Aptech®

Unleash your potential

ELECTRIC LAB AUTOMATION DOCUMENTATION 2025

PRESENTED TO

MISS KIRAN SHAKEEL

PRESENTED BY

S:NO	STD'S ID	STD'S NAME
1.	1547835	FIZZA ZULFIQAR
2.	1548125	RABEEA QASIM
3.	1544717	MADINA RASHID

BATCH CODE: PR2-2024-01B2

TABLE OF CONTENTS

INTRODUCTION

Electronic Lab Automation is a streamlined system designed to enhance the management and testing of electronic products. Built using Laravel and PHP with a MySQL database, it facilitates seamless product handling through three key user roles: Admin, User, and Tester. Admins can add and manage products, users can view and interact with them, and testers verify functionality with approval or disapproval options. The platform also includes a comment section for feedback and collaboration, ensuring an efficient and transparent workflow for electronics testing and validation.

PROBLEM DEFINATION

Traditional electronic product management and testing is slow, error-prone, and lacks transparency. Admins face challenges in managing products, testers struggle to provide structured feedback, and users have limited involvement in the testing process. Electronic Lab Automation provides a digital platform where admins can manage products, testers can approve/disapprove and give feedback, and users can view and participate in discussions—ensuring efficiency, transparency, and accuracy in the workflow.

CUSTOMER REQUIREMENT SPECIFICATION

NON-FINANCIAL REQUIREMENTS

- *The system must store product details and*
- *testing data in a structured database.*
- *Unique 10-digit product ID and 12-digit test ID should be automatically generated.*
- *Advanced search options should be included.*
- *modular and sub-modular structure to manage different types of products and testing.*
- *Detailed remarks should be maintained covering test criteria and outcomes.*
- *The system should allow tracking of testing status.*
- *The name of the testing personnel should be recorded.*

CUSTOMER REQUIREMENT SPECIFICATION

FINANCIAL REQUIREMENTS

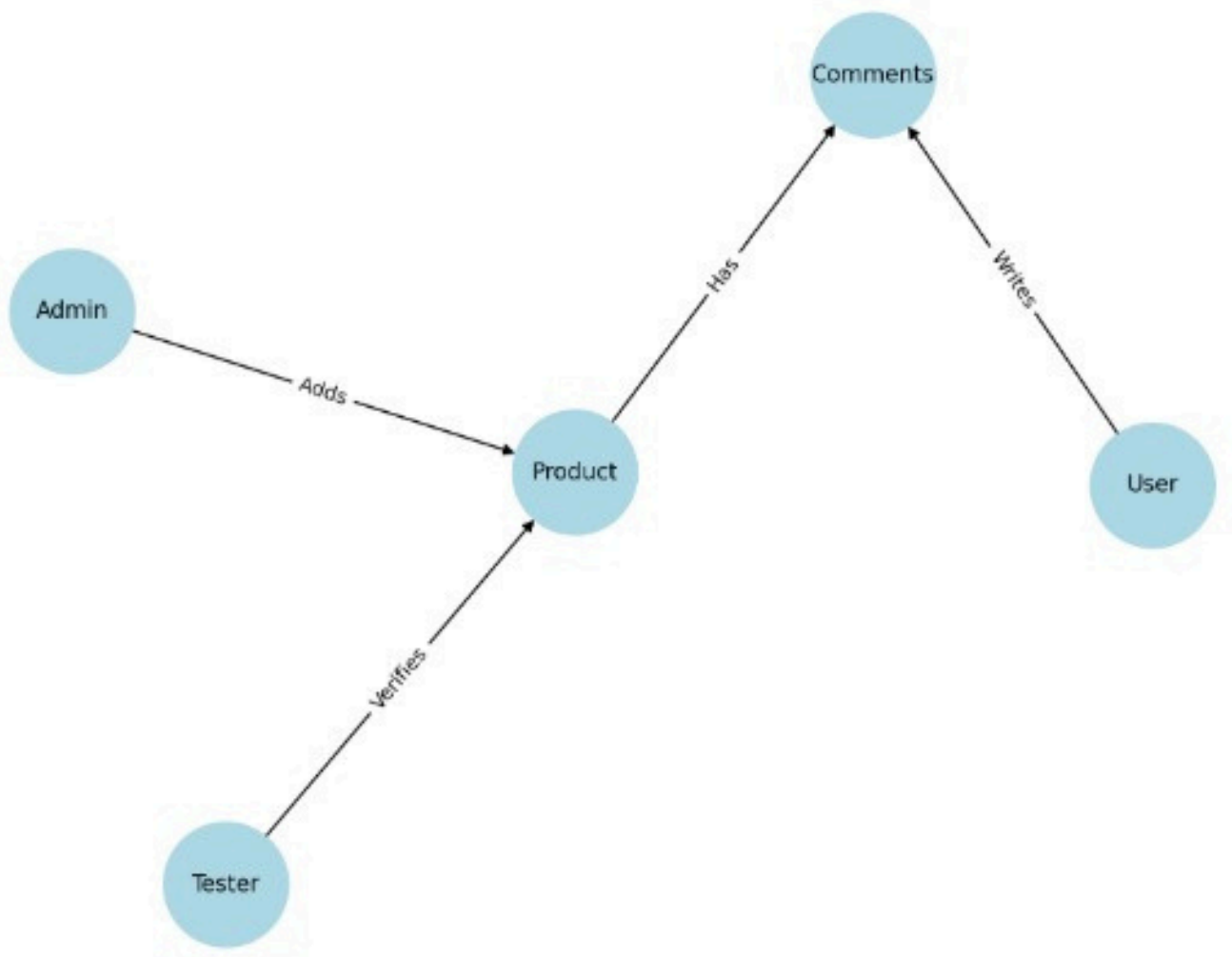
- ***Failed products should be marked for remanufacturing.***
- ***Successful products should proceed forfurther testing at CPRI for final approval***

PROJECT PLAN

PHASE	TASK	DURATION
PHASE 1	REQUIREMENT ANALYSIS	2 days
PHASE 2	DATABASE DESIGN	6 days
PHASE 3	FRONTEND & BACKEND DEVELOPEMENT	2 weeks
PHASE 4	TESTING & DEBUGGING	4 days
PHASE 5	DEPLOYEMENT	2 days

E-R DIAGRAMS

THE ENTITY-RELATIONSHIP DIAGRAM (ERD) REPRESENTS THE LOGICAL STRUCTURE OF THE DATABASE, INCLUDING ENTITIES SUCH AS PRODUCTS, TESTING RECORDS, TESTERS And test results.



ALGORITHMS

- User/Admin/Tester logs in
- Selects a product
- Validate input (ensure it's not empty)

GUI STANDARDS DOCUMENT

- **Dashboard: Displays real-time testing statistics.**
- **Product Management: Interface for adding, modifying, and deleting product entries.**
- **Testing Module: Interface for recording test results and generating reports.**
- **Search & Filters: Advanced search functionality based on product ID, test ID, and test status.**
- **Reports Section: Generates reports for tested, failed, and approved products.**

INTERFACE DESIGN DOCUMENT

1. Login Screen: Secure login for employees.
2. Dashboard: Overview of testing progress and product details.
3. Product Entry Form: Add new products and generate unique IDs.
4. Testing Form: Record test details, results, and remarks.
5. Search Page: Locate products using advanced search filters.
***6. Reports Page: View and export reports on product status*.**

UNIT TESTING CHECKLIST

1. Product ID generation is working correctly.
2. Test ID generation is accurate and unique.
3. Search functionality returns expected results.
4. Testing records stored and retrieved correctly.
5. User authentication working securely.

PROJECT REVIEW AND MONITORING REPORT

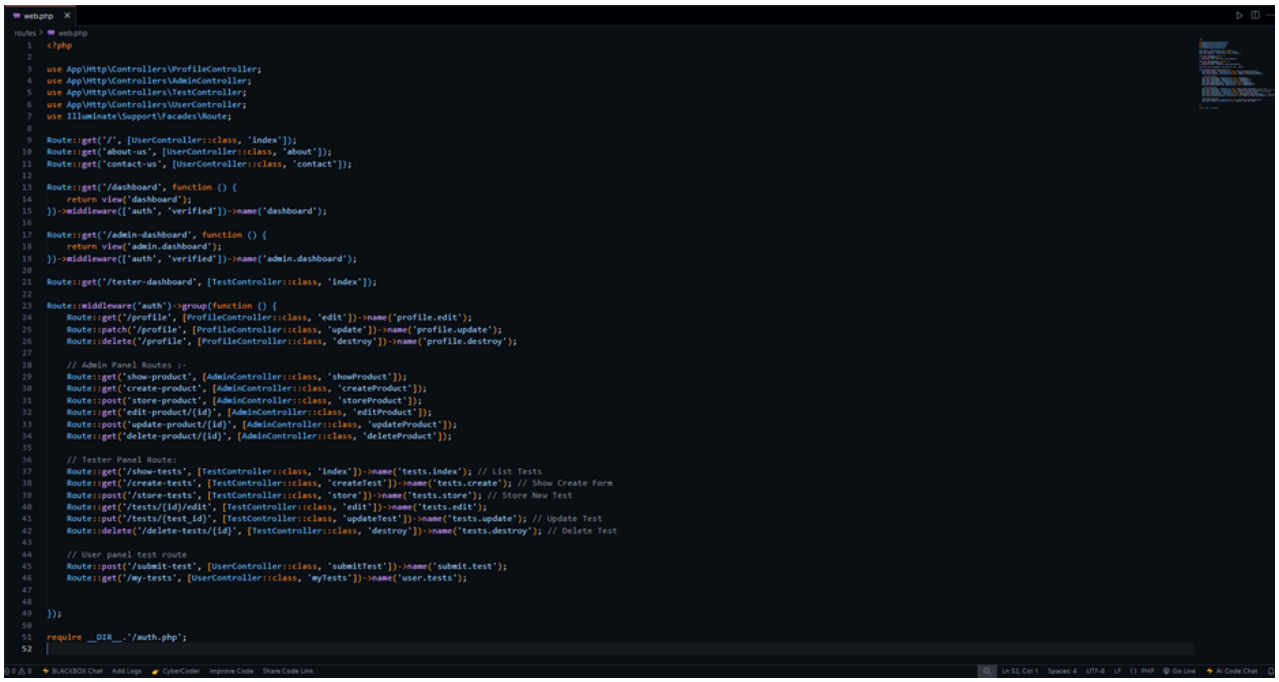
REGULAR PROJECT REVIEWS WILL BECONDUCTED TO ENSURE:

1.Alignment with customer requirements.
2.Timely completion of project phases.
3.Functional and performance testing ofmodules.

