1 What is Response Object in Express?

The Response Object (res) in Express is used to send a response from the server back to the client.
It represents the HTTP response that an Express app sends when it receives an HTTP request.

2 What are the different types/methods available in the Response Object?

Some commonly used response methods are:
- res.send() → Sends response data (text, JSON, HTML, etc.)
- res.json() → Sends JSON response
- res.status() → Sets HTTP status code
- res.redirect() → Redirects the client to another route
- res.render() → Renders a template
- res.sendFile() → Sends a file
- res.end() → Ends the response process

3 How is the res object made available inside route handlers? Do we import it manually?

The res object is automatically provided by Express as a parameter inside route handler functions.
Example:

```
app.get("/", (req, res) => {
   res.send("Hello");
});
```

We do NOT import it manually. Express internally creates and passes it.

4 What kind of data can be sent using the response object?

We can send:
- Plain Text
- JSON Data
- HTML Content
- Files (PDF, images, etc.)
- Status messages

5 What is the use of status codes in HTTP responses?

Status codes indicate whether a request was successful or if an error occurred.
They help the client understand the result of the request.

6 What is a status code? Explain with examples.

A status code is a 3-digit number sent by the server in the HTTP response.

Examples:
200 → OK (Request successful)
201 → Created (Resource created successfully)
400 → Bad Request (Client error)
404 → Not Found (Resource not found)
500 → Internal Server Error (Server-side error)

7 Create an API that runs on port 5000 and sends a response message “Success Done” when accessed.

Example Code:

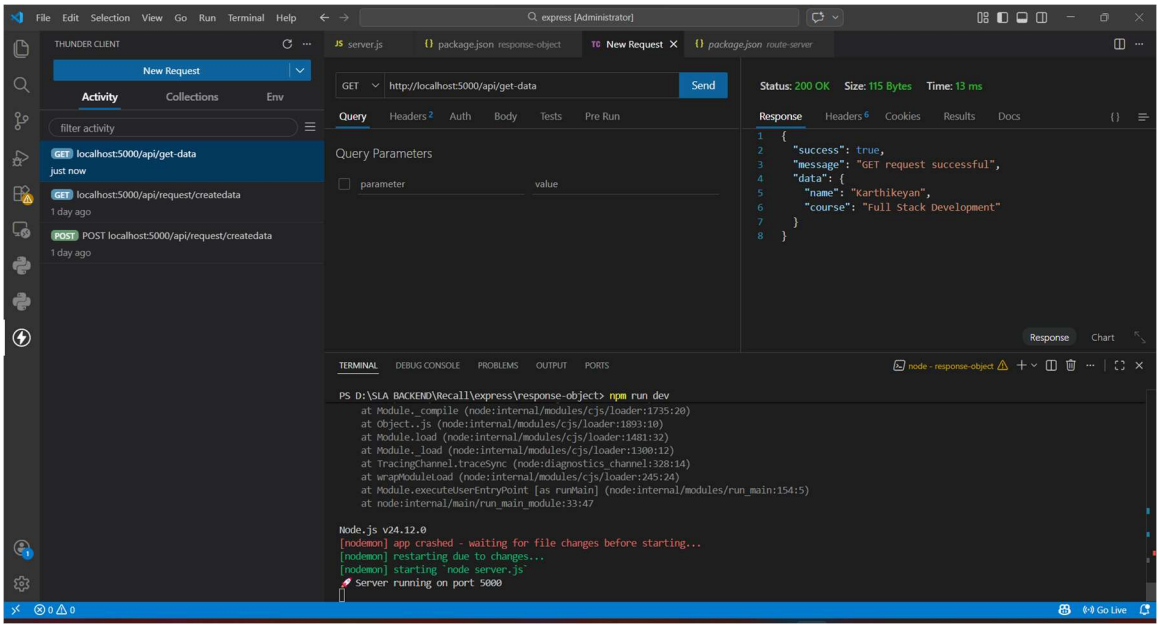
```
const express = require("express");
const app = express();

app.get("/", (req, res) => {
  res.send("Success Done");
});

app.listen(5000, () => {
  console.log("Server running on port 5000");
});
```

