

Bahria University,

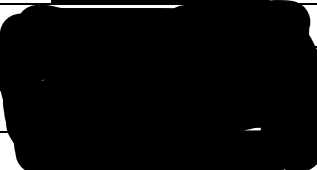
Karachi Campus



COURSE: CSL-210 OBJECT ORIENTED PROGRAMMING
TERM: SPRING 2025, CLASS: BSE- 2 (C)

Military Base Security & Operations Management System

Submitted By:

<u>S.NO</u>	<u>Student Name</u>	<u>Enrollment #</u>
<u>1</u>	Huzaifa Asif	
<u>2</u>	Muhammad Arsalan	
<u>3</u>	Moiz ur Rehman Minhas	
<u>4</u>	Huzaifa AbdulRehman	

Submitted to:

Engr.Muhammad Faisal/ Engr. Saniya Sarim

Signed

Remarks:

Score:

HEADINGS:

Table of Content

Contents

Table of Content	2
INTRODUCTION	2
OOP CONCEPTS USED IN PROJECT	2
UML CLASS DIAGRAM	3
FEATURES OF PROJECT	4
CODE (ONLY MAIN CLASSES AND FORMS)	4
INTERFACES(OUTPUT SCREESHOTS)	7
CONCLUSION	13

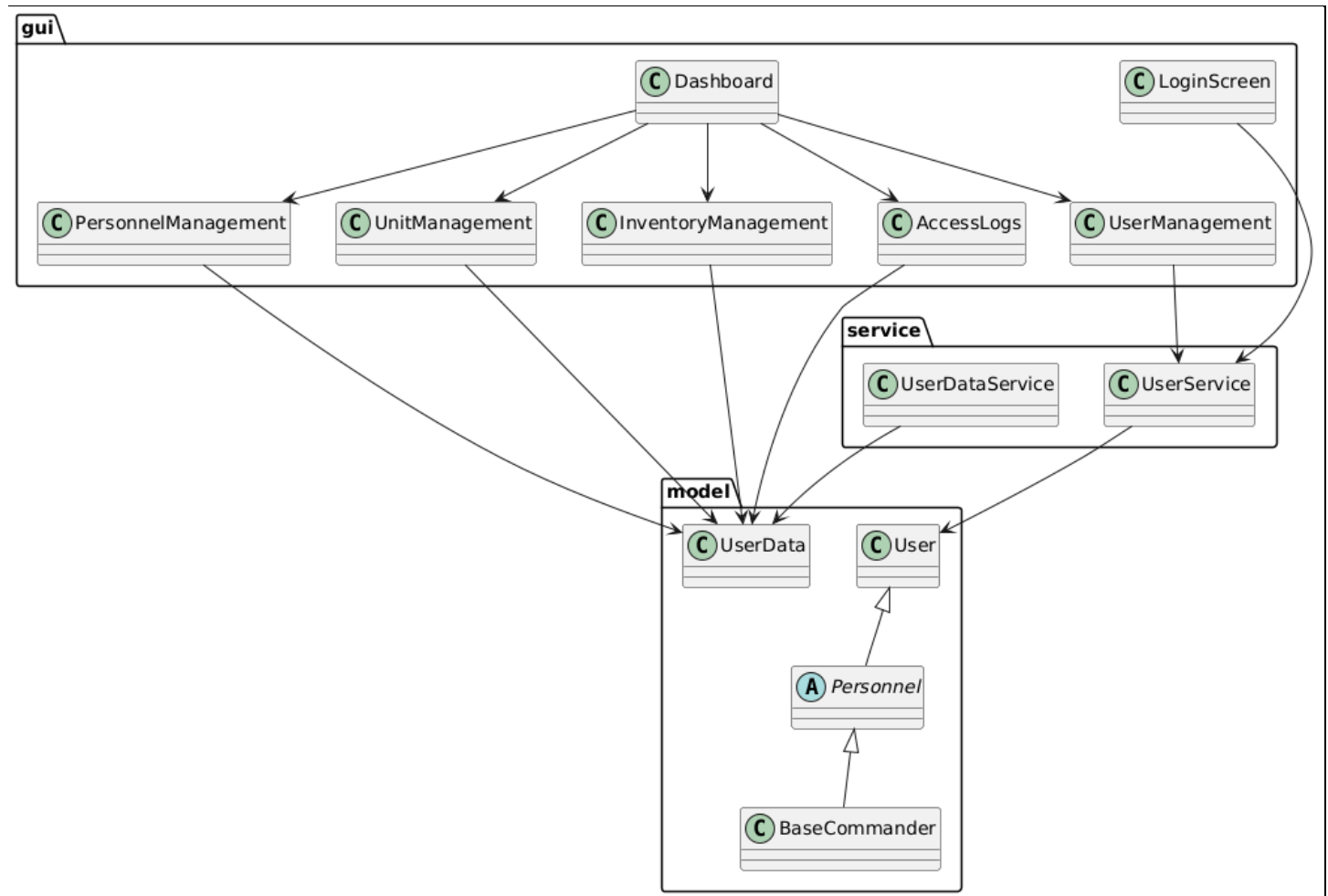
INTRODUCTION

The Military Base Management System is a Java-based desktop application developed as a submission for the Object Oriented Programming Lab. This project simulates the digital management of a military base, including personnel, units, inventory, and security access. The system demonstrates key OOP principles and is built using Java Swing for the graphical user interface.

OOP CONCEPTS USED IN PROJECT

- **Encapsulation:** Data members in classes such as User, Personnel, and UserData are private, with public getters and setters.
- **Inheritance:** BaseCommander extends the abstract class Personnel, and other specific roles can be similarly extended.
- **Abstraction:** Classes like UserService and UserDataService abstract away file operations and business logic from the UI.
- **Polymorphism:** Methods like getPermissions() are overridden in subclasses (BaseCommander).
- **Modularity:** Each feature (Personnel, Inventory, Units, Access Logs, User Management) has its own class and UI panel.

UML CLASS DIAGRAM



Role Based Access Control

Feature/Panel	Personnel	Security Officer	Base Commander
View Personnel List	✓	✓	✓
Add/Edit/Delete Personnel	✗	✗	✓
View Inventory	✓	✓	✓
Add/Edit/Delete Inventory	✗	✓	✓
View Unit List	✓	✓	✓
Add/Edit/Delete Units	✗	✗	✓
View Access Logs	✗	✓	✓
Add Access Log	✗	✓	✓
Delete Access Log	✗	✓	✓
User Management (Add/Remove Users)	✗	✗	✓

FEATURES OF PROJECT

- **User Authentication:** Secure login and registration with role selection (Base Commander, Personnel, etc.)
- **Role-Based Access:** Features and panels are enabled/disabled according to the user's role.
- **Personnel Management:** Add, edit, and delete personnel records.
- **Unit Management:** Manage military units, assign personnel, and track unit details.
