**.NET Project Report**

On

“***Reconciliation***”

M.Sc. (Computer Science) 2017-18

**Team Member:**

**Mahesh Ravindra Bhelke (M.Sc. (C.S.)-I)**

**TABLE OF CONTENTS**

**CHAPTER 1: INTRODUCTION**

1.1 Existing System and Need for System

1.2 Scope of Work

1.3 Operating Environment – Hardware and Software

**CHAPTER 2: PROPOSED SYSTEM**

2.1 Proposed System

2.2 Objectives of System

2.3 User Requirements

**CHAPTER 3: ANALYSIS & DESIGN**

3.1 Data Flow Diagram (DFD)

3.2 Functional Decomposition Diagram (FDD)

3.3 UML Diagrams

3.3.1 Class Diagram

3.3.2 Object Diagram

3.3.3 Use Case Diagram

3.3.4 Sequence Diagram

3.3.5 Collaboration Diagram

3.3.6 State Diagram

3.3.7 Activity Diagram

3.3.8 Component Diagram

3.3.9 Deployment Diagram

3.4 Entity Relationship Diagram (ERD)

3.5 Data Dictionary

3.6 Table Design

**CHAPTER 4: USER MANUAL**

4.1 Operations Manual / Menu Explanation

4.2 Menu Screens

4.3 Project code

**CHAPTER 5: Limitations and Enhancement**

5.1 Drawbacks and Limitations

5.2 Proposed Enhancements

5.3 Conclusions

5.4 Bibliography

**Introduction**

Reconciliation is the process used to resolve the difference between the distributors claimed amount and the bill back amount calculated by the system. When the bill back document is created, the original claim amount is copied from the claim request. The bill back amount calculated by the system can be same or different than the original claim amount. Further, if the calculated amount is different than the claim amount and if the system has a tolerance limit set then the claim amount can be within or outside tolerance of the calculated amount. The tolerance limit can be set up as per the manufacturer's preferences.

After the bill back, amount calculations are carried out in the manufacturer's system, the status of the line items are depicted using colors showing whether the claim amount is within tolerance, outside tolerance or same as the calculated amount. Reconciliation is usually carried out for the line items that are outside tolerance so that both the manufacturer and distributor concur on the amount before accrual and settlement take place.

A procedure for confirming the reliability of a company's accounting records by regularly comparing [balances of transactions]. An account reconciliation may be prepared on a daily, monthly, or annual basis. After reconciliation, the final bill back amount agreed upon by both parties is accrued and settled. If there are any changes after settlement, then they are made effective as a Post Settlement Adjustment.

**Need of the system**

**Manual reconciliation: -**

You can think of the reconciliation and billing process in the major stages outlined below. While these stages represent a general sequence of events, performing reconciliation may require working through some stages concurrently. We'll provide more details about each of these major stages in subsequent sections.

Determine which line items require reconciliation. The first step is to decide which orders and line items require your attention for reconciliation. Generally, these will be line items for which your advertiser used a third-party service to track volume. However, there may be some DFP served line items that need adjustment as well. The question to ask yourself is which line items need volume or revenue adjustments.

Next step: -

Confirm the billing source for each line item. If your advertiser is using a Third-party service to track impressions or clicks for the purposes of reconciliation with DFP line items, you'll need to identify these line items and ensure the source is set to "Third party volume". Orders for line items that do not require your attention can be marked "Complete".

Next Step: -

Incorporate third-party numbers. For third-party line items, you'll need to

incorporate numbers as reported by third-party services. The optimal way to do so is to leverage the DFP API.

Next step: -

Adjust values and complete reconciliation. Modify delivery numbers as necessary and mark each order "Complete" as to indicate the line items that belong to it no longer need your attention.

Next step: -

Integrate reconciliation data with invoicing system. Run a report though the DFP O query tool to import it to your invoicing system, or leverage the DFP API to

transfer finalized reconciliation data into your invoicing system. The goal is to incorporate all revenue and reconciliation data to your invoicing system so that you can generate invoices and bill advertisers.

**MANUAL RECONCILIATION TO**

**AUTOMATION**

In the United States, the passage in 2002 of the Sarbanes-Oxley Act (SOX) has emphasized the need for balance sheet account reconciliation to be included within a company's own procedures, not relying only on external

auditors. The legislation was enacted to protect shareholders and general a public from accounting errors and fraudulent practices in the enterprise, as well as improve the accuracy of corporate disclosures." SOX and other acts like it across the world have increased stress on organizations to comply. As a result, the accounting industry has sought ways to automate a previously strenuous o manual process. The pressure of sox coupled with the perennial need to mitigate erroneous reconciliation in the process.

By using all the information technology available, organizations can easily automate their reconciliation and for each financial close cycle less manual labor would be required. 90% of companies manually reconcile using Microsoft Excel spreadsheets to do so. This process is arduous allowing for further human error. Automating reconciliation can significantly reduce errors and increase efficiency. Further benefits of automated reconciliation include centralized control, improved monitoring, reduced operational costs, increased productivity and efficiency, improved accessibility, data security improved and reduced audit risks and costs.

**Scope of work**

* Upload Transaction Data Files
* View Transaction Files
* Search Transaction Files
* Map two transaction Files
* Create Report
* View Report
* Database of Transaction Files
* Secured

**Operating System: - hardware & Software.**

* Minimum Software Requirement:

1. Windows 7 (service pack one).
2. Dot Net 4.6 Framework.
3. Web Browser (chrome).
4. Sql Server.

* Minimum hardware requirement.

1. Dual-core Processor (Intel).
2. 1 gb ram.

**Proposed System**

The proposed system has better both I/O capabilities of each user skills while interacting with the system. The retrieval of the records is much faster than the Existing System. Hence it causes to saving user time for the further work

The proposed system also provides the information about transaction id, date, customer, name etc. Mapping Algorithm is much faster and more effective it gives result is less time than pervious algorithm.

Searching feature is quite faster than the Existing System. Just by selecting the date the transaction file is pull and is displayed. Hence the proposed system is better than existing system.

**Advantages of Proposed system**

1. Enhanced User Interface.
2. Upload file to the database server.
3. View the Transaction File.
4. Mapping of two transaction file is faster.
5. Reports are generated faster.

**Objective of System**

* Speed
* Manual Reconciliation to Automation
* Increase in performance
* Reducing the process time

**User Requirement**

* Web Base Application for Reconciliation Process.
* Sql Database.
* Login Functionally.
* Upload Transaction File.
* View Transaction File.
* Map Transaction File.
* Report Generation.
* Fast and Effective.

