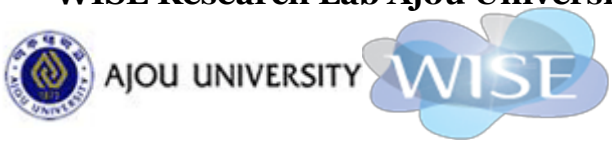
2018-2

웹 시스템 설계
과제

Assignment 2

Vue / Express with Mongoose

Composed by:
WISE Research Lab Ajou University



Preliminary

■ 실습 목표

- Mongoose 를 이용해 MongoDB 기반의 백엔드 API 를 개발 해본다.
- 프론트엔드(Vue)에서 AJAX 를 이용하여 앞서 만든 API 와 통신하는 방법을 익힌다.

■ Mongoose https://mongoosejs.com/docs/index.html

- Node.js 기반의 MongoDB ODM 라이브러리
- MongoDB driver 의 기본 CRUD 기능을 포함하며, Schema, Join(populate) 등 풍부한 기능을 추가로 지원
- Schema 기반의 Collection 모델링이 가능하고,
  Model 로 모델링 된 Document 들이 모인 Collection 을 관리함(CRUD)
- Callback / Promise 를 자유자재로 사용할 수 있는 유연한 API 제공

■ Mongoose Connect https://mongoosejs.com/docs/api.html#Connection

- MongoDB 서버에 연결해서 connection 객체 생성
- 반환된 Connection 객체를 이용해서 연결된 서버와 통신을 수행함

```
const mongoose = require('mongoose');
const conn = await  mongoose.connect('mongodb://example.com:27017/myDB_1',
{useNewUrlParser: true});
```

- **"mongodb://서버주소:포트/db 이름"**
- 포트를 명시하지 않을 경우 기본 포트(27017)로 연결

■ Schema https://mongoosejs.com/docs/guide.html

- 하나의 Collection 에 저장할 데이터 구조를 모델링하는 기능

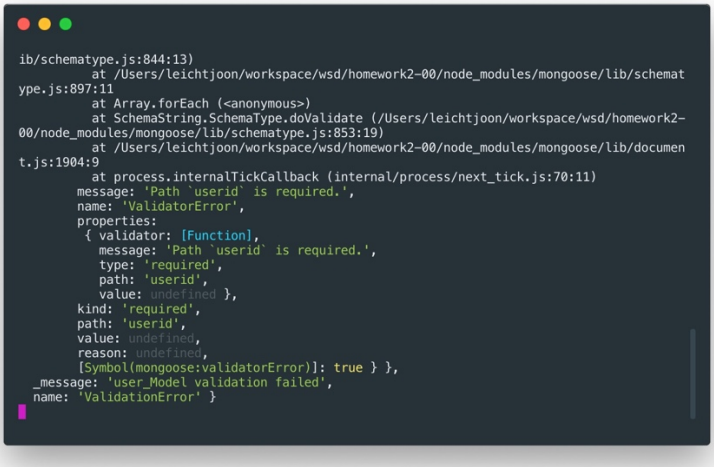
```
const userSchema = new Schema({
  idx: Number,
  userid: {
    type: String,
    unique: true
  name: {
    first: String,
    last: String
  },
  createdAt: {
    type: Date,
    default: Date.now
```

- 각 프로퍼티(userid, name, ...)를 필드라고 표현함
- 위의 userSchema 로 생성한 model 에서 데이터를 저장할 때 userSchema 에 존재하지 않는 필드를 포함한다면 해당 값들은 제외하고 DB 에 저장됨

