Software Requirements Specification Template

The following annotated template shall be used to complete the Software Requirements Specification (SRS) assignment. The instructor must approve any modifications to the overall structure of this document.

**Template Usage:**

Text contained within angle brackets (‘<’, ‘>’) shall be replaced by your project-specific information and/or details. For example, <Project Name> will be replaced with either ‘Smart Home’ or ‘Sensor Network’ etc.

Italicized text is included to briefly annotate the purpose of each section within this template. This text should not appear in the final version of your submitted SRS.

This page is not a part of the final template and should be removed before your SRS is submitted.

Paws and Claws Market Place

Software Requirements Specification

<Version>

<Date>

Sohaib Sikandar , Shahzain Khan

Lead Software Engineer

Prepared for

BS IT

Instructor:

Fall 2023

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Description** | **Author** | **Comments** |
| <date> | <Version 1> | <Your Name> | <First Revision> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

|  |  |  |  |
| --- | --- | --- | --- |
| **Signature** | **Printed Name** | **Title** | **Date** |
|  | <Your Name> | Lead Software Eng. |  |
|  | Dr Baryalai | Instructor |  |
|  |  |  |  |

**Table of Contents**

Revision History ii

Document Approval ii

1. Introduction 1

1.1 Purpose 1

1.2 Scope 1

1.3 Definitions, Acronyms, and Abbreviations 1

1.4 References 1

1.5 Overview 1

2. General Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Characteristics 2

2.4 General Constraints 2

2.5 Assumptions and Dependencies 2

3. Specific Requirements 2

3.1 External Interface Requirements 3

3.1.1 User Interfaces 3

3.1.2 Hardware Interfaces 3

3.1.3 Software Interfaces 3

3.1.4 Communications Interfaces 3

3.2 Functional Requirements 3

3.2.1 <Functional Requirement or Feature #1> 3

3.2.2 <Functional Requirement or Feature #2> 3

3.3 Use Cases 3

3.3.1 Use Case #1 3

3.3.2 Use Case #2 3

3.4 Classes / Objects 3

3.4.1 <Class / Object #1> 3

3.4.2 <Class / Object #2> 3

3.5 Non-Functional Requirements 4

3.5.1 Performance 4

3.5.2 Reliability 4

3.5.3 Availability 4

3.5.4 Security 4

3.5.5 Maintainability 4

3.5.6 Portability 4

3.6 Inverse Requirements 4

3.7 Design Constraints 4

3.8 Logical Database Requirements 4

3.9 Other Requirements 4

4. Analysis Models 4

4.1 Sequence Diagrams 4

4.2 Data Flow Diagrams (DFD) 4

A. Appendices 5

A.1 Appendix 1 5

A.2 Appendix 2 5

# 1. Introduction

The introduction to the Software Requirement Specification (SRS) document should provide an overview of the complete SRS document. While writing this document please remember that this document should contain all the information needed by a software engineer to adequately design and implement the software product described by the requirements listed in this document. (Note: the following subsection annotates are largely taken from the IEEE Guide to SRS).

## 1.1 Purpose

The SRS serves as a contractual agreement between the stakeholders and the development team, ensuring everyone has a shared understanding of the software’s purpose, functionality, and limitations. It is essential for the success of any software project.

## 1.2 Scope

## Scope of the Project: Paws and Claws Marketplace

## The Paws and Claws Marketplace is a web-based platform that integrates pet adoption, buying/selling of pets (focused on dogs, cats, and rabbits), and an online shop for pet products.

## What It Will Do:

## Provide a user-friendly interface for posting and browsing pet listings with detailed descriptions and images.

## Enable ethical pet adoption with essential checkpoints for responsible ownership.

## Offer a centralized marketplace and integrated pet product shop.

## Support secure transactions and ensure mobile responsiveness.

## What It Will Not Do:

## Exclude advanced AI-based matching or veterinary service functionalities.

## Application:

## The platform benefits users by reducing fragmentation, promoting ethical adoption, and providing a seamless experience for all pet-related needs. It aligns with SDG 11 (Sustainable Cities) and SDG 15 (Life on Land) by fostering responsible pet ownership and animal welfare.

## Goal: Deliver an all-in-one, scalable platform for pet enthusiasts, ensuring ease of use and ethical practices.

## 1.3 Definitions, Acronyms, and Abbreviations

1. SRS: Software Requirements Specification – A document that describes the functionalities, constraints, and goals of the software project.

2. SDG: Sustainable Development Goals – Global goals defined by the United Nations, specifically Goal 11 (Sustainable Cities) and Goal 15 (Life on Land), which are relevant to this project.

3. UI/UX: User Interface/User Experience – Refers to the design and functionality of the platform to ensure a seamless user experience.

