## **Software Requirements Specification (SRS) for JPetStore**

## **1. Introduction**

### **1.1 Purpose**

The purpose of this document is to present a detailed specification for the development of **JPetStore**, an online platform that allows users to browse, select, and purchase a variety of pets and pet-related products.  
 This SRS provides a clear understanding of the system’s functionality for:

* Software developers building the system,
* Testers validating the system,
* Project managers tracking progress,
* End users using the website,
* Admins managing catalog and orders.

### **1.2 Scope**

The **JPetStore** platform provides the following features:

* Secure user registration, login, and account management.
* Browsing, searching, and filtering of pets by categories (e.g., Dogs, Cats, Birds, Fish, Reptiles).
* Shopping cart management and checkout with multiple payment options (Debit Card, Credit Card, UPI).
* Wishlist feature for saving pets for later.
* Order tracking, history viewing, and cancellation before shipment.
* Admin dashboard for managing pets, orders, user comments, FAQs, and generating sales reports.
* Mobile-friendly responsive design to cater to users across devices.
* **Exclusions**: No physical delivery handled by the platform itself; assumed to be third-party delivery.
* No in-house payment processing (handled through integrated gateways).

### **1.3 Definitions, Acronyms, and Abbreviations**

* **JPetStore**: Java Pet Store Application
* **SRS**: Software Requirements Specification
* **UI**: User Interface
* **UX**: User Experience
* **UPI**: Unified Payments Interface
* **SKU**: Stock Keeping Unit (Product Identifier)

### **1.4 References**

* IEEE 830-1998 - Recommended Practice for Software Requirements Specifications.
* Live Demo: <https://petstore.octoperf.com/actions/Catalog.action>
* Industry standards for web security (SSL/TLS, OWASP Guidelines)

### **1.5 Overview**

The document provides an overview of the intended functionality and quality characteristics of the JPetStore system, including:

* Overall description,
* Detailed functional and non-functional requirements,
* External interface requirements,
* System constraints and assumptions.

## **2. Overall Description**

### **2.1 Product Perspective**

* JPetStore is an independent, web-based application following a **3-tier architecture**:
* **Frontend**: User Interface developed in JSP/HTML/CSS/JavaScript.
* **Backend**: Application logic and server-side operations written in Java (Servlets).
* **Database**: Stores user, product, order, and transaction data (SQL database).

The platform integrates:

* External payment gateway APIs for transactions,
* Email notification services for user communication.

**Key Integrations**:

* Payment Gateways
* SSL certificate for secure HTTPs communication
* CDN (optional) for faster static content delivery

### **2.2 Product Functions**

* User Registration, Login, Password Recovery
* Pet Listings by Categories
* Advanced Search (Name, Category, Keywords)
* Product Details with Add to Cart/Wishlist
* Cart Review, Modification, and Checkout
* Secure Payment Process
* Order Confirmation, Status Updates
* Admin Tools for Inventory and Order Management
* FAQ and Content Management
* Review and Rating System for Products

### **2.3 User Characteristics**

* **End Users**: General public, minimal technical knowledge required, familiar with online shopping.
* **Admins**: Employees or managers handling order processing, product management, and customer support, moderate technical skills required.

### **2.4 Constraints**

* Must be accessible on popular browsers (Google Chrome, Mozilla Firefox).
* Minimum RAM requirement: 256MB for smooth access.
* Backend server must support Java EE technology.
* Response time for any action (search, add to cart) should be ≤ 2 seconds under normal load.

### **2.5 Assumptions and Dependencies**

* Internet connectivity is available for all users.
* Payment gateways and email notification services are functional.
* The hosting environment supports Java web applications and SQL databases.

## **3. Specific Requirements**

### **3.1 External Interface Requirements**

#### **3.1.1 User Interfaces**

* Responsive layout adapting to mobile, tablet, and desktop screens.
* Navigation bar with Home, Catalog, Cart, Wishlist, Orders, Login/Register.
* Admin dashboard with Product Management, Order Overview, Sales Analytics.
* Error messages and system feedback (e.g., Invalid login, Payment failed).

#### **3.1.2 Hardware Interfaces**

* No specialized hardware needed.
* Hosted on cloud/server supporting standard HTTP(S) protocols.

#### **3.1.3 Software Interfaces**

* SQL database for data storage.
* Payment processing APIs for transactions.
* Email services (SMTP) for communication.

#### **3.1.4 Communications Interfaces**

* HTTPS communication to secure user data.
* RESTful APIs for payment gateway integration.

### **3.2 Functional Requirements**

#### **3.2.1 User Registration/Login**

* New users must register by filling in the required fields (name, email, password, address).
* Password must meet complexity standards (minimum 8 characters, a mix of numbers/symbols).
* Registered users can log in using credentials and recover forgotten passwords via email.

#### **3.2.2 Catalog Management**

* Pets displayed in categorized collections (Fish, Dogs, Cats, Reptiles, Birds).
* Each product shows:
  + Image, Description, Price, Availability (In Stock/Out of Stock).
* Sorting options:
  + Price (Low to High, High to Low)
  + New Arrivals

#### **3.2.3 Search and Wishlist**

* Search by keywords, pet names, or categories.
* Add/remove items to a Wishlist for future reference.

#### **3.2.4 Shopping Cart Management**

* Add pets to the shopping cart.
* View Cart page shows:
  + Item details, Quantity (editable), Price, Remove option.
* Auto-update cart total dynamically.

#### **3.2.5 Checkout and Payment**

* Collect shipping address and contact details.
* Payment Methods:
  + Debit Card
  + Credit Card
  + UPI
* Review page before final submission.
* Post-payment confirmation email and receipt.

#### **3.2.6 Order History and Tracking**

* View list of previous orders with:
  + Order ID, Date, Items, Status (Pending, Shipped, Delivered, Cancelled).
* Users can cancel orders that are pending shipment.

#### **3.2.7 Admin Management**

* Manage pet listings (CRUD operations).
* View/manage all customer orders.
* Update order statuses (Processing, Shipped, Delivered).
* View Monthly/Yearly sales reports and analytics.

#### **3.2.8 Content and FAQ Management**

* Admins can add/edit FAQs to assist users.
* Enable/disable user reviews and ratings for products.
* Support for multilingual content adaptation.

### **3.3 Performance Requirements**

* Load product catalog within 2 seconds.
* System should handle up to 2000 concurrent sessions.
* Order confirmation and payment processes should complete within 10 seconds under standard load.

### **3.4 Design Constraints**

* Follow MVC (Model-View-Controller) architecture.
* Compliance with PCI-DSS (Payment Card Industry Data Security Standard) guidelines for handling payment information.
* All images must be optimized for fast loading.

### **3.5 Software System Attributes**

#### **3.5.1 Reliability**

* System should be available 99.9% of the time.
* Automatic data backup every 24 hours.

#### **3.5.2 Availability**

* Website must be accessible 24/7 except during scheduled maintenance (not exceeding 2 hours/month).

#### **3.5.3 Security**

* Passwords stored with hashing and salting.
* Two-factor authentication (future enhancement).
* All payment data processed through secure, compliant gateways.

#### **3.5.4 Maintainability**

* Code must be modular for easy maintenance and upgrades.
* Proper error logging and monitoring must be implemented.

#### **3.5.5 Portability**

* Application should run on Windows, Linux, and MacOS hosting servers.
* Frontend must support cross-browser compatibility.

## **4. Supporting Information**

* User Journey Diagrams (Login → Catalog → Cart → Checkout → Order).
* Entity Relationship Diagram (ERD) for Database.
* Sample SQL scripts for database setup.
* List of APIs used (payment gateway, email services).