

EPISODE DATE REACT APPLICATION

Good practices

- **Project Initialization:** Using Vite to initialize the Application, following the React recommendations of not using CRA anymore to Initialize any React project.
- **Project structure:** Structuring the project by folders by kind of file and having each Component in its own folder. Also having the API request functions separated in a services folder and segmenting each request call by functions.
- **Componentization:** Componentizing any separable layouts or piece of the application so it can have its own functions, variables, states, and processes. This way of working help us to handle and maintain our code in the best way, making it more scalable.
- **Async/Await:** Using async functions to handle the asynchronous requests.
- **Coding rules handle:** Using Eslint rules in order not to have non-used vars, redundant code, bad-performance importations, etc.
- **Code comments:** Commenting each function, service, mutation, applied method, or any piece of potentially confusing code that could be in the application to make it more scalable.
- **Error prevention:** Preventing any breakable part of the application having **if validations** if a parameter isn't sent so it can handle the missing of a required parameter.
- **Native Hooks use:** Using react hooks in the best way to handle, clear, set, and reset the variable values and behavior.
- **Conditional rendering:** Using ternaries in some Components to handle the friendliness of the Application when something happens.
- **Separation of Concerns:** Dividing the components into smaller functions to handle specific tasks.

Homescreen

Description:

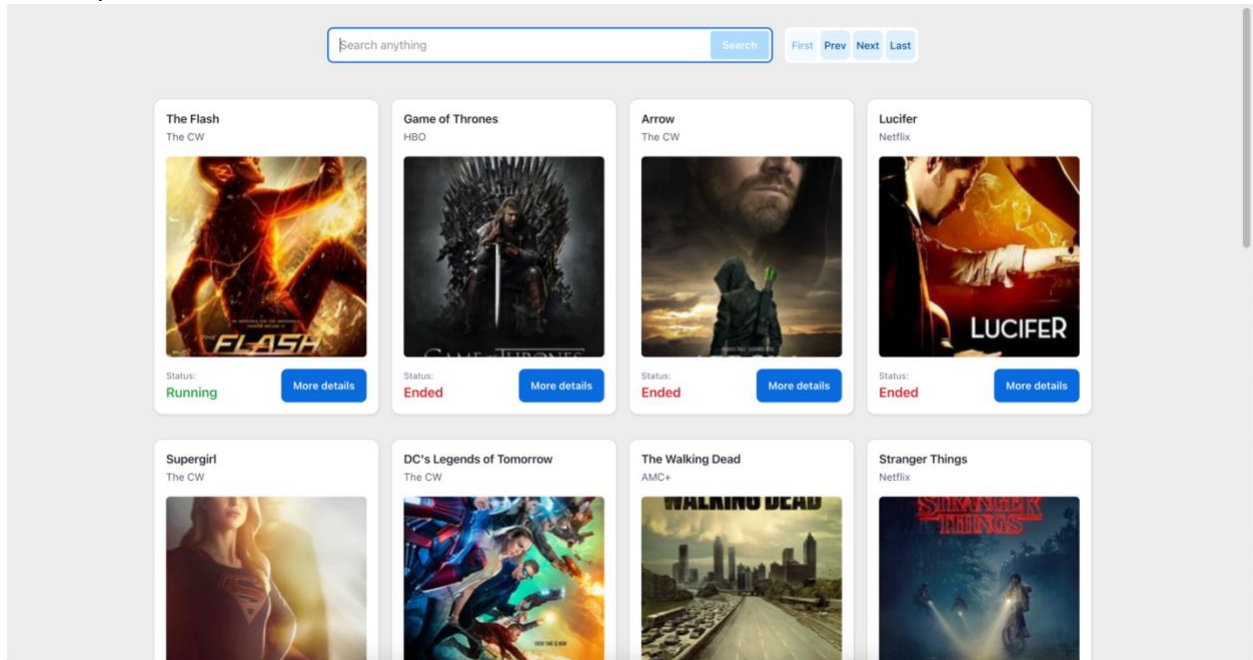
The Homescreen component represents the main screen of the React application. It manages the state and behavior of various components to display a list of movies fetched from an API. The component includes features such as search functionality, pagination, movie details display, loading indicators, and a search history.

