Who am I?

Humera Tariq

PhD, MS, MCS (Computer Science), B.E (Electrical)

Postdoc (Medical Image Processing, Deep Neural Networks)

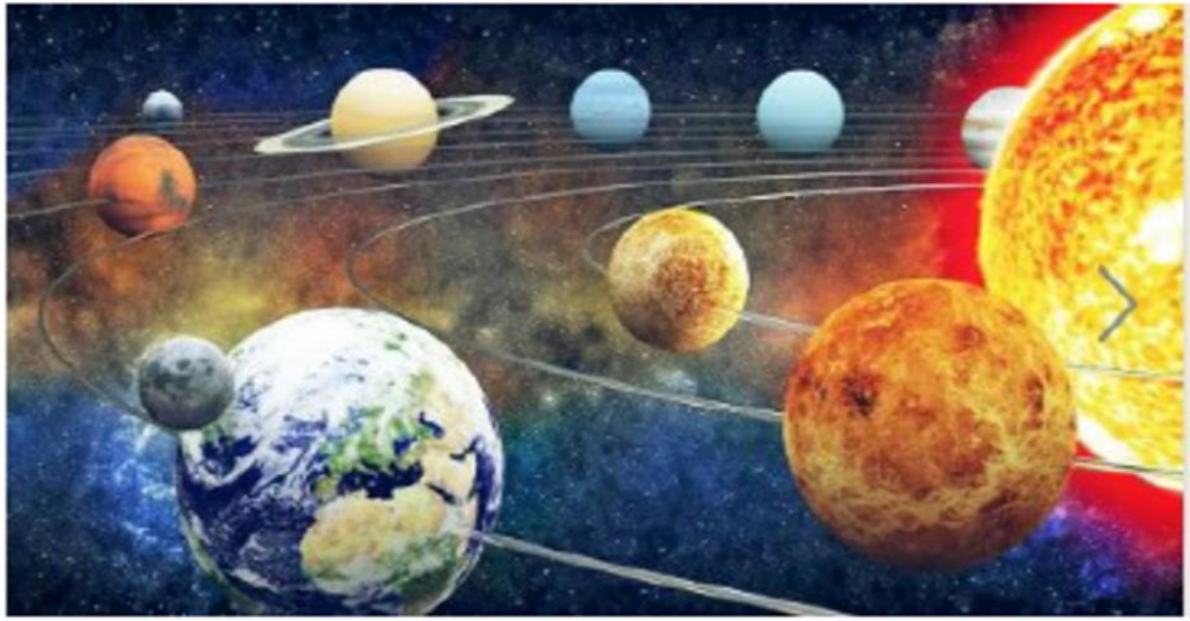
Email: humera@uok.edu.pk

Web: https://humera.pk/

Discord: https://discord.gg/xeJ68vh9

Starting in the name of Allah,

the most beneficial,
the most merciful.



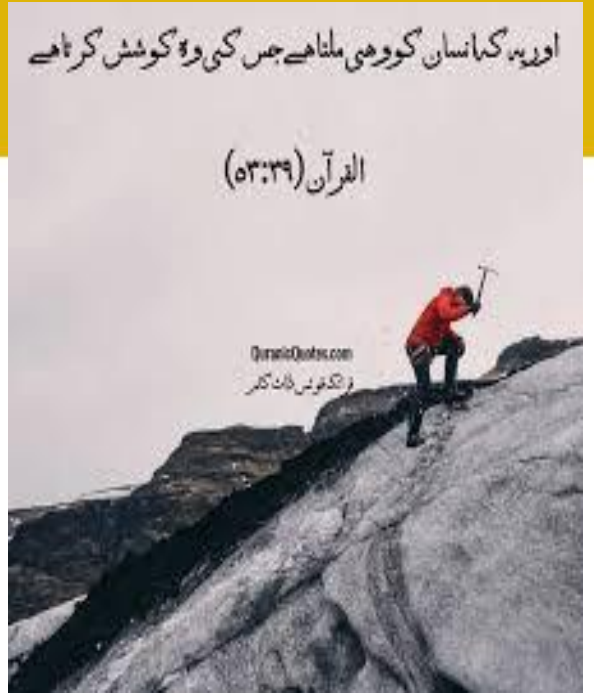
اَمۡ لِلۡاِنۡسَانِ مَا تَمَنّٰى ﴿۲۴﴾ فَلِلّٰهِ الۡاٰخِرَةُ وَالۡاُوۡلٰى ﴿۲۵﴾

کیا انسان کو ہر وہ چیز حاصل ہے جس کی اس نے تمنا کی؟ پس اللہ ہی کے لیے ہے آخرت اور دنیا

UNIVERSITY OF KARACHI

Surah An-Najm Chapter 53 Verse 39

And there is not for man except that [good] for which he strives.



UNIVERSITY OF KARACHI

Week 04

Internet Application Development

Department of Computer Science (DCS/UBIT)
University of Karachi
January 2025

Today's Agenda

- ✓ A little bit from JJ Chapter 1 (W3C history-origin)
- ✓ A little bit about lab challenges
- ✓ Counter variable demo
  - vanilla JS x 2
  - React native x 1
  - Modern React (vite) x 1
  - Redux (vite) x 1
- ✓ Vanilla JS Canvas animation
- ✓ React boiler code overview

UNIVERSITY OF KARACHI

Who initiated these standards ?

Internet Engineering Task Force (IETF)

World Wide Web Consortium (W3C)

Web Hypertext Application Technology Working Group (WHATWG)

ECMA International	ECMAScript (TC39)	Defines JavaScript (ECMAScript) language specs.	Current spec: ECMAScript 2024. Improving async handling and records.
International Organization for Standardization/International Electrotechnical Commission	ISO/IEC JTC 1	Sets international IT standards.	Works on open data formats and security protocols.
Khronos Group	N/A	Defines standards for graphics and media (WebGL, WebGPU).	WebGPU gaining traction over WebGL for better web graphics.
Unicode Consortium	N/A	Develops Unicode character encoding standards.	Unicode 15.1 is the current version (more emojis added).
Internet Assigned Numbers Authority	IANA	Manages IP addresses, protocols, and domain names.	Oversees IPv6 implementation and DNS.
FIDO Alliance	N/A	Works on strong, passwordless authentication.	WebAuthn becoming widely adopted by browsers.

HOW THE WEB BEGAN!

1989

Sir Tim Berners-Lee made his first proposal for the Web to make sharing information across the Internet easier. The proposal wasn't initially accepted!

1990

Tim has created the three core technologies of the Web (HTTP, URI and HTML). From which he created the first website.

1991

People outside of CERN (Tim's workplace) primarily University-based scientific departments and physics laboratories were invited to use the Web.

1993

Tim wanted everyone and anyone to have access to the web so it would achieve its full potential. So announced he'd ensure CERN allowed the royalty-free use of the Web forever.

1994

Tim moved to Massachusetts Institute of technology where he founded the World Wide Web Consortium (W3C) devoted to developing and upkeeping open Web standards.

Who Invent what ? WWW or Internet

Tim Berners-Lee

British scientist https://www.britannica.com/biography/Tim-Berners-Lee

Ask the Chatbot a Question | More Actions

Also known as: Sir Tim Berners-Lee
Written by Michael Aaron Dennis
Fact-checked by The Editors of Encyclopaedia Britannica
Last Updated: Dec 16, 2024 • Article History

Tim Berners-Lee (born June 8, 1955, London, England) is a British computer scientist, generally credited as the inventor of the World Wide Web. In 2004, he was awarded a knighthood by Queen Elizabeth II of the United Kingdom and the inaugural Millennium Technology Prize (€1 million) by the Finnish Technology Award Foundation.

Computing came naturally to Berners-Lee, as both of his parents worked on the Ferranti Mark I, the first commercial computer. (*See* computer: The first stored-program machines.) After graduating in 1976 from the University of Oxford, Berners-Lee designed computer software for two years at Plessey Telecommunications Ltd., located in Poole, Dorset, England. Following this, he had several positions in the computer industry, including a stint from June to December 1980 as a software engineering consultant at CERN, the European particle physics laboratory in Geneva.

Tim Berners-Lee Tim Berners-Lee, 2014.

Quick Facts

In full: Sir Tim Berners-Lee

Born: June 8, 1955, London, England (age 69)

Founder: World Wide Web Consortium

Inventions: World Wide Web • World Wide Web

- ✓ Program for hypertext
- ✓ research to control remote machines
- ✓ Tim Berners-Lee also developed the first web browser called **WorldWideWeb** in 1990

UNIVERSITY OF KARACHI

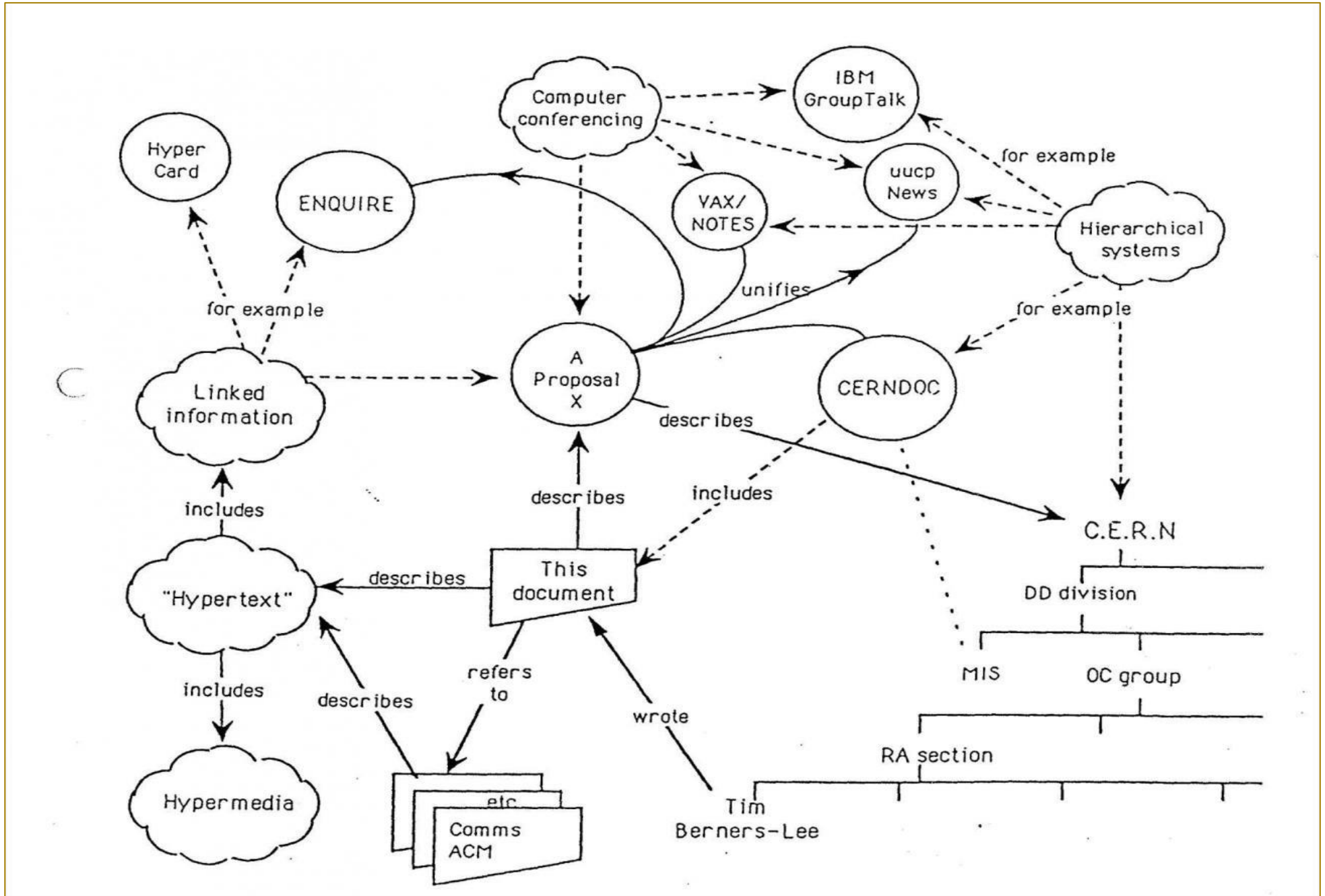
What happenend b/w 40s and 90's ?

The main event of the 1990s was to be the emergence of the World Wide Web application, which brought the Internet into the homes and businesses of millions of people worldwide. The Web served as a platform for enabling and deploying hundreds of new applications that we take for granted today, including search (e.g., Google and Bing) Internet commerce (e.g., Amazon and eBay) and social networks (e.g., Facebook).