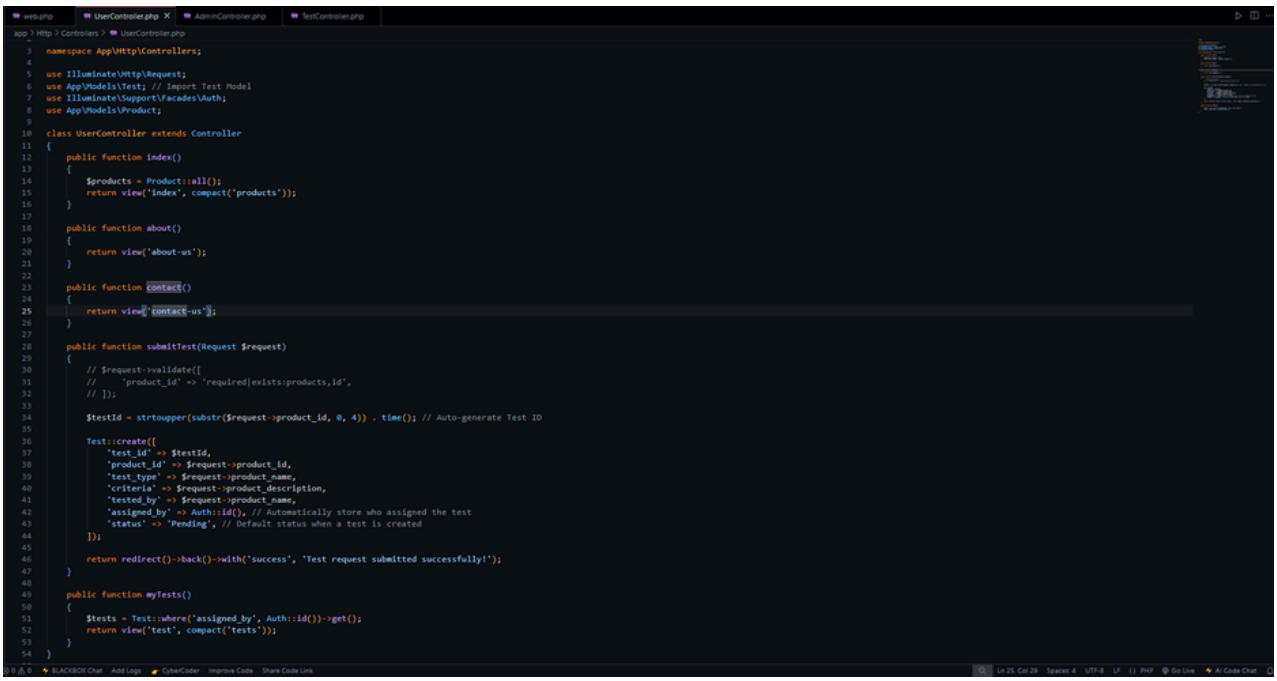
FINAL CHECKLIST

✓ *Dashboard for Admin, User, and Tester*
✓*Admins can add and manage products.*
✓*Testers can review, approve, or disapprove products.*
✓*Users, admins, and testers can comment on products for feedback.*

CODE SCREEN SHOTS

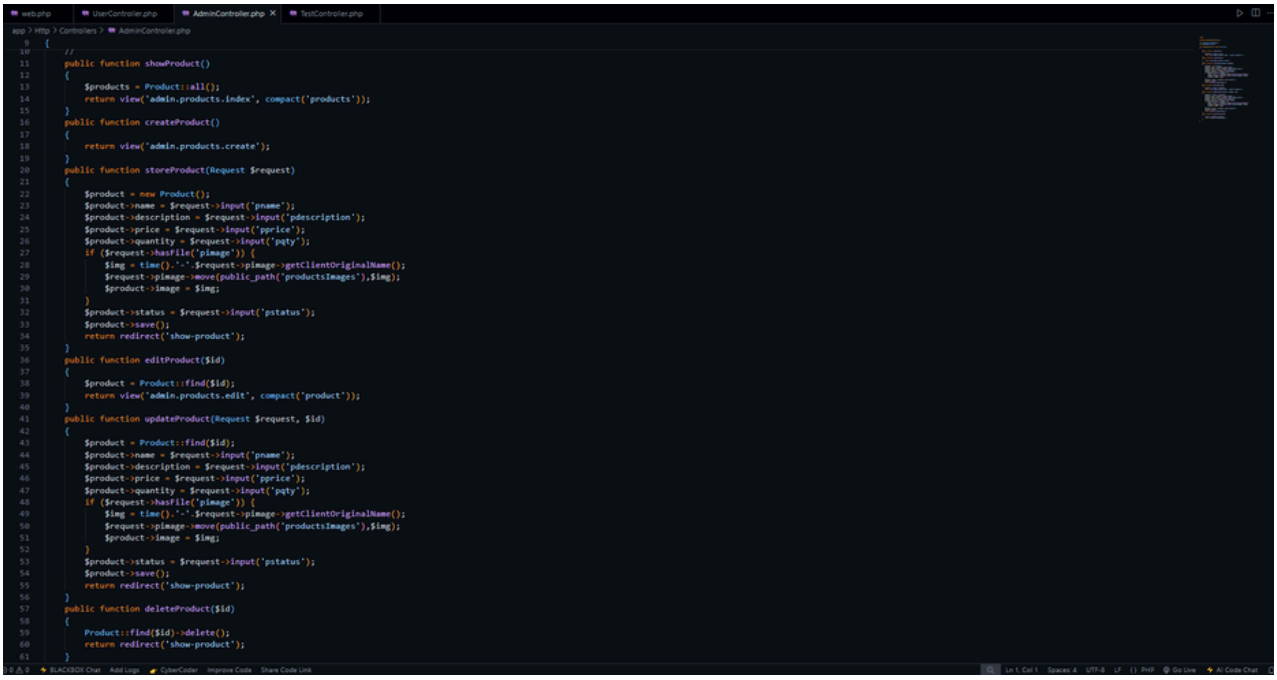


```php
<?php

use App\Http\Controllers\ProfileController;
use App\Http\Controllers\AdminController;
use App\Http\Controllers\TestController;
use App\Http\Controllers\UserController;
use Illuminate\Support\Facades\Route;

Route::get('/', [UserController::class, 'index']);
Route::get('about-us', [UserController::class, 'about']);
Route::get('contact-us', [UserController::class, 'contact']);

Route::get('/dashboard', function () {
    return view('dashboard');
})->middleware(['auth', 'verified'])->name('dashboard');

Route::get('/admin-dashboard', function () {
    return view('admin.dashboard');
})->middleware(['auth', 'verified'])->name('admin.dashboard');

Route::get('/tester-dashboard', [TestController::class, 'index']);

Route::middleware('auth')->group(function () {
    Route::get('/profile', [ProfileController::class, 'edit'])->name('profile.edit');
    Route::patch('/profile', [ProfileController::class, 'update'])->name('profile.update');
    Route::delete('/profile', [ProfileController::class, 'destroy'])->name('profile.destroy');

    // Admin Panel Routes :-
    Route::get('show-product', [AdminController::class, 'showProduct']);
    Route::get('create-product', [AdminController::class, 'createProduct']);
    Route::post('store-product', [AdminController::class, 'storeProduct']);
    Route::get('edit-product/{id}', [AdminController::class, 'editProduct']);
    Route::post('update-product/{id}', [AdminController::class, 'updateProduct']);
    Route::get('delete-product/{id}', [AdminController::class, 'deleteProduct']);

    // Tester Panel Route:
    Route::get('/show-tests', [TestController::class, 'index'])->name('tests.index'); // List Tests
    Route::get('/create-tests', [TestController::class, 'createTest'])->name('tests.create'); // Show Create Form
    Route::post('/store-tests', [TestController::class, 'store'])->name('tests.store'); // Store New Test
    Route::get('/tests/{id}/edit', [TestController::class, 'edit'])->name('tests.edit');
    Route::put('/tests/{test_id}', [TestController::class, 'updateTest'])->name('tests.update'); // Update Test
    Route::delete('/delete-tests/{id}', [TestController::class, 'destroy'])->name('tests.destroy'); // Delete Test

    // User panel test route
    Route::post('/submit-test', [UserController::class, 'submitTest'])->name('submit.test');
    Route::get('/my-tests', [UserController::class, 'myTests'])->name('user.tests');


});

require __DIR__.'/auth.php';
```

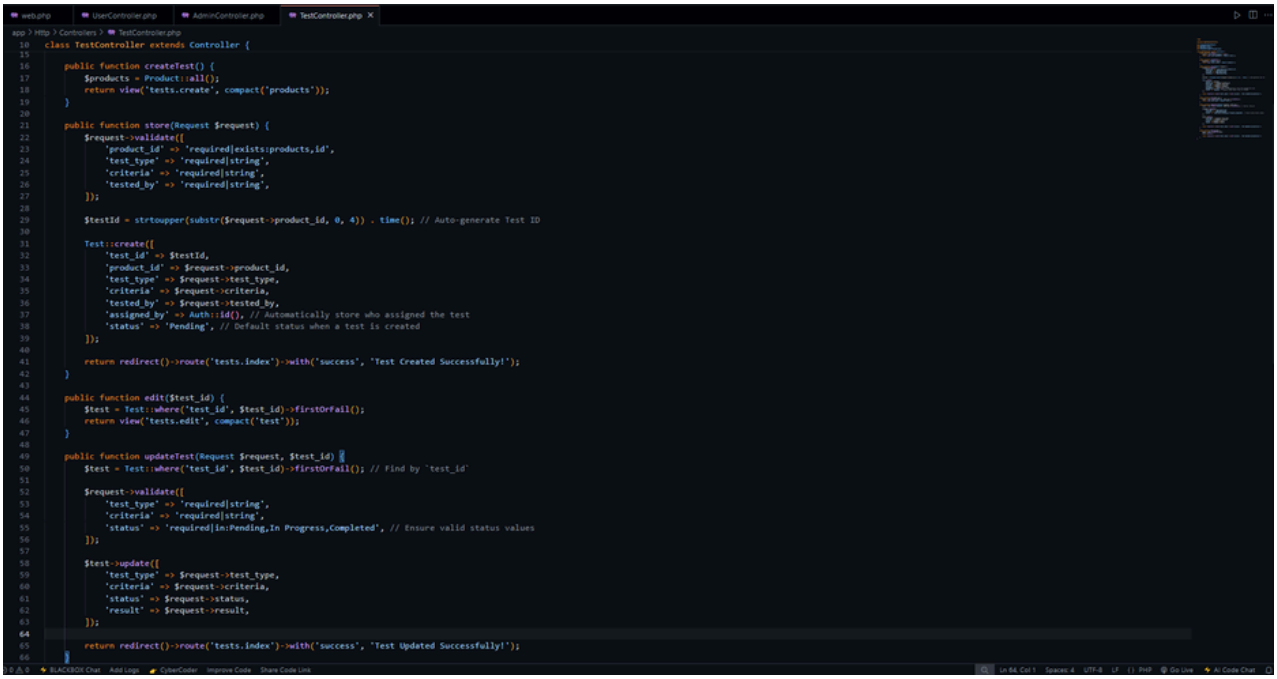


```php
namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Models\Test; // Import Test Model
use Illuminate\Support\Facades\Auth;
use App\Models\Product;

class UserController extends Controller
{
    public function index()
    {
        $products = Product::all();
        return view('index', compact('products'));
    }

    public function about()
    {
        return view('about-us');
    }

    public function contact()
    {
        return view('contact-us');
    }

    public function submitTest(Request $request)
    {
        // $request->validate([
        //     'product_id' => 'required|exists:products,id',
        // ]);

        $testId = strtoupper(substr($request->product_id, 0, 4)) . time(); // Auto-generate Test ID

        Test::create([
            'test_id' => $testId,
            'product_id' => $request->product_id,
            'test_type' => $request->product_name,
            'criteria' => $request->product_description,
            'tested_by' => $request->product_name,
            'assigned_by' => Auth::id(), // Automatically store who assigned the test
            'status' => 'Pending', // Default status when a test is created
        ]);

        return redirect()->back()->with('success', 'Test request submitted successfully!');
    }

    public function myTests()
    {
        $tests = Test::where('assigned_by', Auth::id())->get();
        return view('test', compact('tests'));
    }
}
```