Summary of Homescreen Component Behavior:

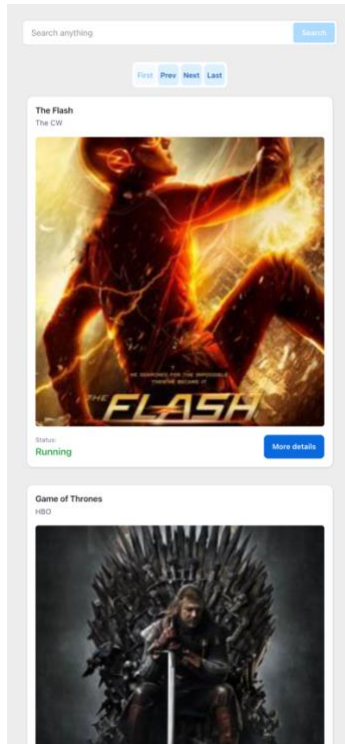
- Manages stative variables for *moviesData*, *requestQueries*, *searchHistory*, *openMovieDetails*, *isLoading*, *isLoadingDetails*, and *movieDetails*.
- Uses the **onGetMovies** function to fetch movies based on provided URL and optional parameters, updating the *moviesData* state and controlling the *isLoading* state.
- Handles query parameter changes with the **onQueryParamChange** function, updating the *requestQueries* state and triggering movie fetching. It also handles direct selection from the search history.
- Resets the search and requests the most popular movies with the **onResetSearch** function.
- Retrieves and sets movie details using the **onGetDetails** function, updating the *movieDetails* state and controlling the *isLoadingDetails* state. It also opens the movie details modal.
- Controls the visibility of the movie details modal with the **onCloseMovieDetails** function.
- Manages the search history by adding and retrieving search history items with the **onAddSearchHistoryParam** and **onGetSearchHistory** functions, respectively.
- Performs a movie search using the **onSearch** function, triggering movie fetching and updating the search history.
- Handles pagination by changing the current page with the **onChangePage** function, which updates the *requestQueries* and triggers movie fetching.
- Uses the useEffect hook to initialize the component, make the initial movie request for the most popular movies, and clean up the *moviesData* state on unmount.
- Renders the **MovieDetails** component when *openMovieDetails* is true, displaying the movie details modal.
- Renders the **SearchBar** component and the **Pagination** component within a Box container, allowing the user to perform searches and navigate through pages of movie data.
- Displays a **CircularProgress** component when *isLoading* is true, indicating that movie data is being fetched.
- Renders the **MovieCard** components within a Grid container when *moviesData.total* is greater than 0, displaying the movie cards with the corresponding movie data. Each **MovieCard** includes an option to show more details.
- Displays the **NoMovieFound** component when *moviesData.total* is 0, indicating that no movies were found.

Homescreen

Desktop view:



Mobile view:



MovieCard

Description:

The **MovieCard** component represents a card that displays information about a movie. It receives props including movie details, an **onShowDetails** function, and loading status.

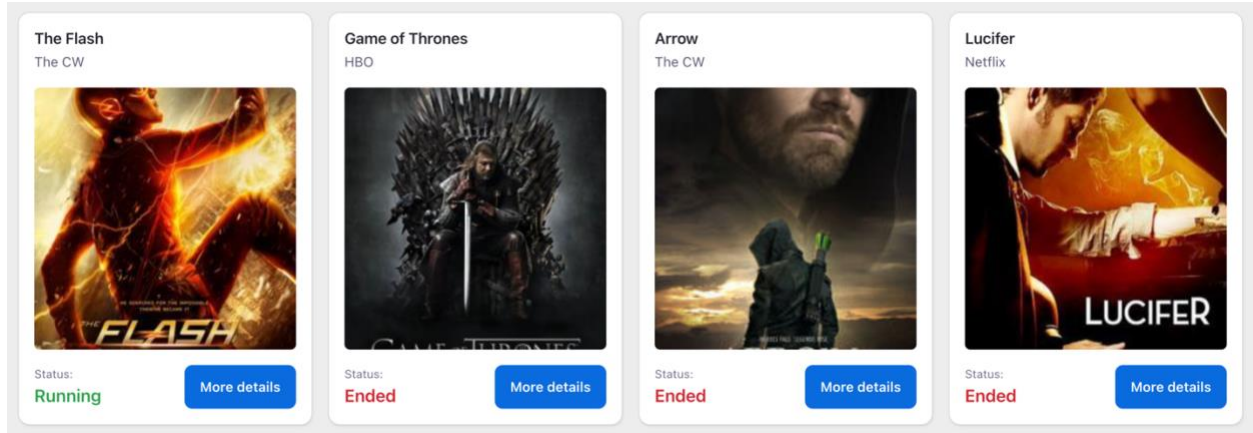
Summary of MovieDetails Component Behavior:

- Receives movie details, **onShowDetails** function, and loading status as props.
- Renders a Material-UI Card component to display the movie card.
- Displays the movie name using a **Typography** component with text overflow handling.
- Shows the network name using another **Typography** component.
- Displays the movie image using an img tag within an **AspectRatio** component.
- Renders a box with the status label and a button for more details.
- The status label is rendered using a **Typography** component, with color based on the movie's status.
- The button is rendered using a Material-UI **Button** component and triggers **onShowDetails** function.
- The button can display a loading state based on the loading prop.

The **MovieCard** component serves as a reusable card representation for displaying movie information. It provides a concise and visually appealing way to present movie details and allows users to access additional information about a movie through a button click.

MovieCard

Desktop view:



Mobile view:



MovieDetails

Description:

The **MovieDetails** component is responsible for displaying detailed information about a movie. It receives movie data as props and presents various details such as the movie *name*, *network*, *status*, *image*, *description*, *rating*, *start date*, *genres*, and *country*. It also showcases a list of pictures related to the movie. The component organizes the episodes into seasons and allows users to navigate through different seasons using tabs. It provides a comprehensive view of the movie's information, enhancing the user experience and providing an engaging presentation of the movie's details.

Summary of **MovieDetails** Component Behavior:

- Receives movie details, open status, and **onClose** function as props.
- Uses the useState hook to manage the seasons state.
- Implements the **divideEpisodesBySeasons** function to divide episodes into seasons.
- Calls **divideEpisodesBySeasons** function on component mount and clears the state on unmount.
- Renders a Material-UI **Modal** component to display movie details.
- Displays the movie *name*, *network*, and *status* at the top of the modal.
- Renders a divider below the movie details section.
- Renders the movie *image* on the left and other details on the right.
- Displays the movie *description*, *rating*, *start date*, *genres*, and *country*.
- Shows a list of pictures related to the movie.
- Renders tabs for each season of the show.
- Displays the episodes for the selected season in a list format.


MovieDetails

Desktop view:

The Flash

The CW

Ended



Description:

Barry Allen is a Central City police forensic scientist with a reasonably happy life, despite the childhood trauma of a mysterious red and yellow being killing his mother and framing his father. All that changes when a massive particle accelerator accident leads to Barry being struck by lightning in his lab. Coming out of coma nine months later, Barry and his new friends at STAR labs find that he now has the ability to move at superhuman speed. Furthermore, Barry learns that he is but one of many affected by that event, most of whom are using their powers for evil. Determined to make a difference, Barry dedicates his life to fighting such threats, as The Flash. While he gains allies he never expected, there are also secret forces determined to aid and manipulate him for their own agenda.

Rating: 9.3171

Start Date: 2014-10-07

Genres: Drama, Action, Science-Fiction

Country: US

Season 1

Season 2

Season 3

Season 4

Season 5

Season 6

Season 7

Season 8

Season 9

1. Pilot

Air date: 2014-10-08 00:00:00

2. Fastest Man Alive

Air date: 2014-10-15 00:00:00

Desktop view:

The Flash

The CW

Ended



Description:

Barry Allen is a Central City police forensic scientist with a reasonably happy life, despite the childhood trauma of a mysterious red and yellow being killing his mother and framing his father. All that changes when a massive particle accelerator accident leads to Barry being struck by lightning in his lab. Coming out of coma nine months later, Barry and his new friends at STAR labs find that he now has the ability to move at superhuman speed. Furthermore, Barry learns that he is but one of many affected by that event, most of whom are using their powers for evil. Determined to make a difference, Barry dedicates his life to fighting such threats, as The Flash. While he gains allies he never expected, there are also secret forces determined to aid and manipulate him for their own agenda.

