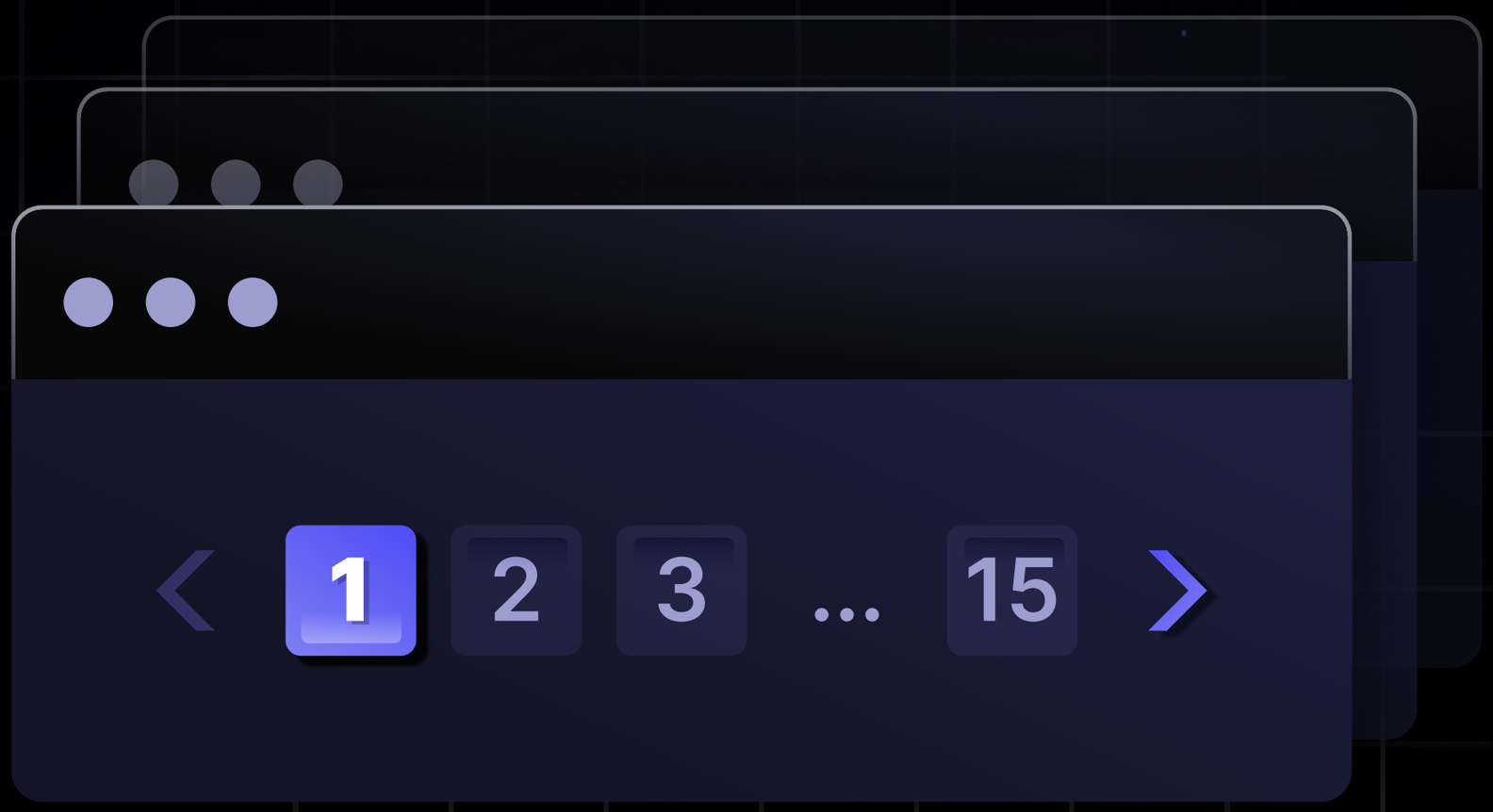
Front End Techniques

API Pagination

What is it?

When your data becomes too large, it is no longer feasible to fetch all the data upfront. Pagination is a technique used to divide large sets of data into manageable chunks or pages and process one chunk / page at a time.