CODE SCREEN SHOTS

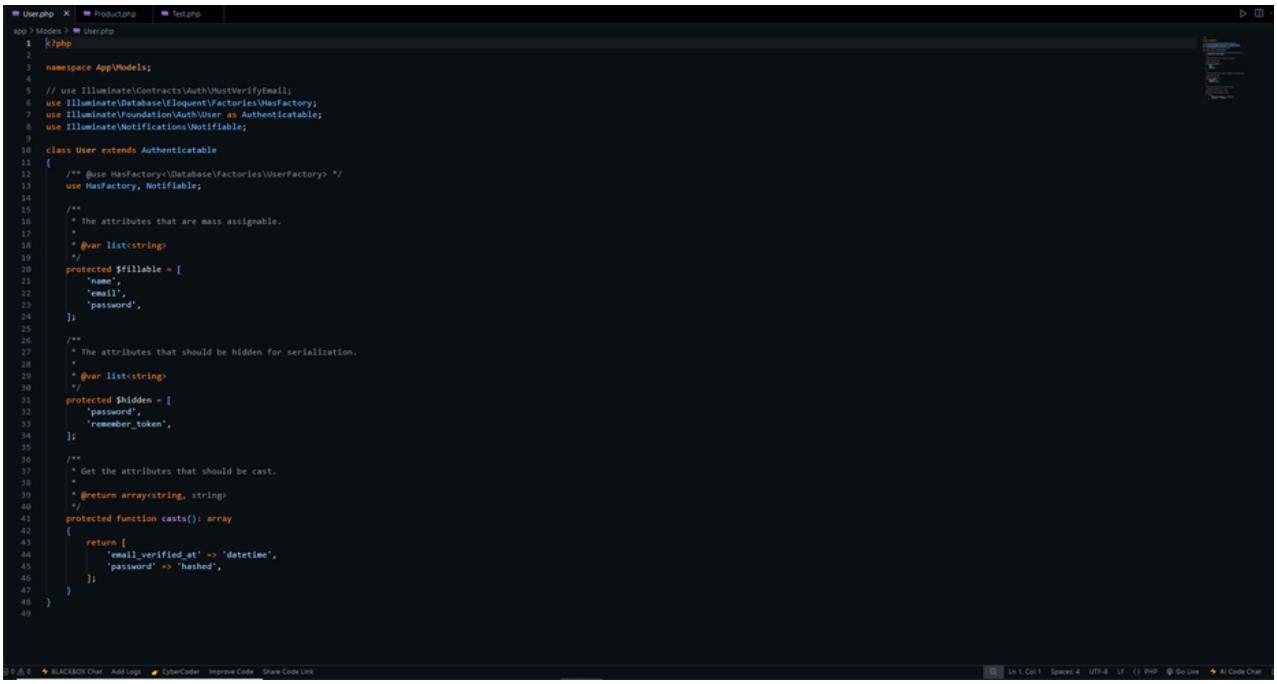


```php
{
    //
    public function showProduct()
    {
        $products = Product::all();
        return view('admin.products.index', compact('products'));
    }
    public function createProduct()
    {
        return view('admin.products.create');
    }
    public function storeProduct(Request $request)
    {
        $product = new Product();
        $product->name = $request->input('pname');
        $product->description = $request->input('pdescription');
        $product->price = $request->input('pprice');
        $product->quantity = $request->input('pqty');
        if ($request->hasFile('pimage')) {
            $img = time().'-'.$request->pimage->getClientOriginalName();
            $request->pimage->move(public_path('productsImages'),$img);
            $product->image = $img;
        }
        $product->status = $request->input('pstatus');
        $product->save();
        return redirect('show-product');
    }
    public function editProduct($id)
    {
        $product = Product::find($id);
        return view('admin.products.edit', compact('product'));
    }
    public function updateProduct(Request $request, $id)
    {
        $product = Product::find($id);
        $product->name = $request->input('pname');
        $product->description = $request->input('pdescription');
        $product->price = $request->input('pprice');
        $product->quantity = $request->input('pqty');
        if ($request->hasFile('pimage')) {
            $img = time().'-'.$request->pimage->getClientOriginalName();
            $request->pimage->move(public_path('productsImages'),$img);
            $product->image = $img;
        }
        $product->status = $request->input('pstatus');
        $product->save();
        return redirect('show-product');
    }
    public function deleteProduct($id)
    {
        Product::find($id)->delete();
        return redirect('show-product');
    }
```

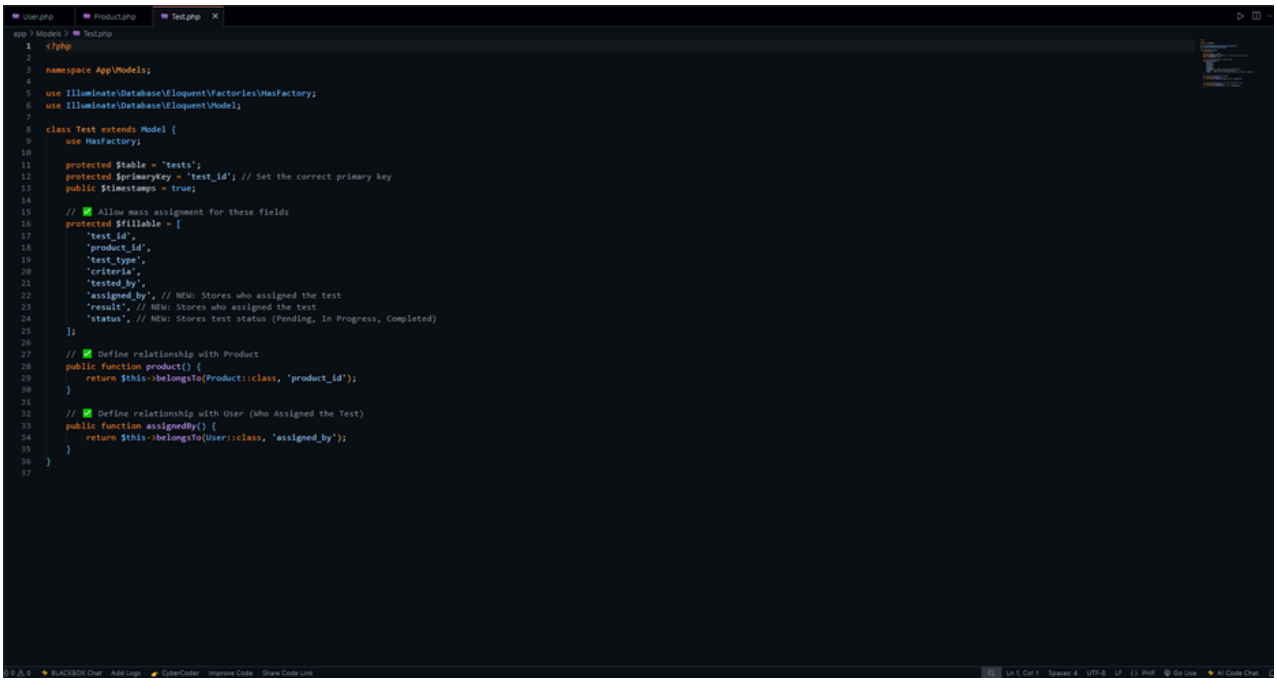


```php
class TestController extends Controller {

    public function createTest() {
        $products = Product::all();
        return view('tests.create', compact('products'));
    }

    public function store(Request $request) {
        $request->validate([
            'product_id' => 'required|exists:products,id',
            'test_type' => 'required|string',
            'criteria' => 'required|string',
            'tested_by' => 'required|string',
        ]);

        $testId = strtoupper(substr($request->product_id, 0, 4)) . time(); // Auto-generate Test ID

        Test::create([
            'test_id' => $testId,
            'product_id' => $request->product_id,
            'test_type' => $request->test_type,
            'criteria' => $request->criteria,
            'tested_by' => $request->tested_by,
            'assigned_by' => Auth::id(), // Automatically store who assigned the test
            'status' => 'Pending', // Default status when a test is created
        ]);

        return redirect()->route('tests.index')->with('success', 'Test Created Successfully!');
    }

    public function edit($test_id) {
        $test = Test::where('test_id', $test_id)->firstOrFail();
        return view('tests.edit', compact('test'));
    }

    public function updateTest(Request $request, $test_id) {
        $test = Test::where('test_id', $test_id)->firstOrFail(); // Find by `test_id`

        $request->validate([
            'test_type' => 'required|string',
            'criteria' => 'required|string',
            'status' => 'required|in:Pending,In Progress,Completed', // Ensure valid status values
        ]);

        $test->update([
            'test_type' => $request->test_type,
            'criteria' => $request->criteria,
            'status' => $request->status,
            'result' => $request->result,
        ]);

        return redirect()->route('tests.index')->with('success', 'Test Updated Successfully!');
    }
```

