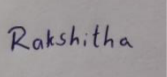


Name: N.A.R. Dilshan

Student Reference Number: 10707181

Module Code: SOFT255SL	Module Name: Software Engineering for the Internet using Java
Coursework Title: SOFT255SL T1 C1	
Deadline Date: Tuesday, 1 December 2020	Member of staff responsible for coursework: Ms. Sulari Fernando
Programme: BSc (Hons) Software Engineering	
Please note that University Academic Regulations are available under Rules and Regulations on the University website <a href="http://www.plymouth.ac.uk/studenthandbook">www.plymouth.ac.uk/studenthandbook</a> .	
<p>Group work: please list all names of all participants formally associated with this work and state whether the work was undertaken alone or as part of a team. Please note you may be required to identify individual responsibility for component parts.</p> <p>N.A.R. Dilshan - 10707181  G.T.G.L.D. Abedeera - 10707120  P. A. H. N. Mihiranga - 10707281  G.T.U. Ariyathilake - 10707133  B.E.R.R. Jayathilaka - 10707229</p> <p><b><i>We confirm that we have read and understood the Plymouth University regulations relating to Assessment Offences and that we are aware of the possible penalties for any breach of these regulations. We confirm that this is the independent work of the group.</i></b></p> <p>Signed on behalf of the group: </p>	
<p>Individual assignment: <b><i>I confirm that I have read and understood the Plymouth University regulations relating to Assessment Offences and that I am aware of the possible penalties for any breach of these regulations. I confirm that this is my own independent work.</i></b></p> <p>Signed :</p>	
<p>Use of translation software: failure to declare that translation software or a similar writing aid has been used will be treated as an assessment offence.</p> <p>I have not used translation software.</p> <p>If used, please state name of software.....</p>	
<p>Overall mark _____ %      Assessors Initials _____      Date _____</p>	

**SOFT255SL**  
**Software Engineering for the Internet using Java**

**Coursework - T1 C1**

**Security Logs System**

**Group No: 31**

N.A.R. Dilshan  
G.T.G.L.D. Abedeera  
P. A. H. N. Mihiranga  
G.T.U. Ariyathilake  
B.E.R.R. Jayathilaka

# Table of Content

Table of Content .....	0
Introduction.....	1
Analysis.....	2
Requirements.....	2
Use Case Diagram .....	2
ER Diagram.....	3
Class Diagram .....	3
Implementation .....	4
Code .....	4
GUI.....	12
Database .....	14
Conclusion .....	15
Future Improvements .....	15

# Introduction

I would like to point out some situations that any of us as students of NSBM University may face in our day today life at NSBM. And also how we deal with those situations.

First one is forgetting to bring your Student ID card. The normal procedure is that the security personnel at the entrance checks your Student ID and lets you in. But if you forgot to bring it, they ask you to write your details in the logbook at the entrance. It could be difficult as well as time wasting task for everyone especially when it is a busy morning at the entrance.

Another situation is you lost something valuable inside the university premises. What we normally do is put a message about the lost item informing others in our WhatsApp chat groups. But the security personnel are the ones who could look out for the lost item and most probably the ones who may find it. But they are not members of your chat group. So they don't know to whom it belongs to when they found something.

Some of us travel to the university by our own vehicles. But we have no proper parking facility and we have to park our vehicles roadside beside the university without proper security.

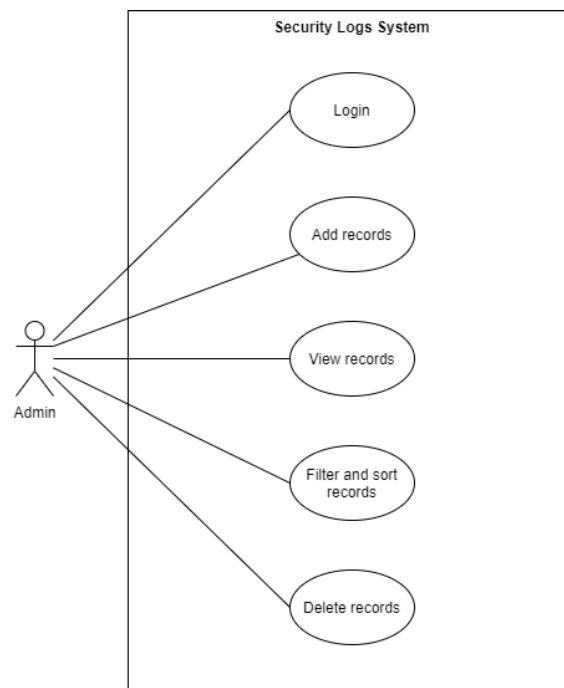
Considering situations like that we came up with a better solution to replace the current system. It is to have a security logs program to keep records of some situations like above and also to manage some processes in a better way. Both the security staff and the students could be benefited by this solution. As an example, if the university could arrange a proper space to park the vehicles and assign a security guard to the place, they can easily record the necessary details and manage the car park properly using this program. It is a great relief to the vehicle owners and also the university can charge a parking fee for their trouble since the program records all the necessary details of the vehicles in and out. So the main objective of our Security Logs Program is to manage some day today processes at university in a better way than they are right now and by that, to increase the productivity.

