

|  |  |
| --- | --- |
| Names: | Registration numbers: |
| Aminah Tariq  Adina Zafar  Ayesha Arshad | Sp24-bse-011-B  Sp24-bse-006-B  Sp24-bse-019-B |
| Assignment number: | 3 |
| Course instructor: | Prof Shahid Bhatti |
| Due Date: | 12-12-2024 |

**SOURCE CODE:**

Adina Zafar: SP24-BSE-006

Aminah Tariq: SP24-BSE-011

Ayesha Arshad:SP24-BSE-019

**User.java**

package com.example.bloodbankjava;  
  
public class User {  
 private int userId;  
 private String name;  
 private String role;  
 private String email;  
 private String password;  
  
 public User(String name, String email) {  
 this.name = name;  
 this.email = email;  
 }  
  
 public User(int userId, String name, String role, String email, String password) {  
 this.userId = userId;  
 this.name = name;  
 this.role = role;  
 this.email = email;  
 this.password = password;  
 }  
   
 public int getUserId() { return userId; }  
 public void setUserId(int userId) { this.userId = userId; }  
 public String getName() { return this.name; }  
 public void setName(String name) { this.name = name; }  
 public String getRole() { return role; }  
 public void setRole(String role) { this.role = role; }  
 public String getEmail() { return email; }  
 public void setEmail(String email) { this.email = email; }  
 public String getPassword() { return password; }  
 public void setPassword(String password) { this.password = password; }  
  
 public String toString() {  
 return name + " (" + email + ")";  
 }  
}

**UserService.java**

package com.example.bloodbankjava;  
  
import javafx.application.Application;  
import javafx.stage.Stage;  
import javafx.scene.control.\*;  
import java.util.List;  
import java.util.ArrayList;  
import java.util.List;  
  
public class Userservice {  
 private static List<User> *users* = new ArrayList<>();  
  
 // Fetch all users  
 public static List<User> getAllUsers() {  
 // Example: Replace with your actual implementation to fetch users  
 List<User> users = new ArrayList<>();  
 users.add(new User("Admin", "admin@example.com")); // Sample user  
 return users;  
 }  
  
  
 // Add a new user  
 public static void addUser(User user) {  
 // Example: Replace with your actual implementation  
 System.*out*.println("User added: " + user.getName() + ", " + user.getEmail());  
 // Add to storage/database  
 }  
  
 // Update an existing user  
 public static void updateUser(User user) {  
 // Example: Replace with your actual implementation  
 for (int i = 0; i < *users*.size(); i++) {  
 if (*users*.get(i).getEmail().equals(user.getEmail())) {  
 *users*.set(i, user);  
 break;  
 }  
 }  
 }  
  
 // System.out.println("User updated: " + user.getName() + ", " + user.getEmail());  
 // Update user in storage/database  
  
 // Remove a user  
 public static void removeUser(User user) {  
 // Check if the user exists in the list  
 if (*users*.removeIf(existingUser -> existingUser.getEmail().equals(user.getEmail()))) {  
 System.*out*.println("User removed: " + user.getName() + ", " + user.getEmail());  
 } else {  
 System.*out*.println("User not found: " + user.getName() + ", " + user.getEmail());  
 }  
 }  
  
 public User login(String email, String password) {  
 return null; }  
}

**Request.java**

package com.example.bloodbankjava;  
  
import java.time.LocalDate;  
  
public class Request {  
 private int requestId;  
 private String hospitalName;  
 private String bloodType;  
 private int quantity;  
 private String urgency;  
 private String status;  
 private LocalDate requestDate;  
  
 public Request(int requestId, String hospitalName, String bloodType, int quantity, String urgency, String status, LocalDate requestDate) {  
 this.requestId = requestId;  
 this.hospitalName = hospitalName;  
 this.bloodType = bloodType;  
 this.quantity = quantity;  
 this.urgency = urgency;  
 this.status = status;  
 this.requestDate = requestDate;  
 }  
 public int getRequestId() { return requestId; }  
 public void setRequestId(int requestId) { this.requestId = requestId; }  
 public String getHospitalName() { return hospitalName; }  
 public void setHospitalName(String hospitalName) { this.hospitalName = hospitalName; }  
 public String getBloodType() { return bloodType; }  
 public void setBloodType(String bloodType) { this.bloodType = bloodType; }  
 public int getQuantity() { return quantity; }  
 public void setQuantity(int quantity) { this.quantity = quantity; }  
 public String getUrgency() { return urgency; }  
 public void setUrgency(String urgency) { this.urgency = urgency; }  
 public String getStatus() { return status; }  
 public void setStatus(String status) { this.status = status; }  
 public LocalDate getRequestDate() { return requestDate; }  
 public void setRequestDate(LocalDate requestDate) { this.requestDate = requestDate; }  
}

**RequestService.java**

package com.example.bloodbankjava;  
import java.io.IOException;  
import java.util.ArrayList;  
import java.util.List;  
  
import java.io.IOException;  
import java.util.ArrayList;  
import java.util.List;  
public class RequestService {  
   
 private static final String *FILE\_PATH* ="C:\\Users\\DELL\\Desktop\\BloodBank-java\\src\\main\\java\\com\\example\\bloodbankjava\\Request.txt";  
 public List<BloodRequest> getAllRequests() throws IOException {  
 List<BloodRequest> requests = new ArrayList<>();  
 List<String> lines = FileReaderWriter.*readFile*(*FILE\_PATH*);  
 for (String line : lines) {  
 String[] parts = line.split(",");  
 requests.add(new BloodRequest(  
 Integer.*parseInt*(parts[0]),  
 Integer.*parseInt*(parts[1]),  
 parts[2],  
 Integer.*parseInt*(parts[3]),  
 parts[4],  
 parts[5],  
 java.time.LocalDate.*parse*(parts[6])  
 ));  
 }  
 return requests;  
 }  
  
