

〈13주차 실습〉

데이터베이스 어플리케이션 프로그래밍(2)

Database Programming



수강신청 시스템 : 삭제, 조회



IntelliJ 설치



Postman 설치



Spring Boot 프로젝트 생성 및 설정



JDBC Template



실습 과제

- **1. JAVA_HOME 새로 만들기 클릭**
변수이름 : JAVA_HOME
변수값 : 자바 jdk를 설치한 위치
예) C:\Program Files\Java\jdk1.8.0_311
- **2. PATH 기존변수 수정**
변수이름 : Path
변수값 : java가 설치된 디렉토리 밑의 bin 디렉토리
예) %JAVA_HOME%\bin
- **3. CLASSPATH 새로 만들기 클릭**
변수이름 : CLASSPATH
변수값 : java가 설치된 디렉토리 밑의 lib 폴더 안에 tools.jar
예) %JAVA_HOME%\lib\tools.jar
- **4. cmd 창 열어서 path, java, javac 각각 실행한 후 잘 되는지 확인**

- JSP에서 Java Beans 규약에 맞게 작성된 클래스를 사용할 수 있음
- 사용 이유
 - 재사용성을 증가시켜 프로그램의 효율이 상승됨
 - 유지 보수가 쉬우며, 코드를 재사용하기 때문에 안정성이 보장됨

- java 파일을 Tomcat 8.5\webapps\ROOT\WEB-INF\classes 경로에 배치함
*classes 폴더 생성해야 함
- jsp:useBean 형태로 접근하기 위해서는 Package를 지정해 주어야 함
- **Syntax**
 <jsp:useBean id= “reference명”
 class= “package명.class명” scope= “page” />

- BTest.java

```
package BTest;

public class BTest{
    private int cnt = 0;

    public String getMsg(){
        return "getMsg() Called";
    }

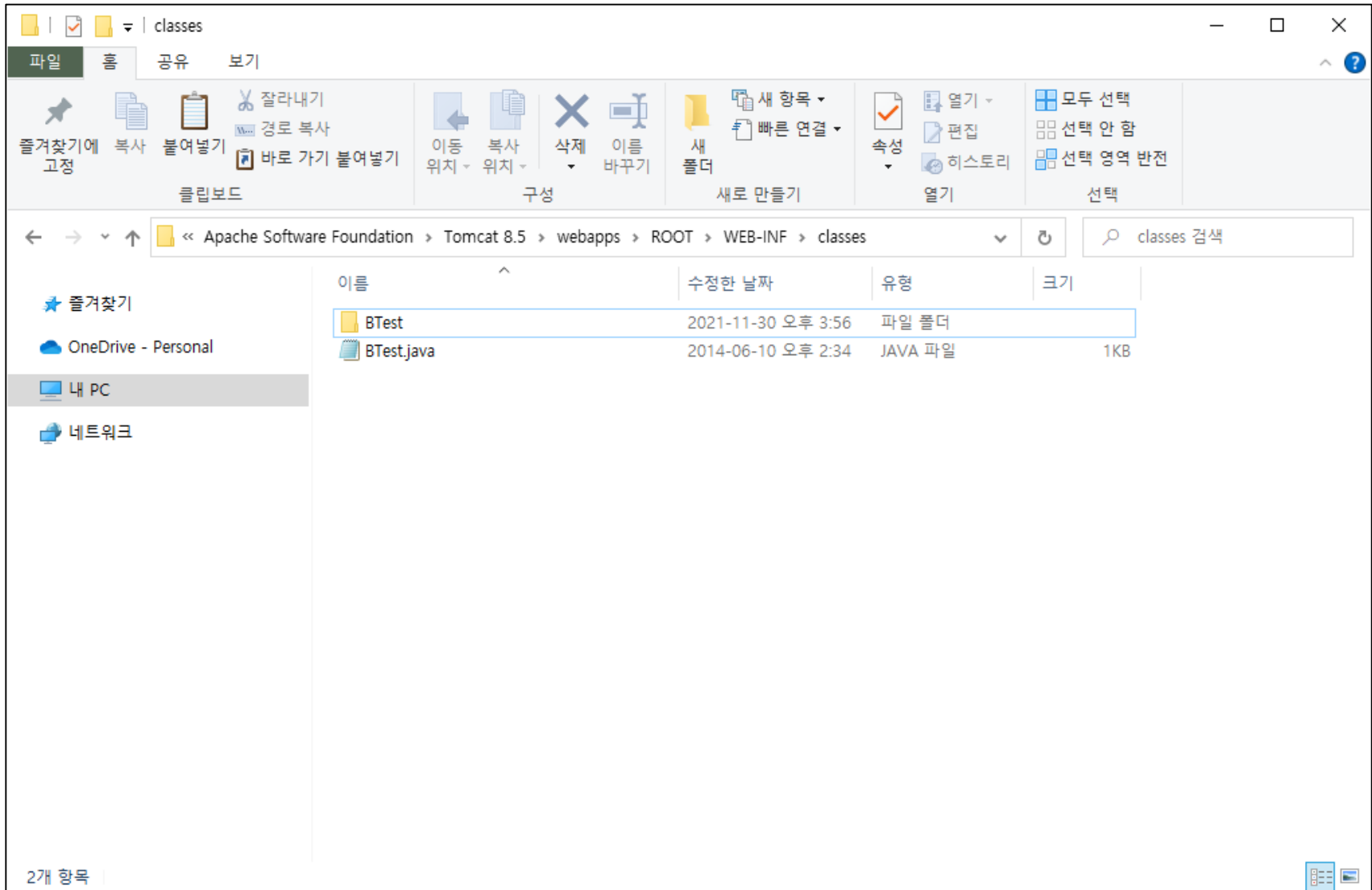
    public int getCnt(){
        return ++cnt;
    }
}
```

관리자: 명령 프롬프트

C:\>cd C:\Program Files\Apache Software Foundation\Tomcat 8.5\webapps\ROOT\WEB-INF\classes

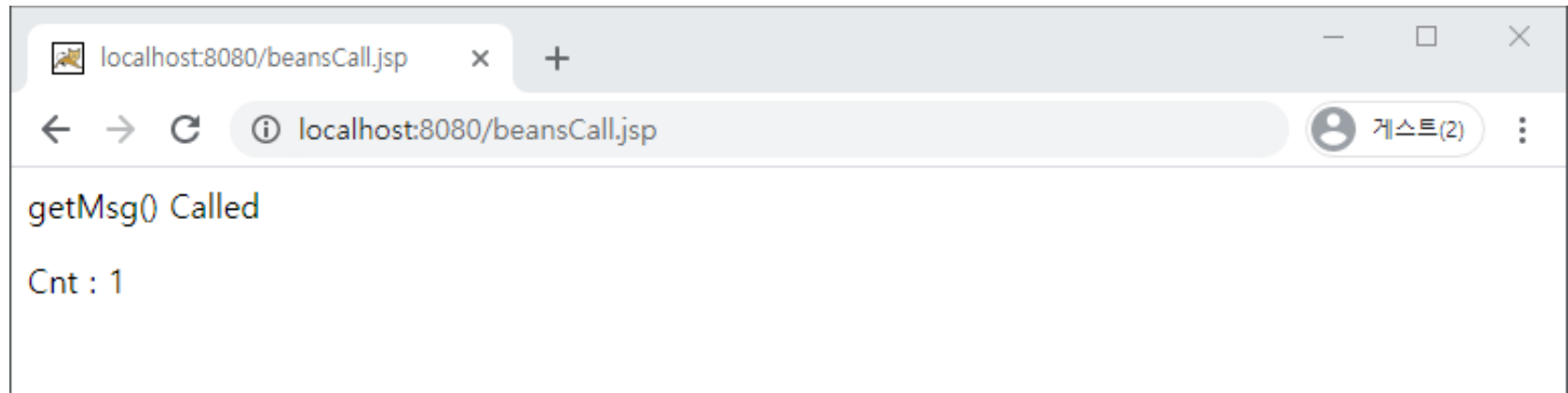
C:\Program Files\Apache Software Foundation\Tomcat 8.5\webapps\ROOT\WEB-INF\classes>javac -d . BTest.java

C:\Program Files\Apache Software Foundation\Tomcat 8.5\webapps\ROOT\WEB-INF\classes>



- beansCall.jsp

```
<%@ page language="java" contentType="text/html; charset=utf-8" pageEncoding="utf-8" %>
<!DOCTYPE html>
<jsp:useBean id="bTest" class="BTest.BTest" scope="page"></jsp:useBean>
<html>
    <p><%= bTest.getMsg() %></p>
    <p>Cnt : <%= bTest.getCnt() %></p>
</html>
```



수강신청 조회

localhost:8080/select.jsp

게스트(2)

로그아웃

사용자 정보 수정

수강신청 입력

수강신청 삭제

수강신청 조회

과목번호	분반	과목명	학점
C100	3	컴퓨터 프로그래밍	3
C200	3	자료구조	3
C300	3	알고리즘	3
C400	3	데이터베이스 시스템	3
C500	3	운영체제	3
C800	3	데이터베이스 프로그래밍	3

총신청과목수	6
총신청학점	18

2023년도 1학기

조회

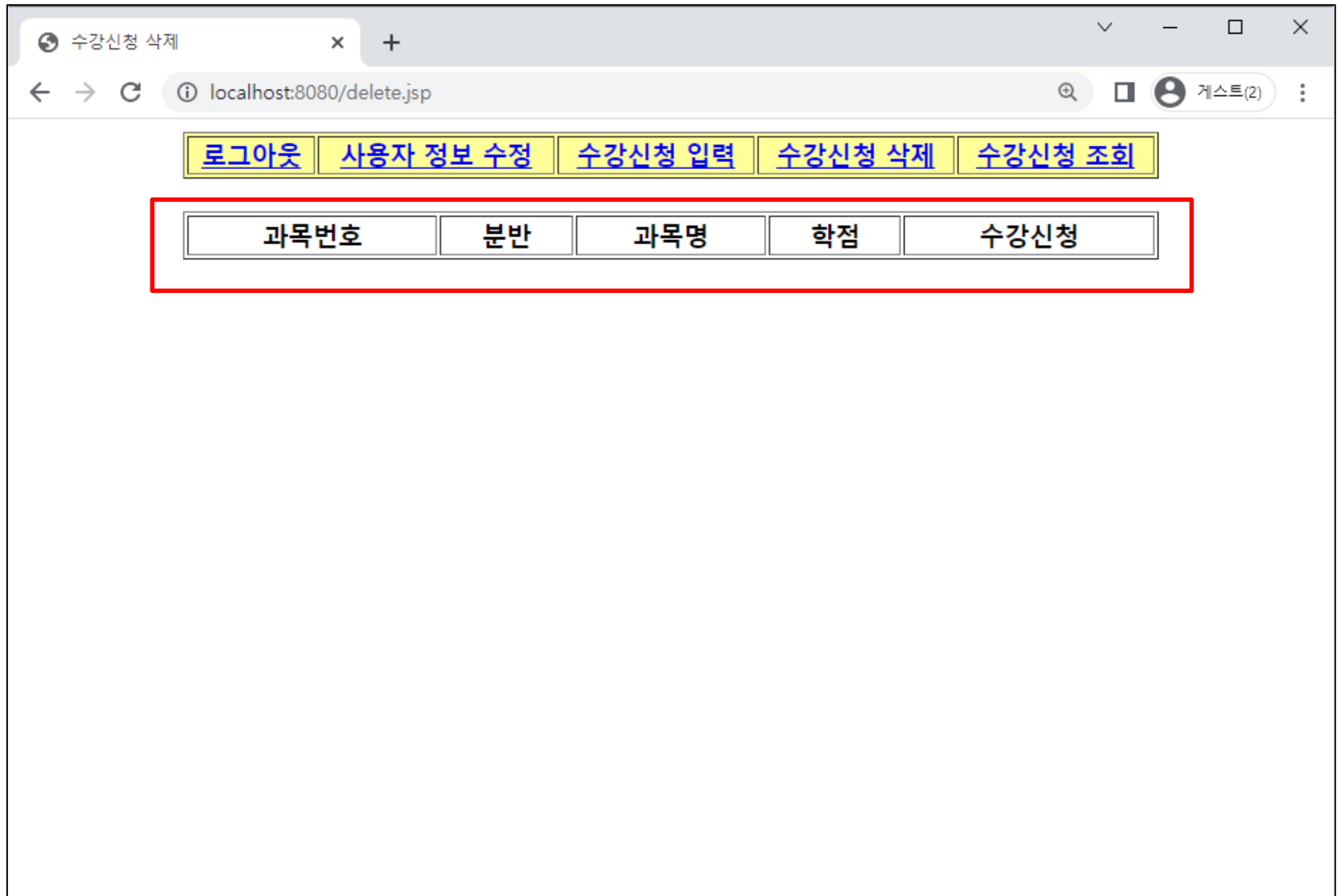
수강신청 삭제

localhost:8080/delete.jsp

게스트(2)

