Project Structure

```
billing-system/
    .env
    .env.example
    README.md
    app/
        Console/
            Kernel.php
        Exceptions/
            Handler.php
        Http/
            Controllers/
                Auth/
                    AuthenticatedSessionController.php
                    ConfirmablePasswordController.php
                    EmailVerificationNotificationController.php
                    EmailVerificationPromptController.php
                    NewPasswordController.php
                    PasswordController.php
                    PasswordResetLinkController.php
                    RegisteredUserController.php
                    VerifyEmailController.php
                ClientController.php
                Controller.php
                DashboardController.php
                InvoiceController.php
                ProductController.php
                ProfileController.php
                SettingsController.php
            Kernel.php
            Middleware/
                Authenticate.php
                EncryptCookies.php
                PreventRequestsDuringMaintenance.php
                RedirectIfAuthenticated.php
                TrimStrings.php
                TrustHosts.php
                TrustProxies.php
                ValidateSignature.php
                VerifyCsrfToken.php
            Requests/
                Auth/
                    LoginRequest.php
                ProfileUpdateRequest.php
                StoreClientRequest.php
                StoreInvoiceRequest.php
                StoreProductRequest.php
                UpdateClientRequest.php
                UpdateInvoiceRequest.php
        Models/
            Client.php
```

```
        Invoice.php
        InvoiceItem.php
        Product.php
        SerialNumber.php
        Settings.php
        User.php
    Providers/
        AppServiceProvider.php
        AuthServiceProvider.php
        BroadcastServiceProvider.php
        EventServiceProvider.php
        RouteServiceProvider.php
    View/
        Components/
            AppLayout.php
            GuestLayout.php
bootstrap/
    app.php
    cache/
        packages.php
        services.php
composer.json
config/
    app.php
    auth.php
    cache.php
    database.php
    filesystems.php
    hashing.php
    logging.php
    mail.php
    queue.php
    services.php
    session.php
    settings.php
    view.php
database/
    factories/
        UserFactory.php
    migrations/
        2023_01_01_000000_create_clients_table.php
        2023_01_01_000001_create_products_table.php
        2023_01_01_000002_create_invoices_table.php
        2023_01_01_000003_create_invoice_items_table.php
        2023_01_01_000005_create_settings_table.php
        2025_05_03_005940_create_users_table.php
        create_serial_numbers_table.php
    seeders/
        DatabaseSeeder.php
        SettingsSeeder.php
package.json
pdd1.py
public/
    index.php
```

```
resources/
    css/
        app.css
    js/
        app.js
        bootstrap.js
    views/
        Client
        auth/
            confirm-password.blade.php
            forgot-password.blade.php
            login.blade.php
            register.blade.php
            reset-password.blade.php
            verify-email.blade.php
        clients/
            create.blade.php
            index.blade.php
        components/
            application-logo.blade.php
            auth-session-status.blade.php
            danger-button.blade.php
            dropdown-link.blade.php
            dropdown.blade.php
            flash-message.blade.php
            input-error.blade.php
            input-label.blade.php
            modal.blade.php
            nav-link.blade.php
            primary-button.blade.php
            responsive-nav-link.blade.php
            secondary-button.blade.php
            text-input.blade.php
        dashboard.blade.php
        invoices/
            create.blade.php
            edit.blade.php
            index.blade.php
            show.blade.php
            template/
                custom.blade.php
            templates/
                custom.blade.php
        layouts/
            app.blade.php
            guest.blade.php
            navigation.blade.php
        products/
            index.blade.php
        profile/
            edit.blade.php
            partials/
                delete-user-form.blade.php
                update-password-form.blade.php
```

```
                update-profile-information-form.blade.php
        settings/
            index.blade.php
        welcome.blade.php
routes/
    api.php
    auth.php
    channels.php
    console.php
    web.php
```

app/Console/Kernel.php

```
<?php

namespace App\Console;

use Illuminate\Console\Scheduling\Schedule;
use Illuminate\Foundation\Console\Kernel as ConsoleKernel;

class Kernel extends ConsoleKernel
{
    /**
     * Define the application's command schedule.
     */
    protected function schedule(Schedule $schedule): void
    {
        // $schedule->command('inspire')->hourly();
    }

    /**
     * Register the commands for the application.
     */
    protected function commands(): void
    {
        $this->load(__DIR__.'/Commands');

        require base_path('routes/console.php');
    }
}
```

app/Exceptions/Handler.php

```
<?php

namespace App\Exceptions;

use Illuminate\Foundation\Exceptions\Handler as ExceptionHandler;
use Throwable;

class Handler extends ExceptionHandler
{
    /**
     * The list of the inputs that are never flashed to the session on validation exceptions.
     *
     * @var array<int, string>
     */
    protected $dontFlash = [
        'current_password',
        'password',
        'password_confirmation',
    ];

    /**
     * Register the exception handling callbacks for the application.
     */
    public function register(): void
    {
        $this->reportable(function (Throwable $e) {
            //
        });
    }
}
```

app/Http/Kernel.php

```
<?php

namespace App\Http;

use Illuminate\Foundation\Http\Kernel as HttpKernel;

class Kernel extends HttpKernel
{
    /**
     * The application's global HTTP middleware stack.
     *
     * These middleware are run during every request to your application.
     *
     * @var array<int, class-string|string>
     */
    protected $middleware = [
        // \App\Http\Middleware\TrustHosts::class,
        \App\Http\Middleware\TrustProxies::class,
        \Illuminate\Http\Middleware\HandleCors::class,
        \App\Http\Middleware\PreventRequestsDuringMaintenance::class,
        \Illuminate\Foundation\Http\Middleware\ValidatePostSize::class,
        \App\Http\Middleware\TrimStrings::class,
        \Illuminate\Foundation\Http\Middleware\ConvertEmptyStringsToNull::class,
    ];

    /**
     * The application's route middleware groups.
     *
     * @var array<string, array<int, class-string|string>>
     */
    protected $middlewareGroups = [
        'web' => [
            \App\Http\Middleware\EncryptCookies::class,
            \Illuminate\Cookie\Middleware\AddQueuedCookiesToResponse::class,
            \Illuminate\Session\Middleware\StartSession::class,
            \Illuminate\View\Middleware\ShareErrorsFromSession::class,
            \App\Http\Middleware\VerifyCsrfToken::class,
            \Illuminate\Routing\Middleware\SubstituteBindings::class,
        ],

        'api' => [
            // \Laravel\Sanctum\Http\Middleware\EnsureFrontendRequestsAreStateful::class,
            \Illuminate\Routing\Middleware\ThrottleRequests::class.':api',
            \Illuminate\Routing\Middleware\SubstituteBindings::class,
        ],
    ];

    /**
     * The application's middleware aliases.
     *
      * Aliases may be used instead of class names to conveniently assign middleware to routes and
```

```
groups.
     *
     * @var array<string, class-string|string>
     */
    protected $middlewareAliases = [
        'auth' => \App\Http\Middleware\Authenticate::class,
        'auth.basic' => \Illuminate\Auth\Middleware\AuthenticateWithBasicAuth::class,
        'auth.session' => \Illuminate\Session\Middleware\AuthenticateSession::class,
        'cache.headers' => \Illuminate\Http\Middleware\SetCacheHeaders::class,
        'can' => \Illuminate\Auth\Middleware\Authorize::class,
        'guest' => \App\Http\Middleware\RedirectIfAuthenticated::class,
        'password.confirm' => \Illuminate\Auth\Middleware\RequirePassword::class,
                                                                    'precognitive'          =>
\Illuminate\Foundation\Http\Middleware\HandlePrecognitiveRequests::class,
        'signed' => \App\Http\Middleware\ValidateSignature::class,
        'throttle' => \Illuminate\Routing\Middleware\ThrottleRequests::class,
        'verified' => \Illuminate\Auth\Middleware\EnsureEmailIsVerified::class,
    ];
}
```

app/Http/Controllers/ClientController.php

```
<?php

namespace App\Http\Controllers;

use App\Models\Client;
use App\Http\Requests\StoreClientRequest;
use App\Http\Requests\UpdateClientRequest;
use Illuminate\Http\Request;

class ClientController extends Controller
{
    /**
     * Display a listing of the clients.
     */
    public function index(Request $request)
    {
        $query = Client::query();

        // Search functionality
        if ($request->has('search')) {
            $search = $request->get('search');
            $query->where(function($q) use ($search) {
                $q->where('name', 'like', "%{$search}%")
                  ->orWhere('email', 'like', "%{$search}%")
                  ->orWhere('phone', 'like', "%{$search}%");
            });
        }

        $clients = $query->latest()->paginate(10);

        return view('clients.index', compact('clients'));
    }

    /**
     * Show the form for creating a new client.
     */
    public function create()
    {
        return view('clients.create');
    }

    /**
     * Store a newly created client in storage.
     */
    public function store(StoreClientRequest $request)
    {
        $client = Client::create($request->validated());

        return redirect()->route('clients.show', $client)
            ->with('success', 'Client created successfully.');
    }
```

```
    /**
     * Display the specified client.
     */
    public function show(Client $client)
    {
        $invoices = $client->invoices()->latest()->paginate(5);
        return view('clients.show', compact('client', 'invoices'));
    }

    /**
     * Show the form for editing the specified client.
     */
    public function edit(Client $client)
    {
        return view('clients.edit', compact('client'));
    }

    /**
     * Update the specified client in storage.
     */
    public function update(UpdateClientRequest $request, Client $client)
    {
        $client->update($request->validated());

        return redirect()->route('clients.show', $client)
            ->with('success', 'Client updated successfully.');
    }

    /**
     * Remove the specified client from storage.
     */
    public function destroy(Client $client)
    {
        // Check if client has invoices
        if ($client->invoices()->count() > 0) {
            return back()->with('error', 'Cannot delete client with existing invoices.');
        }

        $client->delete();

        return redirect()->route('clients.index')
            ->with('success', 'Client deleted successfully.');
    }
}
```

app/Http/Controllers/Controller.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Foundation\Auth\Access\AuthorizesRequests;
use Illuminate\Foundation\Bus\DispatchesJobs;
use Illuminate\Foundation\Validation\ValidatesRequests;
use Illuminate\Routing\Controller as BaseController;

class Controller extends BaseController
{
    use AuthorizesRequests, DispatchesJobs, ValidatesRequests;
}
```

app/Http/Controllers/DashboardController.php

```
<?php

namespace App\Http\Controllers;

use App\Models\Invoice;
use App\Models\Client;
use App\Models\Product;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use Carbon\Carbon;

class DashboardController extends Controller
{
    /**
     * Display the dashboard with statistics.
     */
    public function index()
    {
        // Get counts
        $totalClients = Client::count();
        $totalProducts = Product::count();
        $totalInvoices = Invoice::count();

        // Get total revenue
        $totalRevenue = Invoice::sum('total');

        // Get recent invoices
        $recentInvoices = Invoice::with('client')
            ->latest()
            ->take(5)
            ->get();

        // Get monthly revenue for the current year
        $monthlyRevenue = Invoice::select(
                DB::raw('MONTH(invoice_date) as month'),
                DB::raw('SUM(total) as revenue')
            )
            ->whereYear('invoice_date', Carbon::now()->year)
            ->groupBy('month')
            ->orderBy('month')
            ->get()
            ->keyBy('month')
            ->map(function ($item) {
                return round($item->revenue, 2);
            });

        // Fill in missing months with zero
        for ($i = 1; $i <= 12; $i++) {
            if (!isset($monthlyRevenue[$i])) {
                $monthlyRevenue[$i] = 0;
            }
```

```
        }
        $monthlyRevenue = $monthlyRevenue->sortKeys();

        // Get top clients
         $topClients = Client::select('clients.id', 'clients.name', DB::raw('SUM(invoices.total) as
total_spent'))
            ->join('invoices', 'clients.id', '=', 'invoices.client_id')
            ->groupBy('clients.id', 'clients.name')
            ->orderByDesc('total_spent')
            ->take(5)
            ->get();

        // Get top products
                            $topProducts    =    Product::select('products.id',    'products.name',
DB::raw('SUM(invoice_items.quantity) as total_sold'))
            ->join('invoice_items', 'products.id', '=', 'invoice_items.product_id')
            ->groupBy('products.id', 'products.name')
            ->orderByDesc('total_sold')
            ->take(5)
            ->get();

        return view('dashboard', compact(
            'totalClients',
            'totalProducts',
            'totalInvoices',
            'totalRevenue',
            'recentInvoices',
            'monthlyRevenue',
            'topClients',
            'topProducts'
        ));
    }
}
```

app/Http/Controllers/InvoiceController.php

```
<?php

namespace App\Http\Controllers;

use App\Models\Invoice;
use App\Models\Client;
use App\Models\Product;
use App\Http\Requests\StoreInvoiceRequest;
use App\Http\Requests\UpdateInvoiceRequest;
use Barryvdh\DomPDF\Facade\Pdf;

class InvoiceController extends Controller
{
    public function index()
    {
        $invoices = Invoice::with('client')->latest()->paginate(10);
        return view('invoices.index', compact('invoices'));
    }

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

    public function store(StoreInvoiceRequest $request)
    {
        $invoice = Invoice::create($request->validated());

        foreach ($request->items as $item) {
            $invoiceItem = $invoice->items()->create([
                'product_id' => $item['product_id'],
                'quantity' => $item['quantity'],
                'price' => $item['price']
            ]);

            if (isset($item['serial_numbers'])) {
                $serials = explode(',', $item['serial_numbers']);
                foreach ($serials as $serial) {
                    $invoiceItem->serialNumbers()->create([
                        'product_id' => $item['product_id'],
                        'serial_number' => trim($serial)
                    ]);
                }
            }
        }

        $invoice->calculateTotals();

        return redirect()->route('invoices.show', $invoice)
```

```
        ->with('success', 'Invoice created successfully.');
}

public function show(Invoice $invoice)
{
    $invoice->load('client', 'items.product', 'items.serialNumbers');
    return view('invoices.show', compact('invoice'));
}

public function edit(Invoice $invoice)
{
    $clients = Client::all();
    $products = Product::all();
    $invoice->load('items.product', 'items.serialNumbers');
    return view('invoices.edit', compact('invoice', 'clients', 'products'));
}

public function update(UpdateInvoiceRequest $request, Invoice $invoice)
{
    $invoice->update($request->validated());

    // Delete existing items and serial numbers
    $invoice->items()->delete();

    foreach ($request->items as $item) {
        $invoiceItem = $invoice->items()->create([
            'product_id' => $item['product_id'],
            'quantity' => $item['quantity'],
            'price' => $item['price']
        ]);

        if (isset($item['serial_numbers'])) {
            $serials = explode(',', $item['serial_numbers']);
            foreach ($serials as $serial) {
                $invoiceItem->serialNumbers()->create([
                    'product_id' => $item['product_id'],
                    'serial_number' => trim($serial)
                ]);
            }
        }
    }

    $invoice->calculateTotals();

    return redirect()->route('invoices.show', $invoice)
        ->with('success', 'Invoice updated successfully.');
}

public function destroy(Invoice $invoice)
{
    $invoice->delete();
    return redirect()->route('invoices.index')
        ->with('success', 'Invoice deleted successfully.');
}
```

```
    public function pdf(Invoice $invoice)
    {
        $invoice->load('client', 'items.product', 'items.serialNumbers');
        $template = Settings::get('invoice.default_template', 'custom.blade.php');

        $pdf = PDF::loadView("invoices.templates.$template", compact('invoice'));
        return $pdf->download("invoice_{$invoice->invoice_number}.pdf");
    }
}
```

app/Http/Controllers/ProductController.php

```
<?php

namespace App\Http\Controllers;

use App\Models\Product;
use App\Http\Requests\StoreProductRequest;
use App\Http\Requests\UpdateProductRequest;
use Illuminate\Http\Request;

class ProductController extends Controller
{
    /**
     * Display a listing of the products.
     */
    public function index(Request $request)
    {
        $query = Product::query();

        // Search functionality
        if ($request->has('search')) {
            $search = $request->get('search');
            $query->where(function($q) use ($search) {
                $q->where('name', 'like', "%{$search}%")
                  ->orWhere('description', 'like', "%{$search}%");
            });
        }

        $products = $query->latest()->paginate(10);

        return view('products.index', compact('products'));
    }

    /**
     * Show the form for creating a new product.
     */
    public function create()
    {
        return view('products.create');
    }

    /**
     * Store a newly created product in storage.
     */
    public function store(StoreProductRequest $request)
    {
        $product = Product::create($request->validated());

        return redirect()->route('products.show', $product)
            ->with('success', 'Product created successfully.');
    }
```

```
    /**
     * Display the specified product.
     */
    public function show(Product $product)
    {
        return view('products.show', compact('product'));
    }

    /**
     * Show the form for editing the specified product.
     */
    public function edit(Product $product)
    {
        return view('products.edit', compact('product'));
    }

    /**
     * Update the specified product in storage.
     */
    public function update(UpdateProductRequest $request, Product $product)
    {
        $product->update($request->validated());

        return redirect()->route('products.show', $product)
            ->with('success', 'Product updated successfully.');
    }

    /**
     * Remove the specified product from storage.
     */
    public function destroy(Product $product)
    {
        // Check if product is used in any invoice
        $invoiceItemCount = \App\Models\InvoiceItem::where('product_id', $product->id)->count();

        if ($invoiceItemCount > 0) {
            return back()->with('error', 'Cannot delete product that is used in invoices.');
        }

        $product->delete();

        return redirect()->route('products.index')
            ->with('success', 'Product deleted successfully.');
    }
}
```

app/Http/Controllers/ProfileController.php

```
<?php

namespace App\Http\Controllers;

use App\Http\Requests\ProfileUpdateRequest;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Redirect;
use Illuminate\View\View;

class ProfileController extends Controller
{
    /**
     * Display the user's profile form.
     */
    public function edit(Request $request): View
    {
        return view('profile.edit', [
            'user' => $request->user(),
        ]);
    }

    /**
     * Update the user's profile information.
     */
    public function update(ProfileUpdateRequest $request): RedirectResponse
    {
        $request->user()->fill($request->validated());

        if ($request->user()->isDirty('email')) {
            $request->user()->email_verified_at = null;
        }

        $request->user()->save();

        return Redirect::route('profile.edit')->with('status', 'profile-updated');
    }

    /**
     * Delete the user's account.
     */
    public function destroy(Request $request): RedirectResponse
    {
        $request->validateWithBag('userDeletion', [
            'password' => ['required', 'current_password'],
        ]);

        $user = $request->user();

        Auth::logout();
```

```
        $user->delete();

        $request->session()->invalidate();
        $request->session()->regenerateToken();

        return Redirect::to('/');
    }
}
```

app/Http/Controllers/SettingsController.php

```
<?php

namespace App\Http\Controllers;

use App\Models\Settings;
use Illuminate\Http\Request;

class SettingsController extends Controller
{
    public function index()
    {
        $settings = Settings::all()->pluck('setting_value', 'setting_key');
        return view('settings.index', compact('settings'));
    }

    public function update(Request $request)
    {
        foreach ($request->except('_token') as $key => $value) {
            Settings::set($key, $value);
        }

        return redirect()->route('settings.index')
            ->with('success', 'Settings updated successfully.');
    }
}
```

app/Http/Controllers/Auth/AuthenticatedSessionController.php

```
<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use App\Http\Requests\Auth\LoginRequest;
use App\Providers\RouteServiceProvider;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\View\View;

class AuthenticatedSessionController extends Controller
{
    /**
     * Display the login view.
     */
    public function create(): View
    {
        return view('auth.login');
    }

    /**
     * Handle an incoming authentication request.
     */
    public function store(LoginRequest $request): RedirectResponse
    {
        $request->authenticate();

        $request->session()->regenerate();

        return redirect()->intended(RouteServiceProvider::HOME);
    }

    /**
     * Destroy an authenticated session.
     */
    public function destroy(Request $request): RedirectResponse
    {
        Auth::guard('web')->logout();

        $request->session()->invalidate();

        $request->session()->regenerateToken();

        return redirect('/');
    }
}
```

app/Http/Controllers/Auth/ConfirmablePasswordController.php

```php
<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use App\Providers\RouteServiceProvider;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Validation\ValidationException;
use Illuminate\View\View;

class ConfirmablePasswordController extends Controller
{
    /**
     * Show the confirm password view.
     */
    public function show(): View
    {
        return view('auth.confirm-password');
    }

    /**
     * Confirm the user's password.
     */
    public function store(Request $request): RedirectResponse
    {
        if (! Auth::guard('web')->validate([
            'email' => $request->user()->email,
            'password' => $request->password,
        ])) {
            throw ValidationException::withMessages([
                'password' => __('auth.password'),
            ]);
        }

        $request->session()->put('auth.password_confirmed_at', time());

        return redirect()->intended(RouteServiceProvider::HOME);
    }
}
```

app/Http/Controllers/Auth/EmailVerificationNotificationController.php

```php
<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use App\Providers\RouteServiceProvider;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;

class EmailVerificationNotificationController extends Controller
{
    /**
     * Send a new email verification notification.
     */
    public function store(Request $request): RedirectResponse
    {
        if ($request->user()->hasVerifiedEmail()) {
            return redirect()->intended(RouteServiceProvider::HOME);
        }

        $request->user()->sendEmailVerificationNotification();

        return back()->with('status', 'verification-link-sent');
    }
}
```

app/Http/Controllers/Auth/EmailVerificationPromptController.php

```
<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use App\Providers\RouteServiceProvider;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\View\View;

class EmailVerificationPromptController extends Controller
{
    /**
     * Display the email verification prompt.
     */
    public function __invoke(Request $request): RedirectResponse|View
    {
        return $request->user()->hasVerifiedEmail()
                    ? redirect()->intended(RouteServiceProvider::HOME)
                    : view('auth.verify-email');
    }
}
```

app/Http/Controllers/Auth/NewPasswordController.php

```
<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use Illuminate\Auth\Events\PasswordReset;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Password;
use Illuminate\Support\Str;
use Illuminate\Validation\Rules;
use Illuminate\View\View;

class NewPasswordController extends Controller
{
    /**
     * Display the password reset view.
     */
    public function create(Request $request): View
    {
        return view('auth.reset-password', ['request' => $request]);
    }

    /**
     * Handle an incoming new password request.
     *
     * @throws \Illuminate\Validation\ValidationException
     */
    public function store(Request $request): RedirectResponse
    {
        $request->validate([
            'token' => ['required'],
            'email' => ['required', 'email'],
            'password' => ['required', 'confirmed', Rules\Password::defaults()],
        ]);

        // Here we will attempt to reset the user's password. If it is successful we
        // will update the password on an actual user model and persist it to the
        // database. Otherwise we will parse the error and return the response.
        $status = Password::reset(
            $request->only('email', 'password', 'password_confirmation', 'token'),
            function ($user) use ($request) {
                $user->forceFill([
                    'password' => Hash::make($request->password),
                    'remember_token' => Str::random(60),
                ])->save();

                event(new PasswordReset($user));
            }
        );
```

```
        // If the password was successfully reset, we will redirect the user back to
        // the application's home authenticated view. If there is an error we can
        // redirect them back to where they came from with their error message.
        return $status == Password::PASSWORD_RESET
                    ? redirect()->route('login')->with('status', __($status))
                    : back()->withInput($request->only('email'))
                            ->withErrors(['email' => __($status)]);
    }
}
```

app/Http/Controllers/Auth/PasswordController.php

