CH – 1

EXISTING SYSTEM STUDY

1.1 Introduction

Existing System Study involves Study of an Existing System, Documenting of Existing System & identifying Current Deficiencies and Establishing New Goals. It usually includes an Analysis of the Existing System and the Design of the New System, including the Development of System Specifications which provide a basis for the selection of Equipment.

1.2 Drawbacks

Teachers:

- Teachers had to send notes and assignments to all students individually, either by email, WhatsApp, or by printing them out and distributing them in class. This was a time-consuming and tedious process, and it was difficult to ensure that all students received the same information.
- Teachers also had to keep track of which students had submitted their assignments, and they had to grade and return the assignments to students. This was another time-consuming task.

Students:

- Students had to purchase textbooks and sample papers, which could be expensive, especially for students from low-income families.
- Students often had to carry heavy books to and from school, which could be uncomfortable and inconvenient.
- Students sometimes forgot their books at home, which could disrupt their studies.

Overall, the current system was inefficient and time-consuming for both teachers and students. It was also expensive for students, and it could sometimes disrupt their studies.

CH – 2

SCOPE AND OBJECTIVE

2.1 Project Scope

The project scope is the definition of what work will be done in a project. It should be clear, concise, and complete, and it should be agreed upon by all stakeholders. The project scope should include the following elements:

- The work that needs to be done to complete the project
- The deliverables that will be produced
- The criteria for success
- The constraints on the project, such as budget, timeline, and resources
- Developing custom content for the website
- Integrating the website with the school's existing systems
- Providing technical support to students and teachers

2.2 Project Objective

To develop a functional science website that will:

- Provide students with access to interactive lessons, tutorials, simulations, and experiments on a variety of functional science topics.
- Provide students with opportunities to practice their functional science skills through quizzes and assessments.
- Provide students with a forum to discuss functional science concepts and ideas with their peers and with experts in the field.
- Provide teachers with resources to supplement their functional science teaching.
- Make functional science more accessible and engaging for students of all levels.

CH - 3

REQUIREMENT

3.1 Software

- **Code Editor**
- **Web Browser**
- **Npm**
- **Git and Github**
- **React**

3.1.1 Code Editor

A code editor is a software application that is specifically designed for editing source code. Source code is the text that programmers use to create software programs. Code editors typically provide a variety of features that make it easier to write, edit, and format source code, such as syntax highlighting, auto-completion, and error checking.I used *'VS Code'* as code editor.

3.1.2. Web Browser

A web browser (commonly referred to as a browser) is a software application for accessing information on the World Wide Web. When a user requests a web page from a particular website, the web browser retrieves the necessary content from a web server and then displays the page on the user's device. We used ***'Google Chrome'*** and **'*Microsoft Edge*'** as a browser for our project.

3.1.3 Git & Github

Git and GitHub are two popular tools that are used by software developers to manage and collaborate on code. Git is a distributed version control system that allows developers to track changes to their code over time and to work on different versions of the code simultaneously. GitHub is a code hosting platform that provides a way for developers to share and collaborate on their code.I have installed the Git locally and also I have used Github pages for my project deployment.

3.1.4 npm Package Manager

npm is a package manager for the JavaScript programming language maintained by npm, Inc. npm is the default package manager for the JavaScript runtime environment Node.js and is included as a recommended feature in the Node.js installer.

3.1.5 React,Vite and its required modules

React:- React is a JavaScript library for building user interfaces. It is declarative, efficient, and flexible. React makes it easy to create interactive UIs by using a component-based approach

Vite:- Vite is a build tool for developing modern web applications with Vue, React, and Svelte. It is fast, lightweight, and easy to use. Vite uses a native ESM-based dev server, which means that it can serve code directly from the file system without having to transpile it first. This makes Vite very fast and efficient.

To develop a React application with Vite, you will need to install the following modules:

- react
- react-dom
- Vite
- react-icons

npm create vite

In addition to the required modules, you may also want to install the following modules:

- @types/react
- @types/react-dom
- @vitejs/plugin-react

3.2 Hardware

This website is made on a laptop with configuration as:

- **Latest OS**: Linux Ubuntu version 22 LTS
- **Processor:** i7 11th gen
- **RAM:**16 GB (DDR5)
- **SSD Space:** 512 GB

3.3 Tools & Methodology

3.3.1 Introduction

Web development refers to the building, creating, and maintaining of websites. It includes aspects such as web design, web hosting, web programming, and database management. It is the creation of an application that works over the internet i.e. websites.

It is very much popular nower days.

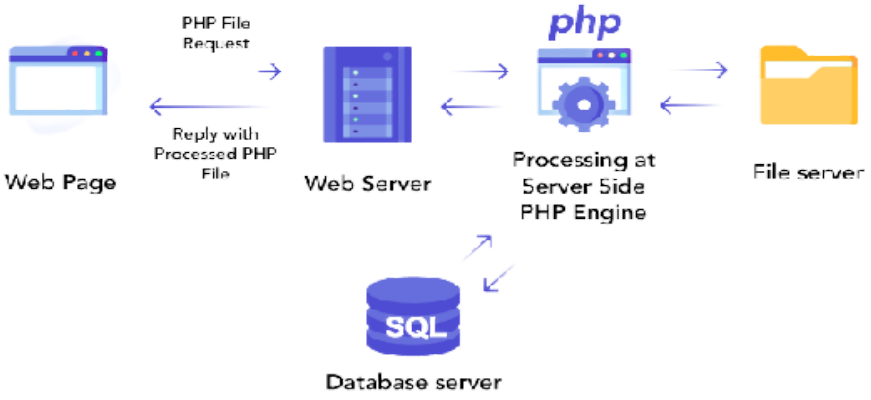
There are two ways to create a websites:

1. Dynamic Websites
2. Static Websites

3.3.2 Dynamic Website

A dynamic website is a website that can generate content on the fly, in response to user input or other factors. This is in contrast to a static website, which simply serves the same pre-generated content to all users.

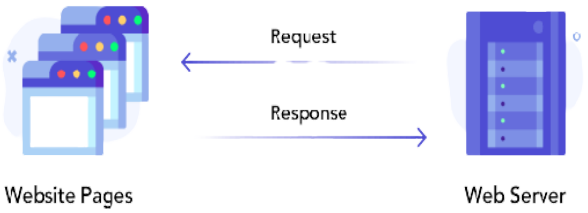
Dynamic websites are typically created using server-side scripting languages, such as PHP, Python,JS, or Ruby. These languages allow developers to write code that can generate different content based on the input that they receive.



3.3.3 Static Website

A static website is a website that serves the same pre-generated content to all users. This content is typically stored in HTML, CSS, and JavaScript files, which are then served to the user's web browser when they request a page.

Static websites are typically easier to create and maintain than dynamic websites, as they do not require any server-side scripting. However, static websites are also less flexible and cannot provide the same level of personalization or interactivity as dynamic websites.



3.3.4 Types of Web Development

Web Development can be classified into two types:

1. Frontend Web Development
2. Backend Web Development

3.3.5 Frontend Web Development

Front-end web development is the process of creating the user interface and user experience of a website. It involves the use of HTML, CSS, and JavaScript to create the layout, style, and interactivity of a website. Front-end developers are responsible for making sure that a website is easy to use and navigate, and that it looks good on all devices.

Roadmap of Frontend Web Development:

- **Structure Language:** HTML
- **Styling Language:** CSS
- **CSS Framework:** Bootstrap, Tailwind CSS..
- **Programming Language :** JavaScript
- **JS Framework:** ReactJS, AngularJS, VueJS.
- **CSS preprocessor:** SCSS

3.3.6 Backend Development

Backend web developers also need to be familiar with databases, web servers, and security. Some of the most popular databases include MySQL, PostgreSQL, and MongoDB. Some of the most popular web servers include Apache, Nginx, and IIS.

Roadmap of Backend Web Development :

- **PHP:** PHP is a server-side scripting language designed specifically for web development.
- **Java:** Java is one of the most popular and widely used programming language. It is highly scalable.
- **Python:** Python is a programming language that lets you work quickly and integrate systems more efficiently.
- **Node.js:** Node.js is an open source and cross-platform runtime environment for executing JavaScript code outside a browser.
- **Back End Frameworks:** The list of backend frameworks are:Express, Django, Rails, Laravel, Spring, etc.
- **SQL Database:** MariaDB, MySQL.
- **NoSQL Database:** Mongo DB.

3.3.7 HTML

3.3.7.1 What is Html?

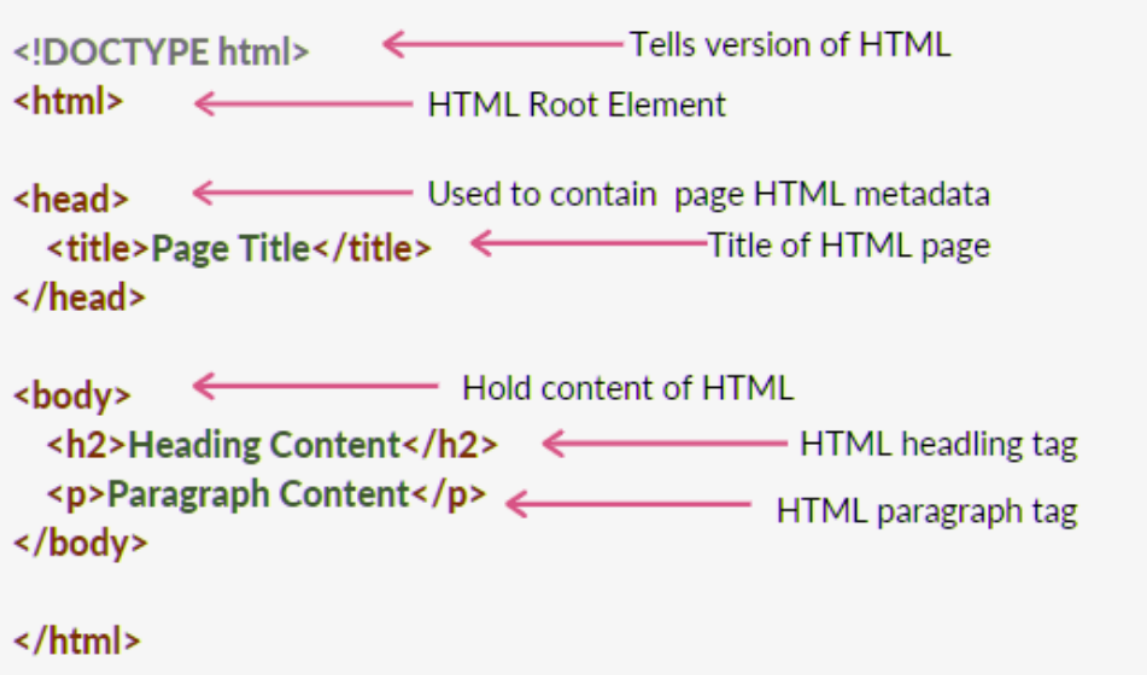
HTML stands for HyperText Markup Language. It is the standard markup language for creating web pages. HTML describes the structure of a web page by using a series of tags. These tags tell the browser how to display the content of the page.

3.3.7.2 What is Html used for?

HTML is used to create the structure and content of web pages. It is the standard markup language for the World Wide Web, and it is used by browsers to render web pages.

HTML is used to define headings, paragraphs, lists, tables, images, links, and other elements of a web page. It can also be used to create forms, embed videos, and add other interactive features to a web page.HTML is a relatively simple language to learn, and it is a good starting point for anyone who wants to learn web development. There are many resources available online and in libraries that can teach you the basics of HTML.

3.3.7.3 Basic Structure of Html



- **<!DOCTYPE HTML> –** A DOCTYPE or document type declaration is an instruction that tells the web browser about the markup language in which the current page is written. It is not an element or tag. The DOCTYPE declaration is not case-sensitive.

- **<HTML> –** This tag is used to define the root element of HTML document. This tag tells the browser that it is an HTML document. It is the second outer container element that contains all other elements within it.
- **<HEAD> –** This tag is used to define the head portion of the HTML document that contains information related to the document. Elements within the head tag are not visible on the front-end of a webpage.
- **<TITLE> –** It is used to create a title of the document and the title appears in the title bar at the top. At least one title appears in every document. The title portion of the document starts with <title> and ends with </title>, and in between, enter the text that you want as the title.
- **<BODY> –** The body tag is used to enclose all the visible content of a webpage. In other words, the body content is what the browser will show on the front end.