4. API: Application Programming Interface – A set of tools and protocols for building and integrating software applications.

5. DBMS: Database Management System – Used to manage and retrieve user and listing data (e.g., MongoDB in this project).

6. Adoption Checkpoints: Steps or criteria set by pet owners to ensure ethical and responsible pet adoption.

7. Marketplace: A virtual platform for buying and selling pets and pet products.

8. Responsive Design: A web design approach ensuring the platform works seamlessly across devices like smartphones, tablets, and desktops.

9. Classified-Style Interface: A layout similar to classified ads, where users can post and browse listings easily.

10. Pet Listings: Ads posted on the platform for pet adoption, buying, or selling.

These terms are critical to understanding the functionality and scope of the project.

## 1.4 References

## 1. \*\*Pet Adoption Platforms\*\*: shelters and rescue organizations to adopt pets

## - \*\*Petfinder\*\*: Website: [https://www.petfinder.com](https://www.petfinder.com)

## - \*\*Adopt-a-Pet\*\*: Similar to Petfinder

## - Website: [https://www.adoptapet.com](https://www.adoptapet.com)

## 2. \*\*Pet Marketplace Platforms\*\*: buying and selling of pets.

## - \*\*Gumtree\*\* and \*\*PetClassifieds\*\*:

## - Gumtree: [https://www.gumtree.com](https://www.gumtree.com)

## - PetClassifieds: [https://www.petclassifieds.com](https://www.petclassifieds.com)

## 3. \*\*Pet Product Retailers\*\*: well-known platforms that offers a wide range of pet supplies.

## - \*\*Chewy\*\*: - Website: [https://www.chewy.com](https://www.chewy.com)

## - \*\*Petco\*\*: Website: [https://www.petco.com](https://www.petco.com)

## 1.5 Overview

# (1) Description of the SRS Contents

# The SRS outlines the purpose, scope, functionalities, and requirements of the Paws and Claws Marketplace. It details the platform's features, performance criteria, system architecture, and constraints to guide development and ensure stakeholder alignment.

# (2) Organization of the SRS

# Introduction: Purpose, scope, and definitions.

# Overall Description: Summary of functionality and constraints.

# Requirements: Functional and non-functional specifications.

# Appendices: Glossary and references.

# This structure ensures clarity and ease of use for all stakeholders

# .2. General Description

## 2.1 Product Perspective

## This platform is a standalone solution that connects pet lovers nationwide, focusing on pet adoption, buying, selling, and offering pet-related products (dogs, cats, and rabbits). It features a classified-style interface, aligning with SDG Goal 11 (Sustainable Cities and Communities) and SDG Goal 15 (Life on Land).

## 2.2 Product Functions

## Pet Listings: Users can view, add, and manage pet listings for adoption, sale, or purchase.

## Search and Filters: Advanced search options to help users find specific pets or products.

## User Profiles: Users can create and manage profiles to list pets, track transactions, and communicate with others.

## Secure Transactions: Integration of payment gateways for secure transactions between buyers and sellers.

## Review and Rating: Users can rate and review transactions, enhancing trust and reliability.

## Product Catalog: Vendors can list pet-related products with details and pricing.

## Admin Dashboard: Admins can manage the platform, review user reports, and oversee the content.

## 2.3 User Characteristics

## Target users include pet lovers, adopters, vendors, and rescue organizations. The platform caters to users with varying technical skills, offering an intuitive interface and easy navigation

## 2.4 General Constraints

The platform must comply with data privacy regulations and support a wide range of devices with an optimized, responsive design.

## 2.5 Assumptions and Dependencies

The platform assumes users have internet access and uses modern web browsers. It may integrate with third-party payment gateways and vendors for product listings.

# 

# 3. Specific Requirements

**User Management**

Users must be able to create, edit, and delete their accounts.

User profiles must include basic information such as name, email, phone number, and location.

Users must be able to reset their passwords via email verification.

Users must be able to view their account activity, including listings posted and transactions.

**Pet Listings**

Users must be able to add, edit, and delete pet listings.

Listings must include details like pet type (dog, cat, rabbit), breed, age, health status, and images.

Listings must be searchable by breed, location, and other relevant attributes.

Users must be able to mark listings as sold or adopted.

Listings must have a contact method (email or phone number) for potential buyers.

**Search and Filters**

The platform must include a search bar for finding pets based on attributes such as breed, location, and type.

Users must be able to apply multiple filters (e.g., breed, size, adoption status).

Search results should display relevant listings with an option to sort by date or price.

**Transactions and Payments**

The platform must support secure payment processing for pet-related products and adoption fees.

Payment transactions must be logged with timestamps for reference.

