

ADAMA SCIENCE AND

TECHNOLOGY UNIVERSITY

**School of Electrical Engineering and computing**

**Software Engineering Department**

Mobile application design and development

Software Requirement Specification Document

**Group Members**   **ID no**

1. Sumeya Awel Ugr/26377/14
2. Oliyad Zelalem Ugr/31100/15
3. Sosina Seife Ugr/31500/15
4. Hana Getu Ugr/30624/15
5. Dagmawit Andargachew Ugr/30373/15
6. Mekdes Urgi Ugr/30874/15

Section 2

Instructor Name:  Yared Tekalegn

Submission Date: May-2-2025

**Quick mart**

**1, introduction** 1.1purpose of documentation  
 1.1.1 user guides  
 1.1.2technical specification  
 1.1.3release notes  
 1.2key future of quick mart  
**2, project overview**  
 2.1 problem domain  
 2.2 current market issues and problems  
 2.3 potential of quick mart to solve these problems  
 2.4 feature scope  
 2.5 target audience with characteristics and notice  
**3, user requirements**  
 3.1 functional requirement  
 3.2 non-functional requirement  
**4, design concept** 4.1 UI design  
 4.2 UX flow  
 4.3 responsiveness and accessibility  
**5, development approach**  
 5.1 agile methodology  
 5.1.1 Justification for choosing agile  
 5.2 collaborative design- development loop  
 5.3 component- based architecture  
 5.4 code management  
 5.5 continuous testing  
 5.6 justification for choosing agile  
**6, technological stack**  
 6.1 front- end  
 6.2 back-end  
 6.3 tools and platforms  
 6.4dependencies  
**7, implementation details**  
 7.1 key features and functionalities  
 7.2 functionality of the app  
 7.3 essential features of the app  
 7.4 screen developed  
 7.5 state management  
 7.6 UI libraries used  
 7.7 navigation  
 7.8 state management  
 7.9 swipe and gesture handling  
 7.10 authentication  
 7.11 checkout logic  
 7.12 UI styling  
**8, testing and quality assurance**  
 8.1 reflection on testing effectiveness  
**9, future enhancements** 9.1 some of future integration  
 9.2 alignment with project objective and user needs  
 9.3 technical restrictions  
 9.4 operational restriction  
 9.5 strategic restriction

1. **Introduction**

Quick Mart is a mobile commerce app fully dedicated to shoes. With a minimalist, easy-to-use interface designed using Figma and developed using React Native, the app provides users with an effortless experience to browse, favorite, and purchase shoes. End-to-end, from discovery to checkout, Quick Mart offers convenience, speed, and modern looks in sync with mobile shoppers.

**1.1 Purpose of Documentation**

This is a comprehensive guide for developers, stakeholders, and users to understand the structure, operation, and development process of the Quick Mart mobile application.

* + 1. **User Guides**

Provides end-users with instructions on how to navigate and utilize the app features efficiently.

* + 1. **Technical Specifications**

Details the architecture, technology stack, and development practices used in building the application.

* + 1. **Release Notes**

Captures updates, bug fixes, and enhancements in different versions of the application.

* 1. **Key Features of Quick Mart**

Discover shoes with swipe gestures

Save favorites for later

Real-time cart and total amount view

User authentication and profile management

1. **Project Overview**

The Quick Mart application aims to provide a minimalist and user-friendly platform for purchasing shoes online. The app features five core screens:

**Loading Screen:** Displays the app logo during startup.

**Discover Page:** Acting as the main page, the user can browse through available shoes, swipe to the right to add items into the cart, swipe to the left to take away, and search by the name of shoes. Each product includes detailed information and the option to favorite it. 

**Favorites Page:** A list of shoes saved by the user from the Discover page for later viewing. 

**Cart Page:** Displays selected shoes with pricing and the total amount. From here, users can proceed to checkout. 

**Profile Page:** Supports user authentication and displays account details. Logged-in users can access settings, order history, help & support, and log out. New users can create an account via email and password.

Quick Mart offers a familiar e-commerce flow while keeping the experience minimal and focused, particularly for mobile-first users who want fast and easy access to popular footwear.

