ABSTRACT

Since the 1980s, video gaming has become a popular form of entertainment and a part of modern popular culture in most parts of the world. Video arcades with large, graphics-decorated coin-operated machines were common at malls and popular, affordable home consoles enabled people to play games on their home TVs. After a series of events, here we are today with everything accessible to us with a click of a button, and video games are no exception. Sitting on the shoulders of giants, we now have the ability to create software that allows us to buy video games online, a popular example is the digital distribution platform, 'Steam', developed by Valve Corporation in 2003. Albeit more convenient, it is nonetheless inexpensive than buying a physical CD. This DBMS project introduces the concept of online video game rentals. When translated to a working concept, the user will be required to download a software on which they will be able to rent their choice of video games, pay for the number of days they had it for and they will not be downloading the game rented, the software will support gameplay hence allowing users to play on the software itself.

ACKNOWLEDGEMENT

With gratitude, I would like the acknowledge all those whose guidance and encouragement allowed us to successfully complete this project

We would like to thank our guide, Anu Jose ma'am, for continually reviewing our project through its progress.

A special thanks to Lohith sir and Prasad MS sir, from the ISE department, for clearing a lot our doubts allowing us to move forward with later aspects of our project.

We think it is important to mention Varsha P ma'am, of the MCA department, for introducing us to the concept of JSP and doing an good job of explaining JDBC allowing us to lay the foundation of our project.

We would like to thank all the teaching and non-teaching staff who directly and indirectly contributed to the completion of this project

Most importantly we want to thank our peers who were all very generous in terms of sharing their knowledge with us.

This project would not have been possible, in its entirety, without the aforementioned.

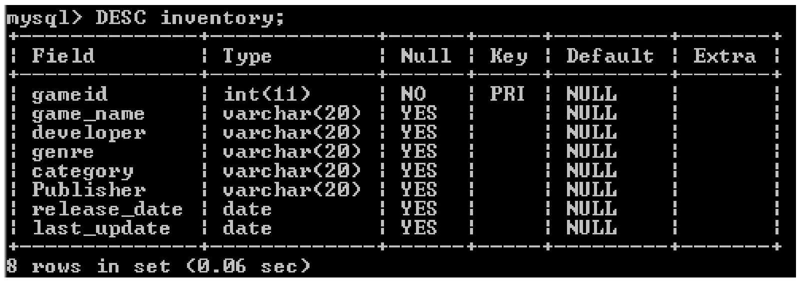
PriyavaniUpadhyay

Bharath VK

TABLE OF CONTENTS

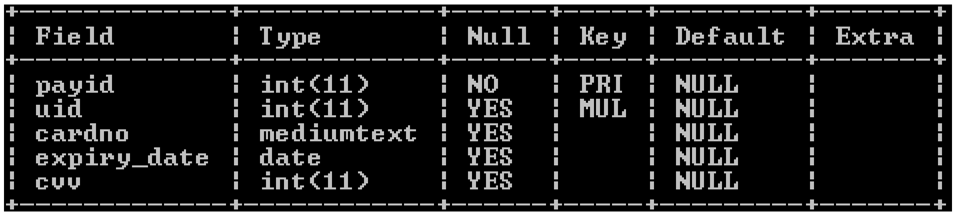
LIST OF TABLES AND FIGURES

1) INVENTORY – THIS TABLE HOLDS THE LIST OF GAMES AVAILABLE



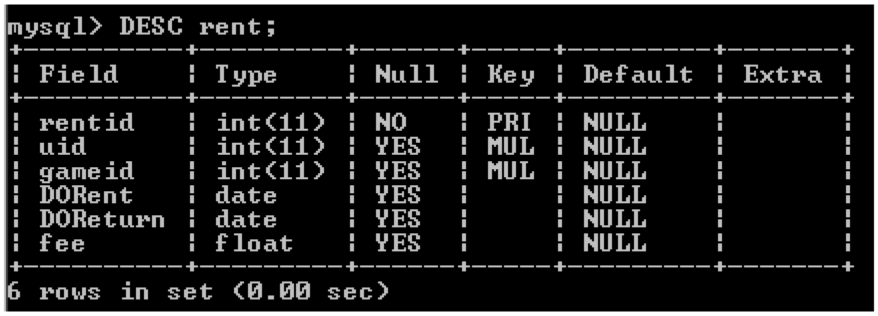
```
mysql> DESC inventory;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| gameid       | int(11)     | NO   | PRI | NULL    |       |
| game_name    | varchar(20) | YES  |     | NULL    |       |
| developer    | varchar(20) | YES  |     | NULL    |       |
| genre        | varchar(20) | YES  |     | NULL    |       |
| category     | varchar(20) | YES  |     | NULL    |       |
| Publisher    | varchar(20) | YES  |     | NULL    |       |
| release_date | date        | YES  |     | NULL    |       |
| last_update  | date        | YES  |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
8 rows in set (0.06 sec)
```