**3.1) Data Flow Diagram (DFD)**

**Context Level DFD**

Gets data for processing

Request s data

Request current status

Gets Client’s info

Gets status information

Request Client’s info

Admin

Client

Reconciler

**1st Level DFD**

Reconciler

Active Directory

Update DB

Report Generation

2.0

1.0

3.0

4.0

5.0

View DB

Map DB

**UML DIAGRAM**

ViewTrans

- date: String

-changedDate: string

+Page\_load()

ViewPos

- date: String

-changedDate: string

+Page\_load()

MapResult

-changeddate: String

-settlement\_status: String

+Button1\_click()

Map

-selecteddate: string

-changedDate: string

+Calender\_Selection\_Changed()

View

- selecteddate: string

+CalenderSelectionChanged()

+ POS\_TransactionObj

Upload

- csvpath: string

-csvdata: string

+TransactionBOUpload()

+PosBOUpload()

+Upload\_to\_server()

Transaction

-transaction\_id: Int64

-total\_transaction\_amount: decimal

+Page\_load()

+Calender\_Selection\_Changed()

**Class diagram**

View

- selecteddate: 03/15/2018

Upload

- csvpath: ~Files/p03/19/2018.csv

-csvdata: “193380428516231,2001-02-16T 12:25:30

+Upload\_to\_server()

