VISVESVARAYA TECHNOLOGICAL UNIVERSITY
"Jnana Sangama", Belagavi-590018, Karnataka

Report

On

DATABASE MANAGEMENT SYSTEM MINI PROJECT (17CSL58)

"SHRI-UPI"

Submitted By

USN	NAME
1BI17CS182	YEETESH PRANAY

For the academic year 2018-19

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
BANGALORE INSTITUTE OF TECHNOLOGY
K.R. Road, V.V. Puram, Bengaluru-560 004

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
"Jnana Sangama", Belagavi-590018, Karnataka

BANGALORE INSTITUTE OF TECHNOLOGY
K.R. Road, V.V. Puram, Bengaluru-560 004

Department of Computer Science & Engineering

Certificate

This is to certify that the implementation of **DBMS MINI PROJECT (17CSL58)** entitled " **SHRI-UPI "** has been successfully completed by **YEETESH PRANAY(1BI17CS182)** of V semester B.E. for the partial fulfillment of the requirements for the Bachelor's degree in **Computer Science & Engineering** of the **Visvesvaraya Technological University** during the academic year **2019-2020**.

Lab In charges:

Prof. Prathima M. G.	**Prof. Varshitha K. C.**	**Dr. ASHA T.**
Assistance Professor	Assistance Professor	Professor and Head
Dept of CS&E, BIT	Dept of CS&E, BIT	Dept of CS&E, BIT

Examiners: 1) 2)

ACKNOWLEDGEMENT

The knowledge & satisfaction that accompany the successful completion of any task would be incomplete without mention of people who made it possible, whose guidance and encouragement crowned my effort with success. I would like to thank all and acknowledge the help I have received to carry out this Mini Project.

I would like to convey my thanks to Head of Department **Dr. ASHA T.** for being kind enough to provide the necessary support to carry out the mini project.

I am most humbled to mention the enthusiastic influence provided by the lab in-charges **Prof. PRATHIMA M. G.** and **Prof. VARSHITHA K. C.** , on the project for their ideas, time to time suggestions for being a constant guide and co-operation showed during the venture and making this project a great success.

I would also take this opportunity to thank my friends and family for their constant support and help. I'm very much pleasured to express my sincere gratitude to the friendly co-operation showed by all the **staff members** of Computer Science Department, BIT.

Name: Yeetesh Pranay
USN: 1BI17CS182

Table of contents

CHAPTER-1

INTRODUCTION

CHAPTER-1

1.1 INTRODUCTION:

SHRI-UPI allows transactions to be digitally handled through out the process of crediting, debiting, checking balance, history and feedback. It also reduces the capital spent on going to the bank, waiting in long lines, paying extra money to transact with another bank account and so on.When a customer opens the website he/she has to provide his/her information, later user_id will be created and displayed. And then (s)he has to remember his/her user_id through out in order to transact money using one's account. Upi_id will be pre-known to the customer hence while transacting (s)he must provide upi_id and user_id. There are options to check history of the respective user (transactions) and to send feedback regarding the website. (S)He is also given an option to view balance in the corresponding account.

1.2 PROBLEM STATEMENT:

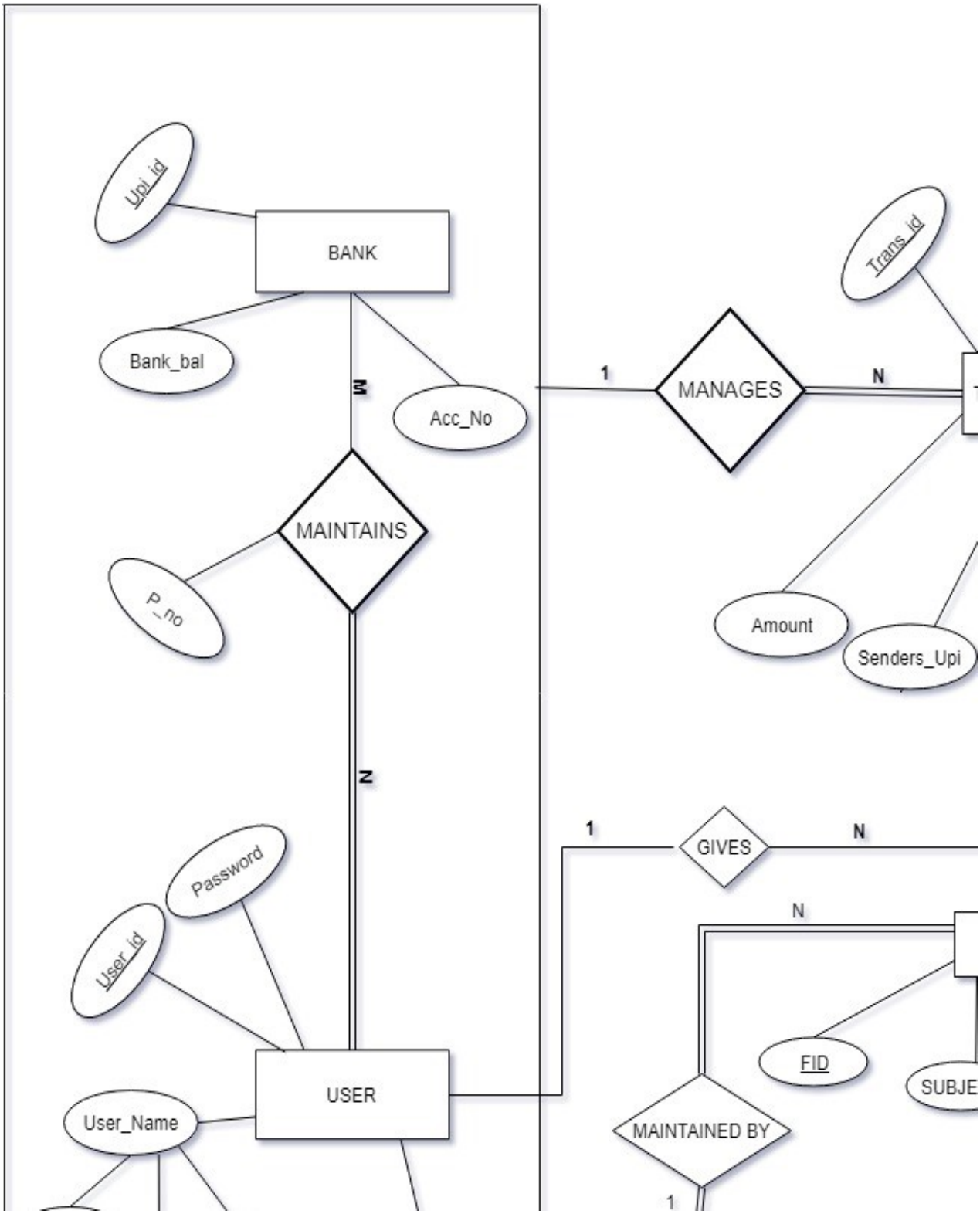
Design and develop a database to clog up the user's online banking and give them a hand way to transact money with appealing applications.

