〈 2022 03 16 WED 〉

오늘은 Project 회의를 진행하고

어떻게 병합해야할 건지 의논할 예정

내가 맡은 파트에서 회원정보 수정 refactoring 수행

serviceImpl (parameterNullProcess)—〉 수정 전

```
/****
 *  parameter 값이 null 일때를 처리하기 위한 MEthod
 */
@Override
public Map<String, String> parameterNullProcess(HttpSession session, String nick, String pw, String answer, String gender, String birthday, String
context) {

    Map<String, String> result = new HashMap<>();

    /** parameter 들의 값들이 null 일 경우
     *
     * model 에서 현재 값들을 가져와서 비어있는 값들에 적용 시킨다.
     * */
    String nickValidError = null;
    String pwValidError = null;
    String answerValidError = null;

    // ErrorOrNot
    boolean errorEmerge = false;

    if(nick==""){
        nick = (String)session.getAttribute("nick");
    }else if(nick!=null){
        if(!validateHandling.nickValid(nick)){
            nickValidError= "Korean, English, numbers excluding special characters, 2-20 digits";
            result.put("nickValidError",nickValidError);
            errorEmerge=true;

        }
    }
    if(pw == ""){
        /**
         * pw 는 복호화할 수 없기 때문에 pw 가 null 일 때 다른 method 수행 시켜야함
         * */
        pw = "noChange";
    }else if(pw!=null){
        if(!validateHandling.pwValid(pw)){
            pwValidError = "Password must be 8 to 16 characters in uppercase and lowercase letters, numbers, and special characters.";
            result.put("pwValidError",pwValidError);
            errorEmerge=true;
        }
    }

    if(gender==null){
        gender=(String)session.getAttribute("userGender");
        if(gender.equals("Male")){
            gender = "m";
        }else if(gender.equals("Female")){
            gender = "f";
        }
    }
    System.out.println(birthday);
    if(birthday == ""){
        LocalDate getBirthday= (LocalDate) session.getAttribute("userBirthday");
        birthday = String.valueOf(getBirthday);
    }

    if(answer == "" ) {
        answer = (String) session.getAttribute("userAnswer");
    }else if(answer!=""){
        if(!validateHandling.answerValid(answer)){
            answerValidError = "answer type is Korean, numeric, English 2-20 characters";
            result.put("answerValidError",answerValidError);
            errorEmerge=true;
        }
    }

    if(context == ""){
        context = (String)session.getAttribute("userContext");

        /** 질문 변경이 없다 ---> 현재 질문에 유지
         *
         * 그에 대한 답도 없다 ----> 질문에 다한 답 또 한 유지
         * */
    }else if(context!="") {
        /**
         * 질문이 null 이 아니고 기존의 질문과 일치하지 않다면 ----> 즉 달라진다면
         * answer 도 무조건 달라져야하는데.
         * Answer 의 값이 null 이라면 error
         *
         * */
        if(!(context.equals((String)session.getAttribute("userContext")))&&answer==null){
            result.put("errorMessage", "If you want to change a question, please enter an answer to the question");
            errorEmerge=true;
        }else if((context.equals((String)session.getAttribute("userContext")))&&answer==null){
            answer = (String) session.getAttribute("userAnswer");
        }
        }


    if(errorEmerge==false) {
        result.put("error","notExists");
        result.put("nick", nick);
        result.put("pw", pw);
        result.put("answer", answer);
        result.put("gender", gender);
        result.put("birthday", birthday);
        result.put("context", context);
    }else{
        result.put("error","exists");

    }

 return result;
}
```

