

v.0.0.1

최병권, 이소우, 김소윤, 김성중



NUTRI CHECK MANAGER

Contents



1. Introduce to Program

- 1-1. 기획의도
- 1-2. 소프트웨어 구성

2. Details

- 2-1. 메인 화면
 - 2-1-1. 구현
- 2-2. 영양제 관리
 - 2-2-1. 영양제 등록
 - 2-2-2. 영양제 리스트
 - 2-2-3. 영양제 상세 조회
 - 2-2-4. 영양제 정보 수정
 - 2-2-5. 영양제 검색
- 2-3. 복용 내역 관리
 - 2-3-1. 복용 내역 목록

3. API

- 2-3. Swagger-ui 화면

A collection of yellow, oval-shaped capsules or softgels scattered across a dark, reflective surface. The capsules are translucent, showing a bright yellow interior. The lighting creates highlights and reflections on the capsules and the surface. The background transitions from a lighter grey on the left to a dark black on the right.

Introduce to Program

1-1. 기획의도

「 건강관리 필수 아이템인 '영양제'
요즘 영양제를 먹는 사람은 많다
바쁜 현대인들에겐 영양제를 챙겨
먹는 것조차 일이 되곤 한다.
까먹고, 더 먹고, 넘기고.....
이런 불상사를 막고자 만든 API
이 API를 이용하면 잊어버리지
않고 영양제 복용이 가능하다
이러한 의도로 이 API는 제작되었다 」

1-2. 소프트웨어 구성



- JAVA
- Spring Boot
 - JPA
- Postgres



- Vue.js
- Javascript
 - Nuxt



Details

2-2-1. 영양제 등록_Java_PillRequest(model)

```

@Getter
@Setter
public class PillRequest {

    no usages
    @ApiModelProperty(notes = "영양제 이름 (1~35자)", required = true)
    @NotNull
    @Length(min = 1, max = 35)
    private String pillName;

    no usages
    @ApiModelProperty(notes = "영양제 제조사 (1~20 자)", required = true)
    @NotNull
    @Length(min = 1, max = 20) // string은 길이조절을 해야해서 min,max lenght를 이용해서 사용
    private String pillCompany;

    no usages
    @ApiModelProperty(notes = "영양제 가격 (0 이상)", required = true)
    @NotNull
    @Min(value = 0) // integer은 값 자체의 숫자의 범위를 지정하는거라 바로 min,max쓸
    private Integer price;

    no usages
    @ApiModelProperty(notes = "영양 정보", required = true)
    @NotNull
    @Enumerated(value = EnumType.STRING)
    private NutriInfo nutriInfo;

    no usages
    @ApiModelProperty(notes = "영양제 복용시작일자 (yyyy-mm-dd)", required = true)
    @NotNull
    private LocalDate startDate;

    no usages
    @ApiModelProperty(notes = "영양제 유통기한 (yyyy-mm-dd)", required = true)
    @NotNull
    private LocalDate expiationDate;

```

```

    @ApiModelProperty(notes = "영양제 총 수량(1회 기준) (1이상)", required = true)
    @NotNull
    @Min(value = 1)
    private Integer totalQuantity;

    no usages
    @ApiModelProperty(notes = "섭취 주기 여부", required = true)
    @NotNull
    private Boolean isDaily;

    no usages
    @ApiModelProperty(notes = "1회 복용 횟수 (1 이상)", required = true)
    @NotNull
    @Min(value = 1)
    private Integer oneTimeQuantity;
}

```

- DB에 저장할 목적으로 영양제 정보를 수집하기 위해 PillRequest를 다음과 같이 구성하였다.
- Enums를 활용하여 영양 정보 생성하였고 EnumType을 String으로 하여 추후 보완 및 저장에 용이하도록 하였다.
- 입력 값 오류를 막기 위한 Valid 사용을 위해 NotNull, Length, Min, Max등 기준 값을 지정하였다.
- Swagger 및 코드 내에 매개변수의 이름,필수 값 등을 지정하기 위해 ApiModelProperty를 활용하였다.

2-2-1. 영양제 등록_Java_PillService, PillController

```
@Service
@RequiredArgsConstructor
public class PillService {

    7 usages
    private final PillRepository pillRepository;

    SO WOO LEE
    public void setPill(PillRequest request) {
        LocalDate today = LocalDate.now(); // today 변수 만들기

        //오늘이 받을 값보다 나중(after)이면 true
        if(today.isAfter(request.getExpiationDate())) throw new CDataPastException();
        Pill addData = new Pill.PillBuilder(request).build();

        pillRepository.save(addData);
    }
}
```

- Repository에 JPA를 구현 및 가동하였고 RequiredArgsConstructor 활용하여 함께 쓰이도록 했다.
- 영양정보를 가져와 setting해주는 method를 생성하였다.
- CDataPastException 생성으로 영양제 유통기한 날짜가 지난 정보 입력 시 아래와 같은 메시지를 띄우게 하였다.

```
, DATE_PAST( code: -20000, msg: "유통기한이 지난 제품입니다.")
```

