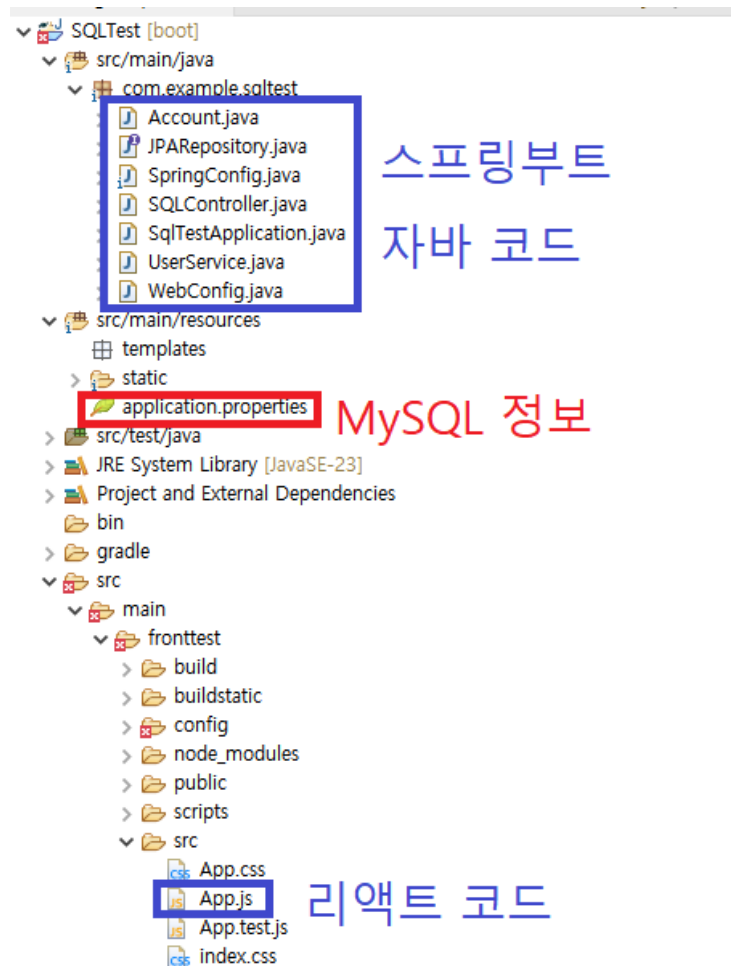


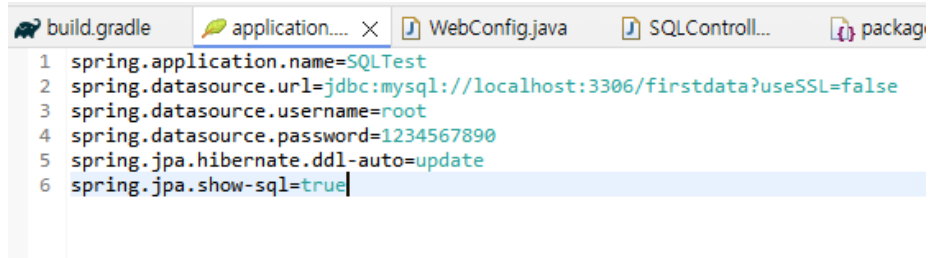
스프링부트 리액트 MySQL 연동

October 16, 2024

1 프로그램 구조



2 application.properties

A screenshot of an IDE window showing the 'application.properties' file. The window has several tabs: 'build.gradle', 'application....', 'WebConfig.java', 'SQLControll...', and 'packag...'. The 'application.properties' tab is active, displaying the following configuration lines:

```
1 spring.application.name=SQLTest
2 spring.datasource.url=jdbc:mysql://localhost:3306/firstdata?useSSL=false
3 spring.datasource.username=root
4 spring.datasource.password=1234567890
5 spring.jpa.hibernate.ddl-auto=update
6 spring.jpa.show-sql=true|
```

핵심

2줄의 'localhost:3306'는 접속할 DB의 주소를 적는 부분, 'firstdata'부분은 MySQL내부에 있는 여러 데이터베이스 중 접속할 특정 데이터베이스를 명시한다.

3줄, 4줄은 MySQL접속할 때 쓰는 계정과 비밀번호다.

5줄은 테이블을 선언한 엔티티에 따라
테이블을 자동으로 관리하는 옵션이다.

6줄은 데이터베이스 내부 sql이 출력되게 하는 옵션이다.