[로그아웃](#) [사용자 정보 수정](#) [수강신청 입력](#) [수강신청 삭제](#) [수강신청 조회](#)

과목번호	분반	과목명	학점	수강신청
C100	3	컴퓨터 프로그래밍	3	삭제
C200	3	자료구조	3	삭제
C300	3	알고리즘	3	삭제
C400	3	데이터베이스 시스템	3	삭제
C500	3	운영체제	3	삭제
C800	3	데이터베이스 프로그래밍	3	삭제



수강신청 조회

localhost:8080/select.jsp

게스트(2)

로그아웃

사용자 정보 수정

수강신청 입력

수강신청 삭제

수강신청 조회

과목번호	분반	과목명	학점
총신청과목수			0
총신청학점			0

2023

년도

1

학기

조회

수강신청 입력 x +

localhost:8080/insert.jsp

게스트(2)

[로그아웃](#) [사용자 정보 수정](#) [수강신청 입력](#) [수강신청 삭제](#) [수강신청 조회](#)

과목번호	분반	과목명	학점	수강신청
C800	3	데이터베이스 프로그래밍	3	신청
M100	3	멀티미디어 개론	3	신청
M300	3	그래픽 활용	3	신청
C200	3	자료구조	3	신청
C500	3	운영체제	3	신청
C300	3	알고리즘	3	신청
C100	3	컴퓨터 프로그래밍	3	신청
M400	3	윈도우즈 프로그래밍	3	신청
C900	3	객체지향 윈도우즈 프로그래밍	3	신청
C400	3	데이터베이스 시스템	3	신청
M200	3	선형대수	3	신청
M700	3	게임 프로그래밍	3	신청

수강신청 조회

localhost:8080/select.jsp

게스트(2)

[로그아웃](#) [사용자 정보 수정](#) [수강신청 입력](#) [수강신청 삭제](#) [수강신청 조회](#)

과목번호	분반	과목명	학점
C800	3	데이터베이스 프로그래밍	3
M100	3	멀티미디어 개론	3
M300	3	그래픽 활용	3

총신청과목수	3
총신청학점	9

2023

년도

1

학기

조회

```
package enrollBean;

public class Enroll {
    private String c_id;
    private int c_id_no;
    private String c_name;
    private int c_unit;

    public Enroll() {
        c_id = null;
        c_id_no = 0;
        c_name = null;
        c_unit = 0;
    }

    public void setCId(String c_id) {
        this.c_id = c_id;
    }

    public void setCIdNo(int c_id_no) {
        this.c_id_no = c_id_no;
    }

    public void setCName(String c_name) {
        this.c_name = c_name;
    }

    public void setCUnit(int c_unit) {
        this.c_unit = c_unit;
    }

    public String getCId() {
        return c_id;
    }

    public int getCIdNo() {
        return c_id_no;
    }

    public String getCName() {
        return c_name;
    }

    public int getCUnit() {
        return c_unit;
    }
}
```

```
package enrollBean;
import java.sql.*;
import java.util.*;
import javax.sql.*;
import oracle.jdbc.driver.*;
import oracle.jdbc.pool.*;
import enrollBean.*;

public class EnrollMgr {

    private OracleConnectionPoolDataSource ocpds = null;
    private PooledConnection pool = null;

    public EnrollMgr() {
        try{

            ocpds = new OracleConnectionPoolDataSource();

            ocpds.setURL("jdbc:oracle:thin:@210.94.199.20:1521:DBLAB");
            ocpds.setUser("사용자 계정");
            ocpds.setPassword("비밀번호");

            pool = ocpds.getPooledConnection();
        }catch(Exception e){
            System.out.println("Error : Connection Failed");
        }
    }

    public Vector getEnrollList(String s_id) {
        Connection conn = null;
        PreparedStatement pstmt = null;
        CallableStatement cstmt1 = null;
        CallableStatement cstmt2 = null;
        ResultSet rs = null;
        Vector vecList = new Vector();
    }
}
```



```
try {
    conn = pool.getConnection();

    cstmt1 = conn.prepareCall("{? = call Date2EnrollYear(SYSDATE)}");
    cstmt1.registerOutParameter(1, java.sql.Types.INTEGER);
    cstmt1.execute();
    int nYear = cstmt1.getInt(1);

    cstmt2 = conn.prepareCall("{? = call Date2EnrollSemester(SYSDATE)}");
    cstmt2.registerOutParameter(1, java.sql.Types.INTEGER);
    cstmt2.execute();
    int nSemester = cstmt2.getInt(1);

    String mySQL = "select e.c_id cid,e.c_id_no cid_no,c.c_name cname,c.c_unit cunit "
        + "from enroll e, course c where e.s_id=? and e.e_year=? and e.e_semester=? "
        + "and e.c_id=c.c_id and e.c_id_no=c.c_id_no";
    pstmt = conn.prepareStatement(mySQL);
    pstmt.setString(1,s_id); pstmt.setInt(2, nYear);
    pstmt.setInt(3, nSemester);
    rs = pstmt.executeQuery();

    while (rs.next()) {
        Enroll en = new Enroll();
        en.setCId(rs.getString("cid"));
        en.setCIdNo(rs.getInt("cid_no"));
        en.setCName(rs.getString("cname"));
        en.setCUnit(rs.getInt("cunit"));
        vecList.add(en);
    }
    cstmt1.close();
    cstmt2.close();
    pstmt.close();
    conn.close();

} catch (Exception ex) {
    System.out.println("Exception" + ex);
}

return vecList;
}
```

```
public Vector getEnrollList(String s_id, int nYear, int nSemester) {
    Connection conn = null;
    PreparedStatement pstmt = null;
    ResultSet rs = null;
    Vector vecList = new Vector();

    try {
        conn = pool.getConnection();

        String mySQL = "select e.c_id cid,e.c_id_no cid_no,c.c_name cname,c.c_unit cunit "
            + "from enroll e, course c where e.s_id=? and e.e_year=? and e.e_semester=? "
            + "and e.c_id=c.c_id and e.c_id_no=c.c_id_no";
        pstmt = conn.prepareStatement(mySQL);
        pstmt.setString(1,s_id);
        pstmt.setInt(2, nYear);
        pstmt.setInt(3, nSemester);
        rs = pstmt.executeQuery();

        while (rs.next()) {
            Enroll en = new Enroll();
            en.setCId(rs.getString("cid"));
            en.setCIdNo(rs.getInt("cid_no"));
            en.setCName(rs.getString("cname"));
            en.setCUnit(rs.getInt("cunit"));
            vecList.add(en);
        }
        pstmt.close();
        conn.close();
    } catch (Exception ex) {
        System.out.println("Exception" + ex);
    }

    return vecList;
}
```

```
public int getCurrentYear()
{
    int nYear=0;
    Connection conn = null;
    CallableStatement cstmt = null;
    try {
        conn = pool.getConnection();

        cstmt = conn.prepareCall("{? = call Date2EnrollYear(SYSDATE)}");
        cstmt.registerOutParameter(1, java.sql.Types.INTEGER);
        cstmt.execute();
        nYear = cstmt.getInt(1);

        cstmt.close();
        conn.close();
    } catch (Exception ex) {
        System.out.println("Exception" + ex);
    }

    return nYear;
}
```

```
public int getCurrentSemester()
{
    int nSemester=0;
    Connection conn = null;
    CallableStatement cstmt = null;

    try {
        conn = pool.getConnection();

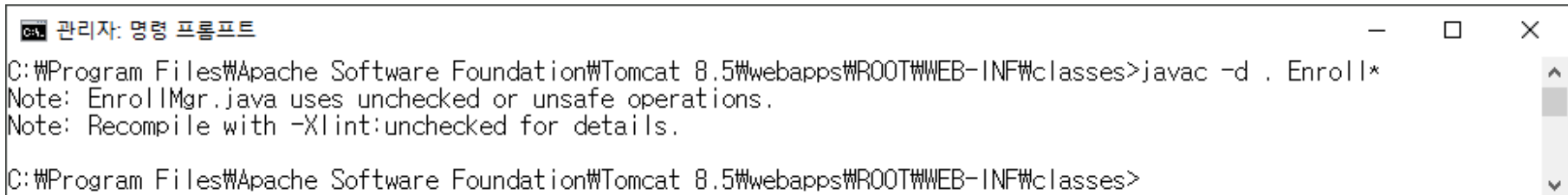
        cstmt = conn.prepareCall("{? = call Date2EnrollSemester(SYSDATE)}");
        cstmt.registerOutParameter(1, java.sql.Types.INTEGER);
        cstmt.execute();
        nSemester = cstmt.getInt(1);

        cstmt.close();
        conn.close();
    } catch (Exception ex) {
        System.out.println("Exception" + ex);
    }

    return nSemester;
}
```

```
public void deleteEnroll(String s_id, String c_id, int c_id_no) {
    Connection conn = null;
    PreparedStatement pstmt = null;
    try {
        conn = pool.getConnection();
        String mySQL = "delete from enroll where s_id=? and c_id=? and c_id_no=?";
        pstmt = conn.prepareStatement(mySQL);
        pstmt.setString(1,s_id);
        pstmt.setString(2, c_id);
        pstmt.setInt(3, c_id_no);
        pstmt.executeUpdate();

        pstmt.close();
        conn.close();
    } catch (Exception ex) {
        System.out.println("Exception" + ex);
    }
}
```



```
관리자: 명령 프롬프트
C:\Program Files\Apache Software Foundation\Tomcat 8.5\webapps\ROOT\WEB-INF\classes>javac -d . Enroll*
Note: EnrollMgr.java uses unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.
C:\Program Files\Apache Software Foundation\Tomcat 8.5\webapps\ROOT\WEB-INF\classes>
```