- Pill entity에서 활용한 builder를 통해 번거로운 setter작업을 없애 실수를 줄여 DB저장에 정확도를 높였다.

```
@Api(tags = "영양제 목록")
@RestController
@RequiredArgsConstructor
@RequestMapping("/v1/pill")
public class PillController {

    6 usages
    private final PillService pillService;

    no usages SO WOO LEE
    @ApiOperation(value = "영양제 등록")
    @PostMapping("/data")
    public CommonResult setPill(@RequestBody @Valid PillRequest request) {
        pillService.setPill(request);

        return ResponseService.getSuccessResult();
    }
}
```

- Api를 사용해 Swagger의 가독성을 높였다.
- Body로 사용자에게 값을 받기 위해 PostMapping을 사용했고 RequestBody의 오류 범실을 줄이기 위해 Valid를 사용하였다.
- ResponseService를 따로 생성하여 결과 값을 아래와 같이 지정해 주어 return값의 질을 높였다.

```
SUCCESS( code: 0, msg: "성공하였습니다.")
, FAILED( code: -1, msg: "실패하였습니다.")
```


2-2-1. 영양제 등록_Vue

```
<template>
  <div>
    <div>
      <h1 class="custom-title-h1">▶ 영양제 등록 </h1>
    </div>
    <div style="text-align: center">
      <el-form :model="ruleForm" :rules="rules" ref="ruleForm" label-width="120px" class="demo-ruleForm">
        <br />
        <br />
        <el-form-item label="영양제 이름" prop="pillName">
          <el-input v-model="ruleForm.pillName"></el-input>
        </el-form-item>
        <br />
        <br />
        <el-form-item label="영양제 제조사" prop="pillCompany">
          <el-input v-model="ruleForm.pillCompany"></el-input>
        </el-form-item>
        <br />
        <br />
        <el-form-item label="영양제 가격" prop="price">
          <el-input v-model="ruleForm.price"></el-input>
        </el-form-item>
        <br />
        <br />
        <el-form-item label="영양 정보" prop="nutriInfo">
          <el-select v-model="ruleForm.nutriInfo" placeholder="영양정보를 선택해주세요">
            <el-option label="루테인" value="LUTEIN"></el-option>
            <el-option label="철분" value="IRON"></el-option>
            <el-option label="미네랄" value="MINERALS"></el-option>
            <el-option label="오메가3" value="OMEGA3"></el-option>
            <el-option label="프로폴리스" value="PROPOLIS"></el-option>
            <el-option label="프로바이오틱스" value="PROBIOTICS"></el-option>
            <el-option label="칼슘" value="CALCIUM"></el-option>
          </el-select>
        </el-form-item>
        <br />
        <br />
        <el-row>
          <el-col :span="11">
            <el-form-item label="복용시작일자" required>
              <el-form-item prop="startDate">
                <el-date-picker type="date" placeholder="Pick a date" v-

```

< Template

```
model="ruleForm.startDate"
                style="width: 100%;"></el-date-picker>
              </el-form-item>
            </el-form-item>
          </el-col>
          <el-col :span="11">
            <el-form-item label="유통기한" required>
              <el-form-item prop="expiationDate">
                <el-date-picker type="date" placeholder="Pick a date" v-
model="ruleForm.expiationDate" style="width: 100%;"></el-date-picker>
              </el-form-item>
            </el-form-item>
          </el-col>
        </el-row>
        <br />
        <br />
        <el-row>
          <el-col :span="8">
            <el-form-item>
              <el-switch
                v-model="ruleForm.isDaily"
                active-text="일간 섭취"
                inactive-text="주간 섭취">
              </el-switch>
            </el-form-item>
          </el-col>
          <el-col :span="8">
            <el-form-item label="영양제 총 수량" prop="totalQuantity">
              <el-input-number v-model="ruleForm.totalQuantity" :min="1"></el-input-
number>
            </el-form-item>
          </el-col>
          <el-col :span="8">
            <el-form-item label="1회 복용 횟수" prop="oneTimeQuantity">
              <el-input-number v-model="ruleForm.oneTimeQuantity" :min="1"></el-input-
number>
            </el-form-item>
          </el-col>
        </el-row>
        <br />
        <br />
        <br />
        <br />
        <el-form-item>
          <el-button type="primary" @click="doSetData">등록</el-button>
          <el-button>취소</el-button>
        </el-form-item>
      </el-form>
    </div>
  </div>
</template>
```

