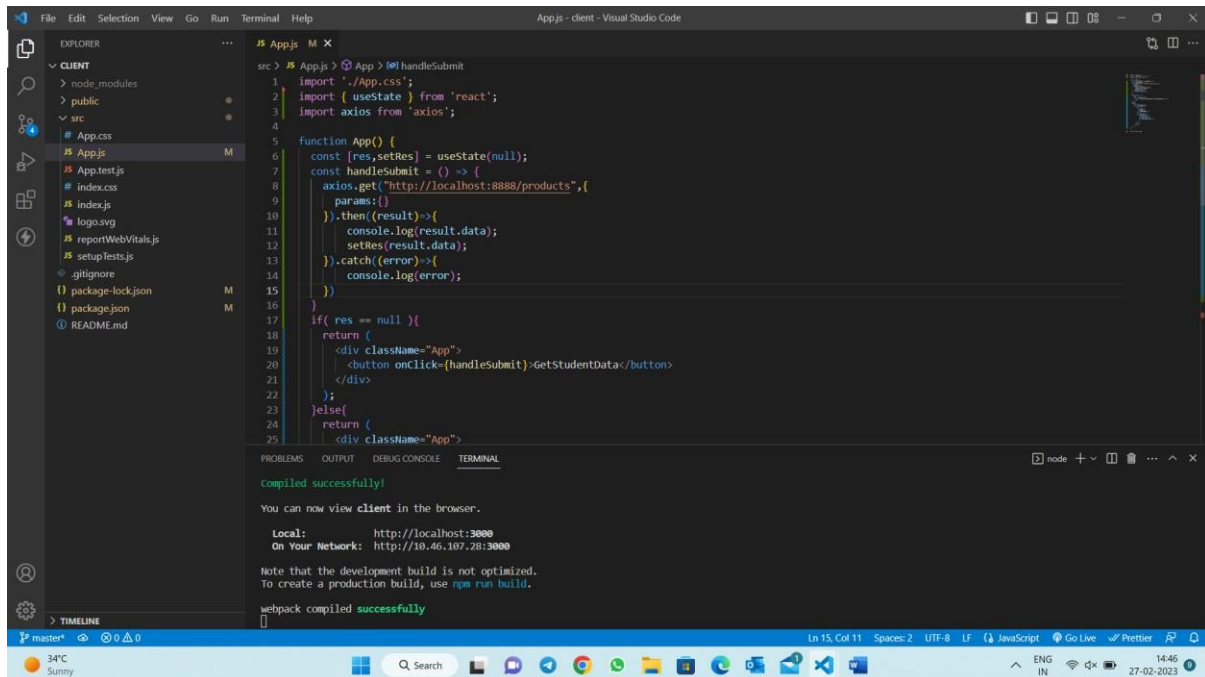


MSWD IN SEM LAB EXAM:-

Name:- K. Vijaya Lakshmi Renuka

ID no:- 2100030609

Client side:-



```
src > App.js > App > handleSubmit
1 import './App.css';
2 import { useState } from 'react';
3 import axios from 'axios';
4
5 function App() {
6   const [res, setRes] = useState(null);
7   const handleSubmit = () => {
8     axios.get("http://localhost:8888/products",{
9       params:{
10         // ...
11       }
12     }).then(result=>{
13       console.log(result.data);
14       setRes(result.data);
15     }).catch(error=>{
16       console.log(error);
17     })
18   }
19   if( res == null ){
20     return (
21       <div className="App">
22         <button onClick={handleSubmit}>GetStudentData</button>
23       </div>
24     );
25   } else{
26     return (
27       <div className="App">
28         // ...
29       </div>
30     );
31   }
32 }
```

