**Documentation: Key Terms and Concepts in Vue.js Code**

**1. Vue Component**

* **Definition:** A Vue component is a reusable piece of UI with its own logic and view template. Components are fundamental building blocks in Vue.js applications.
* **Usage:** Components encapsulate parts of the user interface, such as NavigationBar, SpotProfile, and BookingDetails.

**2. Axios**

* **Definition:** Axios is a popular JavaScript library used for making HTTP requests from the browser. It supports features like request and response interception, and has built-in support for handling JSON data.
* **Usage:** Axios is used extensively in Vue components to fetch data from APIs, as seen in fetchSpot() and bookSpot() methods.

**3. Computed Properties**

* **Definition:** Computed properties in Vue.js are reactive properties that are recalculated automatically whenever their dependencies change. They are used for deriving data based on other data properties.
* **Usage:** calculateTotalCost is a computed property used to dynamically calculate the total booking cost based on checkInDate, checkOutDate, and spot.pricePerNight.

**4. Vue Router**

* **Definition:** Vue Router is the official router for Vue.js applications. It allows you to define multiple routes, map them to Vue components, and handle navigation within your single-page application (SPA).
* **Usage:** this.$route and this.$router are used to access route information and perform programmatic navigation (this.$router.push()) respectively.

**5. Local Storage**

* **Definition:** Local Storage is a browser API that allows you to store key-value pairs persistently in the browser. Data stored in local storage remains even after the browser window is closed.
* **Usage:** localStorage.getItem('user') is used to retrieve user data stored locally, such as userId, username, email, etc., in fetchUserData() method.

**6. Error Handling with Axios**

* **Definition:** Axios provides mechanisms for handling errors that occur during HTTP requests. Errors can include network failures, server-side errors (e.g., 400 Bad Request), and validation errors.
* **Usage:** Errors are handled in catch blocks after Axios requests, such as in fetchSpot() and bookSpot() methods, where console messages or further actions (e.g., displaying validation errors) are implemented.

**7. Vue Lifecycle Hooks**

* **Definition:** Vue.js components have a lifecycle managed by Vue instance, providing methods that allow you to hook into specific stages of a component's lifecycle (e.g., creation, mounting, updating, destruction).
* **Usage:** created() hook is used to fetch initial data (fetchSpot() and fetchUserData()) and initialize component state (initializeDates()).