**Technical Solution Approach**

Contents

[1 Introduction 2](#_gjdgxs)

[1.1 About this document 2](#_30j0zll)

[1.1.1 Purpose & Scope of the document 2](#_1fob9te)

[2 Component Design 2](#_3znysh7)

[2.1 Component Design Diagram 2](#_2et92p0)

[2.1.1 Overall Workflow 3](#_tyjcwt)

[2.1.2 Low level Design 4](#_3dy6vkm)

[3 Technology & Frameworks to be used 4](#_1t3h5sf)

[4 Solution Approach 5](#_4d34og8)

# Introduction

## About this document

The document outlines the purpose and scope of the AdDisplay component, which is responsible for displaying ads on a webpage.

The document also includes a component design diagram and an overall workflow sequence diagram. The AdDisplay component will be built using React JS and the document provides a solution approach that involves creating an AdDisplay component and an AdLoader component to asynchronously load ad data.

### Purpose & Scope of the document

The AdDisplay component is responsible for displaying advertisements to users. The component will be displayed in a designated space on the webpage and will show a variety of ads at same time. The design of the banner will be simple with a clear call-to-action and minimal text. The banner will use a white background with bold, contrasting colors for the text and imagery to catch the user's attention.

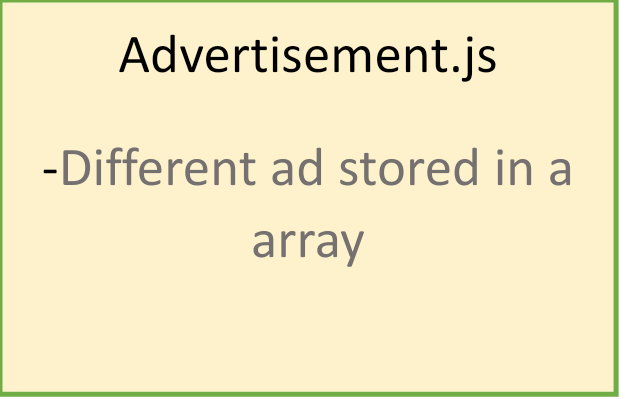
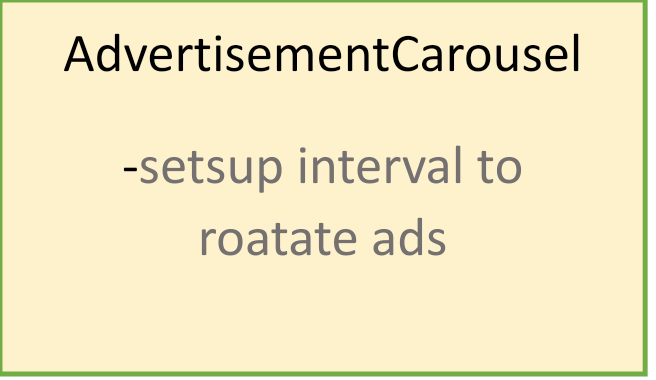
The AdDisplay component will be used to display ads to users as they browse the webpage. The component will be located in a designated space on the webpage and will not interfere with the user's browsing experience.

# Component Design

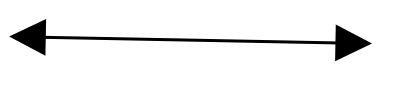
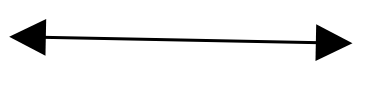
## Component Design Diagram

. React component calls AdRender with advertisement and delay as parameter to check the type.The matched condition calls carousel,the `AdvertisementCarousel` component sets up an interval to rotate through the given `advertisements` array, displaying a different advertisement image based on the current index. The interval and state management are handled using React‘s `useEffect` and useState. The component returns JSX, which displays the advertisement corresponding to the current index in the main page.



