**Table of contents:**

Contents

[Chapter 1: 2](#_Toc198215099)

[About the Project 2](#_Toc198215100)

[Group Members: 3](#_Toc198215101)

[Chapter 2 3](#_Toc198215102)

[Use Case Diagram: 3](#_Toc198215103)

[FA22 BSE 059 – Syed Shah Mansoor Trimzi 3](#_Toc198215104)

[SP23 BSE 096 – Murtaza Tanveer 5](#_Toc198215105)

[SP23 BSE 076 – Naseem Shahid 6](#_Toc198215106)

[FA22 BSE 074 – Amanullah Khan 6](#_Toc198215107)

[Chapter 3 7](#_Toc198215108)

[Full Dressed Usecase: 7](#_Toc198215109)

[FA22 BSE 059 – Syed Shah Mansoor Trimzi 8](#_Toc198215110)

[SP23 BSE 096 – Murtaza Tanveer 9](#_Toc198215111)

[SP23 BSE 076 – Naseem Shahid 11](#_Toc198215112)

[FA22 BSE 074 – Amanullah Khan 12](#_Toc198215113)

[Chapter 4 14](#_Toc198215114)

[SSD 14](#_Toc198215115)

[FA22 BSE 059 – Syed Shah Mansoor Trimzi 14](#_Toc198215116)

[SP23 BSE 096 – Murtaza Tanveer 15](#_Toc198215117)

[SP23 BSE 076 – Naseem Shahid 16](#_Toc198215118)

[FA22 BSE 074 – Amanullah Khan 17](#_Toc198215119)

[Chapter 5 18](#_Toc198215120)

[Class Diagram: 18](#_Toc198215121)

[FA22 BSE 059 – Syed Shah Mansoor Trimz 18](#_Toc198215122)

[SP23 BSE 096 – Murtaza Tanveer 20](#_Toc198215123)

[SP23 BSE 076 – Naseem Shahid 20](#_Toc198215124)

[FA22 BSE 074 – Amanullah Khan 21](#_Toc198215125)

# Chapter 1:

## About the Project

Inventory Management System - Java + MySQL

This is a simple Inventory Management System designed for grocery stores, built as a Software Design and Architecture semester project by a team of four. It follows proper Object-Oriented Programming, UML modeling, and design principles to support real-world maintainability and scalability.

Key Modules

The system is divided into 7 functional modules, each mapped to a clear use case and independently implementable:

Login / Logout

Role-based authentication (Admin, Employee)

Session management

Account Management

Admin can create, edit, or delete employee accounts

Enforces role-based access

Add Item to Inventory

Add new grocery items with attributes like name, quantity, category, price

Delete Item from Inventory

Delete outdated or discontinued items by Admin

Search Item

Search inventory by name, category, or ID

View Inventory

Display complete list of items with filtering options

Generate Reports

Create summary reports based on stock, activity, and item trends

⚙Tech Stack

Layer Technology

Language Java (JDK 17+)

Database MySQL

UI Java Swing

Database Access JDBC

Design Tools Draw.io / Lucidchart for UML

Design Methodology

Object-Oriented Programming (O

P)

We used OOP to ensure modularity, reusability, and clean separation of concerns:

Encapsulation: Each module has its own data and behavior

Abstraction: Users interact with high-level operations

Inheritance: Roles (e.g., Admin, Employee) can extend base User

Polymorphism: Future enhancements can allow custom UI actions per role

Software Design Principles & Patterns

Single Responsibility Principle (SRP): Each class has a well-defined job

GRASP Patterns:

Controller: Each module has its own controller class

Creator: DAOs are responsible for creating model objects

Information Expert: Logic resides in the class that has the data

MVC Architecture:

Model: User, Item, Inventory

View: LoginUI, InventoryUI, etc.