CODE SCREEN SHOTS

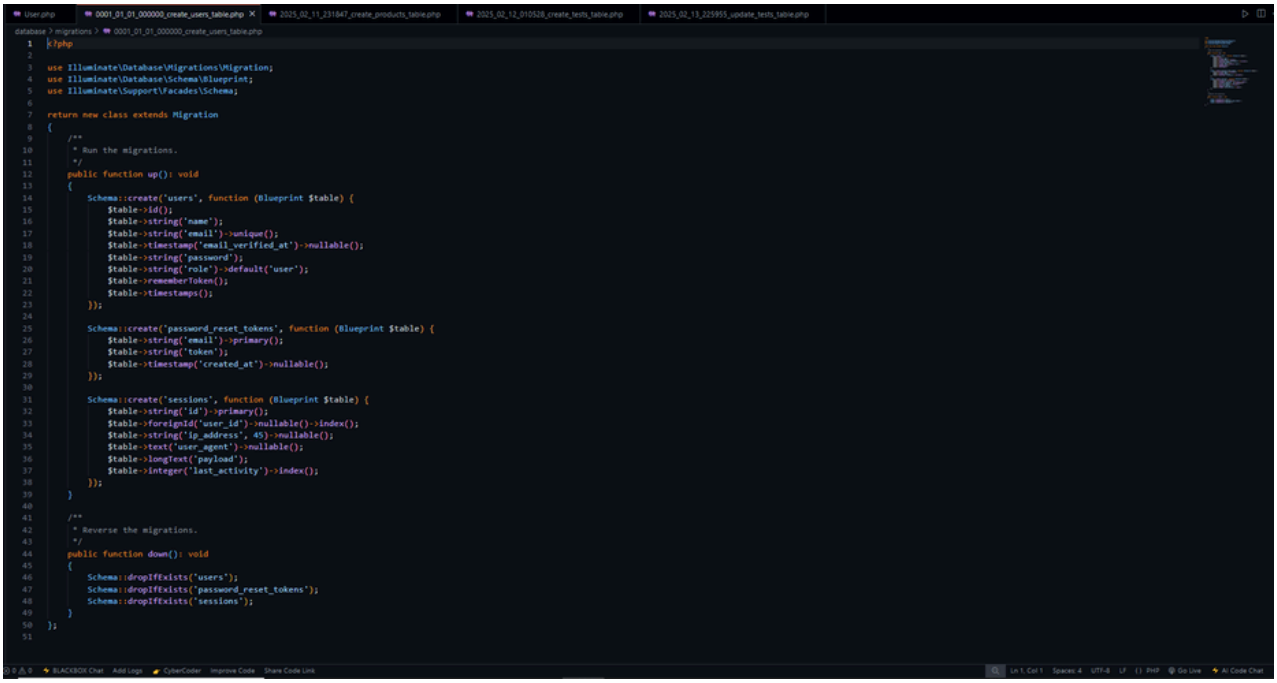


```php
<?php

namespace App\Models;

// use Illuminate\Contracts\Auth\MustVerifyEmail;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Foundation\Auth\User as Authenticatable;
use Illuminate\Notifications\Notifiable;

class User extends Authenticatable
{
    /** @use HasFactory<\Database\Factories\UserFactory> */
    use HasFactory, Notifiable;

    /**
     * The attributes that are mass assignable.
     *
     * @var list<string>
     */
    protected $fillable = [
        'name',
        'email',
        'password',
    ];

    /**
     * The attributes that should be hidden for serialization.
     *
     * @var list<string>
     */
    protected $hidden = [
        'password',
        'remember_token',
    ];

    /**
     * Get the attributes that should be cast.
     *
     * @return array<string, string>
     */
    protected function casts(): array
    {
        return [
            'email_verified_at' => 'datetime',
            'password' => 'hashed',
        ];
    }
}
```

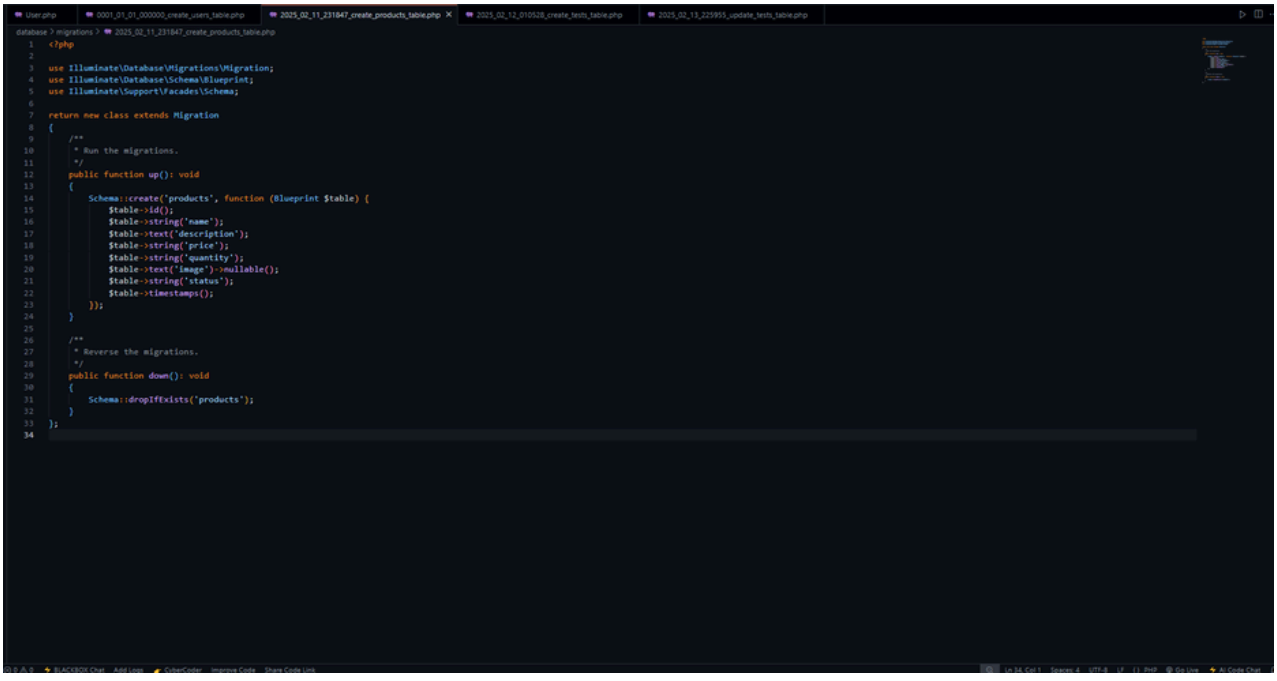


```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Test extends Model {
    use HasFactory;

    protected $table = 'tests';
    protected $primaryKey = 'test_id'; // Set the correct primary key
    public $timestamps = true;

    // ✅ Allow mass assignment for these fields
    protected $fillable = [
        'test_id',
        'product_id',
        'test_type',
        'criteria',
        'tested_by',
        'assigned_by', // NEW: Stores who assigned the test
        'result', // NEW: Stores who assigned the test
        'status', // NEW: Stores test status (Pending, In Progress, Completed)
    ];

    // ✅ Define relationship with Product
    public function product() {
        return $this->belongsTo(Product::class, 'product_id');
    }

    // ✅ Define relationship with User (Who Assigned the Test)
    public function assignedBy() {
        return $this->belongsTo(User::class, 'assigned_by');
    }
}
```

CODE SCREEN SHOTS

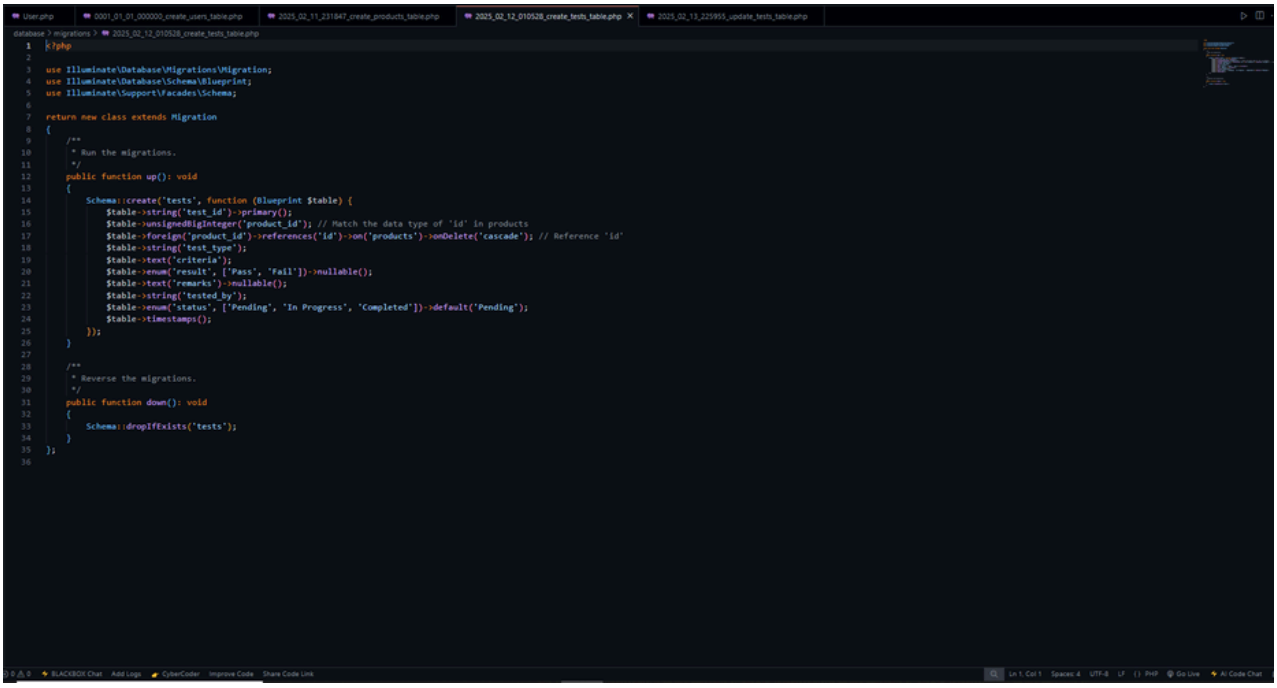


```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('users', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->string('email')->unique();
            $table->timestamp('email_verified_at')->nullable();
            $table->string('password');
            $table->string('role')->default('user');
            $table->rememberToken();
            $table->timestamps();
        });

        Schema::create('password_reset_tokens', function (Blueprint $table) {
            $table->string('email')->primary();
            $table->string('token');
            $table->timestamp('created_at')->nullable();
        });

        Schema::create('sessions', function (Blueprint $table) {
            $table->string('id')->primary();
            $table->foreignId('user_id')->nullable()->index();
            $table->string('ip_address', 45)->nullable();
            $table->text('user_agent')->nullable();
            $table->longText('payload');
            $table->integer('last_activity')->index();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('users');
        Schema::dropIfExists('password_reset_tokens');
        Schema::dropIfExists('sessions');
    }
};
```

