

EverBloom

An Innovative Online Flower Store Management System

Anika de Beer 231312

Open Window, School of Fundamentals

DV200

Tsungai Katsuro

24 July 2025

Table of Contents

1. Client Conceptualisation & Problem Statement.....	3
1.1 Overview of the Project and Business Domain.....	3
1.2 The Core Problem or Need the App Will Solve.....	3
1.3 Necessity of the Software Solution.....	3
1.4 Constraints or Limitations.....	4
1.5 Name and Basic Logo.....	5
2. System Architecture.....	6
2.1 High-Level System Design Diagram.....	6
Interaction Flow.....	6
2.2 Technologies & Frameworks.....	7
2.3 Justification for Tech Stack Selection.....	7
React.js (Frontend).....	7
3. Feature Requirements & Scope.....	8
3.1 Define the Scope of Your System.....	8
3.2 SMART Objectives.....	8
3.3 List of Major Features.....	8
3.4 Feature Prioritization.....	9
3.5 Define User Roles.....	9
3.6 System Flow Diagrams or User Stories.....	10
4. Data Planning.....	10
4.1 Entity-Relationship Diagram (ERD).....	10
4.2 Explanation of Key Tables.....	11
4.3 Data Types and Constraints.....	12
5. Wireframes & UI/UX Considerations.....	13
5.1 Moodboard.....	13
5.2 Basic Wireframes of Key Screens.....	13
5.3 Justification of Frontend Frameworks.....	16
5.4 Accessibility Concerns.....	16
6. Project Timeline & Workflow.....	17
6.1 High-Level Gantt Chart.....	17
6.2 Breakdown of Milestones and Deliverables.....	17
6.3 Estimated Timelines.....	18
6.4 Project Management Methodology.....	18
6.5 Feature Responsibility.....	19
7. Risks, Challenges & Conclusion.....	19
7.1 Potential Risks.....	19
7.2 Mitigation Strategies.....	19
7.3 Final Thoughts/Conclusion.....	19

1. Client Conceptualisation & Problem Statement

1.1 Overview of the Project and Business Domain

EverBloom is a vertically integrated online flower shop with a unique promise: “From our fields to your door.” We grow and sell our own flowers directly to customers, offering a seamless online shopping experience for fresh bouquets and arrangements. Behind the scenes, EverBloom functions as a powerful internal tool, managing inventory and logistics by tracking stem availability and locations across all stores. The direct-from-the-farm approach, combined with advanced business management in a single integrated system, sets EverBloom apart from competitors like Adene’s Farm Flowers, Femme Petale & Petal and Post.

1.2 The Core Problem or Need the App Will Solve

Customers buying flowers traditionally encounter challenges such as restricted choices, uncertainty about freshness or quality, unreliable availability, and a time-consuming shopping process. Meanwhile, local florists struggle to expand their reach and keep inventory accurate and up-to-date in an ever-changing marketplace. This leads to missed opportunities, dissatisfied customers, and inefficiencies on both sides of the transaction.

1.3 Necessity of the Software Solution

EverBloom addresses the challenges of real-time inventory management and limited variety in the traditional flower industry. By tracking flowers from harvest to delivery, EverBloom ensures customers have easy access to a wide selection of high-quality blooms for any event, far beyond what’s available in stores. Florists can also confidently place large pre-orders, avoiding the uncertainty and limited stock typical of visiting the Johannesburg Flower Market. This streamlined, user-friendly platform benefits both customers and florists, guaranteeing freshness, variety, and convenience.

