1. $ md rest_contacts
2. VSCode Terminal로 해당 폴더로 이동
3. $ npm init --> package.json 파일 생성
4. $ npm install express mongoose body-parser cors --save

5. 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...')
})
```

6. db.js

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

7. MongoDB Start
   1)$ md data
   2)$ cd data
   3)$ md db
   4)$ cd ..
   4)mongod --dbpach data/db

8. VSCode Terminal에서

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

9. Browser에서
   http://localhost:8000

```
   {"message":"Welcome! Hello, World!!!"}

10. 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);

11. 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
   }

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

13. 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) => {
```

```
111         Contact.find()
112         .then(contacts => {
113             res.send(contacts);
114         })
115         .catch(err => {
116             res.status(500).send({
117                 message : err.message
118             })
119         });
120     }
121
122     exports.findOne = (req,res) => {
123         Contact.findOne({
124             contactId : req.params.contactId
125         })
126         .then(contact => {
127             if(!contact){
128                 return res.status(404).send({
129                     message:req.params.contactId + '에 해당하는 연락처가 없습니다.'
130                 });
131             }
132             res.send(contact);
133         })
134         .catch(err => {
135             return res.status(500).send({
136                 message : req.params.contactId + '로 검색 중 에러 발생'
137             });
138         })
139     }
140
141     exports.update = (req, res) => {
142         Contact.findOneAndUpdate (
143             {contactId:req.params.contactId},
144             {contactId:req.body.contactId, name:req.body.name, email:req.body.email,
145             gender:req.body.gender, phone:req.body.phone}, {new:true}
146         )
147         .then(contact => {
148             if(!contact){
149                 return res.status(404).send({
150                     message : req.params.contactId + '에 해당하는 연락처(contact) 발견되지 않았습니
                        다.'
151                 })
152             }
153             res.send(contact);
154         })
155         .catch(err => {
156             return res.status(500).send({
157                 message : err.message
158             });
159         })
160     }
161
162     exports.delete = (req, res) => {
163         Contact.findOneAndDelete({contactId:req.params.contactId})
164         .then(contact => {
```

```
165         if(!contact){
166            return res.status(404).send({
167              message:
168                req.params.contactId +
169                "에 해당하는 연락처(contact)가 없습니다."
170            });
171         }
172         res.send({
173            message : '정상적으로 ' + req.params.contactId + ' 연락처가 삭제되었습니다.'
174         })
175      })
176      .catch(err => {
177         return res.status(500).send({
178            message : err.message
179         });
180      });
181   }
182
183 14. VSCode Terminal에서 Ctrl + C로 정지 후 다시 시작한다.
184 15. http://localhost:8000/contacts
185    -데이터가 없기 때문에 [] 만 나타난다.
186
187 16. POSTMAN으로 Test
188 17. POST : 연락처 추가
189    POST : http://localhost:8000/contacts
190    {
191       "contactId" : 1,
192       "name" : "한지민",
193       "email" : "aaa@aaa.com",
194       "gender" : "여성",
195       "phone" : "010-1234-5678"
196    }
197    ----------------------------------------
198    {
199       "_id": "5dfd07923276581cac4f81eb",
200       "contactId": 1,
201       "name": "한지민",
202       "email": "aaa@aaa.com",
203       "gender": "여성",
204       "phone": "010-1234-5678",
205       "register_date": "2019-12-20T17:40:34.100Z",
206       "__v": 0
207    }
208    ----------------------------------
209    http://localhost:8000/contacts와도 동일한 결과가 나온다.
210
211 18. 두명 연락처를 추가한다.
212    {
213       "contactId" : 2,
214       "name" : "박지민",
215       "email" : "bbb@bbb.com",
216       "gender" : "남성",
217       "phone" : "010-7777-8888"
218    }
219    {
```

```
      "contactId" : 3,
      "name" : "김지민",
      "email" : "ccc@ccc.com",
      "gender" : "여성",
      "phone" : "010-5678-9876"
   }

