MERN-AUTH DOCUMENTATION  
By Daniel Mathews -10/2/2024

1. **Dependencies and frameworks used:**

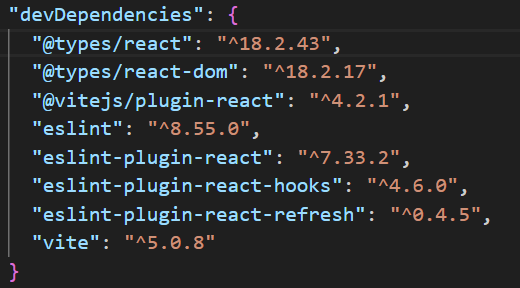
**Server side:**

* NodeJS
* express
* bcrypt
* cookie-parser
* cors
* dotenv
* jsonwebtoken
* mongoose
* nodemon

**Client side:**

* VITE Boilerplate
* ReactJS
* axios
* react-dom
* react-router-dom
* react-hot-toast

**Developer dependencies for client side:**

****

1. **Framework and Dependencies (Details):**
2. **NodeJS:**

With Node.js, you can write this server-side code using JavaScript, which is usually associated with web browsers. But with Node.js, you can use JavaScript to create servers too.

Here's how it works:

* Setting Up: First, you install Node.js on your computer. It's like installing a special tool that lets you use JavaScript for server-side tasks.
* Writing Code: Then, you write JavaScript code that defines how your server should behave. You can create routes, handle requests, and send responses, just like you would with other server technologies.
* Running the Server: After you've written your server code, you use Node.js to run it. This starts up your server, so it's ready to handle requests from other computers or devices.
* Handling Requests: When someone sends a request to your server (like visiting a website hosted by your server), your server's code processes that request and sends back a response. This could be a webpage, data, or anything else your server is programmed to do.

So, in simple terms, Node.js lets you use JavaScript to build and run servers, allowing you to create powerful applications that can handle requests from users all around the world.

1. **Express:**

Imagine you're building a house with Node.js as your toolbox. Node.js gives you the basic tools to create the structure, but if you want to add rooms, windows, and doors quickly and easily, you might use Express.

Express is like a set of blueprints and pre-made parts for your house. It provides a framework or a set of guidelines that helps you organize and build your web applications (or your "house") more efficiently.

Here's how Express helps:

* Routing: Express helps you set up paths or routes in your application, like telling visitors where to go when they type in a certain web address.
* Middleware: It allows you to plug in additional functionality to your application's request-response cycle. For example, you might use middleware to handle authentication, logging, or parsing incoming data.
* Template Engines: Express can work with template engines, which are tools that help you dynamically generate HTML pages. This makes it easier to create dynamic web pages that change based on user input or other factors.
* Simplified Code: With Express, you often write less code to achieve the same functionality compared to using just Node.js. It provides shortcuts and conventions that make common tasks easier and faster.

So, in simple terms, Express is like an assistant that helps you build your web application faster and with less effort by providing ready-made solutions for common tasks like routing, middleware, and working with templates. It's like having a set of pre-designed rooms and features for your house-building project, so you can focus on the creative aspects rather than reinventing the wheel for each room.

1. **Bcrypt**

Imagine you have a secret treasure chest, and you want to keep it safe from thieves. Bcrypt is like a special lock that you put on your treasure chest to keep it secure. It is essential for making secure login applications.

Here's how it works:

* Encrypting Passwords: Bcrypt is often used to encrypt passwords. When a user creates an account on a website, their password is scrambled and turned into a bunch of gibberish using a special algorithm. This scrambled version is stored in the website's database instead of the actual password.
* Protecting Against Thieves: If a thief somehow gets into the database and steals the passwords, they'll only see the scrambled versions. It's extremely difficult for them to unscramble these passwords back into their original form.
* Checking Passwords: When a user tries to log in, the password they enter is also scrambled using the same algorithm. The website then checks if this scrambled version matches the one stored in the database. If they match, the user is granted access.
* Adding Extra Security: Bcrypt is designed to be really hard for attackers to crack. It uses techniques that slow down the process of unscrambling passwords, making it much harder for hackers to break in.

So, in simple terms, Bcrypt is like a super-strong lock that keeps passwords safe from thieves by turning them into unreadable gibberish. It adds an extra layer of security to websites and applications, helping to protect users' accounts and personal information.

1. **Cookie-parser**

Imagine you're hosting a party and you want to give each guest a special sticker so you can recognize them when they come back. Cookie-parser is like the person who hands out these stickers and helps you remember who's who.

Here's how it works:

* Identifying Guests: When someone comes to your party (or visits your website), cookie-parser helps you give them a special sticker (a cookie). This sticker contains information like their name or preferences.
* Storing Information: The cookie-parser helps you store this sticker (cookie) on the guest's computer or device. This way, even if they leave the party (or close the website) and come back later, you still recognize them by the sticker.
* Retrieving Information: When the guest comes back, cookie-parser helps you check their sticker (cookie) to see if they've been to the party before (visited the website). If they have, you can use the information on the sticker to remember their name or preferences.
* Customizing Experience: With cookie-parser, you can use the information from the sticker (cookie) to customize the guest's experience. For example, you might greet them by name or show them content based on their preferences.

So, in simple terms, cookie-parser is like the host of the party who hands out special stickers (cookies) to guests and helps remember who's who. It's a tool that allows websites to remember users and personalize their experience based on their past interactions.

1. **Cors**

Imagine you're hosting a party, and you have a rule that only people from your neighborhood can come in. CORS (Cross-Origin Resource Sharing) is like a bouncer who enforces this rule to keep your party safe.

Here's how it works:

* Defining the Neighborhood: Just like you define your neighborhood, websites also have a domain or "neighborhood." This is the place where all their content lives.
* Enforcing the Rule: When someone from a different neighborhood tries to access your party (or resources on your website) – that's a different domain or origin – CORS steps in. It checks if the request is coming from an allowed neighborhood.
* Allowing Access: If the request is from an allowed neighborhood, CORS lets it in without any trouble. But if it's from a different neighborhood, CORS blocks it to prevent any potential problems or security risks.
* Keeping Things Safe: By enforcing this rule, CORS helps protect your website from unauthorized access or malicious activities that could potentially harm you or your guests.

So, in simple terms, CORS is like a bouncer for your website, making sure that only requests from allowed neighborhoods (domains) can access your content. Cors allows cross-origin requests from a specific frontend server (http://localhost:5173) to access the resources on the backend server where the cors middleware is applied. It also allows credentials to be included with these cross-origin requests. It's a security measure that helps keep your website safe and secure.

1. **Dotenv**

Imagine you're working on a secret project, and you need to store sensitive information, like passwords or API keys. dotenv is like a secure vault where you can keep these secrets hidden from prying eyes.

Here's how it works:

* Storing Secrets: You put your sensitive information, like passwords or API keys, into a special file called ‘.env’. This file is like a secret notebook where you keep all your important stuff.
* Keeping Secrets Safe: dotenv helps you load these secrets into your code so that your application can use them. But here's the clever part: dotenv encrypts this information and keeps it hidden from anyone who shouldn't see it.
* Using Secrets in Your Code: Once dotenv has loaded the secrets from your .env file, you can access them in your code just like any other variables. For example, you might use them to connect to a database or make secure API calls.
* Protecting Your Project: By using dotenv, you keep your sensitive information separate from your codebase, which makes it much harder for hackers or unauthorized users to access it. It's like keeping your valuables in a locked safe instead of leaving them out in the open.

So, in simple terms, dotenv is like a secret vault where you can store sensitive information for your project. It helps keep your secrets safe and your project secure by encrypting and managing important data outside of your code.

1. **Jsonwebtoken**

Imagine you're sending a package to a friend, but you want to make sure only your friend can open it. JSON Web Token (JWT) is like a special lock that you put on the package to ensure it can only be opened by the right person.

Here's how it works:

* Creating a Token: When your friend logs in to your website or application, you create a JWT containing some information about them, like their username or user ID. You then seal this information in the JWT.
* Adding Security: JWT adds an extra layer of security by encoding this information using a special algorithm. This makes it impossible for anyone else to tamper with the data inside the token.
* Sending the Token: Once the JWT is created, you send it to your friend's browser or device. It's like attaching the special lock to the package before sending it off.
* Verifying Identity: When your friend sends a request back to your server, they include the JWT. Your server can then use the same special lock to verify that the JWT hasn't been tampered with and that it came from your friend.
* Granting Access: If everything checks out, your server unlocks the JWT and reads the information inside. This allows your server to recognize your friend and grant them access to their account or certain features of your website or application.

So, in simple terms, JSON Web Token (JWT) is like a special lock that ensures only authorized users can access certain parts of your website or application. It adds an extra layer of security by encoding user information in a tamper-proof way, making it a reliable method for verifying identity and granting access to authorized users.

1. **Mongoose**

Imagine you're building a house, and you need a blueprint to follow. Mongoose is like an architect who helps you design and build your house, but specifically for working with data and databases in your Node.js application.

Here's how it works:

* Designing Data Models: Just like an architect creates blueprints for your house, with Mongoose, you define the structure of your data by creating models. These models represent the different types of data you'll be working with, like users, products, or articles.
* Interacting with Databases: Mongoose acts as a bridge between your Node.js application and your MongoDB database. It provides you with tools and methods to easily perform common database operations, like saving, updating, retrieving, and deleting data.
* Validation and Middleware: Mongoose helps you enforce rules and validate the data before it's saved to the database. You can define rules such as required fields, data types, and custom validation logic. Additionally, Mongoose allows you to define middleware functions that run before or after certain database operations, giving you more control over how data is handled.
* Simplifying Complexities: Working directly with databases can be complex, especially when dealing with relationships between different types of data. Mongoose simplifies this process by providing features like schema definitions, built-in data types, and methods for handling relationships between models.
* Enhancing Productivity: By abstracting away the complexities of working with databases, Mongoose allows developers to be more productive and focus on building features for their applications rather than dealing with low-level database operations.

