@extends('layout.adminPageControl')

@section('content')

<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/toastify-js/src/toastify.min.css">

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/tagify/4.4.1/tagify.css">

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/tagify/4.4.1/tagify.min.js"></script>

<link href="https://cdnjs.cloudflare.com/ajax/libs/dropzone/5.9.2/min/dropzone.min.css" rel="stylesheet">

<script src="https://cdnjs.cloudflare.com/ajax/libs/dropzone/5.9.2/min/dropzone.min.js"></script>

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/froala-editor/3.2.7/css/froala\_editor.pkgd.min.css">

<script src="https://cdnjs.cloudflare.com/ajax/libs/froala-editor/3.2.7/js/froala\_editor.pkgd.min.js"></script>

<script src="https://cdn.jsdelivr.net/npm/toastify-js/src/toastify.min.js"></script>

<style>

.draggable {

cursor: move;

}

.textarea-label {

display: block;

margin-bottom: 5px;

/\* Space between the label and textarea \*/

}

.fr-box.fr-basic.fr-top .fr-wrapper a {

display: none !important;

}

.full-width-textarea {

width: 100%;

/\* Full width \*/

box-sizing: border-box;

/\* Include padding and border in the element's total width \*/

/\* Optional: Add some padding for better UX \*/

padding: 10px;

/\* Optional: Add a border and border radius for styling \*/

border: 1px solid #ccc;

border-radius: 4px;

}

#fr-logo {

float: left;

outline: none;

display: none !important;

}

.container-fluid.mt-5 {

background: #e2ecfa;

padding: 22px;

}

.page-content {

background: #fff;

}

.highlight {

background-color: #f0f0f0;

/\* Highlight color for dragged rows \*/

}

.table {

width: 100%;

border-collapse: collapse;

margin-top: 20px;

}

.table th,

.table td {

padding: 12px 15px;

text-align: left;

}

.table th {

background-color: #f2f2f2;

border-bottom: 2px solid #ddd;

}

.table tr:nth-child(even) {

background-color: #f9f9f9;

}

.table td i {

cursor: pointer;

}

.bx-edit {

color: #007bff;

/\* Blue color for edit \*/

}

.bx-edit:hover {

color: #0056b3;

}

.bx-trash:hover {

color: #b30000;

/\* Darker red for hover \*/

}

/\* Style for drag-and-drop rows \*/

.draggable {

transition: background-color 0.2s ease;

}

.draggable:hover {

background-color: #e0e0e0;

}

hr {

border: none;

height: 2px;

/\* Thicker line \*/

background: linear-gradient(to right, #f8f8fb, #000000, #f8f8fb);

margin-top: 30px;

margin-bottom: 30px;

}

.image-add-input {

display: none;

}

.uploaded-images {

display: flex;

flex-wrap: nowrap;

/\* Disable wrapping \*/

gap: 10px;

overflow-x: auto;

/\* Enable horizontal scroll \*/

white-space: nowrap;

/\* Ensure items stay in a single line \*/

padding-bottom: 10px;

/\* Space for scroll bar \*/

}

.uploaded-image-item {

position: relative;

display: inline-block;

}

.uploaded-image-item img {

display: block;

max-width: 100px;

height: auto;

}

.uploaded-image-item .btn-danger {

position: absolute;

top: 0;

right: 0;

}

.uploaded-images {

overflow-x: auto;

scrollbar-width: none;

}

.uploaded-images::-webkit-scrollbar {

display: none;

}

</style>

<div class="main-content">

<div class="page-content">

<div>Page > {{ $page->name ?? '' }}</div>

<div class="container-fluid mt-5" id="logo-section">

<div class="row">

<div class="col-12">

<div class="page-title-box">

<h4 class="mb-sm-0 font-size-18">Clinet Logo Section</h4>

</div>

<div class="row mb-3 align-items-center">

<div class="col-12">

<form id="logosection">

@csrf

<input type="hidden" id="parent\_id" name="parent\_id" value="{{ $page->id }}">

<div class="row">

<div class="col-md-5">

<label for="client\_name">Title 1:</label>

<input type="text" class="form-control" id="title1" name="title1"

onchange="saveLogoData()" value="{{ $logo->title1 ?? '' }}" required>

</div>

<div class="col-md-7">

<label for="client\_role">Title 2:</label>

<input type="text" class="form-control" id="title2" name="title2"

value="{{ $logo->title2 ?? '' }}" onchange="saveLogoData()" required>

</div>

</div>

<label for="company\_name">Form Title:</label>

<input type="text" class="form-control" id="form\_title" name="form\_title"

value="{{ $logo->form\_title ?? '' }}" onchange="saveLogoData()" required>

</form>

<form action="{{ route('logos.store') }}" method="POST" enctype="multipart/form-data"

class="dropzone mt-2" id="logoDropzone">

@csrf

<input type="file" name="images[]" class="image-add-input" multiple>

<input type="hidden" id="parent\_id" name="parent\_id" value="{{ $page->id }}">

<div class="dz-message">

Drag & Drop your files here or click to upload.

</div>

</form>

<div class="uploaded-images mt-3">

@if (isset($clientlogos))

@foreach ($clientlogos as $logo)

<div class="uploaded-image-item" id="logo-{{ $logo->id }}">

<img src="{{ asset('storage/logos/' . $logo->image) }}" alt="Uploaded Image"

width="100">

<button type="button" class="btn btn-danger btn-sm"

onclick="deleteLogo({{ $logo->id }})">

<i class="fa fa-trash"></i> Delete

</button>

</div>

@endforeach

@endif

</div>

</div>

</div>

</div>

</div>

</div>

<hr>

<div class="container-fluid mt-5" id="year-of-experience">

<div class="row">

<div class="col-12">

<div class="page-title-box">

<h4 class="mb-sm-0 font-size-18">Years of Expertise Section</h4>

</div>

<div class="row mb-3 align-items-center">

<input type="hidden" class="recordId" value="">

<div class="row ms-1">

<div class="col-md-2">

<label for="years">Years:</label>

<input type="text" class="form-control get-touch-us-form title1" value="">

</div>

<div class="col-md-4">

<label>Title 1:</label>

<input type="text" class="form-control get-touch-us-form title3" value="">

</div>

<div class="col-md-6">

<label>Title 2:</label>

<input type="text" class="form-control get-touch-us-form title2" value="">

</div>

</div>

<div class="row">

<div class="col-md-3">

<label for="years">Years:</label>

<input type="text" class="form-control get-touch-us-form years" id="years"

value="">

<small class="form-text text-muted"> &nbsp;e.g., "5,Some text".</small>

</div>

<div class="col-md-3">

<label for="clients">Clients:</label>

<input type="text" class="form-control get-touch-us-form clients" id="clients"

value="">

<small class="form-text text-muted"> &nbsp;e.g., "5,Some text".</small>

</div>

<div class="col-md-3">

<label for="projects">Projects:</label>

<input type="text" class="form-control get-touch-us-form projects" id="projects"

value="">

<small class="form-text text-muted"> &nbsp;e.g., "5,Some text".</small>

</div>

<div class="col-md-3">

<label for="reviews">Reviews:</label>

<input type="text" class="form-control get-touch-us-form reviews" id="reviews"

value="">

<small class="form-text text-muted"> &nbsp;e.g., "5,Some text".</small>

<input type="hidden" class="parent\_id" id="parent\_id" name="parent\_id"

value="{{ $page->id }}">

</div>

</div>

</div>

</div>

</div>

</div>

<hr>

<div class="container-fluid mt-5" id="recent-projects">

<div class="row">

<div class="col-12">

<div class="row mb-3 align-items-center">

<div class="col-12">

<form id="recentProjectForm">

<input type="hidden" id="projectId" name="id">

<div class="row">

<div class="col-md-12 mb-3">

<label for="re\_content" class="textarea-label">Recent Projects Description:</label>

<textarea name="re\_content" id="re\_content" rows="3" class="full-width-textarea">{{ $recentProjectContent->content??'' }}</textarea>

</div>

</div>

<div class="page-title-box pb-2">

<h4 class="mb-sm-0 font-size-18">Recent Projects</h4>

</div>

<div class="row">

<input type="hidden" id="parent\_id" name="parent\_id"

value="{{ $page->id }}">

<div class="col-md-6">

<label for="client\_name">Project Name:</label>

<input type="text" class="form-control" id="name" name="name"

required>

<label for="client\_role">Project Description:</label>

<input type="text" class="form-control" id="description"

name="description" required>

</div>

<div class="col-md-6">

<label for="company\_name">Project Link:</label>

<input type="text" class="form-control" id="link" name="link"

required>

<label for="review\_content">Project Image:</label>

<input type="file" class="form-control" id="image" name="image">

<img id="currentImage" src="" alt="Current Project Image"

style="display:none; margin-top: 10px; max-width: 100px;">

</div>

</div>

<button type="submit" class="btn btn-primary mt-3" id="submitButton">Add

Project</button>

</form>

</div>

</div>

<div class="row mt-3" id="recentProjectContainer2">

<div class="row mt-3">

<div class="col-12">

<table class="table table-bordered" id="recentProjectContainer">

<thead>

<tr>

<th>Project Name</th>

<th>Project Discription</th>

<th>Project Link</th>

<th>Project Image</th>

<th>Actions</th>

</tr>

</thead>

<tbody>

</tbody>

</table>

</div>

</div>

</div>

</div>

</div>

</div>

<hr>

<div class="container-fluid mt-5" id="google-reviews">

<div class="row">

<div class="col-12">

<div class="page-title-box">

<h4 class="mb-sm-0 font-size-18">Google Reviews</h4>

</div>

<div class="row mb-3 align-items-center">

<div class="col-12">

<form id="reviewForm">

<div class="row ms-1">

<div class="col-md-6">

<input type="hidden" id="parent\_id" name="parent\_id"

value="{{ $page->id }}">

<div class="form-group">

<label for="company\_name">Company Name:</label>

<input type="text" class="form-control" id="company\_name"

name="company\_name" required>

</div>

<div class="form-group">

<label for="client\_role">Client URL:</label>

<input type="text" class="form-control" id="client\_role"

name="client\_role" required>

</div>

</div>

<div class="col-md-6">

<div class="form-group">

<label for="stars\_count">Stars Count (1-5):</label>

<select class="form-control" id="stars\_count" name="stars\_count"

required>

<option value="" disabled selected>Select stars</option>

<option value="1">1 Star</option>

<option value="2">2 Stars</option>

<option value="3">3 Stars</option>

<option value="4">4 Stars</option>

<option value="5">5 Stars</option>

</select>

</div>

<div class="form-group">

<label for="client\_image">Client Image:</label>

<input type="file" class="form-control" id="client\_image"

name="client\_image" accept="image/\*">

</div>

</div>

</div>

<label for="review\_content">Review Content:</label>

<textarea class="form-control" id="review\_content" name="review\_content" required></textarea>

<button type="submit" class="btn btn-primary mt-3">Submit Review</button>

</form>

</div>

</div>

<div class="row mt-3" id="reviewsContainer">

<div class="row mt-3">

<div class="col-12">

<table class="table table-bordered" id="reviewsContainer">

<thead>

<tr>

<th>Company Name</th>

<th>Company URL</th>

<th>Review Content</th>

<th>Stars Count</th>

<th>Client Image</th>

<th>Actions</th>

</tr>

</thead>

<tbody>

</tbody>

</table>

</div>

</div>

</div>

</div>

</div>

</div>

<hr>

<div class="container-fluid mt-5" id="seo-package">

<div class="row">

<div class="col-12">

<div class="page-title-box">

<h4 class="mb-sm-0 font-size-18">SEO Section</h4>

</div>

<div class="row mb-3 align-items-center">

<input type="hidden" id="parent\_id" name="parent\_id" value="{{ $page->id }}">

<div class="col-3">

<label for="seo\_name">Title:</label>

<input type="text" class="form-control" id="seo\_name" name="seo\_name"

value="{{ $seo->title ?? '' }}" required>

<label for="seo\_image">Image:</label>

<input type="file" class="form-control" id="seo\_image" name="seo\_image"

accept="image/\*">

<input type="hidden" id="seo\_id" name="seo\_id" value="{{ $seo->id ?? '' }}">

<img id="imagePreview"

src="{{ $seo && $seo->image ? asset('/storage/' . $seo->image) : asset('/path/to/default/image.png') }}"

alt="Uploaded Image" width="100" style="margin-top: 10px;">

</div>

<div class="col-8">

<label for="seo\_content">Content:</label>

<div id="editor2">{!! $seo->content ?? '' !!}</div>

<input type="hidden" id="seo\_content" name="seo\_content"

value="{{ $seo->content ?? '' }}" required>

</div>

</div>

<button type="button" id="saveButton2" class="btn btn-primary mt-3">Save</button>

</div>

</div>

</div>

<hr>

<div class="container-fluid mt-5" id="price-package">

<div class="row">

<div class="col-12">

<div class="page-title-box">

<h4 class="mb-sm-0 font-size-18">Pricing Package</h4>

</div>

<div class="row mb-3 align-items-center">

<div class="table-responsive">

<table class="table mt-3">

<thead>

<tr>

@if (count($price) > 0)

@foreach ($price[0] as $key => $val)

@php

if (

in\_array($key, [

'id',

'section\_id',

'parent\_id',

'updated\_at',

'created\_at',

])

) {

continue;

}

@endphp

<th>{{ ucfirst($key) }}</th>

@endforeach

@endif

<th>Actions</th>

</tr>

</thead>

<tbody id="segmentsTableBody">

@if (isset($price))

@foreach ($price as $item)

<tr data-id="{{ $item->id }}">

<td>{{ $item->future }}</td>

<td>{{ $item->basic }}</td>

<td>{{ $item->medium }}</td>

<td>{{ $item->enterprice }}</td>

<td>{{ $item->advanced }}</td>

<td>

<button class="edit-row btn btn-primary">Edit</button>

</td>

</tr>

@endforeach

@endif

<tr>

<td><input type="text" class="future" placeholder="Future"></td>

<td><input type="text" class="basic" placeholder="Basic"></td>

<td><input type="text" class="medium" placeholder="Medium"></td>

<td><input type="text" class="enterprice" placeholder="Enterprise"></td>

<td><input type="text" class="advanced" placeholder="Advanced"></td>

<td>

<input type="hidden" id="parent\_id" name="parent\_id"

value="{{ $page->id }}">

<button id="add-row" class="btn btn-success">Add Row</button>

</td>

</tr>

</tbody>

</table>

</div>

</div>

</div>

</div>

</div>

<hr>

<div class="container-fluid mt-5" id="payment-terms">

<div class="row">

<div class="col-12">

<div class="page-title-box">

<h4 class="mb-sm-0 font-size-18">Payment Terms Section</h4>

</div>

<div class="row mb-3 align-items-center">

<div class="col-12">

<form id="paymentsection">

@csrf

<input type="hidden" id="parent\_id" name="parent\_id" value="{{ $page->id }}">

<input type="hidden" id="paymentTermId" name="paymentTermId"

value="{{ $paymentTerm->id ?? '' }}"> <!-- Use your actual ID here -->

<label for="payment\_content">Payment Content:</label>

<div id="editor">{!! $paymentTerm->content ?? '' !!}</div>

<input type="hidden" id="payment\_content" name="payment\_content" required>

<button type="button" id="saveButton" class="btn btn-primary mt-3">Save</button>

</form>

</div>

</div>

</div>

</div>

</div>

<hr>

<div class="container-fluid mt-5" id="why-choose">

<div class="row">

<div class="col-12">

<div class="page-title-box">

<h4 class="mb-sm-0 font-size-18">Why Choose Section</h4>

</div>

<div class="row mb-3 align-items-center">

<input type="hidden" id="parent\_id" name="parent\_id" value="{{ $page->id }}">

<div class="col-3">

<label for="why\_name">Title:</label>

<input type="text" class="form-control" id="why\_name" name="why\_name"

value="{{ $why->title ?? '' }}" required>

<label for="why\_image">Image:</label>

<input type="file" class="form-control" id="why\_image" name="why\_image"

accept="image/\*">

<input type="hidden" id="why\_id" name="why\_id" value="{{ $why->id ?? '' }}">

<img id="why-imagePreview"

src="{{ $why && $why->image ? asset('/storage/' . $why->image) : asset('/path/to/default/image.png') }}"

alt="Uploaded Image" width="100" style="margin-top: 10px;">

</div>

<div class="col-8">

<!-- <label for="why\_content">Content:</label> -->

<!-- <div id="editor8">{!! $why->content ?? '' !!}</div> -->

<textarea id="why\_content" name="why\_content"

required>{{ $why->content ?? '' }}

</textarea>

</div>

</div>

<button type="button" id="saveButton3" class="btn btn-primary mt-3">Save</button

</div>

</div>

</div>

</div>

<hr>

<div class="container-fluid mt-5" id="benefits-of">

<div class="row">

<div class="col-12">

<div class="page-title-box">

<h4 class="mb-sm-0 font-size-18">benefitof Section</h4>

</div>

<div class="row mb-3 align-items-center">

<input type="hidden" id="parent\_id" name="parent\_id" value="{{ $page->id }}">

<div class="col-5">

<label for="benefitof\_name">Title:</label>

<input type="text" class="form-control" id="benefitof\_name" name="benefitof\_name"

value="{{ $benefitof->title ?? '' }}" required>

<label for="benefitof\_image">Image:</label>

<input type="file" class="form-control" id="benefitof\_image" name="benefitof\_image"

accept="image/\*">

<input type="hidden" id="benefitof\_id" name="benefitof\_id" value="{{ $benefitof->id ?? '' }}">

<img id="benefit-imagePreview"

src="{{ $benefitof && $benefitof->image ? asset('/storage/' . $benefitof->image) : asset('/path/to/default/image.png') }}"

alt="Uploaded Image" width="400" style="margin-top: 10px; ">

</div>

<div class="col-7">

<div class="">

<!-- Content 1 -->

<label for="content1">Content 1:</label>

<textarea class="form-control" id="content1" rows="4" name="content1">{{ $benefitof->content1 ?? '' }}</textarea>

</div>

<div class="">

<!-- Content 2 -->

<label for="content2">Content 2:</label>

<textarea class="form-control" id="content2" rows="4" name="content2">{{ $benefitof->content2 ?? '' }}</textarea>

</div>

<div class="">

<!-- Content 3 -->

<label for="content3">Content 3:</label>

<textarea class="form-control" id="content3" rows="4" name="content3">{{ $benefitof->content3 ?? '' }}</textarea>

</div>

<div class="">

<!-- Content 4 -->

<label for="content4">Content 4:</label>

<textarea class="form-control" id="content4" rows="4" name="content4">{{ $benefitof->content4 ?? '' }}</textarea>

</div>

</div>

</div>

<button type="button" id="savebenefitButton" class="btn btn-primary mt-3">Save</button>

</div>

</div>

</div>

<hr>

<div class="container-fluid mt-5" id="how-our-websites">

<div class="row">

<div class="col-12">

<div class="page-title-box">

<h4 class="mb-sm-0 font-size-18">How Our Websites</h4>

</div>

<div class="row mb-3 align-items-center">

<input type="hidden" id="parent\_id" name="parent\_id" value="{{ $page->id }}">

<div class="col-6">

<label for="howOurWebsites\_title">Title:</label>

<input type="text" class="form-control" id="howOurWebsites\_title" name="howOurWebsites\_title"

value="{{ $howOurWebsites->title ?? '' }}" required>

</div>

<div id="cards-container" style="cursor:move;">

<label for="cards" class="form-label">Cards</label>

<button type="button" id="add-card" class="btn btn-sm btn-primary mb-2 mt-3">Add1 Card</button>

<!-- <div class="drag-instruction mb-3">

<p><strong>Instructions for Drag and Drop:</strong></p>

<ul>

<li>To reorder the cards, click and hold the card you wish to move.</li>

<li>Drag it to your desired position.</li>

<li>Release the card to drop it in the new position.</li>

<li>Once you are happy with the order, remember to save your changes.</li>

</ul>

</div> -->

<div class="row row-cols-1 row-cols-md-3 g-3">

@foreach(old('cards', $howOurWebsites->cards ?? []) as $index => $card)

