Pet Management System - Project Structure Layout

📁 Root Project Structure

```
pet-management-system/
├── 📄 pom.xml                    # Maven parent project configuration
├── 📄 README.md                  # Project documentation
├── 📄 .gitignore                 # Git ignore rules
├── 📁 pet-management-ejb/        # EJB Backend Module
├── 📁 pet-management-client/     # Swing Desktop Client Module
├── 📁 pet-management-web/        # Spring Boot Web Module
└── 📁 documentation/             # Project documentation
```

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

📦 Maven Configuration

```
pet-management-ejb/
├── 📄 pom.xml                    # EJB module Maven configuration
```

🗂️ Source Structure

```
pet-management-ejb/
├── 📁 src/
│   ├── 📁 main/
│   │   ├── 📁 java/
│   │   │   └── 📁 com/petmanagement/
│   │   │       ├── 📁 entity/              # JPA Entities
│   │   │       │   ├── 📄 Owner.java
│   │   │       │   ├── 📄 Pet.java
│   │   │       │   ├── 📄 PetEntity.java
│   │   │       │   ├── 📄 Vaccination.java
│   │   │       │   ├── 📄 VaccinationEntity.java
│   │   │       │   ├── 📄 MedicalRecord.java
│   │   │       │   ├── 📄 MedicalRecordEntity.java
│   │   │       │   └── 📄 User.java / UserEntity.java
│   │   │       │
│   │   │       ├── 📁 dto/                 # Data Transfer Objects
│   │   │       │   └── 📄 PetDTO.java
│   │   │       │
│   │   │       ├── 📁 service/             # Business Logic Layer
│   │   │       │   ├── 📄 AuthenticationService.java
│   │   │       │   ├── 📄 PetService.java
│   │   │       │   ├── 📄 OwnerService.java
│   │   │       │   ├── 📄 VaccinationService.java
│   │   │       │   ├── 📄 PetManagementService.java (Interface)
│   │   │       │   └── 📄 PetManagementServiceBean.java (Implementation)
│   │   │       │
│   │   │       ├── 📁 scheduler/           # Scheduled Tasks
│   │   │       │   ├── 📄 VaccinationCheckScheduler.java
│   │   │       │   └── 📄 VaccinationReminderScheduler.java
│   │   │       │
│   │   │       ├── 📁 messaging/           # JMS Components
│   │   │       │   ├── 📄 VaccinationAlertConsumer.java
│   │   │       │   └── 📄 VaccinationReminderConsumer.java
│   │   │       │
│   │   │       └── 📁 config/              # Configuration
│   │   │          └── 📄 DatabaseInitializer.java
│   │   │
│   │   └── 📁 resources/
│   │       ├── 📁 META-INF/
│   │       │   └── 📄 persistence.xml       # JPA Configuration
│   │       └── 📁 db/migration/            # Database Migrations
│   │          ├── 📄 V1__Create_initial_schema.sql
│   │          └── 📄 V2__Insert_sample_data.sql
│   │
│   └── 📁 test/                            # Test Resources
│      └── 📁 java/
```

```
    └── 📁 com/petmanagement/
        └── 📁 service/
            ├── 📄 PetServiceTest.java
            └── 📄 AuthenticationServiceTest.java
```

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

📦 Maven Configuration

```
pet-management-client/
├── 📄 pom.xml                    # Swing client Maven configuration
```

🗂️ Source Structure

```
pet-management-client/
├── 📁 src/
│   ├── 📁 main/
│   │   ├── 📁 java/
│   │   │   └── 📁 com/petmanagement/client/
│   │   │       ├── 📄 PetManagementClient.java   # Main Application Class
│   │   │       │
│   │   │       ├── 📁 ui/                   # User Interface Components
│   │   │       │   ├── 📄 LoginFrame.java
│   │   │       │   ├── 📄 LoginDialog.java
│   │   │       │   ├── 📄 MainFrame.java
│   │   │       │   ├── 📄 MainWindow.java
│   │   │       │   │
│   │   │       │   ├── 📁 dialog/              # Dialog Components
│   │   │       │   │   ├── 📄 PetEditDialog.java
│   │   │       │   │   ├── 📄 OwnerEditDialog.java
│   │   │       │   │   ├── 📄 VaccinationEditDialog.java
│   │   │       │   │   ├── 📄 MedicalRecordEditDialog.java
│   │   │       │   │   ├── 📄 VaccinationManagementDialog.java
│   │   │       │   │   └── 📄 MedicalRecordManagementDialog.java
│   │   │       │   │
│   │   │       │   └── 📁 model/              # Client-side Models
│   │   │       │       ├── 📄 Pet.java
│   │   │       │       ├── 📄 Vaccination.java
│   │   │       │       └── 📄 MedicalRecord.java
│   │   │       │
│   │   │       └── 📁 service/              # Client Services
│   │   │           └── 📄 RemoteServiceClient.java
│   │   │
│   │   └── 📁 resources/
│   │       ├── 📁 images/               # UI Images/Icons
│   │       └── 📄 client.properties          # Client Configuration
│   │
│   └── 📁 test/                        # Client Tests
│       └── 📁 java/
│           └── 📁 com/petmanagement/client/
│               └── 📄 ClientIntegrationTest.java
```