CHAPTER-2

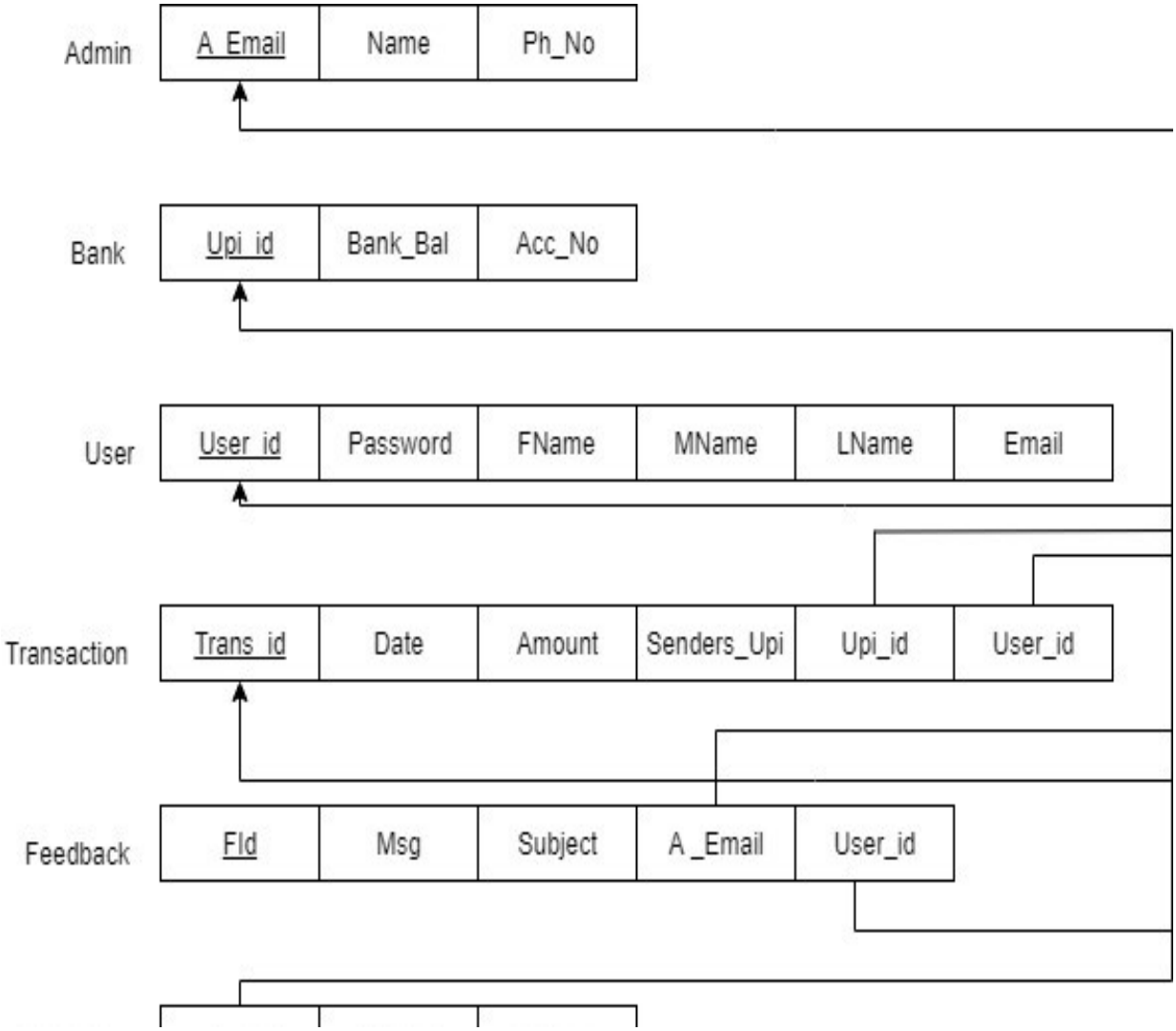
BACK END DESIGN

CHAPTER-2

2.1 CONCEPTUAL DATABASE DESIGN:



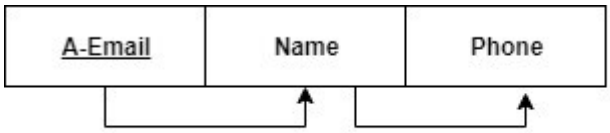
2.2 LOGICAL DATABASE DESIGN:



2.3 NORMALIZATION:

ADMIN:

FD1



First normal form: This table is already in 1NF as it all the attributes are atomic.

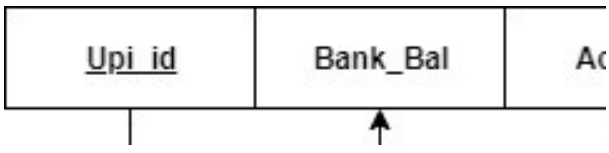
Second normal form: as there are no composite attributes where one of them derives one attribute and the other doesn't it is in 2NF

FD2 FD3

Third normal form: as there is a transitive property from A_Email to name and then Name to Phone in FD1 it is not in 3NF thus it is divided into FD2 and FD3.

BANK:

FD1



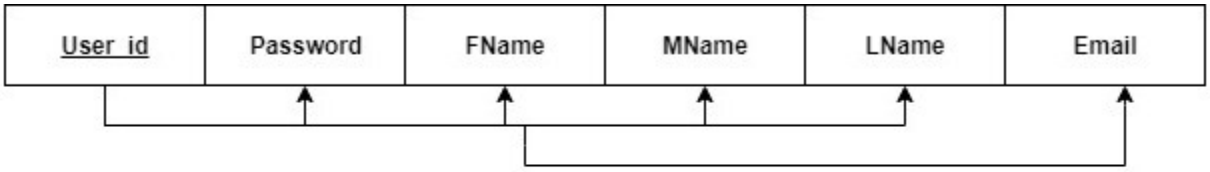
First normal form: This table is already in 1NF as it all the attributes are atomic.