@if($card!=null)

<div class="col mb-3" data-index="{{ $index }}">

<div class="card p-3 border">

<div class="mb-2">

<label>Icon (Enter Font Awesome Class)</label>

<div class="input-group">

<!-- Direct input for the Font Awesome icon class -->

<input type="text" name="cards[{{ $index }}][icon]" id="icon-input-{{ $index }}" class="form-control" placeholder="e.g. fa-home, fa-heart, etc." value="{{ $card['icon'] ?? '' }}">

<span class="input-group-text">

<!-- Display the selected icon -->

<i id="icon-preview-{{ $index }}" class="fa {{ $card['icon'] ?? 'fa-home' }}"></i>

</span>

</div>

</div>

<div class="mb-2">

<label>Title</label>

<input type="text" id="howOurWebsitescards" name="cards[{{ $index }}][title]" class="form-control" value="{{ $card['title'] }}">

</div>

<div class="mb-2">

<label>Content</label>

<textarea rows="4" name="cards[{{ $index }}][content]" class="form-control">{{ $card['content'] }}</textarea>

</div>

<button type="button" class="btn btn-danger btn-sm remove-card">Remove Card</button>

</div>

</div>

@endif

@endforeach

</div>

</div>

</div>

<button type="button" id="savehowourWebbutton" class="btn btn-primary mt-3">Save</button>

</div>

</div>

</div>

<hr>

<div class="container-fluid mt-5" id="take-your-bussiness">

<div class="row">

<div class="col-12">

<div class="page-title-box">

<h4 class="mb-sm-0 font-size-18">Take Your Business</h4>

</div>

<div class="row mb-3 align-items-center">

<input type="hidden" id="parent\_id" name="parent\_id" value="{{ $page->id }}">

<div class="col-6">

<label for="takeYourBusiness\_title">Title:</label>

<input type="text" class="form-control" id="takeYourBusiness\_title" name="takeYourBusiness\_title"

value="{{ $takeYourBusiness->title ?? '' }}" required>

</div>

</div>

<!-- Additional content textarea input -->

<div class="row mb-3 align-items-center">

<div class="col-6">

<label for="takeYourBusiness\_content">Content:</label>

<textarea class="form-control" id="takeYourBusiness\_content" name="takeYourBusiness\_content" rows="4" required>{{ $takeYourBusiness->content ?? '' }}</textarea>

</div>

</div>

<div id="takeYourBusiness-cards-container" style="cursor:move;">

<label for="cards" class="form-label">Cards</label>

<button type="button" id="add-takeYourBusiness-card" class="btn btn-sm btn-primary mb-2 mt-3">Add Card</button>

<div class="row row-cols-1 row-cols-md-3 g-3">

@foreach(old('cards', $takeYourBusiness->cards ?? []) as $index => $card)

@if($card != null)

<div class="col mb-3" data-index="{{ $index }}">

<div class="card p-3 border">

<div class="mb-2">

<label>Icon (Enter Font Awesome Class)</label>

<div class="input-group">

<!-- Direct input for the Font Awesome icon class -->

<input type="text" name="cards2[{{ $index }}][icon]" id="take-icon-input-{{ $index }}" class="form-control" placeholder="e.g. fa-home, fa-heart, etc." value="{{ $card['icon'] ?? '' }}">

<span class="input-group-text">

<!-- Display the selected icon -->

<i id="take-icon-preview-{{ $index }}" class="fa {{ $card['icon'] ?? 'fa-home' }}"></i>

</span>

</div>

</div>

<div class="mb-2">

<label>Title</label>

<input type="text" id="howOurWebsitescards" name="cards2[{{ $index }}][title]" class="form-control" value="{{ $card['title'] }}">

</div>

<div class="mb-2">

<label>Content</label>

<textarea rows="4" name="cards2[{{ $index }}][content]" class="form-control">{{ $card['content'] }}</textarea>

</div>

<button type="button" class="btn btn-danger btn-sm remove-card">Remove Card</button>

</div>

</div>

@endif

@endforeach

</div>

</div>

</div>

</div>

<button type="button" id="saveTakeYourBusinessButton" class="btn btn-primary mt-3">Save</button>

</div>

<hr>

<div class="container-fluid mt-5" id="why-web-siteneed">

<div class="row">

<div class="col-12">

<div class="page-title-box">

<h4 class="mb-sm-0 font-size-18">Why Website Need</h4>

</div>

<div class="row mb-3 align-items-center">

<input type="hidden" id="parent\_id" name="parent\_id" value="{{ $page->id }}">

<div class="col-6">

<label for="whyWebsiteNeed\_title">Title:</label>

<input type="text" class="form-control" id="whyWebsiteNeed\_title" name="whyWebsiteNeed\_title"

value="{{ $whyWebsiteNeed->title ?? '' }}" required>

</div>

<div class="col-6">

<label for="whyWebsiteNeed\_image">Image:</label>

<input type="file" class="form-control" id="whyWebsiteNeed\_image" name="whyWebsiteNeed\_image"

accept="image/\*">

<input type="hidden" id="whyWebsiteNeed\_id" name="whyWebsiteNeed\_id" value="{{ $whyWebsiteNeed->id ?? '' }}">

<img id="whyWebsiteNeed-imagePreview"

src="{{ $whyWebsiteNeed && $whyWebsiteNeed->image ? asset('/storage/' . $whyWebsiteNeed->image) : asset('/path/to/default/image.png') }}"

alt="Uploaded Image" width="100" style="margin-top: 10px;">

</div>

</div>

<div id="whyWebsiteNeed-cards-container" style="cursor:move;">

<label for="cards" class="form-label">Cards</label>

<button type="button" id="add-whyWebsiteNeed-card" class="btn btn-sm btn-primary mb-2 mt-3">Add Card</button>

<div class="row row-cols-1 row-cols-md-3 g-3">

@foreach(old('cards', $whyWebsiteNeed->cards ?? []) as $index => $card)

@if(!is\_null($card))

<div class="col mb-3" data-index="{{ $index }}">

<div class="card p-3 border">

<div class="mb-2">

<label>Icon (Enter Font Awesome Class)</label>

<div class="input-group">

<!-- Direct input for the Font Awesome icon class -->

<input type="text" name="cards3[{{ $index }}][icon]" id="why-icon-input-{{ $index }}" class="form-control" placeholder="e.g. fa-home, fa-heart, etc." value="{{ $card['icon'] ?? '' }}">

<span class="input-group-text">

<!-- Display the selected icon -->

<i id="why-icon-preview-{{ $index }}" class="fa {{ $card['icon'] ?? 'fa-home' }}"></i>

</span>

</div>

</div>

<div class="mb-2">

<label>Title</label>

<input type="text" name="cards3[{{ $index }}][title]" class="form-control" value="{{ $card['title'] }}">

</div>

<div class="mb-2">

<label>Content</label>

<textarea rows="4" name="cards3[{{ $index }}][content]" class="form-control">{{ $card['content'] }}</textarea>

</div>

<button type="button" class="btn btn-danger btn-sm remove-card">Remove Card</button>

</div>

</div>

@endif

@endforeach

</div>

</div>

<button type="button" id="saveWhyWebsiteNeedButton" class="btn btn-primary mt-3">Save</button>

</div>

</div>

</div>

{{-- <div class="container-fluid mt-5">

<div class="row">

<div class="col-12">

<div class="page-title-box">

<h4 class="mb-sm-0 font-size-18">Portfolios</h4>

</div>

<div class="row mb-3 align-items-center">

<form id="createPortfolioForm" method="POST" action="{{ route('portfolios.store') }}"

enctype="multipart/form-data">

@csrf

<div class="row mb-3">

<div class="col">

<select name="type" class="form-control" id="type" required>

<option value="">Select Type</option>

<option value="Static">Static</option>

<option value="Dynamic">Dynamic</option>

<option value="Web Application">Web Application</option>

<option value="Logo">Logo</option>

<option value="Brochure">Brochure</option>

</select>

</div>

<div class="col">

<input type="file" name="image" class="form-control"

accept=".jpg,.png,.jpeg" required>

</div>

<div class="col">

<input type="text" name="link" class="form-control" placeholder="Link"

required>

</div>

<div class="col">

<select name="status" class="form-control" id="portfoliostatus" required>

<option value="">Select Status</option>

<option value="Active">Active</option>

<option value="Inactive">Inactive</option>

</select>

</div>

</div>

<button type="submit" class="btn btn-primary">Add Portfolio</button>

</form>

<!-- Display Portfolios -->

<table class="table mt-4">

<thead>

<tr>

<th>ID</th>

<th>Type</th>

<th>Image</th>

<th>Link</th>

<th>Status</th>

<th>Actions</th>

</tr>

</thead>

<tbody>

@foreach ($portfolios as $portfolio)

<tr data-id="{{ $portfolio->id }}">

<td>{{ $portfolio->id }}</td>

<td>{{ $portfolio->type }}</td>

<td><img src="{{ asset('images/' . $portfolio->image) }}" alt=""

style="width: 50px;"></td>

<td>{{ $portfolio->link }}</td>

<td>{{ $portfolio->status }}</td>

<td>

<button class="btn btn-primary btn-sm editPortfolioModal"

data-id="{{ $portfolio->id }}">Edit</button>

<form action="{{ route('portfolios.destroy', $portfolio->id) }}"

method="POST" class="d-inline">

@csrf

@method('DELETE')

<button class="btn btn-danger btn-sm">Delete</button>

</form>

</td>

</tr>

@endforeach

</tbody>

</table>

</div>

</div>

</div>

</div>

<hr> --}}

</div>

</div>

<div class="modal fade" id="segmentModal" tabindex="-1" aria-labelledby="segmentModalLabel" aria-hidden="true">

<div class="modal-dialog">

<div class="modal-content">

<div class="modal-header">

<h5 class="modal-title" id="segmentModalLabel">Edit Segment</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>

