**FINAL PROJECT**

**ADVANCED VISA PROCESSING SYSTEM**

1. **S.S.L.Prasanna**

**N190367**

**Batch - 6**

**CODE:**

**Main.java:**

**package** com.myvisaapplication;

**public** **class** Main {

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

VisaApplicationManager manager = **new** VisaApplicationManager();

InputScanner inputScanner = **new** InputScanner();

FileHandler FileHandler = **new** FileHandler();

**while** (**true**) {

System.***out***.println("Visa Processing System");

System.***out***.println("1. Submit new visa application");

System.***out***.println("2. Process visa applications");

System.***out***.println("3. Update application status");

System.***out***.println("4. View applications by status");

System.***out***.println("5. View top 10 visa types");

System.***out***.println("6. Exit");

**int** choice = inputScanner.readInt("Enter your choice: ");

**switch** (choice) {

**case** 1:

String name = inputScanner.readString("Enter applicant name: ");

String visaType = inputScanner.readVisaType();

VisaApplication application = **new** VisaApplication(name, visaType);

manager.addApplication(application);

System.***out***.println("Application submitted.");

**break**;

**case** 2:

// Processing logic would go here

**break**;

**case** 3:

String applicantName = inputScanner.readString("Enter applicant name: ");

String status = inputScanner.readString("Enter new status: ");

**try** {

manager.updateApplicationStatus(applicantName, status);

System.***out***.println("Application status updated.");

} **catch** (ApplicationNotFoundException e) {

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

}

**break**;

**case** 4:

String statusFilter = inputScanner.readString("Enter status to filter by: ");

manager.getApplicationsByStatus(statusFilter).forEach(System.***out***::println);

**break**;

**case** 5:

manager.getTopVisaTypes(10).forEach(entry ->

System.out.println(entry.getKey() + ": " + entry.getValue()));

**break**;

**case** 6:

System.***out***.println("Exiting the program.");

**return**;

**default**:

System.***out***.println("Invalid choice. Please try again.");

}

}

}

}

**VisaApplication.Java:**

**package** com.myvisaapplication;

**import** java.time.LocalDateTime;

**public** **class** VisaApplication {

**private** String applicantName;

**private** String visaType;

**private** String status;

**private** LocalDateTime submissionDate;

**private** LocalDateTime processingDate;

**public** VisaApplication(String applicantName, String visaType) {

**this**.applicantName = applicantName;

**this**.visaType = visaType;

**this**.status = "Submitted";

**this**.submissionDate = LocalDateTime.*now*();

}

**public** String getApplicantName() {

**return** applicantName;

}

**public** **void** setApplicantName(String applicantName) {

**this**.applicantName = applicantName;

}

**public** String getVisaType() {

**return** visaType;

}

**public** **void** setVisaType(String visaType) {

**this**.visaType = visaType;

}

**public** String getStatus() {

**return** status;

}

**public** **void** setStatus(String status) {

**this**.status = status;

}

**public** LocalDateTime getSubmissionDate() {

**return** submissionDate;

}

**public** LocalDateTime getProcessingDate() {

**return** processingDate;

}

**public** **void** setProcessingDate(LocalDateTime processingDate) {

**this**.processingDate = processingDate;

}

@Override

**public** String toString() {

**return** "VisaApplication{" +

"applicantName='" + applicantName + '\'' +

", visaType='" + visaType + '\'' +

", status='" + status + '\'' +

", submissionDate=" + submissionDate +

", processingDate=" + processingDate +

'}';

}

**public** **void** setSubmissionDate(LocalDateTime submissionDate2) {

// **TODO** Auto-generated method stub

**throw** **new** UnsupportedOperationException("Unimplemented method 'setSubmissionDate'");

}

}

**VisaApplicationManager.java:**

package com.myvisaapplication;

import java.time.LocalDateTime;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import java.util.stream.Collectors;

public class VisaApplicationManager {

private List<VisaApplication> applications;

private Map<String, Employee> employees;

public VisaApplicationManager() {

applications = new ArrayList<>();

employees = new HashMap<>();

}

public void addApplication(VisaApplication application) {

applications.add(application);

}

public void removeApplication(String applicantName) throws ApplicationNotFoundException {

boolean removed = applications.removeIf(app -> app.getApplicantName().equalsIgnoreCase(applicantName));

if (!removed) {

throw new ApplicationNotFoundException("Application not found for applicant: " + applicantName);