2) PAYMENT_INFO – THIS TABLE CONTAINS THE INFORMATION ABOUT THE USER'S PAYMENT DETAILS



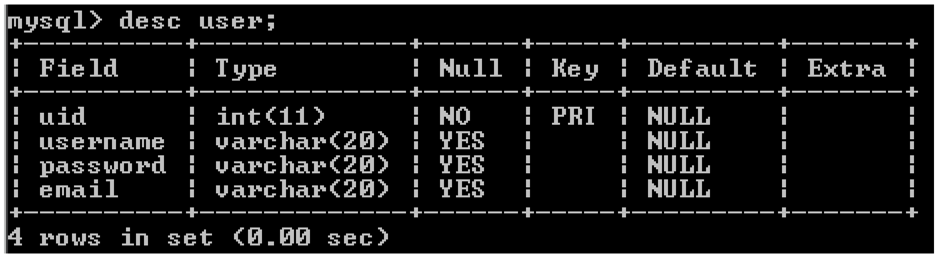
```
+-------------+------------+------+-----+---------+-------+
| Field       | Type       | Null | Key | Default | Extra |
+-------------+------------+------+-----+---------+-------+
| payid       | int(11)    | NO   | PRI | NULL    |       |
| uid         | int(11)    | YES  | MUL | NULL    |       |
| cardno      | mediumtext | YES  |     | NULL    |       |
| expiry_date | date       | YES  |     | NULL    |       |
| cvv         | int(11)    | YES  |     | NULL    |       |
+-------------+------------+------+-----+---------+-------+
```

3) RENT – THIS TABLE CONTAINS THE RENT DETAILS



```
mysql> DESC rent;
+----------+---------+------+-----+---------+-------+
| Field    | Type    | Null | Key | Default | Extra |
+----------+---------+------+-----+---------+-------+
| rentid   | int(11) | NO   | PRI | NULL    |       |
| uid      | int(11) | YES  | MUL | NULL    |       |
| gameid   | int(11) | YES  | MUL | NULL    |       |
| DORent   | date    | YES  |     | NULL    |       |
| DOReturn | date    | YES  |     | NULL    |       |
| fee      | float   | YES  |     | NULL    |       |
+----------+---------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

4) USER – THIS TABLE CONTAINS INFORMATION ABOUT USER DETAILS



