STADIUM SEAT BOOKING SYSTEM

S.Praveen Kumar
B.tech II Year
Department of CSE
SR Engineering College
sambari.praveen444@gmail.com

Sana Farheen
B.tech II Year
Department of CSE
SR Engineering College

Naheer Fatima
B.tech II Year
Department of CSE
SR Engineering College

K.Ravi Chythanya
Assistant Professor
Department of CSE
SR Engineering College
chythu536@gmail.com

ABSTRACT

The main aim of our project is to manage the details of seats, show times, match name, show date, cost. It manages the booking of above things. The project is totally built for the users where as only the admin is guaranteed the access to update or change the details. The purpose of the project is to build an application program to reduce the manual work for managing the seats,booking. This project gives crystal clear information which can be easily understood by the customer**.**

Keywords: booking,details,application program.

INTRODUCTION

The stadium is a centerpiece for the hosting of important events especially sports events such as soccer. It is a big task to handle it. Ticket purchase is an essential aspect of stadium management. The method can also be deemed to be a critical one, as the failure of it would result in a significant economic loss for the company running the stadium, and, in the case of the software failing, probably thousands of angry fans stuck outside the ground, many of whom may decide not to return – against this would have implications for the managers of the stadium, and apparently the software developers too. Therefore, the system must be created using well established.

OBJECTIVE

It provides the searching facilities based on seats, show time, stadium names, show dates. The concept of files is there to store the information for the customer who reserved it. It provides efficiency and security.

ELEMENTS USED IN THE PROJECT

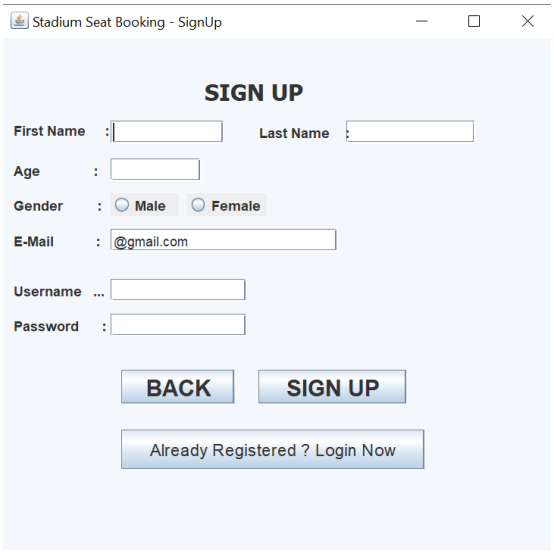
- **Java:**
  Java is simple to learn. It was designed to be easy to use and is therefore easy to write, compile, debug, and study from other programming languages. It allows you to design modular programs and reusable code.Java is platform-independent. One of the most essential advantages of Java is its ability to flow easily from one computer system to another. The capability to run the same program on many different systems is important to World Wide Web software, and Java wins at this by being platform-independent at both the source and binary levels. Because of Java's robustness, ease of use, cross-platform capabilities and security features, it has become a language of choice for providing worldwide Internet solutions.Java provides guarantees for security such as Security-related APIs and Byte-code verification. Java has the characteristic of Stack, which supports to retrieve and save data more efficiently. It is multithreaded as it can perform various tasks simultaneously within a program.
- **Swings:**

Swing is the collection of user interface components for the Java programs. It is portion of Java foundation classes that are referred to as JFC. In easy words, Swing is the graphical user interface toolkit that is utilized for developing the windows based java applications or programs. Swing is the successor of AWT which is known as Abstract window toolkit API for Java and AWT components are a largely heavyweight.The components are lightweight as compared to AWT components. It gives a good interface to the user for all the platforms. It is not specifically for one platform. The components are written in Java and platform independent as well. The Java foundation classes were first appeared in 1997 and then later on it is termed as Swing. To use the swing in java, javax. swing package needs to be used or import. It is also known as Java Swing.

