**MEAN Stack End-to-End Curriculum (100 Hours)**

**1. Course Overview**

This 100-hour curriculum is designed to provide a comprehensive, practical, and industry-focused education in MEAN stack development. Students will progress from foundational web concepts to building and deploying full-stack applications. The course emphasizes hands-on learning, culminating in a capstone project that showcases their abilities.

* **Total Duration:** 100 Hours
* **Suggested Pace:** 10 hours/week over 10 weeks.
* **Learning Path Progression:** The curriculum is structured logically:
  1. **Weeks 1-2:** Web Development & JavaScript Fundamentals
  2. **Weeks 3-4:** Backend Development with Node.js & Express.js
  3. **Weeks 5-6:** Database Integration with MongoDB & Mongoose
  4. **Weeks 7-8:** Frontend Development with Angular
  5. **Weeks 9-10:** Full-Stack Integration, Deployment & Advanced Topics
* **Assessment Methods:**
  1. Weekly coding assignments and practical labs.
  2. Three milestone projects of increasing complexity.
  3. Code reviews to enforce best practices.
  4. A final capstone project presentation and defense.

**2. Detailed Module Structure**

**Module 1: Web Foundations & Modern JavaScript (20 hours)**

* **Learning Objectives:**
  + Set up a professional development environment.
  + Master HTML5, CSS3 (Flexbox/Grid), and responsive design.
  + Understand and use Git for version control.
  + Achieve proficiency in modern JavaScript (ES6+), including asynchronous operations.
  + Get introduced to TypeScript and its benefits.
* **Prerequisites:** None. This is the starting point.
* **Sub-topics:**
  + **Topic 1: Development Environment & Git (3 hours)**
    - Intro to client-server architecture.
    - Installing and configuring VS Code, Node.js, and npm.
    - Git fundamentals: init, add, commit, push, branch, merge.
    - Collaborative workflows with GitHub: forking, pull requests.
  + **Topic 2: HTML5 & CSS3 (7 hours)**
    - HTML5 semantic elements and accessibility (WCAG) basics.
    - Advanced CSS: Flexbox, Grid, and responsive design with media queries.
    - Introduction to SASS/SCSS for maintainable stylesheets.
  + **Topic 3: Advanced JavaScript & Asynchronicity (7 hours)**
    - ES6+ deep dive: arrow functions, destructuring, classes, modules.
    - The event loop, callbacks, Promises, and async/await.
    - Working with APIs: fetch().
  + **Topic 4: Introduction to TypeScript (3 hours)**
    - Why TypeScript? Static typing, interfaces, and basic types.
    - Configuring tsconfig.json.
* **Hands-on Activities:**
  + Create a responsive personal portfolio page using HTML/CSS and deploy it to GitHub Pages.
  + Build a simple "Quote of the Day" app that fetches data from a public API.
* **Assessment/Project Component:** A multi-page, responsive static site built with SASS and JavaScript, managed with Git.

**Module 2: Backend Development with Node.js & Express.js (25 hours)**

* **Learning Objectives:**
  + Build robust RESTful APIs with Node.js and the Express.js framework.
  + Understand the Node.js runtime, module system, and npm ecosystem.
  + Implement middleware, routing, and environment configuration.
  + Master RESTful API design principles.
* **Prerequisites:** Module 1.
* **Sub-topics:**
  + **Topic 1: Node.js Runtime (5 hours)**
    - Node.js architecture and the V8 engine.
    - Core modules: fs, path, http.
    - Understanding npm scripts and package management (package.json).
  + **Topic 2: Express.js Core Concepts (10 hours)**
    - Setting up an Express server.
    - Routing: structuring routes, route parameters, query strings.
    - Middleware: custom middleware, third-party middleware (e.g., cors, morgan).
    - Handling POST, GET, PUT, DELETE requests.
  + **Topic 3: RESTful API Design & Implementation (10 hours)**
    - Designing RESTful endpoints and status codes.
    - Structuring controllers and service layers.
    - Centralized error handling and logging.
    - Managing secrets and environment variables with .env files.
* **Hands-on Activities:**
  + Build a simple web server using only the Node.js http module.
  + Refactor the server to use Express.js.
* **Assessment/Project Component:** **Project 1: To-Do List REST API**.

**Module 3: Database & Security with MongoDB & JWT (20 hours)**

* **Learning Objectives:**
  + Understand NoSQL concepts and MongoDB's document model.
  + Use Mongoose for schema design, validation, and data modeling.
  + Implement secure user authentication and authorization using JWT.
  + Protect API routes based on user roles.
* **Prerequisites:** Module 2.
* **Sub-topics:**
  + **Topic 1: MongoDB & Mongoose (10 hours)**
    - Setting up a free cluster on MongoDB Atlas.
    - Mongoose Schemas, models, and data validation.
    - CRUD operations: create, find, findById, findByIdAndUpdate, findByIdAndDelete.
    - Data modeling: relationships (embedding vs. referencing).
  + **Topic 2: Authentication & Authorization with JWT (10 hours)**
    - Password hashing with bcryptjs.
    - Implementing user registration and login endpoints.
    - Creating, signing, and verifying JSON Web Tokens (JWT).
    - Implementing middleware to protect routes.
    - Role-based access control (user vs. admin).
* **Hands-on Activities:**
  + Integrate MongoDB into the To-Do List API.
  + Add user registration and login functionality to the API.
* **Assessment/Project Component:** **Project 2: Blog Platform API** (start).