```
mysql> desc user;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| uid      | int(11)     | NO   | PRI | NULL    |       |
| username | varchar(20) | YES  |     | NULL    |       |
| password | varchar(20) | YES  |     | NULL    |       |
| email    | varchar(20) | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
4 rows in set (0.00 sec)
```

CHAPTER 1

INTRODUCTION

This project deals with the implementation of an online portal for video game rentals. The implementation is done as a web application. The idea behind this concept is to make video gaming affordable to some extent by creating a web application based on a pay per day model i.e. a rent model which drastically reduces the cost of playing a video game.

The games industry's shift from brick and mortar retail to digital downloads led to a severe sales decline at video game retailers such as GameStop, following other media retailers superseded by Internet delivery, such as Blockbuster, Tower Records, and Virgin Megastores. However, video game retailers do impact the price of video games. First off, the retailer puts pressure on publishers to keep prices on digital platforms as high (or nearly as high) as they are at retail.If publishers don't, they face the risk of losing much needed shelf-space that they currently rely on for the majority of their sales.

Although licensing a game would cost around the same as buying it, it yields profits in the long run since we license it once and rent it out as many times as we need. Renting a video game is economical mostly in scenarios where the user would like to try a game out, if the user would like to play the game only for a day, if the video game has a very short lifespan etc.

With the help of software like MySQL, Eclipse and Apache Tomcat, we attempt to demonstrate the working concept with respect to the database management aspect.

Every user is required to register with the service.Once the user has registered, they will be allowed to rent any game available in the inventory. As a result of attempting to rent a game they will be required to enter their account details for verification purposes. Once verified with the details in the database, an upfront amount will be extracted from the user’s account. The user will be allowed to play the game for as long as he or she does not choose to return the game. Once they return the game, the number of days between the date of rent and date of return is calculated and multiplied with a constant amount i.e. the fee per day, as a result of which the final amount will be extracted from the user account once he or she goes through the verification process again

CHAPTER 2

SYSTEM REQUIREMENTS

3.1 HARDWARE REQUIREMENTS

3.1.1 Processor: Intel(R) Core(TM) i5-7300HQ CPU @ 2.50GHz, 2501 Mhz, 4 Core(s), 4 Logical Processor(s)

3.1.2 Installed Physical Memory (RAM): 8.00 GB (7.88 GB usable)

3.1.3 System Type: 64-bit Operating System, x64-based Processor

3.2 SOFTWARE REQUIREMENTS

3.2.1 MySQL

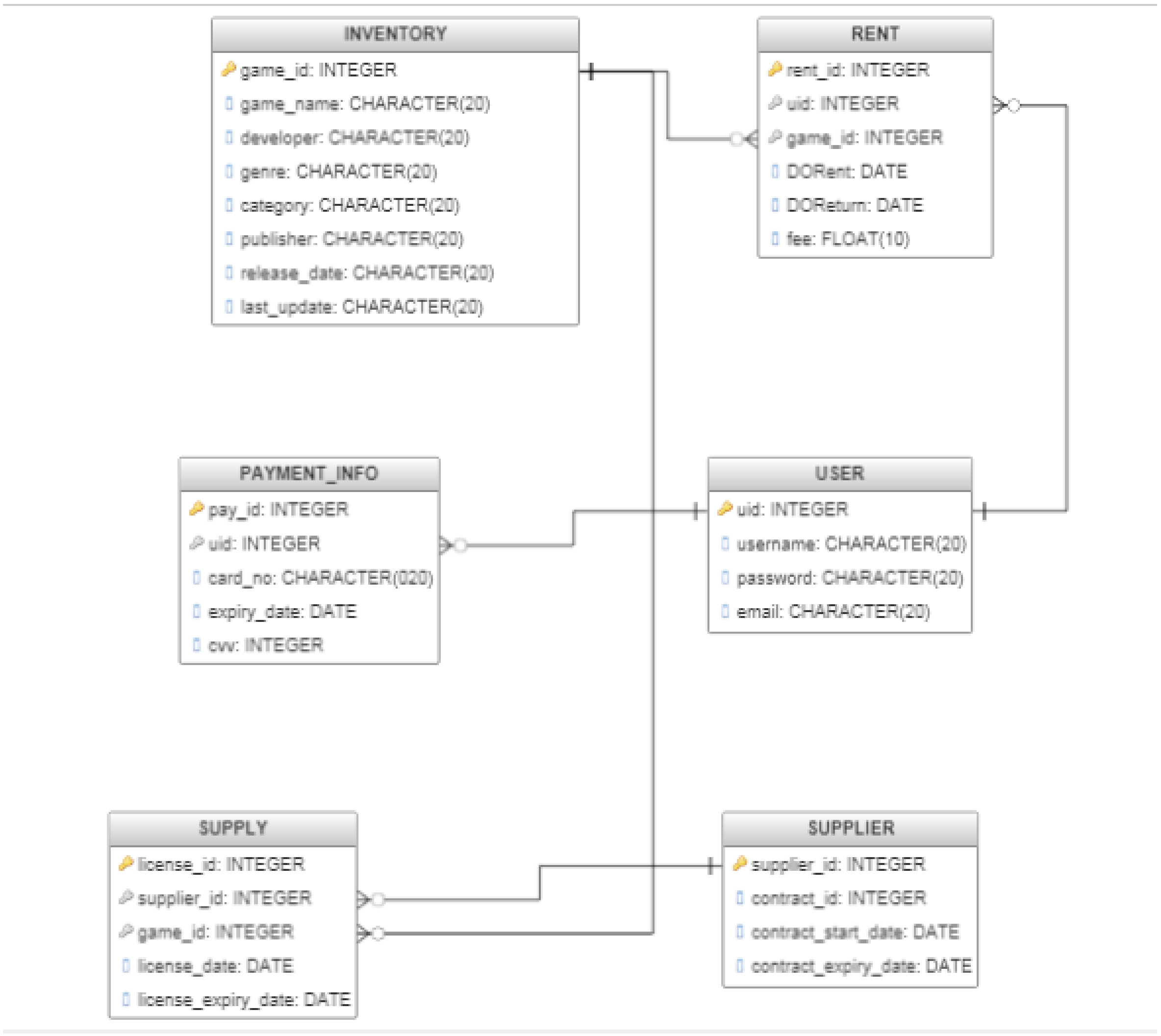
3.2.2 Eclipse IDE for Java EE Developers – Oxygen

3.2.3 Apache Tomcat Web Server

CHAPTER 3

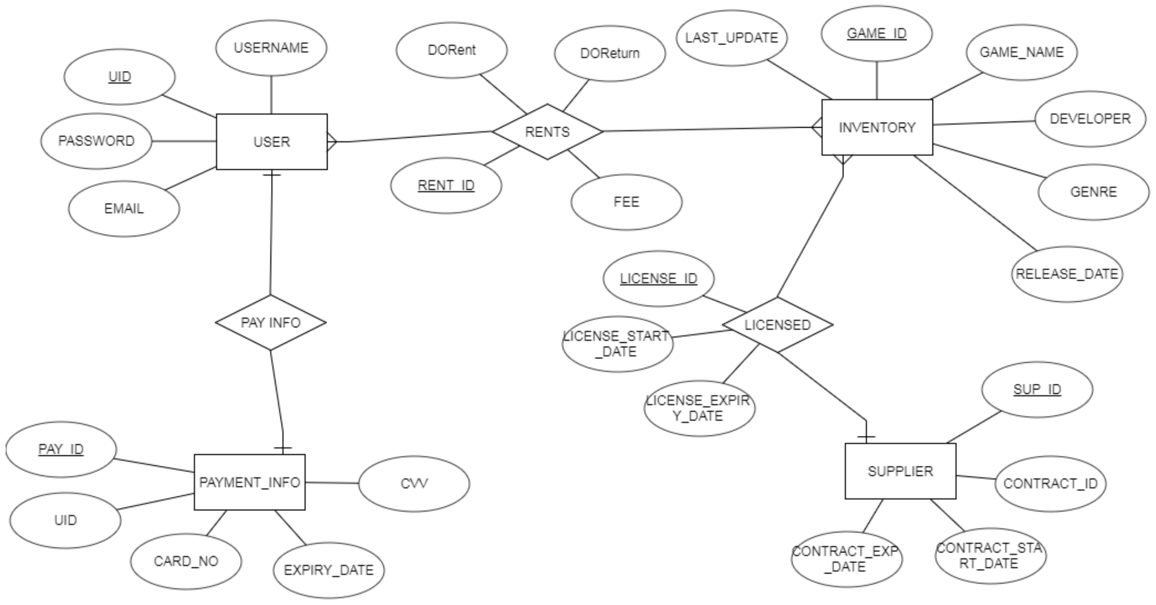
CHAPTER 3.1

SCHEMA DIAGRAM



CHAPTER 3.2

ER DIAGRAM



CHAPTER 4

IMPLEMENTATION

4.1 The following is the code for User Login:

