0. 프로젝트 설치 전, VUE 환경 개발 설정이 필요, 다음을 체크

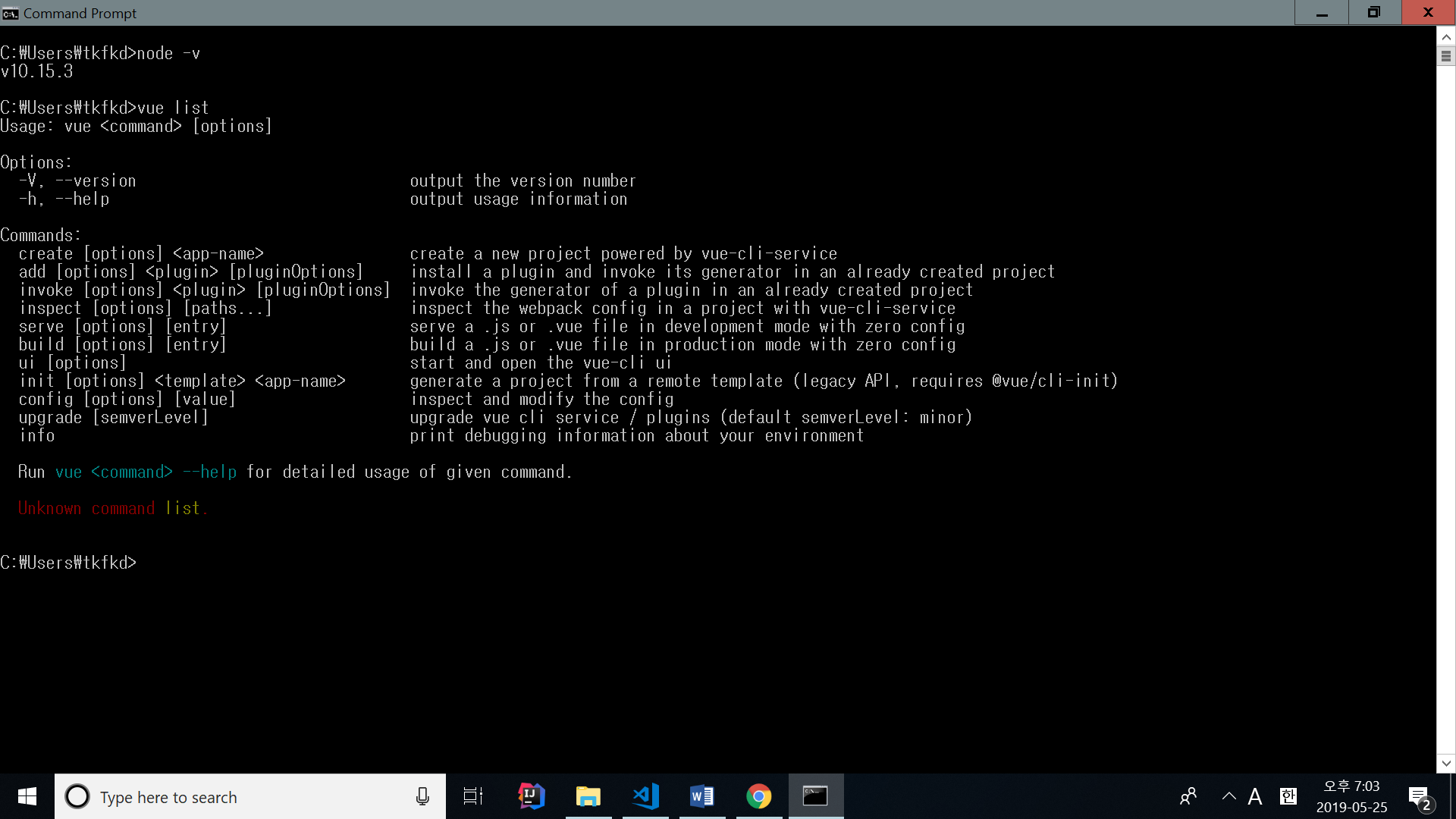
🞏 Node 설치 되어있는가? (인터넷 검색, 다운 그리고 node –v 을 cmd에 입력하여 확인)