Controller: LoginController, InventoryController

UML Diagrams (Created)

For each use case (module), the following were developed:

General Use Case Diagram (with <<include>> and <<extend>> as needed)

Fully Dressed Use Cases (structured table format)

System Sequence Diagrams (SSD)

Communication Diagrams (based on GRASP)

Class Diagrams (showing structure and relationships)

Roles & Access Control

Role Capabilities

Admin Full access: account mgmt, inventory CRUD, reports

Employee Limited access: login, view/search inventory, add items

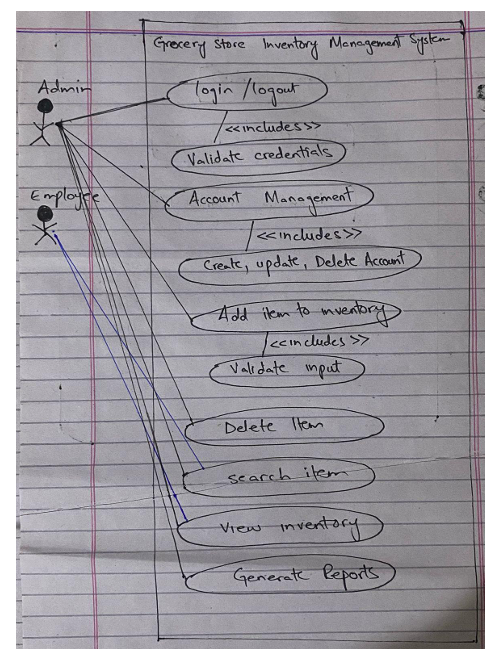
## Group Members:

* FA22 BSE 059 – Syed Shah Mansoor Trimzi
* SP23 BSE 096 – Murtaza Tanveer
* SP23 BSE 076 – Naseem Shahid
* FA22 BSE 074 – Amanullah Khan

