**Final Documentation of the project File**

**Course Name: ITMD 411 (Intermediate programming)**

**Name: Muhammad Hamza Mobeen**

**Source Code:**

**(Dao.java)**

**package** javaapplication1;

**import** javax.swing.\*;

**import** java.io.BufferedReader;

**import** java.io.FileReader;

**import** java.sql.\*;

**import** java.util.ArrayList;

**import** java.util.Arrays;

**import** java.util.List;

**public** **class** Dao {

// instance fields

**static** Connection *connect* = **null**;

Statement statement = **null**;

// constructor

**public** Dao() {

}

**public** Connection getConnection() {

// Setup the connection with the DB

**try** {

*connect* = DriverManager.*getConnection*("jdbc:mysql://www.papademas.net:3307/tickets?autoReconnect=true&useSSL=false" + "&user=fp411&password=411");

} **catch** (SQLException e) {

e.printStackTrace();

}

**return** *connect*;

}

**public** **void** createTables() {

**final** String createTicketsTable = "CREATE TABLE hamza\_jpapa\_tickets (tid INT PRIMARY KEY AUTO\_INCREMENT,user VARCHAR(255) NOT NULL,ticket\_desc VARCHAR(400),start\_date DATE,end\_date DATE);";

**final** String createUsersTable = "CREATE TABLE hamza\_jpapa\_users(uid INT AUTO\_INCREMENT PRIMARY KEY, uname VARCHAR(30), upass VARCHAR(30), admin int)";

**try** {

statement = getConnection().createStatement();

**if** (statement != **null**) {

statement.executeUpdate(createTicketsTable);

statement.executeUpdate(createUsersTable);

System.***out***.println("Created tables in the given database...");

statement.close();

*connect*.close();

} **else** {

System.***out***.println("Failed to establish a connection.");

}

} **catch** (Exception e) {

System.***out***.println(e.getMessage());

}

addUsers();

}

**public** **void** addUsers() {

String sql;

Statement statement;

BufferedReader br;

List<List<String>> array = **new** ArrayList<>();

**try** {

br = **new** BufferedReader(**new** FileReader(**new** String("./userlist.csv")));

String line;

**while** ((line = br.readLine()) != **null**) {

array.add(Arrays.*asList*(line.split(",")));

}

} **catch** (Exception e) {

System.***out***.println("There was a problem loading the file");

}

**try** {

statement = getConnection().createStatement();

**for** (List<String> rowData : array) {

sql = "INSERT INTO hamza\_jpapa\_users(uname,upass,admin) " + "VALUES('" + rowData.get(0) + "'," + " '"

+ rowData.get(1) + "','" + rowData.get(2) + "');";

statement.executeUpdate(sql);

}

System.***out***.println("Inserts completed in the given database...");

statement.close();

} **catch** (Exception e) {

System.***out***.println(e.getMessage());

}

}

**public** **int** insertRecord(String ticketName, String ticketDesc, Date startDate, Date endDate) {

**int** id = 0;

**try** (Connection connection = getConnection();

PreparedStatement preparedStatement = connection.prepareStatement(

"INSERT INTO hamza\_jpapa\_tickets (user, ticket\_desc, start\_date, end\_date) VALUES (?, ?, ?, ?)",

PreparedStatement.***RETURN\_GENERATED\_KEYS***)) {

preparedStatement.setString(1, ticketName);

preparedStatement.setString(2, ticketDesc);

preparedStatement.setDate(3, startDate);

preparedStatement.setDate(4, endDate);

**int** affectedRows = preparedStatement.executeUpdate();

**if** (affectedRows > 0) {

**try** (ResultSet generatedKeys = preparedStatement.getGeneratedKeys()) {

**if** (generatedKeys.next()) {

id = generatedKeys.getInt(1);

} **else** {

**throw** **new** SQLException("Failed to retrieve auto-generated key.");

}

}

}

} **catch** (SQLException e) {

e.printStackTrace();