🞏 Vue-cli 설치되어 있는가? (cmd에서 npm install -g @vue/cli)

1. Frontend 뷰 웹 팩 설치

- 템플릿

vue-cli 에서는 미리 세팅된 몇가지 템플릿을 제공한다. 제공되는 템플릿은 vuejs-templates 에 각각의 레퍼런스로 저장되어 있다. vue list 명령어를 통해서 제공되는 템플릿 리스트를 확인가능



- Webpack

Webpack은 웹 앱의 자원들을 자바스크립트 모듈로 변환하여, 하나로 묶어 웹 성능을 향상시켜 주고, 빌드 자동화 기능까지 포함한 강력한 자바스크립트 모듈 번들러

출처: <https://songta.tistory.com/54> [./Song]

- Vue init webpack projectName 과 vue create project 차이점

|  |
| --- |
| 버전 3.x  vue create helloworld |
| 버전 2.9  vue init webpack-simple 파일 경로 |

|  |  |  |
| --- | --- | --- |
|  | Vue CLI 3.0 | Vue CLI 2.9 |
| 언어 | ES6 기준 | ES5 기준 (ES6 선택 가능) |
| 웹팩 | 웹팩 설정 파일이 숨겨져 있음  (필요 설정 추가하는 형식) | 웹팩 설정 파일에서 바로 수정 가능 |
| GUI | GUI 툴 제공 (vue ui) | X |
| 구성 | 뷰 플러그인 형식 | 프로젝트 템플릿을 깃헙에서 다운로드 |
| 명령어 | vue create '프로젝트 이름' | vue init 템플릿 App 프로젝트 폴더 경로' |

|  |
| --- |
| cd backend  PS C:\Users\tkfkd\immsystem\backend> cd ..  PS C:\Users\tkfkd\immsystem> vue init webpack frontend  vue-cli · Generated "frontend".  # Installing project dependencies ...  # ========================  npm WARN deprecated browserslist@2.11.3: Browserslist 2 could fail on reading Browserslist >3.0 config used in other tools.  npm WARN deprecated bfj-node4@5.3.1: Switch to the `bfj` package for fixes and new features!  npm WARN deprecated flatten@1.0.2: I wrote this module a very long time ago; you should use something else.  npm WARN deprecated browserslist@1.7.7: Browserslist 2 could fail on reading Browserslist >3.0 config used in other tools.  npm WARN deprecated circular-json@0.3.3: CircularJSON is in maintenance only, flatted is its successor.  > core-js@2.6.8 postinstall C:\Users\tkfkd\immsystem\frontend\node\_modules\core-js  > node -e "try { require('./scripts/postinstall'); } catch (e) { /\* empty \*/ }"  Thank you for using core-js ( https://github.com/zloirock/core-js )!  Please consider supporting of core-js on Open Collective or Patreon:  > https://opencollective.com/core-js  > https://www.patreon.com/zloirock  Also, the author of core-js ( https://github.com/zloirock ) is looking for a good job -)  > uglifyjs-webpack-plugin@0.4.6 postinstall C:\Users\tkfkd\immsystem\frontend\node\_modules\webpack\node\_modules\uglifyjs-webpack-plugin  > node lib/post\_install.js  npm notice created a lockfile as package-lock.json. You should commit this file.  npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.9 (node\_modules\fsevents):  npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.9: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})  added 1203 packages from 675 contributors and audited 11959 packages in 50.895s  found 12 vulnerabilities (7 moderate, 5 high)  run `npm audit fix` to fix them, or `npm audit` for details  Running eslint --fix to comply with chosen preset rules...  # ========================  > frontend@1.0.0 lint C:\Users\tkfkd\immsystem\frontend  > eslint --ext .js,.vue src "--fix"  # Project initialization finished!  # ========================  To get started:  cd frontend  npm run dev  Documentation can be found at https://vuejs-templates.github.io/webpack  PS C:\Users\tkfkd\immsystem> cd frontend  PS C:\Users\tkfkd\immsystem\frontend> npm run dev  > frontend@1.0.0 dev C:\Users\tkfkd\immsystem\frontend  > webpack-dev-server --inline --progress --config build/webpack.dev.conf.js  12% building modules 23/29 modules 6 active ...\tkfkd\immsystem\frontend\src\App.vue{ parser: "babylon" } is deprecated; we now treat it as { parser: "babel" }.  95% emitting  DONE Compiled successfully in 7923ms 18:48:25  I Your application is running here: http://localhost:8081  Terminate batch job (Y/N)? y  PS C:\Users\tkfkd\immsystem\frontend> |

