**How Does a Browser Work? | Engineering Side**

**Browser Architecture**

🌐 **Browser Engine**  
The browser engine is a critical component that manages the entire rendering pipeline and executes JavaScript code. It has two main subsystems:

1. **Rendering Engine:**
   * Handles HTML, CSS, and JS for rendering and layout.
   * Converts raw bytes into visual elements on the screen.
2. **JavaScript Engine:**
   * Executes JavaScript code in web pages.
   * Optimized for performance with features like Just-In-Time (JIT) compilation.

**Rendering Process**

🔄 The rendering process involves multiple stages to transform raw web resources into a visible page:

1. **Loading Resources:**
   * Browser sends HTTP requests to fetch HTML, CSS, JS, and images.
2. **Parsing:**
   * **HTML:** Tokenized into individual elements and nodes to form the **DOM (Document Object Model)**.
   * **CSS:** Parsed into the **CSSOM (CSS Object Model)**.
3. **Combining Models:**
   * DOM and CSSOM are merged to create a **Render Tree**, which includes only the elements to be displayed on the screen.
4. **Layout (Reflow):**
   * The browser calculates the size and position of each element based on the render tree.
5. **Painting:**
   * Elements are converted to pixels and drawn on the screen.
6. **Compositing:**
   * Layers are processed for final rendering, ensuring smooth transitions and animations.

**JavaScript Execution**

* Uses the **Event Loop** to manage asynchronous tasks while keeping the page responsive.
* Execution occurs in a single thread but supports asynchronous events (e.g., timers, API calls).

**Optimization Techniques**

* **Caching:** Stores frequently accessed resources to reduce load times.
* **Lazy Loading:** Loads resources only when required.
* **Preloading:** Prioritizes critical resources for faster initial rendering.

