Info packet

Basics of Web Development and React

1. General Structure of web development:
   The main way we communicate over the internet is through user-facing sites with modern design. This obfuscates the entire infrastructure of what needs to happen for information to get sent reliably and securely to where it needs to go.

   At a high level, a web application separates into front-end and back-end components that work together to deliver a seamless user experience. The front end handles the user interface, rendering pages, and responding to interactions in the browser, while the back end processes requests, applies business logic, and manages data in databases. Communication between the two happens over the network, typically using HTTP requests through APIs. These requests pass through network layers, including the transport layer (which ensures reliable delivery) and the internet layer (which routes data packets across the web), before reaching the server. Together, front end, back end, and network layers enable a smooth, interactive web experience.

   Most engineering works with obfuscation, meaning we hide the small details of the implementation to make sure that engineers on other aspects of projects don't need to care about them. For example, a front end developer doesn't need to know too much about the server implementation and network protocols.

2. Your environment today: Vite and React

   Today you will be in the role between storytellers and front end engineers, deciding on the contents of the website, changing the design elements and overall structure. This mostly means using a combination of Java Script [Type Script] for interactivity and logic, CSS for styling and HTML for structure.

   As a front end engineer, you are provided with a pre-made server/backend environment, so you don't have to care about the server implementation too much. Today, we are using Vite, which is a fast, light weight development environment that will run the local development server and bundle the website for export to the server. You will mostly need to worry about just using the React library, as Vite handles re-building the site automatically, every time you save your code.

   React is a front end development framework based on Java Script developed by Facebook (now Meta), with the main goal of producing replicable components that can be updated for their platforms. For example, a Facebook engineer can make one change on how a button looks and it will be applied all over the platform. No need to redesign all the screens from scratch.

React also has some cool features when it comes to live content, but today we will be working only on a static website, with no need to update the number of likes or comments dynamically, so if you are interested you can always research this yourself!

The fundamental unit of react is the **Component**: it's a small, functional unit that should be reused in various parts of the app.

A stereotypical example for one of these is a button: you re-use the design and functionality of the button in different places: you can add a button to a website to re-direct you, you can add a button to navigate photos, submit a form, etc. and these buttons should all have the same or similar design and interactions to ensure best practices for user accessibility and experience.

Components can also contain other components within, for example a card component can have a button within them to expand it. You can think of components as a function that can take in some information and produce a certain view using that information.

Pages, in our context, represent very large components, that are usually made of many smaller components, that represent a single web page. The page should be a unit of the website reachable by URL navigation: the extra bits at the end of a website like www.myweb.com/**this_is_a_page,** And they should be centered around themes in the website like contacts forms, history, services, about, etc…

In React, **props** (short for properties) are inputs that a parent component passes to a child component to configure or display data. They are read-only and cannot be changed by the child.
State, on the other hand, is data that a component manages internally and can change over time, usually in response to user actions or events, triggering the component to re-render. Essentially, props are for passing data into a component, while state is for storing data that a component controls itself.

React is written in something called **JSX**, which is a way of embedding HTML structure code into logical Java Script Code. This means that the component's render section, the part where you tell the browser what to display, is structured similarly to HTML using the common HTML tags and syntax. Then, Vite handles interpreting it for local preview and re-packages it for the server when needed.

- 2 main folders:
- Src
    - Contains the main part of the code
    - Split into pages and components
    - Components: the individual repeating elements that make up the site, such as buttons, paragraphs, info cards, socials images,
- Public
    - Contains the static resources that are not code, such as images, videos, fonts, etc...