# MERN STACK –CRUD OPERATION STEP BY STEP

**Requirement**

Node, Mongodb, React js,Any IDE for mongodb,VScode ,Any command prompt (i suggest Git Bash) etc.

Here is the Step by Step write code.

**Step 1**

Create New Folder, ReactCRUD, if you use git then right-click folder then GIT Bash runs command npm init. If you use cmd you navigate to your folder after running this command.

```
npm init
```

**Step 2**

One by one put answers by cmd: your project name, keyword, entry point etc. After creating your package.json file code like this.

```
{
  "name": "reactcrud",
  "version": "1.0.0",
  "description": "reactCrud",
  "main": "server.js",
  "scripts": {
    "test": "react"
  },
  "keywords": [
          "React"
       ],
       "author": "puneet kankar",
       "license": "MIT"

     }
```

**Step 3**

Add manual dependencies or one by one in our package.json express,mongoose,morgan,body-parser etc.

```
{
  "name": "reactcrud",
  "version": "1.0.0",
  "description": "reactCrud",
  "main": "server.js",
  "scripts": {
    "test": "react"
  },
  "keywords": [
          "React"
        ],
        "author": "puneet kankar",
        "license": "MIT",
         "dependencies": {
          "body-parser": "^1.17.2",
          "express": "^4.15.3",
          "mongoose": "^4.10.2",
          "morgan": "^1.8.2"

        }
      }
```

**Step 4**

Run npm install command . If you went one by one add dependencies run these commands one by one.

```
// when you add manual dependencies in package.json
npm init

//or

// when you not add dependencies in package.json
npm install body-parser
npm install express
npm install mongoose
     npm install morgan
```

**Step 5**

We Create config.js file on root directory for Mongodb database connection.We write code like this. In this code we require mongoose driver for connection to database.

```
var mongo = require("mongoose");
var db =
mongo.connect("mongodb://192.168.1.71:27017/reactcrud", function(
err, response){
   if(err){ console.log('Failed to connect to ' + db); }
   else{ console.log('Connected to ' + db, ' + ', response); }
});


module.exports =db;

    // reactcrud is database name
    // 192.168.1.71:27017 is your mongo server name
```

**Step 6**

We Create server.js file on root directory for writing nodejs apis for our Create ,Insert,Delete,Update. Give port number for application and run on server. In this file we write the code for all required dependancies and create schema of our database collection document and write code for api's for performing operation.

```
var express = require("express");
var path = require("path");
var mongo = require("mongoose");
var bodyParser = require('body-parser');
var morgan = require("morgan");
var db = require("./config.js");

var app = express();
var port = process.env.port || 7777;
    var srcpath  =path.join(__dirname,'/public') ;
    app.use(express.static('public'));
    app.use(bodyParser.json({limit:'5mb'}));
    app.use(bodyParser.urlencoded({extended:true, limit:'5mb'}))
;


    var mongoose = require('mongoose');
    var Schema = mongoose.Schema;
    var studentSchema = new Schema({
        name: { type: String   },
        address: { type: String   },
        email: { type: String },
        contact: { type: String },
    },{ versionKey: false });


    var model = mongoose.model('student', studentSchema, 'studen
t');
```

```
    //api for get data from database
    app.get("/api/getdata",function(req,res){
     model.find({},function(err,data){
                if(err){
                    res.send(err);
                }
                else{
                    res.send(data);
                    }
            });
    })


    //api for Delete data from database
    app.post("/api/Removedata",function(req,res){
     model.remove({ _id: req.body.id }, function(err) {
                if(err){
                    res.send(err);
                }
                else{
                       res.send({data:"Record has been Deleted..!!"});
                   }
            });
    })


    //api for Update data from database
    app.post("/api/Updatedata",function(req,res){
     model.findByIdAndUpdate(req.body.id, { name:  req.body.name, address: req.body.address, contact: req.body.contact,email:req.body.email },
    function(err) {
     if (err) {
     res.send(err);
     return;
     }
     res.send({data:"Record has been Updated..!!"});
     });
    })


    //api for Insert data from database
    app.post("/api/savedata",function(req,res){

        var mod = new model(req.body);
```

```
            mod.save(function(err,data){
                if(err){
                    res.send(err);
                }
                else{
                     res.send({data:"Record has been Inserted..!!"});
                }
            });
    })

    // call by default index.html page
    app.get("*",function(req,res){
        res.sendFile(srcpath +'/index.html');
    })

    //server stat on given port
    app.listen(port,function(){
        console.log("server start on port"+ port);
    })
```