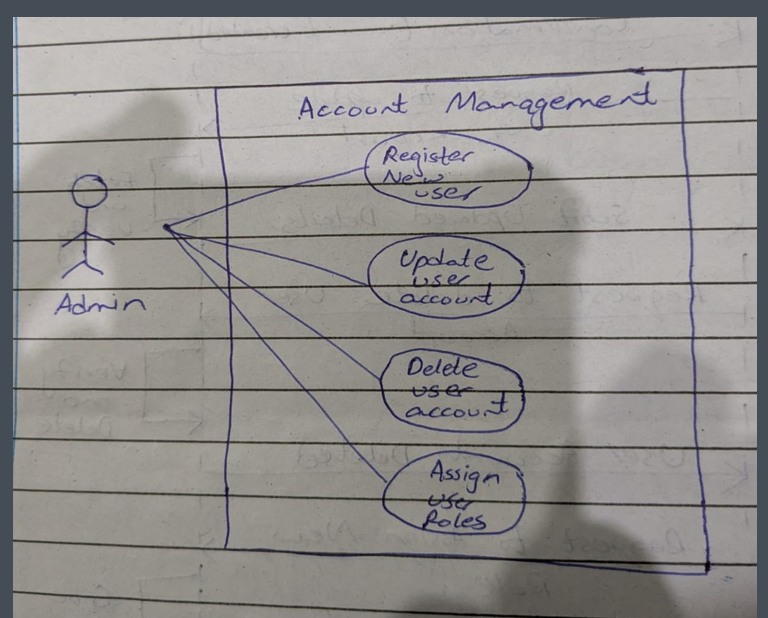
# Chapter 2

## Use Case Diagram:

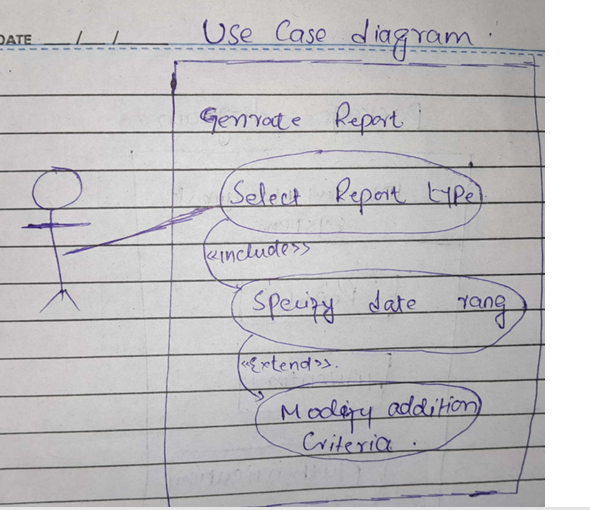
### FA22 BSE 059 – Syed Shah Mansoor Trimzi

****

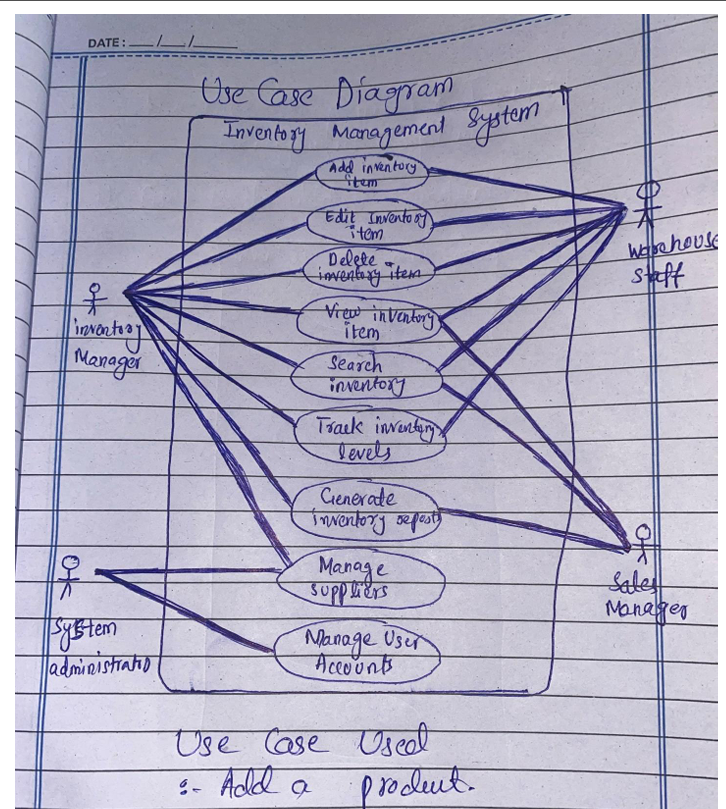
### SP23 BSE 096 – Murtaza Tanveer



### SP23 BSE 076 – Naseem Shahid

****

### FA22 BSE 074 – Amanullah Khan



# Chapter 3

## Full Dressed Usecase:

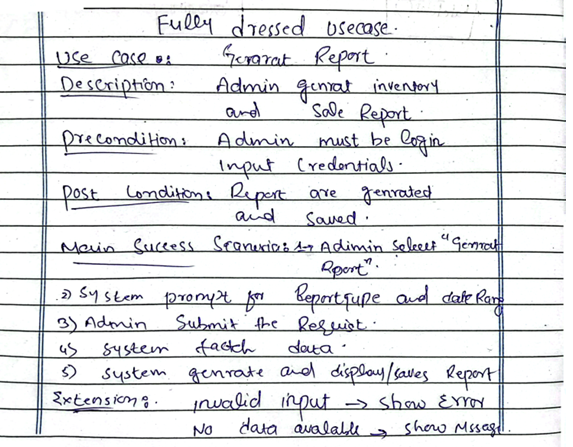
### FA22 BSE 059 – Syed Shah Mansoor Trimzi