</div>

<div class="modal-body">

<input type="hidden" id="segmentId">

<div class="mb-3">

<label for="segmentTitle" class="form-label">Title</label>

<input type="text" class="form-control" id="segmentTitle" required>

</div>

</div>

<div class="modal-footer">

<button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Close</button>

<button type="button" class="btn btn-primary" onclick="saveSegment()">Save changes</button>

</div>

</div>

</div>

</div>

<div class="modal fade" id="editPortfolioModal" tabindex="-1" aria-labelledby="editPortfolioModalLabel"

aria-hidden="true">

<div class="modal-dialog">

<div class="modal-content">

<form id="editPortfolio" method="POST" action="" enctype="multipart/form-data">

@csrf

@method('PUT')

<div class="modal-header">

<h5 class="modal-title" id="editPortfolioModalLabel">Edit Portfolio</h5>

<button type="button" class="btn-close" data-bs-dismiss="modal"></button>

</div>

<div class="modal-body">

<input type="hidden" name="id" id="editPortfolioId">

<div class="mb-3">

<label for="editType" class="form-label">Type</label>

<select name="type" class="form-control" id="editType" required>

<option value="Static">Static</option>

<option value="Dynamic">Dynamic</option>

<option value="Web Application">Web Application</option>

<option value="Logo">Logo</option>

<option value="Brochure">Brochure</option>

</select>

</div>

<div class="mb-3">

<label for="editImage" class="form-label">Image</label>

<input type="file" class="form-control" name="image" id="editImage">

<img id="existingImage" src="" alt="" style="width: 50px;">

</div>

<div class="mb-3">

<label for="editLink" class="form-label">Link</label>

<input type="text" class="form-control" name="link" id="editLink" required>

</div>

<div class="col">

<label for="editStatus" class="form-label">Status</label>

<select name="status" class="form-control" id="editStatus" required>

<option value="Active">Active</option>

<option value="Inactive">Inactive</option>

</select>

</div>

</div>

<div class="modal-footer">

<button type="button" class="btn btn-secondary" data-bs-dismiss="modal">Close</button>

<button type="submit" class="btn btn-primary">Save changes</button>

</div>

</form>

</div>

</div>

</div>

<link rel="stylesheet" href="https://cdn.ckeditor.com/ckeditor5/43.3.1/ckeditor5.css" />

<script type="importmap">

{

"imports": {

"ckeditor5": "https://cdn.ckeditor.com/ckeditor5/43.3.1/ckeditor5.js",

"ckeditor5/": "https://cdn.ckeditor.com/ckeditor5/43.3.1/"

}

}

</script>

<script src="https://cdnjs.cloudflare.com/ajax/libs/Sortable/1.15.0/Sortable.min.js"></script>

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"></script>

<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>

<script>

document.addEventListener("DOMContentLoaded", function() {

const hash = window.location.hash;

if (hash) {

const targetId = hash.substring(1);

const element = document.getElementById(targetId);

if (element) {

const headerOffset = 100;

const elementPosition = element.getBoundingClientRect().top;

const offsetPosition = elementPosition + window.scrollY - headerOffset;

window.scrollTo({

top: offsetPosition,

behavior: "smooth"

});

} else {

console.warn("Element not found for ID:", targetId);

}

} else {

console.warn("No hash in URL, cannot scroll to an element.");

}

});

</script>

<script>

document.getElementById('recentProjectForm').addEventListener('submit', function(e) {

e.preventDefault();

let formData = new FormData(this);

const projectId = document.getElementById('projectId').value;

const url = projectId ? `{{ route('recent-projects.update', '') }}/${projectId}` :

"{{ route('recent-projects.store') }}";

fetch(url, {

method: 'POST',

headers: {

'X-CSRF-TOKEN': document.querySelector('meta[name="csrf-token"]').getAttribute(

'content')

},

body: formData

})

.then(response => {

if (!response.ok) {

throw new Error('Network response was not ok ' + response.statusText);

}

return response.json();

})

.then(data => {

if (data.success) {

loadProjects();

Toastify({

text: projectId ? "Project updated successfully!" :

"Project added successfully!",

backgroundColor: "#4CAF50",

position: "right",

}).showToast();

} else {

Toastify({

text: "An error occurred: " + data

.message,

backgroundColor: "#f44336",

position: "right",

}).showToast();

}

})

.catch(error => {

console.error("Error:", error);

Toastify({

text: "An unexpected error occurred.",

backgroundColor: "#f44336",

position: "right",

}).showToast();

});

});

function loadProjects() {

const url = "{{ route('recent.projects.index', ['parent\_id' => $page->id]) }}";

fetch(url)

.then(response => response.json())

.then(projects => {

let tableBody = document.querySelector('#recentProjectContainer tbody');

tableBody.innerHTML = '';

const path = "{{asset('/')}}";

projects.forEach(project => {

tableBody.innerHTML += `

<tr>

<td>${project.name}</td>

<td>${project.description}</td>

<td><a href="${project.link}" target="\_blank">${project.link}</a></td>

<td><img src="${path}/storage/project\_images/${project.image}" width="100" /></td>

<td>

<button class="btn btn-warning btn-sm" onclick="editProject(${project.id})">Edit</button>

<button class="btn btn-danger btn-sm" onclick="deleteProject(${project.id})">Delete</button>

</td>

</tr>

`;

});

});

}

loadProjects();

function editProject(id) {

fetch(`{{url('/')}}/admin/recentprojects/${id}`)

.then(response => {

if (response.ok) {

return response.json();

} else {

throw new Error('Failed to fetch project data');

}

})

.then(data => {

if (data.success) {

const project = data.project;

document.getElementById('projectId').value = project.id;

document.getElementById('name').value = project.name;

document.getElementById('description').value = project.description;

document.getElementById('link').value = project.link;

document.getElementById('submitButton').textContent = 'Update Project';

const path = "{{asset('/')}}";

const imageUrl = path+`/storage/project\_images/${project.image}`;

document.getElementById('currentImage').src = imageUrl;

document.getElementById('currentImage').style.display = 'block'; // Show the image

}

})

.catch(error => {

console.error("Error:", error);

});

}

function deleteProject(id) {

if (confirm('Are you sure you want to delete this project?')) {

fetch(`url('/')}}/admin/recent-projects/${id}`, {

method: 'DELETE',

headers: {

'X-CSRF-TOKEN': document.querySelector('meta[name="csrf-token"]').getAttribute('content')

}

})

.then(response => response.json())

.then(data => {

if (data.success) {

loadProjects();

Toastify({

text: "Project deleted successfully!",

backgroundColor: "#f44336",

position: "right",

}).showToast();

}

});

}

}

</script>

<script>

Dropzone.options.logoDropzone = {

paramName: "images",

maxFilesize: 2, // MB

acceptedFiles: ".jpeg,.jpg,.png,.gif",

addRemoveLinks: true,

success: function(file, response) {

console.log("Success! Images uploaded.");

},

error: function(file, response) {

console.log("Error uploading images.");

}

};

function deleteLogo(logoId) {

const url = "{{ route('logos.destroy', ':id') }}".replace(':id', logoId);

fetch(url, {

method: 'DELETE',

headers: {

'X-CSRF-TOKEN': document.querySelector('meta[name="csrf-token"]').getAttribute('content')

}

})

.then(response => response.json())

.then(data => {

if (data.success) {

document.getElementById('logo-' + logoId).remove();

Toastify({

text: "Logo deleted successfully!",

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#DC3545", // Red color for delete

}).showToast();

} else {

console.error('Error deleting logo');

}

})

.catch(error => console.error('Error:', error));

}

</script>

<script>

function saveLogoData() {

const title1 = document.getElementById('title1').value;

const parent\_id = document.getElementById('parent\_id').value;

const title2 = document.getElementById('title2').value;

const formTitle = document.getElementById('form\_title').value;

const url = "{{ route('logos.update') }}";

fetch(url, {

method: 'POST',

headers: {

'Content-Type': 'application/json',

'X-CSRF-TOKEN': document.querySelector('meta[name="csrf-token"]').getAttribute('content')

},

body: JSON.stringify({

title1: title1,

parent\_id: parent\_id,

title2: title2,

form\_title: formTitle

})

})

.then(response => {

if (response.ok) {

return response.json();

} else {

throw new Error('Failed to update logo data');

}

})

.then(data => {

if (data.success) {

Toastify({

text: "Logo data updated successfully!",

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#4CAF50",

}).showToast();

} else {

console.error('Error updating logo data');

Toastify({

text: "Error updating logo data",

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#FF0000",

}).showToast();

}

})

.catch(error => {

console.error('Error:', error);

Toastify({

text: "Error: " + error.message,

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#FF0000",

}).showToast();

});

}

</script>

<script>

$(document).on('click', '.editPortfolio', function() {

var id = $(this).data('id');

$.get('/portfolios/' + id + '/edit', function(data) {

$('#editPortfolioId').val(data.id);

document.getElementById('editType').value = portfolio.type;

$('#editImage').val(data.image);

$('#editLink').val(data.link);

$('#editSortOrder').val(data.sort\_order);

$('#editStatus').val(data.status);

$('#editPortfolioForm').attr('action', '/portfolios/' + id);

$('#editPortfolioModal').modal('show');

});

});

$(document).ready(function() {

$('.editPortfolioModal').on('click', function() {

const portfolioId = $(this).data('id');

const editUrl = `{{ route('portfolios.edit', ':id') }}`.replace(':id', portfolioId);

$.ajax({

url: editUrl,

type: 'GET',

dataType: 'json',

success: function(portfolio) {

$('#editPortfolioId').val(portfolio.id);

$('#editType').val(portfolio.type);

$('#editLink').val(portfolio.link);

$('#editStatus').val(portfolio.status);

$('#existingImage').attr('src', `/images/${portfolio.image}`).attr(

'alt', portfolio.type);

$('#editPortfolio').attr('action', `/portfolios/${portfolio.id}`);

$('#editPortfolioModal').modal('show');

},

error: function(xhr, status, error) {

console.error('Error fetching portfolio data:', xhr.responseText);

}

});

});

});

</script>

<script>

$(document).ready(function() {

initializeReviews();

initializeGetTouchUs();

