Coffee Shop Manager

System Requirements Specification V3

Introduction 2
User Characteristics 2
Customer 2
Barista 2
Manager 3
Financial Manager 3
User Stories 3
Customer 3
Barista 3
Manager 4
Financial Manager 4
UML Diagram 4
Use Case Diagram 5
Functional Requirements 7
Point of Sale (POS) 7
Inventory Management 8
Employee Management 8
Customer Relationship Management 8
Data Analytics & Reporting 8
Online & Mobile Ordering 9
Quality Requirements 9
Architectural Requirements 9
Constraints 10
Technology Requirements 10
Deployment Model 11
Service Contracts (API Spec) 11
Coding Standards 12
Technical Installation Manual 12
User Manual 13
Testing 13
CI/CD 13
Security & Roles 14
Wow Factors 14
Versioning 15

Introduction

Small coffee shops often struggle with subpar or fragmented tools that don't fit their use case. This project aims to solve that by designing an integrated system that includes:

- Point-of-sale (POS)
- Inventory management
- Multiple roles (Customer, Barista, Manager, Financial Manager)
- Mobile application for online ordering

User Characteristics

Customer

- Walk-in or online user with basic to moderate tech skills
- Uses mobile or web app to place orders, manage account, track loyalty points
- Expects fast, responsive UI and clear feedback
- High priority on usability and convenience

Barista

- Oversees inventory, staff schedules, and operations
- Uses admin dashboard tools for real-time monitoring
- Elevated access to view/edit product data and run reports
- Good system proficiency but may need training

Manager

- Monitors sales and inventory
- Can assign employee roles and permissions
- Requires timely, accurate reports

Financial Manager

- Less frequent user, mainly for analytics and reporting
- Needs access to daily/monthly performance metrics
- Works with visualized data and CSV exports
- High need for data accuracy and availability

User Stories

Customer

- Register and create an account
- Log in to view past orders, loyalty points, place orders
- Reset password independently
- Access and update account profile
- Browse full menu and descriptions
- Order coffee via mobile to skip the queue

Barista

- See incoming orders in a queue
- Update order status for real-time feedback

- Notify customers when order is ready
- Monitor stock levels and mark items “out of stock”
- Print receipts after payment
- Apply loyalty points discounts