3 빌드 하는 방법

우선 build.gradle파일 최하단에 다음 코드를 추가한다.

```

33 def frontendDir = "$projectDir/src/main" frontend
34
35
36 sourceSets {
37     main {
38         resources { srcDirs = ["$projectDir/src/main/resources"]
39         }
40     }
41 }
42
43 processResources { dependsOn "copyReactBuildFiles" }
44
45 task installReact(type: Exec) {
46     workingDir "$frontendDir"
47     inputs.dir "$frontendDir"
48     group = BasePlugin.BUILD_GROUP
49     if (System.getProperty('os.name').toLowerCase(Locale.ROOT).contains('windows')) {
50         commandLine "npm.cmd", "audit", "fix"
51     } else {
52         commandLine "npm.cmd", 'install'
53     }
54 }
55
56
57 task buildReact(type: Exec) {
58     dependsOn "installReact"
59     workingDir "$frontendDir"
60     inputs.dir "$frontendDir"
61     group = BasePlugin.BUILD_GROUP
62     if (System.getProperty('os.name').toLowerCase(Locale.ROOT).contains('windows')) {
63         commandLine "npm.cmd", "run-script", "build"
64     } else {
65         commandLine "npm", "run-script", "build"
66     }
67 }
68
69 task copyReactBuildFiles(type: Copy) {
70     dependsOn "buildReact"
71     from "$frontendDir/build"
72     into "$projectDir/src/main/resources/static"
73 }

```

코드 추가 할 때

맨 윗 표시된 부분은 프론트엔드의 폴더명에 따른다.

그 후, cmd를 통해 스프링부트 최상위 폴더로 이동한 후, 'gradlew build'을 입력해서 Build Success가 뜨면 된다.

```
C:\Users#82105\Desktop#Springboot#BackendBasic#SQLTest>gradlew build
<-----> 6% EXECUTING [7s]
> :installReact
```

다음은 build—libs로 이동하여 다음과 같이 치면
 React와 springboot가 연동된 서버가 springboot포트로
 실행된다. java -jar (프로젝트)...SNAPSHOT.jar

```
C:\Users#02105\Desktop#Springboot#BackendBasic#SQLTest#build#1\libs>java -jar SQLTest-0.0.1-SNAPSHOT.jar

  ____  _
 / ___|| | | |
| |___| |_| |
|___|_||_|_|_|

:: Spring Boot ::
               (v3.3.4)

2024-10-17T01:54:54.039+09:00 INFO 14392 --- [SQLTest] [ main ] com.example.sqltest.SqlTestApplication : Starting SqlTestApplication v0.0.1-SNAPSHOT using Java 23 with PID 14392 (C:\Users#02105\Desktop#Springboot#BackendBasic#SQLTest#build#1\libs#SQLTest-0.0.1-SNAPSHOT.jar started by 02105 in C:\Users#02105\Desktop#Springboot#BackendBasic#SQLTest#build#1\libs)
2024-10-17T01:54:54.044+09:00 INFO 14392 --- [SQLTest] [ main ] com.example.sqltest.SqlTestApplication : No
```

4 대략적인 코드 설명

Account

@Entity가 붙어있는 클래스로, DB의 테이블에 대응되어
멤버변수를 통해 데이터를 저장한다.

JpaRepository

DB와 통신하는 JpaRepository를 상속받기 위한 객체이다.

SpringConfig

UserService클래스의 수명을 관리하는 클래스이다.

SQLController

springboot의 컨트롤러로 요청을 받아서 세부사항에
맞게 코드를 분기하는 역할을 맡는다.

핵심은 @RequestBody로 선언된 reqData로

Post에서 같이 오는 json파일을 받아서 읽는 역할을 한다.

UserService

@Transactional이 붙어있는 클래스로

여러 명령이 동시에 들어와서 DB의 정보가 섞이지 않게
중계하는 역할을 맡는다.

WebConfig

Cors정책으로 인해 다른 도메인에 있는 서버끼리는

기본적으로 통신을 거부하기 때문에

React가 Springboot에 접근할 수 있도록

허용해주는 역할을 맡는다.

App.js

React의 디자인에 관련된 js파일이다.

중요한 모듈로 axios가 있다. http통신을 관리하는 모듈로

```
axios.get("주소")  
.then((response) => {"정상응답시 실행될 함수"})  
.catch((error) => {"비정상응답시 실행될 함수"})
```

```
axios.post("주소", {"data1":"aa","data2":"ab"})  
.then((response) => {"정상응답시 실행될 함수"})  
.catch((error) => {"비정상응답시 실행될 함수"})  
와 같은 구조를 가지고 있다.
```