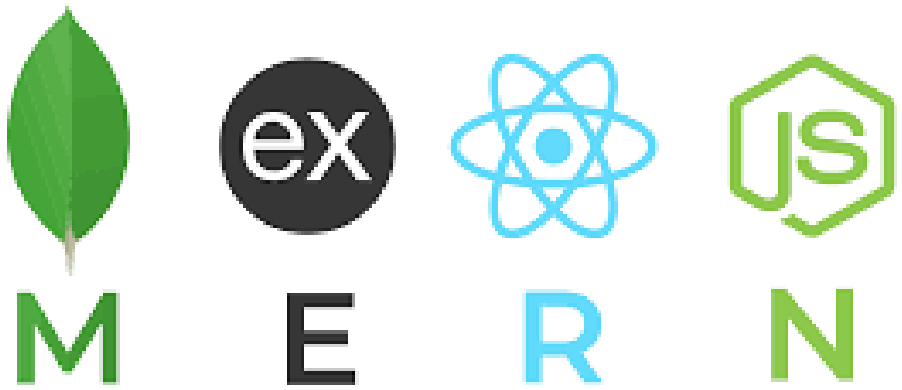
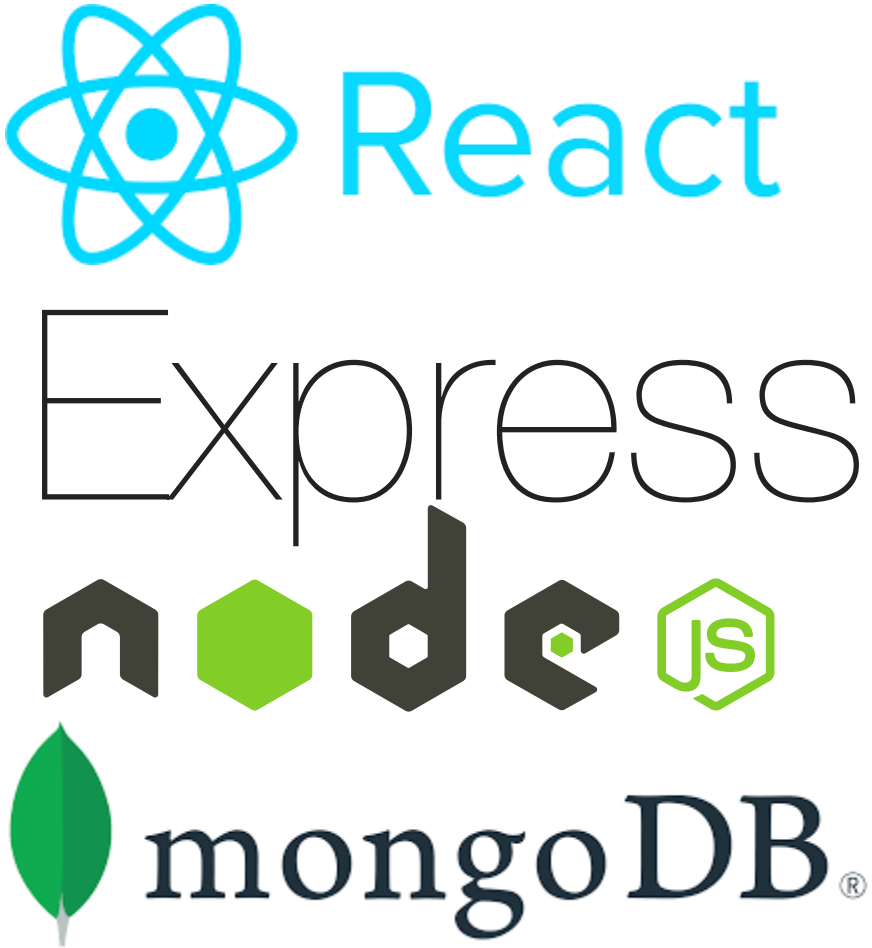
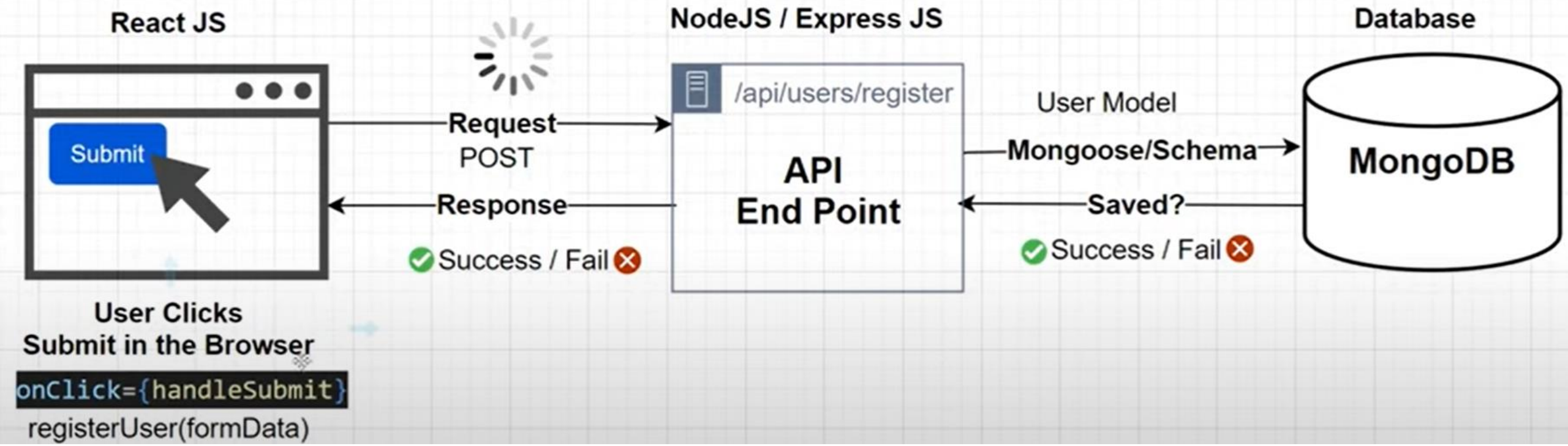




Reference :

https://mongoosejs.com/docs/schematypes.html
https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/mongoose
https://mongoosejs.com/docs/2.7.x/docs/finding-documents.html
https://www.geeksforgeeks.org/how-to-connect-mongodb-with-reactjs/
https://www.makeuseof.com/react-form-data-mongodb-database-store/

Save React JS Form Data into MongoDB Database



Mongo Database MongoDB

Compare

mongoose

Object-Document Modeler ODM

rigid design

mongoDB®

native driver

flexible development

Reference:

https://www.mongodb.com/developer/languages/javascript/mongoose-versus-nodejs-driver/

https://mongoosejs.com/

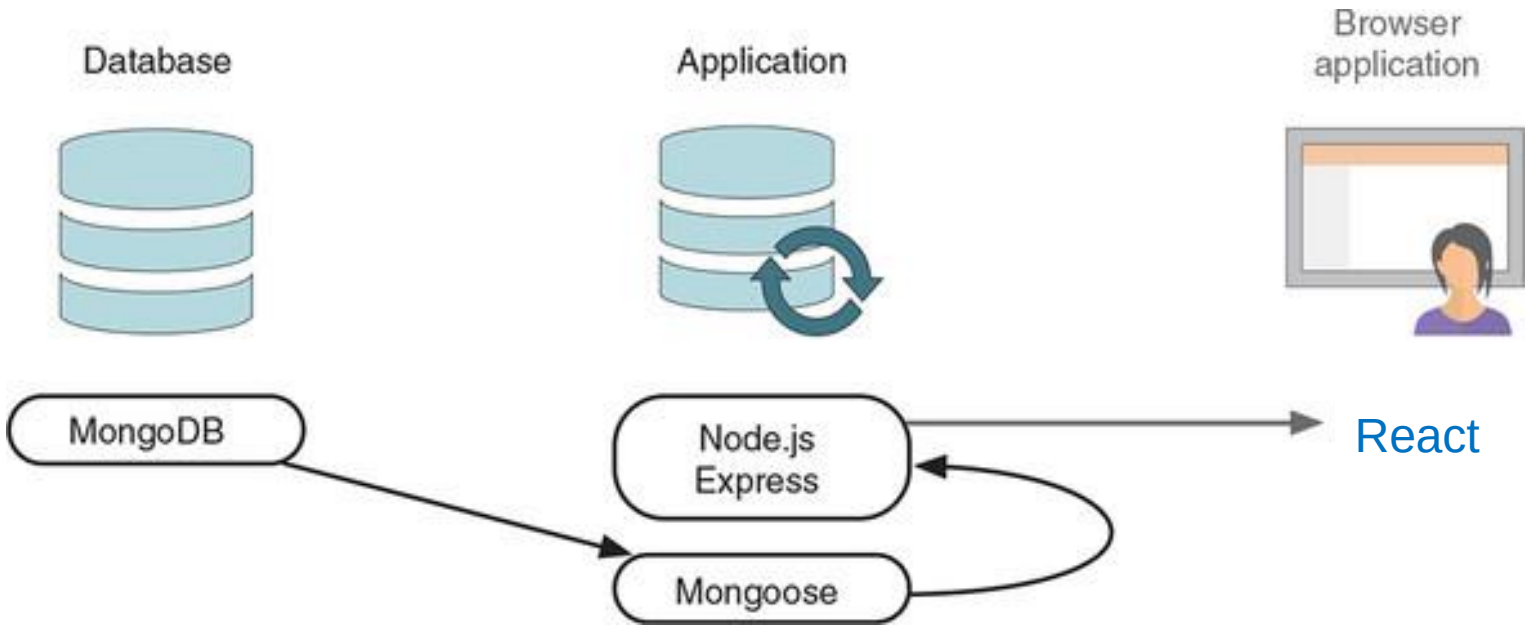
Results and Conclusion

Drumroll please, below are the results:

READS	Native	Mongoose
Throughput	1200 #/sec	583 #/sec
Avg Request	0.83 ms	1.71 ms
WRITES	**Native**	**Mongoose**
Throughput	1128 #/sec	384 #/sec
Avg Request	0.89 ms	2.60 ms

Overall, the native driver is around 2x faster than Mongoose.

