Software Requirements Specification (SRS)

PIDDZ Pizza Delivery Application

Version 1.0
Date: September 29, 2025
Team Members: Daniel Araujo, Prachish Pandey, Ishrak Mulla, Djoulie Saint Louis, Zeyu Zhang

---

1. Introduction

1.1 Purpose

This Software Requirements Specification document provides a complete description of the functional and non-functional requirements for the PIDDZ Pizza Delivery Application. This document is intended for developers, testers, and stakeholders involved in the project.

1.2 Scope

PIDDZ is a web-based pizza ordering application designed for small to medium-sized pizza businesses. The system enables customers to browse menus, customize pizzas, manage shopping carts, and place orders. Business owners can manage incoming orders through a dashboard interface.

1.3 Definitions and Acronyms

SRS refers to Software Requirements Specification, which documents all system requirements. UI represents User Interface, the visual elements users interact with. API stands for Application Programming Interface, used for system communication. PIDDZ is the project name for this Pizza Delivery Application.

1.4 References

This document references the Software Project Management Plan (SPMP), Software Configuration Management Plan (SCMP), and the Project Proposal Document submitted previously.

---

2. Overall Description

2.1 Product Perspective

PIDDZ is a standalone web application that provides an alternative to expensive third-party delivery platforms. The system consists of a customer-facing interface for ordering and a business dashboard for order management. Unlike generic e-commerce platforms, PIDDZ focuses specifically on pizza ordering with industry-specific customization features.

2.2 Product Functions

The system provides interactive menu browsing with category filtering, allowing customers to easily find desired items. Pizza customization enables users to select size, crust type, and additional toppings. Shopping cart management maintains selected items throughout the browsing session. The order checkout process collects customer information and confirms orders. A business dashboard allows staff to view and manage incoming orders. Real-time price calculation ensures customers always see accurate costs based on their selections.

2.3 User Classes and Characteristics

The primary users are customers who order pizza through the web interface. These users possess basic web browsing skills and use the system occasionally to frequently. Their main tasks include browsing the menu, customizing orders according to preferences, and completing checkout.

Secondary users are business staff members who manage orders and operations. These users have moderate computer skills and interact with the system daily. Their responsibilities include viewing incoming orders, updating order status, and managing menu items.

2.4 Operating Environment

The client-side application runs on modern web browsers including Chrome, Firefox, Safari, and Edge. The server-side infrastructure utilizes Firebase hosting and database services. The platform operates cross-platform, supporting both desktop and mobile devices without requiring native applications.

2.5 Design and Implementation Constraints

The system must function without requiring user authentication in the minimum viable product stage. Implementation is limited to Firebase free tier resources to maintain zero cost. Integration with actual payment processors is excluded, with the system operating in

demonstration mode. The application must be deployable locally for presentation and demonstration purposes.

2.6 Assumptions and Dependencies

The system assumes users have stable internet connectivity during ordering. Browser compatibility requires support for HTML5, CSS3, and ES6 JavaScript features. Firebase services must remain available and within free tier limits for the project scope. The development team has access to necessary tools and platforms throughout the project timeline.

---

3. System Features

3.1 Menu Browsing

This high-priority feature allows users to view available menu items organized by category. The system displays all menu items with name, description, and base price clearly visible. Category filters enable users to view all items or filter by specific categories including pizzas, sides, drinks, and desserts. Items appear in a responsive grid layout that adapts to different screen sizes. Each menu item includes an appropriate action button, either "Customize" for pizzas or "Add to Cart" for other items.

Functional Requirements:

FR-1.1 states that the system shall display all menu items with name, description, and price. FR-1.2 requires the system to provide category filters for All, Pizzas, Sides, Drinks, and Desserts. FR-1.3 specifies that the system shall display items in a responsive grid layout. FR-1.4 mandates that each menu item shall include an action button for customization or direct cart addition.

3.2 Pizza Customization

This high-priority feature enables users to customize pizza orders according to their preferences. The customization interface opens in a modal dialog when users select a pizza. Size selection offers small, medium, and large options with corresponding price adjustments. Crust type selection includes thin crust, regular, thick crust, and stuffed crust options. Multiple toppings can be selected from an available list. Special instructions can be entered through a text field for additional preferences. The total price updates dynamically as users make selections.

Functional Requirements:

FR-2.1 requires the system to provide size selection options including Small, Medium, and Large. FR-2.2 mandates crust type selection including Thin, Regular, Thick, and Stuffed options. FR-2.3 specifies that the system shall allow selection of multiple toppings. FR-2.4 states that available toppings must be displayed with individual pricing. FR-2.5 requires a text field for special instructions. FR-2.6 mandates dynamic price calculation based on all selections.

3.3 Shopping Cart Management

This high-priority feature enables users to add, view, and manage items in their shopping cart. When users confirm customized items, they are added to the cart with all selections preserved. A persistent cart counter displays in the page header, updating in real-time as items are added. Cart items remain stored during the entire browsing session. Users can view a cart summary showing all selected items. The system calculates and displays the total cart value including all customizations.

Functional Requirements:

FR-3.1 states that the system shall add customized items to cart when user confirms selections. FR-3.2 requires the system to display cart item count in a persistent header. FR-3.3 mandates that cart items be stored during the browsing session. FR-3.4 specifies that users can view a cart summary. FR-3.5 requires calculation and display of total cart value.

3.4 Order Checkout

This high-priority feature allows users to review their order and submit it for processing. The checkout page displays a complete order summary with all items, customizations, and prices. Customer information fields collect name, delivery address, and phone number. Form validation ensures all required fields are completed before submission. Upon successful submission, an order confirmation page displays with order details and estimated preparation time. The system generates a unique order ID for tracking purposes.

Functional Requirements:

FR-4.1 requires the system to display order summary with all items and prices. FR-4.2 mandates collection of customer information including name, address, and phone. FR-4.3 specifies that required fields must be validated before submission. FR-4.4 states that the system shall provide order confirmation with estimated time. FR-4.5 requires generation of unique order ID for tracking.

3.5 Business Dashboard

This medium-priority feature provides business staff with order management capabilities. The dashboard displays a list of all pending orders in chronological order. Staff can view detailed information for each order including all items, customizations, and customer contact information. Order status can be updated to reflect current preparation stage. The interface provides clear visibility of order queue and allows efficient order processing.

Functional Requirements:

FR-5.1 states that the system shall display a list of pending orders. FR-5.2 requires display of order details including items and customer information. FR-5.3 mandates that staff can update order status. FR-5.4 specifies that orders be displayed in chronological order.

---

4. User Stories

4.1 Customer User Stories

Story US-1: Browse Menu describes a customer who wants to browse available menu items to decide what to order. The acceptance criteria require that the menu displays with clear images, names, descriptions, and prices for all items. This story has high priority as it represents the entry point for all customer interactions.

Story US-2: Filter by Category involves a customer wanting to filter menu items by category to quickly find desired items. The acceptance criteria specify that filter buttons must show only items from the selected category. This high-priority story improves user experience by reducing information overload.

Story US-3: Customize Pizza represents a customer wanting to customize their pizza with different sizes, crusts, and toppings to get exactly what they want. Acceptance criteria require that the customization modal opens with all options and that selections are reflected in the price. This high-priority story delivers core value proposition of the application.

Story US-4: See Real-Time Price describes a customer wanting to see price updates as they customize their order to know the total cost before adding to cart. The acceptance criteria mandate that price updates immediately when options change. This high-priority story builds customer confidence and transparency.

Story US-5: Add to Cart involves a customer wanting to add items to their cart to order multiple items at once. Acceptance criteria require that items are added with all customizations preserved and that the cart counter updates. This high-priority story enables the core ordering workflow.

Story US-6: View Cart represents a customer wanting to view cart contents to review their order before checkout. The acceptance criteria specify that the cart displays all items with customizations and total price. This high-priority story provides order confirmation before commitment.

Story US-7: Complete Checkout describes a customer wanting to enter delivery information and confirm their order so it can be prepared. Acceptance criteria require form validation of all required fields and display of confirmation page with order details. This high-priority story completes the customer ordering process.

4.2 Business User Stories

Story US-8: View Incoming Orders involves a business owner wanting to see all incoming orders to prepare them efficiently. The acceptance criteria require that the dashboard displays all orders with timestamps. This medium-priority story enables order fulfillment workflow.

Story US-9: Update Order Status describes a business owner wanting to update order status so customers know their order progress. Acceptance criteria specify that status can be changed to Preparing, Ready, Out for Delivery, or Completed. This medium-priority story improves customer communication.

Story US-10: View Order Details represents a business owner wanting to see complete order details to prepare orders correctly. The acceptance criteria mandate that order view shows all items, customizations, and customer information. This medium-priority story ensures order accuracy.

---

5. Non-Functional Requirements

5.1 Performance Requirements

The system must meet specific performance standards to ensure satisfactory user experience. Pages shall load within 3 seconds on standard broadband connection (NFR-1). Cart operations including adding and updating items shall complete within 500 milliseconds (NFR-2). The system shall support at least 10 concurrent users without degradation (NFR-3).

5.2 Security Requirements

Security measures protect user data and system integrity. Customer data shall be transmitted securely using HTTPS protocols (NFR-4). The business dashboard shall require

authentication in future implementations (NFR-5). No sensitive payment information shall be stored in the system (NFR-6).

5.3 Usability Requirements

The interface must be accessible and intuitive for target users. The interface shall be intuitive for users with basic web browsing skills (NFR-7). The system shall provide clear feedback for all user actions including confirmations and errors (NFR-8). Error messages shall be clear and actionable, guiding users toward resolution (NFR-9).

5.4 Reliability Requirements

The system must maintain consistent availability and data integrity. The system shall maintain 95% uptime during the demonstration period (NFR-10). Cart data shall persist throughout the browsing session without loss (NFR-11).

5.5 Maintainability Requirements

Code quality standards ensure long-term project sustainability. Code shall follow consistent naming conventions using camelCase for JavaScript variables and functions (NFR-12). All functions shall include comments documenting their purpose and key operations (NFR-13). Code shall be modular and reusable to facilitate future enhancements (NFR-14).

