

2025년 상반기 K-디지털 트레이닝

회원관리-비밀번호 변경(백엔드)

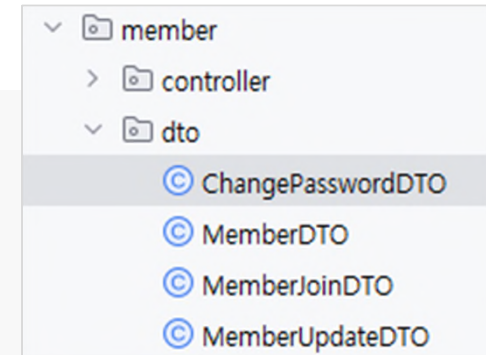
[KB] IT's Your Life

1 회원관리-비밀번호 변경(백엔드)

- ✓ 비밀번호 변경에는 multipart가 없으므로, application/json 인코딩으로 처리
- DTO 앞에 @RequestBody 설정 필요!

ChangePasswordDTO.java

```
@Data
@NoArgsConstructor
@AllArgsConstructor
public class ChangePasswordDTO {
    String username;    // 사용자 ID
    String oldPassword; // 이전 비밀번호
    String newPassword; // 새 비밀번호
}
```



MemberMapper.java

```
public interface MemberMapper {  
    ...  
    int updatePassword(ChangePasswordDTO changePasswordDTO);  
}
```

그냥 DTO 넣었어

MemberMapper.xml

```
<update id="updatePassword">
  UPDATE tbl_member
  SET
    password = #{newPassword},
    update_date = now()
  WHERE username =#{username}
</update>
```

MemberService.java ✓

```
public interface MemberService {
```

```
    ...
```

```
    void changePassword(ChangePasswordDTO changePassword);  
}
```

그냥 DTO 넘겼대

MemberServiceImpl.java

```
@Override
public void changePassword(ChangePasswordDTO changePassword) {
    MemberVO member = mapper.get(changePassword.getUsername());

    if(!passwordEncoder.matches(changePassword.getOldPassword(), member.getPassword())) {
        throw new PasswordMismatchException();
    }

    changePassword.setNewPassword(passwordEncoder.encode(changePassword.getNewPassword()));

    mapper.updatePassword(changePassword);
}
```

MemberController.java


```
@PutMapping("/{username}/changepassword")
public ResponseEntity<?> changePassword(@RequestBody ChangePasswordDTO changePasswordDTO) {
    service.changePassword(changePasswordDTO);
    return ResponseEntity.ok().build();
}
```



2025년 상반기 K-디지털 트레이닝


회원관리-비밀번호 변경(프론트엔드)

[KB] IT's Your Life

비밀번호 변경

 이전 비밀번호:

 새 비밀번호:

 새 비밀번호 확인:

✓ 확인

api/authApi.js

```
import api from '@api';  
  
const BASE_URL = '/api/member';  
  
export default {  
  ...  
  async changePassword(formData) {  
    const { data } = await api.put(`${BASE_URL}/${formData.username}/changepassword`, formData);  
    console.log('AUTH PUT: ', data);  
  
    return data;  
  },  
};
```

회원관리-비밀번호 변경(프론트엔드)

pages/auth/ChangePasswordPage.vue

```
<script setup>
import { computed, reactive, ref } from 'vue';
import { useAuthStore } from '@stores/auth';
import { useRouter } from 'vue-router';
import authApi from '@api/authApi';
```

```
const router = useRouter();
const auth = useAuthStore();
```

```
const changePassword = reactive({
  username: auth.username,
  oldPassword: "",
  newPassword: "",
  newPassword2: "",
});
```

유효성 검사 필수항목 체크

```
✓ const disableSubmit = computed(() => !changePassword.oldPassword || !changePassword.newPassword
|| !changePassword.newPassword2);
```

```
const error = ref("");
```

Schema are comprised of parsing actions (transforms) as well as assertions (tests) about the input value. Validate an input value to parse and run the configured set of assertions. Chain together methods to build a schema.