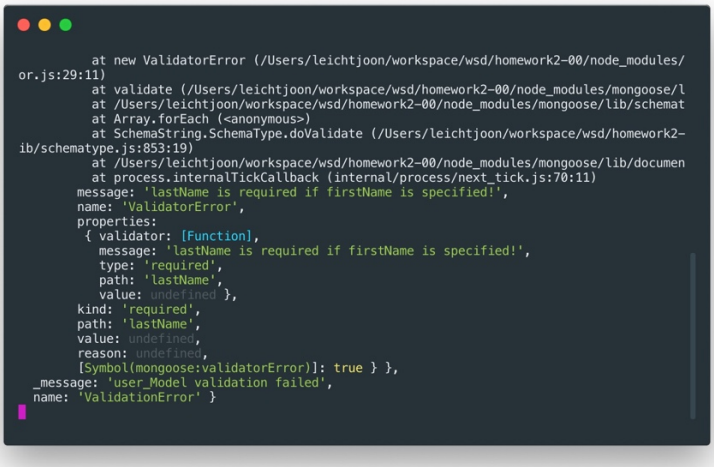
- **Unique**
  - ✓ 해당 Collection 안에서 고유한 값만 저장하는 필드임을 명시
  - ✓ unique:true 인 필드가 이미 저장된 다른 document 와 값이 중복된다면 저장 실패

- **Required**
  - ✓ 저장할 때 무조건 포함해야 하는 필드임을 명시
  - ✓ required:true 인 필드가 포함되지 않았다면 저장 실패
  - ✓ true/false 또는 특정 조건과 함께 에러 발생 시 메시지도 설정 가능

```
const userSchema = new Schema({
    userid: {
      type: String,
      required: true
    },
    fistName: {
      type: String,
    },
    lastName: {
      type: String,
      required: [
        //사용자 정의 메시지
        () => {return this.firstName !== null},
        'lastName is required if firstName is specified!'
      ]
    }
})
```



저장시 required:true 인 필드의 값이 없으면 발생하는 에러 메시지



사용자 정의 메시지 출력

- **Virtual**
  - ✓ mongoose 인스턴스에서만 존재하는 가상의 계산된 필드값
  - ✓ virtual 필드는 DB 에 저장되지 않음

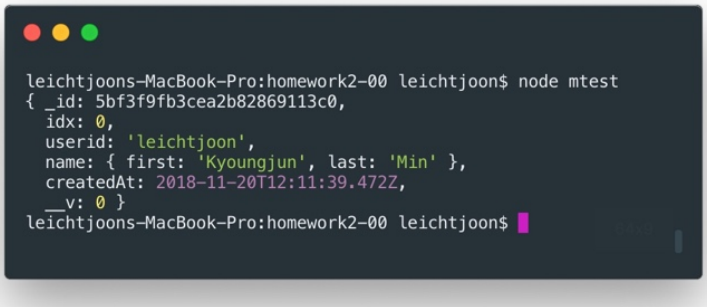
```
userSchema.virtual('fullName', () => {
    return `${this.firstName} ${this.lastName}`
})
// userData.fullName
```

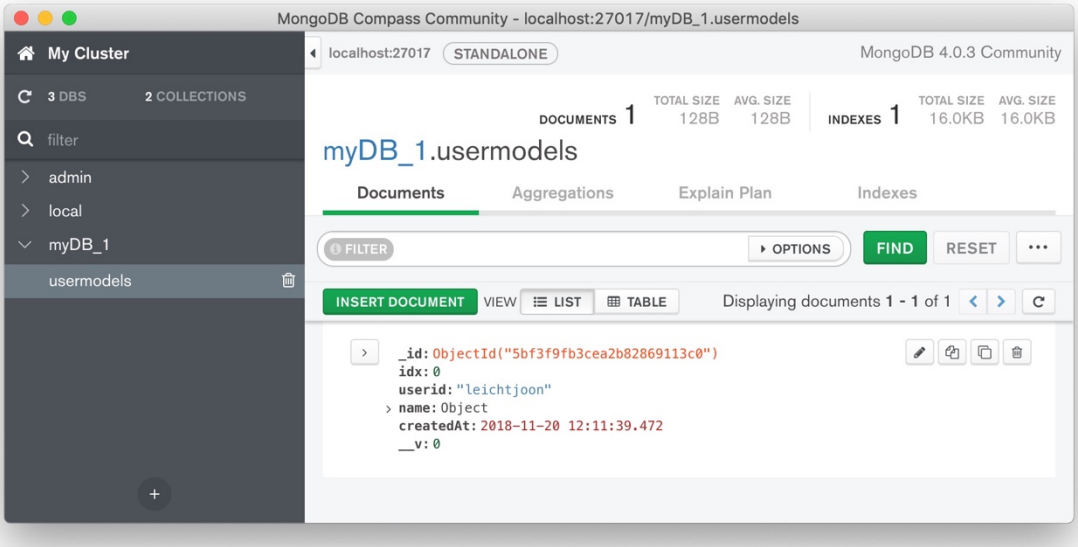
■ Model https://mongoosejs.com/docs/models.html