3.3.7.4 History of HTML

HTML was invented by Tim Berners-Lee in 1989 while he was working at CERN, the European Organization for Nuclear Research. Berners-Lee was looking for a way to share information with other researchers at CERN, and he developed HTML as a way to create linked documents that could be accessed over the internet.

3.3.7.5 Versions of Html?

1. **1.0 Html:**

   HTML 1.0 was the first version of the HyperText Markup Language (HTML), the standard markup language for creating web pages. It was released in 1991 and was a relatively simple language, but it was enough to create basic web pages with text, images, and links.

2. **2.0 Html:**

   HTML 2.0 was the second major version of the HyperText Markup Language (HTML), the standard markup language for creating web pages.

It was published in 1995 and added a number of new features to HTML 1.0, including support for forms, tables, style sheets, scripts, and frames. These new features made it possible to create more complex and interactive web pages.

3. 3.2 Html:

HTML 3.0 was released in 1996 and added new features such as support for maths and scientific notation, applets and other embedded content, multiple style sheets, client-side image maps, and better accessibility features. It also made improvements to existing elements in HTML 2.0, such as adding support for nested tables, images within links, and background images.

4. HTML 4.01:

It was developed in 1999. It extended the support of cascading styling sheets. In version 3.2, CSS were embedded in HTML page itself. Therefore, if the website has various web pages to apply to the style of each page, we must place CSS on each web page. Hence there was a repetition of the same block of CSS.

5. Html 5.0:

This is the latest version of HTML. For a developer, it could be used in 2014. It came up with lots of HTML tags support. HTML5 provided support for new form elements like input element s of different types; geolocations support tags, etc. Some new tags are:

- **Email –** New HTML5 tag, which was added, is the input element of type email. This is a form tag, although it could be used outside of a form tag also. This tag checks the validation of the input value. It checks whether the value inserted is a valid email.

- **Password –** This is another form tag that was added to receive a password from the user. Being the password type field, the user types in the field are not visible directly to the user but are represented by special symbols. These symbols save the password from getting revealed on the browser.
- **Audio tag –** This is a new audio tag that was implemented in HTML5. This tag helps to add audio to our web page. We can use this tag to embed an audio clip into a web page. This audio tag could be played on a webpage.
- **Semantic tags –** Semantic tags are also known as structural tags. Structural tags are the tags that provide structure to the HTML page. It helps it divide the HTML page into different structures. These structures get combined into an HTML page itself to form an HTML web page. Few of the important HTML semantic tags are figcaption, <header>, <footer>
- **Section tag –** This tag is used to semantic a section in an HTML page. A section tag represents a section on a web page.

3.3.7.6 HTML Formatting Tags

HTML Formatting is a process of formatting text for better look and feel. HTML provides us the ability to format text without using CSS. There are many formatting tags in HTML.

3.3.7.7 Some Tags of Html

Structural tags

- <!DOCTYPE html>: Defines the type of document as an HTML document.
- <html> and </html>: Enclose the entire HTML document.
- <head> and </head>: Contain information about the document, such as the title, meta tags, and style sheets.
- <body> and </body>: Contain the content of the document, such as text, images, and links.

Text formatting tags

- <h1> to <h6>: Define headings of different levels.
- <p>: Defines a paragraph.
-
: Inserts a line break.

- <hr>: Inserts a horizontal rule.
- and : Define an unordered list.
- and : Define an ordered list.
- : Defines a list item.
- <dl> and </dl>: Define a definition list.
- <dt>: Defines a term in a definition list.
- <dd>: Defines a definition in a definition list.
- <code> and </code>: Enclose code.
- <pre> and </pre>: Enclose preformatted text.

Link tags

- <a> and : Define a link.
- <img>: Inserts an image.
- <area>: Defines an area within an image that is a link.

Form tags

- <form> and </form>: Define a form.
- <input>: Creates an input field, such as a text field, checkbox, or radio button.
- <select> and </select>: Creates a select box, which allows users to select one or more options from a list.
- <textarea> and </textarea>: Creates a textarea field, which allows users to enter a block of text.
- <button> and </button>: Creates a button.

Table tags

- <table> and </table>: Define a table.
- <tr> and </tr>: Define a row in a table.
- <td> and </td>: Define a cell in a table.
- <th> and </th>: Define a header cell in a table.

Other tags

- <script> and </script>: Enclose JavaScript code.
- <style> and </style>: Enclose CSS code.
- <meta>: Provides meta information about the document, such as the character encoding and keywords.
- <title> and </title>: Define the title of the document, which is displayed in the browser's title bar.

3.3.7.8 Advantages of HTML:

- Basically HTML5 has its many new syntactical features, which include the <video>, <audio>elements, as well as the integration of SVG content. Due to these new elements, it will be very easy to integrate multimedia and graphical content to web without using flash and third party plug-in. There are also another new elements like <section>, <article>, <header> and <nav> which enrich the semantic value of the document. Other major advantages of HTML5 are described below.

- **Mutuality:** Due to usability purpose the web sites made by developers are highly interactive nowadays and for this developers need to include fluid animations, stream video, play music and Social Network sites like Facebook and Twitter into the websites. Till now they have only the option to integrate it with the help of Flash or Silver light, Flex or java script like tools. But these consume so much time to develop and even the complexity of web application also increased.
- **Cleaner markup / Improved Code:** HTML 5 will enable web designers to use cleaner, neater code; we can remove most div tags and replace them with semantic HTML 5 elements. Improved Semantics: Now it is easy to see which parts of the page are headers, nav, footers, aside, etc as the tags are specific for these all and most importantly know what their meaning and purpose is in whole the format
- **Elegant forms:** HTML5 enables designer to use fancier forms. Even it makes form validation native to HTML, User interface enhancements and reduced need

for JavaScript (only needed in browsers that don't support form types). There will be different type of text inputs, search and different fields for different purpose.

- **Consistency:** As websites adopt the new HTML5 elements we will see greater consistency in terms of the HTML used to code a web page on one site compared to another. This will make it easier for designers and developers to immediately understand how a web page is structured.
- **Improved Accessibility:** Different technologies can elaborate on the features with the help of HTML5, as they can immediately make more detailed understanding of the structure of a page by take a look at HTML5 elements it has.
- **Fulfil the need of Web application:** Many new features and standards have emerged as part of HTML 5. Once you detect the available features in today's browsers, you can take advantage of those features in your application. Main focus of HTML5 is to make easier application with easy front-ends, drag and drop tools, discussion boards, wikis and other useful elements.

3.3.7.9 Disadvantages of HTML:

- HTML has a few significant drawbacks, such as its static nature, its inability to render content in an aesthetically pleasing way, its well-known compatibility issues and its overall complexity.
- **Insufficient for Dynamic Pages:** Back in the early days of the World Wide Web, no one expected a Web page to do anything besides displaying static words and images, much like a book does. Nowadays, Internet users expect more out of their favourite websites, from infinite scrolling pages such as the Twitter timeline to search boxes that automatically generate suggestions based on input.
- **Limited for Displaying Content:** HTML is a structuring language that allows you to attach a virtual label to sections of your content. To circumvent that limitation, a separate language had to be invented to handle the presentation of Web pages Cascading Style Sheets. In effect, this limitation forces Web designers and developers to maintain two separate sets of files: HTML files that

contain the content of the website and structures it, and CSS files that describe how a page should look.

3.3.8 What is CSS?

Cascading **S**tyle **S**heets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable.

CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs, variations in display for different devices and screen sizes as well as a variety of other effects.

3.3.8.2 Versions of CSS:

- **CSS 1.0:** The CSS Level 1 W3C Recommendation was made in December 1996. The release of CSS 1 supported: font properties, text attributes, alignment of text, tables, images and more, colors of text and backgrounds, spacing of words, letters and lines, margins, borders, padding and positioning, and unique identification and classification of groups of attributes.
- **CSS 2.0:** In May of 1998, the W3C released CSS 2 which added new capabilities including z-index, media types, bidirectional text, absolute, relative and fixed positioning, and support for aural style sheets. Not long after the release of CSS 2, a new browser, Opera was released which also supported CSS.
- **CSS 3.0:** With the release of CSS 3, capabilities were broken into modules. Between June 2011 and June 2012, the following four modules were released as formal recommendations: color, selectors' level 3, and namespaces and media queries.

3.3.8.3 Types of Stylesheet:

- External Stylesheet
- Internal Stylesheet
- Inline Stylesheet

✔ **External Stylesheet:** With an external style sheet, you can change the look of an entire website by changing just one file! Each page must include a reference to the external style sheet file inside the element. The element goes inside the section:

Syntax:

```
<Head>
<link rel=”stylesheet” href=”style.CSS>
</head>
```

✔ **Internal Stylesheet:** An internal style sheet may be used if one single page has a unique style. Internal styles are defined within the section of an HTML page:

Syntax:

```
<head>
<Style>
<Body> {
Background-color: red;
}
h1 {
Color: maroon;
Margin-left: 40px;
}
</style>
</head>
```

✔ **Inline Stylesheet:** An inline style may be used to apply a unique style for a single element. To use inline styles, add the style attribute to the relevant

element. The style attribute can contain any CSS property. The example below shows how to change the color and the left margin of a element.

Syntax:

<h1 style="color:blue; margin-left: 30px ;">This is a heading</h1>

3.3.8.4 Advantages of CSS:

- **CSS saves time:** When most of us first learn HTML, we get taught to set the font face, size, color, style etc. every time it occurs on a page. This means we find ourselves typing (or copying & pasting) the same thing over and over again. With CSS, you only have to specify these details once for any element. CSS will automatically apply the specified styles whenever that element occurs.
- **Pages load faster:** Less code means faster download times.
- **Easy maintenance:** To change the style of an element, you only have to make an edit in one place.
- **Superior styles to HTML:** CSS has a much wider array of attributes than HTML

3.3.8.5 Disadvantages of CSS:

- **Browser compatibility:** Browsers have varying levels of compliance with Style Sheets. This means that some Style Sheet features are supported and some aren't. To confuse things more, some browser manufacturers decide to come up with their own proprietary tags.
- CSS works differently on different browsers. IE and Opera supports CSS as different logic.

3.3.8.6 Properties of CSS

1.Appearance

- Color: Defines the color of text, borders, and other elements on a page.
- Font: Defines the font family, font size, and other font properties.

- Background: Defines the background color, image, and repeat properties of an element.
- Border: Defines the style, width, and color of an element's border.
- Text: Defines the text alignment, decoration, and other text properties.

2.Layout

- Display: Defines how an element is displayed, such as block, inline, or none.
- Position: Defines the position of an element on a page, such as relative, absolute, or fixed.
- Float: Defines whether an element floats to the left or right.
- Margin: Defines the space around an element.
- Padding: Defines the space between an element's border and its content.
- Width: Defines the width of an element.
- Height: Defines the height of an element.

3.Effects

- Box-shadow: Adds a shadow to an element.
- Border-radius: Rounds the corners of an element's border.
- Transition: Defines how an element's properties change over time.
- Transform: Applies transformations to an element, such as scaling, rotation, and skewing.
- Opacity: Defines the transparency of an element.

4.Other

- Cursor: Defines the cursor that is displayed when hovering over an element.
- Content: Defines the content of an element, such as images, text, or other HTML elements.
- Outline: Adds an outline to an element.
- Visibility: Defines whether an element is visible or hidden.

CSS properties can be combined to create a wide variety of effects. For example, the following CSS code defines a button with a rounded border, a gradient background, and a shadow:

```
button {
  border-radius: 5px;
  background: linear-gradient(to bottom, #0000ff, #fff);
  box-shadow: 0px 0px 5px 0px #000000;
}
```

CSS properties are a powerful tool for controlling the appearance and behavior of web pages. By understanding how CSS properties work, developers can create web pages that are visually appealing and easy to use.

Here are some additional examples of how CSS properties can be used:

- To centre an element on the page:

```
element {
  margin: 0 auto;
}
```

- To create a responsive layout that adapts to different screen sizes:

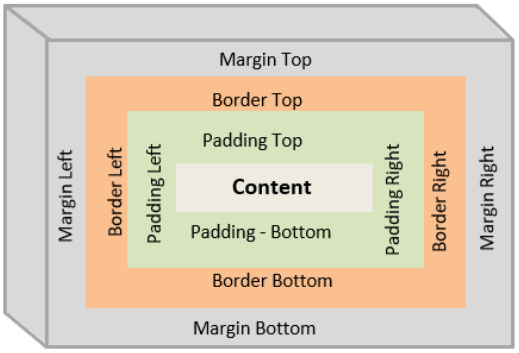
```
@media (max-width: 768px) {
  element {
    width: 100%;
  }
}
```

```
.dropdown {
  position: relative;
  display: inline-block;
}
.dropdown:hover .dropdown-content {
  display: block;
}
```

3.3.8.7 CSS Box Model

The CSS box model is a conceptual model that describes how HTML elements are displayed on a web page. It is a rectangular box that surrounds every element, and it consists of four parts: content, padding, border, and margin.

