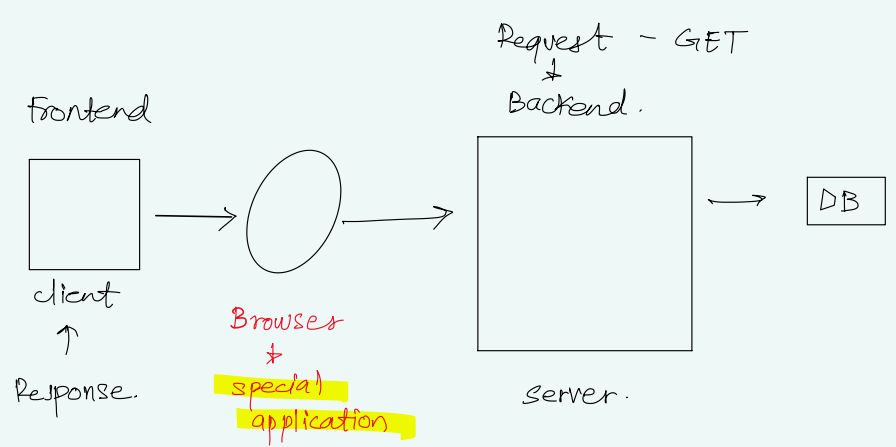
# DAY 1

Monday, 5 May 2025 1:41 AM

React → Library

React → Facebook.
SPA → Single page Application

Freelancing → React



Backend → JS + Express / Java spring / C# Dot net.

Frontend
HTML + CSS + JS → MPA / SPA
HTML → Designing
CSS → Decoration.
JS → (logic) Client side programming.

React → Facebook
Angular → Google → Typescript.
Vue JS → Alibaba.

React is way better than Angular.

* React is client-side library used to develop SPA application using JS. **

React → very simple.

Angular → Framework

- will learn more than syllabus.

*SPA → does not reload often.

CDN → in React

* JSX → JS + XML (HTML)
Markup language → which uses tags are called Markup.
JSX → Imp. to understand React.

Using JSX we create components.
components:
- Used to create UI
- reusable entity used to create User interface.

- Two types
- class based → stateful
- Function based. → stateless. → Hook → state

value stored in component → state.

* React Props → Properties (metadata used in component) → Read only

Interview Qs.

* React state → state used to store metadata inside component.
- Using React Hook.
→ Read writable.

* React Hook → special functions starts with 'use'

- use State(), use Effect(), use Location()

Context vs Redux → global store management
Eg. Update in amazon cart.
Context:
- API
- use Context()
- To share information
Eg. Login

Handling User Inputs

React Router → used for switching b/w components.

Route → method + path + Event Handler.
method → Req, Response. get method
path → url

Advanced Router (Not in syllabus) — Nested routing, Parameterized routing.

Fetch, axios

* Consuming REST APIs
REST → Representational State Transfer

* Server communication
- REST → JSON / XML / YAML → Design Pattern.
- SOAP → Simple Object Access Protocol → set of instructions.
- Graph QL
* (Graph Query Language) — By Meta.
- gRPC → google Remote Procedure call → protobuff
- Websocket → uses socket → method / address for communication

Uploading Files → Form Data

Styled components → component + style (CSS) → Reusability

- checking user (authentication) Eg. 2 factor

Authorization → permission given to authenticated user.
→ JWT Tokens → session storage / local

* Session Handling → Session storage + local storage

* Pagination → limit, skip on server + API.
- Eg. Amazon returns 20 results for more click on 2nd page.

* Cloud Deployment
Cloud → AWS
Deployment → making the application accessible to end-user (client)

Rule 1 : Be on Time. Doors will be closed.
Rule 2 : No Nap.
Rule 3 : Practice.

Kubernetes → Best thing to learn.

CompTIA Certified.

golang → developed by google.

Kotlin → Jetbrains.