1.4 Constraints or Limitations

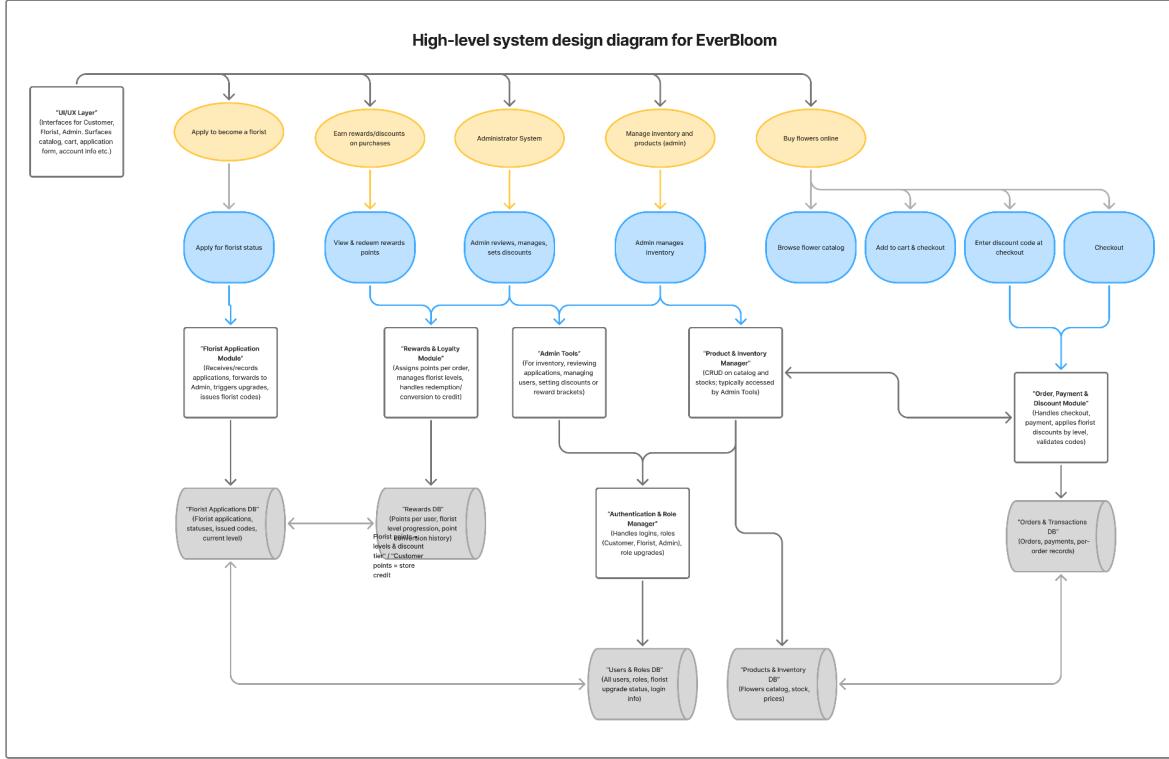
Key challenges include managing constantly fluctuating inventory as flowers are harvested, stored, and distributed; ensuring reliable and timely deliveries, particularly for same-day or large event orders; securely integrating with various payment gateways; and maintaining EverBloom's distinctiveness in a crowded and competitive market. Additionally, coordinating logistics between farms, delivery partners, and customers may present complexities that require robust solutions.

1.5 Name and Basic Logo



2. System Architecture

2.1 High-Level System Design Diagram



The system adopts a three-tier architecture:

- **Frontend** (Client): Runs in the user's browser, handling all user interactions.
- **Backend** (Server/API): Processes business logic, handles authentication, and mediates all data transactions.
- **Database** (Data Layer): Stores persistent data such as users, products, orders, applications, and rewards.

Interaction Flow

- The **frontend** communicates with the **backend** via RESTful APIs (HTTP), sending user requests (login, browsing, ordering, etc.).
- The **backend** interacts with the **database** (MySQL) to read/write data in response to API requests and enforces business logic and security.

- **Frontend** never accesses the **database** directly; all data flows through the backend for security and consistency.

2.2 Technologies & Frameworks

Layer	Technology	Role/ Responsibility
Frontend	React.js	SPA (Single Page Application) UI, AJAX API calls
Backend	Node.js	JavaScript runtime for server-side logic
Backend	Express.js	Web server, RESTful API routing, middleware handling
Database	MySQL	Relational DBMS, structured data with relationships

2.3 Justification for Tech Stack Selection

React.js (Frontend)

- Modern, component-based UI for interactive single-page apps.
- Fast, responsive dashboards with virtual DOM.
- Large ecosystem for easy feature integration and future scalability.

Node.js + Express.js (Backend)

- Non-blocking, event-driven for handling many API requests.
- Shared JavaScript codebase improves productivity.
- Express streamlines API routing; rich NPM ecosystem for rapid feature adoption.

MySQL (Database)

- Ideal for structured, relational business data.
- Stable, well-supported, easy to deploy.
- Ensures data integrity and supports future analytics.

