ETHER INFOTECH

MERN STACK DEVELOPMENT

SYLLABUS

www.etherinfotech.com | +91 95002 95905

Why Choose Ether Infotech for Learning MERN Stack?

MERN Stack continues to be one of the most versatile and in-demand skill sets across industries. At Ether Infotech, Coimbatore, we go beyond teaching individual technologies — we equip you with complete full-stack MERN development skills that make you job-ready. Our career-focused training covers everything from MongoDB, Express.js, and React to Node.js, along with advanced modules like RESTful APIs, DevOps, Testing, Deployment, and AI integration, ensuring you're fully prepared to build real-world applications and succeed in interviews.

Join the No.1 MERN Stack Training Institute in Coimbatore with 100% Placement Assistance

Course Highlights

- 100% Practical Training
- Industry-Trained Mentors
- Beginner to Advanced Level
- Real-Time Projects
- Course Completion, Project Experience and Internship Certificate
- Internship & Placement Support
- Classroom + Practice Lab Environment
- 10000+ Students Trained Successfully
- 50+ Active Hiring Partners

Who Can Join?

- College Students
- Job Seekers
- Non-IT to IT Career Switchers

MERN Stack Development Syllabus 8hrs

FRONTEND

HTML (HyperText Markup Language)

HTML is the backbone of web pages used to structure content.

HTML Tags and Attributes

- Basic tags: div, span, h1 to h6, p, a, img, ul, ol, li, table, etc.
- Global attributes: id, class, style, title, alt
- Structural tags: header, nav, main, section, footer
- Inline vs block-level elements

Forms and Validation

- Creating input forms using form, input, textarea, select, button
- Input types: text, email, password, date, number
- Built-in validations: required, minlength, pattern, etc.
- Form submission using action and method

Semantic HTML

- Using meaningful tags for better accessibility and SEO
- Tags: article, aside, section, figure, figcaption, time, mark, etc.

CSS (Cascading Style Sheets)

CSS styles the HTML structure for appearance, layout, and responsiveness.

Selectors, Box Model, Positioning

- Element, class, ID, attribute selectors
- Box model: margin, border, padding, content
- Positioning: static, relative, absolute, fixed, sticky

Flexbox, Grid

- Flexbox: one-dimensional layout control

Properties: display: flex, justify-content, align-items, flex-direction, etc.

- CSS Grid: two-dimensional layout system

Properties: grid-template-columns, grid-gap, grid-area

Media Queries (Responsive Design)

- Creating responsive designs for mobile, tablet, and desktop
- Mobile-first and desktop-first approaches

CSS Frameworks: Bootstrap / Tailwind

- Bootstrap: utility and component classes
- Grids, cards, navbars, buttons
- Tailwind CSS: utility-first framework
- Custom styling using utility classes (flex, text-lg, bg-blue-500, etc.)

JavaScript (JS)

JS is the programming language of the web, enabling interactivity and logic.

Variables, Data Types, Operators

- Declaring variables using var, let, and const
- Primitive types: string, number, boolean, null, undefined
- Objects and arrays
- Arithmetic, logical, and comparison operators

DOM Manipulation

- Accessing elements using getElementById, querySelector, etc.
- Changing content: innerText, innerHTML
- Modifying styles and attributes
- Creating and removing elements dynamically

Functions, Events

- Function declarations, expressions, arrow functions
- Event handling: onclick, addEventListener
- Event types: click, change, input, mouseover, keypress

ES6 Features

- let and const
- Arrow functions (=>)
- Template literals (Hello ${name})
- Destructuring, spread/rest operators
- Promises and async/await
- Modules: import/export

Fetch API / Axios

- Making HTTP requests to external APIs
- Handling responses with Promises
- GET, POST, PUT, DELETE requests
- Error handling and loading states

Form Validation

- Client-side form validation using JS
- Custom validation rules
- Real-time feedback with events like on input or on blur

Single Page Applications (SPA) Basics

- Concept of SPAs: dynamic page updates without full reload
- History API and URL handling
- Benefits of SPA over traditional MPA

Frontend Framework – React

React is a popular JS library for building component-based UIs.

Components, Props, State

- Function and Class components
- Props: passing data between components
- State: managing dynamic data inside components
- Component re-rendering and lifecycle

Hooks (useState, useEffect)

- UseState: managing component-level state
- UseEffect: side effects (e.g., fetching data, timers)
- Cleanup and dependency array
- Other hooks (intro): useRef, useContext, useMemo, etc.