- **Package**파일이 두 개 이상일 때 한 번에 컴파일 해주어야 함

```
<%@ page language="java" contentType="text/html; charset=utf-8" pageEncoding="utf-8"%>
<%@ page import="java.util.*, enrollBean.*"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="EUC-KR">
<title>수강신청 조회</title>
</head>
<body>
    <%@ include file="top.jsp"%>
    <% if (session_id == null)    response.sendRedirect("login.jsp"); %>

    <table width="75%" align="center" border>
        <tr>
            <th>과목번호</th>
            <th>분반</th>
            <th>과목명</th>
            <th>학점</th>
        </tr>
        <br>
        <jsp:useBean id="enrollMgr" class="enrollBean.EnrollMgr" />
        <%
            Vector vlist = null;
            int year, semester;

            if (request.getParameter("year") == null && request.getParameter("semester") == null) {
                year = enrollMgr.getCurrentYear();
                semester = enrollMgr.getCurrentSemester();
            } else {
                year = Integer.parseInt(request.getParameter("year"));
                semester = Integer.parseInt(request.getParameter("semester"));
            }

            vlist = enrollMgr.getEnrollList(session_id, year, semester);
            int counter = vlist.size();
            int totUnit = 0;

            for (int i = 0; i < vlist.size(); i++) {
                Enroll en = (Enroll) vlist.elementAt(i);
                totUnit += en.getCUnit();
            }
        %>
    </table>
</body>
</html>
```

```
<tr>
  <td align="center"><%=en.getCId() %></td>
  <td align="center"><%=en.getCIdNo() %></td>
  <td align="center"><%=en.getCName() %></td>
  <td align="center"><%=en.getCUUnit() %></td>

  <%
    }
  %>

</tr>
</table>
<br>
<table width="75%" align="center" border>
  <tr>
    <th>충신청과목수</th>
    <td align="center"><%=counter%></td>
  </tr>
  <tr>
    <th>충신청학점</th>
    <td align="center"><%=totUnit%></td>
  </tr>
</table>
<br>
<br>
<table width="30%" align="center">
  <FORM method="post" action="select_verify.jsp">
    <tr>
      <td><input type="text" name="year" value=<%=year%> size=4>
        년도 <input type="text" name="semester" value=<%=semester%> size=1>
        학기 <INPUT TYPE="SUBMIT" NAME="Submit" VALUE="조회"></td>
    </tr>
  </FORM>
</table>
</body>
</html>
```

```
<%@ page language="java" contentType="text/html; charset=utf-8" pageEncoding="utf-8"%>
<%@ page import="java.sql.*"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="EUC-KR">
<title>수강신청 조회</title>
</head>
<body>
    <%
        String year = request.getParameter("year");
        String semester = request.getParameter("semester");

        String url = "select.jsp?year=" + year + "&semester=" + semester;

        response.sendRedirect(url);
    %>
</body>
</html>
```



```
<%@ page language="java" contentType="text/html; charset=utf-8" pageEncoding="utf-8"%>
<%@ page import="java.util.*, enrollBean.*"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="EUC-KR">
<title>수강신청 삭제</title>
</head>
<body>
    <%@ include file="top.jsp"%>
    <%
        if (session_id == null)
            response.sendRedirect("login.jsp");
    %>
    <table width="75%" align="center" border>
        <tr>
            <th>과목번호</th>
            <th>분반</th>
            <th>과목명</th>
            <th>학점</th>
            <th>수강신청</th>
        </tr>
        <br>
        <jsp:useBean id="enrollMgr" class="enrollBean.EnrollMgr" />
        <%
            Vector vlist = enrollMgr.getEnrollList(session_id);
            int counter = vlist.size();
            for (int i = 0; i < vlist.size(); i++) {
                Enroll en = (Enroll) vlist.elementAt(i);
        %>
        <tr>
            <td align="center"><%=en.getCId() %></td>
            <td align="center"><%=en.getCIdNo() %></td>
            <td align="center"><%=en.getCName() %></td>
            <td align="center"><%=en.getCUUnit() %></td>
            <td align="center"><a
                href="delete_verify.jsp?c_id=<%=en.getCId() %>&c_id_no=<%=en.getCIdNo() %>">삭제</a></td>
        <%
            }
        %>
        </tr>
    </table>
</body>
</html>
```

```
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ page import="enrollBean.*"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="EUC-KR">
<title>수강신청 삭제 </title>
</head>
<body>
    <%
        String s_id = (String) session.getAttribute("user");
        String c_id = request.getParameter("c_id");
        int c_id_no = Integer.parseInt(request.getParameter("c_id_no"));
    %>
    <jsp:useBean id="enrollMgr" class="enrollBean.EnrollMgr" scope="page" />
    <%
        enrollMgr.deleteEnroll(s_id, c_id, c_id_no);
    %>
    <script>
        alert("삭제되었습니다.");
        location.href = "delete.jsp";
    </script>
</body>
</html>
```

- 1) <https://www.jetbrains.com/ko-kr/idea/download/> 접속
- 2) 운영체제를 선택하여 다운로드 클릭

IntelliJ IDEA 새 버전이 제공됩니다 새로운 기능 기능 리소스 [가격 책정](#) [다운로드](#)

다운로드 IntelliJ IDEA

Windows macOS Linux

Ultimate

Java 및 Kotlin용 최고의 IDE

[다운로드](#) .exe ▼

30일 무료 평가 이용 가능

Community Edition

순수 Java 및 Kotlin 개발을 위한 IDE

[다운로드](#) .exe ▼

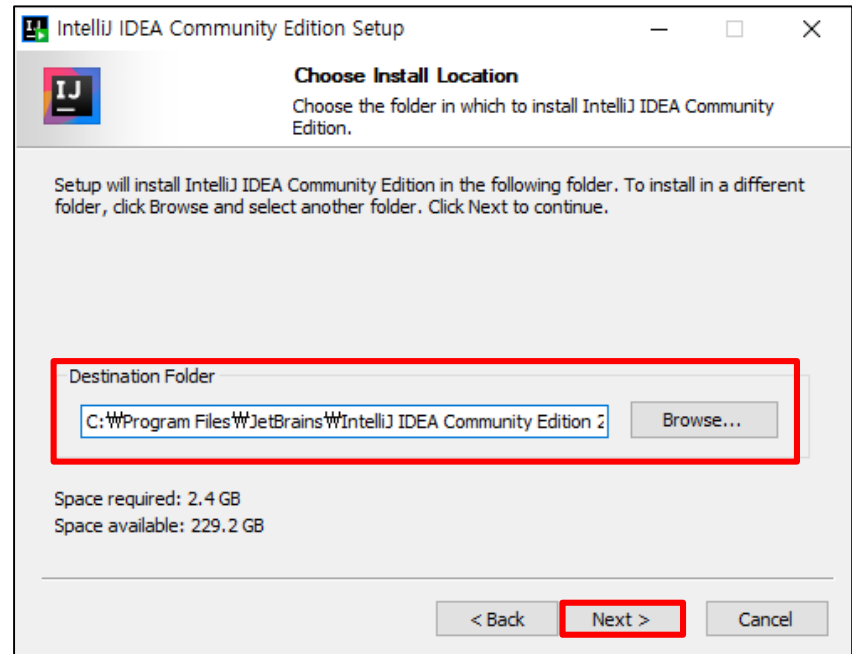
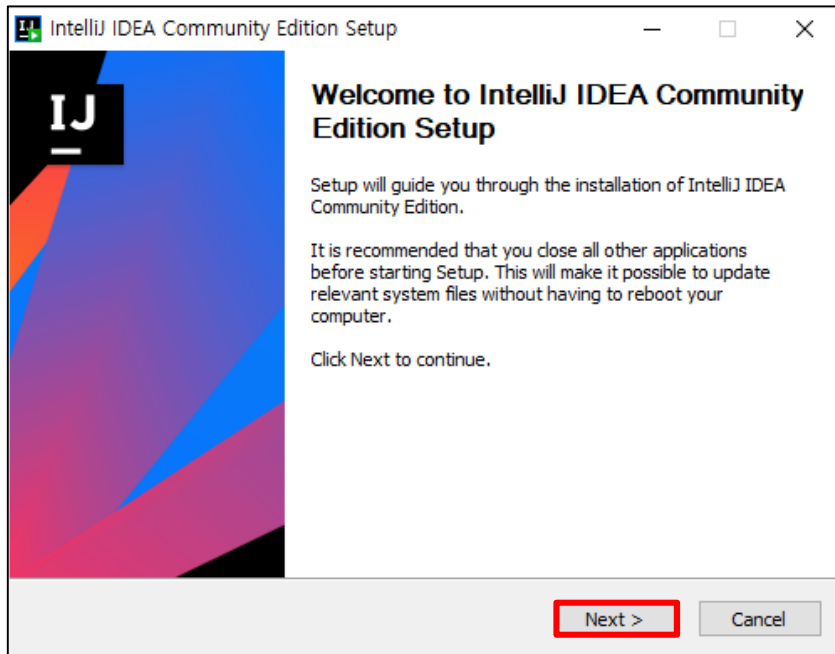
무료, 오픈 소스로 빌드됨

	IntelliJ IDEA Ultimate	IntelliJ IDEA Community Edition ⓘ
Java, Kotlin, Groovy, Scala	✓	✓
Maven, Gradle, sbt	✓	✓

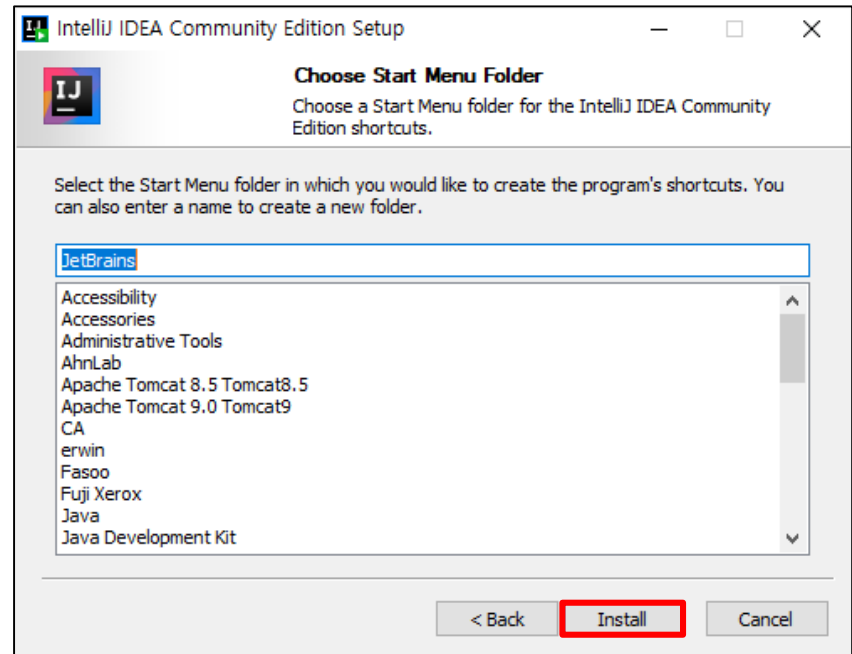
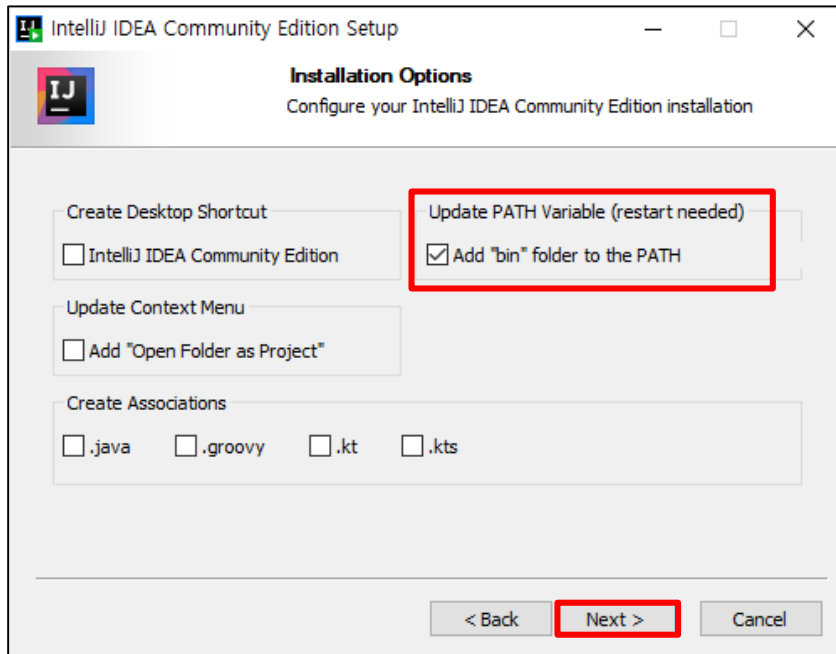
Feedback