The Web was invented at CERN by Tim Berners-Lee between 1989 and 1991 [Berners-Lee 1989], based on ideas originating in earlier work on hypertext from the 1940s by Vannevar Bush [Bush 1945] and since the 1960s by Ted Nelson [Xanadu 2012]. Berners-Lee and his associates developed initial versions of HTML, HTTP, a Web server, and a browser—the four key components of the Web. Around the end

UNIVERSITY OF KARACHI

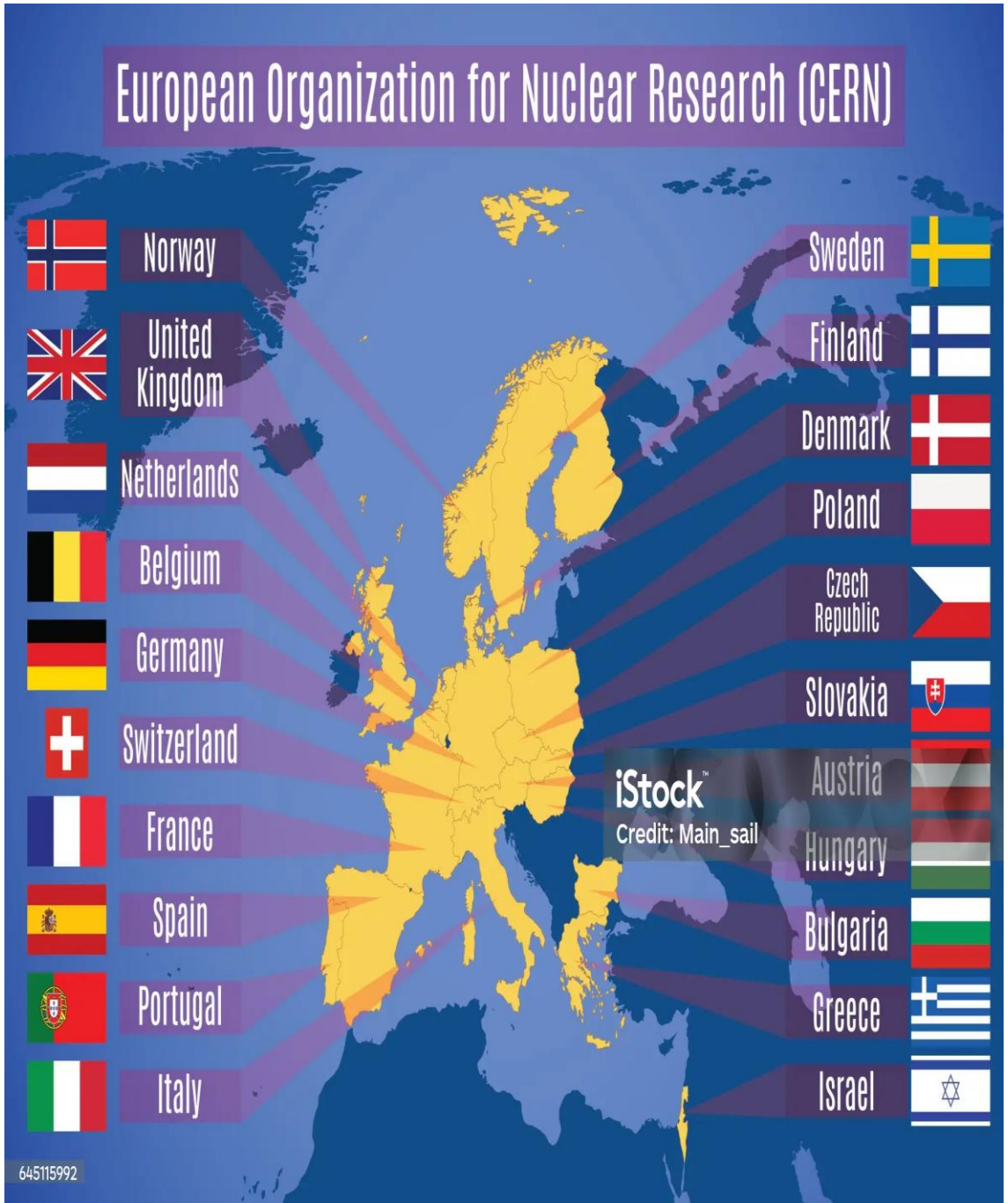
The image on the cover page of Tim Berners-Lee's proposal for the World Wide Web in March 1989 (Image: CERN)

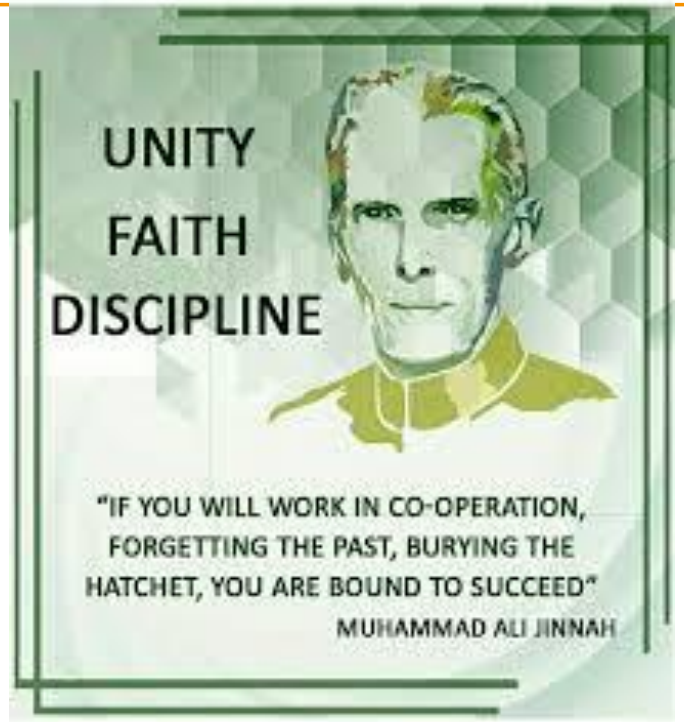


Origins

At the end of the Second World War, European science was no longer world-class. Following the example of international organizations, a handful of visionary scientists imagined creating a European atomic physics laboratory. Raoul Dautry, Pierre Auger and Lew Kowarski in France, Edoardo Amaldi in Italy and Niels Bohr in Denmark were among these pioneers. Such a laboratory would not only unite European scientists but also allow them to share the increasing costs of nuclear physics facilities. French physicist Louis de Broglie put forward the first official proposal for the creation of a European laboratory at the European Cultural Conference, which opened in Lausanne on 9 December 1949. A further push came at the fifth UNESCO General Conference, held in Florence in June 1950, where American physicist and Nobel laureate Isidor Rabi tabled a resolution authorizing UNESCO to "assist and encourage the formation of regional research laboratories in order to increase international scientific collaboration…" At an intergovernmental meeting of UNESCO in Paris in December 1951, the first resolution concerning the establishment of a European Council for Nuclear Research was adopted. Two months later, 11 countries signed an agreement establishing the provisional council – the acronym CERN was born.

https://home.web.cern.ch/resources/video/computing/interview-tim-berners-lee-20th-anniversary-world-wide-web





https://www.nucamp.co/blog/coding-bootcamp-pakistan-pak-top-10-mustattend-tech-meetups-and-conferences-in-pakistan

https://invest.gov.pk/it-ites

Top 10 Must-Attend Tech Meetups and Conferences in Pakistan

By **Chevas Balloun**

Last Updated: **December 25th 2024**

National Centre for Physics Islamabad

Home | About NCP | Organization | Collaborations | Publications | Scientific Events | Photo Gallery | Contact Us

Home / Scientific Events / ISPAD 2025

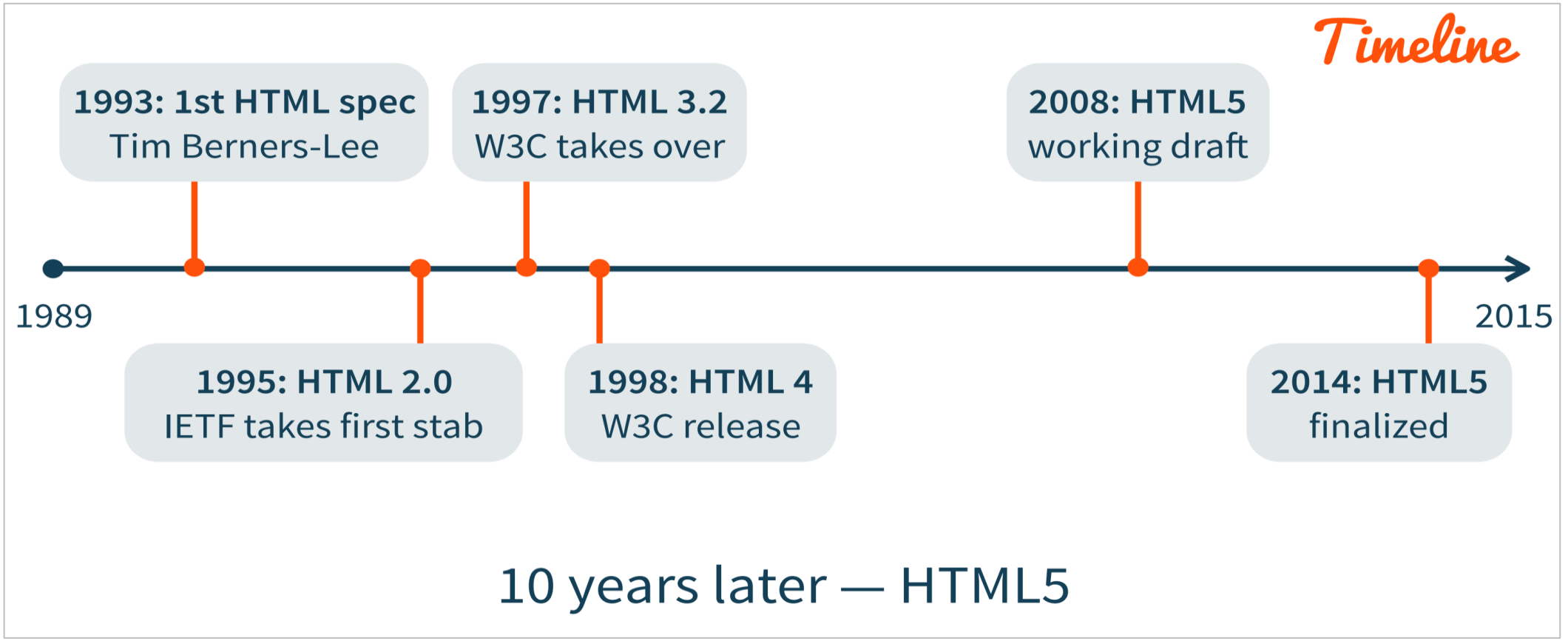
International School on Physics & Allied Disciplines (ISPAD) - 2025

14 - 18 April, 2025

http://ncp.edu.pk/ispad-2025.php

The **National Centre for Physics (NCP), Islamabad, Pakistan**, **The Abdus Salam International Centre for Theoretical Physics (ICTP), Trieste, Italy** and **The European Organization for Nuclear Research (CERN), Switzerland** are jointly organizing the **International School on Physics & Allied Disciplines (ISPAD)** in Islamabad, Pakistan from **April 14 - 18, 2025**.

HTML continued to develop!



UNIVERSITY OF KARACHI

CSS continued to develop!

Timeline

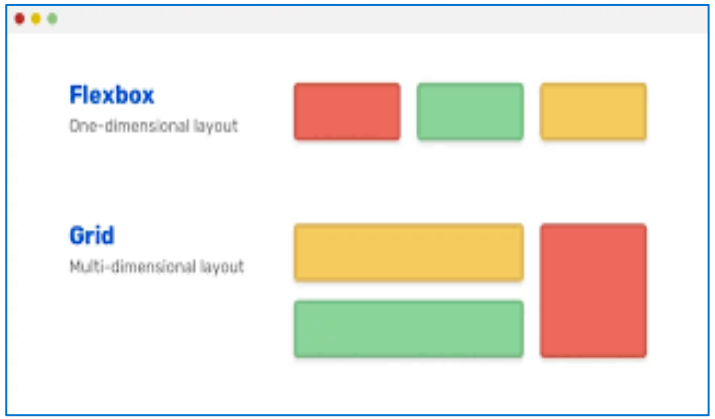
1993: 1st HTML spec
Tim Berners-Lee

1996: CSS 1
W3C release

1994: CSS draft
Håkon Wium Lie

1989 — HTML2 — HTML3 — **1999-2012: CSS3 released in modules** → 2015

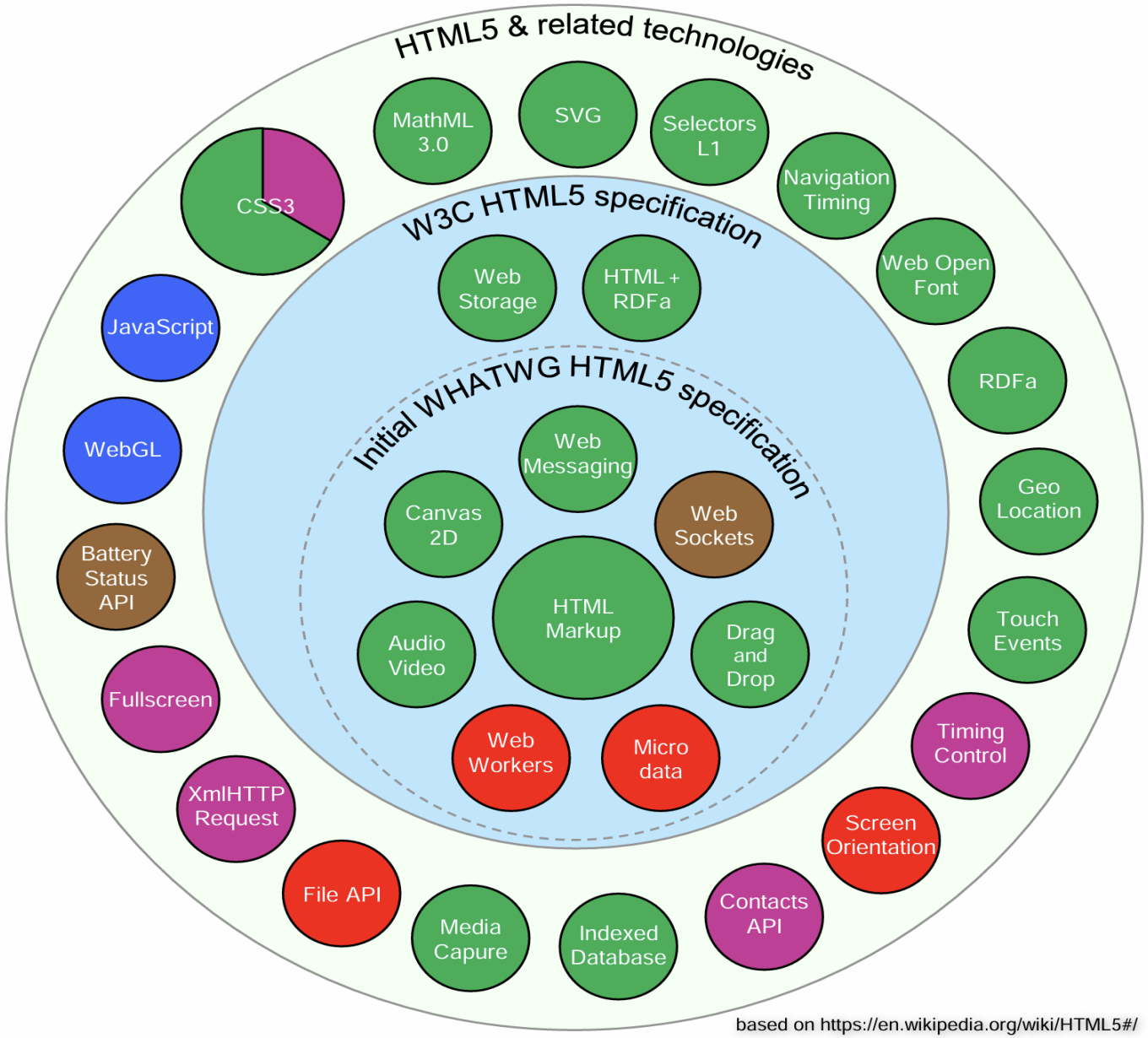
CSS4?



