Customer

01	web	`Route::get('/CustomerPage', [CustomerController::class, 'CustomerPage'])->name('CustomerPage');`	
02	custmerPage method	`public function CustomerPage(Request $request){` `$user_id = $request->header('id');` `$customers = Customer::where('user_id', 5)->get();` `return Inertia::render('CustomerPage', ['customers' => $customers]);` `}`	app/Http/Controllers /CustomerController.php
03	CustomePage.vue	`<template>` `<SideNavLayout>` `<CustomerList/>` `</SideNavLayout>` `</template>` `<script setup>` `import SideNavLayout from "../Layout/SideNavLayout.vue";` `import CustomerList from "../Components/Customer/CustomerList.vue";` `</script>`	resources/js/Pages /CustomerPage.vue
04	CustomerList.vue	`<template>` `<div class="container-fluid">` `<div class="row">` `<div class="col-12"></div>` `<div class="col-12">` `<div class="card">` `<div class="card-body">` `<div>`	resources/js/Components /Customer /CustomerList.vue

```
                            <h3>Customer</h3>
                        </div>
                        <hr />
                        <div class="float-end">
                            <Link href="/CustomerSavePage?id=0"
class="btn btn-success mx-3 btn-sm">
                                Add Customer
                            </Link>
                        </div>

                        <!-- Modal -->

                        <div>
                            <input placeholder="Search..."
class="form-control mb-2 w-auto form-control-sm" type="text"
                                v-model="searchValue">
                            <EasyDataTable
                                buttons-pagination
                                alternating :headers="Header"
                                :items="Item"
                                :rows-per-page="10"
                                :search-field="searchField"
                                :search-value="searchValue" show-
index>
                                <template #item-action="{ id }">
                                    <Link class="btn btn-success
mx-3 btn-sm" :href="`/CustomerSavePage?id=${id}`">Edit
                                    </Link>
                                    <button class="btn btn-danger
btn-sm" @click="DeleteClick(id)">Delete</button>
                                </template>
                            </EasyDataTable>
```

```
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
</template>

<script setup>
import { usePage, router, Link } from '@inertiajs/vue3'
import { createToaster } from "@meforma/vue-toaster";
const toaster = createToaster();
import { ref } from "vue";

let page = usePage()

const Header = [
    { text: "Name", value: "name" },
    { text: "Email", value: "email" },
    { text: "Mobile", value: "mobile" },
    { text: "Action", value: "action" },
];

const Item = ref(page.props.customers)

const searchValue = ref()

const DeleteClick = (id) => {
    let text = "Do you want to delete";
    if (confirm(text) === true) {
        router.get(`/delete-customer/${id}`)
        toaster.success('Customer Deleted Successfully');
```

		`} else {` `text = "You canceled!";` `}` `}` `</script>`	
# Customer Save & update			
01	web	`Route::get('/CustomerSavePage', [CustomerController::class, 'CustomerSavePage'])->name('CustomerSavePage');`	
02	CustomerSavePage method	`public function CustomerSavePage(Request $request){` `$user_id=$request->header('id');` `$id=$request->query('id');` `$customer=Customer::where('id',$id)->where('user_id',$user_id)->first();` `return Inertia::render('CustomerSavePage',['customer'=>$customer]);` `}`	app/Http/Controllers /CustomerController.php
03	CustomerSavePage.vue	`<template>` `<SideNavLayout>` `<CustomerSaveForm/>` `</SideNavLayout>` `</template>` `<script setup>` `import SideNavLayout from "../Layout/SideNavLayout.vue";` `import CustomerSaveForm from "../Components/Customer/CustomerSaveForm.vue";`	resources/js/Pages /CustomerSavePage.vue