Routing (React Router)

- react-router-dom for navigation
- BrowserRouter, Route, Link, useNavigate
- Dynamic routes and URL parameters

Form Handling

- Controlled components (value and onChange)
- Form submission and validation in React
- Using libraries like Formik or React Hook Form (optional)

Conditional Rendering

- Using if statements, ternary operators
- && logical short-circuiting
- Conditional UI elements

API Integration

- Calling REST APIs from React using fetch or axios
- Updating UI based on API responses
- Error and loading state management

Redux (Advanced)

- Centralized state management
- Actions, Reducers, Store
- Middleware like redux-thunk for async logic
- Integration with React using react-redux

Backend Frame Work

Core Node.js

- Introduction to Node.js
- Node.js Architecture (Event Loop, Callbacks, EventEmitter)
- Modules and require()
- File System (fs module)
- Streams and Buffers
- Error Handling (try-catch, async error handling)
- npm & package.json

Express.js Framework

- Introduction to Express.js
- Creating a basic server
- Middleware (built-in, custom, third-party)
- Routing (GET, POST, PUT, DELETE)
- Query Params vs Path Params
- Request and Response Objects
- Error Handling Middleware
- Serving Static Files

Database Integration (MongoDB + Mongoose)

- Introduction to NoSQL & MongoDB
- Installing and using MongoDB locally / with Atlas
- Mongoose Basics
- Schema and Model Definition
- CRUD Operations
- Data Validation
- Indexing & Performance Tips

Authentication & Authorization

- Introduction to Authentication & Authorization
- Password Hashing with bcrypt
- User Registration and Login
- Session-based Authentication (express-session)
- JWT-based Authentication
- Role-based Access Control (RBAC)
- Protecting Routes
- CORS Configuration

RESTful API Development

- Designing REST APIs
- Status Codes & HTTP Methods
- API Versioning
- Structuring a RESTful Project
- Request Validation (express-validator)
- Global Error Handling
- Environment Configuration using dotenv

Advanced Express Patterns

- MVC Pattern in Express
- Service Layer and Repository Pattern
- Asynchronous Programming with async/await
- Dependency Injection
- Security Headers (helmet, express-rate-limit)

Database

SQL (PostgreSQL / MySQL)

CRUD Operations

- SELECT, INSERT, UPDATE, DELETE

Joins, Subqueries

- INNER JOIN, LEFT JOIN, RIGHT JOIN
- Nested queries and aliasing

DDL, DML

- DDL: CREATE, ALTER, DROP
- DML: INSERT, UPDATE, DELETE

MongoDB Basics

- MongoDB Architecture (Database > Collection > Document)
- Data Representation using BSON / JSON
- Mongo Shell vs MongoDB Compass (GUI)

Core CRUD Operations:

- insertOne(), insertMany()
- find(), findOne() with filter, projection, sort
- updateOne(), updateMany() with operators like $set, $inc
- deleteOne(), deleteMany()

Querying Features:

- Filtering with comparison operators ($eq, $gt, $lt, $in)
- Logical operators ($and, $or, $not)
- Sorting, Pagination using limit() and skip()
- Indexing: Creating and using indexes

MongoDB in Express js

Using MongoDB Native Driver

- MongoClient, connecting to MongoDB
- Performing basic operations (find, insertOne, updateOne, deleteOne)
- Working with BSON and JavaScript objects
- Error handling and connection pooling

DevOps, Deployment

Version Control

Git & GitHub

- Initializing repo, commits, branches
- Pull/push/clone
- GitHub collaboration (pull requests, issues)

Deployment

Deploying Express js App

- Deploying application on anyone platform (Heroku, Render, or AWS Ec2)
- Running migrations in production

Environment Variables

- Using .env files
- Keeping secrets secure (API keys, DB creds)

MongoDB in Java

Connecting MongoDB with Java

Using MongoDB Java Driver

- MongoClient, MongoDatabase, MongoCollection
- Inserting and retrieving documents using POJOs

Using Spring Data MongoDB

- @Document, @Id, @Field annotations
- MongoRepository for CRUD operations
- Query Methods and Custom Queries
- Aggregation Framework

MongoDB in Java

Version Control

Git & GitHub

- Initializing repo, commits, branches
- Pull/push/clone
- GitHub collaboration (pull requests, issues)

Deployment

Deploying Springboot App

- Deploying application on anyone platform (Heroku, Render, or AWS Ec2)
- Running migrations in production

Environment Variables

- Using .env files
- Keeping secrets secure (API keys, DB creds)

CI/CD Basics (Optional Advanced)

Docker

- Creating Dockerfiles
- Building and running containers
- Docker Compose for multi-container setups

GitHub Actions

- Creating workflows for testing and deployment
- CI pipelines for automated builds

Cloud (AWS)

- **AWS EC2** – Hosting Springboot apps on virtual servers
- **AWS S3** – Storing static and media files
- **AWS RDS** – Managed relational databases like PostgreSQL or MySQL

Other Must-Know Concepts

REST API Concepts with Spring Boot
- REST Architectural Style
- HTTP Methods: GET, POST, PUT, DELETE, PATCH
- RESTful Resource Naming Conventions
- HTTP Status Codes (200, 201, 400, 404, 500, etc.)
- Using @RestController, @RequestMapping, @GetMapping, etc.

