Q: When and why do we need lazy()?

A:. When the application size is very large so that it includes hundreds and thousands of components then it is not a good idea to bundle all the components together in a single bundle. This can result in making the application very slow and hence affects the performance. So, to create large scale production level application, we need to do chunking that is creating more than one bundle and then dynamically loading them at runtime as per the usecase. This dynamic loading is done with lazy() which is used to defer loading a component's code until its rendered for the first time. lazy() takes a callback as an argument which returns a dynamic import i.e, a promise which is resolved with default export when the bundle is loaded.

```
const Instamart = lazy(()=> import(“./components/Instamart.js”));
```

Q: What is a suspense?

A: A Suspense is used to show a fallback component until its children have finished loading. It is used to handle asynchrounous rendering. The Suspense component works by “suspending” the rendering of a component tree until all the data it needs is available.During this time, React displays a fallback which is used to indicate to user that the content is being fetched. Once the data is available, react resumes rendering the original component tree.

```
<Suspense>
{children}           //This shows nothing on the app in place of lazy component  until the promise is resolved.
</Suspense>
```

```
<Suspense fallback={<Shimmer />}>
{children}         //This shows Shimmer component unil the promise resolved and bundle is loaded.
</Suspense>
```

Q: Why we got this error: *A component suspended while responding to synchronous input. This will cause the UI to be replaced with a loading indicator. To fix, updates that suspend should be wrapped with startTransition.*
How to fix this error?

A: We got this error because as we are waiting for the bundle to be dynamically imported i.e, for the promise to load the bundle to be resolved but since the bundle is not loaded React suspends the rendering operation and throws this exception because it can cause the UI to freeze or become unresponsive.
To fix this issue, we can wrap the lazy components in Suspense boundary which will show a fallback component until the bundle has finished loading and the lazy component is first rendered. We can also use the startTransition API which will show the old UI until the new UI is ready to be loaded.

Q: Advantages and Disadvantages of using code-splitting pattern.

A:

- **Advantages:**
  1. **Faster loading time:** Code splitting reduces the initial load time of an application, making it load faster and improving user experience.
  2. **Performance improved:** Smaller chunks of code can be loaded faster, reducing the time it takes to respond to user interaction
  3. **Improved Caching:** It allows for better caching of code and small chunks can be easily cached and reused

4. **Better Resouce management:** As only the code that is needed for a particular usecase is loaded, it reduces the overall resource requirement of the application.

- **Disadvantages:**
    1. **Increased Complexity:** Code-spliting can add complexity to the codebase. Developers need to carefully manage the split points, dependencies, asynchronous loading which can be time consuming and error prone.
    2. **Additional network requests:** Chunking causes loading many bundles which increases the network requests. Depending on the size and number of chunks, this can have impact on the performance of the application.
    3. **Compatibility Issues:** Code-splitting features like import() may not be compatible with older browsers.

Q: When and why do we need suspense?
A: Suspense is used in case of asynchronous rendering when we are expecting for some data to show but the data is not available yet. So Suspense shows a fallback until the data is available. It tells user that data is being fetched. Once, the data is available then it shows the content.