Users must receive email confirmations after a successful transaction.

Users must be able to review and rate transactions.

**Product Listings (Vendors)**

Vendors must be able to create, edit, and delete product listings (pet supplies, food, accessories).

Product listings must include product name, description, price, and images.

Product listings must be searchable by category (e.g., food, toys, accessories).

Vendors must be able to view transaction history for their products.

**Admin Functions**

Admins must have the ability to manage user accounts, including suspending or deleting accounts if necessary.

Admins must be able to monitor and manage all pet and product listings.

Admins must have access to transaction logs and user activity reports.

**Security and Privacy**

The platform must encrypt sensitive user data, including passwords and payment details.

The platform must comply with data protection regulations (e.g., GDPR).

Users must have the ability to delete their accounts and all associated data.

**User Interface (UI)**

The platform must be responsive and accessible on both desktop and mobile devices.

The UI must be user-friendly, with intuitive navigation and minimal clicks for key tasks.

The design must adhere to accessibility standards (e.g., color contrast, screen reader compatibility).

**Notifications**

Users must receive email notifications for key activities (e.g., new listings, transaction updates).

Users must be able to opt out of non-essential notifications.

**Scalability and Performance**

The platform must support up to 10,000 simultaneous users without significant performance degradation.

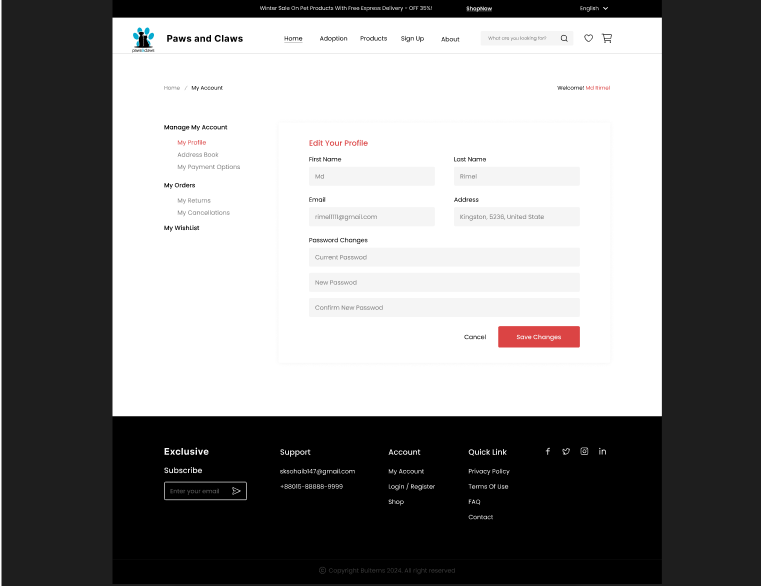
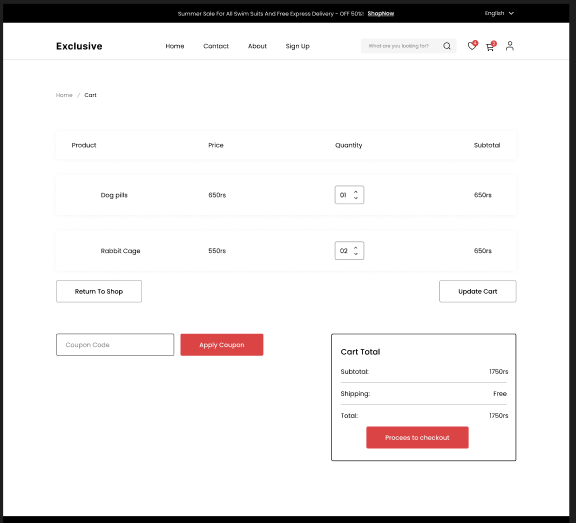
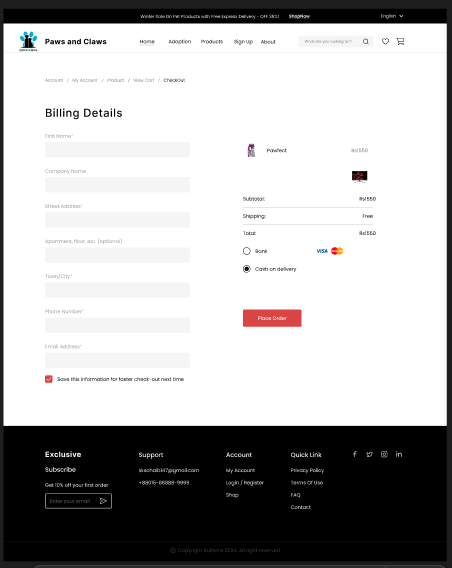
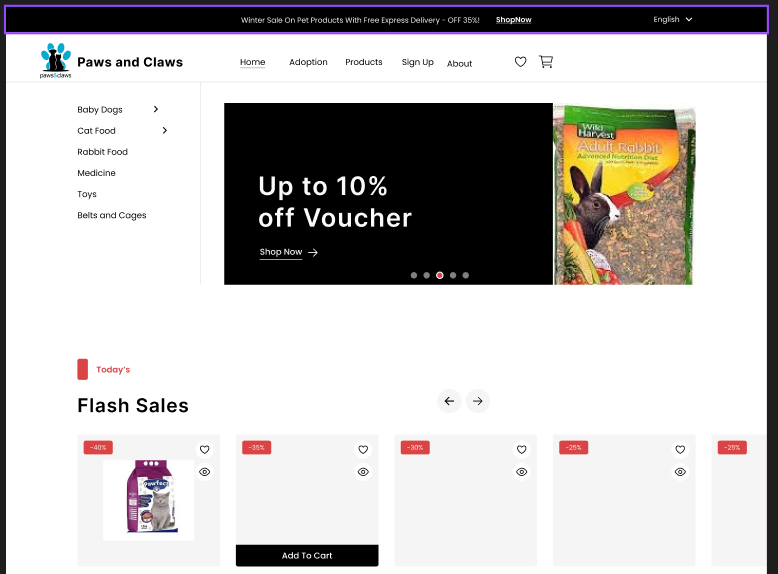
Page load times must be under 3 seconds on average