- **Inventory Management:** Record and update inventory items, including their count and status.
- **Access Logs:** Log base entries and exits for both personnel and vehicles.
- **User Management:** Only base commanders can create, edit, or delete user accounts.

CODE (ONLY MAIN CLASSES AND FORMS)

Dashboard

```
package com.militarybase.gui;

import javax.swing.*.*;
import com.militarybase.model.User;
import com.militarybase.model.UserData;
```

```

import net.miginfocom.swing.*;

public class Dashboard extends JFrame {
    private JTabbedPane tabbedPane2;
    private JPanel personnelPanel;
    private JPanel logPannel;
    private JPanel inventoryPanel;
    private JPanel unitManagementPanel;
    private UserData userData;
    private User user;

    public Dashboard(User user, UserData userData) {
        this.userData = userData;
        this.user = user;
        initComponents();

        // Set content for each default tab
        PersonnelManagement personnel = new PersonnelManagement(user, userData);
        tabbedPane2.setComponentAt(0, personnel.personnelPanel.getContentPane());

        AccessLogs acl = new AccessLogs(user, userData);
        tabbedPane2.setComponentAt(1, acl.accessLogsPanel);

        InventoryManagement inventoryManagement = new InventoryManagement(user,
        userData);
        tabbedPane2.setComponentAt(2,
        inventoryManagement.inventoryPanel.getContentPane());

        UnitManagement unit = new UnitManagement(user, userData);
        tabbedPane2.setComponentAt(3, unit.unitPanel.getContentPane());

        // Only add User Management tab if Base Commander
        String role = user.getRole().trim().toLowerCase();
        if ("base commander".equals(role) || "commander".equals(role)) {
            tabbedPane2.addTab("User Management", new UserManagement(user));
        }

        this.setLayout(new MigLayout("fill"));
        this.add(tabbedPane2, "grow, push");
        this.pack();
        this.setLocationRelativeTo(null);
    }

    private void initComponents() {
        tabbedPane2 = new JTabbedPane();
        personnelPanel = new JPanel();
        logPannel = new JPanel();
        inventoryPanel = new JPanel();
        unitManagementPanel = new JPanel();

        tabbedPane2.addTab("Personnel Management", personnelPanel);
        tabbedPane2.addTab("Access Logs", logPannel);
        tabbedPane2.addTab("Inventory Managemnt", inventoryPanel);
        tabbedPane2.addTab("Unit Management", unitManagementPanel);
    }
}