8 Create an API that sends status code 201 and returns a message to the client.

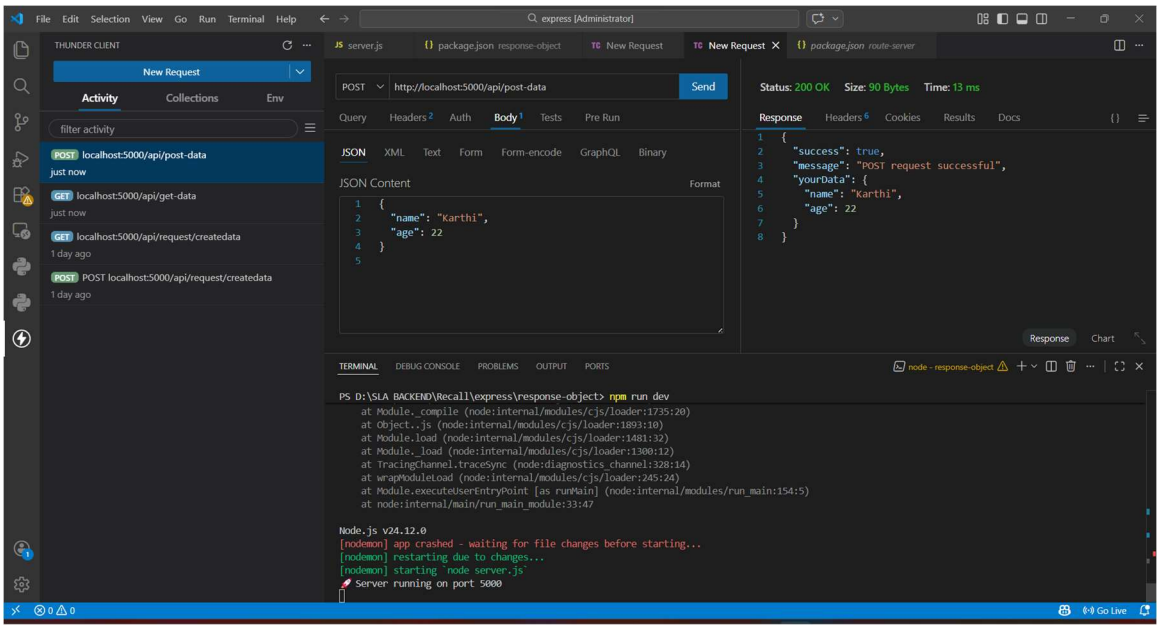
Example Code:

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

app.get("/create", (req, res) => {
  res.status(201).json({ message: "Resource Created Successfully" });
});

