WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

MERN Stack: A Full-Stack Web-Development Solution

Dr. Vishal Shrivastava, Dr. Akhil Pandey, Mrs. Aarti Sharma, Aadarsh Tiwari[1]

B. Tech Scholar[1]
Department: Computer Science Engineering
Email: aadarshtiwari21jul@gmail.com, vishalshrivastava.cs@aryacollege.in, aartisharma.ec@aryacollege.in, akhil@aryacollege.in

ABSTRACT:

The web is one of the most common ways to interact on the Internet. Developing a website from scratch is not an easy task. The MERN stack stands as a prominent choice among web development stacks, primarily due to its user interface rendering and performance, cost-effectiveness, open-source nature, and seamless server transition. It is designed with the primary goal of enhancing overall application performance. The MERN stack facilitates the rapid development of web applications and software. Comprising a robust set of technologies, it empowers the creation of scalable, professional web applications, encompassing front-end, back-end, and database components. The MERN stack is a user-friendly full-stack JavaScript libraries and frameworks, making it the preferred platform for startups. This paper provides an in-depth overview of the MERN stack, encompassing four key technologies: MongoDB, Express.js, React.js, and Node.js. Each of these technologies plays a pivotal role in the development of web applications.

Keywords: MongoDB, Web-Dev, React.js, Node.js, Express.js.

1. Introduction:

M stands for MongoDB, a database system primarily used for managing document data and is a NoSQL (Non-Structured Query Language) database system.

a. M expresses MongoDB, a non-SQL database

b. E expresses Express.js or Express, a framework primarily used for building web applications with Node.js.

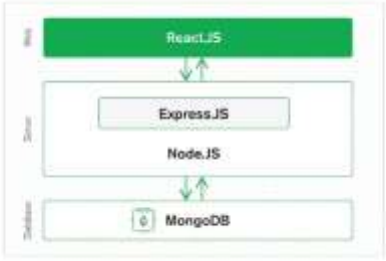
c. R expresses React.js or React, a library primarily used for developing client-side JavaScript applications.

d. N expresses Node.js, a runtime environment primarily used for building server-side JavaScript applications.

All four of these technologies play a significant role in creating a comprehensive framework for developers and are crucial components in the development of web applications.

2. Components Of MERN Stack:

MERN has a 3-tier Architecture System Consisting of 3 Layers



These layers are as follows:

I. Web as front-end layer

II. Server as the middle layer

III. Database as backend

I. WEB AS FRONT-END LAYER

React.js or React, is JavaScript library which is used to create the front-end of web pages. React is widely used because of it the following properties of React, which are as follow:

a. Component-based

b. Declarative

c. JavaScript XML or JSX

These properties make react an optimal choice to create webpages for the web. Another huge advantage of using React is there's huge community of developers using it which makes it easier to find necessary resources to develop

II. SERVER AS MIDDLE LAYER

The server layer is responsible for responses to requests, interactions between web and database as well as processing data. The middle layer or the server layer of MERN stack consists of Node.js and Express.js. Node.js enables us to write server-side logic i.e., how will server serve/respond to a particular HTTP request. On the other hand, Express.js or Express which is framework for web of Node enables building scalable server-side applications.

The server-side logic handles HTTP requests (GET, PUT, DELETE & POST), authorization, authentication and interacting with the database to serve the Front-end of the website.

III. DATABASE AS BACKEND LAYER:

Within the MERN Stack, one of the core components is MongoDB, a database system responsible for storing various types of application data, such as text, metrics, user profiles, comments, and more. Its primary role is to safeguard and manage all application-related information. MongoDB efficiently maintains a structured record of data, making it readily accessible to users when needed.

What sets MongoDB apart is its departure from traditional, table-based relational databases. Instead, it offers a unique approach to data storage and retrieval. MongoDB is a popular choice among NoSQL (Non-SQL or Non-Structured Query Language) databases, offering an open-source, document-oriented database solution. As a NoSQL database, MongoDB doesn't rely on rigid table structures, opting for a more flexible format that differs from traditional relational databases with rows and columns.

3. MongoDB:

I. MongoDB is an open-source cross-platform document-oriented database program. Classified as a NoSQL database system. It uses JSON-like documents with optional schemas.

II. Key Features of MongoDB are the following:

a. **Schema:**
New data fields can be easily added in the MongoDB without changing the structure of the entire database.

b. **JSON-Like Document:**

In a JSON-like Document database, a data structure is a key value pair of complex structure and their respective values. Which makes it easier to store and access the values of to their corresponding keys.

c. **Scalability:**
MongoDB is widely used for scaling applications; it can be scaled by adding the new servers in MongoDB clusters. Which makes it easier for large scale applications.

III. The query language used in MongoDB is called MQL (MongoDB Query Language). Similar to SQL (Structured Query Language) MQL works like relational database however it is built to work with the JSON-Documents. Example:

{

Name: "XYZ",

EmployeeID: 1,

```
    Groups: [ “Web-Development”, “JavaScript”],
}
```

4. Express.js:

II. Express.js is an open-source, minimalist web application framework for Node.js which provides a set of features for building web and mobile applications. It is a very popular choice among developers because of its open-source nature and huge community support.

