



MERN INTERNSHIP – Daily Student Mastery Checklist

Day 1 — MERN Overview & Environment Setup

- I understand the MERN architecture and request flow.
 - I installed Node.js, MongoDB, Git, and VS Code successfully.
 - I can initialize npm and manage package.json.
 - I created and linked a GitHub repository for the LMS project.
 - I configured ESLint & Prettier.
 - I understand commit, push, pull, and branching workflows.
-

Day 2 — HTML5 Essentials

- I can write semantic, accessible HTML.
 - I can create HTML5 forms and multimedia sections.
 - I understand meta tags and SEO basics.
 - I can structure folders for a large application.
 - I created Home & Login pages for the LMS.
-

Day 3 — CSS3 + Tailwind CSS

- I understand CSS animations, transitions, layouts.
 - I installed & configured Tailwind CSS.
 - I can build responsive layouts using Tailwind utilities.
 - I implemented dark mode.
 - I styled Home, Login, and Dashboard pages.
-

Day 4 — JavaScript Fundamentals

- I understand arrays, objects, loops, functions.
- I can manipulate DOM and handle events.

- I can validate forms using JavaScript.
 - I can parse and manage JSON data.
-

Day 5 — Modern JavaScript (ES6+ & APIs)

- I understand modules, arrow functions, destructuring.
 - I can use async/await, Promises, and Fetch API.
 - I can work with localStorage/sessionStorage.
 - I integrated a dummy REST API to load course data.
-

Day 6 — React Setup & Component Architecture

- I created a React project using Vite.
 - I understand JSX and the Virtual DOM.
 - I structured folders (components, pages, assets).
 - I built Navbar, Footer, Sidebar as reusable components.
 - I can pass props and manage component state.
-

Day 7 — State Management & Hooks

- I understand useState, useEffect, useRef, useContext.
 - I implemented global state using Context API.
 - I avoided prop drilling by using context effectively.
 - I built Course List and Add Course pages using hooks.
 - I tested dynamic UI updates successfully.
-

Day 8 — Routing & Form Handling

- I configured React Router v6.
- I implemented protected routes and redirects.
- I used React Hook Form + Yup validation.
- I connected dashboards for Admin/Instructor/Student.
- I checked route-level access restrictions.

Day 9 — Node.js & Express.js Fundamentals

- I created an Express server with routes.
 - I implemented REST API endpoints for Course CRUD.
 - I handled CORS correctly.
 - I tested all APIs using Postman.
 - I used .env for secrets and safety.
-

Day 10 — MongoDB & Mongoose

- I created schemas and models using Mongoose.
 - I connected backend to MongoDB Atlas.
 - I performed CRUD operations using Mongoose.
 - I validated data at schema level.
 - I integrated backend with LMS course page.
-

Day 11 — Authentication & Security

- I implemented JWT authentication.
 - I hashed passwords using bcrypt.
 - I created role-based authorization.
 - I implemented route protection middleware.
 - I tested login/signup/role access using Postman.
-

Day 12 — Cloud & Virtualization Basics

- I understood cloud vs on-premise concepts.
 - I learned VMs, hypervisors, containers.
 - I explored AWS EC2, S3, IAM, VPC.
 - I created an AWS account and launched an EC2 instance.
 - I verified cloud fundamentals through tasks.
-

Day 13 — AWS EC2 & Backend Deployment

- I connected to EC2 via SSH.
 - I installed Node.js, npm, and Git on EC2.
 - I deployed the backend API on EC2.
 - I configured EC2 security groups.
 - I used PM2 to manage the backend server.
-

Day 14 — AWS S3 & Frontend Deployment

- I hosted the React app on S3.
 - I configured IAM roles and bucket policies.
 - I integrated CloudFront CDN for faster delivery.
 - I connected frontend to backend API successfully.
 - I solved common deployment issues.
-

Day 15 — CI/CD with GitHub Actions

- I understood CI/CD workflows.
 - I created GitHub Actions YAML for auto-deploy.
 - I connected workflows to EC2 & S3.
 - I triggered and tested pipeline runs.
 - I fixed errors in GitHub Actions workflow.
-

Day 16 — Monitoring & Cost Optimization

- I configured CloudWatch monitoring.
 - I created AWS Budget alerts for cost control.
 - I applied IAM best practices to secure cloud resources.
 - I optimized deployment pipelines.
 - I tested real-time monitoring and debugging.
-

Final Project Readiness Checklist

A. FRONTEND – React.js Readiness

1. React Fundamentals

- I understand JSX, components, props, and state clearly.
- I can create reusable UI components and compose them effectively.
- I follow proper folder structure for components, pages, hooks, and services.
- I can manage conditional rendering and dynamic UI updates.