Rating: 9.3171

Start Date: 2014-10-07

Genres: Drama, Action, Science-Fiction

NoMovieFound

Description:

The **NoMovieFound** component represents a message displayed when no movies are found for a search query. It includes a message, an image, and a button to navigate back to the homepage. Here's a breakdown of its key features and responsibilities:

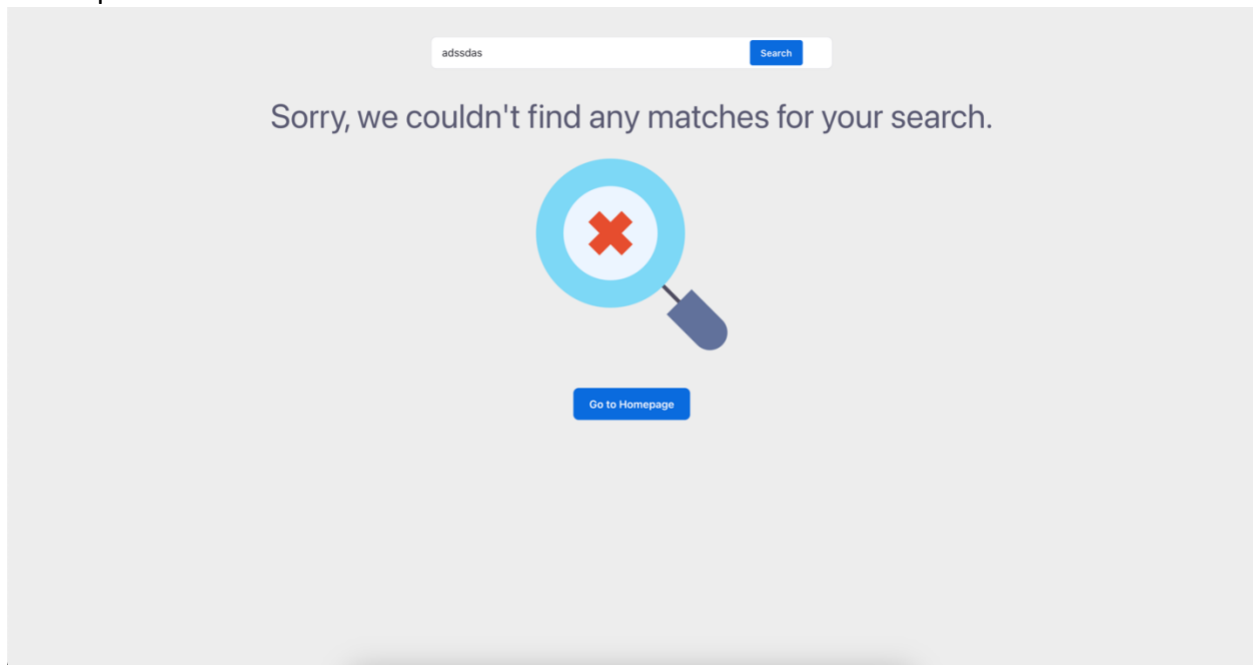
Summary of NoMovieFound Component Behavior:

- Receives the **goToHome** function as a prop, which is called when the "Go to Homepage" button is clicked.
- Renders a **Box** component to contain the message, image, and button.
- Displays a **Typography** component with a custom style for the message. The message informs the user that no matches were found for their search.
- Shows an image using an **img** tag. The image source is specified as `"../..../public/search.png"`, and it has a width of 300 pixels and a margin-top of 20 pixels. The alt attribute is set to "not found".
- Includes a **Box** component with custom styles to align the button to the center.
- Renders a **Button** component with the label "Go to Homepage" and a large size. The `onClick` event triggers the `goToHome` function when the button is clicked.