}

**return** id;

}

**public** ResultSet readRecords() {

ResultSet results = **null**;

**try** {

statement = *connect*.createStatement();

results = statement.executeQuery("SELECT \* FROM hamza\_jpapa\_tickets");

} **catch** (SQLException e1) {

e1.printStackTrace();

}

**return** results;

}

**public** **void** updateRecords(**int** ticketId, String newTicketDesc, Date newStartDate, Date newEndDate) {

**try** {

String updateQuery = "UPDATE hamza\_jpapa\_tickets SET ticket\_desc = ?, start\_date = ?, end\_date = ? WHERE tid = ?";

**try** (PreparedStatement preparedStatement = *connect*.prepareStatement(updateQuery)) {

preparedStatement.setString(1, newTicketDesc);

preparedStatement.setDate(2, newStartDate);

preparedStatement.setDate(3, newEndDate);

preparedStatement.setInt(4, ticketId);

**int** affectedRows = preparedStatement.executeUpdate();

**if** (affectedRows > 0) {

System.***out***.println("Record updated successfully!");

JOptionPane.*showMessageDialog*(**null**, "SUCCESS FULL", "Record updated successfully!", JOptionPane.***INFORMATION\_MESSAGE***);

} **else** {

System.***out***.println("No records were updated.");

JOptionPane.*showMessageDialog*(**null**, "FAIL", "No records were updated.", JOptionPane.***INFORMATION\_MESSAGE***);

}

}

} **catch** (SQLException e) {

e.printStackTrace();

}

}

**public** **void** deleteRecords(**int** ticketId) {

**try** {

String deleteQuery = "DELETE FROM hamza\_jpapa\_tickets WHERE tid = ?";

**try** (PreparedStatement preparedStatement = *connect*.prepareStatement(deleteQuery)) {

preparedStatement.setInt(1, ticketId);

**int** affectedRows = preparedStatement.executeUpdate();

**if** (affectedRows > 0) {

System.***out***.println("Record deleted successfully!");

JOptionPane.*showMessageDialog*(**null**, "SUCCESSS FULL", "Record deleted successfully!", JOptionPane.***INFORMATION\_MESSAGE***);

} **else** {

System.***out***.println("No records were deleted.");

JOptionPane.*showMessageDialog*(**null**, "FAIL", "No records were deleted.", JOptionPane.***INFORMATION\_MESSAGE***);

}

}

} **catch** (SQLException e) {

e.printStackTrace();

}

}

// New method to toggle the status of a ticket

**public** **void** toggleTicketStatus(**int** ticketId) {

**try** {

String toggleQuery = "UPDATE hamza\_jpapa\_tickets SET status = NOT status WHERE tid = ?";

**try** (PreparedStatement preparedStatement = *connect*.prepareStatement(toggleQuery)) {

preparedStatement.setInt(1, ticketId);

**int** affectedRows = preparedStatement.executeUpdate();

**if** (affectedRows > 0) {

System.***out***.println("Ticket status toggled successfully!");

JOptionPane.*showMessageDialog*(**null**, "SUCCESSS FULL", "Ticket status toggled successfully!", JOptionPane.***INFORMATION\_MESSAGE***);

} **else** {

System.***out***.println("No ticket status toggled.");

JOptionPane.*showMessageDialog*(**null**, "FAIL", "No ticket status toggled.", JOptionPane.***INFORMATION\_MESSAGE***);

}

}

} **catch** (SQLException e) {

e.printStackTrace();

}

}

}

**(Login.java)**

**package** javaapplication1;

**import** javax.swing.\*;

**import** java.awt.\*;

**import** java.awt.event.ActionEvent;

**import** java.awt.event.ActionListener;

**import** java.sql.PreparedStatement;

**import** java.sql.ResultSet;

**import** java.sql.SQLException;

@SuppressWarnings("serial")