```
<%
    String email = request.getParameter("EmailLogin");
   String password = request.getParameter("PasswordLogin");
    try {
            Class.forName("com.mysql.jdbc.Driver");
            Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/rental", "root", "sqlpswd");
            Statement stmt = con.createStatement();
            if(email == "" || password == "" || email == null || password == null){
%><script>window.alert("Fill in all the fields")</script><%;
            response.sendRedirect("../JSP/index.jsp");
}
      String query = "SELECT * FROM user WHERE email = '"+email+"';";
      String query2 = "SELECT password FROM user WHERE email = '"+email+"';";
      String query3 = "SELECT username FROM user WHERE email = '"+email+"';";
    ResultSet rs3 = stmt.executeQuery(query3);
    rs3.next();
    String uname = rs3.getString(1);
    ResultSetrs = stmt.executeQuery(query);
    if(rs.next()){
            ResultSet rs2 = stmt.executeQuery(query2);
            rs2.next();
            String pass = rs2.getString(1);
            if(pass.equals(password)){
                    System.out.println("Welcome!");
                    session.setAttribute("username", uname);
                    System.out.println(session.getAttribute("username"));
                    response.sendRedirect("../JSP/browse2.jsp");
    }
    else{
            System.out.println("Incorrect password! Please try again");
```

```
                response.sendRedirect("../html/Register.html");
        }
        }
        else{
                System.out.println("Unregistered email id!");
                System.out.println("Please try again");
                response.sendRedirect("../html/Register.html");
        }
        } catch (SQLException e) {
                // TODO Auto-generated catch block
                e.printStackTrace();
        } catch(ClassNotFoundException e1) {
                e1.printStackTrace();
        }
        %>
```

4.2 The following is the code for displaying the search results:

```
    <%
        System.out.println("Inside browse.jsp");
        String genre = request.getParameter("genre");
        System.out.println(genre);
        String category = request.getParameter("category");
        System.out.println(category);
        String developer = request.getParameter("developer");
        System.out.println(developer);
        intgameid;
        String publisher = request.getParameter("publisher");
        System.out.println(publisher);
        try{
                Class.forName("com.mysql.jdbc.Driver");
                Connection con =
    DriverManager.getConnection("jdbc:mysql://localhost:3306/rental", "root", "sqlpswd");
                Statement stmt = con.createStatement();
                String query = "SELECT * FROM inventory WHERE genre = '"+genre+"' OR
    category = '"+category+"' OR developer = '"+developer+"' OR Publisher = '"+publisher+"' ";
                ResultSetrs = stmt.executeQuery(query);
                if(rs.next()){
```