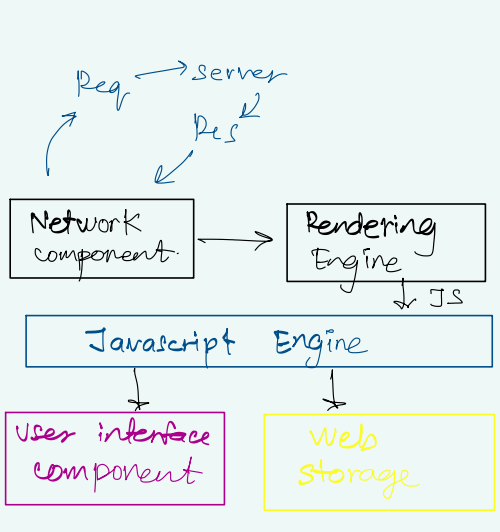
* React → Javascript Library → Less memory Foot print
It requires less memory to run the application.
→ thats why better than Angular.
React → for developing SPA.
- By Meta.

- Efficient rendering (than Angular)

DOM - Document object Model → Tree like structure of objects of each HTML elements in a page.
→ Used to render HTML in JS.

JS → Document object

* Browser architecture. → based on JS Engine.



Network component — Used to communicate with server. (sending Request & Receiving response)

sending php

Server → software ✓
Hardware ×
→ It is an application / software.

Types:-

① Web server → Used to serve http/https
Eg. Apache, nginx, tomcat, express., IIS
tomcat → Java → compiles & returns HTML, CSS, JS
tomcat → both (web & app server)
tomcat → By apache
IIS (Internet Information services) → By Microsoft. → compiles .Net into HTML, CSS, JS

② DB server → Used to persist the Data.
- Eg. RDBMS (MySQL, Oracle)
NoSQL (MongoDB, redis)

③ DNS server → used to get IP address from a Domain name.
Eg. Bind (Linux)

④ SMTP (Simple Mail Transfer protocol)
- Used to send mails Eg. postfix.

⑤ POP (Post office protocol)
- Used to receive emails
- Eg. Dovecot.

⑥ File server → Used to serve Files.
- Eg.
FTP → File Transfer protocol. → windows / linux / mac.
Samba [SMB] → Server Message Block → windows.
NFS → Network File system (Linux)

Server → software
→ machine.

sending php → Apache server → calls php interpreter → converts into HTML, CSS, JS.
mypage.php.

· No matter what it returns HTML, CSS, JS bcz browser understands only it.

* Rendering Engine → Used to convert HTML/CSS to JS objects, it builds the DOM of website
→ Also known as layout Engine.

* JS Engine → Use to execute JS code & produce the O/P.

* UI component → Used to load the UI (output of Webserver)

* web storage component → Used to store data on client.
Eg. history, session storage, local storage.

|  | Rendering Engine | JS Engine. |
|---|---|---|
| Edge | EdgeHTML | chakra. |
| ~~safari~~ | webkit | NitroJS |
| FireFox | Gecko | spiderMonkey |
| Chrome | Blink | V8 |

V8 → Fastest JS Engine in C++
Ryan Dal. NodeJS (Framework) → Extension of V8

* Rendering Engine creates DOM.
JS → Document object

Prerequisites

Function
Promise
Function Reference
Functional programming → map, filter, reduce
Rest operator
Destructuring

HTML → Sir Berner Lee
JS → Brendan Eich
Java → James Gosling

ES6 → ECMAScript → standard version of JS.

Netscape Navigator (FireFox) → First commercial Browser.

Internet Explorer → Microsoft Browser.
→ Free thats why Netscape got out.

ECMAScript use kon prateKane apla apla Browser Banvla.

Mozilla Developer Network (MDN) → Best Documentation of HTML, CSS, JS

· React → Jordan Walke, FaceBook.

MVC (Model view controller) architecture.

MVC
Model → Data
View → UI
Controller. → controls the view.

· view + controller = component.

MPA → Reloads every single time. Eg. Amazon.
SPA → does not reload every time. Eg. Gaana.

AI is good for those who already know.

* React → Fast: similar performance to native apps.
→ Responsive. → size change events.
Bootstrap → Framework
Tailwind → CSS + JS

In Interview → Tailwind asked more than Bootstrap.