* 1. **Problem Domain**

The current digital shopping experience often lacks personalization, speed, or intuitive design for quick purchases.

* 1. **Current Market Issues and Problems** 

Overloaded interfaces on many e-commerce apps 

Slow performance or poor navigation on mobile devices 

Lack of targeted apps focused on footwear

* 1. **Potential of Quick Mart to Solve These Problems**

Quick Mart addresses these issues through a minimalistic interface, intuitive gestures, and a product-specific approach.

* 1. **Feature Scope**

Includes swipe-based product discovery, favorites, cart management, and account creation, with future plans for payment and backend integration. 2.5 Target Audience with Characteristics and Needs

Young adults aged 18–35

Fashion-conscious mobile shoppers

Require fast and easy navigation

Prioritize app responsiveness and visual design

**3. User Requirements**

**3.1 Functional Requirements**

The following functional requirements define the key behaviors and features of the Quick Mart mobile application:

**User Registration and Login:**Users must be able to create an account using email and password and log in securely.

**Product Browsing:**Users can browse a catalog of shoes displayed on the Discover page.

**Swipe Actions on Discover Page:**

 Swiping right adds a shoe to the cart.

 Swiping left dismisses the product.

**Product Search:**Users can search for specific shoes by name.

**Add to Favorites:**Users can favorite shoes from the Discover page for later viewing.

**View Favorites List:**Users can see all favorited shoes on the Favorites page.

**Add to Cart and View Cart:**Shoes added to the cart are displayed on the Cart page with individual and total prices.

**Checkout Process:**Users can proceed to purchase shoes from the cart.If they are not logged in, they are asked to sign-up or log-in in advance.

**User-Profile Management:**Logged-in users can view and edit their profile, e.g., their e-mail, profile picture, and preferences.

**Order History and Support:**Users can view past orders and access help & support options.

**Logout Functionality:**Users can log out from the profile page.

**3.2 Non-Functional Requirements**

The following requirements establish the app's quality and performance features:

**Usability:**The UI must be minimal, clean, and user-friendly for people of all ages.

**Performance:**App should load quickly with responsive transitions between screens.

**Compatibility:**The app must function seamlessly on both Android and iOS devices.

**Security:**User credentials must be securely stored, and sessions should be properly managed.

**Scalability:**The backend architecture should allow future growth (e.g., more products, more users).

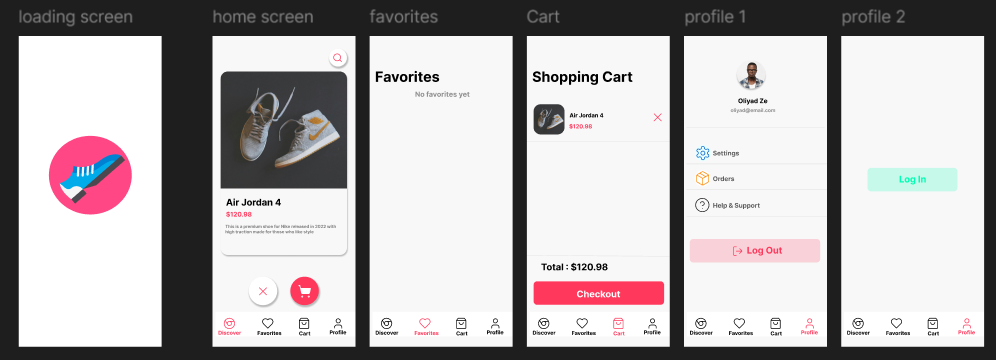
**Availability:**The app should maintain high availability with minimal downtime during updates.

**4. Design Concepts**

The Quick Mart interface is built around a clean, minimalist, and very interactive user interface. It follows modern mobile UI/UX guidelines, with emphasis on clarity, ease of navigation, and efficiency in user interaction.

**4.1 UI Design**

The visual components are designed in Figma with a light color palette and bold, readable typography. The layout relies on a bottom tab navigation bar with four main icons: Discover, Favorites, Cart, and Profile. Each screen is spaced for touch-friendly navigation and is optimized for both small and large mobile screens.