**- proxyTable 역할**

일반적으로 프록시란 다른 클래스의 인터페이스로 동작하는 클래스

Frontend 개발 중 <http://localhost:8080/api> 요청시에 <http://localhostL3000/api> 로 프록시 함

프론트엔드에서 백엔드로 http request가 보내기 위해 필요

Webpack 사용시 proxyTable 필요, 보통 proxy란 이름으로 다음 예시 코드와 같이 설정

|  |
| --- |
| devServer: {  host: '127.0.0.1',  port: 8010,  proxy: {  '/counsel/apply/api': { // /counsel/apply/api 는 예시다  target: '[https://개발서버\_URL/counsel/apply/api](https://xn--_url-9k3p618e2ua570a/counsel/apply/api)',  changeOrigin: true,  pathRewrite: {  '^/counsel/apply/api': ''  }  }  }  } |

<https://lovemewithoutall.github.io/it/webpack-config-for-debug/>

**- http-proxy-middleware 기술**

<https://github.com/chimurai/http-proxy-middleware>

<https://expressjs.com/ko/guide/writing-middleware.html>

**- vue-cli3 proxy 설정 (내가 못 찾았던 부분)**

vue-clli가 3으로 버전업이 되면서 프록시 설정 부분이 살짝 달라짐. 우선 설정 파일이 없어서 생성. root 폴더에 vue.config.js 라는 파일을 만들고, 다음의 코드를 작성

|  |
| --- |
| 'use strict'  module.exports = { devServer: { proxy: { // proxyTable 설정 '/api': { target: '[http://localhost:8080](http://localhost:8080/)', changeOrigin: **true**  }  } } } |

출처: https://cpdev.tistory.com/114 [하루하나]

**- axios 의미**

**Axios는 HTTP통신을 하는데 매우 인기있는 Javascript 라이브러리**

**Axios는 브라우저와 Node.js 플랫폼에서 모두 사용**

HTTP 클라이언트 라이브러리로써, 비동기 방식으로 HTTP 데이터 요청을 실행

내부적으로 AXIOS는 직접적으로 XMLHttpRequest 를 다루지 않고 “AJAX 호출”

<https://velog.io/@rohkorea86/%EB%B9%84%EB%8F%99%EA%B8%B0-%EB%B9%84%EB%8F%99%EA%B8%B0%EC%97%90-%EB%8C%80%ED%95%B4%EC%84%9C-%EC%B0%A8%EA%B7%BC%EC%B0%A8%EA%B7%BC-%EB%8B%A4%EB%A3%A8%EB%A0%A4%EA%B3%A0-%ED%95%A9%EB%8B%88%EB%8B%A4.>

- axios api vs fetch api

둘다 http request(get 요청) 관련

기능이 많냐 적으냐 차이??

- Error solution

|  |
| --- |
| [vue/require-v-for-key]  Elements in iteration expect to have 'v-bind:key' directives.eslint-plugin-vue |

By adding => :key="user.id"

|  |
| --- |
| <div v-for="user in users" :key="user.id" class="user"> |

-eslint 에러

Eslint는 코드 표준화를 위한 도구.

일단 useEslint: true -> false로 바꿔놓고 개발 중.

-계속 똑 같은 페이지… 왜???