2-2-1. 영양제 등록_Vue

```
<script>
export default {
  data() {
    return {
      ruleForm: {
        pillName: '',
        pillCompany: '',
        price: '',
        nutriInfo: '',
        startDate: '',
        expiationDate: '',
        isDaily: true,
        totalQuantity: '',
        oneTimeQuantity: ''
      },
      rules: {
        pillName: [
          { required: true, message: '필수값입니다.', trigger: 'blur' },
          { min: 1, max: 35, message: '1~35자 내로 작성', trigger: 'blur' }
        ],
        pillCompany: [
          { required: true, message: '필수값입니다', trigger: 'blur' },
          { min: 1, max: 20, message: '1~20자 내로 작성 ', trigger: 'blur' }
        ],
        price: [
          { required: true, message: '필수값입니다', trigger: 'blur' },
          { min: 0, message: '0 이상 작성', trigger: 'blur' }
        ],
        nutriInfo: [
          { required: true, message: '필수값입니다', trigger: 'change' }
        ],
        startDate: [
          { type: 'date', required: true, message: '필수값입니다', trigger: 'change' }
        ],
        expiationDate: [
          { type: 'date', required: true, message: '필수값입니다', trigger: 'change' }
        ],
        totalQuantity: [
          { required: true, message: '필수값입니다', trigger: 'blur' },
        ],
        oneTimeQuantity: [
          { required: true, message: '필수값입니다', trigger: 'blur' },
        ]
      }
    }
  },
  methods: {
    doSetData() {
      let formName = 'ruleForm'
      this.$refs[formName].validate(valid => {
        if (valid) {
          this.setData()
        } else {
          return false
        }
      })
    },
    setData() {
      let payload = {
        pillName: this.ruleForm.pillName,
        pillCompany: this.ruleForm.pillCompany,
        price: Number(this.ruleForm.price),
        nutriInfo: this.ruleForm.nutriInfo,
        startDate: this.ruleForm.startDate,
        expiationDate: this.ruleForm.expiationDate,
        isDaily: this.ruleForm.isDaily,
        totalQuantity: this.ruleForm.totalQuantity,
        oneTimeQuantity: this.ruleForm.oneTimeQuantity
      }
      this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, true)
      this.$store.dispatch(this.$pillConstants.DO_CREATE, payload)
        .then(res => {
          this.$toast.success(res.data.msg)
          this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, false)
          this.$router.replace('/pill/all-list')
        })
        .catch(err => {
          this.$toast.error(err.response.data.msg)
          this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, false)
        })
    }
  }
}
</script>
```

< Script Data

```


  methods: {
    doSetData() {
      let formName = 'ruleForm'
      this.$refs[formName].validate(valid => {
        if (valid) {
          this.setData()
        } else {
          return false
        }
      })
    },
    setData() {
      let payload = {
        pillName: this.ruleForm.pillName,
        pillCompany: this.ruleForm.pillCompany,
        price: Number(this.ruleForm.price),
        nutriInfo: this.ruleForm.nutriInfo,
        startDate: this.ruleForm.startDate,
        expiationDate: this.ruleForm.expiationDate,
        isDaily: this.ruleForm.isDaily,
        totalQuantity: this.ruleForm.totalQuantity,
        oneTimeQuantity: this.ruleForm.oneTimeQuantity
      }
      this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, true)
      this.$store.dispatch(this.$pillConstants.DO_CREATE, payload)
        .then(res => {
          this.$toast.success(res.data.msg)
          this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, false)
          this.$router.replace('/pill/all-list')
        })
        .catch(err => {
          this.$toast.error(err.response.data.msg)
          this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, false)
        })
    }
  }
}
</script>

<style scoped>

</style>
```

Script Method >

2-2-1. 영양제 등록_ Vue(구현 화면)



님

HOME

대시보드

영양제 관리

영양제 목록

복용내역 관리

복용내역 목록

▶ 영양제 등록

* 영양제 이름

* 영양제 제조사

* 영양제 가격

* 영양 정보

영양정보를 선택해주세요

* 복용시작일자

Pick a date

* 유통기한

Pick a date

주간 섭취

일간 섭취

* 영양제 총 수량

-

1

+

* 1회 복용 횟수

-

1

+

등록

취소

2-2-2. 영양제 목록_Java_PillItem(model)

```
@Getter
@NoArgsConstructor(access = AccessLevel.PROTECTED)
public class PillItem {

    1 usage
    @ApiModelProperty(notes = "시퀀스")
    private Long id;
    1 usage
    @ApiModelProperty(notes = "영양제 이름")
    private String pillName;
    1 usage
    @ApiModelProperty(notes = "영양제 제조사")
    private String pillCompany;
    1 usage
    @ApiModelProperty(notes = "영양제 가격")
    private Integer pillPrice;
    1 usage
    @ApiModelProperty(notes = "현재 복용 여부")
    private String isEat;

    1 usage SO WOO LEE
    private PillItem(PillItemBuilder builder){
        this.id = builder.id;
        this.pillName = builder.pillName;
        this.pillCompany = builder.pillCompany;
        this.pillPrice = builder.pillPrice;
        this.isEat = builder.isEat;
    }

    3 usages SO WOO LEE
    public static class PillItemBuilder implements CommonModelBuilder<PillItem> {

        2 usages
        private final Long id;
        2 usages
        private final String pillName;
        2 usages
        private final String pillCompany;
        2 usages
        private final Integer pillPrice;
        2 usages
        private final String isEat;
```

```
public PillItemBuilder(Pill pill) {
    this.id = pill.getId();
    this.pillName = pill.getPillName();
    this.pillCompany = pill.getPillCompany();
    this.pillPrice = pill.getPrice();
    this.isEat = pill.getIsEat()? "O" : "X";
}

SO WOO LEE
@Override
public PillItem build() { return new PillItem( builder: this); }
```