2. Hooks & State Management

- I can use useState, useEffect, useRef, useContext effectively.
- I can manage global state using Context API (or Redux if required).
- I can handle complex state updates and side effects.
- I can optimize rendering and avoid unnecessary re-renders.

3. Routing & Navigation

- I can configure React Router v6 with nested routes.
- I can implement protected routes and role-based routing.
- I can redirect users after login/logout based on role.
- I can handle dynamic routes (e.g., /course/:id).

4. Form Handling & Validation

- I can build forms using React Hook Form.
- I can apply Yup validation schemas for strong validation.
- I can handle file uploads (PDF, images, videos).
- I can manage error messages and form submission success flow.

B. BACKEND – Node.js & Express.js Readiness

1. Server & Routing

- I can create an Express server with modular routing.
- I can create controllers, middlewares, and config files following clean architecture.
- I can handle JSON requests, query params, headers, and route parameters.
- I can implement pagination, filtering, and sorting in API responses.

2. API Development

- I can design RESTful APIs for CRUD operations.
- I can structure API responses in a consistent format.
- I can validate inputs using middleware.
- I can test all APIs thoroughly using Postman.

3. Security & Authentication

- I can implement JWT-based authentication and refresh tokens.
- I can hash passwords securely using bcrypt.
- I can secure routes using authentication & authorization middleware.
- I can protect APIs from common attacks (XSS, CORS misconfigurations, rate limiting).

C. DATABASE – MongoDB & Mongoose Readiness

1. Schema Design

- I can design Mongoose schemas with proper data types and validation.
- I can implement relations using references (ObjectId) and embedding.
- I can perform efficient indexing when required.
- I can create reusable models for all LMS modules.

2. CRUD Operations

- I can create, read, update, and delete documents using Mongoose queries.
- I can handle population and nested data retrieval.
- I can manage error handling for database operations.
- I can connect my backend to MongoDB Atlas reliably.

3. Data Sanitization & Validation

- I can validate input data before saving to DB.
- I can sanitize user inputs to prevent NoSQL injection.
- I can create robust validation rules for LMS data (courses, users, submissions).

D. FULL STACK INTEGRATION READINESS

1. Connecting Frontend & Backend

- I can call backend APIs using Fetch/Axios from React.
- I can manage token-based authentication in frontend.
- I can store JWT tokens securely (localStorage/sessionStorage best practices).
- I can display dynamic data received from backend APIs.

2. Error Handling Across Stack

- I understand status codes (200, 400, 401, 403, 500, etc.).
- I can show meaningful feedback messages on the frontend for failed operations.
- I can handle network failures gracefully.
- I can create reusable error boundaries in React.

3. File Uploads & Media Handling

- I can upload files from React to Express backend.
- I can handle multipart/form-data and storage logic.
- I can store file references in MongoDB (PDF, Videos, Images).
- I can serve images/files securely to authenticated users.

E. DEPLOYMENT & CLOUD READINESS (Essential for Final Project)

1. Backend Deployment on EC2

- I can connect to EC2 using SSH.
- I can deploy Node.js + Express app on EC2 with PM2.
- I can configure Security Groups for API access.
- I can update deployment via Git pull + PM2 restart.

2. Frontend Deployment on S3 + CloudFront

- I can build React projects for production.
- I can host static files on S3 and configure public access correctly.
- I can integrate CloudFront for global caching and faster access.
- I can update live deployment with new builds.

3. Environment Variables & Secrets

- I can use .env for local environments.
- I can configure environment variables in EC2 safely.
- I can manage API URLs for dev and production.

F. QUALITY, GIT, & PROJECT MANAGEMENT READINESS

1. Git & Version Control

- I can commit with meaningful messages.
- I can work with branches and resolve merge conflicts.
- I can push/pull to GitHub consistently.
- I can maintain proper folder structure in the repository.

2. Code Quality

- I can use ESLint + Prettier effectively.
- I can follow clean coding practices in JS & React.
- I can name files, functions, and variables meaningfully.

3. Documentation & Presentation

- I can create a README with installation steps.
- I can document API routes using Markdown or Postman Collections.
- I can prepare a project demonstration script (5–7 min).
- I can explain the architecture of my LMS project confidently.