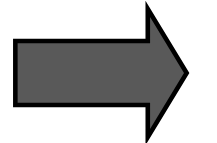


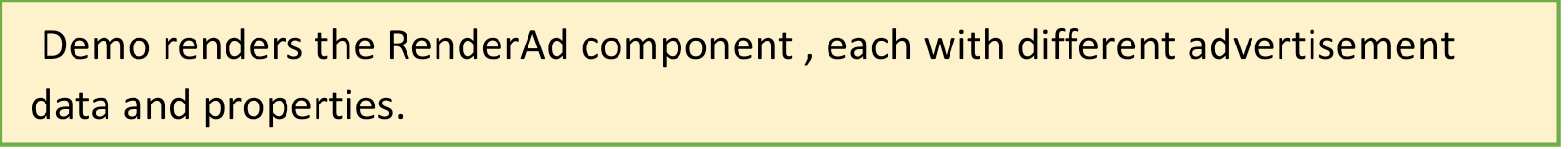


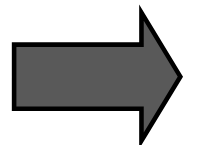


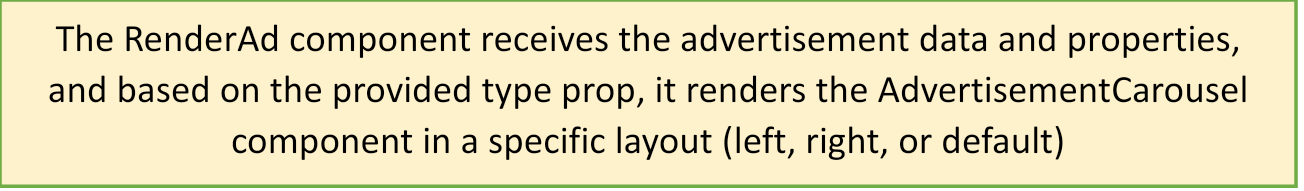
### Overall Workflow

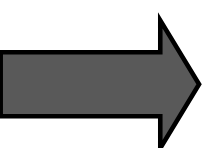


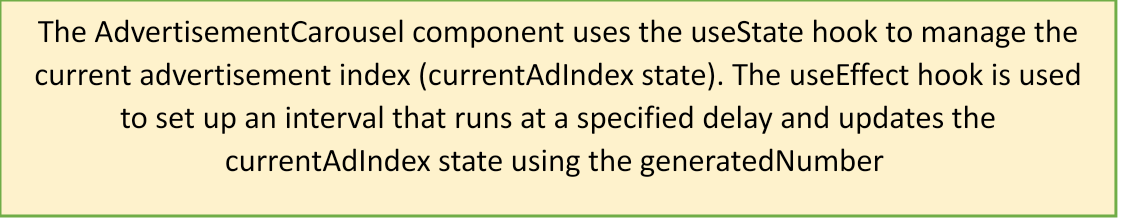




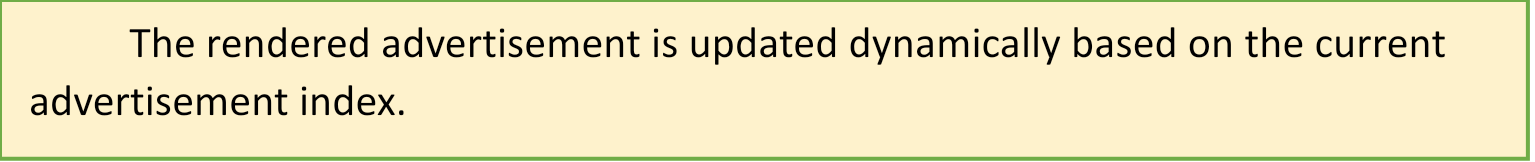


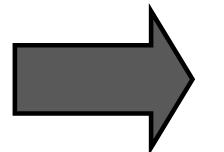


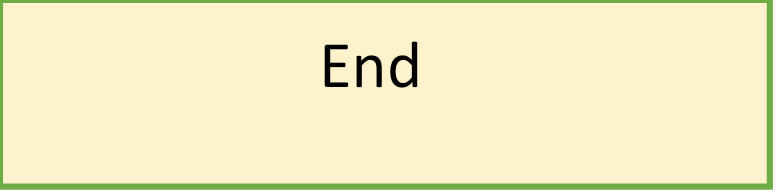






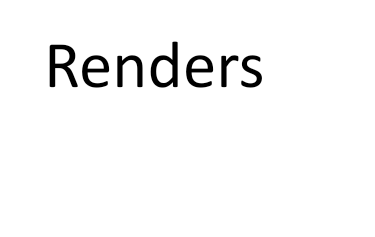
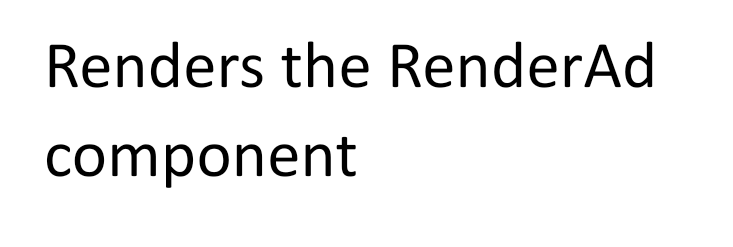


