IT Shaala

FULL STACK DEVELOPMENT

Using MERN

Course duration **: 4 - 6 months**

4.9
Google Rating

Offline/Online Program	**500 Hours of Live Classes**	**LMS Access**	**Internship Certificate**

MASTER MERN STACK DEVELOPMENT

Table of contents

Founder’s Message

IT Shaala is a Leading IT Training Provider for Web Development & Software Development Courses in Pune. It is one of the best Online/Classroom Training Institute, which provides High-Quality Industry Level Training with Real Time Projects. IT Shaala is Institute which broadcasts IT training from basic to advanced technologies.

IT Shaala has specially designed Job Oriented Programme for "Freshers" and "Job Seekers". We provide training by Expert Trainers who are having 15+ years of industry experience, who guide the candidate with the best knowledge and work culture of most IT companies. The training would be accompanied by live project and placement support, which makes the candidates ready to be absorbed in the industry. Our Institute conjointly gives a chance within their educational programs to meet the needs with projected desires of quick developing networking trade.

“

As you embark on your journey from student to professional, remember that every challenge is an opportunity to grow and excel.

”

About IT Shaala

Aim

At IT Shaala, our aim is to empower individuals with the knowledge and skills needed to thrive in the fast-paced world of information technology.
We strive to provide accessible and practical education that opens doors to new opportunities.

Vision

Our vision at IT Shaala is to be a trusted leader in IT education, inspiring lifelong learning and driving positive change in individuals' lives. We envision a future where everyone has the chance to excel in the digital age, regardless of their background or circumstances.

Mission

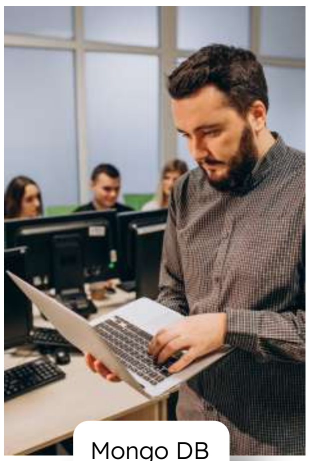
Our mission at IT Shaala is simple: to deliver high-quality IT training that transforms lives. We're dedicated to fostering a supportive and inclusive learning environment where students can grow personally and professionally, reaching their full potential in the world of technology.

Scope in the industry

MERN Stack Developer

React Developer

Node Js Developer



Mongo DB Developer

Course Syallabus

Introduction to MERN

1. How does Internet works?
2. What is HTTP?
3. Browsers and how they work?
4. DNS and how it works?
5. DNS and how it works?
6. DNS and how it works?

HTML

1. Introduction to HTML
   a. What is HTML?
   b. Structure of an HTML document
   c. Basic HTML tags and elements
2. HTML Document Structure
   a. Document type declaration
   b. <html>, <head>, and <body> tags
   c. Headings, paragraphs, and line breaks
3. Text Formatting
   a. <h1> to<h6> headings
   b. <p> and <span> elements
   c. Bold (<b>) and italic (<i>) text
4. Lists
   a. Unordered lists (<ul> and <li>)
   b. Ordered lists (<ol> and<li>)
   c. Definition lists (<dl>, <dt>, and<dd>)
5. Links and Anchors
   a. Creating hyperlinks (<a> tag)
   b. Linking to external websites
   c. Linking within the same page (anchors)
6. Images
   a. Embedding images (<img> tag)
   b. Image attributes (source, alt text, width, height)
7. Tables
   a. Creating tables (<table>, <tr>, and <td>)
   b. Table headers (<th>)
   c. Table captions (<caption>)
8. Forms
   a. Building forms (<form> tag)
   b. Text input fields (<input type="text">)
   c. Checkboxes (<input type="checkbox">)
   d. Radio buttons (<input type="radio">)
   e. Select dropdowns (<select> and <option>)
   f. Submit buttons (<input type="submit">)
9. HTML5 Semantic Elements
   a. <header>, <nav>, <footer>
   b. <section>, <article>, <aside>
   c. <figure> and <figcaption>
   d. <main>, <mark>, <time>
10. Multimedia
   a. Embedding videos (<video> tag)
   b. Adding audio (<audio> tag)
   c. Working with iframes (<iframe> tag)
11. HTML Validation and Best Practices
   a. Validating HTML code
   b. Organizing code using indentation and comments
   c. Using semantic HTML for accessibility