# Analysis

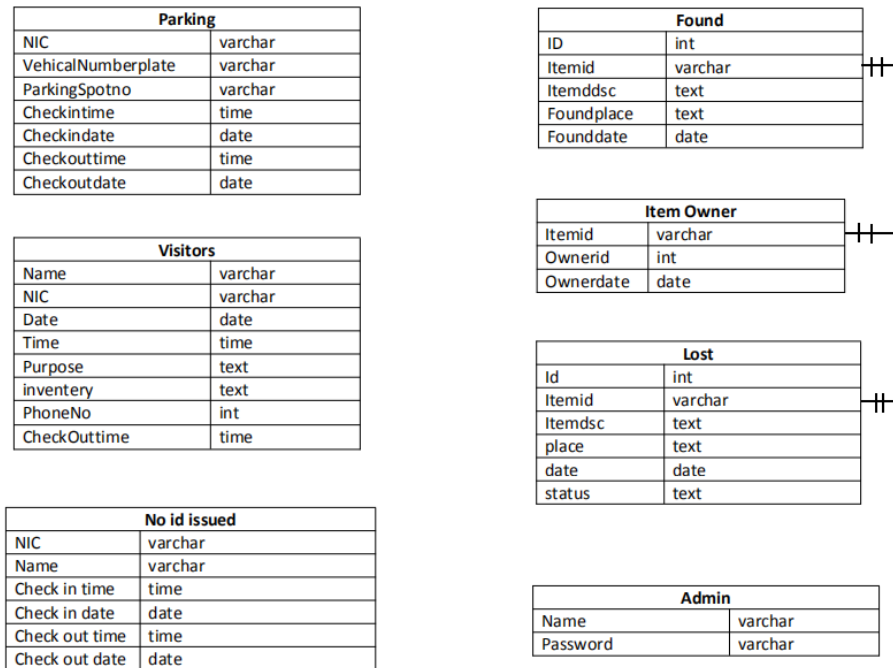
## Requirements

- Functional Requirements
  1. Ability to add records of visitors, parking, lost and found items and people who doesn't have their University ID.
  2. Ability to view all the records in a detailed form.
  3. Ability to sort and filter records.
  4. Ability to delete mistaken records.
  5. Ability to keep track of time of all entries.
  6. Only the security personal with credentials should be able to login to the system.
- Non-functional Requirements
  1. Security
  2. Simple and Functional GUI
  3. Stability of the software

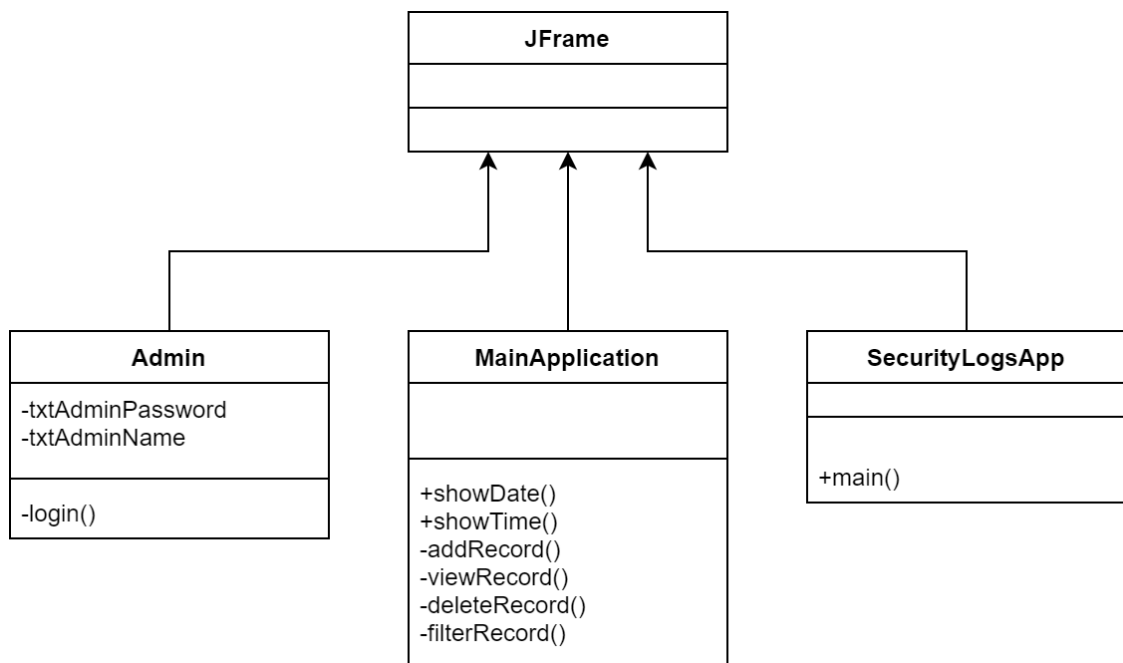
## Use Case Diagram



## ER Diagram



## Class Diagram



# Implementation

## Code

In here we have used coding concepts like encapsulation, inheritance, exception handling to build our program. We have used a tabbed pane and buttons linked to tabs for easy transition between the sections of the software.

