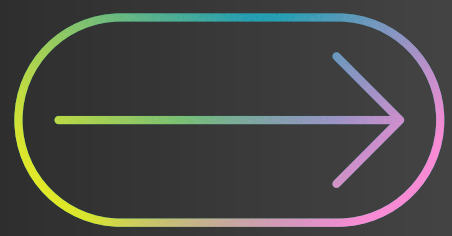
INFINITY SCROLL

With Dynamic params

Slide to know



Blessedraj

Installation

```
//installation
npm install use-dynamic-infinite-scroll

//import
import useDynamicInfiniteScroll from 'use-dynamic-
infinite-scroll';
```

Blessedraj

Fetch Function

myComponent.tsx

```tsx
const fetchReports = async (page: number, params: any) => {
   try{
     const response: any = await apigetData(
       page,
       params.search,
       params.year
   );
   const reports: Report[] = response.data;
   const hasMore = response.hasNext;

     return { data: reports, hasMore };
  }catch(error){
    console.log("error fetching the data")
  }
}
```

api-services.ts

```ts
export const apigetData = (page: number, search?: string, year?: string)
  const params = new URLSearchParams();
  params.append("page", page.toString());
  if (search) {params.append("search", search);}
  if (year) {params.append("year", year);}
  return axios.get(`/url?${params.toString()}`);
};
```

Blessedraj

Usage

- scrollLoading -> True while loading the next page of data.
- initialLoading-> True during the first API call or when search/filter parameters change.
- hasmore->Boolean flag indicating if more data is available to load.
- loaderRef->Ref attached to a sentinel div at the bottom that triggers next page fetch when it enters the viewport.

```
export function mycomponent={
const fetchReports={.......}
const { data,scrollLoading,initialLoading,hasMore,loaderRef,
  updateParams } = useDynamicInfiniteScroll(fetchReports,0);
}
const handleFilter=((value:string)⇒{
  updateParams(value)
})
return (
<>
</>
)
```

Blessedraj

Usage

Add the ref below you container eg table,card etc....

```
export function mycomponent={

return (
<div>
  {initialLoading ? (
    <div>Loading data...</div>  // initial loading
  ) : ( <> //actual mapping
      {data.map((item, index) => (
         <div key={index}>
              {item.title}
              {/* loaderRef at last element */}
              {index === data.length - 1 && <div ref={loaderRef}></div>}
            </div>
      ))}
      //scroll loading
      {scrollLoading && <div>Loading more data...</div>}
      {/* No More Data */}
      {!hasMore && !initialLoading && (
        <div>No more data available.</div>
      )}
    </>
  )}
</div>
```

Blessedraj

Curious about scalable API integration and service architecture strategies???

comment API

Blessedraj