CSS

1. **Introduction to CSS**
   a. What is CSS and why is it used?
   b. CSS syntax and rule structure
   c. Inline, internal, and external CSS
2. **CSS Selectors**
   a. Element selectors
   b. Class selectors
   c. ID selectors
   d. Attribute selectors
   e. Pseudo-classes and pseudo-elements
3. **CSS Box Model**
   a. Understanding the box model concept
   b. Margin, border, padding, and content areas
   c. Box-sizing property
4. **Typography**
   a. Font properties (family, size, weight, style)
   b. Text color, alignment, and decoration
   c. Working with Google Fonts
5. **Colors and Backgrounds**
   a. Color values and color names
   b. Hexadecimal and RGB color codes
   c. Background properties (color, image, position, repeat)
6. **Layout and Positioning**
   a. Display property and values (block, inline, inline-block)
   b. Float and clear properties
   c. Position property (relative, absolute, fixed)
   d. CSS Grid and Flexbox layouts
7. **CSS Units and Measurements**
   a. Absolute units (pixels, inches, centimeters)
   b. Relative units (percentages, em, rem)
   c. Viewport units (vw, vh, vmin, vmax)
8. **CSS Transitions and Animations**
   a. Transition properties (duration, delay, timing function)
   b. Transform properties (translate, rotate, scale)
   c. Keyframe animations
9. **Responsive Web Design**
   a. Media queries and breakpoints
   b. Building responsive layouts
   c. Mobile-first vs. desktop-first approach
10. **CSS Frameworks (Optional)**
    a. Introduction to popular CSS frameworks like Bootstrap or Foundation
    b. Utilizing pre-built CSS classes and components
11. **CSS Preprocessors (Optional)**
    a. Introduction to CSS preprocessors like Sass or Less
    b. Nesting, variables, mixins, and functions
12. **CSS Best Practices and Optimization**
    a. Efficient CSS coding techniques
    b. Minification and optimization tools
    c. Browser compatibility and vendor prefixes

Bootstrap

1. **Introduction to Bootstrap**
   a. What is Bootstrap and why is it used?
   b. Benefits of using Bootstrap for web development
   c. Bootstrap grid system and responsive design

2. **Setting Up Bootstrap**
   a. Downloading Bootstrap or using a CDN
   b. Linking Bootstrap CSS and JavaScript files
   c. Understanding the Bootstrap file structure

3. **Bootstrap Grid System**
   a. Creating responsive layouts with rows and columns
   b. Understanding container classes (container, container-fluid)
   c. Applying column sizes and breakpoints

4. **Typography and Text Styling**
   a. Using Bootstrap typography classes
   b. Working with headings, paragraphs, and inline text
   c. Text alignment, emphasis, and transformations

5. **Buttons and Badges**
   a. Creating buttons with different styles and sizes
   b. Button groups and dropdowns
   c. Adding badges to highlight information

6. **Navigation Components**
   a. Building navigation bars (navbar) and menus
   b. Responsive navigation with collapsing menus
   c. Breadcrumbs, pagination, and tabs

7. **Forms and Form Components**
   a. Styling HTML forms with Bootstrap classes
   b. Input fields, checkboxes, radio buttons, and select dropdowns
   c. Form validation and feedback messages

8. **Bootstrap Components**
   a. Working with alerts, badges, and labels
   b. Creating panels and cards
   c. Accordion, modal, and carousel components

9. **Bootstrap Icons and Glyphicons**
   a. Adding icons using Bootstrap Icons or Glyphicons
   b. Icon classes and customization options

10. **Customizing Bootstrap**
   a. Overriding Bootstrap styles with custom CSS
   b. Modifying Bootstrap variables and Sass files
   c. Creating a custom Bootstrap theme

11. **Responsive Design with Bootstrap**
   a. Understanding responsive breakpoints
   b. Hiding and showing elements on different devices
   c. Creating responsive images and media

12. **Bootstrap Best Practices and Resources**
   a. Following Bootstrap coding conventions
   b. Troubleshooting common issues
   c. Additional Bootstrap resources and documentation

JavaScript

1. **Introduction to Javascript**
   a. What is JavaScript and its role in web development?
   b. JavaScript in the browser and on the server (Node.js)
   c. Setting up a development environment