**Step 7**

We create new public folder. Inside this folder create new HTML file with name index.html. We include react js ,react-dom ,babel,bootstrap CSS,jquery link as we require in our application.

```
<!DOCTYPE HTML>
<html>
    <head>
      <meta charset="utf-8">
       <title>React CRUD</title>
       <script src="https://fb.me/react-15.0.1.js"></script>
       <script src="https://fb.me/react-dom-15.0.1.js"></script>
       <script src="https://cdnjs.cloudflare.com/ajax/libs/babel-core/5.8.23/browser.min.js"></script>
       <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
           <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>
         </head>
          <body>
          <div  id='root'></div>
          <script type="text/babel"  src="./ReactCrud.jsx"  >
          </script>
        </body>
    </html>
```

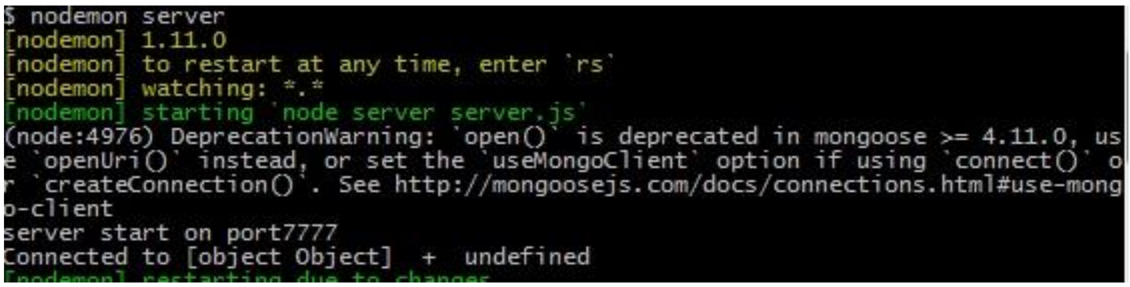
**Step 8**

We create ReactCrud.jsx file for create component with the name StudentAll. Here we write code for insert , select, update, delete operation API calling code.

```

var StudentAll = React.createClass({

  getInitialState: function () {
    return { name: '' ,address: '',email:'',contact:'',id:'',Buttontxt:'Save', data1: []};
  },
   handleChange: function(e) {
        this.setState({[e.target.name]: e.target.value});
    },

       componentDidMount() {

         $.ajax({
            url: "api/getdata",
            type: "GET",
            dataType: 'json',
            ContentType: 'application/json',
            success: function(data) {
              this.setState({data1: data});

            }.bind(this),
            error: function(jqXHR) {
              console.log(jqXHR);

            }.bind(this)
         });
       },

     DeleteData(id){
       var studentDelete = {
             'id': id
                };
         $.ajax({
           url: "/api/Removedata/",
           dataType: 'json',
           type: 'POST',
           data: studentDelete,
           success: function(data) {
             alert(data.data);
              this.componentDidMount();

           }.bind(this),
           error: function(xhr, status, err) {
              alert(err);


           }.bind(this),
           });
         },

         EditData(item){
        this.setState({name: item.name,address:item.address,contact:item.contact,email:item.email,id:item._id,Buttontxt:'Update'});
          },

        handleClick: function() {

        var Url="";
        if(this.state.Buttontxt=="Save"){
           Url="/api/savedata";
            }
           else{
           Url="/api/Updatedata";
           }
           var studentdata = {
             'name': this.state.name,
             'address':this.state.address,
             'email':this.state.email,
             'contact':this.state.contact,
             'id':this.state.id,

         }
         $.ajax({
           url: Url,
           dataType: 'json',
           type: 'POST',
           data: studentdata,