3) 다운로드 받은 설치 파일 실행 → Next 클릭

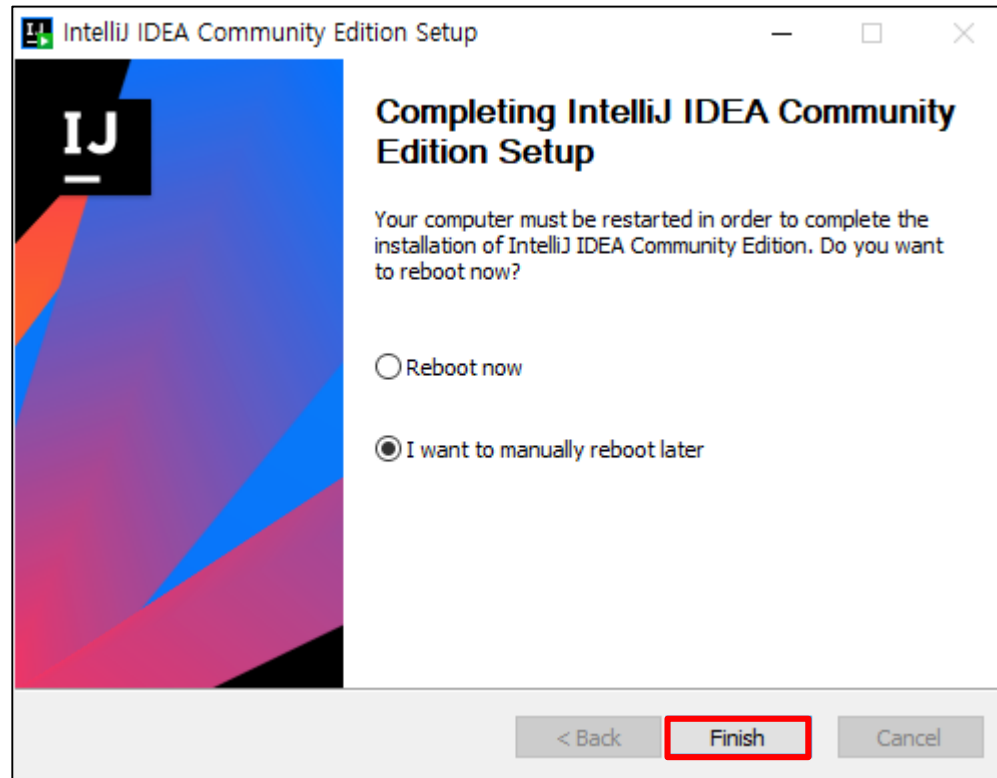
4) 설치 경로 확인 → Next 클릭



- 5) Add "bin" folder to the PATH 체크 (환경 변수 자동 설정) → Next 클릭
- 6) Install 클릭

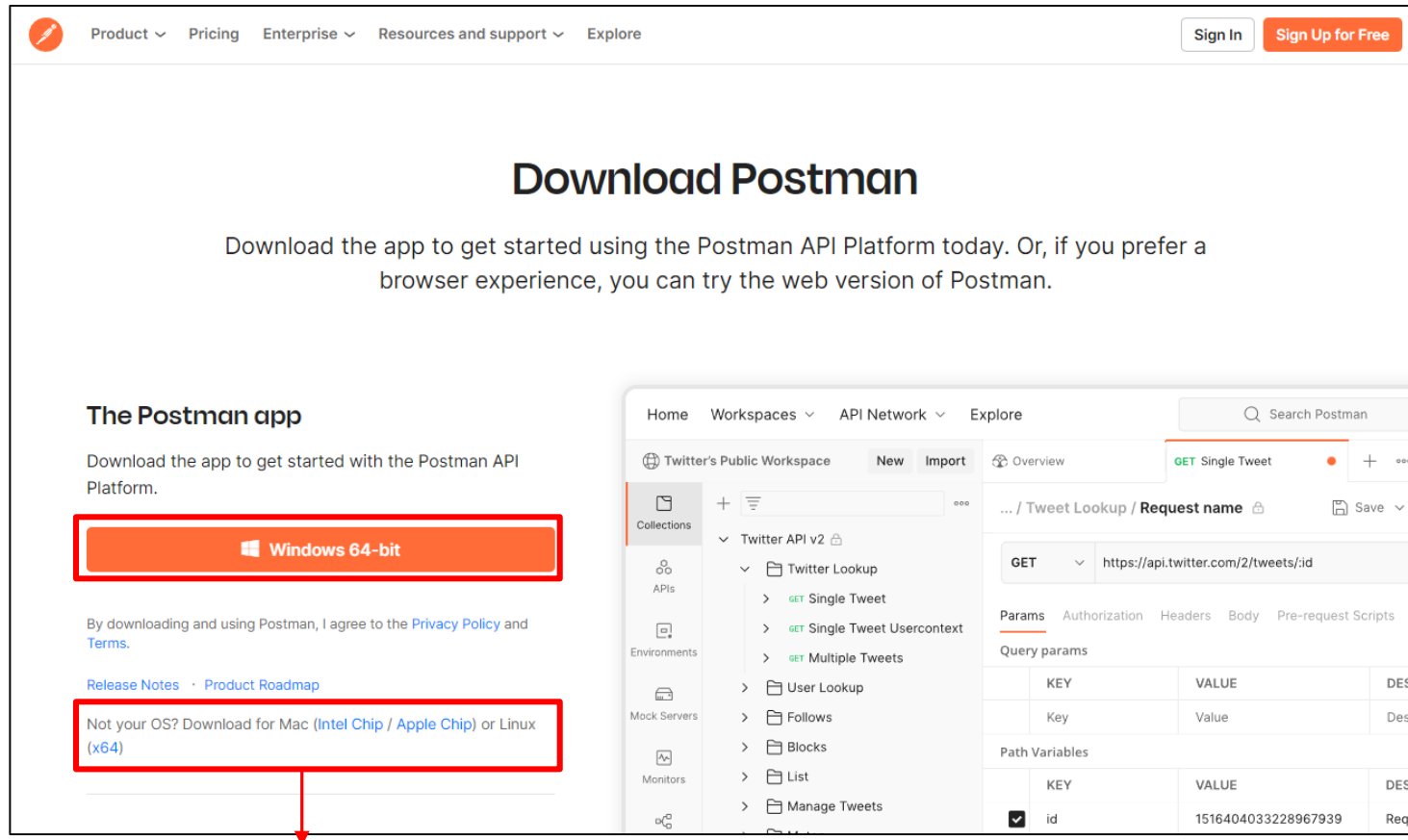


7) Finish 클릭



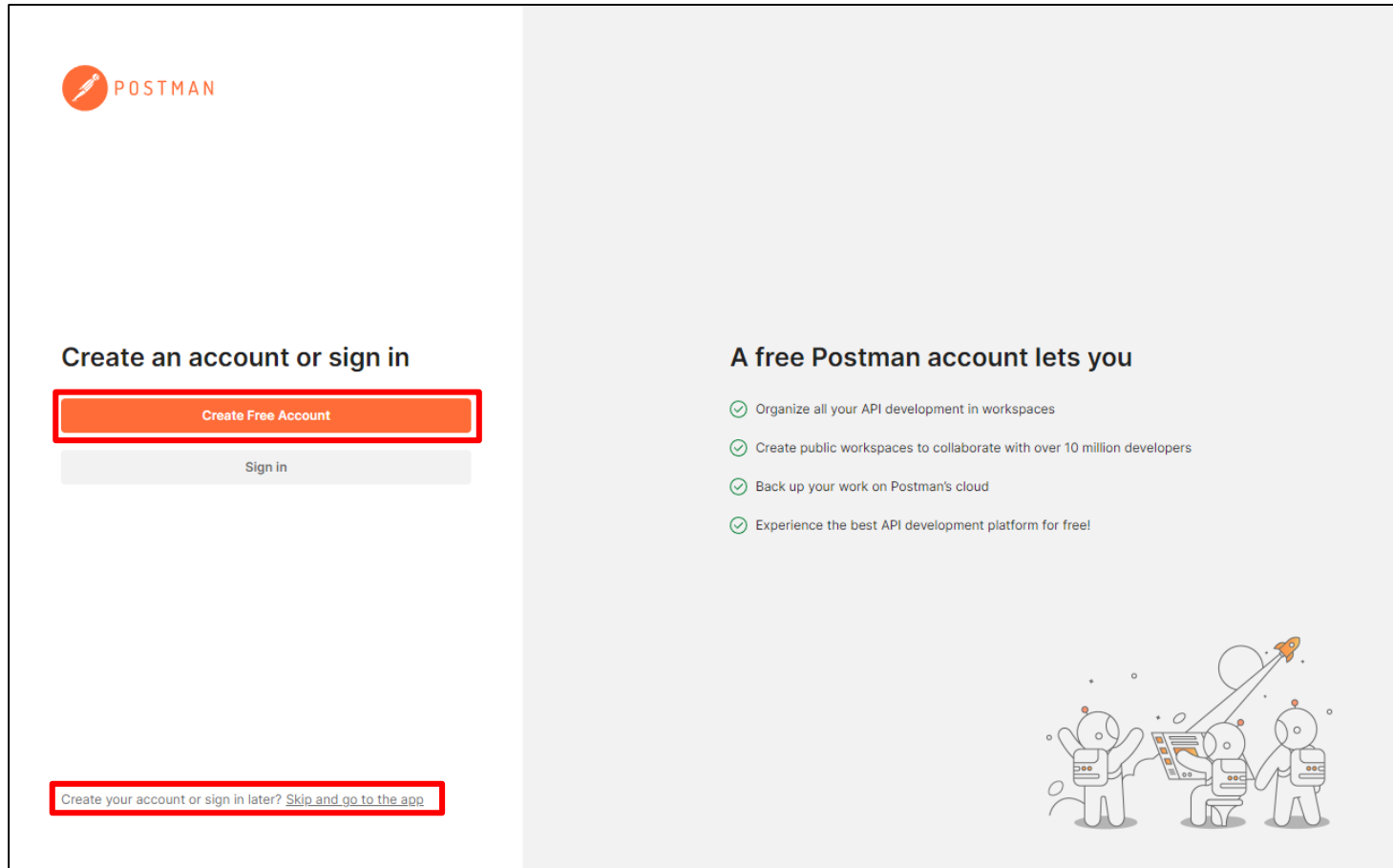
1) <https://www.postman.com/downloads/> 접속

2) "Windows 64-bit" 클릭 (다른 OS 사용 시, Mac 또는 Linux 선택하여 클릭)



Mac 또는 Linux 사용 시 클릭

3) Postman 설치파일 실행 후 "Create Free Account" 클릭하여 회원가입
또는 "Skip and go to the app" 클릭



- 1) Spring initializr (<https://start.spring.io/>) 접속
- 2) Project, Language, Spring Boot, Project Metadata, Packaging, Java 설정
- 3) "ADD DEPENDENCIES..." 클릭

The screenshot shows the Spring Initializr web form with the following settings and annotations:

- 1 Project:** ☒ Gradle - Groovy, ☐ Gradle - Kotlin, ☐ Maven
- 2 Language:** ☒ Java, ☐ Kotlin, ☐ Groovy
- 3 Spring Boot:** ☐ 3.0.1 (SNAPSHOT), ☐ 3.0.0, ☐ 2.7.7 (SNAPSHOT), ☒ 2.7.6
- 4 Project Metadata:**
 - Group: dbp
 - Artifact: UnivJdbcTemplate
 - Name: UnivJdbcTemplate
 - Description: Demo project for Spring Boot
 - Package name: dbp.UnivJdbcTemplate
- 5 Packaging:** ☒ Jar, ☐ War
- 6 Java:** ☐ 19, ☐ 17, ☐ 11, ☒ 8

Dependencies: ADD DEPENDENCIES... CTRL + B
No dependency selected

Summary:

- ① Project : Gradle - Groovy
- ② Language : Java
- ③ Spring Boot : 2.7.6
(3.0.0 버전은 Java 17 이상 지원)
- ④ Project Metadata : Group, Artifact
- ⑤ Packaging : Jar
- ⑥ Java : 버전 선택

Buttons: GENERATE CTRL + G, EXPLORE CTRL + SPACE, SHARE...

4) "MySQL Driver", "Spring Web", "Lombok", "Spring Data JPA", "Spring Data JDBC" 검색 후 Dependencies 추가

The screenshot shows the Spring Boot IDE interface. On the left, the 'Project' sidebar is visible with 'Spring Boot' selected. The 'Project Metadata' section shows 'Group: com.example', 'Artifact: demo', 'Name: demo', 'Description: Demo project', 'Package name: com.example', 'Packaging: Jar', and 'Java: 19'. The main area displays a list of dependencies. A red box highlights the search input field with the text '검색어 입력' (Search input) and an arrow pointing to it. The search results show 'Web, Security, JPA, Actuator, Devtools...' and 'Press Ctrl for multiple adds'. Below this, a list of dependencies is shown, each with a green bar indicating its status. The dependencies are: 'GaalVM Native Support', 'Spring Boot DevTools', 'Lombok', 'Spring Configuration Processor', 'Spring Web', 'Spring Reactive Web', and 'Spring for GraphQL'. To the right, a list of dependencies is shown, each with a green bar indicating its status. The dependencies are: 'MySQL Driver', 'Spring Web', 'Lombok', 'Spring Data JPA', and 'Spring Data JDBC'. A red arrow points from the 'Spring Web' dependency in the main list to the 'Spring Web' dependency in the right list. Another red arrow points from the 'Spring Data JPA' dependency in the main list to the 'Spring Data JPA' dependency in the right list. A third red arrow points from the 'Spring Data JDBC' dependency in the main list to the 'Spring Data JDBC' dependency in the right list. The right list also shows 'MySQL Driver', 'Spring Web', 'Lombok', 'Spring Data JPA', and 'Spring Data JDBC' with their respective status bars and descriptions.

검색어 입력 → Web, Security, JPA, Actuator, Devtools... Press Ctrl for multiple adds

DEVELOPER TOOLS

GaalVM Native Support
Support for compiling Spring applications to native executables using the GraalVM native-image compiler.

Spring Boot DevTools
Provides fast application restarts, LiveReload, and configurations for enhanced development experience.

Lombok
Java annotation library which helps to reduce boilerplate code.

Spring Configuration Processor
Generate metadata for developers to offer contextual help and "code completion" configuration keys (ex.application.properties/.yaml files).

WEB

Spring Web
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Spring Reactive Web
Build reactive web applications with Spring WebFlux and Netty.

Spring for GraphQL
Build GraphQL applications with Spring for GraphQL and GraphQL Java.

MySQL Driver Press Ctrl for multiple adds

MySQL Driver SQL
MySQL JDBC and R2DBC driver.

Spring Web Press Ctrl for multiple adds

Spring Web WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Lombok Press Ctrl for multiple adds

Lombok DEVELOPER TOOLS
Java annotation library which helps to reduce boilerplate code.

Spring Data JPA Press Ctrl for multiple adds

Spring Data JPA SQL
Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

Spring Data JDBC Press Ctrl for multiple adds

Spring Data JDBC SQL
Persist data in SQL stores with plain JDBC using Spring Data.

5) GENERATE 클릭(Spring boot 프로젝트 압축파일 생성)

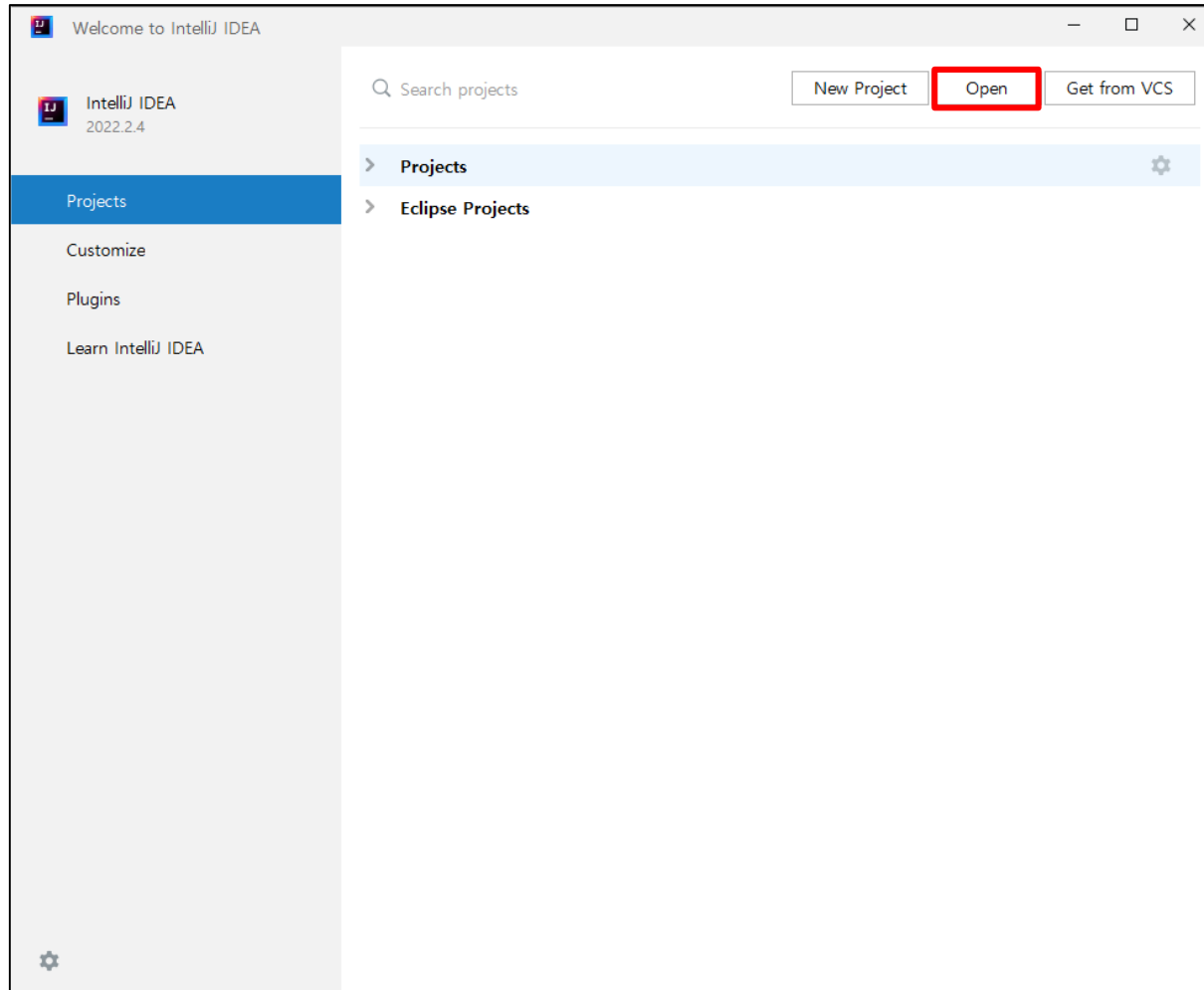
6) 다운로드 받은 파일 압축 해제

The image shows the Spring Initializr web interface. The left sidebar contains a hamburger menu icon and a Twitter icon. The main content area is divided into several sections:

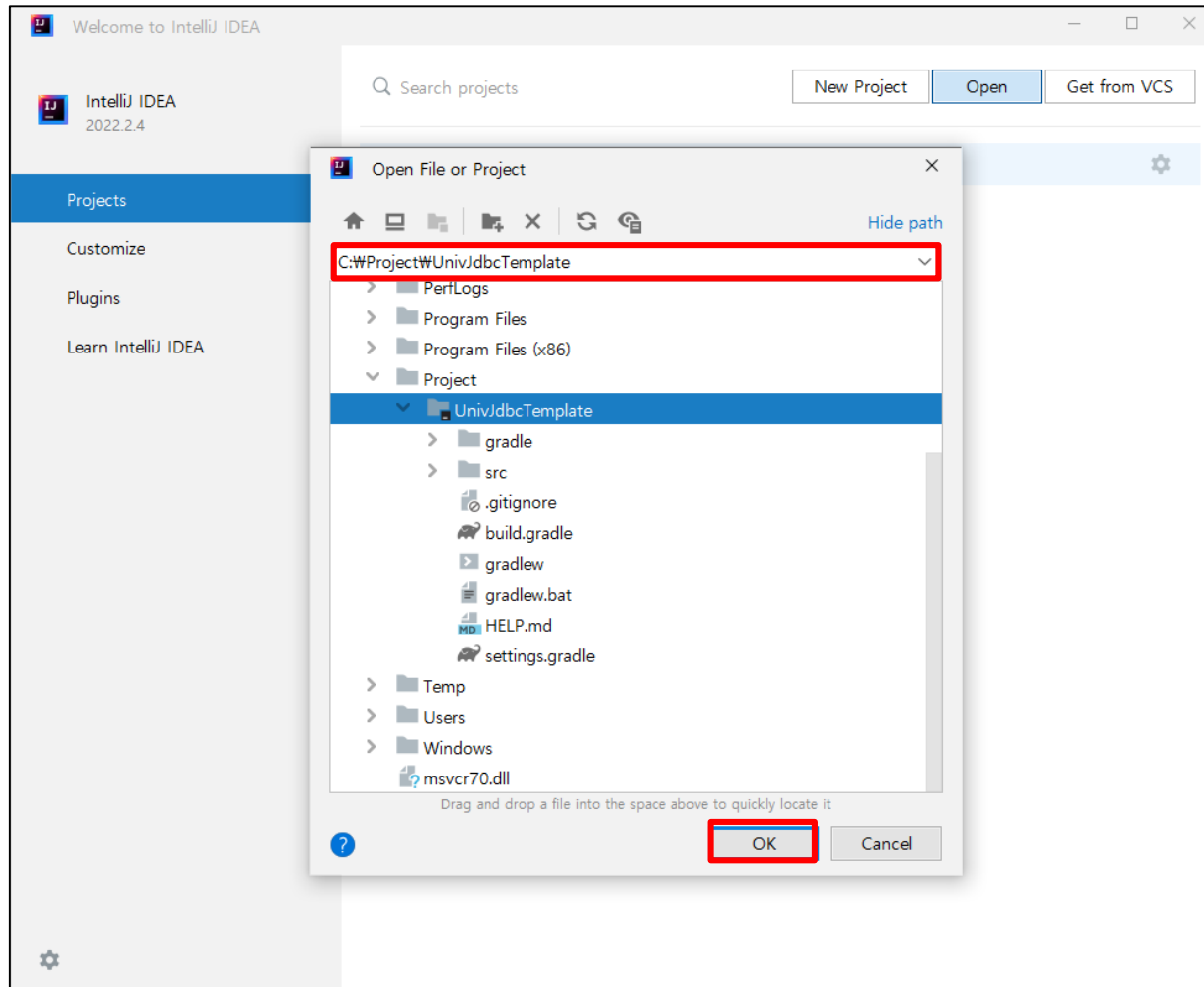
- Project:** Radio buttons for **Gradle - Groovy** (selected), **Gradle - Kotlin**, and **Maven**.
- Language:** Radio buttons for **Java** (selected), **Kotlin**, and **Groovy**.
- Spring Boot:** Radio buttons for **3.0.1 (SNAPSHOT)**, **3.0.0**, **2.7.7 (SNAPSHOT)**, and **2.7.6** (selected).
- Project Metadata:** Text input fields for **Group** (dbp), **Artifact** (UnivJdbcTemplate), **Name** (UnivJdbcTemplate), **Description** (Demo project for Spring Boot), and **Package name** (dbp.UnivJdbcTemplate).
- Packaging:** Radio buttons for **Jar** (selected) and **War**.
- Java:** Radio buttons for **19**, **17**, **11**, and **8** (selected).
- Dependencies:** A list of dependencies with a button **ADD DEPENDENCIES... CTRL + B**. The listed dependencies are:
 - MySQL Driver** (SQL): MySQL JDBC and R2DBC driver.
 - Spring Web** (WEB): Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.
 - Lombok** (DEVELOPER TOOLS): Java annotation library which helps to reduce boilerplate code.
 - Spring Data JPA** (SQL): Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.
 - Spring Data JDBC** (SQL): Persist data in SQL stores with plain JDBC using Spring Data.

At the bottom, there are three buttons: **GENERATE CTRL + G** (highlighted with a red box), **EXPLORE CTRL + SPACE**, and **SHARE...**.

7) IntelliJ 실행 → Open 클릭



8) 압축 해제한 폴더 경로 선택 → OK 클릭



- **build.gradle**

```
plugins {  
    id 'java'  
    id 'org.springframework.boot' version '2.7.6'  
    id 'io.spring.dependency-management' version '1.0.15.RELEASE'  
}
```

```
group = 'dbp'  
version = '0.0.1-SNAPSHOT'  
sourceCompatibility = '1.8'
```

```
configurations {  
    compileOnly {  
        extendsFrom annotationProcessor  
    }  
}
```

```
repositories {  
    mavenCentral()  
}
```

- build.gradle

```
dependencies {  
    implementation 'org.springframework.boot:spring-boot-starter-data-jdbc'  
    implementation 'org.springframework.boot:spring-boot-starter-data-jpa'  
    implementation 'org.springframework.boot:spring-boot-starter-web'  
    compileOnly 'org.projectlombok:lombok'  
    runtimeOnly 'com.mysql:mysql-connector-j'  
    annotationProcessor 'org.projectlombok:lombok'  
    testImplementation 'org.springframework.boot:spring-boot-starter-test'  
}  
  
tasks.named('test') {  
    useJUnitPlatform()  
}
```

- application.properties
 - 경로 : 프로젝트/src/main/resource/application.properties
 - MySQL의 로컬 계정에서 dbp_univ 데이터베이스를 생성해야 함

```
spring.datasource.driver-class-name = com.mysql.cj.jdbc.Driver
spring.datasource.url = jdbc:mysql://localhost:3306/dbp_univ
spring.datasource.username = root
spring.datasource.password = 0000
spring.jpa.show-sql = true
spring.jpa.hibernate.ddl-auto = update
spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL8Dialect
```

데이터베이스 명

MySQL 계정

MySQL 비밀번호

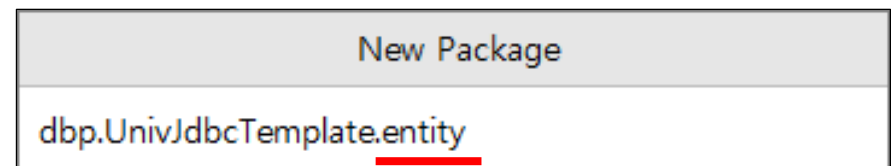
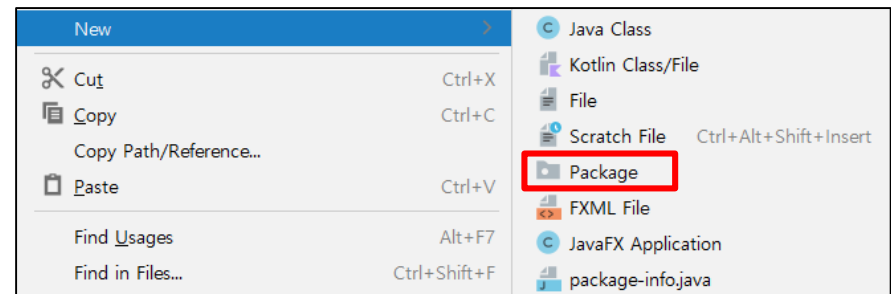
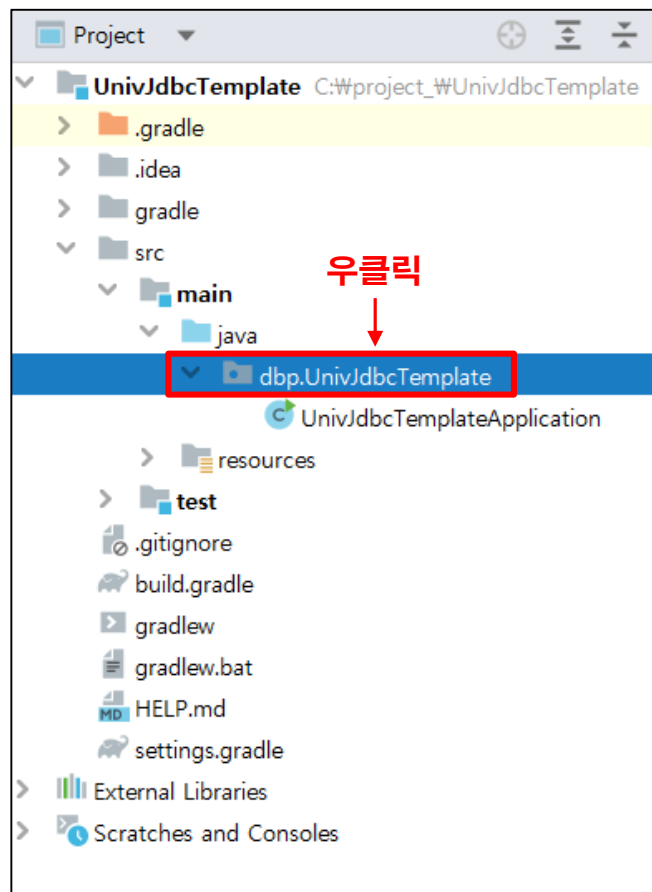
- CLUB

CLUB		
Column Name	Type	Key & Null Status
ID	Int	PRIMARY KEY
NAME	Varchar(10)	NOT NULL
CATEGORY	Varchar(20)	NOT NULL

```
mysql> SELECT * FROM club;
```

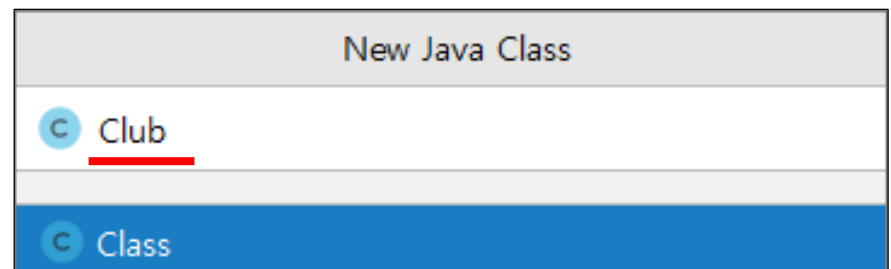
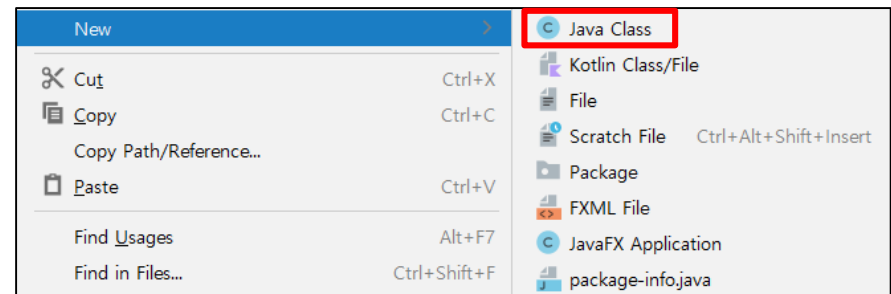
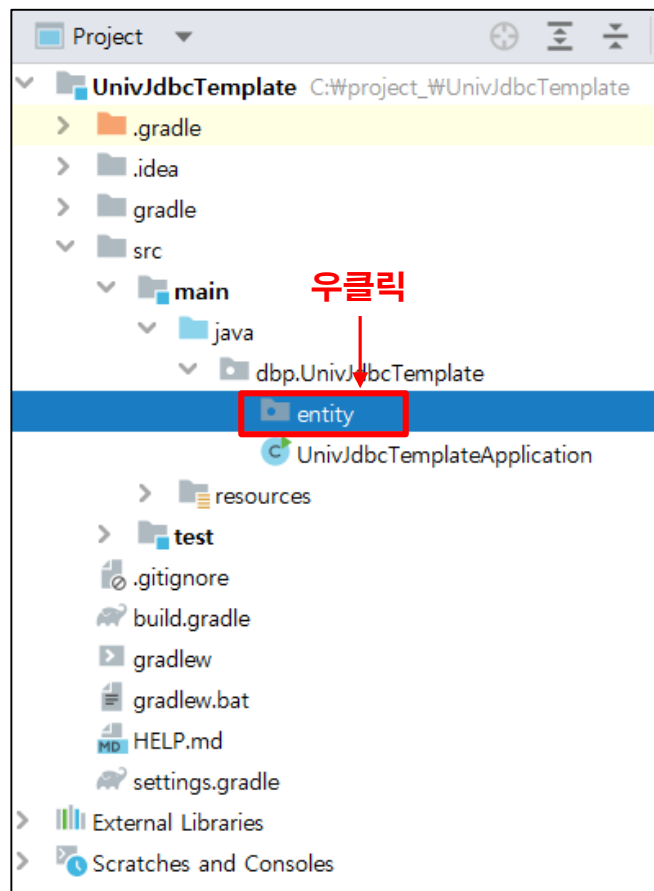
ID	NAME	CATEGORY
1	홍길동	등산
2	김영수	프로그래밍

- Package 생성
 - entity, repository, controller 생성



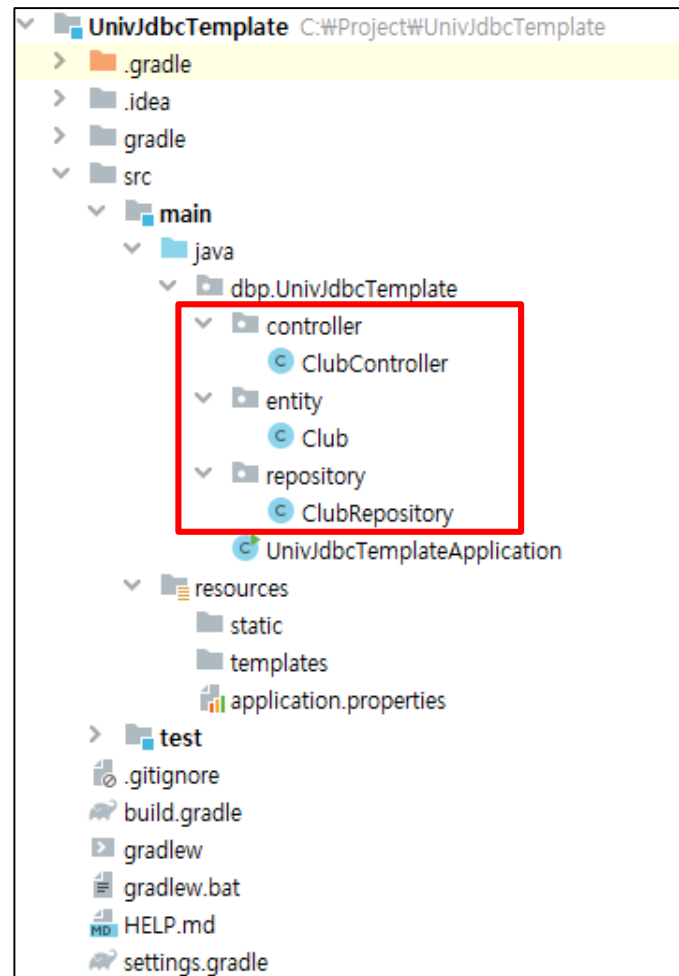
<Package명 입력>

- **Java Class 생성**
 - entity - Club.java, repository - ClubRepository, controller - ClubController.java 생성



〈Java Class명 입력〉

- 패키지 구성



- Club.java

```
package dbp.UnivJdbcTemplate.entity;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

@Entity
public class Club {
    @Id //기본키 매핑
    @GeneratedValue(strategy = GenerationType.IDENTITY) //기본키 생성
    private int id;
    private String name;
    private String category;

    public Club() {
    }

    public Club(int id, String name, String category) {
        this.id = id;
        this.name = name;
        this.category = category;
    }
}
```

- **Club.java**

```
public int getId() {  
    return id;  
}  
  
public void setId(int id) {  
    this.id = id;  
}  
  
public String getName() {  
    return name;  
}  
  
public void setName(String name) {  
    this.name = name;  
}  
  
public String getCategory() {  
    return category;  
}  
  
public void setCategory(String category) {  
    this.category = category;  
}  
}
```

- **ClubRepository.java**

```
package dbp.UnivJdbcTemplate.repository;

import dbp.UnivJdbcTemplate.entity.Club; //Club.java 경로
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.jdbc.core.JdbcTemplate;
import org.springframework.jdbc.core.RowMapper;
import org.springframework.stereotype.Repository;

import java.util.List;
import java.util.Optional;

@Repository
public class ClubRepository {
    @Autowired
    JdbcTemplate jdbcTemplate;

    // club 테이블 전체 검색
    public List<Club> getAllClubs(){
        List<Club> clubs = jdbcTemplate.query("select id, name, category from club",
            (result, rowNum)->new Club(result.getInt("id"), result.getString("name"),
                result.getString("category")));
        return clubs;
    }
}
```

- **ClubRepository.java**

//id로 검색

```
public Optional<Club> getClub(int id) {  
    List<Club> result = jdbcTemplate.query("select * from club where id = ?", clubRowMapper(), id);  
    return result.stream().findAny();  
}
```

//데이터 삽입

```
public int addClub(int id, String name, String category){  
    String query = "insert into club values(?,?,?)";  
    return jdbcTemplate.update(query,id,name,category);  
}
```

//데이터 삭제

```
public int deleteClub(int id){  
    String query = "delete from club where id =?";  
    return jdbcTemplate.update(query,id);  
}
```

```
private RowMapper<Club> clubRowMapper() {  
    return (rs, rowNum) -> {  
        Club club = new Club();  
        club.setId(rs.getInt("id"));  
        club.setName(rs.getString("name"));  
        club.setCategory(rs.getString("category"));  
        return club;  
    };  
}
```


- **ClubController.java**

```
package dbp.UnivJdbcTemplate.controller;

import dbp.UnivJdbcTemplate.entity.Club;    //Club.java 경로
import dbp.UnivJdbcTemplate.repository.ClubRepository; //ClubRepository.java 경로
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;

import java.util.List;
import java.util.Optional;

@RestController
@RequestMapping("/")
public class ClubController {
    @Autowired
    ClubRepository clubRepository;

    @GetMapping("/all")    //club 테이블 전체 검색
    public List<Club> getAllClubs(){
        return clubRepository.getAllClubs();
    }
}
```

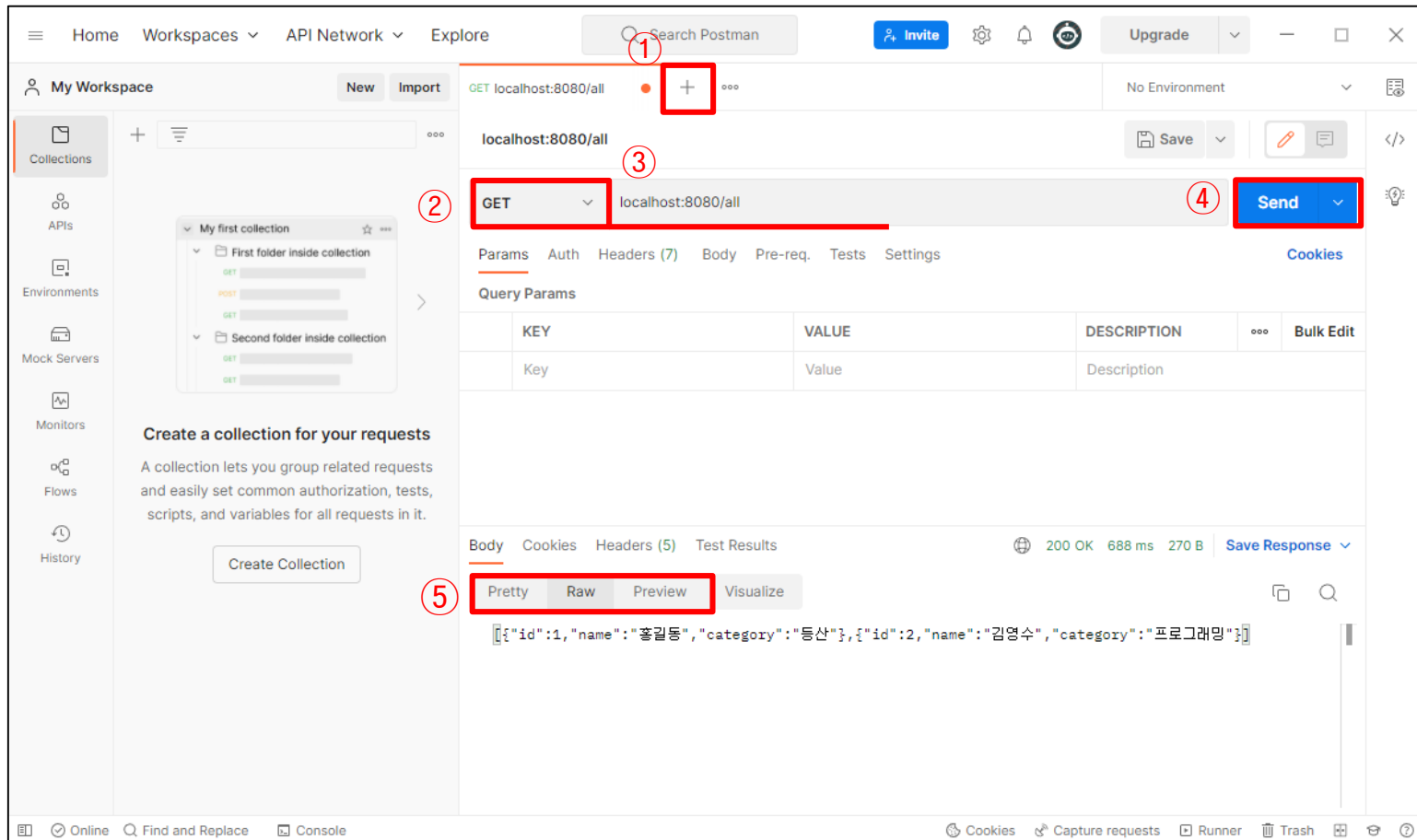
- ClubController.java

```
@GetMapping("/id/{id}")    //id로 검색
public Optional<Club> getClub(@PathVariable int id){
    return clubRepository.getClub(id);
}

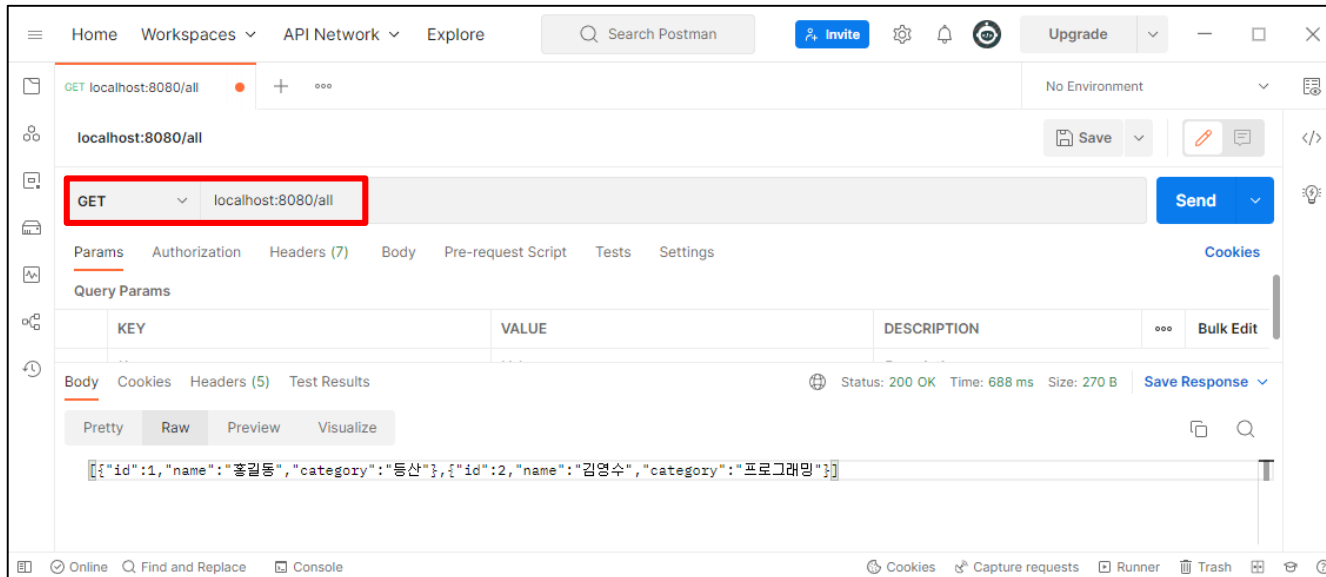
@GetMapping("/addClub")    //데이터 삽입
public String addClub(@RequestParam("id") int id,@RequestParam("name") String name,
                     @RequestParam("category") String category){
    if(clubRepository.addClub(id,name,category) >= 1){
        return "Club Added";
    }else{
        return "Not Added";
    }
}

@DeleteMapping("/delete/{id}")    //데이터 삭제
public String deleteClub(@PathVariable int id) {
    if(clubRepository.deleteClub(id) >= 1){
        return "Club Deleted";
    }else{
        return "Not Deleted";
    }
}
}
```

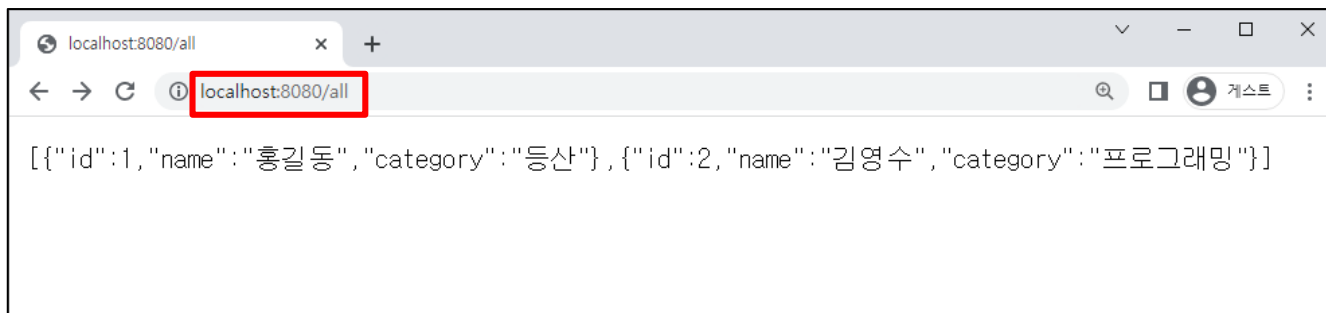
- 새로운 Tab 생성 → Method 방식 선택 → 주소 입력 → Send 클릭
→ View 방식 선택



- 전체 조회 - @GetMapping("/all")

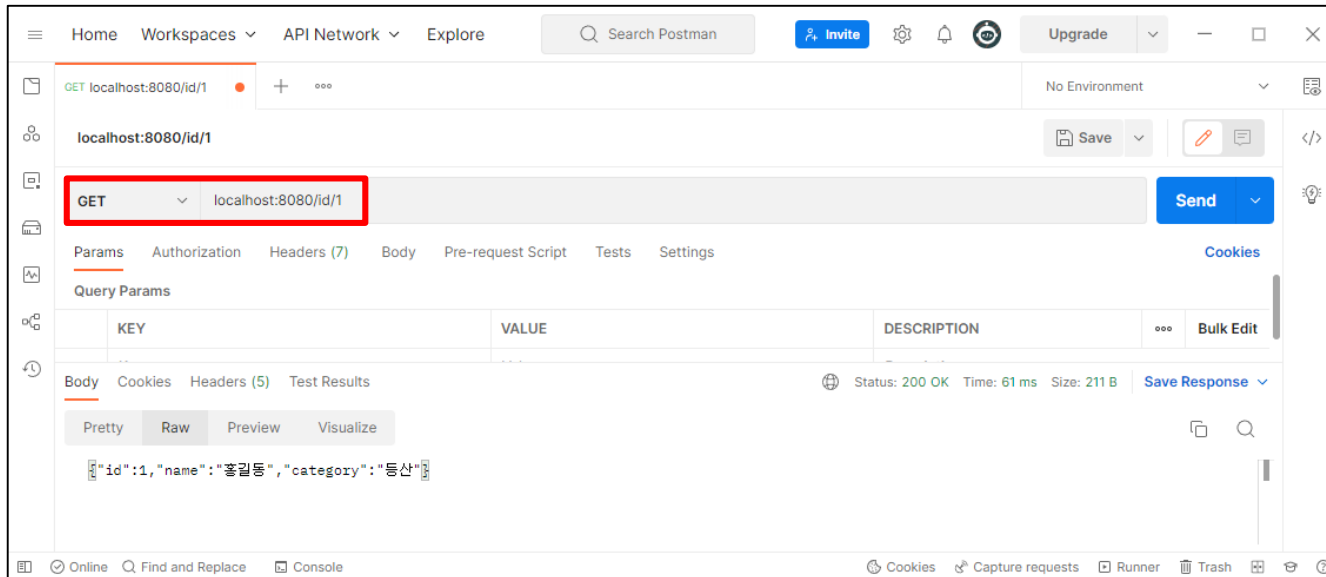


<Postman 실행 결과>

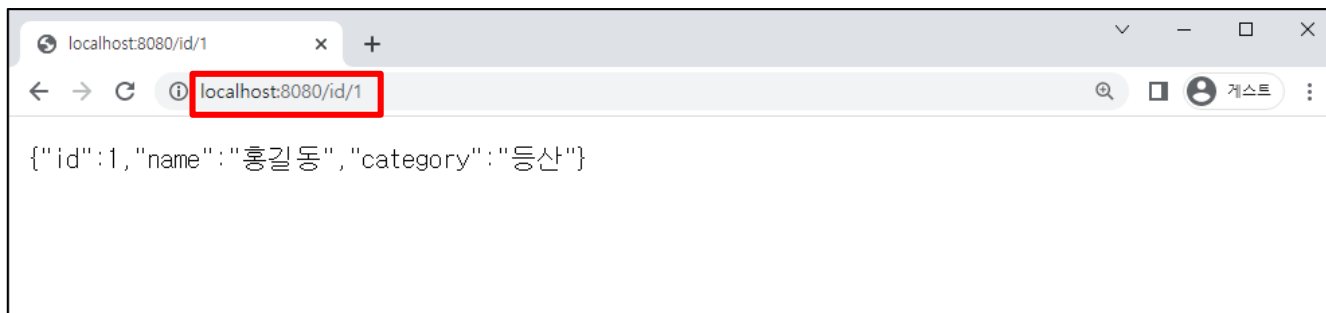


<Chrome 실행 결과>

- 일부 조회 - @GetMapping("/id/{id}")

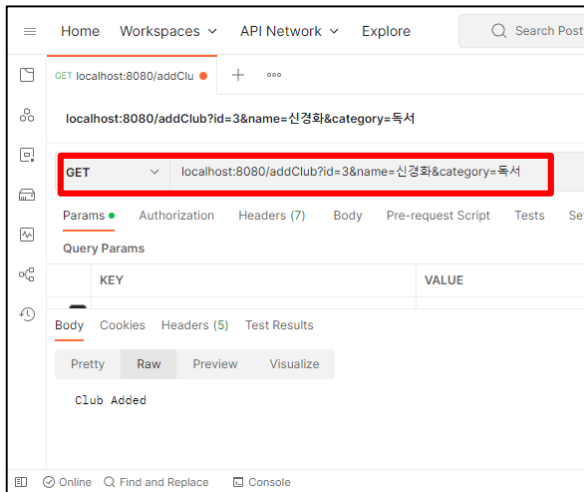


<Postman 실행 결과>



<Chrome 실행 결과>

- 삽입 - @GetMapping("/addClub")

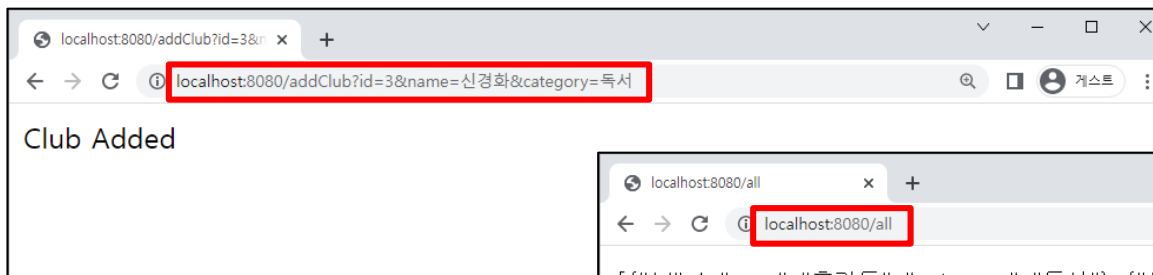


〈Postman 실행 결과〉

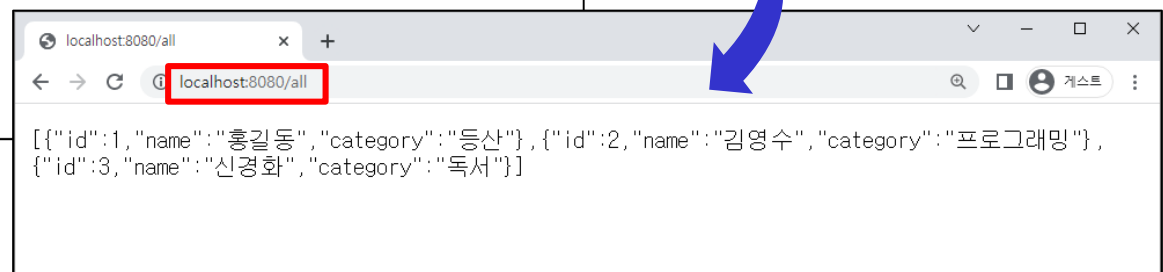
```
mysql> SELECT * FROM club;
```

ID	NAME	CATEGORY
1	홍길동	등산
2	김영수	프로그래밍
3	신경화	독서

〈CLUB TABLE〉

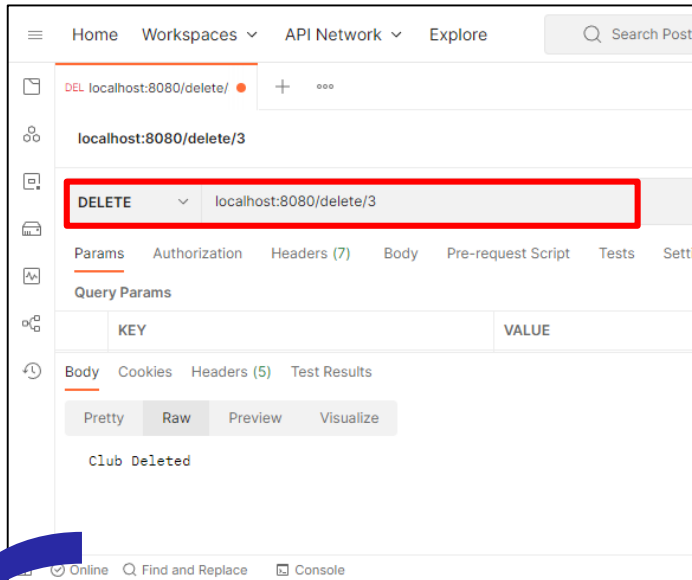


〈Chrome 실행 결과〉



〈삽입 후 전체 결과 조회〉

- 삭제 - @DeleteMapping("/delete/{id}")

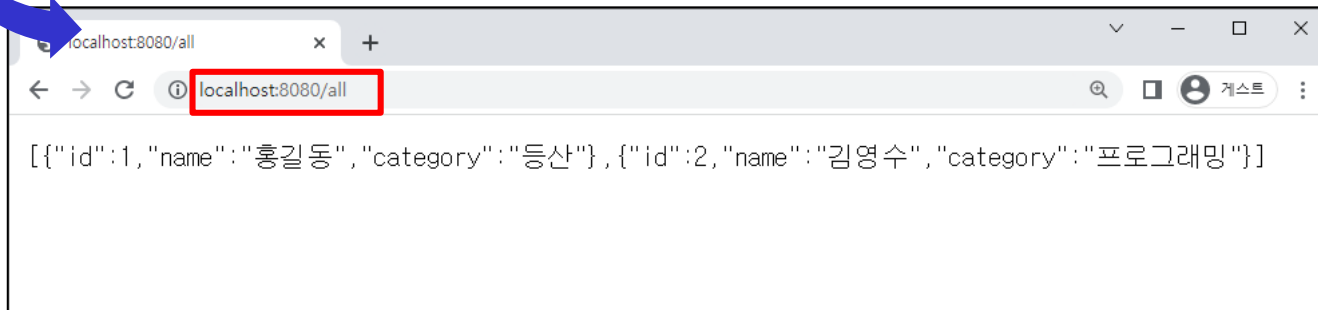


<Postman 실행 결과>

```
mysql> SELECT * FROM club;
```

ID	NAME	CATEGORY
1	홍길동	등산
2	김영수	프로그래밍

<CLUB TABLE>



<삭제 후 전체 결과 조회>

Q1> update.jsp를 Java Beans를 사용하여 작성하시오.
(새로운 Student.java, StudentMgr.java를 작성해야 함)

수강신청 사용자 정보 수정

← → ↻ ⓘ localhost:8080/update.jsp 게스트 ⋮

로그아웃	사용자 정보 수정	수강신청 입력	수강신청 삭제	수강신청 조회
이름	<input type="text" value="신경화"/>			
주소	<input type="text" value="경기도 화성군 송산면 고정1리 540-2"/>			
패스워드	<input type="password" value="...."/>			
학부	<input type="text" value="IT학부"/>			
전공	<input type="text" value="컴퓨터공학"/>			
<input type="button" value="수정"/>				

- 제출 방식 : E-Class를 통하여 제출
- 제출 내용 : ROOT 폴더 압축 파일(jsp, Java파일)
 - Apache Software Foundation\Tomcat\webapps\ROOT 폴더 압축
 - 기존에 작성하였던 수강신청 시스템 관련 파일이 존재해야 함
- 제출 형식 : 학번_이름_주차
 - Ex) 학번_홍길동_13주차.zip
- 제출 기한 : 수업 시작 시간으로 부터 24시간 이내 제출
 - 제출 기한 위반 시 감점 기준
 - 지각 제출 시 과제 점수에서 40% 감점
 - 1일 초과 당 20% 추가 감점 (단, 4일 이후 제출 불가)