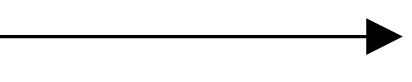




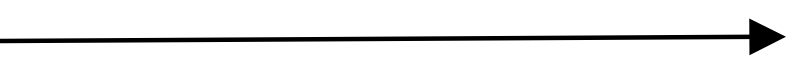
**Sequence diagram for AdDisplay:**

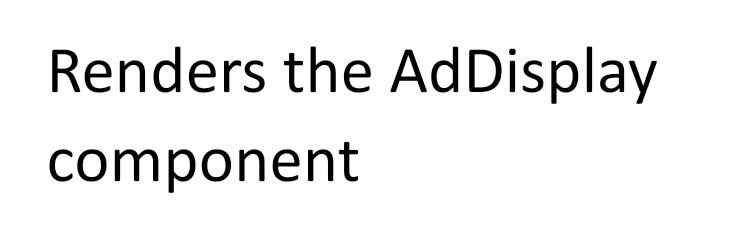
Demo(home page) RenderAd AdvertisementCarousel queue



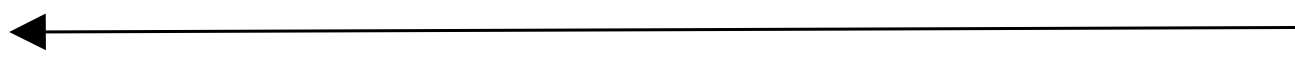




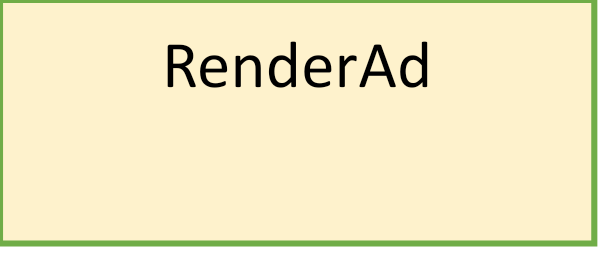
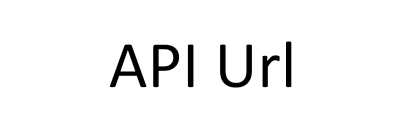
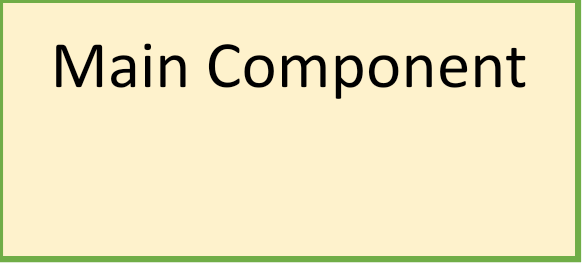


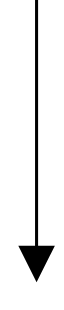
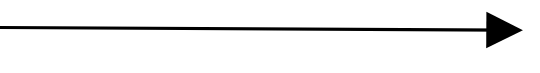




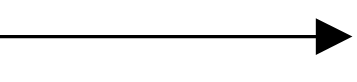
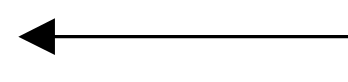
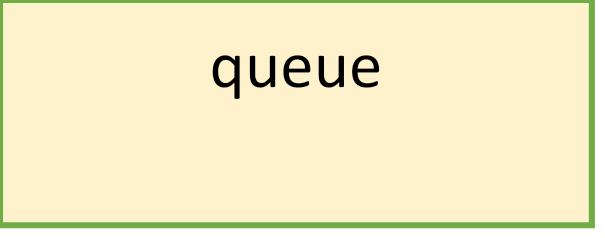


