Backend Bootcamp - Session 3: Node.js + Express Overview

What is Node.js?

- JavaScript runtime environment for running JS on the server
- Built on Chrome's V8 engine (fast execution)
- Created by Ryan Dahl in 2009
- Lets you build servers, APIs, and tools using JavaScript

Why Node.js is Popular

- JavaScript everywhere: frontend + backend
- Non-blocking I/O: handles thousands of requests efficiently
- Large ecosystem via npm
- Perfect for real-time apps, APIs, microservices

How Node.js Works Internally

- Single-threaded but asynchronous using an Event Loop
- Uses 'libuv' library for handling I/O
- Delegates heavy tasks to internal thread pool
- Fast execution using V8 engine and Just-In-Time compilation

Creating a Basic HTTP Server

```js
const http = require('http');
const server = http.createServer((req, res) => {
  res.writeHead(200, { 'Content-Type': 'text/plain' });
  res.end('Welcome to Node.js!');
});
server.listen(3000);
```

Understanding the Server Code

- `http.createServer()`: creates a web server
- `(req, res) => {}`: handles each request
- `res.writeHead()`: sets status and headers
- `res.end()`: ends the response and sends it to the client

Routing Basics (No Framework)

```js
if (req.url === '/about') {
  res.end('About Page');
} else if (req.url === '/contact') {
  res.end('Contact Page');
} else {
  res.end('Home Page');
}
```

Serving JSON Response

```js
if (req.url === '/api') {
  res.writeHead(200, { 'Content-Type': 'application/json' });
  res.end(JSON.stringify({ name: 'Ali', age: 25 }));
}
```

Handling Query Params (Manually)

```js
const url = require('url');
const parsedUrl = url.parse(req.url, true);
console.log(parsedUrl.query);
```

Reading a File Asynchronously

```js
const fs = require('fs');
fs.readFile('data.txt', 'utf8', (err, data) => {
  if (err) throw err;
  console.log(data);
});
```

Creating a Project with Yarn

- `mkdir nodejs-starter && cd nodejs-starter`

- `yarn init -y`
- Create `index.js` with your server code
- `yarn add --dev nodemon`
- Add scripts in package.json:
  - `start`: node index.js
  - `dev`: nodemon index.js

What is Express.js?

- Web application framework for Node.js
- Built on top of Node's `http` module
- Handles routing, middleware, requests/responses with ease
- Most popular Node.js framework for building APIs and web servers

Why Use Express?

- Simplifies server code
- Easy routing (GET, POST, etc.)
- Supports middleware and modular structure
- Works well with databases and frontends
- Massive ecosystem and documentation

Installing Express with Yarn

```bash
yarn add express
```

Creating a Basic Express Server

```js
const express = require('express');
const app = express();

app.get('/', (req, res) => {
  res.send('Welcome to Express!');
});

app.listen(3000, () => {
  console.log('Server running on http://localhost:3000');
});
```

```

### Express Routing Example

```js
app.get('/about', (req, res) => {
  res.send('About Us');
});

app.get('/api/user', (req, res) => {
  res.json({ name: 'Ali', age: 25 });
});
```

Express with Query Parameters

```js
app.get('/greet', (req, res) => {
  const name = req.query.name;
  res.send(`Hello, ${name}`);
});
// Try: http://localhost:3000/greet?name=Asad
```

Adding Middleware

```js
app.use(express.json()); // to parse JSON body

app.post('/api/data', (req, res) => {
  console.log(req.body);
  res.send('Data received');
});
```

Quiz: Express Basics (5 Questions)

1. What does `express()` return?
2. What method handles GET requests?
3. How do you send JSON in a response?
4. Which middleware parses JSON?
```

5. How do you listen on port 4000?