회원관리 만들기

1. 기본 셑팅을 하자

- Dynamic Web Project 작성
- Convert To Tern Project 한다.
- Convert To Maven Project 한다.
- pom.xml파일을 수정한다.

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>kr.human</groupId>
    <artifactId>JSPMember</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <packaging>war</packaging>
    <build>
        <plugins>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-compiler-plugin</artifactId>
                <version>3.10.1</version>
                <configuration>
                    <source>1.8</source>
                    <target>1.8</target>
                </configuration>
            </plugin>
            <plugin>
                <artifactId>maven-war-plugin</artifactId>
                <version>3.2.3</version>
            </plugin>
        </plugins>
        <finalName>JSPMember</finalName>
    </build>
    <dependencies>
        <!-- test -->
        <dependency>
            <groupId>junit</groupId>
            <artifactId>junit</artifactId>
            <version>4.13.2</version>
        </dependency>
        <!-- 롬복 -->
        <dependency>
            <groupId>org.projectlombok</groupId>
            <artifactId>lombok</artifactId>
            <version>1.18.24</version>
        </dependency>
        <!-- JSON - Gson -->
        <dependency>
            <groupId>com.google.code.gson</groupId>
            <artifactId>gson</artifactId>
            <version>2.9.0</version>
```

```
        </dependency>
        <!-- HTML Parser -->
        <dependency>
            <groupId>org.jsoup</groupId>
            <artifactId>jsoup</artifactId>
            <version>1.14.3</version>
        </dependency>
        <!-- JAXB 추가 시작 -->
        <dependency>
            <groupId>javax.xml.bind</groupId>
            <artifactId>jaxb-api</artifactId>
            <version>2.3.1</version>
        </dependency>
        <dependency>
            <groupId>com.sun.xml.bind</groupId>
            <artifactId>jaxb-core</artifactId>
            <version>2.3.0</version>
        </dependency>
        <dependency>
            <groupId>com.sun.xml.bind</groupId>
            <artifactId>jaxb-impl</artifactId>
            <version>2.3.1</version>
        </dependency>
        <dependency>
            <groupId>com.googlecode.jslint4java</groupId>
            <artifactId>jslint4java</artifactId>
            <version>2.0.4</version>
        </dependency>
        <dependency>
            <groupId>javax.activation</groupId>
            <artifactId>activation</artifactId>
            <version>1.1.1</version>
        </dependency>
        <!-- JAXB 추가 종료 -->
        <!-- 마리아 DB JDBC 드라이버 -->
        <dependency>
            <groupId>org.mariadb.jdbc</groupId>
            <artifactId>mariadb-java-client</artifactId>
            <version>3.0.4</version>
        </dependency>
        <!-- MySQL -->
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <version>5.1.47</version>
        </dependency>
        <!-- webjars 사용하기 -->
        <dependency>
            <groupId>org.webjars</groupId>
            <artifactId>jquery</artifactId>
            <version>3.6.0</version>
        </dependency>
        <dependency>
            <groupId>org.webjars</groupId>
            <artifactId>bootstrap</artifactId>
            <version>5.1.3</version>
        </dependency>
        <!-- log4j2 시작 -->
```

```
        <dependency>
            <groupId>org.apache.logging.log4j</groupId>
            <artifactId>log4j-api</artifactId>
            <version>2.12.1</version>
        </dependency>
        <dependency>
            <groupId>org.apache.logging.log4j</groupId>
            <artifactId>log4j-core</artifactId>
            <version>2.12.1</version>
        </dependency>
        <!-- log4j2 종료 -->
        <!-- log4jdbc 시작 -->
        <dependency>
            <groupId>com.googlecode.log4jdbc</groupId>
            <artifactId>log4jdbc</artifactId>
            <version>1.2</version>
        </dependency>
        <dependency>
            <groupId>org.slf4j</groupId>
            <artifactId>slf4j-api</artifactId>
            <version>1.7.5</version>
        </dependency>
        <dependency>
            <groupId>org.slf4j</groupId>
            <artifactId>slf4j-simple</artifactId>
            <version>1.6.4</version>
        </dependency>
        <!-- log4jdbc 끝 -->
        <!-- 태그라이브러리 추가 -->
        <dependency>
            <groupId>jstl</groupId>
            <artifactId>jstl</artifactId>
            <version>1.2</version>
        </dependency>
        <!-- 오라클 -->
        <!-- ojdbc6.jar example -->
        <dependency>
            <groupId>com.oracle</groupId>
            <artifactId>ojdbc6</artifactId>
            <version>11.2.0.3</version>
        </dependency>
        <!-- ibatis 추가 -->
        <dependency>
            <groupId>org.apache.ibatis</groupId>
            <artifactId>ibatis-sqlmap</artifactId>
            <version>2.3.0</version>
        </dependency>
        <!-- Mybatis -->
        <dependency>
            <groupId>org.mybatis</groupId>
            <artifactId>mybatis</artifactId>
            <version>3.5.10</version>
        </dependency>
        <!-- commons-fileupload -->
        <dependency>
            <groupId>commons-fileupload</groupId>
            <artifactId>commons-fileupload</artifactId>
            <version>1.4</version>
```

```
        </dependency>
        <!-- commons-io -->
        <dependency>
            <groupId>commons-io</groupId>
            <artifactId>commons-io</artifactId>
            <version>2.2</version>
        </dependency>
        <!-- COS fileUpload -->
        <dependency>
            <groupId>servlets.com</groupId>
            <artifactId>cos</artifactId>
            <version>05Nov2002</version>
        </dependency>
        <!-- Sitemesh추가 -->
        <dependency>
            <groupId>org.sitemesh</groupId>
            <artifactId>sitemesh</artifactId>
            <version>3.0.1</version>
        </dependency>
        <!-- 메일 보내기 -->
        <!-- https://mvnrepository.com/artifact/javax.mail/mail -->
        <dependency>
            <groupId>javax.mail</groupId>
            <artifactId>mail</artifactId>
            <version>1.4</version>
        </dependency>
    </dependencies>
</project>
```

- Maven Update Project를 수행한다.
- src/main/resources 폴더에 4개의 파일을 복사해서 넣는다.

  db.properties

```
# Maria DB 연결정보
#m.driver=org.mariadb.jdbc.Driver
#m.url=jdbc:mariadb://localhost:3306/javadb
m.driver=net.sf.log4jdbc.DriverSpy
m.url=jdbc:log4jdbc:mysql://localhost:3306/jspdb
m.username=jspuser
m.password=123456

# Oracle DB 연결 정보
# o.driver=oracle.jdbc.driver.OracleDriver
# o.url=jdbc:oracle:thin:@127.0.0.1:1521:XE
# log4jdbc-remix
o.driver=net.sf.log4jdbc.DriverSpy
o.url=jdbc:log4jdbc:oracle:thin:@127.0.0.1:1521:XE
o.username=jspuser
o.password=123456
#o.username=scott
#o.password=tiger
```

mybatis-config.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration
  PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
    <!-- 프로퍼티 파일을 사용 하겠습니다. -->
    <properties resource="db.properties" />
    <!-- 타입의 별칭을 지정한다. 줄여서 사용하겠다. VO를 만들때마다 추가한다. -->
    <typeAliases>
        <package name="kr.human.member.vo"/>
    </typeAliases>
    <!-- DB연결 정보 -->
    <environments default="development">
        <environment id="development">
            <transactionManager type="JDBC" />
            <dataSource type="POOLED">
                <property name="driver" value="${o.driver}" />
                <property name="url" value="${o.url}" />
                <property name="username" value="${o.username}" />
                <property name="password" value="${o.password}" />
            </dataSource>
        </environment>
    </environments>

    <!-- SQL명령이 들어있는 매퍼파일을 지정한다. 맵퍼파일을 여러개 지정이 가능하다.-->
    <mappers>
        <mapper resource="testMapper.xml" />
    </mappers>
</configuration>
```

testMapper.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper
  PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="test">

    <select id="selectToday" resultType="string">
        select sysdate from dual
    </select>

</mapper>
```

log4j2.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<Configuration>
    <!-- Appender, Layout 설정 -->
    <Appenders>
        <Console name="console" target="SYSTEM_OUT">
            <PatternLayout pattern="%d %5p [%c] %m%n" />
        </Console>
        <File name="file" fileName="./logs/file/sample.log"
            append="false">
            <PatternLayout pattern="%d %5p [%c] %m%n" />
```

```
        </File>
    </Appenders>
    <!-- Logger 설정 -->
    <Loggers>
        <!-- SQL문만을 로그로 남기며, PreparedStatement일 경우 관련된 argument 값으로
대체된 SQL문이 보여진다. -->
        <logger name="jdbc.sqlonly">
            <level value="INFO" />
        </logger>
        <!-- SQL문과 해당 SQL을 실행시키는데 수행된 시간 정보(milliseconds)를 포함한다.
-->
        <logger name="jdbc.sqltiming">
            <level value="ERROR" />
        </logger>
        <!-- ResultSet을 제외한 모든 JDBC 호출 정보를 로그로 남긴다. 많은 양의 로그가 생
성되므로 특별히 JDBC 문제를
            추적해야 할 필요가 있는 경우를 제외하고는 사용을 권장하지 않는다. -->
        <logger name="jdbc.audit">
            <level value="ERROR" />
        </logger>
        <!-- ResultSet을 포함한 모든 JDBC 호출 정보를 로그로 남기므로 매우 방대한 양의
로그가 생성된다. -->
        <logger name="jdbc.resultsettable">
            <level value="ERROR" />
            <appender-ref ref="console" />
        </logger>
        <Root level="INFO">
            <AppenderRef ref="console" />
        </Root>
    </Loggers>
</Configuration>
```

