LAB # 9

Backend development with Node + Express js, and testing with postman

OBJECTIVE:

To introduce students to the basic concepts of backend development.

Download and install these softwares to perform lab:

VS Code download link: https://code.visualstudio.com/download

Node js download link: https://nodejs.org/en/download

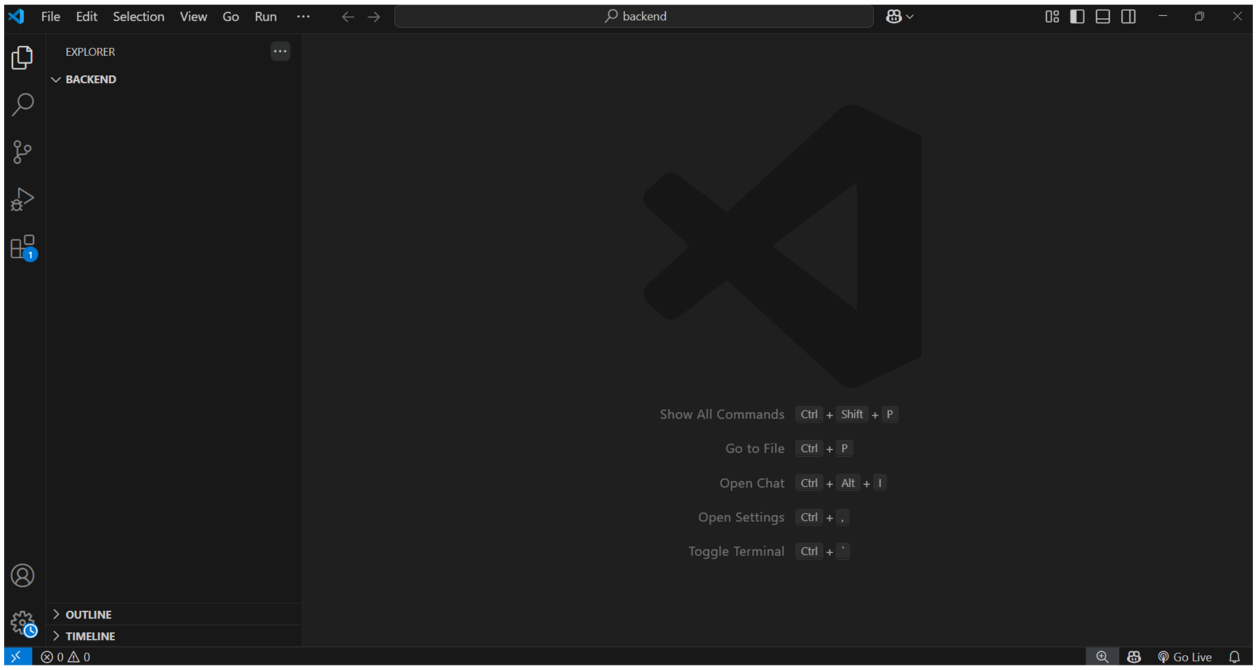
Postman download link: https://www.postman.com/downloads/

Backend development:

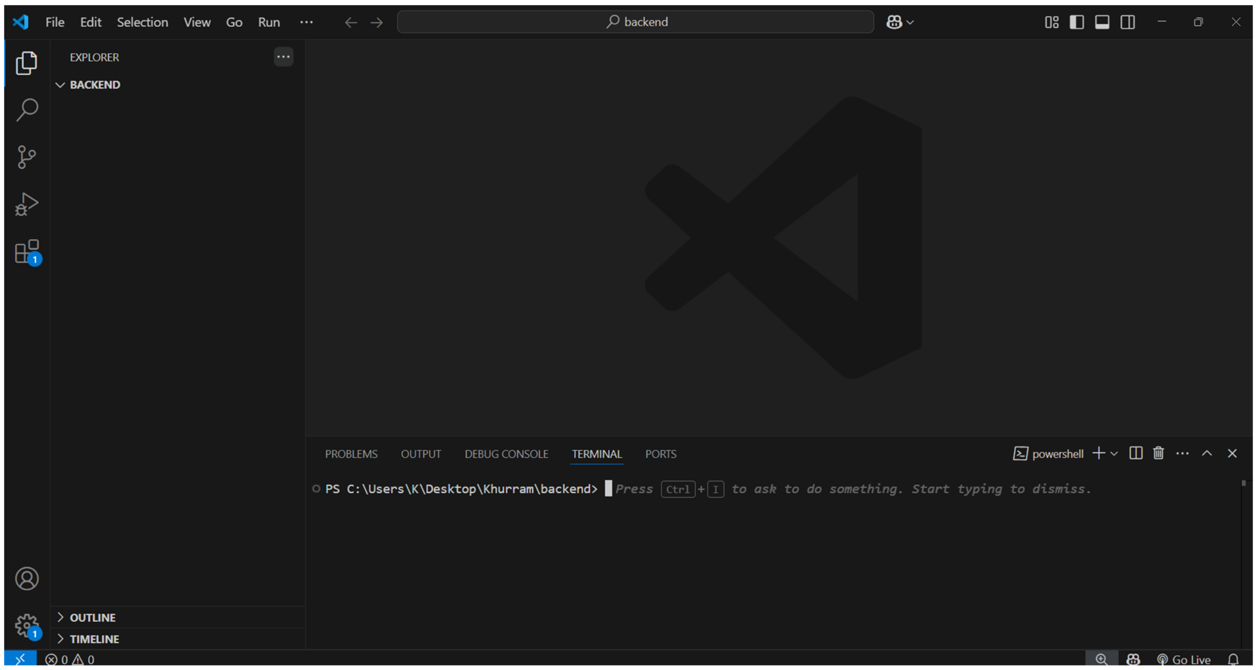
Create a folder name "backend".

Open backend folder in vs code.

You will see screen like this:



Open terminal by using **ctrl** + ` key on keyboard:

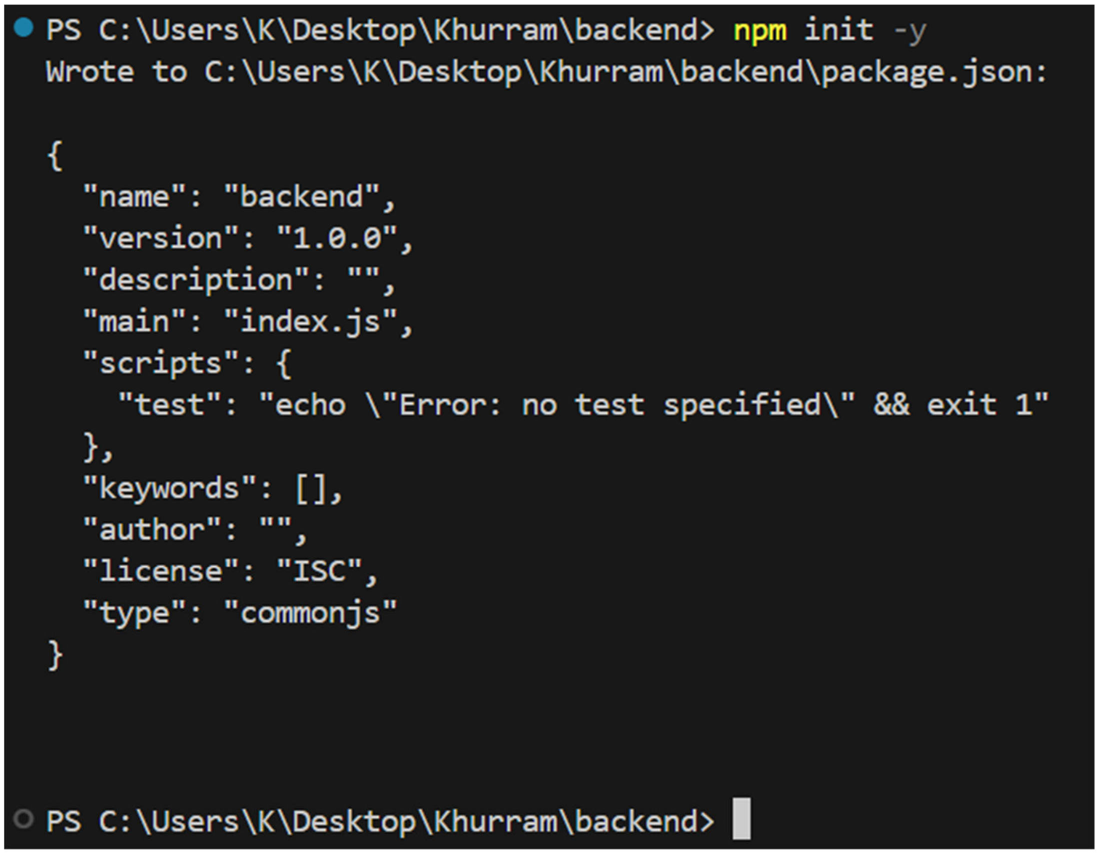

File Edit Selection View Go Run
backend
EXPLORER
BACKEND
OUTLINE
TIMELINE
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
powershell
Go Live

```
PS C:\Users\K\Desktop\Khurram\backend> Press Ctrl+I to ask to do something. Start typing to dismiss.
```

Type this command in terminal:

npm init –y

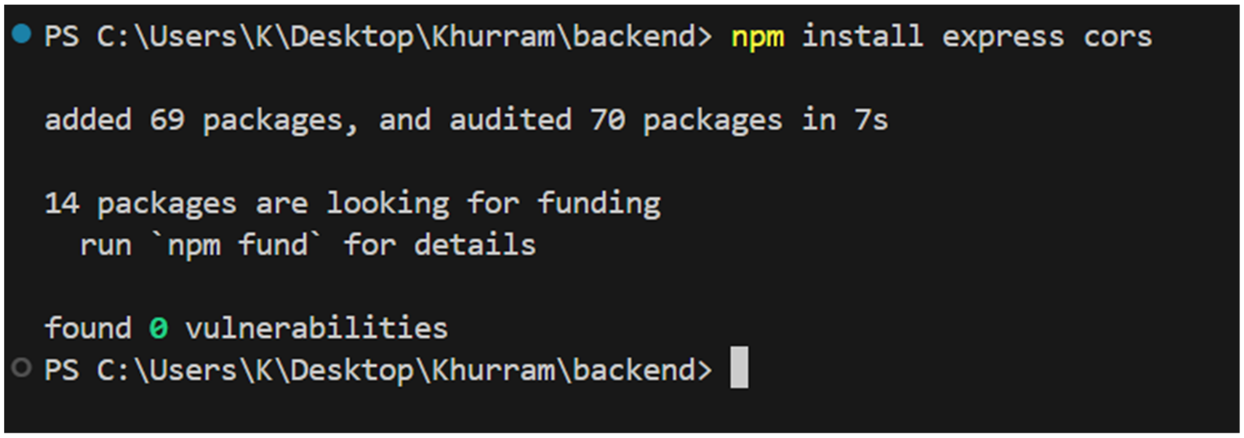
This will create a package.json file:



```
PS C:\Users\K\Desktop\Khurram\backend> npm init -y
Wrote to C:\Users\K\Desktop\Khurram\backend\package.json:

{
  "name": "backend",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "type": "commonjs"
}


PS C:\Users\K\Desktop\Khurram\backend>
```