- MongoDB 에 Document 의 CRUD 작업을 수행하는 객체

```
const userModel = conn.model('userModel', userSchema)
try {
    const result = await userModel.create(/*..*/)
    console.log(result)
    conn.disconnect()
} catch(err) {
    console.error(err)
}
```

- 터미널과 Compass 에서 create 수행 결과 확인





■ Axios https://github.com/imcvampire/vue-axios

- Frontend AJAX 라이브러리인 axios 를 Vue 인스턴스에 전역으로 등록하여 사용

```
import Vue from 'vue'
import axios from 'axios'

Vue.prototype.$http = Vue.prototype.axios = Axios

Vue.axios.get(api, options).then((response) => {
    console.log(response.data)
  })
this.axios.get(api, options).then((response) => {
  console.log(response.data)
})
this.$http.post(api, options).then((response) => {
  console.log(response.data)
})
```

- Single File Component(*.vue)에서 사용하려면 this에서 axios 호출

Homework Assignment

■ 과제 목표

- Mongoose 기반의 Backend API 와 Vue 기반의 병원 진료 웹 애플리케이션을 개발한다.

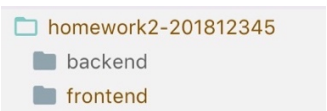
■ 구현 과제

- 환자들은 프론트엔드의 메인 페이지에서 두 가지를 확인할 수 있다.
  - ✓ 진료 진행중인 환자들의 목록
  - ✓ 진료 대기중인 환자들의 목록
- 또한 자신이 진료를 받고 싶으면 이름을 적고 대기번호를 부여받는다. 부여받은 대기번호는 메인페이지의 리스트에 노출된다.
- 의사는 자신의 정보를 등록 후 먼저 대기한 환자 순으로 진료실로 부를 수 있다.
- 의사가 진료를 마치면 성공적으로 완료되었음을 알린다. 진료가 완료되면 진료중인 환자 리스트에서 해당 환자는 사라진다.

■ 과제 참고자료

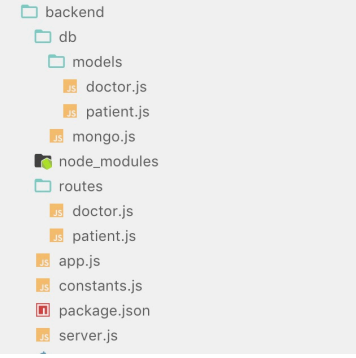
- Postman https://www.getpostman.com/
  - ✓ API 개발환경 툴
  - ✓ Backend 구현할 때 Postman 에서 GET, POST 요청을 보내서 테스트할 수 있음
- Mongoose Official Document https://mongoosejs.com/docs/guides.html
  - ✓ Connect, Schema, Model, Query 등의 API 에 대해 상세하게 설명되어 있음
- 강의노트 10. Express JS
- 강의노트 12. Mongoose
- 강의노트 {15, 16, 17} Vue
- 1018 실습문서#7
  - ✓ Model 에서의 CRUD 관련 API(create, updateOne, findOne 등)의 사용법은 해당 실습문서 참고

■ 전체 프로젝트 구조



- 전체 프로젝트 구조는 두 개의 프로젝트로 나뉘며, backend 와 frontend 는 서로 다른 프로세스로 동작해야 한다.