});

function initializeReviews() {

fetchReviews();

const csrfToken = $('meta[name="csrf-token"]').attr('content');

function fetchReviews() {

const url = "{{ route('client.reviews.index', ['parent\_id' => $page->id]) }}";

$.get(url, function(data) {

$('#reviewsContainer tbody').empty(); // Clear existing reviews

data.forEach(function(review) {

const path = "{{asset('/')}}";

$('#reviewsContainer tbody').append(`

<tr class="review" data-id="${review.id}">

<td>${review.company\_name}</td>

<td>${review.client\_role}</td>

<td>${review.review\_content}</td>

<td>${review.stars\_count}</td>

<td><img src="${path}/storage/${review.client\_image}" alt="${review.client\_name}" style="width: 60px;"></td>

<td><button class="btn btn-danger deleteReview">Delete</button></td>

</tr>

`);

});

});

}

$('#reviewForm').on('submit', function(event) {

event.preventDefault(); // Prevent default form submission

let formData = new FormData(this);

formData.append('\_token', csrfToken);

$.ajax({

url: `{{ url('/') }}/admin/client-reviews`,

type: 'POST',

data: formData,

contentType: false,

processData: false,

success: function(response) {

fetchReviews(); // Refresh reviews

$('#reviewForm')[0].reset(); // Reset form

Toastify({

text: response.success,

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#4CAF50",

}).showToast();

},

error: function(xhr) {

let error = xhr.responseJSON.message || "An error occurred";

Toastify({

text: "Failed to add review: " + error,

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#F44336",

}).showToast();

}

});

});

$(document).on('click', '.deleteReview', function() {

let reviewId = $(this).closest('.review').data('id');

$.ajax({

url: `{{ url('/') }}/admin/client-reviews/${reviewId}`,

type: 'DELETE',

data: {

\_token: csrfToken // Send CSRF token with the delete request

},

success: function(response) {

fetchReviews(); // Refresh reviews

Toastify({

text: response.success,

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#F44336",

}).showToast();

},

error: function(xhr) {

let error = xhr.responseJSON.message || "An error occurred";

Toastify({

text: "Failed to delete review: " + error,

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#F44336",

}).showToast();

}

});

});

}

function initializeGetTouchUs() {

let originalValues = {};

fetchRecord(); // Fetch the existing record data

function fetchRecord() {

const url = "{{ route('getTouchUs.index', ['parent\_id' => $page->id]) }}";

$.get(url, function(data) {

if (data.length > 0) {

let record = data[0];

$('.recordId').val(record.id);

$('.title1').val(record.title1);

$('.title2').val(record.title2);

$('.title3').val(record.title3);

$('.years').val(record.years);

$('.clients').val(record.clients);

$('.projects').val(record.projects);

$('.reviews').val(record.reviews);

originalValues = {

title1: record.title1,

title2: record.title2,

years: record.years,

clients: record.clients,

projects: record.projects,

reviews: record.reviews,

};

}

}).fail(function() {

Toastify({

text: "Failed to load record.",

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#F44336",

}).showToast();

});

}

$('.get-touch-us-form').on('change', function() {

checkChanges(); // Check for changes when any input field changes

});

function checkChanges() {

let id = $('.recordId').val();

let currentData = {

title1: $('.title1').val(),

parent\_id: $('.parent\_id').val(),

title2: $('.title2').val(),

title3: $('.title3').val(),

years: $('.years').val(),

clients: $('.clients').val(),

projects: $('.projects').val(),

reviews: $('.reviews').val(),

};

if (JSON.stringify(originalValues) !== JSON.stringify(currentData)) {

saveChanges(currentData); // Save changes if there are any

}

}

function saveChanges(data) {

data.\_token = $('meta[name="csrf-token"]').attr('content');

var id = $('.parent\_id').val();

const url = "{{ route('getTouchUs.update', [':id']) }}".replace(':id', id);

$.ajax({

url: url,

type: 'PUT',

data: data,

success: function(response) {

Toastify({

text: "Record updated successfully!",

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#4CAF50",

}).showToast();

},

error: function(xhr) {

let error = xhr.responseJSON.message || "An error occurred";

Toastify({

text: "Failed to update: " + error,

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#F44336",

}).showToast();

}

});

}

}

</script>

<script>

$(document).ready(function() {

$('#add-row').click(function() {

var future = $('.future').val();

var parent\_id = $('.parent\_id').val();

var basic = $('.basic').val();

var medium = $('.medium').val();

var enterprice = $('.enterprice').val();

var advanced = $('.advanced').val();

$.ajax({

url: `{{ route('price.store') }}`, // Update with your actual route

method: 'POST',

data: {

future: future,

basic: basic,

medium: medium,

parent\_id: parent\_id,

enterprice: enterprice,

advanced: advanced,

\_token: '{{ csrf\_token() }}' // CSRF token for security

},

success: function(response) {

$('#segmentsTableBody').append(`

<tr data-id="${response.id}">

<td>${response.future}</td>

<td>${response.basic}</td>

<td>${response.medium}</td>

<td>${response.enterprice}</td>

<td>${response.advanced}</td>

<td>

<button class="edit-row btn btn-primary">Edit</button>

</td>

</tr>

`);

$('.future').val('');

$('.basic').val('');

$('.medium').val('');

$('.enterprice').val('');

$('.advanced').val('');

Toastify({

text: "Segment added successfully!",

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#4CAF50", // Green for success

}).showToast();

},

error: function(xhr) {

var errorMessage = xhr.responseJSON.message ||

'An error occurred while adding the segment.';

Toastify({

text: errorMessage,

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#F44336", // Red for error

}).showToast();

}

});

});

$(document).on('click', '.edit-row', function() {

var row = $(this).closest('tr');

var id = row.data('id');

$('.future').val(row.find('td:nth-child(1)').text());

$('.basic').val(row.find('td:nth-child(2)').text());

$('.medium').val(row.find('td:nth-child(3)').text());

$('.enterprice').val(row.find('td:nth-child(4)').text());

$('.advanced').val(row.find('td:nth-child(5)').text());

$('#add-row').text('Update Row').off('click').on('click', function() {

const url = "{{ route('price.update2', [':id']) }}".replace(':id', id);

$.ajax({

url: url,

method: 'PUT',

data: {

future: $('.future').val(),

basic: $('.basic').val(),

medium: $('.medium').val(),

enterprice: $('.enterprice').val(),

advanced: $('.advanced').val(),

\_token: '{{ csrf\_token() }}' // CSRF token for security

},

success: function(response) {

row.find('td:nth-child(1)').text(response.future);

row.find('td:nth-child(2)').text(response.basic);

row.find('td:nth-child(3)').text(response.medium);

row.find('td:nth-child(4)').text(response.enterprice);

row.find('td:nth-child(5)').text(response.advanced);

$('.future').val('');

$('.basic').val('');

$('.medium').val('');

$('.enterprice').val('');

$('.advanced').val('');

$('#add-row').text('Add Row').off('click').on('click',

function() {});

Toastify({

text: "Segment updated successfully!",

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#4CAF50", // Green for success

}).showToast();

},

error: function(xhr) {

var errorMessage = xhr.responseJSON.message ||

'An error occurred while updating the segment.';

Toastify({

text: errorMessage,

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#F44336", // Red for error

}).showToast();

}

});

});

});

});

function editSegment(id) {

fetch(`/sections/${id}`)

.then(response => response.json())

.then(section => {

document.getElementById('segmentId').value = section.id; // Hidden input field for ID

document.getElementById('segmentTitle').value = section.title; // Title input field

const segmentModal = new bootstrap.Modal(document.getElementById('segmentModal'));

segmentModal.show(); // Open the modal

})

.catch(error => {

console.error('Failed to fetch section details:', error);

Toastify({

text: "Failed to fetch section details.",

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#F44336", // Red for error

}).showToast();

});

}

function saveSegment() {

const segmentId = document.getElementById('segmentId').value;

const segmentTitle = document.getElementById('segmentTitle').value;

const token = document.querySelector('meta[name="csrf-token"]').getAttribute('content');

fetch(`/sections/${segmentId}`, {

method: 'PUT',

headers: {

'Content-Type': 'application/json',

'X-CSRF-TOKEN': token

},

body: JSON.stringify({

title: segmentTitle

})

})

.then(response => response.json())

.then(data => {

if (data.success) {

document.querySelector(`#segment-${segmentId} td:nth-child(2)`).innerText = segmentTitle;

Toastify({

text: "Segment updated successfully!",

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#4CAF50", // Green for success

}).showToast();

const segmentModal = bootstrap.Modal.getInstance(document.getElementById('segmentModal'));

segmentModal.hide();

} else {

Toastify({

text: "Failed to update segment.",

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#F44336", // Red for error

}).showToast();

}

})

.catch(error => {

console.error('Error updating segment:', error);

Toastify({

text: "Error updating segment.",

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#F44336", // Red for error

}).showToast();

});

}

function deleteSegment(id) {

const token = document.querySelector('meta[name="csrf-token"]').getAttribute('content');

if (!confirm('Are you sure you want to delete this segment?')) return;

const url = "{{ route('section.delete', [':id']) }}".replace(':id', id);

fetch(url, {

method: 'DELETE',

headers: {

'X-CSRF-TOKEN': token

}

})

.then(response => response.json())

.then(data => {

if (data.success) {

document.getElementById(`segment-${id}`).remove(); // Remove the row

Toastify({

text: "Section deleted successfully!",

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#4CAF50", // Green for success

}).showToast();

} else {

Toastify({

text: 'Failed to delete Section.',

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#F44336", // Red for error

}).showToast();

}

});

}

document.addEventListener('DOMContentLoaded', function() {

const segmentsTableBody = document.getElementById('segmentsTableBody');

let draggedRow = null;

segmentsTableBody.addEventListener('dragstart', function(e) {

if (e.target.nodeName === 'TR') {

draggedRow = e.target;

draggedRow.classList.add('highlight');

}

});

segmentsTableBody.addEventListener('dragover', function(e) {

e.preventDefault();

const currentRow = e.target.closest('tr');

if (currentRow && currentRow !== draggedRow) {

currentRow.classList.add('highlight');

}

});

segmentsTableBody.addEventListener('dragleave', function(e) {

const currentRow = e.target.closest('tr');

if (currentRow) {

currentRow.classList.remove('highlight');