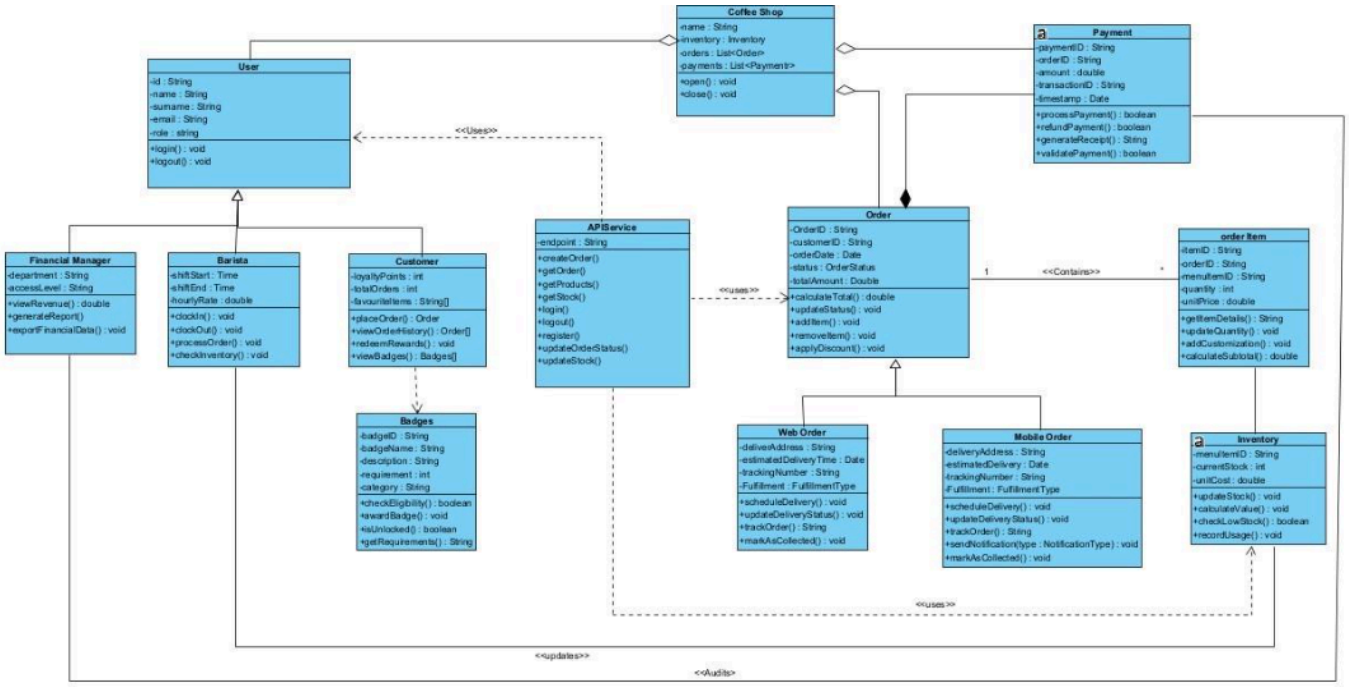
Manager

- View inventory levels for restocking
- View daily sales reports to analyze performance

