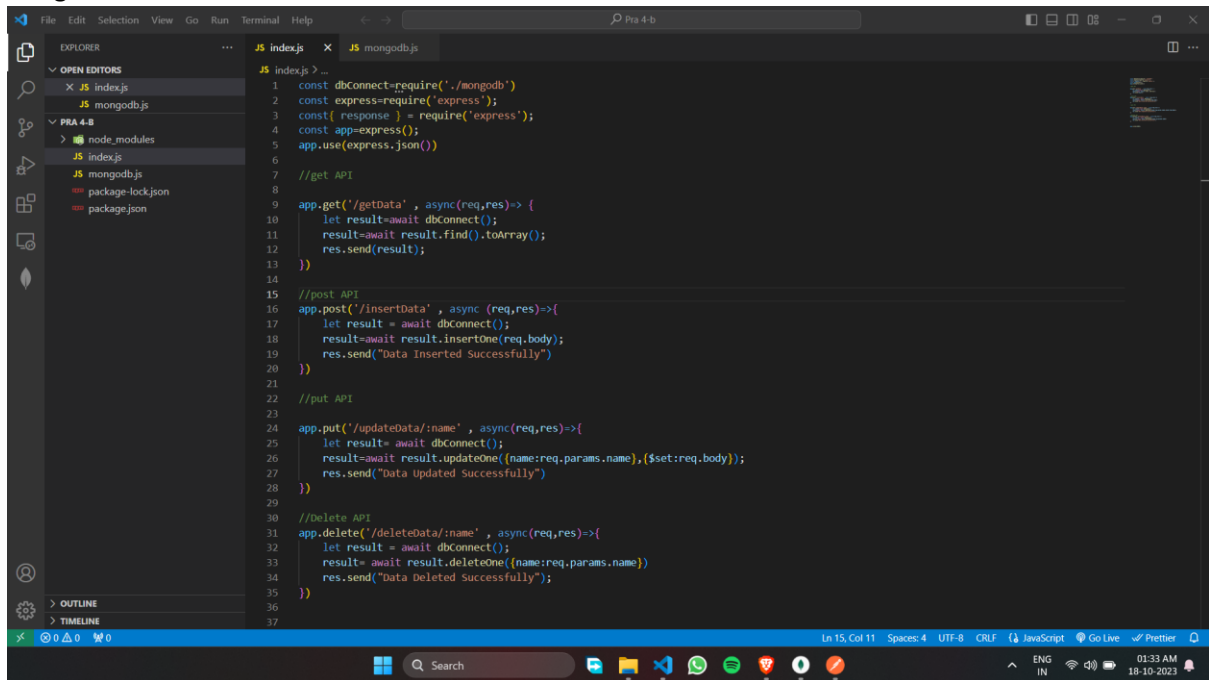


Assignment 4B



The screenshot shows a Visual Studio Code editor window with a dark theme. The Explorer sidebar on the left shows the project structure: 'PRA 4-B' containing 'node_modules', 'index.js', 'mongodb.js', 'package-lock.json', and 'package.json'. The main editor area displays the content of 'index.js', which is a JavaScript file implementing a REST API for MongoDB. The code includes imports for 'mongoose' and 'express', database connection logic, and four API endpoints: GET /getData, POST /insertData, PUT /updateData/:name, and DELETE /deleteData/:name. The status bar at the bottom indicates the cursor is at line 15, column 11, with 4 spaces and UTF-8 encoding. The system tray at the very bottom shows the date and time as 01:33 AM on 18-10-2023.

```
1  const dbConnect=require('./mongodb')
2  const express=require('express');
3  const { response } = require('express');
4  const app=express();
5  app.use(express.json())
6
7  //get API
8
9  app.get('/getData' , async(req,res)=> {
10     let result=await dbConnect();
11     result=await result.find().toArray();
12     res.send(result);
13 })
14
15 //post API
16 app.post('/insertData' , async (req,res)=>{
17     let result = await dbConnect();
18     result=await result.insertOne(req.body);
19     res.send("Data Inserted Successfully")
20 })
21
22 //put API
23
24 app.put('/updateData/:name' , async(req,res)=>{
25     let result= await dbConnect();
26     result=await result.updateOne( {name:req.params.name},{ $set:req.body});
27     res.send("Data Updated Successfully")
28 })
29
30 //Delete API
31 app.delete('/deleteData/:name' , async(req,res)=>{
32     let result = await dbConnect();
33     result= await result.deleteOne( {name:req.params.name})
34     res.send("Data Deleted Successfully");
35 })
36
37
```

This screenshot shows the VS Code editor with the `index.js` file open. The Explorer sidebar on the left shows the project structure with `index.js` and `mongodbjs` files. The main editor area displays the following JavaScript code:

```
4 const app=express();
5 app.use(express.json())
6
7 //get API
8
9 app.get('/getData', async(req,res)=> {
10   let result=await dbConnect();
11   result=await result.find().toArray();
12   res.send(result);
13 })
14
15 //post API
16 app.post('/insertData', async (req,res)=>{
17   let result = await dbConnect();
18   result=await result.insertOne(req.body);
19   res.send("Data Inserted Successfully")
20 })
21
22 //put API
23
24 app.put('/updateData/:name', async(req,res)=>{
25   let result= await dbConnect();
26   result=await result.updateOne( {name:req.params.name},{set:req.body});
27   res.send("data updated Successfully")
28 })
29
30 //Delete API
31 app.delete('/deleteData/:name', async(req,res)=>{
32   let result = await dbConnect();
33   result= await result.deleteOne({name:req.params.name})
34   res.send("Data Deleted Successfully");
35 })
36
37
38
39 app.listen(3000);
```

The status bar at the bottom indicates the file is at line 15, column 11, using UTF-8 encoding and CRLF line endings. The system tray shows the time as 01:33 AM on 18-10-2023.

This screenshot shows the VS Code editor with the `mongodbjs.js` file open. The Explorer sidebar on the left shows the project structure. The main editor area displays the following JavaScript code:

```
1 const {MongoClient}=require('mongodb');
2 const url = "mongodb://0.0.0.0"
3 const database='student';
4 const client=new MongoClient(url);
5
6 const dbConnect= async ()=>{
7   const result= await client.connect();
8   const db= await result.db(database);
9   return db.collection('profile');
10 }
11
12 module.exports=dbConnect;
```

Below the editor, the TERMINAL panel is open, showing the command `node index.js` being executed in a PowerShell window. The status bar at the bottom indicates the file is at line 12, column 26, using UTF-8 encoding and CRLF line endings. The system tray shows the time as 01:33 AM on 18-10-2023.

GET API:

The image displays two software interfaces side-by-side, demonstrating the connection between an API call and a database record.

Top Interface (Postman):

- URL:** `http://localhost:3000/getData`
- Method:** GET
- Status:** 200 OK
- Time:** 75 ms
- Size:** 326 B
- Body (JSON):**

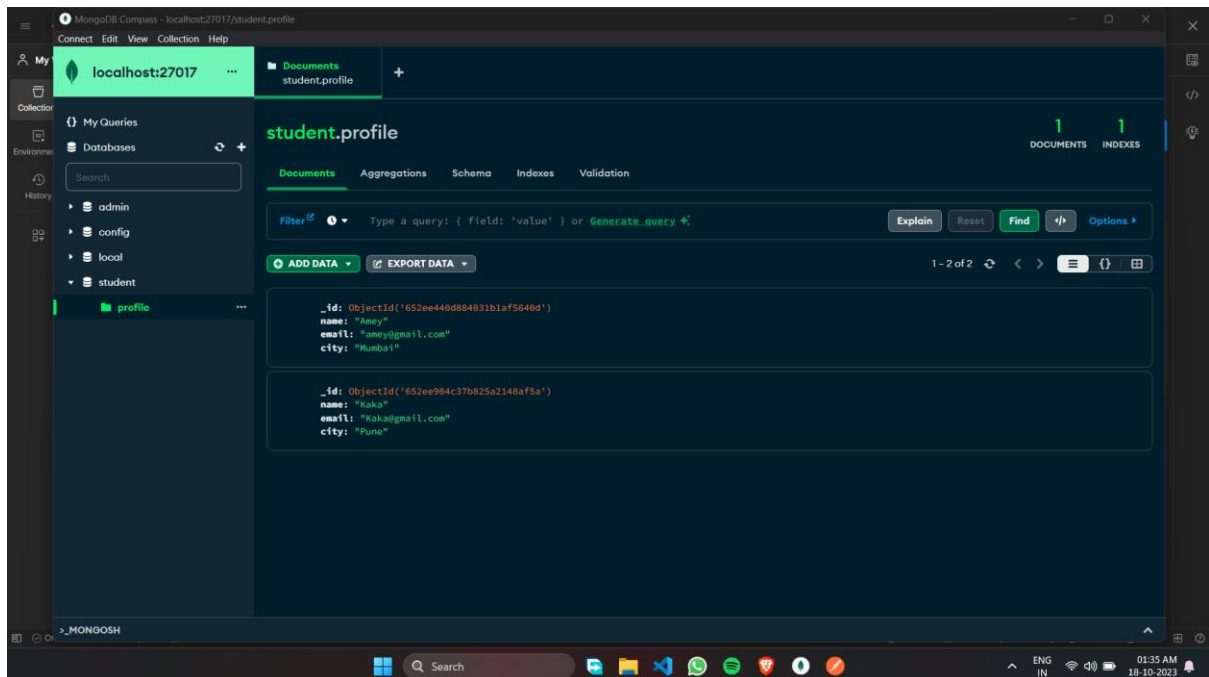
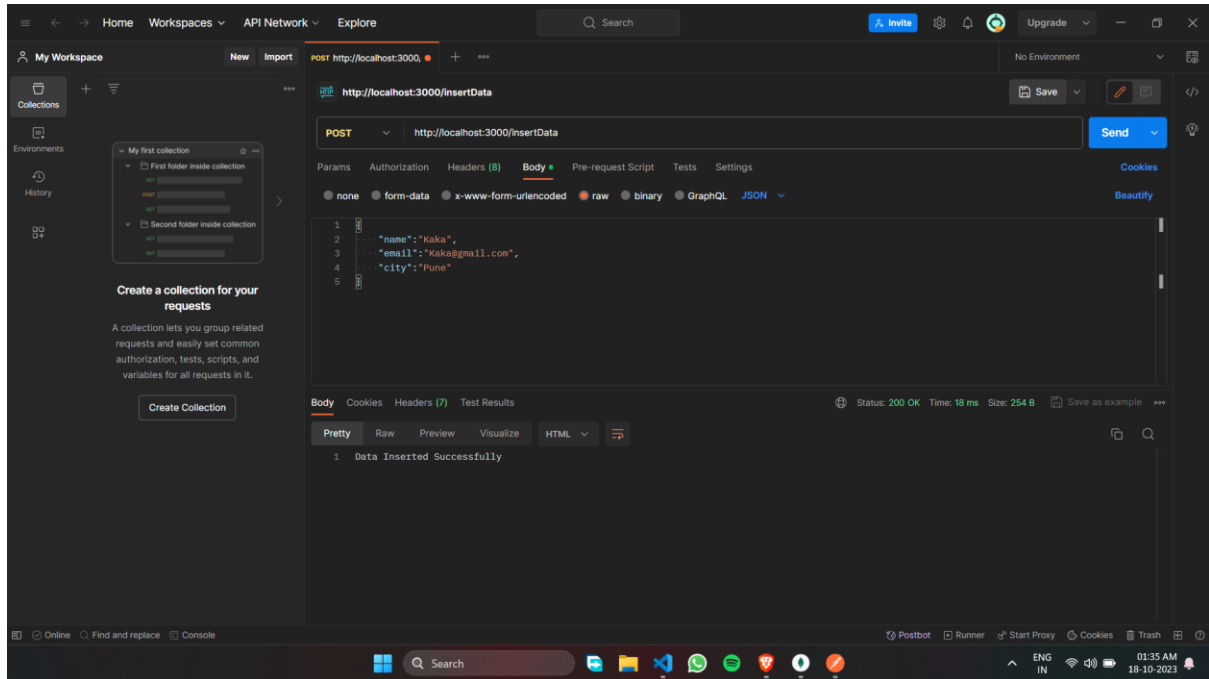
```
{  "id": "652ee44d8d884031b1af5640d",  "name": "Amy",  "email": "amey@gmail.com",  "city": "Mumbai"}
```

Bottom Interface (MongoDB Compass):

- Database:** student
- Collection:** student.profile
- Filter:** (Empty)
- Documents:** 1
- Indexes:** 1
- Document:**

```
{  "_id": ObjectId("652ee44d8d884031b1af5640d"),  "name": "Amy",  "email": "amey@gmail.com",  "city": "Mumbai"}
```

POST API:



Put API:

The image displays two screenshots from a Windows desktop environment, illustrating an API call and its data storage.

Top Screenshot (Postman): Shows a PUT request to `http://localhost:3000/updateData/Amey`. The request body is a JSON object: `{ "name": "Amey", "email": "amey@gmail.com", "city": "Mumbai" }`. The response status is `200 OK` with the message `Data Updated Successfully`.

Bottom Screenshot (MongoDB Compass): Shows the `student.profile` collection in the `student` database. The collection contains three documents:

- `{ "_id": ObjectId('652ee44d8d84031b1af5640d'), "name": "Kaka", "email": "Kaka@gmail.com", "city": "Pune" }`
- `{ "_id": ObjectId('652ee984c37b825a2148af5a'), "name": "Kaka", "email": "Kaka@gmail.com", "city": "Pune" }`
- `{ "_id": ObjectId('652ee98ec37b825a2148af5b'), "name": "Amey", "email": "amey@gmail.com", "city": "Mumbai" }`

Delete API:

The image displays two screenshots from a Windows desktop environment, illustrating the process of deleting data via an API and verifying the result in a database.

Top Screenshot (Postman):

- The interface shows a workspace named "My Workspace".
- The active request is a **DELETE** method targeting the URL `http://localhost:3000/deleteData/Kaka`.
- The "Send" button has been clicked, resulting in a successful response.
- The response body, shown in "Pretty" format, contains the text: `1 Data Deleted Successfully`.
- The status bar at the bottom indicates a status of 200 OK, a time of 18 ms, and a size of 253 B.

Bottom Screenshot (MongoDB Compass):

- The interface shows a connection to `localhost:27017`.
- The selected database is `student`, and the selected collection is `profile`.
- The "Documents" tab is active, showing a single document.
- The document content is:

```
{
  "_id": ObjectId('652ee98ec37b25a2148af5b1'),
  "name": "Aneey",
  "email": "Aneey@gmail.com",
  "city": "Mumbai"
}
```

The status bar at the bottom of the MongoDB Compass window shows `>_MONGOSH`.