- vuetify 추가

npm에서 vue add vuetify, 그리고 main.js에 추가

|  |
| --- |
| import vuetify from 'vuetify' // new 추가  Vue.config.productionTip = false  Vue.prototype.$http = axios  Vue.use(vuetify) // new 추가 |

-현재 에러: backend 페이지 계속 똑 같은 페이지를 리턴(이미 프론트엔드에서 삭제한 페이지임에도 불구하고)

(a) app.use(require('connect-history-api-fallback')()); //5.26 vue router 연동 위한 모듈 추가

해당 코드를 코멘트 처리하면 프론트엔드 연동 없이 백엔드 기본 페이지에서 데이터를 불러옴.

(b) 문제점은 npm run build를 통해 migration같은 걸 하면 해결!

-'connect-history-api-fallback' 모듈이란(front end의 히스토리 모드를 위한 backend history 모드 관련 모듈 설정)

Vue router와 express의 연동을 위해, express(nodejs)에 connect-history-api-fallback이라는 미들웨어 필요함

<좀 더 깊게 알아보기>

기본적으로 vue-router를 사용하게 되면 hash mode와 history mode 중 디폴트로 hash mode가 적용된다.

여기서 hash mode란 URL hash를 사용하여 전체 URL을 시뮬레이트 하는데, URL이 변경될 때 페이지가 다시 로드되지 않는다. 즉 hash 기호 다음의 경로는 페이지 내부의 이름으로 여겨지기 때문에, hash 이후의 경로가 바뀌더라도 페이지가 다시 로드되지 않는다.

Vue-router 사용 중 hash를 제거하기 위해서는 라우터의 history 모드를 설정할 수 있다. 그래서 다시 로드하지 않고 URL을 탐색할 수 있다.

History 모드를 사용할 때 주의사항: 서버는 404 에러를 보고하지 않는 문제가 발생할 수 있음. 그 이유는 발견되지 않는 경로가 이제 inde.html파일에서 제공하기 때문이다. 이를 위해서 Vue 앱에서 catch-all router를 구현하여 404 페이지를 표시하여야 한다. 다음 주소 값이 그 차이이다.

<http://localhost:3000/#/users/1>

<http://localhost:3000/api/users>

- Components 폴더와 Views 폴더의 공통점과 차이점

공통점은 둘 다 확장자가 vue로 끝나는 파일들을 포함하고 있다.

차이점은 router에서 보여주는 component 파일은 views 폴더에, 그 외는 components 폴더에 넣는다.

Views 폴더의 vue: 뷰에서 라우팅을 다룰 때(Vue Router), 라우트는 현재 뷰에서 다른 뷰로 바꿀 때 <router-view>를 사용한다. 이 라우트들은 보통은 src/router/router.js에 찾아볼 수 있다.

Components 폴더의 vue: 상대적으로 라우트로 사용되지 않을 때,

-vuetify 버튼 예제

<https://v1.vuetifyjs.com/ko/components/buttons>

-<v-icon> not working

npm install material-design-icons-iconfont -D

then add

// main.js

import 'material-design-icons-iconfont/dist/material-design-icons.css' // Ensure you are using css-loader

import Vue from 'vue'

import Vuetify from 'vuetify'

Vue.use(Vuetify, {

iconfont: 'md'

})

- <v-navigation-drawer v-model="drawer" app class="indigo">

여기서 app 없애면 drawer 창이 작동X, 왜지??