Fully Dressed Use Case: Login / Logout Use Case Name: Login / Logout Scope: Inventory Management System for Grocery Store Level: User goal Primary Actor: Admin, Employee Stakeholders and Interests: ● Admin / Employee: Want secure access to system functions. ● System: Needs to verify identity and restrict unauthorized access. Preconditions: ● User has an existing account in the system. ● System is running and ready for interaction. Postconditions: ● On successful login, user is granted access to authorized functions. ● On logout, the user's session is terminated securely. Main Success Scenario (Basic Flow): 1. User opens the system and chooses to log in. 2. System prompts for username and password. 3. User enters valid credentials. 4. System validates credentials. 5. System identifies the user role (Admin or Employee). 6. System grants access to role-specific features. 7. User uses the system. 8. User chooses to logout. 9. System ends the session and returns to login screen. Extensions (Alternative Flows): 3a. Invalid Credentials: ● 3a1. System detects invalid username or password. ● 3a2. System displays error message. ● 3a3. User is prompted to re-enter credentials. ● Return to Step 2. 5a. User Role Not Recognized: ● 5a1. System fails to determine user role. ● 5a2. System displays access denied message. ● 5a3. Session is terminated. ● Use Case ends. Special Requirements: ● Passwords must be securely stored (e.g., hashed). ● Login attempts may be limited (optional). ● Session timeout after inactivity (optional). ● UI should clearly distinguish between Admin and Employee post-login.

### SP23 BSE 096 – Murtaza Tanveer

Primary Actor: AdminSecondary Actors: System, Email ServicePreconditions:Admin is logged in with sufficient privileges.System is operational.Main Success Scenario:1. Admin selects an account management action (Register/Update/Delete/Assign Roles).2. System displays the relevant form based on the action:Register New User: Name, email, role fields.Update User: Editable fields (e.g., name, email).Delete User: Confirmation dialog.Assign Roles: Role dropdown list.3. Admin enters required details and submits the request.4. System validates inputs:For registration: Checks email uniqueness.For updates: Verifies user exists.For deletion: Confirms no active dependencies.For role assignment: Ensures admin has permission.5. System processes the request:Creates/updates/deactivates the user account or updates roles.Sends notifications (e.g., welcome email for new users).6. System confirms success to the Admin.Postconditions:• For registration: New user is added to the system.• For updates: User details are modified.• For deletion: User account is deactivated/removed.• For role assignment: User permissions are updated.

### SP23 BSE 076 – Naseem Shahid



### FA22 BSE 074 – Amanullah Khan

Fully Dressed Use Case: Add Item to Inventory

Use Case Name Add Item to Inventory

Primary Actor Admin or Employee

Stakeholders and Interests Admin and Employee want to maintain accurate stock levels. The organization wants reliable inventory data for operations, sales, and reordering.

Preconditions - User must be logged in

- User must have appropriate role (Admin/Employee)

Postconditions - New item is saved in the database

- Inventory is updated with the new item

Main Success Scenario (Basic Flow)

The user selects "Add Item" from the dashboard.

The system displays the item input form (name, quantity, price, category, etc.).

The user fills in all the required fields.

The user clicks the "Add" or "Save" button.

The system validates the input.

The system saves the new item to the inventory database.

The system confirms the addition and updates the inventory list.

The user sees a success message.

Extensions (Alternate Flows)

5a. Input is invalid (e.g., missing name or negative quantity)

The system highlights the invalid fields and shows error messages.

The user corrects the input.

Returns to Step 4.

6a. Database connection fails

The system displays an error message: "Failed to save item. Please try again later."

The system logs the error.

Use case ends with failure.

Special Requirements

Form must validate input before submission.

Quantity must be numeric and non-negative.

Item name should be unique (or warn about duplicates).

The system must log all item additions with timestamps and user IDs for audit.

🔁 Frequency of Use

Frequently used by Admin and Employees during restocking or item updates.

Technology and Data Variations

Inventory is stored in a MySQL database.

