Injecting different database using @Qualifier

To configure multiple datasources in a Spring Boot application using properties files (like `application.properties`), you need to:

1. Define the properties for each datasource.
2. Provide unique bean names for each datasource.
3. Configure your repositories or DAOs to use the appropriate datasource.

Here's how you can do it:

1. **Define Properties for Each Datasource**:

   In your `application.properties` file, define properties for each datasource. For example:

   ```properties
   # First Datasource
   datasource.first.url=jdbc:mysql://localhost:3306/first_db
   datasource.first.username=first_user
   datasource.first.password=first_password

   # Second Datasource
   datasource.second.url=jdbc:mysql://localhost:3306/second_db
   datasource.second.username=second_user
   datasource.second.password=second_password
   ```

2. **Configure Datasources in Spring Boot**:

   In your Spring Boot configuration class, configure the datasources using these properties. You can use the `@ConfigurationProperties` annotation to bind these properties to Java objects representing your datasources. Ensure that each datasource has a unique name.

   ```java
   import org.springframework.boot.context.properties.ConfigurationProperties;
   import org.springframework.context.annotation.Bean;
   import org.springframework.context.annotation.Configuration;
   import org.springframework.jdbc.datasource.DriverManagerDataSource;
   import javax.sql.DataSource;

   @Configuration
   public class DataSourceConfig {

       @Bean(name = "firstDataSource")
   ```

~~~
       @ConfigurationProperties(prefix = "datasource.first")
       public DataSource firstDataSource() {
          return new DriverManagerDataSource();
       }

       @Bean(name = "secondDataSource")
       @ConfigurationProperties(prefix = "datasource.second")
       public DataSource secondDataSource() {
          return new DriverManagerDataSource();
       }
   }
   ```
~~~

3. **Use the Datasources in Repositories or DAOs**:

   Inject the datasources into your repositories or DAOs using the `@Qualifier` annotation to specify which datasource to inject. For example:

~~~
   ```java
   import org.springframework.beans.factory.annotation.Qualifier;
   import org.springframework.stereotype.Repository;
   import javax.sql.DataSource;

   @Repository
   public class MyRepository {

       private final DataSource firstDataSource;
       private final DataSource secondDataSource;

       public MyRepository(@Qualifier("firstDataSource") DataSource
firstDataSource,
                           @Qualifier("secondDataSource") DataSource
secondDataSource) {
          this.firstDataSource = firstDataSource;
          this.secondDataSource = secondDataSource;
       }

       // Use firstDataSource and secondDataSource as needed
   }
   ```
~~~

With this configuration, you have two datasources (`firstDataSource` and `secondDataSource`) configured using properties from `application.properties`. These datasources can be injected into your repositories or DAOs as needed, allowing you to interact with multiple databases in your Spring Boot application.