Task 2 : BackGround Process Of JS FrameWorks

ReactJs :

- **Virtual DOM** : Creates a virtual representation or a copy of DOM called Virtual DOM or vDOM. React compares the virtual DOM to the real DOM to only render components that have changed instead of rendering the whole page.
- **State Management**: React has built-in useState and useReducer hooks for managing state.
- **Performance**: Optimizes re-renders with React.memo, useMemo, and lazy loading.

AngularJS:

- **Two Way DataBinding** : means that data can flow in both directions — from the model to the view and from the view to the model.
- **Change Detection**: Angular works primarily with the real DOM, but it improves performance by using efficient change detection strategies to minimize unnecessary DOM updates.
- **Ahead-of-Time (AOT) Compilation:** Converts Angular HTML and TypeScript code into efficient JavaScript at build time.

VueJs :

- **Reactive Data Binding** : Vue provides a reactive system where changes in data automatically update the DOM.
- **Vuex (State Management) :** Centralized state management library for managing application-wide data.

NextJs :

- **File-Based Routing :**Each file in the pages directory automatically becomes a route.
- **Server-Side Rendering (SSR) :** Pages are rendered on the server at runtime, ensuring content is always up-to-date and SEO-friendly.

SolidJs:

- **Fine-Grained Reactivity :** Updates are fine-grained, meaning only the parts of the DOM affected by changes are updated, without needing a virtual DOM.
- **Built-in Reactivity Primitives :** SolidJS uses primitives like signals, memos, and stores for state management.

Task 3 : NVM vs YARN vs PVPM

Which One Should You Use?

Choose npm if:

- You want the default, built-in package manager.
- You are working on simpler projects or are already familiar with npm.

Choose Yarn if:

- You need faster performance and advanced features like Workspaces.
- You are working in a monorepo setup and require better dependency resolution.

Choose pnpm if: (required knowledge to use it)

- You want the most efficient storage and installation.
- You are working on large projects or monorepos with strict dependency isolation requirements.
- You prioritize speed and disk space optimization.

Task 4 : Which and Where to use Browser Storage.

LocalStorage:

- **Persistent storage** : For to Store persistent Data.
- **Storing non-sensitive data**: Keep data like user interactions, UI states, or caching API responses.

SessionStorage:

- **Storing temporary data**: Keep data that is only needed while the user is interacting with the current session.

Cookies:

- **Tracking and Analytics**: Use cookies for user tracking and storing preferences for analytics purposes.
- When you need to send data to and from the server with each request
- When you want to set an expiration date for your data.

IndexedDB:

- **Purpose**: Advanced, structured storage for large datasets.
- **Storage Limit**: Hundreds of MBs depends on the browser and disk space.

Cache :

- **Purpose**: Storage for HTTP request/response pairs for offline use.
- **Persistence**: Data persists until explicitly deleted or replaced.