 public String getAllRequestsFormatted() throws IOException {  
 StringBuilder report = new StringBuilder();  
 for (BloodRequest request : getAllRequests()) {  
 report.append("Request ID: ").append(request.getRequestId())  
 .append(", Blood Type: ").append(request.getBloodType())  
 .append(", Quantity: ").append(request.getQuantity())  
 .append(", Priority: ").append(request.getPriority())  
 .append(", Status: ").append(request.getStatus())  
 .append("\n");  
 }  
 return report.toString();  
 }  
  
 public int getNextRequestId() throws IOException {  
 return getAllRequests().stream()  
 .mapToInt(BloodRequest::getRequestId)  
 .max()  
 .orElse(0) + 1;  
 }  
  
 public void addRequest(BloodRequest request) throws IOException {  
 List<String> lines = FileReaderWriter.*readFile*(*FILE\_PATH*);  
 String newRequest = request.getRequestId() + "," +  
 request.getHospitalId() + "," +  
 request.getBloodType() + "," +  
 request.getQuantity() + "," +  
 request.getPriority() + "," +  
 request.getStatus() + "," +  
 request.getRequestDate();  
 lines.add(newRequest);  
 FileReaderWriter.*writeFile*(*FILE\_PATH*, lines);  
 }  
  
 public String getRequestStatusForHospital(int hospitalId) throws IOException {  
 StringBuilder statusReport = new StringBuilder();  
 for (BloodRequest request : getAllRequests()) {  
 if (request.getHospitalId() == hospitalId) {  
 statusReport.append("Request ID: ").append(request.getRequestId())  
 .append(", Blood Type: ").append(request.getBloodType())  
 .append(", Quantity: ").append(request.getQuantity())  
 .append(", Status: ").append(request.getStatus())  
 .append(", Priority: ").append(request.getPriority())  
 .append("\n");  
 }  
 }  
 return statusReport.length() > 0 ? statusReport.toString() : "No requests found for this hospital.";  
 }  
}

**FileReaderWriter.java**

package com.example.bloodbankjava;  
import java.io.\*;  
import java.util.ArrayList;  
import java.util.List;  
public class FileReaderWriter {  
  
 public static List<String> readFile(String fileName) {  
 List<String> lines = new ArrayList<>();  
 try (BufferedReader reader = new BufferedReader(new FileReader(fileName))) {  
 String line;  
 while ((line = reader.readLine()) != null) {  
 lines.add(line);  
 }  
 } catch (IOException e) {  
 System.*out*.println("Error reading file: " + fileName);  
 }  
 return lines;  
 }  
  
 public static void writeFile(String fileName, List<String> data) {  
 try (BufferedWriter writer = new BufferedWriter(new FileWriter(fileName, false))) {  
 for (String line : data) {  
 writer.write(line);  
 writer.newLine();  
 }  
 } catch (IOException e) {  
 System.*out*.println("Error writing file: " + fileName);  
 }  
 }  
  
 public static void appendToFile(String fileName, String data) {  
 try (BufferedWriter writer = new BufferedWriter(new FileWriter(fileName, true))) {  
 writer.write(data);  
 writer.newLine();  
 } catch (IOException e) {  
 System.*out*.println("Error appending to file: " + fileName);  
 }  
 }  
}

**SignUp Screen.java**

package com.example.bloodbankjava;  
import javafx.application.Application;  
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 java.io.BufferedWriter;  
import java.io.FileWriter;  
import java.io.IOException;  
  
public class SignUpScreen extends Application {  
 private static final String *FILE\_PATH* = "C:\\Users\\DELL\\Desktop\\BloodBank-java\\src\\main\\java\\com\\example\\bloodbankjava\\user.txt";  
  
 @Override  
 public void start(Stage primaryStage) {  
 VBox layout = new VBox();  
 layout.setAlignment(Pos.*CENTER*);  
 layout.setSpacing(20);  
 layout.setPadding(new Insets(50));  
 layout.setStyle("-fx-background-color: #000000;");  
  
 Label titleLabel = new Label("Sign-Up");  
 titleLabel.setStyle("-fx-font-size: 24px; -fx-font-weight: bold; -fx-text-fill: #ffffff");  
  
 TextField nameField = new TextField();  
 nameField.setPromptText("Enter Name");  
 nameField.setPrefWidth(300);  
  
 TextField emailField = new TextField();  
 emailField.setPromptText("Enter Email");  
 emailField.setPrefWidth(300);  
  
 TextField roleField = new TextField();  
 roleField.setPromptText("Enter Role (Admin/Donor/Hospital)");  
 roleField.setPrefWidth(300);  
  
 PasswordField passwordField = new PasswordField();  
 passwordField.setPromptText("Enter Password");  
 passwordField.setPrefWidth(300);  
  
 Button signUpButton = new Button("Sign Up");  
 signUpButton.setStyle("-fx-background-color: #0d47a1; -fx-text-fill: white; -fx-font-size: 18;");  
 signUpButton.setPrefWidth(300);  
  
 signUpButton.setOnAction(e -> {  
 String name = nameField.getText();  
 String email = emailField.getText();  
 String role = roleField.getText();  
 String password = passwordField.getText();  
  
 if (name.isEmpty() || email.isEmpty() || role.isEmpty() || password.isEmpty()) {  
 showAlert("Error", "All fields are required!", Alert.AlertType.*ERROR*);  
 return;  
 }  
  
 int newUserId = generateUserId();  
  
 User newUser = new User(newUserId, name, role, email, password);  
  
 try {  
 writeUserToFile(newUser);  
 showAlert("Success", "Sign-Up Successful! You can now log in.", Alert.AlertType.*INFORMATION*);  
 primaryStage.close(); // Close the Sign-Up window  
 } catch (IOException ex) {  
 showAlert("Error", "Could not save user data. Please try again.", Alert.AlertType.*ERROR*);  
 ex.printStackTrace();  
 }  
 });  
  
 layout.getChildren().addAll(titleLabel, nameField, emailField, roleField, passwordField, signUpButton);  
  