Second normal form: as there are no composite attributes where one of them derives one attribute and the other doesn't it is in 2NF

Third normal form: The table doesn't include any transitive functional dependencies as well as it is in 2NF thus it is in 3NF.

USER:

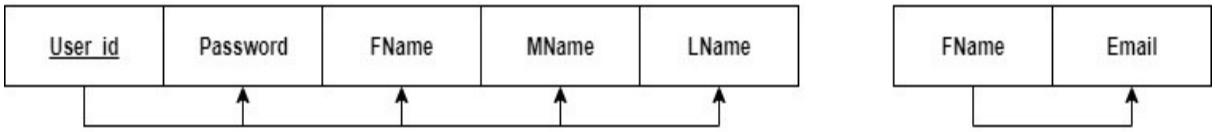
FD1



First normal form: This table is already in 1NF as it all the attributes are atomic.

Second normal form: as there are no composite attributes where one of them derives one attribute and the other doesn't it is in 2NF

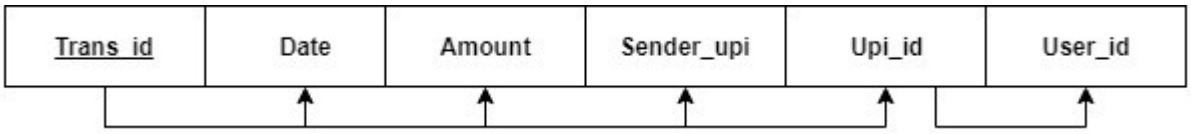
FD2 FD3



Third normal form: as there is a transitive property from User_id to Fname and then Fname to Email in FD1 it is not in 3NF thus it is divided into FD2 and FD3.

TRANSACTION:

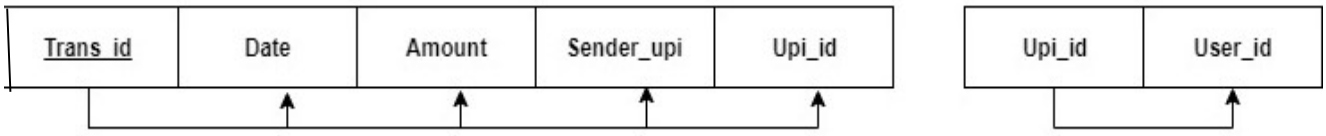
FD1



First normal form: This table is already in 1NF as it all the attributes are atomic.

Second normal form: as there are no composite attributes where one of them derives one attribute and the other doesn't it is in 2NF

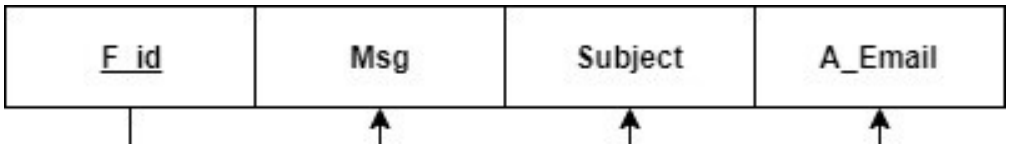
FD2 FD3



Third normal form: as there is a transitive property from Trans_id to Upi_id and then Upi_id to User_id in FD1 it is not in 3NF thus it is divided into FD2 and FD3.

FEEDBACK:

FD1



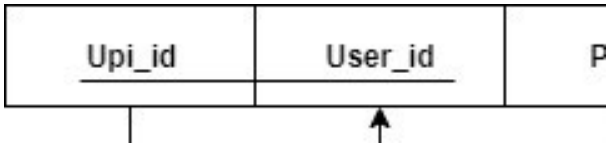
First normal form: This table is already in 1NF as it all the attributes are atomic.

Second normal form: as there are no composite attributes where one of them derives one attribute and the other doesn't it is in 2NF

Third normal form: The table doesn't include any transitive functional dependencies as well as it is in 2NF thus it is in 3NF.

MAINTAINS:

FD1



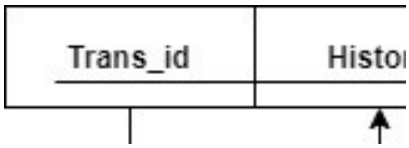
First normal form: This table is already in 1NF as it all the attributes are atomic.

Second normal form: as there are no composite attributes where one of them derives one attribute and the other doesn't it is in 2NF

Third normal form: The table doesn't include any transitive functional dependencies as well as it is in 2NF thus it is in 3NF.

HISTORY

FD1



First normal form: This table is already in 1NF as it all the attributes are atomic.

Second normal form: as there are no composite attributes where one of them derives one attribute and the other doesn't it is in 2NF

Third normal form: The table doesn't include any transitive functional dependencies as well as it is in 2NF thus it is in 3NF.

CHAPTER-3

FRONT END DESIGN

CHAPTER-3

3.1 SCREEN LAYOUT

HTML <p> TAG:

The HTML <p> element represents a paragraph of text. Paragraphs are usually represented in visual media as blocks of text that are separated from adjacent blocks by vertical blank space and/or first-line indentation. Paragraphs are block level elements.

HTML TAG:

The HTML element is the generic container for flow content and does not inherently represent anything. Use it to group elements for purposes such as styling (using the class or id attributes), marking a section of a document in a different language (using the lang attribute), and so on.

**HTML
 TAG:**

The HTML
 element produces a line break in text (carriage-return). It is useful for writing a poem or an address, where the division of lines is significant.

HTML <input> TAG:

The HTML <input> element is used to create interactive controls for web-based forms in order to accept data from the user.An <input> works varies considerably depending on the value of its type attribute,hence the different types are covered in their own separate reference pages. If this attributes is not specified, the default type adopted type is text.

<input> elements of type text create basic, single-line inputs. You should use them anywhere you want the user to enter a single-line value and there isn't a more specific input type available for collecting that value (for example, if it's a date, URL, email, or search term, you've got better options available).You can provide a useful placeholder inside your text input that can provide a hint as to what to enter by including using the placeholder attribute.The available types are as follows:

password: A single-line text field whose value is obscured. Use the maxlength and minlength attributes to specify the maximum length of the value that can be entered.

HTML <table> TAG:

The **HTML <table> element** represents tabular data — that is, information expressed via a two-dimensional data table.

The **HTML <tr> element** defines a row of cells in a table. Those can be a mix of <td> and <th> elements.