UNIVERSITY OF KARACHI

HTML5 and Open Web Platform APIs

- W3C Recommendation
- W3C Candidate Recommendation
- Working Draft
- W3C Working Group Note
- Non-W3C Specification



based on https://en.wikipedia.org/wiki/HTML5#/

- Simplifies implementation of cross-platform applications
- Standard way for accessing specific functionality
- No need for plug-in installation

CSS1 specification published in 1996

- ✓ remember that HTML 3.2 introduced some elements and attributes (e.g. color) for the visual appearance in 1997

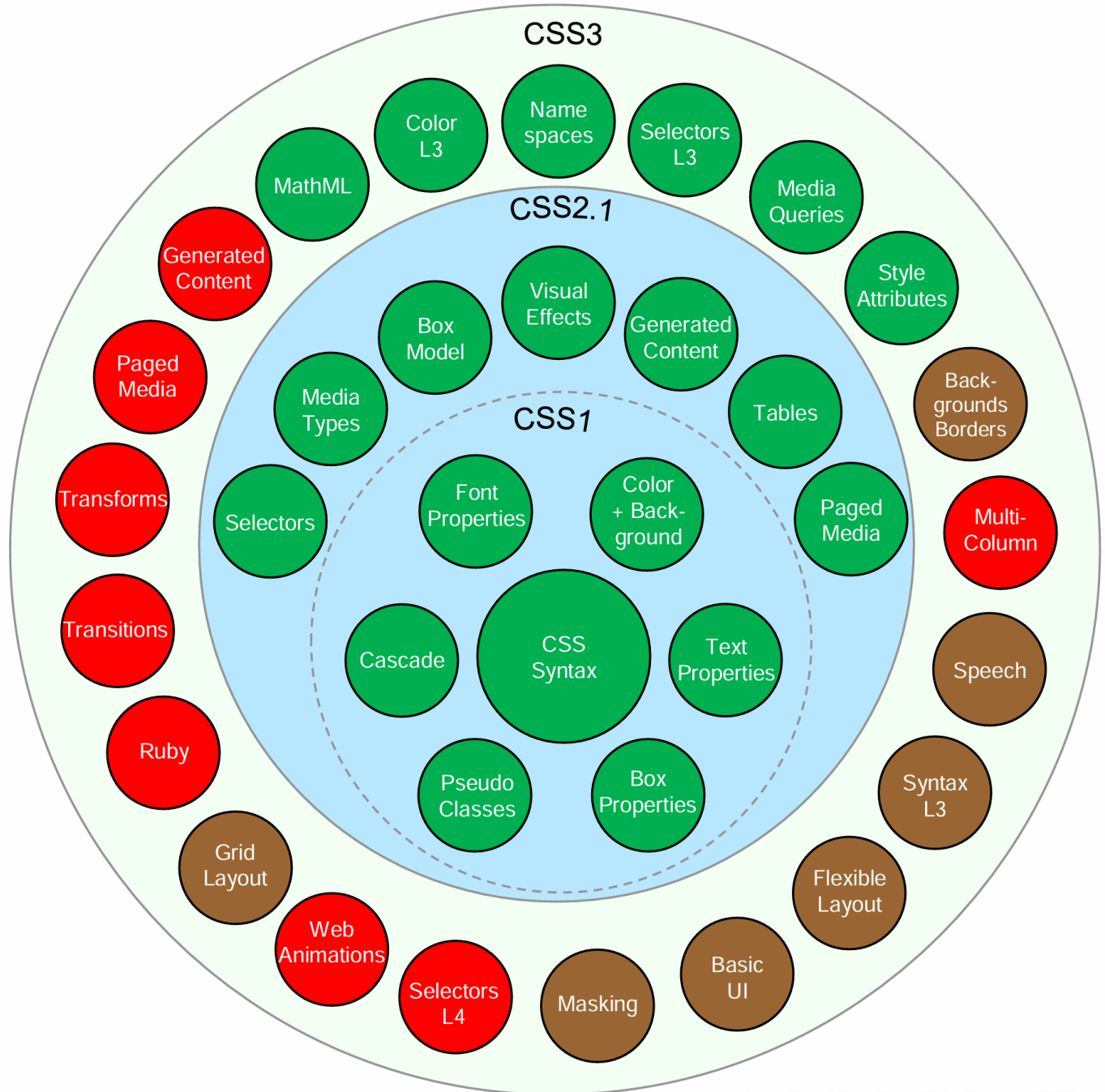
CSS2 specification published in 1998

- ✓ superset of CSS1
- ✓ functionality for relative, absolute and fixed positoning of elements
- ✓ media types

CSS3 divided into separate modules (documents)

- ✓ 2D & 3D transforms
- ✓ transitions
- ✓ Flexbox
- ✓ media queries
- ✓ ……..

- W3C Recommendation
- W3C Candidate Recommendation
- Working Draft



based on https://en.wikipedia.org/wiki/Cascading_Style_Sheets

a-group-5

a-group-6

b-group-10

b-group-1

Week 03

Assignment

HTML-CSS-JS

minimal website

a-group-6

b-group-1

b-group-10

4.23. Consider a two-player game played with pegs of six different colors. One player (the *encrypter*) selects four pegs of different colors and places them in a certain order (leftmost through rightmost). This is the *encrypted message,* and is not shown to the second player (the *decrypter*). The decrypter then selects four pegs and places them in an order from left to right; this is the decrypter's guess about what the encrypted message is. Next, the encrypter *leaks* information about the encrypted message by placing small black or white pegs next to the guessed pegs; these pegs define a *score* for the guess. A black scoring peg indicates that one of the guessed pegs is exactly right: right color and in the right position. A white scoring peg indicates that a guessed peg is the right color but in the wrong position. So two black scoring pegs and one white would mean that three of the guessed pegs are of the right color and that two of these are in the correct position within the guessed peg ordering. Once a guess has been scored, the decrypter guesses a second time, and the encrypter scores this guess. (See Figure 5.35 for a graphical representation of such a game after two guesses have been scored.) This continues until the decrypter fully guesses the secret code (score of four black pegs).

Write a JavaScript program that plays a text version of this game. The program should simulate the encrypter, and the human player will be the decrypter. Specifically, the program will randomly select a secret code consisting of four numbers between 1 and 6. The numbers should all be different. The program will then input and score each guess made by the decrypter (human player). In particular, the user will input four numbers in a prompt box, and the computer will score this guess against the secret. The computer will then output to an alert box the user's guess and score (so many black and so many white pegs). If the score is 4 black, the game ends and the computer displays the total number of guesses made. Otherwise, the computer asks for another guess consisting of four numbers. Notice that the human is expected to write down his or her guesses and scores, because the computer will only output the most recent guess and score by the user each time. Your program does not need to verify that the user enters numbers between 1 and 6, since entering something else can only hurt the user's chances of winning. You may also assume that all of the user's four inputs for a guess are different numbers.

3.24. Give a style sheet rule for the `body` element of a document that will cause a background image to be repeated across the vertical center of the browser client area. The image should remain in the center of the window even if the window is scrolled (see Fig. 3.48, in which "Draft . . ." is an image).

3.25. Describe what the `fixed` value for the `position` style property does when viewing a document in a browser. Give an example of how this feature might be useful. Test to see which browsers (as assigned by your instructor) support this value for `position` (IE6 does not).

a-group-5

4.21. Write a JavaScript program that will allow a user to test the speed at which alert boxes are generated by a particular browser running on a particular system. Specifically, the program will display an alert box, then display a second alert box after OK is clicked in the first alert box, display a third alert box when the second is acknowledged, and so on. The program will stop displaying these alert boxes once 10 seconds has elapsed from the time the first box was displayed (so dividing by 10 will give the number of boxes displayed per second). Then, for 2 seconds, the program should repeatedly display alert boxes that report on the number of clicks per second recorded during the first 10 seconds (the reason for doing this for 2 seconds is so that you can see the results even if you accidentally click OK on one or more of the result alert boxes). For timing, use only the built-in `Date` object and its methods; that is, don't use any other timing-related host objects that might be available in your browser. Report on the results of testing with various browsers and systems.

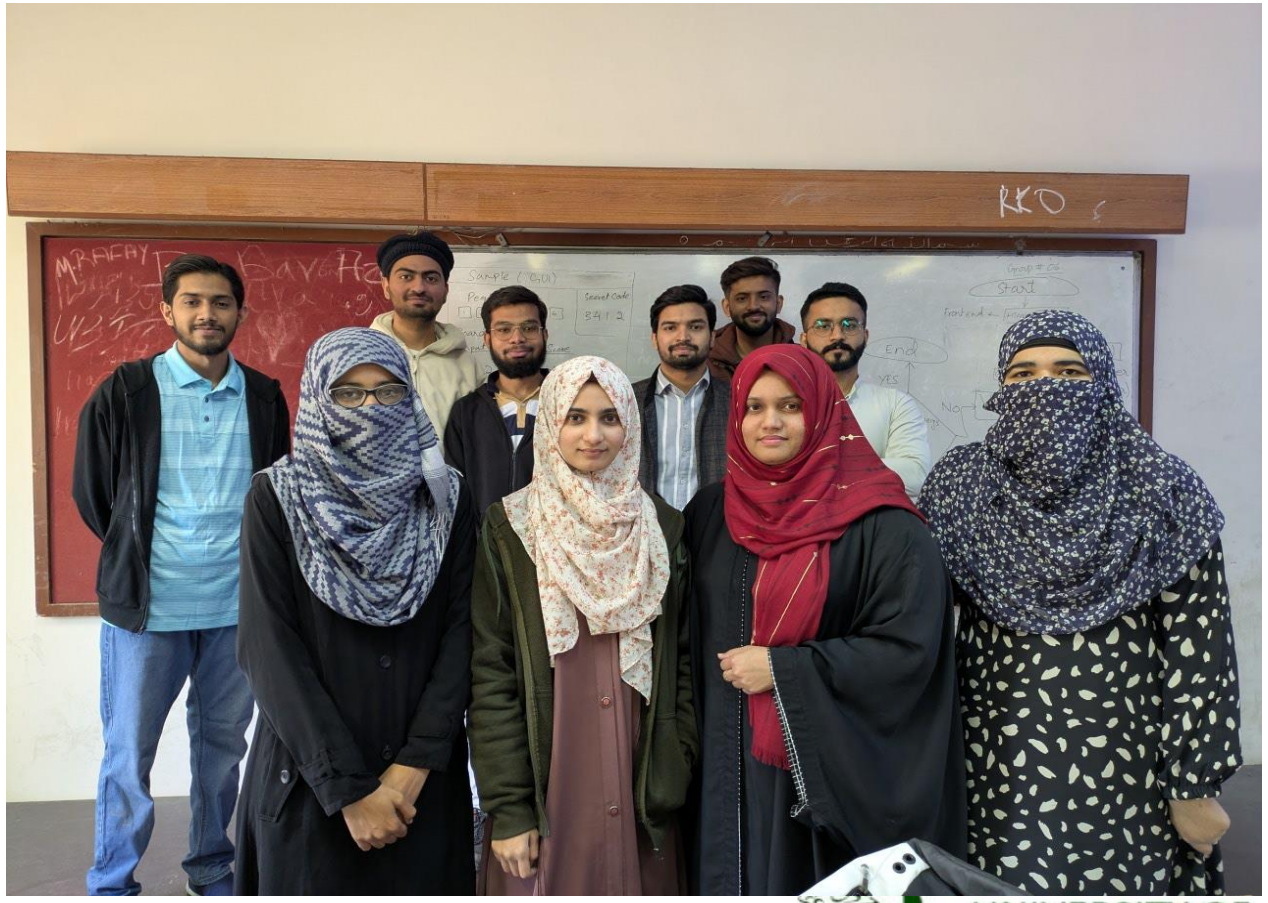
a-group-5

a-group-6

AG6_contribution_website_on_phone

challengeBlog.html - IAD - StackBlitz

https://dinesh-kumar143.github.io/Website/



UNIVERSITY OF KARACHI

b-group-1

https://anashussain2711.github.io/iad-session-1/

b-group-10

https://ias-lab-challenge-website.w3spaces.com/

	10	
#	TEAM LEADER: TAHA ZAHEER	
1	B21110006152	Taha Zaheer
2	B21110006015	Ally Abdullah Zafar
3	B21110006110	Sadaqat Zia
4	B21110006023	Bismah Nasir
5	B21110006112	Saher Hussain
6	B22110004010	Ahmed Hashmi
7	B22110004005	Junaid Khalid

a-group-5

Requirements Debate! (B-G1 and B-G10)

a-group-6

@M_Hammad_waseemB20102089 To be honest we have fulfil the requirement of task than group 6 specially for displaying of images wit...

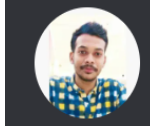
Syed Taha Jameel B20102169 Today at 3:35 PM

If I'm being honest, I would say that their site is also really good.
The fact that they added all the other things could mean two things:

1) Maybe that's how they interpreted what the teacher asked them to do.
2) Or they tried to go above and beyond to show that they can do more.

Either way, I really like their site—it's nice.

1



Nabeel Khan Today at 3:25 PM

@a-group-6 Few suggestions from my end:

Instead of pasting long code in a container, you can simply use max-height prop with overflow-y scroll, this way the whole html wont be that long.