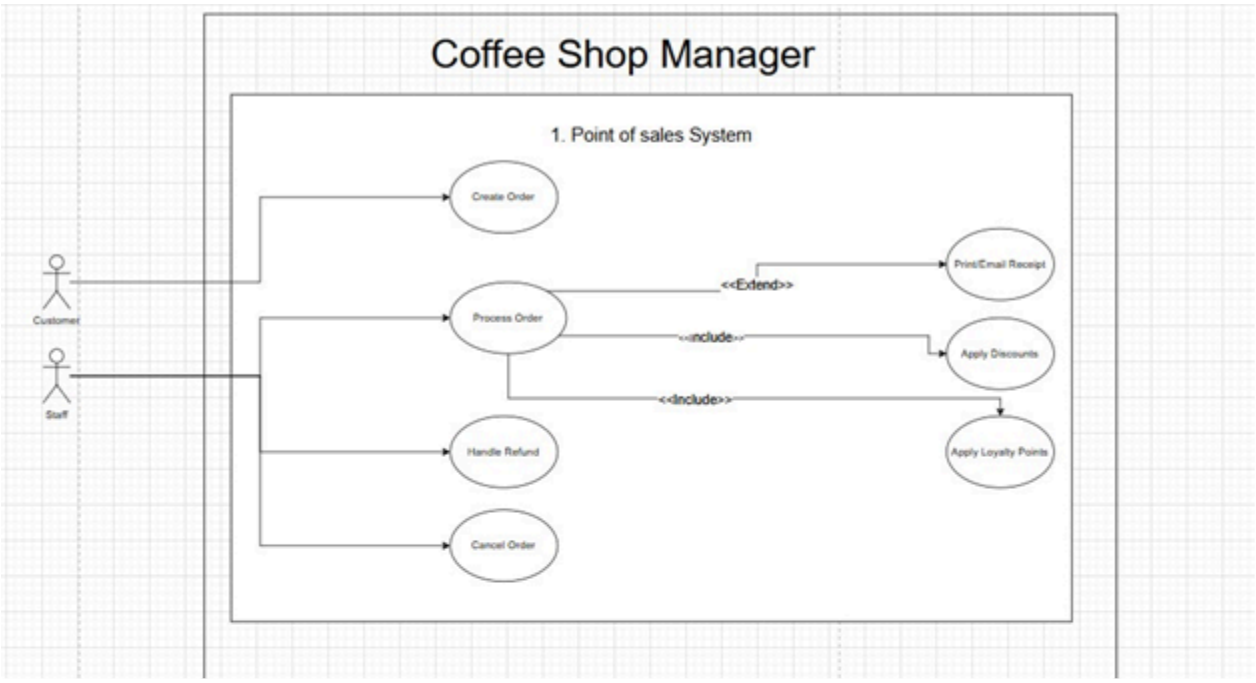
Financial Manager

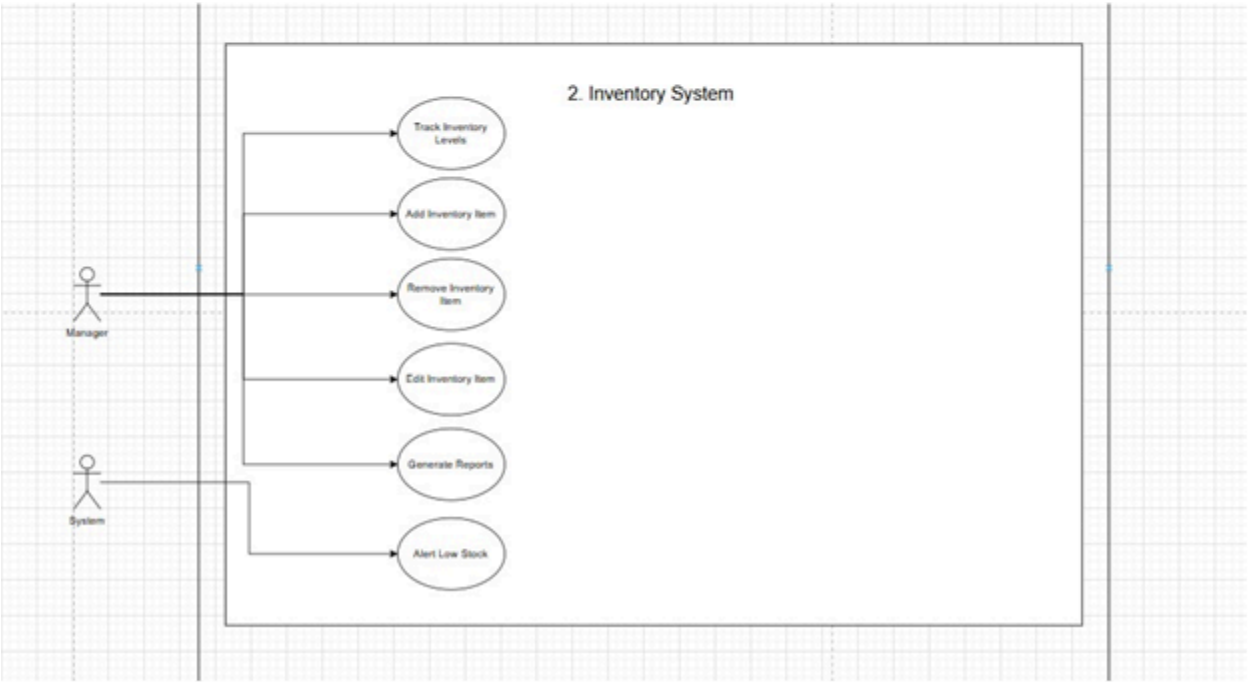
- View daily sales reports to track revenue
- Track expenses to maximize profit
- Track labor costs per employee