- 영양제 목록에서 보여 줄 항목들을 지정하기 위해 PillItem이라는 이름으로 model을 생성하였다.
(현재 복용 중인 영양제를 기준으로 영양제 이름, 제조사, 가격을 보여줄 예정)
- NoArgsConstructor을 사용하여 외부에서 생성자를 만들 수 없도록 AccessLevel.Protected를 지정
- Builder패턴을 활용하여 오류 최소화 및 프로그램 실행 속도를 증가

2-2-2. 영양제 목록_Java_PillService, PillController

```
@Service
@RequiredArgsConstructor
public class PillService {

    7 usages
    private final PillRepository pillRepository;

    SO WOO LEE
    public void setPill(PillRequest request) {...}

    1 usage SO WOO LEE
    public PillResponse getPill(long id) {...}

    1 usage SO WOO LEE
    public ListResult<PillItem> getPills() {
        List<Pill> originList = pillRepository.findAll();

        List<PillItem> result = new LinkedList<>();

        originList.forEach(item -> result.add(new PillItem.PillItemBuilder(item).build()));

        return ListConvertService.settingResult(result);
    }
}
```

- 항목이 담긴 PillItem을 List<>를 사용하여 리스트화 하기 위해 getPills라는 이름의 method를 생성
- 재가공하기 전, 전체 리스트를 먼저 생성시켜야 하므로 LinkedList를 활용하여 리스트를 담을 공간인 result를 지정
- forEach문을 사용하여 원본데이터를 하나씩 builder에게 재가공한 뒤 만들어둔 result안에 저장되도록 하였다.

```
@Api(tags = "영양제 목록")
@RestController
@RequiredArgsConstructor
@RequestMapping("/v1/pill")
public class PillController {

    6 usages
    private final PillService pillService;

    no usages new *
    @ApiOperation(value = "영양제 내역 목록")
    @GetMapping("/all")
    public ListResult<PillItem> getPills() {

        return ResponseService.getListResult(pillService.getPills(), isSuccess: true);
    }
}
```

- Api를 통한 이름 및 값을 명시하여 Swagger 및 코드 작성에 가독성 향상
- PillService에서 값을 받아와야 작동할 수 있도록 하였다.
- ListResult를 활용하여 결과 값이 아래와 같이 도출될 수 있도록 지정

```
private List<T> list;
no usages
private Long totalItemCount;
no usages
private Integer totalPages;
no usages
private Integer currentPage;
```

2-2-2. 영양제 목록_Vue

```
<template>
  <div>
    <div>
      <h1 class="custom-title-h1">▶ 영양제 목록 </h1>
    </div>
    <br />
    <div style="text-align: right">
      <el-button type="success" circle @click.native="moveCreate()">등록</el-button>
    </div>
    <br />
    <el-row>
      <el-col :span="12" v-for="(item, index) in list" v-bind:key="index">
        <el-card class="box-card">
          <div slot="header" class="clearfix">
            <span>{{ item.pillName }}</span>
            <el-button style="float: right; padding: 3px 0" type="text"
              @click.native="moveDetail(item.id)">상세보기</el-button>
          </div>
          <div class="text item">
            제조사 : {{ item.pillCompany }}
          </div>
          <div class="text item">
            가격 : {{ item.pillPrice }} 원
          </div>
          <div class="text item">
            현재 복용상태 : {{ item.isEat }}
          </div>
        </el-card>
      </el-col>
    </el-row>
  </div>
</template>
```

```
<script>
export default {
  data() {
    return {
      list: []
    }
  },
  methods: {
    getList() {
      this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, true)
      this.$store.dispatch(this.$pillConstants.DO_ALL_LIST)
        .then((res) => {
          this.list = res.data.list
          this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, false)
        })
        .catch((err) => {
          this.$toast.error(err.response.data.msg)
          this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, false)
        })
    },
    delData(id) {
      if (confirm('정말 삭제하시겠습니까?')) { // javascript.confirm 검색하기
        this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, true)
        this.$store.dispatch(this.$testDataConstants.DO_DELETE, {id: id})
          .then((res) => {
            this.$toast.success(res.data.msg)
            this.getList()
            this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, false)
          })
          .catch((err) => {
            this.$toast.error(err.response.data.msg)
            this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, false)
          })
    },
    moveCreate() {
      this.$router.push('/pill/form')
    },
    moveDetail(id) {
      let linkBaseUrl = '/pill/detail/'
      this.$router.push(`${linkBaseUrl}${id}`)
    },
    moveEdit(id) {
      let linkBaseUrl = '/customer/edit/'
      this.$router.push(`${linkBaseUrl}${id}`)
    }
  },
  created() {
    this.getList()
  },
  mounted() {
    this.$store.commit(this.$menuConstants.FETCH_SELECTED_MENU, 'MEMBER_LIST')
  },
}
```

```
</script>

<style>
.text {
  font-size: 14px;
}

.item {
  margin-bottom: 10px;
}

.clearfix:before,
.clearfix:after {
  display: table;
  content: "";
}

.clearfix:after {
  clear: both
}


.box-card {
  width: 450px;
}
</style>
```