## Admin

```
158
159 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
160     // TODO add your handling code here:
161
162
163     try {
164         Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
165
166         String DB_URL = "jdbc:sqlserver://MSI:1433;databaseName=Securitylogsgapp db";
167
168         Connection con = DriverManager.getConnection(DB_URL, "sa", "root");
169
170         String sql = "Select * from admin where Name = ? and Password=?";
171         PreparedStatement pst = con.prepareStatement(sql);
172         pst.setString(1, txtAdminName.getText());
173         pst.setString(2, txtAdminPassword.getText());
174         ResultSet rs = pst.executeQuery();
175         if (rs.next())
176         {
177             JOptionPane.showMessageDialog(null, "Username and Password matched");
178             MainApplication m1 = new MainApplication();
179             m1.setVisible(true);
180         }
181         else
182         {
183             JOptionPane.showMessageDialog(null, "Username and Password not matched");
184             txtAdminName.setText("");
185             txtAdminPassword.setText("");
186         }
187
188         con.close();
189     }
190
191     catch (Exception e) {
192         JOptionPane.showMessageDialog(null, e);
193     }
194
195
196
197
198 }
```

```
199 private void jButtonCancelActionPerformed(java.awt.event.ActionEvent evt) {
200     // TODO add your handling code here:
201     this.dispose();
202 }
203
204 private void jButtonResetActionPerformed(java.awt.event.ActionEvent evt) {
205     // TODO add your handling code here:
206     txtAdminName.setText("");
207     txtAdminPassword.setText("");
208 }
209
210 /**
211  * @param args the command line arguments
212  */
213 public static void main(String args[]) {
214     /* Set the Nimbus look and feel */
215     /* Look and feel setting code (optional) */
216
217     /* Create and display the form */
218     java.awt.EventQueue.invokeLater(new Runnable() {
219         public void run() {
220             new Admin().setVisible(true);
221         }
222     });
223 }
224
225 // Variables declaration - do not modify
226 private javax.swing.JButton jButton1;
227 private javax.swing.JButton jButtonCancel;
228 private javax.swing.JButton jButtonReset;
229 private javax.swing.JDialog jDialog1;
230 private javax.swing.JLabel jLabel1;
231 private javax.swing.JLabel jLabel2;
232 private javax.swing.JLabel jLabel3;
233 private javax.swing.JLabel jLabel4;
234 private javax.swing.JLabel jLabel5;
235 private javax.swing.JPanel jPanel1;
236 private javax.swing.JPanel jPanel2;
237 private javax.swing.JTextField txtAdminName;
238 private javax.swing.JPasswordField txtAdminPassword;
239 // End of variables declaration
```

```

1  *
2  * To change this license header, choose License Headers in Project Properties.
3  * To change this template file, choose Tools | Templates
4  * and open the template in the editor.
5  */
6  package securitylogsapp;
7
8  import java.sql.*;
9  import java.util.logging.Level;
10 import java.util.logging.Logger;
11 import javax.swing.JOptionPane;
12
13 /**
14  *
15  * @author PC
16  */
17 public class Admin extends javax.swing.JFrame {
18
19     /**
20      * Creates new form Admin
21      */
22     public Admin() {
23         initComponents();
24     }
25
26     /**
27      * This method is called from within the constructor to initialize the form.
28      * WARNING: Do NOT modify this code. The content of this method is always
29      * regenerated by the Form Editor.
30      */
31     @SuppressWarnings("unchecked")
32     // Generated Code

```

## Main Application

```

2278 private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
2279     // TODO add your handling code here:
2280     try {
2281         Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
2282
2283         String DB_URL = "jdbc:sqlserver://MSI:1433;databaseName=Securitylogsapp db";
2284
2285         Connection con = DriverManager.getConnection(DB_URL, "sa", "root");
2286
2287         Statement st = con.createStatement();
2288         String sql = "SELECT f.id,f.itemid,l.itemid,l.place,l.date,f.foundplace,f.founddate,c.ownerid,c.owndate From fact f INNER JOIN found f ON f.itemid=f.itemid INNER
2289         ResultSet rs = st.executeQuery(sql);
2290         while(rs.next())
2291         {
2292             String id = rs.getString("id");
2293             String itemid = rs.getString("itemid");
2294             String itemdesc = rs.getString("itemdesc");
2295             String lplace = rs.getString("place");
2296             String ldate = rs.getString("date");
2297
2298             String fplace = rs.getString("foundplace");
2299             String fdate = rs.getString("founddate");
2300             String ownerid=rs.getString("ownerid");
2301             String owndate=rs.getString("owndate");
2302
2303             String ttdData[] = {id,itemid,itemdesc,lplace,ldate,fplace,fdate,ownerid,owndate};
2304             DefaultTableModel tblModel = (DefaultTableModel)tblFoundTable1.getModel();
2305
2306             tblModel.addRow(ttdData);
2307
2308             con.close();
2309         }
2310     } catch (Exception e) {
2311         JOptionPane.showMessageDialog(null,e);
2312     }
2313 }