The **HTML <td> element** defines a cell of a table that contains data. It participates in the table model.

The **HTML <thead> element** defines a set of rows defining the head of the columns of the table.

The **HTML <tbody> element** groups one or more <tr> elements as the body of a <table>element.

HTML <form> TAG:

The **HTML <form> element** represents a document section that contains interactive controls to submit information to a web server.It is possible to use the :valid and :invalid CSS pseudo-classes to style a <form>element.The **HTTP method** that the browser uses to submit the form. Possible values are:

post: Corresponds to the HTTP POST method ; form data are included in the body of the form and sent to the server.

get: Corresponds to the HTTP GET method; form data are appended to the action attribute URI with a '?' as separator, and the resulting URI is sent to the server.Use this method when the form has no side-effects and contains only ASCII characters.This value can be overridden by a form method attribute on a <button> or <input>element.

action: The URI of a program that processes the form information. This value can be overridden by a form action attribute on a <button> or <input> element.

SCREEN LAYOUT FORMS

LOGIN FORM: The login form consists of **three text fields** and one **login button**. The text fields consist of Fname and Lname where user enters his/her FIRST NAME and LAST NAME which he has registered and **password** where the user enters the password given when he had registered. The **login button** posts the data to the servlet.

MAINTAINS FORM: Here one enters his/her user_id,uoi_id and phone number in order to combine user_id and upi_id using phone number. Itfinally contains a submit button.

TRANSACTION FORM: The transaction form maintains the primary details of transactions such as the sender's upi, date, amount, upi_id of the user and his/her user_id. Finally it consists of a submit button in order to insert the values into the transaction table.

FEEDBACK FORM: Here one can visualize a feedback mail to the admin wherein a drop down box of admin mail is present along with which a subject and a space to enter message of comment type is present and at the end a submit button.

3.2 CONNECTIVITY:

Connecting to a MySQL database

You need your MySQL server address (if the database is on the same server as the web server it will most likely be **localhost** or **127.0.0.1**), username, password and database name. Create a **filenamehere.php** file and open and close the php code with tags before the html, you can put regular html after it. Open the file in a browser and you should see nothing apart from the title tag, if you see the error the username/password or database name may be wrong.

// Create connection

```
<?php
$connect_error = 'sorry, server is down';
mysql_connect('localhost', 'root', '') or die($connect_error);
mysql_select_db('lr') or die($connect_error);
?>
```

Here the localhost is the servername, root is the username and since I have not given any password that field is empty.

CHAPTER-4

DESCRIPTION OF FUNCTIONALITIES

CHAPTER-4

DESCRIPTION OF FUNCTIONALITIES

LOGIN : Here one logs into the dataset with the correct password and performs the features provided by the website through out using the given user_id.

MAINTAINS: This is to combine the given upi_id with the customer's user_id in order to transact using his/her phone number which is the link to the combination.

TRANSACTION: This is the module where one uses his/her upi_id to credit or debit into the opponent's upi_id with the amount to be specified.

HISTORY: Here one can view the history of the transactions taken place with amount displayed and the date of the transaction by entering one's user_id.

FEEDBACK: One can provide feedback to the admin about the over all view of the webste in order to help the developer to improve his skills.

CHAPTER-5

IMPLEMENTATION USING MYSQL/PHP

CHAPTER-5

5.1 CREATING TABLES:

```
CREATE TABLE `user` (
  `User_id` int(50) NOT NULL AUTO_INCREMENT,
  `FName` varchar(50) NOT NULL,
  `MName` varchar(50) DEFAULT NULL,
  `LName` varchar(50) NOT NULL,
  `Email` varchar(50) NOT NULL,
  `password` varchar(50) NOT NULL,
  PRIMARY KEY (`User_id`)
) ENGINE=InnoDB AUTO_INCREMENT=30 DEFAULT CHARSET=latin1

CREATE TABLE `transaction` (
 `Trans_id` int(11) NOT NULL AUTO_INCREMENT,
 `Amount` decimal(6,2) NOT NULL,
 `Sender_Upi` varchar(20) NOT NULL,
 `Date` date NOT NULL,
 `User_id` int(50) NOT NULL,
 `Upi_id` varchar(20) NOT NULL,
 PRIMARY KEY (`Trans_id`),
 KEY `User_id` (`User_id`),
 KEY `Upi_id` (`Upi_id`),
 CONSTRAINT `transaction_ibfk_1` FOREIGN KEY (`User_id`) REFERENCES `user`
(`User_id`) ON DELETE CASCADE,
 CONSTRAINT `transaction_ibfk_2` FOREIGN KEY (`Upi_id`) REFERENCES `bank`
(`Upi_id`) ON DELETE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=28 DEFAULT CHARSET=latin1

CREATE TABLE `maintains` (
 `Upi_id` varchar(20) NOT NULL,
 `User_id` int(50) NOT NULL,
 `Ph_No` int(10) NOT NULL,
 PRIMARY KEY (`Upi_id`,`User_id`),
 KEY `User_id` (`User_id`),
 CONSTRAINT `maintains_ibfk_2` FOREIGN KEY (`User_id`) REFERENCES `user`
(`User_id`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1


CREATE TABLE `history` ( `Trans_id` int(11) NOT NULL,

`History` varchar(255) NOT NULL,

 PRIMARY KEY (`Trans_id`,`History`),
```

```
CONSTRAINT `history_ibfk_1` FOREIGN KEY (`Trans_id`) REFERENCES
`transaction` (`Trans_id`) ON DELETE CASCADE
) ENGINE=InnoDB DEFAULT CHARSET=latin1

CREATE TABLE `feedback` (
 `Fid` int(11) NOT NULL AUTO_INCREMENT,
 `Msg` blob NOT NULL,
 `Subject` varchar(255) NOT NULL,
 `A_Email` varchar(50) NOT NULL,
 `User_id` int(50) NOT NULL,
 PRIMARY KEY (`Fid`),
 KEY `A_Email` (`A_Email`),
 KEY `User_id` (`User_id`),
 CONSTRAINT `feedback_ibfk_1` FOREIGN KEY (`A_Email`) REFERENCES
`admin` (`A_Email`) ON DELETE CASCADE,
 CONSTRAINT `feedback_ibfk_2` FOREIGN KEY (`User_id`) REFERENCES `user`
(`User_id`)
) ENGINE=InnoDB AUTO_INCREMENT=34 DEFAULT CHARSET=latin1

CREATE TABLE `bank` (
 `Upi_id` varchar(20) NOT NULL,
 `Bank_Bal` decimal(10,5) NOT NULL DEFAULT 5000.00000,
 `Acc_No` varchar(20) NOT NULL,
 PRIMARY KEY (`Upi_id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1

CREATE TABLE `admin` (
 `A_Email` varchar(50) NOT NULL,
 `Name` varchar(50) NOT NULL,
 `Ph_No` int(12) NOT NULL,
 PRIMARY KEY (`A_Email`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1


CREATE TRIGGER `hist` AFTER INSERT ON `transaction`

 FOR EACH ROW INSERT into history values (new.trans_id,Concat(now(),'\tpaid

\t',new.Amount,'\t to \t',new.sender_upi))
```

