What is Redux ?

- Redux is a state management library for JavaScript applications, commonly used with React.
- It helps manage the state of an application in a predictable

What is Redux Thunk used for?

- Redux Thunk is a middleware that allows you to write action creators that return a function instead of an action.
- This can be useful for handling asynchronous operations, such as API calls, within your Redux actions.
- The returned function receives the dispatch and getState methods as arguments, allowing you to dispatch actions and access the state.

What is a Pure Component? When to use Pure Component over Component?

- A Pure Component in React is a component that performs a shallow comparison of props and state to determine whether the component should re-render.
- This can lead to performance improvements by preventing unnecessary renders.

You should use Pure Component over Component when:

- The component renders the same output given the same props and state.
- You want to avoid unnecessary re-renders for performance optimization.

What is the second argument that can optionally be passed to setState and what is its purpose?

- The second argument to setState is a callback function that is executed once the state has been updated and the component has re-rendered.
- This is useful for executing code that depends on the updated state or for performing side effects after the state update.

Create a Table and Search Data from Table using React JS

```
 React, { useState } from "react";
 import export default App = () => {
  const [searchTerm, setSearchTerm] = useState("");
  const [data] = useState([
    { id: 1, name: "John Doe", age: 28 },
    { id: 2, name: "Jane Smith", age: 34 },
    { id: 3, name: "Michael Johnson", age: 45 },
  ]);
  const handleSearch = (event) => {
    setSearchTerm(event.target.value);
  };
  const filteredData = data.filter((item) =>
    item.name.toLowerCase().includes(searchTerm.toLowerCase())
  );

  return (
    <div>
      <h2>Searchable Table</h2>
      <input
        type="text"
        placeholder="Search by name"
        value={searchTerm}
        onChange={handleSearch}
      />
      <table border="1">
```

```
        <thead>
          <tr>
            <th>ID</th>
            <th>Name</th>
            <th>Age</th>
          </tr>
        </thead>
        <tbody>
          {filteredData.map((item) => (
            <tr key={item.id}>
              <td>{item.id}</td>
              <td>{item.name}</td>
              <td>{item.age}</td>
            </tr>
          ))}
        </tbody>
      </table>
    </div>
  );
};
```