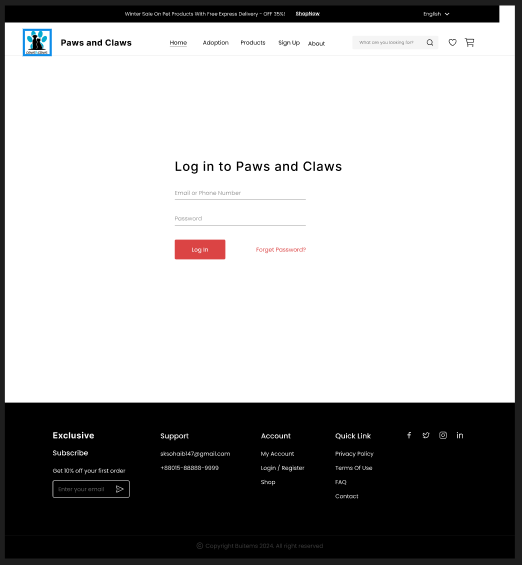
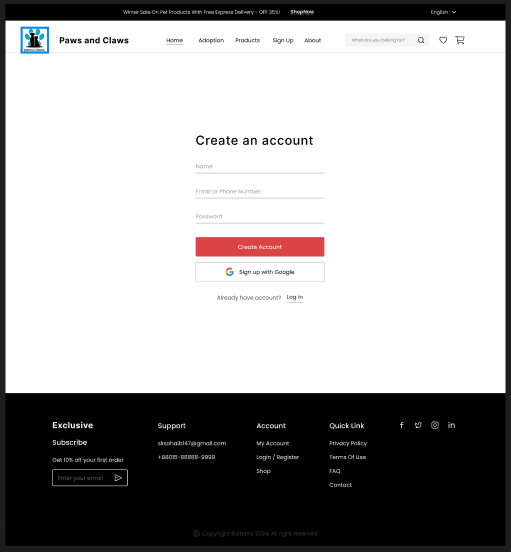
## 3.1 External Interface Requirements

### 

### 3.1.1 User Interfaces

Description: The system must provide an intuitive and responsive interface for users across various devices (e.g., desktop, tablet, mobile).





Key Features:

A home page with search functionality and featured listings.

User registration and login forms.

Dashboard for sellers to manage listings and transactions.

Profile pages for buyers and sellers.

Notifications and messaging systems for user interactions.

Standards: Follow WCAG 2.1 guidelines for accessibility.

### 3.1.2 Hardware Interfaces

Description: The platform is cloud-based and does not depend on specific hardware beyond standard consumer devices.

Requirements:

Compatibility with devices like smartphones, tablets, laptops, and desktops.

Support for peripherals such as keyboards, mice, and touchscreens.

### Specifications:

### Minimum hardware: Dual-core CPU, 2GB RAM, and a modern browser.

### 3.1.3 Software Interfaces

### Description: The system must integrate with various software systems to support its functionality.

### Requirements:

### Database: MongoDB for storing user data, listings, and transactions.

### Payment Gateway: Integration with Stripe and PayPal for secure transactions.

### Cloud Storage: Use AWS S3 or similar for storing pet images and media files.

### APIs: Integration with social media APIs for user authentication (Google, Facebook).

### Email Service: Use SendGrid or equivalent for email notifications.

### 3.1.4 Communications Interfaces

Description: The system must facilitate secure communication between users, components, and third-party services.

