

**Lahore Campus**

**OBJECT ORIENTED PROGRAMMING**

**Crime Reporting System Project**

**Submitted to:** Sir Shahid Bhatti

**Submitted by:** Laraib Qadeer, Namirah Manzoor

**Registration no.:** FA24-BCS-047, FA24-BCS-089.

**Section:** B

**Date:** 29/05/2025

**ABSTRACT**

The Crime Reporting System acts as an innovative platform that advances the practices of crime incident reporting and management along with resolution tasks. The application utilizes Java OOP methods to create a better public security framework which allows for effective crime report access by Public, Police Officers and Admins. The system maintains secure controlled data exchange through its capacity for anonymous reporting and delivers immediate updates.

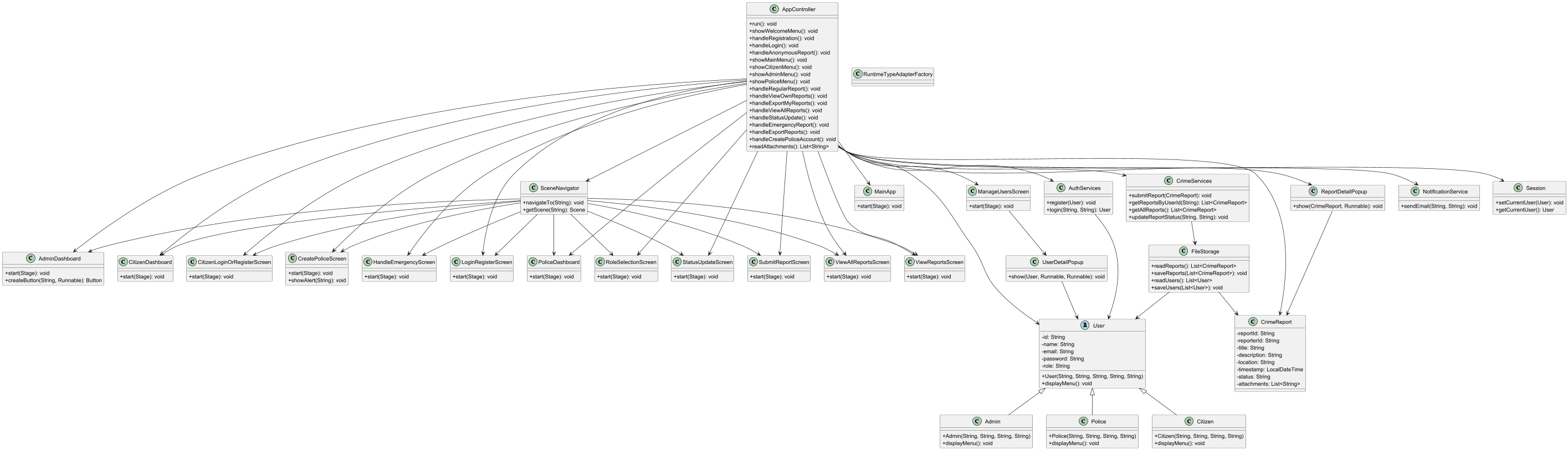
This system employs role-based access control enabled by Inheritance for different user types including Citizen and Police Officer and Admin where each category possesses unique permissions. The technique of encapsulation safeguards important data including passwords and report information by ensuring complete privacy and safety measures. Through polymorphism the system enables users to access reports according to their assigned permissions which enables them to resolve reports when performing officer duties while managing user accounts as administrative personnel.

The report submission process adopts composition through the CrimeReport class that accepts Location permitting users to create complete reporting forms with their positioning information.

The AuthServices and CrimeServices class applies Encapsulation as its main design principle for data persistence duties involving file storage or simple database storage systems. The system uses individual FollowUp objects linked to crime reports to manage their follow-up process.

The Crime Reporting System implements Java’s OOP principles to create an effective platform which combines scalability with security and efficient data management for crime management.

**UML**



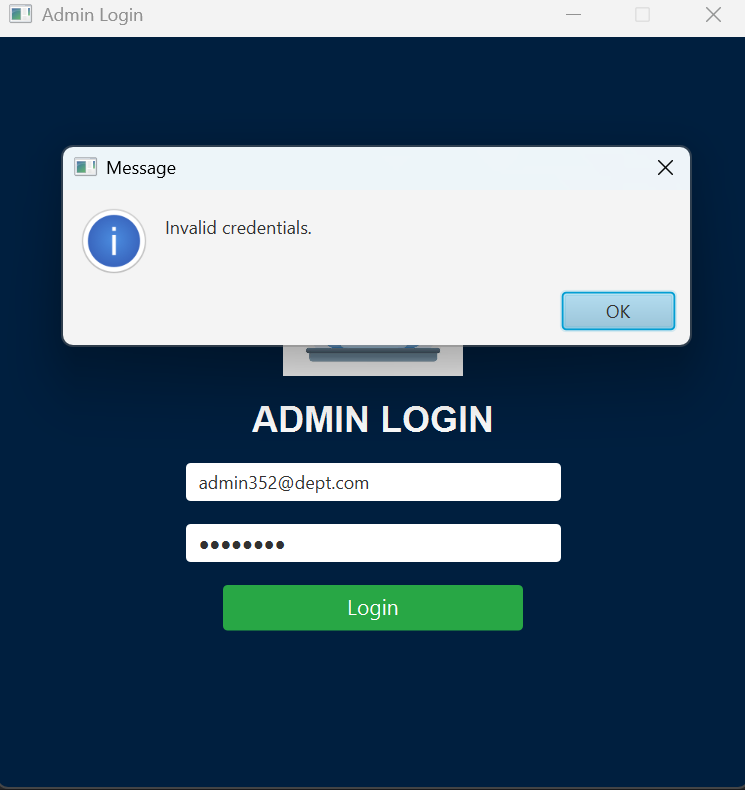
**OUTPUT**

A screenshot of a computer

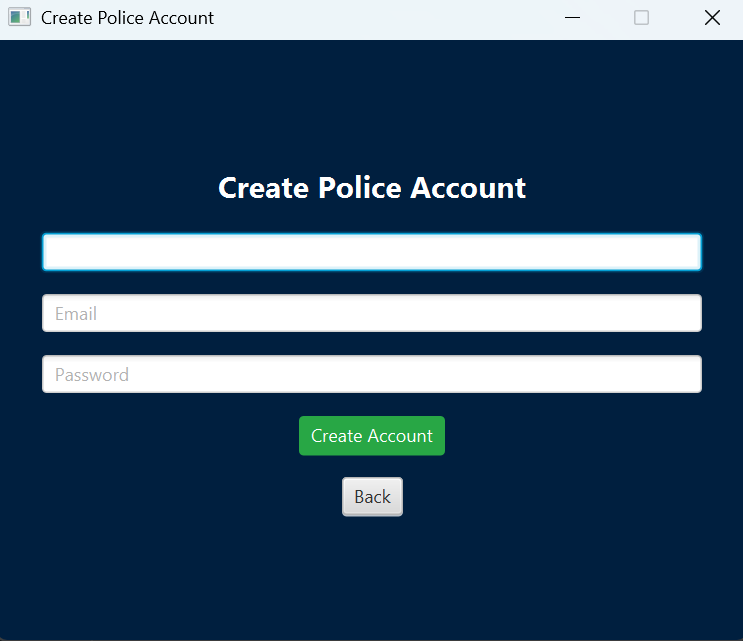
AI-generated content may be incorrect.A screenshot of a login page

AI-generated content may be incorrect.A group of people in uniform

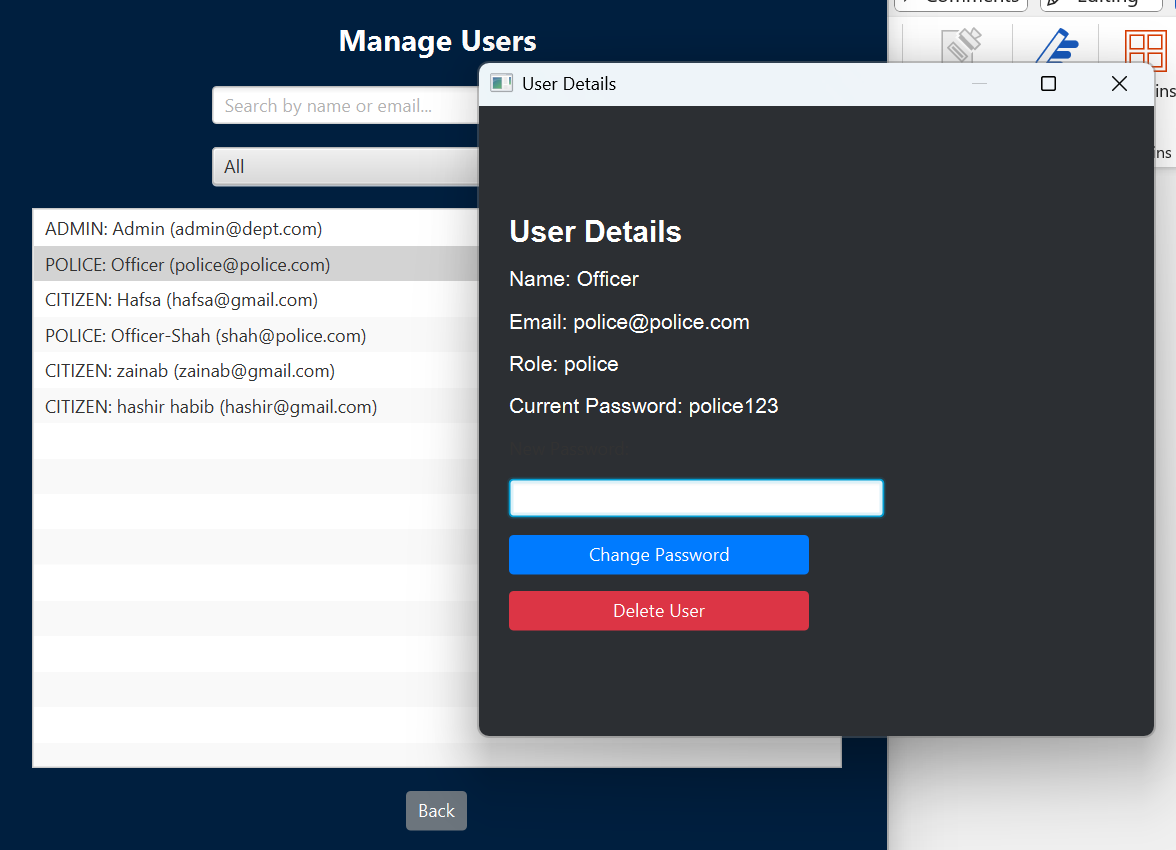
AI-generated content may be incorrect.



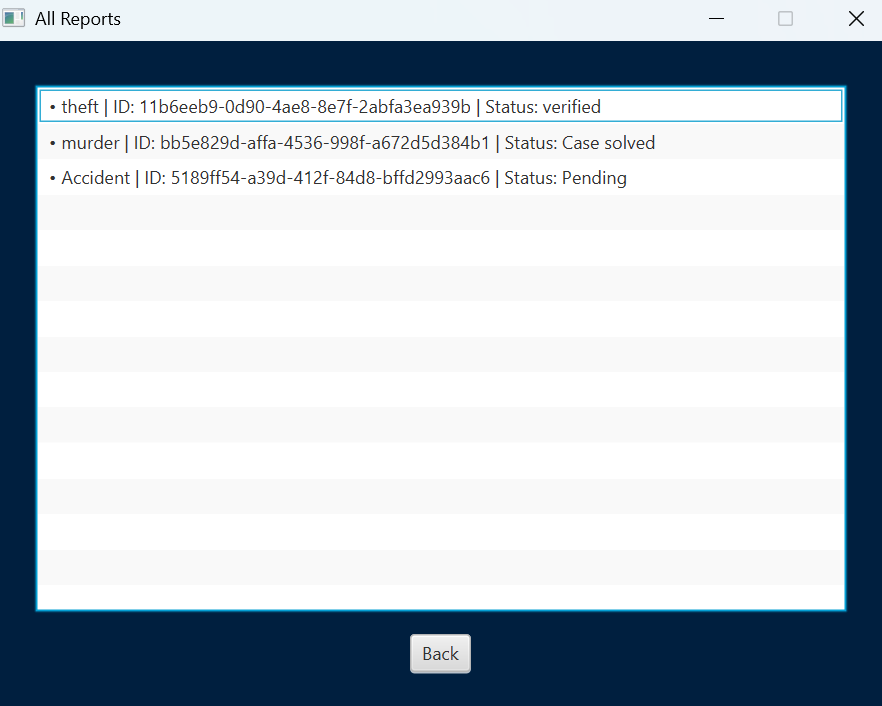
A screenshot of a computer

AI-generated content may be incorrect.

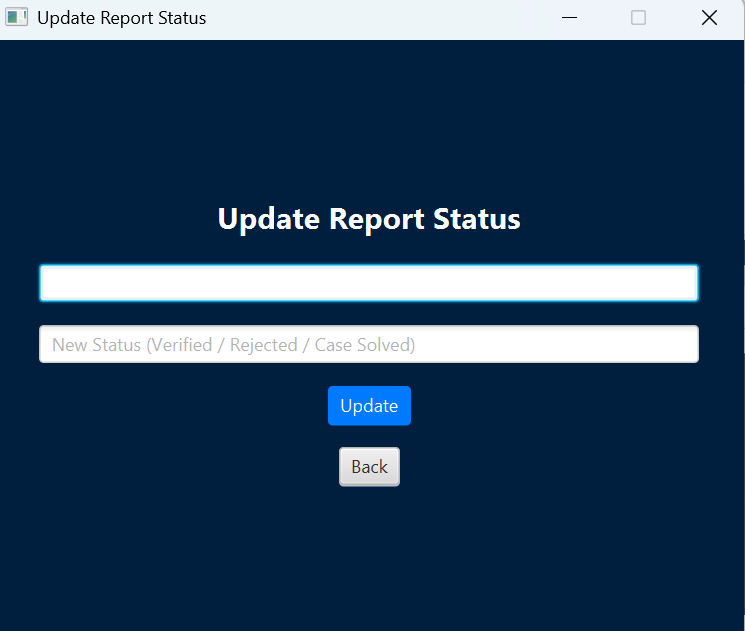
A screenshot of a computer

AI-generated content may be incorrect.

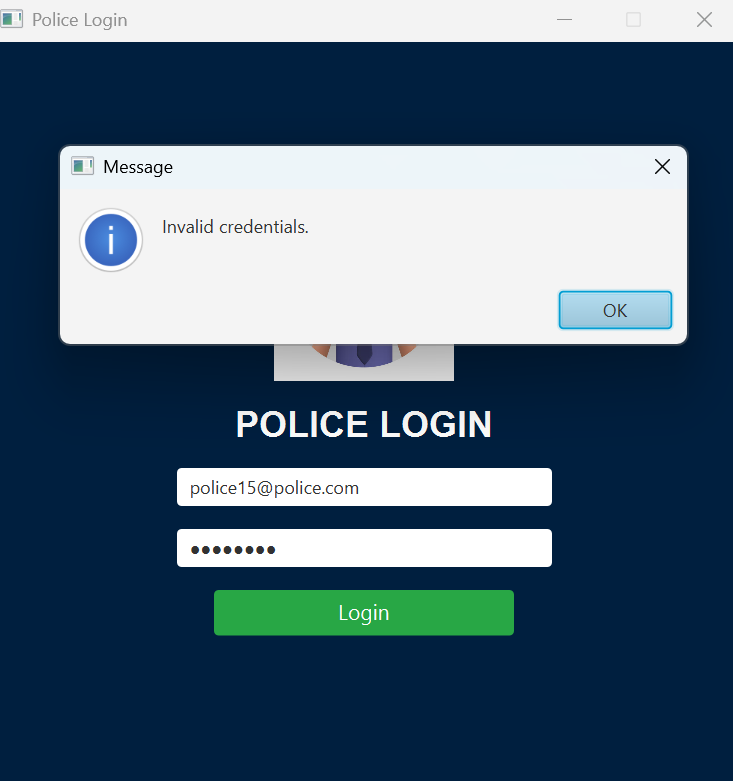
A screenshot of a computer

AI-generated content may be incorrect.

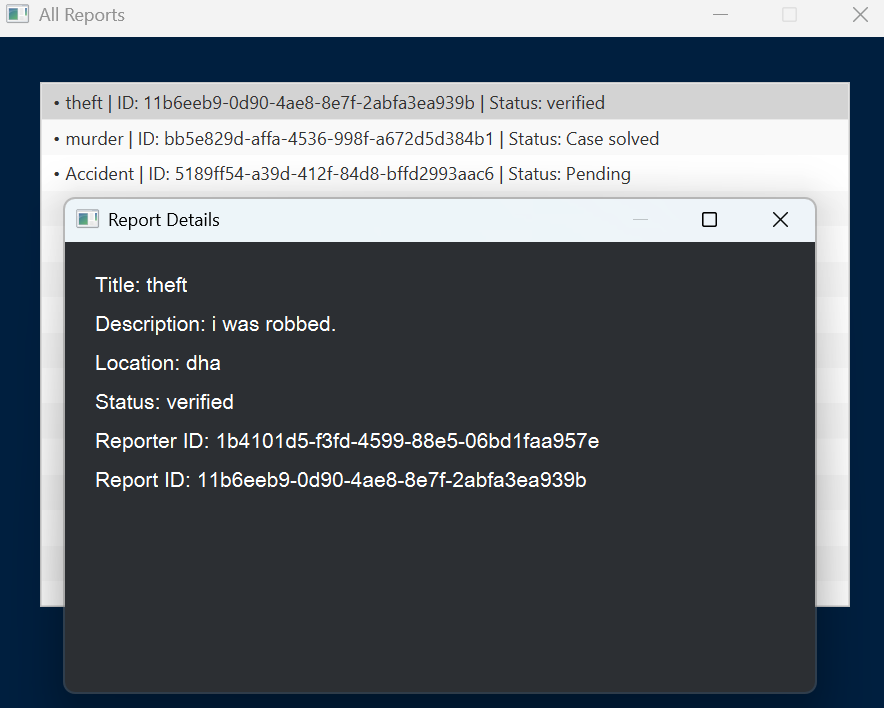
A screenshot of a computer

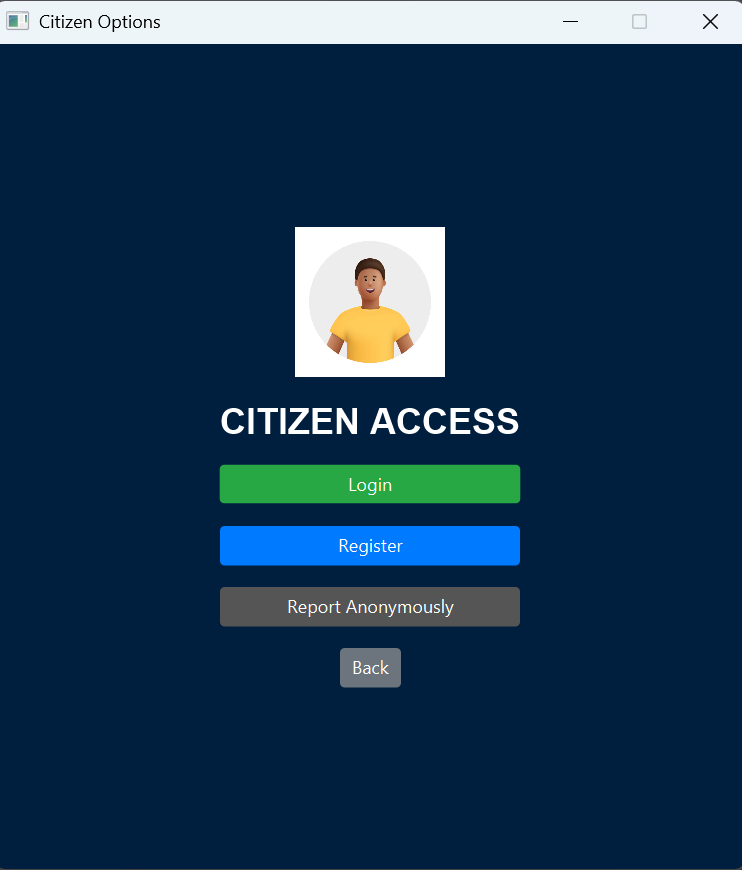
AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

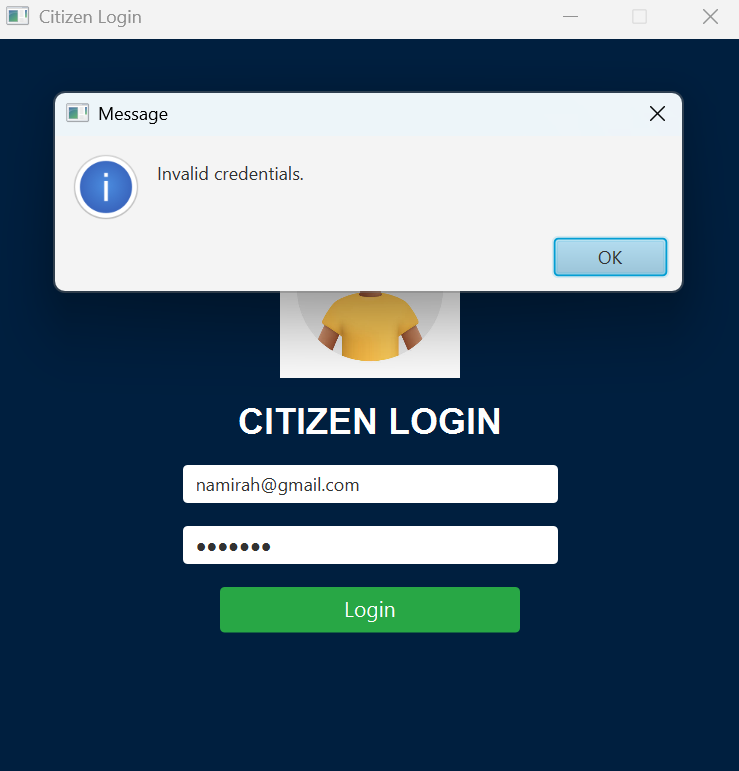
A screenshot of a computer screen

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

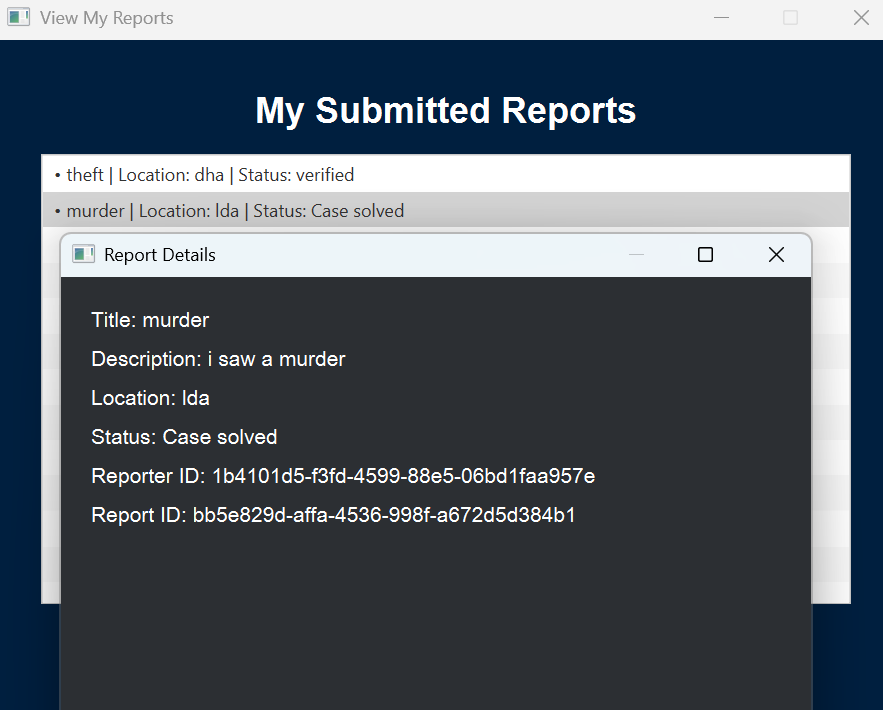
AI-generated content may be incorrect.

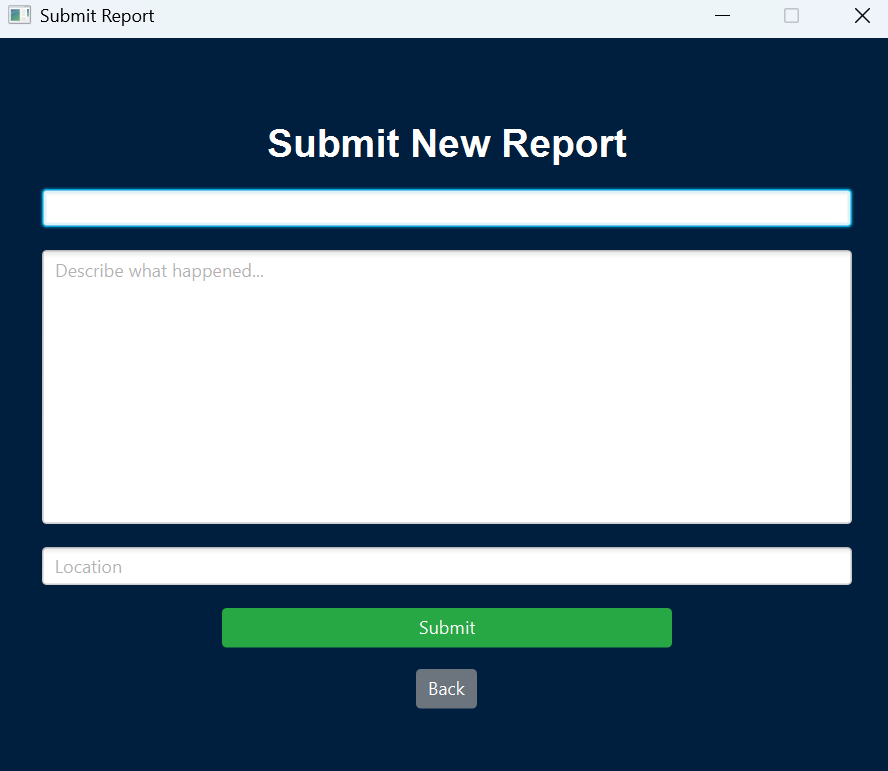
A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.



**CODE**

**User.java**

package models;  
  
public abstract class User {  
 protected String id;  
 protected String name;  
 protected String email;  
 protected String password;  
 protected String role;  
  
 public User(String id, String name, String email, String password, String role) {  
 this.id = id;  
 this.name = name;  
 this.email = email;  
 this.password = password;  
 this.role = role;  
 }  
  
 public String getId() {  
 return id;  
 }  
  
 public String getName() {  
 return name;  
 }  
  
 public String getEmail() {  
 return email;  
 }  
  
 public String getPassword() {  
 return password;  
 }  
  
 public void setPassword(String password) {  
 this.password = password;  
 }  
  
 public String getRole() {  
 return role;  
 }  
  
 public boolean checkPassword(String inputPassword) {  
 return this.password.equals(inputPassword);  
 }  
  
 @Override  
 public String toString() {  
 return "[" + role.toUpperCase() + "] " + name + " <" + email + ">";  
 }  
  
  
 public abstract void displayMenu();  
}

**Admin.java**

package models;  
  
public class Admin extends User {  
 public Admin(String id, String name, String email, String password) {  
 super(id, name, email, password, "admin");  
 }  
  
 @Override  
 public void displayMenu() {  
 System.*out*.println("\n=== Admin Menu ===");  
 System.*out*.println("1. View All Reports");  
 System.*out*.println("2. Update Report Status");  
 System.*out*.println("3. Export Reports");  
 System.*out*.println("4. Create New Police Account");  
 System.*out*.println("0. Exit");  
  
 }  
}

**Citizen.java**

package models;  
  
public class Citizen extends User {  
 public Citizen(String id, String name, String email, String password) {  
 super(id, name, email, password, "citizen");  
 }  
  
 @Override  
 public void displayMenu() {  
 System.*out*.println("\n=== Citizen Menu ===");  
 System.*out*.println("1. Submit Crime Report");  
 System.*out*.println("2. View My Reports");  
 System.*out*.println("3. Emergency Assistance");  
 System.*out*.println("4. Export My Reports");  
 System.*out*.println("0. Exit");  
 }  
}

**Police.java**

package models;  
  
public class Police extends User {  
 public Police(String id, String name, String email, String password) {  
 super(id, name, email, password, “police”);  
 }  
  
 @Override  
 public void displayMenu() {  
 System.*out*.println(“\n=== Police Menu ===”);  
 System.*out*.println(“1. View All Reports”);  
 System.*out*.println(“2. Update Report Status”);  
 System.*out*.println(“3. Export Reports”);  
 System.*out*.println(“0. Exit”);  
 }  
}

**CrimeReport.java**

package models;  
  
import java.time.LocalDateTime;  
import java.util.List;  
  
public class CrimeReport {  
 private String reportId;  
 private String reporterId;  
 private String title;  
 private String description;  
 private String location;  
 private LocalDateTime timestamp;  
 private String status;  
 private List<String> attachments;  
  
 public CrimeReport(String reportId, String reporterId, String title, String description, String location, List<String> attachments) {  
 this.reportId = reportId;  
 this.reporterId = reporterId;  
 this.title = title;  
 this.description = description;  
 this.location = location;  
 this.attachments = attachments;  
 this.timestamp = LocalDateTime.*now*();  
 this.status = “Pending”;  
 }  
  
 public String getReportId() {  
 return reportId;  
 }  
 public String getReporterId() {  
 return reporterId;  
 }  
 public String getTitle() {  
 return title;  
 }  
 public String getDescription() {  
 return description;  
 }  
 public String getLocation() {  
 return location;  
 }  
 public LocalDateTime getTimestamp() {  
 return timestamp;  
 }  
 public String getStatus() {  
 return status;  
 }  
 public List<String> getAttachments() {  
 return attachments;  
 }  
  
  
 public void setStatus(String status) {  
 this.status = status;  
 }  
  
 @Override  
 public String toString() {  
 String attachCount = (attachments != null) ? attachments.size() + “ file(s)” : “No files”;  
 return “[Report: “ + title + “ | Status: “ + status + “ | Media: “ + attachCount + “]”;  
 }  
}

**AuthServices.java**

package services;  
  
import com.google.gson.reflect.TypeToken;  
import models.\*;  
import utils.FileStorage;  
  
import java.io.File;  
import java.lang.reflect.Type;  
import java.util.ArrayList;  
import java.util.List;  
import java.util.UUID;  
  
public class AuthServices {  
 private List<User> users;  
 private final String USER\_FILE = "db/users.json";  
 private final String USER\_LOG\_FILE = "db/user\_log.txt";  
  
 public AuthServices() {  
 Type userListType = new TypeToken<List<User>>() {}.getType();  
 users = FileStorage.*loadFromFile*(USER\_FILE, userListType);  
 if (users == null) {  
 users = new ArrayList<>();  
 }  
 ensureValidUserRoles();  
 }  
  
 private void ensureValidUserRoles() {  
 boolean changed = false;  
  
 if (users == null || users.isEmpty()) {  
 System.*out*.println(" No users found. Adding default admin and police...");  
 users = new ArrayList<>();  
 users.add(new Admin("admin-001", "Admin", "admin@dept.com", "admin123"));  
 users.add(new Police("police-001", "Officer", "police@police.com", "police123"));  
 saveUsers();  
 return;  
 }  
  
 List<User> fixed = new ArrayList<>();  
 for (User u : users) {  
 if (u.getRole() == null || u.getRole().isBlank()) {  
 String email = u.getEmail().toLowerCase();  
 if (email.contains("@police.com")) {  
 u = new Police(u.getId(), u.getName(), u.getEmail(), u.getPassword());  
 } else if (email.contains("@dept.com")) {  
 u = new Admin(u.getId(), u.getName(), u.getEmail(), u.getPassword());  
 } else {  
 u = new Citizen(u.getId(), u.getName(), u.getEmail(), u.getPassword());  
 }  
 changed = true;  
 }  
 fixed.add(u);  
 }  
  
 users = fixed;  
  
 if (changed) {  
 saveUsers();  
 System.*out*.println(" Fixed invalid roles and saved users.");  
 } else {  
 System.*out*.println(" All user roles already valid. No save needed.");  
 }  
 }  
  
 public User register(String name, String email, String password, String role) {  
 if (!role.equalsIgnoreCase("citizen")) {  
 System.*out*.println(" Only citizens can register. Admins and police accounts are department-issued.");  
 return null;  
 }  
  
 for (User user : users) {  
 if (user.getEmail().equalsIgnoreCase(email)) {  
 System.*out*.println(" Email already registered.");  
 return null;  
 }  
 }  
  
 String id = UUID.*randomUUID*().toString();  
 Citizen newUser = new Citizen(id, name, email, password);  
 users.add(newUser);  
 System.*out*.println(" Registering new user: " + newUser.getEmail());  
 saveUsers();  
 System.*out*.println(" saveUsers() called.");  
  
 String logLine = " User registered: " + name + " (citizen), Email: " + email + "\n";  
 FileStorage.*writeToFile*(new File(USER\_LOG\_FILE), logLine);  
  
 return newUser;  
 }  
  
 public User login(String email, String password) {  
 for (User user : users) {  
 if (user.getEmail().equalsIgnoreCase(email) && user.checkPassword(password)) {  
 switch (user.getRole().toLowerCase()) {  
 case "admin" -> {  
 return new Admin(user.getId(), user.getName(), user.getEmail(), user.getPassword());  
 }  
 case "police" -> {  
 return new Police(user.getId(), user.getName(), user.getEmail(), user.getPassword());  
 }  
 case "citizen" -> {  
 return new Citizen(user.getId(), user.getName(), user.getEmail(), user.getPassword());  
 }  
 }  
 }  
 }  
 return null;  
 }  
  
 public void saveUsers() {  
 System.*out*.println(" Saving users: " + users.size());  
 FileStorage.*saveToFile*(users, USER\_FILE);  
 System.*out*.println(" Users saved to: " + USER\_FILE);  
 }  
  
 public List<User> getAllUsers() {  
 return users;  
 }  
 public boolean deleteUserByEmail(String email) {  
 for (int i = 0; i < users.size(); i++) {  
 if (users.get(i).getEmail().equalsIgnoreCase(email)) {  
 users.remove(i);  
 saveUsers();  
 return true;  
 }  
 }  
 return false;  
 }  
 public boolean removeUser(String email) {  
 return users.removeIf(user -> user.getEmail().equalsIgnoreCase(email));  
 }  
}

**CrimeServices.java**

package services;  
  
import com.google.gson.reflect.TypeToken;  
import models.\*;  
import utils.FileStorage;  
  
import java.io.File;  
import java.lang.reflect.Type;  
import java.nio.file.Files;  
import java.nio.file.Paths;  
import java.util.ArrayList;  
import java.util.List;  
import java.util.UUID;  
  
public class CrimeServices {  
 private List<CrimeReport> reports;  
 private final String REPORT\_FILE = “db/reports.json”;  
 private final String REPORT\_LOG\_FILE = “db/report\_log.txt”;  
  
 public CrimeServices() {  
 Type reportListType = new TypeToken<List<CrimeReport>>() {}.getType();  
 reports = FileStorage.*loadFromFile*(REPORT\_FILE, reportListType);  
 if (reports == null) {  
 reports = new ArrayList<>();  
 }  
 }  
  
  
 public CrimeReport submitReport(String reporterId, String title, String description, String location, List<String> attachments) {  
 String id = UUID.*randomUUID*().toString();  
 CrimeReport report = new CrimeReport(id, reporterId, title, description, location, attachments);  
 reports.add(report);  
 saveReports();  
  
 String logLine = “ Report submitted: “ + title + “ | By: “ + reporterId + “ | Status: Pending\n”;  
 FileStorage.*writeToFile*(new File(REPORT\_LOG\_FILE), logLine);  
  
 return report;  
 }  
  
 public CrimeReport submitReport(String reporterId, String title, String description, String location) {  
 return submitReport(reporterId, title, description, location, new ArrayList<>());  
 }  
  
 public List<CrimeReport> getAllReports() {  
 return reports;  
 }  
  
 public List<CrimeReport> getReportsByUser(String userId) {  
 List<CrimeReport> userReports = new ArrayList<>();  
 for (CrimeReport r : reports) {  
 if (r.getReporterId().equals(userId)) {  
 userReports.add(r);  
 }  
 }  
 return userReports;  
 }  
  
 public oolean updateStatus(String reportId, String newStatus) {  
 for (CrimeReport report : reports) {  
 if (report.getReportId().equals(reportId)) {  
 report.setStatus(newStatus);  
 saveReports();  
  
 String logLine = “ Status updated: “ + report.getTitle() + “ → “ + newStatus + “ | ID: “ + reportId + “\n”;  
 FileStorage.*writeToFile*(new File(REPORT\_LOG\_FILE), logLine);  
  
 User user = getUserById(report.getReporterId());  
 if (user != null && !user.getId().equals(“anonymous”)) {  
 NotificationService.*sendEmail*(  
 user.getEmail(),  
 “Report Status Updated”,  
 “Your report titled ‘” + report.getTitle() + “’ is now marked as: “ + newStatus  
 );  
 }  
  
 return true;  
 }  
 }  
 return false;  
 }  
  
 public void exportReportsToTextFile(String filePath) {  
 try {  
 List<CrimeReport> allReports = getAllReports();  
 StringBuilder builder = new StringBuilder();  
  
 for (CrimeReport r : allReports) {  
 builder.append(“Report ID: “).append(r.getReportId()).append(“\n”);  
 builder.append(“Title: “).append(r.getTitle()).append(“\n”);  
 builder.append(“Description: “).append(r.getDescription()).append(“\n”);  
 builder.append(“Location: “).append(r.getLocation()).append(“\n”);  
 builder.append(“Status: “).append(r.getStatus()).append(“\n”);  
 builder.append(“Reporter ID: “).append(r.getReporterId()).append(“\n”);  
 builder.append(“Attachments: “).append(r.getAttachments()).append(“\n”);  
 builder.append(“Timestamp: “).append(r.getTimestamp()).append(“\n”);  
 builder.append(“--------------------------------------------------\n”);  
 }  
  
 Files.*write*(Paths.*get*(filePath), builder.toString().getBytes());  
 System.*out*.println(“ Reports exported to: “ + filePath);  
 } catch (Exception e) {  
 System.*out*.println(“ Failed to export reports: “ + e.getMessage());  
 }  
 }  
 public void exportReportsToTextFile(String filePath, List<CrimeReport> reportList) {  
 try {  
 StringBuilder builder = new StringBuilder();  
  
 for (CrimeReport r : reportList) {  
 builder.append(“Report ID: “).append(r.getReportId()).append(“\n”);  
 builder.append(“Title: “).append(r.getTitle()).append(“\n”);  
 builder.append(“Description: “).append(r.getDescription()).append(“\n”);  
 builder.append(“Location: “).append(r.getLocation()).append(“\n”);  
 builder.append(“Status: “).append(r.getStatus()).append(“\n”);  
 builder.append(“Reporter ID: “).append(r.getReporterId()).append(“\n”);  
 builder.append(“Attachments: “).append(r.getAttachments()).append(“\n”);  
 builder.append(“Timestamp: “).append(r.getTimestamp()).append(“\n”);  
 builder.append(“--------------------------------------------------\n”);  
 }  
  
 java.nio.file.Files.*write*(java.nio.file.Paths.*get*(filePath), builder.toString().getBytes());  
 System.*out*.println(“ Reports exported to: “ + filePath);  
 } catch (Exception e) {  
 System.*out*.println(“ Failed to export reports: “ + e.getMessage());  
 }  
 }  
  
 private void saveReports() {  
 FileStorage.*saveToFile*(reports, REPORT\_FILE);  
 }  
  
 private User getUserById(String id) {  
 AuthServices authService = new AuthServices();  
 for (User u : authService.getAllUsers()) {  
 if (u.getId().equals(id)) {  
 switch (u.getRole().toLowerCase()) {  
 case “admin” -> {  
 return new Admin(u.getId(), u.getName(), u.getEmail(), u.getPassword());  
 }  
 case “police” -> {  
 return new Police(u.getId(), u.getName(), u.getEmail(), u.getPassword());  
 }  
 case “citizen” -> {  
 return new Citizen(u.getId(), u.getName(), u.getEmail(), u.getPassword());  
 }  
 }  
 }  
 }  
 return null;  
 }  
}

**NotificationServices.java**

package services;  
  
public class NotificationService {  
  
 public static void sendEmail(String to, String subject, String message) {  
 System.*out*.println("\n Email Sent To: " + to);  
 System.*out*.println("Subject: " + subject);  
 System.*out*.println("Message: " + message);  
 System.*out*.println("------------------------------");  
 }  
  
 public static void sendSMS(String to, String message) {  
 System.*out*.println("\n SMS Sent To: " + to);  
 System.*out*.println("Message: " + message);  
 System.*out*.println("------------------------------");  
 }  
}

**FileStorage.java**

package utils;  
  
import com.google.gson.\*;  
import models.\*;  
import utils.typeadapters.RuntimeTypeAdapterFactory;  
  
import java.io.\*;  
import java.lang.reflect.Type;  
import java.time.LocalDateTime;  
import java.util.List;  
  
public class FileStorage {  
 private static final RuntimeTypeAdapterFactory<User> *userAdapterFactory* = RuntimeTypeAdapterFactory  
 .*of*(User.class, "role")  
 .registerSubtype(Citizen.class, "citizen")  
 .registerSubtype(Admin.class, "admin")  
 .registerSubtype(Police.class, "police");  
  
 private static final Gson *gson* = new GsonBuilder()  
 .registerTypeAdapter(LocalDateTime.class,  
 (JsonSerializer<LocalDateTime>) (src, typeOfSrc, context) -> new JsonPrimitive(src.toString()))  
 .registerTypeAdapter(LocalDateTime.class,  
 (JsonDeserializer<LocalDateTime>) (json, typeOfT, context) -> LocalDateTime.*parse*(json.getAsString()))  
 .registerTypeAdapterFactory(*userAdapterFactory*)  
 .create();  
  
 public static <T> void saveToFile(List<T> data, String filePath) {  
 try {  
 File file = new File(filePath);  
 File parent = file.getParentFile();  
 if (parent != null && !parent.exists()) {  
 parent.mkdirs();  
 }  
  
 try (FileWriter writer = new FileWriter(file)) {  
 *gson*.toJson(data, writer);  
 }  
 } catch (Exception e) {  
 System.*out*.println(" Error saving to file: " + filePath);  
 System.*out*.println("Reason: " + e.getMessage());  
 }  
 }  
  
 public static <T> List<T> loadFromFile(String filePath, Type typeOfT) {  
 try {  
 File file = new File(filePath);  
 if (!file.exists()) return null;  
  
 try (FileReader reader = new FileReader(file)) {  
 return *gson*.fromJson(reader, typeOfT);  
 }  
 } catch (Exception e) {  
 System.*out*.println("Error loading from file: " + filePath);  
 System.*out*.println("Reason: " + e.getMessage());  
 return null;  
 }  
 }  
  
 public static void writeToFile(File file, String text) {  
 try {  
 FileWriter fileWriter = new FileWriter(file, true);  
 BufferedWriter bufferedWriter = new BufferedWriter(fileWriter);  
 bufferedWriter.write(text);  
 bufferedWriter.close();  
 fileWriter.close();  
 } catch (IOException e) {  
 throw new RuntimeException(e);  
 }  
 }  
  
 public static String readFromFile(File file) {  
 StringBuilder content = new StringBuilder();  
 try {  
 BufferedReader reader = new BufferedReader(new FileReader(file));  
 String line;  
 while ((line = reader.readLine()) != null) {  
 content.append(line).append(System.*lineSeparator*());  
 }  
 reader.close();  
 } catch (IOException e) {  
 System.*out*.println(" Error reading from file: " + file.getAbsolutePath());  
 System.*out*.println("Reason: " + e.getMessage());  
 }  
 return content.toString();  
 }  
}

**AppController.java**

package controller;  
  
import models.\*;  
import services.AuthServices;  
import services.CrimeServices;  
import services.NotificationService;  
import utils.FileStorage;  
  
import java.io.File;  
import java.util.ArrayList;  
import java.util.List;  
import java.util.Scanner;  
import java.util.UUID;  
  
public class AppController {  
 private final Scanner scanner = new Scanner(System.*in*);  
 private final AuthServices authService = new AuthServices();  
 private final CrimeServices crimeService = new CrimeServices();  
 private User currentUser = null;  
  
 public void run() {  
 showWelcomeMenu();  
 showMainMenu();  
 }  
  
 private void showWelcomeMenu() {  
 System.*out*.println(“=== Welcome to Crime Reporting System ===”);  
  
 while (currentUser == null) {  
 System.*out*.println(“\n1. Register”);  
 System.*out*.println(“2. Login”);  
 System.*out*.println(“3. Report a Crime Anonymously”);  
 System.*out*.print(“Choose option: “);  
 int choice = scanner.nextInt();  
 scanner.nextLine();  
  
 switch (choice) {  
 case 1 -> handleRegistration();  
 case 2 -> handleLogin();  
 case 3 -> handleAnonymousReport();  
 default -> System.*out*.println(“Invalid choice. Try again.”);  
 }  
 }  
 }  
  
 private void handleRegistration() {  
 System.*out*.print(“Name: “);  
 String name = scanner.nextLine();  
 System.*out*.print(“Email: “);  
 String email = scanner.nextLine();  
 System.*out*.print(“Password: “);  
 String password = scanner.nextLine();  
 String role = “citizen”;  
  
 currentUser = authService.register(name, email, password, role);  
 if (currentUser != null) {  
 System.*out*.println(“ Registration successful!”);  
 }  
 }  
  
 private void handleLogin() {  
 System.*out*.print(“Email: “);  
 String email = scanner.nextLine();  
 System.*out*.print(“Password: “);  
 String password = scanner.nextLine();  
 currentUser = authService.login(email, password);  
  
 if (currentUser != null) {  
 System.*out*.println(“ Login successful! Welcome “ + currentUser.getName());  
 } else {  
 System.*out*.println(“Invalid email or password.”);  
 }  
 }  
  
 private void handleAnonymousReport() {  
 System.*out*.println(“Anonymous Report Selected”);  
 System.*out*.print(“Title: “);  
 String title = scanner.nextLine();  
 System.*out*.print(“Description: “);  
 String description = scanner.nextLine();  
 System.*out*.print(“Location: “);  
 String location = scanner.nextLine();  
  
 List<String> attachments = readAttachments();  
  
 var report = crimeService.submitReport(“anonymous”, title, description, location, attachments);  
 System.*out*.println(“Anonymous report submitted with ID: “ + report.getReportId());  
  
 FileStorage.*writeToFile*(new File(“db/report\_log.txt”),  
 “Anonymous report submitted: “ + title + “\n”);  
 }  
  
 private void showMainMenu() {  
 if (currentUser instanceof Citizen) {  
 showCitizenMenu();  
 } else if (currentUser instanceof Admin) {  
 showAdminMenu();  
 } else if (currentUser instanceof Police) {  
 showPoliceMenu();  
 }  
 }  
  
 private void showCitizenMenu() {  
 oolean exit = false;  
 while (!exit) {  
 currentUser.displayMenu();  
 System.*out*.print(“Select: “);  
 int option = scanner.nextInt();  
 scanner.nextLine();  
  
 switch (option) {  
 case 1 -> handleRegularReport();  
 case 2 -> handleViewOwnReports();  
 case 3 -> handleEmergencyReport();  
 case 4 -> handleExportMyReports();  
 case 0 -> exit = true;  
 default -> System.*out*.println(“ Invalid option.”);  
 }  
 }  
 }  
  
 private void showAdminMenu() {  
 oolean exit = false;  
 while (!exit) {  
 currentUser.displayMenu();  
 System.*out*.print(“Select: “);  
 int option = scanner.nextInt();  
 scanner.nextLine();  
  
 switch (option) {  
 case 1 -> handleViewAllReports();  
 case 2 -> handleStatusUpdate();  
 case 3 -> handleExportReports();  
 case 4 -> handleCreatePoliceAccount();  
 case 0 -> exit = true;  
 default -> System.*out*.println(“ Invalid option.”);  
 }  
 }  
 }  
  
 private void showPoliceMenu() {  
 oolean exit = false;  
 while (!exit) {  
 currentUser.displayMenu();  
 System.*out*.print(“Select: “);  
 int option = scanner.nextInt();  
 scanner.nextLine();  
  
 switch (option) {  
 case 1 -> handleViewAllReports();  
 case 2 -> handleStatusUpdate();  
 case 3 -> handleExportReports();  
 case 0 -> exit = true;  
 default -> System.*out*.println(“ Invalid option.”);  
 }  
 }  
 }  
  
 public void handleRegularReport() {  
 System.*out*.print(“Title: “);  
 String title = scanner.nextLine();  
 System.*out*.print(“Description: “);  
 String description = scanner.nextLine();  
 System.*out*.print(“Location: “);  
 String location = scanner.nextLine();  
  
 List<String> attachments = readAttachments();  
  
 var report = crimeService.submitReport(currentUser.getId(), title, description, location, attachments);  
 System.*out*.println(“ Report submitted with ID: “ + report.getReportId());  
  
 NotificationService.*sendEmail*(currentUser.getEmail(), “Report Submitted”,  
 “Your report titled ‘” + title + “’ has been successfully submitted.”);  
  
 FileStorage.*writeToFile*(new File(“db/report\_log.txt”),  
 “Report submitted: “ + title + “ | By: “ + currentUser.getEmail() + “\n”);  
 }  
  
 public void handleViewOwnReports() {  
 var reports = crimeService.getReportsByUser(currentUser.getId());  
 if (reports.isEmpty()) {  
 System.*out*.println(“No reports found.”);  
 } else {  
 reports.forEach(System.*out*::println);  
 }  
 }  
  
 public void handleExportMyReports() {  
 List<CrimeReport> myReports = crimeService.getReportsByUser(currentUser.getId());  
 if (myReports.isEmpty()) {  
 System.*out*.println(“No reports to export.”);  
 return;  
 }  
  
 System.*out*.print(“Enter filename to export (e.g., reports.txt): “);  
 String filename = scanner.nextLine();  
 if (!filename.endsWith(“.txt”)) {  
 filename += “.txt”;  
 }  
  
 String fullPath = “db/” + filename;  
 crimeService.exportReportsToTextFile(fullPath, myReports);  
 System.*out*.println(“ Your report(s) have been saved to: “ + fullPath);  
  
 System.*out*.print(“Do you want to view the exported file now? (yes/no): “);  
 if (scanner.nextLine().equalsIgnoreCase(“yes”)) {  
 String content = FileStorage.*readFromFile*(new File(fullPath));  
 System.*out*.println(“\n=== Exported Report Content ===\n” + content);  
 }  
 }  
  
 public void handleViewAllReports() {  
 crimeService.getAllReports().forEach(System.*out*::println);  
 }  
  
 public void handleStatusUpdate() {  
 System.*out*.print(“Enter Report ID: “);  
 String reportId = scanner.nextLine();  
 System.*out*.print(“New Status (Verified/Rejected): “);  
 String newStatus = scanner.nextLine();  
 oolean updated = crimeService.updateStatus(reportId, newStatus);  
 System.*out*.println(updated ? “ Status updated.” : “Report not found.”);  
 }  
  
 public void handleEmergencyReport() {  
 System.*out*.print(“Brief description of emergency: “);  
 String emergencyDesc = scanner.nextLine();  
 String emergencyLocation = “User’s current location (not tracked in console version)”;  
 List<String> emergencyFiles = readAttachments();  
  
 var emergencyReport = crimeService.submitReport(currentUser.getId(), “Emergency Alert”,  
 emergencyDesc, emergencyLocation, emergencyFiles);  
  
 System.*out*.println(“Emergency alert submitted! Report ID: “ + emergencyReport.getReportId());  
  
 NotificationService.*sendSMS*(currentUser.getEmail(),  
 “Emergency alert submitted. Our team will respond shortly.”);  
  
 FileStorage.*writeToFile*(new File(“db/report\_log.txt”),  
 “Emergency alert filed by: “ + currentUser.getEmail() + “ | “ + emergencyDesc + “\n”);  
 }  
  
 public void handleExportReports() {  
 System.*out*.print(“Enter filename to export (e.g., reports.txt): “);  
 String filename = scanner.nextLine();  
 if (!filename.endsWith(“.txt”)) {  
 filename += “.txt”;  
 }  
 crimeService.exportReportsToTextFile(“db/” + filename);  
 }  
  
 public void handleCreatePoliceAccount() {  
 System.*out*.println(“Create New Police Account”);  
 System.*out*.print(“Name: “);  
 String name = scanner.nextLine();  
 System.*out*.print(“Email: “);  
 String email = scanner.nextLine();  
 System.*out*.print(“Password: “);  
 String password = scanner.nextLine();  
  
 User newPolice = new Police(UUID.*randomUUID*().toString(), name, email, password);  
 authService.getAllUsers().add(newPolice);  
  
 System.*out*.println(“Saving police to users.json...”);  
 authService.saveUsers();  
 FileStorage.*writeToFile*(new File(“db/user\_log.txt”),  
 “Police registered by Admin: “ + name + “ (“ + email + “)\n”);  
 System.*out*.println(“ Police account created.”);  
 }  
  
 private List<String> readAttachments() {  
 List<String> files = new ArrayList<>();  
 System.*out*.print(“Enter number of media files to attach: “);  
 int count = scanner.nextInt();  
 scanner.nextLine();  
 for (int i = 1; i <= count; i++) {  
 System.*out*.print(“Enter file name #” + i + “: “);  
 files.add(scanner.nextLine());  
 }  
 return files;  
 }  
}

**Main.java**

import controller.AppController;  
  
public class Main {  
 public static void main(String[] args) {  
 AppController app = new AppController();  
 app.run();  
 }  
}

**Session.java**

package session;  
  
import models.User;  
  
public class Session {  
 private static User *currentUser*;  
  
 public static void setCurrentUser(User user) {  
 *currentUser* = user;  
 }  
  
 public static User getCurrentUser() {  
 return *currentUser*;  
 }  
}

**RoleSelectionScreen.java**

package gui;  
  
import javafx.scene.Scene;  
import javafx.scene.control.Button;  
import javafx.scene.image.Image;  
import javafx.scene.image.ImageView;  
import javafx.scene.layout.Pane;  
import javafx.stage.Stage;  
import javafx.stage.StageStyle;  
  
public class RoleSelectionScreen {  
  
 public void start(Stage stage) {  
 //stage.initStyle(StageStyle.DECORATED);  
 ImageView background = new ImageView(  
 new Image(getClass().getResource("/gui/role\_selection\_bg.png").toExternalForm())  
 );  
 background.setPreserveRatio(false);  
 background.setFitWidth(768);  
 background.setFitHeight(720);  
  
 Button adminBtn = buildButton("ADMIN", 100, 600);  
 adminBtn.setOnAction(e -> new LoginRegisterScreen("admin", "login").start(stage));  
  
 Button policeBtn = buildButton("POLICE", 310, 600);  
 policeBtn.setOnAction(e -> new LoginRegisterScreen("police", "login").start(stage));  
  
 Button citizenBtn = buildButton("CITIZEN", 510, 600);  
 citizenBtn.setOnAction(e -> new CitizenLoginOrRegisterScreen().start(stage));  
  
 Pane root = new Pane();  
 root.getChildren().addAll(background, adminBtn, policeBtn, citizenBtn);  
  
 Scene scene = new Scene(root, 768, 720);  
  
 stage.setTitle("Police Command Center");  
 stage.setScene(scene);  
 stage.setResizable(false);  
 stage.show();  
 }  
  
 private Button buildButton(String label, double x, double y) {  
 Button btn = new Button(label);  
 btn.setLayoutX(x);  
 btn.setLayoutY(y);  
 btn.setPrefWidth(150);  
 btn.setPrefHeight(60);  
  
 btn.setStyle("""  
 -fx-background-color: transparent;  
 -fx-text-fill: white;  
 -fx-font-weight: bold;  
 -fx-font-size: 18px;  
 -fx-border-color: transparent;  
 -fx-text-alignment: center;  
 -fx-font-family: 'Arial', 'Segoe UI', sans-serif;  
 """);  
  
 return btn;  
 }  
}

**AdminDashboard.java**

package gui;  
  
import controller.AppController;  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.Button;  
import javafx.scene.image.Image;  
import javafx.scene.image.ImageView;  
import javafx.scene.layout.VBox;  
import javafx.scene.text.Font;  
import javafx.scene.text.FontWeight;  
import javafx.scene.text.Text;  
import javafx.stage.Stage;  
  
public class AdminDashboard {  
  
 private final AppController controller = new AppController();  
  
 public void start(Stage stage) {  
 ImageView icon = new ImageView(new Image("file:/C://OOP%20PROJECT//CrimeReportingSystem//src//gui//admin\_icon.png"));  
 icon.setFitHeight(100);  
 icon.setPreserveRatio(true);  
  
 Text title = new Text("Admin Dashboard");  
 title.setFont(Font.*font*("Arial", FontWeight.*BOLD*, 26));  
 title.setStyle("-fx-fill: WHITE;");  
  
 Button btn1 = createButton("Create Police Account", () -> new CreatePoliceScreen(controller).start(SceneNavigator.*getStage*()));  
 Button btn2 = createButton("Manage Users", () -> new ManageUsersScreen().start(SceneNavigator.*getStage*()));  
 Button btn3 = createButton("View All Reports", () -> new ViewAllReportsScreen(controller, "admin").start(SceneNavigator.*getStage*()));  
 Button btn4 = createButton("Update Report Status", () -> new StatusUpdateScreen(controller,"admin").start(SceneNavigator.*getStage*()));  
 Button btn5 =createButton("Exit", () -> SceneNavigator.*getStage*().close());  
  
 VBox layout = new VBox(15, icon, title, btn1, btn2, btn3, btn4,btn5);  
 layout.setAlignment(Pos.*CENTER*);  
 layout.setPadding(new Insets(40));  
 layout.setStyle("-fx-background-color: #001f3f;");  
  
 Scene scene = new Scene(layout, 600, 600);  
 SceneNavigator.*setScene*(scene, "Admin Dashboard");  
  
 btn1.setStyle("-fx-background-color: #87CEFA; -fx-text-fill: white;");  
 btn2.setStyle("-fx-background-color: #87CEFA; -fx-text-fill: white;");  
 btn3.setStyle("-fx-background-color: #87CEFA; -fx-text-fill: white;");  
 btn4.setStyle("-fx-background-color: #87CEFA; -fx-text-fill: white;");  
 btn5.setStyle("-fx-background-color: #dc3545; -fx-text-fill: white;");  
 }  
  
 private Button createButton(String text, Runnable action) {  
 Button btn = new Button(text);  
 btn.setMinWidth(250);  
 btn.setOnAction(e -> action.run());  
 return btn;  
 }  
}

**CitizenDashboard.java**

package gui;  
  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.Button;  
import javafx.scene.image.Image;  
import javafx.scene.image.ImageView;  
import javafx.scene.layout.VBox;  
import javafx.scene.paint.Color;  
import javafx.scene.text.Font;  
import javafx.scene.text.FontWeight;  
import javafx.scene.text.Text;  
import javafx.stage.Stage;  
import session.Session;  
  
public class CitizenDashboard {  
  
 public void start(Stage stage) {  
 ImageView avatar = new ImageView(new Image("file:/C://OOP%20PROJECT//CrimeReportingSystem//src//gui//citizen\_icon.png"));  
 avatar.setFitHeight(100);  
 avatar.setPreserveRatio(true);  
  
 Text title = new Text("Citizen Dashboard");  
 title.setFont(Font.*font*("Arial", FontWeight.*BOLD*, 26));  
 title.setFill(Color.*WHITE*);  
  
 Button submitBtn = new Button("Submit New Report");  
 submitBtn.setMinWidth(250);  
 submitBtn.setStyle("-fx-background-color: #87CEFA; -fx-text-fill: white; -fx-font-size: 14;");  
  
 Button viewBtn = new Button("View My Reports");  
 viewBtn.setMinWidth(250);  
 viewBtn.setStyle("-fx-background-color: #87CEFA; -fx-text-fill: white; -fx-font-size: 14;");  
  
 Button logoutBtn = new Button("Logout");  
 logoutBtn.setMinWidth(250);  
 logoutBtn.setStyle("-fx-background-color: #dc3545; -fx-text-fill: white; -fx-font-size: 14;");  
  
 submitBtn.setOnAction(e -> new SubmitReportScreen().start(SceneNavigator.*getStage*()));  
 viewBtn.setOnAction(e -> new ViewReportsScreen().start(SceneNavigator.*getStage*()));  
 logoutBtn.setOnAction(e -> {  
 Session.*setCurrentUser*(null);  
 new RoleSelectionScreen().start(SceneNavigator.*getStage*());  
 });  
  
 VBox layout = new VBox(20, avatar, title, submitBtn, viewBtn, logoutBtn);  
 layout.setAlignment(Pos.*CENTER*);  
 layout.setPadding(new Insets(40));  
 layout.setStyle("-fx-background-color: #001f3f;");  
  
 Scene scene = new Scene(layout, 600, 500);  
 SceneNavigator.*setScene*(scene, "Citizen Dashboard");  
 }  
}

**PoliceDashboard.java**

package gui;  
  
import controller.AppController;  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.Button;  
import javafx.scene.image.Image;  
import javafx.scene.image.ImageView;  
import javafx.scene.layout.VBox;  
import javafx.scene.text.Font;  
import javafx.scene.text.FontWeight;  
import javafx.scene.text.Text;  
import javafx.stage.Stage;  
  
public class PoliceDashboard {  
  
 private final AppController controller = new AppController();  
  
 public void start(Stage stage) {  
 ImageView icon = new ImageView(new Image("file:/C://OOP%20PROJECT//CrimeReportingSystem//src//gui//police\_icon.png"));  
 icon.setFitHeight(100);  
 icon.setPreserveRatio(true);  
  
 Text title = new Text("Police Dashboard");  
 title.setFont(Font.*font*("Arial", FontWeight.*BOLD*, 26));  
 title.setStyle("-fx-fill: white;");  
  
 Button viewBtn = createButton("View All Reports", () -> new ViewAllReportsScreen(controller, "police").start(SceneNavigator.*getStage*()));  
 Button updateBtn = createButton("Update Report Status", () -> new StatusUpdateScreen(controller,"police").start(SceneNavigator.*getStage*()));  
 Button emergencyBtn = createButton("Handle Emergency Reports", () -> new HandleEmergencyScreen(controller).start(SceneNavigator.*getStage*()));  
 Button exitBtn = createButton("Exit", () -> SceneNavigator.*getStage*().close());  
  
 VBox layout = new VBox(15, icon, title, viewBtn, updateBtn, emergencyBtn, exitBtn);  
 layout.setAlignment(Pos.*CENTER*);  
 layout.setPadding(new Insets(40));  
 layout.setStyle("-fx-background-color: #001f3f;");  
  
 Scene scene = new Scene(layout, 600, 600);  
 SceneNavigator.*setScene*(scene, "Police Dashboard");  
  
 viewBtn.setStyle("-fx-background-color: #87CEFA; -fx-text-fill: white;");  
 updateBtn.setStyle("-fx-background-color: #87CEFA; -fx-text-fill: white;");  
 emergencyBtn.setStyle("-fx-background-color: #87CEFA; -fx-text-fill: white;");  
 exitBtn.setStyle("-fx-background-color: #dc3545; -fx-text-fill: white;");  
 }  
  
 private Button createButton(String text, Runnable action) {  
 Button btn = new Button(text);  
 btn.setMinWidth(250);  
 btn.setOnAction(e -> action.run());  
 return btn;  
 }  
  
}

**CitizenLoginOrRegisterScreen.java**

package gui;  
  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.Button;  
import javafx.scene.image.Image;  
import javafx.scene.image.ImageView;  
import javafx.scene.layout.VBox;  
import javafx.scene.text.Font;  
import javafx.scene.text.FontWeight;  
import javafx.scene.text.Text;  
import javafx.stage.Stage;  
import javafx.scene.paint.Color;  
import models.Citizen;  
import session.Session;  
  
public class CitizenLoginOrRegisterScreen {  
  
 public void start(Stage stage) {  
 ImageView avatar = new ImageView(new Image(“file:/C://OOP%20PROJECT//CrimeReportingSystem//src//gui//citizen\_icon.png”));  
 avatar.setFitHeight(100);  
 avatar.setPreserveRatio(true);  
  
 Text title = new Text(“CITIZEN ACCESS”);  
 title.setFont(Font.*font*(“Arial”, FontWeight.*BOLD*, 24));  
 title.setFill(Color.*WHITE*);  
  
  
 Button loginBtn = new Button(“Login”);  
 loginBtn.setMinWidth(200);  
 loginBtn.setStyle(“-fx-background-color: #28a745; -fx-text-fill: white;”);  
  
 Button registerBtn = new Button(“Register”);  
 registerBtn.setMinWidth(200);  
 registerBtn.setStyle(“-fx-background-color: #007bff; -fx-text-fill: white;”);  
  
 Button anonymousBtn = new Button(“Report Anonymously”);  
 anonymousBtn.setMinWidth(200);  
 anonymousBtn.setStyle(“-fx-background-color: #555555; -fx-text-fill: white;”);  
  
 Button backBtn = new Button(“Back”);  
 backBtn.setStyle(“-fx-background-color: #6c757d; -fx-text-fill: white;”);  
 backBtn.setOnAction(e -> new RoleSelectionScreen().start(SceneNavigator.*getStage*()));  
  
  
 loginBtn.setOnAction(e -> new LoginRegisterScreen(“citizen”, “login”).start(SceneNavigator.*getStage*()));  
 registerBtn.setOnAction(e -> new LoginRegisterScreen(“citizen”, “register”).start(SceneNavigator.*getStage*()));  
  
 anonymousBtn.setOnAction(e -> {  
 Citizen anon = new Citizen(“anonymous”, “Anonymous”, “none”, “none”);  
 Session.*setCurrentUser*(anon);  
 new SubmitReportScreen().start(SceneNavigator.*getStage*());  
 });  
  
 Vbox layout = new Vbox(15, avatar, title, loginBtn, registerBtn, anonymousBtn, backBtn);  
 layout.setAlignment(Pos.*CENTER*);  
 layout.setPadding(new Insets(30));  
 layout.setStyle(“-fx-background-color:#001f3f;”);  
  
 Scene scene = new Scene(layout, 500, 550);  
 SceneNavigator.*setScene*(scene, “Citizen Options”);  
 }  
}

**ViewReportsScreen.java**

package gui;  
  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.\*;  
import javafx.scene.layout.VBox;  
import javafx.scene.paint.Color;  
import javafx.scene.text.Font;  
import javafx.scene.text.FontWeight;  
import javafx.stage.Stage;  
  
import java.util.List;  
  
import models.Citizen;  
import models.CrimeReport;  
import services.CrimeServices;  
import session.Session;  
  
public class ViewReportsScreen {  
  
 public void start(Stage stage) {  
 Label title = new Label(“My Submitted Reports”);  
 title.setFont(Font.*font*(“Arial”, FontWeight.*BOLD*, 24));  
 title.setTextFill(Color.*WHITE*);  
  
 ListView<String> reportList = new ListView<>();  
 reportList.setPrefSize(500, 300);  
  
 List<CrimeReport> reports;  
 if (!(Session.*getCurrentUser*() instanceof Citizen)) {  
 reportList.getItems().add(“Error: Only citizens can view their reports.”);  
 reports = null;  
 } else {  
 Citizen citizen = (Citizen) Session.*getCurrentUser*();  
 reports = new CrimeServices().getReportsByUser(citizen.getId());  
  
 if (reports.isEmpty()) {  
 reportList.getItems().add(“No reports submitted yet.”);  
 } else {  
 for (CrimeReport report : reports) {  
 String display = “• “ + report.getTitle() +  
 “ | Location: “ + report.getLocation() +  
 “ | Status: “ + report.getStatus();  
 reportList.getItems().add(display);  
 }  
 }  
 }  
  
 reportList.setOnMouseClicked(e -> {  
 int index = reportList.getSelectionModel().getSelectedIndex();  
 if (index >= 0 && reports != null && index < reports.size()) {  
 ReportDetailPopup.*show*(reports.get(index));  
 }  
 });  
  
 Button backBtn = new Button(“Back”);  
 backBtn.setStyle(“-fx-background-color: #6c757d; -fx-text-fill: white;”);  
 backBtn.setOnAction(e -> new CitizenDashboard().start(SceneNavigator.*getStage*()));  
  
 Vbox layout = new Vbox(15, title, reportList, backBtn);  
 layout.setAlignment(Pos.*CENTER*);  
 layout.setPadding(new Insets(30));  
 layout.setStyle(“-fx-background-color: #001f3f;”);  
  
 Scene scene = new Scene(layout, 600, 450);  
 SceneNavigator.*setScene*(scene, “View My Reports”);  
 }  
}

**SubmitReportScreen.java**

package gui;  
  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.\*;  
import javafx.scene.layout.VBox;  
import javafx.scene.paint.Color;  
import javafx.scene.text.Font;  
import javafx.scene.text.FontWeight;  
import javafx.stage.Stage;  
  
import models.Citizen;  
import services.CrimeServices;  
import session.Session;  
  
import java.util.Collections;  
  
public class SubmitReportScreen {  
  
 public void start(Stage stage) {  
 Label title = new Label(“Submit New Report”);  
 title.setFont(Font.*font*(“Arial”, FontWeight.*BOLD*, 26));  
 title.setTextFill(Color.*WHITE*);  
  
 TextField reportTitle = new TextField();  
 reportTitle.setPromptText(“Report Title”);  
  
 TextArea description = new TextArea();  
 description.setPromptText(“Describe what happened...”);  
 description.setWrapText(true);  
  
 TextField location = new TextField();  
 location.setPromptText(“Location”);  
  
 Button submitBtn = new Button(“Submit”);  
 submitBtn.setStyle(“-fx-background-color: #28a745; -fx-text-fill: white;”);  
 submitBtn.setMinWidth(300);  
  
 Button backBtn = new Button(“Back”);  
 backBtn.setStyle(“-fx-background-color: #6c757d; -fx-text-fill: white;”);  
  
 submitBtn.setOnAction(e -> {  
 String titleText = reportTitle.getText();  
 String desc = description.getText();  
 String loc = location.getText();  
  
 if (titleText.isEmpty() || desc.isEmpty() || loc.isEmpty()) {  
 showAlert(“Please fill all fields.”);  
 return;  
 }  
  
 if (!(Session.*getCurrentUser*() instanceof Citizen)) {  
 showAlert(“Only citizens can submit reports.”);  
 return;  
 }  
  
 Citizen citizen = (Citizen) Session.*getCurrentUser*();  
 new CrimeServices().submitReport(  
 citizen.getId(), titleText, desc, loc, Collections.*emptyList*()  
 );  
  
 showAlert(“Report submitted successfully.”);  
 reportTitle.clear();  
 description.clear();  
 location.clear();  
 });  
  
 backBtn.setOnAction(e -> {  
 if (Session.*getCurrentUser*() != null && “anonymous”.equals(Session.*getCurrentUser*().getId())) {  
 new CitizenLoginOrRegisterScreen().start(SceneNavigator.*getStage*());  
 } else {  
 new CitizenDashboard().start(SceneNavigator.*getStage*());  
 }  
 });  
  
  
 Vbox layout = new Vbox(15, title, reportTitle, description, location, submitBtn, backBtn);  
 layout.setAlignment(Pos.*CENTER*);  
 layout.setPadding(new Insets(30));  
 layout.setStyle(“-fx-background-color: #001f3f;”);  
  
 Scene scene = new Scene(layout, 600, 500);  
 SceneNavigator.*setScene*(scene, “Submit Report”);  
 }  
  
 private void showAlert(String msg) {  
 Alert alert = new Alert(Alert.AlertType.*INFORMATION*);  
 alert.setHeaderText(null);  
 alert.setContentText(msg);  
 alert.showAndWait();  
 }  
}

**LoginRegisterScreen.java**

package gui;  
  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.\*;  
import javafx.scene.image.Image;  
import javafx.scene.image.ImageView;  
import javafx.scene.layout.VBox;  
import javafx.scene.paint.Color;  
import javafx.scene.text.Font;  
import javafx.scene.text.FontWeight;  
import javafx.stage.Stage;  
  
import models.User;  
import services.AuthServices;  
import session.Session;  
  
public class LoginRegisterScreen {  
 private final String role;  
 private final String mode;  
  
 public LoginRegisterScreen(String role, String mode) {  
 this.role = role.toLowerCase();  
 this.mode = mode.toLowerCase();  
 }  
  
 public void start(Stage stage) {  
 // Avatar  
 ImageView avatar = new ImageView();  
 if (role.equals(“admin”)) {  
 avatar.setImage(new Image(“file:/C://OOP%20PROJECT//CrimeReportingSystem//src//gui//admin\_icon.png”));  
 } else if (role.equals(“police”)) {  
 avatar.setImage(new Image(“file:/C://OOP%20PROJECT//CrimeReportingSystem//src//gui//police\_icon.png”));  
 } else {  
 avatar.setImage(new Image(“file:/C://OOP%20PROJECT//CrimeReportingSystem//src//gui//citizen\_icon.png”));  
 }  
 avatar.setFitHeight(120);  
 avatar.setPreserveRatio(true);  
  
 Label title = new Label(role.toUpperCase() + (mode.equals(“register”) ? “ REGISTER” : “ LOGIN”));  
 title.setFont(Font.*font*(“Arial”, FontWeight.*BOLD*, 24));  
 title.setTextFill(Color.*WHITE*);  
  
 TextField nameField = new TextField();  
 nameField.setPromptText(“Enter your name”);  
 styleField(nameField);  
  
 TextField emailField = new TextField();  
 emailField.setPromptText(“Enter your email”);  
 styleField(emailField);  
  
 PasswordField passwordField = new PasswordField();  
 passwordField.setPromptText(“Enter your password”);  
 styleField(passwordField);  
  
 Button loginBtn = new Button(“Login”);  
 loginBtn.setMinWidth(200);  
 loginBtn.setStyle(“-fx-background-color: #28a745; -fx-text-fill: white; -fx-font-size: 14;”);  
  
 Button registerBtn = new Button(“Register”);  
 registerBtn.setMinWidth(200);  
 registerBtn.setStyle(“-fx-background-color: #007bff; -fx-text-fill: white; -fx-font-size: 14;”);  
  
 loginBtn.setOnAction(e -> {  
 String email = emailField.getText().trim();  
 String password = passwordField.getText().trim();  
  
 if (email.isEmpty() || password.isEmpty()) {  
 showAlert(“Please enter email and password.”);  
 return;  
 }  
  
 AuthServices auth = new AuthServices();  
 User user = auth.login(email, password);  
  
 if (user != null) {  
 if (!user.getRole().equalsIgnoreCase(role)) {  
 showAlert(“Incorrect role selected.”);  
 return;  
 }  
  
 Session.*setCurrentUser*(user);  
 switch (role) {  
 case “citizen”:  
 new CitizenDashboard().start(SceneNavigator.*getStage*());  
 break;  
 case “police”:  
 new PoliceDashboard().start(SceneNavigator.*getStage*());  
 break;  
 case “admin”:  
 new AdminDashboard().start(SceneNavigator.*getStage*());  
 break;  
 }  
 } else {  
 showAlert(“Invalid credentials.”);  
 }  
 });  
  
 registerBtn.setOnAction(e -> {  
 String name = nameField.getText().trim();  
 String email = emailField.getText().trim();  
 String password = passwordField.getText().trim();  
  
 if (name.isEmpty() || email.isEmpty() || password.isEmpty()) {  
 showAlert(“Please enter name, email, and password.”);  
 return;  
 }  
  
 AuthServices auth = new AuthServices();  
 User registeredUser = auth.register(name, email, password, role);  
  
 if (registeredUser != null) {  
 showAlert(“Registered successfully. Now login.”);  
 new LoginRegisterScreen(role, “login”).start(SceneNavigator.*getStage*());  
 } else {  
 showAlert(“User already exists.”);  
 }  
 });  
  
 Vbox layout = new Vbox(15);  
 layout.setAlignment(Pos.*CENTER*);  
 layout.setPadding(new Insets(50));  
 layout.setStyle(“-fx-background-color: #001f3f;”);  
  
 if (role.equals(“citizen”) && mode.equals(“register”)) {  
 layout.getChildren().addAll(avatar, title, nameField, emailField, passwordField, registerBtn);  
 } else {  
 layout.getChildren().addAll(avatar, title, emailField, passwordField, loginBtn);  
 }  
  
 Scene scene = new Scene(layout, 500, 500);  
 SceneNavigator.*setScene*(scene, capitalize(role) + “ “ + capitalize(mode));  
 }  
  
 private void styleField(TextField field) {  
 field.setMaxWidth(250);  
 field.setStyle(“-fx-background-color: white; -fx-border-color: navyblue; -fx-border-radius: 5;”);  
 }  
  
 private void showAlert(String msg) {  
 Alert alert = new Alert(Alert.AlertType.*INFORMATION*);  
 alert.setHeaderText(null);  
 alert.setContentText(msg);  
 alert.showAndWait();  
 }  
  
 private String capitalize(String s) {  
 return s.substring(0, 1).toUpperCase() + s.substring(1);  
 }  
}

**CreatePoliceScreen.java**

package gui;  
  
import controller.AppController;  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.\*;  
import javafx.scene.layout.VBox;  
import javafx.scene.text.Font;  
import javafx.scene.text.FontWeight;  
import javafx.stage.Stage;  
import models.Police;  
import services.AuthServices;  
import utils.FileStorage;  
  
import java.io.File;  
import java.util.UUID;  
  
public class CreatePoliceScreen {  
  
 private final AuthServices authService = new AuthServices();  
  
 public CreatePoliceScreen(AppController controller) {}  
  
 public void start(Stage stage) {  
 Label title = new Label(“Create Police Account”);  
 title.setStyle(“-fx-font-size: 20; -fx-font-weight: bold; -fx-text-fill: white;”);  
  
 TextField nameField = new TextField();  
 nameField.setPromptText(“Name”);  
  
 TextField emailField = new TextField();  
 emailField.setPromptText(“Email”);  
  
 PasswordField passwordField = new PasswordField();  
 passwordField.setPromptText(“Password”);  
  
 Button createBtn = new Button(“Create Account”);  
 createBtn.setStyle(“-fx-background-color: #28a745; -fx-text-fill: white;”);  
 createBtn.setOnAction(e -> {  
 String name = nameField.getText().trim();  
 String email = emailField.getText().trim();  
 String password = passwordField.getText().trim();  
  
 if (name.isEmpty() || email.isEmpty() || password.isEmpty()) {  
 showAlert(“All fields are required.”);  
 return;  
 }  
  
 Police newPolice = new Police(UUID.*randomUUID*().toString(), name, email, password);  
 authService.getAllUsers().add(newPolice);  
 authService.saveUsers();  
 FileStorage.*writeToFile*(new File(“db/user\_log.txt”),  
 “Police registered by Admin: “ + name + “ (“ + email + “)\n”);  
  
 showAlert(“Police account created successfully!”);  
 new AdminDashboard().start(SceneNavigator.*getStage*());  
 });  
  
 Button backBtn = new Button(“Back”);  
 backBtn.setOnAction(e -> new AdminDashboard().start(SceneNavigator.*getStage*()));  
  
 Vbox layout = new Vbox(15, title, nameField, emailField, passwordField, createBtn, backBtn);  
 layout.setPadding(new Insets(30));  
 layout.setAlignment(Pos.*CENTER*);  
 layout.setStyle(“-fx-background-color: #001f3f;”);  
 Scene scene = new Scene(layout, 500, 400);  
 SceneNavigator.*setScene*(scene, “Create Police Account”);  
 }  
  
 private void showAlert(String msg) {  
 Alert alert = new Alert(Alert.AlertType.*INFORMATION*, msg, ButtonType.*OK*);  
 alert.showAndWait();  
 }  
}

**HandleEmergencyScreen.java**

package gui;  
  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.\*;  
import javafx.scene.layout.VBox;  
import javafx.stage.Stage;  
import models.CrimeReport;  
import services.CrimeServices;  
  
import java.util.List;  
  
public class HandleEmergencyScreen {  
  
 private final CrimeServices crimeService = new CrimeServices();  
  
 public HandleEmergencyScreen(controller.AppController controller) {}  
  
 public void start(Stage stage) {  
 Label title = new Label("Declare Emergency Report");  
 title.setStyle("-fx-font-size: 20; -fx-font-weight: bold; -fx-text-fill: white;");  
  
 TextField reportIdField = new TextField();  
 reportIdField.setPromptText("Enter Report ID");  
  
 TextField descField = new TextField();  
 descField.setPromptText("Emergency Description");  
 descField.setDisable(true);  
  
 Button declareBtn = new Button("Declare Emergency");  
 declareBtn.setStyle("-fx-background-color: #dc3545; -fx-text-fill: white;");  
 declareBtn.setDisable(true);  
  
 Button sendTeamBtn = new Button("Send Team Immediately");  
 sendTeamBtn.setStyle("-fx-background-color: #28a745; -fx-text-fill: white;");  
 sendTeamBtn.setDisable(true);  
  
 Button validateBtn = new Button("Validate Report ID");  
 validateBtn.setOnAction(e -> {  
 String reportId = reportIdField.getText().trim();  
 List<CrimeReport> allReports = crimeService.getAllReports();  
 boolean exists = allReports.stream().anyMatch(r -> r.getReportId().equals(reportId));  
  
 if (exists) {  
 descField.setDisable(false);  
 declareBtn.setDisable(false);  
 sendTeamBtn.setDisable(false);  
 showAlert("Report ID found. You may now proceed.");  
 } else {  
 showAlert("Report ID not found.");  
 }  
 });  
  
 declareBtn.setOnAction(e -> {  
 showAlert("Emergency declared for: " + reportIdField.getText());  
 });  
  
 sendTeamBtn.setOnAction(e -> {  
 showAlert("Notification sent to team.\nThey will respond as soon as possible.");  
 });  
  
 Button backBtn = new Button("Back");  
 backBtn.setOnAction(e -> new PoliceDashboard().start(SceneNavigator.*getStage*()));  
  
 VBox layout = new VBox(15, title, reportIdField, validateBtn, descField, declareBtn, sendTeamBtn, backBtn);  
 layout.setAlignment(Pos.*CENTER*);  
 layout.setPadding(new Insets(30));  
 layout.setStyle("-fx-background-color: #001f3f;");  
 Scene scene = new Scene(layout, 500, 500);  
 SceneNavigator.*setScene*(scene, "Declare Emergency");  
 }  
  
 private void showAlert(String msg) {  
 Alert alert = new Alert(Alert.AlertType.*INFORMATION*, msg, ButtonType.*OK*);  
 alert.showAndWait();  
 }  
}

**ManageUsersScreen.java**

package gui;  
  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.\*;  
import javafx.scene.layout.VBox;  
import javafx.stage.Stage;  
import models.User;  
import services.AuthServices;  
import java.util.LinkedHashMap;  
import java.util.List;  
import java.util.Map;  
import java.util.stream.Collectors;  
  
public class ManageUsersScreen {  
  
 private final Map<String, User> userMap = new LinkedHashMap<>();  
 private final Map<String, Boolean> userDeletedMap = new LinkedHashMap<>();  
  
 public void start(Stage stage) {  
 Label title = new Label(“Manage Users”);  
 title.setStyle(“-fx-font-size: 20; -fx-font-weight: bold; -fx-text-fill: white;”);  
  
 TextField searchField = new TextField();  
 searchField.setPromptText(“Search by name or email...”);  
 searchField.setMaxWidth(300);  
  
 ComboBox<String> roleFilter = new ComboBox<>();  
 roleFilter.getItems().addAll(“All”, “Admin”, “Police”, “Citizen”);  
 roleFilter.setValue(“All”);  
 roleFilter.setPrefWidth(300);  
  
 ListView<String> userListView = new ListView<>();  
 loadUsersIntoList(userListView, “”, “All”);  
  
 searchField.textProperty().addListener((obs, oldVal, newVal) -> {  
 loadUsersIntoList(userListView, newVal, roleFilter.getValue());  
 });  
  
 roleFilter.setOnAction(e -> {  
 loadUsersIntoList(userListView, searchField.getText(), roleFilter.getValue());  
 });  
  
 userListView.setOnMouseClicked(event -> {  
 String selectedKey = userListView.getSelectionModel().getSelectedItem();  
 if (selectedKey != null) {  
 if (userDeletedMap.getOrDefault(selectedKey, false)) {  
 showAlert(“This user was already deleted and cannot be managed.”);  
 return;  
 }  
  
 User selectedUser = userMap.get(selectedKey);  
 if (selectedUser != null) {  
 UserDetailPopup.*show*(selectedUser,  
 () -> loadUsersIntoList(userListView, searchField.getText(), roleFilter.getValue()), // onUpdate  
 () -> {  
 userDeletedMap.put(selectedKey, true);  
 showAlert(“User has been deleted.\n(Still visible for audit reference)”);  
 }  
 );  
 }  
 }  
 });  
  
 Button backBtn = new Button(“Back”);  
 backBtn.setStyle(“-fx-background-color: #6c757d; -fx-text-fill: white;”);  
 backBtn.setOnAction(e -> new AdminDashboard().start(stage));  
  
 Vbox layout = new Vbox(15, title, searchField, roleFilter, userListView, backBtn);  
 layout.setPadding(new Insets(30));  
 layout.setAlignment(Pos.*CENTER*);  
 layout.setStyle(“-fx-background-color: #001f3f;”);  
 layout.setPrefSize(600, 600);  
  
 Scene scene = new Scene(layout);  
 stage.setScene(scene);  
 stage.setTitle(“Manage Users”);  
 stage.show();  
 }  
  
 private void loadUsersIntoList(ListView<String> listView, String search, String roleFilter) {  
 listView.getItems().clear();  
  
 Map<String, Boolean> oldDeletedMap = new LinkedHashMap<>(userDeletedMap);  
  
 userMap.clear();  
 userDeletedMap.clear();  
  
