```
[HOL] Full stack Project with Vue.js
-----------------------------------------------------
Task1. Back-end Project 준비하기
1. In Visual Studio Code > Terminal
2. $ md rest_contacts
3. $ cd rest_contacts

4. $ npm init
   1)package.json
      {
         "name": "rest_contacts",
         "version": "1.0.0",
         "description": "",
         "main": "index.js",
         "scripts": {
         "test": "echo \"Error: no test specified\" && exit 1"
         },
         "author": "",
         "license": "ISC"
      }

5. $ npm install express mongoose body-parser cors --save


--------------------------------------------------------------
Task2. Back-end Server 준비하기
1. index.js
   const express = require('express')
   const bodyparser = require('body-parser')
   const cors = require('cors')
   const app = express()    //express app 생성

   app.use(bodyparser.json())   //json 형식 parsing하기
   app.use(cors())   //cors 적용

   const dbconfig = require('./db.js')
   const mongoose = require('mongoose')
   //Database 연결 및 상태 경로
   mongoose.connect(dbconfig.url, { useNewUrlParser:true})
   .then(() => {
      console.log('MongoDB Connect Success.')
   }).catch(err => {
      console.log('MongoDB Connect Failure', err)
   })

   app.get('/', (req, res) => {
      console.log(req)
      res.json({'message': 'Welcome! Hello, World!!!'})
   })

   let port = process.env.PORT || 8000  //Server Port 설정

   //Client으로부터 요청 받기
   app.listen(port, () => {
      console.log('Port : ' + port + ' is ready...')
   })
```

2. db.js

```
module.exports = {
    url : 'mongodb://localhost:27017/contactlist'
}
```

3. MongoDB Server Start

```
rest_contacts> $ md .\data\db
rest_contacts> $ mongod --dbpach data/db
```

4. VSCode Terminal에서

```
$ node index.js
Port : 8000 is ready...
MongoDB Connect Success.
```

5. Browser에서

```
http://localhost:8000
{"message":"Welcome! Hello, World!!!"}
```

Task3. Back-end Coding

1. contact.model.js

```
const mongoose = require('mongoose')

var ContractSchema = mongoose.Schema({  //Schema 설정
    contactId: {type : Number, require:true},
    name : { type:String, require : true},
    email : { type : String, require : true},
    gender : String,
    phone : String,
    register_date : { type:Date, default : Date.now}
})
module.exports = mongoose.model('Contact', ContractSchema);
```

2. router.js

```
module.exports = (app) => {
    const contacts = require('./contact.controller.js');  //contact.controller.js 로딩
    app.get('/contacts', contacts.findAll);   //모든 연락처 검색
    app.get('/contacts/:contactId', contacts.findOne);  //특정 연락처 검색
    app.post('/contacts', contacts.create);   //새로운 연락처 추가
    app.put('/contacts/:contactId', contacts.update);  //특정 연락처 업데이트
    app.delete('/contacts/:contactId', contacts.delete);  //특정 연락처 삭제
}
```

3. index.js 수정

```
require('./router.js')(app);   <-- 여기에 추가
let port = process.env.PORT || 8000  //Server Port 설정
```

4. contact.controller.js

```
const Contact = require('./contact.model.js')
```

```
exports.create = (req, res) => {
    //새로운 연락처 만들기
    const contact = new Contact({
        contactId : req.body.contactId,
        name : req.body.name,
        email : req.body.email,
        gender : req.body.gender,
        phone: req.body.phone
    });

    //Database에 새로운 연락처 저장하기
    contact.save()
    .then(data => {
        res.send(data);
    })
    .catch(err => {
        res.status(500).send({
            message : err.message
        })
    });
}

exports.findAll = (req, res) => {
    Contact.find()
    .then(contacts => {
        res.send(contacts);
    })
    .catch(err => {
        res.status(500).send({
            message : err.message
        })
    });
}

exports.findOne = (req,res) => {
    Contact.findOne({
        contactId : req.params.contactId
    })
    .then(contact => {
        if(!contact){
            return res.status(404).send({
                message:req.params.contactId + '에 해당하는 연락처가 없습니다.'
            });
        }
        res.send(contact);
    })
    .catch(err => {
        return res.status(500).send({
            message : req.params.contactId + '로 검색 중 에러 발생'
        });
    })
}

exports.update = (req, res) => {
    Contact.findOneAndUpdate (
        {contactId:req.params.contactId},
        {contactId:req.body.contactId, name:req.body.name, email:req.body.email,
        gender:req.body.gender, phone:req.body.phone}, {new:true}
    )
    .then(contact => {
        if(!contact){
            return res.status(404).send({
                message : req.params.contactId + '에 해당하는 연락처(contact) 발견되지 않았습니다.'
            })
        }
        res.send(contact);
    })
    .catch(err => {
        return res.status(500).send({
            message : err.message
        });
    })
}

exports.delete = (req, res) => {
    Contact.findOneAndDelete({contactId:req.params.contactId})
    .then(contact => {
        if(!contact){
            return res.status(404).send({
              message:
                req.params.contactId +
                "에 해당하는 연락처(contact)가 없습니다."
            });
        }
        res.send({
            message : '정상적으로 ' + req.params.contactId + ' 연락처가 삭제되었습니다.'
        })
    })
    .catch(err => {
        return res.status(500).send({
            message : err.message
        });
    });
}
```