Requirements:

Protocols: HTTPS for all user interactions to ensure data encryption.

Messaging: Enable real-time messaging using WebSocket or similar technology.

Third-Party Integrations: REST APIs for communication with payment gateways, email services, and social media platforms.

Error Handling: All communication interfaces must include mechanisms for retries and error reporting.

## 3.2 Functional Requirements

**3.2.1 :User Registration and Profile Management**

Introduction:

Users should be able to register for an account, create a profile, and manage their personal information.

Inputs:

User’s name, email, password, phone number, and location.

Profile picture (optional).

Processing:

Validate user inputs (e.g., check for valid email format, password strength).

Store user information in the database.

Send a confirmation email for account verification.

Outputs:

Successful account creation confirmation.

Profile page with user information.

Email verification request.

Error Handling:

Invalid email format: display an error message prompting for correction.

Weak password: notify the user to create a stronger password.

Email already in use: inform the user and suggest login or password recovery options.

**3.2.2 :Pet Listing Management**

Introduction:

Users can add, edit, or delete pet listings for adoption or sale.

Inputs:

Pet type (dog, cat, rabbit), breed, age, health status, images, and description.

Contact information (phone or email).

Processing:

Validate input data (e.g., ensure required fields are filled, check image size).

Store pet listing in the database, making it searchable.

Outputs:

Confirmation of pet listing creation.

Display of new listing on the platform.

Email notification for listing confirmation.

Error Handling:

Missing required fields: prompt user to fill in the missing details.

Invalid image file: prompt the user to upload a valid file format.

**3.2.3 Search and Filter Listings**

Introduction:

Users can search for pet listings and filter results by various criteria.

Inputs:

Search keywords (e.g., breed, location).

Filter options (e.g., age, adoption status).

Processing:

Match search keywords with pet listings.

Apply filter criteria to narrow down the search results.

Sort results based on selected criteria (e.g., date, price).

Outputs:

Display filtered and sorted pet listings.

Option to view detailed pet information.

Error Handling:

No search results: display a message indicating no matches found and suggest modifying search terms.

Invalid filter values: prompt the user to select valid options.

**3.2.4 Transaction and Payment Processing**

Introduction:

The platform should support secure payment for transactions related to pet adoption or purchases.

Inputs:

Payment details (credit/debit card, payment gateway credentials).

Transaction amount.

Processing:

Securely transmit payment data to the payment gateway.

Process payment and update transaction records.

Generate transaction confirmation.

Outputs:

Payment success or failure notification.

Transaction record in the user's profile.

Email receipt for the transaction.

Error Handling:

Payment failure: prompt the user to check payment details and try again.

Transaction timeout: inform the user to retry or contact support.

## 3.3 Use Cases

**3.3.1 Use Case: User Registration**

**Use Case ID: UC-01**

Actor(s): User

Description: This use case allows a user to register an account on the platform by providing personal details and creating a profile.

Preconditions: User is not logged in.

Postconditions: User account is created, and an email verification link is sent.

Normal Flow:

1. User navigates to the registration page.

2. User enters required details (name, email, password, phone number).

3. User submits the form.

4. System validates input and creates an account.

5. System sends a verification email to the user.

6. User is redirected to the login page.

Alternative Flow:

If the email is already in use, the system prompts the user to log in or reset the password.

Exceptions:

If the email format is invalid, the system displays an error message.

If the password does not meet criteria, the system asks the user to create a stronger password.

Frequency of Use: Low (one-time action per user)

Assumptions: User has a valid email address.

**3.3.2 Use Case: Add Pet Listing**

**Use Case ID: UC-02**

Actor(s): User (Pet owner or Vendor)

Description: This use case allows a user to add a pet for adoption or sale to the platform.

Preconditions: User must be logged in.

Postconditions: Pet listing is added to the platform and made visible to other users.

Normal Flow:

1. User logs in and navigates to the "Add Pet Listing" page.

2. User fills out the pet's details (type, breed, age, health, description, images).

3. User submits the form.

4. System validates inputs and creates the listing.

5. The new listing is displayed on the platform and made searchable.

Alternative Flow:

If any required fields are missing, the system prompts the user to complete them.

Exceptions:

If the image format is unsupported, the system prompts the user to upload a valid format.

If the description exceeds the character limit, the system prompts the user to shorten it.

Frequency of Use: Medium (users may add multiple listings over time)

Assumptions: User has clear and accurate pet details to add.