- Limit iteration of elements in v-for (해당 숫자만큼 불러오고 나머지는 more 버튼으로 다 불러오기)

- Vuetify Data table

참고 사이트: <https://fkkmemi.github.io/nemv/nemv-069-board-datatable/>

예시 코드

|  |
| --- |
| <template>  <v-container grid-list-md>  <v-layout row wrap>  <!-- <v-flex xs12 sm6 md4 v-for="article in articles" :key="article.\_id">  {{article}}  </v-flex> -->  <v-flex xs12>  <v-data-table  :headers="headers"  :items="articles"  :loading="loading">  <template slot="items" slot-scope="props">  <td :class="headers[0].class">{{ id2date(props.item.\_id)}}</td>  <td :class="headers[1].class">{{ props.item.title }}</td>  <td :class="headers[2].class">{{ props.item.\_user ? props.item.\_user.id : '손님' }}</td>  <td :class="headers[3].class">{{ props.item.cnt.view }}</td>  <td :class="headers[4].class">{{ props.item.cnt.like }}</td>  </template>  </v-data-table>  </v-flex>  </v-layout>  </v-container>  </template>  <script>  export default {  data () {  return {  // ..  articles: [],  headers: [  { text: '날짜', value: '\_id', sortable: true, class: 'hidden-sm-and-down' },  { text: '제목', value: 'title', sortable: true },  { text: '글쓴이', value: '\_user', sortable: false },  { text: '조회수', value: 'cnt.view', sortable: true },  { text: '추천', value: 'cnt.like', sortable: true }  ],  loading: false  // ..  }  },  // ..  methods: {  // ..  list () {  if (this.loading) return  this.loading = true  this.$axios.get(`article/${this.board.\_id}`)  .then(({ data }) => {  this.articles = data.ds  this.loading = false  })  .catch((e) => {  this.pop(e.message, 'error')  this.loading = false  })  },  id2date (val) {  if (!val) return '잘못된 시간 정보'  return new Date(parseInt(val.substring(0, 8), 16) \* 1000).toLocaleString()  }  }  }  </script> |

<다음 코드 관련 부연 설명>

|  |
| --- |
| <template v-slot:activator="{ on }">  <v-btn  v-on="on"  flat  icon  > |

- v-on?

이벤트 리스너, 이를 이용하여 DOM 이벤트를 듣고 트리거 될 때 자바스크립트를 실행할 수 있다.

v-on의 구조는 v-on:이벤트이름=”method 이름”

예시) v-on:click=”rollDice”

- 'slot activator' work in vuetify?

참고 사이트: <https://stackoverflow.com/questions/49079936/how-does-slot-activator-work-in-vuetify>

Vue.js 컴포넌트 재사용(slot), 즉 슬롯(slot)은 컴포넌트의 재사용성을 높여주는 기능

slot은 개념적으로 말해 부모의 컴포넌트의 템플릿에 있는 일부 노드를 자식 컴포넌트로 전달해주는 것.

- Parent component vs Child component

컴포넌트: 기본 HTML 엘리먼트를 확장해 재사용 가능 코드를 캡슐화 하는데 사용, Vue 컴파일러에 의해 동작이 추가된 사용자 지정 엘리먼트

-코드 재사용하기(slot을 이용한 components를 따로 만들기)

-props

부모에서 자식으로의 전달은 props

<child-component v-bind: props-name=”parent data name”></child-component>

자식에서 부모로의 이벤트 호출은 $emit

this.$emit(‘event-name’)

<child-component v-on:evt-name=”parent method name”></child-component>

|  |
| --- |
| 예시 (Child component) |
| <template>  <div>  <v-dialog  v-model="dialog"  width="500"  >  <template>  <v-btn  flat  icon  slot = "activator"  >  <v-icon>delete</v-icon>  </v-btn>  </template>  <v-card>  <v-card-title  class="headline grey lighten-2"  primary-title  >  <strong>{{passedData.item.id}}</strong>  Delete Confirmation  </v-card-title>  <v-card-text>  You won't be able to revert back.  Are you sure to delete?  </v-card-text>  <v-divider></v-divider>  <v-card-actions>  <v-spacer></v-spacer>  <v-btn  color="warning"  flat  v-on:click="sendToParent(passedData.item.id)"  >  Confirm  </v-btn>  <v-btn  color="primary"  flat  >  Back  </v-btn>  </v-card-actions>  </v-card>  </v-dialog>  </div>  </template>  <script>  export default {  props:['passedData'],  data() {  return {  selected: this.passedData  }  },  methods: {  sendToParent: function(id){  this.$emit('sendToParent', id)  // parent에 parameter 값도 보내기  },  }  }  </script> |