Secondly, for code sharing it is always better to use syntax hightlighting

👍 2

https://prismjs.com/

https://dinesh-kumar143.github.io/Website/

@Dr. Humera Tariq @Nabeel Khan @Dinesh Kumar B21110006026 # a-group-6 I am curious if you answer my following questions o...

Dinesh Kumar B21110006026 Today at 1:27 PM

We ll update it

I ll make a navigation bar which user can go through as hi want and ll also make it responsive as well.

Where were we now ?

- ✓ A little bit from JJ Chapter 1 (W3C history-origin)
- ✓ A little bit about lab challenges
- ✓ Counter variable demo

vanilla JS	x 2
React native	x 1
Modern React (vite)	x 1
Redux	x 1

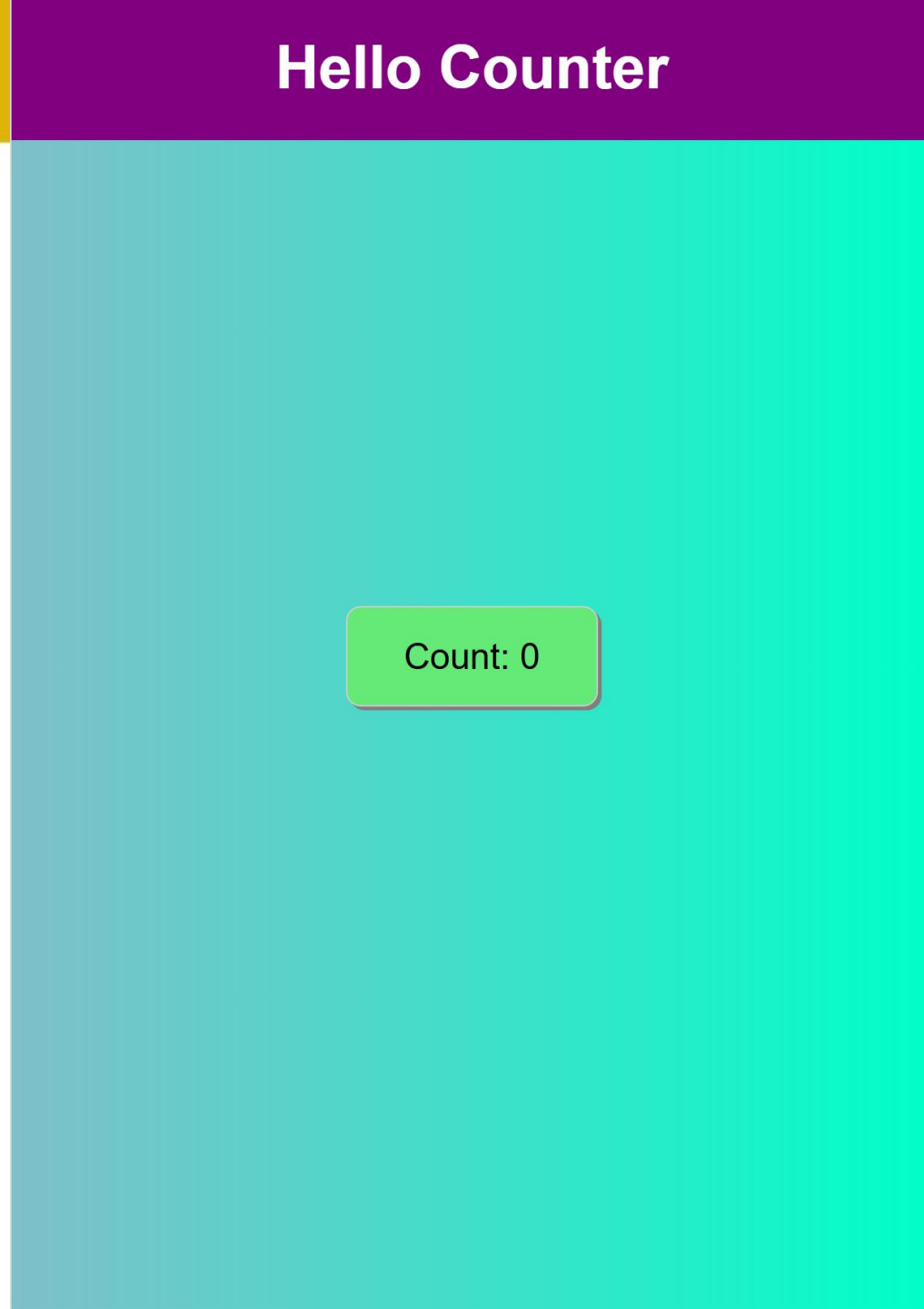
- ✓ Vanilla JS Canvas animation
- ✓ React boiler code overview

UNIVERSITY OF KARACHI

Vanilla JS counter (html, css, js)

Vanilla JS – Inefficient counter

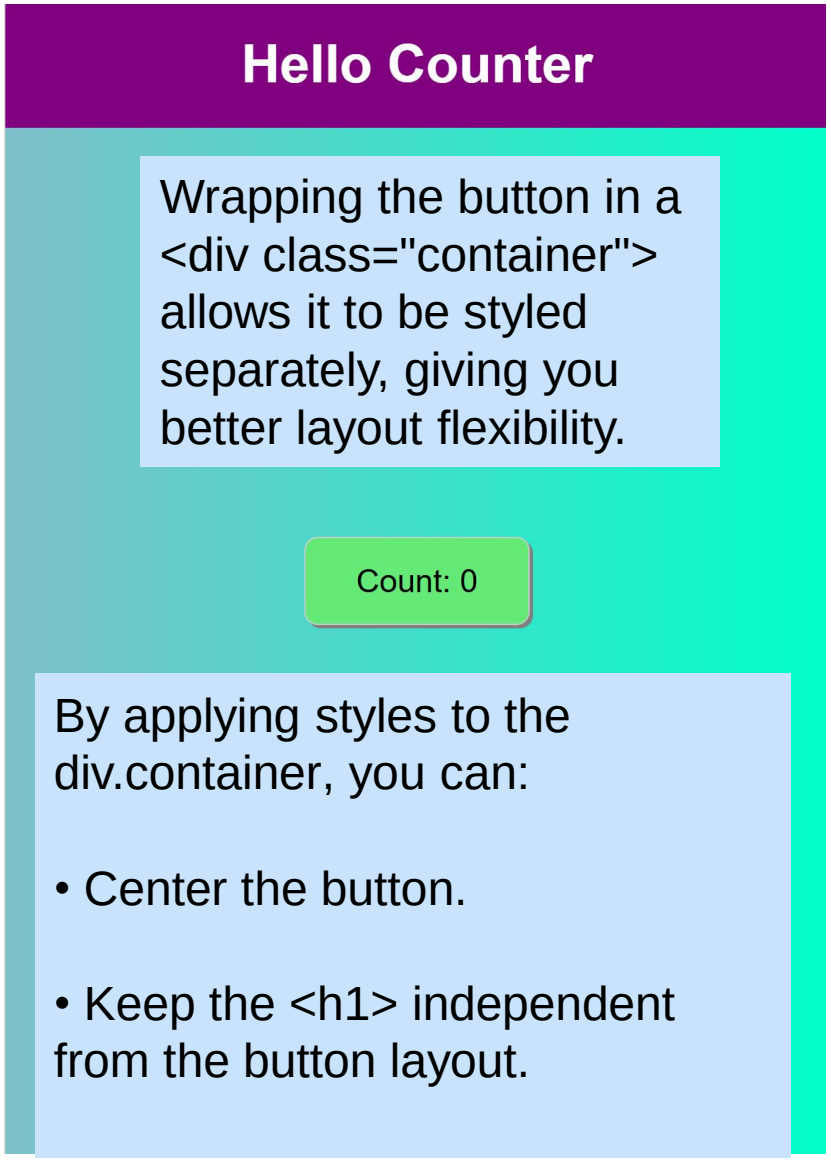
```
html
├── head
│   ├── meta (charset="UTF-8")
│   ├── meta (name="viewport")
│   ├── title ("Counter Button")
│   └── style (CSS rules)
├── body
│   ├── h1 ("Hello Counter") [positioned at the top via CSS
│   └── div.container
│       └── button#btn (aria-live="polite") ("Count: 0")
└── script (JavaScript logic)
```



The body indeed contains two main elements in our example:

(i) ____??____ **:** Displays the heading at the top.

(ii) A ____??____ containing ____??____ Element is used to group and structure the button independently.

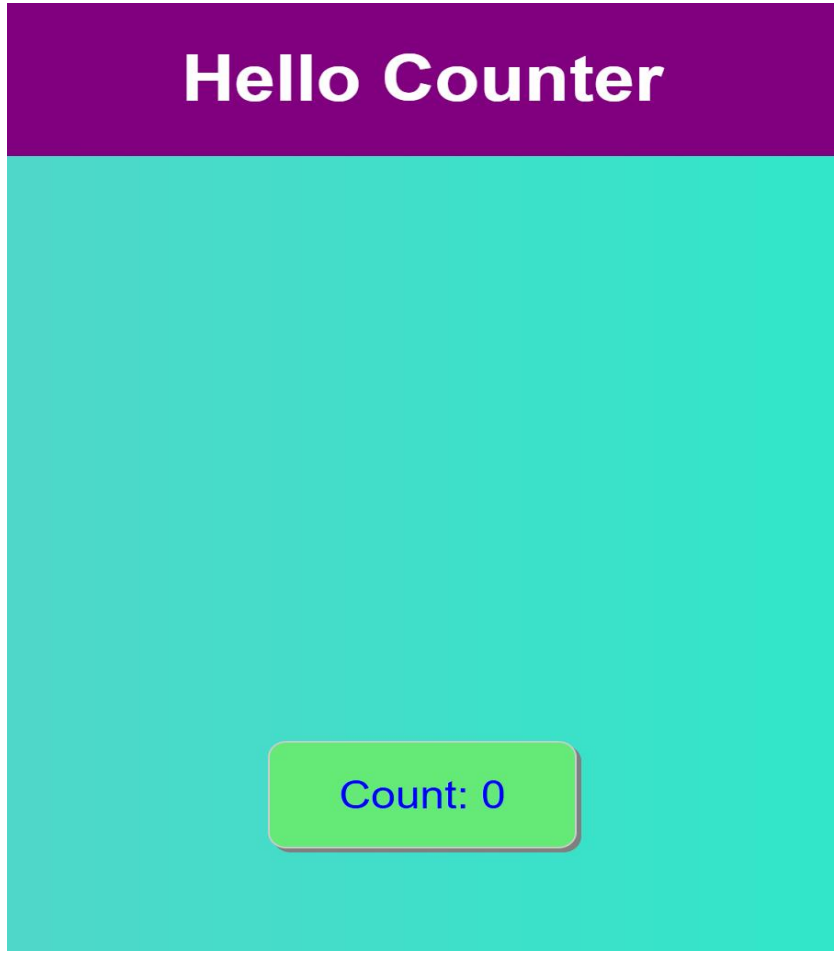



```css
h1 {
    width: 100%;
    background-color: purple;
    color: white;
    text-align: center;
    padding: 20px;
    position: absolute;
    top: 0;
}

.container {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 100%;
    width: 100%;
}

button {
    padding: var(--button-padding);
    font-size: var(--button-font-size);
    cursor: pointer;
    border-radius: var(--button-border-radius);
    border: 1px solid #ccc;
    background-color: #65ea77;
    box-shadow: 2px 2px grey;
    transition: background-color 0.3s ease;
}

button:hover {
    background-color: var(--button-hover-bg);
}
```

When the user hovers over the button. The transition gradually animates from the default background color (#65ea77) to the hover background color (#f0f0f0) over 0.3 seconds, making the interaction visually smooth.

✓ The variables defined under the :root selector are ___?___.(local/global)

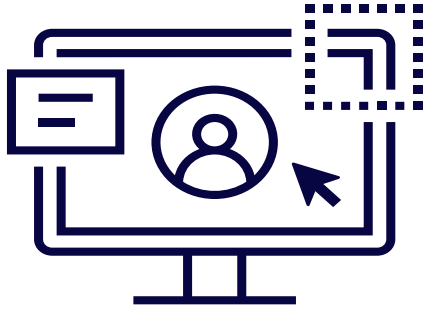
✓ --button-text-color would be a ___?___ **variable**, available only within the button selector and its ___?___ .




```css
button {
    padding: var(--button-padding);
    font-size: var(--button-font-size);
    cursor: pointer;
    border-radius: var(--button-border-radius);
    border: 1px solid #ccc;
    background-color: #65ea77;
    box-shadow: 2px 2px grey;
    transition: background-color 0.3s ease;
    --button-text-color: blue;
    color: var(--button-text-color);
}
```

User (1) → OS (2) → Browser (3) → OS (4) → Browser (5) → Browser (6)

file:///D:/____BSCS_IAD/__week_04/big4_js/vainiall_counter_inefficient.html



- **User (1)**: Double-clicks index.html.
- **OS (2)**: Identifies file type and associates it with the browser.
- **Browser (3)**: Receives file path with file://.
- **OS (4)**: Locates and reads the file.
- **Browser (5)**: Processes(parsing) HTML, CSS, and JS.
- **Browser (6)**: Renders the page and displays the file:// URL.

HTTP-based URLs need a host (www.example.com).

- ✓ Just like http://, the file:/// protocol is also a ____??____ in the broader URI structure.
- ✓ Just as HTTP URLs point to resources on the ____??____ , file URLs point to resources on the ____??____ .