**4.2 UX Flow**

**On Launch:** Users are presented with a loading screen displaying the app logo.

**Discover Flow:** Users scroll or swipe through available shoes. Swiping left dismisses and swipes it out of the cart, and swiping right in pulls it into the cart.

**Search Flow:** A search bar allows users to find specific shoes by name.

**Favorites Flow:** Items marked as favorite are stored for future reference and accessible on a dedicated page.

**Cart and Checkout Flow:** Multiple selected items are shown with individual and total prices. Checkout prompts authentication if the user isn't logged in.

**Authentication Flow:** New users can create accounts with email and password, while returning users can log in. Once signed in, users can access their profile and settings.

**4.3 Responsiveness & Accessibility**

**Responsive Layout:** Components resize smoothly to fit various device sizes.

**Touch Optimization:** Interactions and buttons are all optimized for tap accuracy and thumb reach.

**Accessible Fonts and Contrast:** Offers readability and usability for people with visual impairments.

This section guides the foundation of Quick Mart’s interface design and user experience decisions.

**5. Development Approach**

Development of Quick Mart is on a mobile-first agile methodology, which ensures rapid prototyping, iterative feedback, and continuous improvement. This section provides an overview of the planning and execution strategy adopted while developing the app.

**5.1 Agile Methodology**

Development was structured in sprints with each iteration focused on releasing a functional chunk of the app (i.e., login screen, discover flow, cart logic). regular stand-ups and review sessions enabled feedback integration at an early point and ensured that all were on the same page.

**5.1.1 Justification for Choosing Agile**

The Agile methodology was selected for Quick Mart due to its flexibility, speed of delivery, and incremental nature, which fits well into the rapidly changing environment of mobile app development.

**Agile allowed the development team to:**

Quickly adapt to design and functionality feedback from early user interactions.

Iteratively develop features such as gesture handling, navigation, and user account flows in manageable sprints.

Promote collaboration among design and development teams, especially through tools like Figma and Git for version control.

Realize a Minimum Viable Product (MVP) quickly and plan for further enhancements based on actual user needs and market feedback.

**5.2 Collaborative Design-Development Loop**

The design and development teams worked closely, using Figma for UI prototypes and React Native for implementation. Real-time collaboration ensured that what was designed was accurately translated into code.

**5.3 Component-Based Architecture**

React Native components were organized modularly, allowing for reuse across the app (e.g., button styles, card displays, tab navigation). This approach improved maintainability and scalability.

**5.4 Code Management**

Version control was managed with Git and GitHub, using feature, fix, and release branching models. Pull requests were reviewed as code before merging, to have clean, documented code.

**5.5 Continuous Testing**

Each build was tested on multiple devices and emulators using both manual and automated test scripts to ensure stable releases during each sprint.

**5.6 Justification for Choosing Agile**

The Agile approach was selected for Quick Mart due to its flexibility, speed of delivery, and iterative nature, which works well with the mobile app development dynamic nature. Agile allowed the development team to:

Quickly adapt to design and functionality feedback from early user interactions.

Iteratively develop features such as gesture handling, navigation, and user account flows in manageable sprints.

Foster collaboration between design and development teams, especially via Figma and Git version control usage.

Launch an MVP quickly and iterate afterward based on actual user needs and market feedback.

**6. Technological Stack**

Quick Mart is built using a modern and scalable technology stack tailored for mobile-first development, rapid UI prototyping, and cross-platform deployment.

**6.1 Front-end**

**React Native:** Core framework for building cross-platform mobile applications.

**Figma:** UI/UX design and prototyping tool.

**JavaScript / TypeScript:** For writing clean, modular, and scalable application logic.

**React Navigation:** For managing in-app navigation and routing.

**6.2 Backend (Optional/Future Enhancement)**

While the current implementation may use local state or mock APIs, future enhancements could include:

**Node.js + Express.js:** Backend server for APIs and business logic.

**MongoDB / Firebase:** Database solutions for storing product details, user accounts, cart data, etc.



**Firebase Authentication:** Secure user registration and login system.

**6.3 Tools and Platforms**

**Expo:** For building, testing, and deploying React Native apps faster.

**Git & GitHub:** For version control and collaboration.

**VS Code:** Preferred code editor for development.

**Android Studio / Xcode:** For platform-specific emulators and builds.

This stack enables Quick Mart to operate efficiently on both Android and iOS, supports rapid UI updates, and is scalable for future backend integration and expansion.