- 4개의 패키지를 만든다.

kr.human.member.vo

kr.human.member.dao

kr.human.member.servirce

kr.human.mybatis

- kr.human.mybatis 패키지에 MybatisApp.java 파일을 만든다.

```
package kr.human.mybatis;

import java.io.IOException;
import java.io.InputStream;

import org.apache.ibatis.io.Resources;
import org.apache.ibatis.session.SqlSessionFactory;
import org.apache.ibatis.session.SqlSessionFactoryBuilder;


public class MybatisApp {
```

```
    // 1. 정적멤버로 팩토리 변수 선언하기
    private static SqlSessionFactory sqlSessionFactory;

    // 2. 정적변수를 초기화 하는 정적초기화 블럭을 만든다.
    static {
        String resource = "mybatis-config.xml";
        try {
            InputStream inputStream = Resources.getResourceAsStream(resource);
            sqlSessionFactory = new
SqlSessionFactoryBuilder().build(inputStream);
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    // 3. 외부에서 객체를 얻도록 public 메서드를 작성한다.
    public static SqlSessionFactory getSqlSessionFactory() {
        return sqlSessionFactory;
    }
}
```

- src/test/java폴더의 kr.human.mybatis 패키지에 MybatisAppTest.java 파일을 만든다.

```
package kr.human.mybatis;

import static org.junit.Assert.assertEquals;
import static org.junit.Assert.assertNotNull;

import java.sql.SQLException;

import org.apache.ibatis.session.SqlSession;
import org.apache.ibatis.session.SqlSessionFactory;
import org.junit.Before;
import org.junit.Test;

import kr.human.member.dao.MemberDAOImpl;

public class MybatisAppTest {

    static SqlSessionFactory sqlSessionFactory;

    @Before
    public void beforeClass() {
        sqlSessionFactory = MybatisApp.getSqlSessionFactory();
    }

    @Test
    public void getSession() {
        assertNotNull(MybatisApp.getSqlSessionFactory());
    }
}
```

- 위의 파일을 실행해서 에러가 없으면 기본 셑팅은 끝난 것이다.
- JSP로 테스트 해보자(mybatis.jsp파일을 src/main/webapp폴더에 만든다.)

```
<%@page import="kr.human.mybatis.MybatisApp"%>
<%@page import="org.apache.ibatis.session.SqlSession"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <%
        // 이 모양이 서비스클래스의 메서드의 내용이다.
        SqlSession sqlSession = null;
        try{
            sqlSession =
MybatisApp.getSqlSessionFactory().openSession(false);
            //-----------------------------------------------
            // 이 부분만 변경된다.
            String today = sqlSession.selectOne("test.selectToday");
            out.println("DB 날짜 : " + today + "<br><hr>");
            //-----------------------------------------------
            sqlSession.commit();
        }catch(Exception e){
            sqlSession.rollback();
            e.printStackTrace();
        }finally{
            sqlSession.close();
        }
    %>
</body>
</html>
```

위 파일을 실행해서 날짜가 보이면 DB연동이 잘 된것이다.

2. DB에 테이블을 만들자

```
CREATE SEQUENCE member_idx_seq;

CREATE TABLE member(
    idx NUMBER PRIMARY KEY,
    userid varchar2(100) NOT NULL UNIQUE,
    password varchar2(100) NOT NULL,
    name varchar2(50) NOT NULL,
    birth DATE NOT NULL,
    gender char(1) check(gender IN ('M','F')),
    email varchar2(100) NOT NULL,
    phone varchar2(20) NOT NULL,
    postcode varchar2(10) NOT NULL,
    addr1 varchar2(200) NOT NULL,
    addr2 varchar2(200) NOT NULL,
    use NUMBER check(use IN (0,1,2,3,4,5,6,7,8,9)),
    lev NUMBER check(lev IN (0,1,2,3,4,5,6,7,8,9))
);
INSERT INTO MEMBER VALUES
```

```
(member_idx_seq.nextval,'root','1234','최고관리자','1988-09-05','M'
,'ithuman202204@gmail.com','010-1234-5678',' ',' ',' ',1, 9);
INSERT INTO MEMBER VALUES
(member_idx_seq.nextval,'admin','1234','최고관리자','1988-09-05','M'
,'ithuman202204@gmail.com','010-1234-5678',' ',' ',' ',1, 9);
INSERT INTO MEMBER VALUES
(member_idx_seq.nextval,'master','1234','최고관리자','1988-09-05','M'
,'ithuman202204@gmail.com','010-1234-5678',' ',' ',' ',1, 9);
INSERT INTO MEMBER VALUES
(member_idx_seq.nextval,'webmaster','1234','최고관리자','1988-09-05','M'
,'ithuman202204@gmail.com','010-1234-5678',' ',' ',' ',1, 9);
INSERT INTO MEMBER VALUES
(member_idx_seq.nextval,'system','1234','최고관리자','1988-09-05','M'
,'ithuman202204@gmail.com','010-1234-5678',' ',' ',' ',1, 9);
INSERT INTO MEMBER VALUES
(member_idx_seq.nextval,'sys','1234','최고관리자','1988-09-05','M'
,'ithuman202204@gmail.com','010-1234-5678',' ',' ',' ',1, 9);

SELECT * FROM MEMBER;
```

3. VO클래스를 만들자.

MemberVO.java

```
package kr.human.member.vo;

import java.util.Date;
import lombok.Data;

@Data
public class MemberVO {
    private int idx;
    private String userid;
    private String password;
    private String name;
    private Date   birth;
    private String gender;
    private String email;
    private String phone;
    private String postCode;
    private String addr1;
    private String addr2;
    private int use;
    private int lev;
}
```

4. DAO를 만들자

- memberMapper.xml을 src/main/resources폴더 밑에 만들자.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper
  PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="member">

    <insert id="insert" parameterType="MemberVO">
    INSERT INTO MEMBER VALUES
    (member_idx_seq.nextval,#{userid},#{password},#{name},#{birth},#{gender}
    ,#{email},#{phone},#{postCode},#{addr1},#{addr2}, 0, 0)
    </insert>

    <update id="update" parameterType="MemberVO">
        update member set
            password=#{password},
            email = #{email},
            postCode = #{postCode},
            gender = #{gender},
            addr1 = #{addr1},
            addr2 = #{addr2}
        where
            idx=#{idx}
    </update>

    <delete id="delete" parameterType="int">
        delete from member where idx=#{idx}
    </delete>

    <select id="selectByIdx" parameterType="int" resultType="MemberVO">
        select * from member where idx=#{idx}
    </select>

    <select id="selectByUserid" parameterType="string" resultType="MemberVO">
        select * from member where userid=#{userid}
    </select>

    <select id="selectByName" parameterType="string" resultType="MemberVO">
        select * from member where name=#{name}
    </select>

    <select id="selectList" resultType="MemberVO">
        select * from member order by idx desc
    </select>

    <select id="selectUseridCount" parameterType="string" resultType="int">
        select count(*) from member where userid=#{userid}
    </select>

    <update id="updateUse" parameterType="hashmap">
        update member set use=#{use} where idx=#{idx}
    </update>

    <update id="updateLevel" parameterType="hashmap">
```

```
        update member set lev=#{lev} where idx=#{idx}
    </update>

    <update id="updatePassword" parameterType="hashmap">
        update member set password=#{password} where userid=#{userid}
    </update>
</mapper>
```

- mybatis-config.xml파일을 수정하자

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration
  PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
    <!-- 프로퍼티 파일을 사용 하겠습니다. -->
    <properties resource="db.properties" />
    <!-- 타입의 별칭을 지정한다. 줄여서 사용하겠다. VO를 만들때마다 추가한다. -->
    <typeAliases>
        <package name="kr.human.member.vo"/>
    </typeAliases>
    <!-- DB연결 정보 -->
    <environments default="development">
        <environment id="development">
            <transactionManager type="JDBC" />
            <dataSource type="POOLED">
                <property name="driver" value="${o.driver}" />
                <property name="url" value="${o.url}" />
                <property name="username" value="${o.username}" />
                <property name="password" value="${o.password}" />
            </dataSource>
        </environment>
    </environments>

    <!-- SQL명령이 들어있는 매퍼파일을 지정한다. 맵퍼파일을 여러개 지정이 가능하다.-->
    <mappers>
        <mapper resource="testMapper.xml" />
        <mapper resource="memberMapper.xml" /> <!-- 여기에 맵퍼파일 추가 -->
    </mappers>
</configuration>
```