III. Key features of Express.js are the following:

a. **API Creation:**

APIs with a bunch of HTTP utility strategies and middleware available to you, making a powerful API is fast and simple.

b. **Performance:**
Express provides a thin layer of fundamental web application features, without obscuring widely used Node.js features are often utilized.

c. **Easy to use:**

Express.js is an easy-to-use Node.js framework because it requires only a basic understanding of backend development i.e Node and JavaScript.

IV. A handler is a function that is called to process a request. Express.js provides a variety of methods for defining handlers, including the following:

a. **app.get():** A handler for a GET request.

b. **app.post():** A handler for a POST request.

c. **app.put():** A PUT request.

d. **app.delete():** A DELETE request.

Example:

```
const express = require('express');

const app = express();

app.get('/', (req, res) => {
    res.send('Get request');
});
```

5. React.js:

V. ReactJS is an open-source JavaScript library for building user interfaces in a more flexible and reusable manner (UIs). ReactJS is used to build single-page applications (SPAs), mobile applications, and traditional web applications.

VI. Key feature of React.js or React are the following:

a. **Component-based:**
React components are JavaScript functions. Each react component is part of the application which has it’s own presentation and logic, can be reused. This makes React code modular design and easy to read.

b. **Declarative:**
Unlike imperative programming, declarative programming emphasizes describing the desired state of the UI. This makes updating code more efficient, cleaner, and maintainable.

c. **JSX:**
it stands for JavaScript XML

React is not just a library; it’s also an architecture and a framework that lets you fetch data in asynchronous component that run on the server. In other words, data from database can be passed down to the components of the frontend of the web applications.

Example:

```
import React, { Component } from 'react';
class DataFetchingComponent extends Component {
 constructor() {
  super();
  this.state = {
   data: [],
   loading: true,
   error: null,
  };
 }
 async componentDidMount() {
  try {
   const response = await fetch('https://api.example.com/data'); // Replace with your API endpoint
   if (!response.ok) {
    throw new Error('Network response was not ok');
   }
   const data = await response.json();
   this.setState({ data, loading: false });
  } catch (error) {
   this.setState({ loading: false, error });
  }
 }  render() {
  const { data, loading, error } = this.state;
  if (loading) {
   return <div>Loading...</div>;
  }
  if (error) {
   return <div>Error: {error.message}</div>;
  }
  return (
   <div>
    <h1>Data from the Database</h1>
    <ul>
     {data.map((item) => (
      <li key={item.id}>{item.name}</li>
     ))}
    </ul>
```

```
    </div>
  );
 }
}
export default DataFetchingComponent;
```

6. Node.js:

VII. As an asynchronous event-driven JavaScript runtime, Node.js is designed to build scalable network applications. Thread-based networking is relatively inefficient and very difficult to use. Furthermore, users of Node.js are free from worries of dead-locking the process since there are no locks.

VIII. Key features of Node are the following:

a. **Non-blocking, Asynchronous I/O:**

Node.js is designed to handle asynchronous I/O operations efficiently. It uses an event-driven, non-blocking architecture, which makes it well-suited for handling many concurrent connections, making it ideal for building real-time applications like chat applications, online games, and streaming services.

b. **Cross-Platform:**
Node.js is cross-platform and can run on various operating systems, including Windows, macOS or OS X, and Linux. This portability allows developers to write code that works across different environments and makes code machine independent.

c. **Fast Execution:**

Node.js is known for its performance due to the V8 JavaScript engine, which compiles JavaScript code to machine code leads to faster execution. It's particularly suitable for applications that require low latency and high throughput.

IX. Node.js is used to create web servers. Many popular web server frameworks like int this case Express.js make it easy to handle HTTP requests and responses, route URLs, and perform various server-side tasks.

7. References:

1. "Fullstack React and Node.js: Build Interactive Web Apps with a Modern JavaScript Stack" by Chad Fowler and Brad Traversy (2019). "The MERN Stack: Full-Stack Development with MongoDB, Express, React, and Node.js" by Donovan Hutchinson (2019). "MERN Stack: A Complete Guide to Building Modern Web Applications with MongoDB, Express, React, and Node.js" by David Katz (2019).
2. "Fullstack React and Node.js: Build Interactive Web Apps with a Modern JavaScript Stack" by Chad Fowler and Brad Traversy (2019)/ "The MERN Stack: Full-Stack Development with MongoDB, Express, React, and Node.js" by Donovan Hutchinson (2019).
3. "MERN Stack: A Complete Guide to Building Modern Web Applications with MongoDB, Express, React, and Node.js" by David Katz (2019)
4. https://www.mongodb.com/mern-stack
5. https://expressjs.com
6. https://react.dev/
7. https://nodejs.org/en/about