```

```

2855 private void jButton26ActionPerformed(java.awt.event.ActionEvent evt) {
2856     // TODO add your handling code here:
2857     try {
2858         Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
2859
2860         String DB_URL = "jdbc:sqlserver://MSI:1433;databaseName=Securitylogsapp db";
2861
2862         Connection con = DriverManager.getConnection(DB_URL, "sa", "root");
2863
2864         String sql = "DELETE from noiddisswed WHERE NIC=?";
2865         PreparedStatement pst = con.prepareStatement(sql);
2866         pst.setString(1, txtiddelateitenic.getText());
2867
2868         pst.executeUpdate();
2869         JOptionPane.showMessageDialog(null, "Record Deleted Successfully");
2870         txtiddelateitenic.setText("");
2871
2872         con.close();
2873     } catch (Exception e) {
2874         JOptionPane.showMessageDialog(null,e);
2875     }
2876 }

```



```

9  import java.util.Date;
10 import java.sql.Timestamp;
11 import java.sql.Date;
12 import java.sql.Connection;
13 import java.sql.DriverManager;
14 import java.sql.PreparedStatement;
15 import java.sql.ResultSet;
16 import java.sql.Statement;
17 import java.sql.TimeStamp;
18
19 import java.text.SimpleDateFormat;
20 import java.time.LocalDate;
21 import java.time.format.DateTimeFormatter;
22 import java.time.temporal.ChronoUnit;
23
24 import javax.swing.JOptionPane;
25 import javax.swing.JFrame;
26 import javax.swing.table.DefaultTableModel;
27
28 /**
29  *
30  * @author nmr
31  */
32 public class MainApplication extends javax.swing.JFrame {
33
34     /**
35      * Creates new form MainApplication
36      */
37     public MainApplication() {
38         initComponents();
39         showDate();
40         showTime();
41     }
42
43     public MainApplication() {
44         initComponents();
45         showDate();
46         showTime();
47     }
48
49     void showDate() {
50         // Determine the date
51         DateTimeFormatter dtf = DateTimeFormatter.ofPattern("yyyy/MM/dd");
52         LocalDate now = LocalDate.now();
53
54         //Visitor Checkin Date
55         jLabelVisitorCheckInDate.setText(dtf.format(now));
56
57         //No ID Issued Checkin, CheckOut Date
58         jLabelNoIssuedCheckInDate.setText(dtf.format(now));
59         jLabelNoIssuedCheckOutDate.setText(dtf.format(now));
60
61         //Parking Checkin, Out Date
62         jLabelParkingCheckInDate.setText(dtf.format(now));
63         jLabelParkingCheckOutDate.setText(dtf.format(now));
64     }
65
66     void showTime() {
67         // Determine the time
68         DateTimeFormatter dtf = DateTimeFormatter.ofPattern("HH:mm a");
69         LocalDate now = LocalDate.now();
70         //Visitor Checkin, CheckOut Time
71         jLabelVisitorCheckInTime.setText(dtf.format(now));
72         jLabelVisitorCheckOutTime.setText(dtf.format(now));
73
74         //No ID Issued Checkin, CheckOut Time
75         jLabelNoIssuedCheckInTime.setText(dtf.format(now));
76         jLabelNoIssuedCheckOutTime.setText(dtf.format(now));
77
78         //Parking Checkin, Out Time
79         jLabelParkingCheckInTime.setText(dtf.format(now));
80         jLabelParkingCheckOutTime.setText(dtf.format(now));
81     }
82 }

```

```

1011 private void jButtonActionPerformed(java.awt.event.ActionEvent evt) {
1012
1013     try {
1014         Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
1015
1016         String DB_URL = "jdbc:sqlserver://MSI1433;DatabaseName=SecurityLogapp db";
1017
1018         Connection con = DriverManager.getConnection(DB_URL, "sa", "root");
1019
1020         Date date = new Date();
1021         java.sql.Date sqlDate = new java.sql.Date(date.getTime());
1022
1023         String sql = "insert into visitors(Name,NIC,Date,Time,Purpose,Inventory,PhoneNo) Values(?, ?, ?, ?, ?, ?, ?)";
1024         PreparedStatement pst = con.prepareStatement(sql);
1025         pst.setString(1, txtName.getText());
1026         pst.setString(2, txtNIC.getText());
1027         pst.setDate(3, sqlDate);
1028         pst.setTimeStamp(4, new Timestamp(System.currentTimeMillis()));
1029
1030         pst.setString(5, txtPurpose.getText());
1031         pst.setString(6, txtInventory.getText());
1032         pst.setString(7, txtPhoneNo.getText());
1033
1034         pst.executeUpdate();
1035         JOptionPane.showMessageDialog(null, "Record Inserted Successfully");
1036         txtName.setText("");
1037         txtNIC.setText("");
1038         txtPurpose.setText("");
1039         txtInventory.setText("");
1040         txtPhoneNo.setText("");
1041
1042         con.close();
1043     }
1044 }