M:Map

-selecteddate: 03/15/2018

-changedDate: 03/19/2018

ViewTrans

- date: 03/15/2018

-changedDate: t03/19/2018

ViewPos

- date: 03/15/2018

-changedDate: t03/19/2018

MapResult

-changeddate :t03/19/2018

-changeddate: p03/19/2018

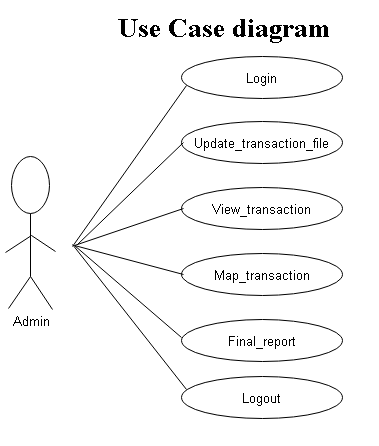
-settlement\_status: 0

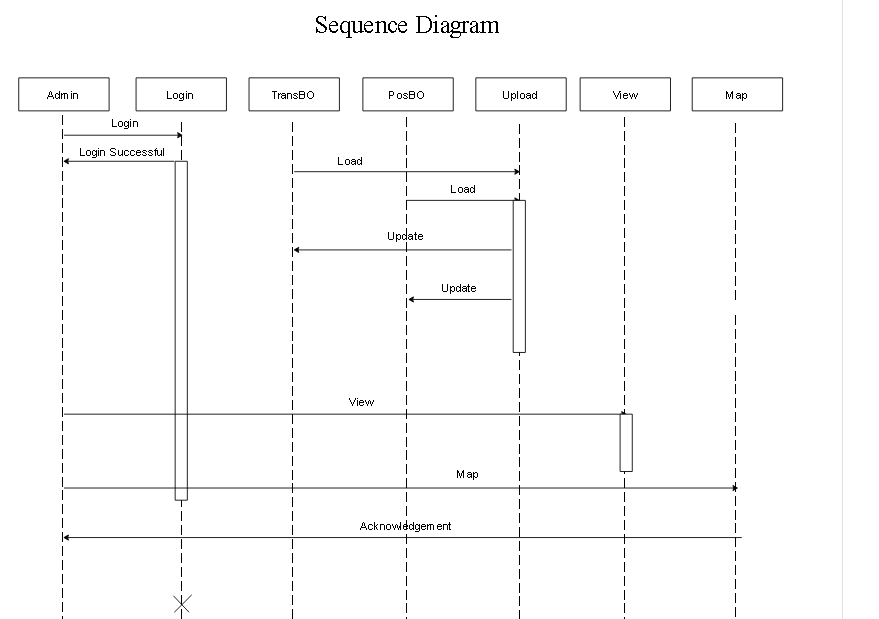
Transaction

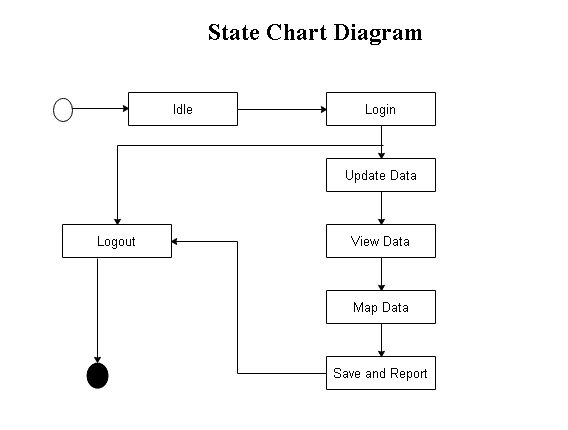
-transaction\_id: 150982332598982

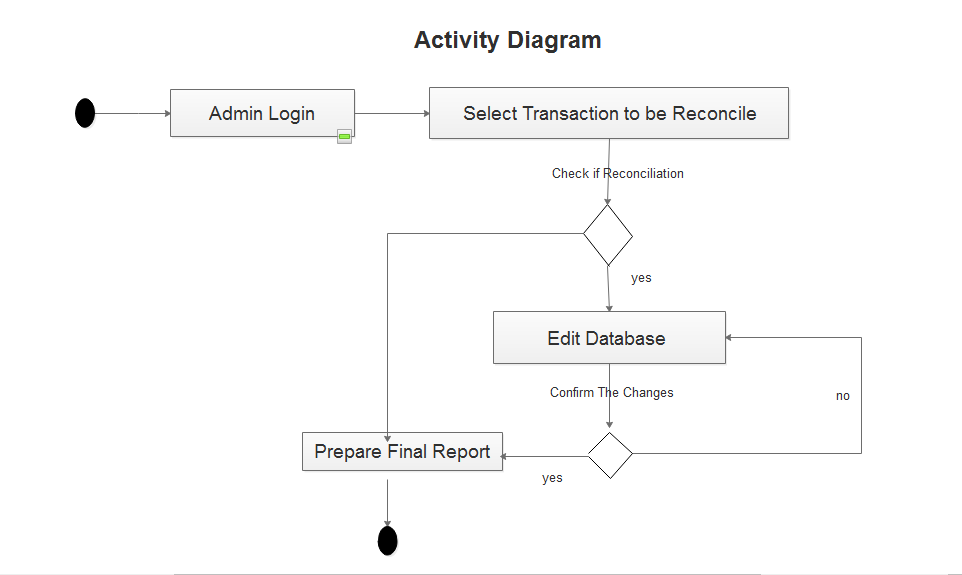
-total\_transaction\_amount: 5162.00

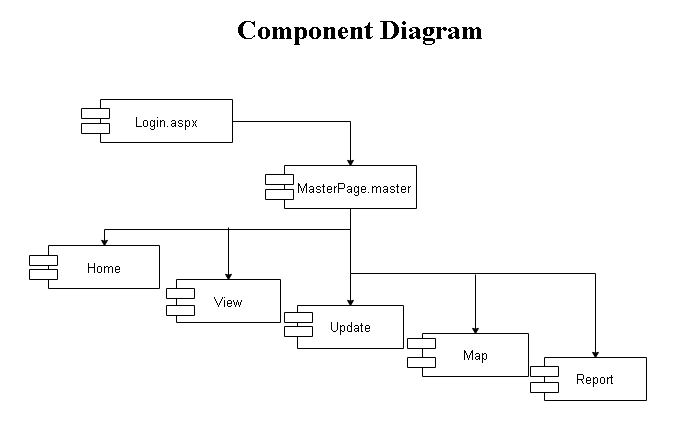
**Object diagram**











Data Dictionary

Database :- dbConnection

TableName:- Transaction\_BO\_Upload

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Constraints | Description |
| transaction\_id | Int64 | Primary Key |  |
| date\_time | DateTime |  |  |
| corporate\_id | Int64 |  |  |
| merchant\_id | Int64 |  |  |
| store\_id | Int64 |  |  |
| terminal\_id | Int64 |  |  |
| terminal\_type | string |  |  |
| terminal\_sequence\_no | Int64 |  |  |
| application\_version | Float |  |  |
| software\_version | Float |  |  |
| os\_version | Float |  |  |
| ip\_address | String |  |  |
| mac\_address | String |  |  |
| associate\_id | Int64 |  |  |
| associate\_name | String |  |  |
| customer\_name | String |  |  |
| customer\_card\_no | Int64 |  |  |
| card\_no | Int64 |  |  |
| card\_expiry\_date | String |  |  |
| billing\_address1 | String |  |  |
| billing\_address2 | String |  |  |
| billing\_city | String |  |  |
| billing\_state | String |  |  |
| billing\_country | String |  |  |
| billing\_zip | Int64 |  |  |
| contact\_no | Int64 |  |  |
| shipping\_address1 | String |  |  |
| shipping\_address2 | String |  |  |
| shipping\_city | String |  |  |
| shipping\_state | String |  |  |