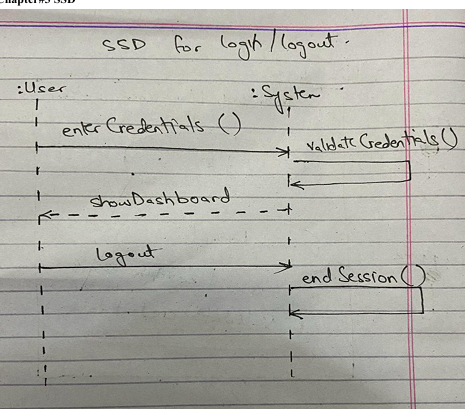
Item addition form is implemented in a Java Swing UI.

Backend interacts with the database using JDBC.

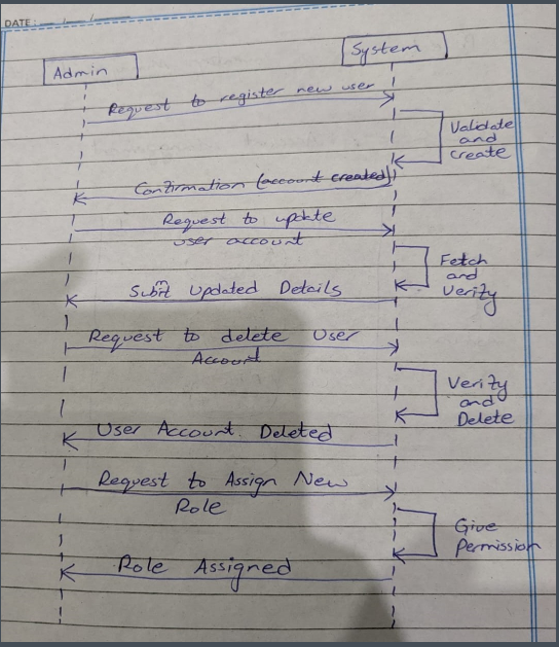
# Chapter 4

## SSD

### FA22 BSE 059 – Syed Shah Mansoor Trimzi



### SP23 BSE 096 – Murtaza Tanveer

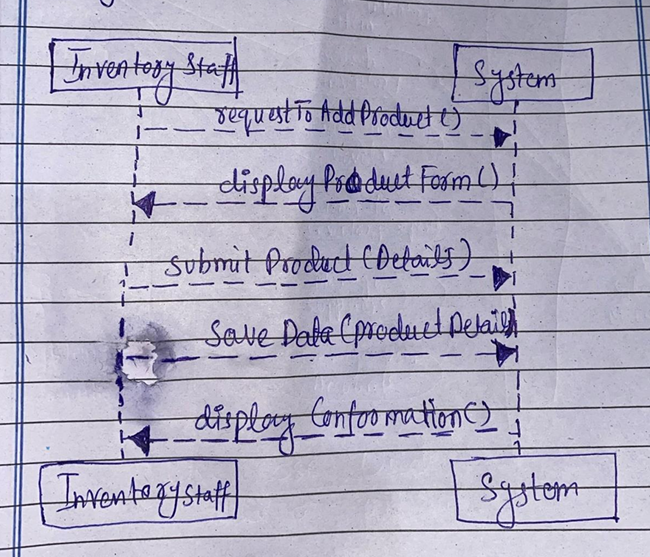


### SP23 BSE 076 – Naseem Shahid

A paper with writing on it

AI-generated content may be incorrect.

### FA22 BSE 074 – Amanullah Khan



# Chapter 5

## Class Diagram:

### FA22 BSE 059 – Syed Shah Mansoor Trimz

A paper with writing on it

AI-generated content may be incorrect.

### SP23 BSE 096 – Murtaza Tanveer

A paper with writing on it

AI-generated content may be incorrect.

### SP23 BSE 076 – Naseem Shahid

A diagram on a lined paper

AI-generated content may be incorrect.

### FA22 BSE 074 – Amanullah Khan

A piece of paper with writing on it

AI-generated content may be incorrect.