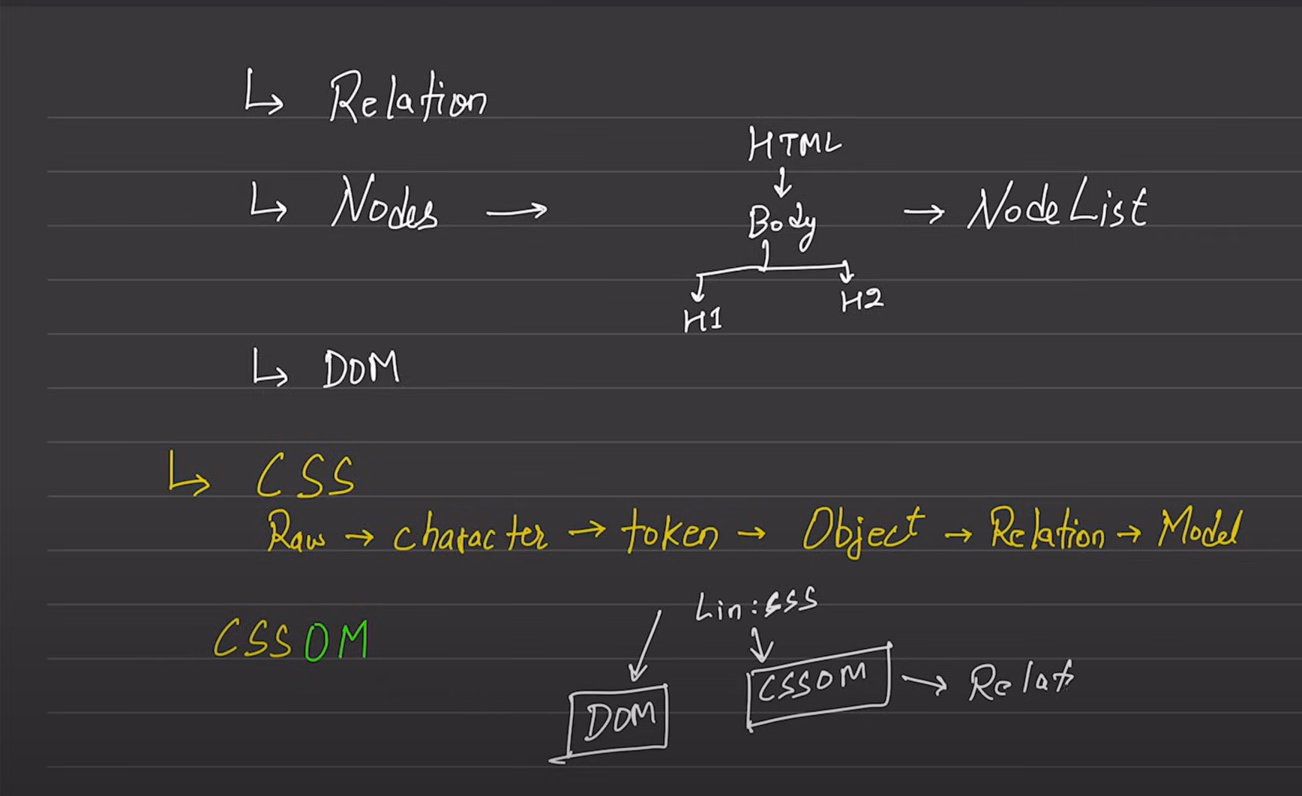
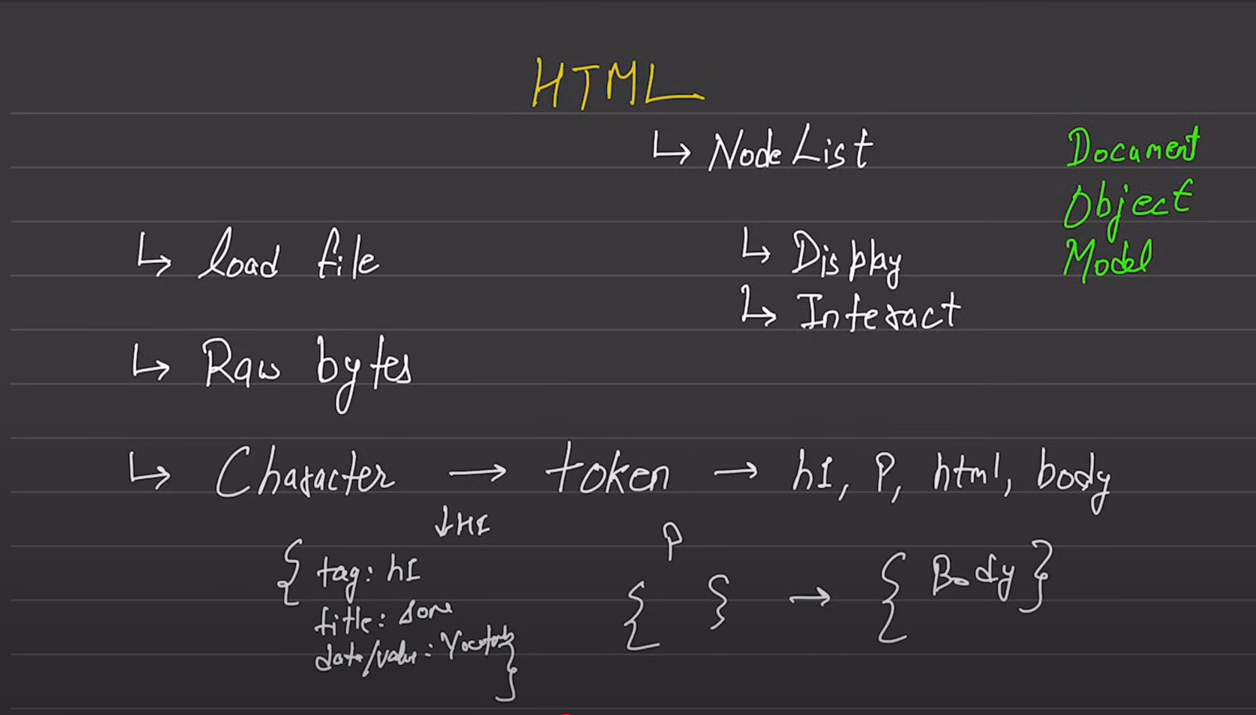
**Security Measures**