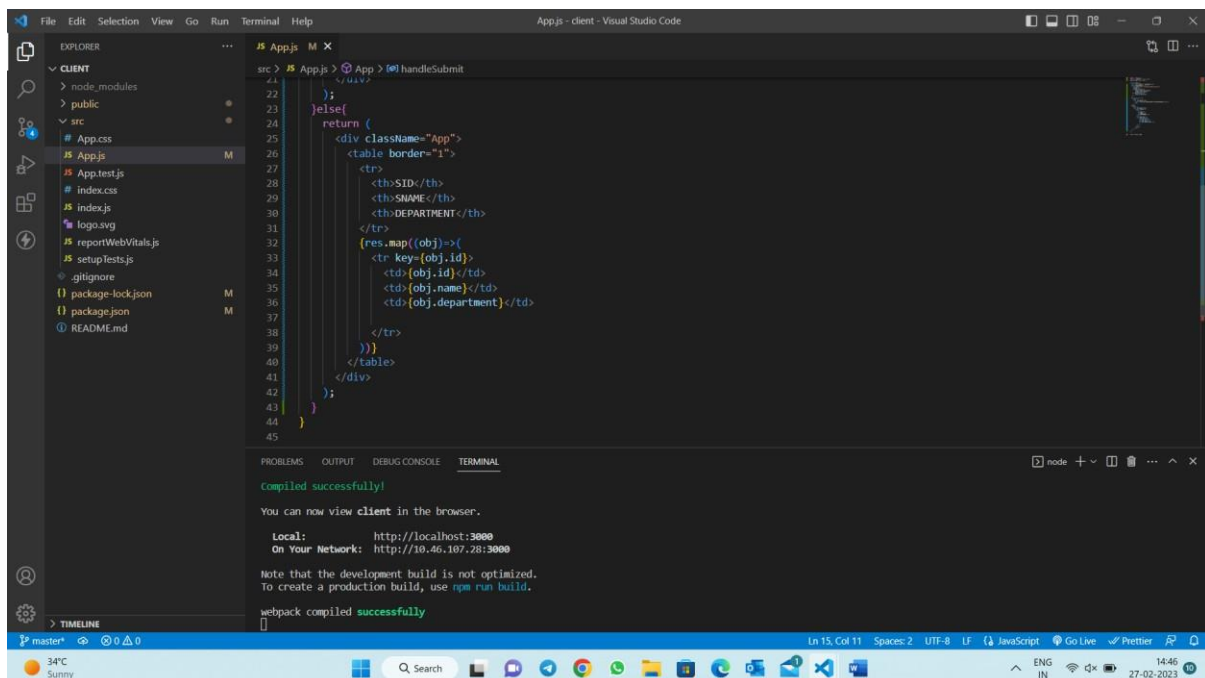
Compiled successfully!

You can now view client in the browser.

Local: http://localhost:3000
On Your Network: http://10.46.107.28:3000

Note that the development build is not optimized.
To create a production build, use `npm run build`.

webpack compiled successfully



```
src > App.js > App > handleSubmit
21 </div>
22
23 } else{
24   return (
25     <div className="App">
26       <table border="1">
27         <tr>
28           <th>SID</th>
29           <th>SNAME</th>
30           <th>DEPARTMENT</th>
31         </tr>
32         {res.map((obj)=>{
33           <tr key={obj.id}>
34             <td>{obj.id}</td>
35             <td>{obj.name}</td>
36             <td>{obj.department}</td>
37           </tr>
38         })}
39       </table>
40     </div>
41   );
42 }
43
44 }
```

Compiled successfully!