app.listen(5000, () => {
  console.log("Server running on port 5000");
});
```

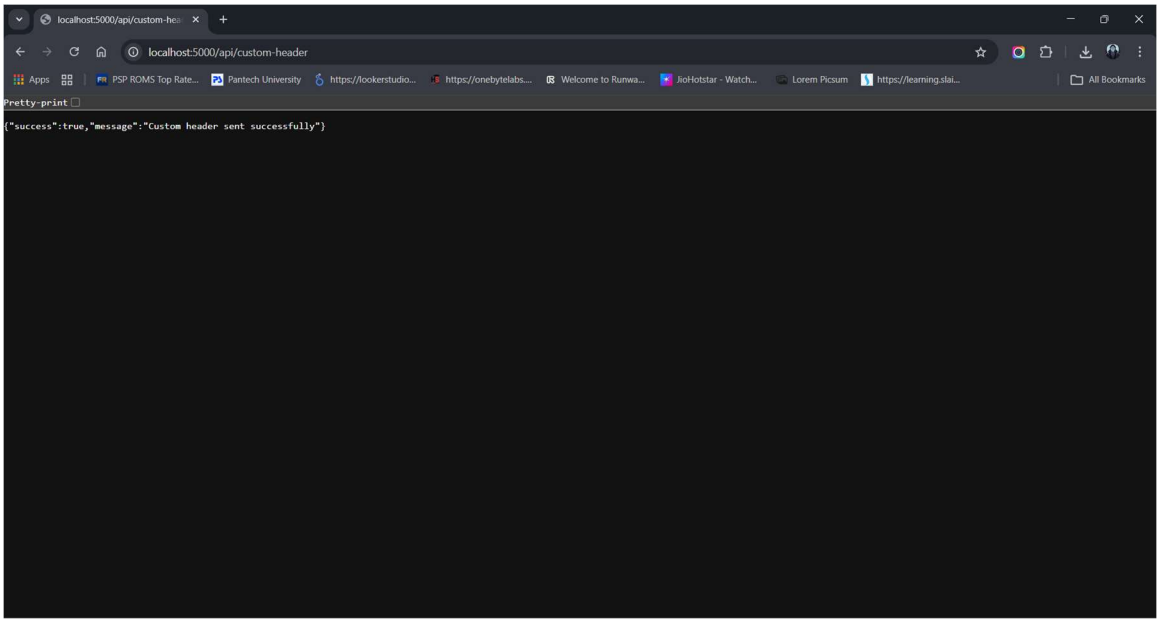
9 Create a GET route that returns JSON data with status code 200.



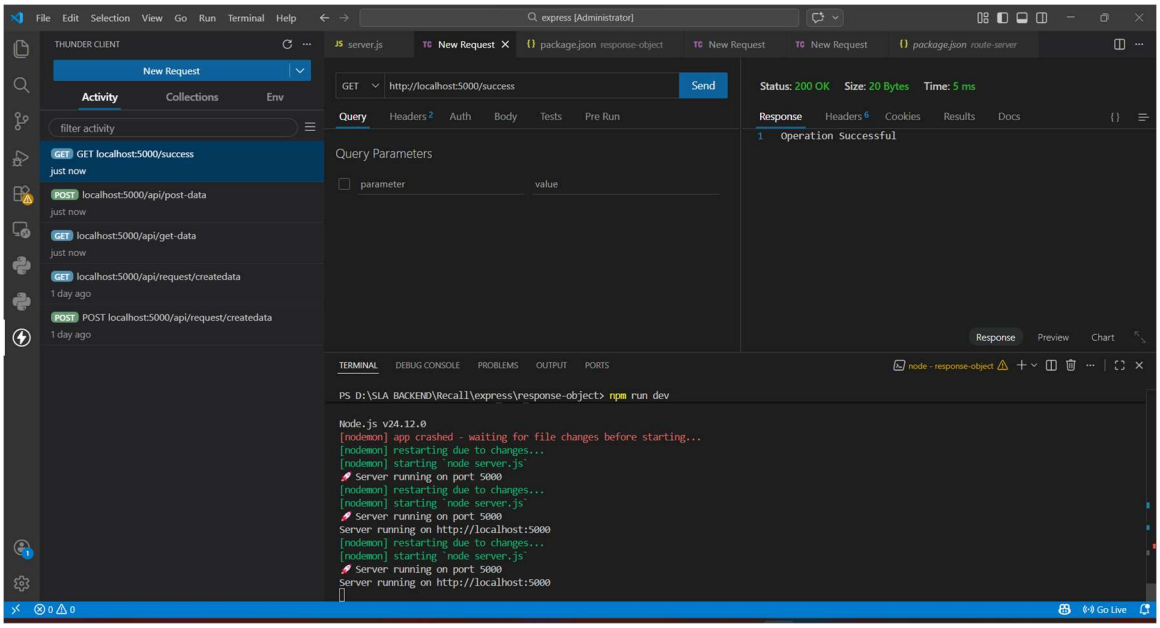
10 Create a POST route that reads data from request body and sends it back in the response.



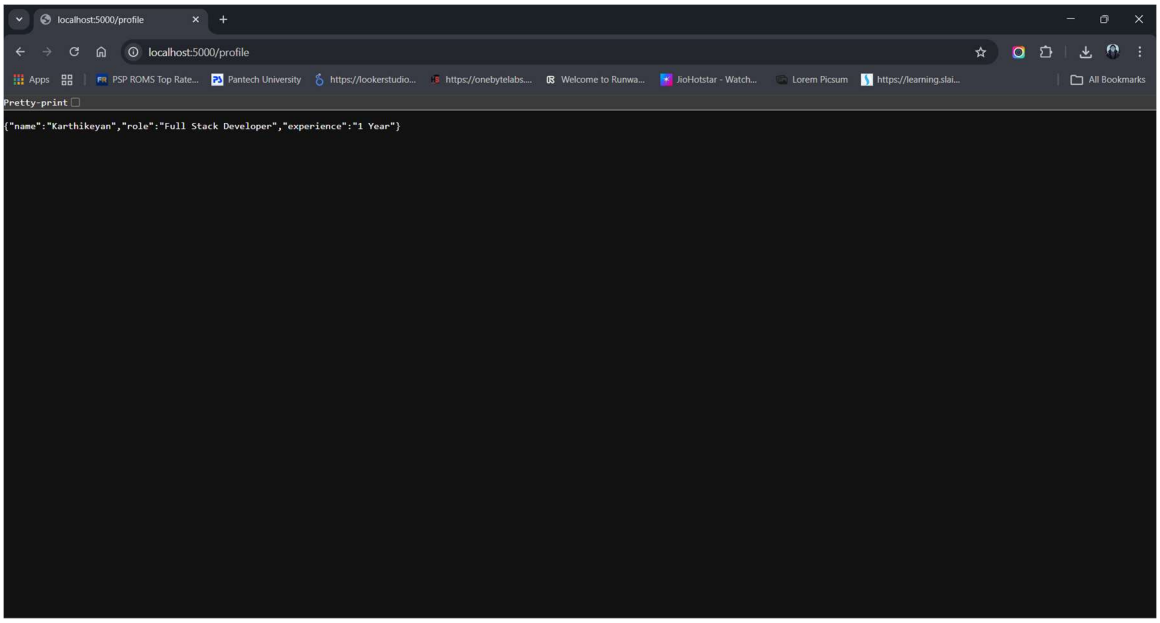
11 Create a route that sends a custom header along with a success response message.



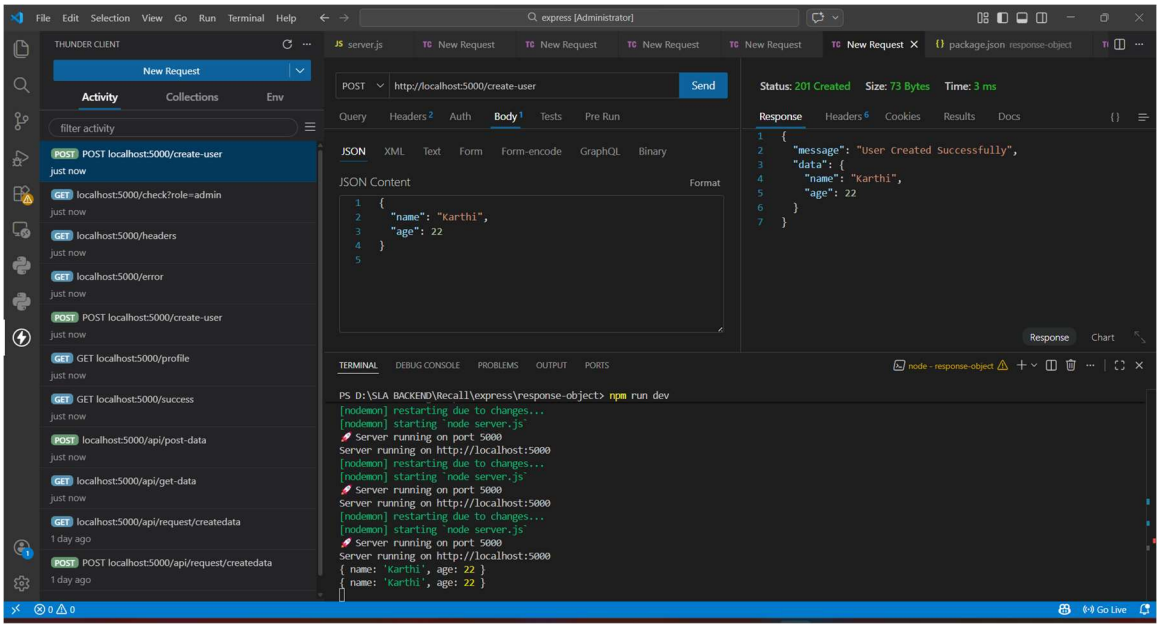
1 Basic Success Response Task.



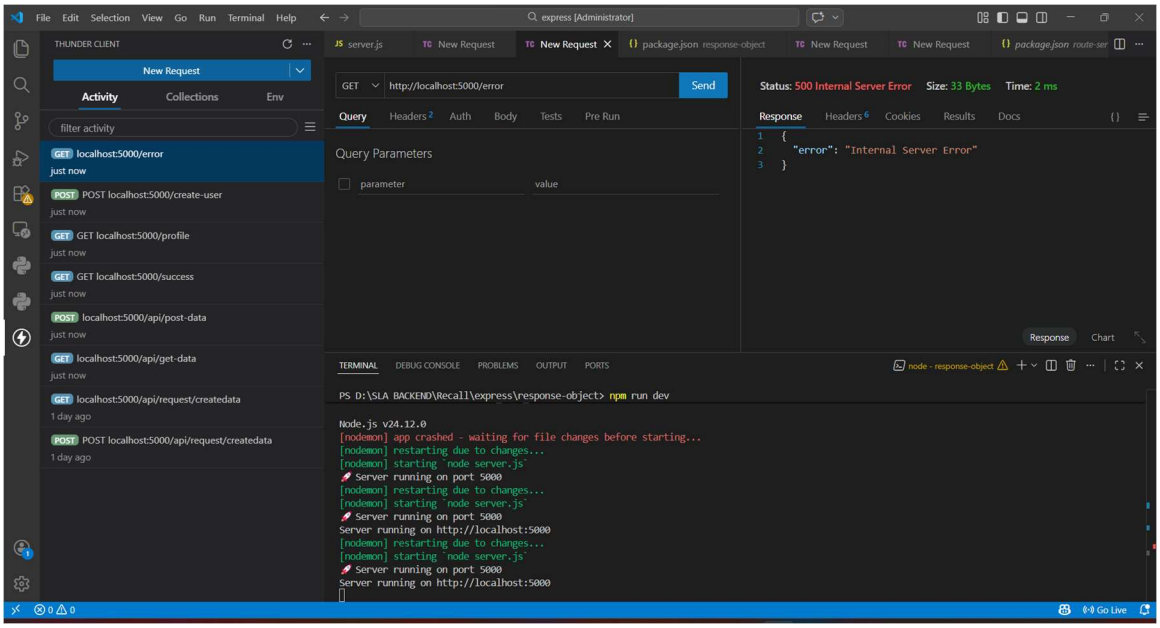
2 JSON Response Task.