5. VSCode Terminal에서 Ctrl + C로 정지 후 다시 시작한다.

6. http://localhost:8000/contacts
    1)데이터가 없기 때문에 [] 만 나타난다.

7. POSTMAN으로 Test
    1)혹시 Chrome에 Postman이 없을 경우 아래 링크를 클릭해서 설치할 것

        https://chrome.google.com/webstore/detail/postman/fhbjgbiflinjbdggehcddcbncdddomop?hl=ko

8. POST : 연락처 추가
    POST : http://localhost:8000/contacts

```
{
    "contactId" : 1,
    "name" : "한지민",
    "email" : "aaa@aaa.com",
    "gender" : "여성",
        "phone" : "010-1234-5678"
    }
    -----------------------------------------
    {
        "_id": "5dfd07923276581cac4f81eb",
        "contactId": 1,
        "name": "한지민",
        "email": "aaa@aaa.com",
        "gender": "여성",
        "phone": "010-1234-5678",
        "register_date": "2020-06-02T13:14:18.258Z",
        "__v": 0
    }
    -----------------------------------
    -http://localhost:8000/contacts와도 동일한 결과가 나온다.


9. 2명 더 연락처를 추가한다.
    {
        "contactId" : 2,
        "name" : "박지민",
        "email" : "bbb@bbb.com",
        "gender" : "남성",
        "phone" : "010-7777-8888"
    }
    {
        "contactId" : 3,
        "name" : "김지민",
        "email" : "ccc@ccc.com",
        "gender" : "여성",
        "phone" : "010-5678-9876"
    }


10. GET : 데이터 검색
    1)전체 검색
        -http://localhost:8000/contacts
        [{"_id":"5ed650aa7c6cdf5e587c0cc1","contactId":1,"name":"한지민","email":"aaa@aaa.com","gender":"여성","phone":"010-1234-5678","register_date":"2020-06-02T13:14:18.258Z","__v":0},{"_id":"5ed650ff7c6cdf5e587c0cc2","contactId":2,"name":"박지민","email":"bbb@bbb.com","gender":"남성","phone":"010-7777-8888","register_date":"2020-06-02T13:15:43.801Z","__v":0},{"_id":"5ed6510d7c6cdf5e587c0cc3","contactId":3,"name":"김지민","email":"ccc@ccc.com","gender":"여성","phone":"010-5678-9876","register_date":"2020-06-02T13:15:57.355Z","__v":0}]

    2)특정 데이터 검색
        -http://localhost:8000/contacts/2
        {"_id":"5ed650ff7c6cdf5e587c0cc2","contactId":2,"name":"박지민","email":"bbb@bbb.com","gender":"남성","phone":"010-7777-8888","register_date":"2020-06-02T13:15:43.801Z","__v":0}


11. PUT : 데이터 업데이트
    -PUT : http://localhost:8000/contacts/3
    {
        "contactId" : 3,
        "name" : "김지민",
        "email" : "ccc@ccc.com",
        "gender" : "여성",
        "phone" : "010-9999-9999"  <--전화번호 수정
    }

    -3번의 전화번호가 변경된 것이 Browser에서 확인
    -http://localhost:8000/contacts/3
        {"_id":"5ed6510d7c6cdf5e587c0cc3","contactId":3,"name":"김지민","email":"ccc@ccc.com","gender":"여성","phone":"010-9999-9999","register_date":"2020-06-02T13:15:57.355Z","__v":0}


12. DELETE : 데이터 삭제
    -DELETE : http://localhost:8000/contacts/2

    {
        "message": "정상적으로 2 연락처가 삭제되었습니다."
    }

    ------------------------------------------
    Browser 에서 http://localhost:8000/contacts/2로 검색하면

    {"message":"2에 해당하는 연락처가 없습니다."}



-----------------------------------------------------------------
Task4. Front-end 준비하기
1. contactsclient Project 생성하기
    1)새로운 Command 창에서

        $ vue create contactsclient --default
        ...
        ...
        $ cd contactsclient
        $ npm run serve
            -Web Browser에서 확인

    2)여기서 새 Visual Studio Code 열기

        contactsclient> $ code .


2. Vuetify 설치
    1)Google Material Design에 기반을 둔 Vue.js에 특화된 UI 컴포넌트 라이브러리.
    2)Command 창에서
        $ vue add vuetify
    3)Default(recommended) 로 설치한다.
    4)설치 후 npm run serve로 확인


3. App.vue

        ...
        ...
        <v-content>
          <h2>연락처 리스트</h2>
          <data-table></data-table>
        </v-content>
    </template>

    <script>
    import DataTable from './components/DataTable';

    export default {
      name: 'App',

      components: {
        DataTable
      },

      data: () => ({
        //
      }),
    };
    </script>

    <style scoped>
      h2 {
        margin-top:20px;
        margin-left:10px;
        color:blue;
      }
    </style>
```

