# SRS FOR DBMS MINIPROJECT

Pet Shop Management System

460: Maitreyee Shendye

468: Tanmayi Kulkarni

472: Yuga Tungar

2459: Kanchan Fadol

Batch: A3

## 1.Introduction

### 1.1 Purpose

The purpose of this software system is to provide a **Pet Shop Management System** that enables efficient management of pets, products, sales, and customer data for a pet shop business. The system will be accessible from both **administrator** and **customer** perspectives.

**Administrator Perspective**: Allows management of pet inventory, product catalog, sales records, employee data, and customer information.

Customer Perspective: Allows customers to view available pets, products, check prices, and place orders.

### 1.2 Scope

This system provides functionalities to:

- Maintain pet inventory details (species, breeds, etc.).

- Manage product catalog (pet food, accessories).

- Handle customer details (registration, orders, etc.).

- Track sales and generate reports.

- Provide a user-friendly interface for customers to browse available pets and products.

The system will be developed using JDBC for database connectivity, and the database will be hosted on a relational database management system like MySQL or PostgreSQL.

### 1.3 Definitions, Acronyms, and Abbreviations

JDBC: Java Database Connectivity

SQL: Structured Query Language

### 1.4 References

- Database schema

- JDBC documentation: [JDBC API Documentation](https://docs.oracle.com/javase/8/docs/api/java/sql/package-summary.html)

- SQL Language Reference: [MySQL Documentation](https://dev.mysql.com/doc/)

### 1.5 Overview

The document is organized as follows:

- Section 2 outlines the system overview and high-level architecture.

- Section 3 details functional requirements.

- Section 4 specifies non-functional requirements.

- Section 5 describes the system design and architecture.

- Section 6 includes external interface requirements.

## 2. System Overview

### 2.1 System Architecture

The Pet Shop Management System is a 2-tier application, consisting of:

Client Tier (java console): This tier is responsible for interacting with users (administrators and customers). It communicates with the Server Tier using JDBC for database operations.

Server Tier (Backend/Database): This tier handles database operations such as storing pet, product, customer, and sales data.

The client application will connect to the backend via JDBC and perform operations like adding pets, querying available pets/products, and processing sales.

### 2.2 Assumptions and Dependencies

- The system relies on the availability of a stable network connection for database connectivity.

- The database is structured to accommodate necessary tables like pets, customers, orders, sales, products, etc.

## 3. Functional Requirements

### 3.1 Administrator Functionalities

Pet Management:

- Add new pet details (species, breed, age, price, etc.).
- Update pet details.
- Remove pets from the inventory.
- View list of all pets with filtering options based on species or breed.

- Product Management:

- Add new product details (e.g., pet food, accessories).
- Update product details.
- Remove products.
- View product catalog and search by category (food, accessories, etc.).

- Customer Management:

- Add new customer records (name, contact info, etc.).
- Update customer details.
- Remove customer records.
- View customer order history.

-Order Management:

- Process customer orders.
- Generate reports on total sales.

### 3.2 Customer Functionalities

- View Available Pets:

- Browse available pets for sale.
- Filter pets based on age and  breed,

- View Products:

- Browse and search for pet products (food, toys, accessories).

- Check product details such as price

- Place Orders:

  - Customers can place orders for pets or products.

  - Checkout process with order summary and payment details.

- Order History:

  - View past orders and track order status.

4. Non-Functional Requirements

**4.1 Performance**

- The system should be able to handle both admin and customer

- Each query should return results within 3 seconds.

4.2 **Security**

- User authentication for both administrator and customer.

- Data encryption for sensitive information such as customer details and payment information.

- Ensure role-based access control: administrators have access to all features, while customers can only view and order pets/products.

4.3 **Usability**

- The system should provide an intuitive and easy-to-use interface for both administrators and customers.

- The customer interface should be simple, allowing users to search for pets, products, and make orders without unnecessary complexity.

4.4 **Maintainability**

- The system should be easy to maintain, with clear documentation and well-structured code.

- Database queries, stored procedures, triggers, and functions should be reusable and modular.

4.5 **Scalability**

- The system should be able to scale horizontally or vertically to support additional pets, products, and users.

## 5. System Design and Architecture

### 5.1 Database Design

- The database will have the following core tables:

**Admin** ( id,name,address,email_id,mobile_number)
**Customer** (customer_id , customer_name,address,email_address, mobile_number)
**Product** (product_id,product_name,type,used_by_pet,cost,quantity)
**Pet** (pet_id,pet_name,pet_type,date_of_rescue,breed)
**Orders** (order_number,order_date, quantity,product_id, customer_id);
**Adopts** (pet_id,customer_id,date_of_adoption)
**Got_adopted** (pet_id,pet_name,pet_type,breed, date_of_rescue)

### 5.2 Key Database Queries

- JOIN: To join `Pets`, `Products`, `Orders` and 'adopts' to generate reports.

- BETWEEN: To filter data within a range- like filter products by range, pet between a particular age

- LIKE: To search for products used by a particular pet type, using type of product or pet product used by, to see pets adopted on a particular date and rescued on a date.

- GROUP BY: To group data from products table having same type or used by which pet.

- ORDER BY: To sort data adoption date and product cost

**- Views: For creating views like "Daily Sales Report" or "Pet Availability Report."**

**- Indexes: To speed up queries, especially on frequently queried columns like `pet_id`, `product_id`, `order_id`.**

**- Triggers: To automatically update inventory when an order is placed.**

**- Stored Procedures/Functions: To encapsulate business logic (e.g., calculating discounts, calculating total sales).**

### 5.3 JDBC Integration

- The application will connect to the database using JDBC.

- Prepared statements will be used for SQL queries to prevent SQL injection.

- Functions and stored procedures will be invoked using `CallableStatement`.

## 6. External Interface Requirements

### 6.1 User Interfaces

Used jar file to connect jdbc and sql

#### 7. **Appendices**

7.1 **Glossary**

- **Pet**: An animal available for sale in the pet shop.

- **Order**: A transaction made by the customer to purchase pets or products.

- **SKU (Stock Keeping Unit)**: A unique identifier for a product in the system.

7.2 **Revision History**

| Version | Date | Description | Author |
|---|---|---|---|
| 1.0 | 2024-11-12 | Initial document draft | [Your Name] |

---

Conclusion

This **SRS** provides a comprehensive overview of the Pet Shop Management System, outlining both functional and non-functional requirements, system architecture, and external interfaces. The

system is designed to handle pet and product management, customer interactions, and sales tracking efficiently, ensuring a seamless experience for both administrators and customers.