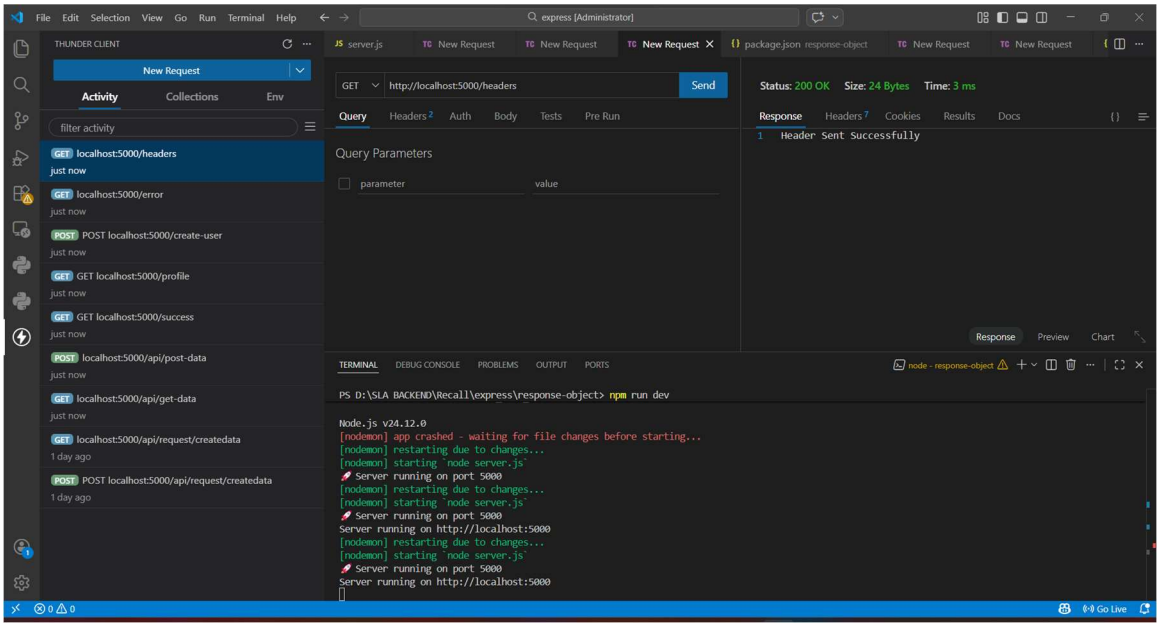
3 Create Route with 201 Status.



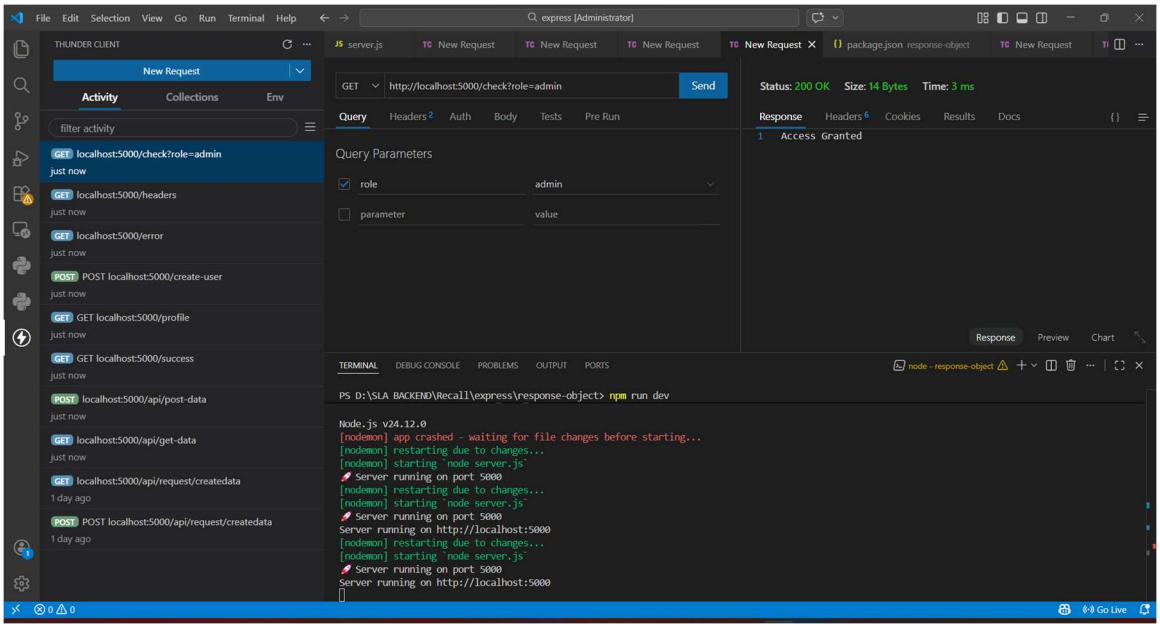
4 Error Handling Task.



5 Custom Header Task.



6 Conditional Status Code Task.


File Edit Selection View Go Run Terminal Help
express [Administrator]
THUNDER CLIENT
New Request
Activity
Collections
Env
filter activity
GET localhost:5000/check?role=admin
just now
GET localhost:5000/headers
just now
GET localhost:5000/error
just now
POST POST localhost:5000/create-user
just now
GET GET localhost:5000/profile
just now
GET GET localhost:5000/success