}

});

segmentsTableBody.addEventListener('drop', function(e) {

e.preventDefault();

const targetRow = e.target.closest('tr');

if (targetRow && targetRow !== draggedRow) {

const isBelow = (targetRow.getBoundingClientRect().top < draggedRow

.getBoundingClientRect().top);

if (isBelow) {

segmentsTableBody.insertBefore(draggedRow, targetRow);

segmentsTableBody.insertBefore(targetRow, draggedRow.nextSibling);

} else {

segmentsTableBody.insertBefore(targetRow, draggedRow);

segmentsTableBody.insertBefore(draggedRow, targetRow.nextSibling);

}

const newOrderId = targetRow.id.split('-')[1];

const oldOrderId = draggedRow.id.split('-')[1];

updateOrder(oldOrderId, newOrderId);

} else {

console.log("Invalid Drop Target");

}

if (draggedRow) {

draggedRow.classList.remove('highlight');

draggedRow = null;

}

});

function updateOrder(draggedId, targetId) {

fetch("{{ route('sortorder.update') }}", {

method: "POST",

headers: {

"Content-Type": "application/json",

"X-CSRF-TOKEN": '{{ csrf\_token() }}'

},

body: JSON.stringify({

draggedId: draggedId,

targetId: targetId

})

})

.then(response => {

if (!response.ok) {

return response.json().then(error => {

console.error("AJAX Error:", error);

Toastify({

text: error.message,

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#F44336", // Red for error

}).showToast();

});

}

return response.json();

})

.then(data => {

Toastify({

text: "Order updated successfully!",

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#4CAF50", // Green for success

}).showToast();

window.location.reload();

})

.catch(error => {

console.error('Failed to update order:', error);

Toastify({

text: "Failed to update order: " + error,

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#F44336",

}).showToast();

});

}

});

</script>

<script>

const editor = new FroalaEditor('#editor', {

toolbarButtons: [

'undo', 'redo', '|',

'bold', 'italic', 'underline', '|',

'paragraphFormat', 'align', 'formatOL', 'formatUL', '|',

'insertLink', 'insertImage', '|',

'clearFormatting', 'html'

],

height: 300

});

const saveContent = () => {

const paymentContent = editor.html.get();

const parent\_id = document.getElementById('parent\_id').value;

document.getElementById('payment\_content').value = paymentContent;

const paymentTermId = document.getElementById('parent\_id').value; // If you're updating, get the ID

const url = `{{ route('payment-terms.update', '') }}/${paymentTermId}`;

fetch(url, {

method: 'POST',

headers: {

'X-CSRF-TOKEN': document.querySelector('meta[name="csrf-token"]').getAttribute('content'),

'Content-Type': 'application/json', // Set content type to JSON

},

body: JSON.stringify({

content: paymentContent,

parent\_id: parent\_id

})

})

.then(response => {

if (!response.ok) {

throw new Error('Network response was not ok ' + response.statusText);

}

return response.json();

})

.then(data => {

if (data.success) {

Toastify({

text: "Payment term updated successfully!",

backgroundColor: "#4CAF50",

position: "right",

}).showToast();

} else {

Toastify({

text: "An error occurred: " + data.message,

backgroundColor: "#f44336",

position: "right",

}).showToast();

}

})

.catch(error => {

console.error("Error:", error);

Toastify({

text: "An unexpected error occurred.",

backgroundColor: "#f44336",

position: "right",

}).showToast();

});

};

document.getElementById('saveButton').addEventListener('click', saveContent);

// editor.events.on('blur', saveContent);

</script>

<script>

const editor2 = new FroalaEditor('#editor2', {

toolbarButtons: [

'undo', 'redo', '|',

'bold', 'italic', 'underline', '|',

'paragraphFormat', 'align', 'formatOL', 'formatUL', '|',

'insertLink', 'insertImage', '|',

'clearFormatting', 'html'

],

height: 300 // Adjust height as needed

});

document.getElementById('seo\_image').addEventListener('change', function(event) {

const file = event.target.files[0];

const reader = new FileReader();

reader.onload = function(e) {

const imgPreview = document.getElementById('imagePreview');

imgPreview.src = e.target.result; // Set the source of the image to the file's result

};

if (file) {

reader.readAsDataURL(file); // Read the file as a Data URL

}

});

const saveContent2 = () => {

const seoContent = editor2.html.get();

document.getElementById('seo\_content').value = seoContent;

const seoId = document.getElementById('seo\_id').value; // Get the ID for updating

const parent\_id = document.getElementById('parent\_id').value; // Get the ID for updating

const title = document.getElementById('seo\_name').value; // Get the title value

const imageFile = document.getElementById('seo\_image').files[0]; // Get the image file

const formData = new FormData();

formData.append('title', title);

if (imageFile) {

formData.append('seo\_image', imageFile); // Only append if an image is selected

}

formData.append('content', seoContent);

formData.append('seo\_id', seoId); // Include the ID for updating

formData.append('parent\_id', parent\_id); // Include the ID for updating

const url = `{{ route('seo-sections.update', '') }}/${parent\_id}`;

fetch(url, {

method: 'POST',

headers: {

'X-CSRF-TOKEN': document.querySelector('meta[name="csrf-token"]').getAttribute(

'content'), // Add CSRF token

},

body: formData, // Send FormData object

})

.then(response => {

if (!response.ok) {

throw new Error('Network response was not ok ' + response.statusText);

}

return response.json();

})

.then(data => {

if (data.success) {

Toastify({

text: "SEO Section updated successfully!",

backgroundColor: "#4CAF50",

position: "right",

}).showToast();

} else {

Toastify({

text: "An error occurred: " + data.message,

backgroundColor: "#f44336",

position: "right",

}).showToast();

}

})

.catch(error => {

console.error("Error:", error);

Toastify({

text: "An unexpected error occurred.",

backgroundColor: "#f44336",

position: "right",

}).showToast();

});

};

document.getElementById('saveButton2').addEventListener('click', saveContent2);

</script>

<script type="module">

let editorInstance;

import {

ClassicEditor,

Essentials,

Bold,

Italic,

Font,

Paragraph,

Underline,

Strikethrough,

Link,

List,

BlockQuote,

Image,

ImageUpload,

Table,

Alignment

} from 'ckeditor5';

ClassicEditor

.create( document.querySelector( '#why\_content' ), {

plugins: [

Essentials, Bold, Italic, Font, Paragraph, Underline, Strikethrough,

Link, List, BlockQuote, Image, ImageUpload, Table, Alignment

],

toolbar: [

'undo', 'redo', '|', 'bold', 'italic', 'underline', 'strikethrough', '|',

'fontSize', 'fontFamily', 'fontColor', 'fontBackgroundColor', '|',

'link', 'bulletedList', 'numberedList', '|',

'blockQuote', 'insertTable', '|', 'alignment', 'imageUpload'

],

image: {

toolbar: [

'imageTextAlternative', 'imageStyle:full', 'imageStyle:side'

]

},

table: {

contentToolbar: [ 'tableColumn', 'tableRow', 'mergeTableCells' ]

}

} )

.then( editor => {

editorInstance = editor;

// Editor ready

} )

.catch( error => {

console.error( error );

} );

document.getElementById('why\_image').addEventListener('change', function(event) {

const file = event.target.files[0];

const reader = new FileReader();

reader.onload = function(e) {

const imgPreview = document.getElementById('why-imagePreview');

imgPreview.src = e.target.result; // Set the source of the image to the file's result

};

if (file) {

reader.readAsDataURL(file); // Read the file as a Data URL

}

});

const saveContent3 = () => {

const whyContent = editorInstance.getData();

const whyId = document.getElementById('why\_id').value; // Get the ID for updating

const parent\_id = document.getElementById('parent\_id').value; // Get the ID for updating

const title = document.getElementById('why\_name').value; // Get the title value

const imageFile = document.getElementById('why\_image').files[0]; // Get the image file

const formData = new FormData();

formData.append('title', title);

if (imageFile) {

formData.append('why\_image', imageFile); // Only append if an image is selected

}

formData.append('content', whyContent);

formData.append('why\_id', whyId); // Include the ID for updating

formData.append('parent\_id', parent\_id); // Include the ID for updating

const url = `{{ route('why-choose.update', '') }}/${parent\_id}`;

fetch(url, {

method: 'POST',

headers: {

'X-CSRF-TOKEN': document.querySelector('meta[name="csrf-token"]').getAttribute(

'content'), // Add CSRF token

},

body: formData, // Send FormData object

})

.then(response => {

if (!response.ok) {

throw new Error('Network response was not ok ' + response.statusText);

}

return response.json();

})

.then(data => {

if (data.success) {

Toastify({

text: "why Section updated successfully!",

backgroundColor: "#4CAF50",

position: "right",

}).showToast();

} else {

Toastify({

text: "An error occurred: " + data.message,

backgroundColor: "#f44336",

position: "right",

}).showToast();

}

})

.catch(error => {

console.error("Error:", error);

Toastify({

text: "An unexpected error occurred.",

backgroundColor: "#f44336",

position: "right",

}).showToast();

});

};

document.getElementById('saveButton3').addEventListener('click', saveContent3);

</script>

<script>

$(document).ready(function() {

$('#re\_content').on('change', function() {

const newText = $(this).val();

const parent\_id = document.getElementById('parent\_id').value;

$.ajax({

url: "{{ route('project.update.route') }}",

type: 'POST',

data: {

content: newText,

parent\_id: parent\_id,

\_token: '{{ csrf\_token() }}'

},

success: function(response) {

// Show a success message or handle successful response

Toastify({

text: "Content updated successfully!",

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#4CAF50", // Green success color

}).showToast();

},

error: function(xhr) {

// Show an error message if the update fails

const errorMsg = xhr.responseJSON?.message || "Update failed. Please try again.";

Toastify({

text: errorMsg,

duration: 3000,

close: true,

gravity: "top",

position: "right",

backgroundColor: "#FF0000", // Red error color

}).showToast();

}

});

});

});

