Spring Boot Folder Structure (Best Practices)

Learn the best Spring Boot project structure for clean, scalable, and production-ready applications. Organize your codebase with real-world folder structure examples.

Munaf Badarpura

July 05, 2025 • 4 min read

When new spring boot developers start working with Spring Boot Project, they didn’t really care much about project structure.

It was one of those things where they just wanted the code to work but as projects grow and more classes and files created, that messy structure comes back to haunt you. so it feels frustrating and adding new features felt chaotic. so, a clean and well-organized folder structure isn't just theoretical best practice it actually makes your life and your team's life way easier.

In this blog, I'm sharing some Spring Boot folder structure best practices that will help you keep your codebase clean, maintainable, and easy to scale.

But first… why should you even care about folder structure?

Here's what I have learned the hard way:

1. Clarity & Organization

2. Scalability

3. Code Reusability

4. Easier Testing & Debugging

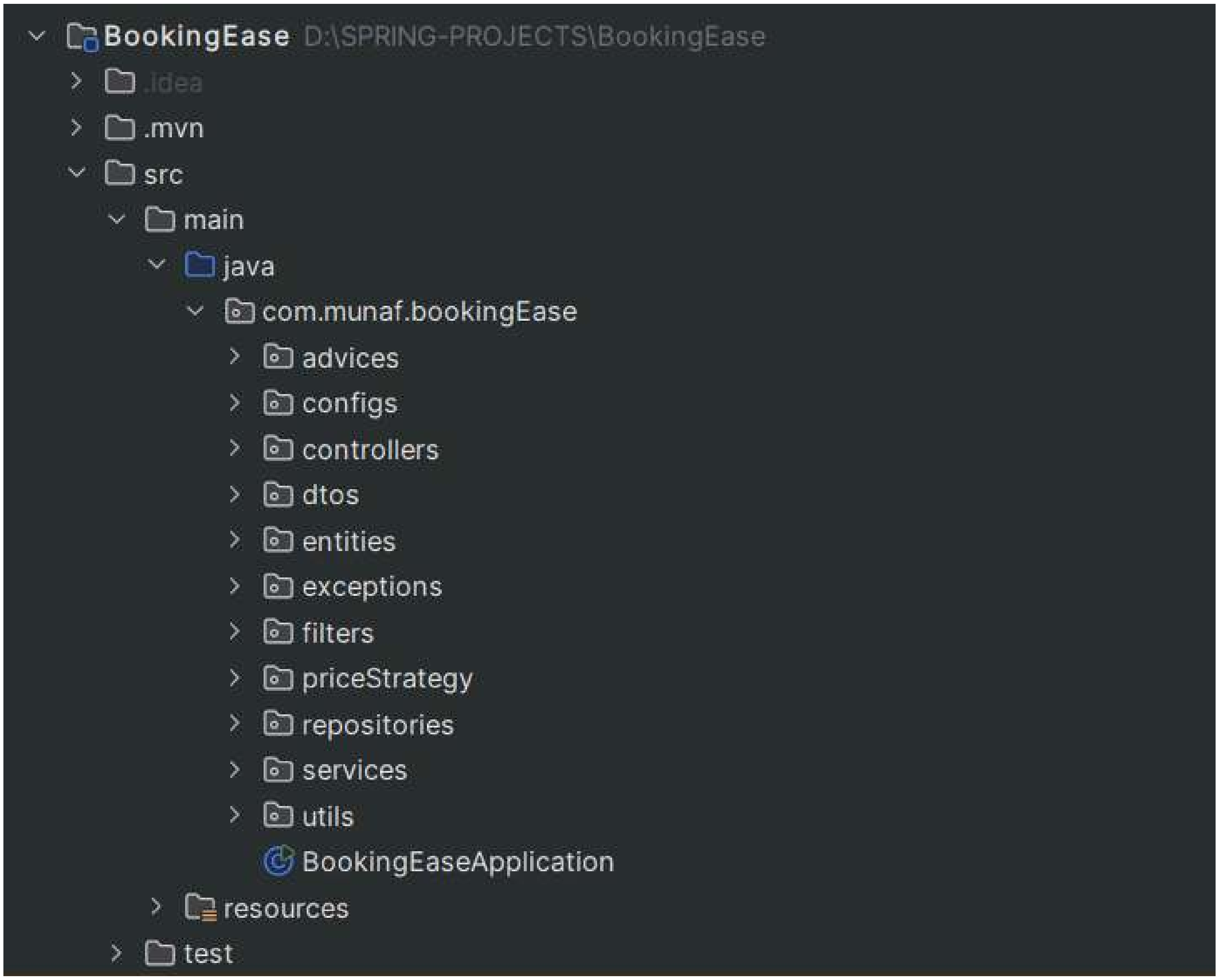
5. Friendly for New Team Members

