Technical Evaluation | ImpactLab

We've created this set of tasks to evaluate how developers tackle real-world problems. You should only spend around 2 hours working on these tasks. The last thing we want you to do is toil away for days on end! They're designed to give us an idea of your problem-solving approaches and skillset.

Programming Challenges (PHP)	
Task 1	Develop a function that accepts an array of integers and calculates the sum of all the even values in the array.
Task 2	A number is a palindrome if it reads the same forwards and backwards, like 101, 555, or 20102. Write a function that checks if a given string has this property.
Task 3	Build a function that identifies the second-highest value in a list of integers.

Creating and Designing APIs (PHP)	

Task 1	Plan an endpoint that retrieves user profile details using a unique ID.
	
	Endpoint: GET /api/users/{user_id}
	
	Example response (JSON):
	
	{
	  "user_id": 1,
	  "name": "Test Name",
	  "email": "test_name@email.com",
	  "bio": "Backend Web Developer",
	  "url": "https://website.com",
	  "created_at": "2022-07-04T10:29:45Z"
	}

Task 2	Create an endpoint to allow users to add a new post by providing a title and body content.
	
	Endpoint: POST /api/posts
	
	Example request (JSON):
	
	{
	  "title": "Creating a Post",
	  "content": "New post body text"
	}
	
	Example response (JSON):
	
	{
	  "status": "success",
	  "message": "Post created successfully",
	  "post_id": 10000
	}

Working with Database Queries (PHP)	
Task 1	Write a SQL statement to get all users' names and email addresses from the table named "users."
Task 2	Write a query to find the average age of all users in the "users" table.
Task 3	Write a SQL command to retrieve the top five products with the highest sales from a table named "sales."

Building APIs (Laravel)	
Task 1	Draft an endpoint to return all product details from the "products" table.
Task 2	Write an API endpoint that lets a user place an order containing multiple products.
Task 3	Develop an endpoint for removing a user entry from the "users" table.

Automated Testing (Laravel)	

Task 1	Write a PHPUnit test to verify that the getAverageRating method in the Product model calculates the average score based on reviews.
	
	Partial solution:
	
	use Tests\TestCase;
	use App\Models\Product;
	use App\Models\Review;
	
	class ProductTest extends TestCase
	{
	    public function testGetAverageRating()
	    {
	        // Implementation here...
	    }
	}

Task 2	Create a Dusk test to validate the functionality of the "Add to Cart" button on the product page.
	
	Partial solution:
	
	use Laravel\Dusk\Browser;
	use Tests\DuskTestCase;
	
	class ProductTest extends DuskTestCase
	{
	    public function testAddToCartButton()
	    {
	        // Implementation here...
	    }
	}

Task 3	Write a test to confirm that attempting to access a non-existent product page triggers a 404 response.
	
	Partial solution:
	
	`use Tests\TestCase;`
	
	`class ProductTest extends TestCase`
	`{`
	`    public function testNonExistentProductPage()`
	`    {`
	`        // Implementation here...`
	`    }`
	`}`

Automating Infrastructure Setup (Laravel)	
Task 1	Demonstrate the steps to configure an environment using Laravel Forge and provision a server for the application.
Task 2	Explain how to automate deployments via a deployment script or continuous integration pipeline.
Task 3	Implement a scheduled process using Laravel's task scheduler to perform periodic maintenance, like clearing cached data or creating reports.

Database Schema and Seeding (Laravel)	
Task 1	Define a schema for a basic e-commerce application, including tables for products, orders, customers, and reviews. Use Schema::create for the table definitions.
Task 2	Write a database migration to add a column named "quantity" to the "products" table.
Task 3	Develop a database seeder to populate the "products" table with sample entries.

2 hours isn't much, so it's ok to not do everything! We want to see what you've prioritised, and a description of what hasn't been done.

You might want to consider:

- API Design Best Practices
- Error Handling
- Performance
- Documentation

Once completed please submit your solution through to us by replying to this email, linking us through to your GitHub repo, or in a ZIP Folder.

Again, thank you for your application and taking the time to work through this next stage with us,

ImpactLab