2. **JavaScript Basics**
    a. Variables, data types, and operators
    b. Control flow (if statements, loops, switch statements)
    c. Functions and scope
    d. Working with arrays and objects
3. **DOM Manipulation**
    a. Introduction to the Document Object Model
    b. Selecting and manipulating HTML elements
    c. Modifying content, styles, and attributes
    d. Handling events and event listeners
4. **Working with Functions**
    a. Function declarations and expressions
    b. Parameters and return values
    c. Arrow functions
    d. Higher-order functions and callbacks
5. **Arrays and Iteration**
    a. Array methods (push, pop, shift, unshift, etc.)
    b. Looping through arrays (for loop, forEach, map, filter)
    c. Array manipulation and transformation
6. **Object-Oriented JavaScript**
    a. Object literals and properties
    b. Constructors and the 'new' keyword
    c. Prototypes and inheritance
    d. Classes and ES6 syntax
7. **Asynchronous JavaScript**
    a. Introduction to asynchronous programming
    b. Callback functions and the event loop
    c. Promises and async/await
    d. Fetch API and working with AJAX
8. **Error Handling and Debugging**
    a. Understanding JavaScript errors
    b. Debugging techniques and tools
    c. Error handling with try-catch blocks
    d. Console methods and logging
9. **Working with JSON and APIs**
    a. Introduction to JSON
    b. Making HTTP requests with XMLHttpRequest and Fetch API
    c. Parsing JSON data
    d. Working with RESTful APIs
10. **Browser Storage**
    a. Working with cookies
    b. Local Storage and Session Storage
    c. Storing and retrieving data from the browser
11. **JavaScript Modules and Bundlers**
    a. Organizing code into modules
    b. Import and export statements
    c. Introduction to module bundlers (Webpack, Rollup)
12. **JavaScript Best Practices**
    a. Clean code principles and best practices
    b. Code organization and naming conventions
    c. Performance optimization techniques
    d. Common JavaScript pitfalls and how to avoid them

Node JS

1. **Introduction to Node.js**
    a. What is Node.js and its key features?
    b. Understanding the event-driven, non-blocking architecture
    c. Installing Node.js and setting up a development environment

2. **Node.js Basics**
    a. Working with the Node.js REPL (Read-Eval-Print Loop)
    b. Writing and running simple Node.js scripts
    c. NPM (Node Package Manager) and managing dependencies
3. **Modules and CommonJS**
    a. Introduction to modules and the CommonJS module system
    b. Exporting and importing modules
    c. Core modules vs. external modules
4. **Asynchronous Programming with Node.js**
    a. Understanding the non-blocking I/O model
    b. Callbacks and handling asynchronous operations
    c. Promises and async/await for asynchronous control flow
5. **File System Operations**
    a. Reading and writing files using the fs module
    b. Working with directories and file paths
    c. Synchronous vs. asynchronous file operations
6. **Building HTTP Servers**
    a. Creating an HTTP server using the http module
    b. Handling HTTP requests and responses
    c. Routing and middleware concepts
7. Express.js Framework
    a. Introduction to Express.js as a web application framework
    b. Setting up an Express.js server
    c. Handling routes, requests, and responses
    d. Middleware and error handling
8. **Working with Databases**
    a. Connecting to databases (e.g., MongoDB, MySQL) with Node.js
    b. Performing CRUD operations (Create, Read, Update, Delete)
    c. Using ORMs (Object-Relational Mappers) with Node.js
9. **Working with APIs**
    a. Consuming and integrating with external APIs
    b. Making HTTP requests with Node.js (using axios, node-fetch, etc.)
10. **Authentication and Authorization**
    a. Implementing user authentication and authorization
    b. Using libraries like Passport.js for authentication strategies
    c. Implementing JWT (JSON Web Tokens) for session management
11. **Websockets and Real-time Applications**
    a. Introduction to Websockets and real-time communication
    b. Building a real-time chat application with Socket.io
    c. Broadcasting events and handling real-time updates
12. **Deployment and Scaling**
    a. Deploying Node.js applications to production servers
    b. Configuring environment variables
    c. Load balancing and scaling strategies

Mongo DB

1. **Introduction to MongoDB**
    a. What is MongoDB and its key features?
    b. Understanding NoSQL databases and MongoDB's document-based approach
    c. Installing MongoDB and setting up a development environment

2. **MongoDB Data Modeling**
    a. Document structure in MongoDB
    b. Designing collections and documents
    c. Relationships between documents (embedded vs. referencing)
3. **CRUD Operations in MongoDB**
    a. Creating a database and collections
    b. Inserting documents
    c. Querying documents using find() and operators
    d. Updating and deleting documents
4. **Querying in MongoDB**
    a. Query operators and their usage
    b. Sorting and limiting results
    c. Aggregation framework for complex queries
    d. Indexing and improving query performance
5. **MongoDB and Node.js**
    a. Connecting a Node.js application to MongoDB
    b. Using the MongoDB Node.js driver (or an ORM like Mongoose)
    c. Performing CRUD operations from Node.js