}

}

public void updateApplicationStatus(String applicantName, String newStatus) throws ApplicationNotFoundException {

for (VisaApplication application : applications) {

if (application.getApplicantName().equalsIgnoreCase(applicantName)) {

application.setStatus(newStatus);

application.setProcessingDate(LocalDateTime.now());

return;

}

}

throw new ApplicationNotFoundException("Application not found for applicant: " + applicantName);

}

public List<VisaApplication> getApplications() {

return applications;

}

public List<VisaApplication> getApplicationsByStatus(String status) {

return applications.stream()

.filter(app -> app.getStatus().equalsIgnoreCase(status))

.collect(Collectors.toList());

}

public Map<String, Long> countVisaTypes() {

return applications.stream()

.collect(Collectors.groupingBy(VisaApplication::getVisaType, Collectors.counting()));

}

public List<Map.Entry<String, Long>> getTopVisaTypes(int limit) {

return countVisaTypes().entrySet().stream()

.sorted(Map.Entry.<String, Long>comparingByValue().reversed())

.limit(limit)

.collect(Collectors.toList());

}

}

**VisaApplicationProcessingSystem.java:**

package com.myvisaapplication;

import java.time.LocalDateTime;

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

import java.util.stream.Collectors;

public class VisaApplicationManager {

private List<VisaApplication> applications;

private Map<String, Employee> employees;

public VisaApplicationManager() {

applications = new ArrayList<>();

employees = new HashMap<>();

}

public void addApplication(VisaApplication application) {

applications.add(application);

}

public void removeApplication(String applicantName) throws ApplicationNotFoundException {

boolean removed = applications.removeIf(app -> app.getApplicantName().equalsIgnoreCase(applicantName));

if (!removed) {

throw new ApplicationNotFoundException("Application not found for applicant: " + applicantName);

}

}

public void updateApplicationStatus(String applicantName, String newStatus) throws ApplicationNotFoundException {

for (VisaApplication application : applications) {

if (application.getApplicantName().equalsIgnoreCase(applicantName)) {

application.setStatus(newStatus);

application.setProcessingDate(LocalDateTime.now());

return;

}

}

throw new ApplicationNotFoundException("Application not found for applicant: " + applicantName);

}

public List<VisaApplication> getApplications() {

return applications;

}

public List<VisaApplication> getApplicationsByStatus(String status) {

return applications.stream()

.filter(app -> app.getStatus().equalsIgnoreCase(status))

.collect(Collectors.toList());

}

public Map<String, Long> countVisaTypes() {

return applications.stream()

.collect(Collectors.groupingBy(VisaApplication::getVisaType, Collectors.counting()));

}

public List<Map.Entry<String, Long>> getTopVisaTypes(int limit) {

return countVisaTypes().entrySet().stream()

.sorted(Map.Entry.<String, Long>comparingByValue().reversed())

.limit(limit)

.collect(Collectors.toList());

}

}

**VisaProcessingSystem.Java:**

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

**import** javax.swing.border.EmptyBorder;

**import** javax.swing.table.DefaultTableCellRenderer;

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

**import** javax.swing.table.TableCellRenderer;

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

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

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

**import** java.util.Vector;

**public** **class** VisaProcessingSystem **extends** JFrame {

**private** JTextField nameField, passportField, nationalityField, dobField, visaTypeField, statusField;

**private** JTextArea logArea;

**private** DefaultTableModel tableModel;

**private** JTable table;

**public** VisaProcessingSystem() {

setTitle("Visa Processing System");

setSize(800, 600);

setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

setLocationRelativeTo(**null**);

// Main panel

JPanel mainPanel = **new** JPanel(**new** BorderLayout());

mainPanel.setBackground(Color.***WHITE***); // Set background color

setContentPane(mainPanel);

// Form panel

JPanel formPanel = **new** JPanel(**new** GridBagLayout());

formPanel.setBorder(BorderFactory.*createEmptyBorder*(20, 20, 20, 20)); // Add padding

formPanel.setBackground(Color.***WHITE***); // Set background color

mainPanel.add(formPanel, BorderLayout.***NORTH***);