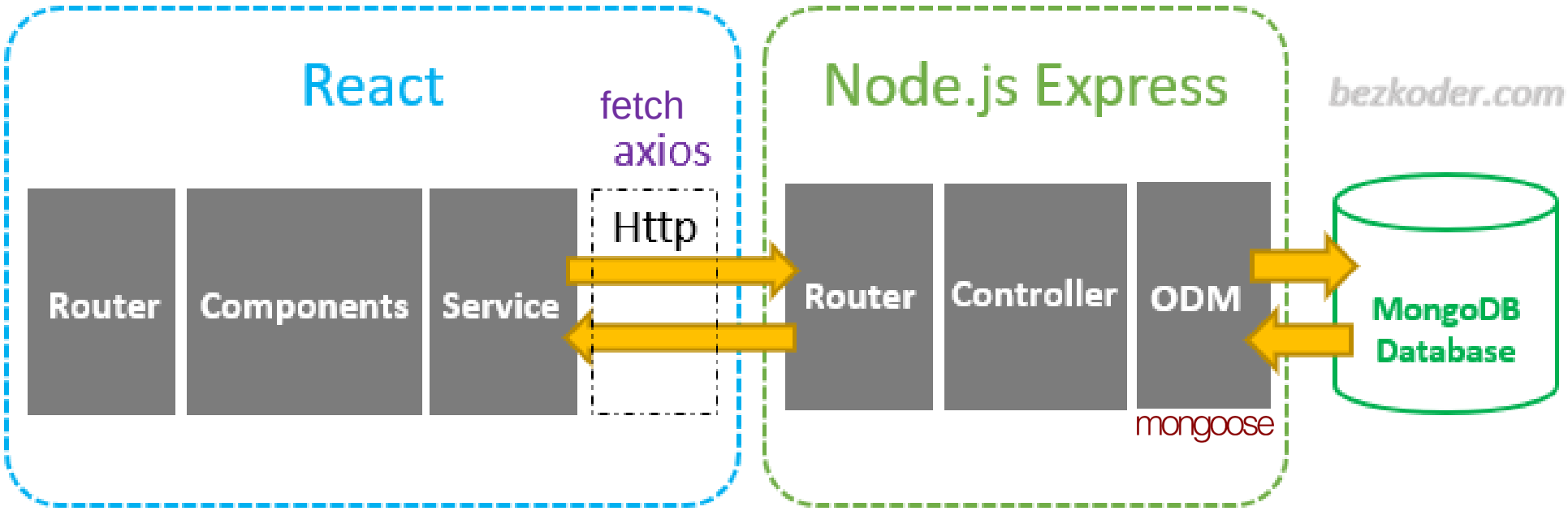
https://blog.jscrambler.com/mongodb-native-driver-vs-mongoose-performance-benchmarks



Mongoose is an Object Data Modeling (ODM) library for MongoDB and Node.

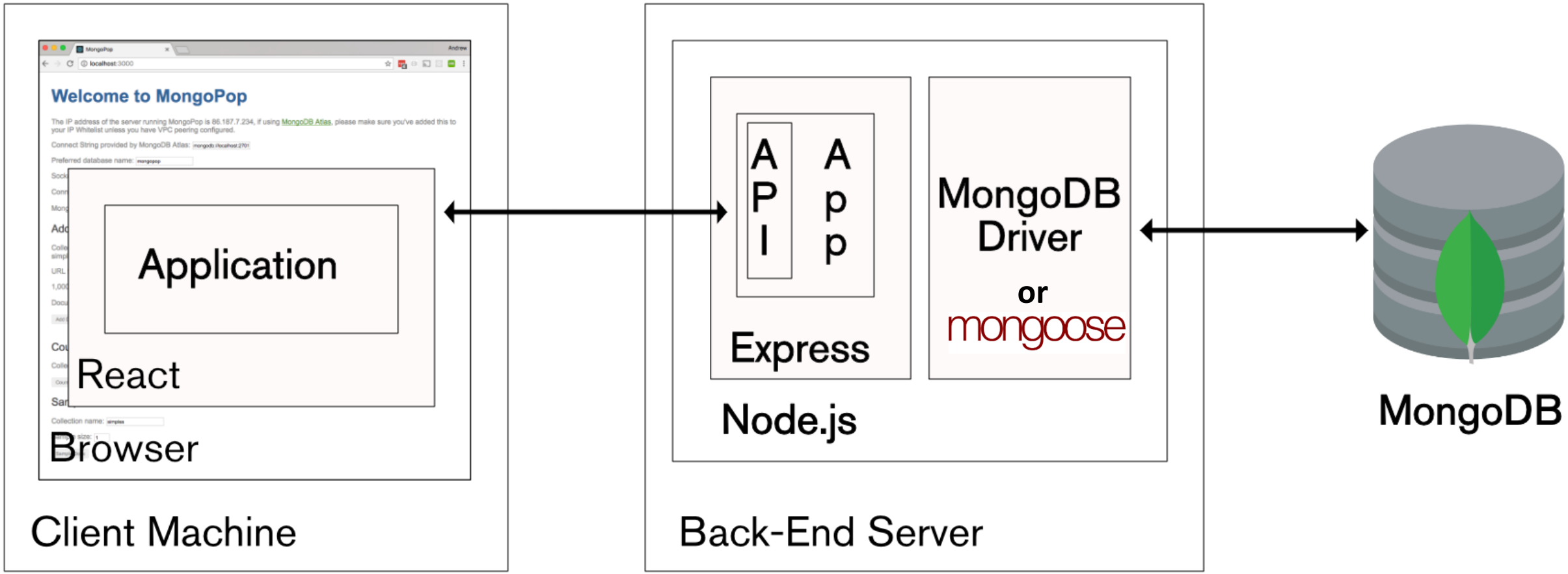
It provides a simplified schema-based method, to model your application data and store it in a MongoDB database.

MERN stack Architecture :



– Node.js Express exports REST APIs & interacts with MongoDB Database using Mongoose ODM.

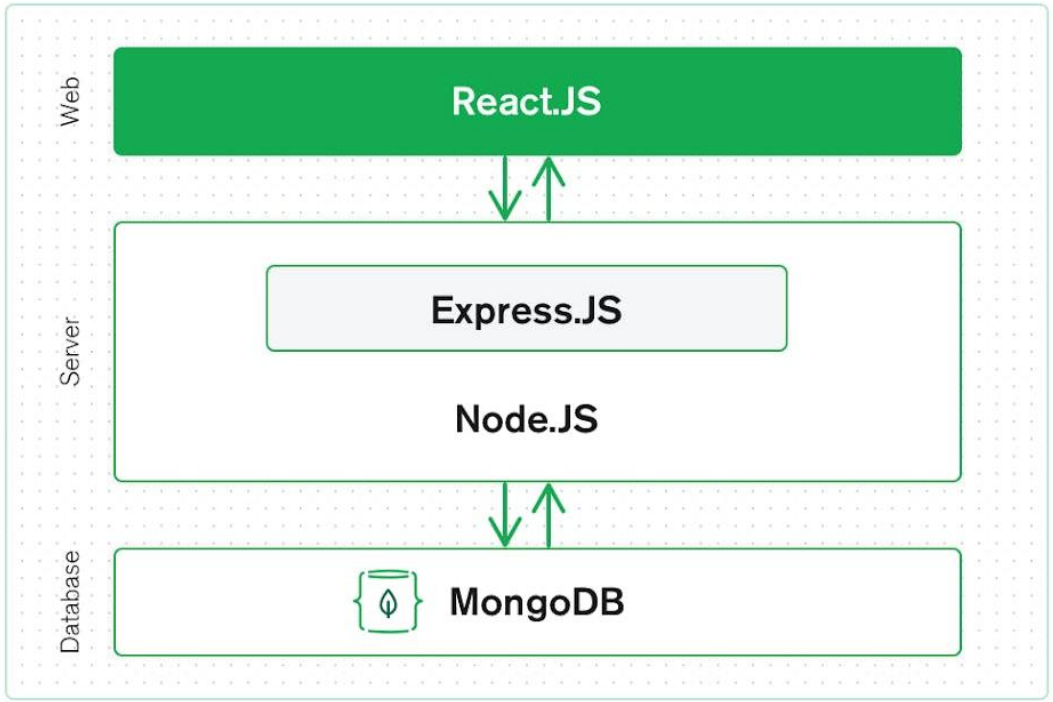
– React sends HTTP Requests and retrieves HTTP Responses using *Axios / Fetch*, and consumes data on the components. React Router is used for navigating to pages.