* KB 국민은행

```
import { object, string, number, date, InferType } from 'yup';

let userSchema = object({
  name: string().required(),
  age: number().required().positive().integer(),
  email: string().email(),
  website: string().url().nullable(),
  createdAt: date().default(() => new Date()),
});

// parse and assert validity
let user = await userSchema.validate(await fetchUser());

type User = InferType<typeof userSchema>;
/* {
  name: string;
  age: number;
```

정규형을 통해 형식 검사도 할 수 있다.
라이브러리를 활용하기도 한다.
yup라이브러리. npmjs에 검색해보자

pages/auth/ChangePasswordPage.vue

```
const inputPassword = () => (error.value = "");

const resetError = () => (error.value = ""); ✓

const onSubmit = async () => {
  if (changePassword.newPassword !== changePassword.newPassword2) {
    error.value = '새 비밀번호가 일치하지 않습니다.';
    return;
  }

  try {
    await authApi.changePassword(changePassword);
    alert('비밀번호를 수정했습니다.');
```

router.push({ name: 'profile' }); 경로 기반이 아닌 라우터 이름 기반으로 이동

```
  } catch (e) {
    error.value = e.response.data;
  }
};
</script>
```

PASSWORDmismatchException 발생하면 백엔드 어드바이스가
잡아서 처리할 예정

pages/auth/ChangePasswordPage.vue

```
<template>
  <div class="mt-5 mx-auto" style="width: 500px">
    <h1 class="my-5">
      <i class="fa-solid fa-lock"> </i>
      비밀번호 변경
    </h1>

    <form @submit.prevent="onSubmit">
      <div class="mb-3">
        <label for="password" class="form-label">
          <i class="fa-solid fa-lock"> </i>
          이전 비밀번호:
        </label>
        <input type="password" class="form-control" placeholder="이전 비밀번호" v-model="changePassword.oldPassword"
          @input="inputPassword" />
      </div>
```

인풋 이벤트 -> 키보드 입력 발생시 바로

chang이벤트는 -> 입력이 끝나거나
포커스가 바뀌거나 할때 호출됨

pages/auth/ChangePasswordPage.vue

```
<div class="mb-3">
  <label for="password" class="form-label">
    <i class="fa-solid fa-lock"> </i>
    새 비밀번호:
  </label>
  <input type="password" class="form-control" placeholder="새 비밀번호" v-model="changePassword.newPassword"
    @input="resetError" />
</div>

<div class="mb-3">
  <label for="password" class="form-label">
    <i class="fa-solid fa-lock"> </i>
    새 비밀번호 확인:
  </label>
  <input type="password" class="form-control" placeholder="새 비밀번호 확인" v-model="changePassword.newPassword2"
    @input="resetError" />
</div>

<div v-if="error" class="text-danger">{{ error }}</div>
```

pages/auth/ChangePasswordPage.vue

```
<button type="submit" class="btn btn-primary mt-4" :disabled="disableSubmit">  
  <i class="fa-solid fa-check"> </i>  
  확인  
</button>  
</form>  
</div>  
</template>
```


Security Config

✓ Spring Security 설정

○ 인증 요구 경로

- POST :: /api/member
- PUT:: /api/member
- PUT:: /api/member/*/changepassword



로그인이 필요하지 않는
/create /;login /checkDuplicate

로그인이 필요한 경로는 따로 있다.

security config의 config메서드의
web.ignoring(). antMatcher에서
인자로 "/api/member/**"가 있다. 이건 모두 무시하라는 건데
좀더 세부적으로 나누게 수정하자.
일단 저 경로를 지우고

SecurityConfig.java

```
@Override
public void configure(HttpSecurity http) throws Exception {
```

```
...
http
```

```
    .authorizeRequests()
```

```
    .antMatchers(HttpMethod.OPTIONS).permitAll()
```

```
    // .antMatchers(HttpMethod.POST, "/api/member").authenticated() 이걸 빼고. 회원가입하는데 인증이 될필요는X
    .antMatchers(HttpMethod.PUT, "/api/member", "/api/member/*/changepassword").authenticated() ✓
    .anyRequest().permitAll();
```

```
http.httpBasic().disable() // 기본 HTTP 인증 비활성화
```

```
    .csrf().disable() // CSRF 비활성화
```

```
    .formLogin().disable() // formLogin 비활성화 □ 관련 필터 해제
```

```
    .sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS); // 세션 생성 모드 설정
```

```
}
```

/api/admin/** == 저 경로를 포함한 모든 하위 경로.
는
hasRole(어드민 권한만) 하면 된다.

admin입장에서 회원관리 기능이 어떤것이 필요할까를 고민해봐야한다.

현재 만들어진 것은 실제 웹 사이트 템플릿. 활용할 수 있는. 그전꺼는 학습을 위한 것.