GridBagConstraints gbc = **new** GridBagConstraints();

gbc.anchor = GridBagConstraints.***WEST***;

gbc.insets = **new** Insets(5, 5, 5, 5);

gbc.gridx = 0;

gbc.gridy = 0;

formPanel.add(**new** JLabel("Name:"), gbc);

gbc.gridx = 1;

gbc.gridy = 0;

nameField = **new** JTextField(20);

formPanel.add(nameField, gbc);

gbc.gridx = 0;

gbc.gridy = 1;

formPanel.add(**new** JLabel("Passport:"), gbc);

gbc.gridx = 1;

gbc.gridy = 1;

passportField = **new** JTextField(20);

formPanel.add(passportField, gbc);

gbc.gridx = 0;

gbc.gridy = 2;

formPanel.add(**new** JLabel("Nationality:"), gbc);

gbc.gridx = 1;

gbc.gridy = 2;

nationalityField = **new** JTextField(20);

formPanel.add(nationalityField, gbc);

gbc.gridx = 0;

gbc.gridy = 3;

formPanel.add(**new** JLabel("DOB (yyyy-MM-dd):"), gbc);

gbc.gridx = 1;

gbc.gridy = 3;

dobField = **new** JTextField(20);

formPanel.add(dobField, gbc);

gbc.gridx = 0;

gbc.gridy = 4;

formPanel.add(**new** JLabel("Visa Type:"), gbc);

gbc.gridx = 1;

gbc.gridy = 4;

visaTypeField = **new** JTextField(20);

formPanel.add(visaTypeField, gbc);

gbc.gridx = 0;

gbc.gridy = 5;

formPanel.add(**new** JLabel("Status:"), gbc);

gbc.gridx = 1;

gbc.gridy = 5;

statusField = **new** JTextField(20);

formPanel.add(statusField, gbc);

// Button panel

JPanel buttonPanel = **new** JPanel(**new** FlowLayout(FlowLayout.***CENTER***));

buttonPanel.setBackground(Color.***WHITE***); // Set background color

mainPanel.add(buttonPanel, BorderLayout.***CENTER***);

JButton addButton = **new** JButton("Add");

addButton.setBackground(**new** Color(30, 144, 255)); // Set background color

addButton.setForeground(Color.***WHITE***); // Set text color

addButton.setFocusPainted(**false**); // Remove focus border

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

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

addApplication();

}

});

buttonPanel.add(addButton);

JButton removeButton = **new** JButton("Remove");

removeButton.setBackground(**new** Color(255, 69, 0)); // Set background color

removeButton.setForeground(Color.***WHITE***); // Set text color

removeButton.setFocusPainted(**false**); // Remove focus border

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

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

removeApplication();

}

});

buttonPanel.add(removeButton);

JButton updateButton = **new** JButton("Update");

updateButton.setBackground(**new** Color(60, 179, 113)); // Set background color

updateButton.setForeground(Color.***WHITE***); // Set text color

updateButton.setFocusPainted(**false**); // Remove focus border

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

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

updateApplication();

}

});

buttonPanel.add(updateButton);

// Table panel

JPanel tablePanel = **new** JPanel(**new** BorderLayout());

tablePanel.setBorder(BorderFactory.*createEmptyBorder*(20, 20, 20, 20)); // Add padding

tablePanel.setBackground(Color.***WHITE***); // Set background color

mainPanel.add(tablePanel, BorderLayout.***SOUTH***);

// Table to display applications

tableModel = **new** DefaultTableModel() {

@Override

**public** Class<?> getColumnClass(**int** columnIndex) {

**return** String.**class**; // Ensure all columns are treated as Strings

}

};

table = **new** JTable(tableModel);

table.setBackground(Color.***WHITE***); // Set background color

// Set renderer for column headers to have colored backgrounds

TableCellRenderer headerRenderer = **new** DefaultTableCellRenderer() {

@Override

**public** Component getTableCellRendererComponent(JTable table, Object value, **boolean** isSelected, **boolean** hasFocus, **int** row, **int** column) {

JLabel label = (JLabel) **super**.getTableCellRendererComponent(table, value, isSelected, hasFocus, row, column);

label.setBackground(**new** Color(70, 130, 180)); // Set background color for headers

label.setForeground(Color.***WHITE***); // Set text color for headers

label.setHorizontalAlignment(JLabel.***CENTER***); // Center align text

**return** label;

}

};

// Apply renderer to each column header

**for** (**int** i = 0; i < tableModel.getColumnCount(); i++) {

table.getTableHeader().getColumnModel().getColumn(i).setHeaderRenderer(headerRenderer);

}

JScrollPane scrollPane = **new** JScrollPane(table);

tablePanel.add(scrollPane, BorderLayout.***CENTER***);