Express

1. **Introduction to Express.js**
    a. What is Express.js and its role in web development?
    b. Understanding the Node.js and Express.js relationship
    c. Installing Express.js and setting up a development environment
2. **Building a Basic Server**
    a. Creating an Express.js server
    b. Handling HTTP requests and responses
    c. Routing and handling different routes
3. **Middleware in Express.js**
    a. Understanding middleware and its role in Express.js
    b. Writing custom middleware functions
    c. Implementing error handling middleware
4. **Routing in Express.js**
    a. Implementing route handlers for different HTTP methods (GET, POST, etc.)
    b. Creating dynamic routes with route parameters
    c. Using query parameters and request body data
5. **Templating Engines**
    a. Introduction to templating engines in Express.js (such as EJS or Handlebars)
    b. Rendering dynamic views and passing data to templates
    c. Layouts, partials, and template inheritance
6. **Working with Forms and Data**
    a. Handling form submissions in Express.js
    b. Validating form data and displaying validation errors
    c. Processing and storing data in a database
7. **Middleware for Authentication and Authorization**
    a. Implementing user authentication with middleware (such as Passport.js)
    b. Managing user sessions and cookies
    c. Protecting routes with authorization middleware
8. **Error Handling and Logging**
    a. Handling errors in Express.js applications
    b. Implementing error logging and debugging techniques
    c. Using third-party error logging tools

9. **RESTful API Development**
   a. Designing and implementing RESTful APIs with Express.js
   b. Defining API routes and handling CRUD operations
   c. Handling authentication and authorization for APIs

10. **File Upload and Download**
   a. Uploading and handling file uploads in Express.js
   b. Downloading files and serving static assets
   c. Working with file storage services (such as Amazon S3)

11. **Advanced Topics in Express.js**
   a. WebSockets and real-time communication with Socket.io
   b. Caching and performance optimization techniques
   c. Implementing pagination and sorting in APIs

12. **Testing and Deployment**
   a. Unit testing and integration testing in Express.js
   b. Deploying Express.js applications to production servers
   c. Configuring environment variables and managing deployment environments

React

1. **Introduction to React**
   a. What is React and its key features?
   b. Understanding React's component-based architecture
   c. Setting up a development environment

2. **JSX and Component Fundamentals**
   a. Introduction to JSX syntax and its relationship to JavaScript
   b. Creating functional and class components
   c. Understanding component lifecycle methods

3. **State and Props**
   a. Managing state within React components
   b. Passing data between components using props
   c. Handling user events and updating state

4. **Handling Forms and User Input**
   a. Creating controlled components for form inputs
   b. Validating form data and handling form submission
   c. Implementing form validation and error handling

5. **Lists and Keys**
   a. Rendering lists of data in React
   b. Using keys for efficient list rendering
   c. Updating lists and handling user interaction

6. **React Router**
   a. Introduction to React Router for client-side routing
   b. Configuring routes and handling navigation
   c. Passing parameters and accessing route data

7. **Component Styling and CSS**
   a. Styling React components using CSS classes and inline styles
   b. Exploring CSS-in-JS libraries (such as styled-components)
   c. Applying component-based styling patterns

8. **React Context**
   a. Managing state and sharing data across components using Context

b. Creating and consuming context providers and consumers

9. **React Hooks**
   a. Introduction to React Hooks (useState, useEffect, useContext, etc.)
   b. Managing state and side effects with Hooks
   c. Custom Hooks and their reuse across components

10. **React and API Integration**
    a. Fetching data from APIs using asynchronous functions
    b. Handling API responses and updating component state
    c. Implementing loading indicators and error handling

11. **React Forms and Validation Libraries**
    a. Working with form libraries in React (such as Formik or React Hook Form)
    b. Simplifying form handling and validation using form libraries

12. **React Best Practices and Performance Optimization**
    a. Writing clean and reusable React code
    b. Performance optimization techniques (memoization, lazy loading, etc.)
    c. Debugging and error handling in React applications

Development Tools

1. **Text Editors and Integrated Development Environments (IDEs):**
   a. Visual Studio Code: A popular and highly customizable text editor with powerful features and extensions for web development.
   b. Sublime Text: A lightweight yet feature-rich text editor with a large user base and extensive plugin ecosystem.
   c. Atom: A hackable text editor built by GitHub, known for its flexibility and extensive package ecosystem.
   d. WebStorm: A full-fledged IDE specifically designed for web development, providing advanced features and tools.