```
<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;
use Illuminate\Validation\Rules\Password;

class PasswordController extends Controller
{
    /**
     * Update the user's password.
     */
    public function update(Request $request): RedirectResponse
    {
        $validated = $request->validateWithBag('updatePassword', [
            'current_password' => ['required', 'current_password'],
            'password' => ['required', Password::defaults(), 'confirmed'],
        ]);

        $request->user()->update([
            'password' => Hash::make($validated['password']),
        ]);

        return back()->with('status', 'password-updated');
    }
}
```

app/Http/Controllers/Auth/PasswordResetLinkController.php

```
<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Password;
use Illuminate\View\View;

class PasswordResetLinkController extends Controller
{
    /**
     * Display the password reset link request view.
     */
    public function create(): View
    {
        return view('auth.forgot-password');
    }

    /**
     * Handle an incoming password reset link request.
     *
     * @throws \Illuminate\Validation\ValidationException
     */
    public function store(Request $request): RedirectResponse
    {
        $request->validate([
            'email' => ['required', 'email'],
        ]);

        // We will send the password reset link to this user. Once we have attempted
        // to send the link, we will examine the response then see the message we
        // need to show to the user. Finally, we'll send out a proper response.
        $status = Password::sendResetLink(
            $request->only('email')
        );

        return $status == Password::RESET_LINK_SENT
                    ? back()->with('status', __($status))
                    : back()->withInput($request->only('email'))
                            ->withErrors(['email' => __($status)]);
    }
}
```

app/Http/Controllers/Auth/RegisteredUserController.php

```
<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use App\Models\User;
use App\Providers\RouteServiceProvider;
use Illuminate\Auth\Events\Registered;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Hash;
use Illuminate\Validation\Rules;
use Illuminate\View\View;

class RegisteredUserController extends Controller
{
    /**
     * Display the registration view.
     */
    public function create(): View
    {
        return view('auth.register');
    }

    /**
     * Handle an incoming registration request.
     *
     * @throws \Illuminate\Validation\ValidationException
     */
    public function store(Request $request): RedirectResponse
    {
        $request->validate([
            'name' => ['required', 'string', 'max:255'],
                        'email'  =>  ['required',  'string',  'lowercase',  'email',  'max:255',
'unique:'.User::class],
            'password' => ['required', 'confirmed', Rules\Password::defaults()],
        ]);

        $user = User::create([
            'name' => $request->name,
            'email' => $request->email,
            'password' => Hash::make($request->password),
        ]);

        event(new Registered($user));

        Auth::login($user);

        return redirect(RouteServiceProvider::HOME);
    }
```

```
}
```

app/Http/Controllers/Auth/VerifyEmailController.php

```
<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use App\Providers\RouteServiceProvider;
use Illuminate\Auth\Events\Verified;
use Illuminate\Foundation\Auth\EmailVerificationRequest;
use Illuminate\Http\RedirectResponse;

class VerifyEmailController extends Controller
{
    /**
     * Mark the authenticated user's email address as verified.
     */
    public function __invoke(EmailVerificationRequest $request): RedirectResponse
    {
        if ($request->user()->hasVerifiedEmail()) {
            return redirect()->intended(RouteServiceProvider::HOME.'?verified=1');
        }

        if ($request->user()->markEmailAsVerified()) {
            event(new Verified($request->user()));
        }

        return redirect()->intended(RouteServiceProvider::HOME.'?verified=1');
    }
}
```

app/Http/Middleware/Authenticate.php

```
<?php

namespace App\Http\Middleware;

use Illuminate\Auth\Middleware\Authenticate as Middleware;
use Illuminate\Http\Request;

class Authenticate extends Middleware
{
    /**
     * Get the path the user should be redirected to when they are not authenticated.
     */
    protected function redirectTo(Request $request): ?string
    {
        // Temporary bypass for testing
        if (app()->environment('local')) {
            return $request->expectsJson() ? null : route('dashboard');
        }

        return $request->expectsJson() ? null : route('login');
    }
}
```

app/Http/Middleware/EncryptCookies.php

```
<?php

namespace App\Http\Middleware;

use Illuminate\Cookie\Middleware\EncryptCookies as Middleware;

class EncryptCookies extends Middleware
{
    /**
     * The names of the cookies that should not be encrypted.
     *
     * @var array<int, string>
     */
    protected $except = [
        //
    ];
}
```

app/Http/Middleware/PreventRequestsDuringMaintenance.php

```
<?php

namespace App\Http\Middleware;

use Illuminate\Foundation\Http\Middleware\PreventRequestsDuringMaintenance as Middleware;

class PreventRequestsDuringMaintenance extends Middleware
{
    /**
     * The URIs that should be reachable while maintenance mode is enabled.
     *
     * @var array<int, string>
     */
    protected $except = [
        //
    ];
}
```

app/Http/Middleware/RedirectIfAuthenticated.php

```
<?php

namespace App\Http\Middleware;

use App\Providers\RouteServiceProvider;
use Closure;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Symfony\Component\HttpFoundation\Response;

class RedirectIfAuthenticated
{
    /**
     * Handle an incoming request.
     *
      * @param  \Closure(\Illuminate\Http\Request): (\Symfony\Component\HttpFoundation\Response)
$next
     */
    public function handle(Request $request, Closure $next, string ...$guards): Response
    {
        $guards = empty($guards) ? [null] : $guards;

        foreach ($guards as $guard) {
            if (Auth::guard($guard)->check()) {
                return redirect(RouteServiceProvider::HOME);
            }
        }

        return $next($request);
    }
}
```

app/Http/Middleware/TrimStrings.php

```
<?php

namespace App\Http\Middleware;

use Illuminate\Foundation\Http\Middleware\TrimStrings as Middleware;

class TrimStrings extends Middleware
{
    /**
     * The names of the attributes that should not be trimmed.
     *
     * @var array<int, string>
     */
    protected $except = [
        'current_password',
        'password',
        'password_confirmation',
    ];
}
```

app/Http/Middleware/TrustHosts.php

```
<?php

namespace App\Http\Middleware;

use Illuminate\Http\Middleware\TrustHosts as Middleware;

class TrustHosts extends Middleware
{
    /**
     * Get the host patterns that should be trusted.
     *
     * @return array<int, string|null>
     */
    public function hosts(): array
    {
        return [
            $this->allSubdomainsOfApplicationUrl(),
        ];
    }
}
```

app/Http/Middleware/TrustProxies.php

```
<?php

namespace App\Http\Middleware;

use Illuminate\Http\Middleware\TrustProxies as Middleware;
use Illuminate\Http\Request;

class TrustProxies extends Middleware
{
    /**
     * The trusted proxies for this application.
     *
     * @var array<int, string>|string|null
     */
    protected $proxies;

    /**
     * The headers that should be used to detect proxies.
     *
     * @var int
     */
    protected $headers =
        Request::HEADER_X_FORWARDED_FOR |
        Request::HEADER_X_FORWARDED_HOST |
        Request::HEADER_X_FORWARDED_PORT |
        Request::HEADER_X_FORWARDED_PROTO |
        Request::HEADER_X_FORWARDED_AWS_ELB;
}
```

app/Http/Middleware/ValidateSignature.php

```
<?php

namespace App\Http\Middleware;

use Illuminate\Routing\Middleware\ValidateSignature as Middleware;

class ValidateSignature extends Middleware
{
    /**
     * The names of the query string parameters that should be ignored.
     *
     * @var array<int, string>
     */
    protected $except = [
        // 'fbclid',
        // 'utm_campaign',
        // 'utm_content',
        // 'utm_medium',
        // 'utm_source',
        // 'utm_term',
    ];
}
```

app/Http/Middleware/VerifyCsrfToken.php

```
<?php

namespace App\Http\Middleware;

use Illuminate\Foundation\Http\Middleware\VerifyCsrfToken as Middleware;

class VerifyCsrfToken extends Middleware
{
    /**
     * The URIs that should be excluded from CSRF verification.
     *
     * @var array<int, string>
     */
    protected $except = [
        //
    ];
}
```

app/Http/Requests/ProfileUpdateRequest.php

```
<?php

namespace App\Http\Requests;

use App\Models\User;
use Illuminate\Foundation\Http\FormRequest;
use Illuminate\Validation\Rule;

class ProfileUpdateRequest extends FormRequest
{
    /**
     * Get the validation rules that apply to the request.
     *
     * @return array<string, \Illuminate\Contracts\Validation\Rule|array|string>
     */
    public function rules(): array
    {
        return [
            'name' => ['required', 'string', 'max:255'],
                        'email'  =>  ['required',  'string',  'lowercase',  'email',  'max:255',
Rule::unique(User::class)->ignore($this->user()->id)],
        ];
    }
}
```

app/Http/Requests/StoreClientRequest.php

```
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class StoreClientRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     */
    public function authorize(): bool
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array<string, \Illuminate\Contracts\Validation\Rule|array|string>
     */
    public function rules(): array
    {
        return [
            'name' => ['required', 'string', 'max:255'],
            'email' => ['required', 'string', 'email', 'max:255', 'unique:clients'],
            'phone' => ['nullable', 'string', 'max:20'],
            'address' => ['nullable', 'string', 'max:500'],
        ];
    }
}
```

app/Http/Requests/StoreInvoiceRequest.php

```
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class StoreInvoiceRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     */
    public function authorize(): bool
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array<string, \Illuminate\Contracts\Validation\Rule|array|string>
     */
    public function rules(): array
    {
        return [
            'client_id' => ['required', 'exists:clients,id'],
            'invoice_number' => ['required', 'string', 'max:255', 'unique:invoices'],
            'invoice_date' => ['required', 'date'],
            'due_date' => ['required', 'date', 'after_or_equal:invoice_date'],
            'notes' => ['nullable', 'string', 'max:1000'],
            'items' => ['required', 'array', 'min:1'],
            'items.*.product_id' => ['required', 'exists:products,id'],
            'items.*.quantity' => ['required', 'integer', 'min:1'],
            'items.*.price' => ['required', 'numeric', 'min:0'],
            'items.*.serial_numbers' => ['nullable', 'string'],
        ];
    }
}
```

app/Http/Requests/StoreProductRequest.php

```php
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;

class StoreProductRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     */
    public function authorize(): bool
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array<string, \Illuminate\Contracts\Validation\Rule|array|string>
     */
    public function rules(): array
    {
        return [
            'name' => ['required', 'string', 'max:255'],
            'description' => ['nullable', 'string', 'max:1000'],
            'price' => ['required', 'numeric', 'min:0'],
            'has_serial' => ['boolean'],
        ];
    }
}
```

app/Http/Requests/UpdateClientRequest.php

```php
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;
use Illuminate\Validation\Rule;

class UpdateClientRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     */
    public function authorize(): bool
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array<string, \Illuminate\Contracts\Validation\Rule|array|string>
     */
    public function rules(): array
    {
        return [
            'name' => ['required', 'string', 'max:255'],
            'email' => [
                'required',
                'string',
                'email',
                'max:255',
                Rule::unique('clients')->ignore($this->client)
            ],
            'phone' => ['nullable', 'string', 'max:20'],
            'address' => ['nullable', 'string', 'max:500'],
        ];
    }
}
```

app/Http/Requests/UpdateInvoiceRequest.php

```
<?php

namespace App\Http\Requests;

use Illuminate\Foundation\Http\FormRequest;
use Illuminate\Validation\Rule;

class UpdateInvoiceRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     */
    public function authorize(): bool
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array<string, \Illuminate\Contracts\Validation\Rule|array|string>
     */
    public function rules(): array
    {
        return [
            'client_id' => ['required', 'exists:clients,id'],
            'invoice_number' => [
                'required',
                'string',
                'max:255',
                Rule::unique('invoices')->ignore($this->invoice)
            ],
            'invoice_date' => ['required', 'date'],
            'due_date' => ['required', 'date', 'after_or_equal:invoice_date'],
            'notes' => ['nullable', 'string', 'max:1000'],
            'items' => ['required', 'array', 'min:1'],
            'items.*.product_id' => ['required', 'exists:products,id'],
            'items.*.quantity' => ['required', 'integer', 'min:1'],
            'items.*.price' => ['required', 'numeric', 'min:0'],
            'items.*.serial_numbers' => ['nullable', 'string'],
        ];
    }
}
```

app/Http/Requests/Auth/LoginRequest.php

```
<?php

namespace App\Http\Requests\Auth;

use Illuminate\Auth\Events\Lockout;
use Illuminate\Foundation\Http\FormRequest;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\RateLimiter;
use Illuminate\Support\Str;
use Illuminate\Validation\ValidationException;

class LoginRequest extends FormRequest
{
    /**
     * Determine if the user is authorized to make this request.
     */
    public function authorize(): bool
    {
        return true;
    }

    /**
     * Get the validation rules that apply to the request.
     *
     * @return array<string, \Illuminate\Contracts\Validation\Rule|array|string>
     */
    public function rules(): array
    {
        return [
            'email' => ['required', 'string', 'email'],
            'password' => ['required', 'string'],
        ];
    }

    /**
     * Attempt to authenticate the request's credentials.
     *
     * @throws \Illuminate\Validation\ValidationException
     */
    public function authenticate(): void
    {
        $this->ensureIsNotRateLimited();

        if (! Auth::attempt($this->only('email', 'password'), $this->boolean('remember'))) {
            RateLimiter::hit($this->throttleKey());

            throw ValidationException::withMessages([
                'email' => trans('auth.failed'),
            ]);
        }
```

```
        RateLimiter::clear($this->throttleKey());
    }

    /**
     * Ensure the login request is not rate limited.
     *
     * @throws \Illuminate\Validation\ValidationException
     */
    public function ensureIsNotRateLimited(): void
    {
        if (! RateLimiter::tooManyAttempts($this->throttleKey(), 5)) {
            return;
        }

        event(new Lockout($this));

        $seconds = RateLimiter::availableIn($this->throttleKey());

        throw ValidationException::withMessages([
            'email' => trans('auth.throttle', [
                'seconds' => $seconds,
                'minutes' => ceil($seconds / 60),
            ]),
        ]);
    }

    /**
     * Get the rate limiting throttle key for the request.
     */
    public function throttleKey(): string
    {
        return Str::transliterate(Str::lower($this->string('email')).'|'.$this->ip());
    }
}
```

app/Models/Client.php

```
<?php

namespace App\Models;

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

class Client extends Model
{
    use HasFactory;

    protected $fillable = ['name', 'email', 'phone', 'address'];

    public function invoices()
    {
        return $this->hasMany(Invoice::class);
    }
}
```

app/Models/Invoice.php

```
<?php

namespace App\Models;

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

class Invoice extends Model
{
    use HasFactory;

     protected $fillable = ['client_id', 'invoice_number', 'invoice_date', 'due_date', 'subtotal',
'gst_amount', 'total', 'notes'];

    protected $casts = [
        'invoice_date' => 'date',
        'due_date' => 'date',
    ];

    public function client()
    {
        return $this->belongsTo(Client::class);
    }

    public function items()
    {
        return $this->hasMany(InvoiceItem::class);
    }

    public function calculateTotals()
    {
        $subtotal = $this->items->sum(function($item) {
            return ($item->price / 1.1) * $item->quantity; // Assuming 10% GST
        });

        $this->subtotal = $subtotal;
        $this->gst_amount = $this->items->sum(function($item) {
            return $item->price * $item->quantity - ($item->price / 1.1) * $item->quantity;
        });

        $this->total = $subtotal + $this->gst_amount;
        $this->save();
    }
}
```

app/Models/InvoiceItem.php

```
<?php

namespace App\Models;

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

class InvoiceItem extends Model
{
    use HasFactory;

    protected $fillable = ['invoice_id', 'product_id', 'quantity', 'price'];

    public function invoice()
    {
        return $this->belongsTo(Invoice::class);
    }

    public function product()
    {
        return $this->belongsTo(Product::class);
    }

    public function serialNumbers()
    {
        return $this->hasMany(SerialNumber::class);
    }
}
```

app/Models/Product.php

```
<?php
namespace App\Models;

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

class Product extends Model
{
    use HasFactory;

    protected $fillable = ['name', 'description', 'price', 'has_serial'];

    public function serialNumbers()
    {
        return $this->hasMany(SerialNumber::class);
    }
}
```

app/Models/SerialNumber.php

```
<?php

namespace App\Models;

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

class SerialNumber extends Model
{
    use HasFactory;

    protected $fillable = [
        'product_id',
        'invoice_item_id',
        'serial_number'
    ];

    public function product()
    {
        return $this->belongsTo(Product::class);
    }

    public function invoiceItem()
    {
        return $this->belongsTo(InvoiceItem::class);
    }
}
```

app/Models/Settings.php

```
<?php

namespace App\Models;

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

class Settings extends Model
{
    use HasFactory;

    protected $table = 'system_settings';

    protected $fillable = [
        'setting_key',
        'setting_value',
        'is_public',
    ];

    public $timestamps = false; // Optional: if your table doesn't have created_at/updated_at

    /**
     * Retrieve a setting by key.
     */
    public static function get(string $key, $default = null)
    {
        $setting = self::where('setting_key', $key)->first();
        return $setting ? $setting->setting_value : $default;
    }

    /**
     * Set or update a setting.
     */
    public static function set(string $key, $value, bool $isPublic = false)
    {
        return self::updateOrCreate(
            ['setting_key' => $key],
            ['setting_value' => $value, 'is_public' => $isPublic]
        );
    }
}
```

app/Models/User.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;
use Laravel\Sanctum\HasApiTokens;

class User extends Authenticatable
{
    use HasApiTokens, HasFactory, Notifiable;

    protected $fillable = [
        'name',
        'email',
        'password',
    ];

    protected $hidden = [
        'password',
        'remember_token',
    ];

    protected $casts = [
        'email_verified_at' => 'datetime',
    ];
}
```

app/Providers/AppServiceProvider.php

```
<?php

namespace App\Providers;

use Illuminate\Support\ServiceProvider;

class AppServiceProvider extends ServiceProvider
{
    /**
     * Register any application services.
     */
    public function register(): void
    {
        //
    }

    /**
     * Bootstrap any application services.
     */
    public function boot(): void
    {
        //
    }
}
```

app/Providers/AuthServiceProvider.php

```php
<?php

namespace App\Providers;

// use Illuminate\Support\Facades\Gate;
use Illuminate\Foundation\Support\Providers\AuthServiceProvider as ServiceProvider;

class AuthServiceProvider extends ServiceProvider
{
    /**
     * The model to policy mappings for the application.
     *
     * @var array<class-string, class-string>
     */
    protected $policies = [
        //
    ];

    /**
     * Register any authentication / authorization services.
     */
    public function boot(): void
    {
        //
    }
}
```

app/Providers/BroadcastServiceProvider.php

```
<?php

namespace App\Providers;

use Illuminate\Support\Facades\Broadcast;
use Illuminate\Support\ServiceProvider;

class BroadcastServiceProvider extends ServiceProvider
{
    /**
     * Bootstrap any application services.
     */
    public function boot(): void
    {
        Broadcast::routes();

        require base_path('routes/channels.php');
    }
}
```

app/Providers/EventServiceProvider.php

```php
<?php

namespace App\Providers;

use Illuminate\Auth\Events\Registered;
use Illuminate\Auth\Listeners\SendEmailVerificationNotification;
use Illuminate\Foundation\Support\Providers\EventServiceProvider as ServiceProvider;
use Illuminate\Support\Facades\Event;

class EventServiceProvider extends ServiceProvider
{
    /**
     * The event to listener mappings for the application.
     *
     * @var array<class-string, array<int, class-string>>
     */
    protected $listen = [
        Registered::class => [
            SendEmailVerificationNotification::class,
        ],
    ];

    /**
     * Register any events for your application.
     */
    public function boot(): void
    {
        //
    }

    /**
     * Determine if events and listeners should be automatically discovered.
     */
    public function shouldDiscoverEvents(): bool
    {
        return false;
    }
}
```

app/Providers/RouteServiceProvider.php

```
<?php

namespace App\Providers;

use Illuminate\Cache\RateLimiting\Limit;
use Illuminate\Foundation\Support\Providers\RouteServiceProvider as ServiceProvider;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\RateLimiter;
use Illuminate\Support\Facades\Route;

class RouteServiceProvider extends ServiceProvider
{
    /**
     * The path to your application's "home" route.
     *
     * Typically, users are redirected here after authentication.
     *
     * @var string
     */
    public const HOME = '/dashboard';

    /**
     * Define your route model bindings, pattern filters, and other route configuration.
     */
    public function boot(): void
    {
        RateLimiter::for('api', function (Request $request) {
            return Limit::perMinute(60)->by($request->user()?->id ?: $request->ip());
        });

        $this->routes(function () {
            Route::middleware('api')
                ->prefix('api')
                ->group(base_path('routes/api.php'));

            Route::middleware('web')
                ->group(base_path('routes/web.php'));
        });
    }
}
```

app/View/Components/AppLayout.php

```
<?php

namespace App\View\Components;

use Illuminate\View\Component;
use Illuminate\View\View;

class AppLayout extends Component
{
    /**
     * Get the view / contents that represents the component.
     */
    public function render(): View
    {
        return view('layouts.app');
    }
}
```

app/View/Components/GuestLayout.php

```
<?php

namespace App\View\Components;

use Illuminate\View\Component;
use Illuminate\View\View;

class GuestLayout extends Component
{
    /**
     * Get the view / contents that represents the component.
     */
    public function render(): View
    {
        return view('layouts.guest');
    }
}
```