</script>

</script>

<script>

// Handle image preview when an image is selected

document.getElementById('benefitof\_image').addEventListener('change', function(event) {

const file = event.target.files[0];

const reader = new FileReader();

reader.onload = function(e) {

const imgPreview = document.getElementById('benefit-imagePreview');

imgPreview.src = e.target.result; // Set the source of the image to the file's result

};

if (file) {

reader.readAsDataURL(file); // Read the file as a Data URL

}

});

// Function to save content with textareas

// Function to save content with textareas

const savebenefitContent = () => {

const benefitofContent1 = document.getElementById('content1').value; // Get content1 value

const benefitofContent2 = document.getElementById('content2').value; // Get content2 value

const benefitofContent3 = document.getElementById('content3').value; // Get content3 value

const benefitofContent4 = document.getElementById('content4').value; // Get content4 value

const benefitofId = document.getElementById('benefitof\_id').value; // Get the ID for updating

const parent\_id = document.getElementById('parent\_id').value; // Get the parent ID

const title = document.getElementById('benefitof\_name').value; // Get the title value

const imageFile = document.getElementById('benefitof\_image').files[0]; // Get the image file

const formData = new FormData();

formData.append('title', title);

// Only append the image if it's selected

if (imageFile) {

formData.append('benefitof\_image', imageFile);

}

// Append content1, content2, content3, content4 to the form data

formData.append('content1', benefitofContent1);

formData.append('content2', benefitofContent2);

formData.append('content3', benefitofContent3);

formData.append('content4', benefitofContent4);

formData.append('benefitof\_id', benefitofId); // Include the ID for updating

formData.append('parent\_id', parent\_id); // Include the parent ID for updating

const url = `{{ route('benefitof-sections.update', '') }}/${parent\_id}`;

// Make the fetch request to save data

fetch(url, {

method: 'POST',

headers: {

'X-CSRF-TOKEN': document.querySelector('meta[name="csrf-token"]').getAttribute('content'), // Add CSRF token

},

body: formData, // Send FormData object

})

.then(response => {

if (!response.ok) {

throw new Error('Network response was not ok ' + response.statusText);

}

return response.json();

})

.then(data => {

if (data.success) {

Toastify({

text: "benefitof Section updated successfully!",

backgroundColor: "#4CAF50",

position: "right",

}).showToast();

} else {

Toastify({

text: "An error occurred: " + data.message,

backgroundColor: "#f44336",

position: "right",

}).showToast();

}

})

.catch(error => {

console.error("Error:", error);

Toastify({

text: "An unexpected error occurred.",

backgroundColor: "#f44336",

position: "right",

}).showToast();

});

};

// Add event listener to save the content when the save button is clicked

document.getElementById('savebenefitButton').addEventListener('click', savebenefitContent);

</script>

<script>

document.addEventListener('DOMContentLoaded', function () {

const maxCards = 20;

const cardsContainer = document.getElementById('cards-container');

const addCardButton = document.getElementById('add-card');

// Initialize SortableJS

const sortable = new Sortable(cardsContainer.querySelector('.row'), {

handle: '.card',

animation: 150,

onEnd(evt) {

updateCardIndexes();

},

onStart(evt) {

evt.item.classList.add('dragging');

}

});

// Add new card

addCardButton.addEventListener('click', function () {

if (cardsContainer.querySelectorAll('.col').length >= maxCards) {

Swal.fire({

icon: 'warning',

title: 'Limit Reached',

text: 'You can only add a maximum of 6 cards.',

confirmButtonText: 'OK',

});

return;

}

const newIndex = cardsContainer.querySelectorAll('.col').length;

const cardHtml = `

<div class="col mb-3" data-index="${newIndex}">

<div class="card p-3 border">

<div class="mb-2">

<label>Icon (Enter Font Awesome Class)</label>

<div class="input-group">

<input type="text" name="cards[${newIndex}][icon]" id="icon-input-${newIndex}" class="form-control" placeholder="e.g. fa-home, fa-heart, etc.">

<span class="input-group-text">

<i id="icon-preview-${newIndex}" class="fa fa-home"></i>

</span>

</div>

</div>

<div class="mb-2">

<label>Title</label>

<input type="text" name="cards[${newIndex}][title]" class="form-control">

</div>

<div class="mb-2">

<label>Content</label>

<textarea rows="4" name="cards[${newIndex}][content]" class="form-control"></textarea>

</div>

<button type="button" class="btn btn-danger btn-sm remove-card">Remove Card</button>

</div>

</div>

`;

cardsContainer.querySelector('.row').insertAdjacentHTML('beforeend', cardHtml);

updateCardIndexes();

});

// Remove card

cardsContainer.addEventListener('click', function (e) {

if (e.target.classList.contains('remove-card')) {

const card = e.target.closest('.col');

card.remove();

updateCardIndexes(); // Re-index cards after removal

}

});

// Handle icon input changes

cardsContainer.addEventListener('input', function (e) {

if (e.target.matches('[id^="icon-input-"]')) {

const cardIndex = e.target.id.split('-')[2]; // Get the card index from ID

updateIconPreview(cardIndex);

}

});

// Function to update the icon preview

function updateIconPreview(cardIndex) {

const iconInput = document.getElementById(`icon-input-${cardIndex}`);

const iconPreview = document.getElementById(`icon-preview-${cardIndex}`);

const iconClass = iconInput.value.trim();

if (iconClass) {

iconPreview.className = `fa ${iconClass}`; // Update preview with entered class

} else {

iconPreview.className = 'fa fa-home'; // Default icon if input is empty

}

}

// Helper function to collect card data

const getCardsData = () => {

const cards = [];

const cardElements = document.querySelectorAll('[name^="cards["]');

cardElements.forEach((input) => {

const cardIndex = input.name.match(/\d+/)[0]; // Extract card index

// Ensure cards[cardIndex] exists

if (!cards[cardIndex]) {

cards[cardIndex] = {}; // Initialize if not present

}

// Assign values to card data

if (input.name.includes('title')) {

cards[cardIndex].title = input.value;

} else if (input.name.includes('content')) {

cards[cardIndex].content = input.value;

} else if (input.name.includes('icon')) {

cards[cardIndex].icon = input.value;

}

});

return cards;

};

// Update card indexes after drag or delete

const updateCardIndexes = () => {

const cardElements = cardsContainer.querySelectorAll('.col');

cardElements.forEach((card, index) => {

card.dataset.index = index; // Update index

const titleInput = card.querySelector('[name\*="[title]"]');

const contentInput = card.querySelector('[name\*="[content]"]');

const iconInput = card.querySelector('[name\*="[icon]"]');

// Update name attributes with new index

titleInput.name = `cards[${index}][title]`;

contentInput.name = `cards[${index}][content]`;

iconInput.name = `cards[${index}][icon]`;

});

};

// Save button click

const savehowourWeb = () => {

const benefitofId = document.getElementById('benefitof\_id').value;

const parent\_id = document.getElementById('parent\_id').value;

const title = document.getElementById('howOurWebsites\_title').value;

const cards = getCardsData(); // Collect cards data

const formData = new FormData();

formData.append('title', title);

formData.append('cards', JSON.stringify(cards)); // Send cards as JSON string

formData.append('howOurWebsites\_id', benefitofId);

formData.append('parent\_id', parent\_id);

const url = `{{ route('how-Our-Websites.update', '') }}/${parent\_id}`;

fetch(url, {

method: 'POST',

headers: {

'X-CSRF-TOKEN': document.querySelector('meta[name="csrf-token"]').getAttribute('content'),

},

body: formData,

})

.then(response => response.json())

.then(data => {

if (data.success) {

Toastify({

text: "Benefit of Section updated successfully!",

backgroundColor: "#4CAF50",

position: "right",

}).showToast();

} else {

Toastify({

text: "An error occurred: " + data.message,

backgroundColor: "#f44336",

position: "right",

}).showToast();

}

})

.catch(error => {

console.error("Error:", error);

Toastify({

text: "An unexpected error occurred.",

backgroundColor: "#f44336",

position: "right",

}).showToast();

});

};

document.getElementById('savehowourWebbutton').addEventListener('click', savehowourWeb);

});

</script>

<script>

document.addEventListener('DOMContentLoaded', function () {

const maxCards = 20;

const cardsContainer = document.getElementById('takeYourBusiness-cards-container');

const addCardButton = document.getElementById('add-takeYourBusiness-card');

// Initialize SortableJS

const sortable = new Sortable(cardsContainer.querySelector('.row'), {

handle: '.card',

animation: 150,

onEnd(evt) {

updateTakeYourBusinessCardIndexes();

},

onStart(evt) {

evt.item.classList.add('dragging');

}

});

// Add new card

addCardButton.addEventListener('click', function () {

if (cardsContainer.querySelectorAll('.col').length >= maxCards) {

Swal.fire({

icon: 'warning',

title: 'Limit Reached',

text: 'You can only add a maximum of 20 cards.',

confirmButtonText: 'OK',

});

return;

}

const newIndex = cardsContainer.querySelectorAll('.col').length;

const cardHtml = `

<div class="col mb-3" data-index="${newIndex}">

<div class="card p-3 border">

<div class="mb-2">

<label>Icon (Enter Font Awesome Class)</label>

<div class="input-group">

<input type="text" name="cards2[${newIndex}][icon]" id="take-icon-input-${newIndex}" class="form-control" placeholder="e.g. fa-home, fa-heart, etc.">

<span class="input-group-text">

<i id="take-icon-preview-${newIndex}" class="fa fa-home"></i>

</span>

</div>

</div>

<div class="mb-2">

<label>Title</label>

<input type="text" name="cards2[${newIndex}][title]" class="form-control">

</div>

<div class="mb-2">

<label>Content</label>

<textarea rows="4" name="cards2[${newIndex}][content]" class="form-control"></textarea>

</div>

<button type="button" class="btn btn-danger btn-sm remove-card">Remove Card</button>

</div>

</div>

`;

cardsContainer.querySelector('.row').insertAdjacentHTML('beforeend', cardHtml);

updateTakeYourBusinessCardIndexes(); // Re-index cards after adding

});