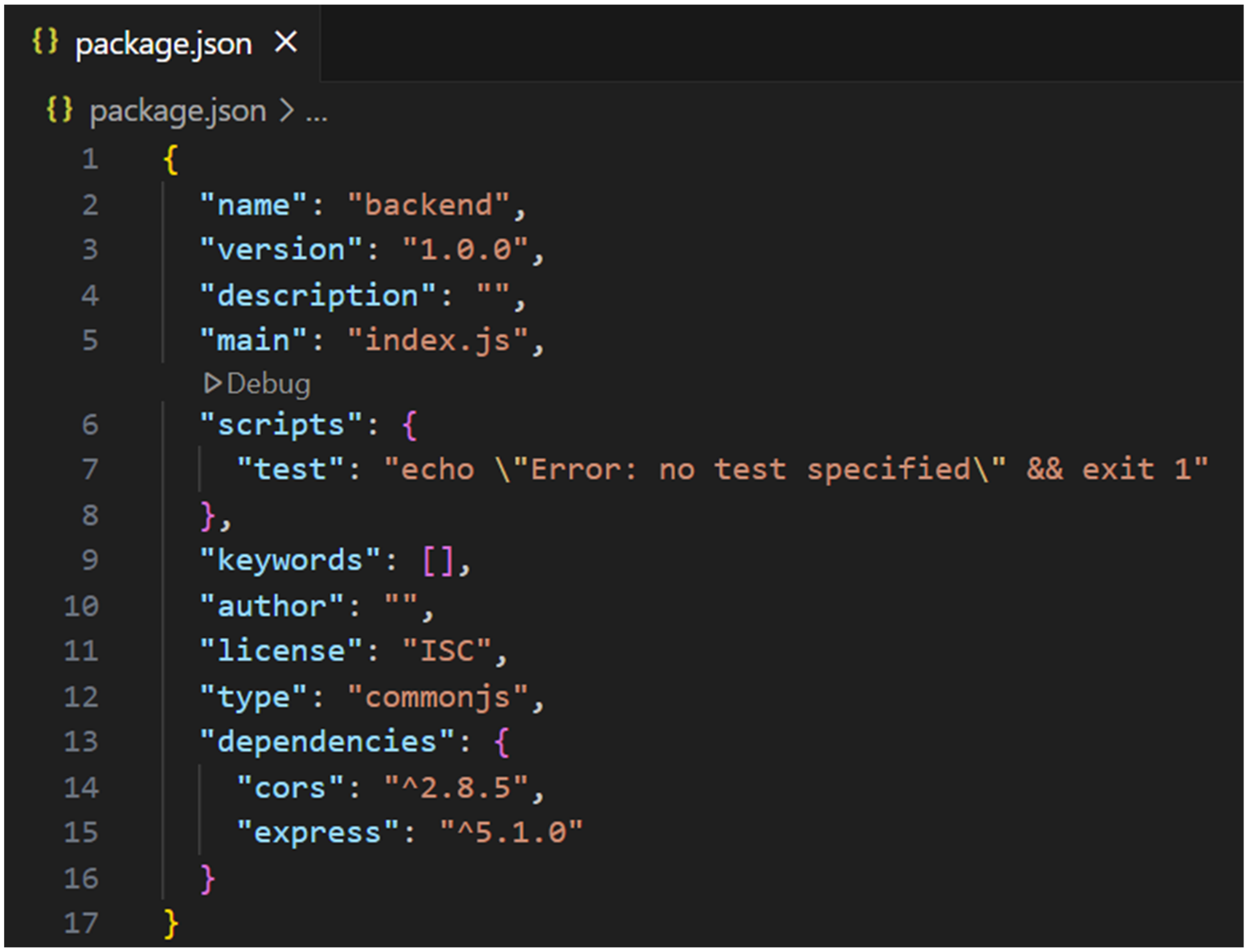
Install required packages by typing the following command:

npm install express cors



```
PS C:\Users\K\Desktop\Khurram\backend> npm install express cors

added 69 packages, and audited 70 packages in 7s

14 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\Users\K\Desktop\Khurram\backend>
```

You can check package.json file:



package.json

package.json > ...

```json
{
  "name": "backend",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  ▷Debug
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "type": "commonjs",
  "dependencies": {
    "cors": "^2.8.5",
    "express": "^5.1.0"
  }
}
```

Now create a file name server.js.

server.js:

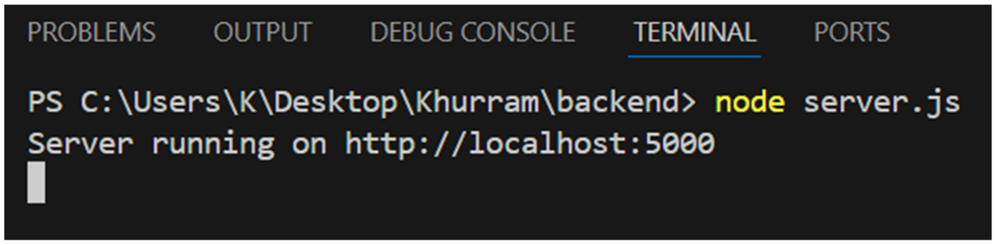
```javascript
// Backend api bana rahay hain,
// by using Express js framework
// jis mai Create, Read, Update or Delete Operations perform ho gain.
// Task ko easy rakhne k liye, only username pay kaam hoga
// Let suppose k username save kia
// Read kia etc etc

// Express aik tarah ka node ka backend framework hai.
// Express server banane k kaam aata hai.
const express = require('express');

// CORS stands for Cross-Origin Resource Sharing.
// Let suppose k ap ka backend http://localhost:5000 pay run kar raha hai.
// And frontend http://localhost:3000 pay run kar raha hai.
// To cors ap ko allow karta hai k ap do different links say.
// data ki sharing kr saktay hain.
// Is wajah say he ye Cross-Origin Resource Sharing (CORS) kehal aata hai.
const cors = require('cors');

// Express ki app bana rahay hain
const app = express();

// App k sath cors use kar rahay hain.
// Takay future mai koi problem nhi ho.
app.use(cors());

// "JSON" stands for JavaScript Object Notation.
// Json yani key:value type k data pay kaam karain gain.

// For example:
/*
{
    "username":"User_1"
}
*/

// To json use kar rahay hain.
app.use(express.json());

// Abhi koi database nhi use kia.
// To data ko array mai save karain gain.
// Or har user ki aik id hogi vo bhi.
let users = [];
let nextId = 1;

// Post request say data create kartay hain.
// Yani data save karne k liye request send ki hai.

// Postman software open kro.
// Post request select karo.
// ye link likho:
// http://localhost:5000/api/post
// Body mai ye data likho:
/*
{
    "username":"Test User"
}
*/
app.post('/api/post', (req, res) => {
    // Request say data ko data variable mai save kar liya.
    let data = req.body;

    // Aik new object create kia id vo use ki jo upper create ki thi.
    // Phir data.username mai say username object mai add kar dia.
    const newUser = {
        id: nextId++,
        username: data.username
    };

    // Ab object ko users k array mai push kar dia.
    users.push(newUser);

    // End mai response send kia.
    // Response mai do cheeze send ki:
    // Aik status code 201 jis ka matlub hota hai k successfully created.
    // Dosra newUser ka object return kia takay pata chale k kia data save kia
hai.
    res.status(201).json(newUser);
});


// Get request say data read kartay hain.
// Ye Get request sab data ko get kr k,
// JSON ki form mai ap ko reqponse day gi.
// Postman software open kro.
// Get request select karo.
// ye link likho:
// http://localhost:5000/api/get
app.get('/api/get', (req, res) => {

    // Sab data lay k 200 status code k sath return kr raha hai
    res.status(200).json(users);
});


// Let suppose k agar only one username chiye,
// to ap ki Get request kuch is tarah ki hogi,
// k ap get request k end mai user id put karain gain.

// Postman software open kro.
// Get request select karo.
// ye link likho:
// http://localhost:5000/api/get/2
// Yaad rahay link k end mai ap id koi bhi day saktay ho.
```

```
app.get('/api/get/:id', (req, res) => {

    // Jo id ap link ko do,
    // vo server ko req.params.id say milay gi,
    // phir id ko userId k variable mai save kia.
    let userId = parseInt(req.params.id);

    // Ab users mai vo id find karay ga,
    // agar username mil gia,
    // to vo user k object ko user variable mai save kare ga.
    let user = users.find(u => u.id === userId);

    // Ab vo aik user k data ko,
    // status code 200 k sath return karde ga.
    res.status(200).json(user);
});


// Put request update karne ka kaam karti hai.

// Postman software open kro.
// Put request select karo.
// ye link likho:
// http://localhost:5000/api/put/2
// Yaad rahay link k end mai ap id koi bhi day saktay ho.

// Body mai ye data ai ga:
/*
{
    "username":"User 2"
}
*/
app.put('/api/put/:id', (req, res) => {

    // Jo id ap link ko do,
    // vo server ko req.params.id say milay gi,
    // phir id ko userId k variable mai save kia.
    let userId = parseInt(req.params.id);

    // Sab data a k data variable mai save hoga.
    let data = req.body;

    // Ab users mai vo id find karay ga,
    // agar username mil gia,
    // to vo user k object ko user variable mai save kare ga.
    let user = users.find(u => u.id === userId);

    // username ko update kr raha hai.
    user.username = data.username;

    // Ab vo aik user k data ko,
    // status code 200 k sath return karde ga.
    res.status(200).json(user);
});


// Delete request delete karne ka kaam karti hai.
// Postman software open kro.
// Delete request select karo.
// ye link likho:
// http://localhost:5000/api/delete/2
// Yaad rahay link k end mai ap id koi bhi day saktay ho.
app.delete('/api/delete/:id', (req, res) => {

    // Jo id ap link ko do,
    // vo server ko req.params.id say milay gi,
    // phir id ko userId k variable mai save kia.
    let userId = parseInt(req.params.id);

    // users k array mai say us object ka index find kare ga,
    // jis k data ko delete karna hai.
    let index = users.findIndex(u => u.id === userId);

    // Data ko delete kr k aik array return kr raha hai,
    // jis array k 0 index pay vo delete ho chuka object hai.
    let deletedUserData = users.splice(index, 1);
    res.status(200).json(deletedUserData[0]);
});


// Aik constant PORT jis mai 5000 save kia or app ko start kia port 5000 pay.
const PORT = 5000;
app.listen(PORT, () => {
    // console.log() ki command print karne ka kaam karti hai
    console.log(`Server running on http://localhost:${PORT}`);
});
```