A good folder structure should help you clearly organize so your team can find files faster. It also promotes testing because you can quickly find the files you need to test or debug, and it

[illegible]
do I really need all of these?"

Trust me, I've been there*.* But once your project starts growing, that structure saves you from chaos. So, here's a breakdown of the well-recognized folder structure for Spring Boot apps, along with why each folder exists

Following image shows the well recognized folder structure for Spring Boot application :



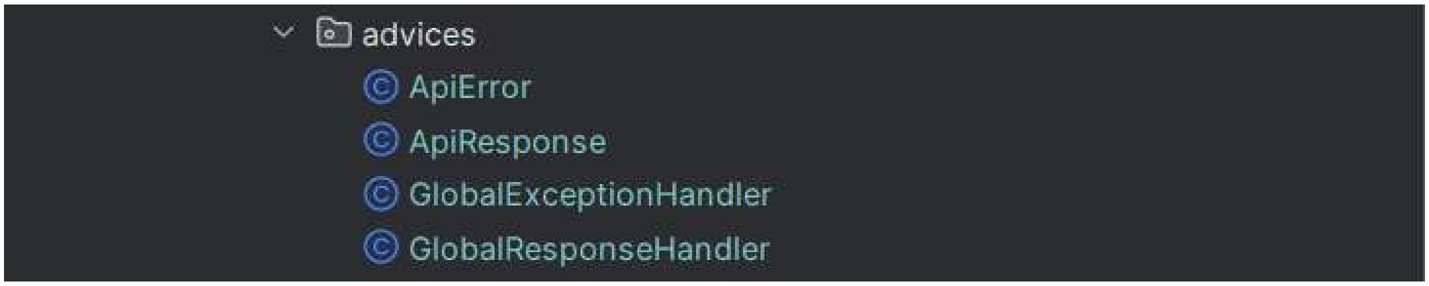
```
BookingEase D:\SPRING-PROJECTS\BookingEase
  .idea
  .mvn
  src
    main
      java
        com.munaf.bookingEase
          advices
          configs
          controllers
          dtos
          entities
          exceptions
          filters
          priceStrategy
          repositories
          services
          utils
          BookingEaseApplication
      resources
    test
```

Spring Boot Project Structure

Let's understand each folder in great detail :

1. advices.

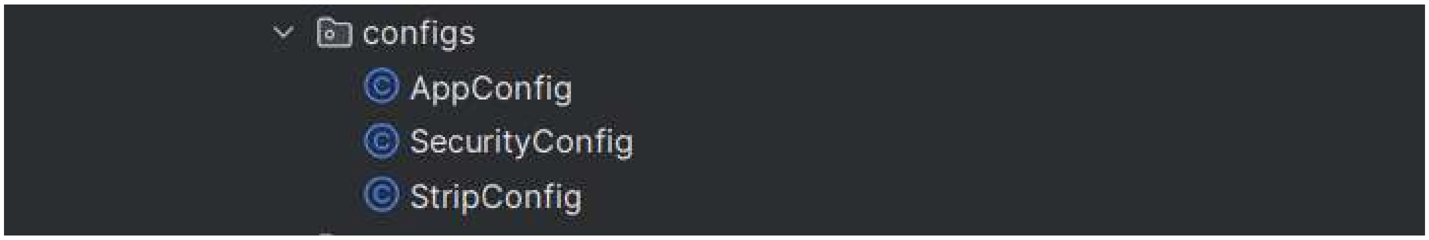
It holds classes like GlobalExceptionHandler to make sure your app returns clean, consistent error responses — no more random stack traces leaking to the client. You can also throw in response wrapper logic here if you want to standardize API responses.