| shipping\_country | String |  |  |
| shipping\_zip | Int64 |  |  |
| sku\_amount | Int64 |  |  |
| tax\_amount | Int64 |  |  |
| tip\_amount | Int64 |  |  |
| discount\_amount | Int64 |  |  |
| donation\_amount | Int64 |  |  |
| total\_transition\_amount | Int64 |  |  |
| approve\_amount | Int64 |  |  |
| balance\_amount | Int64 |  |  |
| processor\_merchant\_id | Int64 |  |  |
| processor\_terminal\_id | Int64 |  |  |
| approve\_code | Int64 |  |  |
| is\_online | String |  |  |
| processor\_transition\_date\_time | DateTime |  |  |
| pos\_reference\_no | Int64 |  |  |
| pos\_register\_no | Int64 |  |  |
| settlement\_status | Int64 |  |  |

Database :- dbConnection

TableName:- POS\_BO\_Upload

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Constraints | Description |
| transaction\_id | Int64 | Primary Key |  |
| pos\_transistion\_id | Int64 |  |  |
| pos\_register\_no | Int64 |  |  |
| merchant\_id | Int64 |  |  |
| store\_id | Int64 |  |  |
| terminal\_id | Int64 |  |  |
| terminal\_type | String |  |  |
| associate\_id | Int64 |  |  |
| associate\_name | String |  |  |
| customer\_card\_no | Int64 |  |  |
| card\_no | Int64 |  |  |
| total\_transistion\_amt | float |  |  |
| approved\_code | Int64 |  |  |

Database :- dbConnection

TableName:- Login

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Constraints | Description |
| **username** | **Varchar(50)** | Primary Key |  |
| **password** | **Varchar(50)** |  |  |

Database :- dbConnection

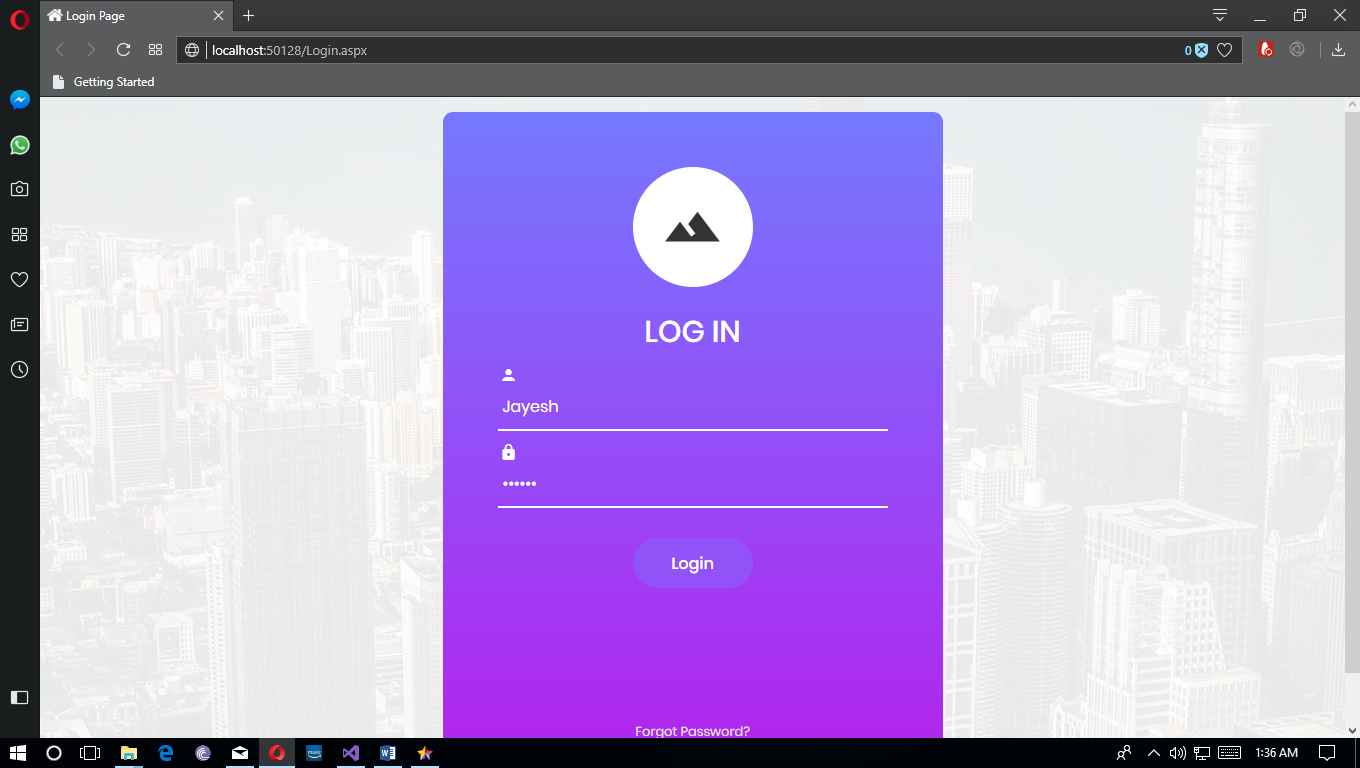
TableName:- Comment

|  |  |  |  |
| --- | --- | --- | --- |
| Column Name | Data Type | Constraints | Description |
| **Transaction\_id** | **bigint** | Primary Key |  |
| **Comment\_text** | **Varchar(50)** |  |  |
| **Date** | **datetime** |  |  |