just now
POST localhost:5000/api/post-data
just now
GET localhost:5000/api/get-data
just now
GET localhost:5000/api/request/createdata
1 day ago
POST POST localhost:5000/api/request/createdata
1 day ago
server.js
New Request
New Request
New Request
New Request
package.json response-object
New Request
GET
http://localhost:5000/check?role=admin
Send
Query
Headers 2
Auth
Body
Tests
Pre Run
Query Parameters
role
admin
parameter
value
Status: 200 OK
Size: 14 Bytes
Time: 3 ms
Response
Headers 6
Cookies
Results
Docs
1 Access Granted
Response
Preview
Chart
TERMINAL
DEBUG CONSOLE
PROBLEMS
OUTPUT
PORTS
node - response-object
PS D:\SLA BACKEND\Recall\express\response-object> npm run dev
Node.js v24.12.0
[nodemon] app crashed - waiting for file changes before starting...
[nodemon] restarting due to changes...
[nodemon] starting `node server.js`
Server running on port 5000
[nodemon] restarting due to changes...
[nodemon] starting `node server.js`
Server running on port 5000
Server running on http://localhost:5000
[nodemon] restarting due to changes...
[nodemon] starting `node server.js`
Server running on port 5000
Server running on http://localhost:5000
0 0
Go Live