Advices Package

2. configs

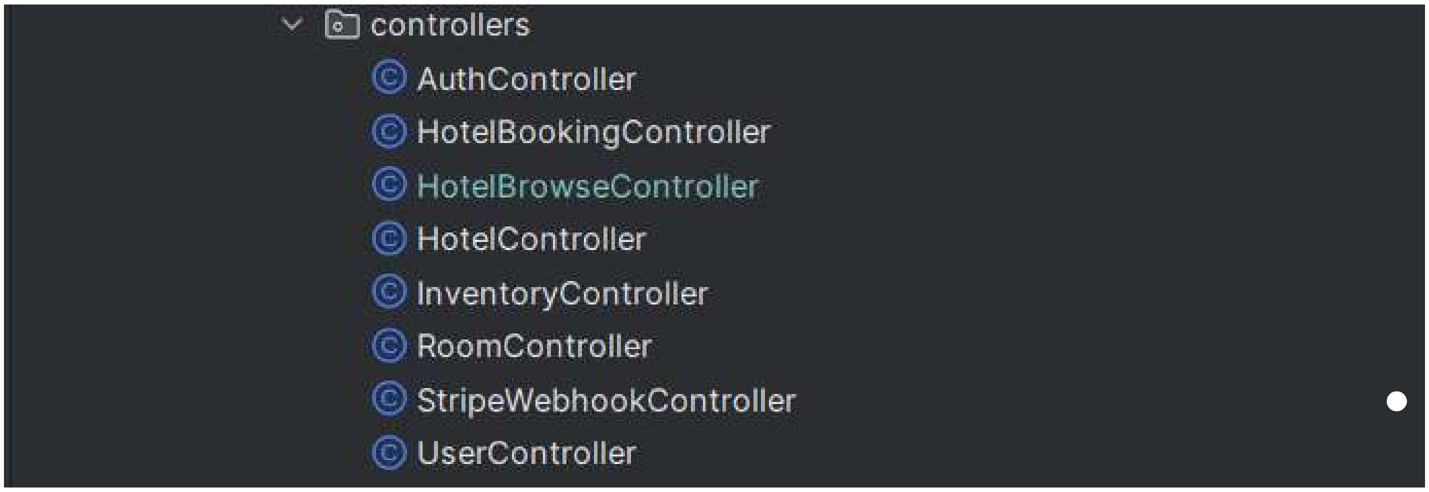
This is where your app-level configurations live. Whether it's setting up beans, defining global settings, or other custom configs — they all go here in neat, predictable files like AppConfig.



