1. Why do we use javascript and its advantages?
2. Difference between ==and === ?
3. Difference Null and undefined ?
4. What is NaN in JavaScript ?
5. Difference between Var let and const ?
6. What is Hoisting in JavaScript ?
7. What is the Lexical Environment ?
8. What is the Temporal Dead Zone ?
9. Different types of loops ? var and let in for loop ?
10. What is This Keyword in JavaScript ? Why do we use it ?
11. What is an API ? how to fetch data from an API ?
12. Difference between SetTimeout and SetInterval ? Is it synchronous or asynchronous ? Is it a part of JavaScript ?
13. What are Event loops ? explain the whole flow
14. What is Call Stack ?
15. What is a micro task queue ? and Difference between Micro and macro stack ?
16. What is Callback hell? What is the inversion of control ? What is a pyramid of Dom ?
17. How to avoid callback hell ?
18. What are Promises ? Drawbacks of Promises ? Methods of promises ? State of Promises ?
19. What is Async awaits ? Is it synchronous or asynchronous ?
20. Can async await and .then and .catch be used together?
21. Difference between Rest operator or Spread Operator ?
22. What is the difference between Call, apply and bind ?
23. What is closure ? Lexical Scope ? What is curring ?
24. What is class and object ? What are Prototypes ?
25. Difference between SessionStorage and local storage ?
26. Difference between map and foreach.
27. How to Handle Error in JavaScript ?
28. .then and .catch and Try ? is .than is synchronous or asynchronous ?
29. **Difference between the promise.all and promise.allsettled.**
30. Reconciliation

React JS

1. Difference between Virtual Dom and Real Dom ? What is Diffing algo ? What is the Batch update in ReactJs?
2. Destructuring of an array
3. Functional component and class components differences ?
4. What is the class Component life cycle
5. What is Rendering ?
6. What are Hooks? Especially useState or useEffect or useReducer ?
7. In the useState() hook , why is it not possible to directly update the state and why are we only able to update the state with setState i,e the updater function?
8. What is Conditional rendering
9. How useEffect() works with empty dependency, no dependency and state inside the dependency?
10. Difference Controlled and Uncontrolled Component
11. What is useRef ?
12. UseEffect hook in depth about the dependent array ? how to store data in local storage using useEffect ?
13. Life cycle methods ? compare life cycle method to reactjs hooks.
14. Why do we use useEffect to fetch an api? Is it possible to fetch an api without useEffect()? If yes then what would be the problems that will be encountered?
15. Difference betw
16. een useMemo and UseCallback?
17. How to use memo and use callback helps in performance optimization.
18. What is the useDispatch() hook?
19. how to pass data from child to parent component ?
20. How to share data between sibling components ?
21. What is the useContext hook? Why we use it
22. What is prop drilling ?
23. What is context API?
24. Why do we use context API instead of prop drilling?
25. What is Redux ? flow of redux ? What is redux thunk and redux saga ? What is a reducer ?
26. What is the difference between Context vs Redux?

27. What is Router? Why do we use the Browser route? create a navigation bar and define its path?
28. Types of components ? What are higher order components ?
29. What are Custom components in ReactJs and why do we use it ?
30. What is Importing and Exporting?
31. What is suspense, call back, useMemo, re-forward?
32. Why do we use keys in React list items?
33. What is createSlice
34. How to create a store in redux
35. Flux architecture,
36. What is phase and methods in react
37. Can we replace redux using react hooks?
38. Task is create an array of objects with id,name,price and use redux show product name and price and a button to add to cart using redux
39. Create a react application for showing answers when clicked on question

DSA

1. Identify and eliminate duplicate words from the given string, then display the modified string. "big black bug bit a big black dog on his big black nose"

   The objective is to find and remove any repeated words within the string, resulting in a modified string that displays each word only once.

