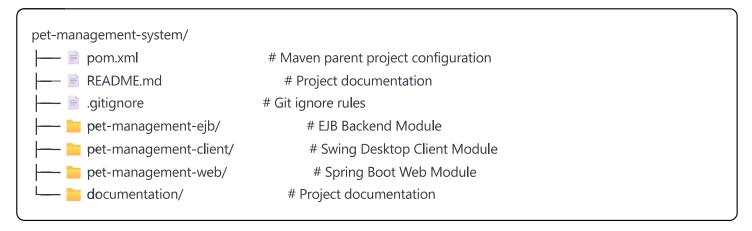
## **Pet Management System - Project Structure Layout**

#### Root Project Structure

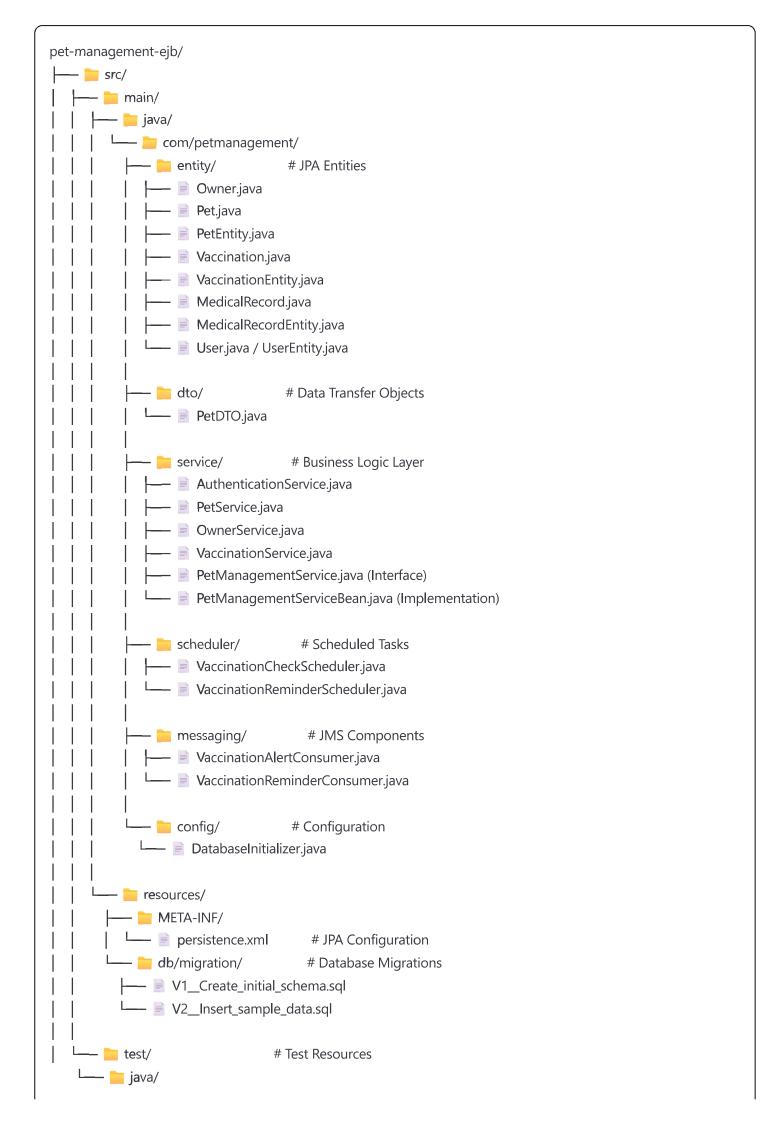


## Module 1: EJB Backend (pet-management-ejb/)

#### **Maven Configuration**

pet-management-ejb/		