```

```

1045 private void jButtonActionPerformed(java.awt.event.ActionEvent evt) {
1046     // TODO add your handling code here:
1047
1048     try {
1049         Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
1050
1051         String DB_URL = "jdbc:sqlserver://MSI1433;DatabaseName=SecurityLogapp db";
1052
1053         Connection con = DriverManager.getConnection(DB_URL, "sa", "root");
1054
1055         Statement st = con.createStatement();
1056         String sql = "select * from visitors";
1057         ResultSet rs = st.executeQuery(sql);
1058         while(rs.next())
1059         {
1060             String name = rs.getString("Name");
1061             String nic = rs.getString("NIC");
1062             String date = rs.getString("Date");
1063             String time = rs.getString("Time");
1064             String purpose = rs.getString("Purpose");
1065             String inv = rs.getString("Inventory");
1066             String phone = rs.getString("PhoneNo");
1067             String outtime = rs.getString("CheckOutTime");
1068
1069             String thData[] = {name,nic,date,time,purpose,inv,phone,outtime};
1070             DefaultTableModel tblModel = (DefaultTableModel) vassonTable1.getModel();
1071
1072             tblModel.addRow(thData);
1073         }
1074         con.close();
1075     }
1076
1077     catch (Exception e) {
1078         JOptionPane.showMessageDialog(null,e);
1079     }
1080 }

```

```

1841 private void btnOnTheSpotActionPerformed(java.awt.event.ActionEvent evt) {
1842     // TODO add your handling code here:
1843     try {
1844         Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
1845
1846         String DB_URL = "jdbc:sqlserver://M11:1433;databaseName=SecurityLogApp db";
1847
1848         Connection con = DriverManager.getConnection(DB_URL, "sa", "root");
1849
1850         Statement st = con.createStatement();
1851         String sql = "Select * From visitors TABLE BY Time DESC";
1852         ResultSet rs = st.executeQuery(sql);
1853         while(rs.next())
1854         {
1855             String name = rs.getString("Name");
1856             String nic = rs.getString("NIC");
1857             String date = rs.getString("Date");
1858             String time = rs.getString("Time");
1859             String purpose = rs.getString("Purpose");
1860             String inv = rs.getString("Inventory");
1861             String phone = rs.getString("PhoneNo");
1862             String outtime = rs.getString("CheckOutTime");
1863
1864             String chData[] = {name,nic,date,time,purpose,inv,phone,outtime};
1865             DefaultTableModel tblModel = (DefaultTableModel) visitorsTable1.getModel();
1866
1867             tblModel.addRow(chData);
1868         }
1869         con.close();
1870     }
1871     catch (Exception e) {
1872         JOptionPane.showMessageDialog(null,e);
1873     }
1874 }

```

```

1880 private void SubmitActionPerformed(java.awt.event.ActionEvent evt) {
1881     // TODO add your handling code here:
1882
1883     try {
1884         Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
1885
1886         String DB_URL = "jdbc:sqlserver://M11:1433;databaseName=SecurityLogApp db";
1887
1888         Connection con = DriverManager.getConnection(DB_URL, "sa", "root");
1889
1890         String sql = "UPDATE visitors SET CheckOutTime=? WHERE NIC=?";
1891         PreparedStatement pst = con.prepareStatement(sql);
1892         pst.setTimestamp(1, new Timestamp(System.currentTimeMillis()));
1893         pst.setString(2, txtCheckOutNIC.getText());
1894
1895         pst.executeUpdate();
1896         JOptionPane.showMessageDialog(null, "Record Inserted Successfully");
1897         txtCheckOutNIC.setText("");
1898
1899         con.close();
1900     }
1901     catch (Exception e) {
1902         JOptionPane.showMessageDialog(null,e);
1903     }
1904 }