**3.3.3 Use Case: Search Pet Listings**

**Use Case ID: UC-03**

Actor(s): User (Adopter or Buyer)

Description: This use case allows a user to search for pet listings using filters like breed, location, and age.

Preconditions: User is logged in.

Postconditions: Search results are displayed based on user criteria.

Normal Flow:

1. User navigates to the search page.

2. User enters search criteria (e.g., breed, location, adoption status).

3. User applies filters if necessary (e.g., age, price range).

4. System retrieves and displays matching listings.

5. User can select a listing to view more details.

Alternative Flow:

If no results are found, the system informs the user and suggests modifying the search.

Exceptions:

If an invalid filter is applied (e.g., a non-existent breed), the system displays an error message.

Frequency of Use: High (frequent searches by users)

Assumptions: Listings are properly tagged with attributes like breed, age, and location.

**3.3.4 Use Case: Process Payment**

**Use Case ID: UC-04**

Use Case Name: Process Payment

Actor(s): User (Buyer)

Description: This use case allows a user to make a secure payment for a pet adoption or product purchase.

Preconditions: User must have a valid payment method.

Postconditions: Payment is processed, and a receipt is generated.

Normal Flow:

1. User selects a pet or product to purchase.

2. User proceeds to checkout and selects a payment method.

3. User enters payment details and confirms the transaction.

4. System processes the payment securely via a payment gateway.

5. The user is notified of payment success and receives a transaction receipt.

Alternative Flow:

If the payment fails, the system displays an error and prompts the user to retry.

Exceptions:

If payment details are invalid (e.g., insufficient funds), the system alerts the user to correct the issue.

Frequency of Use: Low (only for users completing transactions)

Assumptions: User has a valid payment method and sufficient funds.…

## 3.4 Classes / Objects

## 3.4.1 User

## Attributes:

## userID (String): Unique identifier for each user.

## name (String): Full name of the user.

## email (String): User’s email address.

## password (String): Encrypted password for authentication.

## phone (String): Contact number.

## role (String): Indicates whether the user is a buyer or seller.

## Functions:

## registerUser(): Registers a new user with the platform.

## loginUser(): Authenticates the user and starts a session.

## updateProfile(): Allows the user to modify personal details.

## deleteAccount(): Deletes the user account and associated data.

## Reference:

## Functional Requirements: 3.2.1.1 User Registration, 3.2.1.2 User Login.

## Use Cases: 4.2.1 Manage User Profiles.

## 3.4.2 Pet Listing

## Attributes:

## listingID (String): Unique identifier for each listing

## title (String): Short description of the listing.

## description (String): Detailed information about the pet.

## price (Float): Cost of the pet (if applicable).

## petType (String): Type of pet (e.g., dog, cat, rabbit).

## age (Integer): Age of the pet in years or months.

## status (String): Current status (e.g., available, sold, adopted).

## Functions:

## createListing(): Allows sellers to create a new pet listing.

## editListing(): Enables sellers to update details of their listings.

## deleteListing(): Removes a listing from the platform.

## searchListings(): Searches and filters listings based on user inputs.

## Reference:

## Functional Requirements: 3.2.1.3 Pet Listing Management, 3.2.1.4 Search Functionality.

## Use Cases: 4.2.2 Create and Manage Listings.

## 3.4.3 Transaction

## Attributes:

## transactionID (String): Unique identifier for each transaction.

## buyerID (String): Identifier of the buyer involved in the transaction.

## sellerID (String): Identifier of the seller involved.

## listingID (String): Identifier of the listing involved in the transaction.

## amount (Float): Total transaction amount.

## status (String): Current status (e.g., pending, completed, canceled).

## Functions:

## initiateTransaction(): Starts a new transaction between buyer and seller.

## processPayment(): Handles payment processing through a gateway.

## updateTransactionStatus(): Updates the transaction's state based on user actions or payment outcomes.

## refundTransaction(): Handles refunds in case of cancellations.

## Reference:

## Functional Requirements: 3.2.1.5 Transaction Processing.

## Use Cases: 4.2.3 Handle Transactions.

## 3.5 Non-Functional Requirements

## 3.5.1 Performance

## The platform must process 95% of user interactions (e.g., searches, pet listings) within 2 seconds.

## The system must support up to 10,000 concurrent users without performance degradation.

## Pages must load in less than 3 seconds under normal conditions.

## 3.5.2 Reliability

## The system must have a Mean Time Between Failures (MTBF) of at least 30 days.

## 99% of transactions should be completed without errors or failures.

## The system must recover from failures within 5 minutes.

## 3.5.3 Availability

## The platform must be available 99.9% of the time, excluding scheduled maintenance.

## System downtime must not exceed 1 minute per day.

## Scheduled maintenance should not exceed 4 hours per month.

## 3.5.4 Security

## All sensitive data (e.g., passwords, payment information) must be encrypted using industry-standard encryption algorithms (e.g., AES-256).