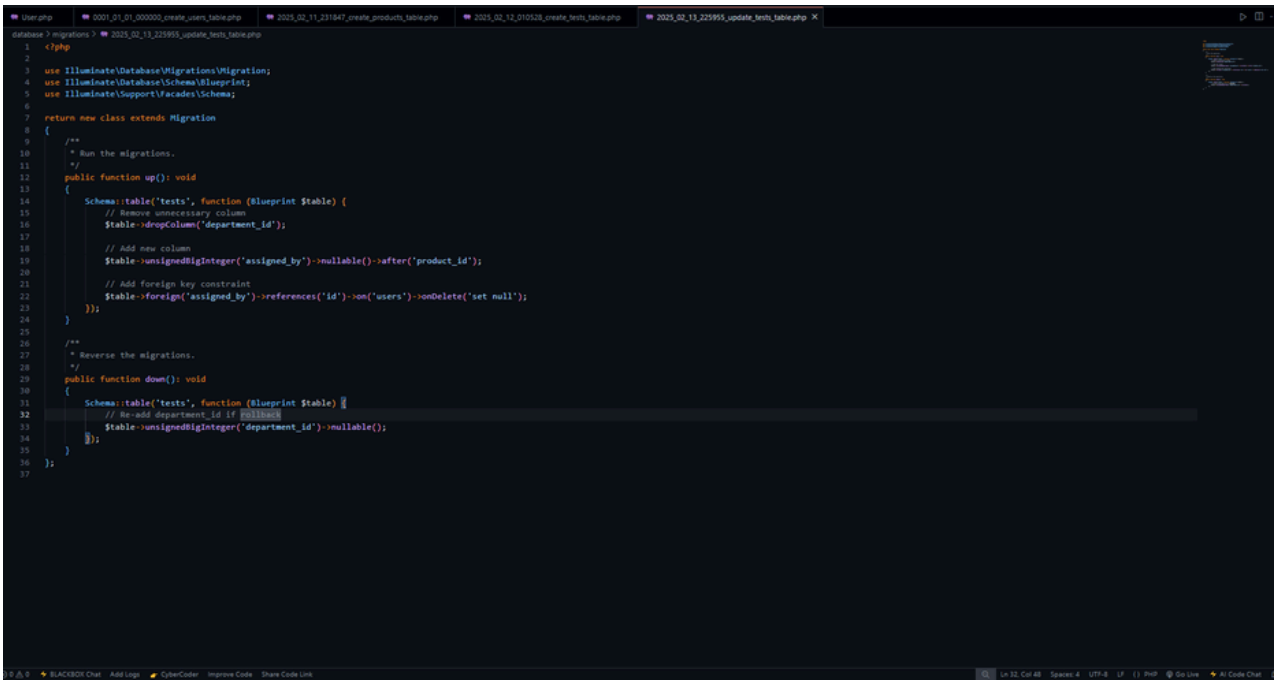


```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('products', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->text('description');
            $table->string('price');
            $table->string('quantity');
            $table->text('image')->nullable();
            $table->string('status');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('products');
    }
};
```

CODE SCREEN SHOTS

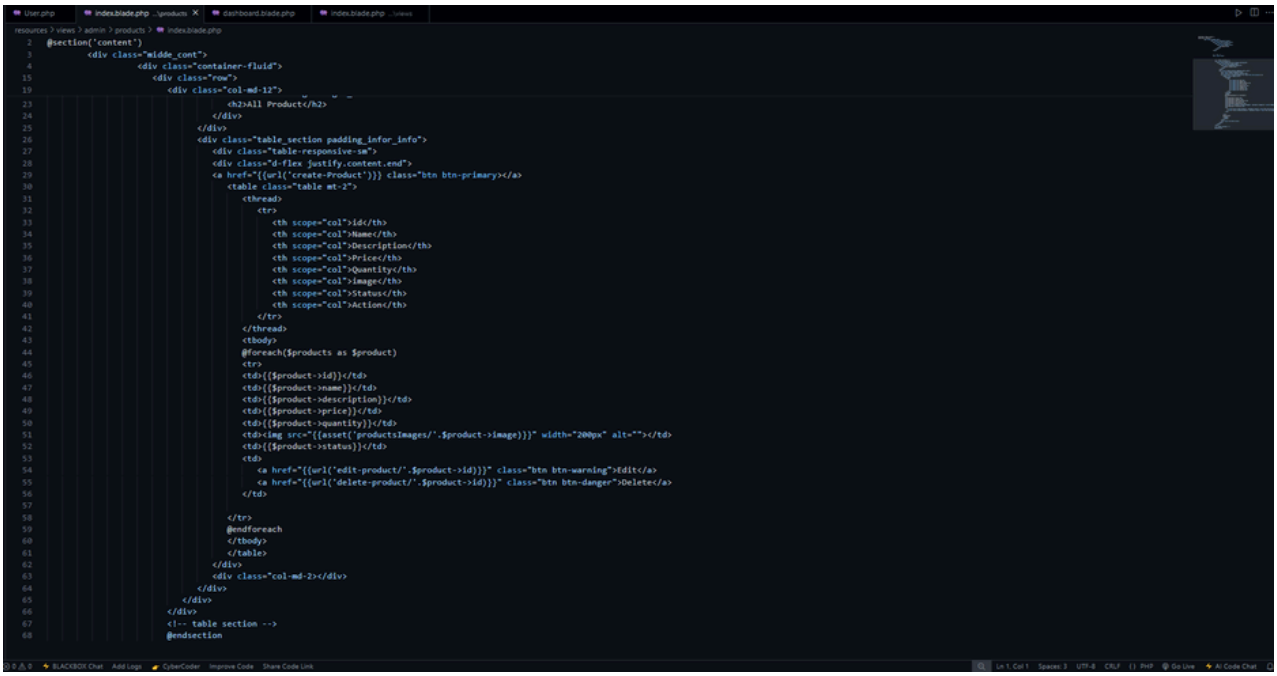


```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::create('tests', function (Blueprint $table) {
            $table->string('test_id')->primary();
            $table->unsignedBigInteger('product_id'); // Match the data type of 'id' in products
            $table->foreign('product_id')->references('id')->on('products')->onDelete('cascade'); // Reference 'id'
            $table->string('test_type');
            $table->text('criteria');
            $table->enum('result', ['Pass', 'Fail'])->nullable();
            $table->text('remarks')->nullable();
            $table->string('tested_by');
            $table->enum('status', ['Pending', 'In Progress', 'Completed'])->default('Pending');
            $table->timestamps();
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::dropIfExists('tests');
    }
};
```

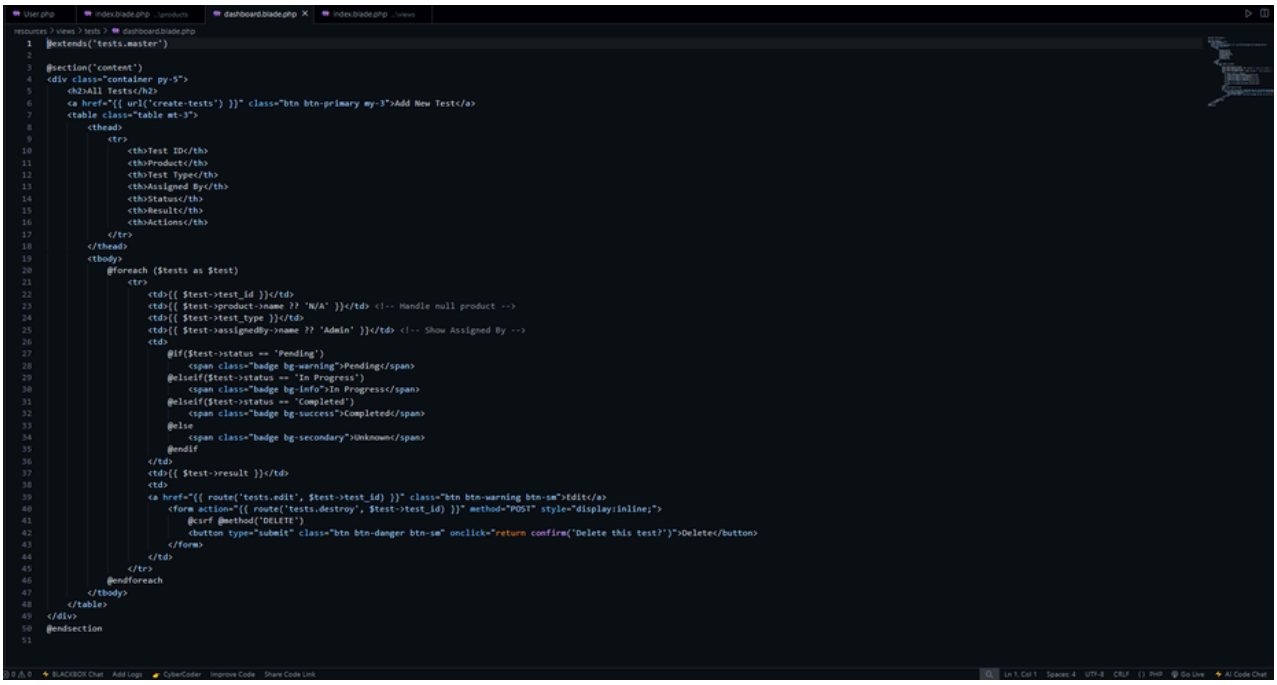


```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

return new class extends Migration
{
    /**
     * Run the migrations.
     */
    public function up(): void
    {
        Schema::table('tests', function (Blueprint $table) {
            // Remove unnecessary column
            $table->dropColumn('department_id');

            // Add new column
            $table->unsignedBigInteger('assigned_by')->nullable()->after('product_id');

            // Add foreign key constraint
            $table->foreign('assigned_by')->references('id')->on('users')->onDelete('set null');
        });
    }

    /**
     * Reverse the migrations.
     */
    public function down(): void
    {
        Schema::table('tests', function (Blueprint $table) {
            // Re-add department_id if rollback
            $table->unsignedBigInteger('department_id')->nullable();
        });
    }
};
```