Configs Package

3. controllers

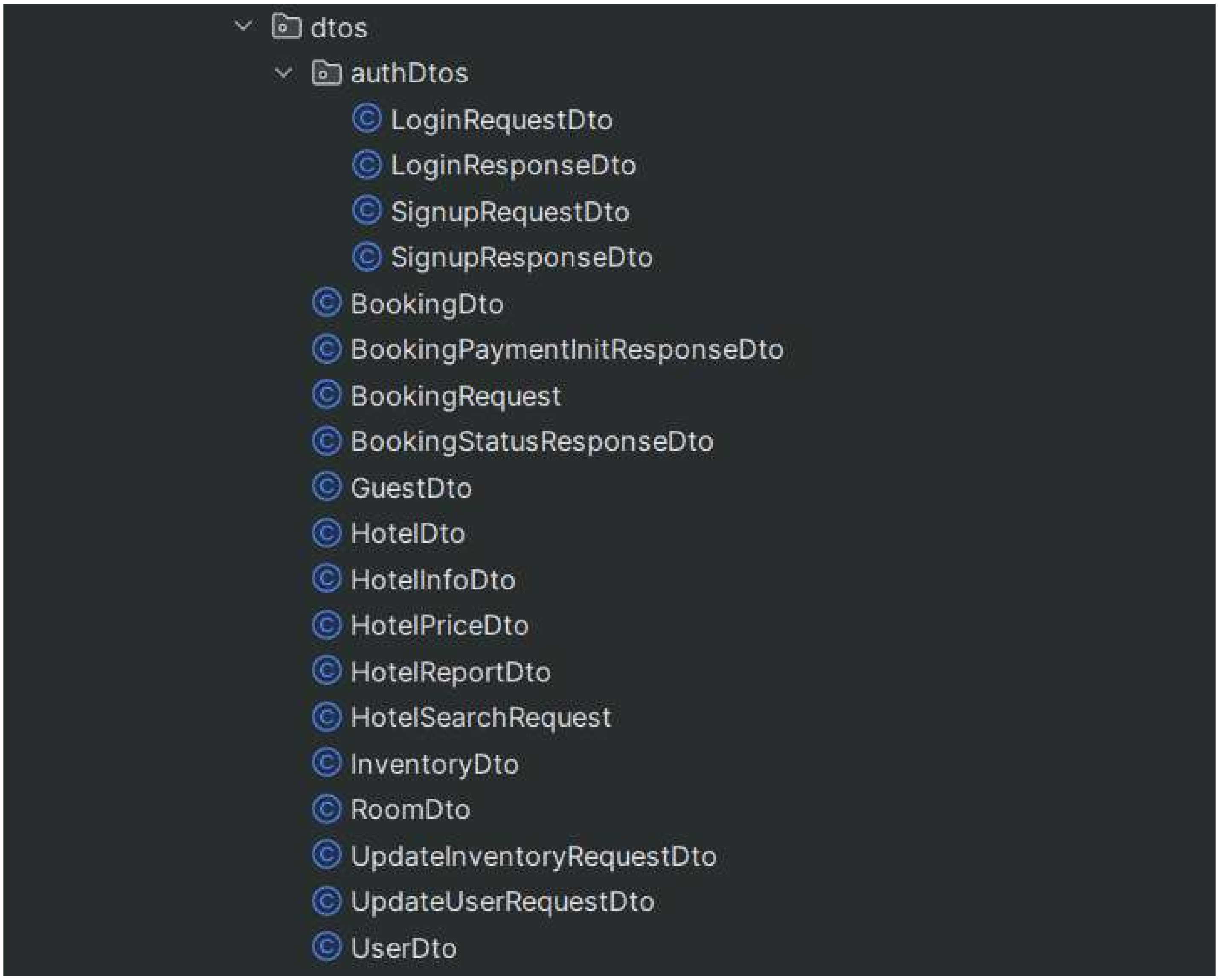
Your entry point to the outside world. This folder holds your REST API endpoints, handling all incoming HTTP requests and routing them to the right services. Basically, it connects your app to users or clients.



Controller Package

4. dtos

DTOs (Data Transfer Objects) are your data messengers. They carry structured data between different layers or between front-end and back-end. Keeping them in one place keeps your project clean and avoids mixing internal models with external-facing data.