HTTP URL

```
http://www.example.org/index.html
```

- Scheme: `http`
- Host: `www.example.org`
- Path: `/index.html`

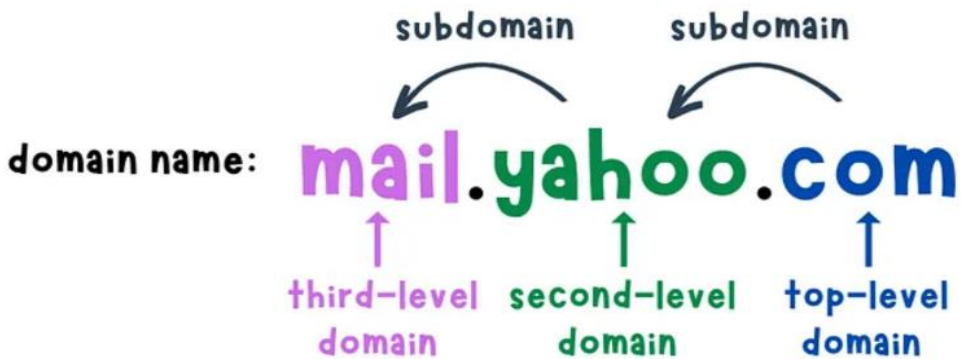
File URL

```
file:///D:/path/to/index.html
```

- Scheme: `file`
- No host component (or empty)
- Path: `D:/path/to/index.html`

UNIVERSITY OF KARACHI

Domain Name



root .

top-level domain com

second-level domain yahoo

third-level domain mail

Parts of url

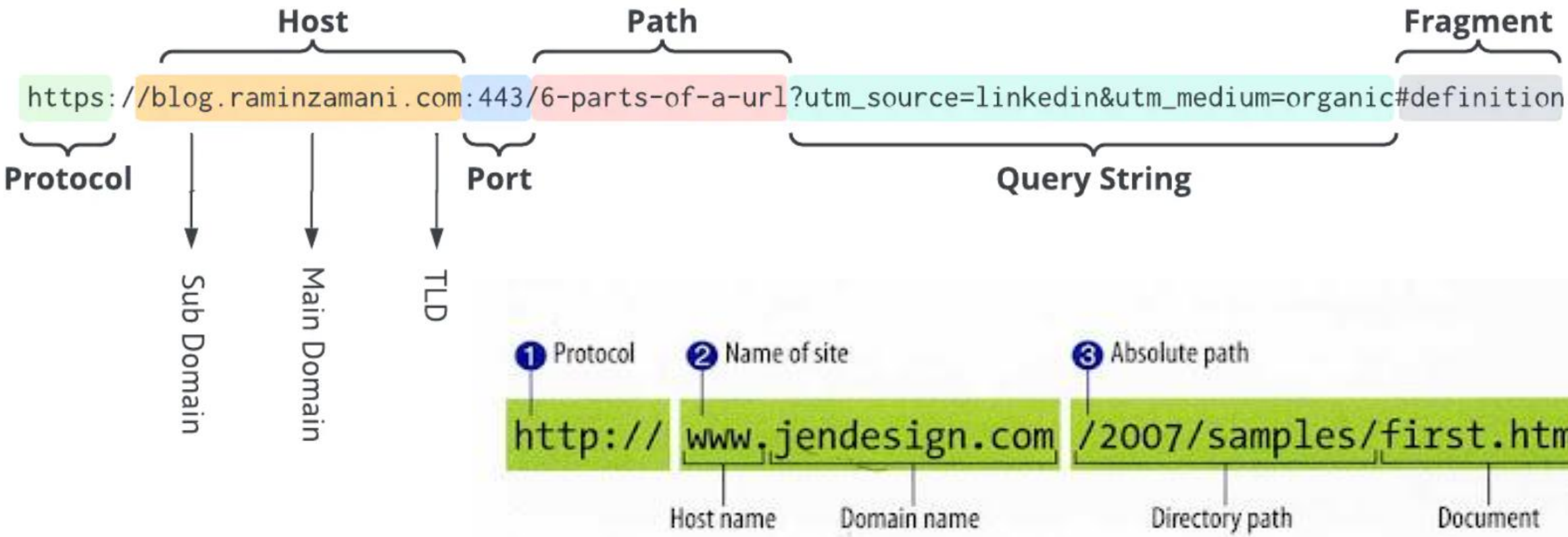


Figure 2-1. The parts of a URL.

Technical Limitation of file url

(i) The default security policy enforced by browsers is called the **Same-Origin Policy**, which blocks ____??____ requests between different origins. Same-Origin Policy prevents scripts on one website from making requests to another website's domain unless explicitly allowed by the server.

(ii) CORS stands for **Cross-Origin Resource Sharing**, a mechanism that allows or restricts ____??____ between different domains. CORS ensures that a client (like a browser) can securely request resources **(data, scripts, APIs)** from a server hosted on a different origin.

(iii) If you see a browser error like: “Access to fetch at 'https://api.example.com/data' from origin 'http://localhost:3000' has been blocked by CORS policy”. This indicates that the backend server does not include the appropriate ____??_____?____ in its response.

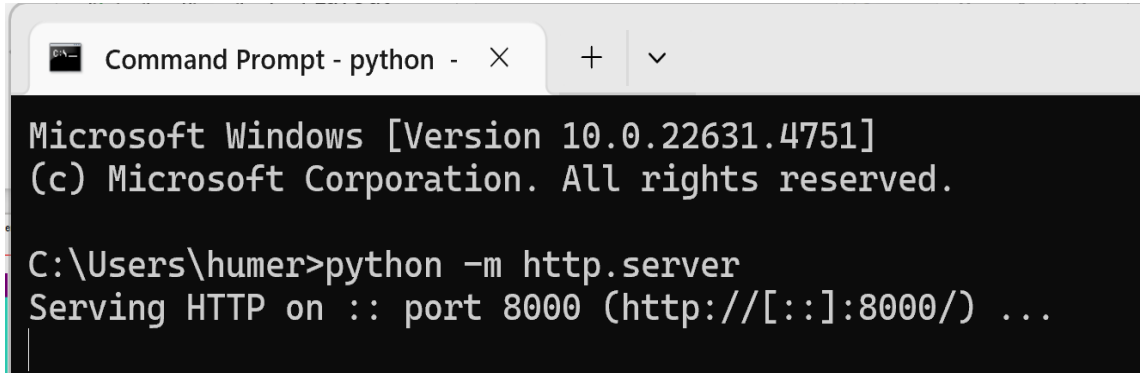
UNIVERSITY OF KARACHI

Solution / file protocol issues resolution

Use a lightweight development server (VSCode Live Server, http-server, or built-in Python server python -m http.server, Node.js, Python (Django/Flask), PHP, Ruby on Rails).

(i) Then Access the app at _____ ?? _____.

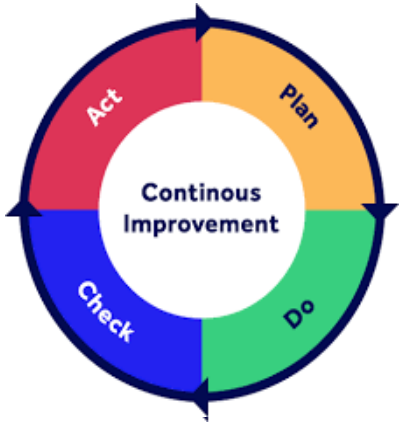
(ii) http://localhost:8080 is a _____ **??** _____ where your app is served (via a server running on port 8080).

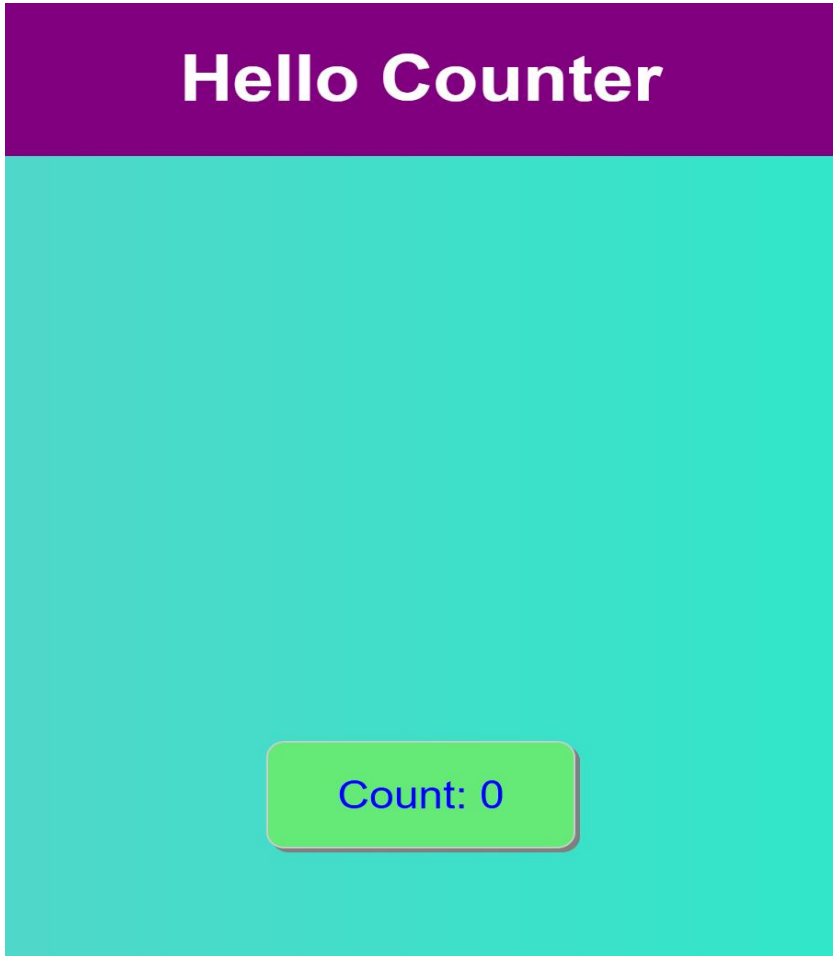


UNIVERSITY OF KARACHI

Vanilla JS counter (html, css, js) Version 1

```
68  <body>
69      <h1>Hello Counter</h1>
70
71      <div class="container">
72          <button id="btn" aria-live="polite">Count: 0</button>
73      </div>
74
75      <script>
76          let count = 0;
77          const button = document.getElementById('btn');
78
79          button.addEventListener('click', () => {
80              count++;
81              button.textContent = `Count: ${count}`;
82          });
83      </script>
84  </body>
85  </html>
86
```

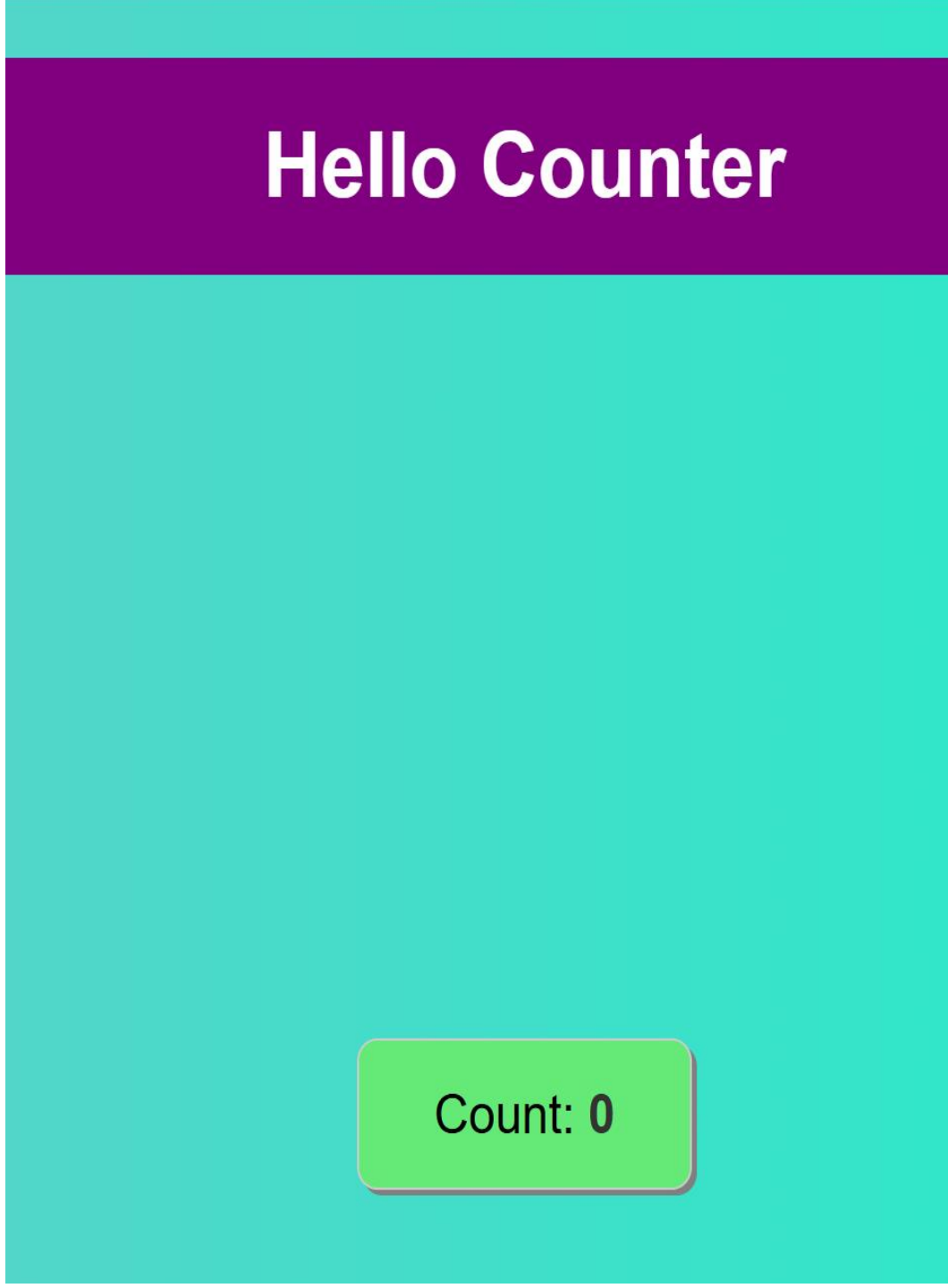




Vanilla JS counter (html, css, js)

Version 2

```
html
 └── head
     ├── meta (charset="UTF-8")
     ├── meta (name="viewport")
     ├── title ("Counter Button")
     └── style (styling rules)
 └── body
     └── div.container
         └── button#btn
             ├── text node ("Count: ")
             └── span#count
                 └── text node ("0")
     └── script
         └── JavaScript code (handles button click and counter update)
```



efficient

Why the inside the button?

- The separates the dynamic number from static text
- Allows us to update **only the number** without affecting the rest of the button
- More efficient than rewriting the entire button content
- Better for accessibility and DOM structure

```
<body>
    <h1>Hello Counter</h1>

    <!-- Wrapper div to group the button and the counter -->
    <div class="container">
        <button id="btn">
            Count: <span id="count">0</span>
        </button>
    </div>
    <script>
        // Initialize counter variable
        let count = 0;
        // Get references to DOM elements
        const button = document.getElementById('btn');
        const countSpan = document.getElementById('count');

        // Add click event listener to button
        button.addEventListener('click', () => {
            // Increment count and update the counter display
            count++;
            countSpan.textContent = count; // Update only the number part
        });
    </script>
</body>
</html>
```

```html
<body>
    <!-- The body contains the visible content of the page -->

    <!-- A button with an ID of 'btn' and initial text 'Count: 0' -->
    <button id="btn" aria-live="polite">Count: 0</button>
    <!-- 'aria-live="polite"' helps screen readers announce changes to the but

    <!-- JavaScript code goes here -->
    <script>
        // Initialize a counter variable to keep track of the number of clicks
        let count = 0;

        // Get a reference to the button element using its ID
        const button = document.getElementById('btn');

        // Add a click event listener to the button
        button.addEventListener('click', () => {
            // Increment the counter by 1 each time the button is clicked
            count++;

            // Update the button's text content to show the new count
            button.textContent = `Count: ${count}`;
        });
    </script>
</body>
```

Key Technical Difference:

- **Without span:** Every click forces the browser to:
    - Destroy the old button content
    - Recreate all text nodes ("Count: " and the number)
    - Recalculate styles/layout for the entire button
- **With span:** The browser only needs to:
    - Update the text node inside the span
    - Perform minimal layout recalculation

Why This Matters:

1.Performance: The span version is ~30-50% faster in repeated updates (JSBench test: jsbench.me/)

2.Accessibility: Screen readers get cleaner updates

3.Maintainability: Separates static/dynamic content

4.DOM Stability: Preserves any child elements/attributes

Live Demo Comparison:

(open browser dev tools > Performance tab to see the difference)

Record Ctrl + E

Live Demo Comparison

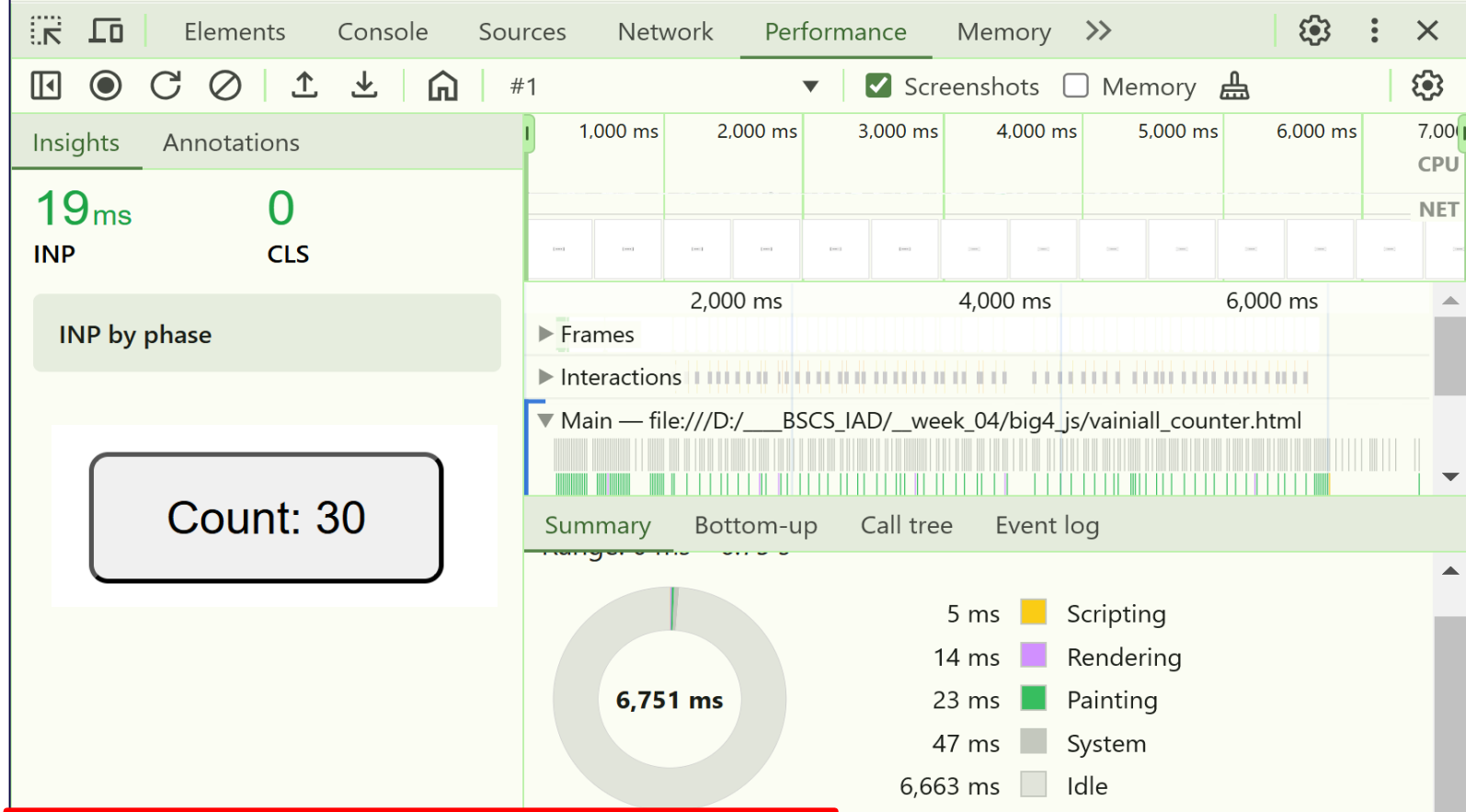
To see the difference in action:

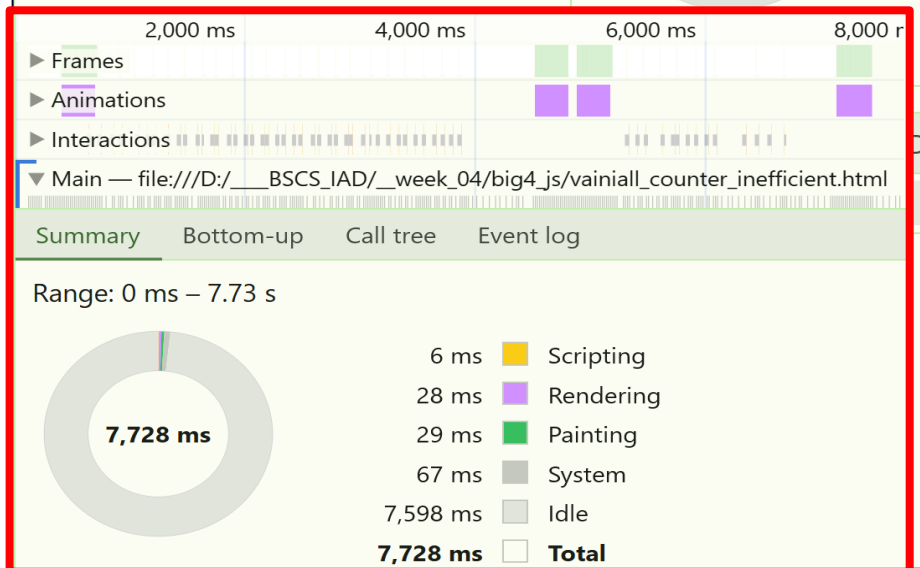
1. Open the browser's **Developer Tools** (F12).

2. Go to the **Performance** tab.

3. Compare the performance and accessibility behavior of the two versions:

- ✓ Without : Full button content is recreated.
- ✓ With : Only the number inside the is updated.





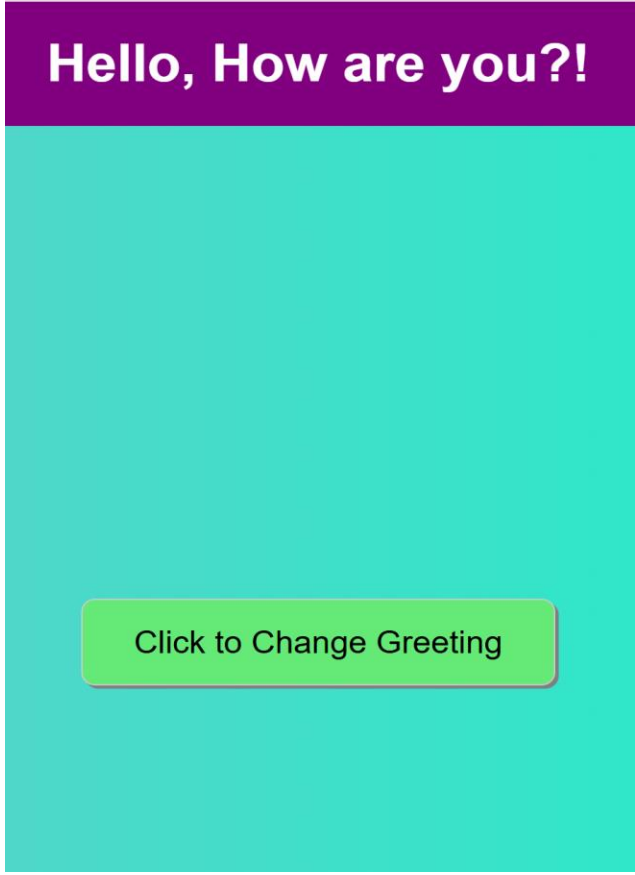
Let's create an interactive web application where users can dynamically update the greeting message displayed in a prominent header (<h1>) by clicking a button.

Demonstrates how JavaScript can directly manipulate the DOM (textContent of the <h1>) without any framework.

Vanilla JS Greetings(html, css, js)

Version 1

```html
<body>
  <!-- Static Greeting Header -->
  <h1>Hello, Vanilla JS!</h1>

  <!-- Container for Button -->
  <div class="container">
    <button id="greetBtn">Click to Change Greeting</button>
  </div>

  <script>
    // Function to handle dynamic greeting updates
    function updateGreeting() {
      const message = prompt("Enter a new greeting message:");
      if (message) {
        document.querySelector("h1").textContent = `Hello, ${message}!`;
      }
    }
    // Attach event listener to button
    document.getElementById("greetBtn").addEventListener("click", updateGreeting);
  </script>
</body>
</html>
```



Comparative approach to look at button click event in native react, modern react and redux app

What is React?

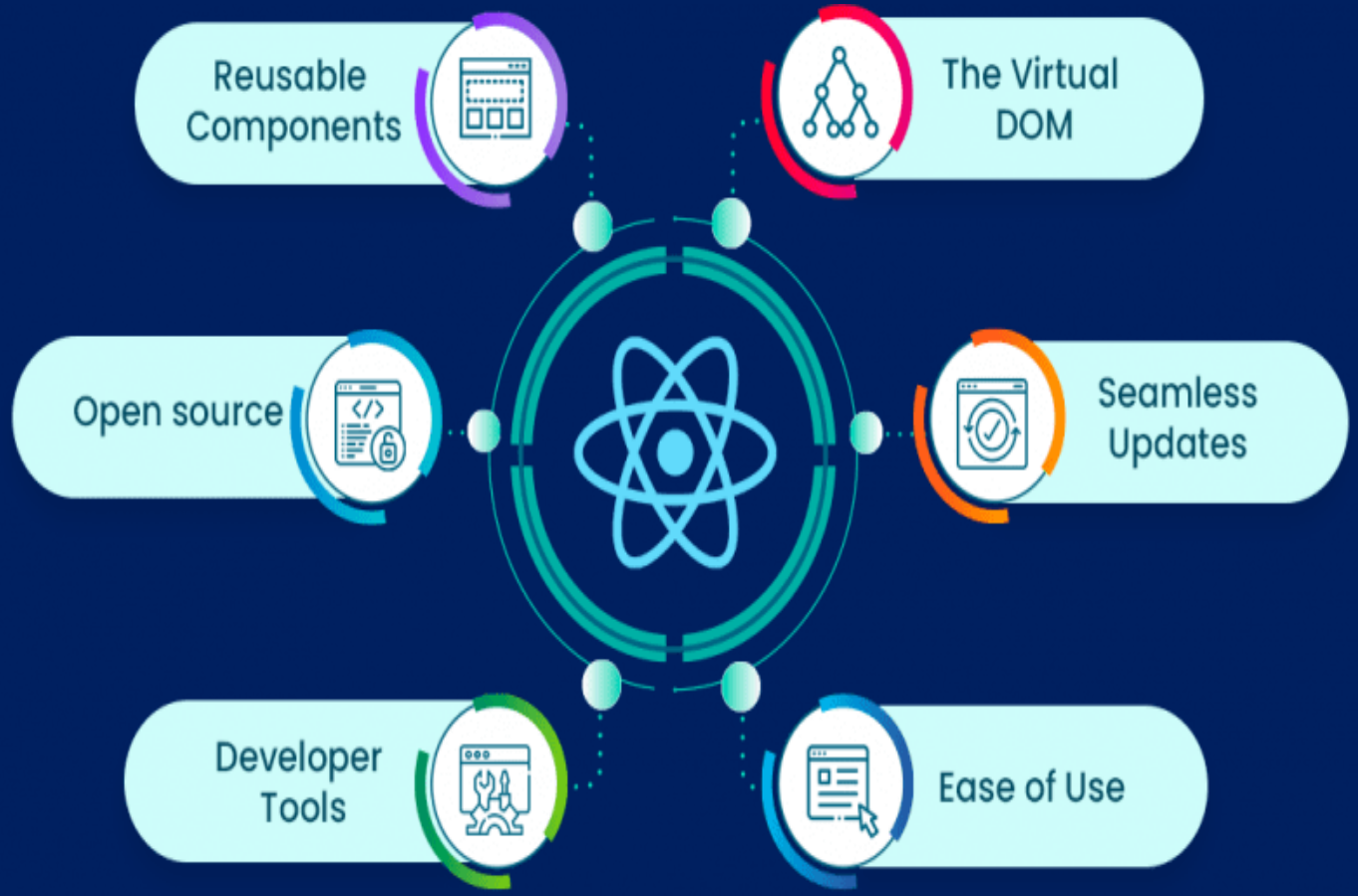
- ✓ React (aka React.js or ReactJS) is an **open-source front-end JavaScript library** that is used for building composable user interfaces, especially for ?? .