19. GET : 데이터 검색
   1)전체 검색
   http://localhost:8000/contacts
   [{"_id":"5dfd07923276581cac4f81eb","contactId":1,"name":"한지
   민","email":"aaa@aaa.com","gender":"여
   성","phone":"010-1234-5678","register_date":"2019-12-20T17:40:34.100Z","__v":0},{
   "_id":"5dfd084d3276581cac4f81ec","contactId":2,"name":"박지
   민","email":"bbb@bbb.com","gender":"남
   성","phone":"010-7777-8888","register_date":"2019-12-20T17:43:41.363Z","__v":0},{
   "_id":"5dfd08643276581cac4f81ed","contactId":3,"name":"김지
   민","email":"ccc@ccc.com","gender":"여
   성","phone":"010-5678-9876","register_date":"2019-12-20T17:44:04.648Z","__v":0}]

   2)특정 데이터 검색
   http://localhost:8000/contacts/2
   {"_id":"5dfd084d3276581cac4f81ec","contactId":2,"name":"박지
   민","email":"bbb@bbb.com","gender":"남
   성","phone":"010-7777-8888","register_date":"2019-12-20T17:43:41.363Z","__v":0}

20. PUT : 데이터 업데이트
   PUT : http://localhost:8000/contacts/3
   {
      "contactId" : 3,
      "name" : "김지민",
      "email" : "ccc@ccc.com",
      "gender" : "여성",
      "phone" : "010-9999-9999"
   }
   ----------------------------------------
   3번의 전화번호가 변경된 것이 Browser에서 확인

21. DELETE : 데이터 삭제
   DELETE : http://localhost:8000/contacts/2
   ------------------------------------------------
   {
      "message": "정상적으로 2 연락처가 삭제되었습니다."
   }
   ----------------------------------------
   Browser 에서 http://localhost:8000/contacts/2
   {"message":"2에 해당하는 연락처가 없습니다."}

-----------------------------------------------------------------
1. $ vue create contactsclient --default
   -npm run serve
2. Vuetify 설치
   1)Google Material Design에 기반을 둔 Vue.js에 특화된 UI 컴포넌트 라이브러리.
   2)Command 창에서
      $ vue add vuetify
```

```
265    3)Default 로 설치한다.
266    4)설치 후 npm run serve로 확인
267
268 3. App.vue
269    ...
270    <v-content>
271      <h2>연락처 리스트</h2>
272      <data-table></data-table>
273    </v-content>
274    ...
275    <script>
276    import DataTable from './components/DataTable';
277
278    export default {
279      name: 'App',
280
281      components: {
282        DataTable
283      },
284
285      data: () => ({
286        //
287      }),
288    };
289    </script>
290    <style scoped>
291      h2 {
292        margin-top:20px;
293        margin-left:10px;
294        color:blue;
295      }
296    </style>
297
298 4. axios 설치
299    $ npm install --save axios
300
301 5. DataTable.vue
302    <template>
303       <div id="app">
304          <v-data-table :headers="headers" :items="items" class="elevation-1">
305             <template v-slot:top>
306                <v-dialog v-model="dialog" persistent max-width="500px">
307                   <v-card>
308                      <v-card-title>
309                         <span class="headline">{{ dialogTitle }}</span>
310                      </v-card-title>
311                      <v-card-text>
312                         <v-container grid-list-md>
313                            <v-layout wrap>
314                               <v-flex xs12>
315                                  <v-text-field label="번호" v-model="contactInfo.contactId"
                                    required>
316                                  </v-text-field>
317                               </v-flex>
318                               <v-flex xs12>
```

