.

Week 1: Introduction to Node.js and Setup

OVERVIEW OF NODE.JS

- ✔ What is Node.js and where is it used
- ✔ Differences between Node.js and traditional server-side frameworks
- ✔ Node.js architecture and event-driven model.

SETTING UP THE ENVIRONMENT

- ✔ Installing Node.js and npm (Node Package Manager)
- ✔ Configuring the development environment (VS Code setup, useful extensions).

NODE.JS BASIC

- ✔ The Node REPL and executing basic commands
- ✔ Understanding package.json and node_modules.

HANDS-ON PRACTICE

- ✔ Creating your first Node.js script.
- ✔ Writing a simple “Hello World” HTTP server.

Week 2: Core Modules and File System Operations

CORE MODULES OVERVIEW

- ➤ Introduction to essential core modules (e.g., `fs`, `path`).
- ➤ File System Module (fs)
- ➤ Reading and writing files.
- ➤ Working with directories.
- ➤ File operations like renaming and deleting.

HANDS-ON PRACTICE

- ➤ Building scripts to manage files and directories.
- ➤ Creating a simple file-based logging utility.

Week 3: Working with Asynchronous Programming

SYNCHRONOUS VS ASYNCHRONOUS PROGRAMMING

- ❖ Overview of blocking vs non-blocking code.

CALLBACK FUNCTIONS

- ❖ Understanding callbacks and callback hell.

PROMISES AND ASYNC/AWAIT

- ❖ Working with promises and promise chaining.
- ❖ Converting callbacks to promises.
- ❖ Using async/await for cleaner asynchronous code.

HANDS-ON PRACTICE

- ❖ Rewriting file system code using async/await.
- ❖ Creating functions using both callback and promise methods.

Week 4: HTTP and Web Servers

HTTP BASICS AND REQUEST/RESPONSE MODEL

- ★ Overview of HTTP protocol and REST principles.

CREATING HTTP SERVERS

- ★ Using the `http` module to create servers.
- ★ Handling request and response.
- ★ Routing basics.

HANDS-ON PRACTICE

- ★ Building a simple RESTful API with plain Node.js.
- ★ Implementing basic CRUD (Create, Read, Update, Delete) operations.

Week 5: Express.js Basics

INTRODUCTION TO EXPRESS.JS

1. Why use Express over plain Node.js.
2. Setting up Express and creating a basic server.

ROUTING IN EXPRESS

1. Creating routes for different endpoints.
2. Handling URL parameters and query strings.

MIDDLEWARE IN EXPRESS

1. Concept of middleware and its uses.
2. Writing custom middleware and using third-party middleware.

HANDS-ON PRACTICE

1. Creating a simple app with Express routes.
2. Using middleware for logging and error handling.

Week 6: Template Engines and Serving Static Files

TEMPLATE ENGINES OVERVIEW

- ✔ Why use a template engine.
- ✔ Installing and using EJS (Embedded JavaScript) as a template engine.

STATIC FILES AND VIEWS

- ✔ Serving static assets (CSS, JavaScript, images).
- ✔ Rendering dynamic content with EJS.

HANDS-ON PRACTICE

- ✔ Building a simple app with dynamic pages using EJS.
- ✔ Creating an Express app with a simple frontend.

Week 7: Working with Databases (MongoDB)

INTRODUCTION TO NOSQL AND MONGODB

a. Difference between SQL and NoSQL.
b. Overview of MongoDB and setting up a local MongoDB server.

INTEGRATING MONGODB WITH NODE.JS USING MONGOOSE

a. Installing Mongoose and defining models.
b. Basic CRUD operations with Mongoose.

HANDS-ON PRACTICE

a. Building a RESTful API that stores data in MongoDB.
b. Implementing CRUD operations with Express and Mongoose.

Week 8: Authentication and Authorization

USER AUTHENTICATION CONCEPTS

- ❖ Understanding authentication vs. authorization.
- ❖ Overview of JWT (JSON Web Tokens) for token-based authentication.

IMPLEMENTING AUTHENTICATION WITH JWT

- ❖ Setting up routes for user registration and login.
- ❖ Securing routes with JWT.

HANDS-ON PRACTICE

- ❖ Creating a login and signup system with JWT.
- ❖ Implementing protected routes in Express.

Week 9: API Development and Documentation

API DESIGN PRINCIPLES

- ★ RESTful API best practices.
- ★ Structuring routes and controllers.

DOCUMENTATION WITH SWAGGER

- ★ Setting up Swagger for API documentation.
- ★ Writing and visualizing API documentation.

HANDS-ON PRACTICE

- ★ Documenting a sample API using Swagger.
- ★ Testing endpoints with Postman and checking documentation.

Week 10: Real-Time Applications with Socket.io

INTRODUCTION TO WEBSOCKETS AND REAL-TIME COMMUNICATION

- ➤ WebSocket basics and how they differ from HTTP.
- ➤ Overview of Socket.io.

SETTING UP A REAL-TIME CHAT APPLICATION

- ➤ Installing Socket.io and setting up a basic server.
- ➤ Handling real-time events with Socket.io.

HANDS-ON PRACTICE

- ➤ Building a simple chat app with Socket.io and Express.
- ➤ Implementing real-time notifications.

Week 11: Error Handling and Debugging

ERROR HANDLING IN NODE.JS

- ❖ Types of errors (e.g., runtime, logical, and syntax errors).
- ❖ Error handling in Express with middleware.

DEBUGGING TECHNIQUES

- ❖ Using Node.js built-in debugger.
- ❖ Working with `console` and debugging with VS Code.

HANDS-ON PRACTICE

- ❖ Implementing error handling in a sample project.
- ❖ Practicing debugging techniques in real applications.

Week 12: Deployment and Final Project

PREPARING AN APPLICATION FOR PRODUCTION

- ✔ Environment variables with dotenv.

- ✔ Setting up a production environment with PM2.

DEPLOYING A NODE.JS APP

- ✔ Deploying to cloud platforms like Heroku or DigitalOcean.
- ✔ Overview of Docker and containerizing a Node.js application.

FINAL PROJECT AND REVIEW

- ✔ Assigning a capstone project where students build a complete application.
- ✔ Reviewing key concepts from the course

ADDITIONAL RESOURCES

- o Reading Material: Node.js documentation, Express documentation, Mongoose
- o documentation.
- o Tools & Utilities: VS Code, Postman, MongoDB Compass, Docker (for advanced
- o deployment).
- o Suggested Practice:Weekly assignments and quizzes for reinforcement.