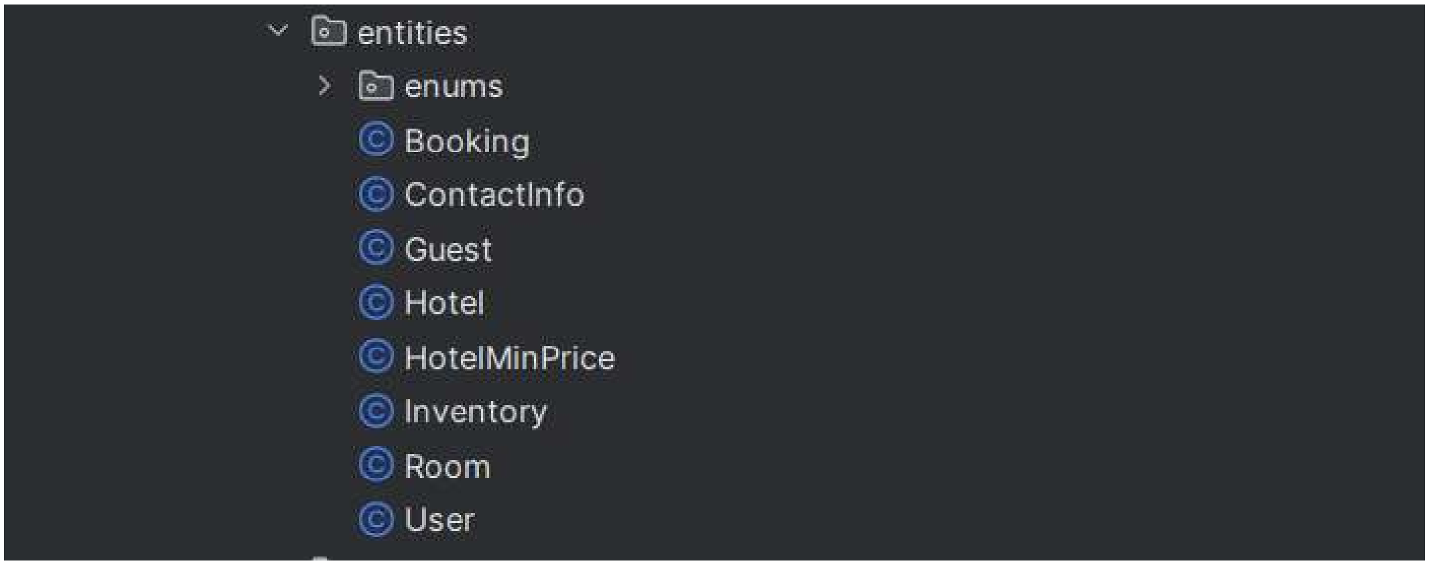


```
dtos
  authDtos
    LoginRequestDto
    LoginResponseDto
    SignupRequestDto
    SignupResponseDto
  BookingDto
  BookingPaymentInitResponseDto
  BookingRequest
  BookingStatusResponseDto
  GuestDto
  HotelDto
  HotelInfoDto
  HotelPriceDto
  HotelReportDto
  HotelSearchRequest
  InventoryDto
  RoomDto
  UpdateInventoryRequestDto
  UpdateUserRequestDto
  UserDto
```

Dtos Package

5. entities

Your database blueprints live here. These classes represent your tables using JPA annotations, defining relationships, fields, and even enums when needed. They're the backbone of your app's data structure.

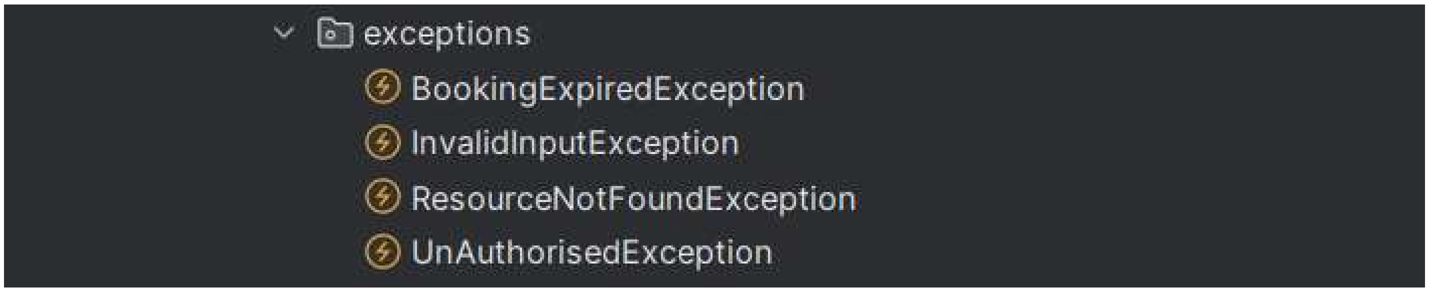


```
entities
  enums
  Booking
  ContactInfo
  Guest
  Hotel
  HotelMinPrice
  Inventory
  Room
  User
```