2. Console.log("2"+2)
3. **Swap KEY : Value**
```
let obj = {
        a : 1
        b : 2
        }
```
4. How will you know if a number given falls under fibonacci series or not?
5. Write a JavaScript program that accepts a number as input and inserts dashes (-) between each even number. For example if you accept 025468 the output should be 0-254-6-8.
6. Using the `reduce` method, calculate and print the sum of elements in an array.
7. Create a timer whose by default value is 10 and value should decrease by 1 per second

Questions 1
arr1 = [{age: 8, name: 'Hman', id: 1}, {age: 9, name: 'Iman', id: 2}, {age: 9.5, name: 'Kman', id: 3}, {age: 10, name: 'Jman', id: 4}]

arr2 = [{id: 1, edu: 'BCom'}, {id: 1, edu: 'Diploma'}, {id: 2, edu: 'BSc'}, {id: 2, edu: 'BA'}, {id: 3, edu: 'MSc'}, {id: 4, edu: 'BTech'}]

Give me combined education data along with id and name
Output: [{name: 'Hman', id: 1, qualification: [‘BCom’, 'Diploma']}]

Q2. **let listData** = [
```
{ id: 1, name: 'Discover Music' },
   { id: 2, pid: 1, name: 'Hot Singles' },
   { id: 3, pid: 1, name: 'Rising Artists' },
   { id: 4, pid: 1, name: 'Live Music' },
   { id: 6, pid: 1, name: 'Best of 2017 So Far' },
{ id: 7, name: 'Sales and Events' },
   { id: 8, pid: 7, name: '100 Albums' },
   { id: 9, pid: 7, name: 'Hip-Hop and R&B Sale' },
   { id: 10, pid: 7, name: 'CD Deals' },
{ id: 11, name: 'Categories' },
   { id: 12, pid: 11, name: 'Songs' },
   { id: 13, pid: 11, name: 'Best Selling Albums' },
   { id: 14, pid: 11, name: 'New Releases' },
   { id: 15, pid: 11, name: 'Best Selling Songs' },
{ id: 16, name: 'MP3 Albums'},
   { id: 17, pid: 16, name: 'Rock' },
   { id: 18, pid: 16, name: 'Gospel' },
   { id: 19, pid: 16, name: 'Latin Music' },
   { id: 20, pid: 16, name: 'Jazz' },
{ id: 21, name: 'More in Music' },
   { id: 22, pid: 21, name: 'Music' },
   { id: 22, pid: 21, name: 'Music Trade-In' },
   { id: 23, pid: 21, name: 'Redeem a Gift Card' },
   { id: 24, pid: 21, name: 'Band T-Shirts' },
]
```

o/p format- { id: 1, name: 'Discover Music', childs:[
```
{ id: 2, pid: 1, name: 'Hot Singles' },
{ id: 3, pid: 1, name: 'Rising Artists' },
 { id: 4, pid: 1, name: 'Live Music' },
 { id: 6, pid: 1, name: 'Best of 2017 So Far' },
] }
```
Question 3. const arrNum = [
```
  [1, 2, 3],
  [4, 5, 6],
  [7, 8, 9],];
// [ [ 6 ], [ 120 ], [ 504 ] ] (Replace with factorial array)
```
4. Flat an array = [1,2,3,[5,[6,7],8],9] , **Output Array =** [1,2,3,4,5,6,7,8,9]

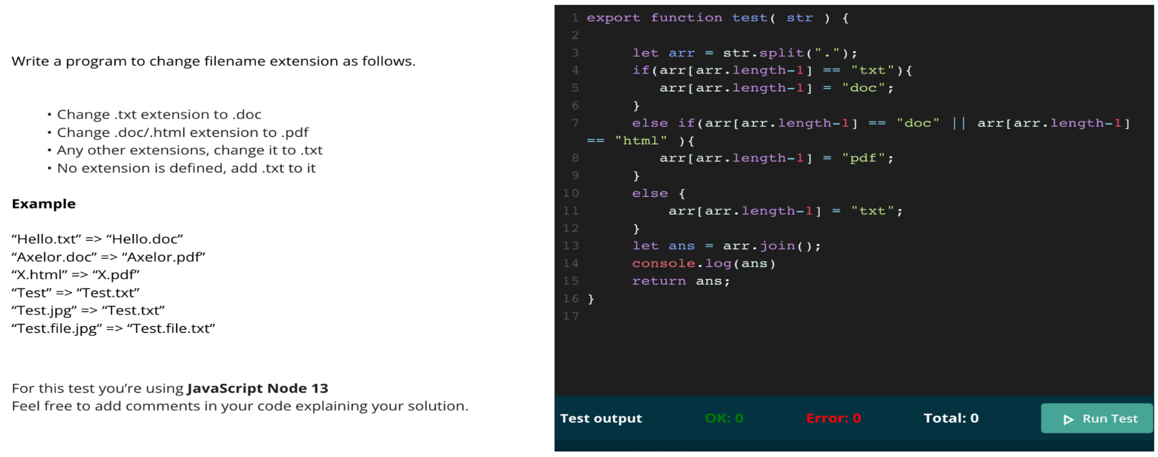
5. const inventory = [
```
  { name: "asparagus", type: "vegetables", quantity: 5 },
  { name: "bananas", type: "fruit", quantity: 0 },
  { name: "goat", type: "meat", quantity: 23 },
  { name: "cherries", type: "fruit", quantity: 5 },
  { name: "fish", type: "meat", quantity: 22 },
];
```