		`</script>`	
04	**CustomerSaveForm.vue**	`<template>` `  <div class="container-fluid">` `  <div class="row d-flex justify-content-center">` `  <div class="col-md-6">` `  <div class="card">` `  <div class="card-body">` `  <div class="float-end">` `  <Link href="/CustomerPage" class="btn btn-success mx-3 btn-sm">` `  Back` `  </Link>` `  </div>` `  <form @submit.prevent="submit">` `  <div class="card-body">` `  <h4>Save Customer</h4>` `  <input id="name" name="name" v-model="form.name" placeholder="Customer Name"` `  class="form-control" type="text" />` `   ` `  <input id="email" name="email" v-model="form.email" placeholder="Customer Email"` `  class="form-control" type="email" />` `   ` `  <input id="mobile" name="phone" v-model="form.mobile" placeholder="Customer Phone"` ` class="form-control" type="text" />	resources/js/Components/Customer/CustomerSaveForm.vue

```
                                <br />
                                <button type="submit" class="btn
w-100 btn-success">Save Change</button>
                            </div>
                        </form>
                    </div>
                </div>
            </div>
        </div>
    </div>
</template>

<script setup>
import { useForm, usePage, router, Link } from '@inertiajs/vue3'
import { createToaster } from "@meforma/vue-toaster";
const toaster = createToaster();
import { ref } from "vue";

const urlParams = new URLSearchParams(window.location.search)
let id = ref(parseInt(urlParams.get('id')))

const form = useForm({ name: '', email: '', mobile: '', id: id })
const page = usePage()

let URL = "/create-customer";
let list = page.props.customer

if (id.value !== 0 && list !== null) {
    URL = "/update-customer";
    form.name = list['name']
    form.id = list['id']
    form.email = list['email']
```

```
    form.mobile = list['mobile']
}

function submit() {
    form.post(URL, {
        onSuccess: () => {
            if (page.props.flash.status === true) {
                toaster.success(page.props.flash.message);
                setTimeout(() => {
                    router.get("/CustomerPage")
                }, 500)
            } else {
                toaster.warning(page.props.flash.message)
            }
        }
    })
}
</script>
```

Create

01	CreateCustomer Method	`public function CreateCustomer(Request $request){` `//  dd($request->all());` `$user_id = $request->header('id');` `$request->validate([` `'name' => 'required',` `'email' => 'required|email|unique:customers,email',` `'mobile' => 'required',` `]);`

		`` Customer::create(['name' => $request->input('name'), 'email' => $request->input('email'), 'mobile' => $request->input('mobile'), 'user_id' => $user_id]); // return response()->json([// 'status' => 'success', // 'message' => 'Customer created successfully' //]); $data = ['message'=>'Customer created successfully','status'=>true,'error'=>'']; return redirect('/CustomerPage')->with($data); } ```	
		Edit	
02	CustomerUpdate method	``` public function CustomerUpdate(Request $request){ $user_id = $request->header('id'); $id = $request->input('id'); Customer::where('id', $id)->where('user_id', $user_id)->update(['name' => $request->input('name'), 'email' => $request->input('email'), 'mobile' => $request->input('mobile'),]); // return response()->json([// 'status' => 'success', // 'message' => 'Customer Updaetd successfully' //]); ```	

		`$data = ['message'=>'Customer updated successfully','status'=>true,'error'=>''];` `return redirect('/CustomerPage')->with($data);` `}//end method`	
# Delete			
	Customer Delete method update	`public function CustomerDelete(Request $request,$id){` `$user_id = $request->header('id');` `Customer::where('user_id', $user_id)->where('id', $id)->delete();` `// return response()->json([` `//  'status' => 'success',` `//  'message' => 'Customer Deleted successfully'` `// ]);` `$data = ['message'=>'Customer Deleted successfully','status'=>true,'error'=>''];` `return redirect('/CustomerPage')->with($data);` `}`	
# Product			
## Product Read display			
01	web	`Route::get('/ProductPage', [ProductController::class, 'ProductPage'])->name('product.page');`	

