

2022



MAJESTIC ADMIN

WEBSITE DOCUMENTATION

By ABDUS SABOOR

Group Members:

<i>Student Id</i>	<i>Student Name</i>
-------------------	---------------------

Student1282680	ABDUS SABOOR
----------------	--------------

Batch Code:

ALFBA_PR_2021_09G

Semester:

HDSE

Instructor:

Sir Faizan

Special Thanks to:

Sir Faizan

Submitted to:

eProject@apttech.ac.in

Table of Contents

ACKNOWLEDGEMENT.....	3
Warning.....	3
Task & Marks.....	4
Problem Statement.....	5
Requirement Specification:	6
Description of Webpages.....	8
Header.....	8
Insights Tab.....	8
Sidebar.....	9
Logo.....	9
Theme Screenshots.....	10
Code Screenshots.....	13

ACKNOWLEDGEMENT

This is to acknowledgement all those without whom this project would not have been reality. Firstly, we would wish to thank our teacher **Sir Faizan** who gave us their immense support, dedicated their time towards it and made us understand how to make this project. Without his guidance, the project would not have been completed.

Thank You

Warning

Before testing the following website make sure you have a **stable internet connection** as it contains CDNs of different libraries such as Bootstrap, JQuery, Material Design Icons etc.

TASKS

Every task is done by ABDUS SABOOR from information gathering and development to template integration.

MARKS

ABDUS SABOOR: 100

Problem Statement

The SRS Electrical appliances are the manufacturers of some electrical products like switch gears, fuses, capacitors, resistors, etc.... These products after being manufactured are sent for testing where the product is subjected to some testing conditions, and then after the testing is successful at their laboratories they are sent to CPRI for the further testing process so that they can get it approved from CPRI and then release the product into the market. If the testing fails then the product is sent back for remanufacturing and then they are again subjected to tests.

Requirement Specification:

- The details of the products and the testing like the product code, product id, testing id, the revise, the testing performed, remarks provided, the result of the testing, etc.... are needed to be maintained in a separate database.
- The product id should be a unique 10-digit code as described by the manufacturing unit and the testing id should be a unique 12-digit code, which should be automatically generated.
- An advanced search option is to be included.
- The application should contain a modular and a sub-modular form based on the type of the product and as well based on the type of the testing.
- One should be able to capture the detailed description of the remarks that include the testing and what should be the output for that testing and the result of testing performed as well as the remarks if any for that particular product.
- One should be able to check the status of the testing.
- After entering the record, it should even ask for the name of the person (s) who has tested the product.

After the product is subjected to testing and if it fails it is to be sent for re-making else it will be sent for further testing procedures that are held by the CPRI and get approved from them so as to release the product into the market.

Functional Requirements: –

- The details are needed to be captured based on the type of testing and as well based on the product.
- The status of the testing and as well the detailed remarks is also needed to be captured.
- An advanced search option is needed to be included so that one can fetch the details of the testing at an ease.

Description of Webpages

Header:

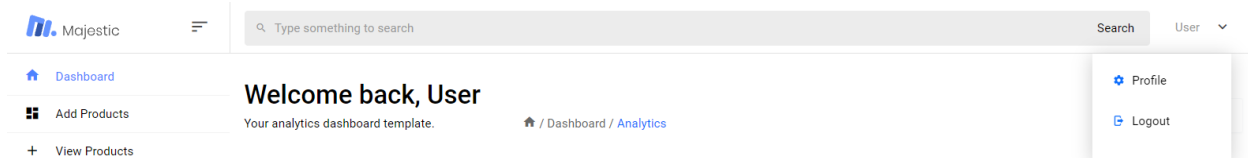


Figure 1: Header

The Header is located on every webpage of the website. It contains the Application's Logo, a search bar and attached with it is the corresponding Search button.

Furthermore, it has a down arrow icon which when clicked opens a dropdown containing Profile and Logout links.

Insights:

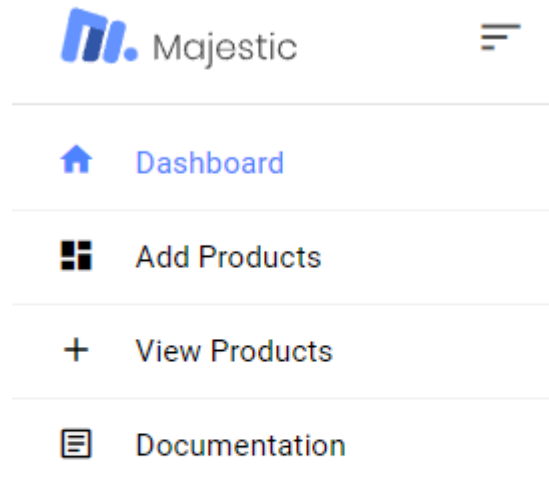


Figure 1: Insights

The Insights tab contains a four columns showing dynamic data named, Total Products, Tested Products, Untested Products, and Tested to Untested Ratio (Products).

Sidebar:

The sidebar contains links to other pages like Add Products, View Products, and Documentation and also includes a collapsible functionality.




Logo:



The logo on the top-left corner of the header is a clickable element. When clicked from any web page it redirects the user to the homepage of the website.

Screenshots:

Admin Panel:






User ▾

[Dashboard](#)
[Add Products](#)
[View Products](#)
[Documentation](#)


Welcome back, User


Your analytics dashboard template.