Output -

```
{
vegetables: [
{ name: 'asparagus', type: 'vegetables', quantity: 5 },
 ],
fruit: [
{ name: "bananas", type: "fruit", quantity: 0 },
{ name: "cherries", type: "fruit", quantity: 5 }
],
meat: [
{ name: "goat", type: "meat", quantity: 23 },
{ name: "fish", type: "meat", quantity: 22 }
] }
```

Q6. Take a input field and button user can enter number in between 1 to 100
Based on the user value it will display a timer

8. Code to give sum of diagonal
```
[[10,1,20,40,20],
 [10,1,21,40,20],
 [10,1,40,40,20],
 [10,1,30,40,20],
 [10,1,60,40,20] ]
```
9. How to convert 24 hours format to 12 hours in Javascript. [Link].
10. Take an input field and the button user can enter numbers in between 1 to 100. Based on the user value it will display a timer



Write a program to change filename extension as follows.

- Change .txt extension to .doc
- Change .doc/.html extension to .pdf
- Any other extensions, change it to .txt
- No extension is defined, add .txt to it

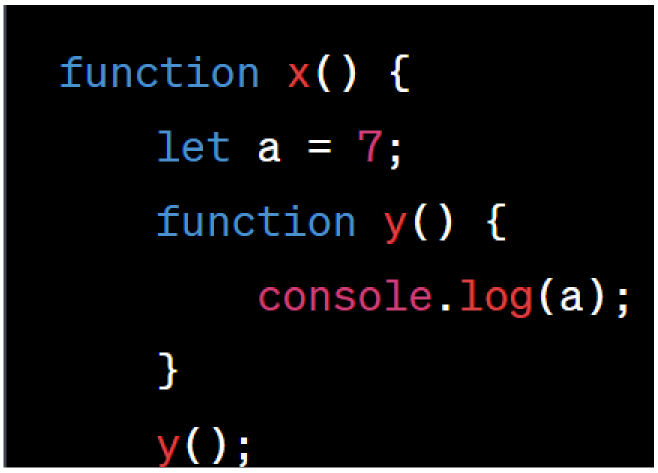
Example

"Hello.txt" => "Hello.doc"
"Axelor.doc" => "Axelor.pdf"
"X.html" => "X.pdf"
"Test" => "Test.txt"
"Test.jpg" => "Test.txt"
"Test.file.jpg" => "Test.file.txt"

For this test you're using **JavaScript Node 13**
Feel free to add comments in your code explaining your solution.

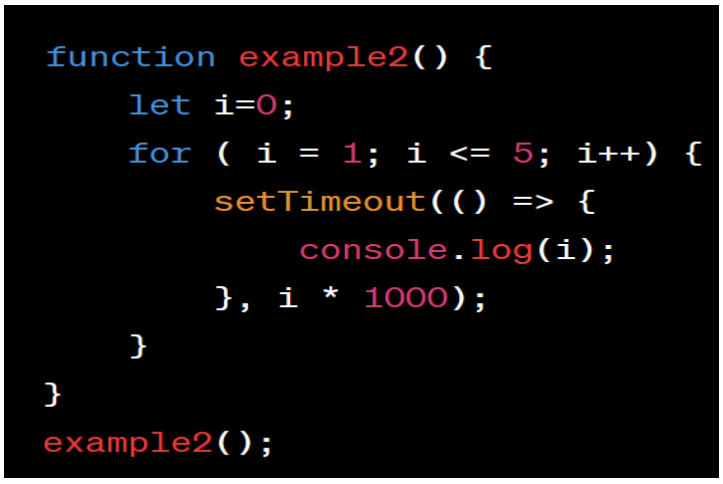
```
export function test( str ) {

      let arr = str.split(".");
      if(arr[arr.length-1] == "txt"){
         arr[arr.length-1] = "doc";
      }
      else if(arr[arr.length-1] == "doc" || arr[arr.length-1]
== "html" ){
         arr[arr.length-1] = "pdf";
      }
      else {
          arr[arr.length-1] = "txt";
      }
      let ans = arr.join();
      console.log(ans)
      return ans;
}
```

