

AI-Powered MERN Full Stack + TypeScript

Build, test and ship production-ready web applications with MongoDB, Express, React and Node.js

Full-Stack Development · intermediate · Live online, instructor-led

This syllabus is a draft awaiting instructor review and may change.

A 108-hour live course in full-stack TypeScript: React with TanStack Query, Express APIs, MongoDB modelling and aggregation, authentication, testing, Docker, CI/CD, caching and a reviewed capstone project.

Length and schedule

108 hours of live teaching, on either schedule:

- Weekday Intensive (Mon–Fri): about 22 weeks before holidays
- Weekend Evening (Sat & Sun evenings): about 27 weeks before holidays

Fees per student, whole course

- Group: Rs. 19,999
- Duo: Rs. 32,999
- One-to-One: Rs. 59,999

Launch fee — GST treatment to be confirmed.

What you will be able to do

- Write modern, typed JavaScript with async/await, modules and TypeScript
- Structure a React application with components, hooks, routing and well-chosen state management
- Design and build a versioned REST API with Express, validation and consistent error handling
- Model data in MongoDB with Mongoose, add the indexes real queries need and write aggregation pipelines
- Implement authentication and role-based authorisation, and defend against common OWASP risks
- Test the stack at unit, API and end-to-end level with Vitest, Supertest and Playwright
- Containerise the application, automate checks with GitHub Actions and deploy it
- Diagnose performance problems, add caching with Redis and debug production issues from logs

Before you start

- Basic programming in any language: variables, conditions, loops and functions
- HTML and CSS basics: you can build a simple static page and style it
- Comfortable using a terminal to run commands and navigate folders
- A laptop with at least 8 GB RAM (16 GB recommended for Docker) running Windows, macOS or Linux

Tools

Node.js 24 LTS, TypeScript, React 19, Vite, React Router, TanStack Query, Zustand, React Hook Form, Express, MongoDB 8, Mongoose, MongoDB Compass, Zod, Redis, Vitest, Supertest, Playwright, Docker and Docker Compose, GitHub Actions, Git and GitHub, Visual Studio Code

Syllabus

Module 1: Modern JavaScript and TypeScript

The language foundations the rest of the course relies on: modern JavaScript, asynchronous code, modules, tooling and TypeScript.

1. Course setup and modern JavaScript essentials (2 h)
2. Functions, closures, objects and arrays (2 h)
3. Asynchronous JavaScript and the event loop (2 h)
4. Modules, packages and project tooling (2 h)
5. TypeScript fundamentals (2 h)
6. Week 2 review and doubt-clearing (2 h)

Module 2: React architecture and hooks

Build user interfaces from typed components, manage local state and effects, and structure a React project that stays maintainable.

7. React and Vite project setup (2 h)
8. State, events and controlled inputs (2 h)
9. Effects, refs and the component lifecycle (2 h)
10. Custom hooks and composition patterns (2 h)
11. Project structure, styling and accessibility (2 h)
12. React review and doubt-clearing (2 h)

Module 3: Routing, forms and state management

Turn screens into an application: client-side routing, robust forms and a clear split between server state and client state.

13. Client-side routing with React Router (2 h)
14. Server state with TanStack Query (2 h)
15. Mutations and optimistic updates (2 h)
16. Client state with Zustand and Context (2 h)
17. Forms with React Hook Form and Zod (2 h)
18. Front-end review and doubt-clearing (2 h)

Module 4: Node.js and Express API design

Design and build a clean, versioned REST API in Express with a layered structure, validation and consistent errors.

19. Node.js runtime and the HTTP model (2 h)
20. Express fundamentals and middleware (2 h)
21. REST API design (2 h)
22. Layered architecture, validation and error handling (2 h)
23. Configuration, file uploads and background-safe code (2 h)
24. API review and doubt-clearing (2 h)

Module 5: MongoDB and Mongoose

Model data for MongoDB, query it efficiently with Mongoose, and answer reporting questions with aggregation pipelines.

