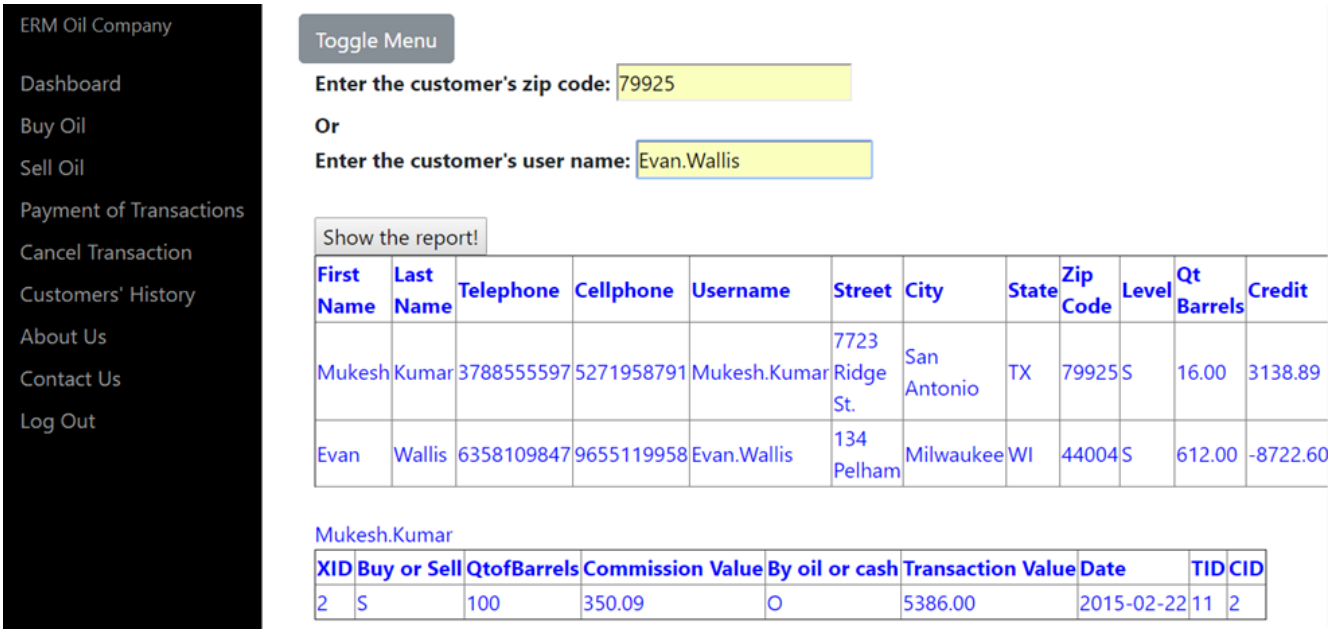

Figure 7: Page for traders inquiry customers historic of transactions.

- Managers' Report: Only managers can navigate through this page to run daily, weekly or monthly reports from a starting date given as input.

