React Presentation

Slide 1

Introduction

Hello everyone,

My name is Anthony Williams, and this is my CS131 Discrete Structures Project about React. This project talks about the history of building websites with JavaScript before React, what problems React attempts to solve, and a brief overview of its core features.

Slide 2

What is React?

React, also known as React JS, is an open-source front-end JavaScript library that helps software developers build user interfaces using UI components. The library has been maintained by Meta (formerly Facebook) and a group of individual companies and developers since 2013. React aims to aid in creating single-page and mobile applications. React provides the framework to manage the application state and render said state to the HTML Document Object Model or DOM for short. To understand what React is, we must first look at the landscape of web applications before its release.

Slide 3

Before React
In the 1990s, most web pages consisted of HTML for the page's text structure and CSS for the basic styling of the page.

Slide 4

In 1995, the programming language javascript was created by an engineer at Netscape to add interactivity to the static pages loaded in Netscape's Navigator browser. Shortly after, a browser war would begin with Microsoft's release of Internet Explorer in 1995. To mimic Netscape's interactivity features, Microsoft reverse-engineered the Navigator's interpreter to create their own standard called JScript.

With the rise of Internet Explorer and other browsers in the following years, different client-side scripting standards made it difficult for developers to make their websites work correctly in the various browsers.

Slide 5

JQuery, a Javascript library, was designed to simplify this headache and allow developers to interact with the DOM across all the different browser implementations. Instead of learning many different browser DOM implementations, developers would use JQuery, which would provide them with a unified API for manipulating the DOM. Jquery was the de facto solution for most developers when implementing Javascript interactivity on their websites at the time.

Slide 6

Unfortunately, this would eventually become a problem. As websites increased their functionality, adding more complex interactions, websites would begin to become full-blown applications offering users different dynamic features like news feeds, messages, and likes to name a few.

Slide 7
Enter Single Page Applications

Libraries like Backbone.js were created to give developers the ability to organize javascript files using the model-view-controller design paradigm along with web technologies like AJAX (short for Asynchronous Javascript and XML), would usher in the birth of the single page application(or SPA for short).

Slide 8

With single-page applications, less time gets spent developing the application in HTML, and more time is spent on Javascript. Instead of requesting the server to load a new page, the page is only loaded once, and the web application is updated using Javascript to repaint the browser screen as needed when the user requests new data.

Slide 9

As time went on, new products were introduced to maintain the growing complexity. Most notably was an offering from Google called Angular JS to tackle the growing problem. Initially, the community adopted Google's offering, but eventually, it became far more complex than the community would have liked.

Slide 10

In 2013, Facebook would eventually release its offering, totally reimagining how front-end applications are built and designed. Google decided to rewrite Angular after realizing its initial implementation needed an overhaul to compete. This move, along with Facebook's total reimagining, eventually caused developers to adopt React in a big way.

Slide 11

Today, some of the biggest companies in tech like Amazon, Airbnb, Netflix, Reddit, and Uber, just to name a few, use React or some form of its ideas to help manage the front-end of their applications.

Slide 12

How does React work

One of the ideas that made React so powerful and led to its widespread adoption was its implementation of a virtual DOM. DOM manipulation is a performance-heavy task because the browser has to repaint the screen and recalculate the layout. React's method to handle DOM manipulation is for the developer to declare a Javascript object telling React what the data should look like at any given time. This Javascript object would be called state, which holds the application's state. This declarative paradigm results in less complexity, faster development times, and better code quality.

Slide 13

The next idea that made React stand out was its implementation of reusable components. You can think of them as legos. The idea was to have small components that could be put together to form bigger components. For example, you could have a navigation component, profile picture component, or menu item component. Reusable components greatly increase code reuse. Components are implemented as simple functions or classes that receive input to display to the user.

Slide 14

The third and final concept that separated React from the rest of the offerings was its implementation of a one-way data flow. One-way data flow makes React applications more performant. Parent components hold the application's state, passing a snapshot of its state to the child components. The state would be an immutable object, and the snapshot sent to the child components would be read-only. In addition to more performant applications, this one-way data flow design allows for easier debugging. Suppose there is something wrong with a hypothetical sign-out button. In that case, a developer-only needs to debug the parent and child component responsible for storing the state and displaying it, respectively.

Slide 15

I want to end this presentation with a straightforward example application utilizing the concepts discussed. The application is a score counter. The application has the counter state passed to the App component to display it. An increment button will update the score when clicked and a decrement button that will subtract from the score state. A clear state button is provided to clear the application's state. You can view an example of this application at the code pen link provided in the repository. Additionally, a sample code has been provided in GitHub as well.

Thank you.