Entities Package

6. exceptions

You will have errors so it's better to handle them properly. This folder contains custom exception classes to manage specific error situations in a clean, predictable way.

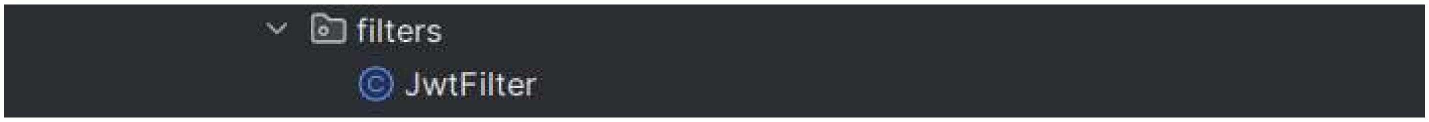


```
exceptions
  BookingExpiredException
  InvalidInputException
  ResourceNotFoundException
  UnAuthorisedException
```

Exceptions Package

7. filters

Filters are used to intercept incoming HTTP requests and outgoing responses, allowing you to perform operations like logging, authentication checks, or modifying requests before they reach controllers. The filters folder contains all the custom filters required in the application like JwtFilter and so on.

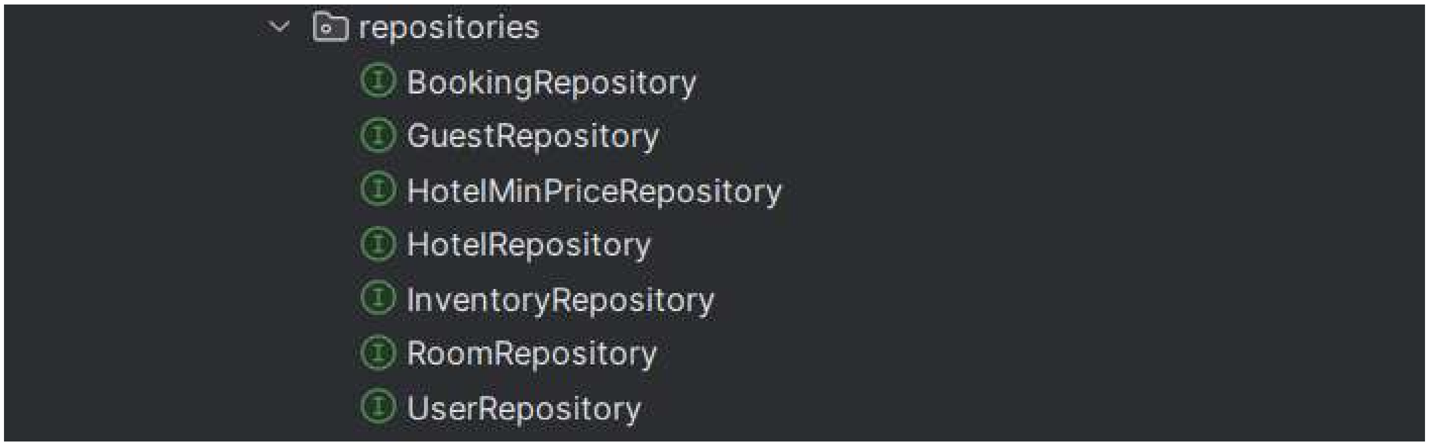


```
filters
    JwtFilter
```

Filters Package

8. repositories

The repositories folder contains interfaces that are responsible for data access operations and interacting with the database. These interfaces typically extend Spring Data JPA repositories like JpaRepository that provide built-in CRUD functionality and allow custom query definitions.