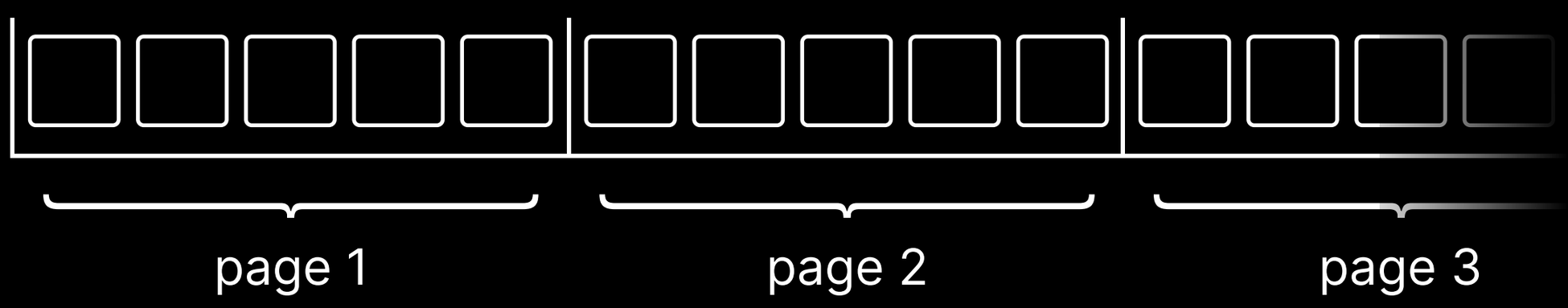


GreatFrontEnd

Offset-based Pagination

This method involves using an offset to specify where to start retrieving data and the number of items to retrieve. The offset can be explicit or derived from the requested page number.

E.g.: `page=3&size=5`



When the client requests for page 3 with a size of 5, it is translated to an offset of 10 (2 pages). When fetching from the database it becomes: `SELECT * FROM table LIMIT 5 OFFSET 10`

Use cases: Best for small to medium-sized datasets where the items aren't updated very frequently.

Offset-based Pagination

Pros vs Cons

👍 Advantages:

- Easy to implement and understand
- Users can jump to any page directly.
- Easily used with various database systems and not tied to specific data storage mechanisms

👎 Disadvantages:

- Performance issues with large datasets, as the offset grows the query can become slower.
- Not suitable for real-time data; if data is added or removed, the pages can show overlapping or missing items.

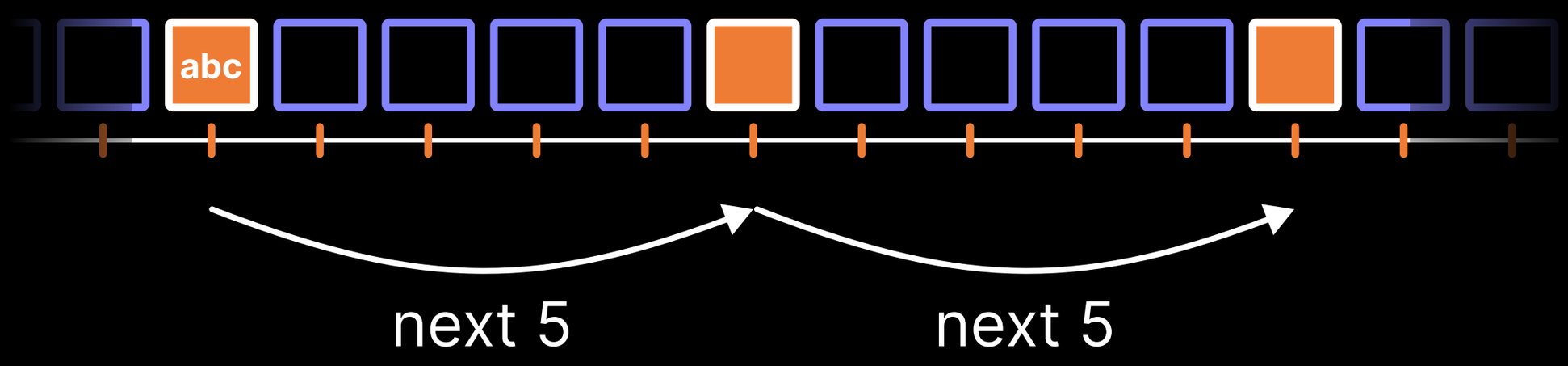
Examples: Common in web applications for displaying lists like search results, where jumping to a specific page is required.

