Spring Boot Project Structure

Consider the below project structure

```
marvellous/
├── src/
│   ├── main/
│   │   ├── java/
│   │   │   └── com/marvellous/
│   │   │       ├── MarvellousApplication.java
│   │   │       ├── controller/
│   │   │       ├── service/
│   │   │       ├── repository/
│   │   │       ├── model/
│   │   │       └── dto/
│   │   └── resources/
│   │       ├── application.properties
│   │       ├── static/
│   │       ├── templates/
│   │       └── banner.txt
│   └── test/
│       └── java/
│           └── com/marvellous/
│               └── MarvellousApplicationTests.java
├── target/
│   ├── marvellous-0.0.1-SNAPSHOT.jar
│   ├── classes/
│   ├── generated-sources/
│   ├── test-classes/
│   └── ...
├── .mvn/
├── mvnw
├── mvnw.cmd
├── pom.xml
└── README.md
```

Detailed Explanation of Each Folder and File

1. src/main/java/com/marvellous/

This is your main Java source code. Follows reverse domain naming.

MarvellousApplication.java

```
@SpringBootApplication

public class MarvellousApplication

{

   public static void main(String[] args)

      {

      SpringApplication.run(MarvellousApplication.class, args);

   }

}
```

controller/

- Contains REST API endpoint logic using @RestController.

service/

- Business logic using @Service.

repository/

- JPA interfaces (UserRepository, etc.) using @Repository.

model/

- POJOs with @Entity for DB table mapping.

dto/ (optional)

- For Data Transfer Objects (pure request/response structures).

2. src/main/resources/

Stores resources needed at runtime.

application.properties

- Main configuration file (e.g., DB connection, port).

static/

- Public assets like CSS, JS, images.

templates/

- For HTML templates using Thymeleaf or other engines.

banner.txt (optional)

- ASCII art displayed on app startup.

3. src/test/java/com/marvellous/

Unit and integration test classes using JUnit/Mockito.

MarvellousApplicationTests.java

- Sample test case with @SpringBootTest.

4. pom.xml

Maven configuration file – dependencies, plugins, etc.

```
<artifactId>marvellous</artifactId>

<dependencies>

    <!-- Example -->

    <dependency>

        <groupId>org.springframework.boot</groupId>

        <artifactId>spring-boot-starter-web</artifactId>

    </dependency>

</dependencies>
```

5. target/ – Generated Folder After mvn package

The target/ directory is automatically created when you run:

```
mvn clean package
```

Contents of target/:

File/Folder	Purpose
`marvellous-0.0.1-SNAPSHOT.jar`	Final **executable JAR** with embedded Tomcat and compiled code
`classes/`	Compiled `.class` files of your source Java code
`generated-sources/`	Auto-generated code by plugins (like MapStruct)
`test-classes/`	Compiled test classes
`maven-status/`	Metadata about compilation status
`surefire-reports/`	Test results generated during `mvn test`

This folder is used during deployment or to run the app:

java -jar target/marvellous-0.0.1-SNAPSHOT.jar

6. Other Files

File/Folders	Description
`.mvn/`, `mvnw`, `mvnw.cmd`	Maven wrapper, allows running Maven without local install
`README.md`	Documentation or project instructions
`.gitignore`	Ignore files in Git (like `/target`, `.class`, `.idea`)

Summary

Component	Keyword / Annotation	Purpose
Main Class	`@SpringBootApplication`	Entry point
Controller	`@RestController`	Handles HTTP Requests
Service	`@Service`	Business Logic
Repository	`@Repository`, `JpaRepository`	Data Access Layer
Model	`@Entity`	Maps to DB Table
Configuration	`application.properties`	Project Settings
Final Output	`target/marvellous-0.0.1-SNAPSHOT.jar`	Executable File