Oil Trading System

Group Project

Group Members

1.Juhi Barai jkb140130
2.Mathumitha Raja mxr142030
3.Sushmitha MohanRaj sxm144630

1.PURPOSE :

To create a convenient and easy-to-use Oil Trading software for oil traders who are trying to buy and sell oil for their clients and managing oil transactions issued by clients.

1.1 SCOPE:

1. Allowing clients to login in the system and perform a transaction with the freedom to make payments later.
2. The software gives an added advantage by allowing a trader to perform a transaction on behalf of a client.
3. The client is provided with multiple options for commission as well as transaction payment.
4. The trader has the authority to cancel a mode of payment selected by client.

1.2 SIZE AND PERFORMANCE :

1. Payment should be authorized by a trader in not more than 30 seconds latency.
2. Credit card payment should finish in 10 seconds.
3. 80% of the transaction should be completed in within a minute.
4. Search queries should return 90% of time in less than 5 seconds.
5. The end user computer should have minimum 4GB RAM.

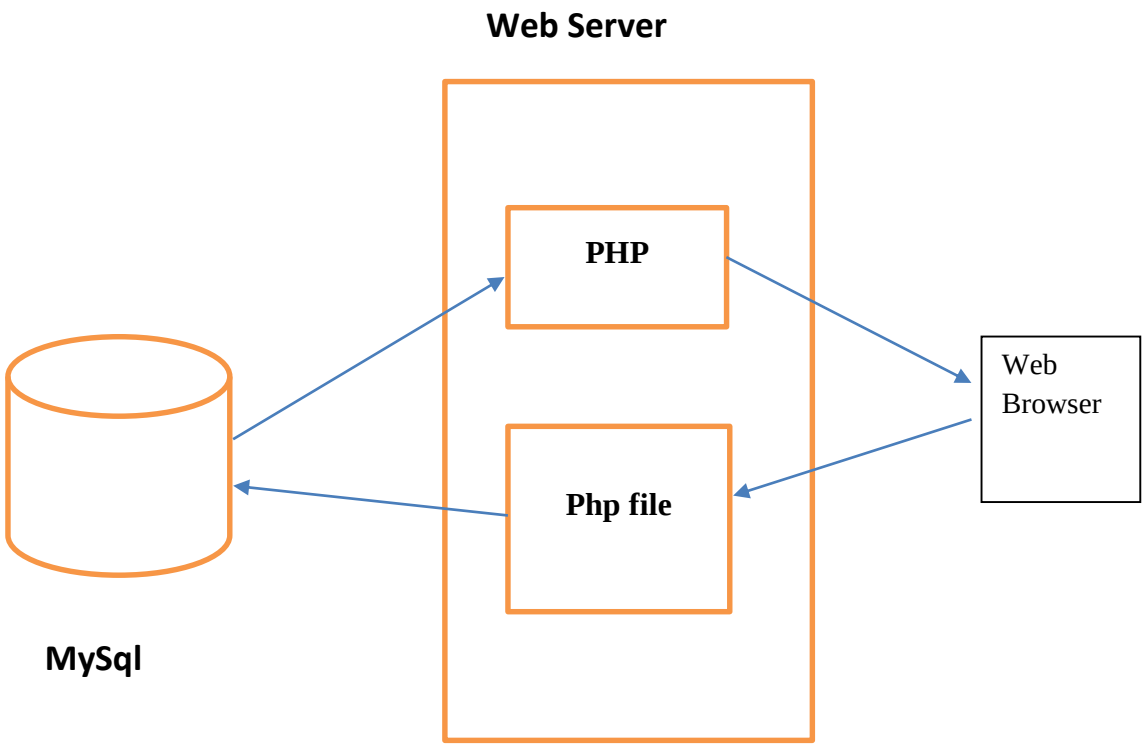
1.3 QUALITY :

1. The desktop user interface should be Windows 7/8/8.1 compliant.
2. The oil trading system shall be available 24 hours a day and 7 days a week, with a 5% down time.
3. The UI of the application is designed for ease-to-use by a computer literate or non-literate user group. No additional training on the system application required.

1.4 SECURITY:

The system should be secure, so that a customer can perform online buy/sell transactions and make online payments. Authentication is employed in the oil trading system using username and password.

Basic Architecture Diagram



2.Relational Schema (SQL Create Table Statements)

1.Client:

```
CREATE TABLE `client` (
  `cid` varchar(5) NOT NULL,
  `cfname` varchar(45) NOT NULL,
  `clname` varchar(45) DEFAULT NULL,
  `phone_no` int(11) NOT NULL,
  `mobile_no` int(11) NOT NULL,
  `email` varchar(45) NOT NULL,
  `level` varchar(45) NOT NULL,
  `quantity` decimal(45,5) NOT NULL,
  `amount_due` decimal(45,5) DEFAULT NULL,
  `status` varchar(45) DEFAULT NULL,
  PRIMARY KEY (`cid`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

2.Address:

```
CREATE TABLE `address` (
  `zip_code` int(11) NOT NULL,
  `state` varchar(45) NOT NULL,
  `cid` varchar(5) NOT NULL,
  `city` varchar(25) NOT NULL,
 KEY `cid_idx` (`cid`),
  CONSTRAINT `cid` FOREIGN KEY (`cid`) REFERENCES `client` (`cid`) ON DELETE
NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

3.Login:

```
CREATE TABLE `login` (
  `username` varchar(10) NOT NULL,
  `password` varchar(45) NOT NULL,
  `role` varchar(45) NOT NULL,
  `id` varchar(45) NOT NULL,
  PRIMARY KEY (`username`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

4.Oil:

```
CREATE TABLE `oil` (
  `oil_type` varchar(45) NOT NULL,
  `oil_quantity` decimal(10,5) NOT NULL,
  `cost_barrel` int(11) NOT NULL,
  PRIMARY KEY (`oil_type`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

5.Payment:

```
CREATE TABLE `payment` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `t_id` varchar(45) DEFAULT NULL,
  `c_id` varchar(45) NOT NULL,
  `amount` int(11) NOT NULL,
  `payment_status` varchar(45) NOT NULL,
  `payment_type` varchar(45) DEFAULT NULL,
  `date_of_payment` varchar(45) DEFAULT NULL,
```

```
  PRIMARY KEY (`id`),

  KEY `c_id_idx` (`c_id`),

  KEY `t_id_idx` (`t_id`),

  CONSTRAINT `c_id` FOREIGN KEY (`c_id`) REFERENCES `client` (`cid`) ON
DELETE NO ACTION ON UPDATE NO ACTION,

  CONSTRAINT `t_id` FOREIGN KEY (`t_id`) REFERENCES `trader` (`trader_id`) ON
DELETE NO ACTION ON UPDATE NO ACTION

) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=latin1;
```

6.Transaction:

```
CREATE TABLE `transaction` (

  `transaction_id` int(11) NOT NULL AUTO_INCREMENT,

  `client_id` varchar(10) NOT NULL,

  `trader_id` varchar(10) DEFAULT NULL,

  `transaction_status` varchar(45) NOT NULL,

  `oil_quantity` int(11) NOT NULL,

  `commission_type` varchar(45) DEFAULT NULL,

  `date_initiated` date NOT NULL,

  `date_approved` date DEFAULT NULL,

  `transaction_type` varchar(45) NOT NULL,

  PRIMARY KEY (`transaction_id`),

  KEY `client_id_idx` (`client_id`),

  KEY `trader_id_idx` (`trader_id`),

  CONSTRAINT `client_id` FOREIGN KEY (`client_id`) REFERENCES `client` (`cid`) ON
DELETE CASCADE ON UPDATE CASCADE,

  CONSTRAINT `trader_id` FOREIGN KEY (`trader_id`) REFERENCES `trader`
(`trader_id`) ON DELETE NO ACTION ON UPDATE NO ACTION
```

```
) ENGINE=InnoDB AUTO_INCREMENT=100 DEFAULT CHARSET=latin1;
```

7.Trader:

```
SELECT * FROM TRANSACTION;CREATE TABLE `trader` (
  `trader_id` varchar(10) NOT NULL,
  `trader_name` varchar(45) DEFAULT NULL,
  PRIMARY KEY (`trader_id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

3.Drop Table Statements

1. DROP TABLE `client`
2. DROP TABLE `transaction`
3. DROP TABLE `trader`
4. DROP TABLE `address`
5. DROP TABLE `oil`
6. DROP TABLE `login`
7. DROP TABLE `payment`

5.Code Overview

Below is a list of the main files in the program logic.

1. Login.php – Allows the Clients, Traders and Manger to login into the application, with their respective username and password by getting the user input from login.html.

2. Client.php – The client is offered 3 options namely Buy oil, Sell oil and Make Payment.
    - If Client selects Buy oil, he/she is prompted to enter the quantity and commission type. In client_trade.html the client enters the desired values.
    - If Client selects Sell oil, he/she is prompted to enter the quantity via client_sell.html page.
    - If Client selects Make Payment, he/she is prompted to enter amount and payment-type, which can be cash/cheque/credit card/debit card through client_payment.html page.

3. Trader.php – Trader is provided with 3 options, namely,
    - Approve/Cancel transaction - which lists the queue of pending transactions waiting for approval. The logic is handled in tarderView.php.
    - Approve/Cancel payment – Trader is prompted to enter the paymentID. Logic is handled in paymentView.php.
    - Initiate Buy – Trader trades on the behalf of a client by entering clientID, oil quantity and commission type in trader_initiate.html page.

4. Manager.php – Manager is offered with 2 important options,
    - View transaction history(oil buy/sell).
    - Change total quantity of oil possessed by the system.

6.SQL INJECTION

Attackers trick the SQL engine into executing unintended commands by supplying specially crafted string input, thus gaining unauthorized access to a database to view or manipulate restricted data.

A few common code vulnerabilities –

1. User supplied table name, column-comparison values can be junk-
   Escapes special characters in the unescaped_string, taking into account the current character set of connection so that it is safe to place it in mysql query.

   Use mysql_real_escape_string() function to prevent SQL injection.

   Example of SQL Injection attack-
   SELECT email, passwd, login_id, full_name
   FROM table
   WHERE email = 'x' AND 1=(SELECT COUNT(*) FROM tabname); --';

2. Cross-site scripting-
   Enables attacker to inject client-side script in web pages which are viewed by others, thus allowing attackers to bypass access control.
   The translations performed are:

   '&' (ampersand) becomes '&'
   '"' (double quote) becomes '"' when ENT_NOQUOTES is not set.
   ''' (single quote) becomes ''' (or ') only when ENT_QUOTES is set.
   '<' (less than) becomes '<'
   '>' (greater than) becomes '>'

   Example of SQL Injection attack-
   <?php
   $new = htmlspecialchars("Test", ENT_QUOTES);
   echo $new; // Test
   ?>