Now press **ctrl + s** to save the file.

Now run this command in terminal, to start server:

node server.js

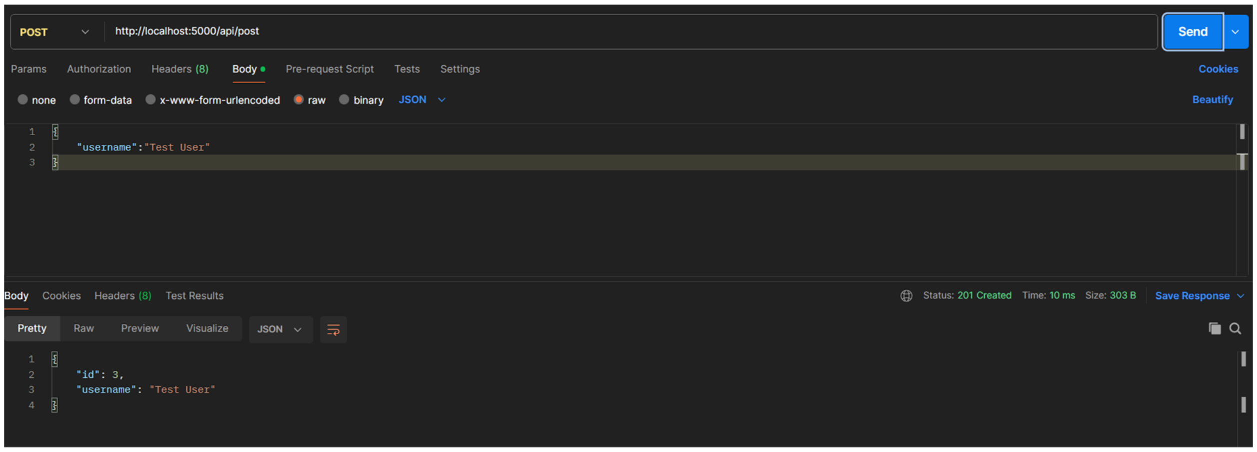


```
PS C:\Users\K\Desktop\Khurram\backend> node server.js
Server running on http://localhost:5000
```

Backend Testing:

Now for testing backend, we use Postman software:

Post request:



POST http://localhost:5000/api/post

```
{
    "username":"Test User"
}
```

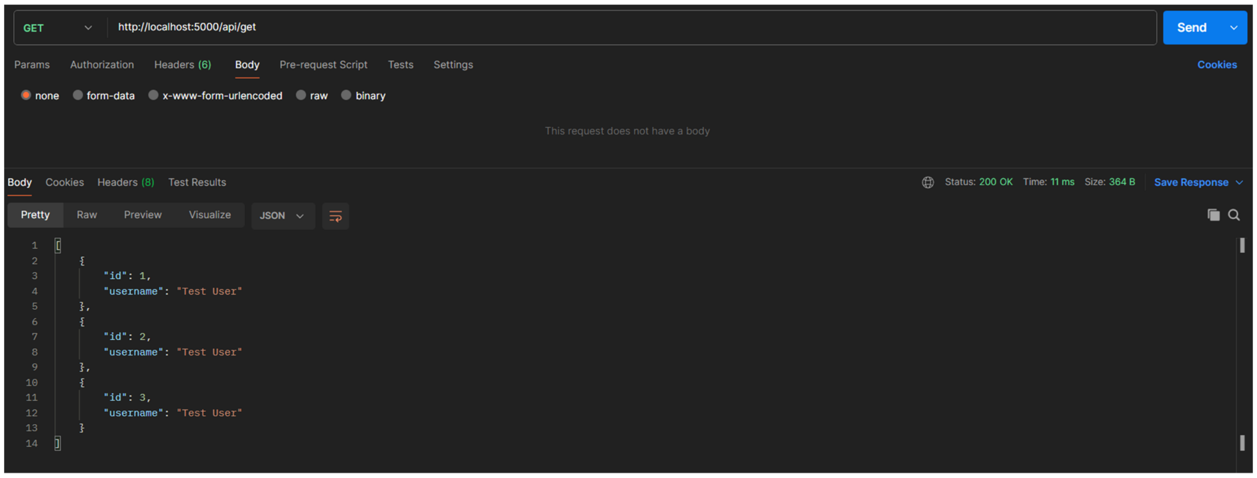
Status: 201 Created Time: 10 ms Size: 303 B

JSON

```
{
    "id": 3,
    "username": "Test User"
}
```

Get request:

Get data of all users:



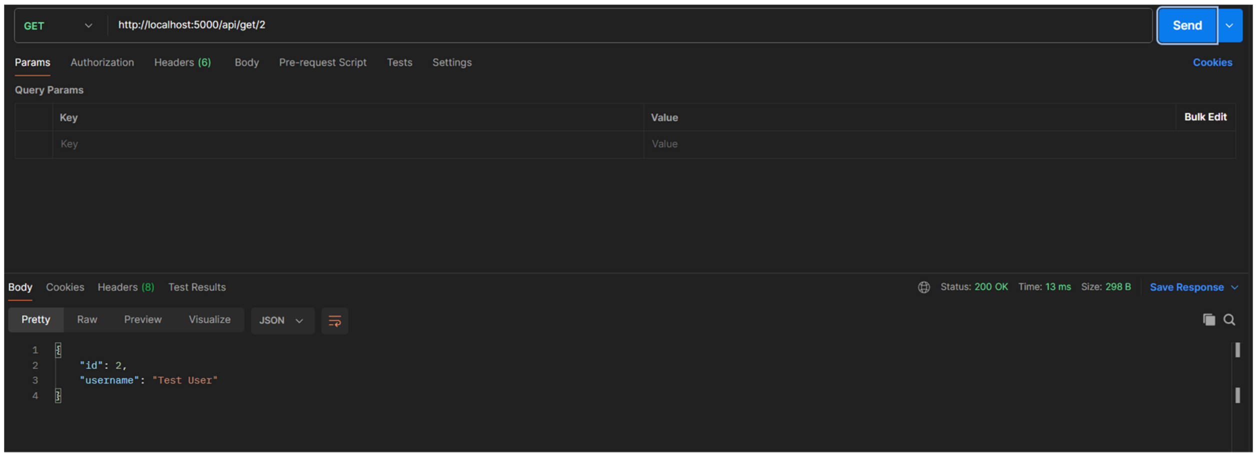
GET http://localhost:5000/api/get

Body: none — This request does not have a body

Status: 200 OK Time: 11 ms Size: 364 B

```
[
    {
        "id": 1,
        "username": "Test User"
    },
    {
        "id": 2,
        "username": "Test User"
    },
    {
        "id": 3,
        "username": "Test User"
    }
]
```

Get data of one user:

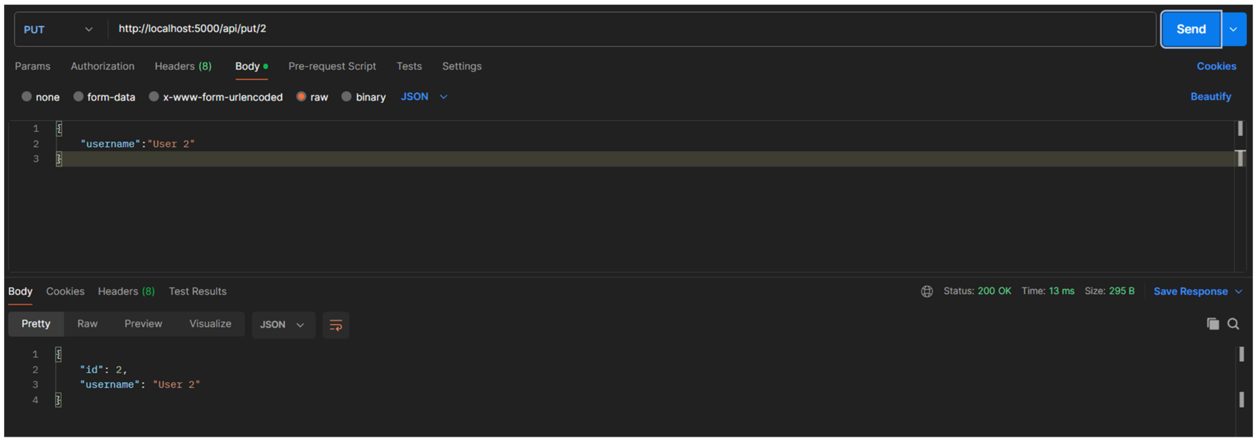


GET http://localhost:5000/api/get/2

Status: 200 OK Time: 13 ms Size: 298 B

```
{
    "id": 2,
    "username": "Test User"
}
```

Put request:



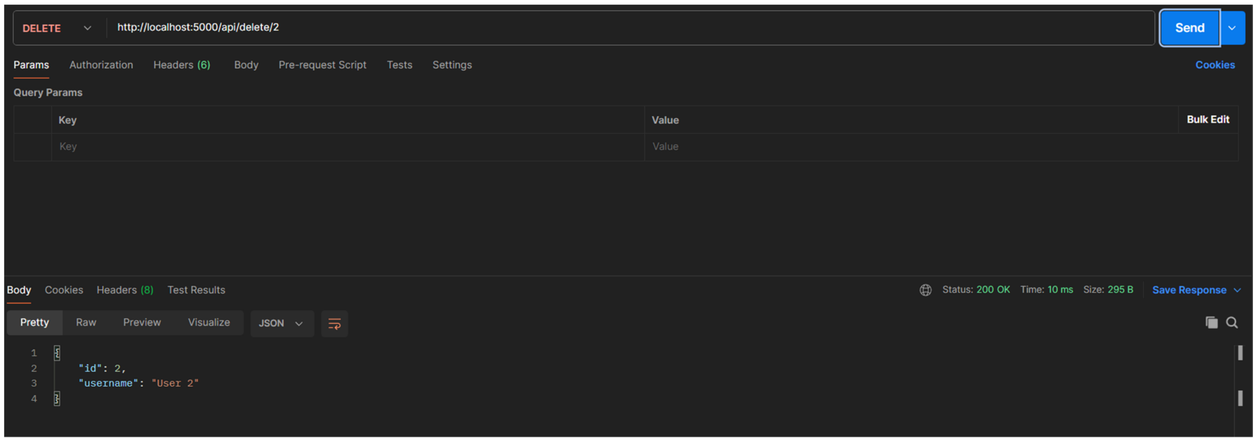
PUT http://localhost:5000/api/put/2

```
{
    "username":"User 2"
}
```

Status: 200 OK Time: 13 ms Size: 295 B

```
{
    "id": 2,
    "username": "User 2"
}
```

Delete request:



http://localhost:5000/api/delete/2

Status: 200 OK Time: 10 ms Size: 295 B

```
{
    "id": 2,
    "username": "User 2"
}
```

<u>Lab Task:</u>

Create a calculator by using Express js and include these methods: post, get, put and delete.