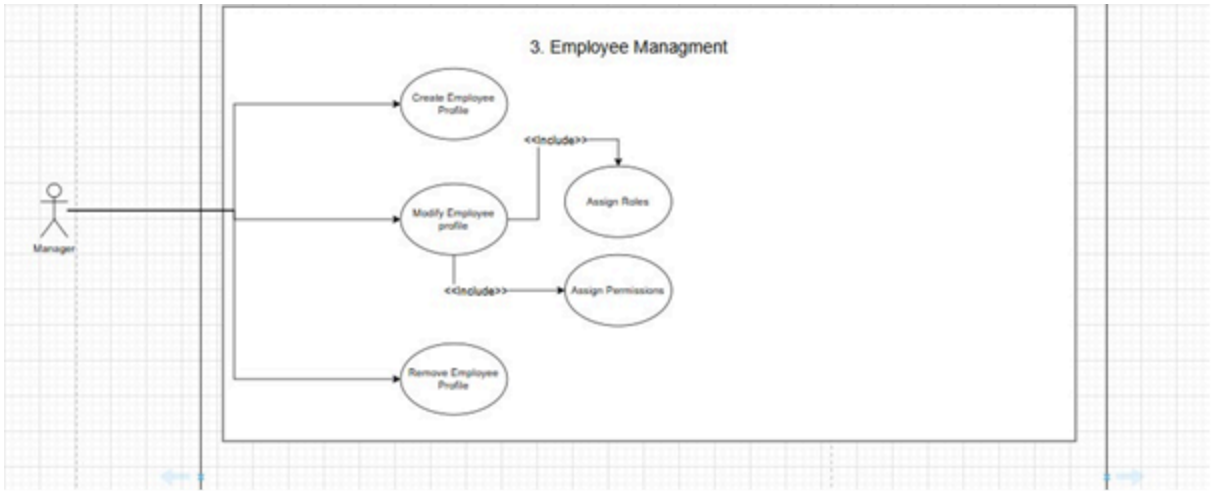
UML Diagram

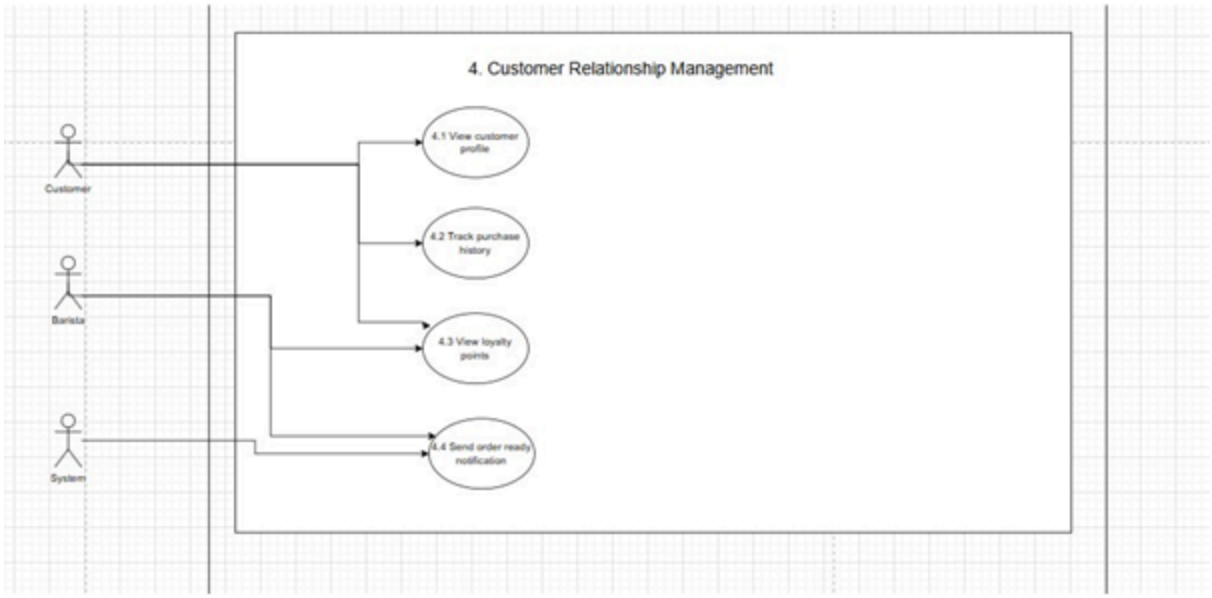


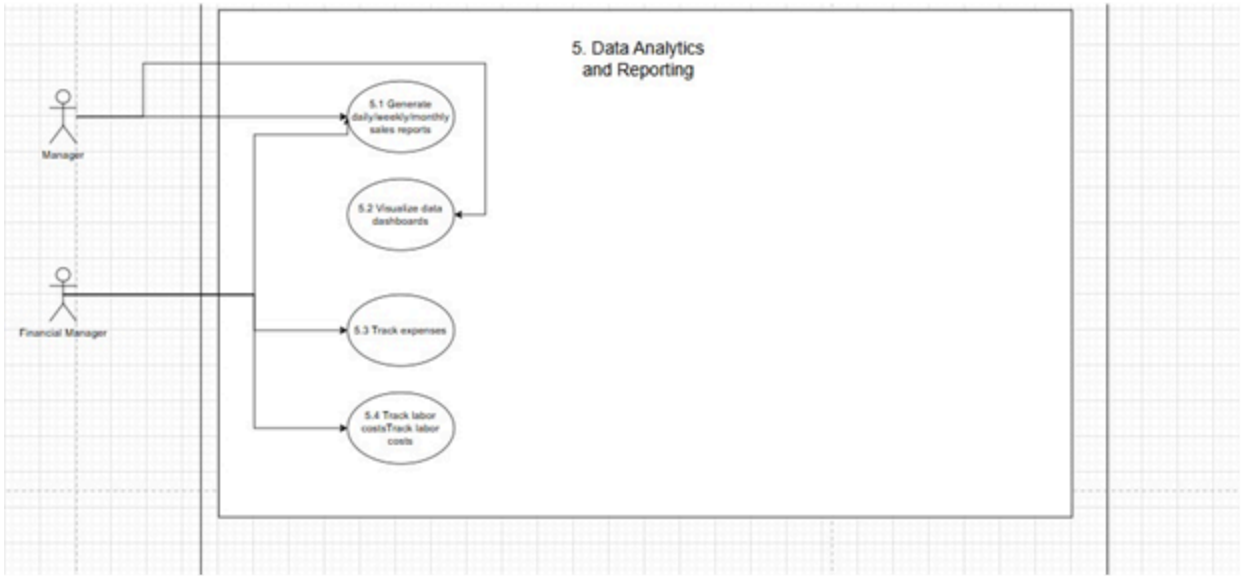
Use Case Diagram

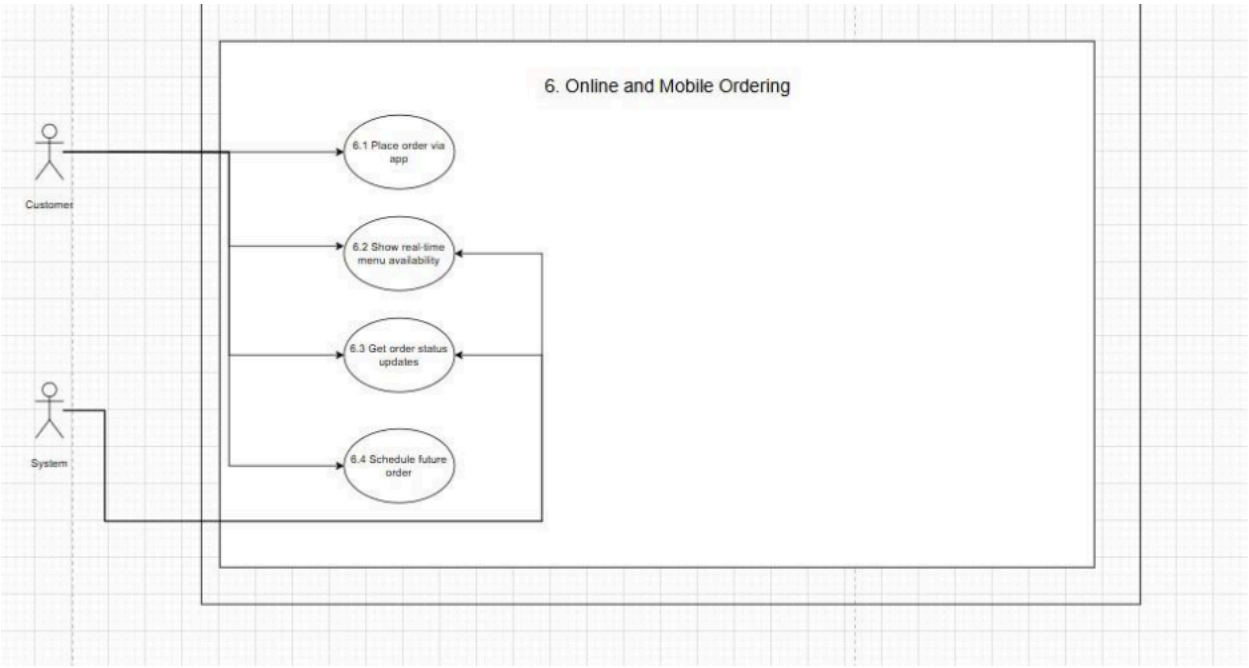





3. Employee Managment
Create Employee Profile
<<Include>>
Assign Roles
Manager
Modify Employee profile
<<Include>>
Assign Permissions
Remove Employee Profile


4. Customer Relationship Management
Customer
4.1 View customer profile
4.2 Track purchase history
Barista
4.3 View loyalty points
System
4.4 Send order ready notification


