6. 회원가입 만들기

1. DTO 생성

dto/user/JoinReqDto.java

```
package shop.mtcoding.bankapp.dto.user;

import lombok.Getter;
import lombok.Setter;

@Setter
@Getter
public class JoinReqDto {
    private String username;
    private String password;
    private String fullname;
}
```

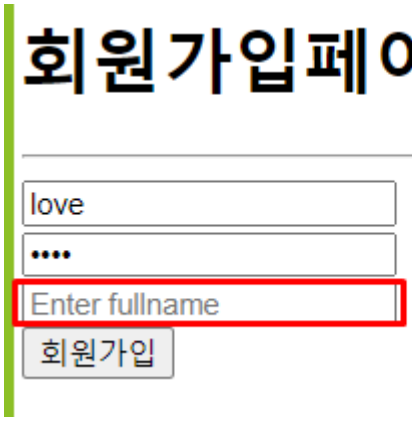
dto/user/JoinReqDto.java

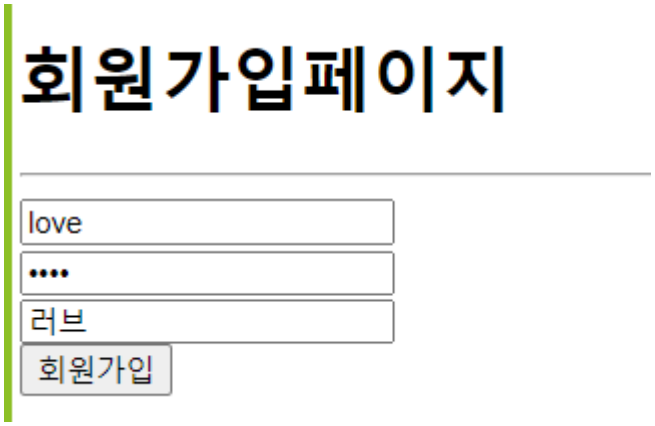
```
package shop.mtcoding.bankapp.dto.user;

import lombok.Getter;
import lombok.Setter;

@Setter
@Getter
public class LoginReqDto {
    private String username;
    private String password;
}
```

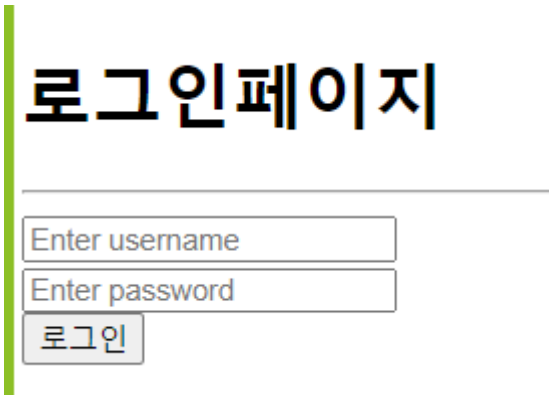
2. Controller 완료

```
/**
 * POST요청과 PUT요청시에만 BODY 데이터가 있다.
 * 해당 BODY데이터는 컨트롤러 메서드의 매개변수에 주입된다.(DS)
 * 스프링은 x-www-form-urlencoded가 기본 파싱전략
 * key=value&key=value (form태그의 기본 전송 전략)
 * 컨트롤러의 메서드는 매개변수에서 두가지 방식으로 데이터를 받는다.
 * 1. 그냥 변수, 2. DTO(Object)
 * 주의 : key이름과 변수이름이 동일해야 한다.
 */
@PostMapping("/join")
```

```
public String join(JoinReqDto joinReqDto) { // DTO로 받는 것이 좋다.
    // 1. POST, PUT일 때만 유효성 검사 (이것보다 우선되는 것이 인증 검사이다)
    if (joinReqDto.getUsername() == null || joinReqDto.getUsername().isEmpty()) {
        throw new CustomException("username을 입력해주세요", HttpStatus.BAD_REQUEST);
    }
    if (joinReqDto.getPassword() == null || joinReqDto.getPassword().isEmpty()) {
        throw new CustomException("password를 입력해주세요", HttpStatus.BAD_REQUEST);
    }
    if (joinReqDto.getFullname() == null || joinReqDto.getFullname().isEmpty()) {
        throw new CustomException("fullname을 입력해주세요", HttpStatus.BAD_REQUEST);
    }

    // 서비스 호출 => 회원가입();

    return "redirect:/loginForm";
}
```