config/app.php

```
<?php

use Illuminate\Support\Facades\Facade;
use Illuminate\Support\ServiceProvider;

return [

    /*
    |--------------------------------------------------------------------------
    | Application Name
    |--------------------------------------------------------------------------
    |
    | This value is the name of your application. This value is used when the
    | framework needs to place the application's name in a notification or
    | any other location as required by the application or its packages.
    |
    */

    'name' => env('APP_NAME', 'Laravel'),

    /*
    |--------------------------------------------------------------------------
    | Application Environment
    |--------------------------------------------------------------------------
    |
    | This value determines the "environment" your application is currently
    | running in. This may determine how you prefer to configure various
    | services the application utilizes. Set this in your ".env" file.
    |
    */

    'env' => env('APP_ENV', 'production'),

    /*
    |--------------------------------------------------------------------------
    | Application Debug Mode
    |--------------------------------------------------------------------------
    |
    | When your application is in debug mode, detailed error messages with
    | stack traces will be shown on every error that occurs within your
    | application. If disabled, a simple generic error page is shown.
    |
    */

    'debug' => (bool) env('APP_DEBUG', false),

    /*
    |--------------------------------------------------------------------------
    | Application URL
    |--------------------------------------------------------------------------
    |
```

```
| This URL is used by the console to properly generate URLs when using
| the Artisan command line tool. You should set this to the root of
| your application so that it is used when running Artisan tasks.
|
*/

'url' => env('APP_URL', 'http://localhost'),

'asset_url' => env('ASSET_URL'),

/*
|--------------------------------------------------------------------------
| Application Timezone
|--------------------------------------------------------------------------
|
| Here you may specify the default timezone for your application, which
| will be used by the PHP date and date-time functions. We have gone
| ahead and set this to a sensible default for you out of the box.
|
*/

'timezone' => 'UTC',

/*
|--------------------------------------------------------------------------
| Application Locale Configuration
|--------------------------------------------------------------------------
|
| The application locale determines the default locale that will be used
| by the translation service provider. You are free to set this value
| to any of the locales which will be supported by the application.
|
*/

'locale' => 'en',

/*
|--------------------------------------------------------------------------
| Application Fallback Locale
|--------------------------------------------------------------------------
|
| The fallback locale determines the locale to use when the current one
| is not available. You may change the value to correspond to any of
| the language folders that are provided through your application.
|
*/

'fallback_locale' => 'en',

/*
|--------------------------------------------------------------------------
| Faker Locale
|--------------------------------------------------------------------------
|
```

```
| This locale will be used by the Faker PHP library when generating fake
| data for your database seeds. For example, this will be used to get
| localized telephone numbers, street address information and more.
|
*/

'faker_locale' => 'en_US',

/*
|--------------------------------------------------------------------------
| Encryption Key
|--------------------------------------------------------------------------
|
| This key is used by the Illuminate encrypter service and should be set
| to a random, 32 character string, otherwise these encrypted strings
| will not be safe. Please do this before deploying an application!
|
*/

'key' => env('APP_KEY'),

'cipher' => 'AES-256-CBC',

/*
|--------------------------------------------------------------------------
| Maintenance Mode Driver
|--------------------------------------------------------------------------
|
| These configuration options determine the driver used to determine and
| manage Laravel's "maintenance mode" status. The "cache" driver will
| allow maintenance mode to be controlled across multiple machines.
|
| Supported drivers: "file", "cache"
|
*/

'maintenance' => [
    'driver' => 'file',
    // 'store' => 'redis',
],

/*
|--------------------------------------------------------------------------
| Autoloaded Service Providers
|--------------------------------------------------------------------------
|
| The service providers listed here will be automatically loaded on the
| request to your application. Feel free to add your own services to
| this array to grant expanded functionality to your applications.
|
*/

'providers' => ServiceProvider::defaultProviders()->merge([
    /*
```

```
         * Package Service Providers...
         */

        /*
         * Application Service Providers...
         */
        App\Providers\AppServiceProvider::class,
        App\Providers\AuthServiceProvider::class,
        // App\Providers\BroadcastServiceProvider::class,
        App\Providers\EventServiceProvider::class,
        App\Providers\RouteServiceProvider::class,
    ])->toArray(),

    /*
    |--------------------------------------------------------------------------
    | Class Aliases
    |--------------------------------------------------------------------------
    |
    | This array of class aliases will be registered when this application
    | is started. However, feel free to register as many as you wish as
    | the aliases are "lazy" loaded so they don't hinder performance.
    |
    */

    'aliases' => Facade::defaultAliases()->merge([
        // 'Example' => App\Facades\Example::class,
    ])->toArray(),

];
```

config/auth.php

```
<?php

return [

    /*
    |--------------------------------------------------------------------------
    | Authentication Defaults
    |--------------------------------------------------------------------------
    |
    | This option controls the default authentication "guard" and password
    | reset options for your application. You may change these defaults
    | as required, but they're a perfect start for most applications.
    |
    */

    'defaults' => [
        'guard' => 'web',
        'passwords' => 'users',
    ],

    /*
    |--------------------------------------------------------------------------
    | Authentication Guards
    |--------------------------------------------------------------------------
    |
    | Next, you may define every authentication guard for your application.
    | Of course, a great default configuration has been defined for you
    | here which uses session storage and the Eloquent user provider.
    |
    | All authentication drivers have a user provider. This defines how the
    | users are actually retrieved out of your database or other storage
    | mechanisms used by this application to persist your user's data.
    |
    | Supported: "session"
    |
    */

    'guards' => [
        'web' => [
            'driver' => 'session',
            'provider' => 'users',
        ],
    ],

    /*
    |--------------------------------------------------------------------------
    | User Providers
    |--------------------------------------------------------------------------
    |
    | All authentication drivers have a user provider. This defines how the
    | users are actually retrieved out of your database or other storage
```

```
| mechanisms used by this application to persist your user's data.
|
| If you have multiple user tables or models you may configure multiple
| sources which represent each model / table. These sources may then
| be assigned to any extra authentication guards you have defined.
|
| Supported: "database", "eloquent"
|
*/

'providers' => [
    'users' => [
        'driver' => 'eloquent',
        'model' => App\Models\User::class,
    ],

    // 'users' => [
    //     'driver' => 'database',
    //     'table' => 'users',
    // ],
],

/*
|--------------------------------------------------------------------------
| Resetting Passwords
|--------------------------------------------------------------------------
|
| You may specify multiple password reset configurations if you have more
| than one user table or model in the application and you want to have
| separate password reset settings based on the specific user types.
|
| The expiry time is the number of minutes that each reset token will be
| considered valid. This security feature keeps tokens short-lived so
| they have less time to be guessed. You may change this as needed.
|
| The throttle setting is the number of seconds a user must wait before
| generating more password reset tokens. This prevents the user from
| quickly generating a very large amount of password reset tokens.
|
*/

'passwords' => [
    'users' => [
        'provider' => 'users',
        'table' => 'password_reset_tokens',
        'expire' => 60,
        'throttle' => 60,
    ],
],

/*
|--------------------------------------------------------------------------
| Password Confirmation Timeout
|--------------------------------------------------------------------------
```

```
    |
    | Here you may define the amount of seconds before a password confirmation
    | times out and the user is prompted to re-enter their password via the
    | confirmation screen. By default, the timeout lasts for three hours.
    |
    */

    'password_timeout' => 10800,

];
```

config/cache.php

```
<?php

use Illuminate\Support\Str;

return [

    /*
    |--------------------------------------------------------------------------
    | Default Cache Store
    |--------------------------------------------------------------------------
    |
    | This option controls the default cache connection that gets used while
    | using this caching library. This connection is used when another is
    | not explicitly specified when executing a given caching function.
    |
    */

    'default' => env('CACHE_DRIVER', 'file'),

    /*
    |--------------------------------------------------------------------------
    | Cache Stores
    |--------------------------------------------------------------------------
    |
    | Here you may define all of the cache "stores" for your application as
    | well as their drivers. You may even define multiple stores for the
    | same cache driver to group types of items stored in your caches.
    |
    | Supported drivers: "apc", "array", "database", "file",
    |         "memcached", "redis", "dynamodb", "octane", "null"
    |
    */

    'stores' => [

        'apc' => [
            'driver' => 'apc',
        ],

        'array' => [
            'driver' => 'array',
            'serialize' => false,
        ],

        'database' => [
            'driver' => 'database',
            'table' => 'cache',
            'connection' => null,
            'lock_connection' => null,
        ],
```

```
    'file' => [
        'driver' => 'file',
        'path' => storage_path('framework/cache/data'),
        'lock_path' => storage_path('framework/cache/data'),
    ],

    'memcached' => [
        'driver' => 'memcached',
        'persistent_id' => env('MEMCACHED_PERSISTENT_ID'),
        'sasl' => [
            env('MEMCACHED_USERNAME'),
            env('MEMCACHED_PASSWORD'),
        ],
        'options' => [
            // Memcached::OPT_CONNECT_TIMEOUT => 2000,
        ],
        'servers' => [
            [
                'host' => env('MEMCACHED_HOST', '127.0.0.1'),
                'port' => env('MEMCACHED_PORT', 11211),
                'weight' => 100,
            ],
        ],
    ],

    'redis' => [
        'driver' => 'redis',
        'connection' => 'cache',
        'lock_connection' => 'default',
    ],

    'dynamodb' => [
        'driver' => 'dynamodb',
        'key' => env('AWS_ACCESS_KEY_ID'),
        'secret' => env('AWS_SECRET_ACCESS_KEY'),
        'region' => env('AWS_DEFAULT_REGION', 'us-east-1'),
        'table' => env('DYNAMODB_CACHE_TABLE', 'cache'),
        'endpoint' => env('DYNAMODB_ENDPOINT'),
    ],

    'octane' => [
        'driver' => 'octane',
    ],

],

/*
|--------------------------------------------------------------------------
| Cache Key Prefix
|--------------------------------------------------------------------------
|
| When utilizing the APC, database, memcached, Redis, or DynamoDB cache
| stores there might be other applications using the same cache. For
| that reason, you may prefix every cache key to avoid collisions.
```

```
    |
    */

    'prefix' => env('CACHE_PREFIX', Str::slug(env('APP_NAME', 'laravel'), '_').'_cache_'),

];
```

config/database.php

```
<?php

use Illuminate\Support\Str;

return [

    /*
    |--------------------------------------------------------------------------
    | Default Database Connection Name
    |--------------------------------------------------------------------------
    |
    | Here you may specify which of the database connections below you wish
    | to use as your default connection for all database work. Of course
    | you may use many connections at once using the Database library.
    |
    */

    'default' => env('DB_CONNECTION', 'mysql'),

    /*
    |--------------------------------------------------------------------------
    | Database Connections
    |--------------------------------------------------------------------------
    |
    | Here are each of the database connections setup for your application.
    | Of course, examples of configuring each database platform that is
    | supported by Laravel is shown below to make development simple.
    |
    |
    | All database work in Laravel is done through the PHP PDO facilities
    | so make sure you have the driver for your particular database of
    | choice installed on your machine before you begin development.
    |
    */

    'connections' => [

        'sqlite' => [
            'driver' => 'sqlite',
            'url' => env('DATABASE_URL'),
            'database' => env('DB_DATABASE', database_path('database.sqlite')),
            'prefix' => '',
            'foreign_key_constraints' => env('DB_FOREIGN_KEYS', true),
        ],

        'mysql' => [
            'driver' => 'mysql',
            'url' => env('DATABASE_URL'),
            'host' => env('DB_HOST', '127.0.0.1'),
            'port' => env('DB_PORT', '3306'),
            'database' => env('DB_DATABASE', 'forge'),
```

```
        'username' => env('DB_USERNAME', 'forge'),
        'password' => env('DB_PASSWORD', ''),
        'unix_socket' => env('DB_SOCKET', ''),
        'charset' => 'utf8mb4',
        'collation' => 'utf8mb4_unicode_ci',
        'prefix' => '',
        'prefix_indexes' => true,
        'strict' => true,
        'engine' => null,
        'options' => extension_loaded('pdo_mysql') ? array_filter([
            PDO::MYSQL_ATTR_SSL_CA => env('MYSQL_ATTR_SSL_CA'),
        ]) : [],
    ],

    'pgsql' => [
        'driver' => 'pgsql',
        'url' => env('DATABASE_URL'),
        'host' => env('DB_HOST', '127.0.0.1'),
        'port' => env('DB_PORT', '5432'),
        'database' => env('DB_DATABASE', 'forge'),
        'username' => env('DB_USERNAME', 'forge'),
        'password' => env('DB_PASSWORD', ''),
        'charset' => 'utf8',
        'prefix' => '',
        'prefix_indexes' => true,
        'search_path' => 'public',
        'sslmode' => 'prefer',
    ],

    'sqlsrv' => [
        'driver' => 'sqlsrv',
        'url' => env('DATABASE_URL'),
        'host' => env('DB_HOST', 'localhost'),
        'port' => env('DB_PORT', '1433'),
        'database' => env('DB_DATABASE', 'forge'),
        'username' => env('DB_USERNAME', 'forge'),
        'password' => env('DB_PASSWORD', ''),
        'charset' => 'utf8',
        'prefix' => '',
        'prefix_indexes' => true,
        // 'encrypt' => env('DB_ENCRYPT', 'yes'),
        // 'trust_server_certificate' => env('DB_TRUST_SERVER_CERTIFICATE', 'false'),
    ],

],

/*
|--------------------------------------------------------------------------
| Migration Repository Table
|--------------------------------------------------------------------------
|
| This table keeps track of all the migrations that have already run for
| your application. Using this information, we can determine which of
| the migrations on disk haven't actually been run in the database.
```

```
    |
    */

    'migrations' => 'migrations',

    /*
    |--------------------------------------------------------------------------
    | Redis Databases
    |--------------------------------------------------------------------------
    |
    | Redis is an open source, fast, and advanced key-value store that also
    | provides a richer body of commands than a typical key-value system
    | such as APC or Memcached. Laravel makes it easy to dig right in.
    |
    */

    'redis' => [

        'client' => env('REDIS_CLIENT', 'phpredis'),

        'options' => [
            'cluster' => env('REDIS_CLUSTER', 'redis'),
                        'prefix'  =>  env('REDIS_PREFIX',  Str::slug(env('APP_NAME',  'laravel'),
'_').'_database_'),
        ],

        'default' => [
            'url' => env('REDIS_URL'),
            'host' => env('REDIS_HOST', '127.0.0.1'),
            'username' => env('REDIS_USERNAME'),
            'password' => env('REDIS_PASSWORD'),
            'port' => env('REDIS_PORT', '6379'),
            'database' => env('REDIS_DB', '0'),
        ],

        'cache' => [
            'url' => env('REDIS_URL'),
            'host' => env('REDIS_HOST', '127.0.0.1'),
            'username' => env('REDIS_USERNAME'),
            'password' => env('REDIS_PASSWORD'),
            'port' => env('REDIS_PORT', '6379'),
            'database' => env('REDIS_CACHE_DB', '1'),
        ],

    ],

];
```

config/filesystems.php

```
<?php

return [

    /*
    |--------------------------------------------------------------------------
    | Default Filesystem Disk
    |--------------------------------------------------------------------------
    |
    | Here you may specify the default filesystem disk that should be used
    | by the framework. The "local" disk, as well as a variety of cloud
    | based disks are available to your application. Just store away!
    |
    */

    'default' => env('FILESYSTEM_DISK', 'local'),

    /*
    |--------------------------------------------------------------------------
    | Filesystem Disks
    |--------------------------------------------------------------------------
    |
    | Here you may configure as many filesystem "disks" as you wish, and you
    | may even configure multiple disks of the same driver. Defaults have
    | been set up for each driver as an example of the required values.
    |
    | Supported Drivers: "local", "ftp", "sftp", "s3"
    |
    */

    'disks' => [

        'local' => [
            'driver' => 'local',
            'root' => storage_path('app'),
            'throw' => false,
        ],

        'public' => [
            'driver' => 'local',
            'root' => storage_path('app/public'),
            'url' => env('APP_URL').'/storage',
            'visibility' => 'public',
            'throw' => false,
        ],

        's3' => [
            'driver' => 's3',
            'key' => env('AWS_ACCESS_KEY_ID'),
            'secret' => env('AWS_SECRET_ACCESS_KEY'),
            'region' => env('AWS_DEFAULT_REGION'),
```

```
            'bucket' => env('AWS_BUCKET'),
            'url' => env('AWS_URL'),
            'endpoint' => env('AWS_ENDPOINT'),
            'use_path_style_endpoint' => env('AWS_USE_PATH_STYLE_ENDPOINT', false),
            'throw' => false,
        ],

    ],

    /*
    |--------------------------------------------------------------------------
    | Symbolic Links
    |--------------------------------------------------------------------------
    |
    | Here you may configure the symbolic links that will be created when the
    | `storage:link` Artisan command is executed. The array keys should be
    | the locations of the links and the values should be their targets.
    |
    */

    'links' => [
        public_path('storage') => storage_path('app/public'),
    ],

];
```

config/hashing.php

```
<?php

return [

    /*
    |--------------------------------------------------------------------------
    | Default Hash Driver
    |--------------------------------------------------------------------------
    |
    | This option controls the default hash driver that will be used to hash
    | passwords for your application. By default, the bcrypt algorithm is
    | used; however, you remain free to modify this option if you wish.
    |
    | Supported: "bcrypt", "argon", "argon2id"
    |
    */

    'driver' => 'bcrypt',

    /*
    |--------------------------------------------------------------------------
    | Bcrypt Options
    |--------------------------------------------------------------------------
    |
    | Here you may specify the configuration options that should be used when
    | passwords are hashed using the Bcrypt algorithm. This will allow you
    | to control the amount of time it takes to hash the given password.
    |
    */

    'bcrypt' => [
        'rounds' => env('BCRYPT_ROUNDS', 12),
        'verify' => true,
    ],

    /*
    |--------------------------------------------------------------------------
    | Argon Options
    |--------------------------------------------------------------------------
    |
    | Here you may specify the configuration options that should be used when
    | passwords are hashed using the Argon algorithm. These will allow you
    | to control the amount of time it takes to hash the given password.
    |
    */

    'argon' => [
        'memory' => 65536,
        'threads' => 1,
        'time' => 4,
        'verify' => true,
```

```
    ],

];
```

config/logging.php

```
<?php

use Monolog\Handler\NullHandler;
use Monolog\Handler\StreamHandler;
use Monolog\Handler\SyslogUdpHandler;
use Monolog\Processor\PsrLogMessageProcessor;

return [

    /*
    |--------------------------------------------------------------------------
    | Default Log Channel
    |--------------------------------------------------------------------------
    |
    | This option defines the default log channel that gets used when writing
    | messages to the logs. The name specified in this option should match
    | one of the channels defined in the "channels" configuration array.
    |
    */

    'default' => env('LOG_CHANNEL', 'stack'),

    /*
    |--------------------------------------------------------------------------
    | Deprecations Log Channel
    |--------------------------------------------------------------------------
    |
    | This option controls the log channel that should be used to log warnings
    | regarding deprecated PHP and library features. This allows you to get
    | your application ready for upcoming major versions of dependencies.
    |
    */

    'deprecations' => [
        'channel' => env('LOG_DEPRECATIONS_CHANNEL', 'null'),
        'trace' => false,
    ],

    /*
    |--------------------------------------------------------------------------
    | Log Channels
    |--------------------------------------------------------------------------
    |
    | Here you may configure the log channels for your application. Out of
    | the box, Laravel uses the Monolog PHP logging library. This gives
    | you a variety of powerful log handlers / formatters to utilize.
    |
    | Available Drivers: "single", "daily", "slack", "syslog",
    |                    "errorlog", "monolog",
    |                    "custom", "stack"
    |
```

```
*/

'channels' => [
    'stack' => [
        'driver' => 'stack',
        'channels' => ['single'],
        'ignore_exceptions' => false,
    ],

    'single' => [
        'driver' => 'single',
        'path' => storage_path('logs/laravel.log'),
        'level' => env('LOG_LEVEL', 'debug'),
        'replace_placeholders' => true,
    ],

    'daily' => [
        'driver' => 'daily',
        'path' => storage_path('logs/laravel.log'),
        'level' => env('LOG_LEVEL', 'debug'),
        'days' => 14,
        'replace_placeholders' => true,
    ],

    'slack' => [
        'driver' => 'slack',
        'url' => env('LOG_SLACK_WEBHOOK_URL'),
        'username' => 'Laravel Log',
        'emoji' => ':boom:',
        'level' => env('LOG_LEVEL', 'critical'),
        'replace_placeholders' => true,
    ],

    'papertrail' => [
        'driver' => 'monolog',
        'level' => env('LOG_LEVEL', 'debug'),
        'handler' => env('LOG_PAPERTRAIL_HANDLER', SyslogUdpHandler::class),
        'handler_with' => [
            'host' => env('PAPERTRAIL_URL'),
            'port' => env('PAPERTRAIL_PORT'),
            'connectionString' => 'tls://'.env('PAPERTRAIL_URL').':'.env('PAPERTRAIL_PORT'),
        ],
        'processors' => [PsrLogMessageProcessor::class],
    ],

    'stderr' => [
        'driver' => 'monolog',
        'level' => env('LOG_LEVEL', 'debug'),
        'handler' => StreamHandler::class,
        'formatter' => env('LOG_STDERR_FORMATTER'),
        'with' => [
            'stream' => 'php://stderr',
        ],
        'processors' => [PsrLogMessageProcessor::class],
```

```
        ],

        'syslog' => [
            'driver' => 'syslog',
            'level' => env('LOG_LEVEL', 'debug'),
            'facility' => LOG_USER,
            'replace_placeholders' => true,
        ],

        'errorlog' => [
            'driver' => 'errorlog',
            'level' => env('LOG_LEVEL', 'debug'),
            'replace_placeholders' => true,
        ],

        'null' => [
            'driver' => 'monolog',
            'handler' => NullHandler::class,
        ],

        'emergency' => [
            'path' => storage_path('logs/laravel.log'),
        ],
    ],

];
```

config/mail.php

```
<?php

return [

    /*
    |--------------------------------------------------------------------------
    | Default Mailer
    |--------------------------------------------------------------------------
    |
    | This option controls the default mailer that is used to send any email
    | messages sent by your application. Alternative mailers may be setup
    | and used as needed; however, this mailer will be used by default.
    |
    */

    'default' => env('MAIL_MAILER', 'smtp'),

    /*
    |--------------------------------------------------------------------------
    | Mailer Configurations
    |--------------------------------------------------------------------------
    |
    | Here you may configure all of the mailers used by your application plus
    | their respective settings. Several examples have been configured for
    | you and you are free to add your own as your application requires.
    |
    | Laravel supports a variety of mail "transport" drivers to be used while
    | sending an e-mail. You will specify which one you are using for your
    | mailers below. You are free to add additional mailers as required.
    |
    | Supported: "smtp", "sendmail", "mailgun", "ses", "ses-v2",
    |            "postmark", "log", "array", "failover", "roundrobin"
    |
    */

    'mailers' => [
        'smtp' => [
            'transport' => 'smtp',
            'url' => env('MAIL_URL'),
            'host' => env('MAIL_HOST', 'smtp.mailgun.org'),
            'port' => env('MAIL_PORT', 587),
            'encryption' => env('MAIL_ENCRYPTION', 'tls'),
            'username' => env('MAIL_USERNAME'),
            'password' => env('MAIL_PASSWORD'),
            'timeout' => null,
            'local_domain' => env('MAIL_EHLO_DOMAIN'),
        ],

        'ses' => [
            'transport' => 'ses',
        ],
```

```
    'postmark' => [
        'transport' => 'postmark',
        // 'message_stream_id' => null,
        // 'client' => [
        //     'timeout' => 5,
        // ],
    ],

    'mailgun' => [
        'transport' => 'mailgun',
        // 'client' => [
        //     'timeout' => 5,
        // ],
    ],

    'sendmail' => [
        'transport' => 'sendmail',
        'path' => env('MAIL_SENDMAIL_PATH', '/usr/sbin/sendmail -bs -i'),
    ],

    'log' => [
        'transport' => 'log',
        'channel' => env('MAIL_LOG_CHANNEL'),
    ],

    'array' => [
        'transport' => 'array',
    ],

    'failover' => [
        'transport' => 'failover',
        'mailers' => [
            'smtp',
            'log',
        ],
    ],

    'roundrobin' => [
        'transport' => 'roundrobin',
        'mailers' => [
            'ses',
            'postmark',
        ],
    ],
],

/*
|--------------------------------------------------------------------------
| Global "From" Address
|--------------------------------------------------------------------------
|
| You may wish for all e-mails sent by your application to be sent from
| the same address. Here, you may specify a name and address that is
```

```
    | used globally for all e-mails that are sent by your application.
    |
    */

    'from' => [
        'address' => env('MAIL_FROM_ADDRESS', 'hello@example.com'),
        'name' => env('MAIL_FROM_NAME', 'Example'),
    ],

    /*
    |--------------------------------------------------------------------------
    | Markdown Mail Settings
    |--------------------------------------------------------------------------
    |
    | If you are using Markdown based email rendering, you may configure your
    | theme and component paths here, allowing you to customize the design
    | of the emails. Or, you may simply stick with the Laravel defaults!
    |
    */

    'markdown' => [
        'theme' => 'default',

        'paths' => [
            resource_path('views/vendor/mail'),
        ],
    ],

];
```