■ Backend 구현



- Express와 MongoDB를 사용하는 API 서버를 구현하는 디렉토리
- models 와 routes 를 제외한 Boilerplate code 를 제공할 것이며, 요구 기능에 만족하는 API 를 routes 에서 구현하고, 각 model 의 Schema 를 정의해야 함.
- 8000 포트에서 동작해야 함. 제공하는 기본 코드의 constants.js 참고
- "npm run start" 또는 "node server" 명령어로 실행 가능해야 함

- **Doctor Model**
  - ✓ 기본 필드 { doctorId, name office }
    - doctorId: index
    - name: 의사 이름
    - office: 해당 의사의 진료실

- **Patient Model**
  - ✓ 기본 필드 { patientId, name, isCalled, office }
  - ✓ patientId: index
  - ✓ name: 환자 이름
  - ✓ isCalled: 의사의 환자 호출 여부. 세 가지 값으로 status 정의
    - (1) 대기
    - (2) 호출됨(진료)
    - (3) 진료 완료
  - ✓ Office: 환자가 호출되면 가야 할 진료실(호출한 의사의 진료실)
  - ✓ 환자의 status 를 정의하는 방법은 상당히 많으므로, 위의 방식을 이용하지 않고 다른 필드를 추가해서 자신만의 구현으로 해결해도 무방함

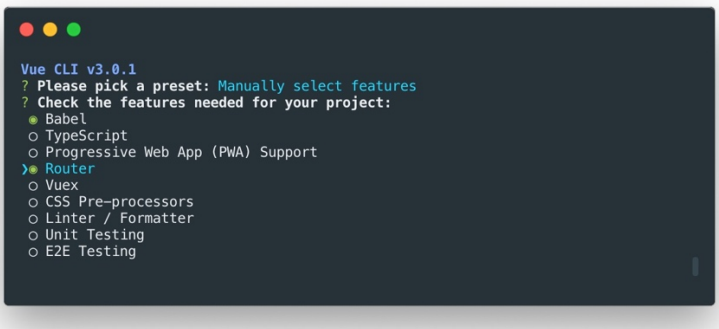
- Routes 에서는 아래의 6 가지 기능을 하는 API 를 구현해야 한다.

API Name	Method	Path	Arguments	API Description
의사 등록	POST	/doctor/register	name, office, password	의사의 정보를 전달 받아서 { doctorId, name, office }의 형태로 db 에 저장하는 API
환자 호출	GET	/doctor/call/:doctorName	:doctorName	의사가 대기자 중에서 가장 먼저 기다린 환자를 호출하는 API(status 변경)
진료 완료	GET	/doctor/done/:patientId	:patientId	진료가 끝난 환자는 호출된 환자의 status 를 진료 완료로 변경
의사 목록	GET	/doctor/list		등록된 모든 의사 목록을 뿌려주는 API
환자 등록	POST	/patient/register	name	환자가 이름을 남기면{ patientId, patientName } 형태로 진료 대기자 목록에 저장 patientId 가 대기번호로 사용됨
환자 목록	GET	/patient/list		{진료대기중인 환자의 목록, 현재 진료중인(호출된) 환자들의 목록}을 뿌려주는 API

✓ Mongoose connect 이후 각 router 파일마다 model을 불러와서 CRUD 작업을 호출한다.

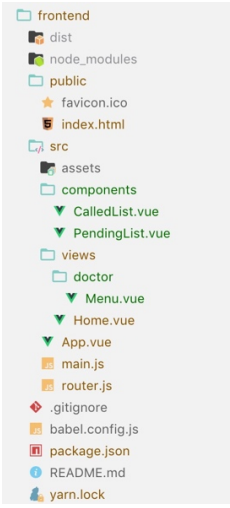
■ Frontend 구현

- @vue/cli v3.0 이상에서 프로젝트 생성
  - ✓ vue create homework2-201812345 (자신의 학번)
  - ✓ Preset 선택에서 Manually select features 선택 후, 아래와 같이 옵션을 고른다



```
Vue CLI v3.0.1
? Please pick a preset: Manually select features
? Check the features needed for your project: Babel, Router
? Use history mode for router? (Requires proper server setup for index fallback i
n production) Yes
? Where do you prefer placing config for Babel, PostCSS, ESLint, etc.? In package
.json
? Save this as a preset for future projects? No
```

