Laravel Livewire 3 Developer Assignment

This assignment is designed to assess your proficiency in Laravel, Livewire 3, and your ability to build a web application from scratch. Please read all instructions carefully before you begin.

Project Overview: Simple Task Management Application

You are required to build a simple, single-page task management application. This application should allow users to manage their tasks efficiently.

Core Functionality:

1. **Task Listing**: Display all tasks in a clear, sortable, and filterable list.
2. **Add New Task**: Allow users to add new tasks with a title and an optional description.
3. **Edit Task**: Enable users to modify existing task details.
4. **Mark as Complete/Incomplete**: Provide a way to toggle the completion status of a task.
5. **Delete Task**: Allow users to remove tasks.
6. **Search/Filter**: Implement a search functionality by task title/description and a filter by completion status (All, Completed, Incomplete).
7. **Sorting**: Tasks should be sortable by creation date (newest/oldest first).

Technical Requirements:

- **Framework**: Laravel (latest stable version).
- **Frontend**: Livewire 3 for all interactive components.
- **JavaScript**: Alpine.js for any necessary client-side interactivity that Livewire doesn't directly handle (e.g., toggling modals, simple animations).
- **Styling**: Tailwind CSS for all styling. Ensure the application is responsive and visually appealing.
- **Database**: SQLite (for simplicity, no need for a full MySQL setup, but ensure migrations are properly defined).
- **Version Control**: The project must be initialized as a Git repository.

Specific Implementation Details:

- **Livewire Components**: Break down the application into logical Livewire components (e.g., `TaskList`, `CreateTaskForm`, `EditTaskModal`).
- **Real-time Updates**: Leverage Livewire's reactivity for immediate UI updates without full page reloads.
- **Validation**: Implement basic form validation for task creation and editing.

- **Error Handling**: Gracefully handle any potential errors and display user-friendly messages.
- **Loading States**: Show loading indicators for Livewire actions that might take a moment.
- **No Page Reloads**: The entire task management experience (CRUD, filtering, sorting) should happen without full page reloads.

Advanced Requirements (Bonus):

To further assess the candidate's capabilities, consider implementing the following:

1. **Many-to-Many Relationship**: Implement a "Tags" feature. A task can have multiple tags, and a tag can be associated with multiple tasks. Users should be able to:
    - Add/remove tags when creating/editing a task.
    - Filter tasks by tags.
2. **Queues**: Use Laravel Queues for a background process. For example, when a task is marked as complete, dispatch a job to a queue that simulates a notification or logging activity (e.g., waiting for a few seconds before updating a `completed_at` timestamp or sending an email).
3. **Data Import**: Implement a feature to import tasks from a CSV or Excel file using the Laravel Excel package.
    - Provide a simple upload form.
    - Display progress or status of the import (can be basic, e.g., "Importing..." then "Import Complete").
    - Handle basic validation for the imported file (e.g., correct columns).

Submission Guidelines:

1. **Repository**: Host your project on a public Git repository (GitHub, GitLab, Bitbucket).
2. **README.md**: Include a `README.md` file at the root of your repository with:
    - Instructions on how to set up and run the project locally.
    - Any assumptions made or design decisions taken.
    - A brief explanation of how you structured your Livewire components.
    - Details on how to test the advanced features (e.g., how to trigger a queue job, how to import data).
3. **Code Quality**: Ensure your code is clean, well-organized, and follows Laravel/PHP best practices. Add comments where necessary to explain complex logic.
4. **Commit History**: Maintain a clear and meaningful Git commit history.

Evaluation Criteria:

Your assignment will be evaluated based on the following:

- **Functionality (40%)**: Does the application meet all the core and technical requirements? Are all features working as expected?
- **Livewire 3 Proficiency (25%)**: Effective and idiomatic use of Livewire 3 features (components, properties, actions, events, validation, loading states).
- **Code Quality & Best Practices (20%)**: Clean code, proper Laravel conventions, maintainability, and readability.
- **Frontend Implementation (10%)**: Responsiveness, visual appeal, and effective use of Tailwind CSS and Alpine.js.
- **Setup & Documentation (5%)**: Ease of setup, clear `README.md`, and good Git commit history.

Good luck! We look forward to reviewing your submission.