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

📦 Maven Configuration

```
pet-management-web/
├── 📄 pom.xml                      # Spring Boot Maven configuration
```

🗂️ Source Structure

```
pet-management-web/
├── 📁 src/
│   ├── 📁 main/
│   │   ├── 📁 java/
│   │   │   └── 📁 com/petmanagement/
│   │   │       ├── 📄 PetManagementApplication.java # Spring Boot Main Class
│   │   │       │
│   │   │       ├── 📁 entity/            # JPA Entities
│   │   │       │   ├── 📄 Pet.java
│   │   │       │   ├── 📄 Appointment.java
│   │   │       │   ├── 📄 Owner.java
│   │   │       │   └── 📄 User.java
│   │   │       │
│   │   │       ├── 📁 dto/               # Data Transfer Objects
│   │   │       │   ├── 📄 PetDTO.java
│   │   │       │   └── 📄 AppointmentDTO.java
│   │   │       │
│   │   │       ├── 📁 repository/        # Data Access Layer
│   │   │       │   ├── 📄 PetRepository.java
│   │   │       │   ├── 📄 AppointmentRepository.java
│   │   │       │   └── 📄 OwnerRepository.java
│   │   │       │
│   │   │       ├── 📁 service/           # Business Logic Layer
│   │   │       │   ├── 📄 PetService.java
│   │   │       │   ├── 📄 AppointmentService.java
│   │   │       │   ├── 📄 ScheduledService.java
│   │   │       │   └── 📄 JmsService.java
│   │   │       │
│   │   │       ├── 📁 controller/        # REST API Controllers
│   │   │       │   ├── 📄 PetController.java
│   │   │       │   └── 📄 AppointmentController.java
│   │   │       │
│   │   │       ├── 📁 config/            # Configuration Classes
│   │   │       │   ├── 📄 JmsConfig.java
│   │   │       │   ├── 📄 SwaggerConfig.java
│   │   │       │   ├── 📄 SecurityConfig.java
│   │   │       │   └── 📄 CacheConfig.java
│   │   │       │
│   │   │       ├── 📁 messaging/         # JMS Components
│   │   │       │   └── 📄 JmsMessageListener.java
│   │   │       │
│   │   │       ├── 📁 exception/         # Exception Handling
│   │   │       │   ├── 📄 ResourceNotFoundException.java
│   │   │       │   ├── 📄 ValidationException.java
│   │   │       │   └── 📄 GlobalExceptionHandler.java
│   │   │       │
│   │   │       └── 📁 util/                  # Utility Classes
│   │   │           ├── 📄 DateUtils.java
│   │   │           └── 📄 ValidationUtils.java
│   │   │
│   │   └── 📁 resources/
│   │       ├── 📄 application.yml          # Main Application Configuration
│   │       ├── 📄 application-test.yml     # Test Configuration
│   │       ├── 📄 application-prod.yml     # Production Configuration
│   │       │
│   │       ├── 📁 db/migration/            # Flyway Migrations
│   │       │   ├── 📄 V1__Create_initial_schema.sql
│   │       │   └── 📄 V2__Insert_sample_data.sql
│   │       │
│   │       └── 📁 static/                  # Static Web Resources
│   │           ├── 📁 css/
│   │           ├── 📁 js/
│   │           └── 📁 images/
│   │
│   └── 📁 test/                            # Test Classes
    ├── 📁 java/
    │   └── 📁 com/petmanagement/
    │       ├── 📁 service/                 # Service Tests
    │       │   ├── 📄 PetServiceTest.java
    │       │   └── 📄 AppointmentServiceTest.java
    │       │
    │       ├── 📁 controller/              # Controller Tests
    │       │   ├── 📄 PetControllerTest.java
    │       │   └── 📄 AppointmentControllerTest.java
    │       │
    │       └── 📁 integration/             # Integration Tests
    │           └── 📄 PetManagementApplicationIntegrationTest.java
    │
    └── 📁 resources/
        └── 📄 application-test.yml         # Test 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
│   ├── 📄 system_architecture.png
│   ├── 📄 database_erd.png
│   └── 📄 component_diagram.png
└── 📁 api/                          # Generated API Docs
    └── 📄 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

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

🏛️ 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.