02	ProductPage method	`public function ProductPage(Request $request){` `$user_id = $request->header('id');` `$products = Product::where('user_id', $user_id)` `->with('category')->latest()->get();` `return Inertia::render('ProductPage', ['products' => $products]);` `}`	app/Http/Controllers /ProductController.php
03	**ProductPage.vue**	`<template>` `<SideNavLayout>` `<ProductList/>` `</SideNavLayout>` `</template>` `<script setup>` `import SideNavLayout from '../Layout/SideNavLayout.vue';` `import ProductList from '../Components/Product/ProductList.vue';` `</script>`	resources/js/Pages /ProductPage.vue
04	**ProductList.vue**	`<template>` `<div class="container-fluid">` `<div class="row">` `<div class="col-12">` `<div class="card">` `<div class="card-body">` `<div>` `<h3>Product</h3>` `</div>` `<hr />` `<div class="float-end">`	resources/js/Components /Product/ProductList.vue

```
                            <Link href="/ProductSavePage?id=0"
class="btn btn-success mx-3 btn-sm">
                                Add Product
                            </Link>
                        </div>
                        <div>
                            <input placeholder="Search..."
class="form-control mb-2 w-auto form-control-sm" type="text"
                                v-model="searchValue">
                            <EasyDataTable
                                buttons-pagination
                                alternating
                                :headers="Header"
                                :items="Item"
                                :rows-per-page="10"
                                :search-field="searchField"
                                :search-value="searchValue" show-
index>
                                <template #item-image="{ image }"
class="pt-2 pb-2">
                                    <img :src="image" :alt="image"
alt="" height="40px" width="40px">
                                </template>
                                <template #item-action="{ id }">
                                    <Link class="btn btn-success
mx-3 btn-sm" :href="`/ProductSavePage?id=${id}`">Edit
                                    </Link>
                                    <button class="btn btn-danger
btn-sm" @click="DeleteClick(id)">Delete</button>
                                </template>
                            </EasyDataTable>
                        </div>
```

```
                    </div>
                </div>
            </div>
        </div>
    </div>
</template>

<script setup>
import { usePage, router, Link } from '@inertiajs/vue3'
import { createToaster } from "@meforma/vue-toaster";
const toaster = createToaster();
import { ref } from "vue";
let page = usePage()

const Header = [
    { text: "Image", value: "image" },
    { text: "Name", value: "name" },
    { text: "Category", value: "category.name" },
    { text: "Price", value: "price" },
    { text: "Quantity", value: "unit" },
    { text: "Action", value: "action" },
];

const Item = ref(page.props.products)
const searchValue = ref()


</script>
```

Product add

01	web	`Route::get('/ProductSavePage', [ProductController::class, 'ProductSavePage'])->name('ProductSavePage');`	
02	ProductSavePage Method	`public function ProductSavePage(Request $request){` `$user_id = $request->header('id');` `$product_id = $request->query('id');` `$categories = Category::where('user_id', $user_id)->get();` `return Inertia::render('ProductSavePage', [ 'categories' => $categories]);` `}`	app/Http/Controllers /ProductController.php
03	ProductSavePage.vue	`<template>` `<SideNavLayout>` `<ProductSaveForm/>` `</SideNavLayout>` `</template>` `<script setup>` `import SideNavLayout from "../Layout/SideNavLayout.vue";` `import ProductSaveForm from "../Components/Product/ProductSaveForm.vue";` `</script>`	resources/js/Pages /ProductSavePage.vue
04	ProductSaveForm.vue	`<template>` `<div class="container-fluid">` `<div class="row d-flex justify-content-center">` `<div class="col-md-6">` `<div class="card">` `<div class="card-body">` `<div class="float-end">` `<Link href="/ProductPage" class="btn btn-success mx-3 btn-sm">`	resources/js/Components /Product/ProductSaveForm.vue

```
                                Back
                            </Link>
                        </div>
                        <form @submit.prevent="submit"
enctype="multipart/form-data">
                            <div class="card-body">
                                <h4>Save Product</h4>
                                <input id="name" name="name" v-
model="form.name" placeholder="Product Name"
                                    class="form-control"
type="text" />
                                <br />
                                <input id="price" name="price" v-
model="form.price" placeholder="Product Price"
                                    class="form-control"
type="number" />
                                <br />
                                <input id="unit" name="unit" v-
model="form.unit" placeholder="Product Unit"
                                    class="form-control"
type="number" />
                                <br />
                                <!-- Category Dropdown -->
                                <div>
                                    <label for="category">Select
Category:</label>
                                    <select v-
model="form.category_id" class="form-control" id="category">
                                        <option value=""
disabled>Select a category</option>
```

```
                                    <option v-for="category in
categories" :key="category.id" :value="category.id">{{
category.name }}</option>
                                    </select>
                                </div>
                                <br />
                                <div>
                                    <label for="image">Product
Image:</label> <br>
                                    <!-- <input type="file"
id="image" @change="handleFileUpload" /> -->
                                    <ImageUpload
:productImage="form.image" @image="(e)=>form.image = e"/>
                                </div>
                                <br />
                                <button type="submit" class="btn
w-100 btn-success">Save Change</button>
                            </div>
                        </form>
                    </div>
                </div>
            </div>
        </div>
    </div>
</template>

<script setup>
import { useForm, usePage, router, Link } from '@inertiajs/vue3'
import { createToaster } from "@meforma/vue-toaster";
const toaster = createToaster();
import { ref } from "vue";
import ImageUpload from './ImageUpload.vue'
```