**6.4 Dependencies**

**react-native-gesture-handler:** Swipe gesture implementation

**react-navigation:** Navigation across screens

**expo:** Build and deployment tool for React Native apps

1. **Implementation Details**

The implementation phase of Quick Mart was focused on translating Figma-based UI designs into fully functional, interactive screens using React Native. Each screen and feature was developed using a component-based approach, enabling reuse and maintainability.

**7.1 Key Features and Functionalities**

Discover, favorite, and cart flows

Swipe gestures for product interactions

User account creation and management

**7.2 Functionality of the App**

The app allows seamless shoe browsing, cart calculation, and checkout initiation with user authentication support.

**7.3 Essential Features of the App**

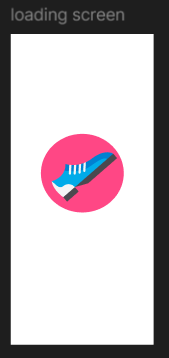
Search shoes by name

Add/remove favorites

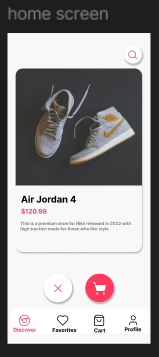
Persistent user sessions (planned)

**7.4 Screens Developed**

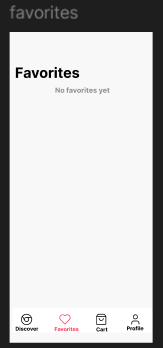
**Loading Screen:** Displays logo animation during app start-up.



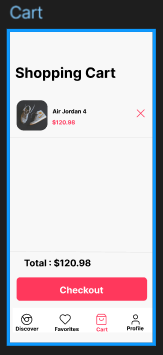
**Discover Screen:** Includes card-based UI for shoe listings, swipe gestures, search input, and favorite icon toggle.



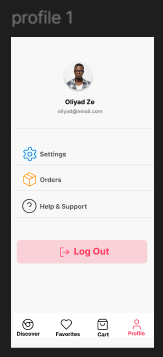
**Favorites Screen:** Displays a list of shoes marked as favorites.



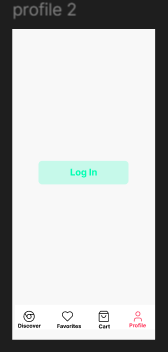
**Cart Screen:** Summarizes selected items, quantity, total price, and checkout button.



**Profile Screen:** Displays user profile, with conditional UI for logged-in vs new users.



**Authentication Screens**: Includes sign-up and sign-in with form validation and Firebase integration.



**7.5 State Management**

**React Context API:** Used to manage cart items, favorites, and user session across the app.

**7.6 UI Libraries Used**

**React Native Paper and React Native Elements:** For pre-styled UI components.

**Lottie for React Native:** For loading animations.

**7.7 Navigation**

**Bottom Tab Navigation:** Implemented using @react-navigation/bottom-tabs.

**Stack Navigation:** Used for screen transitions during authentication and checkout.

This phase ensured that all planned functionalities and interactions from the design stage were implemented with performance and usability in mind.

**7.8 State Management**

For MVP, React's built-in useState and useContext hooks were used for local state management. Future versions may integrate Redux or Context API for more complex state needs.

**7.9 Swipe and Gesture Handling**

The react-native-gesture-handler library enables smooth swiping functionality on the Discover screen, enhancing UX by allowing users to add to cart or dismiss shoes.

**7.10 Authentication**

Implemented using mock email/password logic with plans to upgrade to Firebase Authentication. Auth guards were added to restrict access to the cart and profile for unauthenticated users.

**7.11 Checkout Logic**

Basic logic was created to calculate the total price and validate user authentication before proceeding to checkout. On checkout, users are redirected to login/signup if not already signed in.