 Scene scene = new Scene(layout, 800, 500);  
 primaryStage.setTitle("Sign-Up");  
 primaryStage.setScene(scene);  
 primaryStage.show();  
 }  
  
 private int generateUserId() {  
  
 return (int) (Math.*random*() \* 10000);  
 }  
  
 private void writeUserToFile(User user) throws IOException {  
 try (BufferedWriter writer = new BufferedWriter(new FileWriter(*FILE\_PATH*, true))) {  
 String userData = user.getUserId() + "," + user.getName() + "," + user.getRole() + "," + user.getEmail() + "," + user.getPassword();  
 writer.write(userData);  
 writer.newLine();  
 }  
 }  
  
 private void showAlert(String title, String message, Alert.AlertType type) {  
 Alert alert = new Alert(type);  
 alert.setTitle(title);  
 alert.setContentText(message);  
 alert.showAndWait();  
 }  
}

**Login Controller.java**

package com.example.bloodbankjava;  
import javafx.fxml.FXML;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Scene;  
import javafx.scene.control.Alert;  
import javafx.scene.control.Button;  
import javafx.scene.control.PasswordField;  
import javafx.scene.control.TextField;  
import javafx.stage.Stage;  
  
import java.io.IOException;  
  
public class LoginController {  
 @FXML  
 private TextField emailField;  
 @FXML  
 private PasswordField passwordField;  
 @FXML  
 private Button loginButton;  
 private Userservice userService = new Userservice();  
  
 @FXML  
 private void handleLogin() {  
 String email = emailField.getText();  
 String password = passwordField.getText();  
 try {  
 User user = userService.login(email, password);  
 if (user == null) {  
 showAlert("Login Failed", "Invalid email or password.", Alert.AlertType.*ERROR*);  
 } else {  
 showAlert("Login Successful", "Welcome, " + user.getName(), Alert.AlertType.*INFORMATION*);  
   
 Stage stage = (Stage) loginButton.getScene().getWindow();  
 stage.close();  
 if (user.getRole().equals("Admin")) {  
 openDashboard("admin\_dashboard.fxml");  
 } else if (user.getRole().equals("Donor")) {  
 openDashboard("donor\_dashboard.fxml");  
 } else if (user.getRole().equals("Hospital")) {  
 openDashboard("hospital\_dashboard.fxml");  
 }  
 }  
 } catch (Exception e) {  
 e.printStackTrace();  
 showAlert("Error", "An error occurred while logging in.", Alert.AlertType.*ERROR*);  
 }  
 }  
  
 private void showAlert(String title, String message, Alert.AlertType type) {  
 Alert alert = new Alert(type);  
 alert.setTitle(title);  
 alert.setContentText(message);  
 alert.showAndWait();  
 }  
  
 private void openDashboard(String fxmlFile) {  
 try {  
 FXMLLoader loader = new FXMLLoader(getClass().getResource(fxmlFile));  
 Scene scene = new Scene(loader.load());  
 Stage stage = new Stage();  
 stage.setScene(scene);  
 stage.show();  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
}

**Blood Request.java**

package com.example.bloodbankjava;  
import java.time.LocalDate;  
public class BloodRequest {  
 private int requestId;  
 private int hospitalId;  
 private String bloodType;  
 private int quantity;  
 private String priority;  
 private String status;  
 private LocalDate requestDate;  
  
 public BloodRequest(int requestId, int hospitalId, String bloodType, int quantity, String priority, String status, LocalDate requestDate) {  
 this.requestId = requestId;  
 this.hospitalId = hospitalId;  
 this.bloodType = bloodType;  
 this.quantity = quantity;  
 this.priority = priority;  
 this.status = status;  
 this.requestDate = requestDate;  
 }  
 public int getRequestId() {  
 return requestId;  
 }  
 public void setRequestId(int requestId) {  
 this.requestId = requestId;  
 }  
 public int getHospitalId() {  
 return hospitalId;  
 }  
 public void setHospitalId(int hospitalId) {  
 this.hospitalId = hospitalId;  
 }  
 public String getBloodType() {  
 return bloodType;  
 }  
 public void setBloodType(String bloodType) {  
 this.bloodType = bloodType;  
 }  
 public int getQuantity() {  
 return quantity;  
 }  
 public void setQuantity(int quantity) {  
 this.quantity = quantity;  
 }  
 public String getPriority() {  
 return priority;  
 }  
 public void setPriority(String priority) {  
 this.priority = priority;  
 }  
 public String getStatus() {  
 return status;  
 }  
 public void setStatus(String status) {  
 this.status = status;  
 }  
 public LocalDate getRequestDate() {  
 return requestDate;  
 }  
 public void setRequestDate(LocalDate requestDate) {  
 this.requestDate = requestDate;  
 }  
 @Override  
 public String toString() {  
 return "BloodRequest{" +  
 "Request ID=" + requestId +  
 ", Hospital ID=" + hospitalId +  
 ", Blood Type='" + bloodType + '\'' +  
 ", Quantity=" + quantity +  
 ", Priority='" + priority + '\'' +  
 ", Status='" + status + '\'' +  
 ", Request Date=" + requestDate +  
 '}';  
 }  
}

**Blood Inventory.java**

package com.example.bloodbankjava;  
import java.io.\*;  
import java.nio.file.\*;  
import java.util.\*;  
  
public class BloodInventory {  
 private static final String *INVENTORY\_FILE* ="C:\\Users\\DELL\\Desktop\\BloodBank-java\\src\\main\\java\\com\\example\\bloodbankjava\\inventory.txt";  
  
 public Map<String, Integer> getBloodStock() throws IOException {  
 Map<String, Integer> inventory = new HashMap<>();  
 List<String> lines = Files.*readAllLines*(Paths.*get*(*INVENTORY\_FILE*));  
 for (String line : lines) {  
 String[] parts = line.split(",");  
 inventory.put(parts[0], Integer.*parseInt*(parts[1]));  
 }  
 return inventory;  
 }  
 public void updateStock(String bloodType, int quantity) throws IOException {  
 Map<String, Integer> inventory = getBloodStock();  
 inventory.put(bloodType, inventory.getOrDefault(bloodType, 0) + quantity);  
  