		`` const urlParams = new URLSearchParams(window.location.search) let id = ref(parseInt(urlParams.get('id'))) const form = useForm({ name: '', price: '', unit: '', category_id: '', image: null, id: id.value || null }) const page = usePage() const categories = ref(page.props.categories) let URL = "/create-product"; function submit() { form.post(URL, { onSuccess: () => { if (page.props.flash.status === true) { toaster.success(page.props.flash.message); setTimeout(() => { router.get("/ProductPage") }, 500) } else { toaster.warning(page.props.flash.message) } } }) } </script>```	
05	CreateProduct Method	``` public function CreateProduct(Request $request){ // dd($request->all());```	app/Http/Controllers /ProductController.php

```
         $user_id = $request->header('id');


        $request->validate([
            'name' => 'required',
            'price' => 'required',
            'unit' => 'required',
            'category_id' => 'required',
            'image' =>
'nullable|image|mimes:jpeg,png,jpg,gif,svg,webp|max:2048',
        ]);

        $data = [
            'name' => $request->name,
            'price' => $request->price,
            'unit' => $request->unit,
            'category_id' => $request->category_id,
            'user_id' => $user_id
        ];

        if($request->hasFile('image')){
            $image = $request->file('image');
            $fileName = time().'.'.$image-
>getClientOriginalExtension();
            $filePath = 'uploads/producs/'.$fileName;
            $image->move(public_path('uploads/producs'),
$fileName);
            $data['image'] = $filePath;
        }

        //  dd($data);
         Product::create($data);
```

		`` // return response()->json([// 'status' => 'success', // 'message' => 'Product created successfully' //]); $data = ['message'=>'Product created successfully','status'=>true,'error'=>'']; return redirect('/ProductPage')->with($data); } ```	
Product Delete			
01	ProductList.vue এর ভিতরে javascript tag এর মাঝখানে DeleteClick ফাংশন লিখতে হবে	``` const DeleteClick = (id) => { let text = "Do you want to delete"; if (confirm(text) === true) { router.get(`/delete-product/${id}`) toaster.success('Product Deleted Successfully'); } else { text = "You canceled!"; } } ```	resources/js/Components /Product/ProductList.vue
02	ProductDelete Method Modify	``` public function ProductDelete(Request $request,$id){ try{ // $user_id = $request->header('id'); $product = Product::findOrFail($id); if($product->image && file_exists(public_path($product->image))){ unlink(public_path($product->image)); } ```	app/Http/Controllers /ProductController.php

		`` $product->delete(); // return response()->json([// 'status' => 'success', // 'message' => 'Product Deleted successfully' //]); $data = ['message'=>'Product Deleted successfully','status'=>true,'error'=>'']; return redirect()->back()->with($data); }catch(Exception $e){ return response()->json(['status' => 'failed', 'message' => $e->getMessage()]); $data = ['message'=> 'Something went wrong','status'=>false,'error'=>$e->getMessage()]; return redirect()->back()->with($data); } } ```	
03	ProductList.vue এর ভিতরে Tamplate tag এর মাঝখানে DeleteClick ফাংশন লিখতে হবে	``` <template #item-action="{ id }"> <button class="btn btn-danger btn-sm" @click="DeleteClick(id)">Delete</button> </template> ```	resources/js/Components /Product/ProductList.vue