3. Feature Requirements & Scope

3.1 Define the Scope of Your System

Included:

- User signup/login
- Flower catalog browsing
- Cart, checkout & payment
- Florist application & level upgrade process
- Discount & rewards system
- Admin dashboard for user/product management

Excluded:

- Delivery/fulfilment tracking
- Third-party marketplace integration
- Multi-language/localization
- Mobile app (web-only in MVP)

3.2 SMART Objectives

- Register and onboard users within 2 minutes (Specific, Measurable)
- Process orders with >99% accuracy (Achievable, Measurable)
- Approve or reject florist applications within 24h (Time-bound, Specific)
- Store and retrieve catalog data in <300ms (Relevant, Measurable, Achievable)
- Launch MVP within 6 weeks (Time-bound, Achievable)

3.3 List of Major Features

- Authentication & role-based access
- Flower/product catalog management
- Shopping cart & checkout with payment
- Florist application & admin review workflow
- Rewards and discount management

- Admin dashboard for reporting & inventory and managing florist status
- Order history for users

3.4 Feature Prioritization

Feature	MVP	Nice to Have	Future Consideration
User authentication/roles	✓		
Catalog browsing/search	✓		
Cart & order placement	✓		
Florist application workflow	✓		
Rewards/discount system	✓		
Admin dashboard (basic)	✓		
Order history	✓		
Advanced reporting		✓	
Discount analytics		✓	
Push/email notifications		✓	
Multi-language support			✓
App/mobile version			✓
Third-party integrations			✓

3.5 Define User Roles

- **Admin:** Full access, manage users/products, approve florists, configure rewards/discounts, view reports
- **Florist:** Special discounts, can apply for different levels, earn and redeem rewards, place orders
- **Customer:** Browse, purchase, earn and redeem rewards
- **(Optional) Read-Only User:** Can browse but not purchase (future)

3.6 System Flow Diagrams or User Stories

User Story (Florist Application):

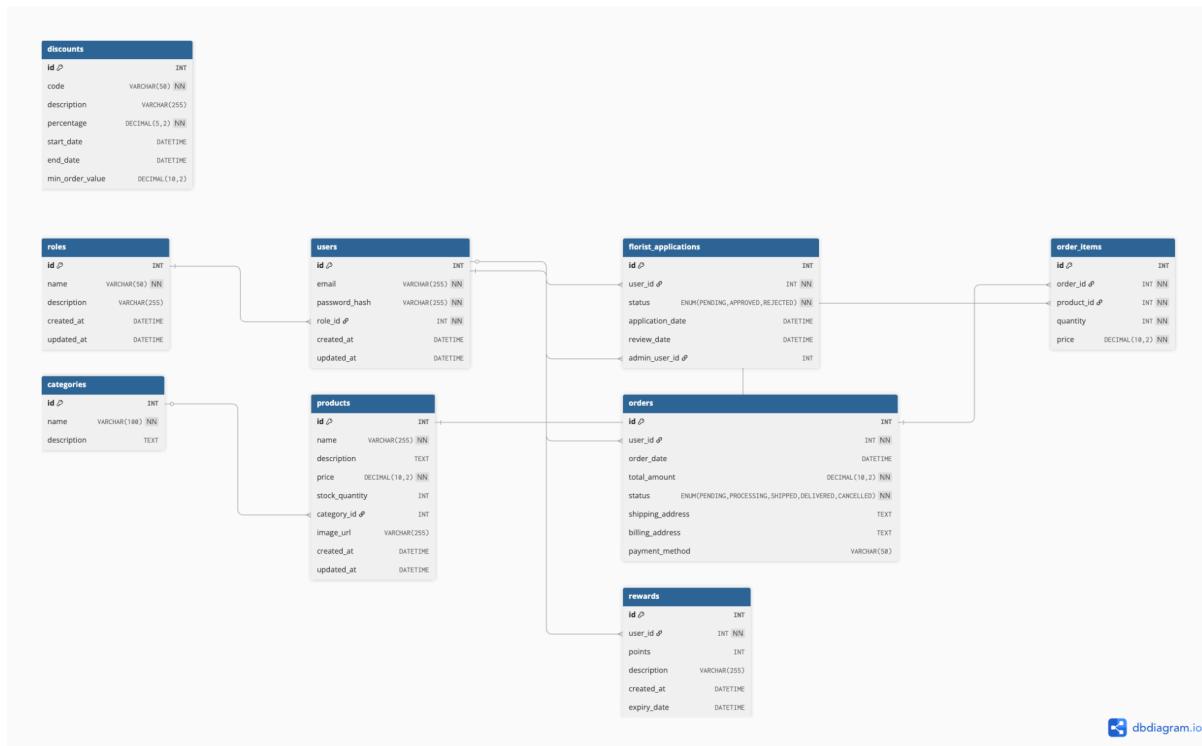
- Customer applies to become a florist.
- Admin reviews and approves/rejects application.
- On approval, user is upgraded and receives discount privileges.