**7.12 UI Styling**

Used StyleSheet in React Native and followed the design from Figma to ensure pixel-perfect implementation. Styles are modularized for easier theme updates in the future.

**8. Testing and Quality Assurance**

Testing and QA were included in the Quick Mart development process to ensure the app functioned consistently across devices, provided a smooth user experience, and appeared as intended.

**Unit Testing**

Focused on core functions like total price calculation and swipe gesture recognition.

**Integration Testing**

Tested combined flows like login-then-checkout, discover-to-cart, etc.

**End to End (E2E) Testing**

Simulated user flows from opening the app to purchasing items.

**Performance Testing**

Measured load times and responsiveness across devices and OS versions.

**8.1 Reflection on Testing Effectiveness**

**Strengths**

Thorough testing across multiple devices

Identified and fixed critical bugs during swiping and navigation

**Challenges**

Lack of real back-end made session testing complex

Manual testing required for most flows due to MVP

Scope Restrictions and limitation of the app

**9.Future Enhancements**

To further enhance functionality, scalability, and user experience, the following improvements are proposed for future versions of Quick Mart:

**Back-end Integration:** Implement a full backend using Node.js and MongoDB or Firebase for dynamic product data, user authentication, and persistent storage.

**Payment Gateway Integration:** Enable secure online payment using services such as Stripe, Razorpay, or PayPal.

**Advanced Search & Filters:** Provide filters (e.g., price range, brand, color, size) to restrict product discovery.

**Push Notifications:** Notify users about offers, new arrivals, and cart reminders.

**User Reviews & Ratings:** Allow users to review products and provide star ratings for better buyer decisions.

**Order Tracking:** Enable real-time order status and delivery tracking within the profile.

**Admin Panel:** Develop a web-based admin dashboard for product and order management.

**Multilingual Support:** Expand accessibility by adding support for multiple languages and regions.

**9.1 Some of Future Integrations**

Real-time database with Firebase or MongoDB

Online payment gateways (Stripe, Razorpay)

Push notifications and order tracking

**9.2 Alignment with Project Objectives and User Needs**

These enhancements aim to strengthen Quick Mart’s value by improving usability, performance, and scalability, aligning with the needs of fast-paced mobile users and modern shopping expectations.

**9.3 Technical Restrictions**

**Device Fragmentation:** Variations in screen sizes and OS versions across Android and iOS devices required thorough cross-device testing.

**Gesture Consistency:** Swipe gestures needed precise implementation using react-native-gesture-handler to avoid misfires or accidental actions.

**Navigation Stack Limits:** Edge cases like back-navigation in deep navigation stacks required handling to prevent cras

**State Persistence:** Without a full back-end, preserving state (cart, favorites) across sessions had technical limitations in the MVP version.

**9.4 Operational Restrictions**

No Backend Integration (MVP):

Due to scope, real-time backend operations like order tracking, payment processing, and inventory management were not implemented.

**Manual Authentication Simulation:** Login/logout flows are handled via mock logic, restricting real user validation or data persistence.

**Static Product Data:** Shoe listings are statically defined in the app. Dynamic updates via a database are a planned enhancement.

**9.5 Strategic Restrictions**

**Focus on Shoes Only:** The app is intentionally scoped to sell only shoes to maintain brand clarity and UX simplicity.

**Limited Checkout Flow:** Payment gateway and shipping functionality are omitted in the MVP, simplifying development but limiting commercial use.

**Offline Functionality:** The app currently does not support offline browsing or caching, which limits usability without internet access.

**Conclusion**

The Quick Mart mobile app properly depicts the design of a neat, effective, and easy-to-use local shoe purchasing optimized e-commerce website. Built using a mobile-first approach, it contains easy-to-use navigation, responsive UI/UX, and main features such as product discovery, favorites, cart management, and user authentication. Built using React Native and designed using Figma, the app emphasizes modular code structure, maintainability, and best shopping experience on both the Android and iOS platforms.

The app succeeds in its fundamental mission: to offer a minimalist but thorough shopping experience with speed, simplicity, and usability as its highest priorities. Through iterative agile development and continuous testing, Quick Mart stands as a scalable foundation for future enhancements in mobile commerce.