Chapter 01 - Inception

[Link to my Code](https://bitbucket.org/namastedev/namaste-react-live/commits/928e6faee91549739adc0c3c97c8f0352a55607e) Theory -

* What is Emmet?

Ans: Emmet Abbreviation saves you time and effort while allowing you to focus on more important matters. By converting acronyms into blocks of structured code in (X)HTML, CSS, XML, XSL, and JSP, Emmet speeds up coding.

Example:



* Difference between a Library and Framework?

Ans: In general, libraries tell you what to do and frameworks let you know what to do. Frameworks are better than libraries, or vice versa; however, it is ultimately a matter of use cases and situations rather than the tool itself.

A framework can relieve you of the headache of dependency trees, what to use and what not to use, how to scaffold your application, and how to design it, but it doesn’t provide you with complete control over your application. It’s very useful if you need to develop something quickly for a client. Using libraries, on the other hand, allows you to design your own application tailored specifically to your needs; however, maintaining dependencies, updating individual libraries, and identifying if one is broken due to another, can be extremely time-consuming. Programming is a complex world, and while neither pattern is inherently better, you need to know which pattern is appropriate for the problem at hand.

Which is better, a library or a framework?

You can probably already see if you have worked with both that normally, code libraries are used to solve a specific problem or to add a specific feature to your program. Frameworks, on the other hand, provide a much more generic and reusable approach. While neither pattern is inherently better, you need to know which pattern is appropriate for the problem at hand.

What is the difference between JS (JavaScript) libraries and frameworks?

JavaScript libraries differ from frameworks in that they contain prewritten code snippets that can be used (and reused) for common JavaScript functions. Unlike JavaScript libraries, JavaScript frameworks are a full toolkit for shaping and organizing your web application. Frameworks describe how developers design applications.

Is flask a framework or library?

Flask is a micro web framework written in Python that does not require any specific tools or libraries.

Is Ajax a framework?

Yes, an Ajax framework is a cross-browser framework that allows developers to easily develop Ajax-based internet applications.

* What is CDN? Why do we use it?

Ans: A content delivery network (CDN) is a network of interconnected servers that speeds up webpage loading for data-heavy applications. CDN can stand for content delivery network or content distribution network. When a user visits a website, data from that website's server has to travel across the internet to reach the user's computer. If the user is located far from that server, it will take a long time to load a large file, such as a video or website image. Instead, the website content is stored on CDN servers geographically closer to the users and reaches their computers much faster.

Why is a CDN important?

The primary purpose of a content delivery network (CDN) is to reduce latency, or reduce the delay in communication created by a network's design. Because of the global and complex nature of the internet, communication traffic between websites (servers) and their users (clients) has to move over large physical distances. The communication is also two-way, with requests going from the client to the server and responses coming back.

A CDN improves efficiency by introducing intermediary servers between the client and the website server. These CDN servers manage some of the client-server communications. They decrease web traffic to the web server, reduce bandwidth consumption, and improve the user experience of your applications.

* Why is React known as React?

Ans: React is a JavaScript library for building user interfaces. It was developed by Facebook, and the name “React” was chosen because it is meant to help developers build user interfaces that are fast and responsive, or “reactive.” The library was designed to “react” to changes in data.

When data in a React application changes, the components that depend on that data are automatically updated, which allows for efficient and seamless updates to the user interface. The name “React” reflects this reactive nature of the library.

The idea behind React is to build reusable components that can be rendered on the front-end, rather than writing a new piece of code every time you need to display something on the screen. This makes it easier and more efficient to build complex user interfaces.

* What is crossorigin in script tag?

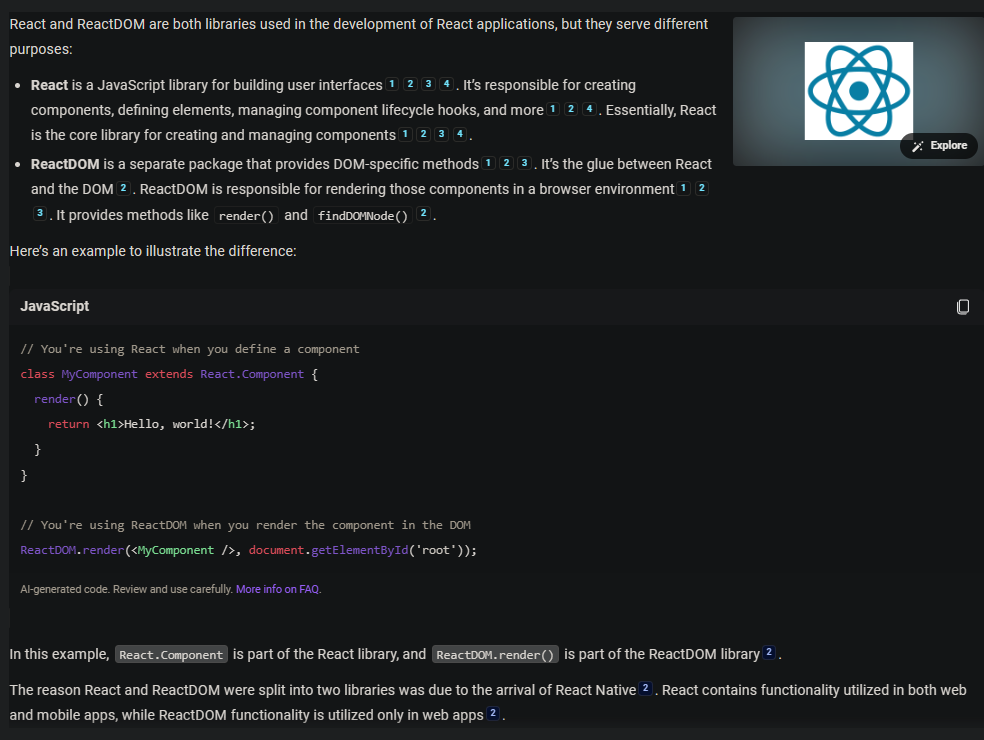
Ans:

A screenshot of a black screen

Description automatically generated

* What is diference between React and ReactDOM

Ans:



* What is difference between react.development.js and react.production.js files via CDN?

Ans:

A screenshot of a computer

Description automatically generated

* What is async and defer?

Ans:

A screenshot of a computer

Description automatically generated

Example:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

<meta http-equiv="X-UA-Compatible" content="IE=edge" />

<meta name="viewport" content="width=device-width, initial-scale=1.0" />

<title>Namaste React</title>

<link rel="stylesheet" href="index.css" />

</head>

<body>

<div id="root1">Not Rendered root1</div>

<div id="root">Not Rendered root</div>

<script

crossorigin

src="https://unpkg.com/react@18/umd/react.development.js"

></script>

<script

crossorigin

src="https://unpkg.com/react-dom@18/umd/react-dom.development.js"

></script>

<script defer>

const heading = React.createElement(

"h1",

{

id: "title",

},

"Hello"

);

const heading2 = React.createElement(

"h2",

{

id: "title",

},

"React"

);

const container = React.createElement(

"div",

{

id: "container",

},

[heading, heading2]

);

console.log(heading);

const root = ReactDOM.createRoot(document.getElementById("root1"));

//passing a react element inside the root

//async defer

root.render(container);

</script>

<script src="App.js"></script>

</body>

</html>

Refer below image for explanation

A screenshot of a computer

Description automatically generated

-----------------------------------------------------END---------------------------------------------------------