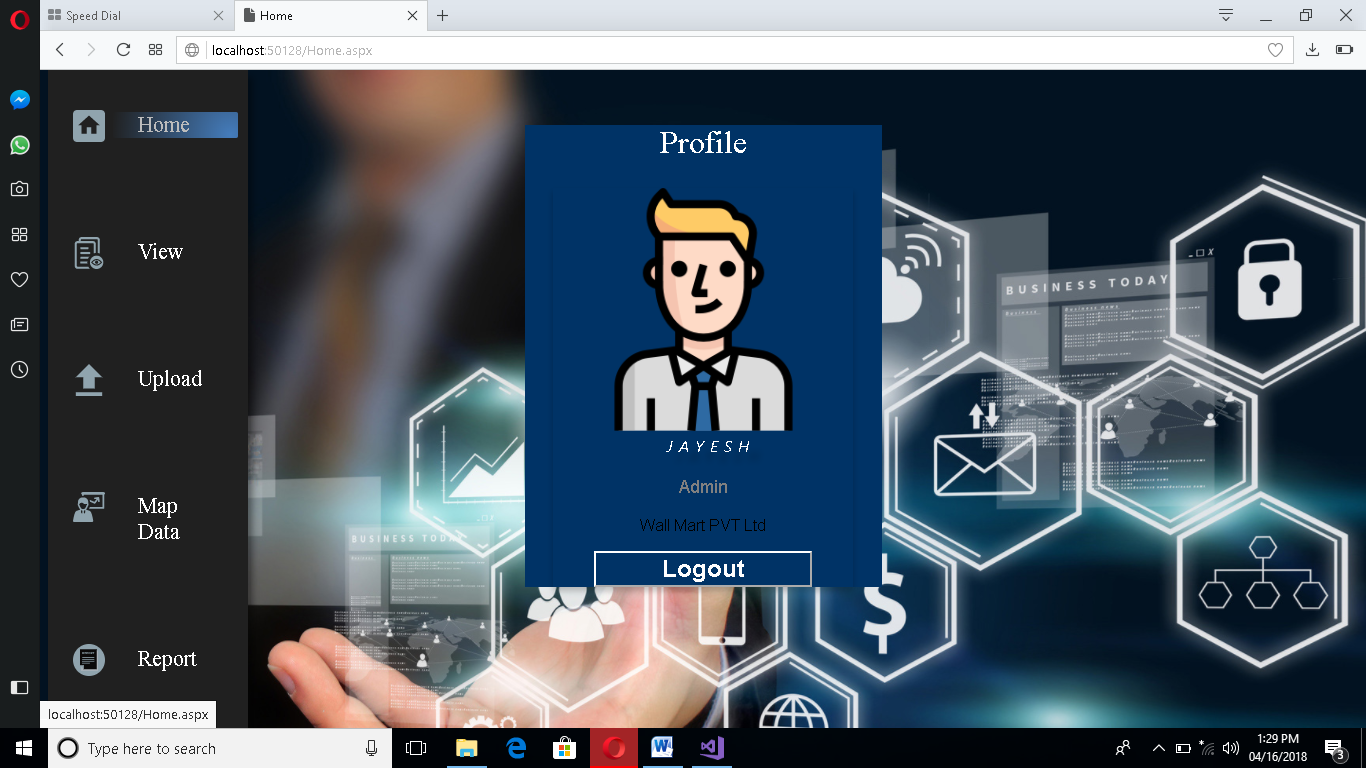
**CHAPTER 4: USER MANUAL**

**4.1 Menu Screens**

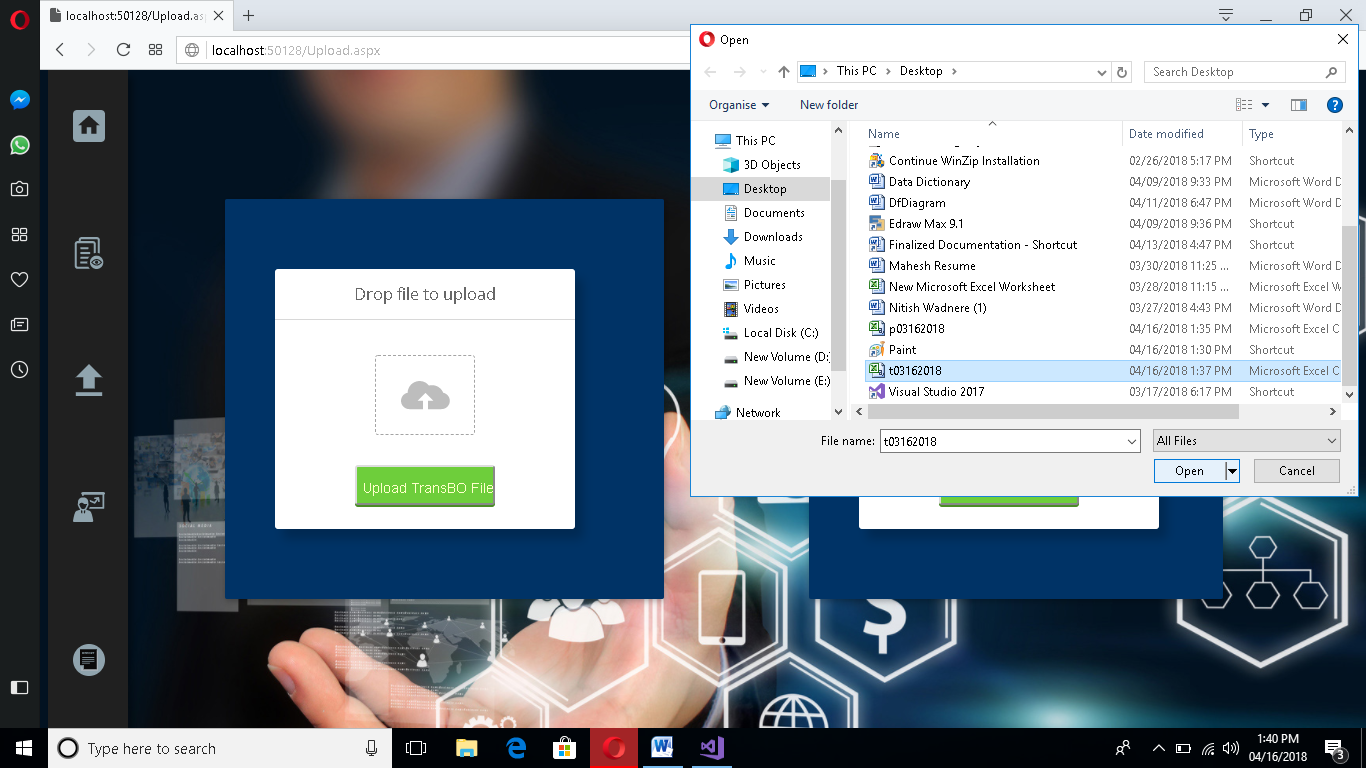
**Login Page:**

****

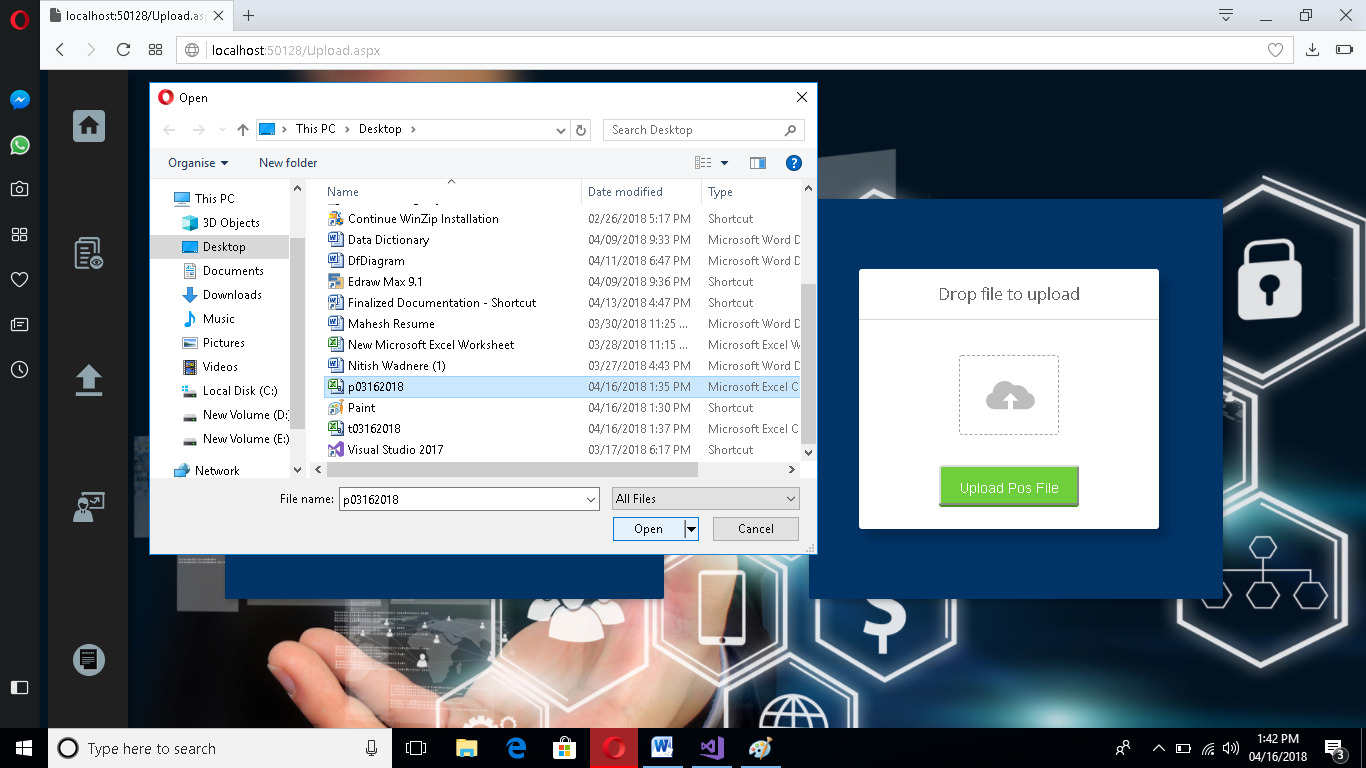
**Home Page:**

****

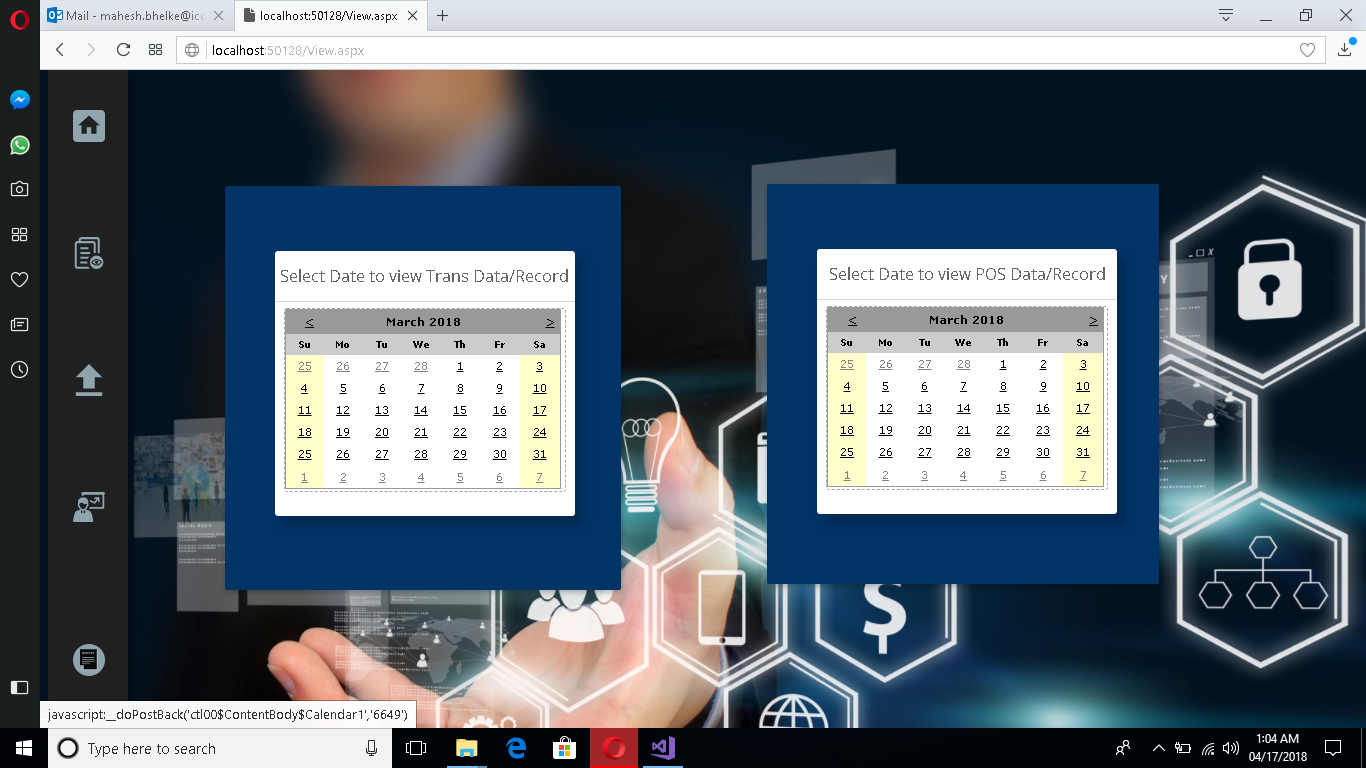
**Upload TransFile Page:**

****

**Upload PosFile Page:**

****

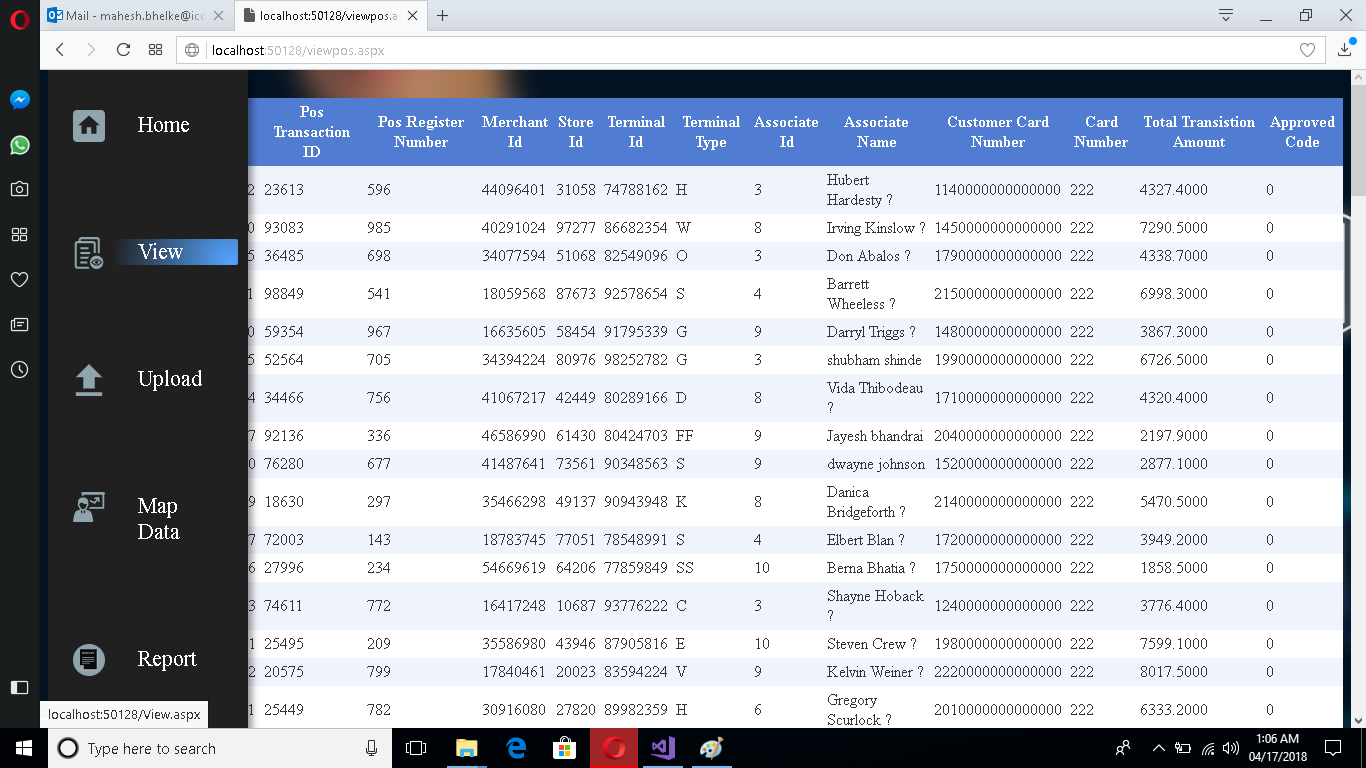
**View Page:**

****

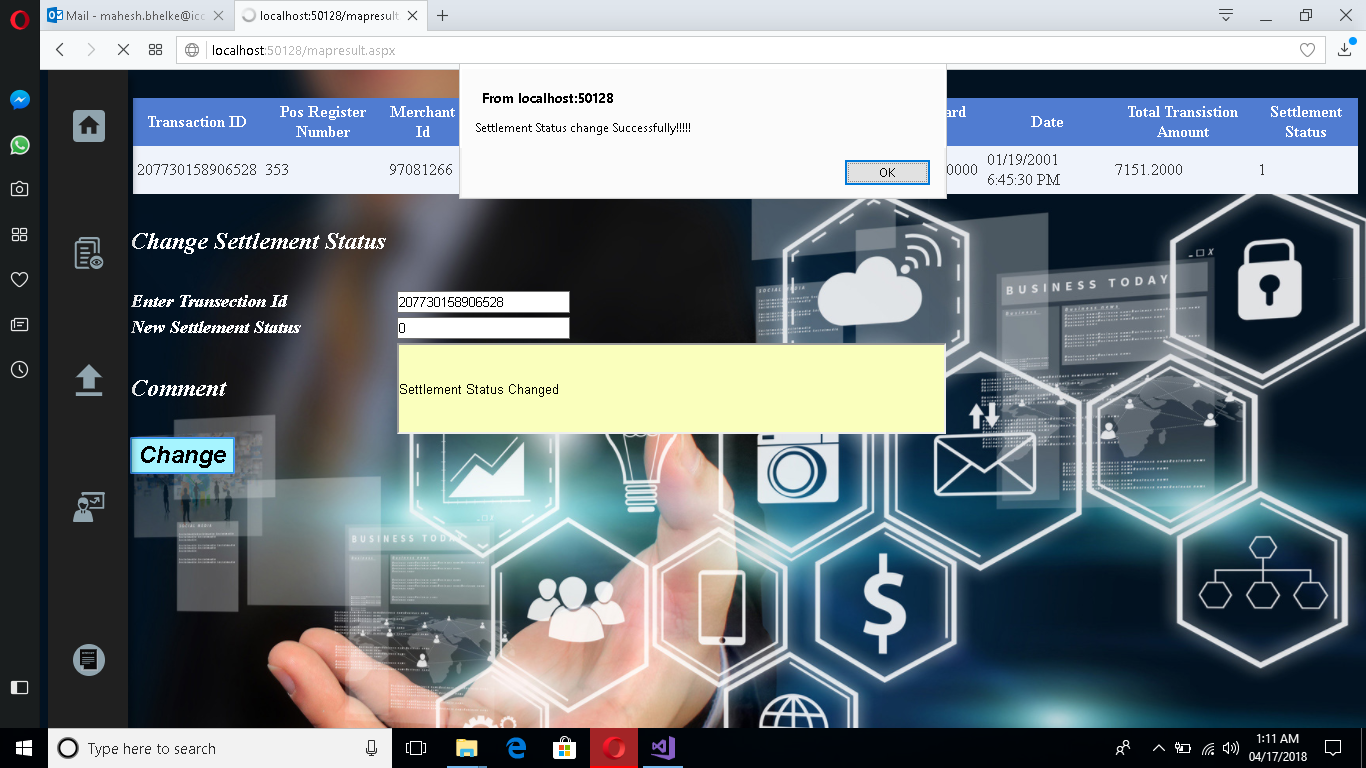
**ViewTransData Page:**

****

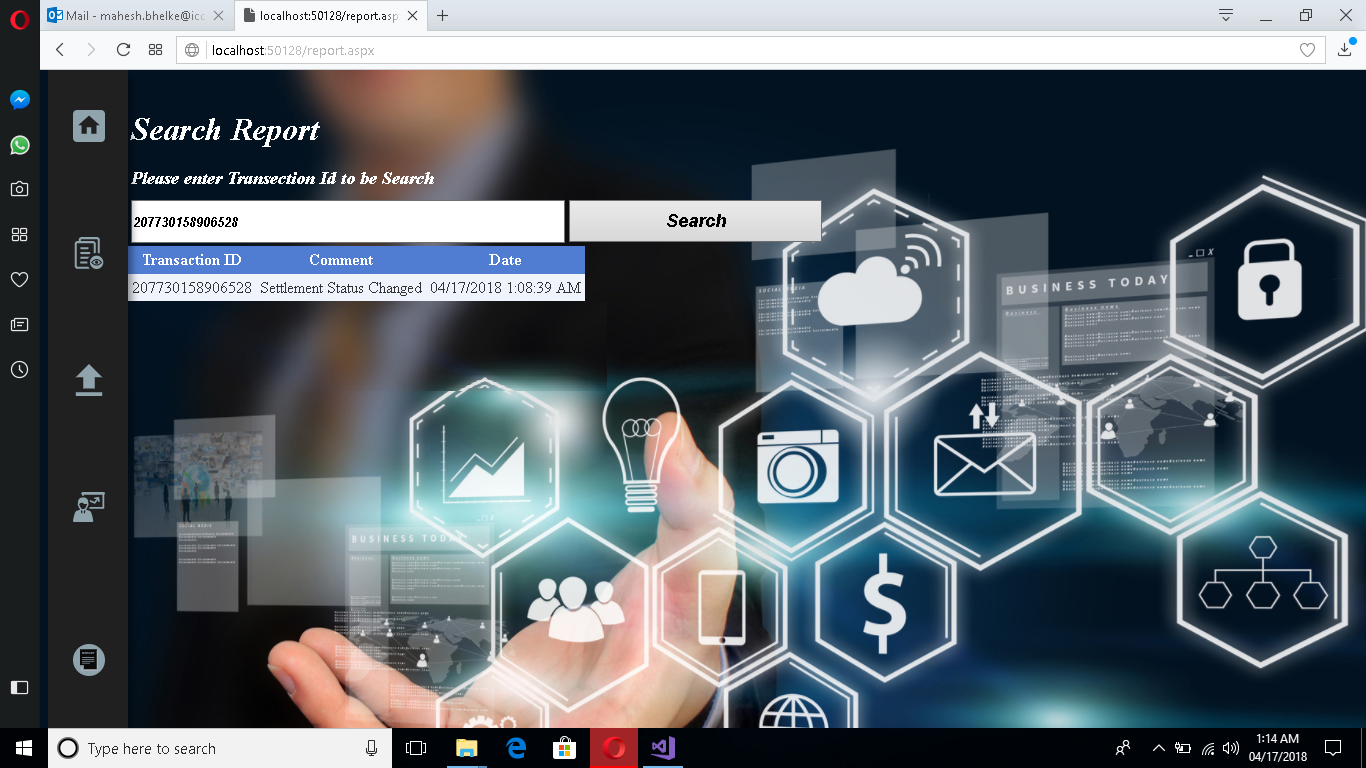
**ViewPosData Page:**

****

**Map Data Page:**

****

**Report Page:**

****

**3.7) Code Design**

**Transaction Class**

namespace Reconcilation

{

class Transaction

{

Int64 transaction\_id; //store transaction id

Decimal Total\_transaction\_amt; //store total transaction amt

public Transaction(Int64 trans\_id, Decimal amt)