```

```

2025 private void getAttendanceDataFromDatabase() { java.sql.event.ActionEvent evt) {
2026 // TODO add your handling code here:
2027 try {
2028     Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
2029
2030     String DB_URL = "jdbc:sqlserver://192.168.1.100:1433;databaseName=AttendanceSystem;encrypt=false";
2031
2032     Connection conn = DriverManager.getConnection(DB_URL, "sa", "root");
2033
2034
2035
2036
2037     Statement st = conn.createStatement();
2038     String sql = "select * from visitors where CheckOutTime IS NOT NULL";
2039     ResultSet rs = st.executeQuery(sql);
2040     while(rs.next())
2041     {
2042         String name = rs.getString("Name");
2043         String nic = rs.getString("NIC");
2044         String date = rs.getString("Date");
2045         String time = rs.getString("Time");
2046         String purpose = rs.getString("Purpose");
2047         String time = rs.getString("Time");
2048         String phone = rs.getString("Phone");
2049         String outtime = rs.getString("CheckOutTime");
2050
2051         String data[] = {name,nic,nic,time,purpose,phone,outtime};
2052         DefaultTableModel tblModel = (DefaultTableModel) visitorsTable.getModel();
2053         tblModel.addRow(data);
2054     }
2055
2056     conn.close();
2057
2058 }
2059
2060 catch (Exception e) {
2061     JOptionPane.showMessageDialog(null,e);
2062 }
2063
2064
2065
2066
2067 private void getAttendanceDataFromDatabase() { java.sql.event.ActionEvent evt) {
2068 // TODO add your handling code here:
2069 // TODO add your handling code here:
2070
2071 }
2072
2073 private void getAttendanceDataFromDatabase() { java.sql.event.ActionEvent evt) {
2074 // TODO add your handling code here:
2075
2076 try {
2077     Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
2078
2079     String DB_URL = "jdbc:sqlserver://192.168.1.100:1433;databaseName=AttendanceSystem;encrypt=false";
2080
2081     Connection conn = DriverManager.getConnection(DB_URL, "sa", "root");
2082
2083     java.sql.Statement st = conn.createStatement();
2084     java.sql.ResultSet rs = st.executeQuery("select * from visitors");
2085     String departure = rs.getString("Departure");
2086
2087     String sql = "insert into visitors (Name, NIC, Date, Time, Purpose, Phone, OutTime) values ('" + rs.getString(1) + "','" + rs.getString(2) + "','" + rs.getString(3) + "','" + rs.getString(4) + "','" + rs.getString(5) + "','" + rs.getString(6) + "','" + rs.getString(7) + "')";
2088     PreparedStatement pstmt = conn.prepareStatement(sql);
2089     pstmt.setString(1, rs.getString(1));
2090     pstmt.setString(2, rs.getString(2));
2091     pstmt.setString(3, rs.getString(3));
2092     pstmt.setString(4, rs.getString(4));
2093     pstmt.setString(5, rs.getString(5));
2094     pstmt.setString(6, rs.getString(6));
2095     pstmt.setString(7, rs.getString(7));
2096
2097     pstmt.executeUpdate();
2098
2099     JOptionPane.showMessageDialog(null, "Record inserted successfully");
2100     visitorsTable.setModel(rs);
2101     visitorsTable.repaint();
2102     visitorsTable.validate();
2103     visitorsTable.setVisible(true);
2104 }

```

```

2080 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
2081     // TODO add your handling code here:
2082     try {
2083         Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
2084
2085         String DB_URL = "jdbc:sqlserver://MSI:1433;databaseName=SecurityLogapp_db";
2086
2087         Connection con = DriverManager.getConnection(DB_URL, "sa", "root");
2088
2089         Statement st = con.createStatement();
2090         String sql = "Select Name, NIC, Date, Time, Purpose, Inventory, PhoneNo from Visitors where CheckOutTime IS Null";
2091         ResultSet rs = st.executeQuery(sql);
2092         while(rs.next())
2093         {
2094             String name = rs.getString("Name");
2095             String nic = rs.getString("NIC");
2096             String date = rs.getString("Date");
2097             String time = rs.getString("Time");
2098             String purpose = rs.getString("Purpose");
2099             String inv = rs.getString("Inventory");
2100             String phone = rs.getString("PhoneNo");
2101
2102             String thData[] = {name, nic, date, time, purpose, inv, phone};
2103             DefaultTableModel tblModel = (DefaultTableModel)jVisitorTable1.getModel();
2104
2105             tblModel.addRow(thData);
2106         }
2107
2108         con.close();
2109     }
2110
2111     catch (Exception e) {
2112         JOptionPane.showMessageDialog(null, e);
2113     }
2114 }

```

```

2115
2116     catch (Exception e) {
2117         JOptionPane.showMessageDialog(null, e);
2118     }
2119 }
2120
2121 private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
2122     // TODO add your handling code here:
2123     try {
2124         Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
2125
2126         String DB_URL = "jdbc:sqlserver://MSI:1433;databaseName=SecurityLogapp_db";
2127
2128         Connection con = DriverManager.getConnection(DB_URL, "sa", "root");
2129
2130         Date departDate = (Date) jowdate.getDate();
2131         SimpleDateFormat oDateFormat = new SimpleDateFormat("yyyy-MM-dd");
2132         String departDateStr = oDateFormat.format(departDate);
2133
2134         String sql = "Insert into tbvisitor (itemid, ownerid, vendate) Values (?,?,?)";
2135         PreparedStatement pst = con.prepareStatement(sql);
2136         pst.setString(1, txtowitemid.getText());
2137         pst.setString(2, txtownerid.getText());
2138         pst.setString(3, departDateStr);
2139
2140         pst.executeUpdate();
2141         JOptionPane.showMessageDialog(null, "Record Inserted Successfully");
2142         txtowitemid.setText("");
2143         txtownerid.setText("");
2144
2145         con.close();
2146     }