🏠 / Dashboard / Analytics






Insights

 Total Products
5


 Tested Products
4

 Untested Products
1

 Ratio: Tested to Untested
4

Latest Products

S.No	Product Name	Category	Testing Type	Remarks
01072209-10025	Resistor	tool	Not Tested Yet	Not satisfied
01072206-10024	Disco Lights	lights	Resistance Testing	Not satisfied
01072225-10019	Simple Button	Buttons	Leakage Testing	Satisfied
29062228-10018	Leminar Button	Buttons	Leakage Testing	Satisfied



User ▾

[Dashboard](#)
[Add Products](#)
[View Products](#)
[Documentation](#)

DEFAULT FORM


Product Name

Category

Testing Type

☐ Earth Testing
 ☐ Resistance Testing
 ☐ Leakage Testing
 ☐ Not Tested Yet

Remarks

 Majestic

Dashboard

Add Products

View Products

Documentation


search

Search

User

Products Details and Testing Type

S.No	Product Name	Category	Testing Type	Remarks		
28062219-10017	20 OHM Resistor	tool	Resistance Testing	Satisfied	Delete	Edit
29062228-10018	Leminar Button	Buttons	Leakage Testing	Satisfied	Delete	Edit
01072225-10019	Simple Button	Buttons	Leakage Testing	Satisfied	Delete	Edit
01072206-10024	Disco Lights	lights	Resistance Testing	Not satisfied	Delete	Edit
01072209-10025	Resistor	tool	Not Tested Yet	Not satisfied	Delete	Edit

 Majestic

Dashboard

Add Products

View Products

Documentation

search

Search

User

PRODUCT UPDATION FORM

Product Name

20 OHM Resistor

Category

tool

Testing Type

☐ Earth Testing

☐ Resistance Testing

☐ Leakage Testing

☐ Not Tested Yet

Remarks

Satisfied

Submit



Hello! let's get started

Sign in to continue.

Don't have an account? [Create](#)



New here?

Signing up is easy. It only takes a few steps

Already have an account? [Login](#)

Code:

```
//Login Routes
Route::get('login', [AuthController::class, 'GetLogin'])->name('login')->middleware('prevent-back-history');
Route::post('custom-login', [AuthController::class, 'customLogin'])->name('login.custom')->middleware('prevent-back-history');

//Registration Routes
Route::get('registration', [AuthController::class, 'registration'])->name('register-user');
Route::post('custom-registration', [AuthController::class, 'customRegistration'])->name('register.custom');

//Logout Routes
Route::get('signout', [AuthController::class, 'signOut'])->name('signout');

//Dashboard Routes
Route::get('dashboard', [ProductController::class, 'dashboard'])->name('dashboard')->middleware('prevent-back-history');

//Add or Create Product
Route::get('add-product', [ProductController::class, 'create'])->name('add-product');
Route::post('product_created', [ProductController::class, 'createProduct'])->name('product_created');

//View Product
Route::get('/view_products', [ProductController::class, 'viewDetails'])->name('view_products');

//Delete Product
Route::delete('/delete_product/{id}', [ProductController::class, 'destroy'])->name('delete_product');

//Search Product
Route::any('search', [ProductController::class, 'search'])->name('search');

//Edit Product
Route::get('/edit_product/{id?}', [ProductController::class, 'edit'])->name('edit_product');

//Update Product Data
Route::post('update_product/{id}', [ProductController::class, 'update'])->name('update_product');

//Update Profile Data
Route::get('/edit_user/{id}', [AuthController::class, 'editUser'])->name('edit_user');
Route::post('update_profile/{id?}', [AuthController::class, 'updateProfile'])->name('update_profile');
```

```
public function __construct()
{
    $this->middleware('guest')->except(['signOut', 'editUser', 'updateProfile']);
}

public function GetLogin()
{
    return view('auth.login');
}

public function customlogin(Request $request)
{
    $request->validate([
        'username' => 'required',
        'password' => 'required',
    ]);

    $credentials = $request->only('username', 'password');

    if(Auth::attempt($credentials)) {
        return redirect()->intended('dashboard')->withSuccess('Signed in');
    }

    return redirect("login")->withSuccess('Login details are not valid');
}
```

```
public function customRegistration(Request $request)
{
    $request->validate([
        'username' => 'required|unique:admin_tb',
        'password' => 'required|min:8',
        'confirm_password' => 'required|same:password'
    ]);

    $data = $request->all();
    $check = $this->create($data);

    return redirect("login")->withSuccess('You have signed-in');
}

public function create(array $data)
{
    return admin::create([
        'username' => $data['username'],
        'password' => Hash::make($data['password'])
    ]);
}

public function signOut() {
    Session::flush();
    Auth::logout();

    return Redirect('login');
}
```



```
public function signOut() {  
    Session::flush();  
    Auth::logout();  
  
    return Redirect('login');  
}  
  
public function editUser($id)  
{  
    $user_edit = admin::find($id);  
  
    return view('auth.profile', [$user_edit]);  
}  
  
public function updateProfile(Request $request, $id)  
{  
    $user = admin::findOrFail($id);  
  
    $user->username = $request->get('username');  
  
    $user->password = $request->get('password');  
  
    $user->save();  
  
    return redirect()->route('edit_user', [$user->id]);  
}
```