Exercise 9: Customizing Data Source Configuration

Spring Boot Auto-Configuration

Spring Boot provides auto-configuration for data sources. It scans for commonly used libraries (e.g., HikariCP) and attempts to configure a data source based on your dependencies.

Externalizing Configuration

It's recommended to externalize your database configuration using `application.properties`. Here's an example:

Properties

```
spring.datasource.url=jdbc:h2:mem:testdb  # H2 in-memory database
spring.datasource.driverClassName=org.h2.Driver
spring.datasource.username=sa
spring.datasource.password=password
```

Managing Multiple Data Sources

Spring Boot allows you to configure multiple data sources by specifying unique names in the `spring.datasource` properties:

Properties

```
spring.datasource.primary.url=jdbc:h2:mem:testdb1  # Primary data source
spring.datasource.primary.driverClassName=org.h2.Driver
spring.datasource.primary.username=sa
spring.datasource.primary.password=password

spring.datasource.secondary.url=jdbc:mysql://localhost:3306/mydatabase  #
Secondary data source (example)
spring.datasource.secondary.driverClassName=com.mysql.cj.jdbc.Driver
```

```
spring.datasource.secondary.username=user
spring.datasource.secondary.password=password
```

Accessing Data Sources

You can inject the desired `DataSource` bean based on its name:

Java

```
@Autowired
@Qualifier("primary")  // Specify the data source name
private DataSource primaryDataSource;

// Use primaryDataSource for database operations
```

Additional Considerations

- **Spring Data JPA Configuration:** If using multiple data sources, configure `@EnableJpaRepositories` with the appropriate `entityManagerFactory` and `transactionManager` beans.
- **DataSource Transaction Management:** Manage transactions appropriately for multi-data source scenarios.
- **Connection Pool Configuration:** You can customize connection pool properties for each data source (e.g., HikariCP configuration).

By understanding these concepts, you can customize your data source configuration and manage multiple data sources within your Spring Boot application, providing flexibility for complex data access needs.