```

```

2248 private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
2249     // TODO add your handling code here:
2250
2251     jTableFoundTable.setModel(new DefaultTableModel(new String[]{"ID", "ItemID", "ItemDesc", "Place", "Date", "FoundPlace", "FoundDate", "OwnerID"},
2252     {
2253     }
2254     ));
2255
2256 private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
2257     // TODO add your handling code here:
2258     try {
2259         Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
2260
2261         String DB_URL = "jdbc:sqlserver://MTI:1433;databaseName=SecurityLogApp.db";
2262
2263         Connection con = DriverManager.getConnection(DB_URL, "sa", "root");
2264
2265         Statement st = con.createStatement();
2266         String sql = "select i.id,i.itemid,i.itemdesc,i.place,i.date,f.foundplace,f.founddate from lost l,found f where l.itemid=f.itemid";
2267         ResultSet rs = st.executeQuery(sql);
2268         while(rs.next())
2269         {
2270             String id = rs.getString("id");
2271             String itemid = rs.getString("itemid");
2272             String itemdesc = rs.getString("itemdesc");
2273             String place = rs.getString("place");
2274             String date = rs.getString("date");
2275             String fplace = rs.getString("foundplace");
2276             String fdate = rs.getString("founddate");
2277
2278             String tbData[] = {id,itemid,itemdesc,place,date,fplace,fdate};
2279             DefaultTableModel tblModel = (DefaultTableModel)jTableFoundTable.getModel();
2280
2281             tblModel.addRow(tbData);
2282         }
2283         con.close();
2284     }
2285 }

```

## Main Method

```

2286 private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
2287     // TODO add your handling code here:
2288     try {
2289         Class.forName("com.microsoft.sqlserver.jdbc.SQLServerDriver");
2290
2291         String DB_URL = "jdbc:sqlserver://MTI:1433;databaseName=SecurityLogApp.db";
2292
2293         Connection con = DriverManager.getConnection(DB_URL, "sa", "root");
2294
2295         Statement st = con.createStatement();
2296         String sql = "select * from item";
2297         ResultSet rs = st.executeQuery(sql);
2298         while(rs.next())
2299         {
2300             String id = rs.getString("id");
2301             String itemid = rs.getString("itemid");
2302             String itemdesc = rs.getString("itemdesc");
2303             String place = rs.getString("place");
2304             String date = rs.getString("date");
2305
2306             String tbData[] = {id,itemid,itemdesc,place,date};
2307             DefaultTableModel tblModel = (DefaultTableModel)jTableFoundTable.getModel();
2308
2309             tblModel.addRow(tbData);
2310         }
2311         con.close();
2312     }
2313     catch (Exception e) {
2314         JOptionPane.showMessageDialog(null,e);
2315     }
2316 }

```

## GUI

The image displays two screenshots of a web application titled "Security Logs System".

The top screenshot shows the "ADMIN LOGIN" interface. It features a large, glowing blue padlock icon on the right. The login form includes fields for "Name:" and "Password:", a "Login" button, a "Reset" button, and a "Cancel" button. The background is dark with a circuit-like pattern.

The bottom screenshot shows the main dashboard. It has a blue header with the title "Security Logs System". On the left is a sidebar with navigation links: "Visitors", "Parking", "Lost and Found", and "No ID Issued". The main content area is divided into three sections:

- Check In:** Includes fields for "NIC", "Vehicle Number PLate", "Parking Spot No", "Time" (12:46 PM), and "Date" (2020/12/01). A "Park" button is at the bottom.
- Pay and Depart:** Includes fields for "NIC", "Time" (12:46 PM), "Date" (2020/12/01), and "Parking Fee" (Rs. 50). A "Pay" button is at the bottom.
- Parking Log:** Includes a "Search By:" section with buttons for "Parked", "By Spot Number", "Took off", and "Clear". Below this is a "Delete Records:" section with a text input and a "Delete" button. At the bottom is a table with the following headers: "NIC", "VehicleNumb...", "ParkingSpotNo", "CheckinTime", "CheckinDate", "CheckOutTime", "CheckOutDate", and "PaymentFee". The table body is currently empty.