1. Content:-The content of the box is the actual content of the element, such as text, images, or other HTML elements.



2.Padding:-The padding is the space between the content of the box and its border.

3.Border:-The border is the line that separates the padding from the margin.

4.Margin:-The margin is the space between the border of the box and the other elements on the page.

The CSS box model is important because it allows developers to control the layout and appearance of web pages. By adjusting the padding, border, and margin of elements, developers can create web pages that are visually appealing and easy to use.

3.3.8.8 FlexBox

CSS Flexbox is a one-dimensional layout module that allows you to align and distribute space among items in a container, even when their size is unknown and/or dynamic. It is a powerful tool for creating responsive and flexible layouts.

Flexbox works by laying out items in a container along a main axis, which can be either horizontal or vertical. The main axis is defined by the flex-direction property.

Flexbox properties

The following are some of the most important Flexbox properties:

- flex-direction: Defines the direction of the main axis. Can be row, row-reverse, column, or column-reverse.
- flex-wrap: Defines whether items should wrap to a new line when there is not enough space on the current line. Can be wrap, nowrap, or wrap-reverse.
- justify-content: Defines how items are distributed along the main axis. Can be flex-start, flex-end, center, space-between, or space-around.
- align-items: Defines how items are aligned along the cross axis. Can be flex-start, flex-end, center, stretch, or baseline.
- align-content: Defines how items are aligned along the cross axis when there is extra space on the main axis. Can be flex-start, flex-end, center, space-between, or space-around.

Flexbox example

The following CSS code creates a flexbox container with two items:

```
.container {
  display: flex;
  flex-direction: row;
  justify-content: space-between;
```

```
}

.item {
  width: 100px;
  height: 100px;
  background-color: blue;
  margin: 5px;
}
```

3.3.8.9 Grid

CSS Grid is a two-dimensional grid layout system that allows you to create complex layouts with ease. It is a powerful tool for creating responsive and flexible layouts, and it can be used for a variety of tasks, including:

- Creating responsive layouts that adapt to different screen sizes and devices.
- Aligning and distributing items in a grid, even when their size is unknown and/or dynamic.
- Creating complex layouts with multiple columns and rows.
- Creating nested grids.
- Creating floating elements.

CSS Grid properties

The following are some of the most important CSS Grid properties:

- display: Defines whether an element is a grid container. Can be grid or inline-grid.
- grid-template-columns: Defines the columns of the grid.
- grid-template-rows: Defines the rows of the grid.

- grid-gap: Defines the spacing between grid items.
- grid-column: Defines the column in which a grid item is placed.
- grid-row: Defines the row in which a grid item is placed.
- grid-area: Defines the area in the grid in which a grid item is placed.
- justify-content: Defines how grid items are distributed along the main axis of the grid.
- align-items: Defines how grid items are aligned along the cross axis of the grid.

CSS Grid example

```
.container {
  display: grid;
  grid-template-columns: 1fr 2fr;
  grid-template-rows: 1fr 2fr;
}
.item {
  background-color: blue;
  margin: 5px;
}
```

This will create a container with four items, each of which is placed in a different grid cell. The items will be distributed evenly across the grid, and the grid will adapt to different screen sizes and devices.

3.3.9.1 Typescript

TypeScript is a programming language developed by Microsoft that builds on JavaScript by adding static typing. This means that TypeScript adds syntax to JavaScript, allowing developers to add types. TypeScript is a "syntactic superset"

of JavaScript, which means that it shares the same base syntax as JavaScript, but adds something to it.

3.3.9.2 Datatypes and Variables in typescript

TypeScript variables and datatypes are similar to JavaScript variables and datatypes, but TypeScript adds the ability to specify the type of a variable when it is declared. This helps to catch errors early and makes the code more readable and maintainable.

Declaring variables in TypeScript

To declare a variable in TypeScript, you use the `let` or `const` keyword, followed by the variable name and a colon, followed by the type of the variable. For example:

```
let myNumber: number = 10;
const myString: string = "Hello, world!";
```

The `let` keyword declares a variable that can be reassigned later. The `const` keyword declares a constant that cannot be reassigned.

TypeScript datatypes

TypeScript has a number of built-in datatypes, including:

- `number`: Represents a numeric value.
- `string`: Represents a sequence of characters.
- `boolean`: Represents a truth value (true or false).
- `void`: Represents the absence of a value.
- `null`: Represents the intentional absence of a value.
- `undefined`: Represents a value that has not yet been assigned.

TypeScript also supports user-defined datatypes, such as enums, classes, and interfaces.

Benefits of using TypeScript datatypes

There are a number of benefits to using TypeScript datatypes:

- Catch errors early: TypeScript can catch errors early, such as trying to assign a string to a number variable.
- Make code more readable and maintainable: TypeScript datatypes make the code more readable and maintainable, because it is clear what type of data each variable can hold.
- Automatic type inference: TypeScript can automatically infer the type of a variable, based on the value that is assigned to it. This means that you do not always need to explicitly specify the type of a variable.

Example of using TypeScript datatypes

The following example shows how to use TypeScript datatypes to create a simple function that calculates the area of a rectangle:

```
function calculateArea(width: number, height: number): number {
  return width * height;
}
const area = calculateArea(10, 20);
console.log(area); // 200
```

3.3.9.3 Objects in Typescripts

Objects in TypeScript are similar to objects in JavaScript. They are collections of key-value pairs, where the keys can be strings or symbols and the values can be any type of data.

Creating objects in TypeScript

To create an object in TypeScript, you can use the object literal syntax:

```
const myObject = {
  name: "John Doe",
  age: 30,
};
```

This will create an object with two properties: `name` and `age`.

We can also use the `new` keyword to create an object from a constructor function:

```
class Person {
  name: string;
  age: number;

  constructor(name: string, age: number) {
    this.name = name;
    this.age = age;
  }
}
const myPerson = new Person("John Doe", 30);
```

- **Accessing object properties**

To access an object property, you can use the dot notation or the square bracket notation:

```
const name = myObject.name;
const age = myObject["age"];
console.log(name); // "John Doe"
```

```
console.log(age); // 30
```

- **Modifying object properties**

To modify an object property, you can simply assign a new value to it:

```
myObject.name = "Jane Doe";
myObject["age"] = 40;
```

- **Adding and removing object properties**

To add a new property to an object, you can simply assign a new value to it:

```
myObject.occupation = "Software Engineer";
```

- **To remove a property from an object, you can use the `delete` operator:**

```
delete myObject.occupation;
```

- **TypeScript object types**

TypeScript allows you to define the type of an object. This can be useful for catching errors early and making the code more readable and maintainable.

To define the type of an object, you can use an object type literal:

```
type Person = {
  name: string;
  age: number;
};
```

```
const myPerson: Person = {
  name: "John Doe",
  age: 30,
};
```

This will create a variable `myPerson` with the type `Person`. This ensures that the variable can only hold objects with the properties `name` and `age`.

Benefits of using TypeScript object types

There are a number of benefits to using TypeScript object types:

- Catch errors early: TypeScript can catch errors early, such as trying to assign a property to an object that does not exist.
- Make code more readable and maintainable: TypeScript object types make the code more readable and maintainable, because it is clear what type of data each object can hold.
- Automatic type inference: TypeScript can automatically infer the type of an object, based on the properties that are assigned to it. This means that you do not always need to explicitly specify the type of an object.

Example of using TypeScript object types

The following example shows how to use TypeScript object types to create a simple function that validates a user object:

```
type User = {
  name: string;
  email: string;
};
```

```
function validateUser(user: User): boolean {
  return user.name && user.email;
}
const myUser = {
  name: "John Doe",
  email: "john.doe@example.com",
};
const isValid = validateUser(myUser);
console.log(isValid); // true
```

The validateUser() function takes a user object as a parameter and returns a boolean value (true if the user object is valid, false otherwise). The user object must have the properties name and email in order to be considered valid.

3.3.9.4 Typealies

Type aliases in TypeScript allow you to create new names for existing types. This can be useful for making your code more readable and maintainable, or for creating new types that are specific to your application.

To create a type alias in TypeScript, you use the type keyword, followed by the name of the type alias and a colon, followed by the type that the alias is for. For example:

```
type UserId = number;
type Username = string;
const userId: UserId = 12345;
const username: Username = "John Doe";
```

In this example, we have created two type aliases: UserId and Username. These type aliases are for the number and string types, respectively. We can then use these type aliases to declare variables and parameters, which makes our code more readable and maintainable.

Type aliases can also be used to create more complex types. For example, we could create a type alias for a user object:

```
type User = {
  id: UserId;
  name: Username;
};
const user: User = {
  id: 12345,
  name: "John Doe",
};
```

This type alias makes it clear that the user variable can only hold objects with the id and name properties.

Type aliases can also be used to union types. For example, we could create a type alias for a user ID or username:

```
type UserIdentifier = UserId | Username;
const identifier: UserIdentifier = 12345;
identifier = "John Doe";
```

- To create a type alias for a function:

```
type GreetFunction = (name: string) => string;

const greet: GreetFunction = (name) => `Hello, ${name}!`;
```

- To create a type alias for an array:

```
type UserArray = Array<User>;

const users: UserArray = [];
```

- To create a type alias for a generic type:

```
type Dictionary<TKey, TValue> = Record<TKey, TValue>;

const dictionary: Dictionary<string, number> = {};
```

Type aliases are a flexible and powerful tool that can be used to improve the quality of your TypeScript code.

3.3.9.5 Loops

Loops in TypeScript are similar to loops in JavaScript. They allow you to repeat a block of code until a certain condition is met.

There are three main types of loops in TypeScript:

- For loops: For loops are used to iterate over a sequence of values, such as an array or a string.
- While loops: While loops are used to execute a block of code as long as a certain condition is met.
- Do...while loops: Do...while loops are similar to while loops, but the block of code is executed at least once, even if the condition is not met.
- For-each:The forEach loop is a versatile tool that can be used to iterate over any type of data structure that has a `forEach()` method.

For loops

For loops are the most common type of loop in TypeScript. They have the following syntax:

```
for (let i = 0; i < 10; i++) {

  // Code block

}
```

This for loop will iterate from 0 to 9 (exclusive). On each iteration, the value of i will be incremented by 1. The code block will be executed on each iteration.

While loops

```
while (condition) {

  // Code block

}
```

This while loop will execute the code block as long as the condition is met. The condition is evaluated before each iteration of the loop. If the condition is false, the loop will terminate.

Do...while loops

```
do {

  // Code block

} while (condition);
```

This do...while loop will execute the code block at least once. After the code block is executed, the condition is evaluated. If the condition is true, the loop will iterate again.

- **forEach loop**

  in TypeScript is similar to the forEach loop in JavaScript. It allows you to iterate over an array and execute a callback function on each element.

The forEach loop has the following syntax:

```
array.forEach(callback, thisArg);
```

The array parameter is the array that you want to iterate over. The callback parameter is a function that will be executed on each element of the array. The thisArg parameter is an optional parameter that specifies the value of the this keyword inside the callback function.

The callback function takes three parameters:

- element: The current element of the array.
- index: The index of the current element in the array.
- array: The array that is being iterated over.

The following example shows how to use the forEach loop to iterate over an array and print each element to the console:

```
const numbers: number[] = [1, 2, 3, 4, 5];

numbers.forEach(element => {

  console.log(element);

});
```

3.3.9.6 Arrays

Arrays in TypeScript have types. This means that you can specify what type of data an array can contain. This helps to catch errors early and make the code more readable and maintainable.

To specify the type of an array, you use the generic array type syntax:

```
const numbers: Array<number> = [1, 2, 3, 4, 5];
```

This declares an array of numbers. The <number> part of the declaration tells TypeScript that the array can only contain numbers.

We can also use type aliases to create custom array types. For example:

```
type NumberArray = Array<number>;
const numbers: NumberArray = [1, 2, 3, 4, 5];
```

This is equivalent to the previous example, but it uses a type alias to make the code more readable and maintainable.