Welcome to MongoPop
Application
React
Browser
Client Machine
API
App
Express
MongoDB Driver
or
mongoose
Node.js
Back-End Server
MongoDB

MERN Stack

MERN stands for MongoDB, Express, React, Node, after the four key technologies that make up the stack.

- MongoDB — document database
- Express(.js) — Node.js web framework
- React(.js) — a client-side JavaScript framework
- Node(.js) — the premier JavaScript web server

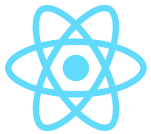


MERN

MEAN

MEVN

MERN Stack

React 

React.js is a front-end JavaScript library for building interactive user interfaces in HTML and communicating with a remote server.

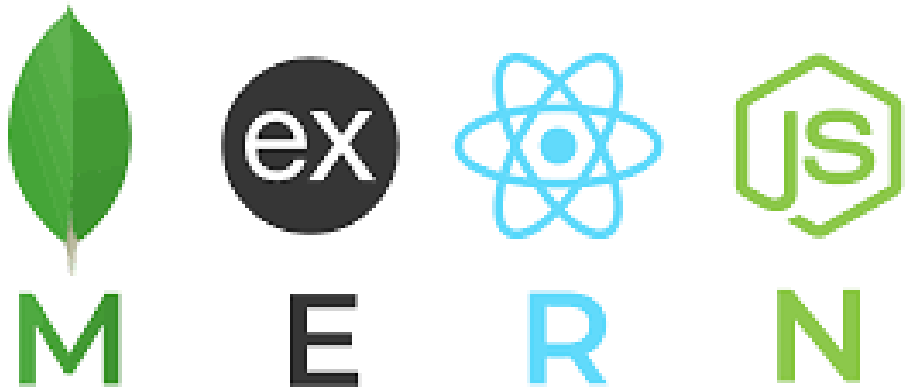
Express Express node

Express.js (running on **Node.js**) is a server-side application framework that wraps HTTP requests and responses and makes it easy to map URLs to server-side functions.

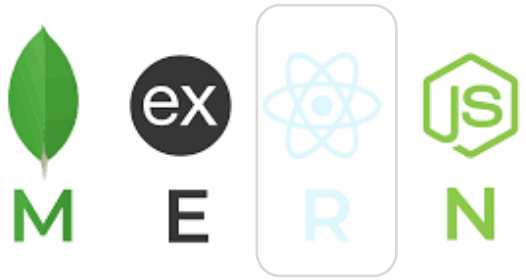
MongoDB database 

JSON documents created in your React.js front end can be sent to the Express.js server, where they can be processed and (assuming they're valid) stored directly in MongoDB for later retrieval.

Reference:
https://www.mongodb.com/mern-stack


ex
JS
M E R N

Prepare Back-end



Abraham N. Aldaco-Gastélum, Ph.D.

Step 1 Prepare the work area in your computer

```
mkdir mern_backend
cd mern_backend
```

Step 2 Create a Javascript file:

```
index.js
```

Step 3 Init node:

```
npm init
```

Step 4 Create a MERN Backend :

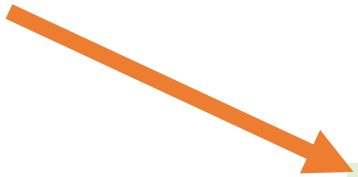
```
npm init -y

npm install express mongoose cors
```

Step 5 Run the application using nodemon :

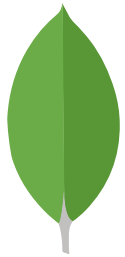
```
nodemon index.js
```

package.json



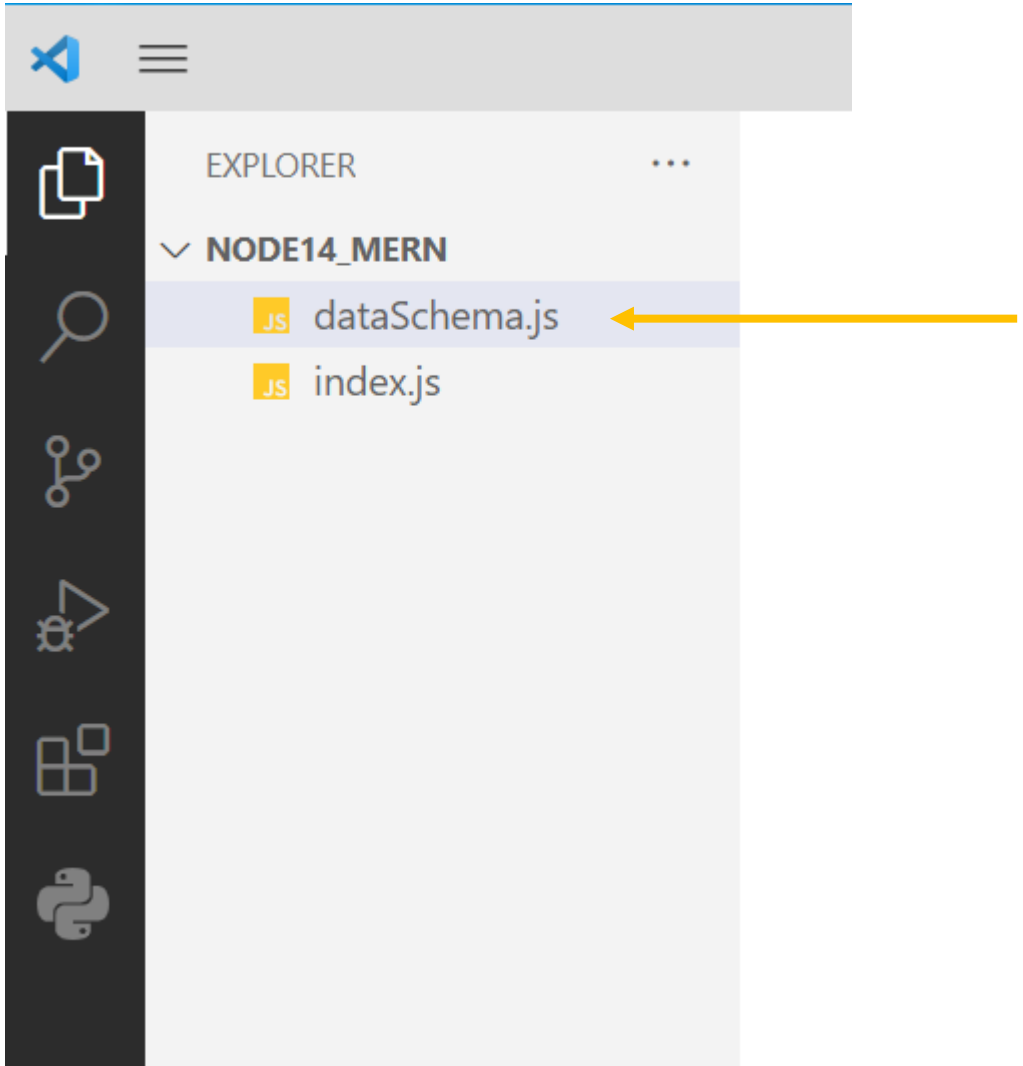
```
{
  "name": "node14_mern_backend",
  "version": "1.0.0",
  "description": "MERN",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [
    "mern",
    "node",
    "express",
    "mongoose",
    "react"
  ],
  "author": "Abraham Aldaco",
  "license": "ISC",
  "dependencies": {
    "cors": "^2.8.5",
    "express": "^4.18.2",
    "mogoose": "^0.0.1-security",
    "mongoose": "^7.0.4"
  }
}
```

Add `dataSchema.js` file



Step 6 Create a Data Schema

Create a new file: **dataSchema.js**.



The schema is used to create a model for the database.

This allows the model to store data in a MongoDB collection according to the structure defined in the Schema.

mongoose requirement.