Security Logs System

Visitors

Parking

Lost and Found

No ID Issued

Check In

Name

NIC Number

Date2020/12/01

Time12:46 PM

Purpose

Inventory

Phone No

Submit

Check Out

NIC Number

Time12:46 PM

Submit

Entries

Search by :

All Entries

Arrived

Departed

Chronological

Clear

Delete Records:

Delete

Name	NIC	Date	Time	Purpose	Inventory	PhoneNo	CheckOutTime
------	-----	------	------	---------	-----------	---------	--------------

Security Logs System

Visitors

Parking

Lost and Found

No ID Issued

Check In

NIC Number

Name

Date2020/12/01

Time12:46 PM

Submit

Check Out

NIC Number

Date2020/12/01

Time12:46 PM

Submit

ID Issued Logs

Search By:

In Campus

Left

Chronological

Clear

Delete Records:

Delete

NIC	Name	CheckInDate	CheckInTime	CheckOutDate	CheckOutTime
-----	------	-------------	-------------	--------------	--------------

13



Security Logs System

Visitors

Parking

Lost and Found

No ID Issued

Report lost item

VID

Item ID

Item Description

lost place

Lost Date

Add lost item

Found Items

VID

Item ID

Item Description

Found Place

Found Date

Report found item

Owner found

Item ID

Owner VID

Return Date

Found owner

Lost and Found logs

Search By:

Lost

Found

Resolved

Clear

Delete Records:

Delete

UID	ItemID	ItemDesc	LostPlace	LostDate	FoundPla	FoundDate	OwnerID	OwnDate
-----	--------	----------	-----------	----------	----------	-----------	---------	---------

## Database

```

graph TD
    subgraph "dbo.parking"
        NIC1[NIC (PK, varchar(12), not null)]
        VehicleNumberPlate[VehicleNumberPlate (varchar(20), null)]
        ParkingSpotNo[ParkingSpotNo (varchar(10), null)]
        CheckInTime[CheckInTime (time(7), null)]
        CheckInDate[CheckInDate (date, null)]
        CheckOutTime[CheckOutTime (time(7), null)]
        CheckOutDate[CheckOutDate (date, null)]
        PaymentFee[PaymentFee (varchar(10), null)]
    end

    subgraph "dbo.itemowner"
        itemid[itemid (varchar(10), null)]
        ownerid[ownerid (int, null)]
        owndate[owndate (date, null)]
    end

    subgraph "dbo.noidissued"
        NIC2[NIC (PK, varchar(10), not null)]
        Name1[Name (varchar(20), null)]
        CheckInDate1[CheckInDate (date, null)]
        CheckInTime1[CheckInTime (time(7), null)]
        CheckOutDate1[CheckOutDate (date, null)]
        CheckOutTime1[CheckOutTime (time(7), null)]
    end

    subgraph "dbo.visitors"
        Name2[Name (varchar(20), null)]
        NIC3[NIC (varchar(12), null)]
        Date[date (date, null)]
        Time[Time (time(7), null)]
        Purpose[Purpose (text, null)]
        Inventory[Inventory (text, null)]
        PhoneNo[PhoneNo (int, null)]
        CheckOutTime2[CheckOutTime (time(7), null)]
    end

    subgraph "dbo.admin"
        Name3[Name (varchar(20), null)]
        Password[Password (varchar(15), null)]
    end

    subgraph "dbo.lost"
        id[id (int, null)]
        itemid1[itemid (varchar(10), null)]
        itemdsc[itemdsc (text, null)]
        place[place (text, null)]
        date1[date (date, null)]
        status[status (text, null)]
    end

    subgraph "dbo.found"
        id1[id (int, null)]
        itemid2[itemid (varchar(10), null)]
        itemdsc1[itemdsc (text, null)]
        foundplace[foundplace (text, null)]
        founddate[founddate (date, null)]
    end
  
```

The diagram illustrates the database schema for a car rental system. It consists of several tables, each with its own set of columns and data types:

- dbo.parking**: Contains columns for vehicle identification (NIC, VehicleNumberPlate), parking location (ParkingSpotNo), and rental details (CheckInTime, CheckInDate, CheckOutTime, CheckOutDate, PaymentFee).
- dbo.itemowner**: Contains columns for item identification (itemid), owner information (ownerid), and ownership date (owndate).
- dbo.noidissued**: Contains columns for item identification (NIC, Name), rental dates (CheckInDate, CheckOutDate), and rental times (CheckInTime, CheckOutTime).
- dbo.visitors**: Contains columns for visitor information (Name, NIC, Date, Time, Purpose, Inventory, PhoneNo) and rental duration (CheckOutTime).
- dbo.admin**: Contains columns for administrator information (Name, Password).
- dbo.lost**: Contains columns for lost item details (id, itemid, itemdsc, place, date, status).
- dbo.found**: Contains columns for found item details (id, itemid, itemdsc, foundplace, founddate).

## **Conclusion**

The users are able to easily keep track of the records of the visitors, parking details, lost and found and people who visits the university without a ID card. Unlike the traditional system, they can filter records and sort records when they need. Users must be given credentials to login the system by a higher position.

### **Future Improvements,**

- The software can be connected to the NSBM main database so that the relevant student and staff information can be viewed through our system.
- Include an alert system for lost and found section to notify people about lost and found items.
- Include a card payment integration for the parking section.