5.2 CODES IN PHP

1.**REGISTER.PHP:**

```
<?php
session_start();
$con = mysqli_connect("localhost","root","") or die("Unable to connect");
mysqli_select_db($con,'shriupi');
?>
<!DOCTYPE html>
<html>
<head>
<title>Registration Page</title>
<link rel="stylesheet" href="style.css">
</head>
<style>
body{
background-image: url("reg.jpg");
background-repeat:no-repeat;
   background-size:cover;
}
</style>
<body style="background-color:#3498db">

        <div id="main-wrapper">
                <center>
                        <h2>Registration Form</h2>
                        <img src="upi.png" class="avatar"/>
                </center>

                <form class="myform" action="register.php"method="post">
                        <label><b>First Name:</b></label><br>
                        <input name="fname" type="text" class="inputvalues"
placeholder="Type your First Name" required/><br>
                        <label><b>Middle Name:</b></label><br>
                        <input name="mname" type="text" class="inputvalues"
placeholder="Type your Middle Name" /><br>
                        <label><b>Last Name:</b></label><br>
                        <input name="lname" type="text" class="inputvalues"
placeholder="Type your Last Name" required/><br>

                        <label><b>Email:</b></label><br>
<input name="email" type="text" class="inputvalues" placeholder="Type your email"
required/><br>
                        <label><b>Password:</b></label><br>
```

```
                        <input name="password" type="password" class="inputvalues"
placeholder="Your password" required/><br>
                        <label><b>Confirm Password:</b></label><br>
                        <input name="cpassword" type="password" class="inputvalues"
placeholder="Confirm password" required/><br>
                        <!input name="submit_btn" type="submit" id="signup_btn"
value="Sign Up"/><br>
                        <!-a href="login.php"><!-input type="button" id="back_btn"
value="Back"/></a>
                        <button name="register" class="sign_up_btn" type="submit">Sign
Up</button>

                                <a href="login.php"><button type="button"
class="back_btn"><< Back to Login</button></a>
                </form>

                <?php
                        if(isset($_POST['register']))
                        {
                                //echo '<script type="text/javascript"> alert("Sign Up
button clicked") </script>';

                                $fname =$_POST['fname'];
                                $mname =$_POST['mname'];
                                $lname =$_POST['lname'];

                                $email =$_POST['email'];
                                $password = $_POST['password'];
                                if($password==$cpassword){
                                        $query= "select * from user WHERE
email='$email'";
                                        $query_run = mysqli_query($con,$query);

                                        if(mysqli_num_rows($query_run)>0){
        echo '<script type="text/javascript"> alert("User already exists.. try another
username") </script>';}
else{
$query= "insert into user values('','$password','$fname','$mname','$lname','$email')";
$query_run = mysqli_query($con,$query);
if($query_run){
echo '<script type="text/javascript"> alert("User Registered.. ") </script>';
$sql_stmt = "SELECT * FROM user where email='$email'" ;
$result = mysqli_query($con,$sql_stmt);
$rows = mysqli_num_rows($result);
if($rows)
{if($row=mysqli_fetch_array($result))
{$user_id = $row['User_id'];}}
$_SESSION['user_id']= $user_id;
```

```
$_SESSION['fname']= $fname;
$_SESSION['email']= $email;
header( "Location: userid.php");
}
else
{
echo '<script type="text/javascript"> alert("'.mysqli_error($con).'") </script>';
}}
}
else{
echo '<script type="text/javascript"> alert("Password and confirm password does not
match!") </script>';}
}
?>
</div>
</body>
</html>
```

2.MAINTAINS.PHP:

```
<!DOCTYPE html>
<html>
<head>
<title>Maintains</title>
<link rel="stylesheet" href="stylem.css">
</head>
<body style="background-color:#f1c40f">
        <div class="main">
        <h1>We are always ready to serve!</h1>
        </div>
        <div class="main-wrapper">
        <form id="main-wrapper" method="POST" action="">
        <input type="text" name="User_id" class="form-control" placeholder="Enter
your User Id" required>
        <br>
                <input type="text" name="Upi_id" class="form-control"
placeholder="Enter your Upi Id" required><br>
                <input type="text" name="Ph_No" class="form-control"
placeholder="Enter your phone number" required><br>
<input type="submit" name="register" class="form-control submit" value="Register"
                </div>
                </form>
<?php

        if(isset($_POST['register']))
        {
                @$User_id=$_POST['User_id'];
                @$Upi_id=$_POST['Upi_id'];
```

```
                @$Ph_No=$_POST['Ph_No'];
                $query = "select * from maintains ";

                                        $query_run = mysqli_query($con,$query);
                                        if($query_run){
        {
                                                $query="INSERT into maintains values
('$Upi_id','$User_id','$Ph_No')";
                                                $query_run = mysqli_query($con,$query);
                                                if($query_run )
{
echo '<script type="text/javascript">alert("User Registered.. Welcome")</script>';
$_SESSION['User_id'] = $User_id;
$_SESSION['Upi_id'] = $Upi_id;
$_SESSION['Ph_No'] = $Ph_No;
header( "Location: pr.php");
}
else
{echo '<p class="bg-danger msg-block">Registration Unsuccessful due to server error.
Please try later</p>';
}
}
else{
echo '<script type="text/javascript">alert("DB error")</script>';}
        }?>
        </div>
</body>
</html>
```

Code of History:

```
<?php
        session_start();
        $con = mysqli_connect("localhost","root","") or die("Unable to connect");
mysqli_select_db($con,'shriupi');
?>
```

