

1. Build a basic HTTP server that can handle different routes and HTTP methods (GET, POST, PUT, DELETE).

```
const http = require('http');

// Helper function to handle sending responses function
sendResponse(res, statusCode, message) {
  res.writeHead(statusCode, { 'Content-Type': 'application/json' });
  res.end(JSON.stringify({ message }));
}

// Create HTTP server const server =
http.createServer((req, res) => {  const {
  method, url } = req;

  // Route: /  if (url === '/') {    if (method === 'GET') {
sendResponse(res, 200, 'Hello, World! This is the root route. ');
    } else {      sendResponse(res, 405, `Method ${method} not allowed on this
route.`);
    }
  }

  // Route: /items
  else if (url === '/items') {    if (method === 'GET') {
sendResponse(res, 200, 'Fetching all items... ');
    } else if (method === 'POST') {
sendResponse(res, 201, 'Creating a new item... ');
```

```
    } else {    sendResponse(res, 405, `Method ${method} not allowed on this
route.`);
```

```
    }
```

```
  }
```

```
  // Route: /items/:id (using simple regex for demonstration)
```

```
  else if (url.match(/^\/items\/\d+$/)) {    const id =
```

```
url.split('/')[2];
```

```
    if (method === 'GET') {    sendResponse(res, 200,
`Fetching item with ID: ${id}`);
```

```
    } else if (method === 'PUT') {    sendResponse(res, 200,
`Updating item with ID: ${id}`);
```

```
    } else if (method === 'DELETE') {    sendResponse(res,
200, `Deleting item with ID: ${id}`);
```

```
    } else {    sendResponse(res, 405, `Method ${method} not allowed on this
route.`);
```

```
    }
```

```
  }
```

```
  // Handle 404 - Not Found  else {
sendResponse(res, 404, 'Route not found.');
```

```
  }
```

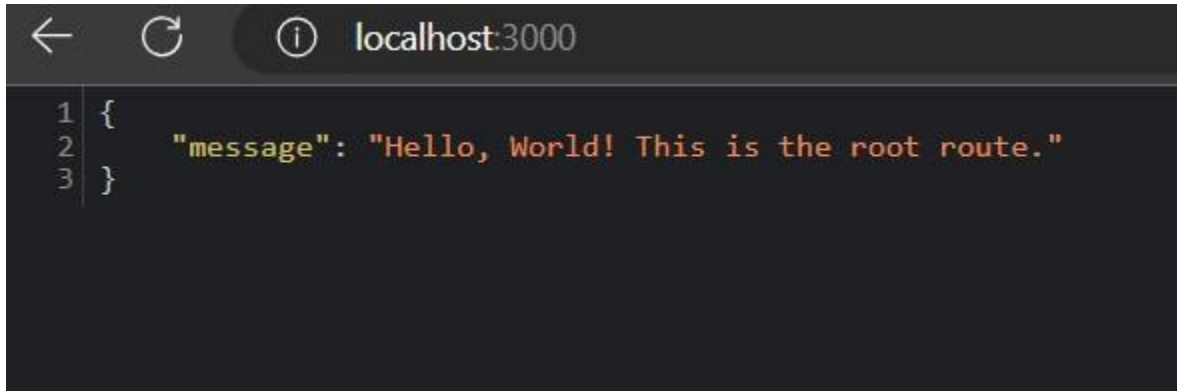
```
});
```

```
// Server listens on port 3000 server.listen(3000, () => {
```

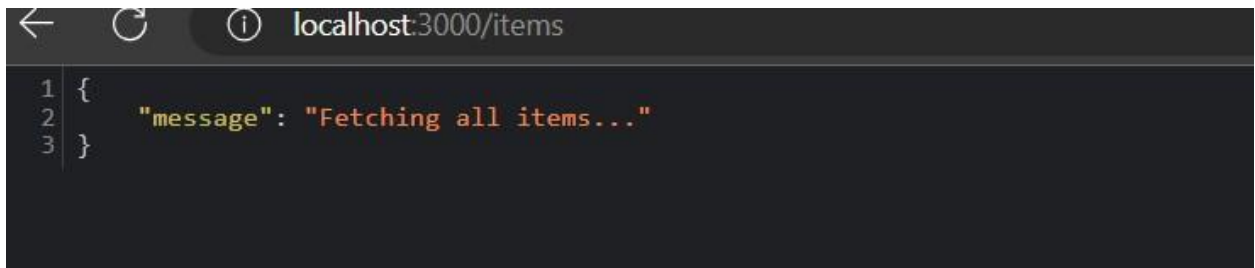
```
console.log('Server is running on http://localhost:3000');
```

```
});
```

Output:

A screenshot of a web browser window. The address bar shows 'localhost:3000'. The page content is a JSON object: { "message": "Hello, World! This is the root route." }. The JSON is displayed with syntax highlighting: curly braces are blue, and the string value is orange. Line numbers 1, 2, and 3 are visible on the left side of the code block.

```
1 {  
2   "message": "Hello, World! This is the root route."  
3 }
```

A screenshot of a web browser window. The address bar shows 'localhost:3000/items'. The page content is a JSON object: { "message": "Fetching all items..." }. The JSON is displayed with syntax highlighting: curly braces are blue, and the string value is orange. Line numbers 1, 2, and 3 are visible on the left side of the code block.

```
1 {  
2   "message": "Fetching all items..."  
3 }
```

2. Create a server that allows users to upload files and save them to the server's filesystem.

```
const http = require('http'); const  
formidable = require('formidable'); const  
fs = require('fs'); const path =  
require('path');  
const hostname = '127.0.0.1'; const port =  
3000; const server = http.createServer((req,  
res) => {   if (req.method.toLowerCase() ===  
'get') {
```

```

    res.writeHead(200, { 'Content-Type': 'text/html' });
    res.end(`
      <form action="/upload" enctype="multipart/form-data" method="post">
        <input type="file" name="fileupload"><br><br>
        <input type="submit" value="Upload File">
      </form>
    `);
  } else if (req.url === '/upload' && req.method.toLowerCase() === 'post') {

    const form = new formidable.IncomingForm();

    form.parse(req, (err, fields, files) => {
      if (err) {
        console.error('Error parsing the
form:', err);
        res.writeHead(500, { 'Content-Type':
'text/plain' });
        res.end('An error occurred during the
file upload.');
        return;
      }

      const uploadedFile = files.fileupload;
      const oldPath = uploadedFile.filepath;
      const newPath =

      path.join(__dirname, 'uploads', uploadedFile.originalFilename);

      fs.rename(oldPath, newPath, (err) => {

```

```

        if (err) {
            console.error('Error saving the
file:', err);
            res.writeHead(500, { 'Content-Type':
'text/plain' });
            res.end('File could not be saved.');
```

```

            return;
        }

        res.writeHead(200, { 'Content-Type': 'text/plain' });
        res.end('File uploaded and saved successfully!');
    });
});
} else {
```

```

        res.writeHead(404, { 'Content-Type': 'text/plain' });
        res.end('Route not found.');
```

```

    }
});

const uploadDir = path.join(__dirname, 'uploads');
if (!fs.existsSync(uploadDir)) {
    fs.mkdirSync(uploadDir);
}
server.listen(port, hostname, () => {
    console.log(`Server
running at http://${hostname}:${port}/`);
});
```

Output:

