# React Presentation

By: Jack Bownman, Jack Udeschini, Paul Katigbak, Peter Regas, Kevin Romero

# Overview

“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.” - https://reactjs.org/

ReactJS is maintained by Meta and and can be used as the basis for the front end of many types of web or mobile applications

ReactJS is a open source javascript library that uses javascript to create components to help create complex UI for your website or app. It is a declarative language that only updates and renders components when data changes.

The main features of react are the Components and Hooks which will be expanded upon in the next slides.

# React Hooks, State, and Lifecycle

- One of Reactjs' biggest features is Hooks.
- What are Hooks?
  - Hooks are functions that let you use class features such as React State and Lifecycle from function components.
- What is React State and Lifecycle?
  - Reactjs Lifecycle aims to solve  static web rendering from vanilla js by continuously reloading  and rendering components.
  - Each component that needs to be constantly reloaded is  done so using React States.
  - Adding a state to a component, tells React to reload that component every life cycle.
  - In order to change static features, into dynamic features, Reactjs uses a virtual DOM, that is constantly changing and pre-rendered and once the life cycle is complete, the virtual DOM is sent to the client to turn it into the actual DOM.
- How does Hooks relate to React's state and lifecycle?
  - Up until recently, state and lifecycle were completely reserved for Classes only. This means that in order to have components that needed to be dynamically updating, they would need to be implemented as a javascript class.
  - Using classes to build components is somewhat redundant, because it takes more space to store the class and the objects, also Classes in general are more complex to program.
  - With Hooks Reactjs moves abstraction away from Classes and brings it into functions. This means that in order to build dynamic components, States no longer need to be in class format, instead they can be used in functions. By doing this, there is no longer any need to create a class and object to create a component, this also saves programmers more time because functions are less complex.
- To summarize:
  - With Hooks, Reactjs adds a greater level of abstraction to JavaScript. Before in order to have some sort of abstraction classes needed to be implemented to accomplish things such as inheritance. With hooks, inheritance is accomplished by rendering different things depending on the State.

# React as a Library vs Vue as a Framework

“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.” - https://reactjs.org/

ReactJS is a open source javascript library that uses javascript to create components to help create complex UI for your website or app.

ReactJS is maintained by Meta and and can be used as the basis for the front end of many types of web or mobile applications

Vue is a progressive lightweight framework that allows  developers to make  solutions easily and fast. Vue.js is also much easier to learn, as it takes less time and the learning curve is much lower.

The main difference is Vue uses HTML as templates as opposed to React JSX. Vue is developed by it’s core team making it’s documentation easier for beginners.

While, React has a lot of third-party libraries and dependencies that might be confusing for beginners.

# How we Implemented Reacts' features

We implemented the Hooks by creating a Wordbank Hook which would load words from an API and a File. See the next slide about what is happening.

```
useEffect(() => {
  async function fetchWords() {
    try {
      getWord().then((words) => {
        setWordSet(words.wordSet);
        console.log(words.wordSet);
        setCorrectWord(words.todaysWord);
      });
    } catch (err) {
      console.log(err);
    }
  }
  fetchWords();
}, []);
```

```
import wordBank from "../../Assets/Wordle.txt";

export const defaultBoard = [
  ["", "", "", "", ""],
  ["", "", "", "", ""],
  ["", "", "", "", ""],
  ["", "", "", "", ""],
  ["", "", "", "", ""],
  ["", "", "", "", ""],
];

export const getWord = async () => {
  var axios = require("axios");
  var config = {
    method: "get",
    url: "http://127.0.0.1:5000/dailyWord",
    headers: {},
  };

  let wordSet;
  let todaysWord;
  axios(config)
    .then(function (response) {
      const selectWord = response.data;
      todaysWord = selectWord;
    })
    .catch(function (error) {
      console.log(error);
    });
  await fetch(wordBank)
    .then((response) => response.text())
    .then((result) => {
      const wordArr = result.split("\n");
      wordSet = new Set(wordArr);
    });

  return { wordSet, todaysWord };
};
```

# Problems Encountered

- What problems were encountered?
  - What makes ReactJs more difficult to learn as compared to a framework such as Vue, is it's Lifecycle and States.
  - With Hooks, a new layer of Asynchronous behaviour is added. Where before Asynchronous behaviour was implemented through API calls. Now Asynchronous behaviour is present in both API calls and Hooks.
  - For our website, whenever we implemented a Hook on a function that dependent on an API call, not only did we have to synchronize the API call but also Synchronize the function.
  - This also would give issues because we Reactjs tries to pre-render Hooks, but nothing can be pre-rendered if the API calls have not finished, and the API Calls can't be made if the function is not yet called. So, if not synchronized correctly this will cause Reactjs to crash.
- How those problems were solved?
  - Thankfully ReactJS implements a useEffect() loads functions before their features are needed, but with API calls inside of Hooks, we needed to add an Async Function as a placeholder inside of use Effect to wait for everything inside the function to be loaded, in order to wait to read the state of the hook. On top of this we need to tell Reactjs to only do this once the page in loaded, or changed so it doesn't do this everytime the Life Cycle restarts.
  - This is also why ReactJS is difficult to learn.

# Final Word

ReactJS is a great Front-End Library, because it’s not a framework it allows faster performance through working around Vanilla JS. Because it’s Vanilla JS, more people can add libraries and extra features to ReactJS and not have to worry about another framework component. With React Lifecycle, State, and Hooks we can Implement another level of abstraction through functions.


# THE END.