NBA Players UI

Tasks:

1. Create a new app for each front (FE)
2. Create a client ui with two lists, arranged side by side:
   1. Left side list
      1. Fetch a list of NBA players from the API
      2. Implement search functionality that allows filtering players by name using the API.
      3. Add pagination support to display a limited number of players per page.
   2. Right side list
      1. Display a list of favorite players with the player stats from the current (or previous) season’s statistics.
3. You may utilize a state management library if the situation calls for it.
4. Additional Requirements:
   1. Optimize the API
   2. Implement error handling and loading states for API requests.
   3. Use CSS Modules or a CSS-in-JS solution for better styling isolation.
   4. Implement responsive design to ensure the UI works well on various screen sizes.
   5. Select your component library or build your own for consistent and polished UI components.
   6. Leverage typescript to the fullest.
   7. Proper breakdown of components and code separation (custom hooks, utility functions, etc).

Guidelines:

1. <https://docs.balldontlie.io/#introduction> API
   1. The API contains data from 1946-current. An API key is required. You can obtain an API key by creating a free account on [our website](https://new.balldontlie.io/). Read the [authentication section](https://docs.balldontlie.io/#authentication) to learn how to use the API key.
2. Begin with a CRA or Vite and TypeScript using one of the provided commands.
3. Create a detailed technical design that covers component breakdown, state management architecture, API interaction, and any complex features you plan to implement.
4. Encourage the use of best practices such as code splitting, lazy loading, and memoization for performance optimization.
5. The submission of both, the technical design document and the final code as a Github repository, is required.
6. The quality of the codebase will be assessed based on factors such as (and not limited to) modularity, reusability, and error handling.
7. Consider providing feedback on alternative design choices and trade-offs to encourage critical thinking.