Benefits of using typed arrays

There are a number of benefits to using typed arrays in TypeScript:

- Catch errors early: TypeScript can catch errors early, such as trying to add a string to a number array.
- Make code more readable and maintainable: Typed arrays make the code more readable and maintainable, because it is clear what type of data each array can contain.
- Automatic type inference: TypeScript can automatically infer the type of an array, based on the values that are assigned to it. This means that you do not always need to explicitly specify the type of an array.

Examples of using typed arrays:

The following example shows how to use typed arrays to create a function that calculates the sum of an array of numbers:

```
function calculateSum(numbers: Array<number>): number {
  let sum = 0;
```

```
  for (const number of numbers) {
    sum += number;
  }
  return sum;
}
const sum = calculateSum([1, 2, 3, 4, 5]);
console.log(sum); // 15
```

3.3.10 React with tsx

React with TSX is a popular combination for building user interfaces. React is a JavaScript library for building user interfaces, and TSX is a TypeScript extension that allows you to write JSX in your TypeScript code. JSX is a syntax extension to JavaScript that allows you to write HTML in your JavaScript code.

Benefits of using React with TSX

There are a number of benefits to using React with TSX:

- Type safety: TypeScript can catch errors early, such as trying to render a component that does not exist.
- Readability and maintainability: TSX makes the code more readable and maintainable, because it is clear what type of data each component is rendering.
- Automatic type inference: TypeScript can automatically infer the type of a component, based on the JSX code that is used to render it. This means that you do not always need to explicitly specify the type of a component.

Examples of using React with TSX

The following example shows how to use React with TSX to create a simple component:

```
import React from "react";


const MyComponent: React.FC = () => {
  return <h1>Hello, world!</h1>;
};


export default MyComponent;
```

3.3.10.1 How to react app

To create a React app with Vite, you can follow these steps:

1. **Install Vite globally:**

```
npm install vite -g
```

2. **Create a new directory for your project and navigate to it:**

```
mkdir my-react-app
cd my-react-app
```

3. **Initialize a new Vite project:**

```
npm create vite my-react-app --template react
```

4. **Install the dependencies:**

```
npm run install
```

5. **Start the development server:**

```
npm run dev
```

3.3.10.2 Components

Components in React TSX are reusable pieces of code that can be combined to create complex user interfaces. They are similar to functions, but they can render HTML to the page.

To create a component in React TSX, you use the function keyword, followed by the name of the component. For example:

```
function Greeting() {
  return <h1>Hello, world!</h1>;
}
```

This will create a component called Greeting that renders a simple heading with the text "Hello, world!".

3.3.10.3 Props

Props in React TSX are data that is passed to a component when it is rendered. They are similar to function parameters, but they are specific to React components.

To pass props to a component, you use the curly braces ({}) syntax. For example:

```
function Greeting({ name }) {
  return <h1>Hello, {name}!</h1>;
}
const App = () => {
  return <Greeting name="John Doe" />;
};
```

In this example, the Greeting component is passed a prop called name with the value "John Doe". The Greeting component will then display the value of the name prop in the heading.

Props can be any type of data, including strings, numbers, objects, and arrays. You can also pass functions as props.

Props are a powerful feature of React that allows you to create reusable and flexible components.

3.3.10.4 How to call Components

There are two ways to call a component in React:

Using the component name: This is the simplest way to call a component. You simply use the name of the component as an HTML element. For example:

// MyComponent.tsx

```
import React from "react";


function MyComponent() {
  return <h1>Hello, world!</h1>;
}


export default MyComponent;
```

- Calling a component

```
// ParentComponent.tsx
import React from "react";
import MyComponent from "./MyComponent";
```

```
function ParentComponent() {
  return (
    <div>
      <MyComponent name="John Doe" />
      <MyComponent name="Jane Doe" />
    </div>
  );
}

export default ParentComponent;
```

This will render two MyComponent components to the page, one for each child of the ParentComponent component.

Calling components in React is a powerful way to build reusable and maintainable UIs. By using the component name or the TSX syntax, you can easily call components from anywhere in your code.

3.3.10.5 Hooks

Hooks in React TSX are functions that allow you to use state and other React features without writing a class. This makes it easier to write reusable and maintainable code.

There are a number of built-in hooks in React, such as useState, useEffect, and useContext. You can also create your own custom hooks.

To use a hook, you simply call it in your function component. For example:

```
import React, { useState } from "react";
```

```
function Greeting() {
  const [name, setName] = useState("John Doe");
  return (
    <div>
      <h1>Hello, {name}!</h1>
      <input
        type="text"
        placeholder="Enter your name"
        value={name}
        onChange={(e) => setName(e.target.value)}
      />
    </div>
  );
}
export default Greeting;
```

In this example, the useState hook is used to create a state variable called name. The name state variable is used to store the user's name. The useState hook also returns a function called setName. This function can be used to update the value of the name state variable.

The Greeting component also has an input field. When the user changes the value of the input field, the onChange event handler is triggered. This event handler calls the setName function to update the value of the name state variable.

Hooks are a powerful feature of React that allow you to write reusable and maintainable code. They are especially useful for writing complex components with state management.

3.3.10.6 Styled Component in React

Styled components is a library that allows you to style React components using CSS-in-JS. This means that you can write your CSS directly in your JavaScript code, which can make your code more concise and readable.

To use styled components, you first need to install the library:

```
npm install styled-components
```

Once the library is installed, you can import it into your code:

```
import styled from "styled-components";

const Button = styled.button`
  background-color: ${(props) => props.color};
  color: ${(props) => props.textColor};
  padding: 10px;
  border-radius: 4px;
`;
```

This will allow you to create different variations of the Button component by passing different props to it.

Styled components is a powerful tool that can help you to write more concise, readable, and maintainable React code.Here is an example of how to use styled components to style a button component:

```
import React from "react";

import styled from
```

```
"styled-components";

const Button = styled.button`

  background-color: blue;

  color: white;

  padding: 10px;

  border-radius: 4px;

`;

function App() {

  return (

    <div>

      <Button>Click me!</Button>

    </div>

  );

}

export default App;
```

This will render a blue button with white text.

3.3.10.7 React Routes and Hash router

React Router is a library for routing in React applications. It allows you to define routes and navigate between them.

There are two types of routes in React Router: browser routes and hash routes.

Browser routes use the browser's history API to keep track of the current route. This means that the URL in the browser's address bar will change as you navigate between routes.

Hash routes use the fragment identifier (#) part of the URL to keep track of the current route. This means that the URL in the browser's address bar will not change as you navigate between routes.

```
function Home() {

  return <h1>Home page</h1>;

}

function About() {

 return <h1>About page</h1>;

}

export default App;
```

Example of a Hash route

```
import React from "react";

import { HashRouter, Route } from "react-router-dom";

function App() {

  return (
```

```
    <HashRouter>
      <Route path="/" exact component={Home} />
      <Route path="/about" component={About} />
    </HashRouter>
  );
}
function Home() {
  return <h1>Home page</h1>;
}
function About() {
  return <h1>About page</h1>;
}
export default App;
```

This code is similar to the previous example, but it uses a HashRouter instead of a BrowserRouter. This means that the URL in the browser's address bar will not change as the user navigates between routes.

3.3.10.8 React Icons

React icons are a set of SVG icons that are designed for use in React applications. They are available as a package on npm and can be installed using the following command:

npm install react-icons

Once the package is installed, you can import the icons into your React application using the following syntax:

import { FaReact } from "react-icons/fa";

This will import the React icon from the Font Awesome icon pack.

To use the icon, you can simply render it like any other React component:

<FaReact />

This will render the React icon to the page.

You can also pass props to the icon to change its appearance. For example, you can pass a size prop to change the size of the icon or a color prop to change the color of the icon.

Here is an example of how to use a React icon to create a button:

```
import React from "react";

import { FaReact } from "react-icons/fa";

const Button = () => {

  return (

    <button>

      <FaReact />

      Click me!

    </button>
```

```
  );

};

export default Button;
```

This code will render a button with the React icon and the text "Click me!".

3.4.1 Git

Git is a distributed version control system that is used to track changes in computer files. It is a popular choice for software development because it is fast, efficient, and reliable.

Git works by storing a complete copy of the repository on each user's machine. This means that everyone working on the project has a complete copy of the code, which makes it easy to collaborate and share changes.

Git also uses a branching system to track changes to the code. This allows developers to work on different features or bug fixes without affecting the main codebase.

To use Git, you first need to install the Git client. Once you have installed the Git client, you can create a new repository or clone an existing repository.

To create a new repository, use the following command:

```
git init
```

This will create a new repository in the current directory.

To clone an existing repository, use the following command:

```
git clone <url of repository>
```

This will clone the repository to your local machine.

Once you have cloned a repository, you can make changes to the code and commit those changes to the repository. To commit changes, use the following command:

git add <file names>

git commit -m "Commit message"

This will add the specified files to the commit staging area and then commit the changes to the repository.

You can also push your changes to a remote repository so that other developers can see them. To push your changes, use the following command:

git push <remote repository name> <branch name>

This will push the changes to the specified remote repository and branch.

Git is a powerful tool for tracking changes in computer files. It is a popular choice for software development because it is fast, efficient, and reliable.

Here are some additional benefits of using Git:

- Collaboration: Git makes it easy for developers to collaborate on projects. Everyone working on the project has a complete copy of the code, which makes it easy to share changes and work on different features or bug fixes.
- Version control: Git allows you to track changes to your code over time. This can be helpful for debugging problems or reverting to a previous version of the code.
- Non-linear workflow: Git supports a non-linear workflow, which means that you can create branches to work on different features or bug fixes without affecting the main codebase.

- Distributed development: Git is a distributed version control system, which means that everyone working on the project has a complete copy of the code. This makes it easy to work on projects from anywhere in the world.

If you are looking for a version control system for your next project, I recommend using Git. It is a powerful and versatile tool that can help you to develop software more efficiently and effectively.

3.5 Github & React Deployment on Github pages

GitHub is a cloud-based hosting service that helps developers store and manage their code, as well as track and control changes to their code. It is a popular choice for software development because it offers a number of features that make it easy to collaborate on projects and share code with others.

To deploy a React application on GitHub, you can follow these steps:

1. Create a new repository on GitHub.
2. Push your React application code to the repository.
3. Create a new deployment environment on GitHub Pages.
4. Connect the deployment environment to your repository.
5. Deploy your React application to the deployment environment.

Once you have deployed your React application to GitHub Pages, it will be accessible at the following URL:

<username>.github.io/<repository-name>

For example, if your username is functional_science and your repository name is my-react-app, then your React application will be accessible at the following URL:

functional_science.github.io/my-react-app

Here is a more detailed explanation of each step:

1. Create a new repository on GitHub:

To create a new repository on GitHub, go to the GitHub website and click on the Create a new repository button.

On the next page, enter a name for your repository and select the Public or Private option.

Click on the Create repository button to create the repository.

2. Push your React application code to the repository:

Once you have created the repository, you can push your React application code to the repository.

To do this, open a terminal window and navigate to the directory where your React application code is located.

Run the following command to push your code to the repository:

```
git push origin main
```

This will push the code in the `main` branch of your local repository to the `main` branch of your remote repository on GitHub.

3. Create a new deployment environment on GitHub Pages:

To create a new deployment environment on GitHub Pages, go to the Settings tab of your repository and click on the Pages link.

On the Pages page, click on the Deploy from branch button.

On the next page, select the `main` branch and click on the Deploy button.

This will create a new deployment environment for your repository.

4. Connect the deployment environment to your repository:

To connect the deployment environment to your repository, click on the Settings link next to the deployment environment name.

On the Settings page, click on the Options tab.

In the Custom domain section, enter the domain name that you want to use for your React application.

Click on the Save button to save your changes.

5. Deploy your React application to the deployment environment:

To deploy your React application to the deployment environment, click on the Deploy button.

GitHub will build your React application and deploy it to the deployment environment.