config/queue.php

```
<?php

return [

    /*
    |--------------------------------------------------------------------------
    | Default Queue Connection Name
    |--------------------------------------------------------------------------
    |
    | Laravel's queue API supports an assortment of back-ends via a single
    | API, giving you convenient access to each back-end using the same
    | syntax for every one. Here you may define a default connection.
    |
    */

    'default' => env('QUEUE_CONNECTION', 'sync'),

    /*
    |--------------------------------------------------------------------------
    | Queue Connections
    |--------------------------------------------------------------------------
    |
    | Here you may configure the connection information for each server that
    | is used by your application. A default configuration has been added
    | for each back-end shipped with Laravel. You are free to add more.
    |
    | Drivers: "sync", "database", "beanstalkd", "sqs", "redis", "null"
    |
    */

    'connections' => [

        'sync' => [
            'driver' => 'sync',
        ],

        'database' => [
            'driver' => 'database',
            'table' => 'jobs',
            'queue' => 'default',
            'retry_after' => 90,
            'after_commit' => false,
        ],

        'beanstalkd' => [
            'driver' => 'beanstalkd',
            'host' => 'localhost',
            'queue' => 'default',
            'retry_after' => 90,
            'block_for' => 0,
            'after_commit' => false,
```

```
    ],

    'sqs' => [
        'driver' => 'sqs',
        'key' => env('AWS_ACCESS_KEY_ID'),
        'secret' => env('AWS_SECRET_ACCESS_KEY'),
        'prefix' => env('SQS_PREFIX', 'https://sqs.us-east-1.amazonaws.com/your-account-id'),
        'queue' => env('SQS_QUEUE', 'default'),
        'suffix' => env('SQS_SUFFIX'),
        'region' => env('AWS_DEFAULT_REGION', 'us-east-1'),
        'after_commit' => false,
    ],

    'redis' => [
        'driver' => 'redis',
        'connection' => 'default',
        'queue' => env('REDIS_QUEUE', 'default'),
        'retry_after' => 90,
        'block_for' => null,
        'after_commit' => false,
    ],

],

/*
|--------------------------------------------------------------------------
| Job Batching
|--------------------------------------------------------------------------
|
| The following options configure the database and table that store job
| batching information. These options can be updated to any database
| connection and table which has been defined by your application.
|
*/

'batching' => [
    'database' => env('DB_CONNECTION', 'mysql'),
    'table' => 'job_batches',
],

/*
|--------------------------------------------------------------------------
| Failed Queue Jobs
|--------------------------------------------------------------------------
|
| These options configure the behavior of failed queue job logging so you
| can control which database and table are used to store the jobs that
| have failed. You may change them to any database / table you wish.
|
*/

'failed' => [
    'driver' => env('QUEUE_FAILED_DRIVER', 'database-uuids'),
    'database' => env('DB_CONNECTION', 'mysql'),
```

```
        'table' => 'failed_jobs',
    ],

];
```

config/services.php

```
<?php

return [

    /*
    |--------------------------------------------------------------------------
    | Third Party Services
    |--------------------------------------------------------------------------
    |
    | This file is for storing the credentials for third party services such
    | as Mailgun, Postmark, AWS and more. This file provides the de facto
    | location for this type of information, allowing packages to have
    | a conventional file to locate the various service credentials.
    |
    */

    'mailgun' => [
        'domain' => env('MAILGUN_DOMAIN'),
        'secret' => env('MAILGUN_SECRET'),
        'endpoint' => env('MAILGUN_ENDPOINT', 'api.mailgun.net'),
        'scheme' => 'https',
    ],

    'postmark' => [
        'token' => env('POSTMARK_TOKEN'),
    ],

    'ses' => [
        'key' => env('AWS_ACCESS_KEY_ID'),
        'secret' => env('AWS_SECRET_ACCESS_KEY'),
        'region' => env('AWS_DEFAULT_REGION', 'us-east-1'),
    ],

];
```

config/session.php

```
<?php

use Illuminate\Support\Str;

return [

    /*
    |--------------------------------------------------------------------------
    | Default Session Driver
    |--------------------------------------------------------------------------
    |
    | This option controls the default session "driver" that will be used on
    | requests. By default, we will use the lightweight native driver but
    | you may specify any of the other wonderful drivers provided here.
    |
    | Supported: "file", "cookie", "database", "apc",
    |            "memcached", "redis", "dynamodb", "array"
    |
    */

    'driver' => env('SESSION_DRIVER', 'file'),

    /*
    |--------------------------------------------------------------------------
    | Session Lifetime
    |--------------------------------------------------------------------------
    |
    | Here you may specify the number of minutes that you wish the session
    | to be allowed to remain idle before it expires. If you want them
    | to immediately expire on the browser closing, set that option.
    |
    */

    'lifetime' => env('SESSION_LIFETIME', 120),

    'expire_on_close' => false,

    /*
    |--------------------------------------------------------------------------
    | Session Encryption
    |--------------------------------------------------------------------------
    |
    | This option allows you to easily specify that all of your session data
    | should be encrypted before it is stored. All encryption will be run
    | automatically by Laravel and you can use the Session like normal.
    |
    */

    'encrypt' => false,

    /*
```

```
|--------------------------------------------------------------------------
| Session File Location
|--------------------------------------------------------------------------
|
| When using the native session driver, we need a location where session
| files may be stored. A default has been set for you but a different
| location may be specified. This is only needed for file sessions.
|
*/

'files' => storage_path('framework/sessions'),

/*
|--------------------------------------------------------------------------
| Session Database Connection
|--------------------------------------------------------------------------
|
| When using the "database" or "redis" session drivers, you may specify a
| connection that should be used to manage these sessions. This should
| correspond to a connection in your database configuration options.
|
*/

'connection' => env('SESSION_CONNECTION'),

/*
|--------------------------------------------------------------------------
| Session Database Table
|--------------------------------------------------------------------------
|
| When using the "database" session driver, you may specify the table we
| should use to manage the sessions. Of course, a sensible default is
| provided for you; however, you are free to change this as needed.
|
*/

'table' => 'sessions',

/*
|--------------------------------------------------------------------------
| Session Cache Store
|--------------------------------------------------------------------------
|
| While using one of the framework's cache driven session backends you may
| list a cache store that should be used for these sessions. This value
| must match with one of the application's configured cache "stores".
|
| Affects: "apc", "dynamodb", "memcached", "redis"
|
*/

'store' => env('SESSION_STORE'),

/*
```

```
|--------------------------------------------------------------------------
| Session Sweeping Lottery
|--------------------------------------------------------------------------
|
| Some session drivers must manually sweep their storage location to get
| rid of old sessions from storage. Here are the chances that it will
| happen on a given request. By default, the odds are 2 out of 100.
|
*/

'lottery' => [2, 100],

/*
|--------------------------------------------------------------------------
| Session Cookie Name
|--------------------------------------------------------------------------
|
| Here you may change the name of the cookie used to identify a session
| instance by ID. The name specified here will get used every time a
| new session cookie is created by the framework for every driver.
|
*/

'cookie' => env(
    'SESSION_COOKIE',
    Str::slug(env('APP_NAME', 'laravel'), '_').'_session'
),

/*
|--------------------------------------------------------------------------
| Session Cookie Path
|--------------------------------------------------------------------------
|
| The session cookie path determines the path for which the cookie will
| be regarded as available. Typically, this will be the root path of
| your application but you are free to change this when necessary.
|
*/

'path' => '/',

/*
|--------------------------------------------------------------------------
| Session Cookie Domain
|--------------------------------------------------------------------------
|
| Here you may change the domain of the cookie used to identify a session
| in your application. This will determine which domains the cookie is
| available to in your application. A sensible default has been set.
|
*/

'domain' => env('SESSION_DOMAIN'),
```

```
/*
|--------------------------------------------------------------------------
| HTTPS Only Cookies
|--------------------------------------------------------------------------
|
| By setting this option to true, session cookies will only be sent back
| to the server if the browser has a HTTPS connection. This will keep
| the cookie from being sent to you when it can't be done securely.
|
*/

'secure' => env('SESSION_SECURE_COOKIE'),

/*
|--------------------------------------------------------------------------
| HTTP Access Only
|--------------------------------------------------------------------------
|
| Setting this value to true will prevent JavaScript from accessing the
| value of the cookie and the cookie will only be accessible through
| the HTTP protocol. You are free to modify this option if needed.
|
*/

'http_only' => true,

/*
|--------------------------------------------------------------------------
| Same-Site Cookies
|--------------------------------------------------------------------------
|
| This option determines how your cookies behave when cross-site requests
| take place, and can be used to mitigate CSRF attacks. By default, we
| will set this value to "lax" since this is a secure default value.
|
| Supported: "lax", "strict", "none", null
|
*/

'same_site' => 'lax',

/*
|--------------------------------------------------------------------------
| Partitioned Cookies
|--------------------------------------------------------------------------
|
| Setting this value to true will tie the cookie to the top-level site for
| a cross-site context. Partitioned cookies are accepted by the browser
| when flagged "secure" and the Same-Site attribute is set to "none".
|
*/

'partitioned' => false,
```

```
];
```

config/settings.php

```
// config/settings.php
return [
    'gst_rate' => 0.10, // 10% GST

    'invoice' => [
        'prefix' => 'INV-',
        'next_number' => 1001,
        'default_template' => 'custom',
    ],

    'email' => [
        'driver' => env('MAIL_DRIVER', 'smtp'),
        'host' => env('MAIL_HOST', 'smtp.mailgun.org'),
        'port' => env('MAIL_PORT', 587),
        'username' => env('MAIL_USERNAME'),
        'password' => env('MAIL_PASSWORD'),
        'encryption' => env('MAIL_ENCRYPTION', 'tls'),
        'from' => [
            'address' => env('MAIL_FROM_ADDRESS', 'hello@example.com'),
            'name' => env('MAIL_FROM_NAME', 'Example'),
        ],
    ],
];
```

config/view.php

```
<?php

return [

    /*
    |--------------------------------------------------------------------------
    | View Storage Paths
    |--------------------------------------------------------------------------
    |
    | Most templating systems load templates from disk. Here you may specify
    | an array of paths that should be checked for your views. Of course
    | the usual Laravel view path has already been registered for you.
    |
    */

    'paths' => [
        resource_path('views'),
    ],

    /*
    |--------------------------------------------------------------------------
    | Compiled View Path
    |--------------------------------------------------------------------------
    |
    | This option determines where all the compiled Blade templates will be
    | stored for your application. Typically, this is within the storage
    | directory. However, as usual, you are free to change this value.
    |
    */

    'compiled' => env(
        'VIEW_COMPILED_PATH',
        realpath(storage_path('framework/views'))
    ),

];
```

database/factories/UserFactory.php

```
<?php

namespace Database\Factories;

use Illuminate\Database\Eloquent\Factories\Factory;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Str;

/**
 * @extends \Illuminate\Database\Eloquent\Factories\Factory<\App\Models\User>
 */
class UserFactory extends Factory
{
    /**
     * The current password being used by the factory.
     */
    protected static ?string $password;

    /**
     * Define the model's default state.
     *
     * @return array<string, mixed>
     */
    public function definition(): array
    {
        return [
            'name' => fake()->name(),
            'email' => fake()->unique()->safeEmail(),
            'email_verified_at' => now(),
            'password' => static::$password ??= Hash::make('password'),
            'remember_token' => Str::random(10),
        ];
    }

    /**
     * Indicate that the model's email address should be unverified.
     */
    public function unverified(): static
    {
        return $this->state(fn (array $attributes) => [
            'email_verified_at' => null,
        ]);
    }
}
```

database/migrations/2023_01_01_000000_create_clients_table.php

```
<?php

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

return new class extends Migration {
    public function up(): void
    {
        Schema::create('clients', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->string('email')->unique();
            $table->string('phone')->nullable();
            $table->text('address')->nullable();
            $table->timestamps();
        });
    }

    public function down(): void
    {
        Schema::dropIfExists('clients');
    }
};
```

database/migrations/2023_01_01_000001_create_products_table.php

```
<?php

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

return new class extends Migration {
    public function up(): void
    {
        Schema::create('products', function (Blueprint $table) {
            $table->id();
            $table->string('name');
            $table->text('description')->nullable();
            $table->decimal('price', 10, 2);
            $table->boolean('has_serial')->default(false);
            $table->timestamps();
        });
    }

    public function down(): void
    {
        Schema::dropIfExists('products');
    }
};
```

database/migrations/2023_01_01_000002_create_invoices_table.php

```
<?php

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

return new class extends Migration {
    public function up(): void
    {
        Schema::create('invoices', function (Blueprint $table) {
            $table->id();
            $table->foreignId('client_id')->constrained();
            $table->string('invoice_number')->unique();
            $table->date('invoice_date');
            $table->date('due_date');
            $table->decimal('subtotal', 10, 2);
            $table->decimal('gst_amount', 10, 2);
            $table->decimal('total', 10, 2);
            $table->text('notes')->nullable();
            $table->timestamps();
        });
    }

    public function down(): void
    {
        Schema::dropIfExists('invoices');
    }
};
```

database/migrations/2023_01_01_000003_create_invoice_items_table.php

```
<?php

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

return new class extends Migration {
    public function up(): void
    {
        Schema::create('invoice_items', function (Blueprint $table) {
            $table->id();
            $table->foreignId('invoice_id')->constrained();
            $table->foreignId('product_id')->constrained();
            $table->integer('quantity');
            $table->decimal('price', 10, 2);
            $table->decimal('total', 10, 2);
            $table->timestamps();
        });
    }

    public function down(): void
    {
        Schema::dropIfExists('invoice_items');
    }
};
```

database/migrations/2023_01_01_000005_create_settings_table.php

```
<?php

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

return new class extends Migration {
    public function up(): void
    {
        Schema::create('system_settings', function (Blueprint $table) {
            $table->id();
            $table->string('setting_key');
            $table->text('setting_value');
            $table->boolean('is_public')->default(false);
            $table->timestamps();
        });
    }

    public function down(): void
    {
        Schema::dropIfExists('system_settings');
    }
};
```

database/migrations/2025_05_03_005940_create_users_table.php

```php
<?php

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

return new class extends Migration
{
    public function up()
    {
        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->rememberToken();
            $table->timestamps();
        });
    }

    public function down()
    {
        Schema::dropIfExists('users');
    }
};
```

database/migrations/create_serial_numbers_table.php

```
<?php

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

return new class extends Migration {
    public function up(): void
    {
        Schema::create('serial_numbers', function (Blueprint $table) {
            $table->id();
            $table->foreignId('product_id')->constrained();
            $table->foreignId('invoice_id')->nullable()->constrained();
            $table->string('serial_number');
            $table->timestamps();
        });
    }

    public function down(): void
    {
        Schema::dropIfExists('serial_numbers');
    }
};
```

database/seeders/DatabaseSeeder.php

```
<?php

namespace Database\Seeders;

// use Illuminate\Database\Console\Seeds\WithoutModelEvents;
use Illuminate\Database\Seeder;

class DatabaseSeeder extends Seeder
{
    /**
     * Seed the application's database.
     */
   public function run(): void
{
    $this->call([
        SettingsSeeder::class,
    ]);
}
}
```

database/seeders/SettingsSeeder.php

```
<?php

namespace Database\Seeders;

use App\Models\Settings;
use Illuminate\Database\Seeder;

class SettingsSeeder extends Seeder
{
    public function run()
    {
        Settings::set('gst_rate', 0.10);
        Settings::set('invoice.prefix', 'INV-');
        Settings::set('invoice.next_number', 1001);
        Settings::set('invoice.default_template', 'custom');
    }
}
```

resources/css/app.css

```
@tailwind base;
@tailwind components;
@tailwind utilities;
```

resources/js/app.js

```
import './bootstrap';

import Alpine from 'alpinejs';

window.Alpine = Alpine;

Alpine.start();
```

resources/js/bootstrap.js

```
/**
 * We'll load the axios HTTP library which allows us to easily issue requests
 * to our Laravel back-end. This library automatically handles sending the
 * CSRF token as a header based on the value of the "XSRF" token cookie.
 */

import axios from 'axios';
window.axios = axios;

window.axios.defaults.headers.common['X-Requested-With'] = 'XMLHttpRequest';

/**
 * Echo exposes an expressive API for subscribing to channels and listening
 * for events that are broadcast by Laravel. Echo and event broadcasting
 * allows your team to easily build robust real-time web applications.
 */

// import Echo from 'laravel-echo';

// import Pusher from 'pusher-js';
// window.Pusher = Pusher;

// window.Echo = new Echo({
//     broadcaster: 'pusher',
//     key: import.meta.env.VITE_PUSHER_APP_KEY,
//     cluster: import.meta.env.VITE_PUSHER_APP_CLUSTER ?? 'mt1',
//             wsHost:   import.meta.env.VITE_PUSHER_HOST   ?   import.meta.env.VITE_PUSHER_HOST   :
`ws-${import.meta.env.VITE_PUSHER_APP_CLUSTER}.pusher.com`,
//     wsPort: import.meta.env.VITE_PUSHER_PORT ?? 80,
//     wssPort: import.meta.env.VITE_PUSHER_PORT ?? 443,
//     forceTLS: (import.meta.env.VITE_PUSHER_SCHEME ?? 'https') === 'https',
//     enabledTransports: ['ws', 'wss'],
// });
```

resources/views/Client

resources/views/dashboard.blade.php

```
@extends('layouts.app')

@section('content')
<div class="py-12">
    <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
        <!-- Stats Cards -->
        <div class="grid grid-cols-1 md:grid-cols-2 lg:grid-cols-4 gap-6 mb-6">
            <!-- Total Clients -->
            <div class="bg-white dark:bg-gray-800 overflow-hidden shadow-sm sm:rounded-lg">
                <div class="p-6">
                    <div class="flex items-center">
                        <div class="p-3 rounded-full bg-blue-500 bg-opacity-10">
                             <svg xmlns="http://www.w3.org/2000/svg" class="h-8 w-8 text-blue-500"
fill="none" viewBox="0 0 24 24" stroke="currentColor">
                                                <path stroke-linecap="round" stroke-linejoin="round"
stroke-width="2" d="M12 4.354a4 4 0 110 5.292M15 21H3v-1a6 6 0 0112 0v1zm0 0h6v-1a6 6 0
00-9-5.197M13 7a4 4 0 11-8 0 4 4 0 018 0z" />
                            </svg>
                        </div>
                        <div class="ml-4">
                                <h2 class="font-semibold text-xl text-gray-800 dark:text-gray-200
leading-tight">{{ $totalClients }}</h2>
                            <p class="text-gray-500 dark:text-gray-400">Total Clients</p>
                        </div>
                    </div>
                </div>
            </div>

            <!-- Total Products -->
            <div class="bg-white dark:bg-gray-800 overflow-hidden shadow-sm sm:rounded-lg">
                <div class="p-6">
                    <div class="flex items-center">
                        <div class="p-3 rounded-full bg-green-500 bg-opacity-10">
                             <svg xmlns="http://www.w3.org/2000/svg" class="h-8 w-8 text-green-500"
fill="none" viewBox="0 0 24 24" stroke="currentColor">
                                                <path stroke-linecap="round" stroke-linejoin="round"
stroke-width="2" d="M20 7l-8-4-8 4m16 0l-8 4m8-4v10l-8 4m0-10L4 7m8 4v10M4 7v10l8 4" />
                            </svg>
                        </div>
                        <div class="ml-4">
                                <h2 class="font-semibold text-xl text-gray-800 dark:text-gray-200
leading-tight">{{ $totalProducts }}</h2>
                            <p class="text-gray-500 dark:text-gray-400">Total Products</p>
                        </div>
                    </div>
                </div>
            </div>

            <!-- Total Invoices -->
            <div class="bg-white dark:bg-gray-800 overflow-hidden shadow-sm sm:rounded-lg">
                <div class="p-6">
```

```
                    <div class="flex items-center">
                        <div class="p-3 rounded-full bg-indigo-500 bg-opacity-10">
                                        <svg xmlns="http://www.w3.org/2000/svg" class="h-8 w-8
text-indigo-500" fill="none" viewBox="0 0 24 24" stroke="currentColor">
                                            <path stroke-linecap="round" stroke-linejoin="round"
stroke-width="2" d="M9 12h6m-6 4h6m2 5H7a2 2 0 01-2-2V5a2 2 0 012-2h5.586a1 1 0 01.707.293l5.414
5.414a1 1 0 01.293.707V19a2 2 0 01-2 2z" />
                            </svg>
                        </div>
                        <div class="ml-4">
                                <h2 class="font-semibold text-xl text-gray-800 dark:text-gray-200
leading-tight">{{ $totalInvoices }}</h2>
                            <p class="text-gray-500 dark:text-gray-400">Total Invoices</p>
                        </div>
                    </div>
                </div>
            </div>

            <!-- Total Revenue -->
            <div class="bg-white dark:bg-gray-800 overflow-hidden shadow-sm sm:rounded-lg">
                <div class="p-6">
                    <div class="flex items-center">
                        <div class="p-3 rounded-full bg-red-500 bg-opacity-10">
                              <svg xmlns="http://www.w3.org/2000/svg" class="h-8 w-8 text-red-500"
fill="none" viewBox="0 0 24 24" stroke="currentColor">
                                            <path stroke-linecap="round" stroke-linejoin="round"
stroke-width="2" d="M12 8c-1.657 0-3 .895-3 2s1.343 2 3 2 3 .895 3 2-1.343 2-3 2m0-8c1.11 0
2.08.402 2.599 1M12 8V7m0 1v8m0 0v1m0-1c-1.11 0-2.08-.402-2.599-1M21 12a9 9 0 11-18 0 9 9 0 0118
0z" />
                            </svg>
                        </div>
                        <div class="ml-4">
                                <h2 class="font-semibold text-xl text-gray-800 dark:text-gray-200
leading-tight">${{ number_format($totalRevenue, 2) }}</h2>
                            <p class="text-gray-500 dark:text-gray-400">Total Revenue</p>
                        </div>
                    </div>
                </div>
            </div>
        </div>

        <div class="grid grid-cols-1 lg:grid-cols-3 gap-6">
            <!-- Recent Invoices -->
                  <div class="lg:col-span-2 bg-white dark:bg-gray-800 overflow-hidden shadow-sm
sm:rounded-lg">
                <div class="p-6">
                     <h3 class="text-lg font-semibold text-gray-800 dark:text-gray-200 mb-4">Recent
Invoices</h3>
                    <div class="overflow-x-auto">
                        <table class="min-w-full divide-y divide-gray-200 dark:divide-gray-700">
                            <thead class="bg-gray-50 dark:bg-gray-700">
                                <tr>
                                    <th scope="col" class="px-6 py-3 text-left text-xs font-medium
text-gray-500 dark:text-gray-300 uppercase tracking-wider">Invoice #</th>
```

```
                                    <th scope="col" class="px-6 py-3 text-left text-xs font-medium
text-gray-500 dark:text-gray-300 uppercase tracking-wider">Client</th>
                                    <th scope="col" class="px-6 py-3 text-left text-xs font-medium
text-gray-500 dark:text-gray-300 uppercase tracking-wider">Date</th>
                                    <th scope="col" class="px-6 py-3 text-left text-xs font-medium
text-gray-500 dark:text-gray-300 uppercase tracking-wider">Total</th>
                                    <th scope="col" class="px-6 py-3 text-left text-xs font-medium
text-gray-500 dark:text-gray-300 uppercase tracking-wider">Actions</th>
                                </tr>
                            </thead>
                              <tbody class="bg-white divide-y divide-gray-200 dark:bg-gray-800
dark:divide-gray-700">
                                @forelse ($recentInvoices as $invoice)
                                    <tr>
                                         <td class="px-6 py-4 whitespace-nowrap text-sm font-medium
text-gray-900 dark:text-gray-100">
                                            {{ $invoice->invoice_number }}
                                        </td>
                                                    <td class="px-6 py-4 whitespace-nowrap text-sm
text-gray-500 dark:text-gray-400">
                                            {{ $invoice->client->name }}
                                        </td>
                                                    <td class="px-6 py-4 whitespace-nowrap text-sm
text-gray-500 dark:text-gray-400">
                                            {{ $invoice->invoice_date->format('M d, Y') }}
                                        </td>
                                                    <td class="px-6 py-4 whitespace-nowrap text-sm
text-gray-500 dark:text-gray-400">
                                            ${{ number_format($invoice->total, 2) }}
                                        </td>
                                                    <td class="px-6 py-4 whitespace-nowrap text-sm
font-medium">
                                                       <a href="{{ route('invoices.show', $invoice) }}"
class="text-blue-600 hover:text-blue-900 dark:text-blue-400 dark:hover:text-blue-600">View</a>
                                        </td>
                                    </tr>
                                @empty
                                    <tr>
                                         <td colspan="5" class="px-6 py-4 whitespace-nowrap text-sm
text-gray-500 dark:text-gray-400 text-center">
                                            No recent invoices.
                                        </td>
                                    </tr>
                                @endforelse
                            </tbody>
                        </table>
                    </div>
                    <div class="mt-4 text-right">
                                     <a  href="{{  route('invoices.index')  }}"  class="text-blue-600
hover:text-blue-900 dark:text-blue-400 dark:hover:text-blue-600 text-sm font-medium">
                            View All Invoices ?
                        </a>
                    </div>
                </div>
```

```
            </div>

            <!-- Top Clients -->
            <div class="bg-white dark:bg-gray-800 overflow-hidden shadow-sm sm:rounded-lg">
                <div class="p-6">
                       <h3 class="text-lg font-semibold text-gray-800 dark:text-gray-200 mb-4">Top
Clients</h3>
                    <div class="space-y-4">
                        @forelse ($topClients as $client)
                            <div class="flex items-center justify-between">
                                <div class="flex items-center">
                                            <div class="w-10 h-10 flex items-center justify-center
rounded-full bg-blue-500 bg-opacity-10 text-blue-500">
                                        {{ strtoupper(substr($client->name, 0, 1)) }}
                                    </div>
                                    <div class="ml-3">
                                                    <p class="text-sm font-medium text-gray-900
dark:text-gray-100">{{ $client->name }}</p>
                                            <p class="text-sm text-gray-500 dark:text-gray-400">${{
number_format($client->total_spent, 2) }}</p>
                                    </div>
                                </div>
                                                <a href="{{ route('clients.show', $client) }}"
class="text-blue-600      hover:text-blue-900      dark:text-blue-400      dark:hover:text-blue-600
text-sm">View</a>
                            </div>
                        @empty
                                <p class="text-sm text-gray-500 dark:text-gray-400 text-center">No
client data available.</p>
                        @endforelse
                    </div>
                    <div class="mt-4 text-right">
                                    <a href="{{ route('clients.index') }}" class="text-blue-600
hover:text-blue-900 dark:text-blue-400 dark:hover:text-blue-600 text-sm font-medium">
                            View All Clients ?
                        </a>
                    </div>
                </div>
            </div>
        </div>
    </div>
</div>
@endsection
```