**Module 4: Frontend Development with Angular (25 hours)**

* **Learning Objectives:**
  + Understand the architecture of an Angular application.
  + Build dynamic user interfaces with components and templates.
  + Manage application flow with services and dependency injection.
  + Create single-page applications (SPAs) with the Angular Router.
  + Handle user input with reactive forms and communicate with a backend.
* **Prerequisites:** Module 1.
* **Sub-topics:**
  + **Topic 1: Angular Fundamentals (8 hours)**
    - Angular CLI, components, templates, and data binding ({{}}, [], (), [()]).
    - Directives: \*ngIf, \*ngFor, [ngClass], [ngStyle].
  + **Topic 2: Services, Routing, and State Management (10 hours)**
    - Dependency Injection and creating reusable services.
    - Client-side routing with the Angular Router and route guards.
    - Communicating with APIs using the HttpClientModule.
    - Introduction to state management with services (RxJS BehaviorSubject).
  + **Topic 3: Angular Forms & HTTP (7 hours)**
    - Template-driven vs. Reactive forms.
    - Building and validating complex forms with FormBuilder.
    - Handling HTTP requests and responses; error handling with interceptors.
* **Hands-on Activities:**
  + Build a simple component-based UI.
  + Create a multi-page application using the Angular Router.
* **Assessment/Project Component:** Develop the frontend for the **Blog Platform** and connect it to the API created in Module 3.

**Module 5: Deployment & Advanced Topics (10 hours)**

* **Learning Objectives:**
  + Prepare a MEAN stack application for production.
  + Deploy a full-stack application to a cloud platform like Azure.
  + Understand the basics of CI/CD pipelines.
  + Explore advanced features like file uploads and real-time communication.
* **Prerequisites:** Module 4.
* **Sub-topics:**
  + **Topic 1: Production Build & Deployment (6 hours)**
    - Building an Angular app for production (ng build).
    - Strategies for deploying a full-stack app (serving the frontend from Express).
    - Deploying to Azure App Service (or similar PaaS).
    - Introduction to CI/CD concepts.
  + **Topic 2: Advanced Features (4 hours)**
    - **File Uploads:** Handling multipart/form-data with libraries like multer.
    - **Real-time Intro:** Conceptual overview of WebSockets with Socket.IO.
    - **API Documentation:** Introduction to tools like Swagger/OpenAPI.
* **Hands-on Activities:**
  + Deploy the completed Blog Platform application.
* **Assessment/Project Component:** **Capstone Project: E-commerce Product Manager**.

**3. Project Timeline & Details**

* **Project 1: To-Do List REST API (Module 2 | 5 hours)**
  + **Description:** A backend-only REST API with full CRUD functionality for to-do items. Data will initially be stored in-memory.
  + **Skills:** Node.js, Express.js, REST API design, Postman testing.
* **Project 2: Full-Stack Blog Platform (Modules 3 & 4 | 15 hours)**
  + **Description:** A full-stack application where users can register, log in, create, and view blog posts.
  + **Backend:** Express API with JWT authentication and MongoDB for persistence.
  + **Frontend:** Angular SPA to consume the API, with protected routes and reactive forms.
* **Project 3: Capstone - E-commerce Product Manager (Module 5 | 20+ hours)**
  + **Description:** A production-ready application for managing an e-commerce inventory.
  + **Features:**
    - Full user authentication with admin/user roles.
    - Admins can perform CRUD operations on products (name, description, price, category).
    - Includes file uploads for product images.
    - Users can only view products.
    - Deployed to a live URL on Azure.
    - (Bonus) Add a real-time notification feature for when a new product is added.

**4. Assessment Strategy**

* **Weekly Assignments (30%):** Small, focused tasks related to the week's topics. Evaluated on correctness and adherence to coding standards.
* **Project 1 & 2 (30%):** Evaluated on functionality, code quality, REST API design (for P1), and frontend-backend integration (for P2).
* **Capstone Project (40%):** The final evaluation. Criteria include:
  + **Functionality:** All required features are implemented and bug-free.
  + **Code Quality:** Clean, modular, and well-commented code.
  + **Security:** Proper implementation of authentication, authorization, and data protection.
  + **Deployment:** The application is successfully deployed and accessible.
  + **Presentation:** The student can clearly explain their architecture and design choices.