3. 서비스 완료

model/user/UserRepository.java

```
public int insert(JoinReqDto joinReqDto);
```

service/UserService.java

```
package shop.mtcoding.bankapp.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.stereotype.Service;
import org.springframework.transaction.annotation.Transactional;

import shop.mtcoding.bankapp.dto.user.JoinReqDto;
import shop.mtcoding.bankapp.handler.ex.CustomException;
import shop.mtcoding.bankapp.model.user.UserRepository;

@Service // IoC
public class UserService {

    @Autowired // DI
    private UserRepository userRepository;

    @Transactional // 회원가입 메서드 호출이 시작될때, 트랜잭션 시작, 끝날때, 트랜잭션 종료
(commit)
    public void 회원가입(JoinReqDto joinReqDto) {
        // mybatis는 인수로 들어온 오브젝트의 변수명으로 자동 매핑해준다.
        int result = userRepository.insert(joinReqDto);
        if (result != 1) {
            throw new CustomException("회원가입 실패",
HttpStatus.INTERNAL_SERVER_ERROR);
        }
    }
}
```

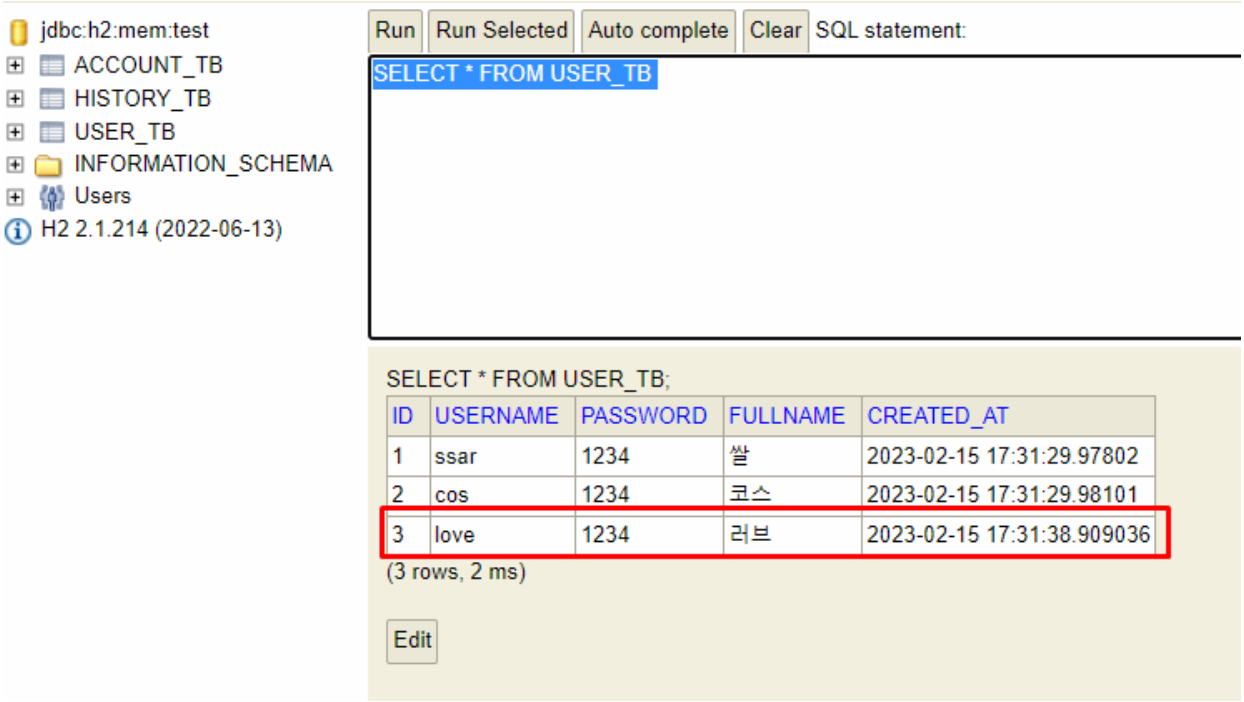
4. Controller 수정

```
@Controller
public class UserController {

    @Autowired
    private UserService userService;

    // ... 생략
}
```

```
@PostMapping("/join")
public String join(JoinReqDto joinReqDto) { // DTO로 받는 것이 좋다.
    // ... 생략

    // 컨벤션 : post, put, delete 할때만 하기
    // 서비스 호출 => 회원가입();
    userService.회원가입(joinReqDto);

    // ... 생략
}
```

5. application.yml 더미 데이터 실행 주석 풀기

```
... 생략

spring:
  sql:
    init:
      schema-locations:
      - classpath:db/table.sql
      data-locations:
      - classpath:db/data.sql

... 생략
```


jdbc:h2:mem:test
ACCOUNT_TB
HISTORY_TB
USER_TB
INFORMATION_SCHEMA
Users
H2 2.1.214 (2022-06-13)
Run
Run Selected
Auto complete
Clear
SQL statement:
SELECT * FROM USER_TB
SELECT * FROM USER_TB;
ID USERNAME PASSWORD FULLNAME CREATED_AT
1 ssar 1234 쌀 2023-02-15 17:31:29.97802
2 cos 1234 코스 2023-02-15 17:31:29.98101
3 love 1234 러브 2023-02-15 17:31:38.909036
(3 rows, 2 ms)
Edit