Shri Vaishnav Vidyapeeth Vishwavidyalaya, Indore

Shri Vaishnav Institute of Information Technology

Subject: INTERNSHIP/PROJECT

Enrollment Number: 19100BTIT06541

Course Code: BTCS801

IV-Year/VII-Semester – IT Section (A)

Submitted By: -

Anushka Shah

Submitted To: -

Prof. Rashmi Chauhan

Assignment 1:

Q1. What is Javascript and some basic features of javascript that you learned during week 1?

Ans - JavaScript is a high-level, dynamic, and interpreted programming language that is commonly used to add interactive elements to web pages. Some basic features of JavaScript include:

1. Variables and data types: JavaScript allows developers to create variables and assign values to them. There are several data types in JavaScript, including strings, numbers, booleans, null, and undefined.
2. Functions: JavaScript allows developers to create reusable blocks of code called functions. Functions can be called with different arguments to perform specific tasks.
3. Conditional statements: JavaScript includes conditional statements like if/else statements, which allow developers to execute code based on certain conditions.
4. Loops: JavaScript includes different types of loops, such as for loops and while loops, which allow developers to execute a block of code repeatedly.
5. Objects and arrays: JavaScript supports objects and arrays, which are data structures that allow developers to store and manipulate collections of data.
6. Events and event handling: JavaScript allows developers to respond to user interactions with a web page, such as clicks or key presses, by writing event handlers that execute specific code in response to those events.

Overall, JavaScript is a versatile language that can be used for a wide range of tasks, from adding interactivity to web pages to building complex web applications.

Q2. What are some of the basic concepts of React.js that you learned during week 1?

Ans – During week 1, some of the basic concepts of React.js that I have learned include:

React JS is a popular JavaScript library for building user interfaces. Some of the basic concepts of React JS include:

1. Components: React JS is a component-based library, which means that it allows developers to create reusable UI components. Components can be thought of as building blocks for a web application's user interface.
2. JSX: React JS uses JSX, which is a syntax extension for JavaScript that allows developers to write HTML-like code within JavaScript. This makes it easier to write and maintain UI code.
3. Virtual DOM: React JS uses a virtual DOM, which is an in-memory representation of the actual DOM. This allows React to update the UI efficiently by only updating the parts of the DOM that have changed.
4. State and props: React JS allows components to have state and props. State represents the internal state of a component, while props represent the data that is passed to a component from its parent component.
5. Lifecycle methods: React JS provides lifecycle methods that allow developers to control what happens when a component is mounted, updated, or unmounted.
6. Event handling: React JS allows developers to handle events like user input or browser events using event handlers.

Overall, these concepts form the foundation of building a React JS application. By understanding them, developers can create powerful and maintainable user interfaces.

Q3. What are some of the advanced concepts of React.js that you learned during week 1?

Ans - During week 1 of learning React.js, some of the advanced concepts I have learned include:
React JS has many advanced concepts that allow developers to build complex and high-performing web applications. Some of these concepts include:

1. Higher-Order Components (HOCs): HOCs are a pattern in React JS that allows developers to create reusable logic that can be shared across multiple components. This makes it easier to create more modular and maintainable code.
2. Context: Context is a feature in React JS that allows data to be passed down the component tree without having to pass props through every level. This can be useful for data that is used by many components throughout an application.
3. Hooks: Hooks are a new feature in React JS that allow developers to use state and other React features in functional components. This simplifies the code and allows for more flexibility in component design.
4. Server-side rendering: React JS can be used to render components on the server, which can improve the initial load time of a web application and improve search engine optimization.
5. Redux: Redux is a popular library that can be used with React JS to manage state in a more structured and predictable way. It allows for a central store of data that can be accessed by any component in the application.
6. Performance optimization: React JS provides many tools for optimizing the performance of web applications, such as code splitting, lazy loading, and memoization. These techniques can help improve the speed and responsiveness of a web application.

Overall, these advanced concepts in React JS allow developers to build more powerful and maintainable web applications. By mastering these concepts, developers can create high-performing and scalable web applications.

Q4. What I learned while creating mixbook clone during week 2?

Ans- This Clone is static and it is a single page web application that includes:

Landing Page- It shows photos from different categories on hover such as calendar, family, travel, love and etc.

Services Page- It shows what you can create using this website that is either Photobooks, Cards, Calendar.

Portfolio- It shows the community of mixbookers who contributed on mixbook website.

Footer and Contact us- It shows all the products, programs, location and also the contact information.

Q5. What specific pages or features did you work on related to task management system during week 3?

Ans - Some specific pages or features that I have Worked on task management system during week 2 could include:

Dashboard: Showing the overview of the task management system that includes list of all tasks and also board view of tasks.

Tasks Page: Page that adds the tasks, shows all the tasks and the status of tasks.

Board View: Page that shows all the tasks divided by the category that is if completed, reviewed or approved by the manager.

Chat Option: Manager and Employee can communicate through this functionality

Additional Features: Also there are some additional features to add,edit, delete the tasks.

Q6. What are the basic concepts of Node.js that I learned during week 4?

Ans - Node.js is a popular runtime environment for executing JavaScript code outside of a web browser. Some of the basic concepts of Node.js include:

1. Asynchronous programming: Node.js is designed to handle asynchronous programming, which means that it can perform multiple operations at the same time without blocking the execution of other operations.
2. Event-driven architecture: Node.js uses an event-driven architecture, which means that it can respond to events such as requests, file I/O, and timers. This allows Node.js to handle large numbers of concurrent connections.
3. Modules: Node.js allows developers to use modules, which are packages of code that can be reused across different applications. Node.js has a large ecosystem of modules that can be easily installed and used in a project.
4. The Node Package Manager (npm): npm is a package manager for Node.js that allows developers to easily manage and install modules. npm has a large repository of open-source modules that can be used in a project.
5. The CommonJS module system: Node.js uses the CommonJS module system, which is a standard for organizing and sharing code between different JavaScript files. This allows developers to create reusable code that can be easily shared between different parts of an application.
6. The file system module: Node.js includes a file system module that allows developers to interact with the file system on the server-side. This can be useful for tasks such as reading and writing files, creating directories, and more.

Overall, these concepts form the foundation of building a Node.js application. By understanding them, developers can create powerful and scalable server-side applications