React → JS Library for Building UI

React / Angular.
- developed in JS / developed in Typescript — Read about Typescript.
- Faster. uses virtual DOM.
- Less memory consumption/footprint.
· automatically datatype change so Typescript → Better JS.
→ Dangerous.

* Environment setup
① using CDN (Content Delivery Network)
② using vite (Package Manager)

```
function ongreet() {
  // get the object of div element
  const div = document.getElementById('root')

  // update the div with req. msg.
  div.innerHTML = <h2> = Hello ${username} ! </h2>
```

${username} → String interpolation.

document. → DOM object

DOM reloads → div1. innerHTML. → expensive action. slows down.
problem → multiple elements adding.
sol^n → React

Array of objects → render persons.

function onclick() { → Button on click assign.

th → Table Head.
td → Table Data

2 prblm : Data in JS, UI in HTML → String interpolation Technique.
can't see error.

Vanilla JS → pure JS without Framework.

sol^n → React Library

we want to mix HTML & JS.

Using CDN links

function on Button click()

2 links. — one Development purpose. one virtual Dom

Jim Ron → Burning Desire.
Brain Tracy

```
const h2 = React.createElement('h2', {}, 'hello world')
```

// create Element
parameters
① Name / Type. → Eg. h1, h2
② Attribute / properties → Eg. class, id, style (this must be object)
③ content / child of element → Eg. Text, html, etc.

React is an object which is an object used to create Elements.

// get the root element → will render element here.

```
const root = document.getElementById('root')
```

ReactDOM.render (h2, root) → React 17 style.
→ it loads the element in root

render = loading

Page 4 :

① create div element id = root
② Button — greet user.
③ Add CDN links in Head.
④ create React Element
⑤ get root Element
⑥ Render react element
⑦ button onclick = " onclick --- "

Note → Div element is considered as root element

React 18 style :

```
const root = ReactDOM.createRoot
  (document.getElementById('root'))
```

→ created element root

```
root.Render(h2)
```

→ rendered element

3lines in 2lines

To maintain / read previous code → React 17.

Program 5

```
<body onLoad = "onPageLoad()"
```

onLoad → when page is loaded it will display

```
<style>
  .greeting {
    color : Black
    font-family : —/—
    font size : —/—
  }
```

greeting → class

```
< div class = greeting
```

class → property
id → property.

Invalid. warning. → class as Keyword. thats why warning.

```
'h2', {
  className
}, 'hello User'
```

className → Inside property
No warning

* Do not use class as it is reserved Keyword in JS. Instead use className.

React will create a Virtual DOM using root element.

* Virtual DOM

- a lightweight copy of a real DOM.
- not to load entire DOM. only what it requires.
- (Browser DOM) Real DOM

* Reconciliation — process where creates virtual DOM & updates
— entire page never gets loaded only changes get loaded.

virtual DOM → performance Boost.

without internet → still works as in cached content. Eg. gaana.

without Internet → Amazon. does not load.

· SPA can cache the content

* create Element using JSX → which we use now.
previous create Element → we dont use.

Internally compiler / interpreter does it. - Babel.

```
const h2 = <h2> Hello User! </h2>
```

→ this converts HTML to react syntax code.

Babel → JS compiler

To create an element using JSX, use Babel.

```
< script type = "text / babel"
```

→ Add this above & mention type in Body

· Babel does the conversion automatically.

· This is called JSX

· Babel → JSX code to JS code.
JSX code → HTML <h2> Hello </h2>
JS code → - React Element

see sir notes while studying.

3 links → 2 for react, 1 for Babel.

Note
· Mention type = " text / babel "

${ username } → not string it is JSX.

Just wrap with { }

{ username }

Q. How many types of Tag in HTML ?
→ 4 types.
opening tag
closing tag
Empty tag
root tag → Starts & ends document. — HTML Tag.

rendering will be done using only one Div.

```
const div = {
  <div>
    <div> name : {   }
    <div> address : {   }
  </div>
}
```

div → variable

parenthesis are optional. here.
→ just interpolation not string interpolation.

Auto Format Document Extension.

· can add multiple divs inside one variable div.

// we can render only one element.

we can use class in CSS style, but while calling using className.