**public** **class** Login **extends** JFrame {

**private** Dao conn;

**public** Login() {

**super**("🚀 IIT HELP DESK LOGIN 🚀");

conn = **new** Dao();

conn.createTables();

setSize(600, 300);

setLayout(**new** GridLayout(5, 2));

setLocationRelativeTo(**null**);

JLabel lblUsername = **new** JLabel("👤 Username", JLabel.***CENTER***);

JLabel lblPassword = **new** JLabel("🔒 Password", JLabel.***CENTER***);

JLabel lblStatus = **new** JLabel(" ", JLabel.***CENTER***);

JTextField txtUname = **new** JTextField(10);

JPasswordField txtPassword = **new** JPasswordField();

JButton btn = **new** JButton("🚀 Submit");

JButton btnExit = **new** JButton("🚪 Exit");

lblStatus.setForeground(Color.***RED***);

lblStatus.setFont(**new** Font("Arial", Font.***BOLD***, 14));

add(lblUsername);

add(txtUname);

add(lblPassword);

add(txtPassword);

add(btn);

add(btnExit);

add(lblStatus);

btn.addActionListener(**new** ActionListener() {

**int** count = 0;

@Override

**public** **void** actionPerformed(ActionEvent e) {

**boolean** admin = **false**;

count = count + 1;

String query = "SELECT \* FROM hamza\_jpapa\_users WHERE uname = ? and upass = ?;";

**try** (PreparedStatement stmt = conn.getConnection().prepareStatement(query)) {

stmt.setString(1, txtUname.getText());

stmt.setString(2, **new** String(txtPassword.getPassword()));

ResultSet rs = stmt.executeQuery();

**if** (rs.next()) {

admin = rs.getBoolean("admin");

**new** Tickets(admin);

setVisible(**false**);

dispose();

} **else** {

lblStatus.setText("❌ Try again! " + (3 - count) + " / 3 attempt(s) left");

}

} **catch** (SQLException ex) {

ex.printStackTrace();

}

}

});

btnExit.addActionListener(e -> System.*exit*(0));

setVisible(**true**);

}

**public** **static** **void** main(String[] args) {

SwingUtilities.*invokeLater*(() -> **new** Login());

}

}

**(Tickets.java)**

**package** javaapplication1;

**import** javax.swing.\*;

**import** java.awt.Color;

**import** java.awt.event.ActionEvent;

**import** java.awt.event.ActionListener;

**import** java.awt.event.WindowAdapter;

**import** java.awt.event.WindowEvent;

**import** java.sql.Date;

**import** java.sql.SQLException;

**import** java.text.ParseException;

**import** java.text.SimpleDateFormat;

@SuppressWarnings("serial")

**public** **class** Tickets **extends** JFrame **implements** ActionListener {

**private** Dao dao = **new** Dao();

**private** Boolean chkIfAdmin = **null**;

**private** JMenu mnuFile = **new** JMenu("File");

**private** JMenu mnuAdmin = **new** JMenu("Admin");

**private** JMenu mnuTickets = **new** JMenu("Tickets");

**private** JMenuItem mnuItemExit;

**private** JMenuItem mnuItemUpdate;

**private** JMenuItem mnuItemDelete;

**private** JMenuItem mnuItemOpenTicket;

**private** JMenuItem mnuItemViewTicket;

**private** JMenuItem mnuItemToggleStatus;

**public** Tickets(Boolean isAdmin) {

chkIfAdmin = isAdmin;

createMenu();

prepareGUI();

}

**private** **void** createMenu() {

mnuItemExit = **new** JMenuItem("Exit");

mnuFile.add(mnuItemExit);

mnuItemUpdate = **new** JMenuItem("Update Ticket");

mnuAdmin.add(mnuItemUpdate);

mnuItemDelete = **new** JMenuItem("Delete Ticket");

mnuAdmin.add(mnuItemDelete);

mnuItemOpenTicket = **new** JMenuItem("Open Ticket");

mnuTickets.add(mnuItemOpenTicket);

mnuItemViewTicket = **new** JMenuItem("View Ticket");

mnuTickets.add(mnuItemViewTicket);