 try (BufferedWriter writer = new BufferedWriter(new FileWriter(*INVENTORY\_FILE*))) {  
 for (Map.Entry<String, Integer> entry : inventory.entrySet()) {  
 writer.write(entry.getKey() + "," + entry.getValue());  
 writer.newLine();  
 }  
 }  
 }  
 public void reduceStock(String bloodType, int quantity) throws IOException {  
 Map<String, Integer> inventory = getBloodStock();  
 if (inventory.containsKey(bloodType)) {  
 int newQuantity = inventory.get(bloodType) - quantity;  
 if (newQuantity < 0) throw new IllegalArgumentException("Insufficient stock.");  
 inventory.put(bloodType, newQuantity);  
  
 try (BufferedWriter writer = new BufferedWriter(new FileWriter(*INVENTORY\_FILE*))) {  
 for (Map.Entry<String, Integer> entry : inventory.entrySet()) {  
 writer.write(entry.getKey() + "," + entry.getValue());  
 writer.newLine();  
 }  
 }  
 } else {  
 throw new IllegalArgumentException("Blood type not found.");  
 }  
 }  
}

**Blood Inventory Service.java**

package com.example.bloodbankjava;  
import java.io.IOException;  
import java.util.ArrayList;  
import java.util.List;  
import java.io.IOException;  
import java.util.Map;  
  
import java.io.IOException;  
import java.util.Map;  
  
public class BloodInventoryService {  
  
 private BloodInventory bloodInventory = new BloodInventory();  
  
 public Map<String, Integer> getBloodStock() throws IOException {  
 return bloodInventory.getBloodStock();  
 }  
  
 public void addStock(String bloodType, int quantity) throws IOException {  
 if (quantity <= 0) {  
 throw new IllegalArgumentException("Quantity must be greater than 0.");  
 }  
 bloodInventory.updateStock(bloodType, quantity);  
 }  
  
 public void reduceStock(String bloodType, int quantity) throws IOException {  
 if (quantity <= 0) {  
 throw new IllegalArgumentException("Quantity must be greater than 0.");  
 }  
 bloodInventory.reduceStock(bloodType, quantity);  
 }  
  
 public String generateBloodInventoryReport() throws IOException {  
 Map<String, Integer> bloodStock = getBloodStock();  
 StringBuilder report = new StringBuilder();  
 report.append("Blood Inventory Report\n");  
 report.append("=====================\n");  
  
 if (bloodStock.isEmpty()) {  
 report.append("No blood available in the inventory.");  
 } else {  
 for (Map.Entry<String, Integer> entry : bloodStock.entrySet()) {  
 report.append("Blood Type: ")  
 .append(entry.getKey())  
 .append(", Quantity: ")  
 .append(entry.getValue())  
 .append("\n");  
 }  
 }  
 return report.toString();  
 }  
}

**Admin Dashboard.java**

package com.example.bloodbankjava;  
import javafx.application.Application;  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.\*;  
import javafx.scene.control.cell.PropertyValueFactory;  
import javafx.scene.image.Image;  
import javafx.scene.image.ImageView;  
import javafx.scene.layout.\*;  
import javafx.scene.text.Font;  
import javafx.scene.text.FontWeight;  
import javafx.stage.Stage;  
  
public class AdminDashboard extends Application {  
  
 private User admin;  
 private RequestService requestService = new RequestService();  
  
 public AdminDashboard(User admin) {  
 this.admin = admin;  
 }  
  
 @Override  
 public void start(Stage primaryStage) {  
 VBox layout = new VBox(20);  
 layout.setPadding(new Insets(20));  
 layout.setAlignment(Pos.*CENTER*);  
  
 Label titleLabel = new Label("Admin Dashboard - Welcome, " + admin.getName());  
 titleLabel.setFont(Font.*font*("Arial", FontWeight.*BOLD*, 24));  
  
 ImageView imageView = new ImageView();  
 try {  
 Image image = new Image("file:C:\\Users\\DELL\\Desktop\\BloodBank-java\\src\\main\\resources\\com\\example\\bloodbankjava\\admin.png");  
 imageView.setImage(image);  
 imageView.setFitWidth(300);  
 imageView.setFitHeight(300);  
 imageView.setPreserveRatio(true);  
 } catch (Exception e) {  
 System.*out*.println("Error loading image: " + e.getMessage());  
 }  
  
 HBox buttonLayout = new HBox(15);  
 buttonLayout.setAlignment(Pos.*CENTER*);  
  
 Button viewReportsButton = new Button("View Donation Reports");  
 Button manageUsersButton = new Button("Manage Users");  
 Button logoutButton = new Button("Logout");  
  
 String buttonStyle = "-fx-background-color: #390a3c; -fx-text-fill: white; -fx-font-size: 14px; -fx-font-weight: bold; -fx-padding: 10px 20px; -fx-border-radius: 5px; -fx-background-radius: 5px;";  
 viewReportsButton.setStyle(buttonStyle);  
 manageUsersButton.setStyle(buttonStyle);  
 logoutButton.setStyle(buttonStyle);  
  
 viewReportsButton.setPrefWidth(250);  
 manageUsersButton.setPrefWidth(200);  
 logoutButton.setPrefWidth(200);  
  
 buttonLayout.getChildren().addAll(viewReportsButton, manageUsersButton, logoutButton);  
 VBox.*setVgrow*(buttonLayout, Priority.*ALWAYS*);  
  
 layout.getChildren().addAll(titleLabel, imageView, buttonLayout);  
  
 viewReportsButton.setOnAction(e -> {  
 try {  
 String reports = requestService.getAllRequestsFormatted();  
 showAlert("Donation Reports", reports, Alert.AlertType.*INFORMATION*);  
 } catch (Exception ex) {  
 ex.printStackTrace();  
 showAlert("Error", "Could not load donation reports.", Alert.AlertType.*ERROR*);  
 }  
 });  
  
 manageUsersButton.setOnAction(e -> {  
 openManageUsersWindow();  
 });  
  
 logoutButton.setOnAction(e -> {  
 primaryStage.close();  
 new Main().start(new Stage());  
 });  
  