25. Document databases and MongoDB basics (2 h)
26. Schema design for MongoDB (2 h)
27. Mongoose models, validation and middleware (2 h)

- 28. Indexing and query performance (2 h)
- 29. Aggregation pipelines (2 h)
- 30. Transactions, data integrity and doubt-clearing (2 h)

Module 6: Authentication, authorisation and security

Sign users in safely, control what each role may do, and defend the application against the most common web vulnerabilities.

- 31. Authentication fundamentals (2 h)
- 32. Sessions versus JWTs (2 h)
- 33. Role-based access control and ownership checks (2 h)
- 34. OWASP basics for MERN applications (2 h)
- 35. Password reset, email verification and OAuth sign-in (2 h)
- 36. Security review and doubt-clearing (2 h)

Module 7: Testing the stack

Write the right tests at the right level so changes can be made with confidence.

- 37. Unit testing with Vitest (2 h)
- 38. Component testing with React Testing Library (2 h)
- 39. API and integration testing with Supertest (2 h)
- 40. End-to-end testing with Playwright and doubt-clearing (2 h)

Module 8: Docker, CI/CD and deployment

Package the application in containers, automate checks on every change and deploy it to a cloud environment.

- 41. Containers and Docker (2 h)
- 42. Docker Compose for local development (2 h)
- 43. CI/CD with GitHub Actions (2 h)
- 44. Deployment and release practice (2 h)

Module 9: Performance, caching and production operations

Keep the application fast and diagnosable in production, and integrate an AI API responsibly as an optional advanced project.

- 45. Front-end and API performance (2 h)
- 46. Caching with Redis (2 h)
- 47. Logging, monitoring and production debugging (2 h)
- 48. Advanced optional project: integrating an AI API (2 h)

Module 10: Capstone project and final assessment

Build, harden and present the capstone application, with structured reviews and the final assessment.

- 49. Capstone kickoff and planning (2 h)
- 50. Capstone build lab: data and API (2 h)
- 51. Capstone build lab: front end and integration (2 h)
- 52. Capstone mid-point review (2 h)
- 53. Final assessment (2 h)
- 54. Capstone presentations and final review (2 h)

Final project

Build and deploy a clinic appointment-booking platform. A small chain of physiotherapy clinics needs a web application where patients book, reschedule and cancel appointments; therapists manage their weekly availability; and clinic administrators see bookings, no-shows and utilisation per clinic. You receive a written brief with user stories and acceptance criteria, and you build it individually or in a pair. Deliverables- A React and TypeScript front end with routing, forms validated by Zod, server state in TanStack Query and loading, empty and error states on every data view.- An Express API with versioned routes, request validation, a consistent error format and role-based access for patients, therapists and administrators.- A MongoDB schema with documented indexes for the main queries, transactions or safe conditional updates that prevent double-booking, and at least two aggregation reports (for example, weekly utilisation per therapist).- Authentication with secure, HTTP-only session cookies or short-lived tokens, password hashing and rate-limited sign-in.- Automated tests: unit tests for booking rules, Supertest API tests including an authorisation test proving one patient cannot read another's bookings, and one Playwright journey for booking an appointment.- A Dockerfile and Docker Compose setup, a GitHub Actions workflow that runs lint, type checks and tests, and a live deployment with environment-based configuration and no secrets in the repository.- Redis caching for one read-heavy endpoint, with a short note on how the cache is invalidated.- A README with setup steps, an architecture diagram and a list of known limitations. Optional extension: an AI-assisted feature, such as summarising a therapist's free-text session notes, that is opt-in, explains what is sent to the AI provider and handles provider errors gracefully. The project is presented and reviewed live in the final weeks, with feedback on code quality, security and design decisions.

Certificate

- At least 80% attendance
- All assignments submitted
- Projects passed
- Final assessment passed (60%)

A ZORK Academy course completion certificate; paying for the course does not by itself earn it.

Next step

Book a free consultation: <https://zorkacademy.online/consultation?course=mern-stack-development>

We help you prepare for work; we do not guarantee jobs, placements or salaries.