Q: useContext vs Redux.
A: Context API:

1. Built-in React. Don't need to install a new library so it does not increase build size.
2. Context API creates different context for different data.
3. Hard to debug and test as compared to redux. Tracking issues in code becomes difficult in case of large codebase.
4. Configuration is easier as compared to redux.
5. Learning is easy.
6. Mostly recommended for small and simple applications.

Redux:

1. Redux is a state management library and needs to be installed separately. It increases bundle size.
2. There is only one Redux store that stores the global state.
3. It is predictable and testable. We can easily understand why, when, what and how the state updation takes place. Redux DevTool is a powerful extension that helps in data flow and debugging.
4. Configuring is relatively difficult as compared to Context API.
5. High learning curve.
6. Mostly recommended for large state that is shared across application, for complex state logic, constantly updating state.
7. It gives structure to the data layer and is more maintainable than Context API.

Q: Advantages of using RTK over Redux:
A:

1. Better learning curve than redux.
2. Less boilerplate code.
3. Do not need many packages to get redux to do anything.
4. Configuring store has become easier.
5. Improves developer experience.
6. createSlice and createReducer allows us to write mutable state updates by using Immer which converts it to immutable state.
7. Auto-genrated action and action creators.

Q: Explain Dispatcher.
A: Dispatcher is used to send the action(event) to the store so that reducer can update the state. Think of it as triggering an event when something happens. It lets the store know that something has happened.
In Redux we use store.dispatch() to dispatch the action but in RTK we use useDispatch() hook which returns us the original dispatch function.

Q: Explain Reducer.
A: Reducer function takes that state and action as an argument and performs logic to update the state. Think of it as an event listener that executes the state updation logic when an event(action) happens. State updates in reducer should be immutable and there should not be any side-effects in it.

Q: Explain slice.
A: Slice is a collection of reducers and actions for a particular feature of your application. Its like a part of state that is only for a single feature. Ex: cart.

Q: Explain selector.
A: Selector lets us extract a piece of information from store state value.
In Redux, selectorFunc(store.getState()) is used to extract information. In RTK, useSelector() hook is used to subscribe to the redux store and read the state value. If the retrieved state value is different from previous state value then this hook triggers a component re-render.

Q: Explain createSlice and the configuration it takes.
A: It is used to create a slice of the redux store and it automatically generates action type strings, action creator functions and action objects. It lets us write mutable state updation logic to reducers with the help of Immer. It also automatically give the action creators the same name as reducer functions.

Configuration it takes are slice name, initialState and reducers.
```
createSlice({
name:”cart”,
initialState:{
items:[]
},
reducers: {
        addItem: addItemReducer
}
});
```