User Story (Order Process):

- User browses flowers, adds items to cart.
- Proceeds to checkout, applies discount/reward.
- Completes payment, order is logged.

4. Data Planning

4.1 Entity-Relationship Diagram (ERD)



4.2 Explanation of Key Tables

Item	Purpose	Key Relationships
users	Stores all user accounts	Each user has one role (role_id → roles.id) Referenced by orders, rewards, florist_apps
roles	Defines user types (admin, florist, etc.)	One-to-many to users (role assigned to multiple users)
florist_applications	Tracks users applying as florists	Each application belongs to a user (user_id → users.id) Optionally reviewed by admin user
products	Catalog of items for sale	Belongs to a category (category_id → categories.id) Linked to order_items
categories	Organizes products by group	One-to-many to products
orders	Tracks user purchases	Each order placed by user (user_id → users.id) Has multiple order_items
order_items	Details of products in each order	Belongs to order (order_id → orders.id) References product (product_id → products.id)
rewards	Tracks user reward points	Each reward belongs to a user (user_id → users.id)
discounts	Promo codes & discount info	Not directly linked, but applied to orders

4.3 Data Types and Constraints

Field Name	Table	Data Type	Constraints
id	ALL (as PK)	INT	Primary Key, Auto-increment
email	users	VARCHAR(255)	Unique, Not Null
password_hash	users	VARCHAR(255)	Not Null
role_id	users	INT	Not Null, Foreign Key → roles(id)
name	roles, categories	VARCHAR(50/100)	Unique, Not Null
user_id	florist_applications, orders, rewards	INT	Foreign Key → users(id), Not Null

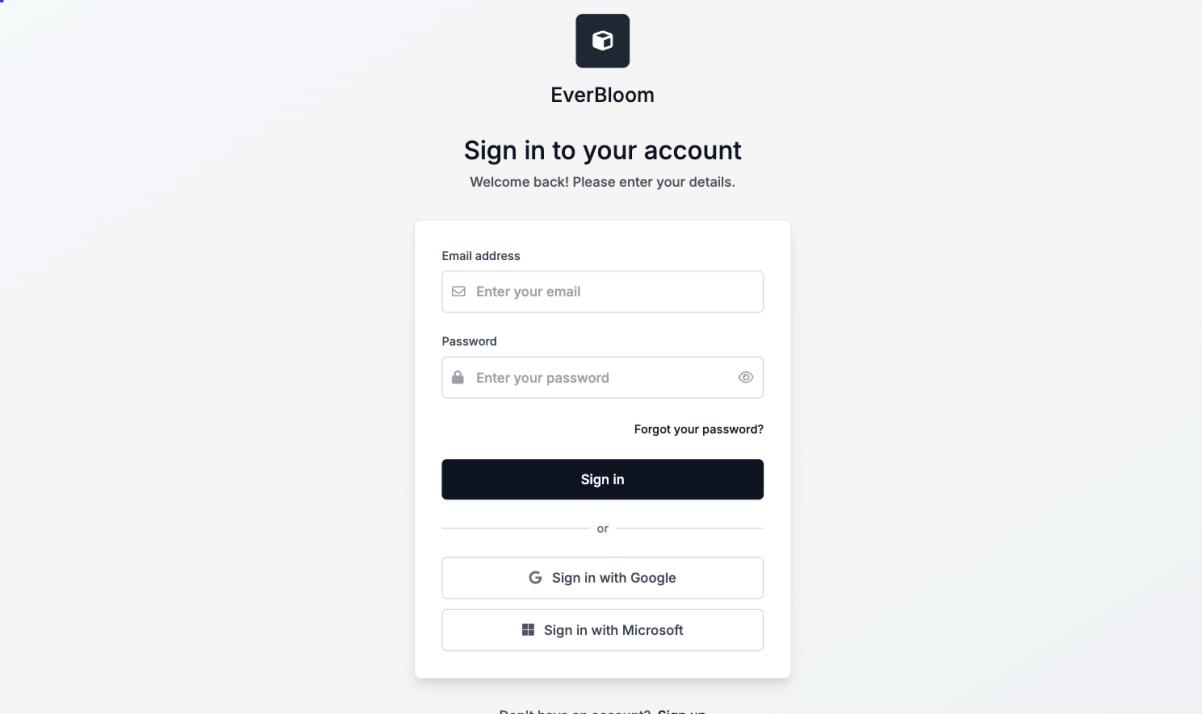
admin_user_id	florist_applications	INT	Foreign Key → users(id), Optional
status	florist_applications, orders	ENUM	Not Null (predefined allowed values)
price	products, order_items	DECIMAL(10,2)	Not Null
stock_quantity	products	INT	Default 0
category_id	products	INT	Foreign Key → categories(id), Optional
total_amount	orders	DECIMAL(10,2)	Not Null
order_date, created_at	Various	DATETIME	Default CURRENT_TIMESTAMP
code	discounts	VARCHAR(50)	Unique, Not Null
percentage	discounts	DECIMAL(5,2)	Not Null
points	rewards	INT	Default 0