// Add a new menu item for toggling ticket status

mnuItemToggleStatus = **new** JMenuItem("Toggle Ticket Status");

mnuTickets.add(mnuItemToggleStatus);

mnuItemToggleStatus.addActionListener(**this**);

}

**private** **void** prepareGUI() {

JMenuBar bar = **new** JMenuBar();

bar.add(mnuFile);

bar.add(mnuAdmin);

bar.add(mnuTickets);

setJMenuBar(bar);

addWindowListener(**new** WindowAdapter() {

**public** **void** windowClosing(WindowEvent wE) {

System.*exit*(0);

}

});

setSize(400, 400);

getContentPane().setBackground(Color.***LIGHT\_GRAY***);

setLocationRelativeTo(**null**);

setVisible(**true**);

}

@Override

**public** **void** actionPerformed(ActionEvent e) {

**if** (e.getSource() == mnuItemExit) {

System.*exit*(0);

} **else** **if** (e.getSource() == mnuItemOpenTicket) {

// ... existing code ...

} **else** **if** (e.getSource() == mnuItemViewTicket) {

getAllDetails();

} **else** **if** (e.getSource() == mnuItemUpdate) {

// ... existing code ...

} **else** **if** (e.getSource() == mnuItemDelete) {

// ... existing code ...

} **else** **if** (e.getSource() == mnuItemToggleStatus) {

toggleTicketStatus();

}

}

**private** **void** getAllDetails() {

**try** {

JTable jt = **new** JTable(ticketsJTable.*buildTableModel*(dao.readRecords()));

jt.setBounds(30, 40, 200, 400);

JScrollPane sp = **new** JScrollPane(jt);

add(sp);

setVisible(**true**);

} **catch** (SQLException e1) {

e1.printStackTrace();

}

}

**private** **void** toggleTicketStatus() {

String ticketIdToUpdate = JOptionPane.*showInputDialog*(**null**, "Enter the ID of the ticket to toggle status:");

**int** ticketId;

**if** (ticketIdToUpdate != **null** && !ticketIdToUpdate.trim().isEmpty()) {

ticketId = Integer.*parseInt*(ticketIdToUpdate);

dao.toggleTicketStatus(ticketId);

getAllDetails();

} **else** {

System.***out***.println("Invalid Ticket ID");

}

}

**private** **static** Date convertStringToSqlDate(String dateString) {

SimpleDateFormat dateFormat = **new** SimpleDateFormat("yyyy-MM-dd");

**try** {

java.util.Date utilDate = dateFormat.parse(dateString);

**return** **new** Date(utilDate.getTime());

} **catch** (ParseException e) {

e.printStackTrace();

**return** **null**;

}

}

**public** **static** **void** main(String[] args) {

SwingUtilities.*invokeLater*(() -> **new** Login());

}

}

**(ticketsJTable.java)**

**package** javaapplication1;

**import** javax.swing.table.DefaultTableModel;

**import** java.sql.ResultSet;

**import** java.sql.ResultSetMetaData;

**import** java.sql.SQLException;

**import** java.util.Vector;

**public** **class** ticketsJTable {

**public** **static** DefaultTableModel buildTableModel(ResultSet rs) **throws** SQLException {

ResultSetMetaData metaData = rs.getMetaData();

// names of columns

Vector<String> columnNames = **new** Vector<String>();

**int** columnCount = metaData.getColumnCount();

**for** (**int** column = 1; column <= columnCount; column++) {

columnNames.add(metaData.getColumnName(column));

}

// data of the table

Vector<Vector<Object>> data = **new** Vector<Vector<Object>>();

**while** (rs.next()) {

Vector<Object> vector = **new** Vector<Object>();

**for** (**int** columnIndex = 1; columnIndex <= columnCount; columnIndex++) {

vector.add(rs.getObject(columnIndex));

}

data.add(vector);

}

// return data/column names for JTable

**return** **new** DefaultTableModel(data, columnNames);

}

}