3.HISTORY:

```
<!DOCTYPE html>
<html>
<head>
<title>History</title>
<link rel="stylesheet" href="style.css">
</head>
<body style="background-color:#f1c40f">
        <div id="main-wrapper">
        <center><h2>History</h2></center>

                <form action="#" method="post">
```

```
                        <div class="inner_container">
                                <label><b>UserId</b></label>
                                <input type="number" placeholder="Enter UserId"
name="User_id" required>
                                <button name="ok" class="sign_up_btn"
type="submit">Search</button>
<style type="text/css">
table {
        border: 1px solid black;
        border-collapse: collapse;
        width: 100%;
        color: 3d6459;
        font-family: monospace;
        font-size: 25px;
text-align: left;
}
th{
        background-color: #D96459;
        color: white;
        border: 1px solid black;
}
tr:nth-child(even) {background-color: #f2f2f2}
</style>
<table style="width:100%">
<tr>
<th>Id </th>
<th>History</th>

</tr>

<?php
        if(isset($_POST['ok']))
{
@$User_id=$_POST['User_id'];
query = "select * from history h where h.Trans_id in (Select t.Trans_id from transaction t
where t.User_id='$User_id')";
$result = mysqli_query($con,$query);
While($row = mysqli_fetch_array($result, MYSQLI_NUM)){
echo "<tr><td>". $row[0] ."</td><td>". $row[1]  ."</td></tr>";
}
echo "</table>";
}
?></table>
</div>
</body>
</html>
```

CHAPTER-6

TESTING AND SNAPSHOTS

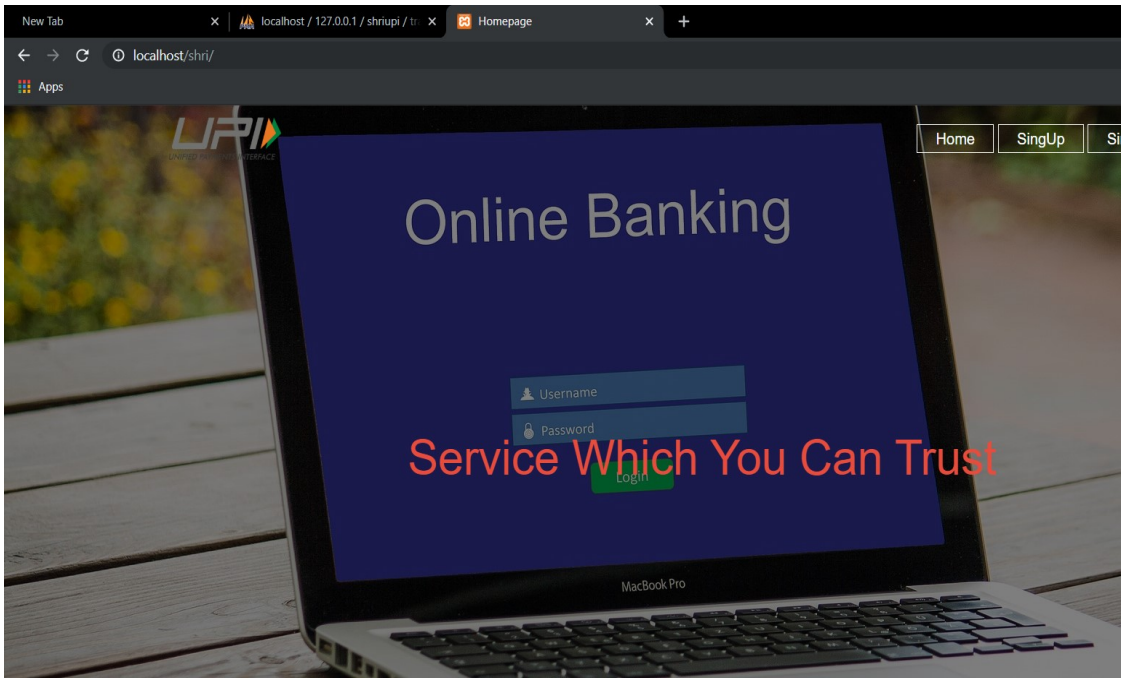
CHAPTER-6

6.1 TEST CASES:

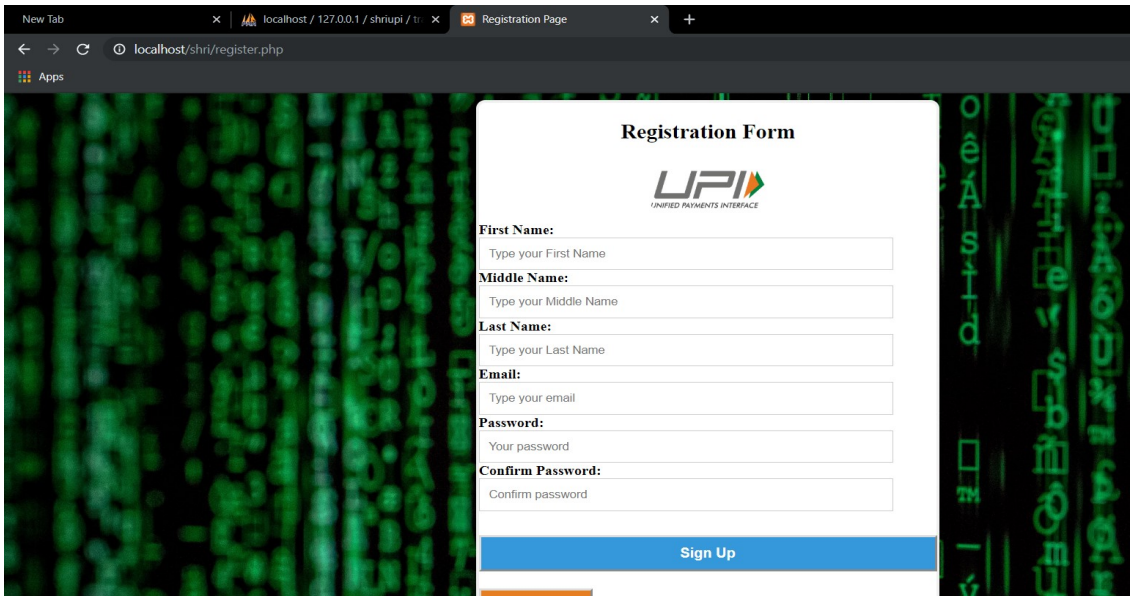
TEST CASE DESCRIPTION	TEST CASE	EXPECTED OUTPUT	OBTAINED OUTPUT	RESULT
First Name	Name<3 characters	Should not accept	Accepted	Fail
First Name	Name>=3 characters	Accepted	Accepted	Pass
Password	**(NOT)** minimum one upper case ,one digit and 6 characters	Should not accept	Accepted	Fail
Password	minimum one upper case ,one digit and 6 characters	Accepted	Accepted	Pass
Confirm password	Password!=Confirm password	Should not accept	Accepted	Fail
Confirm password	Password=Confirm password	Accepted	Accepted	Pass
If Fname, Password, Lname	If Combination does NOT exist then user name is unique	Accepted	Accepted	Pass