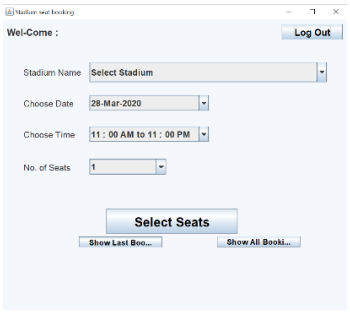
DESIGN

- Login Frame

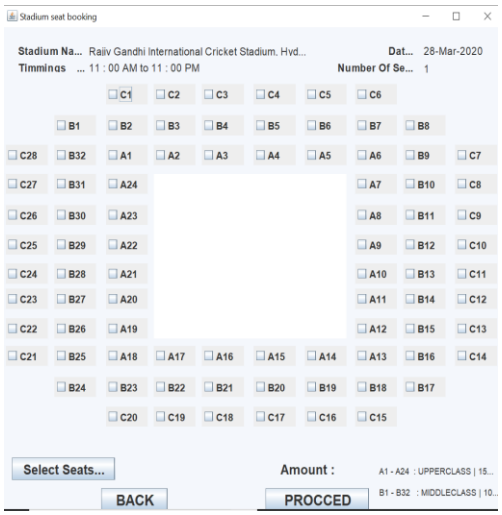
- Registration Frame:



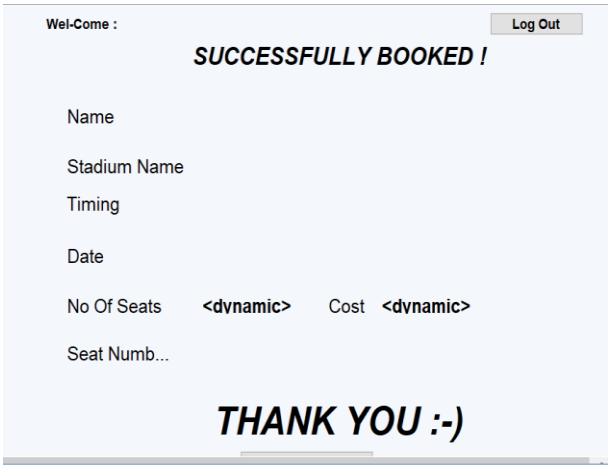
- Third Frame:



- **Fourth Frame:**



- **Fifth Frame:**



IMPLEMENTATION

- Login Page Code

```
LoginButton.addActionListener(new ActionListener() {
                        public void actionPerformed(ActionEvent e) {
                                boolean flag = false;
                                try {
                                        fin = new FileInputStream("LoginDetails.dat");
                                        ois = new ObjectInputStream(fin);
                                        ArrayList<User> al1;
```

```
                    String name = usernamevalue.getText();
                    String pass = passwordvalue.getText();
                    while((al1=(ArrayList)ois.readObject())!=null){
                        User u1 = al1.get(0);
if(pass.equals(u1.getPassword())&&name.equals(u1.getUname())){
                            flag = true;
                            break;
                        }
                    }
                    if(flag==true){
            JOptionPane.showMessageDialog(jf,"Login Successfull !");
```

- Registration Frame code:

```
    SignUpbutton.addActionListener(new ActionListener(){
        public void  actionPerformed(ActionEvent e){
            String u = usernamevalue.getText();
            String p = passwordvalue.getText();
    if(usernamevalue.getText().equals("") || passwordvalue.getText().equals("")) {
        JOptionPane.showMessageDialog(jf, "Please enter Fields !");
            }
            else {
            try {
                al.add(new User(u,p));
                File file = new File("LoginDetails.dat");
                if(file.exists()) {
        oos = new ObjectOutputStream(new FileOutputStream(file,true)) {
            protected void writeStreamHeader() throws IOException {
                        reset();
                    }
                };
                }
                else
        oos = new ObjectOutputStream(new FileOutputStream(file));
                oos.writeObject(al);
                usernamevalue.setText("");
                passwordvalue.setText("");
JOptionPane.showMessageDialog(jf,"Signup Successfull ! You can Login now !!");
```

