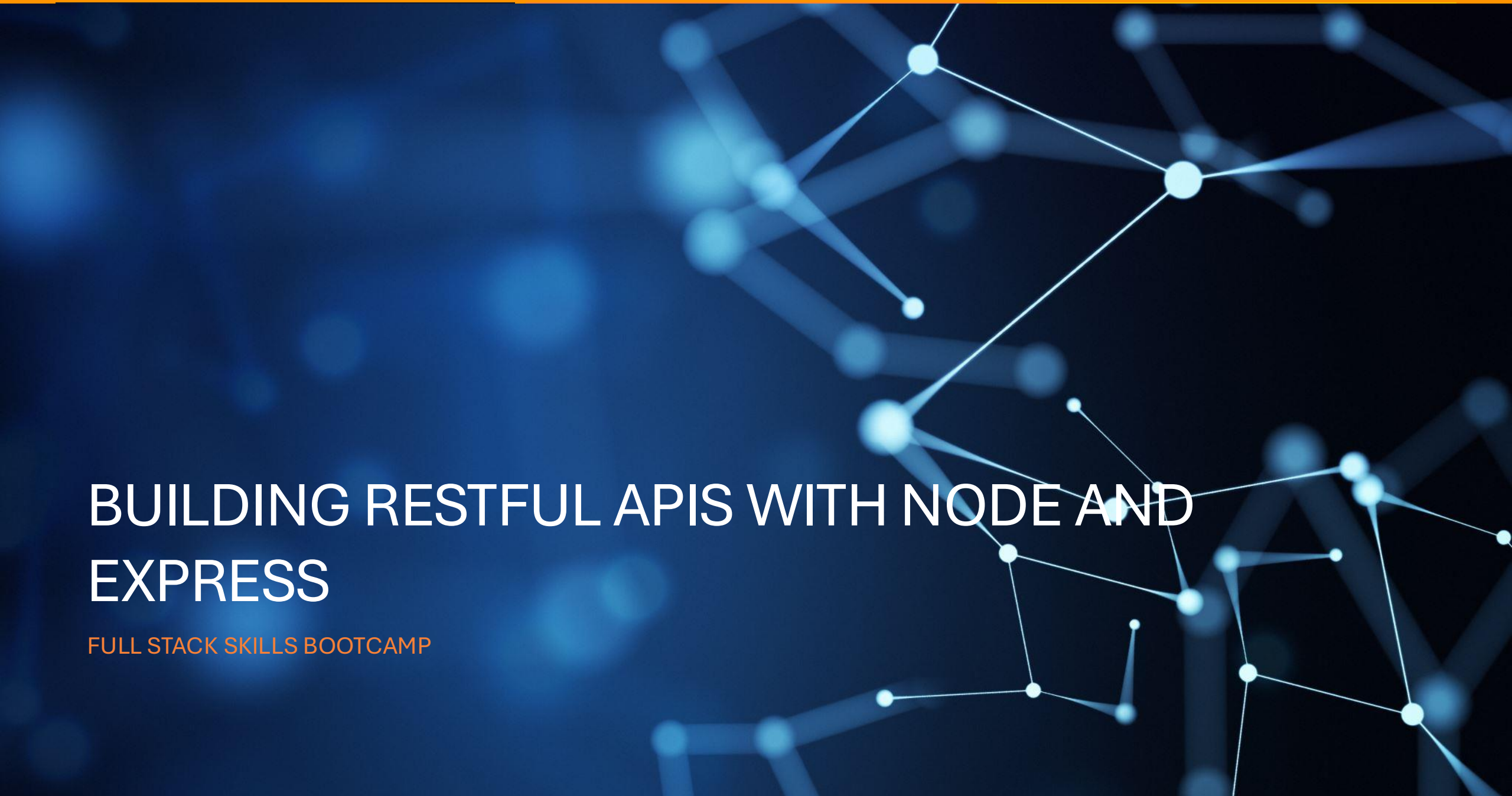

BUILDING RESTFUL APIS WITH NODE AND EXPRESS

FULL STACK SKILLS BOOTCAMP

BUILDING RESTFUL APIS WITH NODE AND EXPRESS

- **Lesson Overview:**
- In this lesson, we will be introduced to:

1. What is RESTful and CRUD
2. Setting up Express and Express routes
3. Working with JSON
4. .gitignore
5. Deploying a Node/Express application

WHAT IS RESTFUL?

- REST (Representational State Transfer) is an architectural style for designing APIs.
- Key principles:
- Stateless
- Client-Server
- Cacheable
- Uniform Interface
- Example: A REST API for a Notes Application

WHAT IS CRUD?

- CRUD Operations:
- Read (GET) – Retrieve existing data
- Create (POST) – Add new data
- Update (PUT/PATCH) – Modify existing data
- Delete (DELETE) – Remove data

```
GET /notes        → Fetch all notes
GET /notes/:id    → Fetch a single note
POST /notes       → Add a new note
PUT /notes/:id    → Update a note
DELETE /notes/:id → Remove a note
```

SETTING UP A SIMPLE EXPRESS SERVER

- **Install Node.js and Express**

```
npm init -y
npm install express
```

- **Create server.js**

```
const express = require("express");
const app = express();
const port = 3000;

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

EXPRESS ROUTES & WILDCARDS

- Defining routes in Express

```
app.get("/", (req, res) => {
    res.send("Welcome to the Notes API!");
});
```

- Using route parameters (wildcards)

```
app.get("/notes/:id", (req, res) => {
    res.send(`Fetching note with ID: ${req.params.id}`);
});
```

REAL WORLD APPLICATIONS FOR TDD

- Handling JSON Data in Express

  Middleware for handling JSON requests

```
app.use(express.json());
```

- Example POST request

```
app.post("/notes", (req, res) => {
    console.log(req.body);
    res.json({ message: "Note created!", note: req.body });
});
```

SERVING STATIC FILES IN EXPRESS

- Serving Static Files in Express
- Serving static frontend files such as HTML, CSS, and JavaScript.
- Setting up Express to serve static files

```
app.use(express.static("public"));
```

- Access frontend files by visiting http://localhost:3000/.

GITIGNORE: HOW TO EXCLUDE FILES AND FOLDERS FROM A REPOSITORY

- **Purpose of .gitignore**:

  Prevent certain files or folders from being tracked by Git.

- **Typical Example**

```
node_modules/
.env
```

REAL WORLD EXAMPLES OF .GITIGNORE

- **Common Scenarios**:

  Node.js projects: Exclude node_modules/.

  Environment files: Exclude .env to protect sensitive information.

  Log files: Prevent log files from cluttering the repository.

DEPLOYING TO RENDER.COM

- Why Render.com?

  Free hosting for Node.js applications
  Easy deployment from GitHub

- Steps to deploy:

1. Push project to GitHub
2. Create a Render account
3. Click **New Web Service** and connect the repository
4. Set the **Start Command** to

```
npm start
```

TESTING THE API WITH POSTMAN

- Testing the API with Postman

  Install and open Postman.

  Send a GET request to http://localhost:3000/notes.

  Send a POST request with JSON data

```
{
    "title": "Learn Express",
    "content": "Build APIs with Node.js and Express!"
}
```

CONCLUSION

- Building RESTful APIs with Express is a crucial skill for full-stack development. Mastering these concepts will allow you to create scalable, efficient, and deployable applications. Keep practicing, experimenting, and improving your API development skills!

Step8Up Academy

QUESTIONS?