**Question 1: How did you manage to fetch the list and what tool did you use?**

To retrieve the Pokémon list, I employed **Redux Toolkit Query (RTK Query)**, which is included in the **@reduxjs/toolkit** package. This tool facilitates API interactions by offering a declarative and efficient method for managing data retrieval and caching.

Here is a brief overview of the process:

1. **API Slice Configuration:** I established an API slice using createAPI from RTK Query. This slice specifies the base URL (https://pokeapi.co/api/v2/) and an endpoint for obtaining the Pokémon list (pokemon?limit=151).
2. **Hook Generation:** RTK Query automatically created a custom hook for data retrieval, specifically useGetPokemonQuery. This hook oversees the complete lifecycle of the request, encompassing loading states, error management, and caching.
3. **Integration:** I incorporated the hook into the App component to fetch the Pokémon list. The data is presented dynamically through a grid-based UI component, which is styled using Tailwind CSS.

By utilizing RTK Query, I ensured a streamlined process, benefiting from built-in caching and re-fetching capabilities for optimal performance.

**Question 2: What steps would you take to future improve this?**

To enhance and future-proof this implementation, I would concentrate on the following areas for improvement:

* 1. Error Handling and Retry Logic
* 2. State Normalization
* 3. Dynamic Caching Strategies
* 4. Lazy Loading
* 5. Improved UI/UX
* 6. Comprehensive Testing and Optimization