- ✓ It is used for handling view layer for web and mobile apps based on components in a ?? .

- ✓ React was created by Jordan Walke, a software engineer working for Facebook. React was first deployed on a ?? . **in 2011** and on a ?? **in 2012**.

UNIVERSITY OF KARACHI

✓ Major Features offered by React

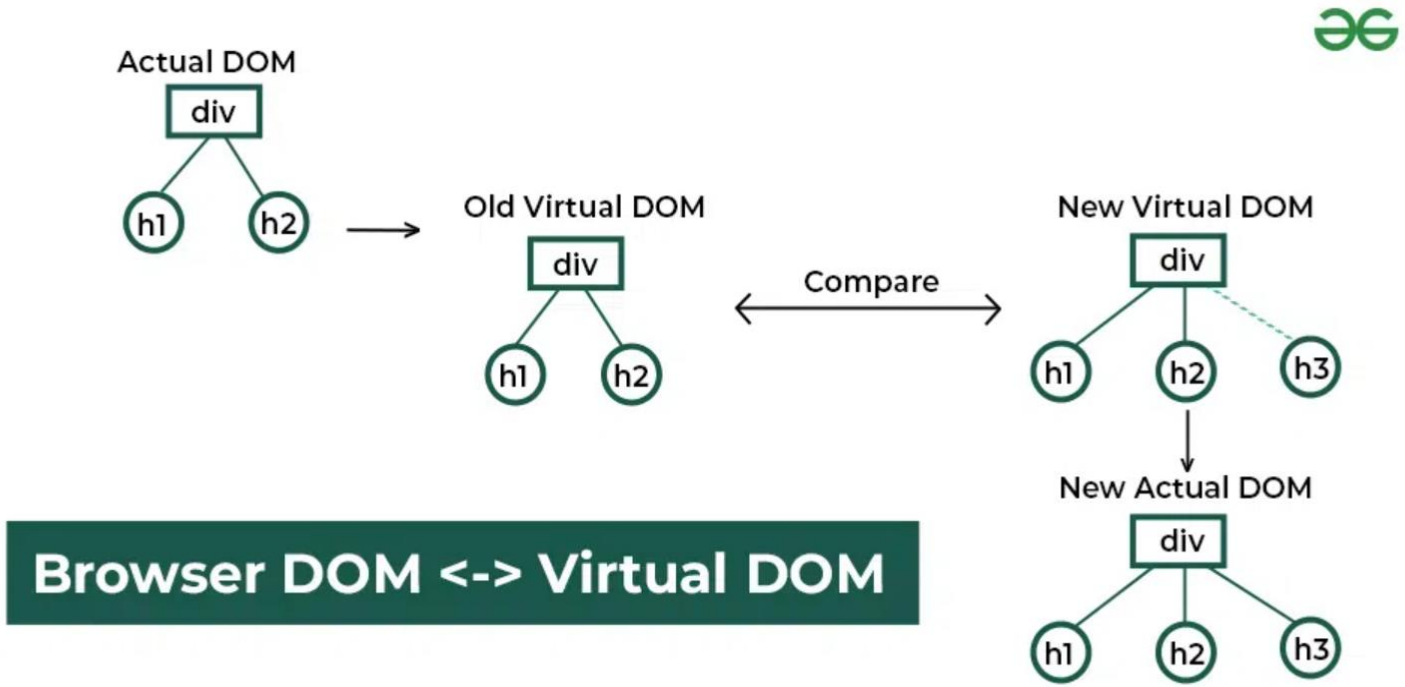


The major features of React are:

- ✓ Uses **JSX syntax**, a syntax extension of JS that allows developers to write HTML in their JS code.
- ✓ It uses **Virtual DOM** instead of Real DOM considering that Real DOM manipulations are expensive.
- ✓ Supports **server-side rendering** which is useful for Search Engine Optimizations(SEO).
- ✓ Follows **Unidirectional or one-way** data flow or data binding.
- ✓ Uses **reusable/composable** UI components to develop the view.

How Does the Virtual DOM Work?

1. **Rendering the Virtual DOM:** React creates a virtual representation of the UI as a tree of JavaScript objects.
2. **Updating State:** It generates a new Virtual DOM tree to reflect the updated state when the application state changes.
3. **Diffing Algorithm:** React compares the new Virtual DOM tree with the previous one using its efficient diffing algorithm to identify the minimal set of changes required.
4. **Updating the Real DOM:** React applies only the necessary changes to the real DOM, optimizing rendering performance.



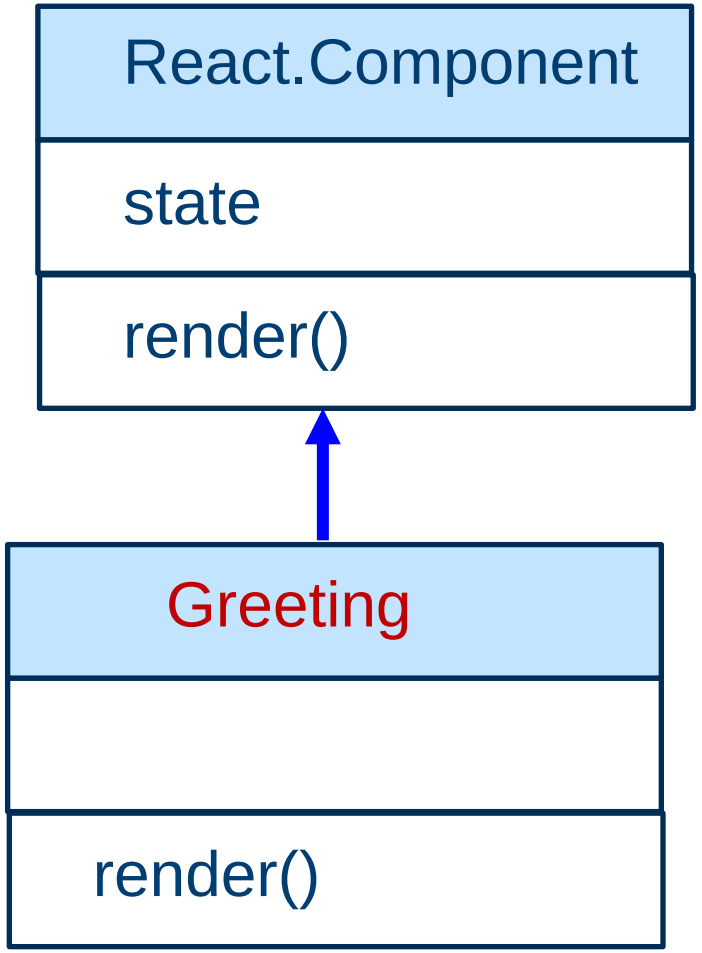
https://www.geeksforgeeks.org/reactjs-virtual-dom/

Virtual DOM	Real DOM
It is a ? of the original DOM	It is a ? representation of HTML elements
It is maintained by ? Libraries.	It is maintained by the ? after parsing HTML elements
After manipulation it only re-renders ? .	After manipulation, it re-render the ? .
Updates are lightweight	Updates are heavyweight
Performance is ? and UX is optimised	Performance is ? and the UX quality is low
Highly efficient as it performs ? .	Less efficient due to re-rendering of DOM after ? .

React Class Component

Class Components

Class components also known as stateful components contain state and lifecycle methods and are written with JavaScript ES6 classes.



UNIVERSITY OF KARACHI

What is state of component in React ?

State of a component is **an object** that holds some information **that may change over the lifetime** of the component. The important point is whenever the state object changes, **the component re-renders**. It is always recommended to **make state as simple as possible and minimize the number of stateful components.**

Whenever React calls your component it gives you a snapshot of the state for that particular render.

state is used for internal communication inside a Component

UNIVERSITY OF KARACHI

What is the difference between state and props?

In React, both **state** and **props** are **plain JavaScript objects** and used to manage the data of a component, but they are used in different ways and have different characteristics.

Props	State
props (short for "properties") are passed to a component by its parent component and are ____?____ meaning that they cannot be modified by the own component itself. **props acts as** an ____?____ **for a function.** Also, props can be used to ____?____ the behavior of a component and to ____?____ data between components. **The components become** ____?____ **with the usage of props.**	**The state entity** is managed by the component itself and can be ____?____ using the setter(setState() for class components) function. Unlike props, state can be modified by the component and is used to manage the internal state of the component. i.e. state acts as a component's memory. Moreover, changes in the state trigger a re-render of ____?____ .The components ____?____ with the usage of state alone.

Standard React component for creating UI (1)

A React **component** can be declared in several different ways.

It can be a **class with a render() method** or it can be **defined as a function**.

In either case, it takes props as an input, and returns a JSX tree as the output

```
class Greeting extends React.Component {
  render() {
    return <h1>{`Hello, ${this.props.message} `}</h1>;
  }  }
```

this.props.message are **placeholders** for dynamic content before rendering.

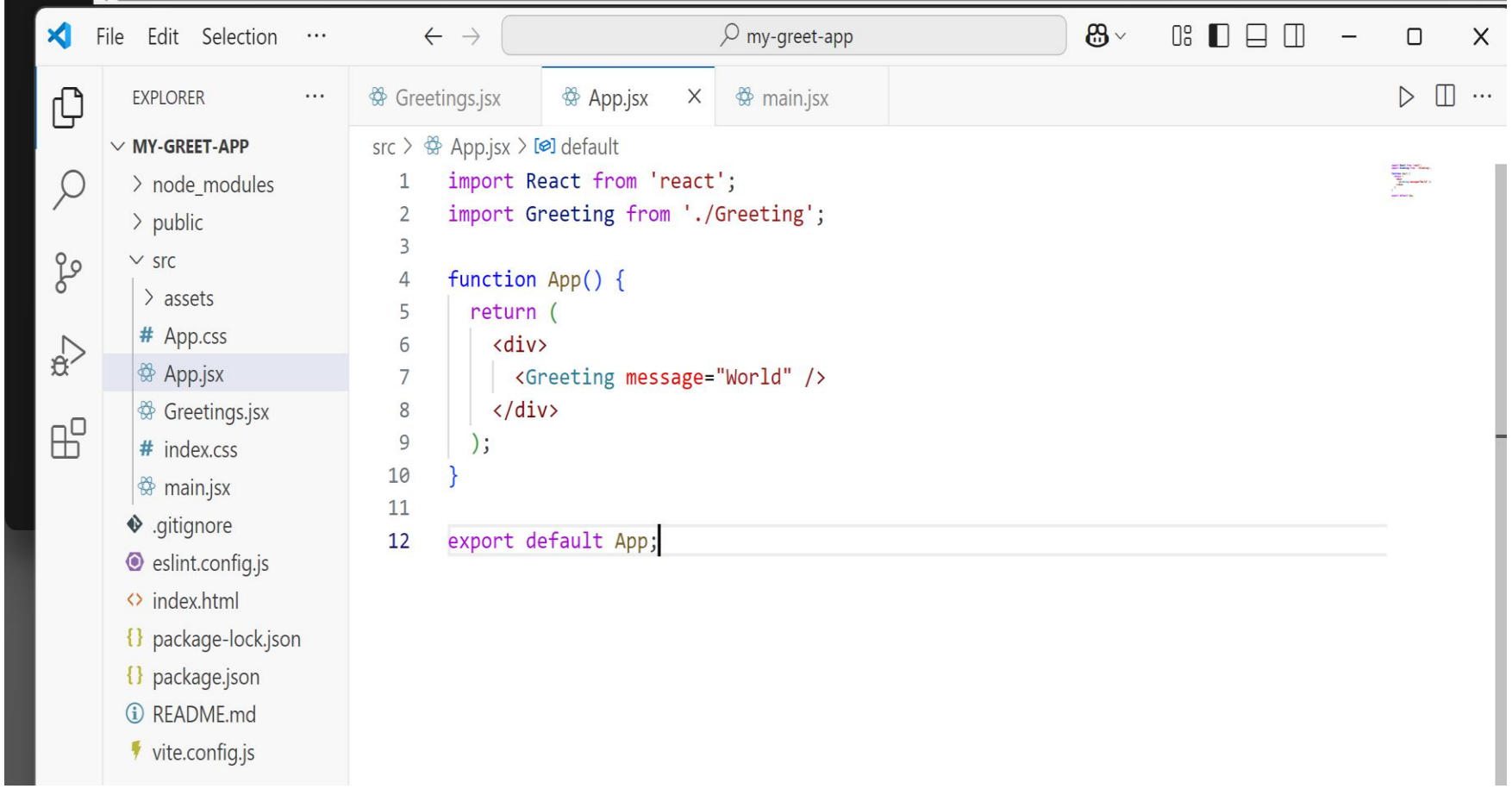
```
√ Select a framework: » React
√ Select a variant: » JavaScript
```

C:\Windows\system32\cmd.e

```
Microsoft Windows [Version 10.0.22631.4751]
(c) Microsoft Corporation. All rights reserved.

D:\____BSCS_IAD\__week_04\big4_js\react-greetings>npm create vite@latest my-greet-app --template react
? Select a framework: » - Use arrow-keys. Return to submit.
    Vanilla
    Vue
>   React
    Preact
    Lit
    Svelte
    Solid
    Qwik
    Angular
    Others
```

```
my-react-app
├── index.html
├── package.json
├── src
│   ├── App.jsx
│   ├── Greeting.jsx
│   └── main.jsx
└── vite.config.js
```

```
[plugin:vite:import-analysis] Failed to resolve import "./Greeting" from "src/App.jsx". Does the file exist?

D:/___BSCS_IAD/__week_04/big4_js/react-greetings/my-greet-app/src/App.jsx:2:21
16 |  }
17 |  import React from "react";
18 |  import Greeting from "./Greeting";
   |                       ^
19 |  function App() {
20 |    return /* @__PURE__ */ jsxDEV("div", { children: /* @__PURE__ */ jsxDEV(Greeting, { message: "World" },
```



Sign in | New tab | Vite + React

localhost:5173

Does our greet app were making any network requests to a backend API (e.g., POST requests) ??