Product Edit

01	ProductList.vue এর ভিতরে Tamplate tag এর মাঝখানে DeleteClick ফাংশন লিখতে হবে	<template #item-action="{ id }"> <Link class="btn btn-success mx-3 btn-sm" :href="`/ProductSavePage?id=${id}`">Edit</Link> </template>	resources/js/Components /Product/ProductList.vue
02	ProductSavePage Method update	public function ProductSavePage(Request $request){ $user_id = $request->header('id'); $product_id = $request->query('id'); $product = Product::where('id',$product_id)->where('user_id',$user_id)->first(); $categories = Category::where('user_id', $user_id)->get(); return Inertia::render('ProductSavePage', ['product' => $product, 'categories' => $categories]); }	
03	ProductUpdate method modify	public function ProductUpdate(Request $request){ // dd($request->all()); $user_id = $request->header('id'); $request->validate(['name' => 'required', 'price' => 'required', 'unit' => 'required', 'category_id' => 'required']);	

```
        $product = Product::where('user_id', $user_id)->findOrFail($request->id);

      //  dd($product);

        $product->name = $request->name;
        $product->price = $request->price;
        $product->unit = $request->unit;
        $product->category_id = $request->category_id;

        if($request->hasFile('image')){
            if($product->image && file_exists(public_path($product->image))){
                unlink(public_path($product->image));
            }

            $request->validate([
                'image' => 'image|mimes:jpeg,png,jpg,gif,svg,webp|max:2048'
            ]);
            $image = $request->file('image');

            $fileName = time().'.'.$image->getClientOriginalExtension();
            $filePath = 'uploads/producs/'.$fileName;

            $image->move(public_path('uploads/producs'), $fileName);
            $product->image = $filePath;
        }

        // dd($product);
```

		<pre>$product->save(); // return response()->json([// 'status' => 'success', // 'message' => 'Product Updated successfully' //]); $data = ['message'=>'Product updated successfully','status'=>true,'error'=>'']; return redirect('/ProductPage')->with($data); }</pre>	
04	ProductSaveForm.vue এর ভিতরে javascript tag এর মাঝখানে লিখতে হবে	<pre>let list = page.props.product if (id.value !== 0 && list !== null) { URL = "/update-product"; form.name = list['name']; form.id = list['id']; form.price = list['price']; form.unit = list['unit']; form.category_id = list['category_id']; form.image = list['image']; }</pre>	
# SalesCreate			
01		Php artisan make:controller SaleController	cmd

