**Module 2**

1. **React.js**
   1. Definition: React is a declarative, efficient, and flexible Javascript library. Uses small and isolated pieces of code called “components”
      1. A component takes in parameters called “props” (properties) and returns a hierarchy of views to display via the render method.
   2. Function: Building complex user interfaces (U.I.)
2. **Flux/Redux**
   1. Definition:Flux is a pattern and Redux is a library.
      1. Flux is a fancy name for the observer pattern modified a little bit to fit React
      2. Both have actions but Redux actions can have functions and promises (im so confused)
   2. Function:Helps you write applications that behave consistently, run in different environments (client, server, and native), and are easy to test.
3. **Git**
   1. Definition:The most widely used modern version control system. It is an example of a DVCS(distributed version control system). Multiple people can work on the same code at once.
   2. Function:Lets you commit your work locally, then sync your copy of the repository with the copy on the server. Works offline!
      1. Better than centralized version control - clients must sync code with server before creating new versions of the code
4. **Babel/Webpack/NPM/Yarn**
   1. Definition:
      1. NPM (node package manager) - a package manager for Node based environments. Keeps tracks of all the packages and their versions and allows the developer to easily update and remove dependencies.
      2. Yarn - a package manager that uses NPM registry as its backend. Very fast. When installing dependencies for a project, NPM installs packages sequentially-slows down performance. Yarn does it in parallel.
      3. Babel - a JS transplier the converts new JS code into old ones. One can add presets such as es2015, es2016, es2017 and Babel compiles them to ES5 (ECMAScript).
      4. Webpack - a modular building tool that has two sets of functionality - Loaders and Plugins.
5. **DOM**
   1. Definition: (Document Object Model) is a programming interface for HTML and XML documents. It represents the document as nodes and objects. A web page is a document. This document can either be displayed in the browser window os as the HTML source.
   2. Function: Allows for the document to be manipulated.
6. **OOP**
   1. Definition: (Oriented Object Programming) - a programming language model organized around objects rather than “actions” and data rather than logic.
   2. Function: Allows developers to create relationships between one object and another - objects can inherit characteristics from other objects.
7. **Handlebars.js**
   1. Definition: A templating engine based on the Mustache template language. Take any HTML and Handlebars expression and compiles them to a Javascript function.
   2. Function: Separate the generation of HTML from the rest of your Javascript and write cleaner code.
8. **AngularJS**
   1. Definition: A structural framework for dynamic web apps. Happens on the browser.
   2. Function: Lets you use HTML as your template language and lets you extend HTML’s syntax to express your application’s components clearly and succinctly.
9. **Python**
   1. Definition: An object oriented programming language with integrated dynamic semantics that offers dynamic typing and dynamic binding options.
   2. Function: For web and app development.
10. **WordPress**
    1. Definition: An online, open source website creation tool written in PHP. One of the most powerful blogging and website content management system (CMS).
    2. Function: Used to create websites - e commerce, business, portfolio websites. etc.