Nexus Retail – Smart Retail Management Software

Final Year B.Tech Project Documentation  
  
Department of Computer Science and Engineering  
LLOYD INSTITUTE OF ENGINEERING AND TECHNOLOGY  
  
Team Members:  
Om Sharma – Backend Development, Inventory Module  
Shivansh Pratap Singh – CRM Module, Database Integration  
Tilak Saini – Frontend Development, UI/UX Design  
Harsh Mishra – Testing, Documentation, Sales Report

# 1. Recommended SDLC Model

## Chosen Model: Iterative Incremental Model

The Iterative Incremental Model is selected for Nexus Retail as it supports incremental releases, allows early deployment of core modules, facilitates continuous stakeholder feedback, and reduces risk by integrating modules gradually. This model aligns well with the MERN stack and agile practices.

## Justification

- Complex feature set benefits from incremental development.  
- Early delivery of core features while adding advanced modules later.  
- Continuous stakeholder feedback integration.  
- Supports modular MERN stack development.

# 2. Phases of the Iterative Incremental Model for Nexus Retail

## Requirement Gathering

Stakeholder interviews to define functional and non-functional requirements.

## System Design

High-level architecture and low-level schema design for frontend, backend, and database.

## Implementation

Frontend in React.js, Backend in Node.js/Express, MongoDB database, Firebase Auth.

## Testing

Unit, Integration, System, and User Acceptance Testing.

## Deployment

Frontend on Vercel, Backend on Render/Railway, DB on MongoDB Atlas.

## Maintenance

Bug fixes, performance optimization, and new feature addition.

# 3. Software Requirements Specification (SRS) – IEEE Format

## 1. Introduction

Purpose: Automate retail operations with GST billing, real-time inventory, CRM, analytics, and multi-store management.

Scope: Streamline retail operations, reduce manual errors, and provide real-time insights.

Definitions: CRM – Customer Relationship Management, GST – Goods and Services Tax, RBAC – Role-Based Access Control.

## 2. Overall Description

Product Perspective: Cloud-based modular system integrating all retail functions.

Product Functions: User Authentication (Admin, Staff, Customer), Billing, inventory management, CRM, analytics, expense tracking, role-based access, multi-store support.

User Classes: Admin, Staff, Customer.

Constraints: Requires internet connection, browser-based.

## 3. Functional Requirements

1. GST billing and invoicing.  
2. Inventory auto-update with alerts.  
3. CRM and promotions.  
4. Sales reporting.  
5. Role-based access control.

## 4. Non-Functional Requirements

Security: JWT & Firebase Auth.  
Performance: <2s API latency.  
Scalability: Multi-store.  
Reliability: 99.9% uptime.

## 5. External Interface Requirements

UI: Responsive React.js interface.  
API: REST endpoints.  
Hardware: Barcode scanner, thermal printer.  
Software: Modern browser.

## 6. Database Design Overview

Collections: users, products, sales, expenses, loyalty.  
Each collection stores specific business data for operations.

# 4. Use Case Descriptions

## Admin: Manage Inventory

Actors: Admin

Preconditions: Admin logged in

Postconditions: Inventory updated

Main Flow:

* - Navigate to Inventory Module
* - Add/Edit/Delete product
* - Save changes

Alternative Flows:

* - If product exists, update instead of adding

## Staff: Generate Invoice

Actors: Staff

Preconditions: Staff logged in, products exist

Postconditions: Invoice generated, stock updated

Main Flow:

* - Scan product barcode
* - Add products to invoice
* - Generate invoice

Alternative Flows:

* - Apply discounts if applicable

## Customer: View Purchase History

Actors: Customer

Preconditions: Customer logged in

Postconditions: Purchase history displayed

Main Flow:

* - Navigate to My Orders
* - Select date range
* - View/download invoices

Alternative Flows:

* - Show empty state if no purchase history

Use Case Diagram (Text-Based)