5. Wireframes & UI/UX Considerations

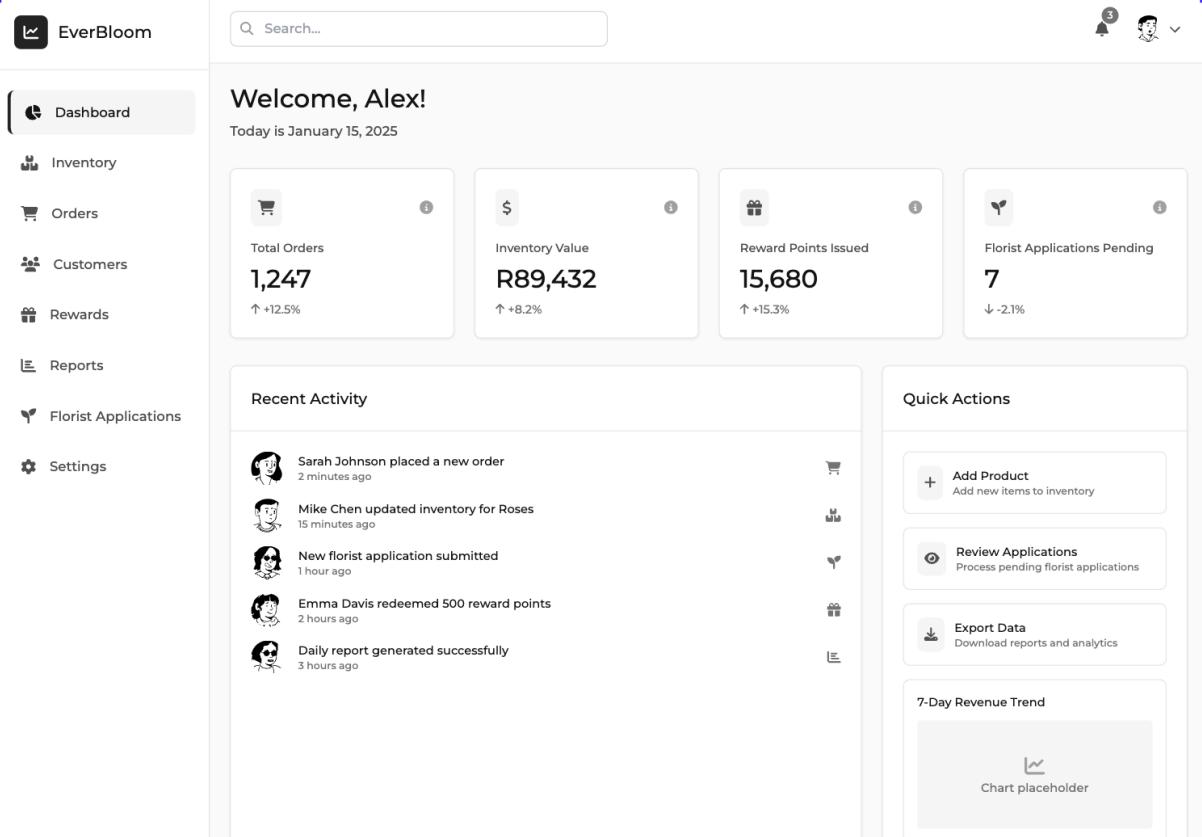
5.1 Moodboard



5.2 Basic Wireframes of Key Screens



The sign-in screen features a logo at the top left and a title "Sign in to your account" in bold. Below it is a sub-instruction "Welcome back! Please enter your details." A large input field for the email address is followed by a password field. Below these are links for "Forgot your password?" and "Sign in". A horizontal line with the word "or" separates this from two social sign-in options: "Sign in with Google" and "Sign in with Microsoft". At the bottom, there's a link for users who don't have an account: "Don't have an account? Sign up".



The dashboard has a header with the EverBloom logo, a search bar, and user notifications. On the left is a sidebar with navigation links: Dashboard (selected), Inventory, Orders, Customers, Rewards, Reports, Florist Applications, and Settings. The main area starts with a welcome message "Welcome, Alex!" and the date "Today is January 15, 2025". It includes four summary cards: "Total Orders" (1,247, +12.5%), "Inventory Value" (R89,432, +8.2%), "Reward Points Issued" (15,680, +15.3%), and "Florist Applications Pending" (7, -2.1%). Below these are sections for "Recent Activity" (listing events like new orders and application submissions) and "Quick Actions" (links for adding products, reviewing applications, and exporting data). A placeholder chart for "7-Day Revenue Trend" is also present.

Inventory Management

Inventory > All Products

[Export](#) [+ Add New Product](#)

⚠ 5 products are below minimum stock level. [View low stock items](#)

Search products... to

Products									Column Settings
	IMAGE	Name \$	SKU \$	Category	Stock \$	Status	Price \$	Last Updated	ACTIONS
<input type="checkbox"/>		Boom Boom White 60cm Stem, Fresh Cut	WHP-001	Dahlias	45	Active	R120	Jan 15, 2025	
<input type="checkbox"/>		Salmon Pink 60cm Stem, Fresh Cut	SWX-002	Dahlias	120	Active	R120	Jan 12, 2025	
<input type="checkbox"/>		Red Roses 60cm Stem, Fresh Cut	ELS-003	Roses	8	Inactive	R100	Jan 10, 2025	

Show per page Showing 1-10 of 247 results ...