CODE SCREEN SHOTS

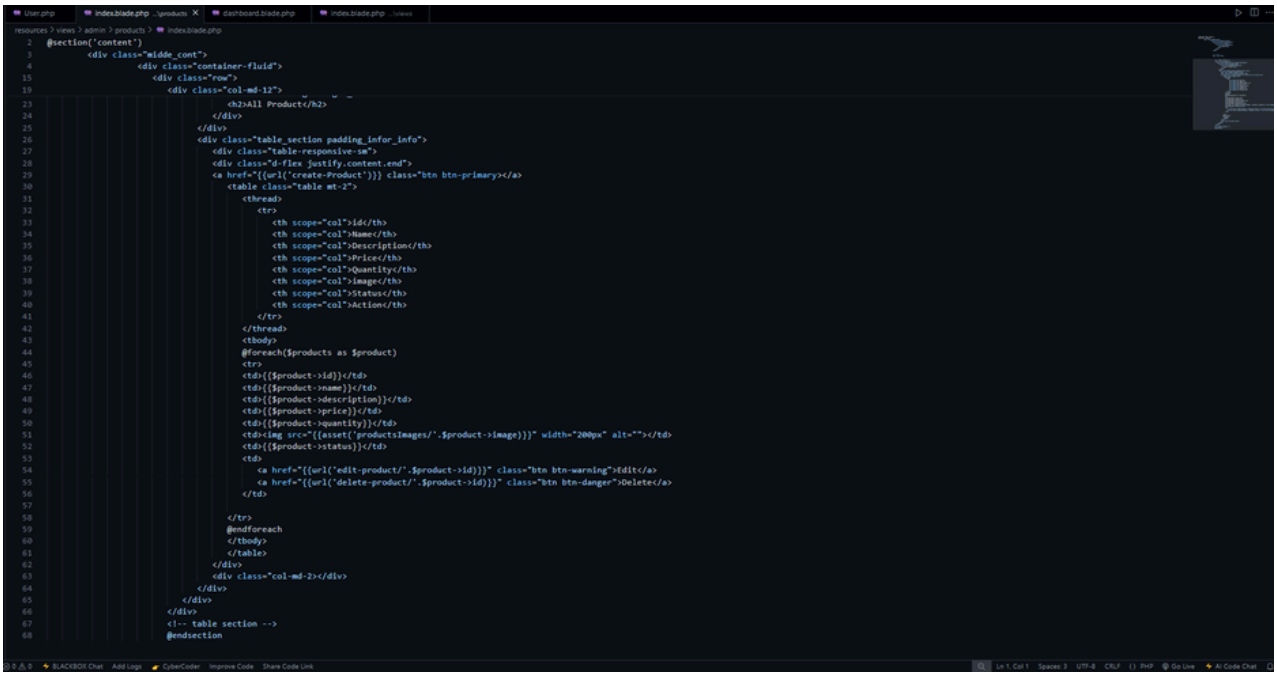


```blade
@section('content')
        <div class="midde_cont">
                <div class="container-fluid">
                   <div class="row">
                      <div class="col-md-12">
                                 <h2>All Product</h2>
                              </div>
                           </div>
                           <div class="table_section padding_infor_info">
                              <div class="table-responsive-sm">
                              <div class="d-flex justify.content.end">
                              <a href="{{url('create-Product')}} class="btn btn-primary></a>
                                 <table class="table mt-2">
                                    <thead>
                                       <tr>
                                          <th scope="col">id</th>
                                          <th scope="col">Name</th>
                                          <th scope="col">Description</th>
                                          <th scope="col">Price</th>
                                          <th scope="col">Quantity</th>
                                          <th scope="col">image</th>
                                          <th scope="col">Status</th>
                                          <th scope="col">Action</th>
                                       </tr>
                                    </thead>
                                    <tbody>
                                    @foreach($products as $product)
                                    <tr>
                                    <td>{{$product->id}}</td>
                                    <td>{{$product->name}}</td>
                                    <td>{{$product->description}}</td>
                                    <td>{{$product->price}}</td>
                                    <td>{{$product->quantity}}</td>
                                    <td><img src="{{asset('productsImages/'.$product->image)}}" width="200px" alt=""></td>
                                    <td>{{$product->status}}</td>
                                    <td>
                                       <a href="{{url('edit-product/'.$product->id)}}" class="btn btn-warning">Edit</a>
                                       <a href="{{url('delete-product/'.$product->id)}}" class="btn btn-danger">Delete</a>
                                    </td>

                                    </tr>
                                    @endforeach
                                    </tbody>
                                    </table>
                                 </div>
                                 <div class="col-md-2></div>
                              </div>
                           </div>
                        </div>
                        <!-- table section -->
                        @endsection
```

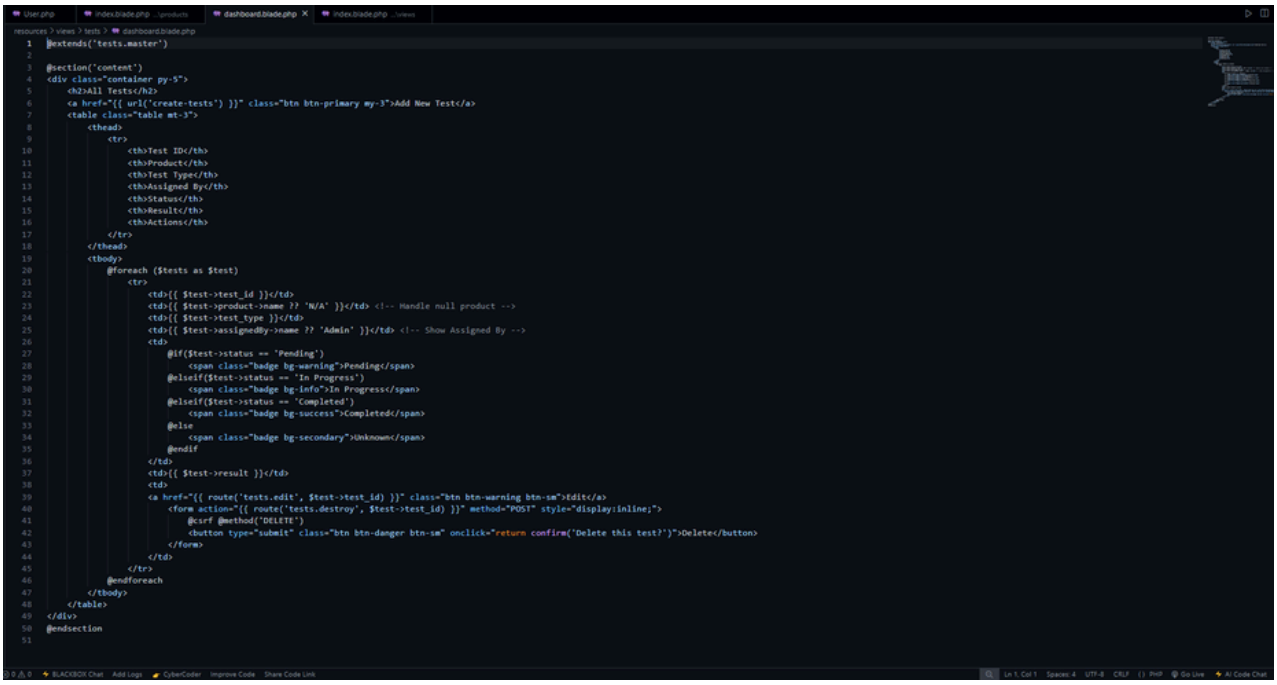


```blade
@extends('tests.master')

@section('content')
<div class="container py-5">
    <h2>All Tests</h2>
    <a href="{{ url('create-tests') }}" class="btn btn-primary my-3">Add New Test</a>
    <table class="table mt-3">
        <thead>
            <tr>
                <th>Test ID</th>
                <th>Product</th>
                <th>Test Type</th>
                <th>Assigned By</th>
                <th>Status</th>
                <th>Result</th>
                <th>Actions</th>
            </tr>
        </thead>
        <tbody>
            @foreach ($tests as $test)
                <tr>
                    <td>{{ $test->test_id }}</td>
                    <td>{{ $test->product->name ?? 'N/A' }}</td> <!-- Handle null product -->
                    <td>{{ $test->test_type }}</td>
                    <td>{{ $test->assignedBy->name ?? 'Admin' }}</td> <!-- Show Assigned By -->
                    <td>
                        @if($test->status == 'Pending')
                            <span class="badge bg-warning">Pending</span>
                        @elseif($test->status == 'In Progress')
                            <span class="badge bg-info">In Progress</span>
                        @elseif($test->status == 'Completed')
                            <span class="badge bg-success">Completed</span>
                        @else
                            <span class="badge bg-secondary">Unknown</span>
                        @endif
                    </td>
                    <td>{{ $test->result }}</td>
                    <td>
                    <a href="{{ route('tests.edit', $test->test_id) }}" class="btn btn-warning btn-sm">Edit</a>
                        <form action="{{ route('tests.destroy', $test->test_id) }}" method="POST" style="display:inline;">
                            @csrf @method('DELETE')
                            <button type="submit" class="btn btn-danger btn-sm" onclick="return confirm('Delete this test?')">Delete</button>
                        </form>
                    </td>
                </tr>
            @endforeach
        </tbody>
    </table>
</div>
@endsection
```