1.Tell me the output and how closure is helping



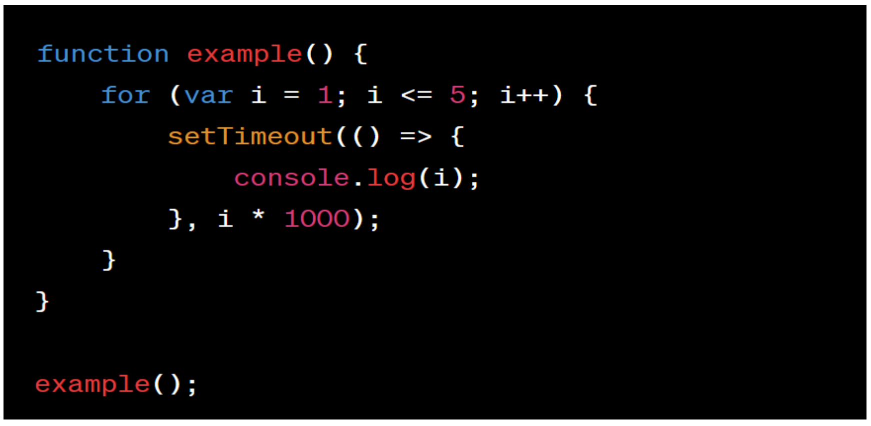
```
function x() {
    let a = 7;
    function y() {
        console.log(a);
    }
    y();
```

2. Tell me the output:



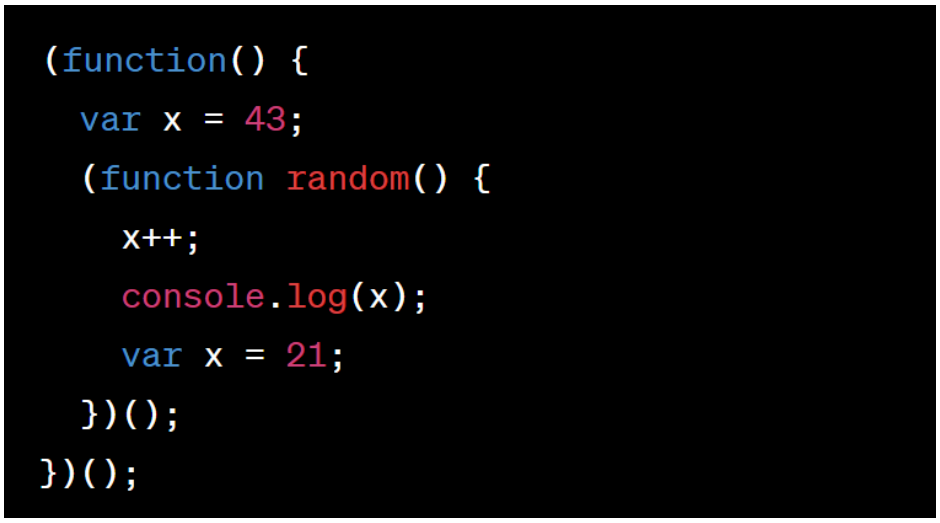
```
function example2() {
    let i=0;
    for ( i = 1; i <= 5; i++) {
        setTimeout(() => {
            console.log(i);
        }, i * 1000);
    }
}
example2();
```

3. What will be the output of the given JavaScript code? How can you modify the code to achieve the desired output using the `let` keyword?



```
function example() {
    for (var i = 1; i <= 5; i++) {
        setTimeout(() => {
            console.log(i);
        }, i * 1000);
    }
}

example();
```

4. What is the output of below code?

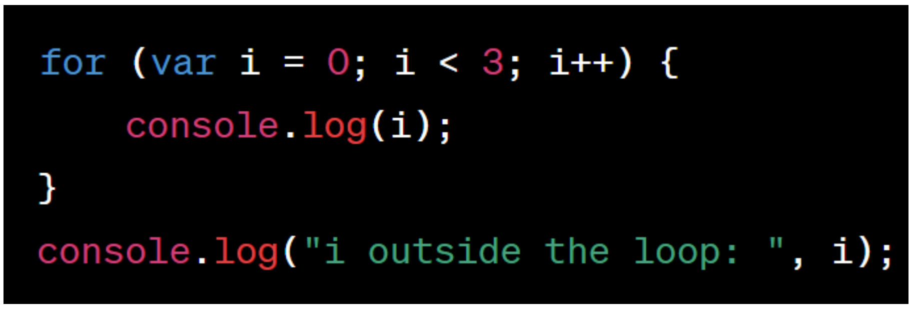


```
(function() {
  var x = 43;
  (function random() {
    x++;
    console.log(x);
    var x = 21;
  })();
})();
```