Add dataSchema.js file:

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

const ReactFormDataSchema = new mongoose.Schema({
  _id: {type: Number},
  title: {type: String},
  price: {type: Number},
  description: {type: String},
  category: {type: String},
  image: {type: String},
  rating: {
    rate : {type: Number},
    count : {type: Number}
  }
} ,
  { collection: "fakestore_catalog" }
)

const Product = mongoose.model('Product', ReactFormDataSchema)
module.exports = Product
```

MODEL

We are using the Fakestore data Structure in JSON format :


```
{
  "_id": 1,
  "title": "Fjallraven - Fckpack, Fits 15
  "price": 109.95,
  "description": "Your perfect pack for ev
  "category": "men's clothing",
  "image": "https://fakestog",
  "rating": { "rate": 3.9, "count": 120 }
}
```


Reference :
https://mongoosejs.com/docs/schematypes.html
https://developer.mozilla.org/en-US/docs/Learn/Server-side/Express_Nodejs/mongoose
https://mongoosejs.com/docs/2.7.x/docs/finding-documents.html

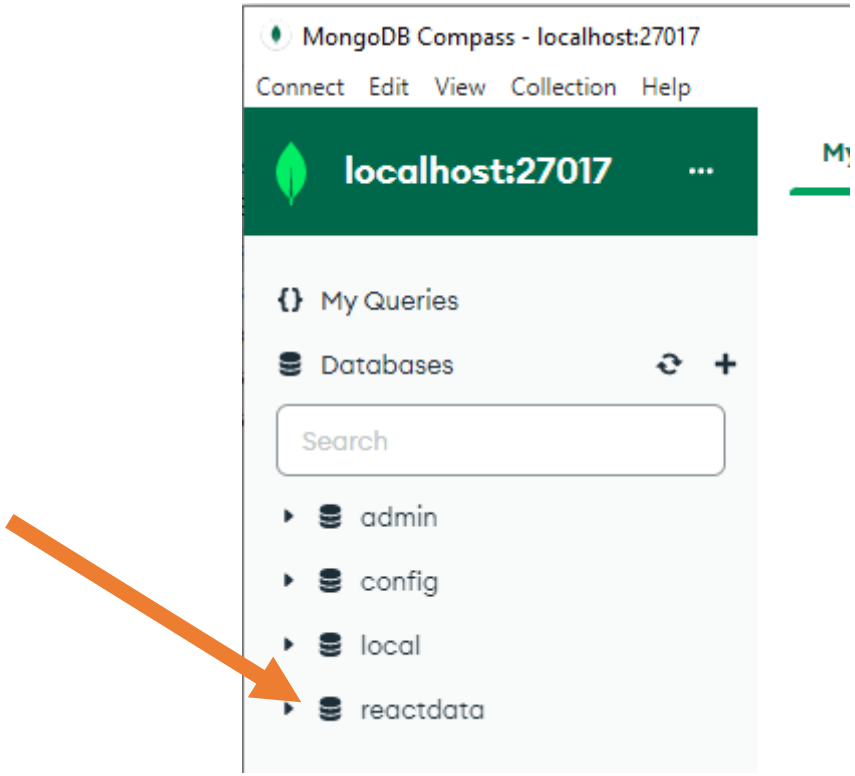
We are going to use Fakestore data in JSON format :

Here, there are three products from the catalog :

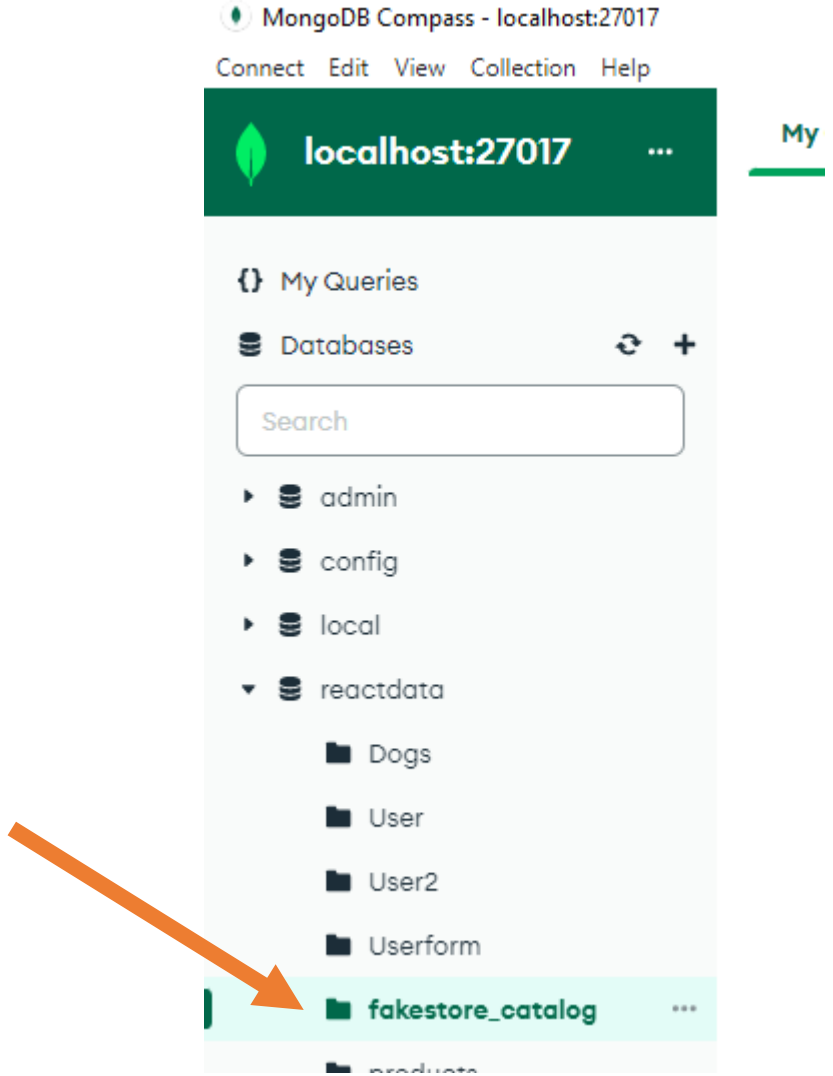

```
[
    {
      "id": 1,
      "title": "Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops",
      "price": 109.95,
      "description": "Your perfect pack for everyday use and walks in the forest. Stash your laptop (up to 15 inches) in the padded sleeve, your
everyday",
      "category": "men's clothing",
      "image": "https://fakestoreapi.com/img/81fPKd-2AYL._AC_SL1500_.jpg",
      "rating": { "rate": 3.9, "count": 120 }
    },
    {
      "id": 2,
      "title": "Mens Casual Premium Slim Fit T-Shirts ",
      "price": 22.3,
      "description": "Slim-fitting style, contrast raglan long sleeve, three-button henley placket, light weight & soft fabric for breathable
and comfortable wearing. And Solid stitched shirts with round neck made for durability and a great fit for casual fashion wear and diehard
baseball fans. The Henley style round neckline includes a three-button placket.",
      "category": "men's clothing",
      "image": "https://fakestoreapi.com/img/71-3HjGNDUL._AC_SY879._SX._UX._SY._UY_.jpg",
      "rating": { "rate": 4.1, "count": 259 }
    },
    {
      "id": 3,
      "title": "Mens Cotton Jacket",
      "price": 55.99,
      "description": "great outerwear jackets for Spring/Autumn/Winter, suitable for many occasions, such as working, hiking, camping,
mountain/rock climbing, cycling, traveling or other outdoors. Good gift choice for you or your family member. A warm hearted love to Father,
husband or son in this thanksgiving or Christmas Day.",
      "category": "men's clothing",
      "image": "https://fakestoreapi.com/img/71li-ujtlUL._AC_UX679_.jpg",
      "rating": { "rate": 4.7, "count": 500 }
    }
]
```

Adding some products to MongoDB by hand

Create a Database “reactdata” :



Create a collection “fakestore_catalog” :



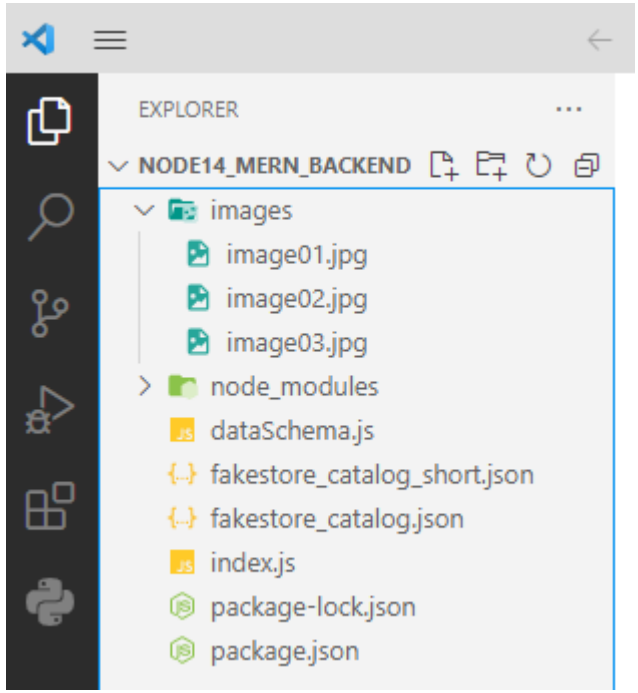
Data to copy-paste to MondoDB :