Once the deployment is complete, your React application will be accessible at the following URL:

<username>.github.io/<repository-name>

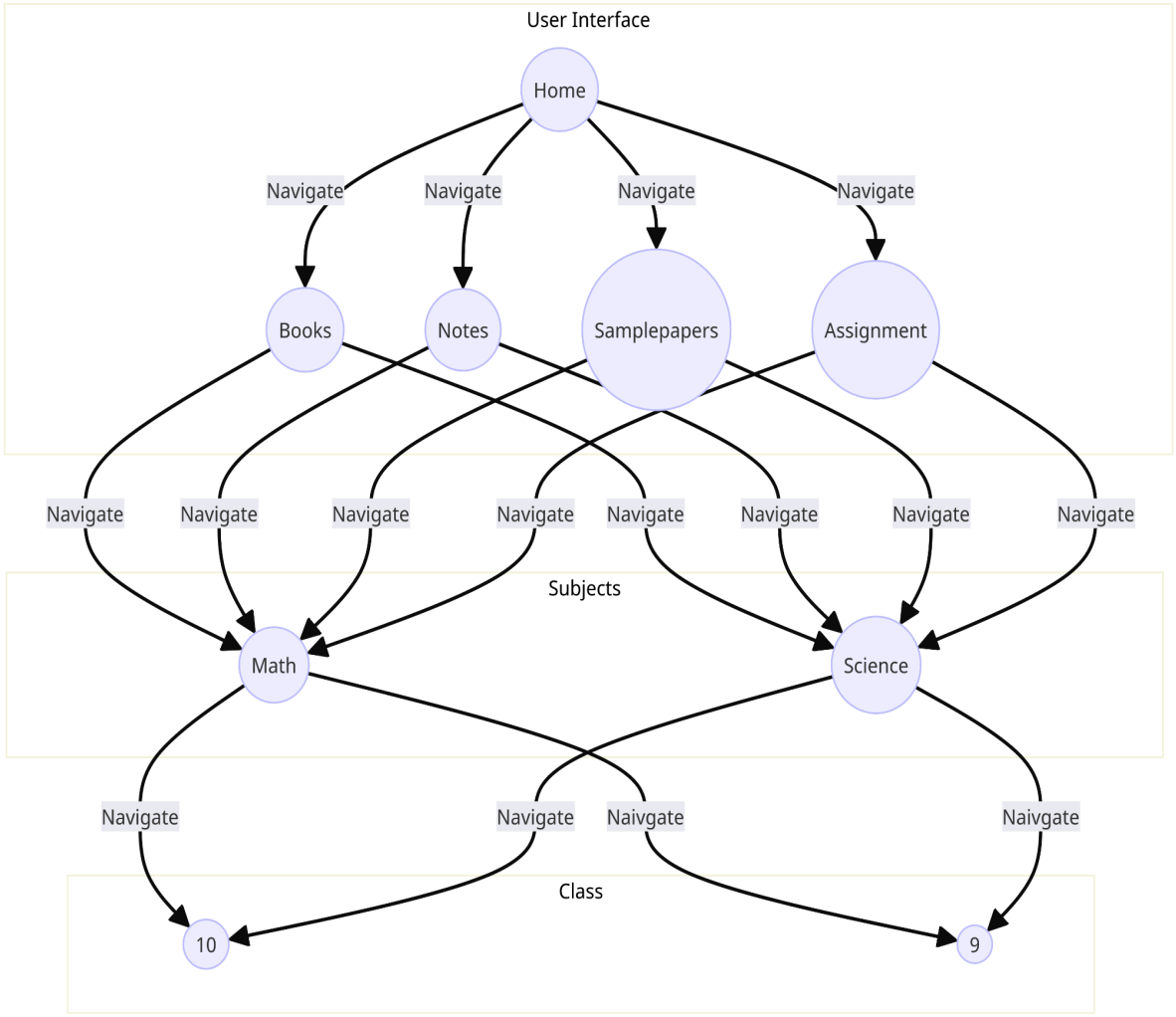
For example, if your username is functional_science and your repository name is my-react-app, then your React application will be accessible at the following URL:

functional_science.github.io/my-react-app

CH - 4

DESIGN PHASE

4.1 DFD of Functional Science



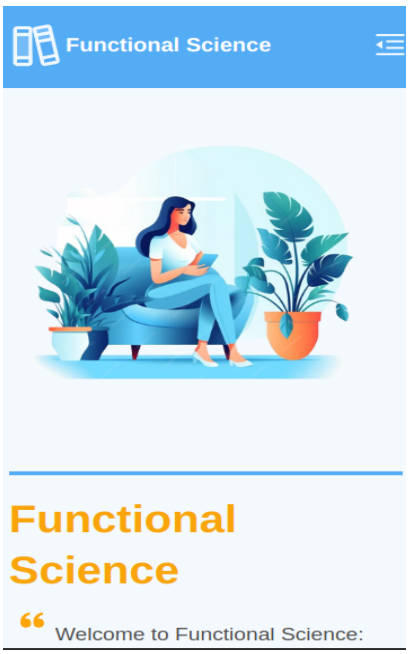
- The "User Interface" subgraph represents the main views of your application (Home, Books, Notes, Samplepapers, Assignment).
- The "Subjects" subgraph represents different subjects (Math and Science) within your application.
- -The arrows indicate the navigation flow from the Home view to Books and other sections, as well as from Books to Math and Science subjects.
- This flowchart provides a clear representation of subjects and their navigation within the main views of your application.

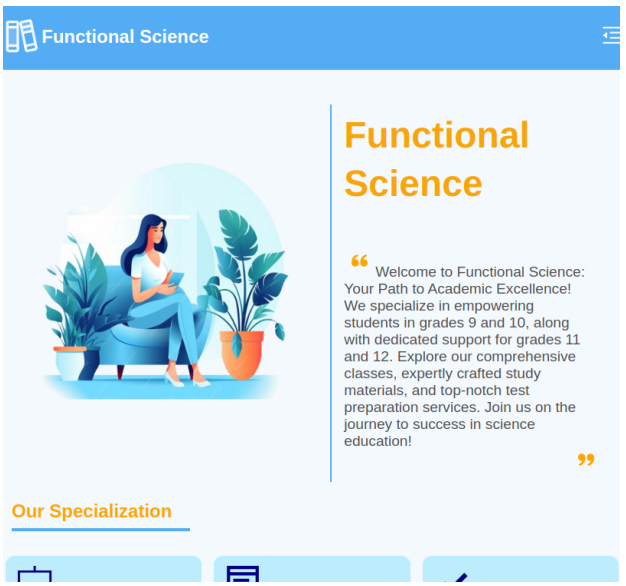
4.2 Responsive Design

Responsive design is an approach to web design that aims to create websites that can be viewed and used on any device, regardless of screen size or orientation. This is important for functional science projects because it allows you to reach a wider audience and improve the user experience for all of your visitors.

By incorporating responsive design principles into the design phase of your functional science project report, you can ensure that your website will look good and function well on all devices.

- Use a responsive layout: A responsive layout allows your report to expand and contract to fit the screen size of the device it is being viewed on.
- Use responsive images: Responsive images can be resized and cropped to fit the screen size of the device they are being viewed on.
- Use media queries: Media queries allow you to target specific devices or screen sizes with your CSS.
- Use a mobile-first approach: When designing your report, start by designing for mobile devices first. This will help you to ensure that your report is easy to use on all devices.
- Test your project on different devices: It is important to test your report on different devices and screen sizes to make sure that it looks good and functions well on all of them.





Functional Science

Home Books Notes Sample Papers Assignments

Functional Science

“Welcome to Functional Science: Your Path to Academic Excellence! We specialize in empowering students in grades 9 and 10, along with dedicated support for grades 11 and 12. Explore our comprehensive classes, expertly crafted study materials, and top-notch test preparation services. Join us on the journey to success in science education!”

Our Specialization

CH - 5
IMPLEMENTATION

In the implementation section, we will see how we will implement the functional science in which I have made the functional website from scratch in which I have used react with typescript and vanilla CSS.

Homepage

The homepage of the project displays an introduction to the project, its features, study material cards, and testimonials. The homepage is implemented using the following React TSX components:

- Introduction: This component displays a brief overview of the project, including its goals and objectives.
- Features: This component displays a list of the project's features.
- Study Material Cards: This component displays a list of study material cards, each of which contains a card to a study material, such as a book, note, or sample paper.
- Testimonials: This component displays a list of testimonials from students who have used the project.

Books Page

The books page allows users to select a book and then download it as a PDF. The books page is implemented using the following React TSX components:

- Book List: This component displays a list of all the books available for download.
- Book Details: This component displays the details of a selected book, such as its title, author, and description.
- Download PDF Button: This component allows users to download the selected book as a PDF.

Notes Page

The notes page allows users to select a note and then download it as a PDF. The notes page is implemented using the following React TSX components:

- Note List: This component displays a list of all the notes available for download.
- Note Details: This component displays the details of a selected note, such as its title, topic, and description.
- Download PDF Button: This component allows users to download the selected note as a PDF.

Sample Papers Page

The sample papers page allows users to select a sample paper and then download it as a PDF. The sample papers page is implemented using the following React TSX components:

- Sample Paper List: This component displays a list of all the sample papers available for download.
- Sample Paper Details: This component displays the details of a selected sample paper, such as its subject, year, and description.
- Download PDF Button: This component allows users to download the selected sample paper as a PDF.

Implementation Details

All of the React TSX components described above are implemented using the following technologies:

- React: A JavaScript library for building user interfaces.
- TypeScript: A superset of JavaScript that adds type safety to the language.
- React icons: A React icons package used for icons on website.

CH - 6
TESTING & PERFORMANCE

6.1 Testing

Here are some test cases that can be used for testing a functional science website:

Homepage

- Verify that the introduction is displayed correctly.
- Verify that the features are displayed correctly.
- Verify that the study material cards are displayed correctly.
- Verify that the testimonials are displayed correctly.
- Verify that the links in the study material cards work correctly.
- Verify that the links in the testimonials work correctly.

Books Page

- Verify that the list of books is displayed correctly.
- Verify that the book details are displayed correctly.
- Verify that the download PDF button works correctly.
- Verify that the downloaded PDF file is valid.

Notes Page

- Verify that the list of notes is displayed correctly.
- Verify that the note details are displayed correctly.
- Verify that the download PDF button works correctly.
- Verify that the downloaded PDF file is valid.

Sample Papers Page

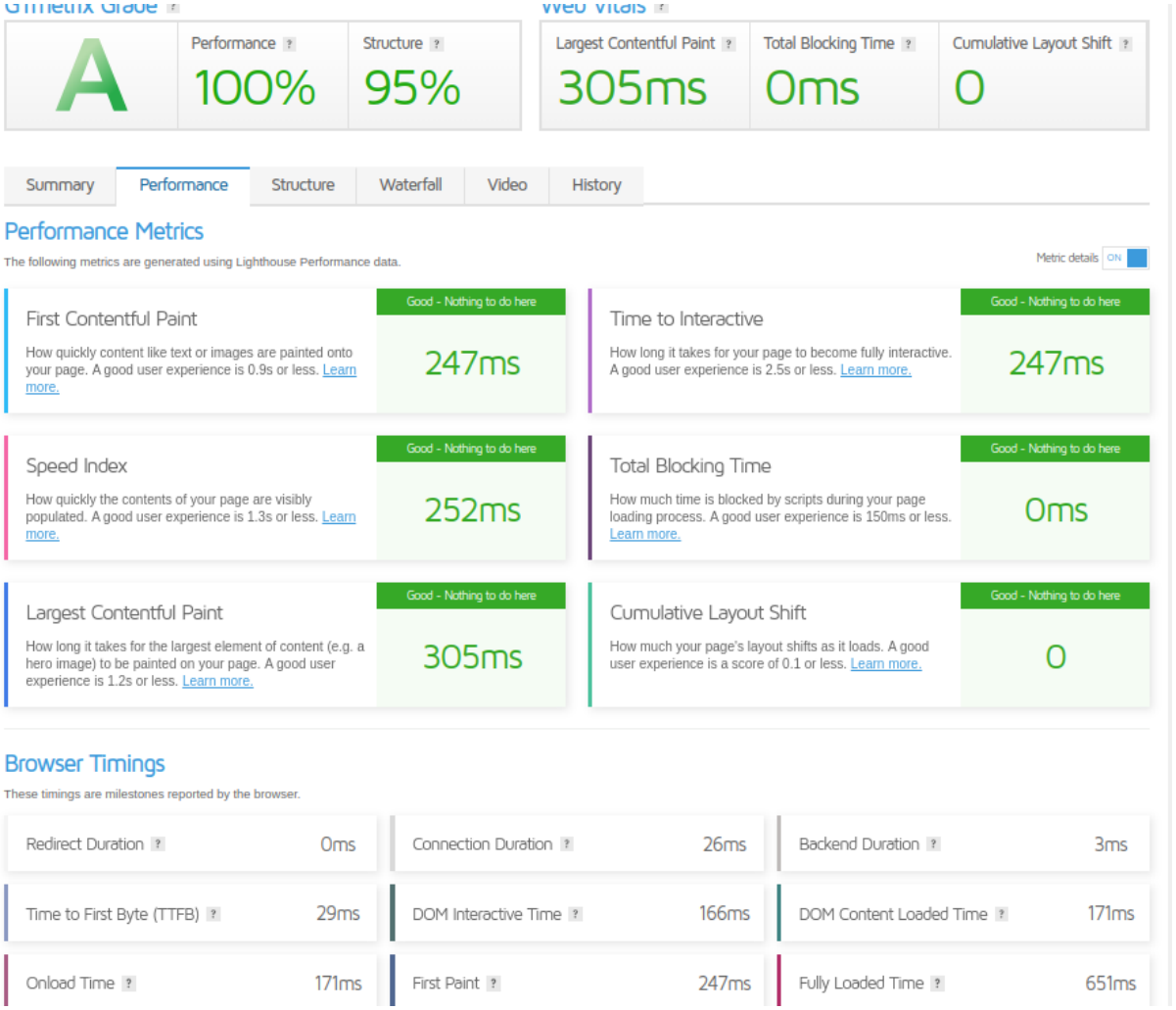
- Verify that the list of sample papers is displayed correctly.
- Verify that the sample paper details are displayed correctly.
- Verify that the download PDF button works correctly.
- Verify that the downloaded PDF file is valid.