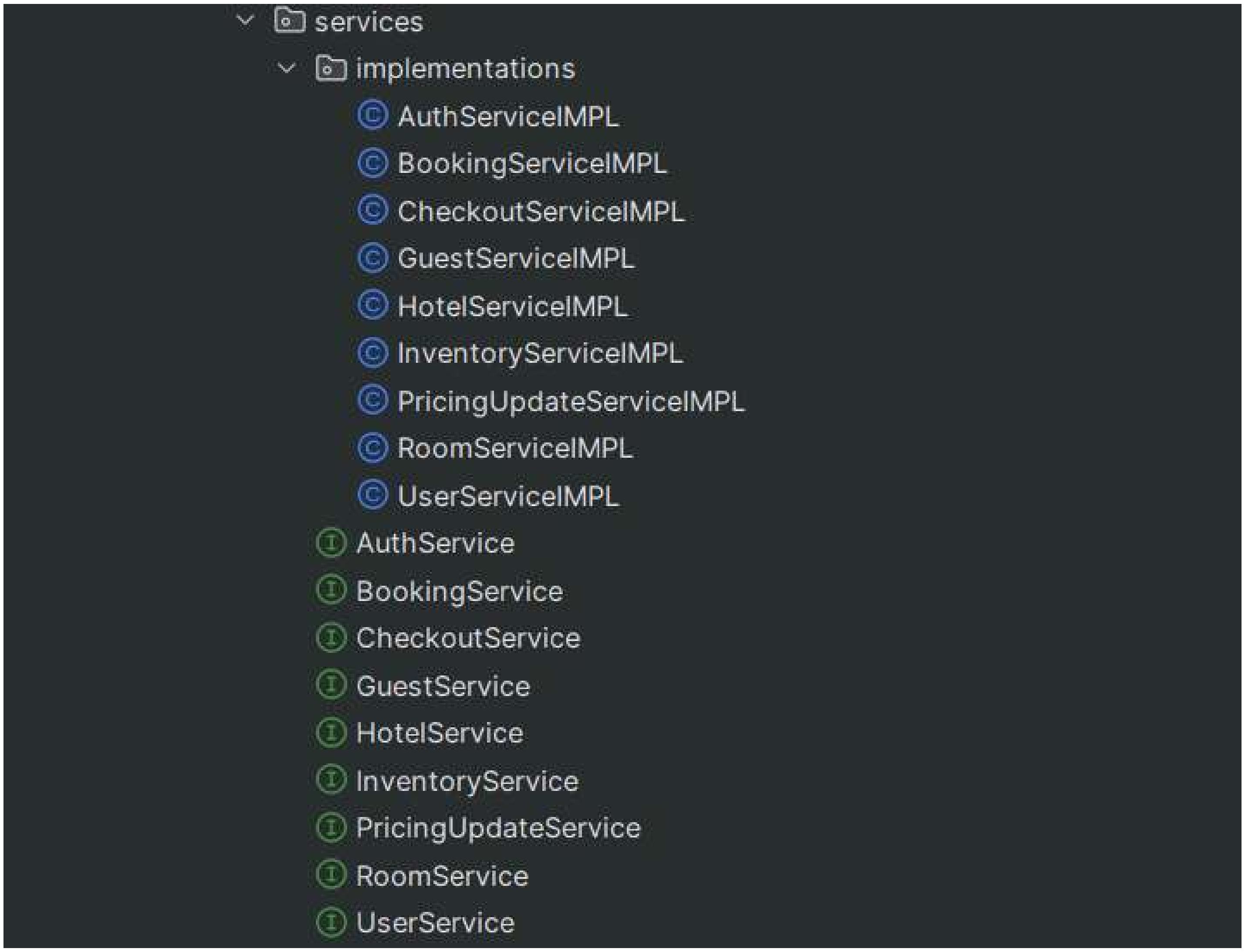


```
repositories
    BookingRepository
    GuestRepository
    HotelMinPriceRepository
    HotelRepository
    InventoryRepository
    RoomRepository
    UserRepository
```

