Project Report: RetailPOS System

# 1. Project Overview

The Fyloqe POS system was developed as a desktop-based application for managing retail store transactions. It supports core functions such as sales billing, inventory control, returns, and analytics reporting. Built using Java and PostgreSQL, with user interfaces crafted through Screen Builder, the system provides a real-time, responsive, and user-friendly experience tailored for small to mid-sized retail environments.

# 2. Objective

The primary objective was to create a robust, easy-to-use, and efficient POS application that simplifies store operations and provides real-time insights into sales and stock levels. The project aimed to reduce manual effort, eliminate billing errors, and ensure fast checkouts with reliable data storage and retrieval.

# 3. Technologies Used

- Java for core application logic and user interface

- PostgreSQL for relational database management

- Screen Builder for designing intuitive user interfaces

- Eclipse IDE for Java development

- pgAdmin for managing the PostgreSQL database

- Agile methodology with sprint planning and user stories

# 4. Modules Implemented

- Sell Module: Barcode scanning, product selection, real-time price calculation, tax and discount handling, and payment support

- Return Module: Initiates product return and updates inventory and sales records

- Inventory Module: Allows stock entry, editing, and removal; real-time updates after transactions

- Report Module: Generates daily, weekly, and monthly sales reports with filters by category or payment method

- Admin Module: Role-based access, settings, and operational control

# 5. System Design

The application follows a modular architecture. Each module operates independently but interacts through a central database to maintain consistency. The GUI screens were built using Screen Builder to ensure rapid design and high usability. The backend handles data manipulation and communication with the PostgreSQL database via JDBC.

# 6. Database Design

- Tables: Products, Sales, Returns, Users, Stock, Transactions

- Each transaction is logged with timestamp, user ID, items sold, payment method, and tax

- Stock table updated in real time to reflect additions or sales

# 7. Individual Responsibilities

- Designed system architecture and built modular codebase in Java

- Created all GUI screens using Screen Builder with consistent UX design patterns

- Designed normalized database schema in PostgreSQL

- Developed SQL queries for sales, returns, and inventory control

- Created test cases and user documentation for usability and compliance

# 8. Challenges and Solutions

- Ensuring real-time inventory update: Solved by using database triggers and optimized SQL queries

- Building dynamic reports: Implemented print preview panels and parameter-based filters

- Maintaining performance under data load: Indexed key columns and minimized I/O operations

# 9. Outcomes and Impact

- 60% reduction in manual processing time at checkout

- Enhanced inventory accuracy through real-time tracking

- Provided reliable, exportable sales insights to store managers

- Delivered project within timeline across three development sprints

# 10. Skills Gained

- Advanced Java and GUI application development

- PostgreSQL query optimization and schema design

- User experience design using Screen Builder

- Agile project tracking and sprint documentation