React Questions

1. What is React JS and what problems does it try and solve?

React JS is an unopinionated JavaScript library for creating User Interfaces. React solves a problem of high data load in modern web applications by making use of a virtual DOM and only rendering the changes in state. React gives us a methodology where we can create modular and succinct applications very fast and develop extremely scalable apps that can be built upon in a modular way.

1. What does it mean to *think* in react?

To think in react is to lend yourself to the modularity and component based paradigm and methodology that react offers. and to keep within the DRY principals that its nature offers. In doing so we must obey a single responsibility principal where each component has a single responsibility and adheres to that trait. If a component starts to do more than one thing we should decompose it in to separate components to modularize the task at hand. React Allows us to do such things with constructs such as Components, State and Props. these are the general principalization of the react ethos as I see it.

1. Briefly describe some of the differences between a Class/Stateful component and a Functional/Presentational component.

Class components are an encapsulated item that can hold the state of an application and have logic and methods to act upon that state. Stateful components hold the state of the application.

Functional components are used for presentation of data. They can be passed data via props and call functions and manipulate the return values of the data that gets passed to them. but they are usually thought of as dumb components (this is not a blanket term as functional components can hold business logic but that is not the intended design or best practice in a React JS application).

1. Describe state.

State is, as the name implies the storage of the applications current state. It seems like it is mutable but we deal with it in a slightly immutable way via a setter called setState. We pass it an object with state changes thus it updates the current state to be what we have told it to be.

5.Describe props.

Props are like arguments in a function. They are data items in an object. These data items can hold variables / properties and functions. They are immutable in effect since they are literally considered arguments to the component on invocation.

1. What are PropTypes used for?

PropTypes are used as a debugging aid to type check the props that are set on a component. They are a great lightweight typechecking library that is stripped away during the build phase so that they do not encumber the overall production build. They are not as robust as using a transpilation unit language such as TypeScript or Flow but they are an excelent light weight alternative when you want to use minimal setup to workflow optimization.

1. What is a lifecycle event in React?

A Lifecycle event is one that happens when a component is being created (mounted) / destroyed (Unmounted). They can also be called by code that the developer writes to perform some desired task that would be deemed useful during the lifecycle of the application. A common action may be using the componentDidMount() lifecycle event to load in data to the application.

1. What is a Higher Order Component?

A Higher Order Component is a Component that is built to take in a callback component as an argument in which it then uses to return the derived component based upon whatever logic is passed in to it. This in its simplest form could be passing in a component and returning it in a certain way.

1. What are three different ways to style components in React?

You can style a component by way of CSS, LES (Preprocessor language / construct) or a styled component (Both custom (styled-component) or module based (reactstrap)). all of the accumulated techniques can be used in custom styled components to leverage the best possible mix of technology and form.

React Routing

1. Explain the differences between client-side routing and server-side routing.

Server side routing is the older way of presenting routes and managing the data from a website or server. This is the process where all of the markup and presentational data are rendered **before** being **sent to the client**. This can be fairly slow depending upon what data is to be rendered and presented. This also takes up more bandwidth as for every new page load we are requesting the entire page render to be pushed over to the client from the server.

Client side routing is the newer and more efficient way to render smaller component sized pieces of a presentational display such as a website or web app. This method does a call to the server usually an ajax request to load in a partial piece or component to be rendered. A front end library will present the data by only rendering the parts that have changed. This also means that the data requested should only be as large as the piece of the presentational layout that needs to be re rendered. The client, usually in the case of a webpage or a webapp, this would be a web browser, is tasked with handling the rendering of the partial based upon what library or framework that may be in use.

1. What does HTTP stand for?

HTTP stands for "**HyperText Transfer Protocol"**

(This is the general term that is sometimes referred to the HTTP Protocol but this is a misuse of the acronym but is still in wide spread use as that would actually be the HyperText Transfer Protocol Protocol which is fairly nonsensical.)

1. What does CRUD stand for?

CRUD is the acronym in data and database design and methodology that stands for "**Create Read Update Delete**".

1. Which HTTP methods can be mapped to the CRUD acronym that we use when interfacing with APIs/Servers.

The relation between CRUD and the HTTP methods are as follows:

* POST => CREATE
* GET => READ
* PUT => UPDATE
* DELETE => DELETE

1. Mention 3 tools we can use to make AJAX requests

* "Postman" / "Advanced Restful Client" (the 2 are fairly Synonymous to each other).
* Axios (This is a library tool that can be installed via npm and can do ajax requests programmatically).
* The Fetch API (this is a simple to understand API that returns data to the client but unlike axios it will require an extra step such as turning data in to Json using the .Json() method).

React with Redux

1. Name 3 JavaScript Array/Object Methods that do not produce side-effects? Which method do we use to create a new object while extending the properties of another object?

* filter() - this returns **a new array** containing the output of the old array **after** a callback filtering function has been invoked. It does it in an a non mutative way.
* map() - this maps over an array and does a specific action upon each iterable item based on a callback function that is passed in as a parameter.
* find() - this returns **a new array** containing the output of the action of a boolean operation to test a condition or set of conditions using a callback function to return the items that come up as true.
* Object.assign() - this returns **a new object** and extends the properties of an object passed in as a parameter in a non-destructive manner.

1. Describe actions, reducers and the store and their role in Redux. What does each piece do? Why is the store known as a 'single source of truth' in a redux application?

* Actions are fired by a dispatcher they move data via a payload **from your application** (containers / components) **to the store**. The fired action goes to the Reducers.
* Reducers act upon the action that has been sent to them and describe the methods to evaluate and change the applications state.

They break down the actions that are fired in to smaller pieces and distribute them down the pipeline to the relevant areas giving the data manipulation / passing a cohesive feels and aid state changes in response to the actions that were fired.

* Store - this is a single source of data and helps the application separate concerns by way of centralizing the data workflow to one place. It is a single source of state that all of the containers / views can use to be a solid fact for the truth of the application state.

This is the centralized data store that will keep state and is a pivotal role in the concepts of the redux ecosystem.

1. What is the difference between Application state and Component state? When would be a good time to use one over the other?

Application State is used for data that is to be accessed by the entire application such as arrays of objects that are to be accessed by components and displayed to the user.

Object/Component State is useful if you want to separate your data concerns in to a more granular approach for instance the input field in a form component.

Application state is global. Any component, anywhere in the app can access it so long as they hook into it. Is better to use on bigger projects with many moving parts.

Component state is local. Component state can only be updated within that component and passed down to its children via props.

1. What is middleware?

Middleware is a function / module that sits between 2 endpoint functions / modules and has an entry point for data and methods to flow in to it and acts upon that data / methods then returns the changed / refactored data to the next function / module down the line like a pipeline component that we can put in the pipe to join 2 of the other components and act upon the data inline. the name has descriptive and practical meaning as the act of what it performs is that of monitoring and acting upon data flow withing the application.

Middleware gives us a way to interact with actions **after** they are dispatched **before** they reach the store's reducer.

1. Describe redux-thunk, what does it allow us to do? How does it change our action-creators?

redux-thunk deals with creation of promises so that the async calls of the methods in the application pipeline can be firing actions after the promise has been resolved and can be used in situations where we would like to do an async call to a server for instance and when the resolution of the promise is complete the actions can be performed on the payload / response data in a meaningful way.

Redux-thunk is middleware that allows us to handle asynchronous actions in Redux. It allows us to write action creators that return a function instead of an action. We can delay the dispatch of an action, or to dispatch only if a certain condition is met.

1. Which react-redux method links up our components with our redux store?

The createStore method links up components with the redux-store.