MVC / MVVM based design?

What is a Component?

In the context of software development, a component is a modular, reusable piece of code that performs a specific function or set of functions within a larger software system. A component can be a standalone piece of software or can be integrated with other components to create a larger application. Components typically have well-defined interfaces that allow them to communicate with other components in the system, and they may be designed to be platform-independent, meaning they can be used across different types of applications and environments.

**React JS Component**

Component-based design in React JS is an approach to building user interfaces by breaking them down into reusable, independent pieces of code called components. Components are designed to be self-contained, encapsulating all the necessary logic and styling for a particular feature or UI element. This makes it easier to manage complex UIs and promotes code reusability, scalability, and maintainability. In React, components are created using either classes or functions, and they can be composed together to create larger, more complex UIs.

**Here are some principles of component design:**

1. **Reusability**: Components should be designed to be reusable in different parts of an application or in different applications altogether.
2. **Single Responsibility**:
3. **Encapsulation**: Components should be self-contained and should not depend on other components or external libraries to function properly. This makes them easier to develop, test, and maintain.
4. **Modularity**: Components should be designed to be modular, meaning that they can be easily combined with other components to create more complex functionality.
5. **Composability**: Components should be designed to be composable, meaning that they can be combined with other components in different ways to create a variety of user interfaces.
6. **Scalability**: Components should be designed to be scalable, meaning that they can handle a large number of inputs and outputs without sacrificing performance.
7. **Accessibility**: Components should be designed with accessibility in mind, so that they can be used by people with disabilities or who are using assistive technologies.
8. **Testability**: Components should be designed to be easily testable, so that developers can quickly identify and fix issues during the development process.

By following these principles, developers can create robust and reusable components that can be used in a wide range of applications and contexts.

Component based technologies

1. Angular framework : google
2. React JS library : Facebook
3. Vue.js

**React JS advantages**

React JS offers several advantages for web development, including:

1. Component-Based: React uses a component-based approach to building UIs, which makes it easier to build reusable and modular code. Components can be composed together to create complex UIs, making it easier to manage and scale large applications.
2. Virtual DOM: React uses a virtual DOM, which improves the performance of the application by minimizing the number of changes that need to be made to the actual DOM. This leads to faster rendering and a better user experience.
3. Declarative Programming: React allows developers to write code in a declarative way, which makes it easier to reason about the code and understand what it does. This also makes it easier to debug and maintain the code.
4. Rich Ecosystem: React has a large and growing ecosystem of tools and libraries, making it easier to build complex applications. It is also backed by Facebook, which provides ongoing support and development for the framework.
5. SEO-Friendly: React is designed to be SEO-friendly, meaning that it can be easily crawled by search engines, resulting in better search engine rankings for websites built with React.
6. Cross-Platform: React can be used to build applications for the web, mobile devices, and desktops, making it a versatile choice for developers.

By leveraging these advantages, developers can build high-performance, scalable, and maintainable applications with React.

In React JS, the virtual DOM (VDOM) is a lightweight copy of the actual Document Object Model (DOM) that represents the current state of the user interface. The VDOM is used to optimize the rendering process by minimizing the number of updates required to the actual DOM, which can be an expensive operation.

When the state of a component changes, React updates the VDOM and compares it to the previous version to determine the minimum set of changes required to update the actual DOM. This approach reduces the number of actual DOM manipulations required, leading to improved performance and a better user experience.

The VDOM is an in-memory representation of the actual DOM and is maintained by React. It allows developers to write code as if the entire page is rendered on every change, without actually having to perform those costly operations.

Overall, the VDOM is a key feature of React and is one of the reasons why React is so performant and popular among web developers.