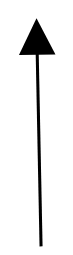
### Low level Design

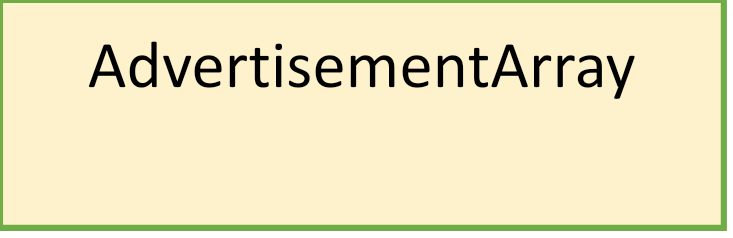






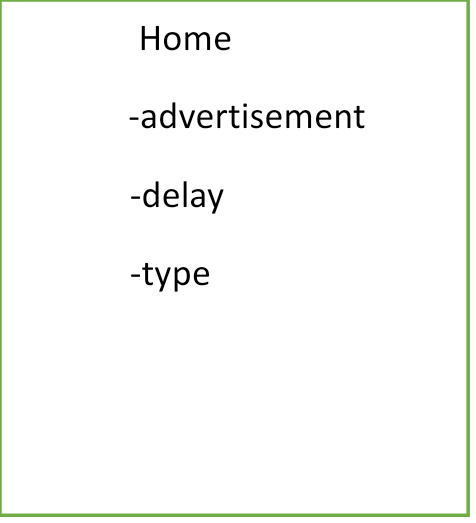


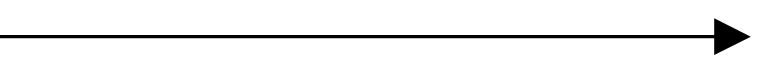






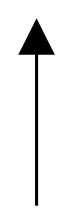
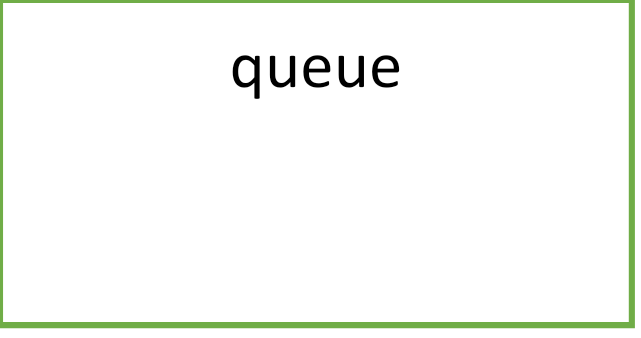
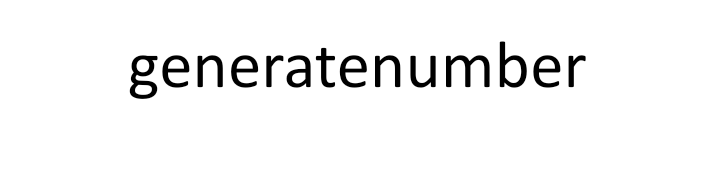
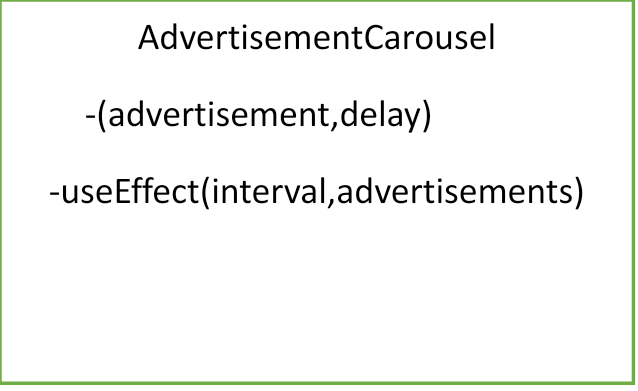












# Technology & Frameworks to be used

The Technology used is a React and the Framework used is javascript

# Solution Approach

1. In the `Demo` component, multiple instances of the `RenderAd` component are rendered, each with different advertisement data and properties.

2. The `RenderAd` component receives the advertisement data and properties, and based on the provided `type` prop, it renders the `AdvertisementCarousel` component in a specific layout (left, right, or default).

3. The `AdvertisementCarousel` component displays a container with an image tag inside, which initially shows the first advertisement from the provided array.

4. The `AdvertisementCarousel` component uses the `useState` hook to manage the current advertisement index (`currentAdIndex` state).

5. The `useEffect` hook is used to set up an interval that runs at a specified `delay` and updates the `currentAdIndex` state using the `generatedNumber` function to get a unique index from the advertisement array.

6. The interval continues to run until the component is unmounted or the `advertisements` prop changes, in which case the interval is cleared.

7. The rendered advertisement is updated dynamically based on the current advertisement index.

8. The `CircularLinkedList` class is used to manage a circular linked list that keeps track of unique indexes generated by the `generatedNumber` function.

9. The `generatedNumber` function generates a random number within the range of the advertisement array length and checks if it is already present in the circular linked list. If not, the number is appended to the list and returned as a unique index. If all attempts to find a unique index are exhausted, it returns null.

10. The `Demo` component and its child components work together to create a dynamic advertisement carousel with customizable layouts and rotation intervals.