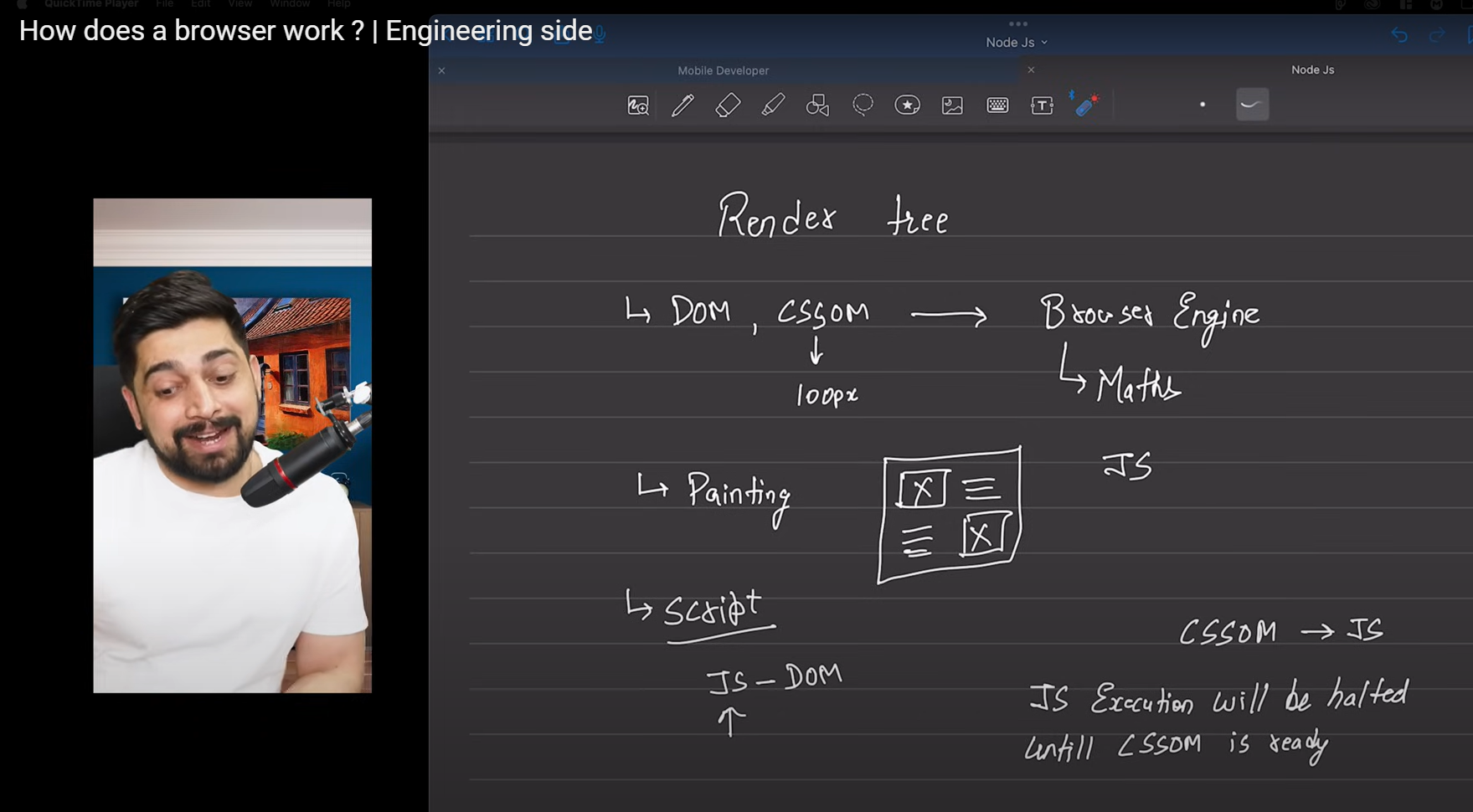
* **Same-Origin Policy:** Prevents unauthorized access between different domains.
* **Sandboxing:** Isolates web pages to prevent malicious code execution.