> Template

> Script_method

> style

2-2-2. 영양제 목록_Vue(구현화면)



님

HOME

대시보드

영양제 관리

영양제 목록

복용내역 관리

복용내역 목록

영양제 목록

등록

지아잔틴 164 500mg

상세보기

제조사 : 뉴트리원
가격 : 7800 원
현재 복용상태 : O

로알 프로폴리스 플라보노이드 17 900mg

상세보기

제조사 : 순수식품
가격 : 13250 원
현재 복용상태 : O

관절엔 콘드로이친 1200 900mg

상세보기

제조사 : 대광약품
가격 : 28710 원
현재 복용상태 : O

장튼튼 유산균 100억가지

상세보기

제조사 : 락토픽
가격 : 34500 원
현재 복용상태 : O

퓨어 알티지 r200

상세보기

제조사 : 고려은단
가격 : 46900 원
현재 복용상태 : O

2-2-3. 영양제 정보 상세 조회_Java_PillResponse

```

@ApiModelProperty(notes = "영양제 시퀀스")
private Long id;

1 usage
@ApiModelProperty(notes = "제조사")
private String pillCompany;

1 usage
@ApiModelProperty(notes = "영양제")
private String pillName;

1 usage
@ApiModelProperty(notes = "가격")
private String pillPriceName;

1 usage
@ApiModelProperty(notes = "영양정보")
private String nutriInfoName;

1 usage
@ApiModelProperty(notes = "복용시작일자 (yyyy-mm-dd)")
private LocalDate startDate;

1 usage
@ApiModelProperty(notes = "유통기한 (yyyy-mm-dd)")
private LocalDate expirationDate;

1 usage
@ApiModelProperty(notes = "총 수량")
private Integer totalQuantity;

1 usage
@ApiModelProperty(notes = "섭취주기")
private Boolean isDaily; // 수정창으로 보낸다

1 usage
@ApiModelProperty(notes = "섭취주기 한글명")
private String isDailyName; // 상세페이지로 보낸다

```

```

public PillResponseBuilder(Pill pill) {
    this.id = pill.getId();
    this.pillCompany = pill.getPillCompany();
    this.pillName = pill.getPillName();
    this.pillPriceName = pill.getPillPrice() + "원";
    this.nutriInfoName = pill.getNutriInfo().getName();
    this.startDate = pill.getStartDate();
    this.expirationDate = pill.getExpirationDate();
    this.totalQuantity = pill.getTotalQuantity();
    this.isDaily = pill.getIsDaily();
    this.isDailyName = pill.getIsDaily()? "일간" : "주간";
    this.oneTimeQuantity = pill.getOneTimeQuantity();
    this.remainingQuantity = pill.getRemainingQuantity();
    this.isEat = pill.getIsEat();
    this.isEatName = pill.getIsEat()? "복용중" : "복용하지 않음";
}

```

- 영양제 정보를 상세 조회할 수 있는 페이지
- Boolean 타입은 Boolean과 String 두가지 타입으로 분류
- Boolean은 정보 수정 시, switch에 기존에 등록되어 있던 값을 전달할 때 사용
- String은 사용자가 해당 Boolean의 값이 어떤 뜻인지 알아 보기 쉽도록 추가

2-2-3. 영양제 상세 조회_Vue

< Template

```
<template>
  <div>
    <div v-if="detailInfo != null">
      <el-descriptions class="margin-top" title="상세내역" :column="3" :size="size" border>
        <template slot="extra">
          <el-button type="primary" size="small" @click.native="moveEdit(id)">수정</el-button>
          <el-button size="small" @click.native="moveList()">뒤로</el-button>
        </template>
        <el-descriptions-item>
          <template slot="label">
            <i class="el-icon-orange"></i>
            영양제
          </template>
          {{ detailInfo.pillName }}
        </el-descriptions-item>
        <el-descriptions-item>
          <template slot="label">
            <i class="el-icon-office-building"></i>
            제조사
          </template>
          {{ detailInfo.pillCompany }}
        </el-descriptions-item>
        <el-descriptions-item>
          <template slot="label">
            <i class="el-icon-coin"></i>
            가격
          </template>
          {{ detailInfo.pillPriceName }}
        </el-descriptions-item>
      </el-descriptions>
    </div>
  </div>
</template>
```