           success: function(data) {
               alert(data.data);
               this.setState(this.getInitialState());
               this.componentDidMount();

           }.bind(this),
           error: function(xhr, status, err) {
              alert(err);
           }.bind(this)
         });
       },

       render: function() {
         return (
           <div  className="container"  style={{marginTop:'50px'}}>
            <p className="text-center" style={{fontSize:'25px'}}><b> CRUD Opration Using React,Nodejs,Express,MongoDB</b></p>
       <form>
         <div className="col-sm-12 col-md-12"  style={{marginLeft:'400px'}}>
       <table className="table-bordered">
          <tbody>
         <tr>
           <td><b>Name</b></td>
           <td>
              <input className="form-control" type="text" value={this.state.name}    name="name" onChange={ this.handleChange } />
               <input type="hidden" value={this.state.id}    name="id"  />
           </td>
         </tr>

         <tr>
           <td><b>Address</b></td>
           <td>
           <input type="text" className="form-control" value={this.state.address}  name="address" onChange={ this.handleChange } />
           </td>
         </tr>

         <tr>
           <td><b>Email</b></td>
           <td>
             <input type="text"  className="form-control" value={this.state.email}  name="email" onChange={ this.handleChange } />
         </td>
       </tr>


       <tr>
         <td><b>Contact</b></td>
         <td>
           <input type="text"  className="form-control" value={this.state.contact}  name="contact" onChange={ this.handleChange } />
         </td>
       </tr>

       <tr>
         <td></td>
         <td>
           <input className="btn btn-primary" type="button" value={this.state.Buttontxt} onClick={this.handleClick} />
         </td>
       </tr>

    </tbody>
       </table>
   </div>


   <div className="col-sm-12 col-md-12 "  style={{marginTop:'50px',marginLeft:'300px'}} >

    <table className="table-bordered"><tbody>
      <tr><th><b>S.No</b></th><th><b>NAME</b></th><th><b>ADDRESS</b></th><th><b>EMAIL</b></th><th><b>CONTACT</b></th><th><b>Edit</b></th><th><b>Delete</b></th></tr>
       {this.state.data1.map((item, index) => (
           <tr key={index}>
              <td>{index+1}</td>
             <td>{item.name}</td>
             <td>{item.address}</td>
             <td>{item.email}</td>
             <td>{item.contact}</td>
              <td>

              <button type="button" className="btn btn-success" onClick={(e) => {this.EditData(item)}}>Edit</button>
             </td>
             <td>
                <button type="button" className="btn btn-info" onClick={(e) => {this.DeleteData(item._id)}}>Delete</button>
             </td>
           </tr>
       ))}
       </tbody>
       </table>
        </div>
   </form>
         </div>
       );
     }
   });

   ReactDOM.render(<StudentAll  />, document.getElementById('root'))
```