pom.xml	# EJB module Maven configuration	

#### Source Structure

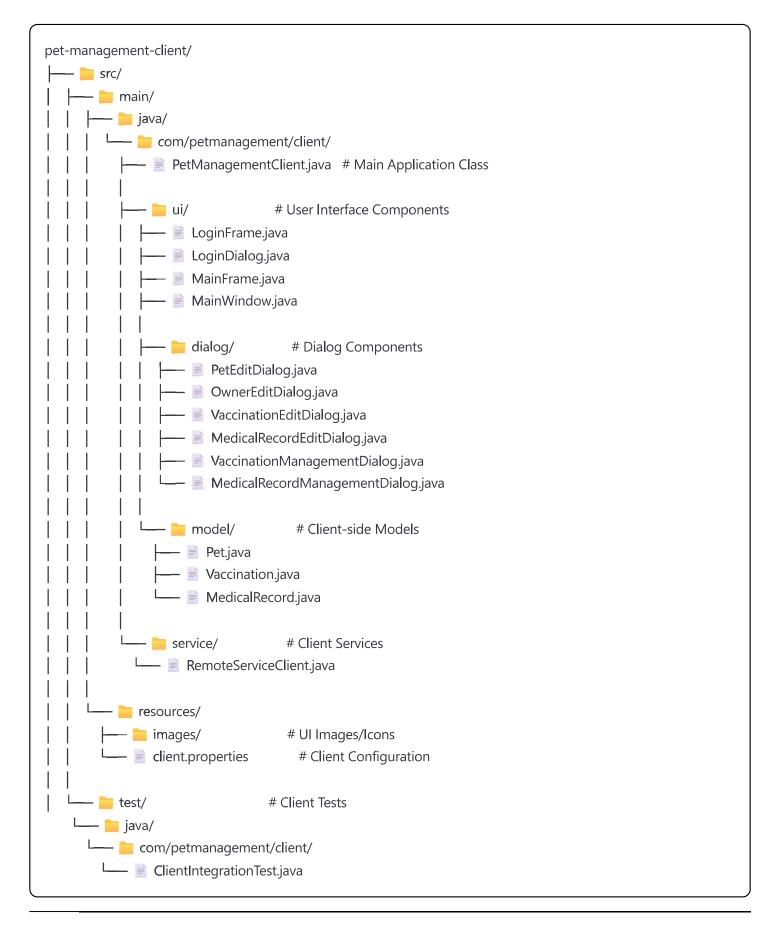




# Module 2: Desktop Client (pet-management-client/)

Maven Configuration

**Source Structure** 



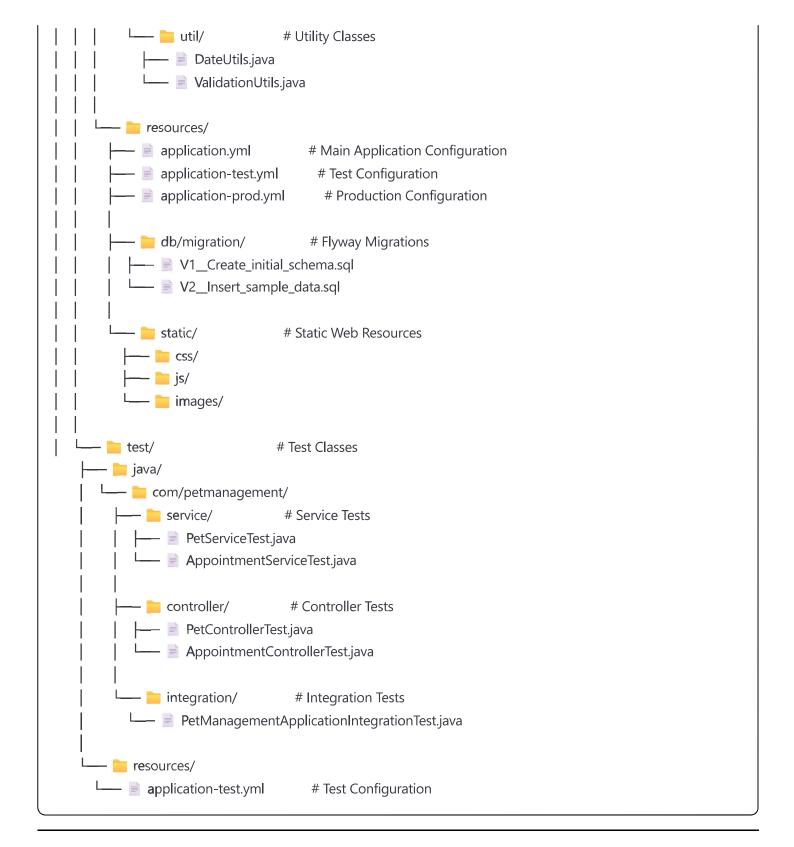
# Module 3: Spring Boot Web (pet-management-web/)