If Fname, Password, Lname	If Combination exists then user name already exists	Should not accept	Accepted	Fail
Check Balance	If balance<3000	Should not accept	Accepted	Fail
Check Balance	If balance>=3000	accepted	Accepted	Pass
Email	Has to be in xxx@gmail.com	Should not accept	Accepted	Fail
Email	xxx@gmail.com	Accepted	Accepted	Pass

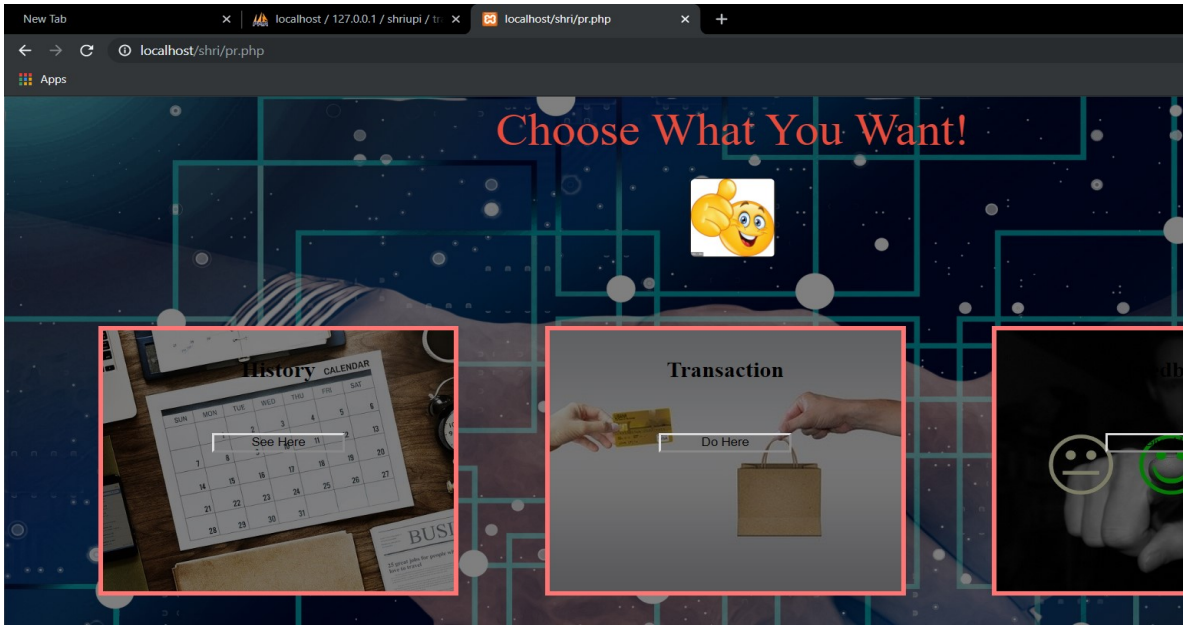
6.2 SNAPSHOTS:



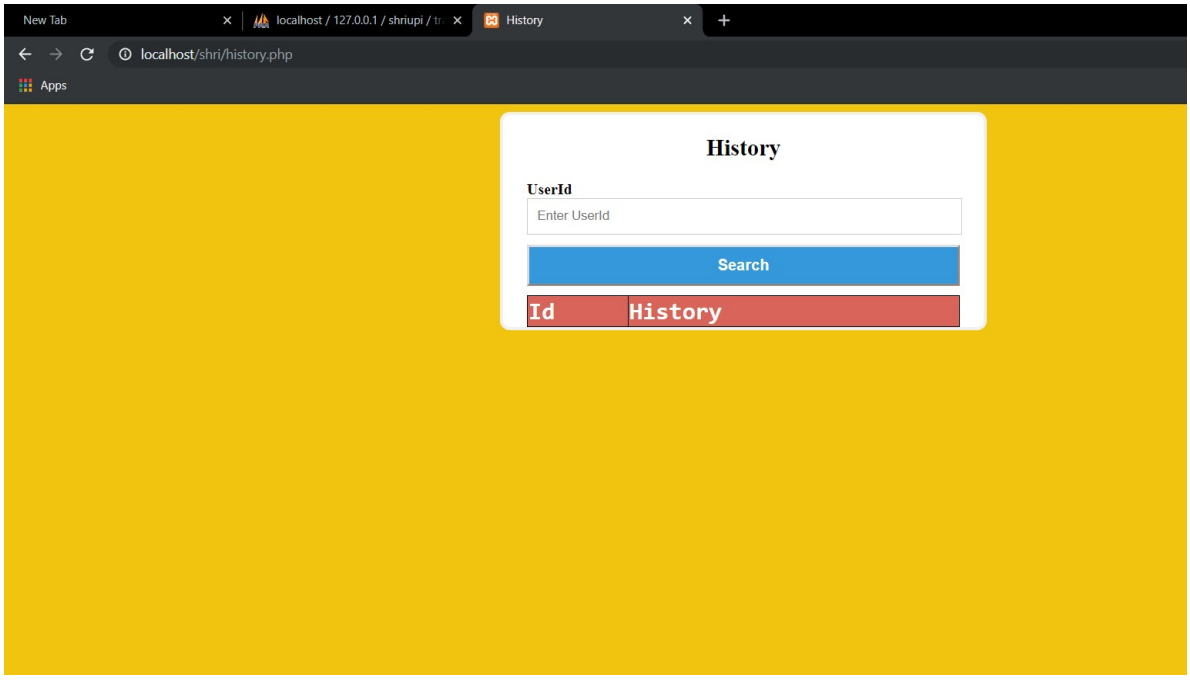
Snapshot 6.2.1 Home Page



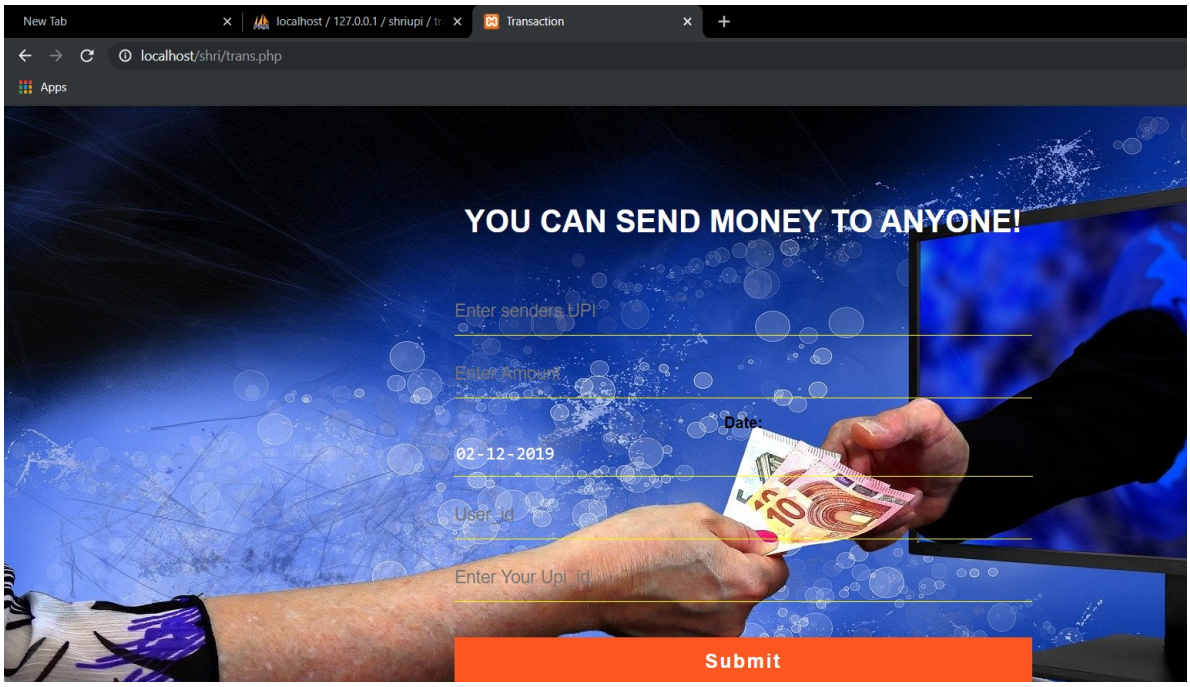
Snapshot 6.2.2 Registration Page



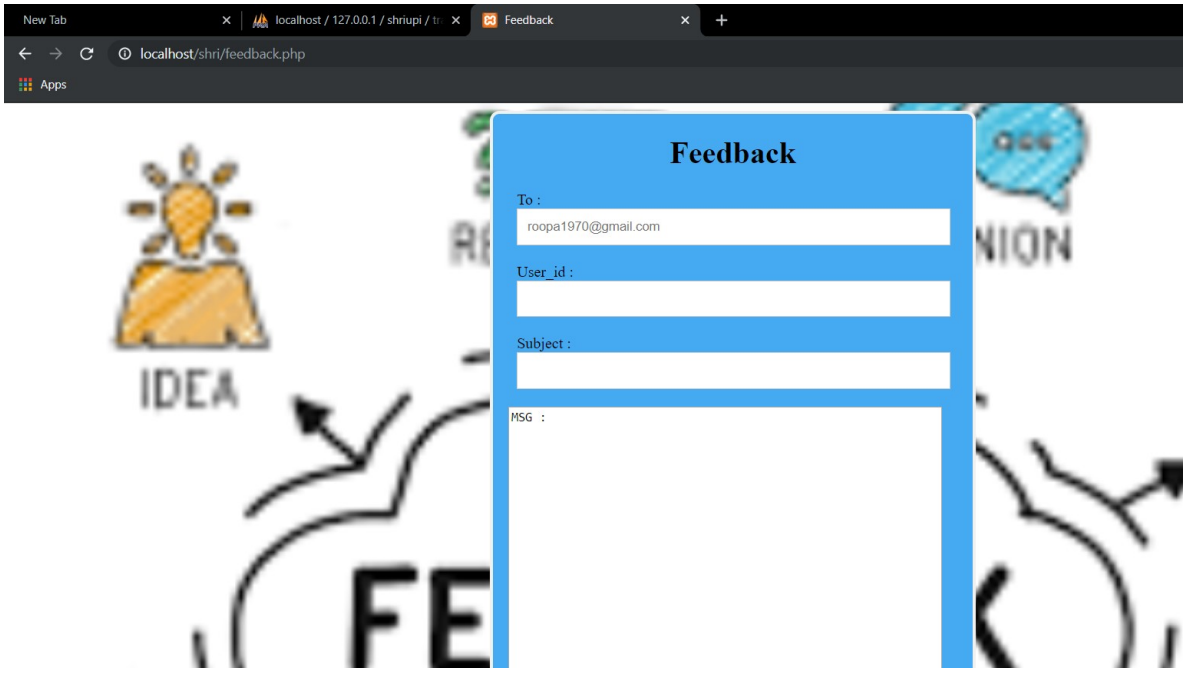
Snapshot 6.2.3 Home Page2



Snapshot 6.2.4 History Page



Snapshot 6.2.5 Transaction Page



Snapshot 6.2.6 Feedback Form

CHAPTER-7

APPLICATIONS

CHAPTER-7

APPLICATIONS:

The UPI-System option is relatively new, and it is gaining popularity in India. Using this, customer can pay for their expenses. They can pay to anyone across India by using their UPI_ID.

Crediting and debiting of money can be done in a simpler way with just your UPI_ID in one's hand. After registering the bank account plays on your fingertips. Also one can view their history of transactions and in some advanced websites can even beget rewards (which we will be updating soon). With feedback provisions one can send the improvements or changes to be incurred to the developer so that necessary transitions can be made.

CHAPTER-8

CONCLUSION

CHAPTER-8

CONCLUSION:

This application was designed to address the various inadequacies we identified in the existing systems such as standing in long queues in banks, payment for movies after reaching the site paying extra money for transacting with other bank accounts, remembering card number or account number in order to transact and so on for the back-end user. Hopefully this newly designed system not only addresses the problems effectively but also inspires further innovation in the field of user interfaces and data transparency. Developed is useful for small size to medium organisation for handling the inventory.