```
                do{     %>

<div id="searchres" class="row w3-black w3-text-white">
<div class="col-md-6" style="padding: 0 0 5px 0; border-right: 1px solid
white;"><imgsrc="../images/gamer.jpg" width="100%" style="float: left;"></div>
<table class="col-md-6 row">
<!-- Repeat for number of attributes-->
<tr><td class="col-sm-2">Name</td><td class=col-sm-4><%=rs.getString(2)%></td></tr>
<tr><tdclass="col-sm-4">Developer</td><td class="col-sm-
8"><%=rs.getString(3)%></td></tr>
<tr><td class="col-sm-4">Genre</td><td class="col-sm-
8"><%=rs.getString(4)%></td></tr>
<tr><td class="col-sm-3"></td><td class="col-sm-2"><form action="confirm.jsp" method =
post><button value = <%= rs.getInt(1) %> id = "rentbtn" name = "gameID" class="w3-btn
w3-display-bottomright w3-center w3-green" style="margin-right: 20px; width:
80px;">Rent</button></form><br><br>
</td></tr><br><br>
</table>
<!-- Till here-->
</div>
</div>
<% }while(rs.next());
    }
    } catch (SQLException e) {
    // TODO Auto-generated catch block
            e.printStackTrace();
    } catch(ClassNotFoundException e1) {
            e1.printStackTrace();
    }%>
    </div>
```

4.3 The following is the code for renting the game:

```
<%
     String uname = session.getAttribute("username").toString();
    if(uname == "" || uname == null){
     %>
```

```
<script>window.alert("You need to login or register to rent a game")</script>
<%;          response.sendRedirect("../html/Register.html");
}
String cardno = request.getParameter("cardno");
System.out.println("cardno:" + cardno);
String cvv = request.getParameter("cvv");
System.out.println("cvv" + cvv);
String expdate = request.getParameter("expdate");
System.out.println("expdate" + expdate);
//String gamename = (String)session.getAttribute("rentgamename");
//System.out.println(gamename);
LocalDateDORent = LocalDate.now();
System.out.println(DORent);
intgid = Integer.parseInt(request.getParameter("gameID"));
System.out.println("gameid is:" + gid);
// Date DOReturn = null;
// float fee = Float.parseFloat(null);
Random rand = new Random();
intrentid = rand.nextInt(100);
try {
    Class.forName("com.mysql.jdbc.Driver");
    Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/rental",
"root", "sqlpswd");
    Statement stmt = con.createStatement();
    if(cardno.equals("") || cvv.equals("") || expdate.equals("") || cardno.equals(null) ||
cvv.equals(null) || expdate.equals(null)){
            response.sendRedirect("../JSP/browse.jsp");
    }
  String query0 = "SELECT uid FROM user where username = '"+uname+"';";
ResultSet rs0 = stmt.executeQuery(query0);
rs0.next();
intuid = rs0.getInt(1);
System.out.println("uid:" + uid);
  String query = "SELECT cardno, expiry_date, cvv FROM payment_info WHERE uid =
'"+uid+"'; ";
ResultSetrs = stmt.executeQuery(query);
if(rs.next()){
    String cno = rs.getString("cardno");
    System.out.println("cno:" + cno);
        String expiry_date = rs.getDate("expiry_date").toString();
    System.out.println("expiry_date:" + expiry_date);
    int cv = rs.getInt("cvv");
    System.out.println("cv:" + cv);
        String query2 = "SELECT * FROM rent WHERE rentid = '"+rentid+"' ;";
    ResultSet rs2 = stmt.executeQuery(query2);
        do{
                rentid = rand.nextInt(100);
                rs2 = stmt.executeQuery(query2);
                }while(rs2.next());
        System.out.println("rentid:" + rentid);
        if(cardno.equals(cno)        &&expdate.equals(expiry_date)        &&cvv.equals(
    String.valueOf(cv))){
        System.out.println("Equal");
         String query3 = "INSERT INTO rent VALUES ('"+rentid+"', '"+uid+"', '"+gid+"',
    '"+DORent+"', null, null )";
        int count = stmt.executeUpdate(query3);
        System.out.println("count:" + count);
        response.sendRedirect("../JSP/user.jsp");

}

}

}catch (SQLException e) {

    // TODO Auto-generated catch block
        e.printStackTrace();
    } catch(ClassNotFoundException e1) {
        e1.printStackTrace();
    }

%>
```

4.4 The following is the code for returning the game:

```
    <%
        String uname = session.getAttribute("username").toString();
        if(uname == "" || uname == null){
    %>
    <script>window.alert("You need to login or register ")</script><%;
```

```
response.sendRedirect("../html/Register.html");
}
    String cardno = request.getParameter("cardno");
    System.out.println("cardno:" + cardno);
    String cvv = request.getParameter("cvv");
    System.out.println("cvv" + cvv);
    String expdate = request.getParameter("expdate");
    System.out.println("expdate" + expdate);
    // intgameid = Integer.parseInt(request.getParameter("gameid"));
    //need to retrieve gameid.
    LocalDateDOReturn = LocalDate.now();
    System.out.println(DOReturn);
    intrentid = Integer.parseInt(request.getParameter("rentID"));
    try {
            Class.forName("com.mysql.jdbc.Driver");
            Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/rental", "root", "sqlpswd");
            Statement stmt = con.createStatement();
            if(cardno.equals("") || cvv.equals("") || expdate.equals("") || cardno.equals(null) ||
cvv.equals(null) || expdate.equals(null)){
            response.sendRedirect("../html/user.html");
    }
    String query0 = "SELECT uid FROM user WHERE username = '"+uname+"'";
    System.out.println("query0 executed");
    ResultSet rs0 = stmt.executeQuery(query0);
    rs0.next();
    intuid = rs0.getInt(1);
    System.out.println("uid:" + uid);
    String query = "SELECT cardno, expiry_date, cvv FROM payment_info WHERE uid  =
'"+uid+"'; ";
    ResultSetrs = stmt.executeQuery(query);
    if(rs.next()){
         String cno = rs.getString("cardno");
            System.out.println("cno:" + cno);
        String expiry_date = rs.getDate("expiry_date").toString();
    System.out.println("expiry_date:" + expiry_date);
    int cv = rs.getInt("cvv");
    System.out.println("cv:" + cv);
    if(cardno.equals(cno)            &&expdate.equals(expiry_date)            &&cvv.equals(
String.valueOf(cv))){
        String query1 = "UPDATE rent set DOReturn = '"+DOReturn+"' WHERE rentid =
'"+rentid+"' ;";
    int count = stmt.executeUpdate(query1);
    System.out.println(count);
    response.sendRedirect("../JSP/user.jsp");
        }
    }
}catch (SQLException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
} catch(ClassNotFoundException e1) {
    e1.printStackTrace();
}
%>
```

4.5 The following is the code for User Logout:

```
<%
    session.invalidate();

    response.sendRedirect("../JSP/index.jsp");

    %>
```

CHAPTER 5

DISCUSSION AND SCREENSHOTS

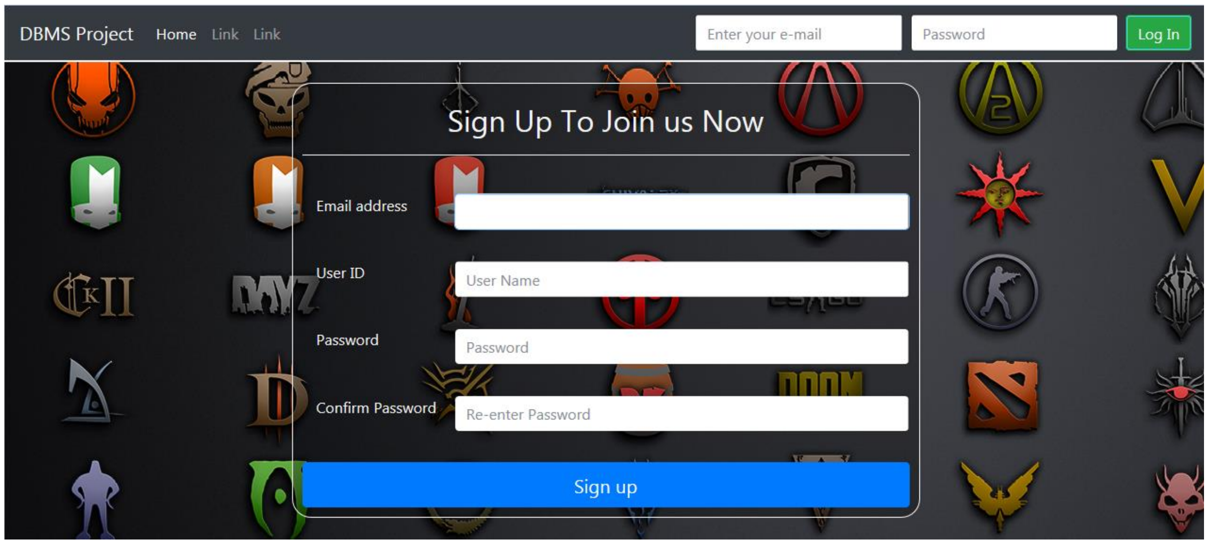
5.1 FLOW OF EVENTS

5.1.1 USER REGISTRATION

STEP 1: User enters their username, password and email address in the appropriate fields.

STEP 2: The application checks the database to see if the given username or email address already exists.

STEP3: If either one already exists, then the user is sent back to the registration page repeat the process, else they are sent to the browse page.

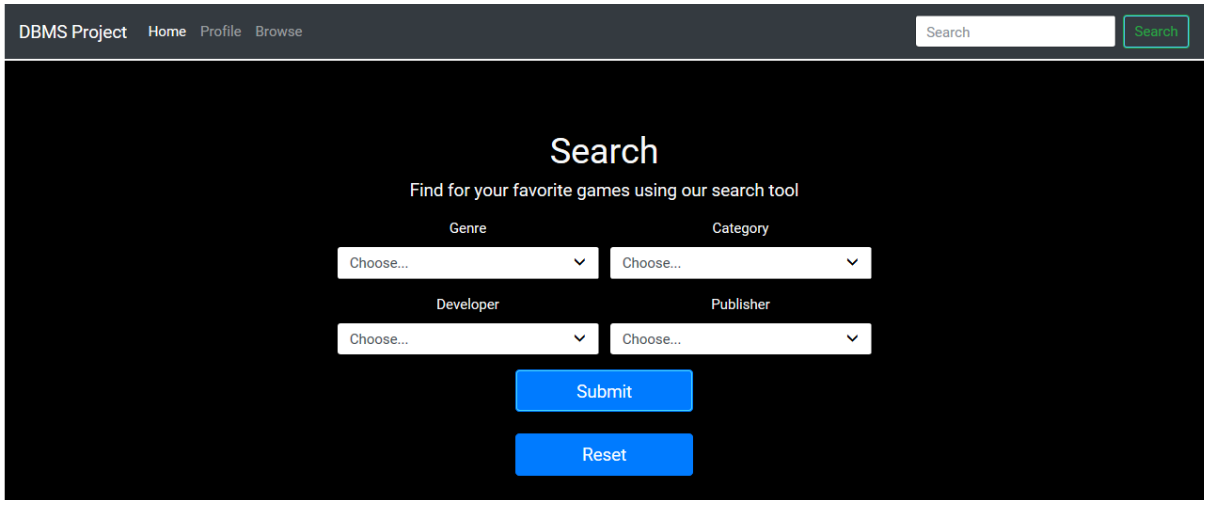


5.1.2 RENT

STEP 1: User enter their preference into the given fields

STEP 2: Based on user preferences, the games are displayed

STEP 3: User chooses to rent a game among the displayed by clicking on rent

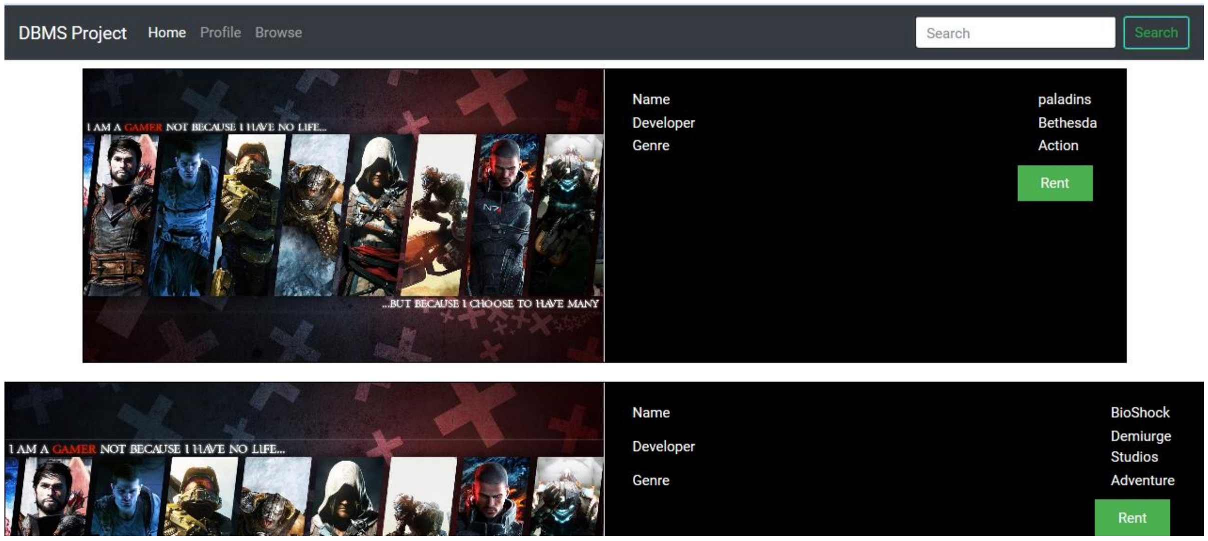


Search

Find for your favorite games using our search tool

Genre	Category
Choose...	Choose...

Developer	Publisher
Choose...	Choose...


DBMS Project
Home
Profile
Browse
Search
Search
I AM A GAMER NOT BECAUSE I HAVE NO LIFE...
...BUT BECAUSE I CHOOSE TO HAVE MANY
Rent
I AM A GAMER NOT BECAUSE I HAVE NO LIFE...
Rent

Name	paladins
Developer	Bethesda
Genre	Action

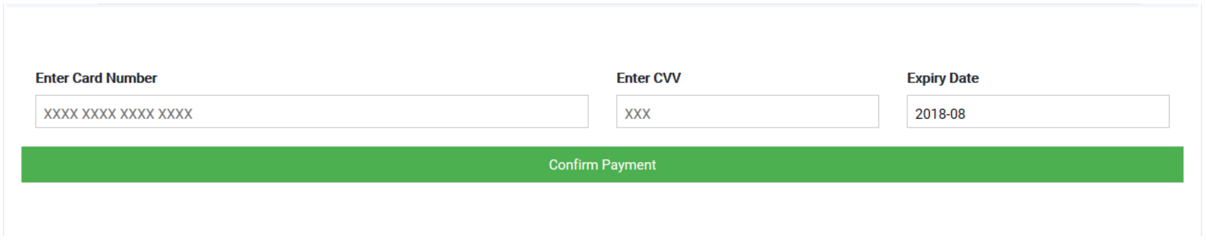
Name	BioShock
Developer	Demiurge Studios
Genre	Adventure

5.1.3 PAYMENT

STEP 1: As a result of selecting a game to rent, the user is prompted to enter their account details

STEP 2: The entered account details will be verified with the account details present in the database

STEP 3: Once the verification is successful, the option to return the game will be displayed on the user's profile page for as long as the user has the game

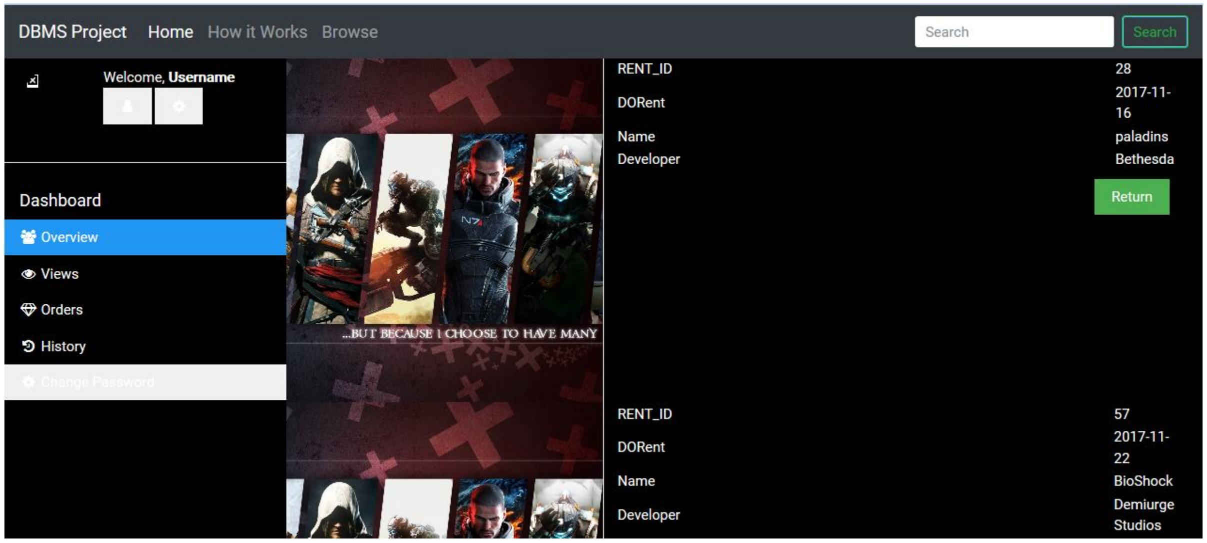


5.1.4 RETURN

STEP 1: User clicks on the return button, on their profile page, besides the game they would like to return.

STEP 2: The user is prompted to enter their account details.

STEP 3: Once the verification process is successful, the returned game, along with all of the previously played and returned games, will be displayed in the orders history page.



RENT_ID	28
DORent	2017-11-16
Name	paladins
Developer	Bethesda
RENT_ID	57
DORent	2017-11-22
Name	BioShock
Developer	Demiurge Studios

CHAPTER 6

CONCLUSION AND FUTURE SCOPE

Our project aims to deliver a full-fledged gaming experience for a fraction of the cost you would be spending in actually buying it.

By being an online web application, our project makes sure that users don't have to be worried about the physical damage of their games that may occur in the normal game CDs. The Users download the game and return it after they finish it.

To conclude, we would like to say that our project offers an efficient, hassle-free and inexpensive way of gaming.

In the future, we could expand our project with more games and more diversity in games. We can also improve the gaming experience by adding more features like a community where like-minded gamers can talk to each other and play together, and/or timely competitions among the users, an improved user experience, and an option for people who already bought the game to make money by renting it out etc.

CHAPTER 7

BIBLIOGRAPHY

WEBSITES

[1] www.w3schools.com

[2] www.en.wikipedia.org

[3] www.tutorialspoint.com

[4] www.stackoverflow.com

[5] www.quora.com

[6] www.mysqltutorial.org

[7] www.bootstrap.com