* +-------------+
* | Admin |
* +-------------+
* / | \
* Manage Inventory CRM Analytics
* +-------------+
* | Staff |
* +-------------+
* | \
* Inventory Billing
* +-------------+
* | Customer |
* +-------------+
* |

View Purchase History

Project Structure

* nexus-retail/
* │
* ├── README.md # Project overview, setup guide
* ├── LICENSE # MIT license
* ├── .gitignore # Ignore node\_modules, env files
* ├── package.json # Root scripts (optional)
* ├── docs/ # All documentation (SDLC, SRS, diagrams)
* │ ├── SDLC.md
* │ ├── SRS.md
* │ ├── usecase-diagrams/
* │ └── architecture.png
* │
* ├── frontend/ # React.js Frontend
* │ ├── package.json
* │ ├── public/
* │ │ ├── index.html
* │ │ └── favicon.ico
* │ └── src/
* │ ├── assets/ # Images, icons, styles
* │ ├── components/ # Reusable UI components
* │ │ ├── Navbar.jsx
* │ │ ├── Sidebar.jsx
* │ │ ├── Footer.jsx
* │ │ └── Loader.jsx
* │ ├── pages/ # Page-level components
* │ │ ├── Dashboard.jsx
* │ │ ├── Inventory.jsx
* │ │ ├── Billing.jsx
* │ │ ├── CRM.jsx
* │ │ ├── Reports.jsx
* │ │ └── Login.jsx
* │ ├── context/ # React Context for global state
* │ │ └── AuthContext.jsx
* │ ├── services/ # API calls to backend
* │ │ ├── api.js
* │ │ └── inventoryService.js
* │ ├── hooks/ # Custom React hooks
* │ ├── utils/ # Helper functions
* │ ├── App.jsx
* │ └── index.js
* │
* ├── backend/ # Node.js + Express Backend
* │ ├── package.json
* │ ├── server.js # Entry point
* │ ├── config/
* │ │ ├── db.js # MongoDB connection
* │ │ └── firebase.js # Firebase Auth config
* │ ├── models/ # Mongoose schemas
* │ │ ├── User.js
* │ │ ├── Inventory.js
* │ │ ├── Invoice.js
* │ │ ├── Customer.js
* │ │ └── Expense.js
* │ ├── routes/ # API endpoints
* │ │ ├── authRoutes.js
* │ │ ├── inventoryRoutes.js
* │ │ ├── billingRoutes.js
* │ │ ├── crmRoutes.js
* │ │ └── reportRoutes.js
* │ ├── controllers/ # Business logic
* │ │ ├── authController.js
* │ │ ├── inventoryController.js
* │ │ ├── billingController.js
* │ │ ├── crmController.js
* │ │ └── reportController.js
* │ ├── middleware/ # Auth, error handling
* │ │ ├── authMiddleware.js
* │ │ └── errorHandler.js
* │ ├── utils/ # Utility functions
* │ │ └── generateInvoice.js
* │ └── tests/ # Jest test cases
* │
* ├── .env.example # Example env vars
* │
* └── deployment/ # Deployment configs
* ├── vercel.json
* ├── render.yaml

└── firebase.json

**Team Member Work Division (Code-wise)**

| **Member** | **Assigned Area** | **Example Folders/Files** |
| --- | --- | --- |
| **Om Sharma** | Backend Inventory Module | backend/models/Inventory.js, backend/routes/inventoryRoutes.js, backend/controllers/inventoryController.js |
| **Shivansh Pratap Singh** | CRM & Billing Module, DB Integration | backend/routes/crmRoutes.js, backend/routes/billingRoutes.js, backend/models/Customer.js |
| **Tilak Saini** | Frontend UI/UX | frontend/src/components/, frontend/src/pages/, frontend/src/assets/ |
| **Harsh Mishra** | Testing & Reports | backend/tests/, backend/routes/reportRoutes.js, docs/ |