```
    {
      "_id": 1,
      "title": "Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops",
      "price": 109.95,
      "description": "Your perfect pack for everyday use and walks in the forest. Stash your laptop (up to 15 inches) in the padded sleeve, your
everyday",
      "category": "men's clothing",
      "image": "http://127.0.0.1:4000/images/image01.jpg",
      "rating": { "rate": 3.9, "count": 120 }
    }


    {
      "_id": 2,
      "title": "Mens Casual Premium Slim Fit T-Shirts ",
      "price": 22.3,
      "description": "Slim-fitting style, contrast raglan long sleeve, three-button henley placket, light weight & soft fabric for breathable and
comfortable wearing. And Solid stitched shirts with round neck made for durability and a great fit for casual fashion wear and diehard baseball
fans. The Henley style round neckline includes a three-button placket.",
      "category": "men's clothing",
      "image": "http://127.0.0.1:4000/images/image02.jpg",
      "rating": { "rate": 4.1, "count": 259 }
    }


    {
      "_id": 3,
      "title": "Mens Cotton Jacket",
      "price": 55.99,
      "description": "great outerwear jackets for Spring/Autumn/Winter, suitable for many occasions, such as working, hiking, camping, mountain/rock
climbing, cycling, traveling or other outdoors. Good gift choice for you or your family member. A warm hearted love to Father, husband or son in
this thanksgiving or Christmas Day.",
      "category": "men's clothing",
      "image": "http://127.0.0.1:4000/images/image03.jpg",
      "rating": { "rate": 4.7, "count": 500 }
    }
```

Create a folder `./images` in the Back-end



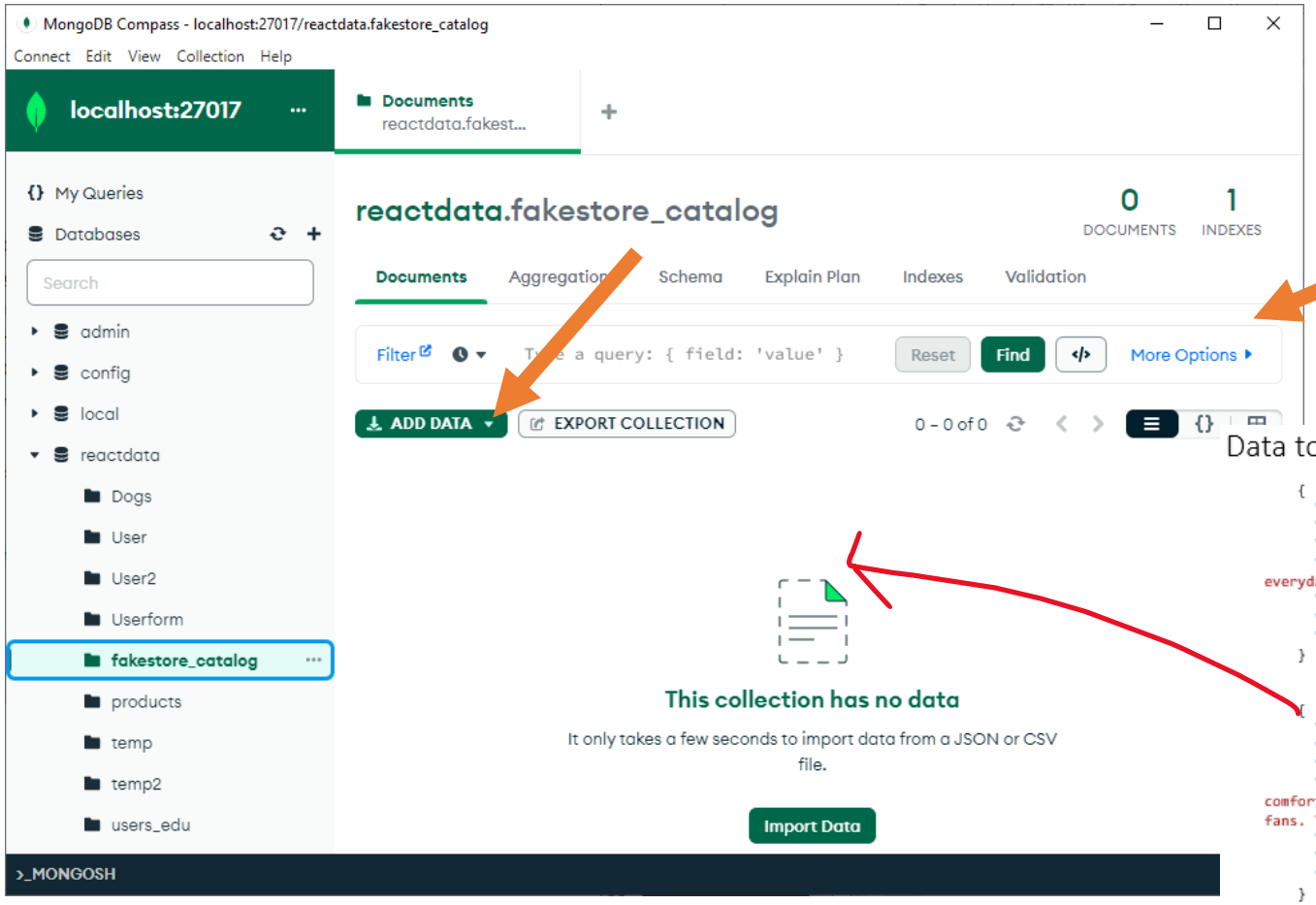
\mern_backend\images\

Copy the images given in Canvas :

The image name most coincide with :

```
"category": "men's clothing",
"image": "http://127.0.0.1:4000/images/image01.jpg",
"rating": { "rate": 3.9, "count": 120 }
```

Add three products by hand to `fakestore_catalog` **collection** to have three **documents**:



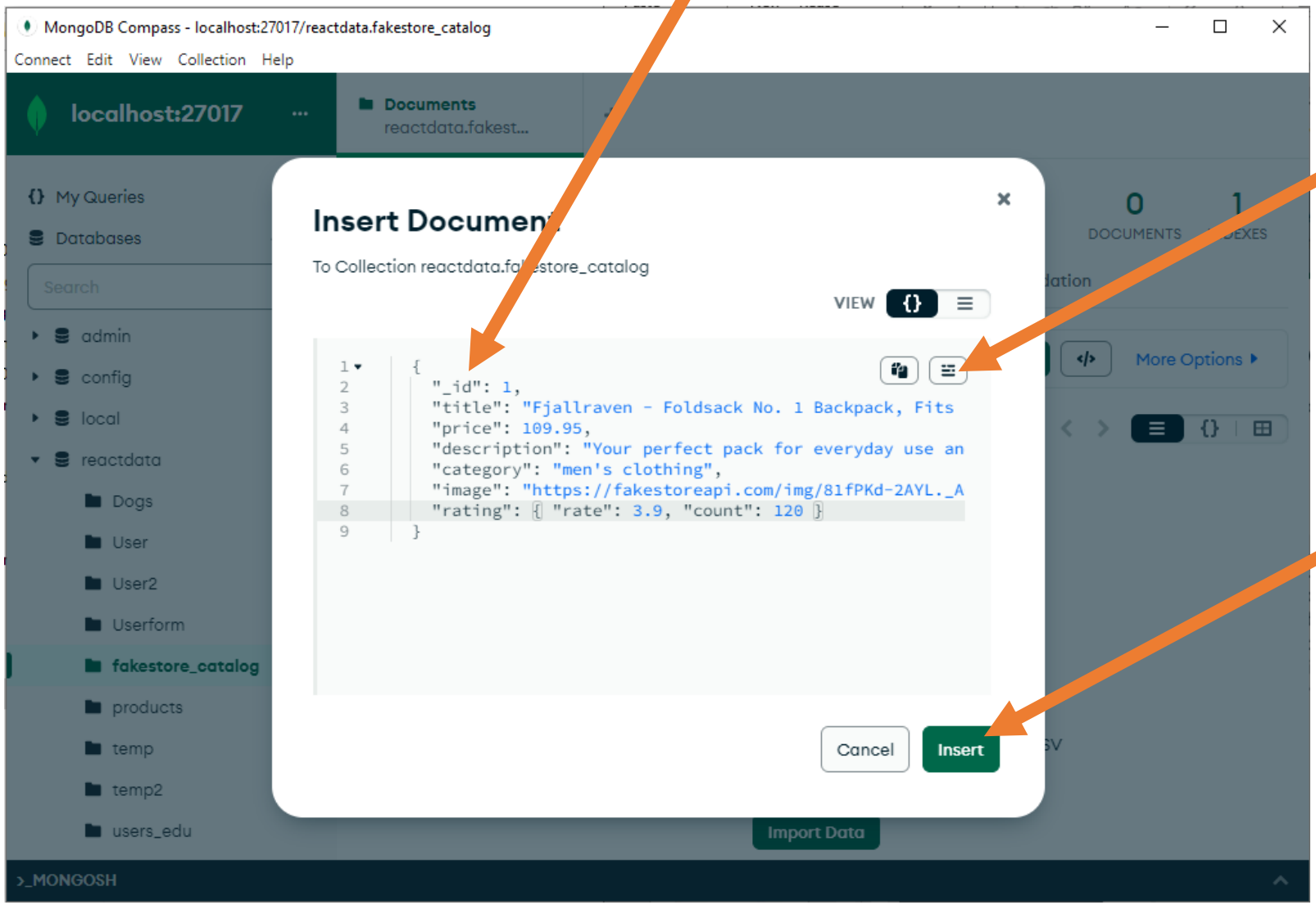
Data to copy-paste to MondoDB :