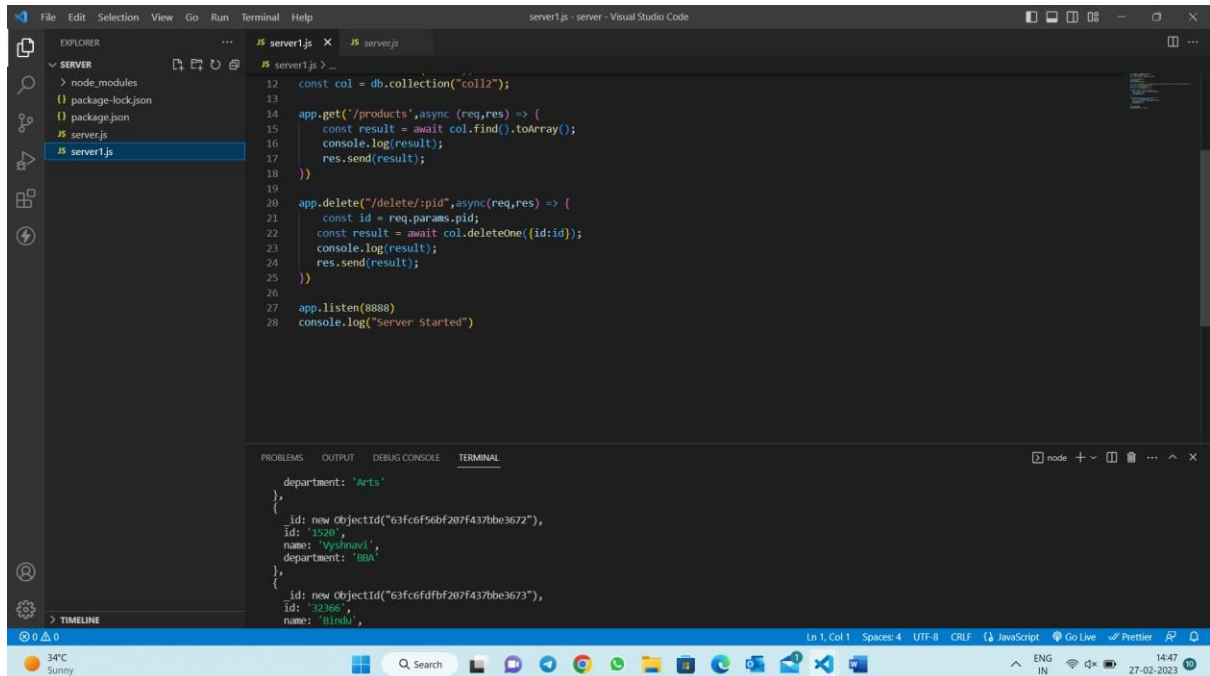
You can now view client in the browser.

Local: http://localhost:3000
On Your Network: http://10.46.107.28:3000

Note that the development build is not optimized.
To create a production build, use `npm run build`.