## The system must comply with data privacy regulations such as GDPR.

## User accounts must be protected by multi-factor authentication (MFA) for sensitive actions (e.g., payment processing, profile changes).

## The system must be protected against common security threats (e.g., SQL injection, cross-site scripting).

## 3.5.5 Maintainability

## The system should allow for easy bug fixes and updates with minimal downtime.

## The platform’s source code should be modular and well-documented to facilitate future enhancements and maintenance.

## System logs should be maintained for at least 30 days to facilitate debugging and issue tracking.

## 3.5.6 Portability

## The platform must be compatible with the latest versions of major browsers (Chrome, Firefox, Safari, Edge).

## The platform must be fully responsive and accessible across all devices (desktop, tablet, mobile).

## The platform must support deployment on various cloud platforms (e.g., AWS, Google Cloud, Microsoft Azure).

## 3.6 Inverse Requirements

## Inverse requirements describe the constraints or limitations that the system should not exhibit. These are important to define to avoid undesirable behaviors. Below are some useful inverse requirements for your project:

## 3.6.1 Performance

## The system must not take more than 5 seconds to process any user interaction (e.g., search or form submission).

## The system must not experience performance degradation when more than 10,000 concurrent users are active.

## 3.6.2 Reliability

## The system must not experience more than 1 failure per week.

## The platform should not have more than 3 minutes of unplanned downtime per month.

## 3.6.3 Availability

## The system should not experience more than 0.1% downtime annually, excluding scheduled maintenance.

## The platform should not have any planned downtime exceeding 4 hours per month.

## 3.6.4 Security

## The platform must not store sensitive data (e.g., passwords, payment information) in an unencrypted format.

## The system should not expose any user data to unauthorized parties, even in case of a breach.

## The platform must not allow weak or easily guessable passwords during user registration.

## 3.6.5 Maintainability

## The platform should not have complex, monolithic code that hinders future updates or bug fixes.

## The system should not have more than 3 days of downtime for routine maintenance tasks.

## 3.6.6 Portability

## The platform should not be restricted to a specific browser or device, limiting user access.

## The system should not be tied to a single cloud provider, limiting deployment flexibility.

## 3.7 Design Constraints

## 3.7.1 Standards Compliance

## The system must comply with web accessibility standards (WCAG 2.1) to ensure the platform is usable for all users, including those with disabilities

## The system must adhere to industry-standard security practices, including OWASP Top 10 security guidelines, to protect user data and prevent common security vulnerabilities.

## The platform must be GDPR-compliant, ensuring proper handling of user data and privacy rights within the EU and other applicable regions.

## 3.7.2 Company Policies

## The platform must use the company's preferred tech stack (HTML, CSS, JavaScript, React, MongoDB), and cannot integrate with third-party frameworks or libraries without prior approval.

## All source code must be version-controlled using Git, and code should be reviewed before deployment to ensure quality and security standards are met.

## The platform should follow company coding standards and naming conventions for consistency across development teams.

## 3.7.3 Hardware Limitations

## The platform must be optimized to run efficiently on standard web hosting environments and must not require specialized or high-performance hardware.

## The system must be scalable, capable of running on cloud infrastructure (e.g., AWS, Google Cloud) to accommodate potential spikes in traffic.

## The platform must be designed to function within the limits of typical consumer devices (e.g., smartphones, tablets, laptops) for optimal user experience.

## 3.7.4 Third-Party Integrations

## Payment processing must be handled through an approved third-party gateway (e.g., Stripe, PayPal) to ensure secure and reliable transactions.

## Integration with social media platforms (e.g., Facebook, Google) for user login must adhere to their API limits and terms of use.

## The platform must support integration with email services (e.g., SendGrid) for user notifications, ensuring compliance with email deliverability standards.

## 3.7.5 Performance Constraints

## The platform must be designed to operate efficiently, with no significant delays during peak usage hours, to meet the performance targets outlined in the non-functional requirements.

## The system must ensure that images and media files are optimized for fast loading, without compromising quality, by using appropriate compression techniques and file formats.3.8

## Logical Database Requirements

## 

## 3.8.1 Database Usage

## A MongoDB database will be used for data storage, providing a NoSQL solution to handle dynamic and varied data structures, such as pet listings, user profiles, and transaction details.

## 3.8.2 Data Formats

## Data will be stored in JSON-like BSON format for flexibility and scalability.

## User data (e.g., name, email, password) will be stored as strings, while numerical values (e.g., pet age, price) will be stored as integers or floats.