02	web	`Route::get('/create-sale', [SaleController::class, 'SalePage'])->name('SalePage');`	
03	SalePage Method	`public function SalePage(Request $request){` `$user_id = $request->header('id');` `$customers = Customer::where('user_id', $user_id)->get();` `$products = Product::where('user_id', $user_id)->get();` `return Inertia::render('SalePage',['products'=> $products,'customers'=> $customers]);` `}`	app/Http/Controllers /SaleController.php
04	SalePage.vue	`<template>` `<SideNavLayout>` `<CreateSalePage/>` `</SideNavLayout>` `</template>` `<script setup>` `import CreateSalePage from '../Components/Sale/CreateSalePage.vue';` `import SideNavLayout from '../Layout/SideNavLayout.vue';` `</script>`	resources/js/Pages /SalePage.vue
05	CreateSalePage.vue	`<template>` `<div class="container-fluid">` `<div class="row">` `<!-- Billing Selection -->` `<div class="col-md-4 col-lg-4 p-2">`	resources/js/Components /Sale/CreateSalePage.vue

```
                <div class="card">
                    <div class="card-body">
                        <h4>Billed To</h4>
                        <div class="shadow-sm h-100 bg-white
rounded-3 p-3 mt-4">
                            <div class="row">
                                <div class="col-8">
                                    <span class="text-bold text-
dark">BILLED TO</span>
                                    <p class="text-xs mx-0 my-
1">Name: <span>{{ selectedCustomer?.name }}</span></p>
                                    <p class="text-xs mx-0 my-
1">Email: <span>{{ selectedCustomer?.email }}</span></p>
                                    <p class="text-xs mx-0 my-
1">User ID: <span>{{ selectedCustomer?.id }}</span></p>
                                </div>
                                <div class="col-4">
                                    <p class="text-bold mx-0 my-1
text-dark">Invoice</p>
                                    <p class="text-xs mx-0 my-
1">Date: {{ new Date().toISOString().slice(0,10) }}</p>
                                </div>
                            </div>
                            <hr class="mx-0 my-2 p-0 bg-
secondary"/>
                            <div class="row">
                                <div class="col-12">
                                    <table class="table w-100">
                                        <thead>
                                            <tr class="text-xs">
                                                <th>Name</th>
                                                <th>Qty</th>
```

```
                                        <th>Total</th>
                                        <th>Remove</th>
                                    </tr>
                                </thead>
                                <tbody>
                                    <tr v-
if="selectedProduct.length > 0"
                                        v-for="(product,
index) in selectedProduct" :key="index"
                                        class="text-
center">
                                        <td> {{
product.name }} </td>
                                        <td> {{
product.unit }}</td>
                                        <td> {{
product.price }}</td>
                                        <td>
                                            <button
@click="removeQty(product.id)" class="">-</button>
                                            <button
@click="addQty(product.id)" class="">+</button>
                                            <button
@click="removeProductFromSale(index)" class="btn btn-danger btn-
sm">Remove</button>
                                        </td>
                                    </tr>
                                </tbody>
                            </table>
                        </div>
                    </div>
```

```
                            <hr class="mx-0 my-2 p-0 bg-
secondary"/>
                            <div class="row">
                                <div class="col-12">
                                    <p class="text-bold text-xs
my-1 text-dark">Total: <i class="bi bi-currency-dollar"></i> {{
calculateTotal() }}</p>
                                    <p class="text-bold text-xs
my-1 text-dark">VAT (5%): <i class="bi bi-currency-dollar"></i> {{
vatAmount }}</p>
                                    <p>
                                        <button @click="applyVat"
class="btn btn-info btn-sm my-1 bg-gradient-primary w-40">Apply
VAT</button>
                                    </p>

                                    <p><button @click="removeVat"
class="btn btn-secondary btn-sm my-1 bg-gradient-primary w-
40">Remove VAT</button></p>

                                    <p><span class="text-
xxs">Discount Mode:</span></p>
                                    <select v-
model="usePercentageDiscount">
                                        <option
:value="false">Flat Discount</option>
                                        <option
:value="true">Percentage Discount</option>
                                    </select>
```

```
                                    <p class="text-bold text-xs
my-1 text-dark">Discount: <i class="bi bi-currency-dollar"></i> {{
discountAmount }} </p>

                                    <div v-
if="!usePercentageDiscount">
                                        <span class="text-
xxs">Flat Discount:</span>
                                        <input v-
model="flatDiscount" type="number" class="form-control w-40"
min="0" />
                                        <p>
                                            <button
@click="applyDiscount" class="btn btn-warning btn-sm my-1 bg-
gradient-primary w-40">Apply Flat Discount</button>
                                        </p>
                                    </div>

                                    <div v-else>
                                        <span class="text-
xxs">Discount (%):</span>
                                        <input v-
model="discountPercent" type="number" class="form-control w-40"
min="0" max="100" step="0.25" />
                                        <p><button
@click="applyDiscount" class="btn btn-warning btn-sm my-1 bg-
gradient-primary w-40" >Apply Percentage Discount</button></p>
                                    </div>

                                    <p><button
@click="removeDiscount" class="btn btn-secondary btn-sm my-1 bg-
gradient-primary w-40">Remove Discount</button></p>
```

```
                                    <hr class="mx-0 my-2 p-0 bg-
secondary"/>
                                    <p class="text-bold text-xs
my-1 text-dark">Payable: <i class="bi bi-currency-dollar"></i> {{
payable }}</p>
                                    <p><button
@click="createInvoice" class="btn btn-success btn-sm my-3 bg-
gradient-primary w-40">Confirm</button></p>
                                </div>
                            </div>
                        </div>
                    </div>
                </div>

                <!-- Product Selection -->
                <div class="col-md-4 col-lg-4 p-2">
                    <div class="card">
                        <div class="card-body">
                            <h4>Select Product</h4>
                            <input
                                placeholder="Search..."
                                class="form-control mb-2 w-auto form-
control-sm"

                                type="text"
                                v-model="searchProductValue"