2-2-3. 영양제 정보 상세 조회_Vue

```
<script>
export default {
  asyncData({params}) {
    return {
      id: params.id
    }
  },
  validate({params}) {
    return /^d+$/ .test(params.id)
  },
  data() {
    return {
      id: null,
      detailInfo: null,
      size: ''
    }
  },
  methods: {
    async getDetail() {
      this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, true)
      await this.$store.dispatch(this.$pillConstants.DO_DETAIL, {id: this.id})
        .then((res) => {
          this.detailInfo = res.data.data
          this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, false)
        })
        .catch((err) => {
          this.$toast.error(err.response.data.msg)
          this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, false)
        })
    },
    moveEdit(id) {
      let linkBaseUrl = '/pill/edit/'
      this.$router.push(`${linkBaseUrl}${id}`)
    },
    moveList() {
      this.$router.push('/pill/list')
    }
  },
  created() {
    this.getDetail()
  },
  mounted() {
    this.$store.commit(this.$menuConstants.FETCH_SELECTED_MENU, 'MEMBER_DETAIL')
  },
}
</script>
```

```
<div v-if="detailInfo != null">
  <el-descriptions class="margin-top" title="상세내역" :column="3" :size="size" border>
    <template slot="extra">
      <el-button type="primary" size="small" @click.native="moveEdit(id)">수정</el-button>
      <el-button size="small" @click.native="moveList()">뒤로</el-button>
    </template>
    <el-descriptions-item>
```

- 영양제 정보를 상세 조회할 수 있는 페이지입니다.
- 가독성을 위해 el-descriptions를 사용해 표로 만들었습니다.

2-2-3. 영양제 정보 상세 조회 _ Vue(구현화면)

상세내역

수정

뒤로

⊗ 영양제	루테인 1000	Ⓜ 제조사	가상제약	Ⓜ 가격	10000원
Ⓜ 영양정보	루테인	Ⓜ 복용시작일자	2023-02-10	Ⓜ 유통기한	2023-12-31
Ⓜ 총 수량	50	Ⓜ 섭취주기	일간	⊗ 1회 복용량	2
⊗ 잔여량	10	Ⓜ 복용여부	복용중		

2-2-4. 영양제 정보 수정_Java_PillInfoUpdateRequest

```
public class PillInfoUpdateRequest {  
    no usages  
    @ApiModelProperty(notes = "섭취주기", required = true)  
    @NotNull  
    private Boolean isDaily;  
  
    no usages  
    @ApiModelProperty(notes = "1회 복용량 (1 이상)", required = true)  
    @NotNull  
    @Min(value = 1)  
    private Integer oneTimeQuantity;  
  
    no usages  
    @ApiModelProperty(notes = "잔여량", required = true)  
    @NotNull  
    @Min(value = 0)  
    private Integer remainingQuantity;  
  
    no usages  
    @ApiModelProperty(notes = "복용여부", required = true)  
    @NotNull  
    private Boolean isEat;  
}
```

- 등록되어 있는 영양제 정보를 수정하는 기능
- 섭취 주기, 1회 복용량, 잔여 량, 복용 여부 등을 수정 가능

2-2-4. 영양제 정보 수정_Vue

```
<template>
  <div v-if="detailInfo != null">
    <el-form :model="ruleForm" :rules="rules" ref="ruleForm" label-width="auto">
      <el-row :gutter="15">
        <el-col :span="8">
          <el-form-item label="섭취주기" prop="isDaily" class="pb-4">
            <el-switch
              class="ml-3"
              v-model="ruleForm.isDaily"
              active-text="일간"
              inactive-text="주간">
            </el-switch>
          </el-form-item>
        </el-col>
      </el-row>
      <el-row :gutter="15">
        <el-col :span="8">
          <el-form-item label="1회 복용량" prop="oneTimeQuantity" class="pb-4">
            <el-input-number class="ml-3" size="small" v-model="ruleForm.oneTimeQuantity"
              @change="handleChange"
              :min="1">
            </el-input-number>
          </el-form-item>
        </el-col>
      </el-row>
      <el-row :gutter="15">
        <el-col :span="8">
          <el-form-item label="잔여량" prop="remainingQuantity" class="pb-4">
            <el-input-number class="ml-3" size="small" v-model="ruleForm.remainingQuantity"
              @change="handleChange"
              :min="0">
            </el-input-number>
          </el-form-item>
        </el-col>
      </el-row>
      <el-row :gutter="15">
        <el-col :span="8">
          <el-form-item label="복용여부" prop="isEat" class="pb-4">
            <el-switch
              class="ml-3"
              v-model="ruleForm.isEat"
              active-text="복용중"
              inactive-text="복용하지 않음">
            </el-switch>
          </el-form-item>
        </el-col>
      </el-row>
      <el-row :gutter="15">
        <el-col>
          <el-form-item>
            <el-button type="primary" @click="doPutData()">수정</el-button>
            <el-button @click="moveDetail(id)">취소</el-button>
          </el-form-item>
        </el-col>
      </el-row>
    </el-form>
  </div>
</template>
```

```
<script>
export default {
  asyncData({params}) {
    return {
      id: params.id
    }
  },
  validate({params}) {
    return /^d+$/i.test(params.id)
  },
  data() {
    return {
      id: null,
      detailInfo: null,
      ruleForm: {
        isDaily: '',
        oneTimeQuantity: '',
        remainingQuantity: '',
        isEat: ''
      },
      rules: {
        oneTimeQuantity: [
          {required: true, message: '이 값은 필수입니다.', trigger: 'blur'}
        ],
        remainingQuantity: [
          {required: true, message: '이 값은 필수입니다.', trigger: 'blur'}
        ]
      }
    }
  },
  methods: {
    async getDetail() {
      this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, true)
      await this.$store.dispatch(this.$pillConstants.DO_DETAIL, {id: this.id})
      .then((res) => {
        this.detailInfo = res.data.data
        this.ruleForm.isDaily = res.data.data.isDaily
        this.ruleForm.oneTimeQuantity = res.data.data.oneTimeQuantity
        this.ruleForm.remainingQuantity = res.data.data.remainingQuantity
        this.ruleForm.isEat = res.data.data.isEat
        this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, false)
      })
      .catch((err) => {
        this.$toast.error(err.response.data.msg)
        this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, false)
      })
    },
    doPutData() {
      let formName = 'ruleForm'
      this.$refs[formName].validate(valid => {
        if (valid) {
          this.putData()
        } else {
          return false
        }
      })
    }
  }
}
```