5.6 Compatibility Requirements

The application must function across diverse user environments. The system shall work correctly on Chrome, Firefox, Safari, and Edge browsers in their current versions (NFR-15). The interface shall be responsive for mobile and desktop devices (NFR-16). The system shall display correctly on screen sizes from 320 pixels to 1920 pixels wide (NFR-17).

---

6. Interface Requirements

6.1 User Interfaces

The user interface employs a clean, modern design with a consistent color scheme based on red tones for branding. The layout follows a mobile-first responsive approach, ensuring optimal display across devices. Navigation flows clearly between menu, cart, and checkout sections. All interactive elements provide visual feedback through hover states and click animations.

6.2 Software Interfaces

The application integrates with Firebase Realtime Database for persistent data storage of menu items and orders. Firebase Hosting provides web deployment infrastructure. Browser localStorage may be used for non-critical session persistence where appropriate.

---

7. Testing Requirements

7.1 Functional Testing

Functional testing verifies that all features work according to specifications. Unit tests validate individual components such as price calculation functions and cart management. Integration tests verify complete user flows from menu browsing through order confirmation. Selenium automated tests simulate user interactions with the interface to ensure proper functionality.

7.2 Usability Testing

Usability testing ensures the interface meets user experience standards. User testing with sample customers provides feedback on navigation and clarity. Mobile responsiveness verification confirms proper display on smartphones and tablets. Cross-browser compatibility testing validates functionality across different browsers.

7.3 Performance Testing

Performance testing measures system responsiveness and capacity. Load time measurement verifies that pages meet the 3-second requirement. Cart operation speed verification ensures sub-500ms response times. Concurrent user simulation tests system behavior under multiple simultaneous users.

---

8. Team Member Contributions (Since Last Submission)

Ishrak Mulla – Menu Developer

Since the last submission, Ishrak has completed the interactive menu browsing page with full functionality. He implemented the menu grid layout with responsive design, created the category filtering system that allows users to view items by type, and built the complete pizza customization modal. The customization feature includes size selection, topping checkboxes with real-time price updates, and integration with the cart system. He also developed comprehensive Selenium test scripts to verify all menu functionality, including browsing, filtering, customization, and cart integration. His work ensures that the menu page is fully functional and well-tested.

Prachish Pandey – Shopping Cart Developer

Prachish has completed the shopping cart functionality, implementing the core cart state management system. He developed the cart counter that displays in real-time as items are added, created the cart data structure that maintains all item details including customizations, and implemented the logic for adding both customized pizzas and simple items to the cart. He ensured that cart data persists during the browsing session and integrated the cart preview functionality. His work provides the critical bridge between menu selection and checkout, ensuring customer selections are accurately tracked throughout their ordering experience.

Zeyu Zhang – Checkout Flow Developer

Zeyu has been developing the checkout interface and order confirmation system. He designed the checkout page layout that displays the complete order summary, created form fields for customer information collection including name, address, and contact details, and implemented form validation to ensure all required information is provided. He also built the order confirmation page that displays order details and estimated delivery time. His work focuses on making the final step of ordering clear and user-friendly while collecting all necessary information for order fulfillment.

Djoulie Saint Louis – UI/Styling Specialist

Djoulie has refined the visual design and ensured consistency across all components. She created the unified color scheme using red tones for branding, implemented responsive CSS using Flexbox and Grid for optimal display on all devices, and styled the modal popup for customization with smooth transitions. She also designed the cart preview badge and ensured all buttons and interactive elements have clear hover states and visual feedback. Her contributions ensure the application looks professional and provides an excellent user experience across desktop and mobile devices.

Daniel Araujo – Integration & Navigation Developer

Daniel has worked on connecting all team components into a cohesive application. He set up the GitHub repository structure with clear branching strategy, created the navigation flow between menu, customization modal, and cart views, and implemented the modal open/close functionality. He also integrated Firebase configuration for the backend, established the project file structure for easy collaboration, and documented the setup process for other team members. His integration work ensures that individually developed features work together seamlessly as a unified application.

---

9. GitHub Repository

The project repository is available at: https://github.com/DanDaMan2121/PIDDZ.git

The repository contains complete source code written in HTML, CSS, and JavaScript. Test scripts include Selenium Python tests for automated validation. Documentation encompasses this SRS, the SPMP, SCMP, and Project Proposal. Setup and deployment instructions guide new developers through environment configuration. Meeting notes and progress tracking documents maintain project history.

---

10. Appendices

Appendix A: Glossary

The cart represents temporary storage for items a customer intends to order during their session. Customization refers to the process of modifying a pizza by selecting size, crust type, and additional toppings. A modal is a popup dialog box that overlays the main interface for focused user interaction. Toppings are additional ingredients that can be added to a pizza base for an extra charge.

Appendix B: Change Log

Version 1.0 was created on September 29, 2025, representing the initial SRS document for the project.

Document Approval

Prepared by: Group 4 PIDDZ Team
Date: September 29, 2025
Status: Approved for Development Phase