serviceImpl (parameterNullProcess)—〉수정 후

```
/**
 *  parameter 값이 null 인지 판별하는 methode
-------------------------------------------------------------------------------------------
-----------------------------------
 */
@Override
public Map<String, Object> parameterValidCheck(HttpSession session, String nick, String
pw, String answer, String gender, String birthday, String context)
{

    Map<String, Object> result = new HashMap<>();

    /** parameter 들의 값들이 null 일 경우
     *  session 에서 현재 값들을 가져와서 비어있는 값들에 적용 시킨다.
     * */

    // ErrorOrNot
    boolean isValid = true;

    //----------------------------------- nick 이 유효한지 확인
    if(nick.isEmpty())
    {
        // 입력 되지 않았다면 session 에서 nick 값을 받음
        nick = (String)session.getAttribute("nick");
    }
    else
    {
        /**
         *     true --> nick 값이 유효
         *     false --> nick 값이 유효하지 않음
         */
        if(!validateHandling.nickValid(nick))
        {
            result.put("nickValidError","Korean, English, numbers excluding special
characters, 2~20 digits");
            isValid=false;

        }
    }

    //----------------------------------- pw가 유효한지 확인
    if(pw.isEmpty())
    {
        /**
         * pw 는 복호화할 수 없기 때문에 pw 가 null 일 때 다른 method 수행 시켜야함
         * */
        pw = "noChange";
    }
    else
    {
        /**    pwValid : pw 유효성 검사
         *     true --> pw 값이 유효
         *     false --> pw 값이 유효하지 않음
         */
        if(!validateHandling.pwValid(pw))
        {
            result.put("pwValidError","Password must be 8 to 16 characters in uppercase
and lowercase letters, numbers, and special characters.");
            isValid=false;
        }
    }
```

```
    // ---------------------------------- gender 가 유효한지


    if(gender==null)
    {
        gender=(String)session.getAttribute("userGender");
        if(gender.equals("Male"))
        {
            gender = "m";
        }
        else
        {
            gender = "f";
        }
    }


    // ---------------------------------- birthday 가 유효한지
    if(birthday.isEmpty())
    {
        LocalDate getBirthday= (LocalDate) session.getAttribute("userBirthday");
        birthday = String.valueOf(getBirthday);
    }


    // ---------------------------------- answer 가 null 인지 아닌지
    if(answer.isEmpty())
        {
            answer = (String) session.getAttribute("userAnswer");
        }

    else
        {
            if(!validateHandling.answerValid(answer))
            {
                result.put("answerValidError", "answer type is Korean, numeric, English
2~20 characters");
                isValid=false;
            }
        }
```

```
    // ---------------------------------- 질문이 null 인지 아닌지
    if(context.isEmpty())
        {
            context = (String)session.getAttribute("userContext");

            /** 질문 변경이 없다 ---> 현재 질문에 유지
             *
             * 그에 대한 답도 없다 ----> 질문에 다한 답 또한 유지
             * */
        }

    else
        {
            /**
            * 질문이 null 이 아니고 기존의 질문과 일치하지 않다면 ----> 즉 달라진다면
            * answer 도 무조건 달라져야하는데.
            * Answer 의 값이 null 이라면 error
             *
            * */
            if((!
context.equals((String)session.getAttribute("userContext")))&&answer==null)
                {
                    result.put("errorMessage", "If you want to change a question, please
enter an answer to the question");
                    isValid=false;
                }


if((context.equals((String)session.getAttribute("userContext")))&&answer==null)
                {
                    answer = (String) session.getAttribute("userAnswer");
                }
        }

    result.put("isValid", isValid);

    if(isValid)
    {
        result.put("nick", nick);
        result.put("pw", pw);
        result.put("answer", answer);
        result.put("gender", gender);
        result.put("birthday", birthday);
        result.put("context", context);
    }
 return result;
}
```

1. method 명을 변경

2. 기능 마다 구별할 수 있도록 주석처리

3. member 변수 선언을 제거하고

map 에서 바로 사용할 수 있도록 변경

4. isEmpty() 사용

serviceImpl (userInfoUpdate)—〉수정 전

```
public Map<String, Object> userInfoUpdate(String currentNick, String nick, String pw, String pwCheck, String gender, String
birthday){
    /** 닉네임이 유효한지 아닌지에 대해서 return
     * 중복 된다면 false
     * 중복 되지 않는 다면 true
     * */
    boolean nickDuplicate = pwAndDupCheck.nickDuplicateCheck(nick, currentNick);


    /***
     * pw 와 pwcheck 가 일치하는지 확인
     */
    boolean pwCheckResult = false;

    /***
     * pw 를 바꾹 싶지 않다는 것을 확인
     */
    boolean wantPwChange =true;


    if(pw.equals("noChange")){
        /***
         *  즉 비밀번호에 값을 아무것도 넣지 않았을 경우를 말함
         *  pw 에 아무것도 없는데 pwcheck 에 값이 있는 것은 이상함으로
         * */
        if(pwCheck.equals("")){
            pwCheckResult=true;
            wantPwChange = false;
        }
    }else{
        pwCheckResult = pwAndDupCheck.pwAndPwCheck(pw,pwCheck);
        if(pwCheckResult == true){
            pw = BCrypt.hashpw(pw,BCrypt.gensalt());
        }
    }

    DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd");
    LocalDate birthdayChanged = LocalDate.parse(birthday, formatter);

    Map<String, Object> result = new HashMap<>();

    if(nickDuplicate==true&&pwCheckResult==true){
        result.put("result",true);
        if(wantPwChange==true) {
            userRepository.changeUserInfo(currentNick, nick, pw, gender, birthdayChanged);
        }else{
            userRepository.byNickUpdateUserInfo(currentNick,nick,gender,birthdayChanged);
        }

    }else{
        result.put("result",false);
        if(nickDuplicate==false) {
            result.put("nickErrorMessage", "someone Using that nick name, please setting another one");
        }
        if(pwCheckResult==false){
            result.put("pwErrorMessage","The passwords do not match each other.");
        }
    }

    return result;
}
```