 AuthServices auth = new AuthServices();  
 List<User> allUsers = auth.getAllUsers();  
  
 List<User> filtered = allUsers.stream()  
 .filter(user -> {  
 oolean matchesSearch = search == null || search.isEmpty() ||  
 user.getName().toLowerCase().contains(search.toLowerCase()) ||  
 user.getEmail().toLowerCase().contains(search.toLowerCase());  
 oolean matchesRole = roleFilter.equalsIgnoreCase(“All”) ||  
 user.getRole().equalsIgnoreCase(roleFilter);  
 return matchesSearch && matchesRole;  
 })  
 .collect(Collectors.*toList*());  
  
 for (User user : filtered) {  
 String label = user.getRole().toUpperCase() + “: “ + user.getName() + “ (“ + user.getEmail() + “)”;  
 userMap.put(label, user);  
 userDeletedMap.put(label, oldDeletedMap.getOrDefault(label, false));  
 listView.getItems().add(label);  
 }  
 }  
  
 private void showAlert(String msg) {  
 Alert alert = new Alert(Alert.AlertType.*INFORMATION*);  
 alert.setHeaderText(null);  
 alert.setContentText(msg);  
 alert.showAndWait();  
 }  
}

**ReportDetailPopup.java**

package gui;  
  
import javafx.geometry.Insets;  
import javafx.scene.Scene;  
import javafx.scene.control.Label;  
import javafx.scene.layout.VBox;  
import javafx.scene.paint.Color;  
import javafx.scene.text.Font;  
import javafx.scene.text.FontWeight;  
import javafx.stage.Modality;  
import javafx.stage.Stage;  
import models.CrimeReport;  
  
public class ReportDetailPopup {  
  
 public static void show(CrimeReport report) {  
 Stage popup = new Stage();  
 popup.initModality(Modality.*APPLICATION\_MODAL*);  
 popup.setTitle("Report Details");  
  
 Label title = new Label("Title: " + report.getTitle());  
 Label desc = new Label("Description: " + report.getDescription());  
 Label location = new Label("Location: " + report.getLocation());  
 Label status = new Label("Status: " + report.getStatus());  
 Label reporter = new Label("Reporter ID: " + report.getReporterId());  
 Label id = new Label("Report ID: " + report.getReportId());  
  
 for (Label label : new Label[]{title, desc, location, status, reporter, id}) {  
 label.setTextFill(Color.*WHITE*);  
 label.setFont(Font.*font*("Arial", FontWeight.*NORMAL*, 14));  
 label.setWrapText(true);  
 }  
  
 VBox layout = new VBox(10, title, desc, location, status, reporter, id);  
 layout.setPadding(new Insets(20));  
 layout.setStyle("-fx-background-color: #2c2f33;");  
  
 Scene scene = new Scene(layout, 500, 300);  
 popup.setScene(scene);  
 popup.showAndWait();  
 }  
}

**StatusUpdateScreen.java**

package gui;  
  
import controller.AppController;  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.\*;  
import javafx.scene.layout.VBox;  
import javafx.stage.Stage;  
import services.CrimeServices;  
  
public class StatusUpdateScreen {  
  
 private final CrimeServices crimeService = new CrimeServices();  
 private final String caller;  
  
 public StatusUpdateScreen(AppController controller, String caller) {  
 this.caller = caller.toLowerCase();  
 }  
  
 public void start(Stage stage) {  
 Label title = new Label(“Update Report Status”);  
 title.setStyle(“-fx-font-size: 20; -fx-font-weight: bold; -fx-text-fill: white;”);  
  
 TextField reportIdField = new TextField();  
 reportIdField.setPromptText(“Report ID”);  
  
 TextField statusField = new TextField();  
 statusField.setPromptText(“New Status (Verified / Rejected / Case Solved)”);  
  
 Button updateBtn = new Button(“Update”);  
 updateBtn.setStyle(“-fx-background-color: #007bff; -fx-text-fill: white;”);  
 updateBtn.setOnAction(e -> {  
 String id = reportIdField.getText().trim();  
 String status = statusField.getText().trim();  
  
 if (id.isEmpty() || status.isEmpty()) {  
 showAlert(“Please fill in both fields.”);  
 return;  
 }  
  
 oolean updated = crimeService.updateStatus(id, status);  
 showAlert(updated ? “Report status updated!” : “Report not found.”);  
 if (updated) navigateBack();  
 });  
  
 Button backBtn = new Button(“Back”);  
 backBtn.setOnAction(e -> navigateBack());  
  
 Vbox layout = new Vbox(15, title, reportIdField, statusField, updateBtn, backBtn);  
 layout.setPadding(new Insets(30));  
 layout.setAlignment(Pos.*CENTER*);  
 layout.setStyle(“-fx-background-color: #001f3f;”);  
 Scene scene = new Scene(layout, 500, 400);  
 SceneNavigator.*setScene*(scene, “Update Report Status”);  
 }  
  
 private void navigateBack() {  
 if (caller.equals(“police”)) {  
 new PoliceDashboard().start(SceneNavigator.*getStage*());  
 } else {  
 new AdminDashboard().start(SceneNavigator.*getStage*());  
 }  
 }  
  
 private void showAlert(String msg) {  
 Alert alert = new Alert(Alert.AlertType.*INFORMATION*, msg, ButtonType.*OK*);  
 alert.showAndWait();  
 }  
}

**UserDetailPopup.java**

package gui;  
  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.\*;  
import javafx.scene.layout.VBox;  
import javafx.scene.paint.Color;  
import javafx.scene.text.Font;  
import javafx.scene.text.FontWeight;  
import javafx.stage.Modality;  
import javafx.stage.Stage;  
import models.User;  
import services.AuthServices;  
  
public class UserDetailPopup {  
  
 public static void show(User user, Runnable onUpdated, Runnable onDeleted) {  
 Stage popup = new Stage();  
 popup.initModality(Modality.*APPLICATION\_MODAL*);  
 popup.setTitle(“User Details”);  
  
 Label titleLabel = new Label(“User Details”);  
 titleLabel.setFont(Font.*font*(“Arial”, FontWeight.*BOLD*, 20));  
 titleLabel.setTextFill(Color.*WHITE*);  
  
 Label nameLabel = new Label(“Name: “ + user.getName());  
 Label emailLabel = new Label(“Email: “ + user.getEmail());  
 Label roleLabel = new Label(“Role: “ + user.getRole());  
 Label passwordLabel = new Label(“Current Password: “ + user.getPassword());  
  
 for (Label label : new Label[]{nameLabel, emailLabel, roleLabel, passwordLabel}) {  
 label.setTextFill(Color.*WHITE*);  
 label.setFont(Font.*font*(“Arial”, 14));  
 }  
  
 PasswordField newPasswordField = new PasswordField();  
 newPasswordField.setPromptText(“Enter new password”);  
 newPasswordField.setMaxWidth(250);  
  
 Button changePasswordBtn = new Button(“Change Password”);  
 changePasswordBtn.setStyle(“-fx-background-color: #007bff; -fx-text-fill: white;”);  
 changePasswordBtn.setPrefWidth(200);  
  
 Button deleteUserBtn = new Button(“Delete User”);  
 deleteUserBtn.setStyle(“-fx-background-color: #dc3545; -fx-text-fill: white;”);  
 deleteUserBtn.setPrefWidth(200);  
  
 AuthServices auth = new AuthServices();  
  
 changePasswordBtn.setOnAction(e -> {  
 String newPassword = newPasswordField.getText().trim();  
 if (!newPassword.isEmpty()) {  
 user.setPassword(newPassword);  
 *showAlert*(“Password changed for: “ + user.getName() + “ (“ + user.getEmail() + “)”);  
 onUpdated.run();  
 popup.close();  
 } else {  
 *showAlert*(“Password cannot be empty.”);  
 }  
 });  
  
 deleteUserBtn.setOnAction(e -> {  
 if (user.getRole().equalsIgnoreCase(“admin”)) {  
 *showAlert*(“You cannot delete an admin user.”);  
 return;  
 }  
  
 Alert confirm = new Alert(Alert.AlertType.*CONFIRMATION*);  
 confirm.setTitle(“Confirm Deletion”);  
 confirm.setHeaderText(“Are you sure you want to delete this user?”);  
 confirm.setContentText(user.getName() + “ (“ + user.getEmail() + “)”);  
  
 ButtonType yes = new ButtonType(“Yes”, ButtonBar.ButtonData.*OK\_DONE*);  
 ButtonType no = new ButtonType(“No”, ButtonBar.ButtonData.*CANCEL\_CLOSE*);  
 confirm.getButtonTypes().setAll(yes, no);  
  
 confirm.showAndWait().ifPresent(response -> {  
 if (response == yes) {  
 auth.removeUser(user.getEmail());  
 *showAlert*(“User deleted successfully: “ + user.getName() + “ (“ + user.getEmail() + “)”);  
 onDeleted.run();  
 popup.close();  
 }  
 });  
 });  
  
 Vbox layout = new Vbox(12);  
 layout.setPadding(new Insets(20));  
 layout.setAlignment(Pos.*CENTER\_LEFT*);  
 layout.setStyle(“-fx-background-color: #2c2f33;”);  
 layout.getChildren().addAll(  
 titleLabel,  
 nameLabel,  
 emailLabel,  
 roleLabel,  
 passwordLabel,  
 new Label(“New Password:”),  
 newPasswordField,  
 changePasswordBtn,  
 deleteUserBtn  
 );  
  
 Scene scene = new Scene(layout, 450, 420);  
 popup.setScene(scene);  
 popup.showAndWait();  
 }  
  
 private static void showAlert(String msg) {  
 Alert alert = new Alert(Alert.AlertType.*INFORMATION*);  
 alert.setHeaderText(null);  
 alert.setContentText(msg);  
 alert.showAndWait();  
 }  
}

**ViewAllReportsScreen.java**

package gui;  
  
import controller.AppController;  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.\*;  
import javafx.scene.layout.VBox;  
import javafx.stage.Stage;  
import models.CrimeReport;  
import services.CrimeServices;  
  
import java.util.List;  
  
public class ViewAllReportsScreen {  
  
 private final AppController controller;  
 private final String caller; // "admin" or "police"  
 private final List<CrimeReport> reports = new CrimeServices().getAllReports();  
  
 public ViewAllReportsScreen(AppController controller, String caller) {  
 this.controller = controller;  
 this.caller = caller.toLowerCase();  
 }  
  
 public void start(Stage stage) {  
 ListView<String> reportList = new ListView<>();  
  
 for (CrimeReport r : reports) {  
 String summary = "• " + r.getTitle() +  
 " | ID: " + r.getReportId() +  
 " | Status: " + r.getStatus();  
 reportList.getItems().add(summary);  
 }  
  
 reportList.setOnMouseClicked(e -> {  
 int i = reportList.getSelectionModel().getSelectedIndex();  
 if (i >= 0) {  
 ReportDetailPopup.*show*(reports.get(i));  
 }  
 });  
  
 Button backBtn = new Button("Back");  
 backBtn.setOnAction(e -> {  
 if (caller.equals("police")) {  
 new PoliceDashboard().start(SceneNavigator.*getStage*());  
 } else {  
 new AdminDashboard().start(SceneNavigator.*getStage*());  
 }  
 });  
  
 VBox layout = new VBox(15, reportList, backBtn);  
 layout.setPadding(new Insets(30));  
 layout.setAlignment(Pos.*CENTER*);  
 layout.setStyle("-fx-background-color: #001f3f;");  
 Scene scene = new Scene(layout, 600, 450);  
 SceneNavigator.*setScene*(scene, "All Reports");  
 }  
}

**SceneNavigator.java**

package gui;  
  
import javafx.scene.Scene;  
import javafx.stage.Stage;  
  
public class SceneNavigator {  
 private static Stage *primaryStage*;  
  
 public static void setStage(Stage stage) {  
 *primaryStage* = stage;  
 }  
  
 public static void setScene(Scene scene, String title) {  
 *primaryStage*.setTitle(title);  
 *primaryStage*.setScene(scene);  
 *primaryStage*.show();  
 }  
  
 public static Stage getStage() {  
 return *primaryStage*;  
 }  
}

**MainApp.java**

package gui;  
  
import javafx.application.Application;  
import javafx.stage.Stage;  
import gui.RoleSelectionScreen;  
  
public class MainApp extends Application {  
  
 @Override  
 public void start(Stage primaryStage) {  
 SceneNavigator.*setStage*(primaryStage);  
 new RoleSelectionScreen().start(primaryStage);  
 }  
  
 public static void main(String[] args) {  
 *launch*(args);  
 }  
}