5. Data Analytics and Reporting
Manager
5.1 Generate daily/weekly/monthly sales reports
5.2 Visualize data dashboards
Financial Manager
5.3 Track expenses
5.4 Track labor costsTrack labor costs



Functional Requirements

Point of Sale (POS)

- Create and process orders
- Apply discounts and loyalty points
- Print or email receipts
- Handle refunds and cancellations

Inventory Management

- Track inventory in real time
- Alert managers on low stock
- Add/edit/remove inventory items
- Generate usage and wastage reports
- Support manual and bulk stock updates

Employee Management

- Create, modify, remove employee profiles
- Assign roles and permissions

Customer Relationship Management

- Store profiles and preferences
- Track purchase history
- Display loyalty points
- Notify customers when orders are ready

Data Analytics & Reporting

- Daily, weekly, monthly sales reports
- Visualized dashboards

Online & Mobile Ordering

- Place orders via web or mobile
- Show real-time menu availability
- Real-time order status updates
- Schedule orders

Quality Requirements

- **Availability:** 99.5% uptime
- **Performance:** 95th percentile API latency < 300ms
- **Throughput:** ≥200 orders per minute
- **Security:** JWT auth, role-based access
- **Maintainability:** Microservices, CI/CD, builds < 8 min
- **Testability:** ≥90% unit test coverage on critical modules
- **Usability:** Order completion <6 taps (mobile)
- **Logging:** Logins, failed logins, and orders with timestamps

---

Architectural Requirements

- **3-tier architecture:**
    - Presentation: Next.js (Web), React Native (Mobile)
    - Business Logic: Node.js/Express APIs
    - Data: PostgreSQL through Supabase

- **Patterns:** Singleton (DB), Factory (object creation), Observer (notifications), Strategy (discounts/loyalty), MVC, Client-Server
- **Extended patterns for Demo 3:** Microservices

Constraints

- Must use PostgreSQL
- Must use React Native for mobile
- Must run on Node.js, deployable via Docker
- Only open-source libraries
- Web & mobile share backend API

Technology Requirements

Component	Technology
Web Frontend	Next.js (React, TypeScript)
Mobile App	React Native
Backend API	Node.js + Express
Database	PostgreSQL + Drizzle ORM

Authentication	JWT + Role-based access
Styling	Tailwind CSS
DevOps / CI/CD	GitHub Actions, Docker
Testing	Jest, React Testing Library

Deployment Model

- **Environment:** Cloud-hosted containers
- **Topology:** Multi-tier, containerized microservices (Orders, Inventory, Loyalty, Analytics)
- **Tools:** Docker, GitHub Actions CI/CD

Service Contracts (API Spec)

- API endpoints for Auth, Orders, Inventory, Loyalty, Analytics
- JSON request/response formats
- REST/HTTPS protocols
- Error codes: 400, 401, 404, 500, etc.
- Versioning strategy

Coding Standards

- Languages: TypeScript, Next.js, React Native, Node.js
- Style: ESLint, Prettier, naming conventions
- Folder/repo structure
- Commit messages: Conventional Commits
- Branching & PR review process
- Testing standards: Jest, React Testing Library

We expand on this in more detail in our Coding Standard document

---

Technical Installation Manual

- Overview of system components
- Prerequisites: Node.js, Docker, Git, etc.
- Installation steps: Clone repo, install dependencies, environment variables
- Running services: Docker Compose, npm start
- Mobile setup: Expo, emulator/phone
- Testing instructions with screenshots

Testing

- Unit testing: order totals, stock updates
- Integration testing: end-to-end service flows
- Frontend UI tests: React Testing Library
- Automated execution: GitHub Actions workflows
- Coverage: 90% on critical modules

CI/CD

- GitHub Actions workflows overview
- CI: linting, unit tests, integration tests
- CD: Docker build & push, deploy
- Rules: PR must pass tests before merge
- Example YAML snippets

Security & Roles

- Authentication: JWT
- Role-based access control: Customer, Barista, Manager, Financial Manager
- Security: HTTPS, CORS, rate limiting, input validation

- Logging: sensitive events (logins, failed attempts)

Wow Factors

- Push notifications
- Gamification
- Low-stock alerts

Versioning

- History of SRS updates:
    - v1: Initial
    - v2: Design patterns & constraints
    - v3: Full documentation for Demo 3