**Step 9**

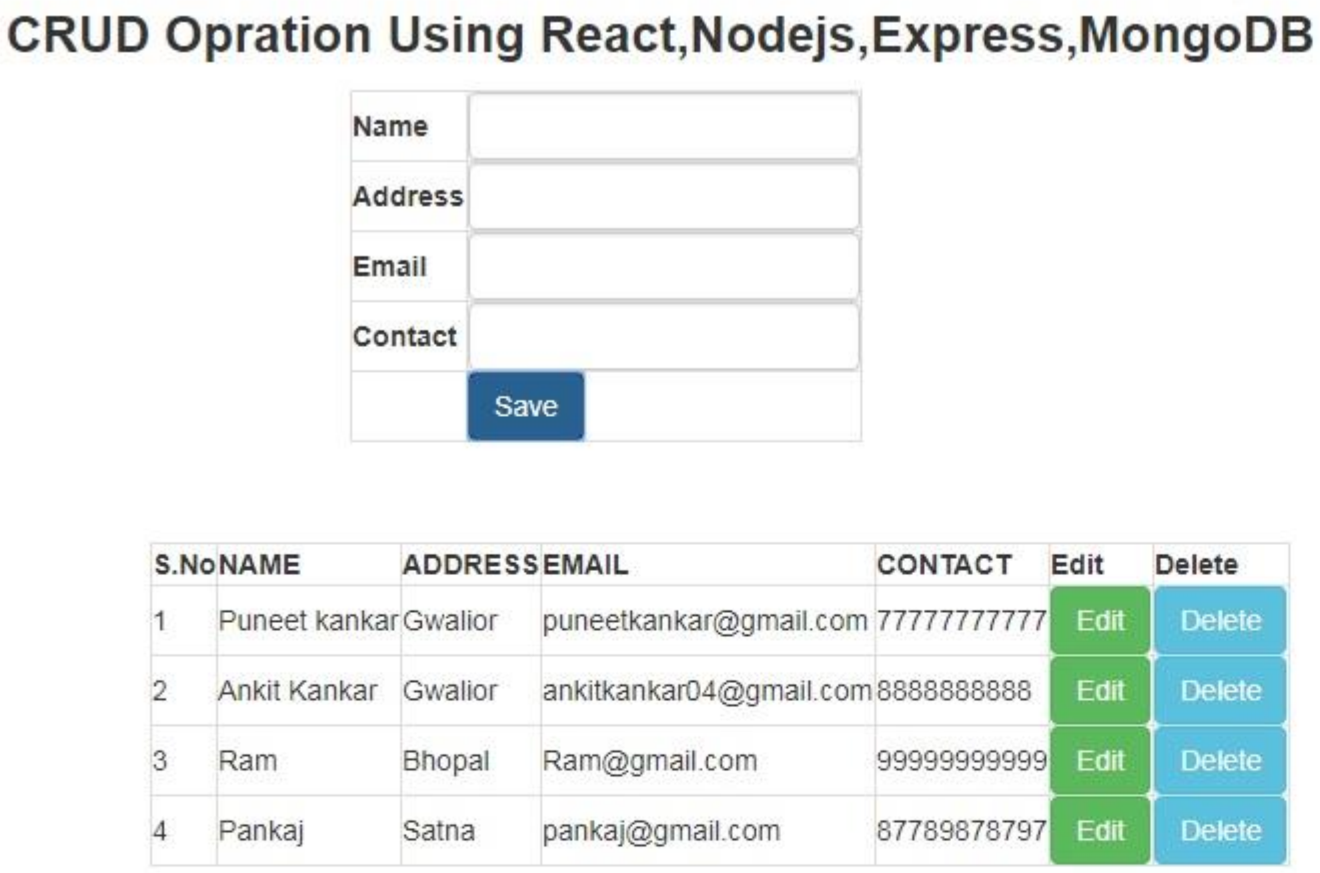
We are almost finished with all the required tasks. We are going to command prompt run node server.js or if you are installing morgan then use nodemon server.



```
$ nodemon server
[nodemon] 1.11.0
[nodemon] to restart at any time, enter `rs`
[nodemon] watching: *.*
[nodemon] starting `node server server.js`
(node:4976) DeprecationWarning: `open()` is deprecated in mongoose >= 4.11.0, use `openUri()` instead, or set the `useMongoClient` option if using `connect()` or `createConnection()`. See http://mongoosejs.com/docs/connections.html#use-mongo-client
server start on port7777
Connected to [object Object]  +  undefined
[nodemon] restarting due to changes
```

**Step 10**

We will go and browse our application on given port number 7777 in our server.js file so we run this URL http://localhost:7777/  our output display looks like below image.



## CRUD Opration Using React,Nodejs,Express,MongoDB

| | |
|---|---|
| Name | |
| Address | |
| Email | |
| Contact | |
| | Save |

| S.No | NAME | ADDRESS | EMAIL | CONTACT | Edit | Delete |
|---|---|---|---|---|---|---|
| 1 | Puneet kankar | Gwalior | puneetkankar@gmail.com | 77777777777 | Edit | Delete |
| 2 | Ankit Kankar | Gwalior | ankitkankar04@gmail.com | 8888888888 | Edit | Delete |
| 3 | Ram | Bhopal | Ram@gmail.com | 99999999999 | Edit | Delete |
| 4 | Pankaj | Satna | pankaj@gmail.com | 87789878797 | Edit | Delete |