```
{
  "_id": 1,
  "title": "Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops",
  "price": 109.95,
  "description": "Your perfect pack for everyday use and walks in the forest. Stash your lap
everyday",
  "category": "men's clothing",
  "image": "http://127.0.0.1:4000/images/image01.jpg",
  "rating": { "rate": 3.9, "count": 120 }
}
```

```
{
  "_id": 2,
  "title": "Mens Casual Premium Slim Fit T-Shirts ",
  "price": 22.3,
  "description": "Slim-fitting style, contrast raglan long sleeve, three-button henley plack
comfortable wearing. And Solid stitched shirts with round neck made for durability and a great t
fans. The Henley style round neckline includes a three-button placket.",
  "category": "men's clothing",
  "image": "http://127.0.0.1:4000/images/image02.jpg",
  "rating": { "rate": 4.1, "count": 259 }
}
```

```
{
  "_id": 3,
  "title": "Mens Cotton Jacket",
  "price": 55.99,
  "description": "great outerwear jackets for Spring/Autumn/Winter, suitable for many occasi
climbing, cycling, traveling or other outdoors. Good gift choice for you or your family member.
this thanksgiving or Christmas Day.",
  "category": "men's clothing",
  "image": "http://127.0.0.1:4000/images/image03.jpg",
  "rating": { "rate": 4.7, "count": 500 }
}
```

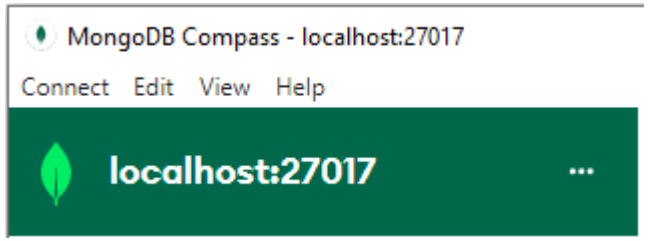
Modify "id" by "_id"



This button helps to `Format` the data.

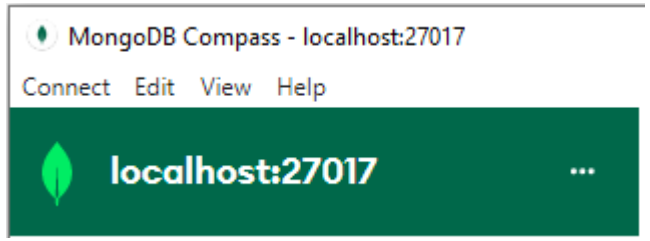
Insert

Test MongoDB database and collection using >_MONGOSH :




```
reactdata>db.fakestore_catalog.find({})
```

```
>_MONGOSH
> db.fakestore_catalog.find({})
< [
    {
      _id: 1,
      title: 'Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops',
      price: 109.95,
      description: 'Your perfect pack for everyday use and walks in the forest. Stash your laptop (up to 15 inches) in the padd
      category: "men's clothing",
      image: 'https://fakestoreapi.com/img/81fPKd-2AYL._AC_SL1500_.jpg',
      rating: {
        rate: 3.9,
        count: 120
      }
    }
    {
      _id: 2,
      title: 'Mens Casual Premium Slim Fit T-Shirts ',
      price: 22.3,
      description: 'Slim-fitting style, contrast raglan long sleeve, three-button henley placket, light weight & soft fabric fo
      category: "men's clothing",
      image: 'https://fakestoreapi.com/img/71-3HjGNDUL._AC_SY879._SX._UX._SY._UY_.jpg',
      rating: {
        rate: 4.1,
        count: 259
      }
    }
    {
```




```
reactdata>db.fakestore_catalog.find({_id:1})
```

```
> db.fakestore_catalog.find({_id:1})
<   {
      _id: 1,
      title: 'Fjallraven - Foldsack No. 1 Backpack, Fits 15 Laptops',
      price: 109.95,
      description: 'Your perfect pack for everyday use and walks in the forest. Stash your laptop (up to 15 inches) in the padd
      category: "men's clothing",
      image: 'https://fakestoreapi.com/img/81fPKd-2AYL._AC_SL1500_.jpg',
      rating: {
        rate: 3.9,
        count: 120
      }
    }
reactdata>
```

Set up the Express server in `index.js`

index.js

mongoose

```
const express = require("express");
const mongoose = require("mongoose");
const cors = require("cors");
const app = express();

app.use(express.json());
app.use(cors());
```

database

```
mongoose.connect("mongodb://127.0.0.1:27017/reactdata",
  {
    dbName: "reactdata",
    useNewUrlParser: true,
    useUnifiedTopology: true,
  }
);
```

web server port

```
const port = process.env.PORT || 4000;
const host = "localhost";
app.listen(port, () => {
  console.log(`App listening at http://%s:%s`, host, port);
});
```

GET Method

Set it up in Express in `index.js` file

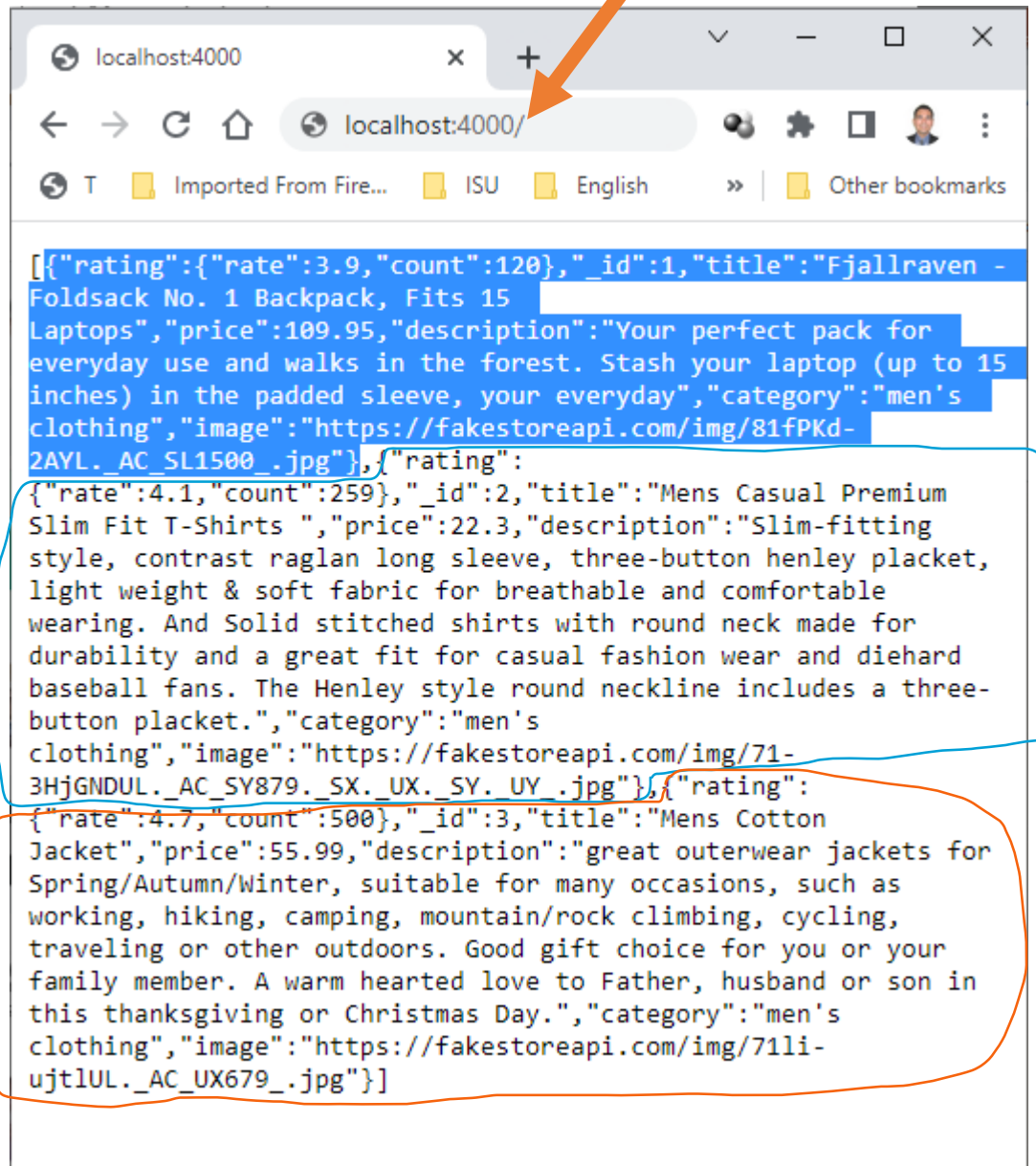
query all products

find