// Add columns to the table

String[] columnNames = {"Name", "Passport", "Nationality", "DOB", "Visa Type", "Status"};

tableModel.setColumnIdentifiers(columnNames);

// Log area

logArea = **new** JTextArea(10, 40);

logArea.setEditable(**false**);

logArea.setBackground(Color.***WHITE***); // Set background color

JScrollPane logScrollPane = **new** JScrollPane(logArea);

tablePanel.add(logScrollPane, BorderLayout.***EAST***);

setVisible(**true**);

}

**private** **void** addApplication() {

String name = nameField.getText();

String passport = passportField.getText();

String nationality = nationalityField.getText();

String dob = dobField.getText();

String visaType = visaTypeField.getText();

String status = statusField.getText();

**if** (!name.isEmpty() && !passport.isEmpty() && !nationality.isEmpty() && !dob.isEmpty() && !visaType.isEmpty() && !status.isEmpty()) {

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

row.add(name);

row.add(passport);

row.add(nationality);

row.add(dob);

row.add(visaType);

row.add(status);

tableModel.addRow(row);

clearFields();

log("Application added to table:");

log("Name: " + name);

log("Passport: " + passport);

log("Nationality: " + nationality);

log("DOB: " + dob);

log("Visa Type: " + visaType);

log("Status: " + status);

} **else** {

JOptionPane.*showMessageDialog*(**this**, "Please fill in all fields.", "Error", JOptionPane.***ERROR\_MESSAGE***);

}

}

**private** **void** removeApplication() {

**int** selectedRow = table.getSelectedRow();

**if** (selectedRow != -1) {

tableModel.removeRow(selectedRow);

log("Application removed from table.");

} **else** {

JOptionPane.*showMessageDialog*(**this**, "Please select a row to remove.", "Error", JOptionPane.***ERROR\_MESSAGE***);

}

}

**private** **void** updateApplication() {

**int** selectedRow = table.getSelectedRow();

**if** (selectedRow != -1) {

tableModel.setValueAt(nameField.getText(), selectedRow, 0);

tableModel.setValueAt(passportField.getText(), selectedRow, 1);

tableModel.setValueAt(nationalityField.getText(), selectedRow, 2);

tableModel.setValueAt(dobField.getText(), selectedRow, 3);

tableModel.setValueAt(visaTypeField.getText(), selectedRow, 4);

tableModel.setValueAt(statusField.getText(), selectedRow, 5);

log("Application updated in table:");

log("Name: " + nameField.getText());

log("Passport: " + passportField.getText());

log("Nationality: " + nationalityField.getText());

log("DOB: " + dobField.getText());

log("Visa Type: " + visaTypeField.getText());

log("Status: " + statusField.getText());

clearFields();

} **else** {

JOptionPane.*showMessageDialog*(**this**, "Please select a row to update.", "Error", JOptionPane.***ERROR\_MESSAGE***);

}

}

**private** **void** clearFields() {

nameField.setText("");

passportField.setText("");

nationalityField.setText("");

dobField.setText("");

visaTypeField.setText("");

statusField.setText("");

}

**private** **void** log(String message) {

logArea.append(message + "\n");

}

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

SwingUtilities.*invokeLater*(**new** Runnable() {

**public** **void** run() {

**new** VisaProcessingSystem();

}

});

}

}

**MYSQL-Database Queries:**

--Create database

CREATE DATABASE visa\_processing\_system;

-- Use the database

USE visa\_processing\_system;

