Senior Laravel Recruitment Assessment

Laravel Framework Assessment Overview

This assessment evaluates your understanding of PHP design principles and their application within the Laravel framework. The tasks are structured across three levels, each building upon the knowledge and skills demonstrated in the previous one.

Assessment Duration
You have 3-7 days to complete this assessment. For any questions, please contact us.

Submission Instructions
Submit your work via a link to your GitHub Repository. Write proper instructions in your readme file, so that we can get it up & running following your instructions. Add necessary migrations & seedings for that.

Level 1: Basic User Management and Authentication

Goals:

- Implement Laravel's built-in authentication.
- Create basic CRUD (Create, Read, Update, Delete) functionalities for users.

Instructions:

1. Initialize a new project using latest Laravel version.
2. Utilize Laravel's default login feature.
3. Create pages for listing all users, displaying a single user, creating new users, editing existing users, and deleting users.
4. Implement soft delete functionality and create pages to list, restore, or permanently delete soft-deleted users.

Notes:

- All user routes should use the 'auth' middleware to ensure authentication.
- User forms should enable photo uploads; ensure forms have the correct encoding type. Adding a user avatar with the uploaded photo is a plus.

- Validate user inputs where necessary.

Level 2: Advanced User Management

Goals:

- Implement the Service Pattern for user management.
- Write unit tests for your service class.
- Implement validation for user data.

Instructions:

1. Ensure all functionalities from Level 1 are complete.
2. Create a service class for handling user-related operations.
3. Define an interface for the service class with necessary user management methods.
4. Develop unit tests for your service class to cover key functionalities such as listing users, adding new users, updating existing users, soft deleting, and handling trashed users.
5. Integrate the user service into your user controller and replace direct model calls with service methods.
6. Define and apply validation rules for user data in a dedicated request class.

Notes:

- Consider additional test cases to improve coverage.
- Customize method names and validation rules according to your requirements.
- Ensure proper service binding in your application's service provider.

Level 3: Extended Features

Goals:

- Manage user addresses. One user can have multiple addresses.
- Implement an event and listener for user save operations.

Instructions:

1. Generate a new table for storing user addresses and create a corresponding Eloquent model.

2. Establish a one-to-many relationship between the User model and the new addresses model.
3. Create an event to trigger whenever a user is created or updated.
4. Develop a listener for the user-saved event, which should automatically handle saving additional user addresses.

Notes:

- The 'user_id' column in the address table should be a foreign key that references the users' table.
- Ensure your listener handles the saving of addresses appropriately and works for CRUD.

Remember, the focus of this assessment is to evaluate your practical skills with the Laravel framework and your understanding of design principles in PHP. Good luck!