Full Stack Development – OA25FSD01

Beginner to Expert Course

Course Description

Master both frontend and backend development to build complete web applications from scratch. This course combines HTML, CSS, JavaScript, React, Node.js, and NestJS to create scalable, full-stack applications. Learn to integrate user interfaces with server-side logic and deploy to the cloud. Perfect for 2025 web development.

What You'll Learn

- Build responsive frontend interfaces with React
- Develop secure backend APIs with Node.js and NestJS
- Integrate frontend and backend for full-stack apps
- Work with databases and real-time features
- Deploy and scale full-stack applications

Course Requirements

- No prior programming knowledge required
- A computer with Node.js, VS Code, and a browser

Who This Course Is For

- Beginners aiming to become full-stack developers
- Developers looking to combine frontend and backend skills
- Professionals building modern web applications

Syllabus (35 Sessions)

1. **Web Development Overview**
   - Frontend vs. backend vs. full stack
   - Setting up VS Code, Node.js, and Git
   - Introduction to HTML and CSS
   - Coding shortcuts
2. **Frontend Basics: HTML and CSS**
   - HTML tags and semantic structure
   - CSS basics: Selectors, Box Model
   - Flexbox for layouts
   - Responsive design with media queries
3. **JavaScript Fundamentals**
   - Variables, functions, and scope

- DOM manipulation and events
- ES6+ features: Arrow functions, modules
- Debugging with browser tools

4. **Backend Basics: Node.js and Express**
   - Setting up a Node.js server
   - Creating Express routes
   - Handling HTTP requests
   - Testing with Postman
5. **Version Control**
   - Git and GitHub basics
   - Commits, branches, and merges
   - Collaborative workflows
   - Resolving conflicts
6. **Introduction to Databases**
   - MongoDB setup with Mongoose
   - CRUD operations
   - Connecting Express to MongoDB
   - Error handling
7. **Frontend with React**
   - Setting up React with Vite
   - JSX, components, and props
   - State management with useState
   - Styling with Tailwind CSS
8. **REST APIs and Integration**
   - Building a REST API with Express
   - Fetching data in React with fetch
   - Handling JSON and errors
   - Displaying API data in UI
9. **Practice Session**
   - Build a to-do app with React and Express
   - Congratulations! You've built a full-stack foundation.
10. **React Hooks and Routing**
    - useEffect and custom Hooks
    - React Router for navigation
    - Building forms in React
    - Validating user input
11. **Authentication**
    - JWT-based authentication
    - User registration and login
    - Securing Express routes
    - Integrating auth in React
12. **State Management**
    - Context API in React
    - Introduction to Redux
    - Managing backend state

- Persisting data across sessions

13. **Advanced Backend with NestJS**
    - Setting up NestJS
    - Creating controllers and services
    - Connecting to MongoDB
    - Dependency injection
14. **Advanced Frontend with Next.js**
    - Server-side rendering with Next.js
    - Static site generation
    - API routes in Next.js
    - Optimizing React apps
15. **Database Design**
    - Introduction to PostgreSQL
    - Creating schemas with Sequelize
    - Joins and relationships
    - MongoDB aggregations
16. **Practice Session**
    - Build a blog app with React and NestJS
    - Congratulations! You're an intermediate full-stack developer.
17. **TypeScript Basics**
    - TypeScript setup for frontend and backend
    - Types and interfaces
    - Type-safe React components
    - Type-safe APIs with NestJS
18. **Real-Time Features**
    - WebSockets with Socket.IO
    - Building a real-time chat feature
    - Integrating with React
    - Testing real-time functionality
19. **API Documentation and Testing**
    - Documenting APIs with Swagger
    - Unit testing with Jest
    - Integration testing for full-stack
    - Mocking APIs and databases
20. **Performance Optimization**
    - Caching with Redis
    - Lazy loading in React
    - Optimizing database queries
    - Measuring performance with Lighthouse
21. **Security Best Practices**
    - Securing APIs with CORS and CSRF
    - Preventing XSS and SQL injection
    - Environment variable management
    - Auditing application security
22. **Microservices Introduction**

- Microservices with NestJS
- Inter-service communication
- Using message queues (RabbitMQ)
- Scaling microservices

23. **Deploying Full-Stack Apps**
    - Deploying frontend to Vercel
    - Deploying backend to Heroku
    - Setting up CI/CD with GitHub Actions
    - Configuring production environments
24. **Practice Session**
    - Deploy a full-stack CRUD app
    - Congratulations! You've mastered full-stack development.
25. **Project 1: Task Manager**
    - Build a task manager with React and Express
    - Implement CRUD and authentication
    - Style with Tailwind CSS
    - Deploy to Vercel and Heroku
26. **Project 2: E-Commerce App**
    - Create an e-commerce UI with React
    - Build a product API with NestJS
    - Integrate Stripe for payments
    - Add search and filters
27. **Project 3: Real-Time Chat**
    - Build a chat app with React and Socket.IO
    - Store messages in MongoDB
    - Implement user presence
    - Deploy with CI/CD
28. **Project 4: Blog with CMS**
    - Create a blog with Next.js
    - Build a CMS API with NestJS
    - Manage posts and users
    - Deploy to Vercel and Heroku
29. **Project 5: Social Media App**
    - Build a social feed with React
    - Create a backend API with NestJS
    - Add like and comment features
    - Test with Jest
30. **Project 6: Analytics Dashboard**
    - Create a dashboard with React and Chart.js
    - Build an analytics API with NestJS
    - Store data in PostgreSQL
    - Deploy with Docker
31. **Advanced Testing**
    - End-to-end testing with Cypress
    - Performance testing for full-stack

- Mocking APIs and databases
- Load testing with Artillery

32. **Serverless Full Stack**
    - Serverless APIs with AWS Lambda
    - Serverless Framework for backend
    - Integrating with Next.js
    - Deploying serverless apps
33. **Project 7: Portfolio with CMS**
    - Build a portfolio with Next.js
    - Create a headless CMS with Strapi
    - Manage content and users
    - Deploy with CI/CD
34. **Scaling Full-Stack Apps**
    - Scaling with Kubernetes basics
    - Monitoring with New Relic
    - Caching with Redis
    - Handling high traffic
35. **Final Exam and Wrap-Up**
    - Comprehensive exam on full-stack concepts
    - Review of key skills and best practices
    - Guidance on full-stack career paths
    - Certificate of completion