섭취주기 주간 ☒ 일간

* 1회 복용량 - 2 +

* 잔여량 - 35 +

복용여부 복용하지 않음 ☒ 복용중

수정 취소

> 구현 화면

- 1회 복용량과 잔여량은 필수 값이기 때문에 validate를 이용하여 null이 되지 않도록 작성
- 섭취 주기와 복용 여부는 switch를 이용하여 값이 필수로 입력되기 때문에 validate를 생략

> Template

> Script

2-2-4. 영양제 검색_Java_PillService

1 usage

```
public ListResult<PillItem> getPillsByLikeName(String searchName) { // -> 이름이 일치하는 것
    // 1. 영양제 이름에 검색어가 포함된 모든 리스트를 가져옴
    List<Pill> originList = pillRepository.findAllByPillNameLikeOrderByPillNameAsc( searchName: "%" + searchName + "%");
    // 2. 결과 값을 담을 리스트를 준비함
    // %searchName% 즉, searchName 출력
    // %김 = 김으로 끝나는 단어, 김% = 김으로 시작하는 단어, %김% = 김이라는 단어가 포함된 단어
    List<PillItem> result = new LinkedList<>();
    // 3. 원본리스트 originList 를 하나씩 던져주면서 결과 값을 담을 리스트인 result 에 PillItem 을 새로 생성하고 반복해서 넣음
    // 근데 PillItem 은 그 안에 빌더를 통해서만 생성이 가능하므로 그 안에 빌더 PillItemBuilder 에게 원본인 item 을 주면서 PillItem 생성을 요청
    originList.forEach(item -> result.add(new PillItem.SearchItemBuilder(item).build()));
    // List<PillItem> 모양의 result 가 전부 가공되었으므로 ListConvertService 에 settingResult 라는 사람에게 ListResult 로 바꿔달라고 요청
    // ListConvertService.settingResult 는 List<>를 바꿔주는 함수
    // ListResult 는 List 와 다르게 리스트 + 그 리스트가 총 아이템의 개수, 총 페이지 개수, 현재 페이지 개수를 세팅해주는 함수
    return ListConvertService.settingResult(result);
}
```

- 1, 2, 3 순서로 리스트를 조회 후 String(제조약 한글명)형으로 검색 조회 가능

2-3-1. 복용 내역 목록_Java_Entity(TakeHistory)

```
@Entity
@Getter
@NoArgsConstructor(access = AccessLevel.PROTECTED)
public class TakeHistory {
    no usages
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    1 usage
    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "pillId", nullable = false)
    private Pill pill;

    2 usages
    @Column(nullable = false)
    private LocalDate takeDate;

    2 usages
    @Column(nullable = false)
    private LocalTime takeTime;

    2 usages
    private String memo;

    public static class TakeHistoryBuilder implements CommonModelBuilder<TakeHistory> {
        2 usages
        private final Pill pill;
        2 usages
        private final LocalDate takeDate;
        2 usages
        private final LocalTime takeTime;
        2 usages
        private String memo;
```

```
public TakeHistoryBuilder(Pill pill, TakeHistoryRequest request) {
    this.pill = pill;
    this.takeDate = request.getTakeDate();
    this.takeTime = LocalTime.of(request.getTakeTimeHour(), request.getTakeTimeMinute());
}

2 usages
public TakeHistoryBuilder setMemo(String memo) {
    this.memo = memo;

    return this;
}

@Override
public TakeHistory build() { return new TakeHistory( builder: this); }
```

- TakeHistoryBuilder를 통해 Pill, takeDate, takeTime, memo 데이터를 받아옴
- GeneratedValue를 통해 id가 순차적으로 생성될 수 있도록 지정
- Entity에 pill, takeDate, takeTime줄을 추가
- 필수 값인 pill, takeDate, takeTime, memo 앞에 final을 타이핑하여 추가