JSON Handling in Spring Boot
- Automatic Serialization/Deserialization using Jackson (ObjectMapper)
- Annotations: @JsonIgnore, @JsonProperty, @JsonInclude
- Customizing JSON responses
- Content negotiation with @ResponseBody / @RequestBody

Integration (Frontend + Spring Boot)
- Making asynchronous API calls using fetch() or Axios (from React/Angular)
- Updating the UI without page reload
- Handling JSON responses from REST APIs
- Error handling in frontend for API failures

Postman for API Testing
- Testing Spring Boot REST APIs:
    1. Setting headers, query params, path variables
    2. Sending form data or JSON payload
- Authentication testing using Bearer Tokens (JWT)
- Automating tests using Postman Collections
- Environment variables for dynamic testing

CORS (Cross-Origin Resource Sharing)
- CORS Issues between Frontend and Backend
- Enabling CORS in Spring Boot:
    1. @CrossOrigin annotation
    2. Global CORS config using WebMvcConfigurer
- Configuring allowed origins, methods, headers, credentials

Web Security (CSRF, XSS Basics)

1. CSRF (Cross-Site Request Forgery)
- Enabled by default in Spring Security
- CSRF tokens in forms (if using Thymeleaf or form-based login)
- Disabling CSRF for stateless APIs (.csrf().disable() in REST APIs)

2. XSS (Cross-Site Scripting)
- Understanding injection vectors
- Output sanitization in frontend
- Setting security headers using Spring Security

JWT Authentication / OAuth2 (Spring Security)

1. Stateless Authentication with JWT:
- Generating and signing JWT tokens
- Sending JWT via Authorization headers
- Verifying and parsing JWT in filters

2. Role-Based Access Control:
- Using @PreAuthorize, @Secured, or hasRole() in configuration
- Custom UserDetailsService for user roles

3. Introduction to OAuth2 (Optional Advanced)
- Using Spring Security OAuth2 Client or Resource Server

Unit Testing with JUnit and Spring Boot
- Writing test cases with JUnit 5
- Using @SpringBootTest, @WebMvcTest, @DataJpaTest
- Mocking with Mockito
- Testing Controllers, Services, and Repositories
- Asserting status codes and response JSON using MockMvc

Data Structures and algorithm

Time and Space Complexity
- Big O Notation (O(n), O(1), O(log n), etc.)

Arrays and Strings
- Basics of arrays (1D, 2D)
- Searching and sorting arrays
- Sliding Window Technique
- Prefix Sum, Difference Arrays

Hashing (HashMap / HashSet)
- Frequency Counting
- HashMap vs TreeMap

Recursion and Backtracking
- Factorial, Fibonacci
- Subsets, Permutations

Linked Lists
- Singly, Doubly Linked Lists

Sorting Algorithms
- Bubble, Selection, Insertion (basic)
- Merge Sort, Quick Sort (optional)
- Sorting before two-pointer
- Sorting before binary search

Two Pointers and Sliding Window
- Two Sum (Sorted)
- Container With Most Water
- Longest Substring Without Repeating Characters

Stacks and Queues
- Stack using Array
- Next Greater Element, Balanced Parentheses
- Queue, Circular Queue
- Implement Stack using Queue and vice versa

FRONTEND

HTML (HyperText Markup Language)
HTML is the backbone of web pages used to structure content.

HTML Tags and Attributes
- Basic tags: div, span, h1 to h6, p, a, img, ul, ol, li, table, etc.
- Global attributes: id, class, style, title, alt
- Structural tags: header, nav, main, section, footer
- Inline vs block-level elements

Forms and Validation
- Creating input forms using form, input, textarea, select, button
- Input types: text, email, password, date, number
- Built-in validations: required, minlength, pattern, etc.
- Form submission using action and method

Semantic HTML
- Using meaningful tags for better accessibility and SEO
- Tags: article, aside, section, figure, figcaption, time, mark, etc.

CSS (Cascading Style Sheets)
CSS styles the HTML structure for appearance, layout, and responsiveness.