resources/views/welcome.blade.php

```
<!DOCTYPE html>
<html>
<head>
    <title>Billing System</title>
</head>
<body>
    <h1>Welcome to the Billing System</h1>
    <div>
        <a href="/login">Login</a> or <a href="/register">Register</a>
    </div>
</body>
</html>
```

resources/views/auth/confirm-password.blade.php

```
<x-guest-layout>
    <div class="mb-4 text-sm text-gray-600 dark:text-gray-400">
          {{ __('This is a secure area of the application. Please confirm your password before
continuing.') }}
    </div>

    <form method="POST" action="{{ route('password.confirm') }}">
        @csrf

        <!-- Password -->
        <div>
            <x-input-label for="password" :value="__('Password')" />

            <x-text-input id="password" class="block mt-1 w-full"
                          type="password"
                          name="password"
                          required autocomplete="current-password" />

            <x-input-error :messages="$errors->get('password')" class="mt-2" />
        </div>

        <div class="flex justify-end mt-4">
            <x-primary-button>
                {{ __('Confirm') }}
            </x-primary-button>
        </div>
    </form>
</x-guest-layout>
```

resources/views/auth/forgot-password.blade.php

```
<x-guest-layout>
    <div class="mb-4 text-sm text-gray-600 dark:text-gray-400">
         {{ __('Forgot your password? No problem. Just let us know your email address and we will email you a password reset link that will allow you to choose a new one.') }}
    </div>

    <!-- Session Status -->
    <x-auth-session-status class="mb-4" :status="session('status')" />

    <form method="POST" action="{{ route('password.email') }}">
        @csrf

        <!-- Email Address -->
        <div>
            <x-input-label for="email" :value="__('Email')" />
                   <x-text-input id="email" class="block mt-1 w-full" type="email" name="email" :value="old('email')" required autofocus />
            <x-input-error :messages="$errors->get('email')" class="mt-2" />
        </div>

        <div class="flex items-center justify-end mt-4">
            <x-primary-button>
                {{ __('Email Password Reset Link') }}
            </x-primary-button>
        </div>
    </form>
</x-guest-layout>
```

resources/views/auth/login.blade.php

```
<x-guest-layout>
    <!-- Session Status -->
    <x-auth-session-status class="mb-4" :status="session('status')" />

    <form method="POST" action="{{ route('login') }}">
        @csrf

        <!-- Email Address -->
        <div>
            <x-input-label for="email" :value="__('Email')" />
                   <x-text-input id="email" class="block mt-1 w-full" type="email" name="email"
:value="old('email')" required autofocus autocomplete="username" />
            <x-input-error :messages="$errors->get('email')" class="mt-2" />
        </div>

        <!-- Password -->
        <div class="mt-4">
            <x-input-label for="password" :value="__('Password')" />

            <x-text-input id="password" class="block mt-1 w-full"
                            type="password"
                            name="password"
                            required autocomplete="current-password" />

            <x-input-error :messages="$errors->get('password')" class="mt-2" />
        </div>

        <!-- Remember Me -->
        <div class="block mt-4">
            <label for="remember_me" class="inline-flex items-center">
                          <input id="remember_me" type="checkbox" class="rounded dark:bg-gray-900
border-gray-300 dark:border-gray-700 text-indigo-600 shadow-sm focus:ring-indigo-500
dark:focus:ring-indigo-600 dark:focus:ring-offset-gray-800" name="remember">
                  <span class="ms-2 text-sm text-gray-600 dark:text-gray-400">{{ __('Remember me')
}}</span>
            </label>
        </div>

        <div class="flex items-center justify-end mt-4">
            @if (Route::has('password.request'))
                  <a class="underline text-sm text-gray-600 dark:text-gray-400 hover:text-gray-900
dark:hover:text-gray-100 rounded-md focus:outline-none focus:ring-2 focus:ring-offset-2
focus:ring-indigo-500 dark:focus:ring-offset-gray-800" href="{{ route('password.request') }}">
                    {{ __('Forgot your password?') }}
                </a>
            @endif

            <x-primary-button class="ms-3">
                {{ __('Log in') }}
            </x-primary-button>
        </div>
```

```
    </form>
</x-guest-layout>
```

resources/views/auth/register.blade.php

```
<x-guest-layout>
    <form method="POST" action="{{ route('register') }}">
        @csrf

        <!-- Name -->
        <div>
            <x-input-label for="name" :value="__('Name')" />
                     <x-text-input  id="name"  class="block  mt-1  w-full"  type="text"  name="name"
:value="old('name')" required autofocus autocomplete="name" />
            <x-input-error :messages="$errors->get('name')" class="mt-2" />
        </div>

        <!-- Email Address -->
        <div class="mt-4">
            <x-input-label for="email" :value="__('Email')" />
                   <x-text-input  id="email"  class="block  mt-1  w-full"  type="email"  name="email"
:value="old('email')" required autocomplete="username" />
            <x-input-error :messages="$errors->get('email')" class="mt-2" />
        </div>

        <!-- Password -->
        <div class="mt-4">
            <x-input-label for="password" :value="__('Password')" />

            <x-text-input id="password" class="block mt-1 w-full"
                          type="password"
                          name="password"
                          required autocomplete="new-password" />

            <x-input-error :messages="$errors->get('password')" class="mt-2" />
        </div>

        <!-- Confirm Password -->
        <div class="mt-4">
            <x-input-label for="password_confirmation" :value="__('Confirm Password')" />

            <x-text-input id="password_confirmation" class="block mt-1 w-full"
                          type="password"
                          name="password_confirmation" required autocomplete="new-password" />

            <x-input-error :messages="$errors->get('password_confirmation')" class="mt-2" />
        </div>

        <div class="flex items-center justify-end mt-4">
                 <a  class="underline  text-sm  text-gray-600  dark:text-gray-400  hover:text-gray-900
dark:hover:text-gray-100    rounded-md    focus:outline-none    focus:ring-2    focus:ring-offset-2
focus:ring-indigo-500 dark:focus:ring-offset-gray-800" href="{{ route('login') }}">
                {{ __('Already registered?') }}
            </a>

            <x-primary-button class="ms-4">
```

```
                {{ __('Register') }}
            </x-primary-button>
        </div>
    </form>
</x-guest-layout>
```

resources/views/auth/reset-password.blade.php

```
<x-guest-layout>
    <form method="POST" action="{{ route('password.store') }}">
        @csrf

        <!-- Password Reset Token -->
        <input type="hidden" name="token" value="{{ $request->route('token') }}">

        <!-- Email Address -->
        <div>
            <x-input-label for="email" :value="__('Email')" />
                   <x-text-input  id="email"  class="block  mt-1  w-full"  type="email"  name="email"
:value="old('email', $request->email)" required autofocus autocomplete="username" />
            <x-input-error :messages="$errors->get('email')" class="mt-2" />
        </div>

        <!-- Password -->
        <div class="mt-4">
            <x-input-label for="password" :value="__('Password')" />
             <x-text-input id="password" class="block mt-1 w-full" type="password" name="password"
required autocomplete="new-password" />
            <x-input-error :messages="$errors->get('password')" class="mt-2" />
        </div>

        <!-- Confirm Password -->
        <div class="mt-4">
            <x-input-label for="password_confirmation" :value="__('Confirm Password')" />

            <x-text-input id="password_confirmation" class="block mt-1 w-full"
                                type="password"
                                  name="password_confirmation" required autocomplete="new-password"
/>

            <x-input-error :messages="$errors->get('password_confirmation')" class="mt-2" />
        </div>

        <div class="flex items-center justify-end mt-4">
            <x-primary-button>
                {{ __('Reset Password') }}
            </x-primary-button>
        </div>
    </form>
</x-guest-layout>
```

resources/views/auth/verify-email.blade.php

```
<x-guest-layout>
    <div class="mb-4 text-sm text-gray-600 dark:text-gray-400">
        {{ __('Thanks for signing up! Before getting started, could you verify your email address by clicking on the link we just emailed to you? If you didn\'t receive the email, we will gladly send you another.') }}
    </div>

    @if (session('status') == 'verification-link-sent')
        <div class="mb-4 font-medium text-sm text-green-600 dark:text-green-400">
            {{ __('A new verification link has been sent to the email address you provided during registration.') }}
        </div>
    @endif

    <div class="mt-4 flex items-center justify-between">
        <form method="POST" action="{{ route('verification.send') }}">
            @csrf

            <div>
                <x-primary-button>
                    {{ __('Resend Verification Email') }}
                </x-primary-button>
            </div>
        </form>

        <form method="POST" action="{{ route('logout') }}">
            @csrf

                <button type="submit" class="underline text-sm text-gray-600 dark:text-gray-400 hover:text-gray-900 dark:hover:text-gray-100 rounded-md focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-indigo-500 dark:focus:ring-offset-gray-800">
                {{ __('Log Out') }}
            </button>
        </form>
    </div>
</x-guest-layout>
```

resources/views/clients/create.blade.php

```
@extends('layouts.app')

@section('content')
<div class="py-12">
    <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
        <div class="bg-white dark:bg-gray-800 overflow-hidden shadow-sm sm:rounded-lg">
                        <div  class="p-6  bg-white  dark:bg-gray-800  border-b  border-gray-200
dark:border-gray-700">
                <div class="mb-6">
                     <h2 class="text-xl font-semibold text-gray-800 dark:text-gray-200">Create New
Client</h2>
                </div>

                <form action="{{ route('clients.store') }}" method="POST">
                    @csrf

                    <div class="grid grid-cols-1 md:grid-cols-2 gap-6">
                        <!-- Name -->
                        <div>
                                 <label for="name" class="block text-sm font-medium text-gray-700
dark:text-gray-300">Name</label>
                                <input type="text" name="name" id="name" value="{{ old('name') }}"
required
                                    class="mt-1 block w-full border-gray-300 dark:border-gray-700
dark:bg-gray-900    dark:text-gray-300    focus:border-indigo-500    dark:focus:border-indigo-600
focus:ring-indigo-500 dark:focus:ring-indigo-600 rounded-md shadow-sm">
                            @error('name')
                                <p class="text-red-600 text-sm mt-1">{{ $message }}</p>
                            @enderror
                        </div>

                        <!-- Email -->
                        <div>
                                 <label for="email" class="block text-sm font-medium text-gray-700
dark:text-gray-300">Email</label>
                             <input type="email" name="email" id="email" value="{{ old('email') }}"
required
                                    class="mt-1 block w-full border-gray-300 dark:border-gray-700
dark:bg-gray-900    dark:text-gray-300    focus:border-indigo-500    dark:focus:border-indigo-600
focus:ring-indigo-500 dark:focus:ring-indigo-600 rounded-md shadow-sm">
                            @error('email')
                                <p class="text-red-600 text-sm mt-1">{{ $message }}</p>
                            @enderror
                        </div>

                        <!-- Phone -->
                        <div>
                                 <label for="phone" class="block text-sm font-medium text-gray-700
dark:text-gray-300">Phone</label>
                            <input type="text" name="phone" id="phone" value="{{ old('phone') }}"
                                    class="mt-1 block w-full border-gray-300 dark:border-gray-700
```

```
dark:bg-gray-900    dark:text-gray-300    focus:border-indigo-500    dark:focus:border-indigo-600
focus:ring-indigo-500 dark:focus:ring-indigo-600 rounded-md shadow-sm">
                            @error('phone')
                                <p class="text-red-600 text-sm mt-1">{{ $message }}</p>
                            @enderror
                        </div>
                    </div>

                    <!-- Address -->
                    <div class="mt-6">
                              <label for="address" class="block text-sm font-medium text-gray-700
dark:text-gray-300">Address</label>
                        <textarea name="address" id="address" rows="3"
                                    class="mt-1 block w-full border-gray-300 dark:border-gray-700
dark:bg-gray-900    dark:text-gray-300    focus:border-indigo-500    dark:focus:border-indigo-600
focus:ring-indigo-500    dark:focus:ring-indigo-600    rounded-md    shadow-sm">{{    old('address')
}}</textarea>
                        @error('address')
                            <p class="text-red-600 text-sm mt-1">{{ $message }}</p>
                        @enderror
                    </div>

                    <div class="mt-6 flex items-center justify-end">
                                     <a  href="{{  route('clients.index')  }}"  class="text-gray-500
hover:text-gray-700 dark:text-gray-300 dark:hover:text-gray-100 mr-4">
                            Cancel
                        </a>
                              <button type="submit" class="px-4 py-2 bg-blue-500 hover:bg-blue-700
text-white font-bold rounded">
                            Create Client
                        </button>
                    </div>
                </form>
            </div>
        </div>
    </div>
</div>
@endsection
```

resources/views/clients/index.blade.php

```
@extends('layouts.app')

@section('content')
<div class="py-12">
    <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
        <div class="bg-white dark:bg-gray-800 overflow-hidden shadow-sm sm:rounded-lg">
                        <div  class="p-6  bg-white  dark:bg-gray-800  border-b  border-gray-200
dark:border-gray-700">
                <div class="flex justify-between items-center mb-6">
                                                <h2  class="text-xl  font-semibold  text-gray-800
dark:text-gray-200">Clients</h2>
                            <a  href="{{  route('clients.create')  }}"  class="px-4  py-2  bg-blue-500
hover:bg-blue-700 text-white font-bold rounded">
                        Add New Client
                    </a>
                </div>

                <!-- Search Form -->
                <div class="mb-6">
                    <form action="{{ route('clients.index') }}" method="GET" class="flex">
                        <input type="text" name="search" placeholder="Search clients..." value="{{
request('search') }}"
                                class="flex-1 border-gray-300 dark:border-gray-700 dark:bg-gray-900
dark:text-gray-300  focus:border-indigo-500  dark:focus:border-indigo-600  focus:ring-indigo-500
dark:focus:ring-indigo-600 rounded-l-md shadow-sm">
                        <button type="submit" class="px-4 py-2 bg-gray-800 dark:bg-gray-200 border
border-transparent  rounded-r-md  font-semibold  text-xs  text-white  dark:text-gray-800  uppercase
tracking-widest  hover:bg-gray-700  dark:hover:bg-white  focus:bg-gray-700  dark:focus:bg-white
active:bg-gray-900  dark:active:bg-gray-300  focus:outline-none  focus:ring-2  focus:ring-indigo-500
focus:ring-offset-2 dark:focus:ring-offset-gray-800">
                            Search
                        </button>
                    </form>
                </div>

                <!-- Clients Table -->
                <div class="overflow-x-auto">
                    <table class="min-w-full divide-y divide-gray-200 dark:divide-gray-700">
                        <thead class="bg-gray-50 dark:bg-gray-700">
                            <tr>
                                    <th scope="col" class="px-6 py-3 text-left text-xs font-medium
text-gray-500 dark:text-gray-300 uppercase tracking-wider">Name</th>
                                    <th scope="col" class="px-6 py-3 text-left text-xs font-medium
text-gray-500 dark:text-gray-300 uppercase tracking-wider">Email</th>
                                    <th scope="col" class="px-6 py-3 text-left text-xs font-medium
text-gray-500 dark:text-gray-300 uppercase tracking-wider">Phone</th>
                                    <th scope="col" class="px-6 py-3 text-left text-xs font-medium
text-gray-500 dark:text-gray-300 uppercase tracking-wider">Invoices</th>
                                    <th scope="col" class="px-6 py-3 text-left text-xs font-medium
text-gray-500 dark:text-gray-300 uppercase tracking-wider">Actions</th>
                            </tr>
```

```
                    </thead>
                            <tbody class="bg-white divide-y divide-gray-200 dark:bg-gray-800
dark:divide-gray-700">
                        @forelse ($clients as $client)
                            <tr>
                                <td class="px-6 py-4 whitespace-nowrap">
                                                <div class="text-sm font-medium text-gray-900
dark:text-gray-100">
                                        {{ $client->name }}
                                    </div>
                                </td>
                                <td class="px-6 py-4 whitespace-nowrap">
                                    <div class="text-sm text-gray-500 dark:text-gray-400">
                                        {{ $client->email }}
                                    </div>
                                </td>
                                <td class="px-6 py-4 whitespace-nowrap">
                                    <div class="text-sm text-gray-500 dark:text-gray-400">
                                        {{ $client->phone ?? 'N/A' }}
                                    </div>
                                </td>
                                <td class="px-6 py-4 whitespace-nowrap">
                                    <div class="text-sm text-gray-500 dark:text-gray-400">
                                                                        {{ $client->invoices_count ??
$client->invoices()->count() }}
                                    </div>
                                </td>
                                <td class="px-6 py-4 whitespace-nowrap text-sm font-medium">
                                            <a href="{{ route('clients.show', $client) }}"
class="text-blue-600      hover:text-blue-900      dark:text-blue-400      dark:hover:text-blue-600
mr-3">View</a>
                                            <a href="{{ route('clients.edit', $client) }}"
class="text-indigo-600   hover:text-indigo-900   dark:text-indigo-400   dark:hover:text-indigo-600
mr-3">Edit</a>
                                        <form action="{{ route('clients.destroy', $client) }}"
method="POST" class="inline">
                                        @csrf
                                        @method('DELETE')
                                                <button type="submit" class="text-red-600
hover:text-red-900  dark:text-red-400  dark:hover:text-red-600"  onclick="return  confirm('Are  you
sure you want to delete this client?')">Delete</button>
                                        </form>
                                </td>
                            </tr>
                        @empty
                            <tr>
                                    <td colspan="5" class="px-6 py-4 whitespace-nowrap text-sm
text-gray-500 dark:text-gray-400 text-center">
                                        No clients found.
                                </td>
                            </tr>
                        @endforelse
                    </tbody>
                </table>
```

```
                </div>

                <!-- Pagination -->
                <div class="mt-4">
                    {{ $clients->links() }}
                </div>
            </div>
        </div>
    </div>
</div>
@endsection
```