{

transaction\_id = trans\_id;

Total\_transaction\_amt = amt;

}

public Transaction()

{

}

public void Set\_trans\_id(Int64 trans\_id)

{

transaction\_id = trans\_id;

}

public Int64 Get\_trans\_id()

{

return transaction\_id;

}

public void Set\_trans\_amt(Decimal amt)

{

Total\_transaction\_amt = amt;

}

public Decimal Get\_trans\_amt()

{

return Total\_transaction\_amt;

}

public override int GetHashCode()

{

int hash = 19;

hash = hash \* 31 + transaction\_id.GetHashCode();

hash = hash \* 31 + Total\_transaction\_amt.GetHashCode();

return hash;

}

public override bool Equals(object obj)

{

Transaction t = obj as Transaction;

if (t == null)

{

return false;

}

return

transaction\_id == t.transaction\_id &&

Total\_transaction\_amt == t.Total\_transaction\_amt;

}

}

**Class Map**

public partial class mapresult : System.Web.UI.Page

{

private string changeddatet;

private string changeddatep;

int counter = 0;

SqlConnection con = new SqlConnection(ConfigurationManager.ConnectionStrings["dbconnection"].ConnectionString);

protected void Page\_Load(object sender, EventArgs e)

{

try

{

if (Request.Cookies["changeddate"] != null)

{

changeddatet = "t" + Request.Cookies["changeddate"].Value.ToString();

changeddatep = "p" + Request.Cookies["changeddate"].Value.ToString();

}

int n = 0, b = 0;

con.Open();

string query = "SELECT COUNT(\*) FROM " + changeddatet + ";"; //query to count number of row

SqlCommand cmd1 = new SqlCommand(query, con);

int numberofrow = (int)cmd1.ExecuteScalar();

Transaction[] transobj = new Transaction[numberofrow]; //makeing array of object of size number of rows(tranaction)

