1. **What is the difference between Named Export, Default export and * as an export?**

   JavaScript modules provide a mechanism to organize and share code across different files. To export and import values between modules, we use the `export` and `import` keywords.

   Named Exports:
   You can export multiple values from a single module, each with a specific name. When importing, you must specify the exact names of the values you want to import.

```
export const name = "Alice";
export const age = 30;


import { name, age } from './module.js';
```

   Default Export:
   You can export a single value as the default export from a module. When importing, you can assign any name to the imported value.

```
export default resList;
```

```
import resList from "../utils/mockData";
```

   *** as export:**
   You can re-export all named exports from another module.

   E.g module.js:

```
export const name = "Alice";
export const age = 30;
```

   reexport.js:

```
export * from './module.js';
```

   main.js:

```
import { name, age } from './reexport.js';
```

2. **What is the importance of config.js file?**
   A `config.js` file is a crucial component in many React.js projects, serving as a centralized repository for various configuration settings. Its primary purpose is to separate configuration from the main application logic, making the codebase more maintainable, flexible, and adaptable to different environments**.**

   Common Configuration Settings in a `config.js` File:

- **API URLs:** Base URLs for API endpoints
- **Database Credentials:** Connection strings and credentials for databases
- **Authentication Settings:** Authentication tokens, secret keys, and other authentication-related information
- **Logging Levels:** Controls the level of logging verbosity (debug, info, warn, error)
- **Feature Flags:** Enables or disables specific features based on configuration
- **Third-Party Service Credentials:** Credentials for services like Google Analytics, Stripe, or AWS

3. **What are React Hooks?**

   React Hooks are functions that allow developers to use React features in functional components without writing class components:

   What they do

   - Hooks "hook into" React's state and lifecycle features, allowing developers to:
   - Manage state
   - Access lifecycle features
   - Add features like keeping track of data
   - Do things when a component loads or updates

   Some common React Hooks include:

   - useState()
   - useReducer()
   - useEffect()
   - useLayoutEffect()
   - useCallback()
   - useMemo(

4. **Why do we need a useState Hook?**

   Think of `useState` as a magical tool that allows your component to remember things. It helps your component keep track of its own state,

   for example like the number of items in your to-do list or the current text in a text input.

   .