

Backend (Express)

1. Understand Node.js and JavaScript Fundamentals

- **Node.js Basics:** Learn the fundamentals of Node.js since Express is built on top of it.
 - Learn about event-driven programming in Node.js.
 - Understand how Node.js handles asynchronous code using callbacks, promises, and `async/await`.
- **npm (Node Package Manager):** Learn how to use npm to install libraries, manage dependencies, and work with `package.json`.
- **File System (fs) module:** Get comfortable with basic file handling in Node.js.
- **Event Loop:** Understand how the event loop works in Node.js and why it's important for performance.

2. Install and Set Up Express

- **Install Express:** Learn how to install Express and set up a basic Express server.
- **Hello World Example:** Create a simple "Hello World" app to understand how routing and requests work in Express.
- **Understand HTTP Methods:** Learn about HTTP methods (GET, POST, PUT, DELETE) and how they map to Express routes.

3. Routing in Express

- **Basic Routes:** Learn how to define routes for handling different HTTP requests (GET, POST, etc.).
- **Route Parameters:** Learn how to handle dynamic route parameters (e.g., `/users/:id`).

- **Query Strings:** Understand how to access and handle query parameters in Express.
- **Express Router:** Learn how to break up your application into smaller, modular route handlers using `express.Router()`.

4. Middleware in Express

- **What is Middleware?**: Understand what middleware is and how it's used to intercept requests.
- **Built-in Middleware**: Learn about built-in middleware functions like `express.json()`, `express.urlencoded()`, `express.static()`, etc.
- **Custom Middleware**: Learn how to create your own middleware to handle requests before they reach the route handler.
- **Error Handling Middleware**: Learn how to handle errors gracefully with custom error-handling middleware.

5. Handling Requests and Responses

- **Request Object (`req`)**: Learn about the request object and how to access request data (headers, body, params, query strings).
- **Response Object (`res`)**: Learn how to send responses using the response object (`res.send()`, `res.json()`, `res.status()`, etc.).
- **Request Body Parsing**: Learn how to parse JSON and form data in the request body using middleware like `express.json()` and `express.urlencoded()`.

6. Working with Databases

- **Connecting to a Database**:
 - Learn how to connect your Express app to a database (MongoDB with Mongoose or SQL databases like PostgreSQL, MySQL, etc.).
- **CRUD Operations**:
 - **Create**: Learn how to handle POST requests to add data to your database.
 - **Read**: Learn how to handle GET requests to retrieve data from your database.

- **Update:** Learn how to handle PUT/PATCH requests to modify data.
- **Delete:** Learn how to handle DELETE requests to remove data.
- **Database Models:** Learn how to create models/schemas in Mongoose (if using MongoDB) or use an ORM like `typeorm` to interact with the database.
- **Basic Relations :**
 - One to one
 - One to many
 - Many to one
 - Many to one between table

7. Authentication and Authorization

- **JWT (JSON Web Tokens):** Learn how to implement JWT for stateless authentication.
- **User Registration and Login:** Implement user registration and login functionality with password hashing (using bcrypt).
- **Passport.js:** Optionally, explore Passport.js for more advanced authentication strategies (e.g., OAuth, social login).
- **Authorization:** Learn how to control access to specific routes by verifying JWTs and checking user roles/permissions.

8. Security Best Practices

- **CORS (Cross-Origin Resource Sharing):** Learn how to handle CORS for allowing cross-origin requests.
- **Environment Variables:** Use environment variables (via `.env` files) to store sensitive data like API keys, database credentials, etc.
- **Helmet.js:** Use `helmet` middleware to set HTTP headers for security.
- **Rate Limiting:** Implement rate limiting to prevent abuse and denial-of-service attacks using libraries like `express-rate-limit`.

- How to prevent sql injections

9. Building a RESTful API

- **REST Principles:** Learn about RESTful design principles for building APIs, such as statelessness, resource-based URLs, and HTTP method usage.
- **API Design:** Learn how to design a well-structured API with proper endpoints (e.g., `/users`, `/products`, `/orders`).
- **Versioning:** Implement versioning for your API (e.g., `/v1/users`, `/v2/users`).
- **Handling JSON Responses:** Learn how to structure and send responses in JSON format.

10. API Testing

- Learn how to test API using [Postman](#)
- API testing with [artillery](#)

11. Asynchronous Programming in Express

- **Async/Await:** Learn how to handle asynchronous code in Express (e.g., database queries, external API calls) using `async/await`.
- **Error Handling in Async Functions:** Learn how to handle errors properly in asynchronous code using `try/catch` blocks or middleware.