resources/views/components/application-logo.blade.php

```
<svg viewBox="0 0 316 316" xmlns="http://www.w3.org/2000/svg" {{ $attributes }}>
    <path d="M305.8 81.125C305.77 80.995 305.69 80.885 305.65 80.755C305.56 80.525 305.49 80.285
305.37 80.075C305.29 79.935 305.17 79.815 305.07 79.685C304.94 79.515 304.83 79.325 304.68
79.175C304.55 79.045 304.39 78.955 304.25 78.845C304.09 78.715 303.95 78.575 303.77 78.475L251.32
48.275C249.97 47.495 248.31 47.495 246.96 48.275L194.51 78.475C194.33 78.575 194.19 78.725 194.03
78.845C193.89 78.955 193.73 79.045 193.6 79.175C193.45 79.325 193.34 79.515 193.21 79.685C193.11
79.815 192.99 79.935 192.91 80.075C192.79 80.285 192.71 80.525 192.63 80.755C192.58 80.875 192.51
80.995 192.48 81.125C192.38 81.495 192.33 81.875 192.33 82.265V139.625L148.62
164.795V52.575C148.62 52.185 148.57 51.805 148.47 51.435C148.44 51.305 148.36 51.195 148.32
51.065C148.23 50.835 148.16 50.595 148.04 50.385C147.96 50.245 147.84 50.125 147.74 49.995C147.61
49.825 147.5 49.635 147.35 49.485C147.22 49.355 147.06 49.265 146.92 49.155C146.76 49.025 146.62
48.885 146.44 48.785L93.99 18.585C92.64 17.805 90.98 17.805 89.63 18.585L37.18 48.785C37 48.885
36.86 49.035 36.7 49.155C36.56 49.265 36.4 49.355 36.27 49.485C36.12 49.635 36.01 49.825 35.88
49.995C35.78 50.125 35.66 50.245 35.58 50.385C35.46 50.595 35.38 50.835 35.3 51.065C35.25 51.185
35.18 51.305 35.15 51.435C35.05 51.805 35 52.185 35 52.575V232.235C35 233.795 35.84 235.245 37.19
236.025L142.1 296.425C142.33 296.555 142.58 296.635 142.82 296.725C142.93 296.765 143.04 296.835
143.16 296.865C143.53 296.965 143.9 297.015 144.28 297.015C144.66 297.015 145.03 296.965 145.4
296.865C145.5 296.835 145.59 296.775 145.69 296.745C145.95 296.655 146.21 296.565 146.45
296.435L251.36 236.035C252.72 235.255 253.55 233.815 253.55 232.245V174.885L303.81 145.945C305.17
145.165 306 143.725 306 142.155V82.265C305.95 81.875 305.89 81.495 305.8 81.125ZM144.2
227.205L100.57 202.515L146.39 176.135L196.66 147.195L240.33 172.335L208.29 190.625L144.2
227.205ZM244.75 114.995V164.795L226.39 154.225L201.03 139.625V89.825L219.39 100.395L244.75
114.995ZM249.12 57.105L292.81 82.265L249.12 107.425L205.43 82.265L249.12 57.105ZM114.49
184.425L96.13 194.995V85.305L121.49 70.705L139.85 60.135V169.815L114.49 184.425ZM91.76
27.425L135.45 52.585L91.76 77.745L48.07 52.585L91.76 27.425ZM43.67 60.135L62.03 70.705L87.39
85.305V202.545V202.555V202.565C87.39 202.735 87.44 202.895 87.46 203.055C87.49 203.265 87.49
203.485 87.55 203.695V203.705C87.6 203.875 87.69 204.035 87.76 204.195C87.84 204.375 87.89 204.575
87.99 204.745C87.99 204.745 87.99 204.755 88 204.755C88.09 204.905 88.22 205.035 88.33
205.175C88.45 205.335 88.55 205.495 88.69 205.635L88.7 205.645C88.82 205.765 88.98 205.855 89.12
205.965C89.28 206.085 89.42 206.225 89.59 206.325C89.6 206.325 89.6 206.325 89.61 206.335C89.62
206.335 89.62 206.345 89.63 206.345L139.87 234.775V285.065L43.67 229.705V60.135ZM244.75
229.705L148.58 285.075V234.775L219.8 194.115L244.75 179.875V229.705ZM297.2 139.625L253.49
164.795V114.995L278.85 100.395L297.21 89.825V139.625H297.2Z"/>
</svg>
```

resources/views/components/auth-session-status.blade.php

```
@props(['status'])

@if ($status)
            <div   {{   $attributes->merge(['class'   =>   'font-medium   text-sm   text-green-600
dark:text-green-400']) }}>
        {{ $status }}
    </div>
@endif
```

resources/views/components/danger-button.blade.php

```
<button {{ $attributes->merge(['type' => 'submit', 'class' => 'inline-flex items-center px-4 py-2 bg-red-600 border border-transparent rounded-md font-semibold text-xs text-white uppercase tracking-widest hover:bg-red-500 active:bg-red-700 focus:outline-none focus:ring-2 focus:ring-red-500 focus:ring-offset-2 dark:focus:ring-offset-gray-800 transition ease-in-out duration-150']) }}>
    {{ $slot }}
</button>
```

resources/views/components/dropdown-link.blade.php

```
<a {{ $attributes->merge(['class' => 'block w-full px-4 py-2 text-start text-sm leading-5 text-gray-700 dark:text-gray-300 hover:bg-gray-100 dark:hover:bg-gray-800 focus:outline-none focus:bg-gray-100 dark:focus:bg-gray-800 transition duration-150 ease-in-out']) }}>{{ $slot }}</a>
```

resources/views/components/dropdown.blade.php

```
@props(['align' => 'right', 'width' => '48', 'contentClasses' => 'py-1 bg-white dark:bg-gray-700'])

@php
switch ($align) {
    case 'left':
        $alignmentClasses = 'ltr:origin-top-left rtl:origin-top-right start-0';
        break;
    case 'top':
        $alignmentClasses = 'origin-top';
        break;
    case 'right':
    default:
        $alignmentClasses = 'ltr:origin-top-right rtl:origin-top-left end-0';
        break;
}

switch ($width) {
    case '48':
        $width = 'w-48';
        break;
}
@endphp

<div class="relative" x-data="{ open: false }" @click.outside="open = false" @close.stop="open = false">
    <div @click="open = ! open">
        {{ $trigger }}
    </div>

    <div x-show="open"
            x-transition:enter="transition ease-out duration-200"
            x-transition:enter-start="opacity-0 scale-95"
            x-transition:enter-end="opacity-100 scale-100"
            x-transition:leave="transition ease-in duration-75"
            x-transition:leave-start="opacity-100 scale-100"
            x-transition:leave-end="opacity-0 scale-95"
            class="absolute z-50 mt-2 {{ $width }} rounded-md shadow-lg {{ $alignmentClasses }}"
            style="display: none;"
            @click="open = false">
        <div class="rounded-md ring-1 ring-black ring-opacity-5 {{ $contentClasses }}">
            {{ $content }}
        </div>
    </div>
</div>
```

resources/views/components/flash-message.blade.php

```
@if (session('success'))
      <div x-data="{ show: true }" x-show="show" x-init="setTimeout(() => show = false, 5000)" class="mb-4 bg-green-100 border-l-4 border-green-500 text-green-700 p-4 dark:bg-green-800 dark:text-green-100 dark:border-green-600" role="alert">
        <div class="flex">
            <div class="flex-shrink-0">
                                    <svg class="h-5 w-5 text-green-500 dark:text-green-400" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 20 20" fill="currentColor" aria-hidden="true">
                    <path fill-rule="evenodd" d="M10 18a8 8 0 100-16 8 8 0 000 16zm3.707-9.293a1 1 0 00-1.414-1.414L9 10.586 7.707 9.293a1 1 0 00-1.414 1.414l2 2a1 1 0 001.414 0l4-4z" clip-rule="evenodd" />
                </svg>
            </div>
            <div class="ml-3">
                <p class="text-sm">{{ session('success') }}</p>
            </div>
            <div class="ml-auto pl-3">
                <div class="-mx-1.5 -my-1.5">
                        <button @click="show = false" type="button" class="inline-flex rounded-md p-1.5 text-green-500 hover:bg-green-200 dark:text-green-400 dark:hover:bg-green-700 focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-green-500 dark:focus:ring-offset-green-800">
                        <span class="sr-only">Dismiss</span>
                           <svg class="h-5 w-5" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 20 20" fill="currentColor" aria-hidden="true">
                                    <path fill-rule="evenodd" d="M4.293 4.293a1 1 0 011.414 0L10 8.586l4.293-4.293a1 1 0 111.414 1.414L11.414 10l4.293 4.293a1 1 0 01-1.414 1.414L10 11.414l-4.293 4.293a1 1 0 01-1.414-1.414L8.586 10 4.293 5.707a1 1 0 010-1.414z" clip-rule="evenodd" />
                        </svg>
                    </button>
                </div>
            </div>
        </div>
    </div>
@endif

@if (session('error'))
      <div x-data="{ show: true }" x-show="show" x-init="setTimeout(() => show = false, 5000)" class="mb-4 bg-red-100 border-l-4 border-red-500 text-red-700 p-4 dark:bg-red-800 dark:text-red-100 dark:border-red-600" role="alert">
        <div class="flex">
            <div class="flex-shrink-0">
                                        <svg class="h-5 w-5 text-red-500 dark:text-red-400" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 20 20" fill="currentColor" aria-hidden="true">
                    <path fill-rule="evenodd" d="M10 18a8 8 0 100-16 8 8 0 000 16zM8.707 7.293a1 1 0 00-1.414 1.414L8.586 10l-1.293 1.293a1 1 0 101.414 1.414L10 11.414l1.293 1.293a1 1 0 001.414-1.414L11.414 10l1.293-1.293a1 1 0 00-1.414-1.414L10 8.586 8.707 7.293z" clip-rule="evenodd" />
                </svg>
            </div>
            <div class="ml-3">
```

```
                <p class="text-sm">{{ session('error') }}</p>
            </div>
            <div class="ml-auto pl-3">
                <div class="-mx-1.5 -my-1.5">
                        <button @click="show = false" type="button" class="inline-flex rounded-md p-1.5 text-red-500 hover:bg-red-200 dark:text-red-400 dark:hover:bg-red-700 focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-red-500 dark:focus:ring-offset-red-800">
                        <span class="sr-only">Dismiss</span>
                           <svg class="h-5 w-5" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 20 20" fill="currentColor" aria-hidden="true">
                                    <path fill-rule="evenodd" d="M4.293 4.293a1 1 0 011.414 0L10 8.586l4.293-4.293a1 1 0 111.414 1.414L11.414 10l4.293 4.293a1 1 0 01-1.414 1.414L10 11.414l-4.293 4.293a1 1 0 01-1.414-1.414L8.586 10 4.293 5.707a1 1 0 010-1.414z" clip-rule="evenodd" />
                        </svg>
                    </button>
                </div>
            </div>
        </div>
    </div>
@endif
```

resources/views/components/input-error.blade.php

```
@props(['messages'])

@if ($messages)
    <ul {{ $attributes->merge(['class' => 'text-sm text-red-600 dark:text-red-400 space-y-1']) }}>
        @foreach ((array) $messages as $message)
            <li>{{ $message }}</li>
        @endforeach
    </ul>
@endif
```

resources/views/components/input-label.blade.php

```
@props(['value'])

<label {{ $attributes->merge(['class' => 'block font-medium text-sm text-gray-700 dark:text-gray-300']) }}>
    {{ $value ?? $slot }}
</label>
```

resources/views/components/modal.blade.php

```
@props([
    'name',
    'show' => false,
    'maxWidth' => '2xl'
])

@php
$maxWidth = [
    'sm' => 'sm:max-w-sm',
    'md' => 'sm:max-w-md',
    'lg' => 'sm:max-w-lg',
    'xl' => 'sm:max-w-xl',
    '2xl' => 'sm:max-w-2xl',
][$maxWidth];
@endphp

<div
    x-data="{
        show: @js($show),
        focusables() {
            // All focusable element types...
              let selector = 'a, button, input:not([type=\'hidden\']), textarea, select, details,
[tabindex]:not([tabindex=\'-1\'])'
            return [...$el.querySelectorAll(selector)]
                // All non-disabled elements...
                .filter(el => ! el.hasAttribute('disabled'))
        },
        firstFocusable() { return this.focusables()[0] },
        lastFocusable() { return this.focusables().slice(-1)[0] },
                    nextFocusable()   {   return   this.focusables()[this.nextFocusableIndex()]   ||
this.firstFocusable() },
                    prevFocusable()   {   return   this.focusables()[this.prevFocusableIndex()]   ||
this.lastFocusable() },
          nextFocusableIndex() { return (this.focusables().indexOf(document.activeElement) + 1) %
(this.focusables().length + 1) },
                                              prevFocusableIndex()      {      return      Math.max(0,
this.focusables().indexOf(document.activeElement)) -1 },
    }"
    x-init="$watch('show', value => {
        if (value) {
            document.body.classList.add('overflow-y-hidden');
            {{ $attributes->has('focusable') ? 'setTimeout(() => firstFocusable().focus(), 100)' :
'' }}
        } else {
            document.body.classList.remove('overflow-y-hidden');
        }
    })"
    x-on:open-modal.window="$event.detail == '{{ $name }}' ? show = true : null"
    x-on:close-modal.window="$event.detail == '{{ $name }}' ? show = false : null"
    x-on:close.stop="show = false"
    x-on:keydown.escape.window="show = false"
```

```
    x-on:keydown.tab.prevent="$event.shiftKey || nextFocusable().focus()"
    x-on:keydown.shift.tab.prevent="prevFocusable().focus()"
    x-show="show"
    class="fixed inset-0 overflow-y-auto px-4 py-6 sm:px-0 z-50"
    style="display: {{ $show ? 'block' : 'none' }};"
>
    <div
        x-show="show"
        class="fixed inset-0 transform transition-all"
        x-on:click="show = false"
        x-transition:enter="ease-out duration-300"
        x-transition:enter-start="opacity-0"
        x-transition:enter-end="opacity-100"
        x-transition:leave="ease-in duration-200"
        x-transition:leave-start="opacity-100"
        x-transition:leave-end="opacity-0"
    >
        <div class="absolute inset-0 bg-gray-500 dark:bg-gray-900 opacity-75"></div>
    </div>

    <div
        x-show="show"
            class="mb-6 bg-white dark:bg-gray-800 rounded-lg overflow-hidden shadow-xl transform
transition-all sm:w-full {{ $maxWidth }} sm:mx-auto"
        x-transition:enter="ease-out duration-300"
        x-transition:enter-start="opacity-0 translate-y-4 sm:translate-y-0 sm:scale-95"
        x-transition:enter-end="opacity-100 translate-y-0 sm:scale-100"
        x-transition:leave="ease-in duration-200"
        x-transition:leave-start="opacity-100 translate-y-0 sm:scale-100"
        x-transition:leave-end="opacity-0 translate-y-4 sm:translate-y-0 sm:scale-95"
    >
        {{ $slot }}
    </div>
</div>
```

resources/views/components/nav-link.blade.php

```
@props(['active'])

@php
$classes = ($active ?? false)
                        ? 'inline-flex items-center px-1 pt-1 border-b-2 border-indigo-400 dark:border-indigo-600 text-sm font-medium leading-5 text-gray-900 dark:text-gray-100 focus:outline-none focus:border-indigo-700 transition duration-150 ease-in-out'
                        : 'inline-flex items-center px-1 pt-1 border-b-2 border-transparent text-sm font-medium leading-5 text-gray-500 dark:text-gray-400 hover:text-gray-700 dark:hover:text-gray-300 hover:border-gray-300 dark:hover:border-gray-700 focus:outline-none focus:text-gray-700 dark:focus:text-gray-300 focus:border-gray-300 dark:focus:border-gray-700 transition duration-150 ease-in-out';
@endphp

<a {{ $attributes->merge(['class' => $classes]) }}>
    {{ $slot }}
</a>
```

resources/views/components/primary-button.blade.php

```
<button {{ $attributes->merge(['type' => 'submit', 'class' => 'inline-flex items-center px-4 py-2 bg-gray-800 dark:bg-gray-200 border border-transparent rounded-md font-semibold text-xs text-white dark:text-gray-800 uppercase tracking-widest hover:bg-gray-700 dark:hover:bg-white focus:bg-gray-700 dark:focus:bg-white active:bg-gray-900 dark:active:bg-gray-300 focus:outline-none focus:ring-2 focus:ring-indigo-500 focus:ring-offset-2 dark:focus:ring-offset-gray-800 transition ease-in-out duration-150']) }}>
    {{ $slot }}
</button>
```

resources/views/components/responsive-nav-link.blade.php

```
@props(['active'])

@php
$classes = ($active ?? false)
            ? 'block w-full ps-3 pe-4 py-2 border-l-4 border-indigo-400 dark:border-indigo-600 text-start text-base font-medium text-indigo-700 dark:text-indigo-300 bg-indigo-50 dark:bg-indigo-900/50 focus:outline-none focus:text-indigo-800 dark:focus:text-indigo-200 focus:bg-indigo-100 dark:focus:bg-indigo-900 focus:border-indigo-700 dark:focus:border-indigo-300 transition duration-150 ease-in-out'
            : 'block w-full ps-3 pe-4 py-2 border-l-4 border-transparent text-start text-base font-medium text-gray-600 dark:text-gray-400 hover:text-gray-800 dark:hover:text-gray-200 hover:bg-gray-50 dark:hover:bg-gray-700 hover:border-gray-300 dark:hover:border-gray-600 focus:outline-none focus:text-gray-800 dark:focus:text-gray-200 focus:bg-gray-50 dark:focus:bg-gray-700 focus:border-gray-300 dark:focus:border-gray-600 transition duration-150 ease-in-out';
@endphp

<a {{ $attributes->merge(['class' => $classes]) }}>
    {{ $slot }}
</a>
```

resources/views/components/secondary-button.blade.php

```
<button {{ $attributes->merge(['type' => 'button', 'class' => 'inline-flex items-center px-4 py-2 bg-white dark:bg-gray-800 border border-gray-300 dark:border-gray-500 rounded-md font-semibold text-xs text-gray-700 dark:text-gray-300 uppercase tracking-widest shadow-sm hover:bg-gray-50 dark:hover:bg-gray-700 focus:outline-none focus:ring-2 focus:ring-indigo-500 focus:ring-offset-2 dark:focus:ring-offset-gray-800 disabled:opacity-25 transition ease-in-out duration-150']) }}>
    {{ $slot }}
</button>
```

resources/views/components/text-input.blade.php

```
@props(['disabled' => false])

<input {{ $disabled ? 'disabled' : '' }} {!! $attributes->merge(['class' => 'border-gray-300 dark:border-gray-700 dark:bg-gray-900 dark:text-gray-300 focus:border-indigo-500 dark:focus:border-indigo-600 focus:ring-indigo-500 dark:focus:ring-indigo-600 rounded-md shadow-sm']) !!}>
```