// Remove card

cardsContainer.addEventListener('click', function (e) {

if (e.target.classList.contains('remove-card')) {

const card = e.target.closest('.col');

card.remove();

updateTakeYourBusinessCardIndexes(); // Re-index cards after removal

}

});

// Handle icon input changes

cardsContainer.addEventListener('input', function (e) {

if (e.target.id.startsWith('take-icon-input-')) {

const cardIndex = e.target.id.split('-')[3]; // Assuming `why-icon-input-1` format

updateIconPreview(cardIndex);

}

});

function updateIconPreview(cardIndex) {

const iconInput = document.getElementById(`take-icon-input-${cardIndex}`);

const iconPreview = document.getElementById(`take-icon-preview-${cardIndex}`);

if (!iconInput || !iconPreview) {

console.error(`Elements not found for index: ${cardIndex}`);

return;

}

const iconClass = iconInput.value.trim();

if (iconClass) {

iconPreview.className = `fa ${iconClass}`;

} else {

iconPreview.className = 'fa fa-rocket';

}

}

// Helper function to collect card data

const getTakeYourBusinessCardsData = () => {

const cards = {};

const cardElements = document.querySelectorAll('[name^="cards2["]');

cardElements.forEach((input) => {

const match = input.name.match(/\[(\d+)\]\[(\w+)\]/);

if (match) {

const cardIndex = match[1];

const fieldName = match[2];

if (!cards[cardIndex]) {

cards[cardIndex] = {};

}

cards[cardIndex][fieldName] = input.value;

}

});

const normalizedCards = Object.keys(cards)

.sort((a, b) => a - b)

.map((key) => cards[key]);

console.log("Normalized Cards Data:", normalizedCards);

return normalizedCards;

};

const updateTakeYourBusinessCardIndexes = () => {

const cardElements = cardsContainer.querySelectorAll('.col');

cardElements.forEach((card, index) => {

card.dataset.index = index; // Update index

const titleInput = card.querySelector('[name\*="[title]"]');

const contentInput = card.querySelector('[name\*="[content]"]');

const iconInput = card.querySelector('[name\*="[icon]"]');

// Update name attributes with new index

titleInput.name = `cards2[${index}][title]`;

contentInput.name = `cards2[${index}][content]`;

iconInput.name = `cards2[${index}][icon]`;

});

};

// Save button click

const saveTakeYourBusiness = () => {

const benefitofId = document.getElementById('benefitof\_id').value;

const parent\_id = document.getElementById('parent\_id').value;

const title = document.getElementById('takeYourBusiness\_title').value;

const content = document.getElementById('takeYourBusiness\_content').value;

const cards = getTakeYourBusinessCardsData();

const formData = new FormData();

formData.append('title', title);

formData.append('content', content);

formData.append('cards', JSON.stringify(cards)); // Send cards as JSON string

formData.append('takeYourBusiness\_id', benefitofId);

formData.append('parent\_id', parent\_id);

const url = `{{ route('take-Your-Business.update', '') }}/${parent\_id}`;

fetch(url, {

method: 'POST',

headers: {

'X-CSRF-TOKEN': document.querySelector('meta[name="csrf-token"]').getAttribute('content'),

},

body: formData,

})

.then(response => response.json())

.then(data => {

if (data.success) {

Toastify({

text: "Take Your Business Section updated successfully!",

backgroundColor: "#4CAF50",

position: "right",

}).showToast();

} else {

Toastify({

text: "An error occurred: " + data.message,

backgroundColor: "#f44336",

position: "right",

}).showToast();

}

})

.catch(error => {

console.error("Error:", error);

Toastify({

text: "An unexpected error occurred.",

backgroundColor: "#f44336",

position: "right",

}).showToast();

});

};

document.getElementById('saveTakeYourBusinessButton').addEventListener('click', saveTakeYourBusiness);

});

</script>

<script>

document.addEventListener('DOMContentLoaded', function () {

const maxCards = 20;

const cardsContainer = document.getElementById('whyWebsiteNeed-cards-container');

const addCardButton = document.getElementById('add-whyWebsiteNeed-card');

// Initialize SortableJS

const sortable = new Sortable(cardsContainer.querySelector('.row'), {

handle: '.card',

animation: 150,

onEnd(evt) {

updateWhyWebsiteNeedCardIndexes();

},

onStart(evt) {

evt.item.classList.add('dragging');

}

});

// Add new card

addCardButton.addEventListener('click', function () {

if (cardsContainer.querySelectorAll('.col').length >= maxCards) {

Swal.fire({

icon: 'warning',

title: 'Limit Reached',

text: 'You can only add a maximum of 20 cards.',

confirmButtonText: 'OK',

});

return;

}

const newIndex = cardsContainer.querySelectorAll('.col').length;

const cardHtml = `

<div class="col mb-3" data-index="${newIndex}">

<div class="card p-3 border">

<div class="mb-2">

<label>Icon (Enter Font Awesome Class)</label>

<div class="input-group">

<input type="text" name="cards3[${newIndex}][icon]" id="why-icon-input-${newIndex}" class="form-control" placeholder="e.g. fa-rocket, fa-cogs, etc.">

<span class="input-group-text">

<i id="why-icon-preview-${newIndex}" class="fa fa-rocket"></i>

</span>

</div>

</div>

<div class="mb-2">

<label>Title</label>

<input type="text" name="cards3[${newIndex}][title]" class="form-control" id="why-title-input-${newIndex}">

</div>

<div class="mb-2">

<label>Content</label>

<textarea rows="4" name="cards3[${newIndex}][content]" class="form-control" id="why-content-input-${newIndex}"></textarea>

</div>

<button type="button" class="btn btn-danger btn-sm remove-card">Remove Card</button>

</div>

</div>

`;

cardsContainer.querySelector('.row').insertAdjacentHTML('beforeend', cardHtml);

updateWhyWebsiteNeedCardIndexes(); // Re-index cards after adding

});

cardsContainer.addEventListener('click', function (event) {

if (event.target.classList.contains('remove-card')) {

const card = event.target.closest('.col');

card.remove();

updateWhyWebsiteNeedCardIndexes(); // Re-index cards after removal

}

});

// Event Listener for the input container

cardsContainer.addEventListener('input', function (e) {

if (e.target.id.startsWith('why-icon-input-')) {

const cardIndex = e.target.id.split('-')[3]; // Assuming `why-icon-input-1` format

updateIconPreview(cardIndex);

}

});

// Update the icon preview based on user input

function updateIconPreview(cardIndex) {

const iconInput = document.getElementById(`why-icon-input-${cardIndex}`);

const iconPreview = document.getElementById(`why-icon-preview-${cardIndex}`);

if (!iconInput || !iconPreview) {

console.error(`Elements not found for index: ${cardIndex}`);

return;

}

const iconClass = iconInput.value.trim();

if (iconClass) {

iconPreview.className = `fa ${iconClass}`;

} else {

iconPreview.className = 'fa fa-rocket';

}

}

function updateWhyWebsiteNeedCardIndexes() {

const cardElements = cardsContainer.querySelectorAll('.col');

cardElements.forEach((card, index) => {

card.dataset.index = index; // Update index

const titleInput = card.querySelector('[name\*="[title]"]');

const contentInput = card.querySelector('[name\*="[content]"]');

const iconInput = card.querySelector('[name\*="[icon]"]');

// Update name attributes with new index

titleInput.name = `cards3[${index}][title]`;

contentInput.name = `cards3[${index}][content]`;

iconInput.name = `cards3[${index}][icon]`;

});

}

const getWhyWebsiteNeedCardsData = () => {

const cards = {};

const cardElements = document.querySelectorAll('[name^="cards3["]');

cardElements.forEach((input) => {

const match = input.name.match(/\[(\d+)\]\[(\w+)\]/);

if (match) {

const cardIndex = match[1];

const fieldName = match[2];

if (!cards[cardIndex]) {

cards[cardIndex] = {};

}

cards[cardIndex][fieldName] = input.value;

}

});

const normalizedCards = Object.keys(cards)

.sort((a, b) => a - b)

.map((key) => cards[key]);

console.log("Normalized Cards Data:", normalizedCards);

return normalizedCards;

};

// Update card indexes after drag or delete

document.getElementById('whyWebsiteNeed\_image').addEventListener('change', function(event) {

const file = event.target.files[0];

const reader = new FileReader();

reader.onload = function(e) {

const imgPreview = document.getElementById('whyWebsiteNeed-imagePreview');

imgPreview.src = e.target.result; // Set the source of the image to the file's result

};

if (file) {

reader.readAsDataURL(file); // Read the file as a Data URL

}

});

const saveWhyWebsiteNeed = () => {

const benefitofId = document.getElementById('whyWebsiteNeed\_id').value;

const parent\_id = document.getElementById('parent\_id').value;

const title = document.getElementById('whyWebsiteNeed\_title').value;

const imageFile = document.getElementById('whyWebsiteNeed\_image').files[0];

const cards = getWhyWebsiteNeedCardsData(); // Collect cards data

const formData = new FormData();

formData.append('title', title);

formData.append('cards', JSON.stringify(cards)); // Send cards as JSON string

formData.append('howOurWebsites\_id', benefitofId);

formData.append('parent\_id', parent\_id);

if (imageFile) {

formData.append('image', imageFile); // Append the image file if available

}

const url = `{{ route('why-website-need.update', '') }}/${parent\_id}`;

fetch(url, {

method: 'POST',

headers: {

'X-CSRF-TOKEN': document.querySelector('meta[name="csrf-token"]').getAttribute('content'),

},

body: formData,

})

.then(response => response.json())

.then(data => {

if (data.success) {

Toastify({

text: "Why Website Need Section updated successfully!",

backgroundColor: "#4CAF50",

position: "right",

}).showToast();

} else {

Toastify({

text: "An error occurred: " + (data.message || 'Unknown error'),

backgroundColor: "#f44336",

position: "right",

}).showToast();

}

})

.catch(error => {

console.error("Error:", error);

Toastify({

text: "An unexpected error occurred. Check console logs.",

backgroundColor: "#f44336",

position: "right",

}).showToast();

});

};

document.getElementById('saveWhyWebsiteNeedButton').addEventListener('click', saveWhyWebsiteNeed);

});

</script>

@endsection