## Images will be stored as links to cloud storage (e.g., AWS S3) rather than directly in the database.

## 3.8.3 Storage Capabilities

## The database should be scalable, able to handle growing data over time (e.g., increasing number of users, listings, transactions).

## Each pet listing will be stored with fields for pet details (e.g., breed, health, price) and media (e.g., image links).

## User profiles will include fields for personal details (e.g., name, email, phone number), account status, and transaction history.

## 3.8.4 Data Retention

## User data will be retained for as long as the user maintains an active account or until the user requests account deletion.

## Transaction and pet listing data will be archived after 2 years, unless it is associated with an ongoing user dispute or legal issue.

## Log files related to system activity will be retained for a period of 30 days for troubleshooting and compliance.

## 3.8.5 Data Integrity

## All data must be validated before insertion into the database to prevent inconsistencies (e.g., invalid email formats, missing required fields).

## Database transactions must be atomic, ensuring that all operations within a transaction are completed successfully or none at all (e.g., during user registration or payment processing).

## The system must implement referential integrity, ensuring that related data (e.g., user profiles and pet listings) is consistent and remains linked properly in the database.

## Backup and recovery processes must be in place to protect against data loss, with daily backups and the ability to restore the database to the previous state in case of failure.

## 3.8.6 Data Security

## All sensitive user data (e.g., passwords, payment details) will be encrypted both at rest and in transit using industry-standard encryption methods (e.g., AES-256 for storage, TLS 1.2+ for transmission).

## Access to the database will be restricted using role-based access control (RBAC), ensuring that only authorized users (e.g., administrators) can modify critical data.

## 3.9 Other Requirements

# 

# 3.9.1 Legal and Compliance

# The platform must adhere to all applicable laws and regulations, including GDPR, CCPA, and local e-commerce regulations.

# All user transactions must comply with tax requirements based on their geographical location.

# 3.9.2 User Interface (UI) and Experience (UX)

# The interface must be intuitive, allowing users to perform key actions (e.g., search, list pets) in fewer than three clicks.

# The platform must support multi-language functionality, starting with English, and allow easy expansion to additional languages.

# The design must use a responsive layout to ensure usability across devices (desktop, tablet, mobile).

# 3.9.3 Search and Filtering

# The platform must include advanced search capabilities, allowing users to filter by location, pet type, age, and price range.

# Search results must display within 2 seconds for optimal user experience.

# 3.9.4 Payment Integration

# The system must integrate with multiple payment gateways (e.g., Stripe, PayPal) to support secure transactions.

# All payment activities must comply with PCI-DSS standards.

# 3.9.5 Support and Feedback

# The platform must provide a support system (e.g., chatbot or contact form) for user queries and complaints.

# Users must be able to submit feedback and rate sellers or buyers for transparency and trust-building.

# 3.9.6 Analytics and Reporting

# Admins must have access to dashboards providing analytics on user activity, listings, and revenue.

# The system must support exporting reports in standard formats (e.g., CSV, PDF).

# 3.9.7 Environmental Considerations

# The platform’s hosting services must use environmentally friendly data centers aligned with sustainable practices.

# These additional requirements ensure that the platform remains legally compliant, user-friendly, and operationally efficient while supporting sustainability and scalability.

# 4. Analysis Models

This section outlines the analysis models used to develop the specific requirements of the project. Each model is traceable to the requirements outlined in Section 3.

## 4.1 Sequence Diagrams

## 

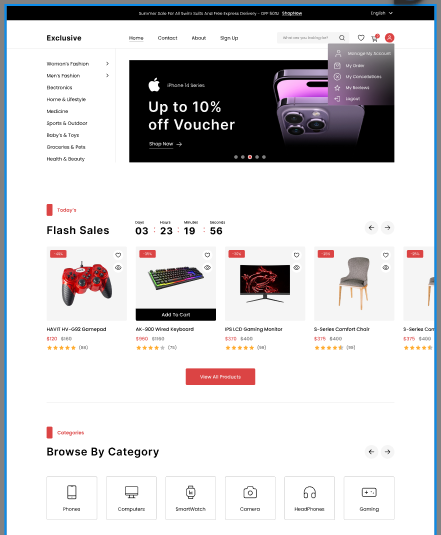
## 4.2 Data Flow Diagrams (DFD)

# A. Appendices

Appendices may be used to provide additional (and hopefully helpful) information. If present, the SRS should explicitly state whether the information contained within an appendix is to be considered as a part of the SRS’s overall set of requirements.

Example Appendices could include (initial) conceptual documents for the software project, marketing materials, minutes of meetings with the customer(s), etc.

## A.1 Appendix 1



## A.2 Appendix 2