|  |
| --- |
| 예시 (Parent component) |
| <template>  <v-card class=products>  <v-card-title>  Products  <v-spacer></v-spacer>  <v-text-field  v-model="search"  append-icon="search"  label="Search"  single-line  hide-details  ></v-text-field>  </v-card-title>  <v-data-table  :headers="headers"  :items="products"  class="elevation-1"  >  <template v-slot:items="props">  <td>{{ props.item.id }}</td>  <td>{{ props.item.name }}</td>  <td>{{ props.item.category }}</td>  <td>{{ props.item.unit }}</td>  <td>  <UpdateBtn />  </td>  <td>  <DeleteBtn  v-bind:passed-data="props"  v-on:sendToParent="deleteItem"  />  </td>  </template>  <template v-slot:no-results>  <v-alert :value="true" color="error" icon="warning">  Your search for "{{ search }}" found no results.  </v-alert>  </template>  </v-data-table>  </v-card>  </template>  <script>  import DeleteBtn from '@/components/DeleteBtn'  import UpdateBtn from '@/components/UpdateBtn'  export default {  components: { DeleteBtn, UpdateBtn } ,  data() {  return {  headers: [  { text: 'ID', value: 'id' },  { text: 'Name', value: 'name' },  { text: 'Category', value: 'category' },  { text: 'Unit', value: 'unit' },  { text: 'Update', sortable: false },  { text: 'Delete', sortable: false }  ],  products: [  {  id: 1,  name: 'Brocoli',  category: 'Vege',  unit: 'ea'  },  {  id: 2,  name: 'Orange',  category: 'Fruit',  unit: 'ea'  },  {  id: 3,  name: 'Onion',  category: 'Vege',  unit: 'kg'  },  ]  }  },  methods:{  deleteItem: function(id){  console.log("tests")  console.log(id)  for(var i =0; i<this.products.length; i++){  console.log(this.products[i])  }  }  }  }  </script>  <style scoped>  </style> |

참고로 하위에서 상위 컴포넌트로 데이터는 전달할 수 없다. (뷰는 단방향 데이터 흐름)

그렇다면 같은 레벨의 컴포넌트간에 통신은?? Event Bus를 이용

- Vue에서 템플릿이란?

- dialog=true는 무슨 의미?

-dialog 에러 관련 해결방법

|  |  |
| --- | --- |
|  | 버튼 컴포넌트의 확인 다이얼로그 에러 |

해결책:

In childVueComponent.vue(dialog 관련 자식 컴포넌트)에서 dialog = false 추가하기

data() {

return {

dialog: false,

selected: this.passedData

}

- delete 버튼 라우팅에 대하여 찾을 수 없음 404 에러가 발생

Localhost:3000/product 요청

GET /api/products 응답 => Working

Delete button 요청

<http://localhost:3000/api/products/delete/0> 응답 => Not working

첫 번째 에러:

|  |
| --- |
| in products.js (backend)에서 delete 메서드로 정의 |
| router.delete('/delete/:id', function(req, res, next) |
| in product.vue (frontend)에서 get 메서드로 불러옴(그래서 404 에러) |
| this.$http.get(`/api/products/delete/${productId}`)  .then((response) => {  this.products = response.data  }) |

두 번째 에러:

|  |
| --- |
| Error [ERR\_HTTP\_HEADERS\_SENT]: Cannot set headers after they are sent to the client  at ServerResponse.setHeader (\_http\_outgoing.js:470:11)  at ServerResponse.header (C:\Users\tkfkd\immsystem\backend\node\_modules\express\lib\response.js:767:10)  at ServerResponse.location (C:\Users\tkfkd\immsystem\backend\node\_modules\express\lib\response.js:884:15)  at ServerResponse.redirect (C:\Users\tkfkd\immsystem\backend\node\_modules\express\lib\response.js:922:18)  at Query.<anonymous> (C:\Users\tkfkd\immsystem\backend\routes\products.js:47:24)  at Query.<anonymous> (C:\Users\tkfkd\immsystem\backend\node\_modules\mysql\lib\Connection.js:525:10)  at Query.\_callback (C:\Users\tkfkd\immsystem\backend\node\_modules\mysql\lib\Connection.js:491:16)  at Query.Sequence.end (C:\Users\tkfkd\immsystem\backend\node\_modules\mysql\lib\protocol\sequences\Sequence.js:83:24)  at Query.\_handleFinalResultPacket (C:\Users\tkfkd\immsystem\backend\node\_modules\mysql\lib\protocol\sequences\Query.js:139:8)  at Query.OkPacket (C:\Users\tkfkd\immsystem\backend\node\_modules\mysql\lib\protocol\sequences\Query.js:72:10) |

기본적으로 res.redirect처리가 잘못 되었을 때 발생

|  |
| --- |
| req.getConnection(function(error, conn) {  conn.query('Delete From Product Where Product\_ID = ?', [delProduct], function(err, rows, fields){  if(error) throw error;  res.send(rows);  res.redirect('/');  })  }) |

Res.send랑 res.redirect랑 같이 쓰니까 나는 에러였음. (비교 <https://luckyyowu.tistory.com/346>)

- Data 업데이트 후, 페이지 refresh를 server 사이드에서 해야하나 front 사이드에서 해야할까?

일단은 client 사이드에서 window.location.reload()추가함 -> working

- 라우트 컴포넌트 의미

|  |
| --- |
| this.$route.params.id |

- parse 에러

|  |
| --- |
| code: 'ER\_PARSE\_ERROR',  errno: 1064,  sqlMessage:  'You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near \'\'33\', \'aa\', \'22\'\' at line 1',  sqlState: '42000',  index: 0,  sql: 'Insert Into persons values \'33\', \'aa\', \'22\'' } |

서버 DB 쿼리문 에러: SYNTAX 에러였음

- setTimeout() 함수란?

일정시간 후에 특정 코드, 함수를 지연한 뒤 실행하고 싶을 때

setTimeout(function() { 코드 }, delay )

코드: 호출될 콜백 함수, delay: 지연 시간

예시:

setTimeout(function() {

console.log(“함수”) //3초뒤에 실행 됌

}, 3000)

활용방법:

접속 후 몇 초 후에 팝업창 띄우기 등등

Computed vs methods vs watch

Methods: 자신이 참고하지 않은 이벤트나 데이터 변경 발생할 때도 실행 O

Computed: 자신이 참고하지 않은 이벤트나 데이터 변경 발생시 실행 X

(어떤 값이 변경될 때 그 값과 아무 상관이 없는 함수가 모두 실행된다면, 결과적으로 비효율적임)

<https://medium.com/@hozacho/%EB%A7%A8%EB%95%85%EC%97%90-vuejs-%EA%B3%84%EC%82%B0%EB%90%9C-%EC%86%8D%EC%84%B1-vuejs-instance-computed-93cb6ad7dca9>

watch: 지정된 대상의 값이 변경될 때마다 정의한 함수가 실행

(computed는 참조하고 있는 값이 변경될 때마다 정의된 계산식에 따라 값을 출력하지만,

Watch는 값이 변경될 때 실행되는 함수를 우리가 지정할 수 있다.

즉 computed는 계산된 값을 출력하는 용도라면 watch는 어떤 조건이 되었을 때, 함수를 실행시키기 위한 트리거로 사용하면 OK)

- vue validation

예를 들어 required, maxLength, email 같은 validation 체크가 필요할 때

Npm install vuelidate –save

그리고 import를 해서 사용!

- v-model이랑 :items 무슨 차이?

|  |
| --- |
| <v-select  v-model = "supplier"  :items = "supplier"  label = "Supplier"  item-text = "Suplier\_Name"  item-value = "Supplier\_ID"  ></v-select> |

- v-autocomplete 기능 알아보기