Selectors, Box Model, Positioning
- Element, class, ID, attribute selectors
- Box model: margin, border, padding, content
- Positioning: static, relative, absolute, fixed, sticky

Flexbox, Grid
- Flexbox: one-dimensional layout control

Properties: display: flex, justify-content, align-items, flex-direction, etc.
- CSS Grid: two-dimensional layout system

Properties: grid-template-columns, grid-gap, grid-area

Media Queries (Responsive Design)
- Creating responsive designs for mobile, tablet, and desktop
- Mobile-first and desktop-first approaches

JavaScript (JS)
JS is the programming language of the web, enabling interactivity and logic.

Variables, Data Types, Operators
- Declaring variables using var, let, and const
- Primitive types: string, number, boolean, null, undefined
- Objects and arrays
- Arithmetic, logical, and comparison operators

DOM Manipulation
- Accessing elements using getElementById, querySelector, etc.
- Changing content: innerText, innerHTML
- Modifying styles and attributes
- Creating and removing elements dynamically

Functions, Events
- Function declarations, expressions, arrow functions
- Event handling: onclick, addEventListener
- Event types: click, change, input, mouseover, keypress

ES6 Features
- let and const
- Arrow functions (=>)
- Template literals (Hello ${name})
- Destructuring, spread/rest operators
- Promises and async/await
- Modules: import/export

Fetch API / Axios
- Making HTTP requests to external APIs
- Handling responses with Promises
- GET, POST, PUT, DELETE requests
- Error handling and loading states

Form Validation
- Client-side form validation using JS
- Custom validation rules
- Real-time feedback with events like on input or on blur

Single Page Applications (SPA) Basics
- Concept of SPAs: dynamic page updates without full reload
- History API and URL handling
- Benefits of SPA over traditional MPA

Frontend Framework – React

Components, Props, State
- Function and Class components
- Props: passing data between components
- State: managing dynamic data inside components
- Component re-rendering and lifecycle

Hooks (useState, useEffect)
- UseState: managing component-level state
- UseEffect: side effects (e.g., fetching data, timers)
- Cleanup and dependency array
- Other hooks (intro): useRef, useContext, useMemo, etc.

Routing (React Router)
- React-router-dom for navigation
- BrowserRouter, Route, Link, useNavigate
- Dynamic routes and URL parameters

Form Handling
- Controlled components (value and onChange)
- Form submission and validation in React
- Using libraries like Formik or React Hook Form (optional)

Conditional Rendering
- Using if statements, ternary operators
- && logical short-circuiting
- Conditional UI elements

API Integration
- Calling REST APIs from React using fetch or axios
- Updating UI based on API responses
- Error and loading state management

Redux (Advanced)
- Centralized state management
- Actions, Reducers, Store
- Middleware like redux-thunk for async logic
- Integration with React using react-redux

AI Integration in Fullstack Applications

24/7 Support Assistance

AI technologies enhance this support by automating responses, resolving common queries, and escalating complex issues.

Key Features:

- AI-Driven Chatbots: Deploy intelligent conversational agents that can understand, interpret, and respond to user queries in real-time.
- Real-Time Query Handling: Offer solutions instantly without human intervention, reducing response time and operational costs.

Chat Bot and Query Resolver

Developing a smart assistant to handle common queries, guide users, and support issue resolution

Key Features:

- Use REST APIs or WebSocket to connect chatbot with backend.
- Store conversation logs and user interactions in database.

Code and Text Generator

Use Cases:

- Auto-generate frontend/backend code based on user input.
- Draft emails, summaries, or blog content.
- Assist with debugging or code explanation.

Tools:

- OpenAI GPT APIs for text/code generation.
- Frontend interfaces integrated with Backend to input prompts and display results.

Generative AI

- Introduction to LLMs (e.g., OpenAI GPT, Hugging Face)
- Use cases in full-stack apps (summarization, translation, etc.)

About Ether Infotech

Ether Infotech offers courses designed to equip students with the essential skills required for mastering Python programming. Located in Coimbatore, this institute is renowned for its practical approach to learning, industry-relevant curriculum, and experienced trainers.

The Python training at Ether Infotech covers a wide range of topics, from basic syntax and data structures to advanced concepts such as web development, data analysis, and machine learning. Each module is carefully designed to ensure students gain hands-on experience through real-world projects and assignments.

In addition to Python, Ether Infotech also provides training in various other technologies, making it a hub for aspiring tech professionals. The institute's commitment to quality education, coupled with state-of-the-art facilities and a supportive learning environment, makes it an ideal choice for anyone looking to advance their career in the tech industry.

Contact Info

35/4, 2ndFloor, Arputham Towers,Desabandhu Street, Ram Nagar, Coimbatore – 641009

+91 95002 95905 | +91 82205 71905

prasanth@etherinfotech.com

www.etherinfotech.com