- Third Frame Code:

```
ShowBookings.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {

        int c=1;
```

```
                    String[] PastBookings=new String[10];
                    String Incr=new String("   ALL BOOKINGS \n");
                    try {
                            fin = new FileInputStream("Bookings.dat");
                            ois = new ObjectInputStream(fin);
                            ArrayList<Bookings> al2;
                            while((al2=(ArrayList)ois.readObject())!=null){
                                    Iterator<Bookings>i=al2.iterator();
                                    while(i.hasNext()){
                                            Bookings b1=i.next();
                                            if(username.equals(b1.getUsername1())){
PastBookings[c] ="----------------------------------------------------------------------------------------------------
-----------------------------------------------------------------\n"+ c +". Tickets Booked on Day :
"+b1.getBookingdate()+"\t Time : "+b1.getBookingtime()+"\n    Stadium Name :
"+b1.getSname1()+"\t"+"Date :" +b1.getDate1()+"\t"+"Cost :" +b1.getCost1()+"\n"+"
Timmings" + b1.getTime1()+"\tNumber of seats :" +b1.getN1()+"\t"+"Seats :"
+b1.getSeats1()+"\n";
                    c++;
                            }catch(Exception er){}
                            if(c>1) {
                            int c1;
                            for(c1=c-1;c1>0;c1--) {
                            Incr +=PastBookings[c1];
                            YourBookings.setText(Incr);
                                    }
                            }
                            if(c==1) {
                                    YourBookings.setText(“No Bookings Till Now !");
                            }
                    }
            });
            LastBooking.addActionListener(new ActionListener() {
                    public void actionPerformed(ActionEvent e) {
                            int c=1;
                            String lastbooking;
                            try {
                                    fin = new FileInputStream("Bookings.dat");
                                    ois = new ObjectInputStream(fin);
                                    ArrayList<Bookings> al3;
                                    while((al3=(ArrayList)ois.readObject())!=null){
```

```
                                        Iterator<Bookings>i=al3.iterator();
                                        while(i.hasNext()){
                                                Bookings b2=i.next();
                                                if(username.equals(b2.getUsername1())){
                                                        lastbooking ="
LAST BOOKING \n"+"----------------------------------------------------------------------------------------
-------------------------------------------------------------------------\n"+ c +". Tickets Booked on Day :
"+b2.getBookingdate()+"\t Time : "+b2.getBookingtime()+"\n    Stadium Name :
"+b2.getSname1()+"\t"+"Date :" +b2.getDate1()+"\t"+"Cost :" +b2.getCost1()+"\n"+"
Timmings" + b2.getTime1()+"\tNumber of seats :" +b2.getN1()+"\t"+"Seats :"
+b2.getSeats1()+"\n";
                                YourBookings.setText(lastbooking);
                                c++;
                                        }}}
}catch(Exception er){}
                                if(c==1) {
                                        YourBookings.setText("  No Bookings Till Now !");
                                }}});
        jf.setVisible(true);
        }
}
```

- Fourth Frame:

```
public int n,upperclass=1500,middleclass=1000,lowerclass=500,seatcost=0,cost=0;
DisplaySeats.addActionListener(new ActionListener() {
public void actionPerformed(ActionEvent e) {
int i=0;
if(Seata1.isSelected())
i++;
//same for all the seats!
if(i==n) {
if(upperclass == 1500) {
if(Seata1.isSelected()) {
a1="A1 ";
seatcost+=upperclass;
Seata1.setEnabled(false);
}
else
a1="";
if(Seata2.isSelected()) {
```

```
a2="A2 ";
seatcost+=upperclass;
Seata2.setEnabled(false);
}
if( middleclass == 1000) {
if(Seatb1.isSelected()) {
b1="B1 ";
seatcost+=middleclass;
Seatb1.setEnabled(false);
}
else
b1="";
if(Seatb2.isSelected()) {
b2="B2 ";
seatcost+=middleclass;
Seatb2.setEnabled(false);
}
if(lowerclass == 500) {
if(Seatc1.isSelected()) {
c1="C1 ";
seatcost+=lowerclass;
Seatc1.setEnabled(false);
}
else
c1="";
if(Seatc2.isSelected()) {
c2="C2 ";
seatcost+=lowerclass;
Seatc2.setEnabled(false);
}
else
c2="";
if(Seatc3.isSelected()) {
c3="C3 ";
Seatc3.setEnabled(false);
seatcost+=lowerclass;
if(i<n) {
JOptionPane.showMessageDialog(jf, "Selected Less number of seats "+i+"<"+n);
}
if(i>n) {
```

```
JOptionPane.showMessageDialog(jf, "Selected More number of seats "+i+">"+n);
}
}
});
public void actionPerformed(ActionEvent e) {
if(seats != null) {
if(Seata1.isSelected())
nsa1=false;
//Similar for all!
// c series lower class
if(Seatc1.isSelected())
nsc1=false;
```

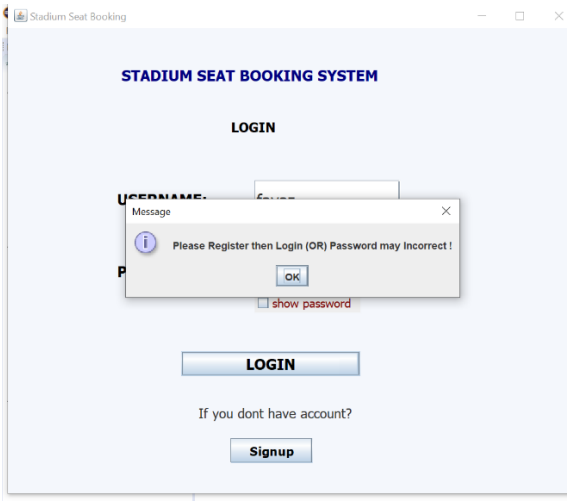
- Fifth Frame**:**

```
Bookings(String username1,String sname1,String date1,String time1,int n1,int cost1,String seats1,String bookingdate,Stringbookingtime) {

this.username1 = username1;

this.sname1 = sname1;

this.date1 = date1;

this.time1= time1;

this.n1=n1;

this.cost1=cost1;

this.seats1=seats1;

this.bookingdate=bookingdate;

this.bookingtime=bookingtime;

}

try {

if(!file1.exists()){

file1.createNewFile();

}

FileWriterfw = new FileWriter(file1,true);
```

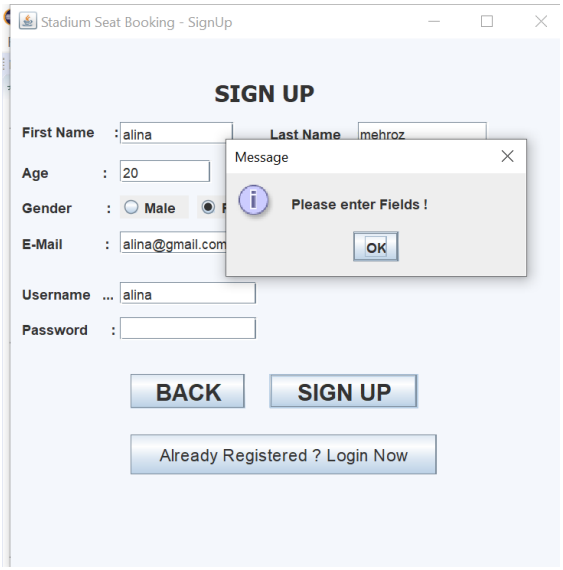
```
BufferedWriterbw = new BufferedWriter(fw);

bw.write("\n\n");

bw.write("================================================================
========");

bw.write("\n");

bw.write("Tickets Booked on Day : "+bookingdate + "     Time : "+bookingtime);

bw.write("\n");

alb.add(new Bookings(u,s,d,t,n_1,c,s1,bd,bt));

try {

File file = new File("Bookings.dat");

if(file.exists()) {

oos = new ObjectOutputStream(new FileOutputStream(file,true))
        protected void writeStreamHeader() throws IOException {

reset();

}

};}

else

oos = new ObjectOutputStream(new FileOutputStream(file));

oos.writeObject(alb);

alb.clear();

oos.close();

}catch(Exception ae){

ae.printStackTrace();

}
```

TESTING

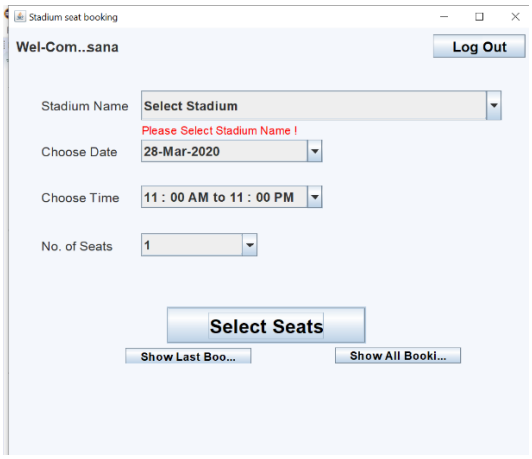
- If new customers directly logins pop up displays first to register



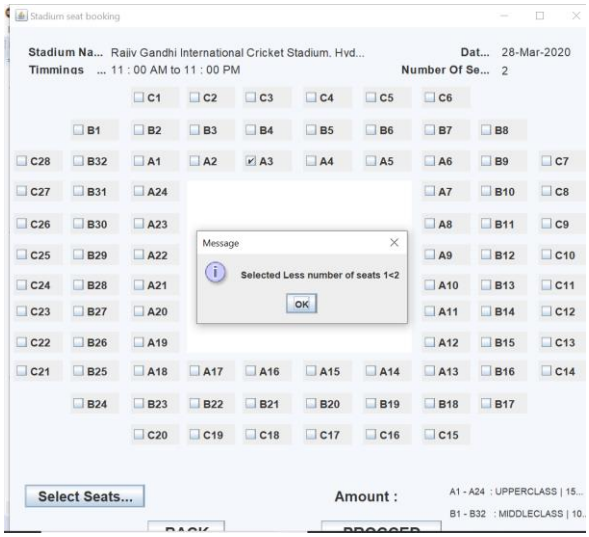
- If customer doesn’t register properly

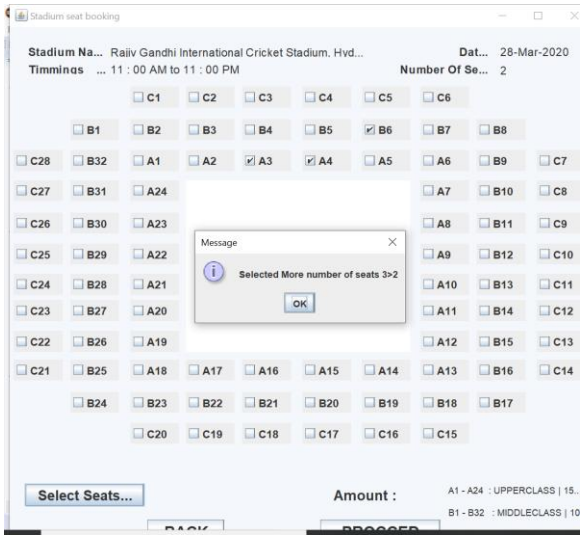


- If customer doesn’t select Stadium name



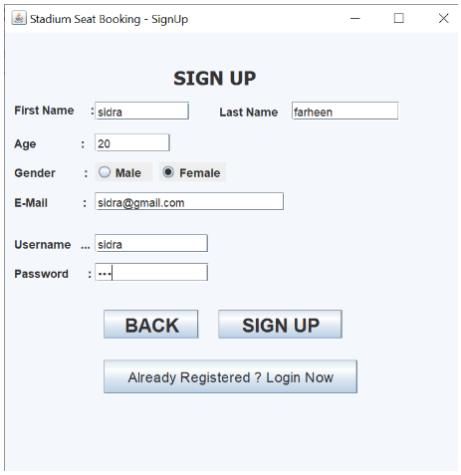
- If customer selects more or less number of seats than required

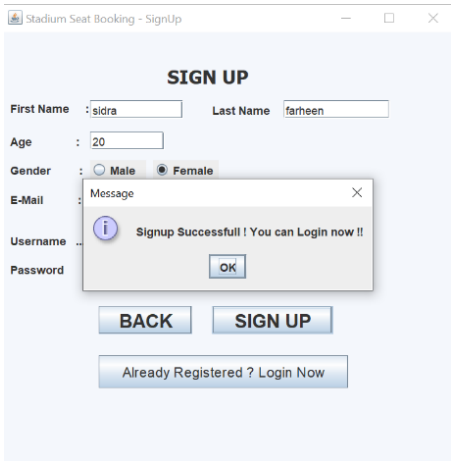




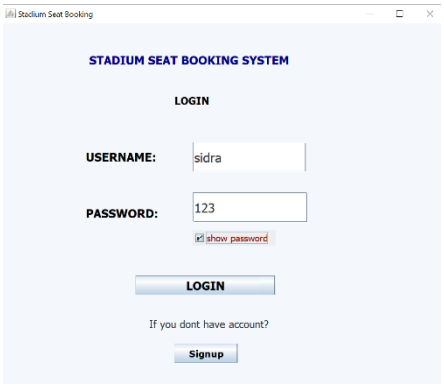
RESULTS

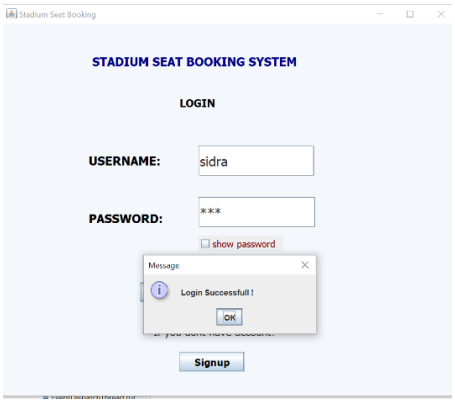
- If a customer doesn't have an account he/she needs to register:





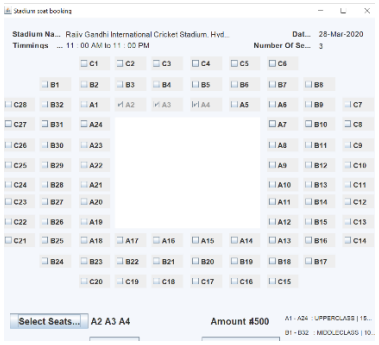
- After Registering customer needs to login:



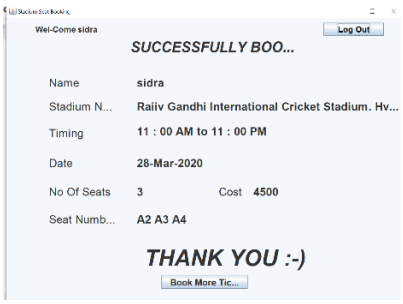


- Screen for customer to select Stadium name, time, date and number of seats:

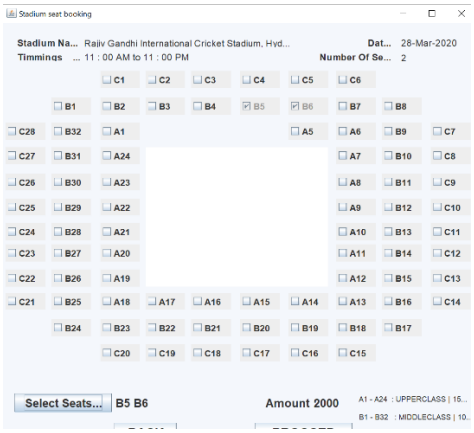
- Screen for selecting seats according to customers convenience:



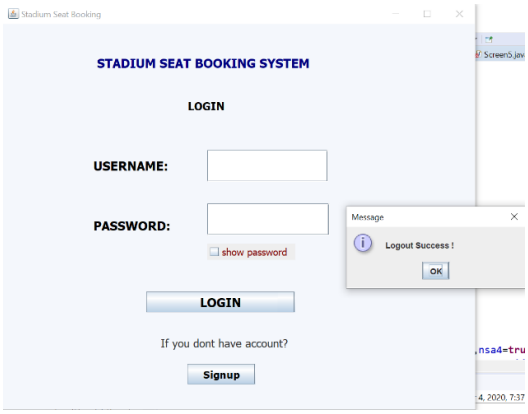
- Final screen displaying all customer details:



- If same customer wants to book seats again:



- After booking seats customer needs to logout using 5th or 3rd screen logout button:



REFERENCES

https://www.studentprojectguide.com/java-projects/stadium-seat-booking-system/

https://www.javatpoint.com/java-swing