serviceImpl (userInfoUpdate)—〉 수정 후

```
/**
 * @return
 * key
 * result ---> true : 값을 정상적으로 전달
 * result ---> false : error 발생
 *
 * error
 * key
 * nickErrorMessage
 * pwErrorMessage
 */
public Map<String, Object> userInfoUpdate(String currentNick,
                                          String nick,
                                          String pw,
                                          String pwCheck,
                                          String gender,
                                          String birthday)
{


    /** 닉네임 유효성 검사
     * 중복 x --> true
     * 중복 o --> false
     * */
    boolean isNickValid = pwAndDupCheck.nickDuplicateCheck(nick, currentNick);


    // pw 와 pwCheck 가 서로 일치하는지 판별
    // true --> 일치
    // false --> 불일치
    boolean isPwCheckValid = false;

    // 비밀번호를 변경 여부를 판별
    // true --> 변경
    // false --> 변경하지 않음
    boolean isPwChange =true;

    // 비밀번호 ============================================================
    // 비밀번호를 바꾸지 않는 경우 ----------------------------
    // 바꾸지 않는 이유에 method 가 다르게 적용
    if(pw.equals("noChange"))
    {
        // pwCheck 가 비어있는 경우
        if(pwCheck.isEmpty())
        {
            isPwCheckValid=true;
            isPwChange = false;
        }
    }

    // 비밀번호를 바꾸지 않는 경우 ---------------------------
    else
    {
        /**
         * true --> 일치
         * false ---> 불일치
         */
        isPwCheckValid = pwAndDupCheck.pwAndPwCheck(pw,pwCheck);
        if(pwAndDupCheck.pwAndPwCheck(pw,pwCheck))
        {
            pw = BCrypt.hashpw(pw,BCrypt.gensalt());
        }
    }
```

```
    // ------- 생일  Data  형식을 변경 ------------------------------------------------
    DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd");
    LocalDate birthdayChanged = LocalDate.parse(birthday, formatter);



    Map<String, Object> result = new HashMap<>();

    // 아이디 중복 없고, pw 확인 까지 서로 맞을 때 ----------------------------------------
    if( (isNickValid) && (isPwCheckValid) )
    {
        result.put("result",true);
        // 비밀번호 변경을 포함 할 때 -------------------------
        if(isPwChange)
           {
            userRepository.changeUserInfo(currentNick, nick, pw, gender, birthdayChanged);
           }
        // 비밀번호 변경을 포함하지 않을 때 ---------------------
        else
            {
            userRepository.changeUserInfoExceptPw(currentNick,nick,gender,birthdayChanged);
            }

    }
    // error 가 발생한 경우  ---------------------------------------------------------------------------
    else
    {
        result.put("result",false);

        // 닉네임이 중복일 경우
        if(isNickValid==false)
        {
            result.put("nickErrorMessage", "someone Using that nick name, please setting another one");
        }
        if(isPwCheckValid==false)
        {
            // password 가 일치하지 않을 경우
            result.put("pwErrorMessage","The passwords do not match each other.");
        }
    }
    return result;
}
```