```

```

package com.militarybase.model;

import java.io.Serializable;

public class User implements Serializable {
    private String id;
    private String password;
    private String role;

    public User(String id, String password, String role) {
        this.id = id;
        this.password = password;
        this.role = role;
    }

    public String getId() {
        return id;
    }

    public String getPassword() {
        return password;
    }

    public String getRole() {
        return role;
    }
}

```

User Data

```

package com.militarybase.model;

import java.io.Serializable;
import java.util.ArrayList;
import java.util.List;

/*
 * This class store all user-specific info, it gets data from each gui
 */
public class UserData implements Serializable {

    private List<Object[]> personnelRows = new ArrayList<>();
    private List<Object[]> inventoryRows = new ArrayList<>();
    private List<Object[]> unitRows = new ArrayList<>();
    private List<Object[]> accessLogRows = new ArrayList<>();

    public List<Object[]> getPersonnelRows() { return personnelRows; }
    public List<Object[]> getInventoryRows() { return inventoryRows; }
    public List<Object[]> getUnitRows() { return unitRows; }
    public List<Object[]> getAccessLogRows() { return accessLogRows; }

}

```

UserDataService

```
package com.militarybase.service;

import com.militarybase.model.UserData;
import java.io.*;

public class UserDataService {
    public static void saveUserData(String userId, UserData userData) {
        try (ObjectOutputStream out = new ObjectOutputStream(new
FileOutputStream("userdata_" + userId + ".dat"))) {
            out.writeObject(userData);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public static UserData loadUserData(String userId) {
        try (ObjectInputStream in = new ObjectInputStream(new
FileInputStream("userdata_" + userId + ".dat"))) {
            return (UserData) in.readObject();
        } catch (Exception e) {
            return new UserData(); // Return empty if file not found/corrupt
        }
    }
}
```

INTERFACES(OUTPUT SCREESHOTS)

Login Screen


Military Base Security & Operations Management System

Id:

Password:

Type: ▼

Personnel Management



Personnel Management

Access Logs

Inventory Managemnt

Unit Management

User Management

ID	Name	Role
213	Asad	Sentry

ID:

Name:

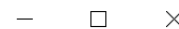
Role:

Add

Edit

Delete

Access Logs



Personnel Management Access Logs Inventory Managemnt Unit Management User Management

Log ID	Personnel Name	ID	Access Type	Gate	Timestamp	Vehicle Plate	Remarks/Purpose
LOG-1	Delivery	23	Commercial an...	Main Gate	2025-06-03 11...	AHK-324	For delivery of ...

Name ID Access Type Gate

Vehicle Plate Remarks/Purpose Timestamp

Personnel Management

Access Logs

Inventory Managemnt

Unit Management

User Management

Inventory Management

Item	Count	Status
G3 Gun	23	Available

Item

Count

Status

Available

▼

Add

Edit

Delete

Unit Managemnet

Personnel Management

Access Logs

Inventory Managemnt

Unit Management

User Management

Unit Management

Unit ID	Unit Name	Type	Commander	Number of ...	Status
2321	FP BN	23	Security	Rab Nawaz	Active

ID

Name

Type

Commander

Number

Status

Active

Add Unit

Edit Unit

Delete Unit

JFormDesigner Evaluation

User Management

Personnel Management

Access Logs

Inventory Managemnt

Unit Management

User Management

User ID	Role
123	Personnel
admin	Base Commander
Arsalan	personnel
Moiz4sm	security
Huzaifa	commander
1234	Personnel
abcd	Base Commander

ID:

Password:

Role:

personnel

Add User

Delete User

CONCLUSION

The Military Base Management System effectively demonstrates the principles of Object Oriented Programming in a real-world scenario. Through modular design, encapsulation, inheritance, and abstraction, the project delivers a secure and organized management solution suitable for a military context. Collaborative development enabled clear separation of concerns, and each module was independently tested and integrated. The project fulfills all OOP Lab requirements and serves as a valuable learning experience in software engineering and teamwork.