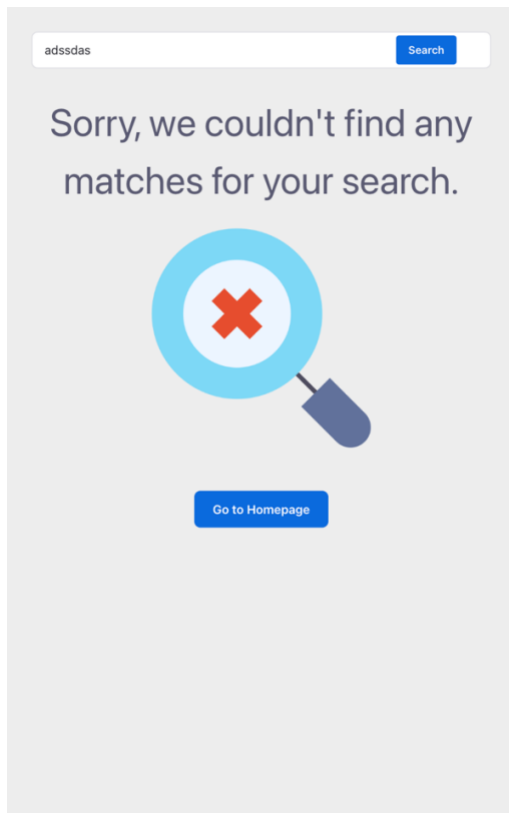
Overall, the **NoMovieFound** component provides a visually appealing message with an image and a button to navigate back to the homepage when no movie matches are found for a search query.

NoMovieFound

Desktop view:



Mobile view:



SearchBar

Description:

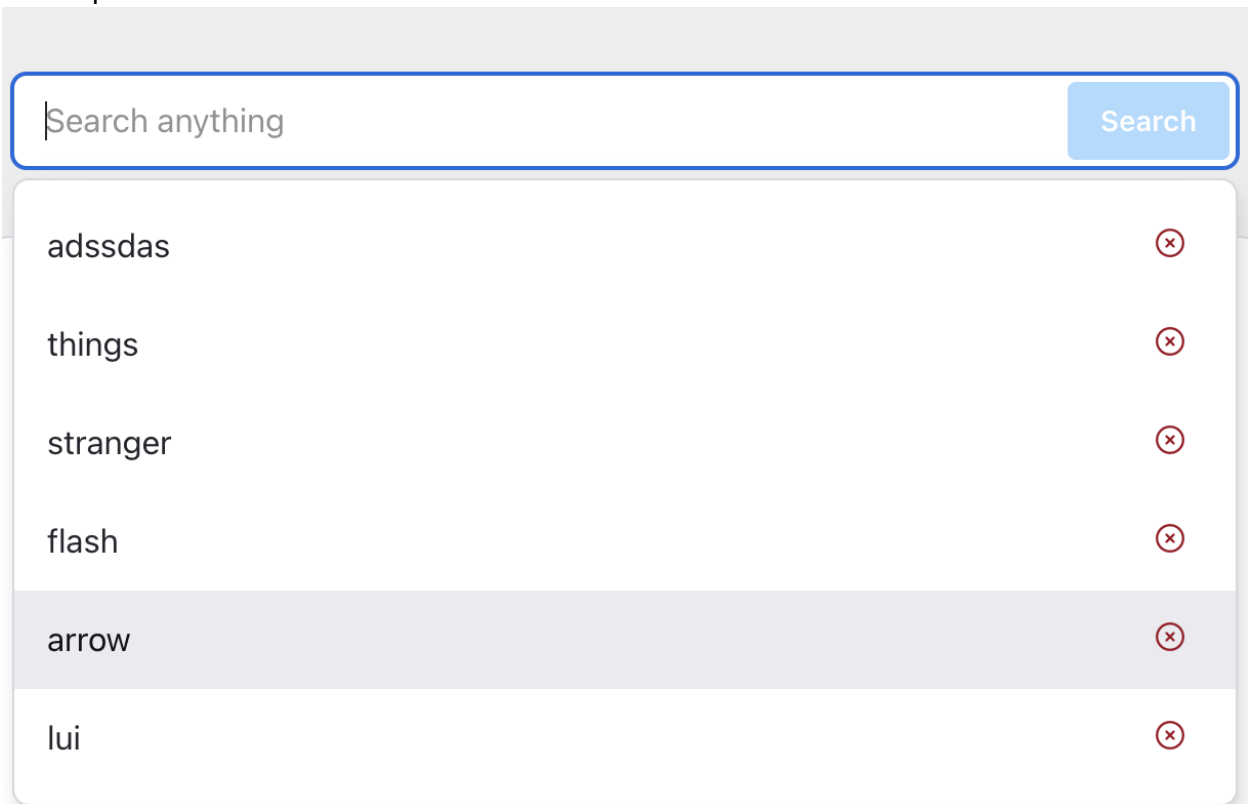
The **SearchBar** component represents a search input field with autocomplete functionality. It allows users to enter search queries, select from a search history, delete history items, and trigger a search action. Here's a breakdown of its key features and responsibilities:

Summary of **SearchBar** Component Behavior:

- Manages the input value state using the useState hook to handle changes in the input field.
- Handles input value changes with the **onInputChange** function, which triggers the **onChange** function and updates the input value state.
- Handles changes in the history value clicked with the **onParamChange** function. It updates the input value state and triggers the **onChange** function with the clicked value.
- Deletes an item from the search history using the **onDeleteHistoryItem** function. It prevents the click event propagation, calls the **deleteHistoryElement** function, and updates the search history by calling the **onFocus** function.
- Triggers the search function when the "Enter" or "Return" key is pressed using the **onEnterKeyUp** function.
- Resets the input value to an empty string if the queries prop has a nullish value using the useEffect hook.
- Renders an **Autocomplete** component with specific configurations, such as width, name, event handlers, input value, and options.
- Provides a *renderOption* function to customize the appearance of each autocomplete option. It includes the option text and a delete button to remove the history item.
- Populates the autocomplete options with values from the *historyList* prop.
- Displays a placeholder text and enables free text input.
- Renders an *endDecorator* component, which is a Button with the label "Search" and triggers the **onSearch** function when clicked or when the "Enter" key is pressed.
- Overall, the **SearchBar** component provides a user-friendly search input field with autocomplete functionality, search history support, and the ability to trigger search actions.

SearchBar

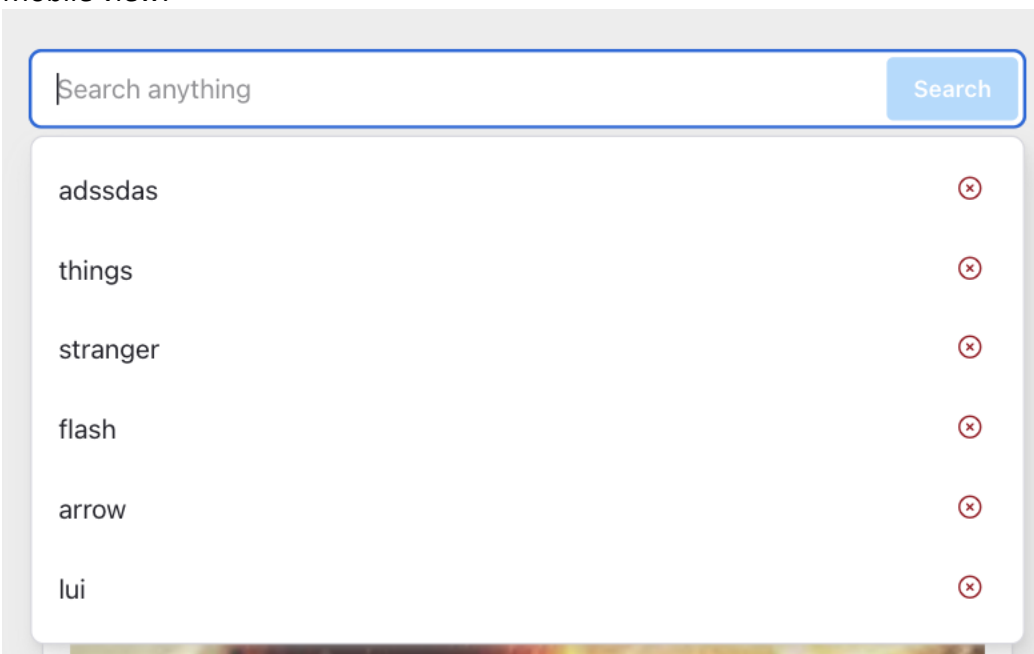
Desktop view:



The desktop view of the SearchBar component features a wide search input field with a blue border and a light blue 'Search' button on the right. Below the input field is a list of search results. The 'arrow' result is highlighted with a light gray background. Each result has a red 'x' icon in a circle on the right side.

Search Results	Action
adssdas	✕
things	✕
stranger	✕
flash	✕
arrow	✕
lui	✕

Mobile view:



The mobile view of the SearchBar component features a search input field with a blue border and a light blue 'Search' button on the right. Below the input field is a list of search results. Each result has a red 'x' icon in a circle on the right side.