E.g. blogs, travel booking, e-commerce websites.

Cursor-based Pagination

This method uses a pointer (the cursor) to a specific item in a dataset. Instead of saying “give me 5 items after the first 10 items” it says “give me 5 items starting after [specified item].”

E.g.: `cursor=abc&next=5`



The cursor is a unique identifier, which can be the item id or timestamp. Subsequent requests use the identifier of the last item to fetch the next set of items. In SQL, an example is: `SELECT * FROM table WHERE id > cursor LIMIT 5.`

Use cases: Ideal for large, dynamic datasets, especially where data is frequently added or modified.

Cursor-based Pagination

Pros vs Cons

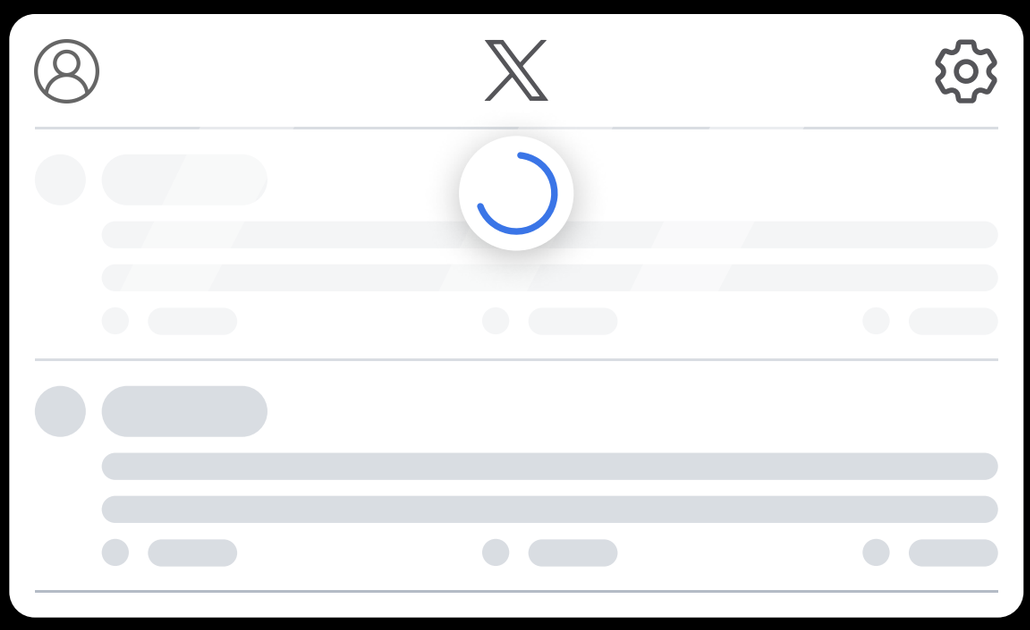
Advantages:

- More efficient and faster on large datasets.
- Handles real-time data well; no overlapping or missing items as data is added or removed.
- Consistent user experience even when data changes.

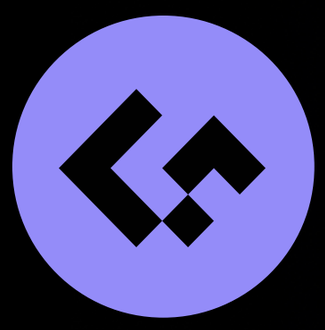
Disadvantages:

- Users cannot jump to a specific page without going through the previous pages.
- Slightly more complex to implement compared to offset-based pagination.

Examples: Common in social media feeds, where new data is constantly being added and the order of items may change. E.g Facebook, Pinterest, Twitter



Follow us for more system design tips



GreatFrontEnd