4. axios 설치

```
    $ npm install --save axios
```

Task5. Front-end Coding

1. DataTable.vue

```
    <template>
        <div id="app">
            <v-data-table :headers="headers" :items="items" class="elevation-1">
                <template v-slot:top>
                    <v-dialog v-model="dialog" persistent max-width="500px">
                        <v-card>
                            <v-card-title>
                                <span class="headline">{{ dialogTitle }}</span>
                            </v-card-title>
                            <v-card-text>
                                <v-container grid-list-md>
                                    <v-layout wrap>
                                        <v-flex xs12>
                                            <v-text-field label="번호" v-model="contactInfo.contactId" required>
                                            </v-text-field>
                                        </v-flex>
                                        <v-flex xs12>
                                            <v-text-field label="이름" v-model="contactInfo.name" required>
                                            </v-text-field>
                                        </v-flex>
                                        <v-flex xs12>
                                            <v-text-field label="이메일" v-model="contactInfo.email" required>
                                            </v-text-field>
                                        </v-flex>
                                        <v-flex xs12>
                                            <v-text-field label="성별" v-model="contactInfo.gender" required>
                                            </v-text-field>
                                        </v-flex>
                                        <v-flex xs12>
                                            <v-text-field label="전화" v-model="contactInfo.phone" required>
                                            </v-text-field>
                                        </v-flex>
                                        <v-flex xs12>
                                            <v-text-field label="등록일" v-model="contactInfo.register_date" required>
                                            </v-text-field>
                                        </v-flex>
                                    </v-layout>
                                </v-container>
                            </v-card-text>
                            <v-card-actions>
                                <v-spacer></v-spacer>
                                <v-btn color="blue darken-1" text @click="btnClick($event)">취소</v-btn>
                                <v-btn color="blue darken-1" text @click="btnClick($event)">확인</v-btn>
                            </v-card-actions>
                        </v-card>
                    </v-dialog>
                </template>
                <template v-slot:item.action="{ item }">
                    <v-btn color="primary" class="mr-2" v-on:click.native="addContact()">추가</v-btn>
                    <v-btn color="primary" class="mr-2" v-on:click.native="updateContact(item)">수정</v-btn>
                    <v-btn color="primary" class="mr-2" v-on:click.native="deleteContact(item)">삭제</v-btn>
                </template>
            </v-data-table>
        </div>
    </template>
    <script>
    import axios from 'axios'
    export default {
        data(){
            return {
                urlinfo:'http://localhost:8000/contacts',
                contactInfo : {
                    _id : null,
                    contactId:null,
                    name:null,
                    email:null,
                    gender:null,
                    phone:null,
                    register_date:null
                },
                dialog:false,
                dialogTitle :null,
                headers : [
                    { text:'번호', align:'right', value:'contactId'},
                    { text:'이름', align:'center', value:'name'},
                    { text:'이메일', align:'right', value:'email'},
                    { text:'성별', align:'center', value:'gender'},
                    { text:'전화', align:'center', value:'phone'},
                    { text:'등록일', align:'right', value:'register_date'},
                    { text:'작업', align:'center', value:'action'}
                ],
                items : []
            }
        },
        created(){
            axios.get(this.urlinfo) //Server에 요청하기
            .then((res) => {