Rewards & Promotions

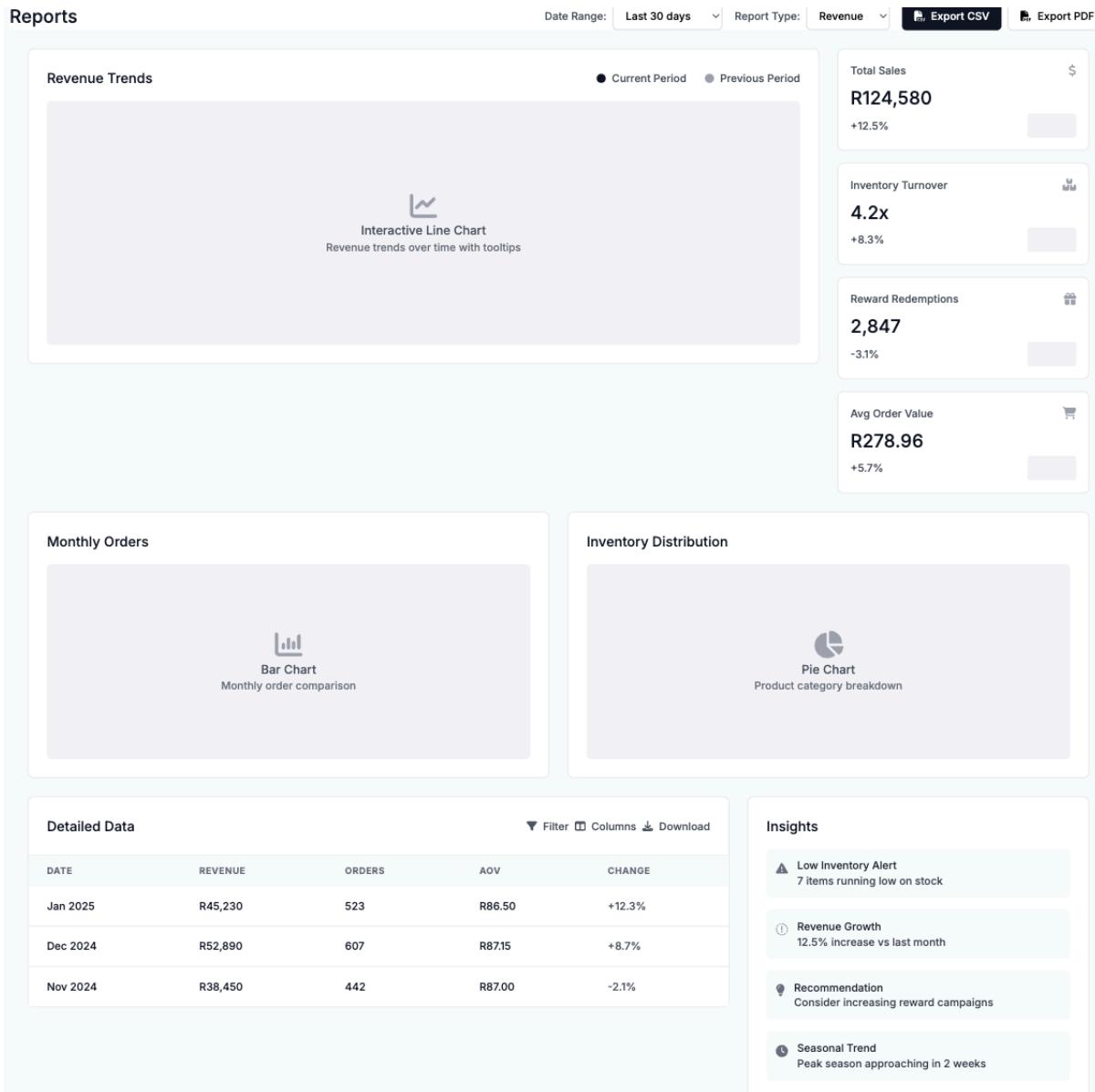
[+ Create New](#)

[Rewards Points](#) [Discount Codes](#)

User Points Balance				
USER	TOTAL POINTS	EARNED THIS MONTH	SPENT THIS MONTH	ACTIONS
Sarah Johnson sarah@example.com	2,450	+320	-150	
Michael Chen michael@example.com	1,890	+280	-100	
Emma Wilson emma@example.com	3,120	+450	-200	

Recent Redemptions

- Alex Rodriguez
Redeemed 500 points for \$5 discount
2 hours ago
- Lisa Park
Redeemed 1000 points for free shipping
5 hours ago
- David Kim
Redeemed 750 points for product discount
1 day ago



Application Management 247 Total Applications Export Settings

Status:	All Statuses	Search:	Search by name or email	Date Range:	yyyy/mm/dd	to	yyyy/mm/dd	Apply Filters	Clear All
<input type="checkbox"/>	Sarah Johnson sarah.johnson@email.com Submitted: Jan 15, 2025	Pending	2 past applications						
<input type="checkbox"/>	Michael Chen michael.chen@email.com Submitted: Jan 14, 2025	Approved	1 past application						
<input type="checkbox"/>	Emily Rodriguez emily.rodriguez@email.com Submitted: Jan 13, 2025	Rejected	0 past applications						

Showing 1 to 3 of 247 applications

< 1 2 3 ... 83 >

5.3 Justification of Frontend Frameworks

I chose React as our frontend framework because of its proven component architecture, robust community support, and suitability for scalable, responsive dashboards. Styling uses Tailwind CSS for its utility-first approach, rapid prototyping, and consistent design across screens.

- Usability considerations: Large click areas, logical grouping of controls, and clear feedback on user actions are prioritized.
 - Responsive design ensures full usability on desktop and tablets.
 - Consistent color palette and spacing support focus and minimize cognitive load.