1. 주석 추가

2. boolean 변수 이름 변경

serviceImpl (changeUserInfo)—〉 수정 전

```
/***
 *  user info 변경시
 *
 *  parameter null 값을 처리해주는 method
 */
@Override
public Map<String, Object> changeUserInfo(HttpSession session, String currentNick,
                               String nick,
                               String pw,
                               String pwCheck,
                               String gender,
                               String birthday,
                               String answer,
                               String context) {


    /****
     *  요소들 중 하나라도 안넣으면 Error 가 발생하기 때문에
     *
     *  정보를 변경하려면 전체를 변경해야 한다 .. 비효율적이다 ..
     *
     *  그래서 변경하지 않을 정보를 null 로 두면
     *
     *  model 에서 지금 현재 정보를 가져와서 적용 시킨다 .
     *
     */

    /***
     *  user 정보를 수정하는 method
     *
     *  result : true ---> 정보 변경이 잘됨
     *  result : false ---> 정보 변경중 오류 발생
     *          --> nickErrorMessage = nickName 설정시 Error 발생 message
     *          --> pwErrorMessage = pw 변경시 Error 발생 message
     */
    Map<String, Object> userInfoUpdate =
userInfoUpdate(currentNick,nick,pw,pwCheck,gender,birthday);


    // User code 를 사용해서 나머지 정보들을 불러올수 있음으로 UserCode 를 구하는 method 실행
    Long code = (Long) session.getAttribute("code");


    User userCode = User.builder()
            .code(code)
            .build();

    // ano를 구하기 위한 method
    Long ano = answerRepository.anoByCode(userCode);
    System.out.println(ano);



    // context 를 수정하기 위해서 AnswerRepository 를 사용해서 qno 를 return
    Question questionQno = answerRepository.getQnoByCode(userCode);
    Long qno = questionQno.getQno();
    System.out.println(qno);


    //answer 를 수정하기 위한 method
    int resultAnswer = answerRepository.updateUserAnswer(ano, answer);


    // context 를 수정하기 위해서 qno 와  context  를 입력받는 method
    int resultQuestion = questionRepository.updateUserContext(qno, context);



    return userInfoUpdate;
}
```

serviceImpl (changeUserInfo)—〉 수정 후

```
/**
 * user 의 개인 정보 변경 method
 * - method 들이 합쳐져있음
 * /**
 *          * @return
 *          * key
 *          * result ---> true : 값을 정상적으로 전달
 *          * result ---> false : error 발생
 *          *
 *          * error
 *          * key
 *          * nickErrorMessage
 *          * pwErrorMessage
 *
 */
@Override
public Map<String, Object> changeUserInfo(HttpSession session,
                                          String currentNick,
                                          String nick,
                                          String pw,
                                          String pwCheck,
                                          String gender,
                                          String birthday,
                                          String answer,
                                          String context) {

    /**
     * @return
     * key
     * result ---> true : 값을 정상적으로 전달
     * result ---> false : error 발생
     *
     * error
     * key
     * nickErrorMessage
     * pwErrorMessage
     */
    Map<String, Object> userInfoUpdate = userInfoUpdate(currentNick,nick,pw,pwCheck,gender,birthday);

    // ------------------------------------ session 에서 user code 받는다
    Long code = (Long) session.getAttribute("code");

    // User Entity 혀태로 변경
    User userCode = User.builder()
            .code(code)
            .build();

    // ano를 구하기 위한 method ------------------------------------------------------------
    Long ano = answerRepository.anoByCode(userCode);



    // context 를 수정하기 위해서 AnswerRepository 를 사용해서 qno 를 return ---------------------------------
    Question questionQno = answerRepository.getQnoByCode(userCode);
    Long qno = questionQno.getQno();


    /**
     * answer 를 수정하기 위한 method ----------------------------------------------------
     * 0 < result ---> 변경 성공
     * 0 > result ---> 변경 실패
     */
    answerRepository.updateUserAnswer(ano, answer);


    /**
     * context 를 수정하기 위한 method ----------------------------------------------------
     * 0 < result ---> 변경 성공
     * 0 > result ---> 변경 실패
     */
    questionRepository.updateUserContext(qno, context);

    return userInfoUpdate;
}
```

1. 주석 추가

controller 〈editingUserInfo〉 —〉 수정 전

```
@PostMapping("editing")
public String editingUserInfo(HttpSession session ,
                              String nick,
                              String pw,
                              String answer,
                              Model model,
                              String pwCheck,
                              String gender,
                              String birthday,
                              String context){



    /***
     * session 에서 현재 nick name 을 받아온다 .
     */

    String currentNick = (String)session.getAttribute("nick");


    /***
     * null 처리를 해야한다 .
     */
    System.out.println(birthday);

     Map<String, String> nullProcess = editService.parameterNullProcess(session, nick, pw, answer, gender, birthday, context);

    if(nullProcess.get("error").equals("notExists")) {
         nick = nullProcess.get("nick");
         pw = nullProcess.get("pw");
        answer = nullProcess.get("answer");
        birthday = nullProcess.get("birthday");
        gender = nullProcess.get("gender");
        context = nullProcess.get("context");

        Map<String, Object> result = editService.changeUserInfo(session, currentNick, nick, pw, pwCheck, gender, birthday, answer, context);

        if ((boolean) result.get("result")==true){
            session.setAttribute("nick", nick);
        }else{
            if((String)result.get("nickErrorMessage")!=null){
                model.addAttribute("nickErrorMessage",(String)result.get("nickErrorMessage"));
            }
            if((String)result.get("pwErrorMessage")!=null){
                model.addAttribute("pwErrorMessage",(String)result.get("pwErrorMessage"));
            }
            return "checkplan/forUser/editUser";
        }


    }else if(nullProcess.get("error").equals("exists")){
        if((String)nullProcess.get("nickValidError")!=null){
            model.addAttribute("nickValidError",(String)nullProcess.get("nickValidError"));
        }
        if((String)nullProcess.get("pwValidError")!=null){
            model.addAttribute("pwValidError",(String)nullProcess.get("pwValidError"));
        }
        if((String)nullProcess.get("answerValidError")!=null){
            model.addAttribute("answerValidError",(String)nullProcess.get("answerValidError"));
        }
        model.addAttribute("errorMessage",nullProcess.get("errorMessage"));

        return "checkplan/forUser/editUser";
    }
    return "checkplan/forUser/editUserInfo";
}
```

