In Vue.js, a component is essentially a self-contained instance of a Vue.js application. Each component may have its own data, computed properties, methods and lifecycle hooks which makes it easy to reuse code across your application.

When you define a new component in Vue.js using the Vue.component() method or single file components (SFCs), you are creating a new constructor function that can be used to create instances of that component throughout your application.

To create an instance of a Vue.js component, you use the new keyword to call the constructor function for that component:

```
// Define our custom "my-component" component
Vue.component('my-component', {
  // Component options go here...
})

// Create an instance of our custom "my-component" component
const myComponentInstance = new Vue({
  el: '#app',
})
```

In this example, we've defined a new my-component component using the Vue.component() method. We then created an instance of this component by calling its constructor with some configuration options and providing it with an element (#app) where it should be mounted on the page.

Once you have created an instance of your Vue.js components, they become reactive just like any other part of your app's state. This means that when their data changes, they will automatically update their template and re-render themselves as needed.

You can also communicate between instances in different ways such as props drilling or Vuex store depending on what suits best for your specific use case.