**Key Takeaways**

Browsers are sophisticated tools that combine rendering, networking, and execution engines to provide a seamless user experience. Understanding their architecture and processes can help developers optimize web applications for better performance and security.

For more insights, watch the video: [How Does a Browser Work? | Engineering Side](https://youtu.be/5rLFYtXHo9s).



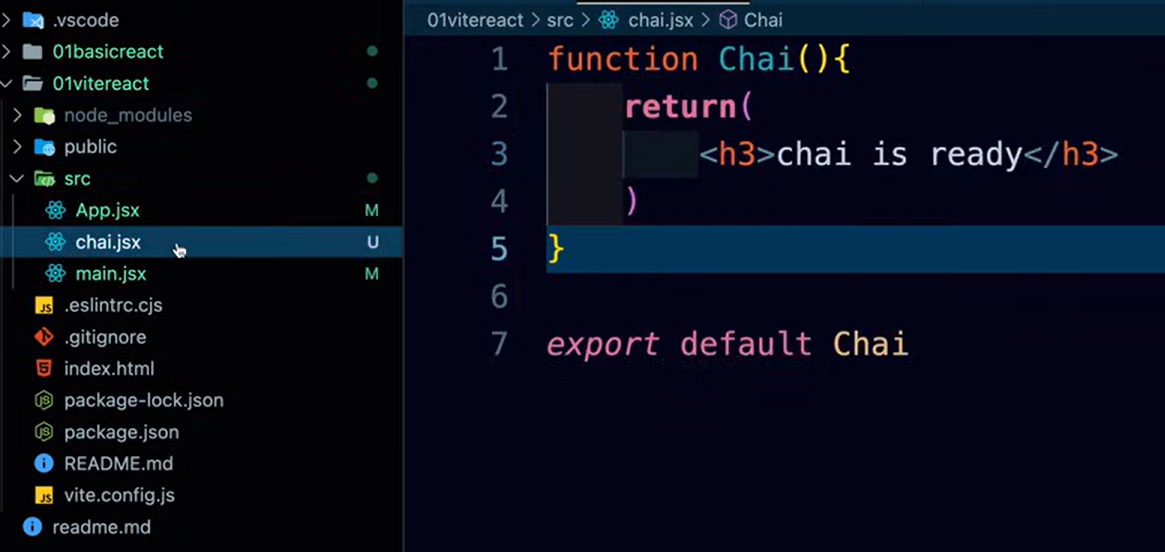


React is a library of javascript only

Vite or create react these are one of the two methods for building react projects

Dono main hii jab function banayenge to capital letter se start krna hai

Vite main components --- js file ko .jsx se save krna padega it is rule



HOOKS are used for UI updation!

createROOT ek DOM create krta hai jaise browser khudke liye ek DOM create krta haai waise hii

virtual DOM main wo compare krta hai main DOM se and jo jo changes huye hain sirf unhi ko change krta hai , browser har baar poora DOM create krta hai



JSX se DOM elements bante hain

Behind the scenes of virtual DOM is reconciliation algorithm only

**CREATING REACT PROJECT**

**GO TO VITE**

**GET STARTED**

**COPY CODE ON TERMINAL -npm create vite@latest**

**NAME PROJ**

**SELECT framework and variant**

**DONE**

**Inserting tailwind steps  
  
npm install -D tailwindcss postcss autoprefixer**

**npx tailwindcss init -p**

**Add the paths to all of your template files in your tailwind.config.js file.  
content: [**

**"./index.html",**

**"./src/\*\*/\*.{js,ts,jsx,tsx}",**

**]**

**Add the @tailwind directives for each of Tailwind’s layers to your ./src/index.css file.**

**@tailwind base;**

**@tailwind components;**

**@tailwind utilities;**

**DONE**

**PROPS**  are used for component reusability

To run a proj   
**npm run dev**

**Call back function in JS**

()=>{}

\*\* do not forget KEYS while using loop in jsx