webpack compiled successfully

Server side:-

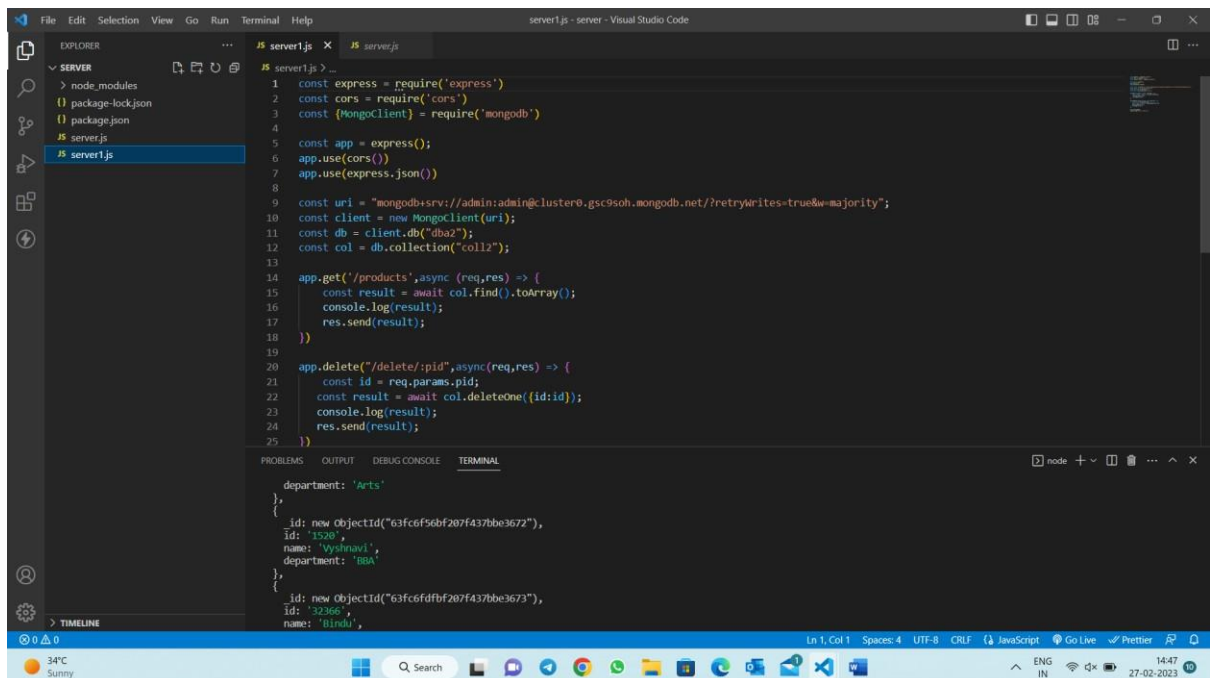


The screenshot shows a Visual Studio Code editor with a file named `server.js` open. The Explorer sidebar on the left shows a project structure with `node_modules`, `package-lock.json`, `package.json`, `server.js`, and `server`. The `server.js` file contains the following JavaScript code:

```
12 const col = db.collection("coll2");
13
14 app.get('/products', async (req, res) => {
15   const result = await col.find().toArray();
16   console.log(result);
17   res.send(result);
18 })
19
20 app.delete("/delete/:pid", async (req, res) => {
21   const id = req.params.pid;
22   const result = await col.deleteOne({id:id});
23   console.log(result);
24   res.send(result);
25 })
26
27 app.listen(8888)
28 console.log("Server: Started")
```

The terminal at the bottom shows the output of the server:

```
department: 'Arts'
{
  _id: new ObjectId("63fc6f5ebf207f437bbe3672"),
  id: '1520',
  name: 'Vyshnavi',
  department: 'BBA'
},
{
  _id: new ObjectId("63fc6f5ebf207f437bbe3673"),
  id: '32366',
  name: 'Bindu',
```

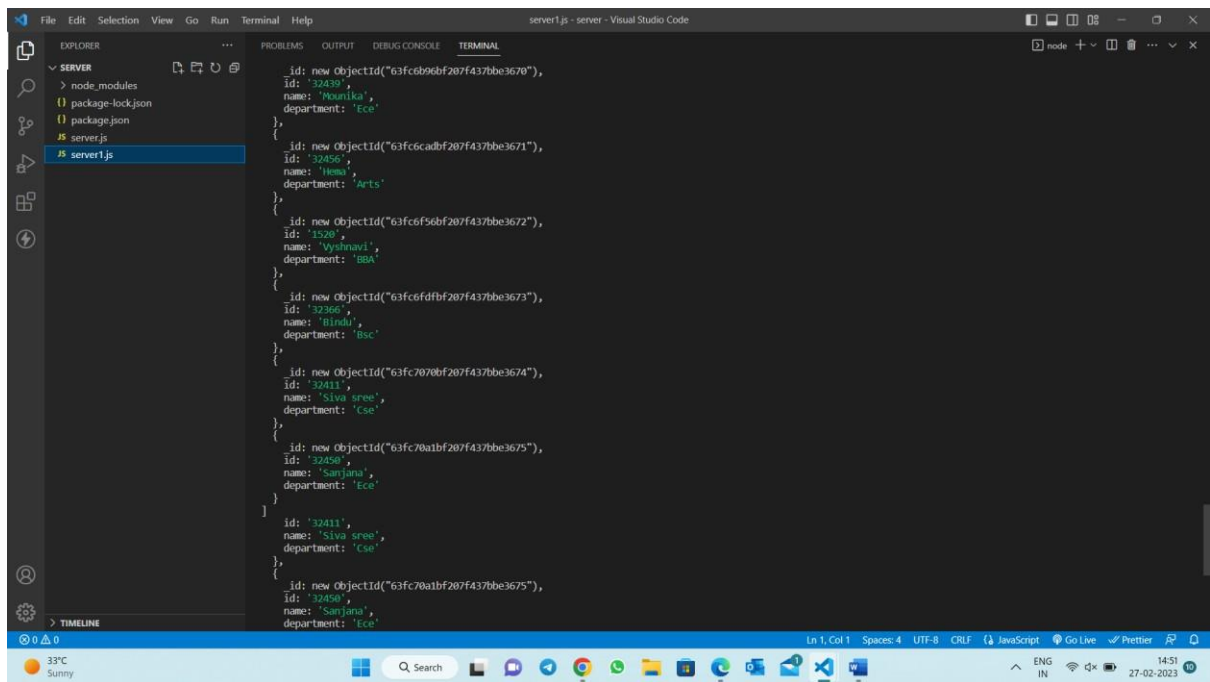


The screenshot shows a Visual Studio Code editor with a file named `server.js` open. The Explorer sidebar on the left shows a project structure with `node_modules`, `package-lock.json`, `package.json`, `server.js`, and `server`. The `server.js` file contains the following JavaScript code:

```
1 const express = require('express')
2 const cors = require('cors')
3 const {MongoClient} = require('mongodb')
4
5 const app = express();
6 app.use(cors());
7 app.use(express.json())
8
9 const url = "mongodb+srv://admin:admin@cluster0.gsc9soh.mongodb.net/?retryWrites=true&majority";
10 const client = new MongoClient(url);
11 const db = client.db("db2");
12 const col = db.collection("coll2");
13
14 app.get('/products', async (req, res) => {
15   const result = await col.find().toArray();
16   console.log(result);
17   res.send(result);
18 })
19
20 app.delete("/delete/:pid", async (req, res) => {
21   const id = req.params.pid;
22   const result = await col.deleteOne({id:id});
23   console.log(result);
24   res.send(result);
25 })
```

The terminal at the bottom shows the output of the server:

```
department: 'Arts'
{
  _id: new ObjectId("63fc6f5ebf207f437bbe3672"),
  id: '1520',
  name: 'Vyshnavi',
  department: 'BBA'
},
{
  _id: new ObjectId("63fc6f5ebf207f437bbe3673"),
  id: '32366',
  name: 'Bindu',
```



```
server1.js - server - Visual Studio Code

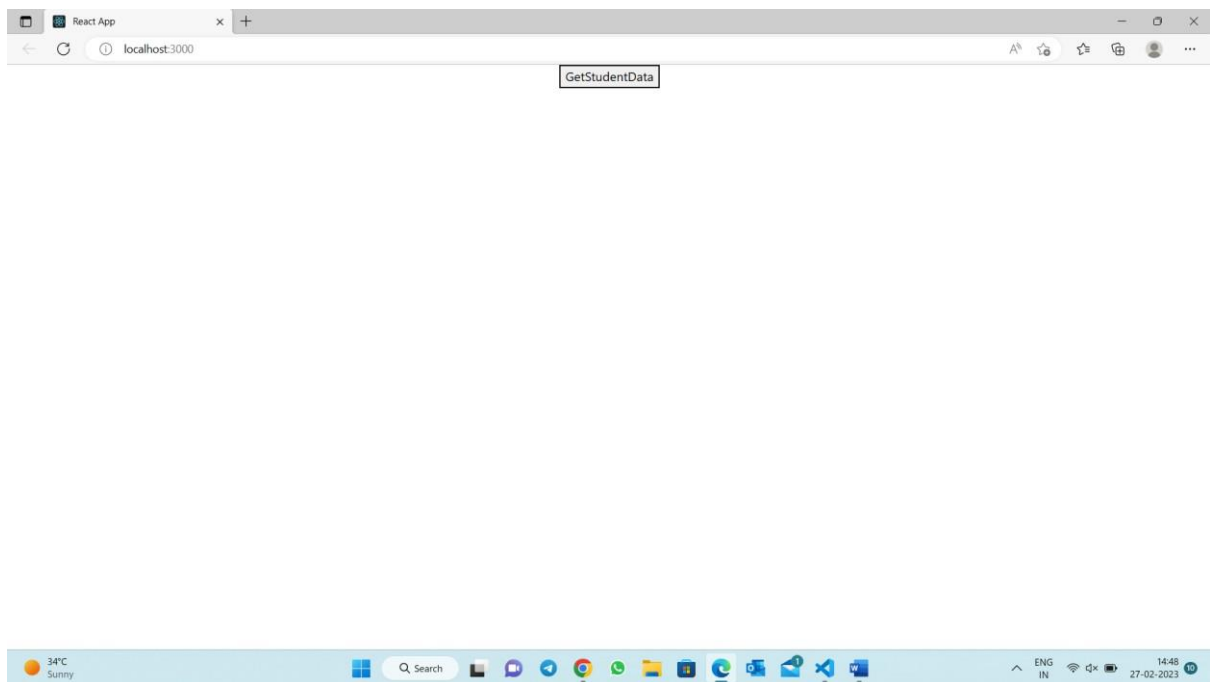
EXPLORER
  SERVER
    > node_modules
    {} package-lock.json
    {} package.json
    JS server.js
    JS server1.js

server1.js
  {
    _id: new ObjectId("63fc6b96bf207f437bbe3670"),
    id: '32439',
    name: 'Mounika',
    department: 'Ece'
  },
  {
    _id: new ObjectId("63fc6cadbf207f437bbe3671"),
    id: '32456',
    name: 'Neesa',
    department: 'Arts'
  },
  {
    _id: new ObjectId("63fc6f56bf207f437bbe3672"),
    id: '1520',
    name: 'Vyshnavi',
    department: 'BBA'
  },
  {
    _id: new ObjectId("63fc6fd9bf207f437bbe3673"),
    id: '32366',
    name: 'Bindu',
    department: 'Bsc'
  },
  {
    _id: new ObjectId("63fc7070bf207f437bbe3674"),
    id: '32411',
    name: 'Siva Sree',
    department: 'Cse'
  },
  {
    _id: new ObjectId("63fc70a1bf207f437bbe3675"),
    id: '32450',
    name: 'Sanjana',
    department: 'Ece'
  }
]
  id: '32411',
  name: 'Siva Sree',
  department: 'Cse'
},
  {
    _id: new ObjectId("63fc70a1bf207f437bbe3675"),
    id: '32450',
    name: 'Sanjana',
    department: 'Ece'
  }
]

TIMELINE
  33°C Sunny

Ln 1, Col 1  Spaces: 4  UTF-8  CRLF  JavaScript  Go Live  Prettier
```

Output screen shots:-



ON CLICKING GETSTUDENTDATA we will get stored data from server side

SID	SNAME	DEPARTMENT
32417	Sowmya	Cse
32439	Mounika	Ece
32456	Hema	Arts
1520	Vyshnavi	BBA
32366	Bindu	Bsc
32411	Siva sree	Cse
32450	Sanjana	Ece

Mongodb data storing:-

The screenshot displays the MongoDB Atlas web interface. On the left sidebar, the 'Database' section is expanded, showing a hierarchy of 'db1' > 'dba2' > 'coll2'. The main panel shows the 'Find' tab for the 'coll2' collection. A filter is applied: `{ field: 'value' }`. The results show two documents:

```

{
  department: "Arts"
}

{
  _id: ObjectId("63fc6f56bf287f437bbe3672"),
  id: "1520",
  name: "Vyshnavi",
  department: "BBA"
}

{
  _id: ObjectId("63fc6fd6bf287f437bbe3673"),
  id: "32366",
  name: "Bindu",
  department: "Bsc"
}

```

At the top of the main panel, statistics are shown: STORAGE SIZE: 36KB, LOGICAL DATA SIZE: 129B, TOTAL DOCUMENTS: 2, INDEXES TOTAL SIZE: 36KB. An 'INSERT DOCUMENT' button is visible in the top right corner of the results area.

Lab-Insem-Exam-Question x Data | Cloud: MongoDB Cloud x +

cloud.mongodb.com/v2/63ea0e62e516ec175f226b56#/metrics/replicaSet/63ea0edec3971cdfe9d21ac7/explorer/dba2/coll2/find

Atlas Bommisetty'... Access Manager Billing All Clusters Get Help Bommisetty

Project 0 Data Services App Services Charts

DEPLOYMENT Database Data Lake PREVIEW SERVICES Triggers Data API Data Federation Search SECURITY Quickstart Database Access Network Access Advanced Goto

Search Namespaces db1 dba2 coll2 s3

STORAGE SIZE: 36KB LOGICAL DATA SIZE: 129B TOTAL DOCUMENTS: 2 INDEXES TOTAL SIZE: 36KB

Find Indexes Schema Anti-Patterns Aggregation Search Indexes

INSERT DOCUMENT

FILTER { field: 'value' } OPTIONS Apply Reset

```
{
  "_id": ObjectId("63fc7870bf207f437b0e3674"),
  "id": "32411",
  "name": "Siva sree",
  "department": "Cse"
}
```

```
{
  "_id": ObjectId("63fc78a1bf207f437b0e3675"),
  "id": "32450",
  "name": "Sanjana",
  "department": "Ece"
}
```

System Status: All Good
©2023 MongoDB, Inc. Status Terms Privacy Atlas Blog Contact Sales

33°C Sunny Search ENG IN 14:55 27-02-2023