-- Create table for visa applications

CREATE TABLE applications (

id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(100) NOT NULL,

passport VARCHAR(50) NOT NULL,

nationality VARCHAR(50) NOT NULL,

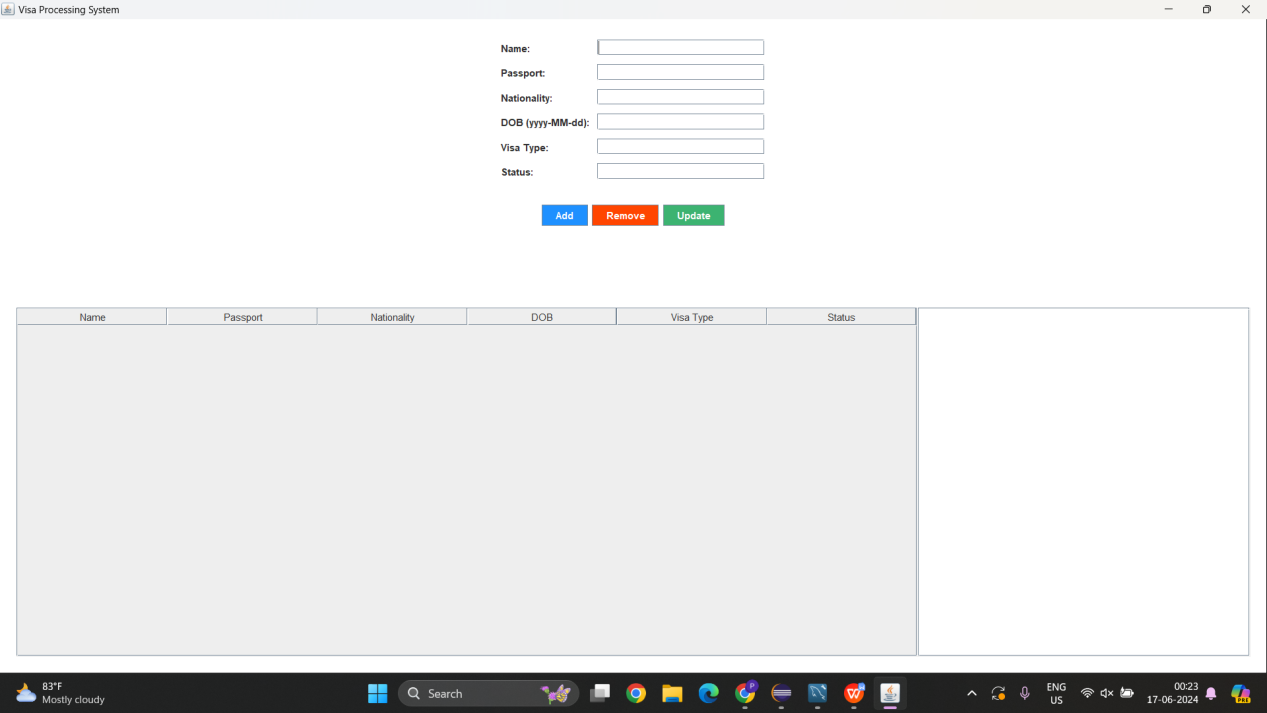
dob DATE NOT NULL,

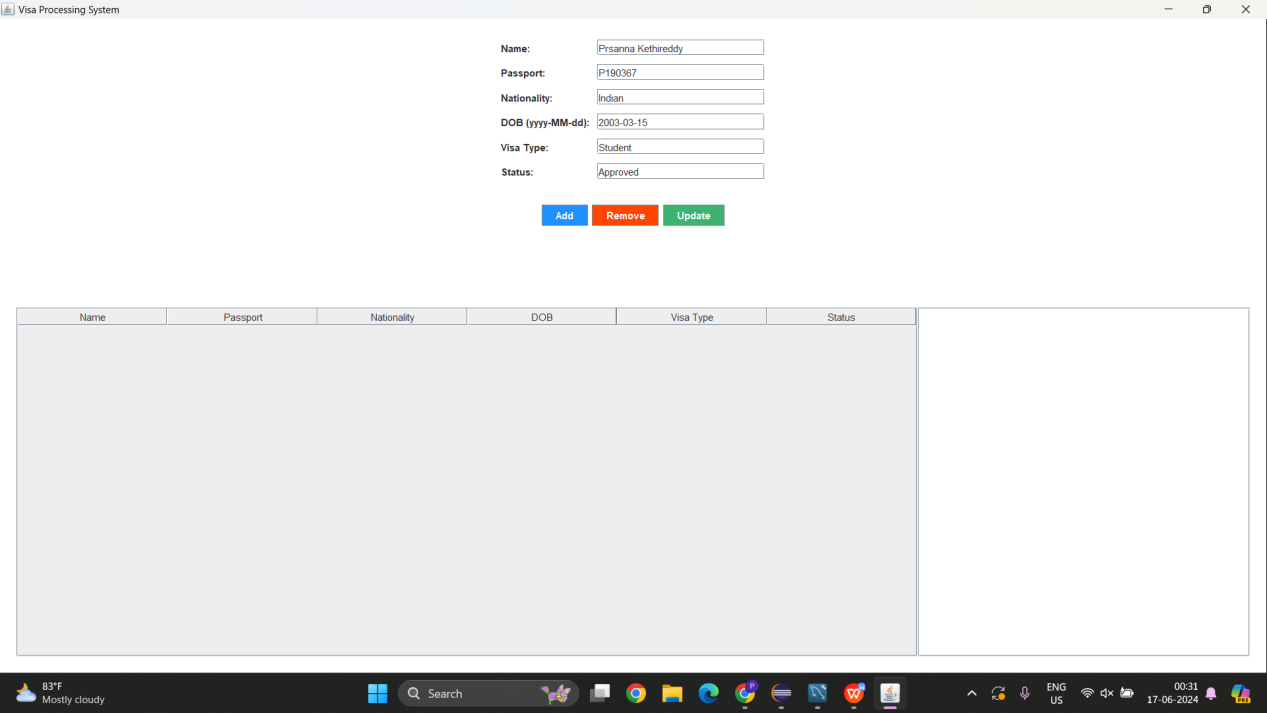
visa\_type VARCHAR(50) NOT NULL,

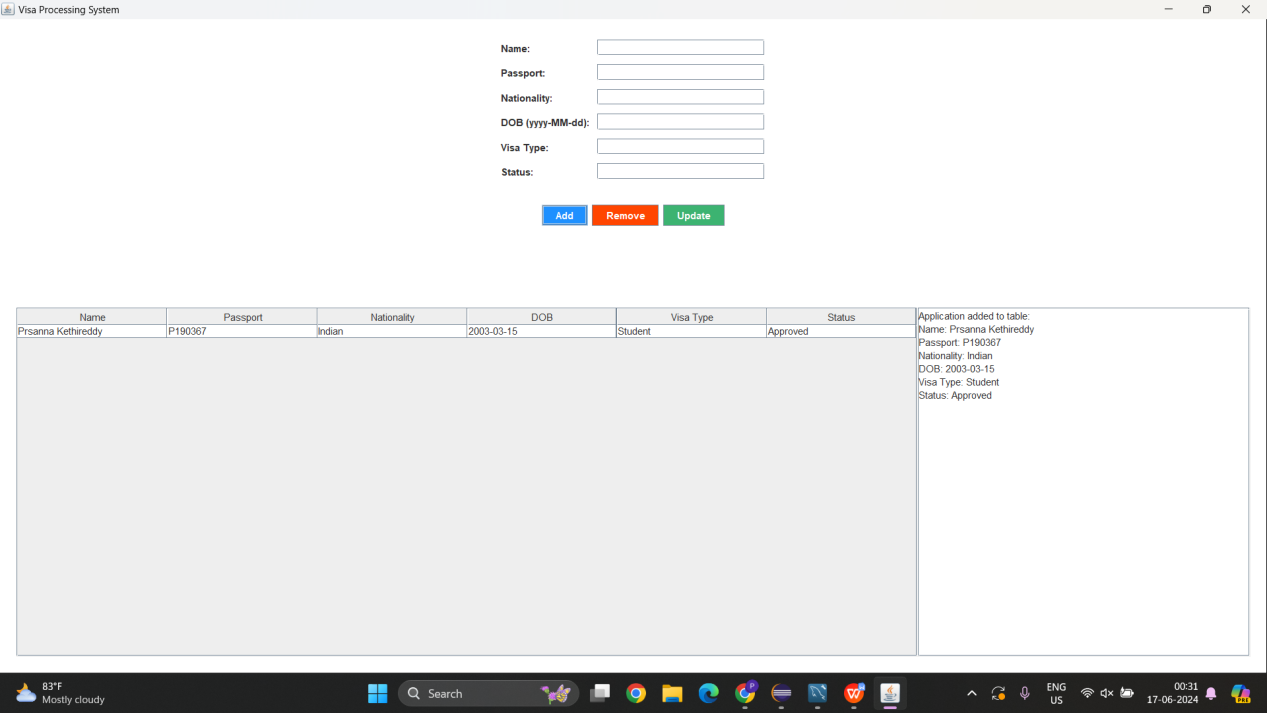
status VARCHAR(50) NOT NULL

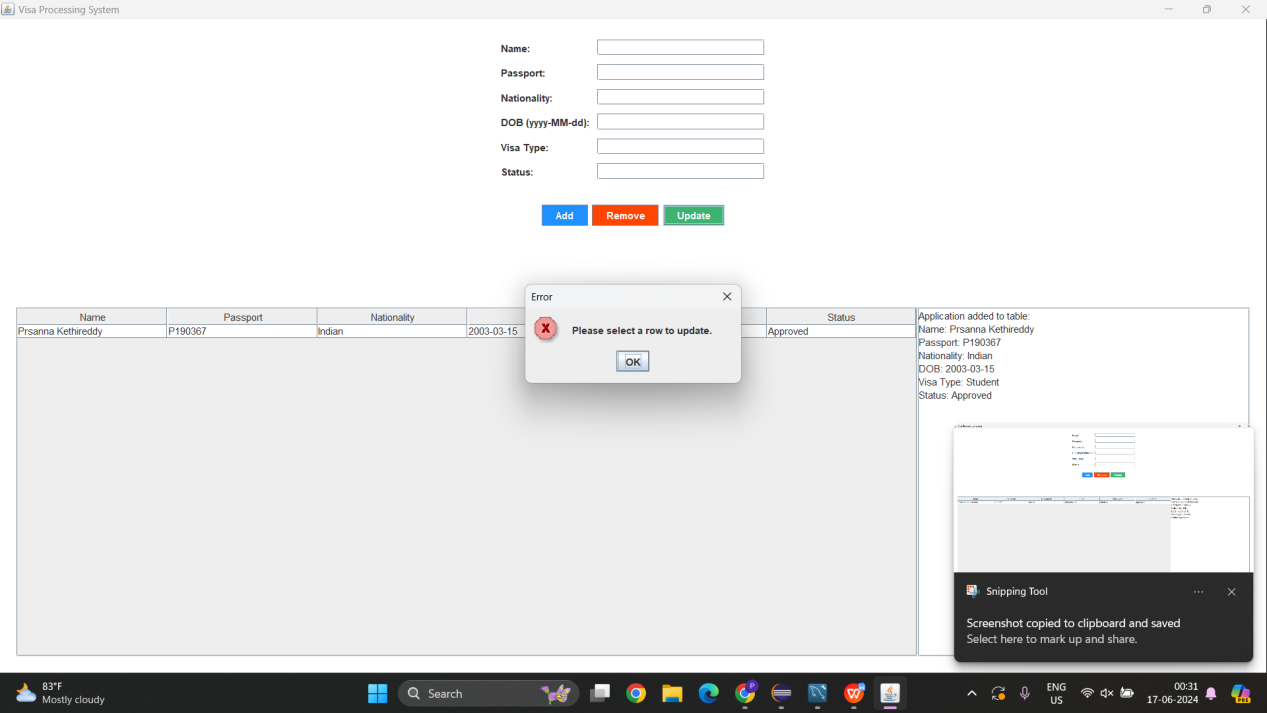
);