```
app.get("/", async (req, resp) => {
  const query = {};
  const allProducts = await Product.find(query);
  console.log(allProducts);
  resp.send(allProducts);
});
```

Test GET -`find({})` in a browser

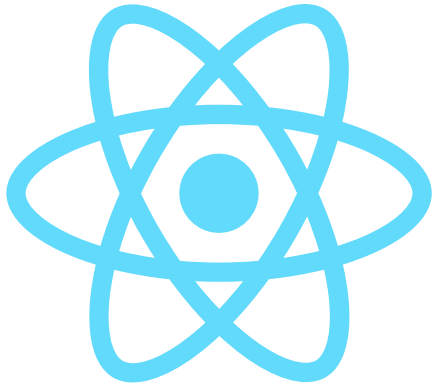
query all products



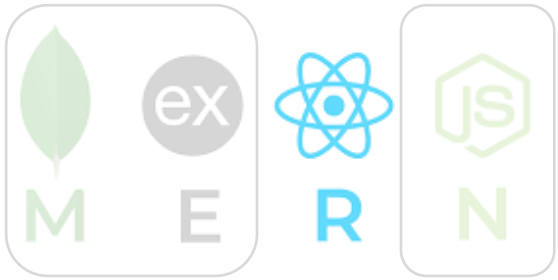
"_id": 1

"_id": 2

"_id": 3



Prepare Front-end



Abraham N. Aldaco-Gastélum, Ph.D.

Step 1 Prepare the work area in your computer

```
npx create-react-app mern_frontend

cd mern_frontend
```

Step 2 We are going to use the files :

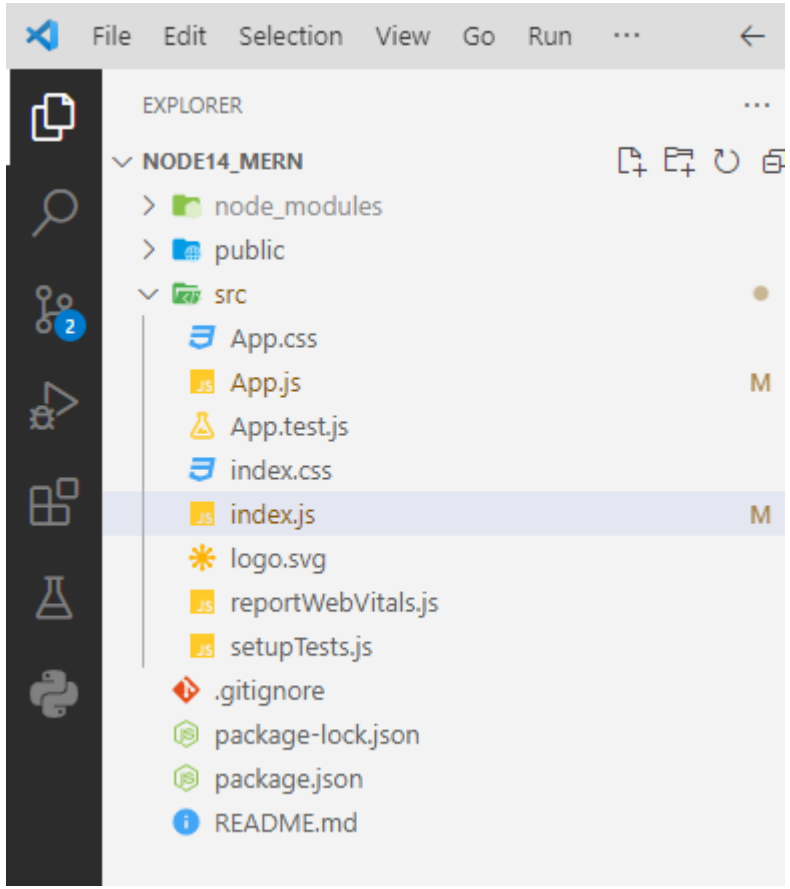
`App.js`

`index.js`

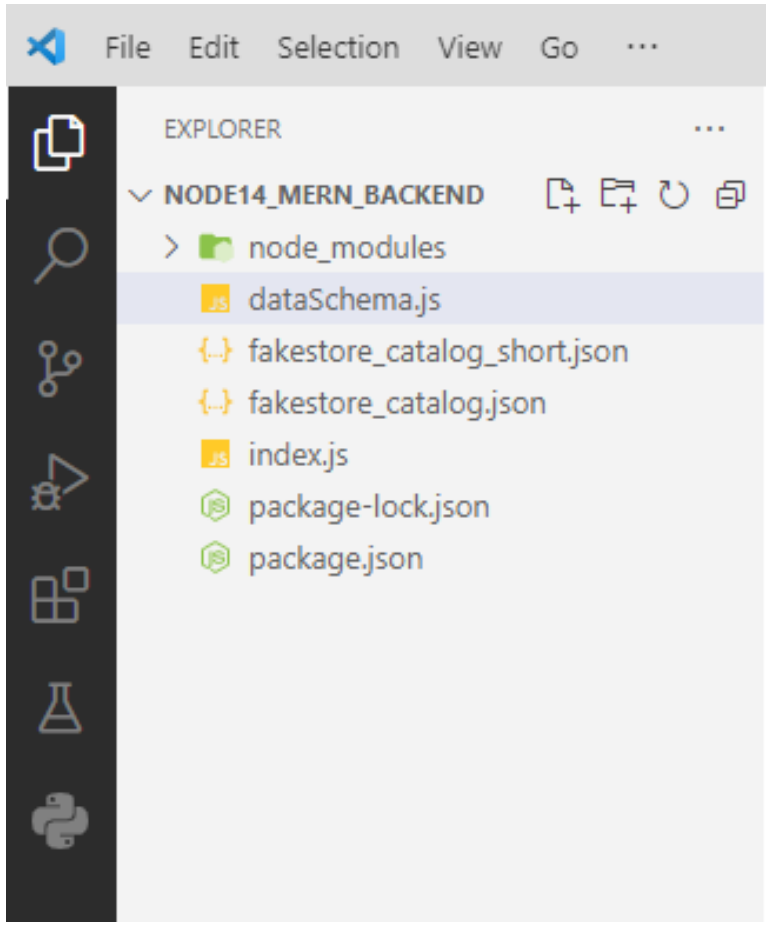
Don't confuse this with the `index.js` in the Back-end

You can use two windows of VSC :

Front-end



Back-end



Front-end - Set `index.js` :

```
import React from "react";
import ReactDOM from "react-dom/client";
import "./index.css";
import App from "./App";

const root = ReactDOM.createRoot(document.getElementById("root"));
root.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>
);
```

Front-end - State variables in **App.js**

```
import { useState, useEffect } from "react";
function App() {
  const [product, setProduct] = useState([]);
  const [viewer1, setViewer1] = useState(false);


return (

     . . .

}  // App end

export default App;
```

Return in App.js to show all Products from MongoDB :

```
return (
    <div>
      <h1>Catalog of Products</h1>

      <button onClick={() => getAllProducts()}>Show All products</button>

      <h1>Show all available Products.</h1>
      <hr></hr>
      {viewer1 && <div>Products {showAllItems}</div>}
      <hr></hr>

    </div>
  );

}  // App end

export default App;
```

getAllProducts()

viewer1

showAllItems

Front-end - getAllProducts() in App.js