resources/views/invoices/create.blade.php

```
@extends('layouts.app')

@section('content')
<div class="py-12">
    <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
        <div class="bg-white dark:bg-gray-800 overflow-hidden shadow-sm sm:rounded-lg">
                        <div  class="p-6  bg-white  dark:bg-gray-800  border-b  border-gray-200
dark:border-gray-700">
                <div class="mb-6">
                     <h2 class="text-xl font-semibold text-gray-800 dark:text-gray-200">Create New
Invoice</h2>
                </div>

                <form action="{{ route('invoices.store') }}" method="POST" id="invoice-form">
                    @csrf

                    <div class="grid grid-cols-1 md:grid-cols-2 gap-6">
                        <!-- Client Selection -->
                        <div>
                             <label for="client_id" class="block text-sm font-medium text-gray-700
dark:text-gray-300">Client</label>
                            <select name="client_id" id="client_id" required
                                    class="mt-1 block w-full border-gray-300 dark:border-gray-700
dark:bg-gray-900     dark:text-gray-300     focus:border-indigo-500     dark:focus:border-indigo-600
focus:ring-indigo-500 dark:focus:ring-indigo-600 rounded-md shadow-sm">
                                <option value="">Select Client</option>
                                @foreach($clients as $client)
                                         <option value="{{ $client->id }}" {{ old('client_id') ==
$client->id ? 'selected' : '' }}>
                                            {{ $client->name }}
                                        </option>
                                @endforeach
                            </select>
                            @error('client_id')
                                <p class="text-red-600 text-sm mt-1">{{ $message }}</p>
                            @enderror
                        </div>

                        <!-- Invoice Number -->
                        <div>
                                     <label for="invoice_number" class="block text-sm font-medium
text-gray-700 dark:text-gray-300">Invoice Number</label>
                             <input type="text" name="invoice_number" id="invoice_number" value="{{
old('invoice_number') }}" required
                                    class="mt-1 block w-full border-gray-300 dark:border-gray-700
dark:bg-gray-900     dark:text-gray-300     focus:border-indigo-500     dark:focus:border-indigo-600
focus:ring-indigo-500 dark:focus:ring-indigo-600 rounded-md shadow-sm">
                            @error('invoice_number')
                                <p class="text-red-600 text-sm mt-1">{{ $message }}</p>
                            @enderror
                        </div>
```

```
                        <!-- Invoice Date -->
                        <div>
                                    <label for="invoice_date" class="block text-sm font-medium
text-gray-700 dark:text-gray-300">Invoice Date</label>
                                <input type="date" name="invoice_date" id="invoice_date" value="{{
old('invoice_date', date('Y-m-d')) }}" required
                                    class="mt-1 block w-full border-gray-300 dark:border-gray-700
dark:bg-gray-900    dark:text-gray-300    focus:border-indigo-500    dark:focus:border-indigo-600
focus:ring-indigo-500 dark:focus:ring-indigo-600 rounded-md shadow-sm">
                            @error('invoice_date')
                                <p class="text-red-600 text-sm mt-1">{{ $message }}</p>
                            @enderror
                        </div>

                        <!-- Due Date -->
                        <div>
                              <label for="due_date" class="block text-sm font-medium text-gray-700
dark:text-gray-300">Due Date</label>
                                    <input type="date" name="due_date" id="due_date" value="{{
old('due_date', date('Y-m-d', strtotime('+30 days'))) }}" required
                                    class="mt-1 block w-full border-gray-300 dark:border-gray-700
dark:bg-gray-900    dark:text-gray-300    focus:border-indigo-500    dark:focus:border-indigo-600
focus:ring-indigo-500 dark:focus:ring-indigo-600 rounded-md shadow-sm">
                            @error('due_date')
                                <p class="text-red-600 text-sm mt-1">{{ $message }}</p>
                            @enderror
                        </div>
                    </div>

                    <!-- Invoice Items -->
                    <div class="mt-6">
                                <h3 class="text-lg font-medium text-gray-700 dark:text-gray-300
mb-3">Invoice Items</h3>

                        <div id="invoice-items" class="space-y-4">
                            <div class="invoice-item bg-gray-50 dark:bg-gray-700 p-4 rounded-md">
                                <div class="grid grid-cols-1 md:grid-cols-4 gap-4">
                                    <!-- Product -->
                                    <div>
                                            <label for="items[0][product_id]" class="block text-sm
font-medium text-gray-700 dark:text-gray-300">Product</label>
                                        <select name="items[0][product_id]" class="product-select
mt-1 block w-full border-gray-300 dark:border-gray-700 dark:bg-gray-900 dark:text-gray-300
focus:border-indigo-500 dark:focus:border-indigo-600 focus:ring-indigo-500
dark:focus:ring-indigo-600 rounded-md shadow-sm" required>
                                            <option value="">Select Product</option>
                                            @foreach($products as $product)
                                                <option value="{{ $product->id }}" data-price="{{
$product->price }}" data-has-serial="{{ $product->has_serial ? 'true' : 'false' }}">
                                                    {{ $product->name }} - ${{
number_format($product->price, 2) }}
                                                </option>
                                            @endforeach
```

```
                                    </select>
                                </div>

                                <!-- Quantity -->
                                <div>
                                    <label for="items[0][quantity]" class="block text-sm font-medium text-gray-700 dark:text-gray-300">Quantity</label>
                                    <input type="number" name="items[0][quantity]" class="quantity-input mt-1 block w-full border-gray-300 dark:border-gray-700 dark:bg-gray-900 dark:text-gray-300 focus:border-indigo-500 dark:focus:border-indigo-600 focus:ring-indigo-500 dark:focus:ring-indigo-600 rounded-md shadow-sm" min="1" value="1" required>
                                </div>

                                <!-- Price -->
                                <div>
                                    <label for="items[0][price]" class="block text-sm font-medium text-gray-700 dark:text-gray-300">Price</label>
                                    <input type="number" name="items[0][price]" class="price-input mt-1 block w-full border-gray-300 dark:border-gray-700 dark:bg-gray-900 dark:text-gray-300 focus:border-indigo-500 dark:focus:border-indigo-600 focus:ring-indigo-500 dark:focus:ring-indigo-600 rounded-md shadow-sm" step="0.01" min="0" required>
                                </div>

                                <!-- Serial Numbers -->
                                <div class="serial-numbers-container hidden">
                                    <label for="items[0][serial_numbers]" class="block text-sm font-medium text-gray-700 dark:text-gray-300">Serial Numbers (comma separated)</label>
                                    <input type="text" name="items[0][serial_numbers]" class="serial-numbers mt-1 block w-full border-gray-300 dark:border-gray-700 dark:bg-gray-900 dark:text-gray-300 focus:border-indigo-500 dark:focus:border-indigo-600 focus:ring-indigo-500 dark:focus:ring-indigo-600 rounded-md shadow-sm" placeholder="SN001, SN002, ...">
                                </div>
                            </div>

                            <!-- Remove Button (hidden for first item) -->
                            <div class="mt-2 text-right">
                                <button type="button" class="remove-item text-red-600 hover:text-red-900 dark:text-red-400 dark:hover:text-red-600 text-sm hidden">
                                    Remove Item
                                </button>
                            </div>
                        </div>
                    </div>

                    <!-- Add Item Button -->
                    <div class="mt-4">
                        <button type="button" id="add-item" class="px-4 py-2 bg-green-500 hover:bg-green-700 text-white font-bold rounded">
                            Add Item
                        </button>
                    </div>
                </div>

                <!-- Notes -->
```

```
                    <div class="mt-6">
                                <label for="notes" class="block text-sm font-medium text-gray-700
dark:text-gray-300">Notes</label>
                        <textarea name="notes" id="notes" rows="3"
                                    class="mt-1 block w-full border-gray-300 dark:border-gray-700
dark:bg-gray-900     dark:text-gray-300     focus:border-indigo-500     dark:focus:border-indigo-600
focus:ring-indigo-500     dark:focus:ring-indigo-600     rounded-md     shadow-sm">{{     old('notes')
}}</textarea>
                        @error('notes')
                            <p class="text-red-600 text-sm mt-1">{{ $message }}</p>
                        @enderror
                    </div>

                    <div class="mt-6 flex items-center justify-end">
                                    <a  href="{{  route('invoices.index')  }}"  class="text-gray-500
hover:text-gray-700 dark:text-gray-300 dark:hover:text-gray-100 mr-4">
                            Cancel
                        </a>
                             <button type="submit" class="px-4 py-2 bg-blue-500 hover:bg-blue-700
text-white font-bold rounded">
                            Create Invoice
                        </button>
                    </div>
                </form>
            </div>
        </div>
    </div>
</div>

<script>
    document.addEventListener('DOMContentLoaded', function() {
        let itemCount = 0;

        // Initialize the first item
        initializeItem(0);

        // Add new item
        document.getElementById('add-item').addEventListener('click', function() {
            itemCount++;

            const itemsContainer = document.getElementById('invoice-items');
            const newItem = document.querySelector('.invoice-item').cloneNode(true);

            // Update input names and IDs
            newItem.querySelectorAll('select, input').forEach(input => {
                const name = input.getAttribute('name');
                if (name) {
                    input.setAttribute('name', name.replace(/\[\d+\]/, `[${itemCount}]`));
                }

                // Clear values
                if (input.tagName === 'SELECT') {
                    input.selectedIndex = 0;
                } else if (input.type === 'number') {
```

```
            if (input.classList.contains('quantity-input')) {
                input.value = 1;
            } else {
                input.value = '';
            }
        } else {
            input.value = '';
        }
    });

    // Show remove button
    const removeButton = newItem.querySelector('.remove-item');
    removeButton.classList.remove('hidden');

    // Add event listeners
    initializeItemEvents(newItem, itemCount);

    // Add to container
    itemsContainer.appendChild(newItem);
});

// Initialize the first item's event listeners
function initializeItem(index) {
    const item = document.querySelector('.invoice-item');
    initializeItemEvents(item, index);

    // Show remove button for all but the first item
    if (index > 0) {
        item.querySelector('.remove-item').classList.remove('hidden');
    }
}

// Initialize event listeners for an item
function initializeItemEvents(item, index) {
    const productSelect = item.querySelector('.product-select');
    const quantityInput = item.querySelector('.quantity-input');
    const priceInput = item.querySelector('.price-input');
    const serialContainer = item.querySelector('.serial-numbers-container');
    const removeButton = item.querySelector('.remove-item');

    // Product selection changes
    productSelect.addEventListener('change', function() {
        const selectedOption = this.options[this.selectedIndex];
        if (selectedOption.value) {
            priceInput.value = selectedOption.dataset.price;

            // Show/hide serial numbers field
            if (selectedOption.dataset.hasSerial === 'true') {
                serialContainer.classList.remove('hidden');
            } else {
                serialContainer.classList.add('hidden');
            }
        } else {
            priceInput.value = '';
```

```
                    serialContainer.classList.add('hidden');
                }
            });

            // Remove item
            removeButton.addEventListener('click', function() {
                item.remove();
            });
        }
    });
</script>
@endsection
```

resources/views/invoices/edit.blade.php

```
@extends('layouts.app')

@section('content')
<div class="py-12">
    <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
        <div class="bg-white overflow-hidden shadow-sm sm:rounded-lg">
            <div class="p-6 bg-white border-b border-gray-200">
                    <h1 class="text-2xl font-bold mb-4">Edit Invoice #{{ $invoice->invoice_number }}</h1>

                <form action="{{ route('invoices.update', $invoice) }}" method="POST">
                    @csrf
                    @method('PUT')

                    <!-- Client Selection -->
                    <div class="mb-4">
                        <label for="client_id" class="block text-gray-700">Client</label>
                                <select name="client_id" id="client_id" class="mt-1 block w-full rounded-md border-gray-300 shadow-sm">
                            @foreach($clients as $client)
                                    <option value="{{ $client->id }}" {{ $invoice->client_id == $client->id ? 'selected' : '' }}>
                                {{ $client->name }}
                            </option>
                            @endforeach
                        </select>
                    </div>

                    <!-- Invoice Items -->
                    <div class="mb-6">
                        <h2 class="text-lg font-semibold mb-2">Invoice Items</h2>
                        <div id="items-container">
                            @foreach($invoice->items as $index => $item)
                            <div class="item-row mb-4 flex gap-4">
                                    <select name="items[{{ $index }}][product_id]" class="flex-1 rounded-md border-gray-300 shadow-sm">
                                @foreach($products as $product)
                                    <option value="{{ $product->id }}" {{ $item->product_id == $product->id ? 'selected' : '' }}>
                                        {{ $product->name }} - ${{ $product->price }}
                                    </option>
                                @endforeach
                            </select>
                                    <input type="number" name="items[{{ $index }}][quantity]" value="{{ $item->quantity }}" min="1" class="w-20 rounded-md border-gray-300 shadow-sm">
                                <input type="number" name="items[{{ $index }}][price]" value="{{ $item->price }}" step="0.01" min="0" class="w-24 rounded-md border-gray-300 shadow-sm">
                                <button type="button" class="remove-item bg-red-500 text-white px-2 py-1 rounded">Remove</button>
                            </div>
                            @endforeach
```

```
                        </div>
                            <button type="button" id="add-item" class="bg-blue-500 text-white px-4
py-2 rounded mt-2">Add Item</button>
                    </div>

                    <div class="flex items-center justify-end">
                        <button type="submit" class="bg-green-500 text-white px-4 py-2 rounded">
                            Update Invoice
                        </button>
                    </div>
                </form>
            </div>
        </div>
    </div>
</div>

<script>
    // Same JavaScript as in create.blade.php
    document.getElementById('add-item').addEventListener('click', function() {
        const container = document.getElementById('items-container');
        const itemCount = container.querySelectorAll('.item-row').length;
        const newItem = document.createElement('div');
        newItem.className = 'item-row mb-4 flex gap-4';
        newItem.innerHTML = `
                        <select  name="items[${itemCount}][product_id]"  class="flex-1  rounded-md
border-gray-300 shadow-sm">
                @foreach($products as $product)
                     <option value="{{ $product->id }}">{{ $product->name }} - ${{ $product->price
}}</option>
                @endforeach
            </select>
                    <input  type="number"  name="items[${itemCount}][quantity]"  value="1"  min="1"
class="w-20 rounded-md border-gray-300 shadow-sm">
             <input type="number" name="items[${itemCount}][price]" step="0.01" min="0" class="w-24
rounded-md border-gray-300 shadow-sm" placeholder="Price">
                      <button  type="button"  class="remove-item  bg-red-500  text-white  px-2  py-1
rounded">Remove</button>
        `;
        container.appendChild(newItem);
    });

    document.addEventListener('click', function(e) {
        if (e.target.classList.contains('remove-item')) {
            e.target.closest('.item-row').remove();
        }
    });
</script>
@endsection
```

resources/views/invoices/index.blade.php

```
@extends('layouts.app')

@section('content')
<div class="py-12">
    <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
        <div class="bg-white overflow-hidden shadow-sm sm:rounded-lg">
            <div class="p-6 bg-white border-b border-gray-200">
                <div class="flex justify-between items-center mb-4">
                    <h1 class="text-2xl font-bold">Invoices</h1>
                       <a href="{{ route('invoices.create') }}" class="bg-blue-500 text-white px-4
py-2 rounded">
                        Create New Invoice
                    </a>
                </div>

                <div class="overflow-x-auto">
                    <table class="min-w-full">
                        <thead>
                            <tr>
                                <th class="px-4 py-2">Invoice #</th>
                                <th class="px-4 py-2">Client</th>
                                <th class="px-4 py-2">Date</th>
                                <th class="px-4 py-2">Due Date</th>
                                <th class="px-4 py-2">Total</th>
                                <th class="px-4 py-2">Actions</th>
                            </tr>
                        </thead>
                        <tbody>
                            @foreach($invoices as $invoice)
                            <tr>
                                <td class="px-4 py-2">{{ $invoice->invoice_number }}</td>
                                <td class="px-4 py-2">{{ $invoice->client->name }}</td>
                                  <td class="px-4 py-2">{{ $invoice->invoice_date->format('m/d/Y')
}}</td>
                                      <td class="px-4 py-2">{{ $invoice->due_date->format('m/d/Y')
}}</td>
                                       <td class="px-4 py-2">${{ number_format($invoice->total, 2)
}}</td>
                                <td class="px-4 py-2 flex gap-2">
                                            <a href="{{ route('invoices.show', $invoice) }}"
class="bg-blue-500 text-white px-2 py-1 rounded">View</a>
                                            <a href="{{ route('invoices.edit', $invoice) }}"
class="bg-yellow-500 text-white px-2 py-1 rounded">Edit</a>
                                        <form action="{{ route('invoices.destroy', $invoice) }}"
method="POST">
                                    @csrf
                                    @method('DELETE')
                                       <button type="submit" class="bg-red-500 text-white px-2
py-1 rounded">Delete</button>
                                </form>
                                            <a href="{{ route('invoices.pdf', $invoice) }}"
```

```
class="bg-green-500 text-white px-2 py-1 rounded">PDF</a>
                                    </td>
                                </tr>
                                @endforeach
                            </tbody>
                        </table>
                    </div>
                </div>
            </div>
        </div>
    </div>
@endsection
```

resources/views/invoices/show.blade.php

```
@extends('layouts.app')

@section('content')
<div class="py-12">
    <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
        <div class="bg-white overflow-hidden shadow-sm sm:rounded-lg">
            <div class="p-6 bg-white border-b border-gray-200">
                <h1 class="text-2xl font-bold mb-4">Invoice #{{ $invoice->invoice_number }}</h1>

                <div class="mb-4">
                    <h2 class="text-lg font-semibold">Client Information</h2>
                    <p>{{ $invoice->client->name }}</p>
                    <p>{{ $invoice->client->email }}</p>
                </div>

                <table class="min-w-full mb-6">
                    <thead>
                        <tr>
                            <th class="px-4 py-2">Product</th>
                            <th class="px-4 py-2">Quantity</th>
                            <th class="px-4 py-2">Price</th>
                            <th class="px-4 py-2">Total</th>
                        </tr>
                    </thead>
                    <tbody>
                        @foreach($invoice->items as $item)
                        <tr>
                            <td class="px-4 py-2">{{ $item->product->name }}</td>
                            <td class="px-4 py-2">{{ $item->quantity }}</td>
                            <td class="px-4 py-2">${{ number_format($item->price, 2) }}</td>
                                        <td class="px-4 py-2">${{ number_format($item->price *
$item->quantity, 2) }}</td>
                        </tr>
                        @endforeach
                    </tbody>
                </table>

                <div class="text-right">
                    <p class="font-bold">Subtotal: ${{ number_format($invoice->subtotal, 2) }}</p>
                    <p>GST (10%): ${{ number_format($invoice->gst_amount, 2) }}</p>
                         <p class="text-xl font-bold">Total: ${{ number_format($invoice->total, 2)
}}</p>
                </div>

                <div class="mt-4">
                     <a href="{{ route('invoices.pdf', $invoice) }}" class="bg-blue-500 text-white
px-4 py-2 rounded">
                        Download PDF
                    </a>
                       <a href="{{ route('invoices.index') }}" class="ml-2 bg-gray-500 text-white
px-4 py-2 rounded">
```

```
                    Back to List
                </a>
            </div>
        </div>
    </div>
</div>
@endsection
```

resources/views/invoices/template/custom.blade.php

```
<!-- resources/views/invoices/templates/custom.blade.php -->
<!DOCTYPE html>
<html>
<head>
    <title>Invoice {{ $invoice->invoice_number }}</title>
    <style>
        body { font-family: Arial, sans-serif; }
        .header { text-align: center; margin-bottom: 20px; }
        .details { margin-bottom: 30px; }
        table { width: 100%; border-collapse: collapse; }
        th, td { border: 1px solid #ddd; padding: 8px; text-align: left; }
        .totals { float: right; width: 300px; margin-top: 20px; }
        .serial-numbers { font-size: 0.8em; color: #666; margin-top: 2px; }
    </style>
</head>
<body>
    <div class="header">
        <h2>INVOICE</h2>
        <p>#{{ $invoice->invoice_number }}</p>
    </div>

    <div class="details">
        <p><strong>Date:</strong> {{ $invoice->invoice_date->format('d/m/Y') }}</p>
        <p><strong>Due Date:</strong> {{ $invoice->due_date->format('d/m/Y') }}</p>
    </div>

    <table>
        <thead>
            <tr>
                <th>Description</th>
                <th>Qty</th>
                <th>Unit Price (inc GST)</th>
                <th>Total (inc GST)</th>
            </tr>
        </thead>
        <tbody>
            @foreach($invoice->items as $item)
            <tr>
                <td>
                    {{ $item->product->name }}
                    @if($item->serialNumbers->count())
                    <div class="serial-numbers">
                        <strong>Serial Numbers:</strong>
                        {{ $item->serialNumbers->pluck('serial_number')->implode(', ') }}
                    </div>
                    @endif
                </td>
                <td>{{ $item->quantity }}</td>
                <td>${{ number_format($item->price, 2) }}</td>
                <td>${{ number_format($item->price * $item->quantity, 2) }}</td>
            </tr>
```

```
            @endforeach
        </tbody>
    </table>

    <div class="totals">
        <table>
            <tr>
                <th>Subtotal (ex GST):</th>
                <td>${{ number_format($invoice->subtotal, 2) }}</td>
            </tr>
            <tr>
                <th>GST (10%):</th>
                <td>${{ number_format($invoice->gst_amount, 2) }}</td>
            </tr>
            <tr>
                <th>Total:</th>
                <td>${{ number_format($invoice->total, 2) }}</td>
            </tr>
        </table>
    </div>
</body>
</html>
```

resources/views/invoices/templates/custom.blade.php

```
<!DOCTYPE html>
<html>
<head>
    <title>Invoice - {{ $invoice->invoice_number }}</title>
    <style>
        body { font-family: Arial, sans-serif; }
        .header { text-align: center; margin-bottom: 20px; }
        .details { margin-bottom: 30px; }
        table { width: 100%; border-collapse: collapse; }
        th, td { border: 1px solid #ddd; padding: 8px; text-align: left; }
        th { background-color: #f2f2f2; }
        .totals { float: right; margin-top: 20px; }
    </style>
</head>
<body>
    <div class="header">
        <h2>INVOICE</h2>
        <p>#{{ $invoice->invoice_number }}</p>
    </div>

    <div class="details">
        <p><strong>Date:</strong> {{ $invoice->invoice_date->format('d/m/Y') }}</p>
        <p><strong>Due Date:</strong> {{ $invoice->due_date->format('d/m/Y') }}</p>
        <p><strong>Client:</strong> {{ $invoice->client->name }}</p>
    </div>

    <table>
        <thead>
            <tr>
                <th>Product</th>
                <th>Qty</th>
                <th>Price (inc GST)</th>
                <th>Total (inc GST)</th>
            </tr>
        </thead>
        <tbody>
            @foreach($invoice->items as $item)
            <tr>
                <td>
                    {{ $item->product->name }}
                    @if($item->serialNumbers->isNotEmpty())
                        <div style="font-size: smaller;">
                            <strong>Serial Numbers:</strong>
                            {{ $item->serialNumbers->pluck('serial_number')->implode(', ') }}
                        </div>
                    @endif
                </td>
                <td>{{ $item->quantity }}</td>
                <td>${{ number_format($item->price, 2) }}</td>
                <td>${{ number_format($item->price * $item->quantity, 2) }}</td>
            </tr>
```

```
            @endforeach
        </tbody>
    </table>

    <div class="totals">
        <p><strong>Subtotal:</strong> ${{ number_format($invoice->subtotal, 2) }}</p>
        <p><strong>GST (10%):</strong> ${{ number_format($invoice->gst_amount, 2) }}</p>
        <p><strong>Total:</strong> ${{ number_format($invoice->total, 2) }}</p>
    </div>
</body>
</html>
```

resources/views/layouts/app.blade.php

```
<!DOCTYPE html>
<html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
<head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>{{ config('app.name', 'Laravel') }}</title>

    @vite(['resources/css/app.css', 'resources/js/app.js'])
</head>
<body class="font-sans antialiased">
    <div class="min-h-screen bg-gray-100 dark:bg-gray-900">
        @include('layouts.navigation')

        <!-- Page Heading -->
        @if (isset($header))
            <header class="bg-white dark:bg-gray-800 shadow">
                <div class="max-w-7xl mx-auto py-6 px-4 sm:px-6 lg:px-8">
                    {{ $header }}
                </div>
            </header>
        @endif

        <!-- Page Content -->
        <main>
            @yield('content')
        </main>
    </div>
</body>
</html>
```

resources/views/layouts/guest.blade.php

```
<!DOCTYPE html>
<html lang="{{ str_replace('_', '-', app()->getLocale()) }}">
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <meta name="csrf-token" content="{{ csrf_token() }}">

        <title>{{ config('app.name', 'Laravel') }}</title>

        <!-- Fonts -->
        <link rel="preconnect" href="https://fonts.bunny.net">
                  <link  href="https://fonts.bunny.net/css?family=figtree:400,500,600&display=swap"
rel="stylesheet" />

        <!-- Scripts -->
        @vite(['resources/css/app.css', 'resources/js/app.js'])
    </head>
    <body class="font-sans text-gray-900 antialiased">
             <div  class="min-h-screen  flex  flex-col  sm:justify-center  items-center  pt-6  sm:pt-0
bg-gray-100 dark:bg-gray-900">
            <div>
                <a href="/">
                    <x-application-logo class="w-20 h-20 fill-current text-gray-500" />
                </a>
            </div>

               <div class="w-full sm:max-w-md mt-6 px-6 py-4 bg-white dark:bg-gray-800 shadow-md
overflow-hidden sm:rounded-lg">
                {{ $slot }}
            </div>
        </div>
    </body>
</html>
```

resources/views/layouts/navigation.blade.php

```
<nav x-data="{ open: false }" class="bg-white dark:bg-gray-800 border-b border-gray-100 dark:border-gray-700">
    <!-- Primary Navigation Menu -->
    <div class="max-w-7xl mx-auto px-4 sm:px-6 lg:px-8">
        <div class="flex justify-between h-16">
            <div class="flex">
                <!-- Logo -->
                <div class="shrink-0 flex items-center">
                    <a href="{{ route('dashboard') }}">
                            <x-application-logo class="block h-9 w-auto fill-current text-gray-800 dark:text-gray-200" />
                    </a>
                </div>

                <!-- Navigation Links -->
                <div class="hidden space-x-8 sm:-my-px sm:ms-10 sm:flex">
                                                        <x-nav-link :href="route('dashboard')" :active="request()->routeIs('dashboard')">
                        {{ __('Dashboard') }}
                    </x-nav-link>
                </div>
            </div>

            <!-- Settings Dropdown -->
            <div class="hidden sm:flex sm:items-center sm:ms-6">
                <x-dropdown align="right" width="48">
                    <x-slot name="trigger">
                                    <button class="inline-flex items-center px-3 py-2 border border-transparent text-sm leading-4 font-medium rounded-md text-gray-500 dark:text-gray-400 bg-white dark:bg-gray-800 hover:text-gray-700 dark:hover:text-gray-300 focus:outline-none transition ease-in-out duration-150">
                            <div>{{ Auth::user()->name }}</div>

                            <div class="ms-1">
                                                            <svg class="fill-current h-4 w-4" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 20 20">
                                        <path fill-rule="evenodd" d="M5.293 7.293a1 1 0 011.414 0L10 10.586l3.293-3.293a1 1 0 111.414 1.414l-4 4a1 1 0 01-1.414 0l-4-4a1 1 0 010-1.414z" clip-rule="evenodd" />
                                    </svg>
                            </div>
                        </button>
                    </x-slot>

                    <x-slot name="content">
                        <x-dropdown-link :href="route('profile.edit')">
                            {{ __('Profile') }}
                        </x-dropdown-link>

                        <!-- Authentication -->
                        <form method="POST" action="{{ route('logout') }}">
```

```
                        @csrf

                        <x-dropdown-link :href="route('logout')"
                                onclick="event.preventDefault();
                                            this.closest('form').submit();">
                            {{ __('Log Out') }}
                        </x-dropdown-link>
                    </form>
                </x-slot>
            </x-dropdown>
        </div>

        <!-- Hamburger -->
        <div class="-me-2 flex items-center sm:hidden">
             <button @click="open = ! open" class="inline-flex items-center justify-center p-2
rounded-md   text-gray-400   dark:text-gray-500   hover:text-gray-500   dark:hover:text-gray-400
hover:bg-gray-100        dark:hover:bg-gray-900        focus:outline-none        focus:bg-gray-100
dark:focus:bg-gray-900   focus:text-gray-500   dark:focus:text-gray-400   transition   duration-150
ease-in-out">
                <svg class="h-6 w-6" stroke="currentColor" fill="none" viewBox="0 0 24 24">
                                    <path  :class="{'hidden': open, 'inline-flex': ! open }"
class="inline-flex"  stroke-linecap="round"  stroke-linejoin="round"  stroke-width="2"  d="M4  6h16M4
12h16M4 18h16" />
                        <path :class="{'hidden': ! open, 'inline-flex': open }" class="hidden"
stroke-linecap="round" stroke-linejoin="round" stroke-width="2" d="M6 18L18 6M6 6l12 12" />
                </svg>
            </button>
        </div>
    </div>

    <!-- Responsive Navigation Menu -->
    <div :class="{'block': open, 'hidden': ! open}" class="hidden sm:hidden">
        <div class="pt-2 pb-3 space-y-1">
                                        <x-responsive-nav-link    :href="route('dashboard')"
:active="request()->routeIs('dashboard')">
                {{ __('Dashboard') }}
            </x-responsive-nav-link>
        </div>

        <!-- Responsive Settings Options -->
        <div class="pt-4 pb-1 border-t border-gray-200 dark:border-gray-600">
            <div class="px-4">
                    <div  class="font-medium  text-base  text-gray-800  dark:text-gray-200">{{
Auth::user()->name }}</div>
                <div class="font-medium text-sm text-gray-500">{{ Auth::user()->email }}</div>
            </div>

            <div class="mt-3 space-y-1">
                <x-responsive-nav-link :href="route('profile.edit')">
                    {{ __('Profile') }}
                </x-responsive-nav-link>

                <!-- Authentication -->
```

```
                <form method="POST" action="{{ route('logout') }}">
                    @csrf

                    <x-responsive-nav-link :href="route('logout')"
                            onclick="event.preventDefault();
                                        this.closest('form').submit();">
                        {{ __('Log Out') }}
                    </x-responsive-nav-link>
                </form>
            </div>
        </div>
    </div>
</nav>
```