✓ 프로젝트 생성이 끝나면 아래와 같은 구조로 프로젝트가 구성되며, 사용자가 접근 가능한 페이지는 총 2 개이다("/", "**/doctor/menu**")



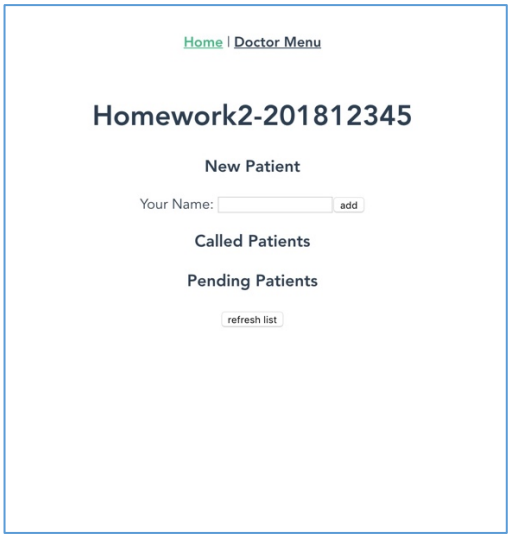
✓ Views/App.vue – 네비게이션을 상단에 두고 router path 에 따라 다르게 렌더링하는 router-view 배치

```
<template>
  <div id="app">
    <div id="nav">
      <router-link to="/">Home</router-link> |
      <router-link to="/doctor/menu">Doctor Menu</router-link>
    </div>
    <router-view/>
  </div>
</template>
```

✓ Views/Home.vue – 메인페이지("/")의 컴포넌트. 아래 4 개를 포함해야 한다.

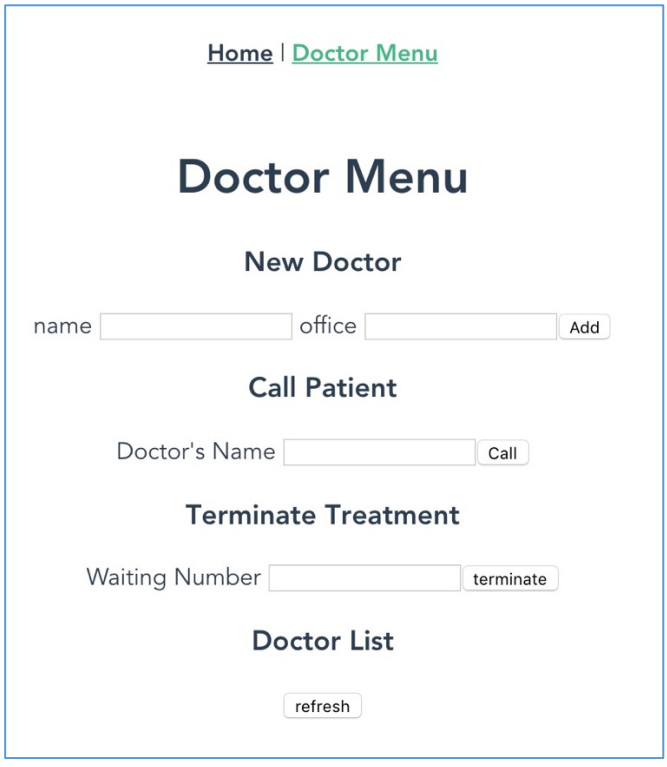
- 환자를 대기열에 추가하는 폼
- 진료중인(호출된) 환자 목록

- 대기중인 환자 목록
- 목록 새로고침 버튼



✓ Views/doctor/Menu.vue – 의사용 페이지("/doctor/menu")의 컴포넌트. 아래 3 개를 포함해야 한다.