5.4 Accessibility Concerns

- Colour Contrast: Colors from the moodboard will be validated for at least AA contrast.
 - Keyboard Navigation: All interactive elements (buttons, forms, tables) are tab-accessible.
 - ARIA Labels: Used for form fields, charts, and dynamic content to ensure screen reader compatibility.
 - Font Sizes: Minimum 16px for comfortable reading, with scalable options.
 - Focus Indicators: Clearly styled focus outlines for accessibility clarity.

6. Project Timeline & Workflow

6.1 High-Level Gantt Chart

6.2 Breakdown of Milestones and Deliverables

Milestone	Deliverable	Deadline
Project Proposal	Project Plan PDF (ideation, planning)	Week 3
Backend Ready	DB schema, Express.js API live	Week 5
First Progress Check	Demo backend endpoints	Week 5
Frontend Core Screen Ready	Main UI screens (React)	Week 6
Integration Complete	FE + BE functional, basic CRUD	Week 7
SQL Exam (individual task)	Proctored SQL knowledge check	Week 7
Progress/Milestone Check	Codebase review, integration demo	Week 9
Testing & Debugging	Test report, bugfix log	Week 9-11
Deployment Presentation	Presentation slides / Deployment demo	Week 11-14
Deployment to GitHub/Git Live	Hosted live app, code repo updated	Week 14
Final Demo & Delivery	Project demo, code + doc submission	Week 16

6.3 Estimated Timelines

- **Backend (Database + Express.js APIs):**

Weeks 4–6 (with post-integration fixes up to Week 9)

- **Frontend (React App, UI):**

Weeks 5–7 (initial build), with ongoing fixes to Week 10

- **Integration (Connect FE & BE):**

Weeks 6–8

- **Testing (Manual & Automated):**

Weeks 8–11 (starts after first integration)

Deployment & Presentation:

Weeks 11–16 (includes iterations based on feedback)

6.4 Project Management Methodology

Agile-inspired Workflow:

The project will use an iterative, milestone-driven approach modeled after Agile:

- Weekly sprints with clear deliverables
- Frequent progress checks and functional demos
- Rapid feedback and adaptation after reviews

Collaboration & Tracking Tools:

- GitHub: Version control, code reviews, and milestones
- Miro: Visual planning, user flows, wireframes, roadmap visualization

Task & Deliverable Tracking:

- Use GitHub Issues & Projects (Kanban board) to create, assign, and track tasks
- Milestone labeling for deadlines

6.5 Feature Responsibility

Project Owner:

- Responsible for all phases - ideation, backend, frontend, integration, testing, deployment, and documentation
- Managing the GitHub repo and Miro boards
- Presenting project and handling Q&A

7. Risks, Challenges & Conclusion

7.1 Potential Risks

A) Technical Risks

Risk	Description	Mitigation Strategy
Scope Creep	Unplanned features or changes may extend beyond the timeline	Lock scope early, get sign-off, use a change request process
Integration Issues	Trouble connecting backend (APIs/Database) with frontend	Early integration testing; use Postman for API validation
Data Loss or Corruption	Unexpected data loss due to coding or deployment errors	Versioned backups, transactional code, regular DB export
Deployment & Hosting Failures	Errors during deployment, misconfigured servers, or 3rd-party outages	Follow deployment checklist, use stable platforms, backup DNS/config
Security Vulnerabilities	Weak auth, exposure of sensitive info, or database leaks	Follow secure coding guidelines, validate inputs, use .env files
Lack of Testing	Insufficient unit/QA testing may result in undetected bugs	Schedule test phases, automate critical tests, peer/manual reviews

B) Non-Technical Risks

Risk	Description	Mitigation Strategy
Time Constraints	Overlapping coursework or underestimating workload	Stick to timeline, weekly progress checks, prioritize core features
Resource Limitations	Limited access to certain tools, platforms, or hardware	Choose free/dev-tier tools, develop locally, use open-source options
Requirements Miscommunication	Misalignment between expected and delivered features	Confirm requirements, regular reviews/feedback, document changes
Motivation/Fatigue	Burnout from intense solo workload	Break tasks into small goals, celebrate milestones, schedule breaks

7.2 Final Thoughts/Conclusion

This proposed system provides a robust, user-friendly solution for inventory and data management. It uses reliable, modern technologies and a clear workflow to ensure deliverability within your timeline. With strong planning, collaboration via GitHub and Miro, and built-in flexibility for future improvements, it's well-suited to meet your project's goals and user needs.