resources/views/products/index.blade.php

```
@extends('layouts.app')

@section('content')
<div class="py-12">
    <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
        <div class="bg-white dark:bg-gray-800 overflow-hidden shadow-sm sm:rounded-lg">
                        <div  class="p-6  bg-white  dark:bg-gray-800  border-b  border-gray-200
dark:border-gray-700">
                <div class="flex justify-between items-center mb-6">
                                            <h2  class="text-xl  font-semibold  text-gray-800
dark:text-gray-200">Products</h2>
                            <a href="{{ route('products.create') }}" class="px-4 py-2 bg-blue-500
hover:bg-blue-700 text-white font-bold rounded">
                        Add New Product
                    </a>
                </div>

                <!-- Search Form -->
                <div class="mb-6">
                    <form action="{{ route('products.index') }}" method="GET" class="flex">
                                <input type="text" name="search" placeholder="Search products..."
value="{{ request('search') }}"
                               class="flex-1 border-gray-300 dark:border-gray-700 dark:bg-gray-900
dark:text-gray-300   focus:border-indigo-500   dark:focus:border-indigo-600   focus:ring-indigo-500
dark:focus:ring-indigo-600 rounded-l-md shadow-sm">
                        <button type="submit" class="px-4 py-2 bg-gray-800 dark:bg-gray-200 border
border-transparent  rounded-r-md  font-semibold  text-xs  text-white  dark:text-gray-800  uppercase
tracking-widest   hover:bg-gray-700   dark:hover:bg-white   focus:bg-gray-700   dark:focus:bg-white
active:bg-gray-900  dark:active:bg-gray-300  focus:outline-none  focus:ring-2  focus:ring-indigo-500
focus:ring-offset-2 dark:focus:ring-offset-gray-800">
                            Search
                        </button>
                    </form>
                </div>

                <!-- Products Table -->
                <div class="overflow-x-auto">
                    <table class="min-w-full divide-y divide-gray-200 dark:divide-gray-700">
                        <thead class="bg-gray-50 dark:bg-gray-700">
                            <tr>
                                    <th scope="col" class="px-6 py-3 text-left text-xs font-medium
text-gray-500 dark:text-gray-300 uppercase tracking-wider">Name</th>
                                    <th scope="col" class="px-6 py-3 text-left text-xs font-medium
text-gray-500 dark:text-gray-300 uppercase tracking-wider">Price</th>
                                    <th scope="col" class="px-6 py-3 text-left text-xs font-medium
text-gray-500 dark:text-gray-300 uppercase tracking-wider">Has Serial</th>
                                    <th scope="col" class="px-6 py-3 text-left text-xs font-medium
text-gray-500 dark:text-gray-300 uppercase tracking-wider">Actions</th>
                            </tr>
                        </thead>
                                <tbody class="bg-white divide-y divide-gray-200 dark:bg-gray-800
```

```
dark:divide-gray-700">
                            @forelse ($products as $product)
                                <tr>
                                    <td class="px-6 py-4 whitespace-nowrap">
                                                    <div class="text-sm font-medium text-gray-900
dark:text-gray-100">
                                            {{ $product->name }}
                                        </div>
                                        <div class="text-sm text-gray-500 dark:text-gray-400">
                                            {{ Str::limit($product->description, 50) }}
                                        </div>
                                    </td>
                                    <td class="px-6 py-4 whitespace-nowrap">
                                        <div class="text-sm text-gray-500 dark:text-gray-400">
                                            ${{ number_format($product->price, 2) }}
                                        </div>
                                    </td>
                                    <td class="px-6 py-4 whitespace-nowrap">
                                                    <span class="px-2 inline-flex text-xs leading-5
font-semibold    rounded-full    {{    $product->has_serial    ?    'bg-green-100    text-green-800
dark:bg-green-800    dark:text-green-100'    :    'bg-gray-100    text-gray-800    dark:bg-gray-700
dark:text-gray-300' }}">
                                            {{ $product->has_serial ? 'Yes' : 'No' }}
                                        </span>
                                    </td>
                                    <td class="px-6 py-4 whitespace-nowrap text-sm font-medium">
                                                <a href="{{ route('products.show', $product) }}"
class="text-blue-600      hover:text-blue-900      dark:text-blue-400      dark:hover:text-blue-600
mr-3">View</a>
                                                <a href="{{ route('products.edit', $product) }}"
class="text-indigo-600   hover:text-indigo-900   dark:text-indigo-400   dark:hover:text-indigo-600
mr-3">Edit</a>
                                            <form action="{{ route('products.destroy', $product) }}"
method="POST" class="inline">
                                                @csrf
                                                @method('DELETE')
                                                        <button type="submit" class="text-red-600
hover:text-red-900  dark:text-red-400  dark:hover:text-red-600"  onclick="return  confirm('Are  you
sure you want to delete this product?')">Delete</button>
                                            </form>
                                    </td>
                                </tr>
                            @empty
                                <tr>
                                        <td colspan="4" class="px-6 py-4 whitespace-nowrap text-sm
text-gray-500 dark:text-gray-400 text-center">
                                            No products found.
                                    </td>
                                </tr>
                            @endforelse
                        </tbody>
                    </table>
                </div>
```

```
                <!-- Pagination -->
                <div class="mt-4">
                    {{ $products->links() }}
                </div>
            </div>
        </div>
    </div>
</div>
@endsection
```

resources/views/profile/edit.blade.php

```
<x-app-layout>
    <x-slot name="header">
        <h2 class="font-semibold text-xl text-gray-800 dark:text-gray-200 leading-tight">
            {{ __('Profile') }}
        </h2>
    </x-slot>

    <div class="py-12">
        <div class="max-w-7xl mx-auto sm:px-6 lg:px-8 space-y-6">
            <div class="p-4 sm:p-8 bg-white dark:bg-gray-800 shadow sm:rounded-lg">
                <div class="max-w-xl">
                    @include('profile.partials.update-profile-information-form')
                </div>
            </div>

            <div class="p-4 sm:p-8 bg-white dark:bg-gray-800 shadow sm:rounded-lg">
                <div class="max-w-xl">
                    @include('profile.partials.update-password-form')
                </div>
            </div>

            <div class="p-4 sm:p-8 bg-white dark:bg-gray-800 shadow sm:rounded-lg">
                <div class="max-w-xl">
                    @include('profile.partials.delete-user-form')
                </div>
            </div>
        </div>
    </div>
</x-app-layout>
```

resources/views/profile/partials/delete-user-form.blade.php

```
<section class="space-y-6">
    <header>
        <h2 class="text-lg font-medium text-gray-900 dark:text-gray-100">
            {{ __('Delete Account') }}
        </h2>

        <p class="mt-1 text-sm text-gray-600 dark:text-gray-400">
            {{ __('Once your account is deleted, all of its resources and data will be permanently deleted. Before deleting your account, please download any data or information that you wish to retain.') }}
        </p>
    </header>

    <x-danger-button
        x-data=""
        x-on:click.prevent="$dispatch('open-modal', 'confirm-user-deletion')"
    >{{ __('Delete Account') }}</x-danger-button>

    <x-modal name="confirm-user-deletion" :show="$errors->userDeletion->isNotEmpty()" focusable>
        <form method="post" action="{{ route('profile.destroy') }}" class="p-6">
            @csrf
            @method('delete')

            <h2 class="text-lg font-medium text-gray-900 dark:text-gray-100">
                {{ __('Are you sure you want to delete your account?') }}
            </h2>

            <p class="mt-1 text-sm text-gray-600 dark:text-gray-400">
                    {{ __('Once your account is deleted, all of its resources and data will be permanently deleted. Please enter your password to confirm you would like to permanently delete your account.') }}
            </p>

            <div class="mt-6">
                <x-input-label for="password" value="{{ __('Password') }}" class="sr-only" />

                <x-text-input
                    id="password"
                    name="password"
                    type="password"
                    class="mt-1 block w-3/4"
                    placeholder="{{ __('Password') }}"
                />

                <x-input-error :messages="$errors->userDeletion->get('password')" class="mt-2" />
            </div>

            <div class="mt-6 flex justify-end">
                <x-secondary-button x-on:click="$dispatch('close')">
                    {{ __('Cancel') }}
                </x-secondary-button>
```

```
                <x-danger-button class="ms-3">
                    {{ __('Delete Account') }}
                </x-danger-button>
            </div>
        </form>
    </x-modal>
</section>
```

resources/views/profile/partials/update-password-form.blade.php

```
<section>
    <header>
        <h2 class="text-lg font-medium text-gray-900 dark:text-gray-100">
            {{ __('Update Password') }}
        </h2>

        <p class="mt-1 text-sm text-gray-600 dark:text-gray-400">
            {{ __('Ensure your account is using a long, random password to stay secure.') }}
        </p>
    </header>

    <form method="post" action="{{ route('password.update') }}" class="mt-6 space-y-6">
        @csrf
        @method('put')

        <div>
             <x-input-label for="update_password_current_password" :value="__('Current Password')"
/>
                      <x-text-input  id="update_password_current_password"  name="current_password"
type="password" class="mt-1 block w-full" autocomplete="current-password" />
                       <x-input-error  :messages="$errors->updatePassword->get('current_password')"
class="mt-2" />
        </div>

        <div>
            <x-input-label for="update_password_password" :value="__('New Password')" />
                     <x-text-input  id="update_password_password"  name="password"  type="password"
class="mt-1 block w-full" autocomplete="new-password" />
            <x-input-error :messages="$errors->updatePassword->get('password')" class="mt-2" />
        </div>

        <div>
                   <x-input-label  for="update_password_password_confirmation"  :value="__('Confirm
Password')" />
              <x-text-input id="update_password_password_confirmation" name="password_confirmation"
type="password" class="mt-1 block w-full" autocomplete="new-password" />
                  <x-input-error  :messages="$errors->updatePassword->get('password_confirmation')"
class="mt-2" />
        </div>

        <div class="flex items-center gap-4">
            <x-primary-button>{{ __('Save') }}</x-primary-button>

            @if (session('status') === 'password-updated')
                <p
                    x-data="{ show: true }"
                    x-show="show"
                    x-transition
                    x-init="setTimeout(() => show = false, 2000)"
                    class="text-sm text-gray-600 dark:text-gray-400"
                >{{ __('Saved.') }}</p>
```

```
            @endif
        </div>
    </form>
</section>
```

resources/views/profile/partials/update-profile-information-form.blade.php

```
<section>
    <header>
        <h2 class="text-lg font-medium text-gray-900 dark:text-gray-100">
            {{ __('Profile Information') }}
        </h2>

        <p class="mt-1 text-sm text-gray-600 dark:text-gray-400">
            {{ __("Update your account's profile information and email address.") }}
        </p>
    </header>

    <form id="send-verification" method="post" action="{{ route('verification.send') }}">
        @csrf
    </form>

    <form method="post" action="{{ route('profile.update') }}" class="mt-6 space-y-6">
        @csrf
        @method('patch')

        <div>
            <x-input-label for="name" :value="__('Name')" />
                     <x-text-input  id="name"  name="name"  type="text"  class="mt-1  block  w-full"
:value="old('name', $user->name)" required autofocus autocomplete="name" />
            <x-input-error class="mt-2" :messages="$errors->get('name')" />
        </div>

        <div>
            <x-input-label for="email" :value="__('Email')" />
                    <x-text-input  id="email"  name="email"  type="email"  class="mt-1  block  w-full"
:value="old('email', $user->email)" required autocomplete="username" />
            <x-input-error class="mt-2" :messages="$errors->get('email')" />

                         @if  ($user  instanceof  \Illuminate\Contracts\Auth\MustVerifyEmail  &&  !
$user->hasVerifiedEmail())
                <div>
                    <p class="text-sm mt-2 text-gray-800 dark:text-gray-200">
                        {{ __('Your email address is unverified.') }}

                            <button form="send-verification" class="underline text-sm text-gray-600
dark:text-gray-400  hover:text-gray-900  dark:hover:text-gray-100  rounded-md  focus:outline-none
focus:ring-2 focus:ring-offset-2 focus:ring-indigo-500 dark:focus:ring-offset-gray-800">
                                {{ __('Click here to re-send the verification email.') }}
                        </button>
                    </p>

                    @if (session('status') === 'verification-link-sent')
                        <p class="mt-2 font-medium text-sm text-green-600 dark:text-green-400">
                             {{ __('A new verification link has been sent to your email address.')
}}
                        </p>
                    @endif
```

```
                </div>
            @endif
        </div>

        <div class="flex items-center gap-4">
            <x-primary-button>{{ __('Save') }}</x-primary-button>

            @if (session('status') === 'profile-updated')
                <p
                    x-data="{ show: true }"
                    x-show="show"
                    x-transition
                    x-init="setTimeout(() => show = false, 2000)"
                    class="text-sm text-gray-600 dark:text-gray-400"
                >{{ __('Saved.') }}</p>
            @endif
        </div>
    </form>
</section>
```

resources/views/settings/index.blade.php

```
<x-app-layout>
    <x-slot name="header">
        <h2 class="font-semibold text-xl text-gray-800 leading-tight">
            {{ __('System Settings') }}
        </h2>
    </x-slot>

    <div class="py-12">
        <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
            <div class="bg-white overflow-hidden shadow-sm sm:rounded-lg">
                <div class="p-6 bg-white border-b border-gray-200">
                    <form method="POST" action="{{ route('settings.update') }}">
                        @csrf

                        <div class="grid grid-cols-1 gap-6 mt-4">
                            <div>
                                 <x-input-label for="invoice_prefix" :value="__('Invoice Prefix')"
/>
                                <x-text-input id="invoice_prefix" class="block mt-1 w-full"
                                    type="text" name="invoice_prefix"
                                    value="{{ $settings['invoice.prefix'] ?? 'INV-' }}" />
                            </div>

                            <div>
                                <x-input-label for="gst_rate" :value="__('GST Rate (%)')" />
                                <x-text-input id="gst_rate" class="block mt-1 w-full"
                                    type="number" step="0.01" name="gst_rate"
                                    value="{{ ($settings['gst_rate'] ?? 0.10) * 100 }}" />
                            </div>

                            <div>
                                 <x-input-label for="default_template" :value="__('Default Invoice
Template')" />
                                <select id="default_template" name="default_template" class="block
mt-1 w-full border-gray-300 focus:border-indigo-500 focus:ring-indigo-500 rounded-md shadow-sm">
                                    <option value="default" {{
($settings['invoice.default_template'] ?? 'custom') == 'default' ? 'selected' : ''
}}>Default</option>
                                    <option value="custom" {{
($settings['invoice.default_template'] ?? 'custom') == 'custom' ? 'selected' : ''
}}>Custom</option>
                                </select>
                            </div>
                        </div>

                        <div class="flex items-center justify-end mt-4">
                            <x-primary-button class="ml-3">
                                {{ __('Save Settings') }}
                            </x-primary-button>
                        </div>
                    </form>
```

```
                </div>
            </div>
        </div>
    </div>
</x-app-layout>
```

routes/api.php

```
<?php

use Illuminate\Http\Request;
use Illuminate\Support\Facades\Route;

/*
|--------------------------------------------------------------------------
| API Routes
|--------------------------------------------------------------------------
|
| Here is where you can register API routes for your application. These
| routes are loaded by the RouteServiceProvider and all of them will
| be assigned to the "api" middleware group. Make something great!
|
*/

Route::middleware('auth:sanctum')->get('/user', function (Request $request) {
    return $request->user();
});
```

routes/auth.php

```
<?php

use App\Http\Controllers\Auth\AuthenticatedSessionController;
use App\Http\Controllers\Auth\ConfirmablePasswordController;
use App\Http\Controllers\Auth\EmailVerificationNotificationController;
use App\Http\Controllers\Auth\EmailVerificationPromptController;
use App\Http\Controllers\Auth\NewPasswordController;
use App\Http\Controllers\Auth\PasswordController;
use App\Http\Controllers\Auth\PasswordResetLinkController;
use App\Http\Controllers\Auth\RegisteredUserController;
use App\Http\Controllers\Auth\VerifyEmailController;
use Illuminate\Support\Facades\Route;

Route::middleware('guest')->group(function () {
    Route::get('register', [RegisteredUserController::class, 'create'])
                ->name('register');

    Route::post('register', [RegisteredUserController::class, 'store']);

    Route::get('login', [AuthenticatedSessionController::class, 'create'])
                ->name('login');

    Route::post('login', [AuthenticatedSessionController::class, 'store']);

    Route::get('forgot-password', [PasswordResetLinkController::class, 'create'])
                ->name('password.request');

    Route::post('forgot-password', [PasswordResetLinkController::class, 'store'])
                ->name('password.email');

    Route::get('reset-password/{token}', [NewPasswordController::class, 'create'])
                ->name('password.reset');

    Route::post('reset-password', [NewPasswordController::class, 'store'])
                ->name('password.store');
});

Route::middleware('auth')->group(function () {
    Route::get('verify-email', EmailVerificationPromptController::class)
                ->name('verification.notice');

    Route::get('verify-email/{id}/{hash}', VerifyEmailController::class)
                ->middleware(['signed', 'throttle:6,1'])
                ->name('verification.verify');

                                                    Route::post('email/verification-notification',
[EmailVerificationNotificationController::class, 'store'])
                ->middleware('throttle:6,1')
                ->name('verification.send');

    Route::get('confirm-password', [ConfirmablePasswordController::class, 'show'])
```

```
                ->name('password.confirm');

    Route::post('confirm-password', [ConfirmablePasswordController::class, 'store']);

    Route::put('password', [PasswordController::class, 'update'])->name('password.update');

    Route::post('logout', [AuthenticatedSessionController::class, 'destroy'])
                ->name('logout');
});
```

routes/channels.php

```
<?php

use Illuminate\Support\Facades\Broadcast;

/*
|--------------------------------------------------------------------------
| Broadcast Channels
|--------------------------------------------------------------------------
|
| Here you may register all of the event broadcasting channels that your
| application supports. The given channel authorization callbacks are
| used to check if an authenticated user can listen to the channel.
|
*/

Broadcast::channel('App.Models.User.{id}', function ($user, $id) {
    return (int) $user->id === (int) $id;
});
```

routes/console.php

```
<?php

use Illuminate\Foundation\Inspiring;
use Illuminate\Support\Facades\Artisan;

/*
|--------------------------------------------------------------------------
| Console Routes
|--------------------------------------------------------------------------
|
| This file is where you may define all of your Closure based console
| commands. Each Closure is bound to a command instance allowing a
| simple approach to interacting with each command's IO methods.
|
*/

Artisan::command('inspire', function () {
    $this->comment(Inspiring::quote());
})->purpose('Display an inspiring quote');
```

routes/web.php

```
<?php
use App\Http\Controllers\ProfileController;
use App\Http\Controllers\InvoiceController;
use App\Http\Controllers\ClientController;
use App\Http\Controllers\ProductController;
use App\Http\Controllers\DashboardController;
use App\Http\Controllers\SettingsController;
use Illuminate\Support\Facades\Route;
use Illuminate\Support\Facades\Auth;

/*
|--------------------------------------------------------------------------
| Web Routes
|--------------------------------------------------------------------------
*/

// Simple home redirect
Route::get('/', function () {
    return redirect()->route('login');
});

// Development-only routes
if (app()->environment('local')) {
    Route::get('/dev/login', function() {
        // Create test user if doesn't exist
        $user = \App\Models\User::firstOrCreate(
            ['email' => 'test@example.com'],
            [
                'name' => 'Test User',
                'password' => bcrypt('password')
            ]
        );

        Auth::login($user);
        return redirect('/dashboard');
    });

    Route::get('/dev/invoice', function() {
        // Ensure we're logged in
        if (!Auth::check()) {
            return redirect('/dev/login');
        }
        // Create test client if doesn't exist
        $client = \App\Models\Client::firstOrCreate(
            ['email' => 'client@example.com'],
            [
                'name' => 'Test Client',
                'phone' => '1234567890',
                'address' => '123 Test St'
            ]
        );
```

```
        // Create test invoice
        $invoice = \App\Models\Invoice::firstOrCreate(
            ['invoice_number' => 'TEST-001'],
            [
                'client_id' => $client->id,
                'invoice_date' => now(),
                'due_date' => now()->addDays(30),
                'subtotal' => 100,
                'gst_amount' => 10,
                'total' => 110,
                'notes' => 'Test invoice'
            ]
        );

        return redirect()->route('invoices.show', $invoice);
    });
}

// Authenticated routes
Route::middleware(['auth', 'verified'])->group(function () {
    // Dashboard
    Route::get('/dashboard', [DashboardController::class, 'index'])->name('dashboard');

    // Profile Routes
    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');

    // Invoice Routes
    Route::resource('invoices', InvoiceController::class);
    Route::get('invoices/{invoice}/pdf', [InvoiceController::class, 'pdf'])->name('invoices.pdf');

    // Client Routes
    Route::resource('clients', ClientController::class);

    // Product Routes
    Route::resource('products', ProductController::class);

    // Settings Routes
    Route::get('/settings', [SettingsController::class, 'index'])->name('settings.index');
    Route::post('/settings', [SettingsController::class, 'update'])->name('settings.update');
});

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