Additional Test Cases

- **V**erify that the website is responsive and displays correctly on different devices and screen sizes.
- Verify that the website is accessible to users with disabilities.
- Verify that the website is secure and does not contain any vulnerabilities.

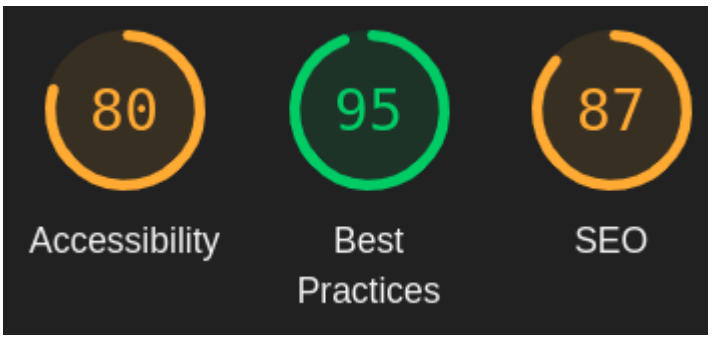
6.2 Performance

6.2.1 Performance:-



This report is from Gmatrix a trusses source for website testing.They test website

1. How much time to load images
2. Image format used
3. Text font and contrast matching
4. Technology Used



This website is also tested by the official google lighthouse performance analyzers

CH - 7

MAINTENANCE

Website Maintenance is the act of regularly checking your website for issues and mistakes and keeping it updated and relevant. This should be done on a consistent basis in order to keep your website healthy, encourage continued traffic growth, deliver a positive user experience, and strengthen your SEO and Google rankings. Given below are some common ways to maintain our website:

- Regularly check that all the web pages are loading without errors.
- Review SEO and Meta titles and descriptions to ensure they are as effective as possible.
- Keep your content up-to-date. This means regularly updating your study materials, practice tests, and other resources to ensure that they reflect the latest curriculum and exam requirements.
- Ensure that your study materials are aligned with the latest curriculum and exam requirements. This means keeping an eye on changes to the CBSE syllabus and NCERT textbooks.
- Focus on providing students with the skills and knowledge they need to succeed in science. Your study materials should go beyond simply teaching students the content; they should also help students to develop their problem-solving, critical thinking, and analytical skills.
- Provide students with opportunities to practice what they have learned. This includes offering practice tests, quizzes, and other interactive activities.
- Offer feedback to students on their work. This will help them to identify their strengths and weaknesses and track their progress over time.
- Build a community around your website. This could involve creating a forum where students can ask questions and share tips, or hosting live Q&A sessions with experts.

CH – 8

CONCLUSION

In this project, I have learned a great deal about React Typescript deep dives and how to increase the UX of a website.

React Typescript deep dives are a great way to learn more about the inner workings of React and Typescript. By going through a deep dive, you can learn about the different components of React, how they work together, and how to use them to build complex applications.

In this project, I learned about the following React Typescript:

- The React component lifecycle: The React component lifecycle is the process that a React component goes through when it is created, rendered, and updated. By understanding the component lifecycle, you can better understand how to use React components to build dynamic and interactive applications.
- React Hooks: React Hooks are a new way to use React state and other features without writing classes. Hooks make it easier to write concise and reusable React components.
- Typescript: Typescript is a superset of JavaScript that adds type safety to the language. By using Typescript, you can catch errors early and write more reliable code.

How to increase the UX of a website

The user experience (UX) of a website is the overall experience that a user has when interacting with a website. A good UX website is easy to use, efficient, and enjoyable.

There are many ways to increase the UX of a website. Here are a few tips:

- Make sure your website is responsive. This means that your website should look good and function well on all devices, including desktop computers, laptops, tablets, and smartphones.
- Use clear and concise navigation menus. Users should be able to easily find the information they are looking for on your website.
- Use high-quality images and videos. Visual content can make your website more visually appealing and engaging.
- Write clear and concise content. Your content should be easy to read and understand.
- Test your website with users. Get feedback from users to identify areas where your website can be improved.

I am grateful for the opportunity to have worked on this project. I have learned a great deal about React Typescript and how to increase the UX of a website. I will use this knowledge to build better and more user-friendly web applications in the future.

CH – 9

FUTURE SCOPE

The future of functional science websites is bright. With the advent of artificial intelligence (AI), there are many new possibilities for how these websites can be used to help students learn and succeed.

One area where AI can be used to improve functional science websites is in doubt solving. AI-powered chatbots can be used to answer student questions 24/7, even if they are complex or challenging. This can be a huge help for students who are struggling to understand a particular concept or who need help preparing for an exam.

AI can also be used to implement courses and online videos on functional science websites. AI-powered video tutorials can be personalised to the individual student's needs and learning style. This can help students to learn at their own pace and in a way that works best for them.

Here are some specific examples of how AI can be used to improve functional science websites:

- AI-powered chatbots can be used to:
    - Answer student questions about science concepts, homework problems, and exam preparation.
    - Provide personalised feedback on student work.
    - Recommend study materials and practice problems to students based on their individual needs.
- AI-powered video tutorials can be used to:
    - Create personalised learning experiences for each student.
    - Provide interactive lessons that allow students to learn by doing.
    - Track student progress and identify areas where students need more help.

In addition to providing flexibility and convenience, recorded videos can also be used to personalize the learning experience for each student. For example, recorded videos can be created to address different learning styles, such as visual learners, auditory learners, and kinesthetic learners. Recorded videos can also be created to address different levels of ability, from beginner to advanced.

Here are some specific examples of how recorded videos can be used to improve the delivery of courses and online videos on functional science websites:

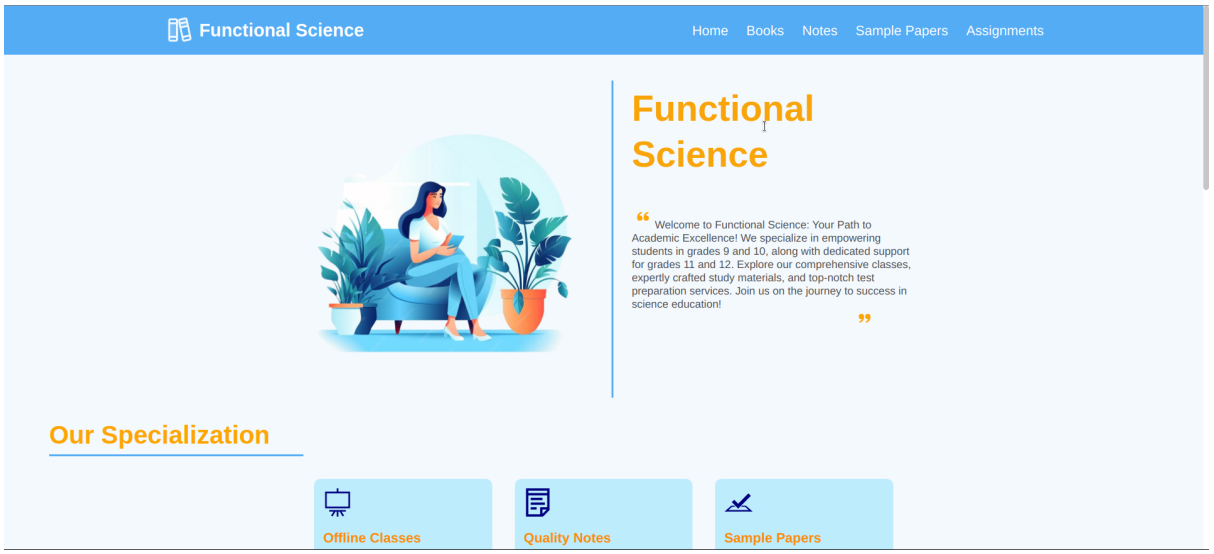
- Recorded videos of lectures: Recorded videos of lectures can be used to provide students with access to high-quality lectures from experienced instructors, regardless of their location or schedule.
- Recorded videos of experiments: Recorded videos of experiments can be used to teach students about complex scientific concepts in a more hands-on way. For example, a recorded video of an experiment on the effects of gravity on different objects could show students how different objects fall at different speeds in a vacuum.
- Recorded videos of dissections: Recorded videos of dissections can be used to teach students about the anatomy of living organisms. For example, a recorded video of a frog dissection could show students the different organs and systems of a frog's body.
- Recorded videos of procedures: Recorded videos of procedures can be used to teach students how to perform complex tasks, such as building a circuit or conducting a chemical analysis. For example, a recorded video of the procedure for conducting a titration could show students how to measure the concentration of an acid or base.

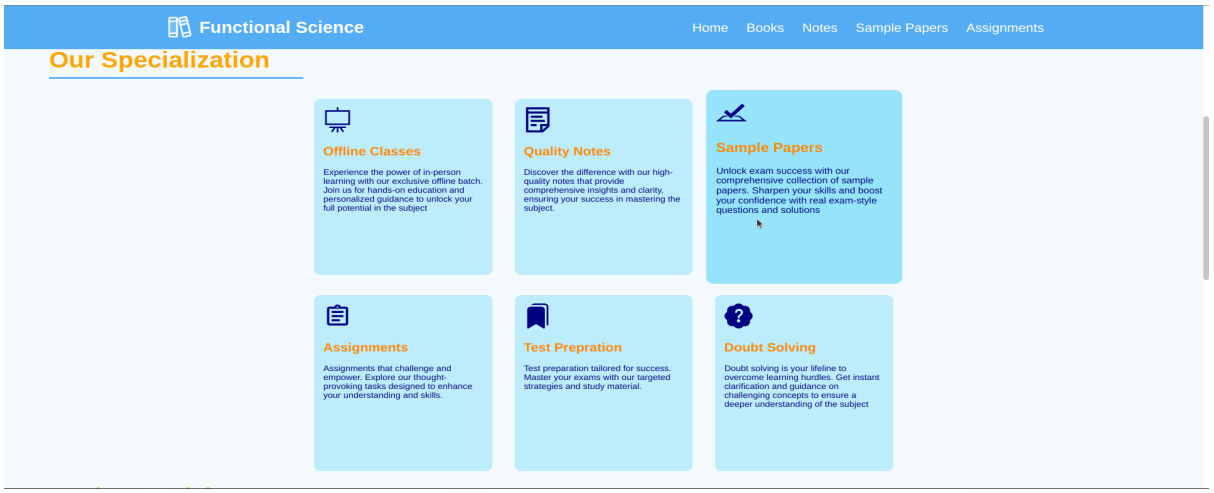
By incorporating recorded videos into the delivery of courses and online videos, functional science websites can become even more effective resources for students of all ages and abilities.