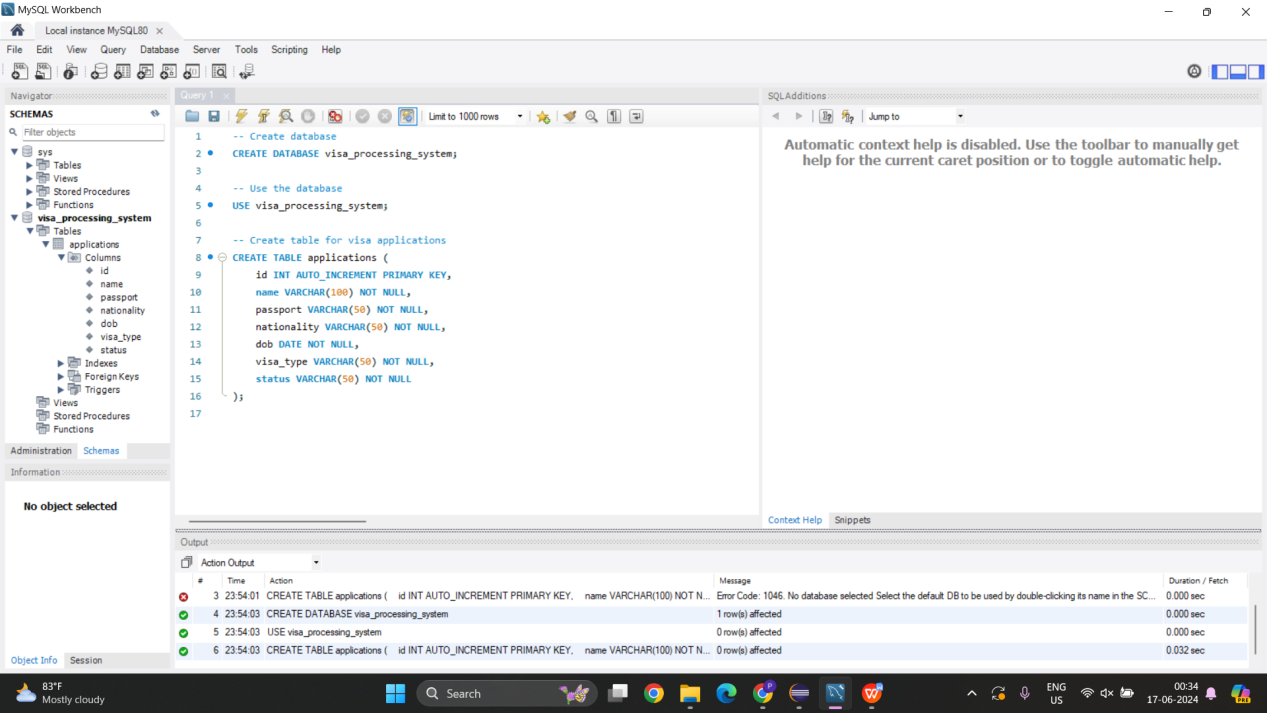
**OUTPUTS:**

****

****

****

****

****

**SUMMARY:**

#### Project Overview

The Visa Processing System is a Java-based desktop application designed to streamline and manage visa applications. The application allows users to submit new visa applications, review existing applications, and update the status of applications. It provides an intuitive graphical user interface (GUI) built using Java Swing and AWT, ensuring ease of use for users.

#### Technologies Used

* **Programming Language:** Java
* **GUI Toolkit:** Swing and AWT
* **Database:** Initially, the application was designed without a database connection for simplicity, but it can be extended to use MySQL or any other relational database for persistent storage.

#### Application Structure

* **Main Class:** VisaProcessingSystem
  + Manages the main application window and menu.
* **Panels and Forms:**
  + **FormPanel:** Collects user input for visa application details.
  + **ButtonPanel:** Contains action buttons like Add, Remove, and Update.
  + **TablePanel:** Displays a table of visa applications.
  + **LogPanel:** Displays logs and messages to the user.

#### Functionalities

* **Submit New Application:**
  + Users can enter details such as name, passport number, nationality, date of birth, visa type, and application status.
  + On clicking "Add," the application data is added to a table displayed in the interface.
* **Review Applications:**
  + Users can view all submitted applications in a tabular format.
* **Update Application Status:**
  + Users can select an application from the table and update its details using the form.
* **Remove Application:**
  + Users can remove a selected application from the table.

#### User Interface

* **Form Fields:**
  + **Name:** Text field for the applicant's name.
  + **Passport:** Text field for the passport number.
  + **Nationality:** Text field for the applicant's nationality.
  + **DOB (Date of Birth):** Text field for the date of birth in yyyy-MM-dd format.
  + **Visa Type:** Text field for the type of visa being applied for.
  + **Status:** Text field for the current status of the application.
* **Buttons:**
  + **Add:** Adds the input data to the table.
  + **Remove:** Removes the selected application from the table.
  + **Update:** Updates the selected application's details with the new input data.
* **Table:**
  + Displays all the submitted visa applications with columns for each field.
* **Log Area:**
  + Displays messages and logs related to user actions and application status.

#### Example Workflow

1. **Submit New Application:**
   * The user fills in the form with details such as John Doe, passport number A12345678, nationality American, date of birth 1985-06-15, visa type Tourist, and status Pending.
   * The user clicks "Add," and the application is added to the table below the form.
2. **Review Applications:**
   * The user views the table to see all submitted applications.
3. **Update Application:**
   * The user selects an application from the table.
   * The form fields are populated with the selected application's details.
   * The user modifies the details and clicks "Update," updating the application's details in the table.
4. **Remove Application:**
   * The user selects an application from the table and clicks "Remove," deleting it from the table.

#### Conclusion

The Visa Processing System project is a comprehensive Java application that effectively demonstrates the use of Swing and AWT for building desktop GUIs. It allows for easy management of visa applications through a user-friendly interface, facilitating the submission, review, updating, and removal of visa applications. This project can be further extended by integrating with a database for persistent storage and enhancing the application's functionalities to cater to real-world requirements.