WEEK 2 - REPORT

This week we familiarized ourselves with the concepts of react and javascript as well as git.

React Introduction

React is a JavaScript library for building user interfaces. It is used to build single page applications and also allows us to create reusable UI components. Instead of manipulating the browser's DOM directly, React creates a virtual DOM in memory, where it does all the necessary manipulating, before making the changes in the browser DOM. React only changes what needs to be changed.

SPAs

Single Page Application that works inside a browser and does not require page reload during use. It serves outstanding User Experience - no page reloads, no extra time. It is just just one HTML page that you visit which then loads all other content using Javascript.
Eg: Gmail, Facebook, Instagram, Trello.

GIT:

We familiarized ourselves with git, especially with the basic commands like add. , init, commit, push. How to navigate from one commit to another, delete commits, un-push commits and compile different commits into a single one. We also learned the different branches and different importance levels of the same. On the same topic we also learned how to navigate between different branches and how to avoid clashes between different pushes. This has enabled us to now make our project together and coordinate with each other more efficiently with less chances of error and bugs.

Key points

Create-react-app tool: Installs all the libraries and packages required to build a React app. It creates a default application for our react project and also adds some starter files in the newly created application.

Npm : Node Package Manager for JS. It helps you manage all the third party packages and libraries that you will be installing for your application. It is installed automatically when you install Nodejs.

Npx : Node Package Runner. It is used to download and run a package temporarily. We'll need create-react-app only once to run our project.

JSX

JSX is a syntax extension for JS. It allows us to define React elements using syntax that looks very similar to HTML. It is used to define the look of UI. In other words, it is used to define the structure of React components. Unlike HTML, JSX couples the rendering logic with other UI logic such as event handling, state changing, data displaying, etc.

React Components

Components are independent and reusable bits of code. They serve the same purpose as JavaScript functions, but work in isolation and return HTML via a render() function. Components come in two types, Class components and Function components.

React components implement a render() method that takes input data and returns what to display. This example uses an XML-like syntax called JSX. Input data that is passed into the component can be accessed by render() via this.props.

React makes it painless to create interactive UIs. Design simple views for each state in your application, and React will efficiently update and render just the right components when your data changes. Declarative views makes the code more predictable and easier to debug.

React also allows us to interface with other libraries and frameworks. This example uses remarkable, an external Markdown library, to convert the <textarea>'s value in real time.

Props

Another way of handling component properties is by using Props. Props are like function arguments, and you send them into the component as attributes.Props are arguments passed into React components.Props are passed to components via HTML attributes.React Props are like function arguments in JavaScript *and* attributes in HTML.