```
319                                 <v-text-field label="이름" v-model="contactInfo.name"
                                    required>
320                                 </v-text-field>
321                             </v-flex>
322                             <v-flex xs12>
323                                 <v-text-field label="이메일" v-model="contactInfo.email"
                                    required>
324                                 </v-text-field>
325                             </v-flex>
326                             <v-flex xs12>
327                                 <v-text-field label="성별" v-model="contactInfo.gender"
                                    required>
328                                 </v-text-field>
329                             </v-flex>
330                             <v-flex xs12>
331                                 <v-text-field label="전화" v-model="contactInfo.phone"
                                    required>
332                                 </v-text-field>
333                             </v-flex>
334                             <v-flex xs12>
335                                 <v-text-field label="등록일"
                                    v-model="contactInfo.register_date" required>
336                                 </v-text-field>
337                             </v-flex>
338                         </v-layout>
339                     </v-container>
340                 </v-card-text>
341                 <v-card-actions>
342                     <v-spacer></v-spacer>
343                     <v-btn color="blue darken-1" text @click="btnClick($event)">취소
                        </v-btn>
344                     <v-btn color="blue darken-1" text @click="btnClick($event)">확인
                        </v-btn>
345                 </v-card-actions>
346             </v-card>
347         </v-dialog>
348     </template>
349     <template v-slot:item.action="{ item }">
350         <v-btn color="primary" class="mr-2" v-on:click.native="addContact()">추가
            </v-btn>
351         <v-btn color="primary" class="mr-2"
            v-on:click.native="updateContact(item)">수정</v-btn>
352         <v-btn color="primary" class="mr-2"
            v-on:click.native="deleteContact(item)">삭제</v-btn>
353     </template>
354 </v-data-table>
355 </div>
356 </template>
357 <script>
358 import axios from 'axios'
359 export default {
360     data(){
361         return {
362             urlinfo:'http://localhost:8000/contacts',
363             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){
```

```
419             this.dialog = false;
420             if($event.target.innerHTML == "확인"){
421                 if(this.dialogTitle =="추가"){
422                 axios.post(this.urlinfo,{
423                     contactId: this.contactInfo.contactId, name: this.contactInfo.name,
                    email:  this.contactInfo.email,
424                     gender: this.contactInfo.gender,
425                     phone:this.contactInfo.phone  //,
                    register_date:this.contactInfo.register_date
426                 })
427                 .then(() => {
428                     axios.get(this.urlinfo) //서버에 요청하기
429                     .then((res) => {
430                     //console.log(res.data); //성공시
431                     this.items = res.data;
432                     alert("연락처 추가 성공");
433                     })
434                     .catch((err) => {
435                     alert('에러 발생: ' + err); //에러 발생
436                     });
437 
438                 })
439                 .catch((err) => {
440                     alert('에러 발생: ' + err); //에러 발생
441                 });
442             }
443             else if(this.dialogTitle=="수정"){
444                 //alert("입력된 정보 : " + "이메일 : " + this.contactInfo.email + " 패스워드 : " +
                this.contactInfo.password);
445                 axios.put(this.urlinfo + '/'+ this.contactInfo.contactId, {
446                     contactId: this.contactInfo.contactId, name: this.contactInfo.name,
                    email:  this.contactInfo.email,
447                     gender: this.contactInfo.gender,
448                     phone:this.contactInfo.phone,
                    register_date:this.contactInfo.register_date
449                 })
450                 .then(() => {
451                     axios.get(this.urlinfo) //서버에 요청하기
452                     .then((res) => {
453                     //console.log(res.data); //성공시
454                     this.items = res.data;
455                     alert("업데이트 성공");
456                     })
457                     .catch((err) => {
458                     alert('에러 발생: ' + err); //에러 발생
459                     });
460                 })
461                 .catch((err) => {
462                     alert('에러 발생: ' + err); //에러 발생
463                 });
464             }
465             else {
466                 axios.delete(this.urlinfo + '/'+ this.contactInfo.contactId, { data: {
                contactId: this.contactInfo.contactId } })
467                 .then(() => {
```

```
468                     //console.log("삭제 후" + result); //성공시
469                     axios.get(this.urlinfo) //서버에 요청하기
470                     .then((res) => {
471                     this.items = res.data;
472                     alert("삭제 성공");
473                     })
474                     .catch((err) => {
475                     alert(' 삭제 후 데이터 가져오는 중 에러 발생: ' + err); //에러 발생
476                     });
477                 })
478                 .catch((err) => { alert('에러 발생: ' + err); });
479                 }
480             }
481             this.contactInfo.contactId = null;
482             this.contactInfo.name = null;
483             this.contactInfo.email = null;
484             this.contactInfo.gender = null;
485             this.contactInfo.phone = null;
486             this.contactInfo.register_date = null;
487         }
488     }
489 }
490 </script>
491 <style scoped>
492     div {
493         margin:0 5px 0 5px;
494     }
495 </style>
```