                            />
                            <EasyDataTable
                                buttons-paginations
                                alternating
                                :items="ProductItem"
```

```
                            :headers="ProductHeader"
                            :rows-per-page="10"
                            :search-value="searchProductValue"
                            :seach-field="searchProductField"
                        >
                            <template #item-image="{image}">
                                <img :src="image" alt="Product
Image" height="auto" width="40px"/>
                            </template>

                            <template #item-action="{id,
image,name, price, unit}">
                                <button :class="['btn btn-sm',
unit > 0 ? 'btn-success' : 'btn-danger']"
                                        :disabled="unit <= 0"
                                        @click="addProductToSale(id,
image, name, price, unit)"
                                        >
                                        {{ unit > 0 ? 'Add' : 'Stock
Out' }}
                                </button>
                            </template>
                        </EasyDataTable>
                        <!-- <p>{{ selectedProduct }}</p> -->
                    </div>
                </div>
            </div>

            <!-- Customer Selection -->
            <div class="col-md-4 col-lg-4 p-2">
                <div class="card">
                    <div class="card-body">
```

```
                        <h4>Select Customer</h4>
                        <input
                            placeholder="Search..."
                            class="form-control mb-2 w-auto form-
control-sm"
                            type="text" v-
model="searchCustomerValue"
                        />
                        <EasyDataTable
                            buttons-pagination
                            alternating
                            :headers="CustomerHeader"
                            :items="CustomerItem"
                            :rows-per-page="10"
                            :search-field="searchCustomerField"
                            :search-value="searchCustomerValue"
                            show-index
                        >
                            <template #item-number="{ id, name,
email }">
                                <button
@click="addCustomerToSale({ id, name, email })" class="btn btn-
success btn-sm">Add</button>
                            </template>
                        </EasyDataTable>
                    </div>
                </div>
            </div>
        </div>
    </div>
</template>
```

```
<script setup>
import { useForm,usePage, router } from '@inertiajs/vue3'
import { createToaster } from "@meforma/vue-toaster";
import { watch } from "vue";
const toaster = createToaster();
import { ref } from "vue";

let page = usePage()

const selectedProduct = ref([]);
const selectedCustomer = ref(null);

//customer
const CustomerHeader = [
    { text: "Name", value: "name" },
    { text: "Pick", value: "number" },
];

const CustomerItem = ref(page.props.customers);
const addCustomerToSale = (Customer)=>{
    selectedCustomer.value = (Customer);
}
const searchCustomerField = ref('name');
const searchCustomerValue = ref();
//end customer

//product
const ProductHeader = [
    { text: "Image", value: "image" },
    { text: "Name", value: "name" },
    { text: "QTY", value: "unit" },
    { text: "Action", value: "action" },
```

```
];

const ProductItem = ref(page.props.products);

const searchProductField = ref('name');
const searchProductValue = ref();

const addProductToSale = (id, image, name, price, productUnit)=>{
    const existingProduct = selectedProduct.value.find(product =>
product.id === id);
    if(existingProduct){
        if(existingProduct.existingQty > 0){
            existingProduct.unit++;
            existingProduct.existingQty--;
        }else{
            toaster.warning(`Product ${name} is out of stock`);
        }
    }else{
        if(productUnit > 0){
            const product = {
                id: id,
                image: image,
                name: name,
                price: price,
                unit: 1,
                existingQty: productUnit-1
            };
            selectedProduct.value.push(product);
            calculateTotal();
        }else{
            toaster.warning(`Product ${name} is out of stock`);
        }
```

```
    }
};//end method

//end product

//Start Billed to
const addQty = (id)=>{
    const product = selectedProduct.value.find(product =>
product.id === id);
    if(product.existingQty > 0){
        product.unit++;
        product.existingQty--;
        calculateTotal();
    }else{
        toaster.warning(`Product ${product.name} is out of
stock`);
    }
};//end addQty

const removeQty = (id)=>{
    const product = selectedProduct.value.find(product =>
product.id === id);
    if(product.unit > 1){
        product.existingQty++;
        product.unit--;
        calculateTotal();
    }
};//end addQty

const removeProductFromSale = (index)=>{
    selectedProduct.value.splice(index,1);
    calculateTotal();
```

```
    calculatePayable();
    removeVat();
    removeDiscount();
    toaster.success('Product removed from sale');
};//end method

const vatRate = ref(5);
const flatDiscount = ref(0);
const discountPercent = ref(0);
const total = ref(0);
const vatAmount = ref(0);
const discountAmount = ref(0);
const usePercentageDiscount = ref(false);

const calculateTotal = () =>{
    return selectedProduct.value.reduce((sum, product) => sum +
product.price * product.unit, 0);
};

const applyVat = ()=>{
    vatAmount.value = (calculateTotal() * vatRate.value) / 100;
    calculateTotal();
}//end method

const removeVat = ()=>{
    vatAmount.value = 0;
    calculateTotal();
}//end method

const applyDiscount = () =>{
    if(usePercentageDiscount.value){
```

```
        discountAmount.value = (calculateTotal() *
discountPercent.value) / 100;
    }else{
        discountAmount.value = flatDiscount.value;
    }
    calculatePayable();
}//end method

const removeDiscount = ()=>{
    discountAmount.value = 0;
    flatDiscount.value = 0;
    discountPercent.value = 0;
    calculatePayable();
}//end method

const calculatePayable = () =>{
    const totalAmount = calculateTotal();
    payable.value = totalAmount + vatAmount.value -
discountAmount.value;
}//end method

watch(
  [selectedProduct, vatAmount, discountAmount],
  () => {
    calculatePayable();
  },
  { deep: true }
);

const payable = ref(0);

const form = useForm({
```

```
    customer_id: '',
    products: '',
    vat: '',
    discount: '',
    payable: calculateTotal(),
    total: '',
});

const createInvoice = ()=>{
    if(!selectedCustomer.value){
        toaster.warning('Please select a customer');
        return;
    }
    if(selectedProduct.value.length === 0){
        toaster.warning('Please select at least one product');
        return;
    }

    form.customer_id = selectedCustomer.value.id;
    form.products = selectedProduct.value;
    form.total = total.value
    form.vat = vatAmount.value;
    form.discount = discountAmount.value;
    form.payable = payable.value;

    const calculatedTotal = calculateTotal();
    form.total = calculatedTotal;
    form.payable = payable.value;

    form.post('/invoice-create',{
        onSuccess: ()=>{
            if(page.props.flash.status === true){
```