2-3-1. 복용 내역 목록_Java_Model,Controller(TakeHistoryRequest, PillStatusController)

```

@Getter
@Setter
public class TakeHistoryRequest {

    no usages
    @ApiModelProperty(value = "복용 날짜(date)", required = true)
    @NotNull
    private LocalDate takeDate;

    no usages
    @ApiModelProperty(value = "복용 시간(hour)", required = true)
    @NotNull
    @Min(0)
    @Max(23)
    private Integer takeTimeHour;

    no usages
    @ApiModelProperty(value = "복용 시간(minute)", required = true)
    @NotNull
    @Min(0)
    @Max(59)
    private Integer takeTimeMinute;

    no usages
    private String memo;
}

```

```

@Api(tags = "영양제 조회")
@RestController
@RequiredArgsConstructor
@RequestMapping("/v1/pill-state")
public class PillStatusController {

    2 usages
    private final PillTakeService pillTakeService;

    no usages
    @ApiOperation(value = "영양제 이름, 제조사, 권장 복용 수")
    @PostMapping("/take/{id}")
    public CommonResult doTake(@PathVariable long id, @RequestBody @Valid TakeHistoryRequest request) {
        pillTakeService.doTake(id, request);

        return ResponseService.getSuccessResult();
    }
}

```

- 1-1. PillStatusControll는 pillTakeService와 함께 작업
- 1-2. ApiModelProperty를 통해 영양제 이름, 제조사, 권장 복용 수 라고 swagger에 표시
- 2-1. TakeHistoryRequest에도 마찬가지로 ApiModelProperty를 복용 날짜(년, 월, 일), 복용 시간(hour), 복용 분(minute)이라고 swagger에 표시

2-3-1. 복용 내역 목록 (1)_Vue

```
<template>
  <div>
    <div>
      <h1 class="custom-title-h1"> ▶ 복용내역 목록 </h1>
    </div>
    <el-table :data="list" stripe style="width: 100%">
      <el-table-column prop="pillFullName" label="영양제 이름 및 제조사"></el-table-column>
      <el-table-column prop="eatDate" label="복용일자"></el-table-column>
      <el-table-column prop="eatTime" label="복용시간"></el-table-column>
      <el-table-column prop="memo" label="메모"></el-table-column>
    </el-table>
  </div>
</template>

<script>
export default {
  data() {
    return {
      list: [],
    },
  },
  methods: {
    getList() {
      this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, true)
      this.$store.dispatch(this.$historyConstants.DO_LIST)
        .then((res) => {
          this.list = res.data.list
          this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, false)
        })
        .catch((err) => {
          this.$toast.error(err.response.data.msg)
          this.$store.commit(this.$customLoadingConstants.FETCH_LOADING_SHOW, false)
        })
    },
  },
  created() {
    this.getList()
  },
}
</script>
```


-> Template

복용내역 목록에 필요한 데이터

복용내역 리스트 메서드 생성

-> Script Method

2-3-1. 복용 내역 목록 (2)_Vue(구현)

<div></div> <div>님</div> <div><div>HOME</div><div>대시보드</div><div>영양제 관리</div><div>영양제 목록</div><div>복용내역 관리</div><div>복용내역 목록</div></div>	복용내역 목록			
	영양제 이름 및 제조사	복용일자	복용시간	메모
	관절엔 콘드로이친 1200 900mg 대광약품	2022-12-19	13:30:00	
	관절엔 콘드로이친 1200 900mg 대광약품	2022-12-24	13:30:00	
	관절엔 콘드로이친 1200 900mg 대광약품	2022-12-22	09:30:00	
	관절엔 콘드로이친 1200 900mg 대광약품	2022-12-18	19:30:00	
	관절엔 콘드로이친 1200 900mg 대광약품	2022-12-23	13:30:00	
	관절엔 콘드로이친 1200 900mg 대광약품	2022-12-20	14:30:00	
	관절엔 콘드로이친 1200 900mg 대광약품	2022-12-17	11:30:00	
	관절엔 콘드로이친 1200 900mg 대광약품	2022-12-21	10:30:00	
	관절엔 콘드로이친 1200 900mg 대광약품	2022-12-25	13:30:00	
	관절엔 콘드로이친 1200 900mg 대광약품	2022-12-26	13:30:00	

Swagger-ui 화면

swagger

Select a specV1

영양제 섭취 관리 프로그램 API V1

[Base URL: localhost:8081/]
<http://localhost:8081/v2/api-docs?group=V1>

Swagger API Docs
[Terms of service](#)
[cbk - Website](#)
[Send email to cbk](#)
[Licenses](#)

Authorize

복용 내역History Controller

GET/v1/history/all복용 내역 목록

영양제 목록Pill Controller

GET/v1/pill/all영양제 내역 목록

POST/v1/pill/data영양제 등록

GET/v1/pill/detail/{id}영양제 상세 내역

PUT/v1/pill/info/{id}영양제 정보 수정

GET/v1/pill/search영양제 이름으로 검색

오늘의 복용 현황Today Pill Eat Status Controller

POST/v1/nutri-status/do-eat/pill-id/{pillId}복용 내역 등록

GET/v1/nutri-status/today복용 현황 목록



The End