                //console.log(res.data);
                this.items = res.data;
            })
            .catch((err) => {
                alert('Error 발생:', err)
            });
        },
        methods : {
            addContact(){
                this.dialog = true;
                this.dialogTitle = "추가";
                this.contactInfo.register_date = Date.now();
            },
            updateContact(data){
                this.dialog = true;
                this.dialogTitle = "수정";
                this.contactInfo.contactId = data.contactId;
                this.contactInfo.name = data.name;
                this.contactInfo.email = data.email;
                this.contactInfo.gender = data.gender;
                this.contactInfo.phone = data.phone;
                this.contactInfo.register_date = data.register_date;
            },
            deleteContact(data){
                this.dialog = true;
                this.dialogTitle = "삭제";
                this.contactInfo.contactId = data.contactId;
                //console.log('deleteContact : ' + data);
            },
            btnClick($event){
                this.dialog = false;
                if($event.target.innerHTML == "확인"){
                    if(this.dialogTitle =="추가"){
                    axios.post(this.urlinfo,{
                        contactId: this.contactInfo.contactId, name: this.contactInfo.name, email: this.contactInfo.email,
                        gender: this.contactInfo.gender,
                        phone:this.contactInfo.phone  //, register_date:this.contactInfo.register_date
                    })
                    .then(() => {
                        axios.get(this.urlinfo) //서버에 요청하기
                        .then((res) => {
                        //console.log(res.data); //성공시
                        this.items = res.data;
                        alert("연락처 추가 성공");
                        })
                        .catch((err) => {
                        alert('에러 발생: ' + err); //에러 발생
                        });

                    })
                    .catch((err) => {
                        alert('에러 발생: ' + err); //에러 발생
                    });
                }
                else if(this.dialogTitle=="수정"){
                    //alert("입력된 정보 : " + "이메일 : " + this.contactInfo.email + " 패스워드 : " + this.contactInfo.password);
                    axios.put(this.urlinfo + '/'+ this.contactInfo.contactId, {
                        contactId: this.contactInfo.contactId, name: this.contactInfo.name, email: this.contactInfo.email,
                        gender: this.contactInfo.gender,
                        phone:this.contactInfo.phone, register_date:this.contactInfo.register_date
                    })
                    .then(() => {
                        axios.get(this.urlinfo) //서버에 요청하기
                        .then((res) => {
                        //console.log(res.data); //성공시
                        this.items = res.data;
                        alert("업데이트 성공");
                        })
                        .catch((err) => {
                        alert('에러 발생: ' + err); //에러 발생
                        });
                    })
                    .catch((err) => {
                        alert('에러 발생: ' + err); //에러 발생
                    });
                }
                else {
                    axios.delete(this.urlinfo + '/'+ this.contactInfo.contactId, { data: { contactId: this.contactInfo.contactId } })
                    .then(() => {
                        //console.log("삭제 후" + result); //성공시
                        axios.get(this.urlinfo) //서버에 요청하기
                        .then((res) => {
                        this.items = res.data;
                        alert("삭제 성공");
                        })
                        .catch((err) => {
                        alert(' 삭제 후 데이터 가져오는 중 에러 발생: ' + err); //에러 발생
                        });
                    })
                    .catch((err) => { alert('에러 발생: ' + err); });
                    }
                }
                this.contactInfo.contactId = null;
                this.contactInfo.name = null;
```

```
            this.contactInfo.email = null;
            this.contactInfo.gender = null;
            this.contactInfo.phone = null;
            this.contactInfo.register_date = null;
        }
    }
}
</script>
<style scoped>
    div {
        margin:0 5px 0 5px;
    }
</style>
```

2. Test

```
$ npm run serve
```