 Scene scene = new Scene(layout, 600, 500);  
 primaryStage.setTitle("Admin Dashboard");  
 primaryStage.setScene(scene);  
 primaryStage.show();  
 }  
 private void openManageUsersWindow() {  
 Stage manageUsersStage = new Stage();  
 manageUsersStage.setTitle("Manage Users");  
  
 VBox manageUsersLayout = new VBox(20);  
 manageUsersLayout.setPadding(new Insets(20));  
 manageUsersLayout.setAlignment(Pos.*CENTER*);  
 Label manageUsersTitle = new Label("Manage Users");  
 manageUsersTitle.setFont(Font.*font*("Arial", FontWeight.*BOLD*, 20));  
  
 TableView<User> userTable = new TableView<>();  
 TableColumn<User, String> nameColumn = new TableColumn<>("Name");  
 nameColumn.setCellValueFactory(new PropertyValueFactory<>("name"));  
 TableColumn<User, String> emailColumn = new TableColumn<>("Email");  
 emailColumn.setCellValueFactory(new PropertyValueFactory<>("email"));  
  
 userTable.getColumns().addAll(nameColumn, emailColumn);  
 userTable.getItems().addAll(Userservice.*getAllUsers*());  
  
 Button addUserButton = new Button("Add User");  
 Button editUserButton = new Button("Edit User");  
 Button removeUserButton = new Button("Remove User");  
  
 String buttonStyle = "-fx-background-color: #007bff; -fx-text-fill: white; -fx-font-size: 14px; -fx-font-weight: bold; -fx-padding: 10px 20px; -fx-border-radius: 5px; -fx-background-radius: 5px;";  
 addUserButton.setStyle(buttonStyle);  
 editUserButton.setStyle(buttonStyle);  
 removeUserButton.setStyle(buttonStyle);  
 HBox manageButtonLayout = new HBox(15);  
 manageButtonLayout.setAlignment(Pos.*CENTER*);  
 manageButtonLayout.getChildren().addAll(addUserButton, editUserButton, removeUserButton);  
 manageUsersLayout.getChildren().addAll(manageUsersTitle, userTable, manageButtonLayout);  
 Scene manageUsersScene = new Scene(manageUsersLayout, 600, 400);  
 manageUsersStage.setScene(manageUsersScene);  
 manageUsersStage.show();  
 }  
 private void showAlert(String title, String message, Alert.AlertType type) {  
 Alert alert = new Alert(type);  
 alert.setTitle(title);  
 alert.setContentText(message);  
 alert.showAndWait();  
 }  
}

**Admin Dashboard Controller.java**

package com.example.bloodbankjava;  
import javafx.scene.control.Alert;  
import java.io.IOException;  
import java.util.Map;  
  
public class AdminDashboardController {  
  
 private BloodInventoryService bloodInventoryService = new BloodInventoryService();  
  
 public String viewBloodInventory() throws IOException {  
 Map<String, Integer> stock = bloodInventoryService.getBloodStock();  
 StringBuilder stockDetails = new StringBuilder();  
 stock.forEach((bloodType, quantity) -> stockDetails.append("Blood Type: ")  
 .append(bloodType)  
 .append(", Quantity: ")  
 .append(quantity)  
 .append("\n"));  
 return stockDetails.toString();  
 }  
 public void addStock(String bloodType, int quantity) throws IOException {  
 try {  
 bloodInventoryService.addStock(bloodType, quantity);  
 showAlert("Success", "Stock updated successfully.", Alert.AlertType.*INFORMATION*);  
 } catch (IllegalArgumentException ex) {  
 showAlert("Error", ex.getMessage(), Alert.AlertType.*ERROR*);  
 }  
 }  
 private void showAlert(String title, String message, Alert.AlertType type) {  
 Alert alert = new Alert(type);  
 alert.setTitle(title);  
 alert.setContentText(message);  
 alert.showAndWait();  
 }  
}

**Donor.java**

package com.example.bloodbankjava;  
import java.time.LocalDate;  
public class Donor extends User {  
 private int age;  
 private String gender;  
 private String bloodType;  
 private String contact;  
 private String lastDonationDate;  
  
 public Donor(int id, String name, String role, String email, String password, int age, String gender, String bloodType, String contact, String lastDonationDate) {  
 super(id, name, role, email, password);  
 this.age = age;  
 this.gender = gender;  
 this.bloodType = bloodType;  
 this.contact = contact;  
 this.lastDonationDate = lastDonationDate;  
 }  
  
 public Donor(String part, int i, String part1, String part2, String part3, String part4) {  
 super(i, part, part1, part2, part3);  
 this.lastDonationDate= part4;  
 }  
  
 public int getAge() {  
 return age;  
 }  
  
 public String getGender() {  
 return gender;  
 }  
  
 public String getBloodType() {  
 return bloodType;  
 }  
  
 public String getContact() {  
 return contact;  
 }  
  
 public String getLastDonationDate() {  
 return lastDonationDate;  
 }  
  
 public void setLastDonationDate(String lastDonationDate) {  
 this.lastDonationDate = lastDonationDate;  
 }  
  
 public void donateBlood(String donationDate) {  
 setLastDonationDate(donationDate);  
 System.*out*.println("Blood donation successful! Last donation date updated.");  
 }  
 @Override  
 public String toString() {  
 return super.toString() + "," + age + "," + gender + "," + bloodType + "," + contact + "," + lastDonationDate;  
 }  
}