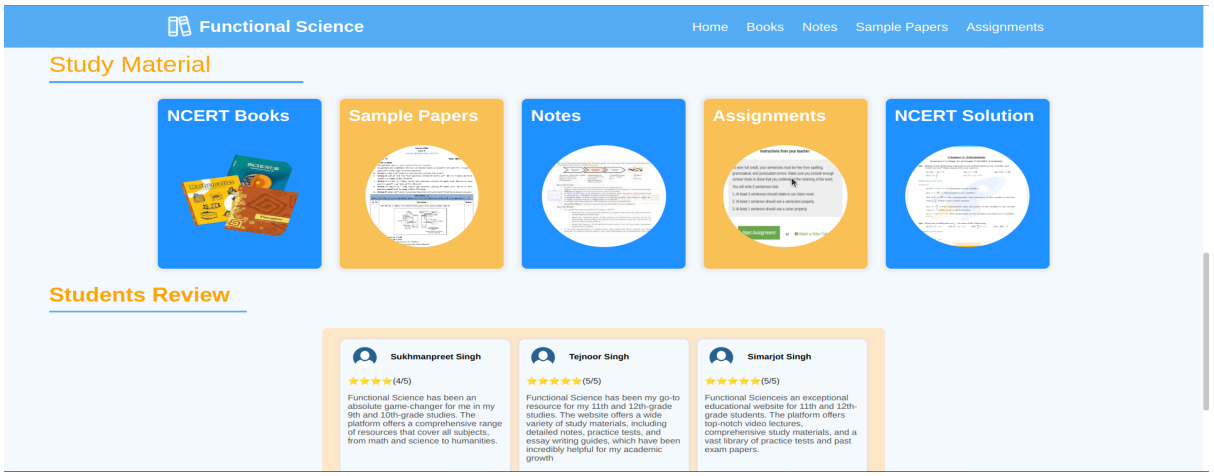
CH – 10

ANNEXURE

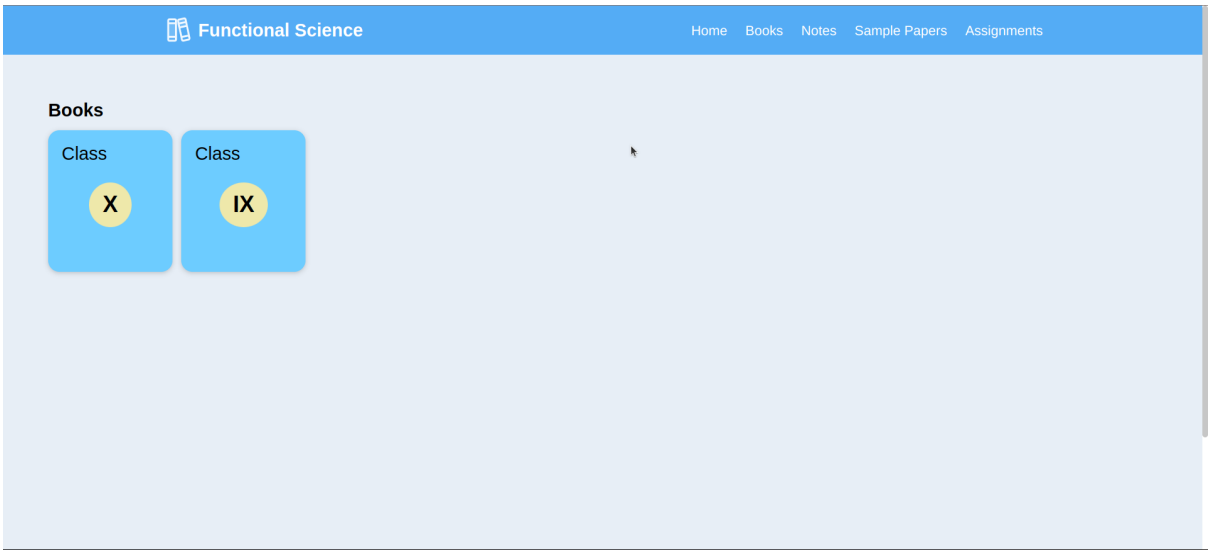
Home page

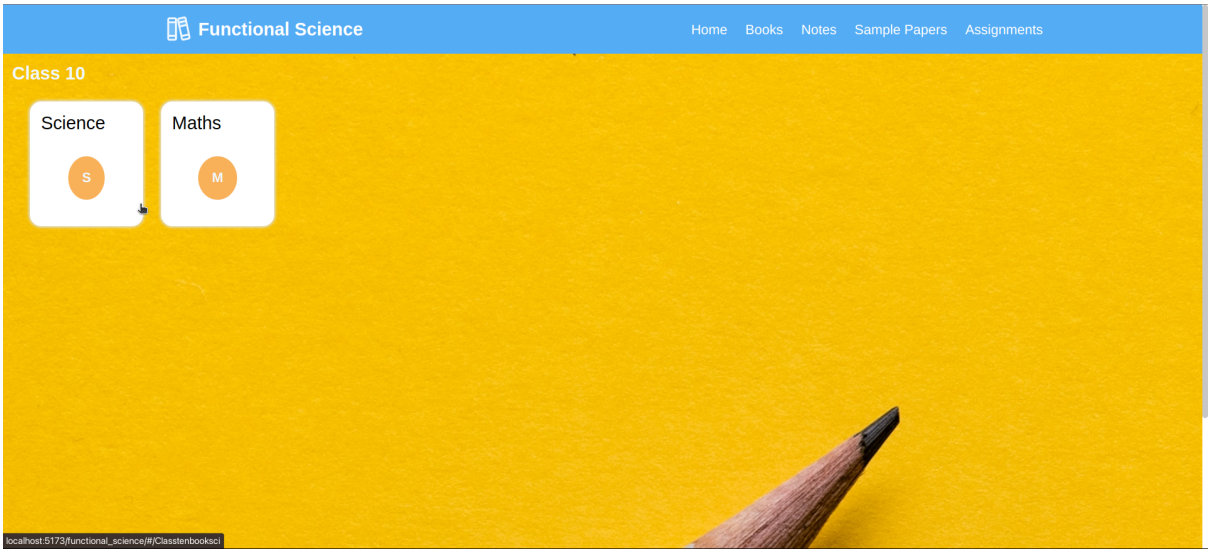


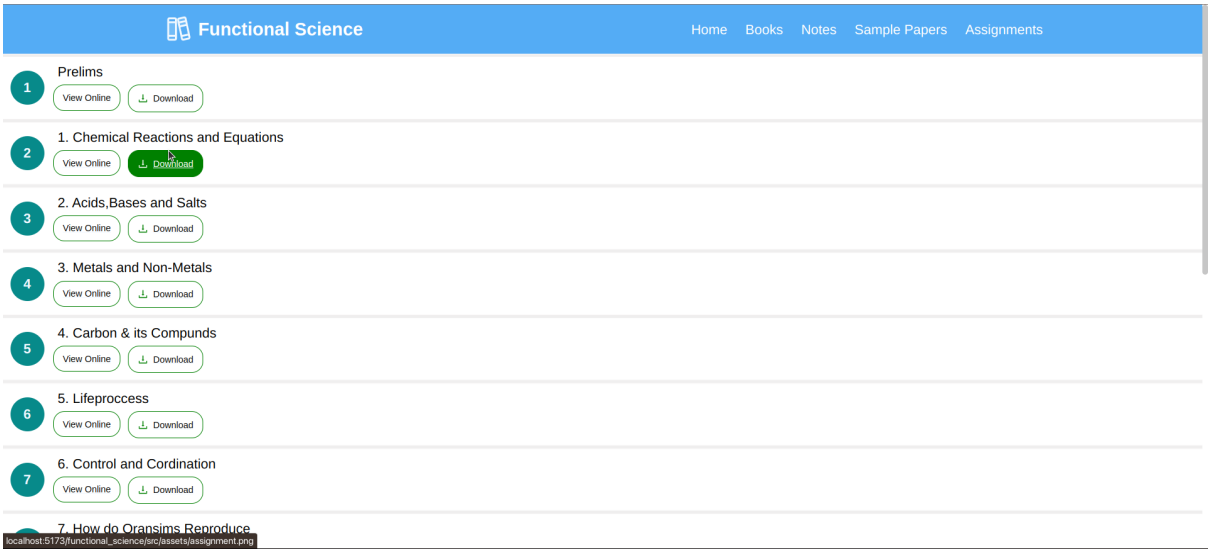




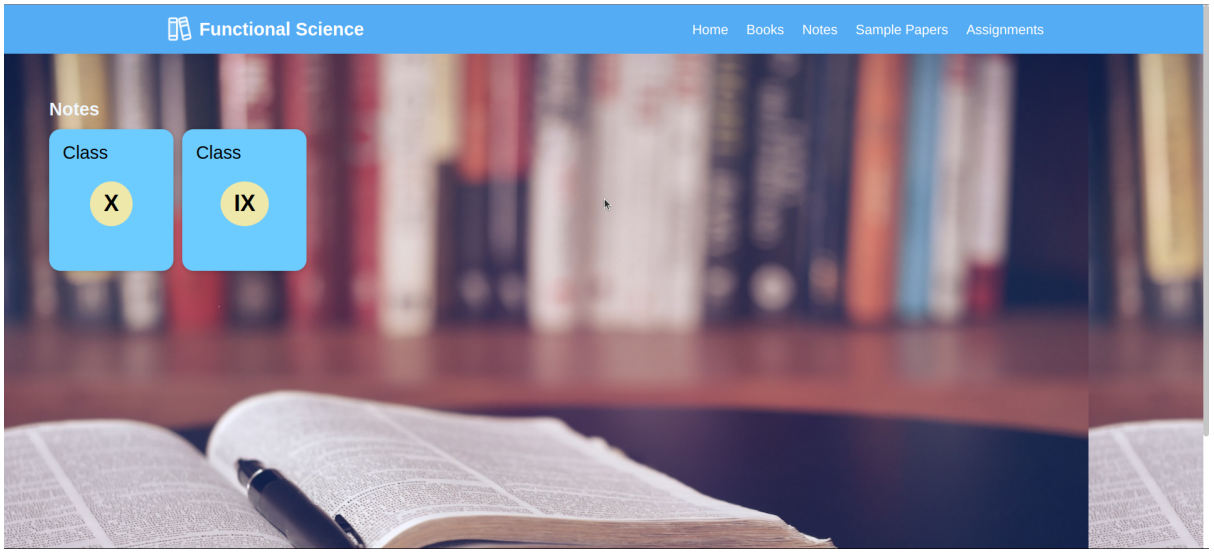
Books Page

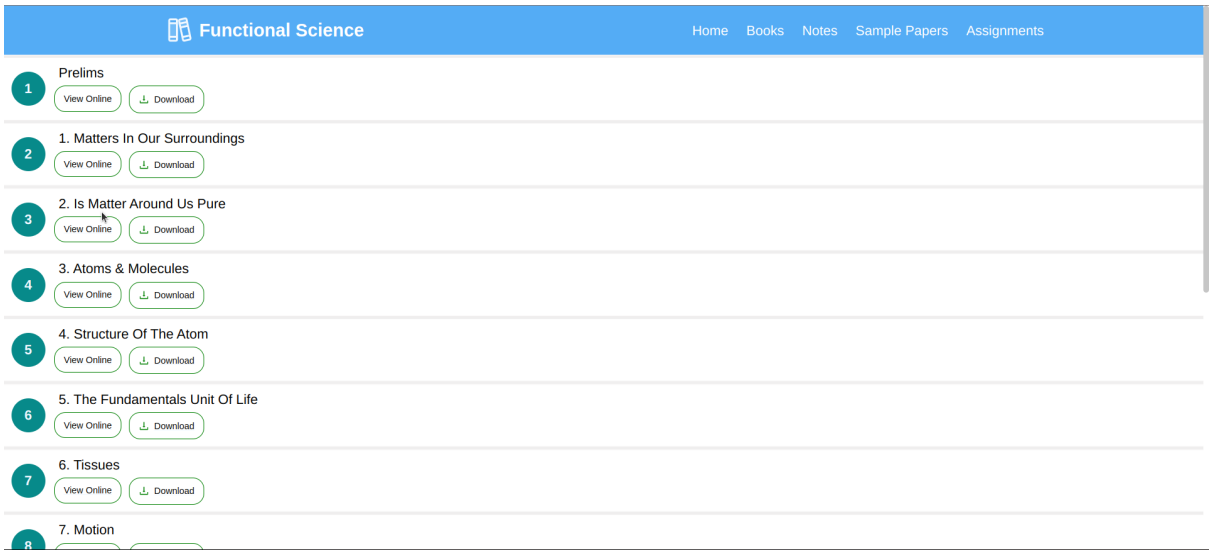

Functional Science
Home
Books
Notes
Sample Papers
Assignments
Books
Class
X
Class
IX