## Maven Configuration









#### Documentation Structure

documentation/		
├── 📄 API_Documentation.md	# REST API Documentation	
├── 📄 Database_Schema.md	# Database Design	
- Architecture_Overview.md	# System Architecture	
├── 📄 Deployment_Guide.md	# Deployment Instructions	
├—— 📄 User_Manual.md	# End User Guide	
🗀 diagrams/	# System Diagrams	
Component_diagram.png		
└── 🚞 api/ #	Generated API Docs	
느 📄 swagger.json		

# Configuration Files Summary

## **Maven Configuration Files**

File	Purpose	Location
pom.xml	Parent project configuration	Root directory
pet-management-ejb/pom.xml	EJB module dependencies	EJB module
pet-management-client/pom.xml	Swing client dependencies	Client module
pet-management-web/pom.xml	Spring Boot dependencies	Web module
•	·	•

## **Application Configuration Files**

File	Purpose	Module
persistence.xml	JPA configuration	EJB
application.yml	Spring Boot main config	Web
application-test.yml	Test configuration	Web
(application-prod.yml	Production configuration	Web
<b>▲</b>	·	•

## **Database Migration Files**

File	Purpose	Location
V1_Create_initial_schema.sql	Initial database schema	Both EJB & Web modules
V2_Insert_sample_data.sql	Sample data insertion	Both EJB & Web modules
4		

# **m** Architecture Layers

## 1. Presentation Layer

- **Desktop Client**: Swing-based GUI (pet-management-client/)
- Web API: REST Controllers (pet-management-web/controller/)

#### 2. Business Logic Layer

- **EJB Services**: Enterprise Java Beans (pet-management-ejb/service/)
- Spring Services: Spring Boot services (pet-management-web/service/)

#### 3. Data Access Layer

- EJB Entities: JPA entities with EJB (pet-management-ejb/entity/)
- Spring Repositories: Spring Data JPA (pet-management-web/repository/)

#### 4. Integration Layer

- JMS Messaging: Event-driven communication
- Scheduled Tasks: Automated processes
- **REST APIs**: External system integration

#### 😉 Component Relationships

```
mermaid

graph TB

A[Desktop Client] --> B[EJB Backend]

C[Web Client] --> D[Spring Boot Backend]

B --> E[Database]

D --> E[Database]

B --> F[JMS Queue]

D --> F[JMS Queue]

G[Scheduler] --> F

H[Message Consumers] --> F
```

#### 🚀 Build & Deployment Order

- 1. **Build Parent Project**: (mvn clean install) (from root)
- 2. **Build EJB Module**: mvn clean package (from ejb directory)
- 3. **Build Client Module**: mvn clean package (from client directory)
- 4. **Build Web Module**: (mvn clean package) (from web directory)
- 5. **Deploy EJB**: Deploy to application server
- 6. **Run Client**: Execute Swing application
- 7. Run Web: Start Spring Boot application

This structure provides a clear separation of concerns, maintains modularity, and follows enterprise Java development best practices. Each module can be developed, tested, and deployed independently while maintaining clean interfaces between them.