* scaler variable → which holds single value.

Object in JS → collection of properties & methods.

name : John
name → property
John → value.

* previously, we render variable, Now object.

Error → object are not valid as React child.

{ person } ✗
{ person.name } ✓

Note: interpolation looks for scaler values

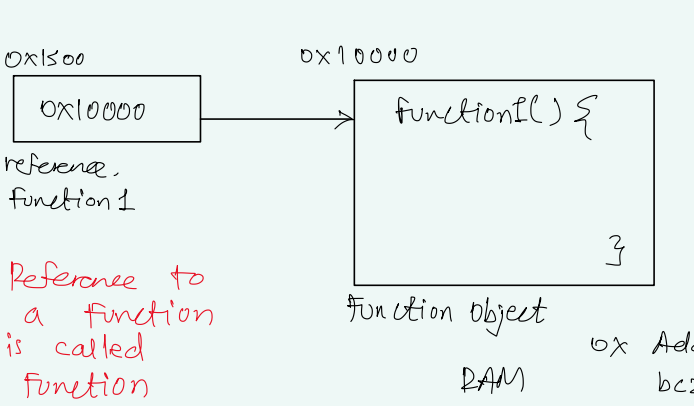
- Interpolation always require a scalar value & cannot render an object
- Interpolation → { }

{ person ['age'] } → can use [ ] operator.
{ person.address.city } ✓
{ person ['address'] ['state'] } ✓

* Rendering Array

* Functional programming language.
- Function is considered as First class citizen.
- Function is created as a variable of type Function.
- Function can be passed as an argument to another Function.
- Function can be returned from another Function as return value.

* Function reference (V. Imp) interview Q.



Reference to a function is called function reference.

0x Address → bcz Hexa Decimal.

RAM divided into Blocks.

function1 is fun. reference.

${ function } ✓ → Function reference.
${ function() } ✗ we are not calling we are pointing to another Function.

let mutable
const immutable

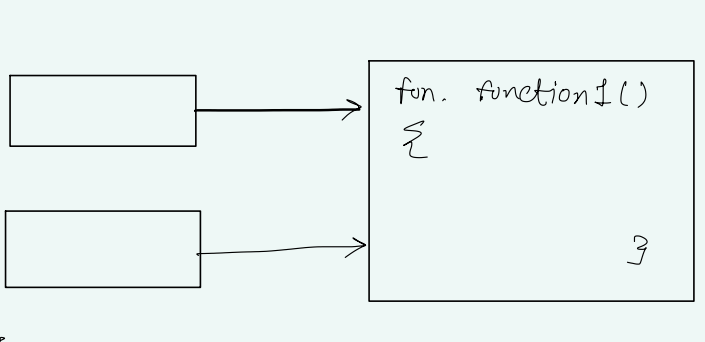
```
const num1 = 100

const num2 = num1
```

→ it create separate memory.
First copy ji ahe tich rahnr. change nhi honr.

now num1 = 200.
still num2 = 100

```
const function 2 = function 1
```



Function 1 → variable

Function 2 → Function reference.

By both reference, we can call Function1().

Tomorrow → Map., render array by Map.
Filter, reduce.
- vite
- component in React.
How to create component.
why not class component but Function component.

Read and come. so you will understand better.

Difference between library and framework in simple terms with example

A **library** is a collection of reusable code that you can call to perform specific tasks. You control the flow of your program and choose when to use the library. A **framework**, on the other hand, provides a structure for your application and dictates the flow, where you plug in your code to fit its design.

**Simple analogy**:

- A library is like a toolbox—you pick the tools (functions) you need and use them how you want.
- A framework is like a house blueprint—you build within its predefined structure, filling in the details.

**Examples**:

- **Library**: jQuery (JavaScript library). You include jQuery and call its functions (e.g., $.ajax() for HTTP requests) wherever you need them in your code. You decide the flow.
- **Framework**: Angular (JavaScript framework). Angular provides a full structure for building web apps, with predefined patterns like components and services. You follow its rules and plug your code into its lifecycle.

**Key difference**:

- Library: You call it, you control the flow.
- Framework: It calls your code, it controls the flow.