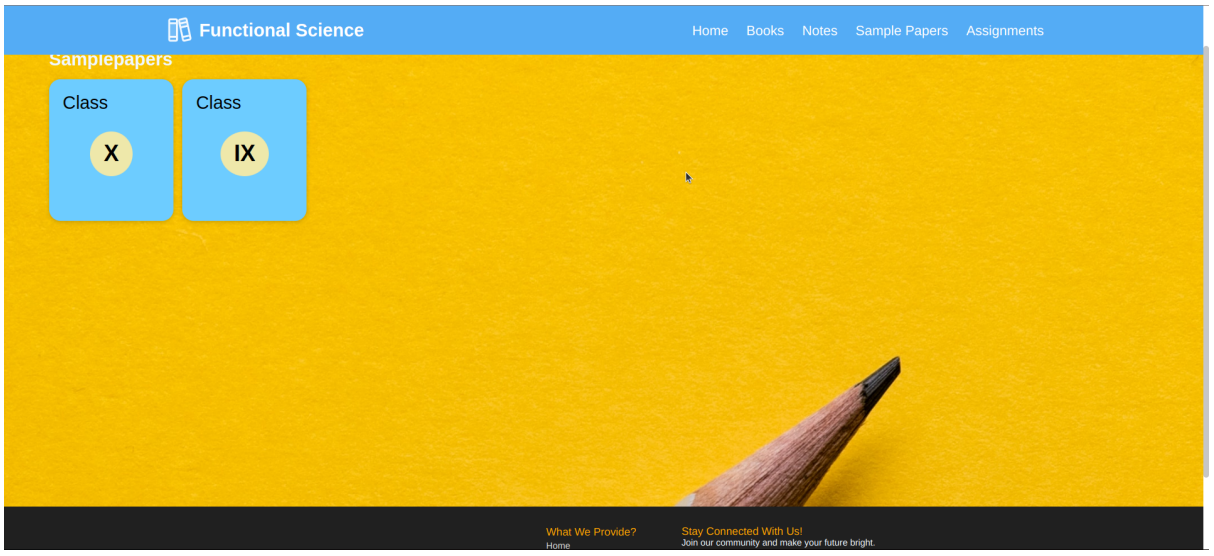

Functional Science
Home
Books
Notes
Sample Papers
Assignments
Class 10
Science
S
Maths
M
localhost:5173/functional_science/#/Classtenbooksci


Functional Science
Home
Books
Notes
Sample Papers
Assignments
1
Prelims
View Online
Download
2
1. Chemical Reactions and Equations
View Online
Download
3
2. Acids,Bases and Salts
View Online
Download
4
3. Metals and Non-Metals
View Online
Download
5
4. Carbon & its Compunds
View Online
Download
6
5. Lifeproccess
View Online
Download
7
6. Control and Cordination
View Online
Download
7. How do Oranisms Reproduce
localhost:5173/functional_science/src/assets/assignment.png

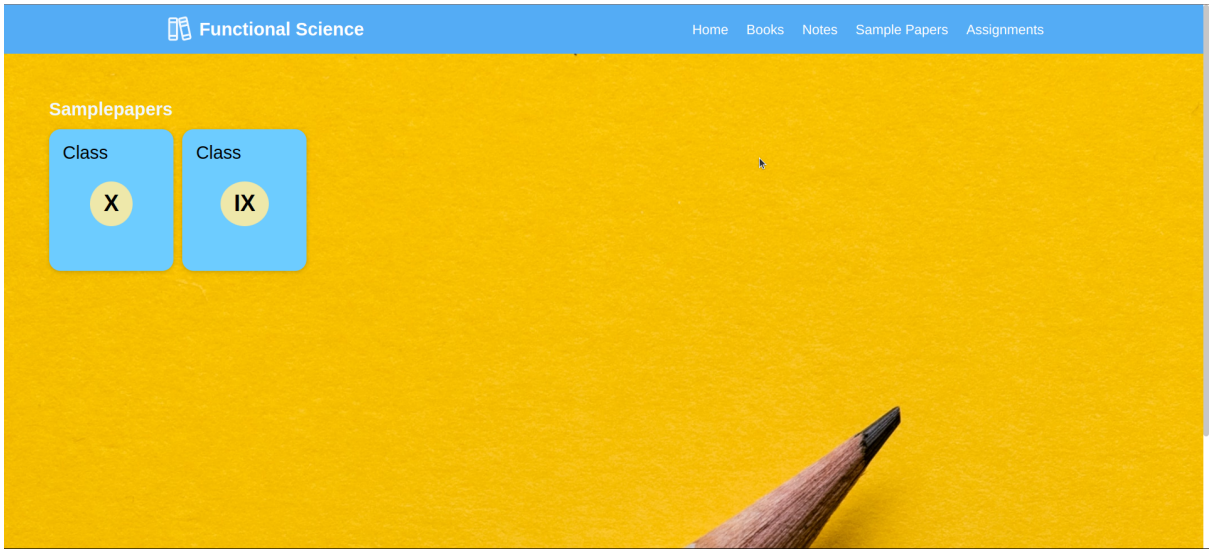
Notes


Functional Science
Home
Books
Notes
Sample Papers
Assignments
Notes
Class
X
Class
IX

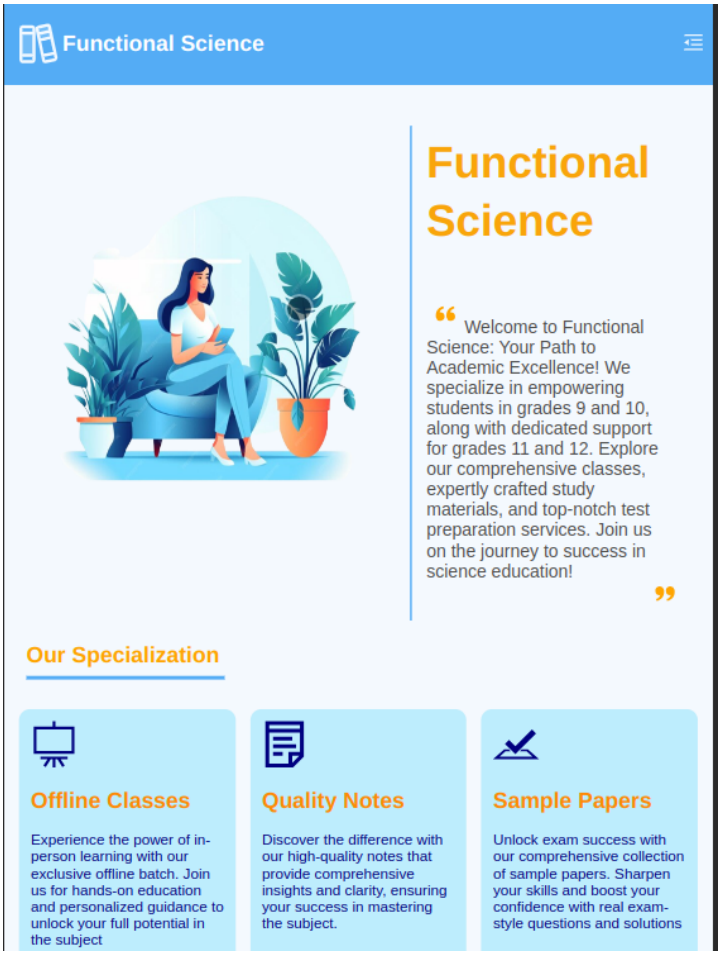

Functional Science
Home
Books
Notes
Sample Papers
Assignments
1
Prelims
View Online
Download
2
1. Matters In Our Surroundings
View Online
Download
3
2. Is Matter Around Us Pure
View Online
Download
4
3. Atoms & Molecules
View Online
Download
5
4. Structure Of The Atom
View Online
Download
6
5. The Fundamentals Unit Of Life
View Online
Download
7
6. Tissues
View Online
Download
7. Motion

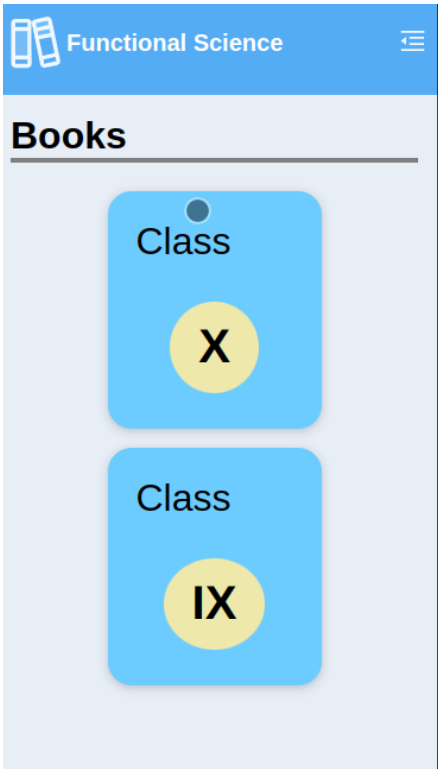

Functional Science
Home
Books
Notes
Sample Papers
Assignments
Samplepapers
Class
X
Class
IX
What We Provide?
Home
Stay Connected With Us!
Join our community and make your future bright.

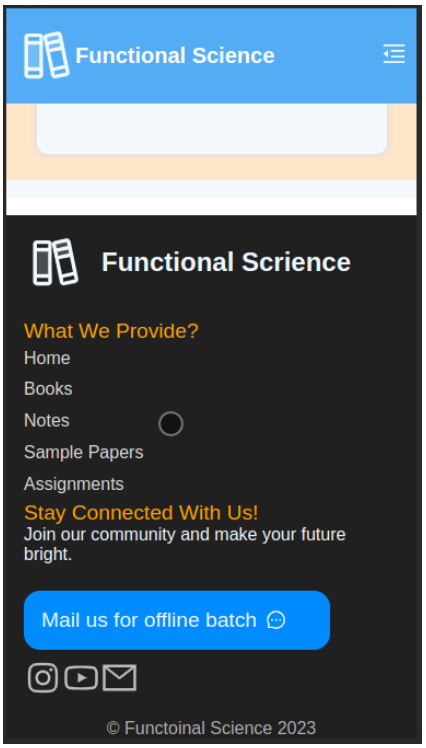
Samplepaper



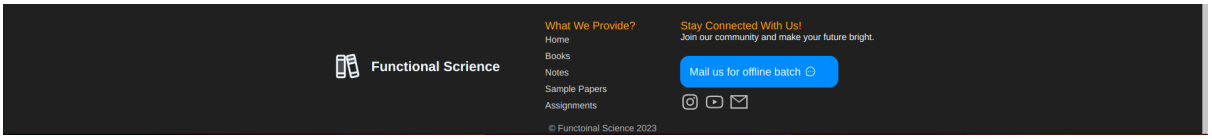
Some Responsive sample pictures

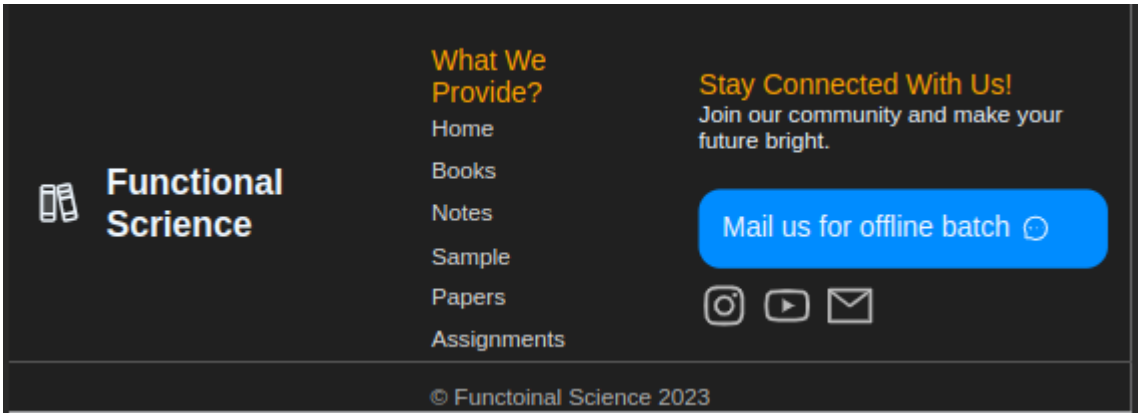






Footer Design





My Project Is Also Deployed on github pages:-

https://simarjot0032.github.io/functional_science/

CH - 11

BIBLIOGRAPHY

https://react.dev/- React Reference

https://www.typescriptlang.org/docs/ -Typescript Code References

https://developer.mozilla.org/en-US/docs/Web/css - CSS Code References

https://www.stackoverflow.com - Coding Error related problem solving from here

https://www.youtube.com - Design Reference from YouTube

https://github.com/ -github pages

https://react-icons.github.io/react-icons/ - React icons