5. What is the output of below code?

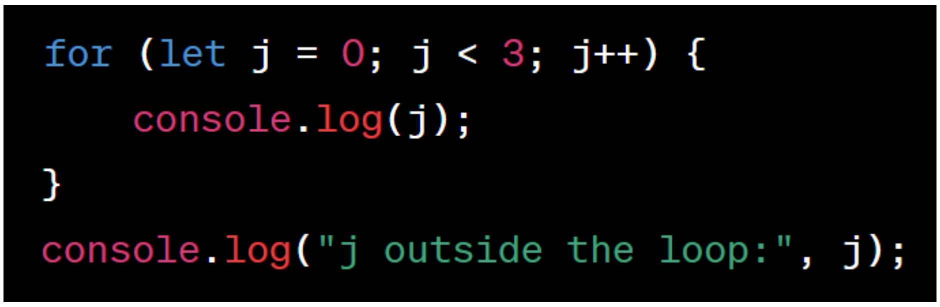
- console.log(0.1 + 0.2 = 0.3);
- console.log(0.1 + 0.2 == 0.3);
- console.log(0.1 + 0.2 === 0.3);

6. Give output of following code .



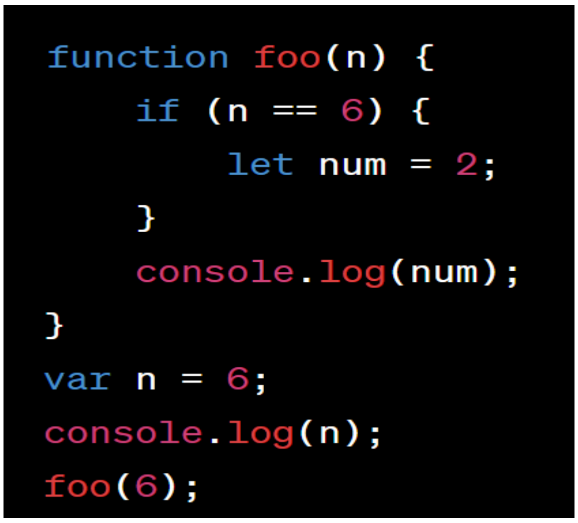
```
for (var i = 0; i < 3; i++) {
    console.log(i);
}
console.log("i outside the loop: ", i);
```

7. Give output of following code .



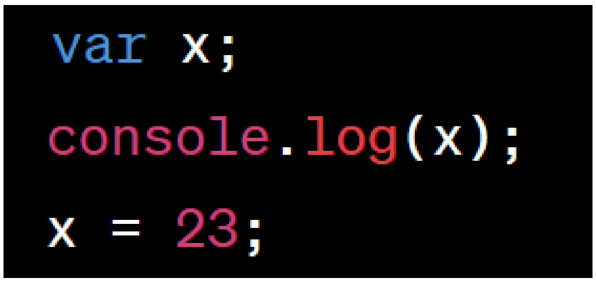
```
for (let j = 0; j < 3; j++) {
    console.log(j);
}
console.log("j outside the loop:", j);
```

8. Give output of following code .



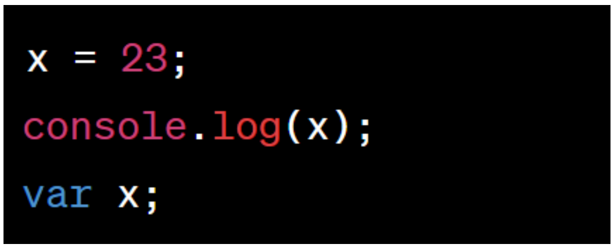
```
function foo(n) {
    if (n == 6) {
        let num = 2;
    }
    console.log(num);
}
var n = 6;
console.log(n);
foo(6);
```

