useReducer Hook

# What is useReducer hook?

* A more advanced and complex way of **MANAGING state**
* Works with a **REDUCER** function
* Usually we use reducer when there is some **COMPLEX**  state to manage

## Syntax

* Const [count, dispatch] = useReducer(reducer, initialValue)
  + Count – initial State
  + Dispatch – function to update the state
    - Dispatching an ACTION
  + reducer – the reducer function
  + initialValue – initialValue

### Reducer function

* function declared outside of the Component
* function reducer(currState, action){
  + return currState + action
  + }
  + takes 2 arguments
    - currentState
    - action
* The **ACTION** param will be passed as an object that has the action and the value
  + **OBJECT -** {type: ‘’, payload:’’}
* A screen shot of a computer code

  Description automatically generated

## Dispatching an action

A screen shot of a computer program

Description automatically generated

## The big advanced of useReducer is that we are managinig all states inside a singe function

A screen shot of a computer program

Description automatically generated- and the only thing we are doing is to **DISPATCH Actions**

## Advantages of useReducer

1. Alternative way of setting and managing state
   1. Used for complex state and related pieces of state

A screen shot of a computer code

Description automatically generated

1. Stores related pieces of state in a **SINGLE state** object
2. useReducer uses the **Reducer function**
   1. decouples the STATE LOGIC from the component
   2. contains all the state updates
3. Reducer is a **PURE** function
   1. State is **Immutable** so we are using the same {…state} operator
4. Dispatch
   1. We use the **DISPATCH** to trigger a state update instead of setState

# Fake API with a Fake Server

* Install the json-server
* Run it with
  + Globally – json-server –watch ‘file’
  + Project only – npx json-server – -watch ‘file’
* Add the following script to the package,.json
* "server": "json-server --watch public/questions.json"
  + We can run it before we start the application

# useState vs useReducer

## useState

* Ideal for single pieces of states that are independent of each other
  + Numbers, strings, arrays, objects
* Logic to update the state is in Event Handlers or Effects
  + Spread all over one or multiple components

## useReducer

* Ideal when we have multiple states that are related to each other and complex states
  + Objects with many values and nested arrays
* Logic is centralized into the reducer function and decoupled from the components
* State is updated using the dispatcher to dispatch actions

A diagram of a flowchart

Description automatically generated

# useState should be top choice for State but useReducer must be used when the state is complex