**Donor Service.java**

package com.example.bloodbankjava;  
import java.io.IOException;  
import java.nio.file.Files;  
import java.nio.file.Paths;  
import java.nio.file.StandardOpenOption;  
import java.time.LocalDate;  
import java.util.ArrayList;  
import java.util.List;  
  
public class DonorService {  
 private static final String *DONOR\_FILE* = "C:\\Users\\DELL\\Desktop\\BloodBank-java\\src\\main\\java\\com\\example\\bloodbankjava\\donor.txt";  
 private static final String *DONATION\_FILE* = "C:\\Users\\DELL\\Desktop\\BloodBank-java\\src\\main\\java\\com\\example\\bloodbankjava\\donations.txt";  
  
 public static List<Donor> getAllDonors() {  
 List<String> lines = FileReaderWriter.*readFile*(*DONOR\_FILE*);  
 List<Donor> donors = new ArrayList<>();  
 for (String line : lines) {  
 String[] parts = line.split(",");  
 donors.add(new Donor(parts[0], Integer.*parseInt*(parts[1]), parts[2], parts[3], parts[4], parts[5]));  
 }  
 return donors;  
 }  
  
 public static void addDonor(Donor donor) {  
 FileReaderWriter.*appendToFile*(*DONOR\_FILE*, donor.toString());  
 }  
 public static void addDonation(int donorId, String bloodType) throws IOException {  
 String donationRecord = donorId + "," + bloodType + "," + LocalDate.*now*() + "\n";  
 Files.*write*(Paths.*get*(*DONATION\_FILE*), donationRecord.getBytes(), StandardOpenOption.*CREATE*, StandardOpenOption.*APPEND*);  
 }  
  
 public static String getDonationHistory(int donorId) throws IOException {  
 List<String> lines = Files.*readAllLines*(Paths.*get*(*DONATION\_FILE*));  
 StringBuilder history = new StringBuilder();  
 for (String line : lines) {  
 String[] parts = line.split(",");  
 if (Integer.*parseInt*(parts[0]) == donorId) {  
 history.append("Blood Type: ").append(parts[1])  
 .append(", Date: ").append(parts[2]).append("\n");  
 }  
 }  
 return history.length() > 0 ? history.toString() : "No donation history found.";  
 }  
 public static boolean canDonate(String lastDonationDate) {  
 return true;  
 }  
}

**Donor Dashboard .java**

package com.example.bloodbankjava;  
  
import javafx.application.Application;  
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.HBox;  
import javafx.scene.layout.Priority;  
import javafx.scene.layout.VBox;  
import javafx.scene.text.Font;  
import javafx.scene.text.FontWeight;  
import javafx.stage.Stage;  
  
import java.io.IOException;  
  
public class DonorDashboard extends Application {  
  
 private DonorService donorService = new DonorService();  
 private User donor;  
  
 public DonorDashboard(User donor) {  
 this.donor = donor;  
 }  
  
 @Override  
 public void start(Stage primaryStage) {  
 VBox layout = new VBox(20);  
 layout.setPadding(new Insets(20));  
 layout.setAlignment(Pos.*CENTER*);  
  
 Label titleLabel = new Label("Donor Dashboard - Welcome, " + donor.getName());  
 titleLabel.setFont(Font.*font*("Arial", FontWeight.*BOLD*, 24));  
  
 ImageView imageView = new ImageView();  
 try {  
 Image image = new Image("file:C:\\Users\\DELL\\Desktop\\BloodBank-java\\src\\main\\resources\\com\\example\\bloodbankjava\\donor.jpg");  
 imageView.setImage(image);  
 imageView.setFitWidth(300);  
 imageView.setPreserveRatio(true);  
 } catch (Exception e) {  
 System.*out*.println("Error loading image: " + e.getMessage());  
 }  
  
 HBox buttonLayout = new HBox(15);  
 buttonLayout.setAlignment(Pos.*CENTER*);  
  
 Button viewHistoryButton = new Button("View Donation History");  
 Button addDonationButton = new Button("Add Donation");  
 Button logoutButton = new Button("Logout");  
  
 String buttonStyle = "-fx-background-color: #4f1111; -fx-text-fill: white; -fx-font-size: 14px; -fx-font-weight: bold; -fx-padding: 10px 20px; -fx-border-radius: 5px; -fx-background-radius: 5px;";  
 viewHistoryButton.setStyle(buttonStyle);  
 addDonationButton.setStyle(buttonStyle);  
 logoutButton.setStyle(buttonStyle);  
  
 viewHistoryButton.setPrefWidth(250);  
 addDonationButton.setPrefWidth(200);  
 logoutButton.setPrefWidth(150);  
  
 buttonLayout.getChildren().addAll(viewHistoryButton, addDonationButton, logoutButton);  
 VBox.*setVgrow*(buttonLayout, Priority.*ALWAYS*);  
  
 layout.getChildren().addAll(titleLabel, imageView, buttonLayout);  
  
 viewHistoryButton.setOnAction(e -> {  
 try {  
 String history = donorService.*getDonationHistory*(donor.getUserId());  
 showAlert("Donation History", history, Alert.AlertType.*INFORMATION*);  
 } catch (IOException ex) {  
 ex.printStackTrace();  
 showAlert("Error", "Could not load donation history.", Alert.AlertType.*ERROR*);  
 }  
 });  
  
 addDonationButton.setOnAction(e -> {  
 TextInputDialog dialog = new TextInputDialog();  
 dialog.setTitle("Add Donation");  
 dialog.setHeaderText("Enter Blood Type:");  
 dialog.setContentText("Blood Type:");  
  
 dialog.showAndWait().ifPresent(bloodType -> {  
 try {  
 donorService.*addDonation*(donor.getUserId(), bloodType);  
 showAlert("Success", "Donation added successfully.", Alert.AlertType.*INFORMATION*);  
 } catch (IOException ex) {  
 ex.printStackTrace();  
 showAlert("Error", "Could not add donation.", Alert.AlertType.*ERROR*);  
 }  
 });  
 });  
  
 logoutButton.setOnAction(e -> {  
 primaryStage.close();  
 new Main().start(new Stage());  
 });  
  
 Scene scene = new Scene(layout, 600, 500);  
 primaryStage.setTitle("Donor Dashboard");  
 primaryStage.setScene(scene);  
 primaryStage.show();  
 }  
  
 private void showAlert(String title, String message, Alert.AlertType type) {  
 Alert alert = new Alert(type);  
 alert.setTitle(title);  
 alert.setContentText(message);  
 alert.showAndWait();  
 }  
}