2. **Version Control Systems (VCS):**
   a. Git: A distributed version control system widely used in web development to track and manage changes in code repositories.
   b. GitHub: A popular web-based hosting service for Git repositories, facilitating collaboration, code review, and version control.

3. **Package Managers:**
   a. npm (Node Package Manager): The default package manager for Node.js, used to install and manage JavaScript packages and dependencies.
   b. Yarn: A fast and reliable package manager developed by Facebook, offering enhancements over npm in terms of speed and stability.

4. **Task Runners:**
   a. Gulp: A popular task runner that automates repetitive tasks such as minification, concatenation, and compilation of frontend assets..

b. Grunt: A JavaScript task runner used for automating build processes, running tests, and performing other development tasks

5. **Module Bundlers:**
   a. webpack: A widely used module bundler that bundles JavaScript modules, assets, and resources, enabling efficient loading and deployment of web applications.
   b. Parcel: A zero-config bundler that automatically handles module bundling, code splitting, and asset management without requiring complex configurations.
6. **CSS Preprocessors:**
   a. Sass: A popular CSS preprocessor that extends CSS syntax with features like variables, nesting, mixins, and functions.
   b. Less: Another CSS preprocessor that simplifies writing and organizing CSS code by introducing variables, mixins, and other dynamic features.
7. **Task Runners:**
   a. Babel: A widely used JavaScript compiler that transpiles modern JavaScript code into backward-compatible versions, allowing developers to use the latest language features while ensuring cross-browser compatibility.
8. **Browser Developer Tools:**
   a. Chrome DevTools: Built into the Google Chrome browser, it provides a comprehensive set of tools for debugging, profiling, and inspecting web applications.
   b. Firefox Developer Tools: Similar to Chrome DevTools, it offers a range of debugging and development tools for web developers using the Firefox browser.
9. **Live Reloading and Code Editors:**
   a. BrowserSync: A development server that enables live reloading and synchronization of code changes across multiple devices and browsers.
   b. Live Server: A lightweight development server that provides live reloading functionality for HTML, CSS, and JavaScript changes.
10. **Linters and Code Formatters:**
    a. ESLint: A popular linter that helps enforce consistent coding styles, detect errors, and highlight potential issues in JavaScript code.
    b. Prettier: A code formatter that automatically formats code according to predefined rules, ensuring consistent code style and formatting across projects.

Student Reviews

A testament to the power of determination, hard work, and dedication.
Read what our alumni have to say about their experinece with us.

Komal Kale

★ ★ ★ ★ ★

The instructors at IT SHAALA are undoubtedly experts in their fields. They were not only knowledgeable but also adept at conveying complex concepts in a clear and understandable manner.

Sandesh Dongardive

★ ★ ★ ★ ★

Great learning experience at ITshaala, ITshaala provides quality training and the best infrastructure. I got best guidance from Sandeep sir. IT shaala helped me to shape my carrier as IT professional.

Kishor Pol

★ ★ ★ ★ ★

Had a fanstastic learning experience at IT Shaala, one of the top institute for Backend Developer training. The institute offers excellent training and top-notch infrastructure

Pruthviraj Gaikwad

★ ★ ★ ★ ★

I have been taking classes for a month now and I am getting good knowledge here .Sandeep sir's teaching method is very nice, nonteaching staff is also supportive .If you want to get knowledge then this is the right place.

Student Placements

Vaibhav Kadam
FullStack Developer

Pratik Gole
Software Engineer

Kedar Thakur
Software Engineer

Akshay Jadhav
Backend Developer

Rohan Bhadke
FullStack Developer

Avanti Urkude
FullStack Developer

Vinay Divekar
Sr.Associate Developer

Pratik Yawalkar
Software Developer

Life at IT Shaala

"In the lifecycle of an IT student, each phase represents a stepping stone towards mastery. From the eager anticipation of enrollment to the relentless pursuit of knowledge, and finally, the triumphant transition into a successful career.

"Embark on the journey of IT education with enthusiasm, navigate through challenges with determination, and emerge as a skilled professional ready to make your mark in the digital landscape. "

IT Shaala

Address : Office No - 211, 2nd floor, Kakade Bizz Icon
Shivajinagar, Pune - 15

For any query, Connect us at:

+91 88620 64497 info@itshaala.com