CODE SCREEN SHOTS

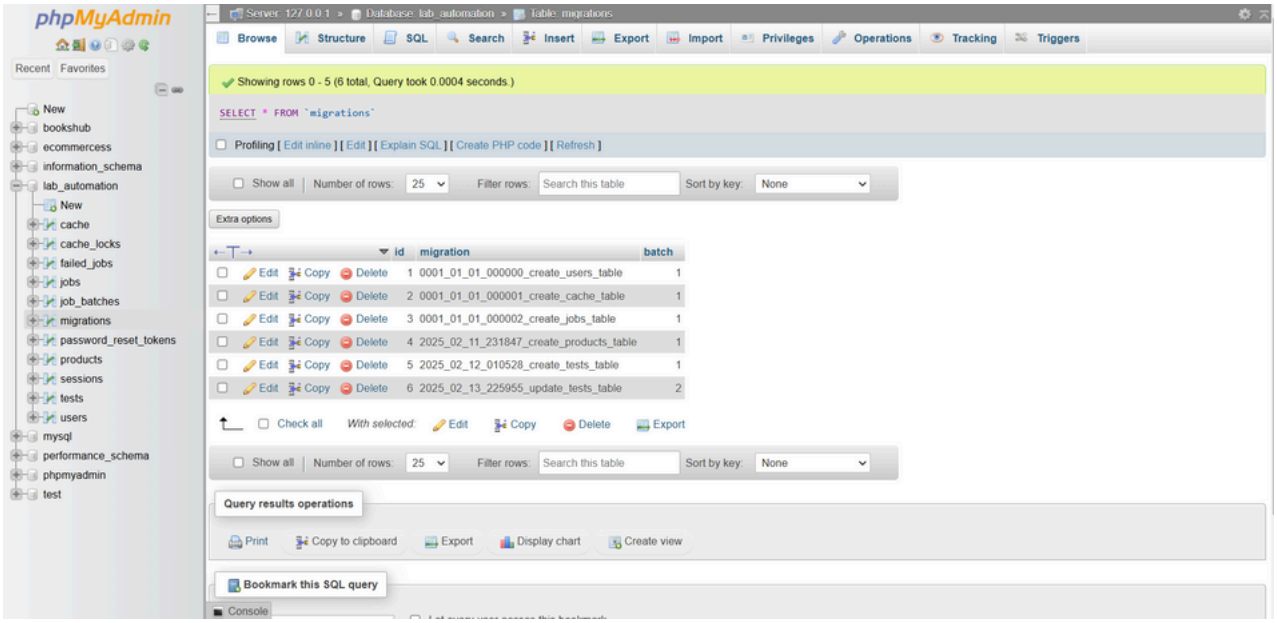


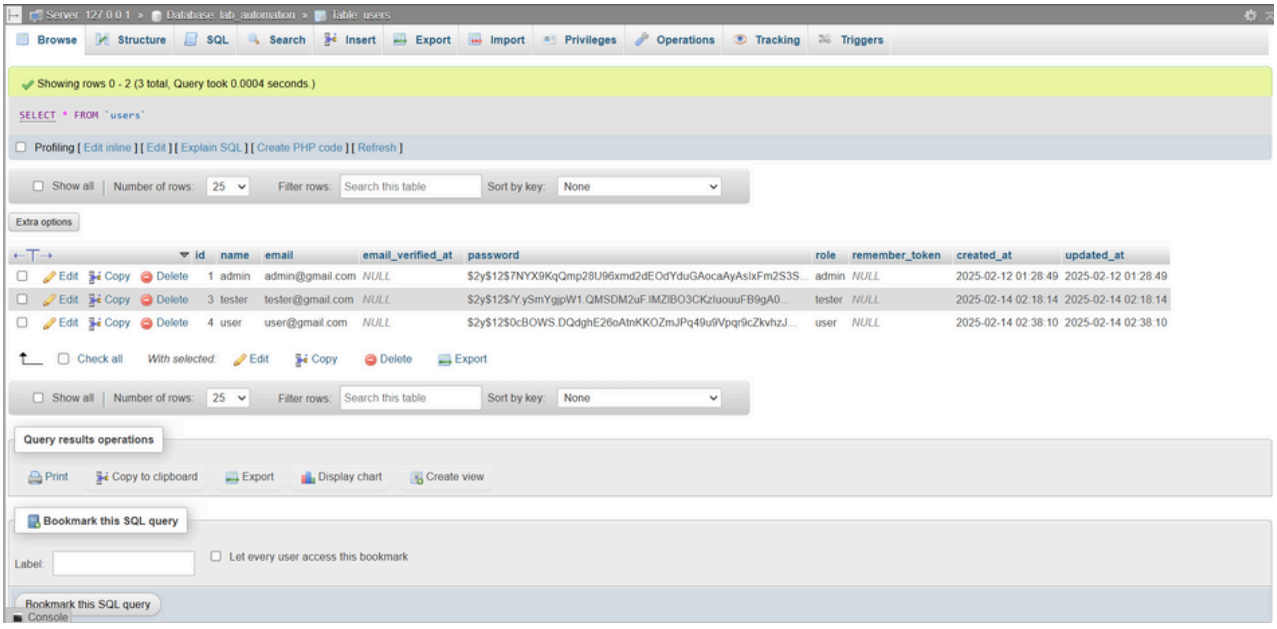
```blade
@section('content')
        <div class="midde_cont">
                <div class="container-fluid">
                   <div class="row">
                      <div class="col-md-12">
                                 <h2>All Product</h2>
                              </div>
                           </div>
                           <div class="table_section padding_infor_info">
                              <div class="table-responsive-sm">
                              <div class="d-flex justify.content.end">
                              <a href="{{url('create-Product')}} class="btn btn-primary></a>
                                 <table class="table mt-2">
                                    <thead>
                                       <tr>
                                          <th scope="col">id</th>
                                          <th scope="col">Name</th>
                                          <th scope="col">Description</th>
                                          <th scope="col">Price</th>
                                          <th scope="col">Quantity</th>
                                          <th scope="col">image</th>
                                          <th scope="col">Status</th>
                                          <th scope="col">Action</th>
                                       </tr>
                                    </thead>
                                    <tbody>
                                    @foreach($products as $product)
                                    <tr>
                                    <td>{{$product->id}}</td>
                                    <td>{{$product->name}}</td>
                                    <td>{{$product->description}}</td>
                                    <td>{{$product->price}}</td>
                                    <td>{{$product->quantity}}</td>
                                    <td><img src="{{asset('productsImages/'.$product->image)}}" width="200px" alt=""></td>
                                    <td>{{$product->status}}</td>
                                    <td>
                                       <a href="{{url('edit-product/'.$product->id)}}" class="btn btn-warning">Edit</a>
                                       <a href="{{url('delete-product/'.$product->id)}}" class="btn btn-danger">Delete</a>
                                    </td>

                                    </tr>
                                    @endforeach
                                    </tbody>
                                    </table>
                                 </div>
                                 <div class="col-md-2></div>
                              </div>
                           </div>
                        </div>
                        <!-- table section -->
                        @endsection
```



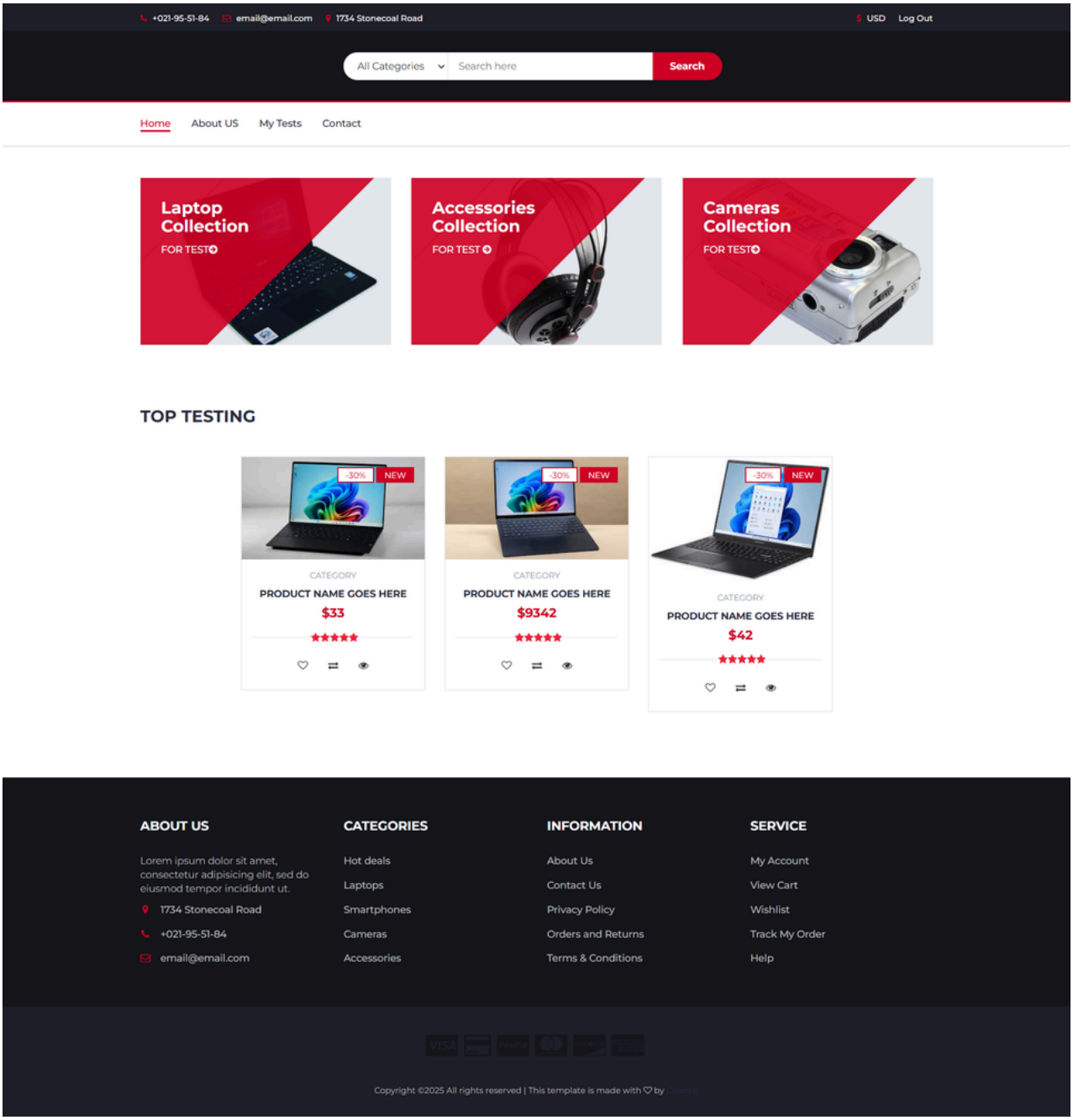
```blade
@extends('tests.master')

@section('content')
<div class="container py-5">
    <h2>All Tests</h2>
    <a href="{{ url('create-tests') }}" class="btn btn-primary my-3">Add New Test</a>
    <table class="table mt-3">
        <thead>
            <tr>
                <th>Test ID</th>
                <th>Product</th>
                <th>Test Type</th>
                <th>Assigned By</th>
                <th>Status</th>
                <th>Result</th>
                <th>Actions</th>
            </tr>
        </thead>
        <tbody>
            @foreach ($tests as $test)
                <tr>
                    <td>{{ $test->test_id }}</td>
                    <td>{{ $test->product->name ?? 'N/A' }}</td> <!-- Handle null product -->
                    <td>{{ $test->test_type }}</td>
                    <td>{{ $test->assignedBy->name ?? 'Admin' }}</td> <!-- Show Assigned By -->
                    <td>
                        @if($test->status == 'Pending')
                            <span class="badge bg-warning">Pending</span>
                        @elseif($test->status == 'In Progress')
                            <span class="badge bg-info">In Progress</span>
                        @elseif($test->status == 'Completed')
                            <span class="badge bg-success">Completed</span>
                        @else
                            <span class="badge bg-secondary">Unknown</span>
                        @endif
                    </td>
                    <td>{{ $test->result }}</td>
                    <td>
                    <a href="{{ route('tests.edit', $test->test_id) }}" class="btn btn-warning btn-sm">Edit</a>
                        <form action="{{ route('tests.destroy', $test->test_id) }}" method="POST" style="display:inline;">
                            @csrf @method('DELETE')
                            <button type="submit" class="btn btn-danger btn-sm" onclick="return confirm('Delete this test?')">Delete</button>
                        </form>
                    </td>
                </tr>
            @endforeach
        </tbody>
    </table>
</div>
@endsection
```