**Donor Dashboard Controller .java**

package com.example.bloodbankjava;  
import javafx.scene.control.Alert;  
  
public class DonorDashboardController {  
 private DonorService donorService = new DonorService();  
  
 public String getDonationHistory(int donorId) {  
 try {  
 return donorService.*getDonationHistory*(donorId);  
 } catch (Exception ex) {  
 ex.printStackTrace();  
 showAlert("Error", "Could not load donation history.", Alert.AlertType.*ERROR*);  
 return null;  
 }  
 }  
 public void addDonation(int donorId, String bloodType) {  
 try {  
 donorService.*addDonation*(donorId, bloodType);  
 showAlert("Success", "Donation added successfully.", Alert.AlertType.*INFORMATION*);  
 } catch (Exception ex) {  
 ex.printStackTrace();  
 showAlert("Error", "Could not add donation.", Alert.AlertType.*ERROR*);  
 }  
 }  
 private void showAlert(String title, String message, Alert.AlertType type) {  
 Alert alert = new Alert(type);  
 alert.setTitle(title);  
 alert.setContentText(message);  
 alert.showAndWait();  
 }  
}

**Hospital Dashboard Controller .java**

package com.example.bloodbankjava;  
  
import javafx.application.Application;  
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.HBox;  
import javafx.scene.layout.Priority;  
import javafx.scene.layout.VBox;  
import javafx.scene.text.Font;  
import javafx.scene.text.FontWeight;  
import javafx.stage.Stage;  
  
class HospitalDashboard extends Application {  
  
 private User hospital;  
 private RequestService requestService = new RequestService();  
  
 public HospitalDashboard(User hospital) {  
 this.hospital = hospital;  
 }  
  
 @Override  
 public void start(Stage primaryStage) {  
 VBox layout = new VBox(20);  
 layout.setPadding(new Insets(20));  
 layout.setAlignment(Pos.*CENTER*);  
 layout.setStyle("-fx-background-color: #f4f4f4;");  
  
 Label titleLabel = new Label("Hospital Dashboard - " + hospital.getName());  
 titleLabel.setFont(Font.*font*("Arial", FontWeight.*BOLD*, 24));  
 titleLabel.setStyle("-fx-text-fill: #333333;");  
  
 Image hospitalImage = new Image("file:C:\\Users\\DELL\\Desktop\\BloodBank-java\\src\\main\\java\\com\\example\\bloodbankjava\\admin.webp"); // Replace with your image path  
 ImageView imageView = new ImageView(hospitalImage);  
 imageView.setFitWidth(200);  
 imageView.setFitHeight(100);  
 imageView.setPreserveRatio(true);  
  
 Button viewRequestsButton = new Button("View Request Status");  
 Button makeRequestButton = new Button("Make New Blood Request");  
 Button logoutButton = new Button("Logout");  
  
 String buttonStyle = "-fx-background-color: #2a4b63; -fx-text-fill: white; -fx-font-size: 16; -fx-padding: 10;";  
 viewRequestsButton.setStyle(buttonStyle);  
 makeRequestButton.setStyle(buttonStyle);  
 logoutButton.setStyle(buttonStyle);  
  
 HBox buttonLayout = new HBox(10, viewRequestsButton, makeRequestButton, logoutButton);  
 buttonLayout.setAlignment(Pos.*CENTER*);  
  
 layout.getChildren().addAll(titleLabel, imageView, buttonLayout);  
  
 viewRequestsButton.setOnAction(e -> viewRequests());  
 makeRequestButton.setOnAction(e -> makeNewRequest());  
 logoutButton.setOnAction(e -> {  
 primaryStage.close();  
 new Main().start(new Stage());  
 });  
  
 Scene scene = new Scene(layout, 600, 500);  
 primaryStage.setTitle("Hospital Dashboard");  
 primaryStage.setScene(scene);  
 primaryStage.show();  
 }  
  
 private void viewRequests() {  
 try {  
 String rawStatus = requestService.getRequestStatusForHospital(hospital.getUserId());  
 if (rawStatus == null || rawStatus.isEmpty()) {  
 showAlert("Request Status", "No requests found.", Alert.AlertType.*INFORMATION*);  
 } else {  
 String[] requests = rawStatus.split(";");  
 StringBuilder formattedStatus = new StringBuilder();  
 for (String request : requests) {  
 String[] details = request.split(",");  
 formattedStatus.append("Blood Type: ").append(details[0].trim()).append("\n")  
 .append("Quantity: ").append(details[1].trim()).append("\n")  
 .append("Priority: ").append(details[2].trim()).append("\n")  
 .append("Status: ").append(details[3].trim()).append("\n")  
 .append("Date: ").append(details[4].trim()).append("\n\n");  
 }  
 showAlert("Request Status", formattedStatus.toString(), Alert.AlertType.*INFORMATION*);  
 }  
 } catch (Exception ex) {  
 ex.printStackTrace();  
 showAlert("Error", "Could not load request status.", Alert.AlertType.*ERROR*);  
 }  
 }  
  
 private void makeNewRequest() {  
 TextInputDialog dialog = new TextInputDialog();  
 dialog.setTitle("New Blood Request");  
 dialog.setHeaderText("Enter Blood Type and Quantity:");  
 dialog.setContentText("Blood Type, Quantity (comma separated):");  
  
 dialog.showAndWait().ifPresent(input -> {  
 try {  
 String[] requestDetails = input.split(",");  
 String bloodType = requestDetails[0].trim();  
 int quantity = Integer.*parseInt*(requestDetails[1].trim());  
  
 requestService.addRequest(new BloodRequest(  
 requestService.getNextRequestId(),  
 hospital.getUserId(),  
 bloodType,  
 quantity,  
 "High", // Default Priority  
 "Pending", // Default Status  
 java.time.LocalDate.*now*()  
 ));  
 showAlert("Success", "Request for " + bloodType + " created successfully.", Alert.AlertType.*INFORMATION*);  
 } catch (Exception ex) {  
 ex.printStackTrace();  
 showAlert("Error", "Could not create request.", Alert.AlertType.*ERROR*);  
 }  
 });  
 }  
  
 private void showAlert(String title, String message, Alert.AlertType type) {  
 Alert alert = new Alert(type);  
 alert.setTitle(title);  
 alert.setContentText(message);  
 alert.showAndWait();  
 }  
}