Search Results	Action
adssdas	✕
things	✕
stranger	✕
flash	✕
arrow	✕
lui	✕

Pagination

Description:

The **Pagination** component is responsible for rendering a set of navigation buttons that allow the user to navigate through the pages of movie data displayed on the **Homescreen** component. It receives various props including the *moviesData* object, the current page number, the **setRequestQueries** function, and the **onChangePage** function.

Summary of **Pagination** Component Behavior:

- Renders a set of navigation buttons for pagination using the **IconButton** component.
- The navigation buttons include "First," "Prev," "Next," and "Last" options.
- Clicking the "First" button calls the **firstPage** function, which updates the requestQueries and triggers a request for the first page of movies.
- Clicking the "Prev" button calls the **previousPage** function, which updates the requestQueries and triggers a request for the previous page of movies based on the current page number.
- Clicking the "Next" button calls the **nextPage** function, which updates the requestQueries and triggers a request for the next page of movies based on the current page number.
- Clicking the "Last" button calls the **lastPage** function, which updates the requestQueries and triggers a request for the last page of movies based on the total number of pages in the *moviesData* object.
- The navigation buttons are disabled under certain conditions to prevent invalid navigation.
- The *paginationOptions* array maps the buttons into an object format that is passed as props to the **IconButton** component, including the button's label, **onClick** function, and disabled status.
- The **Pagination** component is displayed as a **Box** container with a white background, displaying the navigation buttons horizontally.
- The **IconButton** component is used to render each navigation button, with a font size of 14 and the corresponding label as its text content.

Overall, the **Pagination** component provides an intuitive navigation interface for the user to move between pages of movie data displayed on the Homescreen component.

Pagination

Desktop view:



Mobile view:



Extra pictures

Search anything

Cancel

Clear

More

More

More

More

More

The Flash

The CW

Running

More details

Game of Thrones

HBO

Ended

More details

Arrow

The CW

Ended

More details

Lucifer

Netflix

Ended

More details

Supergirl

The CW

Ended

More details

DC's Legends of Tomorrow

The CW

Ended

More details

The Walking Dead

AMC

Ended

More details

Stranger Things

Netflix

Running

More details

Dragon Ball Super

Fuji TV

Ended

More details

Boku no Hero Academia

HBO

Running

More details

The 100

The CW

Ended

More details

Sherlock

BBC One

Ended

More details

Supernatural

The CW

Ended

More details

The Big Bang Theory

CBS

Ended

More details

Marvel's Agents of S.H.I.E.L.D.

ABC

Ended

More details

Marvel's Daredevil

Netflix

Ended

More details

Season 1

Season 2

Season 3

Season 4

Season 5

Season 6

Season 7

Season 8

Season 9

1. Pilot

2. Fastest Man Alive

3. Things You Can't Outrun

4. Going Rogue

5. Plastique

6. The Flash Is Born

7. Power Outage

8. Flash vs. Arrow

9. The Man in the Yellow Suit

10. Revenge of the Rogues

11. The Sound and the Fury

12. Crazy for You

13. The Nuclear Man

14. Fallout

15. Out of Time

FLASH

ALL NEW TUES 8/7c CW

Season 1

Season 2

Season 3

Season 4

Season 5

Season 6

Season 7

Season 8

Season 9

1. Pilot

2. Fastest Man Alive

3. Things You Can't Outrun

4. Going Rogue

5. Plastique

6. The Flash Is Born

7. Power Outage

8. Flash vs. Arrow

9. The Man in the Yellow Suit

10. Revenge of the Rogues

11. The Sound and the Fury

12. Crazy for You

13. The Nuclear Man

14. Fallout

Air date: 2014-10-08 00:00:00

Air date: 2014-10-15 00:00:00

Air date: 2014-10-22 00:00:00

Air date: 2014-10-29 00:00:00

Air date: 2014-11-12 01:00:00

Air date: 2014-11-19 01:00:00

Air date: 2014-11-26 01:00:00

Air date: 2014-12-03 01:00:00

Air date: 2014-12-10 01:00:00

Air date: 2015-01-21 01:00:00

Air date: 2015-01-28 01:00:00

Air date: 2015-02-04 01:00:00

Air date: 2015-02-11 01:00:00

Air date: 2015-02-18 01:00:00