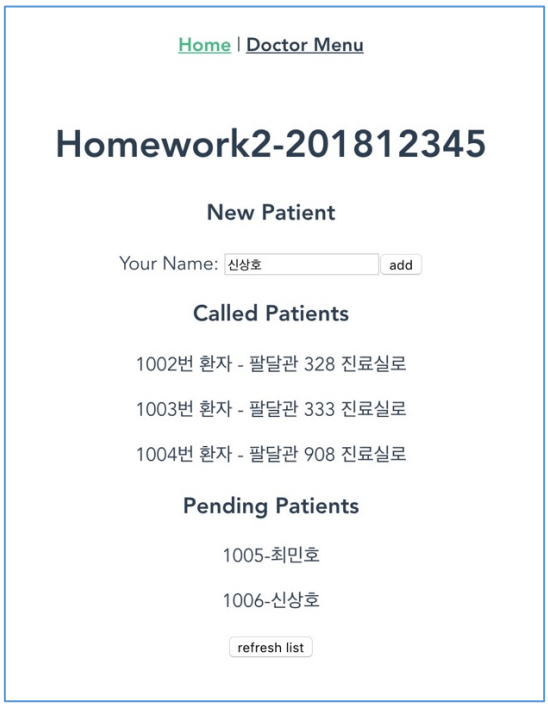
- 새로운 의사를 추가하는 폼
- 준비를 마친 의사가 환자를 호출하는 폼
- 자신이 진료를 마친 환자를 대기표에서 삭제하는 폼
- 현재 등록된 의사 목록 / 새로고침 버튼



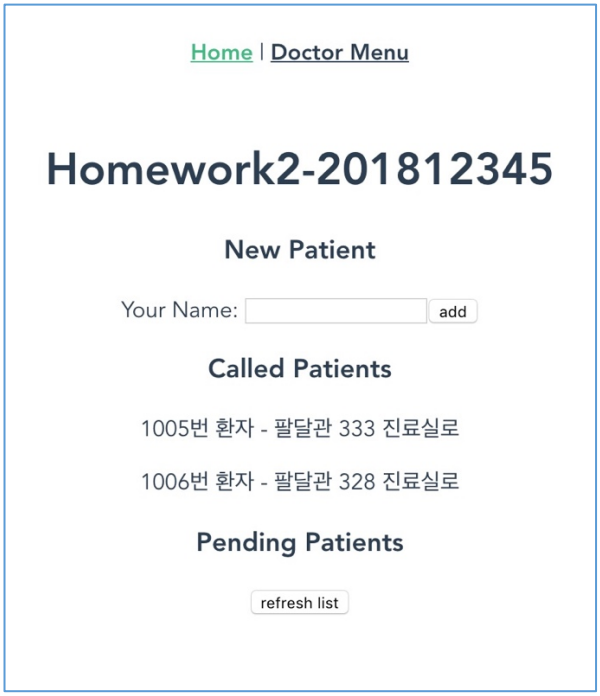
■ 실행 예시

- 메인 페이지의 모습 (5명의 대기중인 환자만 있는 경우)

- 메인 페이지의 모습 (대기중인 5명 중 3명이 호출되었을 경우)



- 메인 페이지의 모습 (2명의 진료가 끝나고 2명을 호출한 경우)



- 의사 메뉴 페이지의 모습 (등록된 의사 목록 확인)

Home | Doctor Menu

Doctor Menu

New Doctor

name office Add

Call Patient

Doctor's Name Call

Terminate Treatment

Waiting Number terminate

Doctor List

김의사-팔달관 328

박의사-팔달관 333

배의사-팔달관 908

최의사-원천관101

이국종-종합관 1004

refresh

■ 제출양식

- 프로젝트 폴더명은 homework2-{학번}으로 한다
  - Ex) homework2-201812345
- 제출하는 압축파일의 이름은 homework2-{학번}.zip 으로 한다
  - Ex) homework2-201812345.zip
- 명시된 파일명, 폴더명을 반드시 지키기 바랍니다. (감점요인)

■ 제출기한

- 11/28 23:59 까지 제출: 100% 점수인정
- 그 이후 제출 시 0%

■ 주의사항

- 모든 과제는 본인이 수행한 결과물만 제출
- 처음 Copy 적발 시 해당 과제 0 점 처리
- Copy 재적발 시 해당 과제 전체점수 0 점 처리
- Q&A: leichtjoon@ajou.ac.kr 민경준 조교