Converting to a Spring Boot Application with @SpringBootApplication Annotation

Spring Feature Description

Spring Boot simplifies the setup of Spring applications by providing auto-configuration and an embedded server. The @SpringBootApplication annotation combines @Configuration, @EnableAutoConfiguration, and @ComponentScan annotations, streamlining the configuration process.

Why Do We Need It in Flight Application?

By converting to a Spring Boot application, we can streamline configuration, simplify dependency management, and prepare the application for future enhancements, such as adding RESTful APIs or web interfaces.

Refactoring Application

We will:

- Add Spring Boot dependencies to the project.
- Annotate the main application class with @SpringBootApplication.
- Utilize Spring Boot's auto-configuration and dependency management.

The Updated Code

- **Modify FlightsApplication**

Change FlightsApplication to be a Spring Boot application:

```
package com.luxoft.flights;

import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.context.annotation.Bean;

@SpringBootApplication
public class FlightsApplication {

    public static void main(String[] args) {
        SpringApplication.run(FlightsApplication.class, args);
    }

    @Bean
```

```
    CommandLineRunner run(DataService dataService, CSVDataLoader dataLoader) {
        return args -> {
            dataLoader.loadData();
            System.out.println("Data loaded successfully.");
            // Existing code for menu options...
        };
    }
}
```

- **Remove AppConfig Class**

The @SpringBootApplication annotation includes component scanning, so the AppConfig class is no longer necessary.

Benefits of Using @SpringBootApplication

- **Simplified Configuration**: Reduces boilerplate code and simplifies setup.
- **Auto-Configuration**: Automatically configures components based on classpath settings.
- **Embedded Server**: Allows running the application standalone without external servers.
- **Ease of Deployment**: Simplifies packaging and deployment processes.

Further Advices

Explore Spring Boot starters to include additional functionalities and consider leveraging Spring Boot's testing features for unit and integration tests.