

Tishk International University – Sulaimani

Relevant Courses:

Database II

and

Programming Paradigm, Object Oriented Programming Course

Report Title:

Inventory Management System

**Prepared by:**

Lana Abdulla

Adam Muhammed

*Course Lecturer:*

Ms. Yusra Mohammed & Ms. Bakhan

*Assigned Date:*

May 18, 2025

*Submission Due:*

May 27, 2025

**Table of Contents**

[**1.** **Introduction** 2](#_Toc199231421)

[**2.** **Literature Review of the Project** 2](#_Toc199231422)

[**3.** **Problem statement (Define the problems you want to solve)** 2](#_Toc199231423)

[**4.** **Proposed Solution** 3](#_Toc199231424)

[**5.** **Methodology** 3](#_Toc199231425)

[**5.1.** **UML Diagram** 3](#_Toc199231426)

[**5.2.** **Database ER & EER Diagram** 3](#_Toc199231427)

[**5.3.** **Java Classes** 4](#_Toc199231428)

[**5.4.** **GIU Design** 4](#_Toc199231429)

[**6.** **Implementation:** 5](#_Toc199231430)

[**6.1. Workflow diagram of the system** 5](#_Toc199231431)

[**6.2. SQL Command** 6](#_Toc199231432)

[**6.3. Java Classes** 7](#_Toc199231433)

[**6.4. Windows Frame snapshot based on the workflow diagram** 8](#_Toc199231434)

[**7.** **Coding Language Conversion** 9](#_Toc199231435)

[**8.** **Conclusion and Future Work** 9](#_Toc199231436)

[**9.** **References** 9](#_Toc199231437)

# **Introduction**

This project integrates three core courses: Object-Oriented Programming (OOP), Database II, and Programming Paradigm into a unified Inventory Management System. The system supports two user roles: Admin and User, and provides functionality for managing inventory entries, exports, and user accounts. It uses Java Swing for the graphical interface and MySQL as the backend database.

The system handles multiple accounting types (e.g., Accounting 70 for inbound inventory, 71A for direct distribution), user authentication, and secure database operations using JDBC and prepared statements .

# **Literature Review of the Project**

Existing inventory systems often focus on either transactional efficiency or user role-based access control. However, this project combines both aspects with a strong emphasis on OOP design patterns, secure database practices, and GUI usability. The use of UML diagrams and ER models ensures a well-documented and maintainable architecture, aligning with modern software engineering standards .

# **Problem statement (Define the problems you want to solve)**

The manual tracking of inventory in government warehouses leads to inefficiencies, errors, and lack of accountability. There is a need for a centralized system that supports:

- Secure login and role-based access

- Multiple accounting types (Accounting 70, 71, 71A, 13, 16)

- Real-time data entry and reporting

- User management (Add/Delete/Update)

- Integration with a relational database for auditing and querying

# **Proposed Solution**

The manual tracking of inventory in government warehouses leads to inefficiencies, errors, and lack of accountability. There is a need for a centralized system that supports:

- Secure login and role-based access

- Multiple accounting types (Accounting 70, 71, 71A, 13, 16)

- Real-time data entry and reporting

- User management (Add/Delete/Update)

- Integration with a relational database for auditing and querying

# **Methodology**

# **UML Diagram**

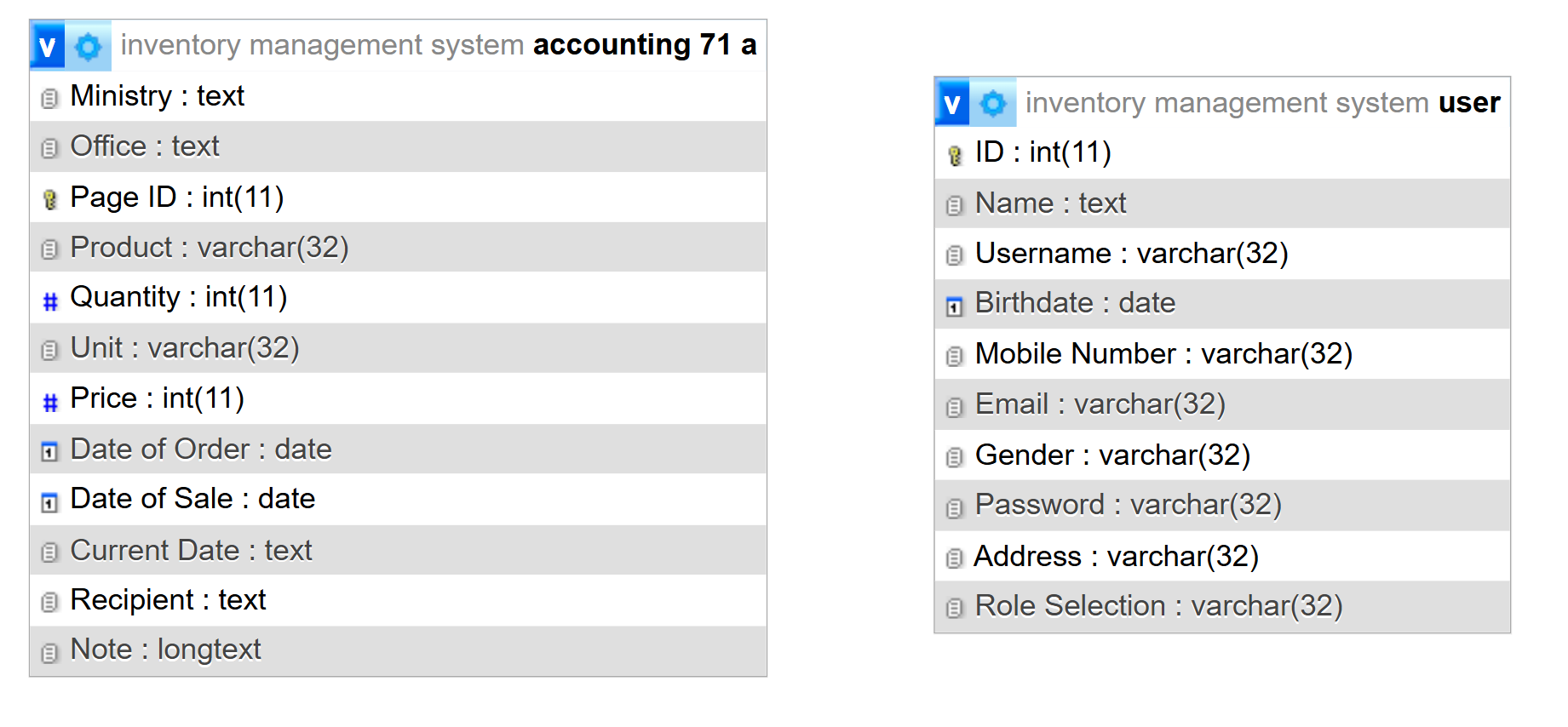
Includes:

- Class Diagram: Shows relationships between `Login`, `User\_Dashboard`, `Import`, `Users`, etc.

- Use Case Diagram: Admin vs. User functionalities

- Sequence Diagram: Login process, data entry, and database interaction

# **Database ER & EER Diagram**



# **Java Classes**

Based on the code files you provided, the following classes are used:

- `Login.java`: Handles authentication

- `User\_Dashboard.java` / `Admin\_Dashboard.java`: Role-based UI

- `Import\_GUI.java` / `Import.java`: Data entry for inventory

- `View\_Accounting\_71A.java`: Query and display records

- `ManageUsers.java` / `Users.java`: User management

These classes demonstrate encapsulation, inheritance, and polymorphism .

# **GIU Design**

The system uses Java Swing for the GUI:

- Buttons: `jButton1`, `saveButton`, `deleteButton`

- Text fields: `usernameField`, `ProductNameField`, `QuantityField`

- Menus: `jMenu1`, `jMenuFile`, `jMenuItemExit`

- Tabbed Panels: `TabbedPane` for Accounting 70, 71A, etc.

- Date Pickers: `DateChooser`, `Date\_Of\_OrderChooser`

# **Implementation:**

## **6.1. Workflow diagram of the system**

Example: Adding an Import Record

Start

↓

Open Import Form

↓

Enter Data (Product, Quantity, Price, Dates, etc.)

↓

Validate Input

↓

[No] ← Is Input Valid? → [Yes]

↓ ↓

Show Error Message Save to Database

↓

Show Success Message

↓

Clear Form Fields

↓

End

## **6.2. SQL Command**

sql

-- Create User Table

CREATE TABLE `users` (

`ID` int NOT NULL AUTO\_INCREMENT,

`username` varchar(45) NOT NULL,

`password` varchar(45) NOT NULL,

`role` varchar(45) NOT NULL,

`birthdate` date NOT NULL,

PRIMARY KEY (`ID`)

);

-- Insert Sample User

INSERT INTO `users` (`username`, `password`, `role`, `birthdate`)

VALUES ('admin', 'admin123', 'Admin', '1990-01-01');

-- Insert Accounting 71A Record

INSERT INTO `accounting 71 a` (`Ministry`, `Office`, `Product`, `Quantity`, `Unit`, `Price`, `Date of Order`, `Date of Sale`, `Recipient`, `Note`)

VALUES ('Ministry of Transport', 'Office A', 'Steel Bars', 100, 'Tons', 5000.00, '2025-05-20', '2025-05-22', 'Department A', '');

```

## **6.3. Java Classes**

Example: `Import.java`

```java

public class Import {

private String ministry;

private String office;

private String product;

private int quantity;

private String unit;

private double price;

private Date dateOfOrder;

private Date dateOfSale;

private String recipient;

private String note;

public void addImport() {

try {

String sql = "INSERT INTO `accounting 71 a` (...) VALUES (?,?,...)";

PreparedStatement pst = con.prepareStatement(sql);

// Set parameters

pst.executeUpdate();

} catch (SQLException e) {

JOptionPane.showMessageDialog(null, "Error: " + e.getMessage());

}}}

## **6.4. Windows Frame snapshot based on the workflow diagram**

Example:

- Login Screen: Username, Password, Login Button

- Admin Dashboard: Manage Users, Import Records, View Records

- Import Form: Fields for Ministry, Office, Product, etc.

- View Records: JTable displaying data from `accounting 71 a`

# **Coding Language Conversion**

The system uses Java for GUI and business logic, and SQL for database operations. There is a clear separation between the presentation layer (Swing) and the data layer (MySQL). The use of prepared statements ensures secure communication between Java and SQL .

# **Conclusion and Future Work**

This project successfully integrates OOP principles, database design, and GUI programming into a functional Inventory Management System. It supports secure login, multiple accounting types, and user management.

Future Work:

- Add export to Excel/PDF functionality

- Integrate mobile compatibility

- Implement advanced reporting features

- Add real-time inventory tracking

# **References**

* Barewbarayate benasazi slimanyi
* Al-khalifa, K. N., & Aspinwall, E. M. (2000). TQM age versus quality: An empirical investigation . Production & Inventory Management Journal, 1, 18–23.
* Baral, P., Larsen, M., & Archer, M. (2019). Does money grow on trees? (Report No. 12345). Washington, DC: World Bank Publications.
* Emerald Insight. (n.d.). REFERENCES: Inventory Management Models: A Tutorial . [https://www.emeraldinsight.com](https://www.emeraldinsight.com/)