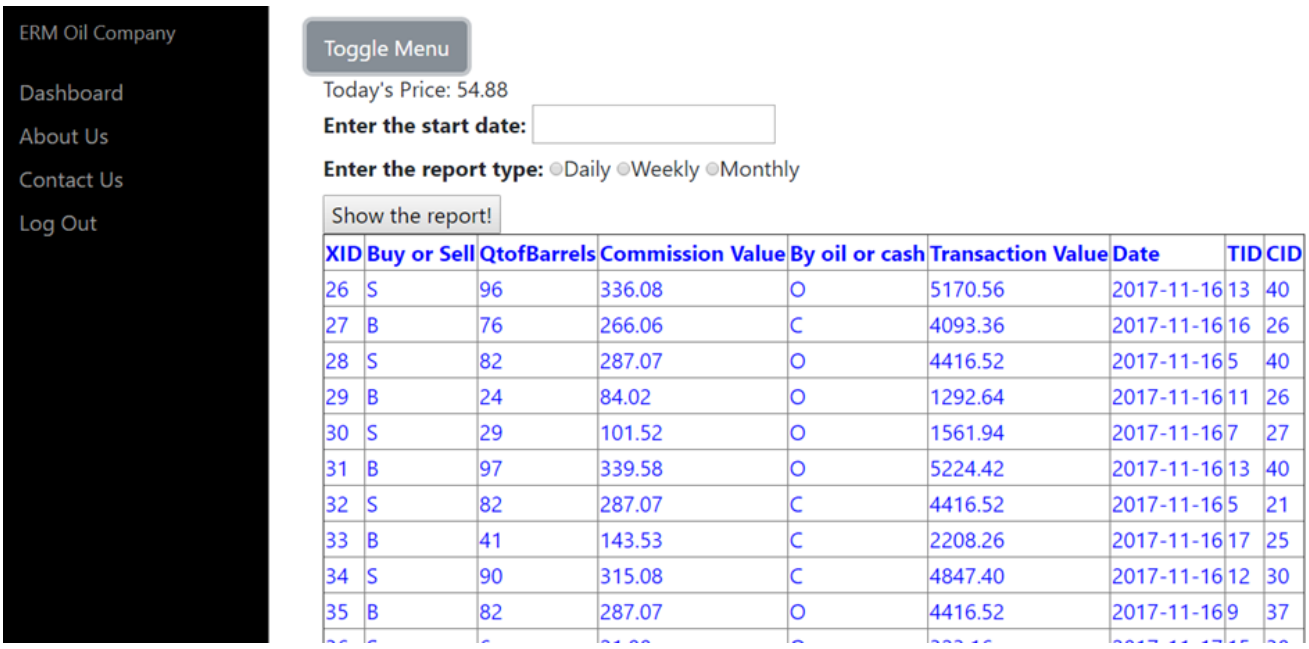

Figure 8: Page for managers run daily, weekly or monthly reports.

3 Overview of the Code

In this section we briefly present the source files created for the OilComp system. The web pages were developed using PHP and SQL programming was implemented following the MySQL

standard. Note that we adopted prepared statements with bind variables using PDO in PHP

Please find below the names of the source files and the description of the contents of each one of them.

- **CreateOilCompany-MySQL.txt**: SQL statementsqueries used to build the oilcomp schema and populate the databases;
- **index.php**: PHP code for login page shown in Figure 1;
- **tradersBuyTransaction.php, costumersBuyTransaction.php**: PHP codes for buying transactions for Traders and Customers respectively;
- **tradersSellTransaction.php, costumersSellTransaction.php**: PHP codes for selling transactions for Traders and Customers respectively;
- **tradersPayment.php, tradersPaymentTransaction.php**: PHP codes for traders performing payments, like shown in Figure 5;
- **tradersCancel.php, tradersCancelTransaction.php**: PHP codes for traders canceling transactions, like shown in Figure 6;
- **tradersSearchHistory.php, tradersSearchHistoryReport.php**: PHP codes for traders inquiring customers's history of transactions, like shown in Figure 7;
- **managersDashboard.php**: PHP code for managers to run monthly, weekly or daily reports, like shown in Figure 8;

A ER Diagram

Please find below the Entity Relationship Diagram implemented in OilComp system.

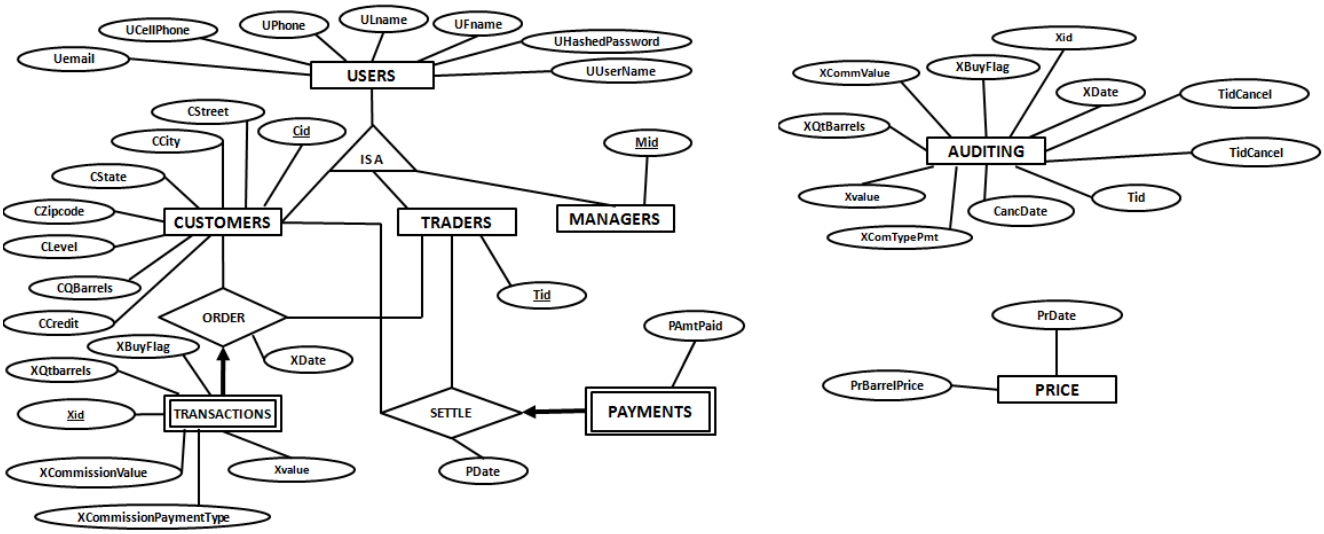

Figure 9: Entity Relationship Diagram.