**Main.java**

package com.example.bloodbankjava;  
import javafx.application.Application;  
import javafx.geometry.Insets;  
import javafx.geometry.Pos;  
import javafx.scene.Scene;  
import javafx.scene.control.\*;  
import javafx.scene.paint.Color;  
import javafx.scene.text.Font;  
import javafx.scene.text.FontWeight;  
import javafx.stage.Stage;  
import javafx.scene.image.Image;  
import javafx.scene.image.ImageView;  
import javafx.scene.layout.HBox;  
import javafx.scene.layout.VBox;  
  
import java.io.\*;  
import java.util.List;  
  
public class Main extends Application {  
 private static final String *FILE\_PATH* = "C:\\Users\\DELL\\Desktop\\BloodBank-java\\src\\main\\java\\com\\example\\bloodbankjava\\user.txt";  
  
 @Override  
 public void start(Stage primaryStage) {  
 HBox mainLayout = new HBox();  
 mainLayout.setAlignment(Pos.*CENTER\_LEFT*);  
 mainLayout.setStyle("-fx-background-color: #000000;");  
  
 VBox formLayout = new VBox();  
 formLayout.setAlignment(Pos.*CENTER\_LEFT*);  
 formLayout.setSpacing(20);  
  
 Label title = new Label("Blood Donation System");  
 title.setFont(Font.*font*("Arial", FontWeight.*BOLD*, 32));  
 title.setTextFill(Color.*web*("#FFFFFF"));  
  
 TextField usernameField = new TextField();  
 usernameField.setPromptText("Enter Username");  
 usernameField.setFont(Font.*font*(16));  
 usernameField.setPrefWidth(300);  
  
 PasswordField passwordField = new PasswordField();  
 passwordField.setPromptText("Enter Password");  
 passwordField.setFont(Font.*font*(16));  
 passwordField.setPrefWidth(300);  
  
 HBox buttonLayout = new HBox();  
 buttonLayout.setAlignment(Pos.*CENTER\_LEFT*);  
 buttonLayout.setSpacing(20);  
  
 Button loginButton = new Button("Login");  
 loginButton.setStyle("-fx-background-color: #0d47a1; -fx-text-fill: white; -fx-font-size: 18; -fx-padding: 10;");  
 loginButton.setPrefWidth(200);  
  
 Button signUpButton = new Button("Sign Up");  
 signUpButton.setStyle("-fx-background-color: #4caf50; -fx-text-fill: white; -fx-font-size: 18; -fx-padding: 10;");  
 signUpButton.setPrefWidth(200);  
  
 buttonLayout.getChildren().addAll(loginButton, signUpButton);  
  
 formLayout.setPadding(new Insets(50));  
 formLayout.setSpacing(30);  
 formLayout.getChildren().addAll(title, usernameField, passwordField, buttonLayout);  
  
 Image image = new Image("file:C:\\Users\\DELL\\Desktop\\BloodBank-java\\src\\main\\java\\com\\example\\bloodbankjava\\WhatsApp Image 2024-12-12 at 20.41.14\_dca6454d.jpg");  
 ImageView imageView = new ImageView(image);  
 imageView.setFitWidth(400);  
 imageView.setFitHeight(600);  
 imageView.setPreserveRatio(true);  
  
 mainLayout.getChildren().addAll(formLayout, imageView);  
  
 signUpButton.setOnAction(e -> {  
 SignUpScreen signUpScreen = new SignUpScreen();  
 signUpScreen.start(new Stage());  
 });  
  
 loginButton.setOnAction(e -> {  
 String username = usernameField.getText();  
 String password = passwordField.getText();  
  
 User loggedInUser = getUserByUsername(username);  
 if (loggedInUser != null && loggedInUser.getPassword().equals(password)) {  
 if ("Hospital".equalsIgnoreCase(loggedInUser.getRole())) {  
 new HospitalDashboard(loggedInUser).start(new Stage());  
 } else if ("Donor".equalsIgnoreCase(loggedInUser.getRole())) {  
 new DonorDashboard(loggedInUser).start(new Stage());  
 } else if ("Admin".equalsIgnoreCase(loggedInUser.getRole())) {  
 new AdminDashboard(loggedInUser).start(new Stage());  
 } else {  
 showAlert("Error", "Unknown role. Please check your credentials.", Alert.AlertType.*ERROR*);  
 }  
 primaryStage.close();  
 } else {  
 showAlert("Error", "Invalid username or password.", Alert.AlertType.*ERROR*);  
 }  
 });  
  
 Scene scene = new Scene(mainLayout, 800, 500);  
 primaryStage.setTitle("Login");  
 primaryStage.setScene(scene);  
 primaryStage.show();  
 }  
  
 private User getUserByUsername(String username) {  
 try {  
 List<String> lines = FileReaderWriter.*readFile*(*FILE\_PATH*);  
  
 for (String line : lines) {  
 String[] parts = line.split(",");  
 if (parts.length == 5) {  
 int id = Integer.*parseInt*(parts[0].trim());  
 String name = parts[1].trim();  
 String role = parts[2].trim();  
 String email = parts[3].trim();  
 String password = parts[4].trim();  
  
 if (username.equals(email)) {  
 return new User(id, name, role, email, password);  
 }  
 }  
 }  
 } catch (Exception e) {  
 e.printStackTrace();  
 }  
 return null;  
 }  
  
 private void showAlert(String title, String message, Alert.AlertType type) {  
 Alert alert = new Alert(type);  
 alert.setTitle(title);  
 alert.setContentText(message);  
 alert.showAndWait();  
 }  
  
 public static void main(String[] args) {  
 *launch*(args);  
 }  
}