```
function getAllProducts() {
  fetch("http://localhost:4000/")
    .then((response) => response.json())
    .then((data) => {
      console.log("Show Catalog of Products :");
      console.log(data);
      setProduct(data);
    });
  setViewer1(!viewer1);
}
```

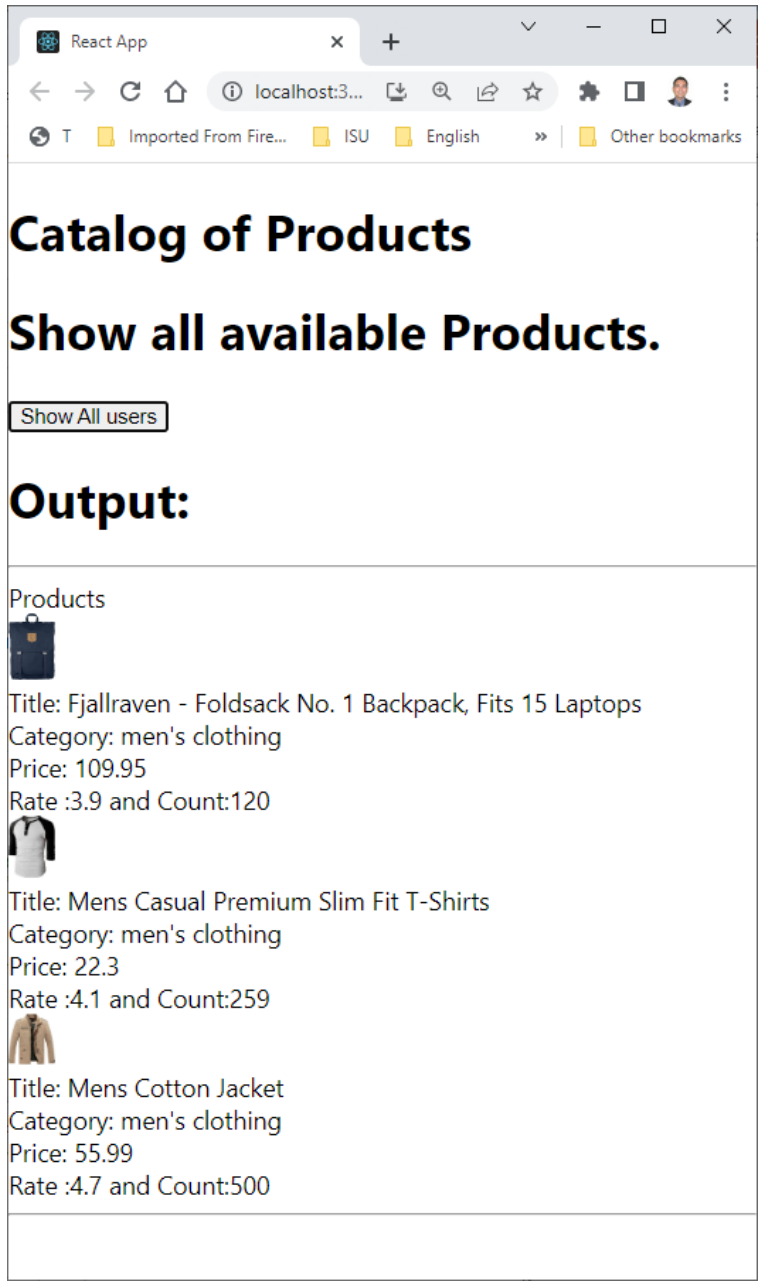
State Variable product

State Variable viewer1

Front-end - showAllItems in App.js


```
const showAllItems = product.map((el) => (
  <div key={el._id}>
    <img src={el.image} width={30} /> <br />
    Title: {el.title} <br />
    Category: {el.category} <br />
    Price: {el.price} <br />
    Rate :{el.rating.rate} and Count:{el.rating.count} <br />
  </div>
));
```

Show all products in Front-end `npm start`:



Improve this view using Bootstrap.

GET Method by `id`
Set it up in Express in `index.js` file

Back-end : app.get("/:id",

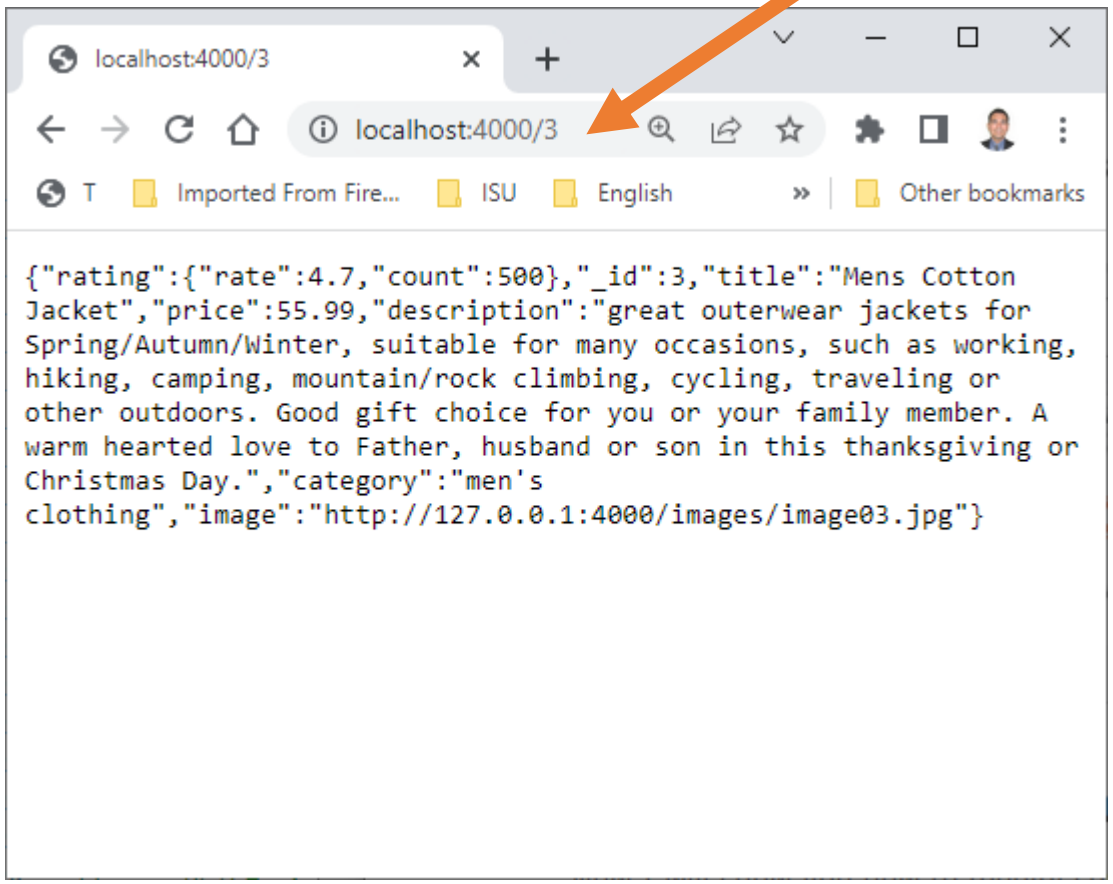
query one id

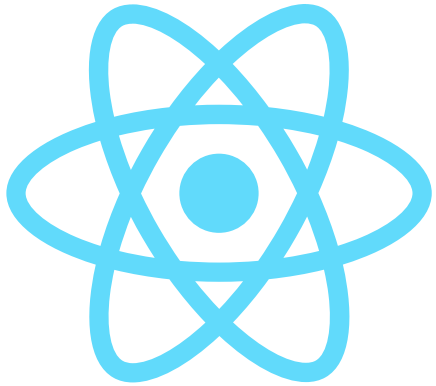
findOne

```
app.get("/:id", async (req, resp) => {
  const id = req.params.id;
  const query = { _id: id };
  const oneProduct = await Product.findOne(query);
  console.log(oneProduct);
  resp.send(oneProduct);
});
```

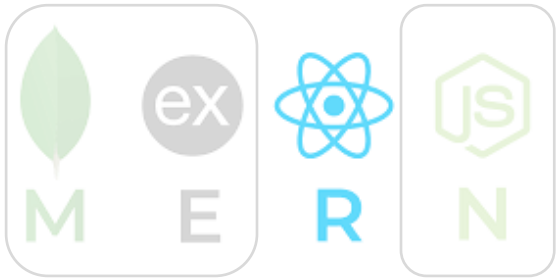
Test `"/:id"` in a browser :

query one id :
http://localhost:4000/3





Prepare Front-end



Abraham N. Aldaco-Gastélum, Ph.D.

Front-end : State variables in **App.js**

```
import { useState, useEffect } from "react";
function App() {
  const [product, setProduct] = useState([]);
  const [viewer1, setViewer1] = useState(false);

  const [oneProduct, setOneProduct] = useState([]);
  const [viewer2, setViewer2] = useState(false);

. . .
```

Front-end : Modify `return` in `App.js` :

```
return (
    <div>
      <h1>Catalog of Products </h1>

      <button onClick={() => getAllProducts()}>Show All users</button>

      <input type="text" id="message" name="message" placeholder="id" onChange={(e) =>getOneProduct(e.target.value)} />


      <h1>Show all available Products:</h1>
      <hr></hr>
      {viewer1 && <div>Products {showAllItems}</div>}
      <hr></hr>
      <h1>Show one Product by Id:</h1>
      {viewer2 && <div>Product: {showOneItem}</div>}
      <hr></hr>
    </div>
  );
}

export default App;
```

getOneProduct

id

viewer2

showOneItem

Front-end : getOneProduct(id) in App.js

parameter id

```
function getOneProduct(id) {
  console.log(id);
  if (id >= 1 && id <= 20) {
    fetch("http://localhost:4000/" + id)
      .then((response) => response.json())
      .then((data) => {
        console.log("Show one product :", id);
        console.log(data);
        const dataArr = [];
        dataArr.push(data);
        setOneProduct(dataArr);
      });
    setViewer2(!viewer2);
  } else {
    console.log("Wrong number of Product id.");
  }
}
```

state variable viewer2

state variable oneProduct

Front-end : showOneItem in App.js


```
const showOneItem = oneProduct.map((el) => (
  <div key={el._id}>
    <img src={el.image} width={30} /> <br />
    Title: {el.title} <br />
    Category: {el.category} <br />
    Price: {el.price} <br />
    Rate :{el.rating.rate} and Count:{el.rating.count} <br />
  </div>
));
```

Show one product in Front-end by id: