THEORY QUESTIONS ASSIGNMENT

Full Stack Stream
(Maximum Score: 100)

KEY NOTES

- This assignment is to be completed at the student's own pace and submitted before the given deadline.
- There are **8** questions in total and each question is marked on a scale 1 to 20. The maximum possible grade for this assignment is 100 points.
- Students are welcome to use any online or written resources to answer these questions.
- The answers need to be explained clearly and illustrated with relevant examples where necessary. Your examples can include code snippets, diagrams or any other evidence-based representation of your answer.

Theory questions	**Points allocated per Question**

1. What is React? (*E.g. Consider: what is it? What is the benefit of using it? What is its virtual DOM? Why would someone choose it over the standard HTML / CSS stack?)* (15 marks)

React is a free open source JavaScript library. It is used to create user interfaces quickly with UI components which can be stacked and reused to make streamlined code. The component based nature is a huge benefit, as once you have created a web element it is usable in any react project. A significant feature of React is the virtual DOM, which sits separately from the actual DOM. It keeps component tree data and constantly compares this to the actual DOM's object hierarchy, and then selectively renders nodes based on state changes in the code. React only applies the changes to the virtual DOM from one instance to the other instead of reloading the whole thing. This saves resources and increases productivity speeds without needing to drain CPU resources or battery by constantly rebuilding. It can also be used to develop hybrid apps and mobile applications for both android and IOS through react native, that makes it useful to a wide variety of programmers. As react is so versatile it is a desired framework to use

over the standard HTML/CSS stack. It can lose some functionality between IOS and android when building apps. However, its one size fits all approach makes it very popular, and its ease of scalability contributes to this also.

2. What are Props? What is the State? What is the difference between them? (10 marks)

Components are built upon JavaScript functions. Props are variables in a React component that are passed as an argument into the JS functions. They are traditionally called props but can be called any variable name. It is read only so data coming through props cannot be changed by the function itself. All React components can have one props that cannot be changed after the argument. State allows components to create and manage their own data as long as they do not pass it elsewhere.To update it you use the syntax setstate() this updates the virtual DOM and becomes re-rendered. The main differences are that components receive data externally through props, whereas state sets data internally to the component. States can't be modified externally but props can't be manipulated internally. State calling can only be done on Class components, whereas props can be used on both functional and class components. Props can be thought of as passing an argument to a function, and the state works as a local data storage of a constructor for a class.

3. What are React Hooks? How do they differ from existing lifecycle methods? (10 marks)

React Hooks allow you to use state and other React features without the need to write a new class. With hooks you can split one component into smaller functions based on what parts are related to different tasks (i.e., fetching data). Another function of Hooks is that you can also group comparable logic into a single component. They can also transfer data between components that don't have props or classes. Hooks also let you reuse stateful logic without having to change your entire component hierarchy. There are 3 stages during the lifecycle of a react component, the phases have methods with each stage. Mounting, Updating and Unmounting, with componentdIdMount, componentDidUpdate and componentWillUnmount this helps perform the life cycle operations. The most common Hooks used in React are useState which allows you to add State to a functional component and useEffect which allows you to add side effects within the component. useEffect works similarly to the lifecycle methods. The differences are that Hooks used functional based components vs lifecycle methods incorporating class based components. Hooks do not require a constructor to initialise. Hooks can also only be used in components and not regular JS functions.

4. Design the perfect door - what should it look like, what are the components for it? What design heuristics should it follow, and how does your design match? What made you choose this design? (20 marks).

1. *<u>Consider in particular (likely need to do independent learning)</u>: who are your stakeholders? What is their personas? What is the doors requirements and how will your stakeholders benefit from your solution?*

The stakeholders for this door would be the average older person, as they need functionality in their products not necessarily over complicated products. This door fits the requirements of this demographic of :

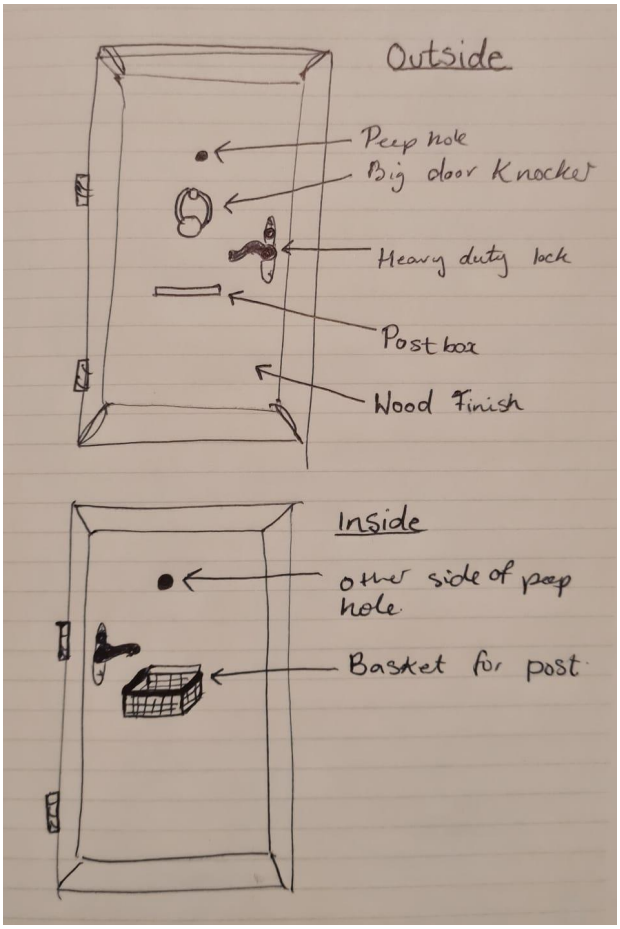
-Ease of use

-Recognisability/ familiarity

-Effortless Security/ sense of ease

-Functionality

-Efficiency



I chose this design because I thought it had a nice balance of aesthetic and the extra features making it above standard but not over complicated. Other draft ideas had extra functions including doggie doors and lights but they could be incorporated separately to the door. Some of the design heuristics I examined were security, multifunctionality and simplicity/ minimalist design.

The components for this door are all listed in the image above: Peep hole, door knocker, heavy duty lock, post box. The idea of the individual components is that stakeholders have both the opportunity to visualise the design in front of them and separate/ replace any components that they deem indesirable. It allows for clearer definition between functionality and aesthetic between the heavy duty lock and the wood finish for instance. The emphasis on heavy duty locks provides stakeholders with a sense of security along with the peep hole for seeing outside. Ease of use was also prioritised with a simple aesthetic and recognisable components such as a door knocker to ensure most users have a familiarity to use it. The post box serves as a multifunctional component for the door and reduces bending down for the stakeholders.

5. What is Angular, and how does it differ from React? *You may need to conduct independent research and learning for this* (10 marks)

Angular is a Google developed, full featured framework which provides greater insight into how your code should be designed versus React which is developed by Facebook, a javascript library that focuses on UI components. The data binding in Angular is supported in both one and two way binding. Binding means if you modify the UI input the model state will update and vice versa. In React data can only bind in one way meaning it cannot affect a component's state. The language used between the two is also different, in Angular TypeScript is used, which is statically typed, whereas in React you use JSX. The DOMs and their usage are also different between Angular and React. Angular uses an incremental DOM; this means when changes are made it compares to the previous DOM and the differences are applied. React has a virtual DOM, which it will compare to the real DOM and the differences are updated. The usage of virtual DOM is much more efficient and lightweight on React's side.

6. Please describe Redux in as much detail - especially consider: *why would someone use it? What is it? What's the benefit of using it? Are there any potential drawbacks to using it? How can it be added to a project? What is dispatch, provider, actions, etc?* (15 marks)

Redux is an open-source JavaScript library used to manage an applications state. It is used to store a whole app state into a single JavaScript object which can then be accessed from the app via the component tree. Redux acts as a global store that all components can fetch and withdraw data from, regardless of location in component hierarchy. Because of this, redux can be very useful for testing React websites by allowing us to quickly change the data React has access to. It encourages good React architecture, and provides a selection of custom react hooks that can be utilised to

locate, send and receive data with ease. Selector allows us to select the state we currently want. Dispatch is used to update a state, this is by taking the action object as an argument, and returning the new value. Provider gives access to any components it calls for to the global store, usually found at the top level. Some of the downsides to Redux is the lack of encapsulation on the data, excessive memory use, increased complexity and time consuming. Redux can be manually added to a project, or if creating a new project through the create react-app command using redux as a template.

7. Please describe Linux in as much detail as possible (feel free to use notes made during lessons, or draw from the lesson directly!). Especially consider: *what is its history? Why would someone use it over other existing operating systems? How does Windows and Mac OSX differ from Linux? How does Linux function, what are some unique features to it? How can it be installed today?* (10 marks)

Linux is an operating system, this is the main kernel for GNU. Linux is favoured for use on servers and is very popular. Linus Torvalds started developing what is known as the Linux terminal to use, and created the name Linux after himself adapted UNIX which was actually based on GNU. A lot of Window's powershell and MacOX terminals accept Linux commands. Linux is popularly used because of its end-to-end encrypted security feature.

One of the key differences between Mac/ Linux and Windows is the filing system. Mac OS and GNU/Linux have no drives with everything stored in the computer as a file, all files are organised into directories that descend from a single root directory. This structure is formed as a tree with a unique root. In Windows there are drives, usually a C or D “drive” and separates drives for external devices. The next difference is the default shell of bash compared to Windows which has separate syntax. They each also have different package managers, Mac has Homebrew, Windows doesn’t not have a default but can install one like chocolatey, and Linuxes uses different ones dependent on the distro, i.e Ubuntu uses APT. Most Linux/GNU distros are free to run which makes them cheaper than Mac with similar qualities making it a more desirable OS to have.

If you want to install Linux today,the first step is to choose the best distribution for your computer. The older distributions such as Dabian and Mint, are more popular and have a larger community could help troubleshoot installation issues. Ubuntu has gained popularity again in recent decades so is another viable option.

8. What are they, and which is better between Class components and Functional components? Provide a discussion. Consider: *Go deep - how does each one work? What is the unique properties or behaviours to each one? Why would someone use one over the other? What are the advantages and disadvantages of each one? Who benefits from these advantages and disadvantages, who is it suitable for?* (10 marks)

A Functional component is essentially a JavaScript function which accepts props as an argument and returns a React element (JSX). Class component's house multiple functions and are extended using `React.Component` and return a react element. Functional components require no rendering method and are stateless, predominantly responsible for display data provided to them and rendering most of the UI components. Class components require a rendering method () which returns a JSX which is similar to HTML, and known as stateful because they use logic and state. React lifecycle methods can be used in class components but not in Functional components. Hooks can make Functional components stateful. Advantage of Functional components is their readability, easier to test and debug. Disadvantages of Functional components are that their simplicity make them lose some of their optimising purpose. Advantages of Class components are their versatility; they can be used for a wider variety of things with more functions. Disadvantages of Class components are their complexity, in writing and how to test it, also the compiling time.

```
Functional component with Hooks example:
const [name,SetName]=
React.useState(' ')

Class components with hooks example:
constructor(props) {
   super(props);
   this.state = {name: ' '}
}
```