**ITEW-6**

**CASE STUDY (PRELIM)**

**GROUP 6**

**Members:**

Pitogo, Richard

Pliego, John Dale

Retardo, Mark Ivan Shane

Rulona, Reinheart

Sanchez, Neil Renzo

Tailon, Ralph Joshua

Villa, Mark Lester

Viray, John Michael

**Case Study: Building a Task Management Application with Vue.js**

**Background:**

You are a junior developer working for a startup company that specializes in project management solutions. Your team has decided to build a new task management application using Vue.js framework due to its flexibility, performance, and ease of use. Your task is to develop the front-end interface of the application.

**Scenario:**

The task management application will allow users to create, update, delete, and prioritize tasks. Users should be able to view their tasks in a list format and mark them as completed. Additionally, users should have the ability to navigate between different views of the application using Vue Router.

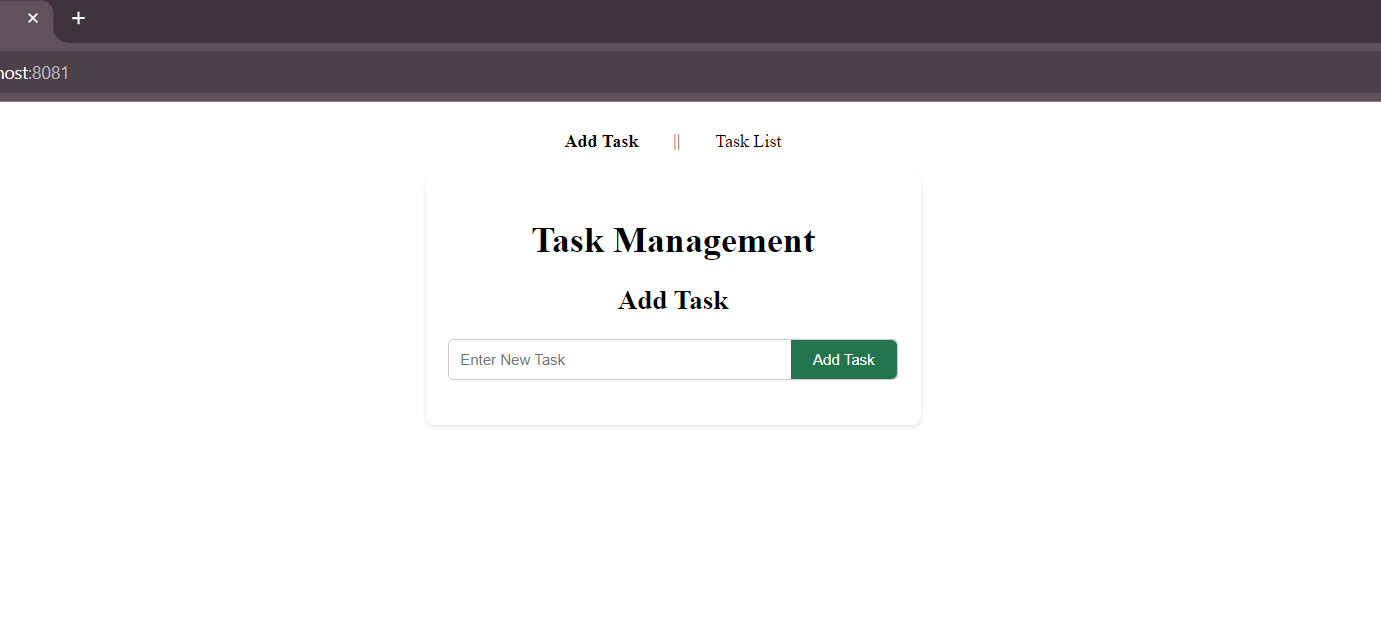
**Your program should apply the following:**

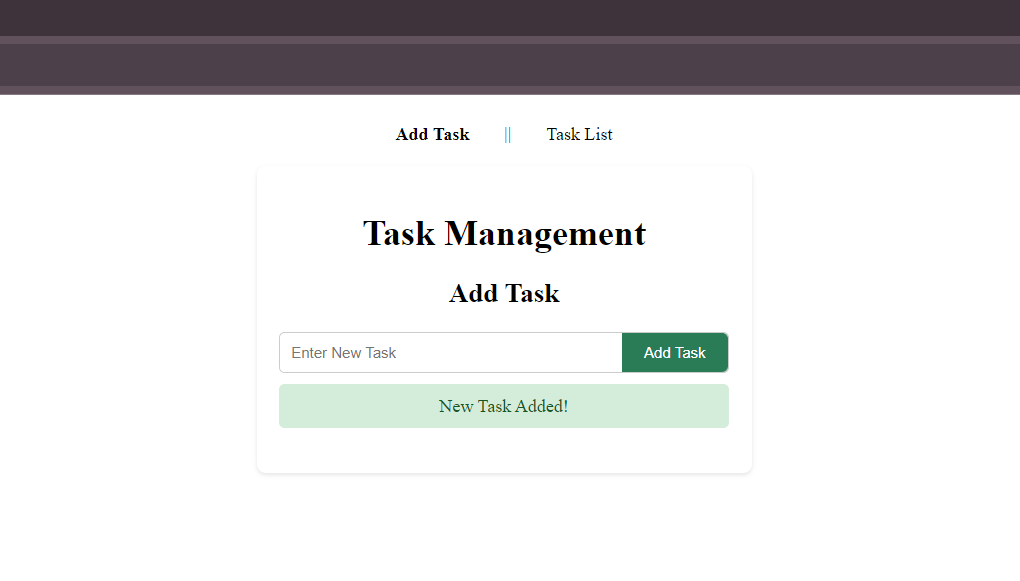
1. **Vue Instances and Lifecycle:**
   * Create a Vue instance for the application.
   * Familiarize yourself with the Vue instance lifecycle hooks and understand when each hook is called.
2. **Data Binding and Interpolation:**
   * Implement data binding to display the list of tasks fetched.
   * Utilize interpolation to dynamically display task details such as title, description, and status.
3. **Methods and Computed Properties:**
   * Implement methods to handle task manipulation actions such as adding, updating, and deleting tasks.
   * Utilize computed properties to calculate derived data, such as the total number of tasks and the number of completed tasks.
4. **Event Handling in Vue.js:**
   * Implement event handling to respond to user interactions, such as clicking on a task to mark it as completed or deleting a task.
   * Handle user input events for adding or editing tasks.
5. **Vue Directives:**
   * Utilize Vue directives such as v-if, v-for, and v-bind to conditionally render elements, iterate over tasks, and bind attributes respectively.
6. **Creating and Using Components:**
   * Create reusable components for displaying individual tasks and the task list.
   * Utilize components to modularize the application and improve maintainability.
7. **Communication Between Components:**
   * Implement communication between components using props, events, and slots.
   * Pass data between parent and child components to display task details and handle user interactions.
8. **Global vs. Local Registration of Components:**
   * Register components globally for commonly used components such as the task list.
   * Register components locally within parent components for components with specific functionality.
9. **Vue Router:**
   * Configure Vue Router to define routes for different views of the application, such as the task list view and task details view.
   * Implement navigation between routes using router links.

**GitHub Link:**

[https://github.com/Astraeus2204/casestudy.git](https://github.com/Astraeus2204/casestudy.git" \t "_blank)

**Sample Running Outputs/Screenshot:**



****