DB SCREEN SHOTS

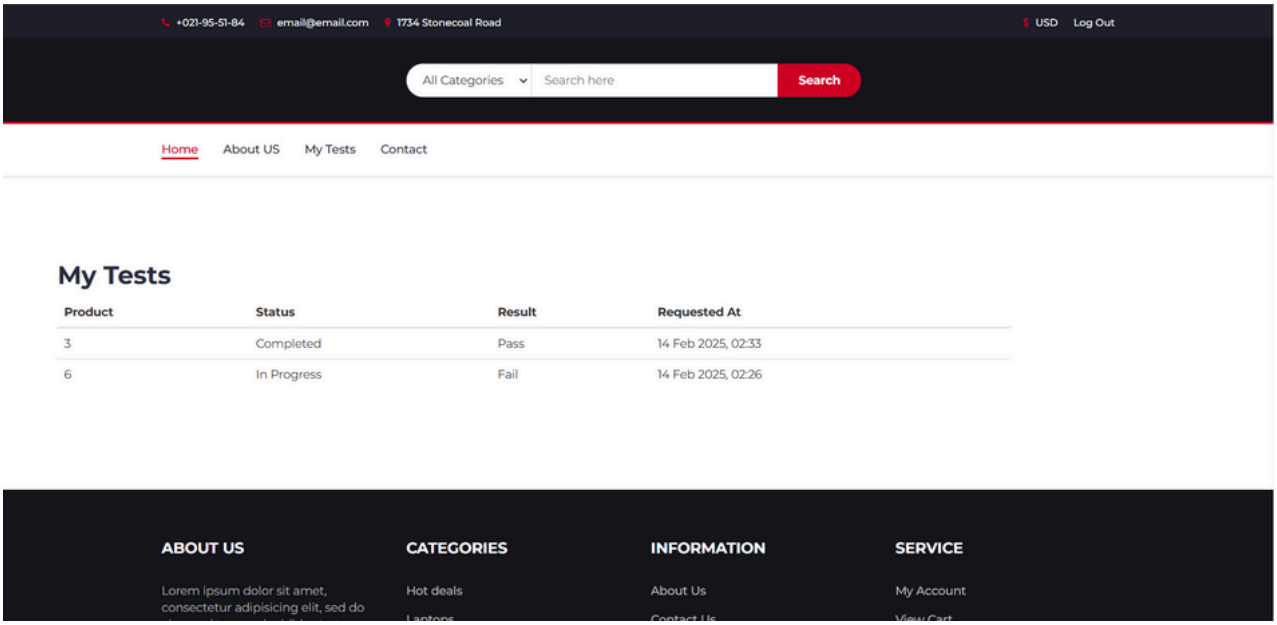




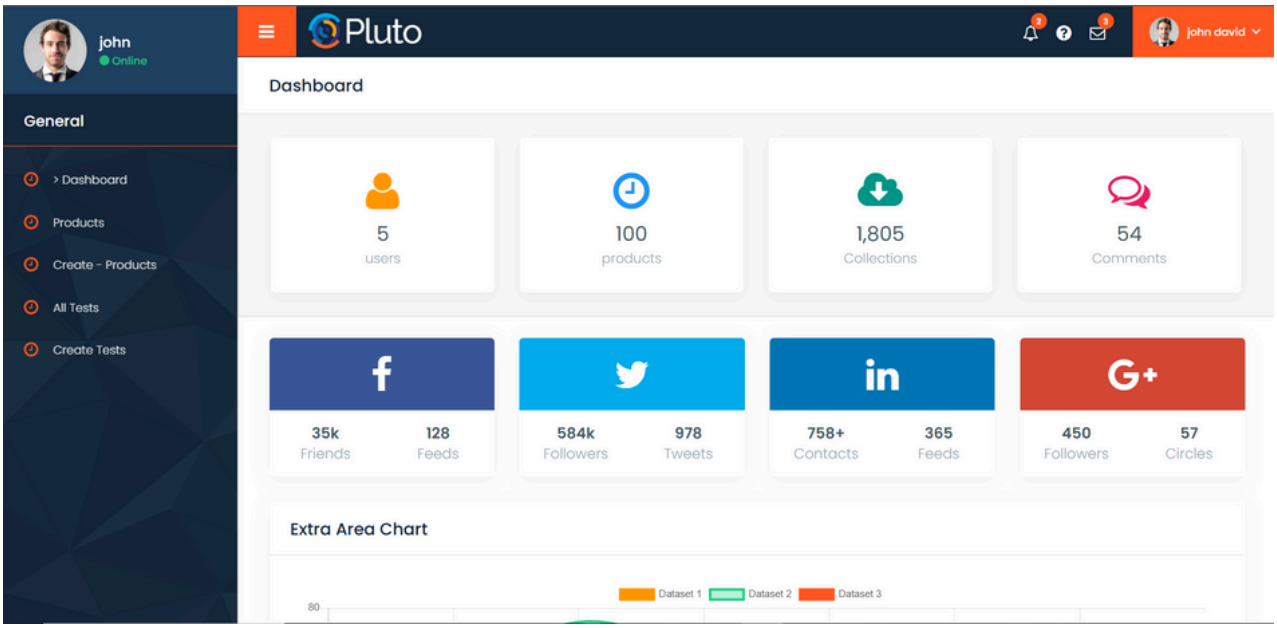
WEB SCREEN SHOTS

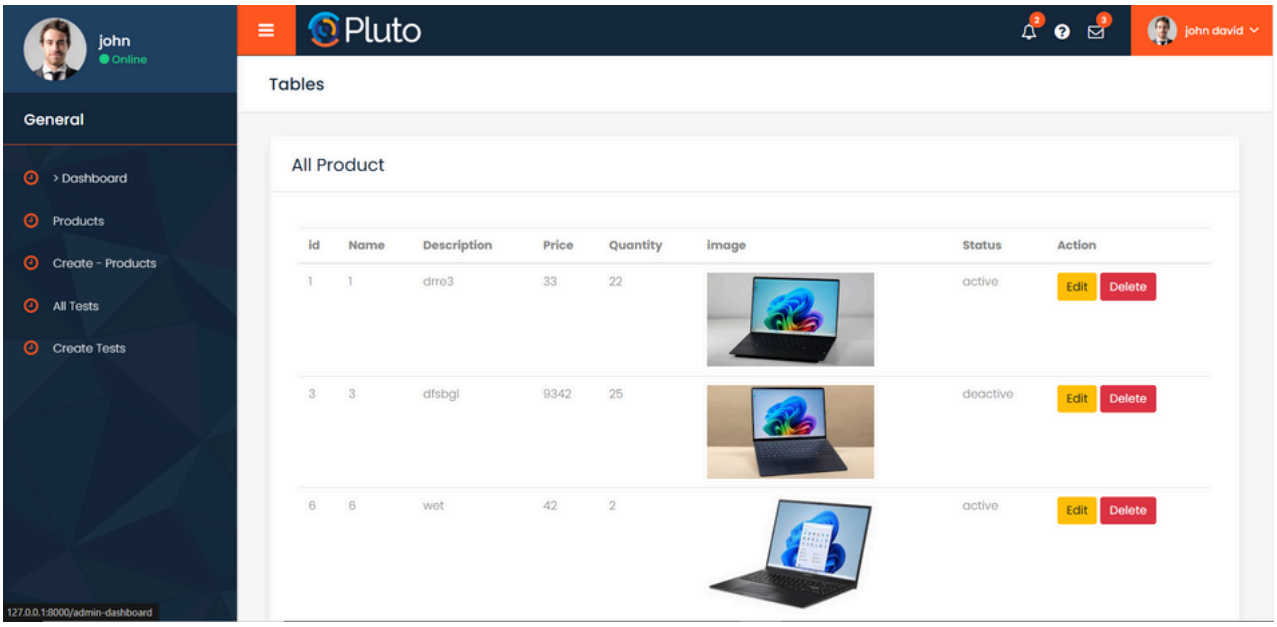


WEB SCREEN SHOTS

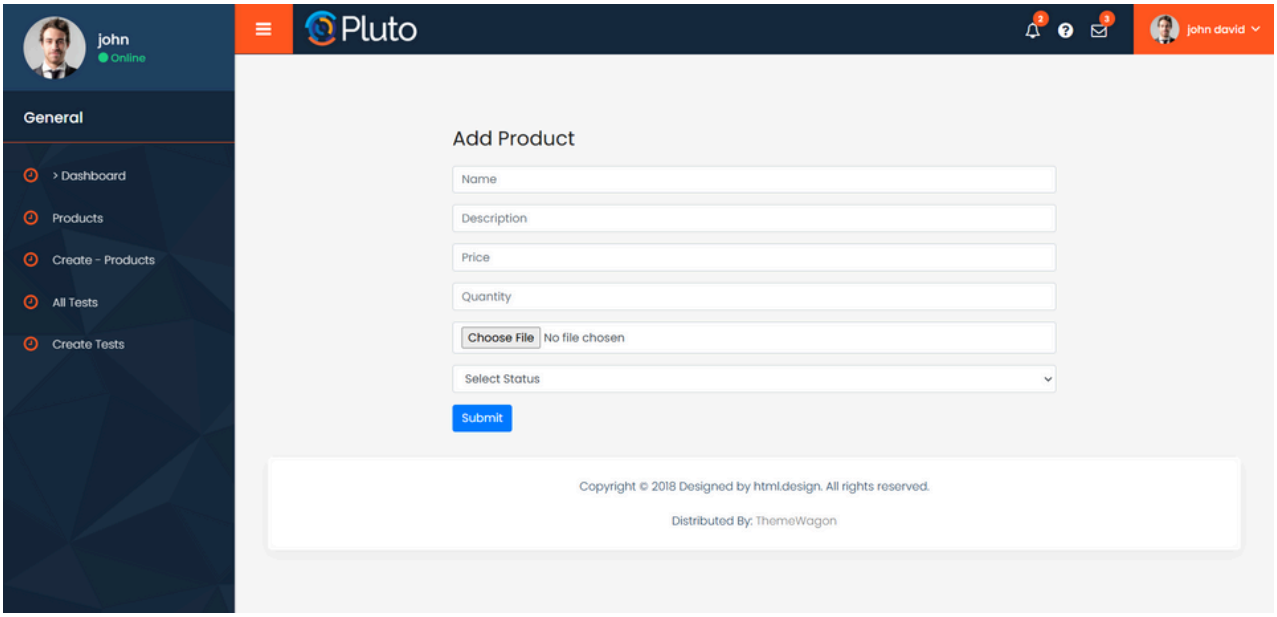


ADMIN SCREEN SHOTS





ADMIN SCREEN SHOTS



TESTER SCREEN SHOTS