Transaction[] posobj = new Transaction[numberofrow]; //makeing array of object of size number of rows(pos)

//fetching data from transaction table

string fetchquery1 = "select transaction\_id,total\_transition\_amount from " + changeddatet + ";";

SqlCommand cmd2 = new SqlCommand(fetchquery1, con);

SqlDataReader rd = cmd2.ExecuteReader(CommandBehavior.CloseConnection);

while (rd.Read())

{

transobj[n] = new Transaction();

transobj[n].Set\_trans\_id(rd.GetInt64(0));

transobj[n].Set\_trans\_amt(rd.GetDecimal(1));

n++;

}

//fetching data from pos table

string fetchquery2 = "select transaction\_id,total\_transistion\_amt from " + changeddatep + ";";

SqlCommand cmd3 = new SqlCommand(fetchquery2, con);

SqlDataReader rd1 = cmd3.ExecuteReader(CommandBehavior.CloseConnection);

while (rd1.Read())

{

posobj[b] = new Transaction();

posobj[b].Set\_trans\_id(rd1.GetInt64(0));

posobj[b].Set\_trans\_amt(rd1.GetDecimal(1));

b++;

}

// storeing problem matic transection id

Int64[] problem\_trans\_id = new Int64[numberofrow];

for (int j = 0; j < numberofrow; j++)

{

bool areEqual = transobj[j].Equals(posobj[j]);

if (areEqual)

{

}

else

{

problem\_trans\_id[counter] = transobj[j].Get\_trans\_id();

counter++;

}

}

for (int m = 0; m < counter; m++)

{

string query1= "select \* from " + changeddatet + " where" + " transaction\_id=" + problem\_trans\_id[m] + ";";

SqlCommand cmd4 = new SqlCommand(query1, con);

SqlDataReader rd2 = cmd4.ExecuteReader(CommandBehavior.CloseConnection);

while (rd2.Read())

{

Int64 trans\_id = rd2.GetInt64(0);

Int64 pos\_reg\_no = rd2.GetInt64(46);

Int64 mer\_id = rd2.GetInt64(3);

Int64 st\_id = rd2.GetInt64(4);

Int64 ter\_id = rd2.GetInt64(5);

String ter\_type = rd2.GetString(6);

Int64 ass\_id = rd2.GetInt64(13);

String ass\_name = rd2.GetString(14);

Int64 cus\_card\_no = rd2.GetInt64(16);

DateTime Date = rd2.GetDateTime(1);

Decimal tot\_trans\_amt = rd2.GetDecimal(37);

byte set\_status = rd2.GetByte(47);

String insertquery = "Insert INTO " + "" + "temp(transaction\_id,pos\_register\_no,merchant\_id,store\_id,terminal\_id,terminal\_type,associate\_id,associate\_name," + "customer\_card\_no,date,total\_transistion\_amt,settlement\_status) values ('" + trans\_id + "','" + pos\_reg\_no + "','" + mer\_id + "','" + st\_id + "','" + ter\_id + "','" + ter\_type + "','" + ass\_id + "','" + ass\_name + "','" + cus\_card\_no + "','" + Date + "','" + tot\_trans\_amt + "','" + set\_status + "');";

SqlCommand cmd5 = new SqlCommand(insertquery, con);

cmd5.ExecuteNonQuery();

}

}

DataSet ds = new DataSet();

string tempquery = "select \* from temp;";

SqlDataAdapter da = new SqlDataAdapter(tempquery, con);

da.Fill(ds, "Record");

DataTable table = ds.Tables["Record"];

DataRow row = table.Rows[0];

GridView1.DataSource = ds;

GridView1.DataBind();

con.Close();

con.Open();

string clear\_temp\_query = "truncate table temp;";

SqlCommand clrcmd = new SqlCommand(clear\_temp\_query, con);

clrcmd.ExecuteNonQuery();

con.Close();

}

catch(SqlException)

{

Response.Write("<script>alert('Data Not Available for Mapping Please Upload Data!!!!!')</script>");

Server.Transfer("Upload.aspx", true);

}

}

protected void Button1\_Click(object sender, EventArgs e)

{

Int64 trans\_id = Int64.Parse(TextBox1.Text);

byte setl\_status = byte.Parse(TextBox2.Text);

string txtcomment = TextBox3.Text;

string date = DateTime.Now.ToString();

con.Open();

string updatestring = "UPDATE " + changeddatet + " SET settlement\_status ='" + setl\_status + "' WHERE transaction\_id = '" + trans\_id + "';";

string addcomment = "insert into comment(Transaction\_Id,Comment\_text,Date)values('" + trans\_id + "','" + txtcomment + "','" + date + "')" + ";";

SqlCommand updatecmd = new SqlCommand(updatestring, con);

updatecmd.ExecuteNonQuery();

SqlCommand updatecmt = new SqlCommand(addcomment, con);

updatecmt.ExecuteNonQuery();

con.Close();

Response.Write("<script>alert('Settlement Status change Successfully!!!!!')</script>");

Server.Transfer("map.aspx", true);

}

protected void TextBox2\_TextChanged(object sender, EventArgs e)

{

}

protected void TextBox1\_TextChanged1(object sender, EventArgs e)

{

}

}

}

**Drawbacks and Limitations**

* Only valid user can make use of this software
* User has to Manual Adjust the money matter.
* Records may get corrupted.
* If server stop Application stop

Athough the design and development is carried out in order to cover most of the functions of the system still there remains certain limitation in the system so far developed. The main drawback is that files are not automatically pulled from the server. This can be considered in further development of the project.

**PROPOSED ENHANCEMENT**

As per requirement the project can developed more. One main proposed enhancement file that are to be map can be pull from sources accordingly to date automatically by removing the option of upload file or by putting it as optional. View of data can be more optimized accordingly

**CONCLUSION**

Thus, this project will help make order entry easier, make customer happier and help build a good business. Our project is only a humble venture to satisfy the needs in an Institution. Several user friendly coding have also adopted. This package shall prove to be a powerful package in satisfying all the requirements of the organization.

The objective of software planning is to provide a frame work that enables the manger to make reasonable estimates made within a limited time frame at the beginning of the software project and should be updated regularly as the project progresses

**Bibliography**

During development of any project some good sources should be referred. During the development of our project we referred some useful books which good sources of knowledge are.

* C# COMPLETE REFERENCE

BY Herbert Scheldt

* Professional WCF Programming .NET Development with the Windows® Communication Foundation

BY Scott Klein

* Windows Communication Foundation Unleashed

By Craig McMurtry

* Beginning Web Development, Silverlight, and ASP.NET AJAX

By Laurence Moroney

Wrox Silverlight™ 1.0

By Devin Rader

* Silverlight™ 4 UNLEASHED

By Laurent Bugnion

* SQL: Practical Guide for Developers

By Michael J. Donahoo.