Hello, World!

File Edit Selection ··· my-greet-app

EXPLORER

- MY-GREET-APP
  - node_modules
  - public
  - src
    - assets
    - App.css
    - App.jsx
    - Greetings.jsx
    - index.css
    - main.jsx
  - .gitignore
  - eslint.config.js
  - index.html
  - package-lock.json
  - package.json

Greetings.jsx | App.jsx | main.jsx

src > App.jsx > App

```jsx
import React from 'react';
import Greeting from './Greetings';

function App() {
  return (
    <div>
      <Greeting message="World" />
    </div>
  );
}

export default App;
```

Standard React component for creating UI (2)

A React **component** can be declared in several different ways.

It can be a **class with a render() method** or it can be **defined as a function**.

Functional Components

Functional components are a simple, fast, and easy way to design and develop a component in React. They are used to create components that return JSX and don't have their state.

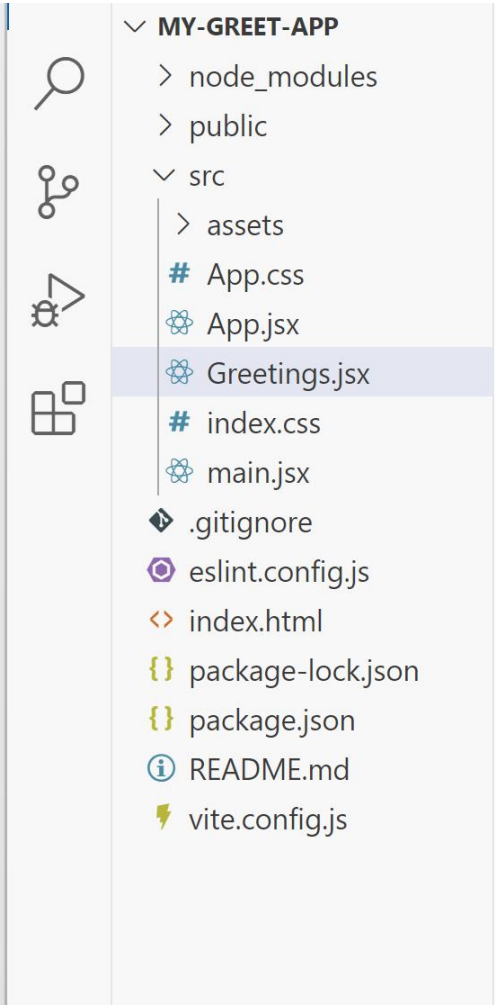
Functional components are pure JavaScript functions that accept props object as the one and only one parameter and return React elements to render the output:

```
function Greeting( { message } ) {
  return <h1> {`Hello, ${message}`} </h1>;
}
```

UNIVERSITY OF KARACHI

Replacing CC with FC

Hello, BSCS-633!



src > Greetings.jsx > default

```jsx
import React from 'react';


// class Greeting extends React.Component {
//   render() {
//     return <h1>{`Hello, ${this.props.message}!`}</h1>;
//   }
// }

// // Functional Component
// const Greeting = (props) => {
//     return <h1>{`Hello, ${props.message}!`}</h1>;
//   };

  // Functional Component with destructuring
const Greeting = ({ message }) => {
    return <h1>{`Hello, ${message}!`}</h1>;
  };



export default Greeting;
```

UNIVERSITY OF KARACHI

Component life cycle in OOP frameworks

Unity (C#)

- `Start()` : Called when the script is enabled.
- `Update()` : Called every frame.
- `OnDestroy()` : Called when the script is destroyed.

Android (Java/Kotlin)

- `onCreate()` : Called when an activity is created.
- `onStart()` : Called when the activity becomes visible.
- `onDestroy()` : Called when the activity is destroyed.

iOS (Swift/Objective-C)

- `viewDidLoad()` : Called when the view is loaded.
- `viewWillAppear()` : Called when the view is about to appear.
- `viewWillDisappear()` : Called when the view is about to disappear.

Lifecycle methods are functions that are automatically called at specific stages of a component's existence. They allow you to run code at key moments, such as when a component is created, updated, or destroyed.

UNIVERSITY OF KARACHI

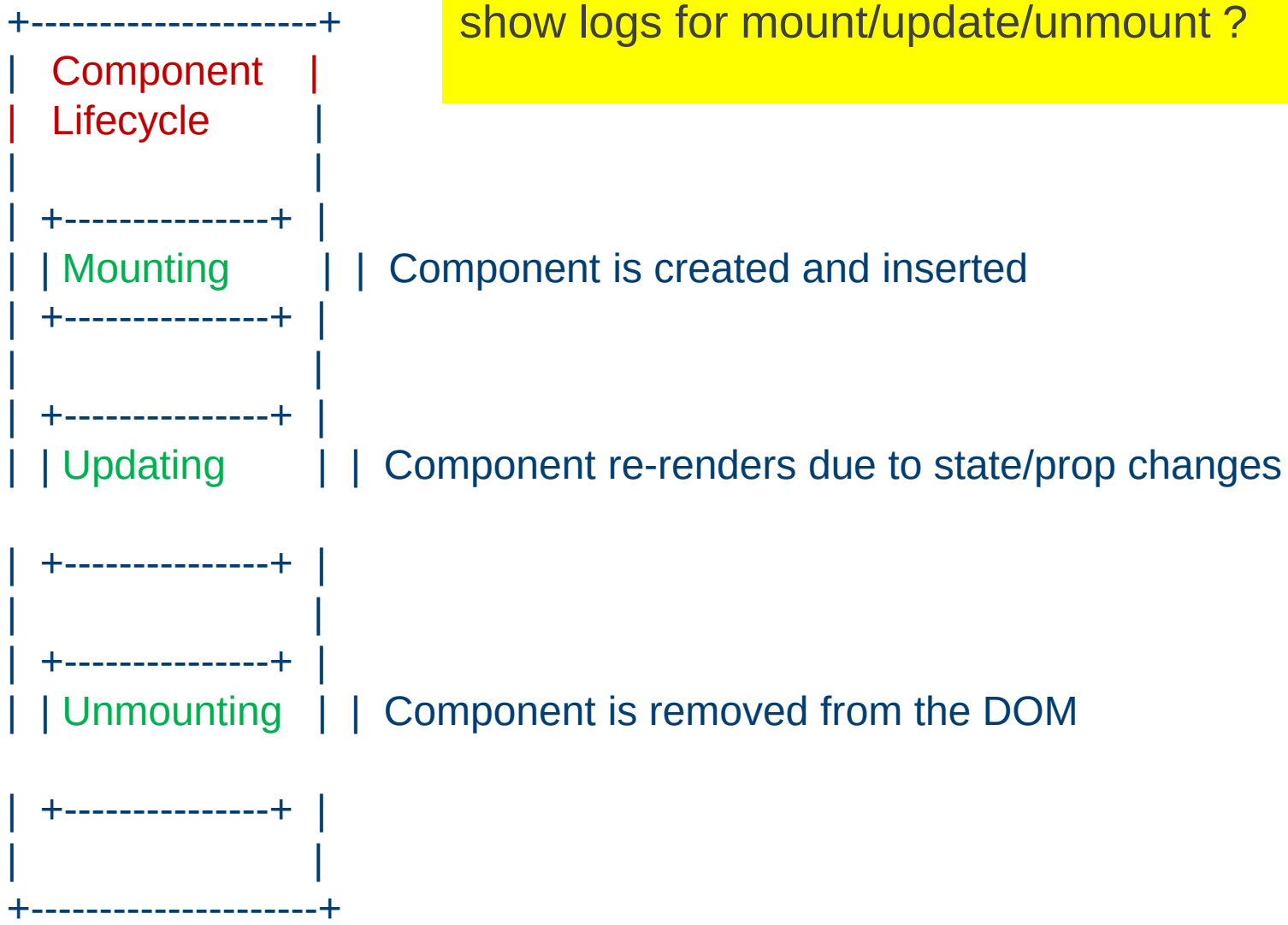
✓ Brief comparison: JS, native react, modern react

Aspect	Vanilla JS	Class Component	Functional Component (Modern)
State Storage	Global variables	`this.state` object	`useState` hook
Updates	Direct DOM manipulation	`this.setState()`	`setState` function from `useState`
Re-renders	Manual	Automatic on state change	Automatic on state change
Code Structure	Procedural	Class-based OOP	Functional programming
Boilerplate	None	High (classes, `this` binding)	Low (no classes, direct functions)

Component life cycle

Can we Build a simple app with useEffect and show logs for mount/update/unmount ?

Practice Assignment



UNIVERSITY OF KARACHI

Here's a simple and concise table to highlight the syntax differences between **native JavaScript**, **native React**, **modern React**, and **Redux** using a **counter example**:

Aspect	Native JavaScript	Native React (Class)	Modern React (Hooks)	Redux (Reducer)
State Initialization	`let count = 0;`	`this.state = { count: 0 };`	`const [count, setCount] = useState(0);`	`const initialState = { count: 0 };`
Event Handling	`button.addEventListener('click', () => { count++; });`	`this.handleClick = () => { this.setState({ count: this.state.count + 1 }); }`	`onClick={ () => setCount(count + 1) }`	`return { ...state, count: state.count + 1 };`

Assignment

Deadline: 15th March 2025

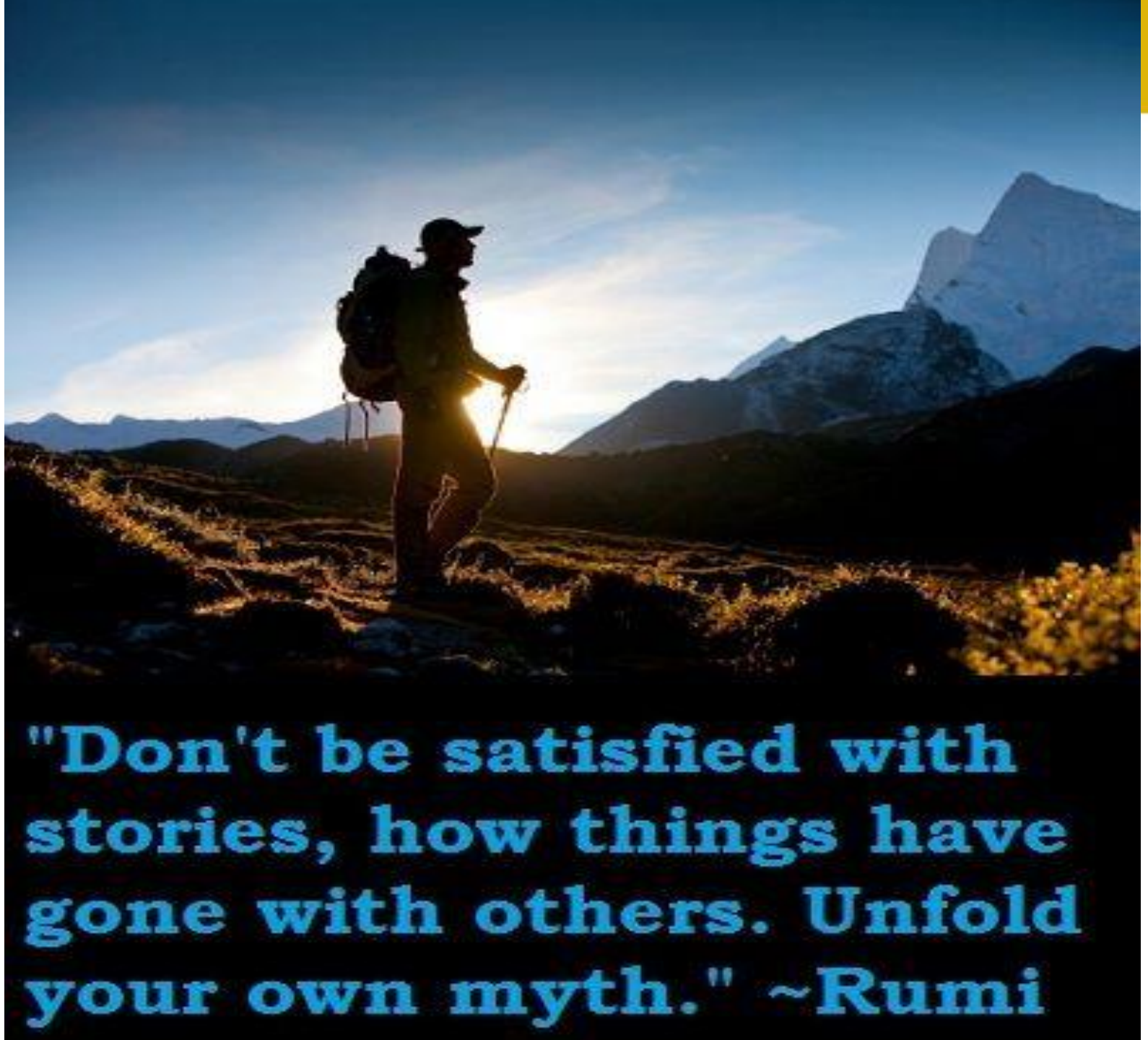
Exercise writing and running minimalist webapp

for comparison purpose in

Vanilla js, native react

Modern react and Redux

UNIVERSITY OF
KARACHI



UNIVERSITY OF KARACHI

Department of Compute Science (UBIT Building), Karachi, Pakistan.

1200 Acres (5.2 Km sq.)
53 Departments
19 Institutes
25000 Students

UNIVERSITY OF KARACHI

My Homeland Pakistan