9. Give output of following code .



```
var x;
console.log(x);
x = 23;
```

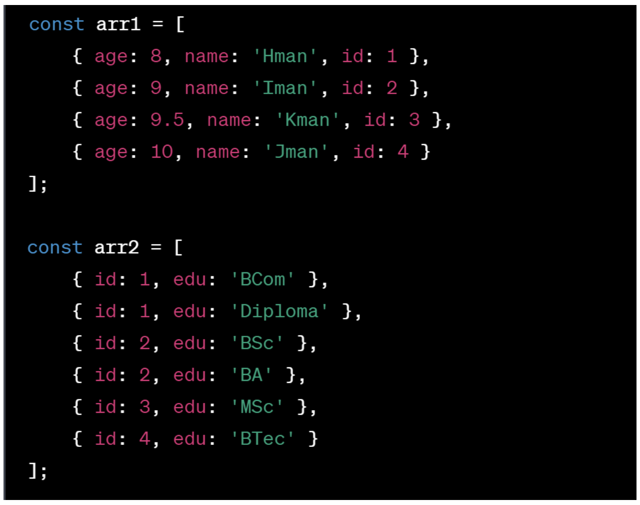
10. Give output of following code .



```
x = 23;
console.log(x);
var x;
```

11. Give me combined education data along with id and name

Output: [{name: 'Hman', id: 1, qualification: [‘BCom’, 'Diploma']}]



```
const arr1 = [
    { age: 8, name: 'Hman', id: 1 },
    { age: 9, name: 'Iman', id: 2 },
    { age: 9.5, name: 'Kman', id: 3 },
    { age: 10, name: 'Jman', id: 4 }
];

const arr2 = [
    { id: 1, edu: 'BCom' },
    { id: 1, edu: 'Diploma' },
    { id: 2, edu: 'BSc' },
    { id: 2, edu: 'BA' },
    { id: 3, edu: 'MSc' },
    { id: 4, edu: 'BTec' }
];
```

React Implementation Task

1. Ultimez - ReactJs. Developer
   Task 1:- https://live-interview-tasks.vercel.app/tasks/four
   Task 2:- https://live-interview-tasks.vercel.app/tasks/three
2. Create a responsive card layout using a combination of grid, Bootstrap, and Flexbox. Organise the layout into rows, each containing three cards. Additionally, implement an interactive feature where clicking on a card changes its colour, and clicking it again restores the original colour - essentially toggling the colour on each click.
3. Explain how to pass data from a child component to a parent component using both Redux and Context API in a React application.
4. In a React application, pass a setter function from a child component to a parent component. Create a button within the parent component that, when clicked, triggers the setter function to modify a state value.
5. Create a React component that fetches data from an API called "abc" and saves the response in a state variable. Utilise an asynchronous function and include the useEffect hook within your component to trigger the data fetching upon the component's mount. Here is the initial code structure for the component:

```
import React, { useState, useEffect } from 'react';

function Component() {
    const [state, setState] = useState([]);

    async function apiCall() {
        const api = await fetch("https://your-api-url-here.com");
        const response = await api.json();
        setState(response);
    }

    useEffect(() => {
        apiCall();
    }, []);

    return (
        <>
            {/* Your JSX content */}
        </>
    )
}
```

To retrieve data from the URL "https://jsonplaceholder.typicode.com/todos?_start=0&_limit=10":

1. Filter the array of objects based on the unique title.(remove the objects with duplicate titles).
2. Incorporate an input checkbox alongside the fetched data by generating a checkbox element through HTML and populating it with the fetched data.
3. Utilise React to analyse the URL, extract the data, and consequently manage the checkbox's state for automatic checking or unchecking, aligned with the data sourced from the URL.
4. Implement a mechanism with React to enable toggling a specific checkbox while keeping the other checkboxes unaffected. This entails keeping track of the state of each checkbox and updating the URL correspondingly.
5. Integrate Next and Prev buttons into the interface, facilitating navigation through the obtained data. These buttons can be generated using React components and programmed to manipulate both the URL and checkbox states upon user interaction.
6. To retain prior data when the Next button is triggered, employ React to store and manage cached data for seamless navigation.
7. Enhance efficiency by checking the cache before fetching data via the URL. Leverage React to determine if the required data already resides in the cache, and if so, retrieve it from there instead of making redundant requests.

In addition, an aspect of the implementation is to introduce a Search feature based on the Title attribute. This would involve designing a search input field, integrating it with the React application, and programming it to filter and display data entries that match the search query in their titles.