In simple terms, Mongoose is like an architect for your data in a Node.js application. It helps you design, manage, and interact with your database in a structured and efficient way, making it easier to build robust and scalable applications.

1. **Nodemon**

Imagine you're cooking in the kitchen and you have a magical helper who keeps an eye on your pot. Whenever you make a change to the recipe, this helper automatically tastes the food and tells you if it's ready or if you need to adjust something. That's what nodemon does, but for your Node.js applications.

Here's how it works:

* Automatically Restarting: When you're developing a Node.js application, you're constantly making changes to your code. Nodemon is like that magical helper who watches your code files. Whenever you make a change and save your file, nodemon automatically restarts your Node.js application.
* Instant Feedback: Just like how your magical helper gives you instant feedback on your cooking, nodemon gives you instant feedback on your code changes. You don't have to manually stop and restart your application every time you make a change – nodemon does it for you, so you can see the results of your changes immediately.
* Saves Time and Effort: Nodemon saves you time and effort by eliminating the need to constantly stop and restart your application during development. This means you can focus more on writing code and testing your changes, rather than managing the development server.

In simple terms, nodemon is like a helpful assistant for your Node.js development. It automatically restarts your application whenever you make changes to your code, giving you instant feedback and saving you time and effort during the development process.

1. Axios

Imagine you want to send a letter to a friend who lives far away. You could either write and mail the letter yourself, or you could ask a reliable delivery service to handle it for you. Axios is like that reliable delivery service for sending requests to servers when you're building a web application.

Here's how it works:

* Sending Requests: Just like you want to send a letter to your friend, in web development, you often need to send requests to servers to get data or send data back and forth. Axios helps you do this by providing a simple and easy-to-use way to make HTTP requests from your web application.
* Handling Responses: Once you've sent a request, you'll usually get a response back from the server. Axios helps you handle these responses by giving you access to the data returned by the server, so you can use it in your application.
* Error Handling: Sometimes things go wrong, like if your letter gets lost in the mail. Axios helps you handle errors gracefully by providing mechanisms to catch and handle errors that occur during the request process.
* Convenience and Flexibility: Axios is known for its simplicity and flexibility. It supports various features like sending data with different request methods (GET, POST, PUT, DELETE), setting headers, handling authentication, and more, making it a versatile tool for interacting with servers.

In simple terms, Axios is like a trusted delivery service for your web application. It helps you send requests to servers, handle responses, and manage errors, making it easier for your application to communicate with other parts of the internet.

1. **React-DOM**

* React: React is a JavaScript library for building user interfaces, commonly used for creating interactive web applications.
* DOM: DOM stands for Document Object Model. It's a programming interface for web documents. In simple terms, it's a structured representation of your HTML document that your browser creates when it loads a webpage. This model allows JavaScript to access and manipulate the elements of the webpage.

Now, onto react-dom:

* React-DOM: React-DOM is a package that serves as the bridge between React and the DOM. It's responsible for taking the virtual representation of your React components and rendering them into the actual DOM elements that you see on the webpage.

In simpler terms, think of React-DOM as the translator between React's virtual world (where components are just JavaScript objects) and the real world of the web browser's DOM (where elements are displayed on the webpage). It helps React components come to life on the screen by translating them into HTML elements that your browser understands and displays.

1. **React-router-DOM**

React-Router-DOM: React-Router-DOM is a package that extends React Router to be compatible with web applications built using React and the DOM. It includes components like <BrowserRouter>, <Route>, <Switch>, and <Link> that allow you to set up routing within your React application and manage navigation between different views or pages in a way that's optimized for web development.

In simpler terms, React-Router-DOM is like a navigation system for your React web application. It helps you set up routes and links so that users can navigate between different pages or views within your app without having to reload the entire page. It's essential for building single-page applications (SPAs) where content changes dynamically without full page reloads.

1. **React-hot-toast**

React: React is a JavaScript library for building user interfaces, commonly used for creating interactive web applications.

Toast: In web development, a toast is a small, temporary notification that appears on the screen to provide feedback to the user. It typically contains a short message or alert.

Now, onto react-hot-toast:

* React-Hot-Toast: React-Hot-Toast is a package that makes it easy to create and display toast notifications in React applications. It provides a set of components and functions that allow you to quickly show notifications to users in response to certain events or actions within your application.
* Hot: The "hot" in react-hot-toast refers to the feature of displaying toast notifications in a visually appealing and dynamic way, typically with animations or transitions. It's like serving "hot" notifications straight to the user's screen.

In simpler terms, React-Hot-Toast is like a tool that helps you quickly and easily show temporary notifications (toasts) to users in your React application. These notifications can be used to provide feedback, alerts, or updates, enhancing the user experience by keeping them informed about important events or actions within the app.