Repositories Package

9. services

Where your business logic lives. Service interfaces define the logic, while their implementations (usually inside an implementations subfolder) contain the actual code. Keeping interfaces separate from implementations makes your app easier to test, extend, and refactor later.

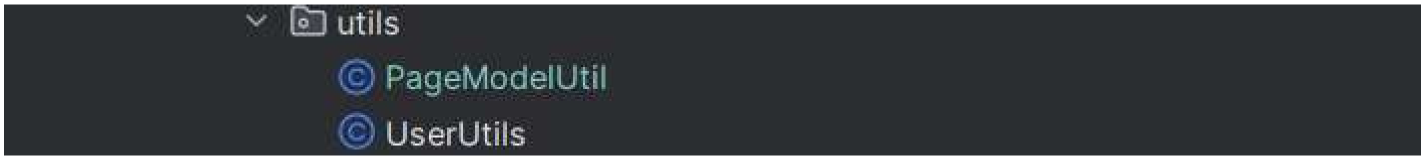


```
services
  implementations
    AuthServiceIMPL
    BookingServiceIMPL
    CheckoutServiceIMPL
    GuestServiceIMPL
    HotelServiceIMPL
    InventoryServiceIMPL
    PricingUpdateServiceIMPL
    RoomServiceIMPL
    UserServiceIMPL
  AuthService
  BookingService
  CheckoutService
  GuestService
  HotelService
  InventoryService
  PricingUpdateService
  RoomService
  UserService
```

Services Package

10. utils

- utils folder contains utility classes to keep the codebase organized and modular.



```
utils
  PageModelUtil
  UserUtils
```

Utils Package

11. resources

The heart of your config files. It stores things like **`application.properties`** or **`application.yml`** for environment-specific settings.

Resources Package

Conclusion:

By following a clean and consistent folder structure in your Spring Boot project, you make your codebase easy to navigate, maintainable, and ready to scale. It avoids chaos as your project grows, simplifies debugging.

> **Want to Master Spring Boot and Land Your Dream Job?**
>
> Struggling with coding interviews? Learn Data Structures & Algorithms (DSA) with our expert-led course. Build strong problem-solving skills, write optimized code, and crack top tech interviews with ease
>
> Learn more

Spring Boot | Spring | Spring Boot Folder Structure

MORE ARTICLES

12 Spring Boot REST API Best Practices to Build Scalable & Secure Applications

Munaf Badarpura • *Aug 23, 2025*

The Spring Boot HandBook
powered by CODING SHUTTLE
Best way to master Spring Boot complete roadmap [2025]
www.codingshuttle.com

Best Way To Master Spring Boot - A Complete Roadmap in 2025

Santosh Mane • *Aug 23, 2025*

SHUTTLE

COMPANY

About Us

COMPILERS

Java Compiler

MUST-READ BLOGS

Operating Systems In One Shot

#BetterEveryday

FOLLOW US ON

Terms & Conditions

Privacy Policy

Pricing & Refund Policy

Contact Us

Our Blogs

C++ Compiler

Python Compiler

Javascript Compiler

Go Compiler

C# Compiler

C Compiler

How to Master Spring Boot in 100 Days: Study Plan

Top 8 Design Patterns in Java Every Developer Should Know

Top 20 Spring Boot Interview Questions for 3 years of experience Professionals

Top 10 Java 8 Interview Questions

Want to read more? Click here