Steps to create a node server that connects with mongodb and work as backend for react app

1. Create a folder named "backend"
2. Open folder in vscode
3. open command window in vs code
4. execute command `npm init` (press enter key multiple time to complete the command)
5. execute following command in command window

   ```
   npm install express mongoose body-parser cors --save
   ```

6. After instalation create a file named server.js
7. create a server file using following code as reference

```
const express = require('express');
const mongoose = require('mongoose');
const bodyParser = require('body-parser');
const cors = require('cors');

// Create an Express application
const app = express();

// Middleware
app.use(bodyParser.json()); // parse req from client into JSON
app.use(cors()); // Enable CORS for all routes

// MongoDB Connection -- Change mydatabase with Your DB name
mongoose.connect('mongodb://127.0.0.1:27017/mydatabase');

const db = mongoose.connection;

db.on('error', (err) => {
  console.error('MongoDB connection error:', err);
});

db.once('open', () => {
  console.log('Connected to MongoDB');
});

// Define a sample Mongoose model
// If your application has more then one collections(tables)
// create one model for one collection(table)
// following the example given bellow
// replace Item with your collection(table) name
const Item = mongoose.model('Item', {
  name: String,
  description: String,
  // you have add all fields(column name) in this format
  // fieldName : DataType,
});

// CRUD endpoints
// all this endpoints are for one collection(table) called Item
// you have to repeat all this endpoints
// for each of your collections(tables)

//API ENDPOINT to Add New item
app.post("/Additem", async (req, res) => {
    try {
      let data = await req.body;
      const item = new item(data);
      const result = item.save();
      res.send(result);
    } catch (error) {
```

```
      res.send(error);
    }
  });
  //API ENDPOINT to Get All item
  app.get("/getitem", async (req, res) => {
    try {
      let response = await item.find().exec();
      res.send(response);
    } catch (error) {
      console.log("error", error);
    }
  });
  //API ENDPOINT to Get filtered item
  app.get("/getitem/:id", async (req, res) => {
    try {
      let id = req.params.id;
      const result = await item.findById(id);
      res.send(result);
    } catch (error) {
      res.send(error);
    }
  });
  //Api ENDPOINT to update record
  app.patch("/updateitem/:id", async (req, res) => {
    try {
      let id = req.params.id;
      let data = req.body;
      let response = await item.findByIdAndUpdate(id, data);
      res.send(response);
    } catch (error) {
       res.send(error);
    }
  });

  app.delete("/deleteitem/:id", async (req, res) => {
    try {
      let id = req.params.id;
      const result = await item.findByIdAndRemove(id);
      res.send(result);
    } catch (error) {
       res.send(error);
    }
  });

// Start the server
const port = process.env.PORT || 5000;
app.listen(port, () => {
  console.log(`Server is running on port ${port}`);
});
```

8. open packege.json file and add `"start":"node server.js"` under `scripts` section consider the following code as reference

```
{
 "name": "backend",
 "version": "1.0.0",
 "description": "",
 "main": "server.js",
 "scripts": {
   "start": "node server.js", // Add this line
   "test": "echo \"Error: no test specified\" && exit 1"
 },
 "author": "",
 "license": "ISC",
 "dependencies": {
   "cors": "^2.8.5",
   "express": "^4.18.2"
 }
}
```

9. after completeing it go to command window of vscode editor and execute `npm start` it will start your server connected with mongodb on given port number

10. your server IP(localhost) and port number given by you in server.js file will be used in your react app for sending request to the server.

```
`eg. http://localhost:3000`
```