- 이전에 만든 테스트파일을 실행 해서 에러가 없다면 XML은 일단 에러가 아니다.
- MemberDAO.java 인터페이스 파일을 만들자

```
package kr.human.member.dao;

import java.sql.SQLException;
import java.util.HashMap;
import java.util.List;

import org.apache.ibatis.session.SqlSession;

import kr.human.member.vo.MemberVO;

public interface MemberDAO {
```

```
    // 저장
    void insert(SqlSession sqlSession, MemberVO memberVO) throws SQLException;
    // 수정
    void update(SqlSession sqlSession, MemberVO memberVO) throws SQLException;
    // 삭제
    void delete(SqlSession sqlSession, int idx) throws SQLException;
    // 1개얻기(idx로 얻기)
    MemberVO selectByIdx(SqlSession sqlSession, int idx) throws SQLException;
    // 1개얻기(userid로 얻기)
    MemberVO selectByUserid(SqlSession sqlSession, String userid) throws
SQLException;

    // name로 얻기
    List<MemberVO> selectByName(SqlSession sqlSession, String name) throws
SQLException;

    // 모두얻기(관리자)
    List<MemberVO> selectList(SqlSession sqlSession) throws SQLException;
    // 동일한 아이디 개수 얻기(중복확인)
    int selectUseridCount(SqlSession sqlSession, String userid) throws
SQLException;

    // 인증정보 변경
    void updateUse(SqlSession sqlSession, HashMap<String, Integer> map) throws
SQLException;

    // 레벨 변경
    void updateLevel(SqlSession sqlSession, HashMap<String, Integer> map) throws
SQLException;

    // 비번 변경
    void updatePassword(SqlSession sqlSession, HashMap<String, String> map)
throws SQLException;
}
```

- MemberDAOImpl.java 파일을 만든다.

```
package kr.human.member.dao;

import java.sql.SQLException;
import java.util.HashMap;
import java.util.List;

import org.apache.ibatis.session.SqlSession;

import kr.human.member.vo.MemberVO;

public class MemberDAOImpl implements MemberDAO {
    private static MemberDAO instance = new MemberDAOImpl();
    private MemberDAOImpl() {}
    public static MemberDAO getInstance(){
        return instance;
    }
    //------------------------------------------------------------------------
----
    @Override
```

```
    public void insert(SqlSession sqlSession, MemberVO memberVO) throws
SQLException {
        sqlSession.insert("member.insert", memberVO);
    }
    @Override
    public void update(SqlSession sqlSession, MemberVO memberVO) throws
SQLException {
        sqlSession.update("member.update", memberVO);
    }
    @Override
    public void delete(SqlSession sqlSession, int idx) throws SQLException {
        sqlSession.delete("member.delete", idx);
    }
    @Override
    public MemberVO selectByIdx(SqlSession sqlSession, int idx) throws
SQLException {
        return sqlSession.selectOne("member.selectByIdx", idx);
    }
    @Override
    public MemberVO selectByUserid(SqlSession sqlSession, String userid) throws
SQLException {
        return sqlSession.selectOne("member.selectByUserid", userid);
    }
    @Override
    public List<MemberVO> selectList(SqlSession sqlSession) throws SQLException
{
        return sqlSession.selectList("member.selectList");
    }
    @Override
    public int selectUseridCount(SqlSession sqlSession, String userid) throws
SQLException {
        return sqlSession.selectOne("member.selectUseridCount", userid);
    }
    @Override
    public void updateUse(SqlSession sqlSession, HashMap<String, Integer> map)
throws SQLException {
        sqlSession.update("member.updateUse", map);
    }
    @Override
    public void updateLevel(SqlSession sqlSession, HashMap<String, Integer> map)
throws SQLException {
        sqlSession.update("member.updateLevel", map);
    }
    @Override
    public List<MemberVO> selectByName(SqlSession sqlSession, String name)
throws SQLException {
        return sqlSession.selectList("member.selectByName", name);
    }
    @Override
    public void updatePassword(SqlSession sqlSession, HashMap<String, String>
map) throws SQLException {
        sqlSession.update("member.updatePassword", map);
    }
}
```

- MybatisTest.java 파일을 수정하여 DAO각각의 메서드를을 테스트해 본다.

```java
package kr.human.mybatis;

import static org.junit.Assert.assertEquals;
import static org.junit.Assert.assertNotNull;

import java.sql.SQLException;

import org.apache.ibatis.session.SqlSession;
import org.apache.ibatis.session.SqlSessionFactory;
import org.junit.Before;
import org.junit.Test;

import kr.human.member.dao.MemberDAOImpl;

public class MybatisAppTest {

    static SqlSessionFactory sqlSessionFactory;

    @Before
    public void beforeClass() {
        sqlSessionFactory = MybatisApp.getSqlSessionFactory();
    }

    @Test
    public void getSession() {
        assertNotNull(MybatisApp.getSqlSessionFactory());
    }

    @Test
    public void selectByIdx() {
        SqlSession sqlSession = sqlSessionFactory.openSession();
        try {
            assertEquals(MemberDAOImpl.getInstance().selectByIdx(sqlSession,
1).getUserid(), "root");
            //assertEquals(MemberDAOImpl.getInstance().selectByIdx(sqlSession,
2).getName(), "최고관리자");
        } catch (SQLException e) {
            e.printStackTrace();
        } finally {
            sqlSession.close();
        }
    }

    @Test
    public void selectByUserid() {
        SqlSession sqlSession = sqlSessionFactory.openSession();
        try {
            assertEquals(MemberDAOImpl.getInstance().selectByUserid(sqlSession,
"root").getName(), "최고관리자");
        } catch (SQLException e) {
            e.printStackTrace();
        } finally {
            sqlSession.close();
        }
    }
}
```

4. Service를 만들자

- email을 발송해주는 서비스 클래스를 만들자(EmailService.java)

```
package kr.human.member.service;

import java.util.Properties;

import javax.mail.Address;
import javax.mail.Authenticator;
import javax.mail.Message;
import javax.mail.MessagingException;
import javax.mail.PasswordAuthentication;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.AddressException;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;

import lombok.extern.slf4j.Slf4j;

@Slf4j
public class EmailService {
    private static Session mailSession;

    static {
        Properties p = new Properties(); // 정보를 담을 객체
        p.put("mail.smtp.host","smtp.naver.com"); // 네이버 SMTP 또는 gmail SMTP
        p.put("mail.smtp.port", "465");
        p.put("mail.smtp.starttls.enable", "true");
        p.put("mail.smtp.auth", "true");
        p.put("mail.smtp.debug", "true");
        p.put("mail.smtp.socketFactory.port", "465");
        p.put("mail.smtp.socketFactory.class",
"javax.net.ssl.SSLSocketFactory");
        p.put("mail.smtp.socketFactory.fallback", "false");

        mailSession = Session.getInstance(p, new Authenticator(){
            protected PasswordAuthentication getPasswordAuthentication(){
                return new PasswordAuthentication("자기 네이버 계정@naver.com", "자
기비번");
            }
        });
    }

    public static void sendMail(String to, String subject, String content) {
        try {
            MimeMessage message = new MimeMessage(mailSession); // 메일의 내용을 담
을 객체
            Address fromAddress = new InternetAddress("자기 네이버 계정
@naver.com");
            message.setFrom(fromAddress);
            Address toAddress = new InternetAddress(to);     // 받는 사람
            message.addRecipient(Message.RecipientType.TO, toAddress);
            message.setSubject(subject);    // 제목
```

```
            message.setContent(content, "text/html;charset=UTF-8"); // 내용
            // 전송
            Transport.send(message);
            log.info("{}에게 메일전송 성공!!!", to);
        } catch (AddressException e) {
            log.info("에러발생 : {}", e.getMessage());
            e.printStackTrace();
        } catch (MessagingException e) {
            log.info("에러발생 : {}", e.getMessage());
            e.printStackTrace();
        }
    }
}
```

- email이 발송되는지 테스트해보자(MailServiceTest.java)

```
package kr.human.member.service;

public class MailServiceTest {
    public static void main(String[] args) {
        EmailService.sendMail("ithuman202204@gmail.com", "제목", "<h1>내용입니다.
</h1>");
    }
}
```

- 새로운 비밀번호를 무작위로 만들어 주는 서비스 클래스를 만들자.(PasswordService.java)

```
package kr.human.member.service;

import java.util.Random;

public class PasswordService {
    public static String makeNewPassword() {
        Random random  = new Random();
        String newPassword="";

        for(int i=0;i<10;i++) {
            switch (random.nextInt(4)) {
            case 0:
                newPassword += random.nextInt(10);
                break;
            case 1:
                newPassword += (char)('A'+ random.nextInt(26));
                break;
            case 2:
                newPassword += (char)('a'+ random.nextInt(26));
                break;
            case 3:
                newPassword += "~!@#$%^&*+-".charAt(random.nextInt(11));
            }
        }
        return newPassword;
    }
```

```
    // 실행해서 테스트해보기
    public static void main(String[] args) {
        for(int i=0;i<10;i++) {
            System.out.println(makeNewPassword());
        }
    }
}
```

- 인터페이스를 만들자!!!(MemberService.java)

```
package kr.human.member.service;

import java.util.List;

import javax.servlet.http.HttpSession;

import kr.human.member.vo.MemberVO;

public interface MemberService {
    // 저장(회원가입)
    void insert(MemberVO memberVO, String urlAddress);
    // 수정(정보수정)
    void update(MemberVO memberVO, String newPassword, HttpSession httpSession);
    // 삭제(회원탈퇴)
    void delete(MemberVO memberVO, HttpSession httpSession);
    // 목록보기(관리자)
    List<MemberVO> selectList();
    // 아이디 찾기
    MemberVO searchUserid(String name, String phone);
    // 비번 찾기
    MemberVO searchPassword(String userid, String phone);
    // 로그인
    boolean login(String userid, String password, HttpSession httpSession);
    // 로그아웃
    void logout();
    // 인증하기
    boolean emailConfirm(String userid);
    // 아이디 중복확인
    int idCheck(String userid);
}
```

- 인터페이스를 구현한 클래스를 만들자!!!(MemberServiceImpl.java)

```
package kr.human.member.service;

import java.sql.SQLException;
import java.util.HashMap;
import java.util.List;

import javax.servlet.http.HttpSession;

import org.apache.ibatis.session.SqlSession;

import kr.human.member.dao.MemberDAO;
import kr.human.member.dao.MemberDAOImpl;
import kr.human.member.vo.MemberVO;
```

```
import kr.human.mybatis.MybatisApp;
import lombok.extern.slf4j.Slf4j;

@Slf4j
public class MemberServiceImpl implements MemberService{
    private static MemberService instance = new MemberServiceImpl();
    private MemberServiceImpl() {}
    public static MemberService getInstance(){
        return instance;
    }
    //-------------------------------------------------------------------------
----
    @Override
    public void insert(MemberVO memberVO, String urlAddress) {
        log.info("MemberServiceImpl의 insert호출 : {}, {}", memberVO,
urlAddress);

        SqlSession sqlSession = null;
        MemberDAO memberDAO = null;
        try {
            sqlSession = MybatisApp.getSqlSessionFactory().openSession(false);
            memberDAO = MemberDAOImpl.getInstance();
            //-----------------------------------------------------------------
            if(memberVO!=null) {
                // DB에 저장을 하고
                memberDAO.insert(sqlSession, memberVO);
                // 환영이메일 발송
                EmailService.sendMail(memberVO.getEmail(),
                        memberVO.getName() + "님 회원가입을 환영합니다.",
                        "다음 링크를 클릭하여 인증을 하셔야만 회원 가입이 완료됩니다.
<br>"
                        +"<a href='" + urlAddress + "&userid=" +
memberVO.getUserid()+"'>인증하기</a>");
            }
            //-----------------------------------------------------------------
            sqlSession.commit();
        }catch (SQLException e) {
            sqlSession.rollback();
            e.printStackTrace();
        } finally {
            if(sqlSession!=null) sqlSession.close();
        }

    }
    @Override
    public void update(MemberVO memberVO, String newPassword, HttpSession
httpSession) {
        log.info("MemberServiceImpl의 update호출 : {}, {}", memberVO,
newPassword);

        SqlSession sqlSession = null;
        MemberDAO memberDAO = null;
        try {
            sqlSession = MybatisApp.getSqlSessionFactory().openSession(false);
            memberDAO = MemberDAOImpl.getInstance();
            //-----------------------------------------------------------------
            if(memberVO!=null) {
                // DB에서 해당 idx을 회원 정보를 가져온다.
```

```
                MemberVO dbVO = memberDAO.selectByIdx(sqlSession,
memberVO.getIdx());
                // 비번이 같으면
                if(dbVO!=null &&
dbVO.getPassword().equals(memberVO.getPassword())) {
                    // 변경을 수행한다.
                    memberVO.setPassword(newPassword); // 새로운 비번으로 바꿔서
                    memberDAO.update(sqlSession, memberVO);

                    // 세션의 값을 변경된 값으로 바꿔준다.
                    httpSession.setAttribute("memberVO", memberVO);

                }
            }
            //-------------------------------------------------------------------
            sqlSession.commit();
        }catch (SQLException e) {
            sqlSession.rollback();
            e.printStackTrace();
        } finally {
            if(sqlSession!=null) sqlSession.close();
        }
    }
    @Override
    public void delete(MemberVO memberVO, HttpSession httpSession) {
        log.info("MemberServiceImpl의 delete호출 : {}", memberVO);

        SqlSession sqlSession = null;
        MemberDAO memberDAO = null;
        try {
            sqlSession = MybatisApp.getSqlSessionFactory().openSession(false);
            memberDAO = MemberDAOImpl.getInstance();
            //-------------------------------------------------------------------
            if(memberVO!=null) {
                // DB에서 해당 userid의 회원 정보를 가져온다.
                MemberVO dbVO = memberDAO.selectByUserid(sqlSession,
memberVO.getUserid());
                // 비번이 같으면
                if(dbVO!=null &&
dbVO.getPassword().equals(memberVO.getPassword())) {
                    // 삭제를 수행한다.
                    // DB에서 바로 지우면 않됨
                    // use값에 탈퇴라고 저장을 하고 관리자 모드에서 탈퇴하고 6개월이 지나
면 일괄 삭제하는
                    // 코드를 만들어 주면 된다.
                    HashMap<String, Integer> map = new HashMap<String, Integer>
();
                    map.put("use", 3); // use 0이면 미인증 1이면 인증 2인면 휴면 3이
면 탈퇴라고 하자
                    map.put("idx", memberVO.getIdx());

                    memberDAO.updateUse(sqlSession, map);

                    // 세션의 값을 삭제해 준다.
                    httpSession.removeAttribute("memberVO");

                }
            }
```

```
            //-------------------------------------------------------------------
            sqlSession.commit();
        }catch (SQLException e) {
            sqlSession.rollback();
            e.printStackTrace();
        } finally {
            if(sqlSession!=null) sqlSession.close();
        }
    }
    @Override
    public List<MemberVO> selectList() {
        return null;
    }
    @Override
    public MemberVO searchUserid(String name, String phone) {
        log.info("MemberServiceImpl의 searchUserid호출 : {}, {}", name, phone);
        MemberVO memberVO = null;

        SqlSession sqlSession = null;
        MemberDAO memberDAO = null;
        try {
            sqlSession = MybatisApp.getSqlSessionFactory().openSession(false);
            memberDAO = MemberDAOImpl.getInstance();
            //-------------------------------------------------------------------
            // 해당 아이디의 회원 정보를 읽어온다
            List<MemberVO> list = memberDAO.selectByName(sqlSession, name);
            if(list!=null && list.size()>0) {
                for(MemberVO vo : list) {
                    if(vo.getPhone().equals(phone)) {
                        memberVO = vo;
                        break;
                    }
                }
            }
            //-------------------------------------------------------------------
            sqlSession.commit();
        }catch (SQLException e) {
            sqlSession.rollback();
            e.printStackTrace();
        } finally {
            if(sqlSession!=null) sqlSession.close();
        }

        log.info("MemberServiceImpl의 searchUserid리턴 : {}", memberVO);
        return memberVO;
    }
    @Override
    public MemberVO searchPassword(String userid, String phone) {
        log.info("MemberServiceImpl의 searchPassword호출 : {}, {}", userid,
phone);
        MemberVO memberVO = null;

        SqlSession sqlSession = null;
        MemberDAO memberDAO = null;
        try {
            sqlSession = MybatisApp.getSqlSessionFactory().openSession(false);
            memberDAO = MemberDAOImpl.getInstance();
            //-------------------------------------------------------------------
```

```
            // 해당 아이디의 회원 정보를 읽어온다
            memberVO = memberDAO.selectByUserid(sqlSession, userid);
            if(memberVO!=null && memberVO.getPhone().equals(phone)) {
                // 임시비번을 만들어서
                String newPassword = PasswordService.makeNewPassword();
                // DB의 비번을 변경하고
                HashMap<String, String> map = new HashMap<String, String>();
                map.put("userid", userid);
                map.put("password", newPassword);
                memberDAO.updatePassword(sqlSession, map);
                // 변경된 비번을 메일로 발송해 준다.
                EmailService.sendMail(memberVO.getEmail(),
                        userid + "님의 비밀번호 변경입니다.",
                        userid + "님의 임시 비밀번호입니다<br>"+
                        "임시비밀번호는 \""+newPassword+"\"입니다. <br>"+
                        "로그인 하신후 반드시 변경하시기 바랍니다.");
            }
            //-------------------------------------------------------------------
            sqlSession.commit();
        }catch (SQLException e) {
            sqlSession.rollback();
            e.printStackTrace();
        } finally {
            if(sqlSession!=null) sqlSession.close();
        }

        log.info("MemberServiceImpl의 searchPassword리턴 : {}", memberVO);
        return memberVO;

    }
    @Override
    public boolean login(String userid, String password, HttpSession
httpSession) {
        log.info("MemberServiceImpl의 login호출 : {}, {}", userid, password);
        boolean isLogin = false;

        SqlSession sqlSession = null;
        MemberDAO memberDAO = null;
        try {
            sqlSession = MybatisApp.getSqlSessionFactory().openSession(false);
            memberDAO = MemberDAOImpl.getInstance();
            //-------------------------------------------------------------------
            // 해당 아이디의 회원 정보를 읽어온다
            MemberVO memberVO = memberDAO.selectByUserid(sqlSession, userid);
            if(memberVO!=null) { // 해당 아이디의 회원정보가 있다면
                if(memberVO.getPassword().equals(password) &&
memberVO.getUse()==1) {
                    httpSession.setAttribute("memberVO", memberVO);
                    isLogin = true;
                }
            }
            //-------------------------------------------------------------------
            sqlSession.commit();
        }catch (SQLException e) {
            sqlSession.rollback();
            e.printStackTrace();
        } finally {
            if(sqlSession!=null) sqlSession.close();
```

```
        }

        log.info("MemberServiceImpl의 login리턴 : {}", isLogin);
        return isLogin;
    }
    @Override
    public void logout() {

    }
    @Override
    public boolean emailConfirm(String userid) {
        log.info("MemberServiceImpl의 emailConfirm호출 : {}", userid);
        boolean isConfirm = false;

        SqlSession sqlSession = null;
        MemberDAO memberDAO = null;
        try {
            sqlSession = MybatisApp.getSqlSessionFactory().openSession(false);
            memberDAO = MemberDAOImpl.getInstance();
            //-------------------------------------------------------------------
            // 해당 아이디의 회원 정보를 읽어온다
            MemberVO memberVO = memberDAO.selectByUserid(sqlSession, userid);
            if(memberVO!=null) { // 해당 아이디의 회원정보가 있다면
                // 인증값 변경
                // 레벨값 변경
                HashMap<String, Integer> map = new HashMap<String, Integer>();
                map.put("use", 1);
                map.put("lev", 1);
                map.put("idx", memberVO.getIdx());
                memberDAO.updateUse(sqlSession, map);
                memberDAO.updateLevel(sqlSession, map);
                isConfirm = true;
            }
            //-------------------------------------------------------------------
            sqlSession.commit();
        }catch (SQLException e) {
            sqlSession.rollback();
            e.printStackTrace();
        } finally {
            if(sqlSession!=null) sqlSession.close();
        }

        log.info("MemberServiceImpl의 emailConfirm리턴 : {}", isConfirm);
        return isConfirm;
    }
    @Override
    public int idCheck(String userid) {
        log.info("MemberServiceImpl의 idCheck호출 : {}", userid);
        int count =0;

        SqlSession sqlSession = null;
        MemberDAO memberDAO = null;
        try {
            sqlSession = MybatisApp.getSqlSessionFactory().openSession(false);
            memberDAO = MemberDAOImpl.getInstance();
            //-------------------------------------------------------------------
            count = memberDAO.selectUseridCount(sqlSession, userid);
            //-------------------------------------------------------------------
```

```
            sqlSession.commit();
        }catch (SQLException e) {
            sqlSession.rollback();
            e.printStackTrace();
        } finally {
            if(sqlSession!=null) sqlSession.close();
        }

        log.info("MemberServiceImpl의 idCheck리턴 : {}", count);
        return count;
    }
}
```

한번에 만들지 말고 서비스의 한개 메서드를 만들면 JSP파일을 바로 만들어 가면서 작업을 하자!!!

5. View를 구현해보자

- index.jsp파일을 만든다.

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
<%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <c:if test="${empty sessionScope.memberVO }">
        <a href="insertForm.jsp">회원가입</a>
        <a href="login.jsp">로그인</a>
    </c:if>
    <c:if test="${not empty sessionScope.memberVO }">
        ${sessionScope.memberVO.userid }(${sessionScope.memberVO.name })님 반갑습
니다 <br>

        <a href="updateForm.jsp">정보수정</a>
        <a href="deleteForm.jsp">회원탈퇴</a>
        <a href="logout.jsp">로그아웃</a>
    </c:if>
</body>
</html>
```

- 회원가입 폼을 만들자(insertForm.jsp)

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
```

```
<meta charset="UTF-8">
<title>회원가입</title>
<%-- 부트스트랩을 사용하기 위한 준비 시작 --%>
<meta name="viewport" content="width=device-width, initial-scale=1">
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-
datepicker/1.9.0/css/bootstrap-datepicker3.min.css">
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.j
s"></script>
<script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.10.2/dist/umd/popper.min.js"
></script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.min.js">
</script>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js">
</script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-
datepicker/1.9.0/js/bootstrap-datepicker.min.js"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/bootstrap-
datepicker/1.9.0/locales/bootstrap-datepicker.kr.min.js"></script>
<%-- 부트스트랩을 사용하기 위한 준비 끝 --%>
<%-- 다음 우편번호 API --%>
<script src="//t1.daumcdn.net/mapjsapi/bundle/postcode/prod/postcode.v2.js">
</script>
<script type="text/javascript">
$(function(){
    $("#birth").datepicker({
        format : "yyyy-mm-dd", // 달력에서 클릭시 표시할 값 형식
        language : "kr", // 언어(js 추가가 필요하다.)
        startDate : "-30y",
    });
    // 아이디 입력시 Ajax를 호출하여 아이디 중복확인하기
    $("#userid").keyup(function() {
        var value = $(this).val();
        if(value!=null && value.length>=4){
            // alert(value);
            $.ajax("idCheck.jsp", {
                type : "GET",
                data : {"userid":value},
                success : function(data){
                    //alert(typeof(data) + "\n" + data )
                    if(data*1==0){
                        $("#msg").html("사용가능").css('color','blue');
                    }else{
                        $("#msg").html("사용불가능").css('color','red');
                    }
                },
                error : function(){
                    alert('에러다!!!!')
                }
            });
        }else{
            $("#msg").html("").css('color','black');
        }
```

```
    });
});

function daumPostcode() {
    new daum.Postcode({
        oncomplete: function(data) {
            var addr = ''; // 주소 변수

            //사용자가 선택한 주소 타입에 따라 해당 주소 값을 가져온다.
            if (data.userSelectedType === 'R') { // 사용자가 도로명 주소를 선택했을
경우
                addr = data.roadAddress;
            } else { // 사용자가 지번 주소를 선택했을 경우(J)
                addr = data.jibunAddress;
            }
            // 우편번호와 주소 정보를 해당 필드에 넣는다.
            document.getElementById('postCode').value = data.zonecode;
            document.getElementById("addr1").value = addr;
            // 커서를 상세주소 필드로 이동한다.
            document.getElementById("addr2").focus();
        }
    }).open();
}

// 폼검증하는 자바스크립트 함수
function formCheck(){
    var value = $("#userid").val();
    if(value==null || value.trim().length==0){
        alert('아이디는 반드시 입력해야 합니다.');
        $("#userid").val("");
        $("#userid").focus();
        return false;
    }
    if($("#msg").html()!="사용가능"){
        alert('사용 불가능한 아이디입니다.');
        $("#userid").val("");
        $("#msg").text("");
        $("#userid").focus();
        return false;
    }
    var value = $("#password").val();
    if(value==null || value.trim().length==0){
        alert('사용자 비빌번호는 반드시 입력해야 합니다.');
        $("#password").val("");
        $("#password").focus();
        return false;
    }
    var value = $("#name").val();
    if(value==null || value.trim().length==0){
        alert('사용자 이름은 반드시 입력해야 합니다.');
        $("#name").val("");
        $("#name").focus();
        return false;
    }
    var value = $("#email").val();
    if(value==null || value.trim().length==0){
        alert('이메일 주소는 반드시 입력해야 합니다.');
        $("#email").val("");
```

```
        $("#email").focus();
        return false;
    }
    var regEmail = /^[0-9a-zA-Z]([-_\.]?[0-9a-zA-Z])*@[0-9a-zA-Z]([-_\.]?[0-9a-
zA-Z])*\.[a-zA-Z]{2,3}$/;
    if (!regEmail.test(value)) {
        alert('이메일 주소 형식이 잘못되었습니다.');
        $("#email").val("");
        $("#email").focus();
        return false;
    }
    var value = $("#birth").val();
    if(value==null || value.trim().length==0){
        alert('생년월일은 반드시 입력해야 합니다.');
        $("#birth").val("");
        $("#birth").focus();
        return false;
    }
    var value = $("#phone").val();
    if(value==null || value.trim().length==0){
        alert('전화번호는 반드시 입력해야 합니다.');
        $("#phone").val("");
        $("#phone").focus();
        return false;
    }
    var value = $("#postCode").val();
    if(value==null || value.trim().length==0){
        alert('우편번호는 반드시 입력해야 합니다.');
        $("#postCode").val("");
        $("#postCode").focus();
        return false;
    }
    var value = $("#addr2").val();
    if(value==null || value.trim().length==0){
        alert('상세 주소는 반드시 입력해야 합니다.');
        $("#addr2").val("");
        $("#addr2").focus();
        return false;
    }
}
</script>

<style type="text/css">
    .title { font-size: 18pt;text-align: center; padding: 10px; font-weight:
bold;}
</style>
</head>
<body>
    <div class="container" style="border: 1px solid gray;padding: 15px;margin-
top: 30px;border-radius: 30px;">
        <form action="insertOk.jsp" method="post" onsubmit="return
formCheck();">
            <div class="title" >회원가입하기</div>
            <div class="mb-3 row">
                <label for="userid" class="col-sm-2 col-form-label">사용자 아이디
</label>
                <div class="col-sm-2">
```

```
                <input type="text" class="form-control" id="userid"
name="userid" placeholder="아이디입력" required>
                </div>
                <div class="col-sm-1 col-form-label" id="msg"></div>
                <div class="col-sm-2"></div>
                <label for="password" class="col-sm-2 col-form-label">사용자 비밀
번호</label>
                <div class="col-sm-3">
                <input type="password" class="form-control" id="password"
name="password" placeholder="비밀번호입력"  required>
                </div>
            </div>
            <div class="mb-3 row">
                <label for="name" class="col-sm-2 col-form-label">사용자 이름
</label>
                <div class="col-sm-3">
                <input type="text" class="form-control" id="name" name="name"
placeholder="사용자 이름 입력" required>
                </div>
                <div class="col-sm-2"></div>
                <label for="birth" class="col-sm-2 col-form-label">생년월일
</label>
                <div class="col-sm-3">
                <input type="text" class="form-control" id="birth" name="birth1"
placeholder="사용자 생년월일 입력" required>
                </div>
            </div>
            <div class="mb-3 row">
                <label for="email" class="col-sm-2 col-form-label">사용자 이메일
</label>
                <div class="col-sm-3">
                <input type="email" class="form-control" id="email" name="email"
placeholder="사용자 이메일 입력" required>
                </div>
                <div class="col-sm-2"></div>
                <label for="phone" class="col-sm-2 col-form-label">사용자 전화번호
</label>
                <div class="col-sm-3">
                <input type="tel" class="form-control" id="phone" name="phone"
placeholder="사용자 전화번호 입력" required>
                </div>
            </div>
            <div class="mb-3 row">
                <label class="col-sm-2 col-form-label">성별</label>
                <div class="col-sm-2">
                  <label for="gender1" class="col-sm-2 col-form-label">남자
</label>
                  <input class="form-check-input" type="radio" name="gender"
id="gender1" value="M" checked>
                </div>
                <div class="col-sm-2">
                  <label for="gender2" class="col-sm-2 col-form-label">여자
</label>
                  <input class="form-check-input" type="radio" name="gender"
id="gender2" value="F">
                 </div>
            </div>
            <div class="mb-3 row">
```

```
                <label class="col-sm-2 col-form-label" for="postCode">우편번호
</label>
                <div class="col-sm-2">
                  <input class="form-control" type="text" name="postCode"
id="postCode" readonly  required onclick="daumPostcode();">
                </div>
                <div class="col-sm-1"></div>
                <div class="col-sm-2">
                  <input type="button" class="btn-check" id="zipCodebtn"
onclick="daumPostcode();">
                  <label class="btn btn-outline-primary" for="zipCodebtn">우편번호
찾기</label>
                </div>
            </div>
            <div class="mb-3 row">
                <label class="col-sm-2 col-form-label" for="addr1">주소</label>
                <div class="col-sm-10">
                  <input class="form-control" type="text" name="addr1"
id="addr1" readonly  required onclick="daumPostcode();">
                </div>
            </div>
            <div class="mb-3 row">
                <label class="col-sm-2 col-form-label" for="addr2">상세주소
</label>
                <div class="col-sm-10">
                  <input class="form-control" type="text" name="addr2"
id="addr2"  required>
                </div>
            </div>
            <div class="mb-3 row">
                <div class="col-sm-12" style="text-align: right;">
                      <input type="submit" class="btn-check" id="submitBtn" >
                      <label class="btn btn-outline-success" for="submitBtn">회원
가입</label>
                      <input type="reset" class="btn-check" id="resetBtn" >
                      <label class="btn btn-outline-success" for="resetBtn">다시
쓰기</label>
                      <input type="button" class="btn-check" id="cancelBtn"
onclick="location.href='${pageContext.request.contextPath}'">
                      <label class="btn btn-outline-success" for="cancelBtn">돌아
가기</label>
                </div>
            </div>
        </form>
    </div>
</body>
</html>
```

- 회원 가입을 완료하는 뷰를 만들자.(insertOk.jsp)

```
<%@page import="java.util.UUID"%>
<%@page import="kr.human.member.service.MemberServiceImpl"%>
<%@page import="java.text.SimpleDateFormat"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<% request.setCharacterEncoding("UTF-8"); %>
<!DOCTYPE html>
```

```
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <jsp:useBean id="memberVO" class="kr.human.member.vo.MemberVO"/>
    <jsp:setProperty property="*" name="memberVO"/>
    <%
        // String 을 날짜 형식으로 변경하여 넣는다.
        String birth = request.getParameter("birth1");
        SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
        memberVO.setBirth(sdf.parse(birth));
    %>
    <%//=memberVO %>
    <%
        // 서비스를 호출하여 저장을 하고
        String urlAddress = "http://"+request.getServerName()+":" +
                    request.getServerPort()+ request.getContextPath() +
"/confirm.jsp?fdagd=" + UUID.randomUUID();
        MemberServiceImpl.getInstance().insert(memberVO, urlAddress);
        // 어디론가 간다.
        out.println(memberVO.getEmail() + "로 인증메일이 발송되었습니다. 인증을 진행하
시고 로그인하시기 바랍니다.<br>");
        out.println("<a href='" + request.getContextPath() + "'>홈으로</a><br>"
);
        /*
        out.println(request.getRequestURI() + "<br>" );
        out.println(request.getRequestURL() + "<br>" );
        out.println(request.getContextPath() + "<br>" );
        out.println(request.getServerName() + "<br>" );
        out.println("http://"+request.getServerName()+":" +
                    request.getServerPort()+ request.getContextPath() +
"/confirm.jsp<br>" );
        */
    %>
</body>
</html>
```

- 회원가입을 완료하면 메일이 발송된다. 이메일을 확인하면 인증을 하는 링크가 보이는데 인증을 처리하는 파일을 만들어 보자.(confirm.jsp)

```
<%@page import="kr.human.member.service.MemberServiceImpl"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
<%
    // 사용자 아이디를 읽는다.
    String userid = request.getParameter("userid");
    // 서비스를 호출하여 해당 아이디의 use(인증), lev(권한)값을 변경해주고
    boolean isConfirm = false;
```

```
    if(userid!=null){
        isConfirm = MemberServiceImpl.getInstance().emailConfirm(userid);
    }

%>
<% if(isConfirm){ %>
    <h2>반갑습니다. <%=userid %>님 인증에 성공하셨습니다.</h2>
    <h2>즐거운 시간되시기 바랍니다.</h2>
    <a href="${pageContext.request.contextPath }">홈</a>
    <a href="${pageContext.request.contextPath }/login.jsp">로그인</a>
<% }else{ %>
    <h2><%=userid %>님 인증에 실패하셨습니다.</h2>
    <h2>장난치시면 죽어요~~~~</h2>
    <a href="${pageContext.request.contextPath }">홈</a>
    <a href="${pageContext.request.contextPath }/insertForm.jsp">회원가입</a>
<% } %>
</body>
</html>
```

- 로그인 폼을 만들자.(login.jsp)

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Login page</title>
<link rel="stylesheet"

href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css">
<script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
<script
    src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js">
</script>
<script

src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js">
</script>

<script src="https://kit.fontawesome.com/3c36eed32b.js" ></script>

<link rel="stylesheet" type="text/css"
    href="${pageContext.request.contextPath }/css/app.css" />
</head>

<body>
    <div id="mainWrapper">
        <div class="login-container">
            <div class="login-card">
                <div class="login-form">
                    <form action="${pageContext.request.contextPath
}/loginOk.jsp" method="post" class="form-horizontal">
                        <%-- 로그인 실패시 에러메세지 출력 --%>
```

```
                        <c:if test="${not empty error }">
                            <div style="color: red;font-size: 15pt;">${error }
</div>
                        </c:if>
                        <%-- 로그아웃시 메세지 출력 --%>
                        <c:if test="${not empty msg }">
                            <div style="color: green;font-size: 15pt;">${msg }
</div>
                        </c:if>

                        <div class="input-group input-sm">
                            <label class="input-group-addon" for="username">
                            <i class="fa-solid fa-user" style="font-size:
20pt;margin-right: 5px;color:green;"></i>
                            </label> <input type="text" class="form-control"
                                id="username" name="userid" placeholder="Enter
Username" required>
                        </div>
                        <div class="input-group input-sm">
                            <label class="input-group-addon" for="password"><i
class="fa-solid fa-lock" style="font-size: 20pt;margin-right: 5px;color:red;">
</i></label> <input type="password"
                                class="form-control" id="password"
name="password"
                                placeholder="Enter Password" required>
                        </div>
                        <div class="form-actions">
                            <input type="submit"
                                class="btn btn-block btn-primary btn-default"
value="Log in">
                        </div>
                        <div style="text-align: center;margin: 15px;">
                            [<a href="findUserid.jsp">아이디찾기</a>]
                               
                            [<a href="findPassword.jsp">비밀번호찾기</a>]
                               
                            [<a href="${pageContext.request.contextPath }">홈으로
</a>]
                        </div>
                    </form>
                </div>
            </div>
        </div>
    </div>

</body>
</html>
```

- 로그인을 완료하는 파일을 만들자.(loginOk.jsp)

```
<%@page import="kr.human.member.service.MemberServiceImpl"%>
<%@page import="kr.human.member.service.MemberService"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
```

```
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <%
        if(request.getMethod().equals("GET")){
            response.sendRedirect(request.getContextPath());
            return;
        }
        String userid = request.getParameter("userid");
        String password = request.getParameter("password");

        if(userid!=null && password!=null){
            // 서비스를 호출하여 로그인 처리를 한다.
            boolean isLogin = MemberServiceImpl.getInstance().login(userid,
password, session);
            if(isLogin){
                response.sendRedirect(request.getContextPath());
                return;
            }else{
                request.setAttribute("error", "잘못된 정보입니다.");
                pageContext.forward("login.jsp");
            }
        }else{
            request.setAttribute("error", "잘못된 정보입니다.");
            pageContext.forward("login.jsp");
            return;
        }
    %>
</body>
</html>
```

- 로그아웃을 처리하는 파일을 만들자.(logout.jsp)

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <%
        if(session.getAttribute("memberVO")!=null){
            session.removeAttribute("memberVO");
        }
        response.sendRedirect(request.getContextPath());
    %>
</body>
</html>
```

- 회원 정보를 수정하는 폼을 만들자.(updateForm.jsp)

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
```

```
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%
    // 로그인이 되어있지 않으면 여기는 진입 불가
    if(session.getAttribute("memberVO")==null){
        response.sendRedirect(request.getContextPath());
        return;
    }
%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>회원 정보 수정</title>
<%-- 부트스트랩을 사용하기 위한 준비 시작 --%>
<meta name="viewport" content="width=device-width, initial-scale=1">
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet">
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.j
s"></script>
<script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.10.2/dist/umd/popper.min.js"
></script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.min.js">
</script>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js">
</script>
<%-- 부트스트랩을 사용하기 위한 준비 끝 --%>
<%-- 다음 우편번호 API --%>
<script src="//t1.daumcdn.net/mapjsapi/bundle/postcode/prod/postcode.v2.js">
</script>
<script type="text/javascript">
$(function(){

});

function daumPostcode() {
    new daum.Postcode({
        oncomplete: function(data) {
            var addr = ''; // 주소 변수

            //사용자가 선택한 주소 타입에 따라 해당 주소 값을 가져온다.
            if (data.userSelectedType === 'R') { // 사용자가 도로명 주소를 선택했을
경우
                addr = data.roadAddress;
            } else { // 사용자가 지번 주소를 선택했을 경우(J)
                addr = data.jibunAddress;
            }
            // 우편번호와 주소 정보를 해당 필드에 넣는다.
            document.getElementById('postCode').value = data.zonecode;
            document.getElementById("addr1").value = addr;
            // 커서를 상세주소 필드로 이동한다.
            document.getElementById("addr2").focus();
        }
    }).open();
}
```

```
// 폼검증하는 자바스크립트 함수
function formCheck(){
    var value = $("#password").val();
    if(value==null || value.trim().length==0){
        alert('사용자 비밀번호는 반드시 입력해야 합니다.');
        $("#password").val("");
        $("#password").focus();
        return false;
    }
    var value = $("#name").val();
    if(value==null || value.trim().length==0){
        alert('사용자 이름은 반드시 입력해야 합니다.');
        $("#name").val("");
        $("#name").focus();
        return false;
    }
    var value = $("#email").val();
    if(value==null || value.trim().length==0){
        alert('이메일 주소는 반드시 입력해야 합니다.');
        $("#email").val("");
        $("#email").focus();
        return false;
    }
    var value = $("#phone").val();
    if(value==null || value.trim().length==0){
        alert('전화번호는 반드시 입력해야 합니다.');
        $("#phone").val("");
        $("#phone").focus();
        return false;
    }
    var value = $("#postCode").val();
    if(value==null || value.trim().length==0){
        alert('우편번호는 반드시 입력해야 합니다.');
        $("#postCode").val("");
        $("#postCode").focus();
        return false;
    }
    var value = $("#addr2").val();
    if(value==null || value.trim().length==0){
        alert('상세 주소는 반드시 입력해야 합니다.');
        $("#addr2").val("");
        $("#addr2").focus();
        return false;
    }
}
</script>

<style type="text/css">
    .title { font-size: 18pt;text-align: center; padding: 10px; font-weight:
bold;}
</style>
</head>
<body>
    <div class="container" style="border: 1px solid gray;padding: 15px;margin-
top: 30px;border-radius: 30px;">
        <form action="updateOk.jsp" method="post" onsubmit="return
formCheck();">
```

```
            <div class="title" >회원 정보 수정</div>
            <div class="mb-3 row">
                <%-- idx는 숨겨서 넘기자 --%>
                <input type="hidden" name="idx" value="${memberVO.idx }" />

                <label for="userid" class="col-sm-2 col-form-label">사용자 아이디
</label>
                <div class="col-sm-2">
                <input type="text" class="form-control" id="userid"
name="userid" readonly value="${memberVO.userid }">
                </div>
                <div class="col-sm-1 col-form-label" id="msg"></div>
                <div class="col-sm-2"></div>
                <label for="password" class="col-sm-2 col-form-label">사용자 비밀
번호</label>
                <div class="col-sm-3">
                <input type="password" class="form-control" id="password"
name="password" placeholder="비밀번호입력"  required>
                </div>
            </div>
            <div class="mb-3 row">
                <label for="name" class="col-sm-2 col-form-label">사용자 이름
</label>
                <div class="col-sm-3">
                <input type="text" class="form-control" id="name" name="name"
value="${sessionScope.memberVO.name }" required>
                </div>
                <div class="col-sm-2"></div>
                <label for="newPassword" class="col-sm-2 col-form-label">새로운 비
번</label>
                <div class="col-sm-3">
                <input type="password" class="form-control" id="newPassword"
name="newPassword">
                </div>
            </div>
            <div class="mb-3 row">
                <label for="email" class="col-sm-2 col-form-label">사용자 이메일
</label>
                <div class="col-sm-3">
                <input type="email" class="form-control" id="email" name="email"
value="${sessionScope.memberVO.email }"  readonly>
                </div>
                <div class="col-sm-2"></div>
                <label for="phone" class="col-sm-2 col-form-label">사용자 전화번호
</label>
                <div class="col-sm-3">
                <input type="tel" class="form-control" id="phone" name="phone"
value="${sessionScope.memberVO.phone }"  required>
                </div>
            </div>
            <div class="mb-3 row">
                <label class="col-sm-2 col-form-label">성별</label>
                <div class="col-sm-2">
                  <label for="gender1" class="col-sm-2 col-form-label">남자
</label>
                  <input class="form-check-input" type="radio" name="gender"
id="gender1"  value="M" ${memberVO.gender=='M' ? "checked='checked'" : ""}>
                </div>
```

```
                <div class="col-sm-2">
                  <label for="gender2" class="col-sm-2 col-form-label">여자
</label>
                  <input class="form-check-input" type="radio" name="gender"
id="gender2" value="F" ${memberVO.gender=='F' ? "checked='checked'" : ""}>
                 </div>
            </div>
            <div class="mb-3 row">
                <label class="col-sm-2 col-form-label" for="postCode">우편번호
</label>
                <div class="col-sm-2">
                  <input class="form-control" type="text" name="postCode"
id="postCode" readonly  required value="${memberVO.postCode }">
                </div>
                <div class="col-sm-1"></div>
                <div class="col-sm-2">
                  <input type="button" class="btn-check" id="zipCodebtn"
onclick="daumPostcode();">
                  <label class="btn btn-outline-primary" for="zipCodebtn">우편번호
찾기</label>
                </div>
            </div>
            <div class="mb-3 row">
                <label class="col-sm-2 col-form-label" for="addr1">주소</label>
                <div class="col-sm-10">
                  <input class="form-control" type="text" name="addr1"
id="addr1" readonly  required value="${memberVO.addr1 }">
                </div>
            </div>
            <div class="mb-3 row">
                <label class="col-sm-2 col-form-label" for="addr2">상세주소
</label>
                <div class="col-sm-10">
                  <input class="form-control" type="text" name="addr2"
id="addr2"  required  value="${memberVO.addr2 }">
                </div>
            </div>
            <div class="mb-3 row">
                <div class="col-sm-12" style="text-align: right;">
                      <input type="submit" class="btn-check" id="submitBtn" >
                      <label class="btn btn-outline-success" for="submitBtn">정보
수정</label>
                      <input type="reset" class="btn-check" id="resetBtn" >
                      <label class="btn btn-outline-success" for="resetBtn">다시
쓰기</label>
                      <input type="button" class="btn-check" id="cancelBtn"
onclick="location.href='${pageContext.request.contextPath}'">
                      <label class="btn btn-outline-success" for="cancelBtn">돌아
가기</label>
                </div>
            </div>
        </form>
    </div>
</body>
</html>
```

- 회원정보 수정을 완료하는 파일을 만들자(updateOk.jsp)

```jsp
<%@page import="java.util.UUID"%>
<%@page import="kr.human.member.service.MemberServiceImpl"%>
<%@page import="java.text.SimpleDateFormat"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<% request.setCharacterEncoding("UTF-8"); %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <jsp:useBean id="memberVO" class="kr.human.member.vo.MemberVO"/>
    <jsp:setProperty property="*" name="memberVO"/>
    <%
        // 새로운 비밀번호는 별도로 받는다.
        String newPassword = request.getParameter("newPassword");
        // 새로운 비번이 없으면 새로운 비번을 기존의 비번으로 만들어 준다.
        if(newPassword==null || newPassword.trim().length()==0){
            newPassword = memberVO.getPassword();
        }
    %>
    <%//=memberVO %>
    <%
        // 서비스를 호출하여 수정을 하고
        MemberServiceImpl.getInstance().update(memberVO, newPassword, session);

        // 어디론가 간다.
        response.sendRedirect(request.getContextPath()); // 홈으로
    %>
</body>
</html>
```

- 회원 탈퇴하는 폼을 만들자(deleteForm.jsp)

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>회원탈퇴</title>
<link rel="stylesheet"

href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css">
<script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
<script
    src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js">
</script>
<script

src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js">
</script>
```

```
<script src="https://kit.fontawesome.com/3c36eed32b.js" crossorigin="anonymous">
</script>

<link rel="stylesheet" type="text/css"
    href="${pageContext.request.contextPath }/css/app.css" />
</head>

<body>
    <div id="mainWrapper">
        <div class="login-container">
            <div class="login-card">
                <div class="login-form">
                    <form action="${pageContext.request.contextPath
}/deleteOk.jsp" method="post" class="form-horizontal">
                        <%-- 로그인 실패시 에러메세지 출력 --%>
                        <c:if test="${not empty error }">
                            <div style="color: red;font-size: 15pt;">${error }
</div>
                        </c:if>
                        <div class="input-group input-sm">
                            <input type="hidden" name="idx"
value="${memberVO.idx }">
                            <label class="input-group-addon" for="userid">
                            <i class="fa-solid fa-user" style="font-size:
20pt;margin-right: 5px;color:green;"></i>
                            </label> <input type="text" class="form-control"
                                id="userid" name="userid" placeholder="Enter
Userid" required>
                        </div>
                        <div class="input-group input-sm">
                            <label class="input-group-addon" for="password"><i
class="fa-solid fa-lock" style="font-size: 20pt;margin-right: 5px;color:red;">
</i></label> <input type="password"
                                class="form-control" id="password"
name="password"
                                placeholder="Enter Password" required>
                        </div>
                        <div class="input-group input-sm" style="margin-bottom:
15px;">
                            <textarea name="content" id="content" cols="70"
rows="10" required placeholder="탈퇴 이유를 남겨주세요"></textarea>
                        </div>

                        <div class="form-actions">
                            <input type="submit"
                                class="btn btn-block btn-primary btn-default"
value="회원탈퇴">
                        </div>
                        <div style="text-align: center;margin: 15px;">
                            [<a href="${pageContext.request.contextPath }">홈으로
</a>]
                        </div>
                    </form>
                </div>
            </div>
        </div>
    </div>
```

```
</body>
</html>
```

- 회원탈퇴를 처리하는 파일을 만들자(deleteOk.jsp)

```
<%@page import="java.util.UUID"%>
<%@page import="kr.human.member.service.MemberServiceImpl"%>
<%@page import="java.text.SimpleDateFormat"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<% request.setCharacterEncoding("UTF-8"); %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <jsp:useBean id="memberVO" class="kr.human.member.vo.MemberVO"/>
    <jsp:setProperty property="*" name="memberVO"/>
    <%//=memberVO %>
    <%
        // 서비스를 호출하여 삭제를 하고
        MemberServiceImpl.getInstance().delete(memberVO, session);

        // 어디론가 간다.
        response.sendRedirect(request.getContextPath()); // 홈으로
    %>
</body>
</html>
```

- 아이디 찾기를 만들자(findUserid.jsp)

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>아이디찾기</title>
<link rel="stylesheet"

href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css">
<script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
<script
    src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js">
</script>
<script

src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js">
</script>

<script src="https://kit.fontawesome.com/3c36eed32b.js" crossorigin="anonymous">
</script>
```

```
<link rel="stylesheet" type="text/css"
    href="${pageContext.request.contextPath }/css/app.css" />
</head>

<body>
    <div id="mainWrapper">
        <div class="login-container">
            <div class="login-card">
                <div class="login-form">
                    <form action="${pageContext.request.contextPath
}/findUseridOk.jsp" method="post" class="form-horizontal">
                        <%-- 로그인 실패시 에러메세지 출력 --%>
                        <c:if test="${not empty error }">
                            <div style="color: red;font-size: 13pt;">${error }
</div>
                        </c:if>
                        <div class="input-group input-sm">
                            <label class="input-group-addon" for="name">
                            <i class="fa-solid fa-user" style="font-size:
20pt;margin-right: 5px;color:green;"></i>
                            </label> <input type="text" class="form-control"
                                id="name" name="name" placeholder="사용자 이름 입
력" required>
                        </div>
                        <div class="input-group input-sm">
                            <label class="input-group-addon" for="phone">
                                <i class="fa-solid fa-square-phone" style="font-
size: 20pt;margin-right: 5px;color:red;"></i></label>
                                <input type="tel"   class="form-control"
id="phone" name="phone"
                                placeholder="전화 번호 입력" required>
                        </div>
                        <div class="form-actions">
                            <input type="submit"
                                class="btn btn-block btn-primary btn-default"
value="아이디 찾기">
                        </div>
                        <div style="text-align: center;margin: 15px;">
                            [<a href="${pageContext.request.contextPath
}/insertForm.jsp">회원가입</a>]
                               
                            [<a href="${pageContext.request.contextPath }">홈으로
</a>]
                        </div>
                    </form>
                </div>
            </div>
        </div>
    </div>

</body>
</html>
```

- 아이디 찾기 완료를 만들자(findUseridOk.jsp)

```
<%@page import="kr.human.member.vo.MemberVO"%>
<%@page import="kr.human.member.service.MemberServiceImpl"%>
```

```
<%@page import="kr.human.member.service.MemberService"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <%
        request.setCharacterEncoding("UTF-8");
        if(request.getMethod().equals("GET")){
            response.sendRedirect(request.getContextPath());
            return;
        }
        String name = request.getParameter("name");
        String phone = request.getParameter("phone");

        if(name!=null && phone!=null){
            // 서비스를 호출하여 해당회원의 정보를 가져온다.
            MemberVO memberVO =
MemberServiceImpl.getInstance().searchUserid(name, phone);
            if(memberVO==null){
                request.setAttribute("error", "잘못된 정보입니다.");
                pageContext.forward("findUserid.jsp");
            }else{
                out.println(name + "님의 아이디는 \"" + memberVO.getUserid() +
"\"입니다.<br>");
                out.println("<a href='login.jsp'>로그인하러가기</a>");
            }
        }else{
            request.setAttribute("error", "잘못된 정보입니다.");
            pageContext.forward("findUserid.jsp");
            return;
        }
    %>
</body>
</html>
```

- 비밀번호 찾기를 만들자(findPassword.jsp)

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>비밀번호 찾기</title>
<link rel="stylesheet"

href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css">
<script src="https://code.jquery.com/jquery-3.4.1.min.js"></script>
<script
    src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js">
</script>
```

```
<script

src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js">
</script>

<script src="https://kit.fontawesome.com/3c36eed32b.js" crossorigin="anonymous">
</script>

<link rel="stylesheet" type="text/css"
    href="${pageContext.request.contextPath }/css/app.css" />
</head>

<body>
    <div id="mainWrapper">
        <div class="login-container">
            <div class="login-card">
                <div class="login-form">
                    <form action="${pageContext.request.contextPath
}/findPasswordOk.jsp" method="post" class="form-horizontal">
                        <%-- 비번찾기 실패시 에러메세지 출력 --%>
                        <c:if test="${not empty error }">
                            <div style="color: red;font-size: 13pt;">${error }
</div>
                        </c:if>
                        <div class="input-group input-sm">
                            <label class="input-group-addon" for="userid">
                            <i class="fa-solid fa-user" style="font-size:
20pt;margin-right: 5px;color:green;"></i>
                            </label> <input type="text" class="form-control"
                                id="userid" name="userid" placeholder="사용자 아이
디 입력" required>
                        </div>
                        <div class="input-group input-sm">
                            <label class="input-group-addon" for="phone">
                                <i class="fa-solid fa-square-phone" style="font-
size: 20pt;margin-right: 5px;color:red;"></i></label>
                                <input type="tel"   class="form-control"
id="phone" name="phone"
                                placeholder="전화 번호 입력" required>
                        </div>
                        <div class="form-actions">
                            <input type="submit"
                                class="btn btn-block btn-primary btn-default"
value="비밀번호 찾기">
                        </div>
                        <div style="text-align: center;margin: 15px;">
                            [<a href="${pageContext.request.contextPath
}/login.jsp">로그인하기</a>]
                               
                            [<a href="${pageContext.request.contextPath }">홈으로
</a>]
                        </div>
                    </form>
                </div>
            </div>
        </div>
    </div>
```

```
</body>
</html>
```

- 비밀번호 찾기완료를 만들자(findPasswordOk.jsp)

```
<%@page import="kr.human.member.vo.MemberVO"%>
<%@page import="kr.human.member.service.MemberServiceImpl"%>
<%@page import="kr.human.member.service.MemberService"%>
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Insert title here</title>
</head>
<body>
    <%
        request.setCharacterEncoding("UTF-8");
        if(request.getMethod().equals("GET")){
            response.sendRedirect(request.getContextPath());
            return;
        }
        String userid = request.getParameter("userid");
        String phone = request.getParameter("phone");

        if(userid!=null && phone!=null){
            // 서비스를 호출하여 해당회원의 정보를 가져온다.
            MemberVO memberVO =
MemberServiceImpl.getInstance().searchPassword(userid, phone);
            if(memberVO==null){
                request.setAttribute("error", "잘못된 정보입니다.");
                pageContext.forward("findPassword.jsp");
            }else{
                out.println(userid + "님의 임시비밀번호가 \"" + memberVO.getEmail()
+ "\"로 발송되었습니다.<br>");
                out.println("<a href='login.jsp'>로그인하러가기</a>");
            }
        }else{
            request.setAttribute("error", "잘못된 정보입니다.");
            pageContext.forward("findPassword.jsp");
            return;
        }
    %>
</body>
</html>
```