controller 〈editingUserInfo〉 —〉 수정 후

```
/**
 * 회원 정보 변경 page 에서 변경하는 method ==========================================================
 */
@PostMapping("editing")
public String editingUserInfo(HttpSession session ,
                              String nick,
                              String pw,
                              String answer,
                              Model model,
                              String pwCheck,
                              String gender,
                              String birthday,
                              String context)
{
    /**
     *  paramValid : key---> isValid
     *  true --> error 존재 x
     *  false --> error 존재함
     */
    Map<String, Object> paramValid = editService.parameterValidCheck
            (
            session,
            nick,
            pw,
            answer,
            gender,
            birthday,
            context
            );

    // error 가 존재하지 않음 : 변수 전체는 유효한 값 ------------------------------------------------------
    if((boolean)paramValid.get("isValid"))
        {
         nick = (String)paramValid.get("nick");
         pw = (String)paramValid.get("pw");
        answer = (String)paramValid.get("answer");
        birthday = (String)paramValid.get("birthday");
        gender = (String)paramValid.get("gender");
        context = (String)paramValid.get("context");


        String currentNick = (String)session.getAttribute("nick");

            /**
             * @return ------------------------------------------------------------------
             * key
             * result ---> true : 값을 정상적으로 전달
             * result ---> false : error 발생
             *
             * error
             * key
             * nickErrorMessage
             * pwErrorMessage
             */
        Map<String, Object> result = editService.changeUserInfo
                (
                        session,
                        currentNick,
                        nick,
                        pw,
                        pwCheck,
                        gender,
                        birthday,
                        answer,
                        context
                );
```

```
// 정상적으로 회원정보가 변경 됨 ------------------------------------------------
        if ((boolean) result.get("result"))
            {
            session.setAttribute("nick", nick);
            }

        // error 가 발생했을 때 ------------------------------------------------------
        // 닉네임 중복, 비밀번호 비밀번호 체크 불일치
        else
            {

                // 닉네임이 중복 되었을 경우 ----------------------------
                if(result.containsKey("nickErrorMessage"))
                    {
                    model.addAttribute("nickErrorMessage",(String)result.get("nickErrorMessage"));
                    }

                // 비밀번호와 비밀번호 확인 값이 일치하지 않을 경우 ----------------------------
                if(result.containsKey("pwErrorMessage"))
                    {
                    model.addAttribute("pwErrorMessage",(String)result.get("pwErrorMessage"));
                    }

                return "checkplan/forUser/editUser";
            }


    }
    // parameter 값이 유효한 값이 아닐 때  ------------------------------------------------------
    else if(!(boolean)paramValid.get("isValid"))
        {
            // 닉네임 유효성을 불만족 ------------------
            if(paramValid.containsKey("nickValidError"))
                {
                model.addAttribute("nickValidError",(String)paramValid.get("nickValidError"));
                }


            // 비밀번호 유효성을 불만족----------
            if(paramValid.containsKey("pwValidError"))
                {
                model.addAttribute("pwValidError",(String)paramValid.get("pwValidError"));
                }


            // 질문에 대한 답 불만족 ----------
            if(paramValid.containsKey("answerValidError"))
                {
                model.addAttribute("answerValidError",(String)paramValid.get("answerValidError"));
                }
                // 질문에 대한 답 불만족 ----------


            // 질문이 변경되었는데 답을 입력하지 않았을 때 ----------
            if(paramValid.containsKey("errorMessage"))
                {
                model.addAttribute("errorMessage", paramValid.get("errorMessage"));
                }

            return "checkplan/forUser/editUser";
    }

    return "checkplan/forUser/editUserInfo";
}
```

1. 주석 추가

2. 줄맞춤

3. map. containsKey를 사용