		`` toaster.success(page.props.flash.message); setTimeout(()=>{ router.get('/InvoiceListPage'); },500); }else{ toaster.warning(page.props.flash.message); } } }) }; //end Billed to </script> ```	
06	InvoiceCreate method Modify	``` public function InvoiceCreate(Request $request){ // dd($request->all()); DB::beginTransaction(); try { $user_id = $request->header('id'); $data = ['user_id' => $user_id, 'customer_id' => $request->customer_id, 'total' => $request->total, 'vat' => $request->vat, 'Payable' => $request->payable, 'discount' => $request->discount]; ```	app/Http/Controllers /InvoiceController.php

```
            $invoice = Invoice::create($data);

            $products = $request->input('products');

            foreach($products as $product){
                $existUnit = Product::where('id', $product['id'])-
>first();

                if(!$existUnit){
                    return response()->json([
                        'status' => 'failed',
                        'message' => "Product with ID
{$product['id']} not found"
                    ]);
                }

                if($existUnit->unit < $product['unit']){
                    return response()->json([
                        'status' => 'failed',
                        'message' => "Only {$existUnit->unit}
units available in stock for product id {$product['unit']}"
                    ]);
                }
                Invoice_product::create([
                    'invoice_id' => $invoice->id,
                    'product_id' => $product['id'],
                    'user_id' => $user_id,
                    'qty' => $product['unit'],
```

```
                    'sale_price' => $product['price']
                ]);
                Product::where('id', $product['id'])->update([
                    'unit' => $existUnit->unit - $product['unit']
                ]);
            }//end foreach

            DB::commit();
            // return response()->json([
            //     'status' => 'success',
            //     'message' => 'Invoice created successfully'
            // ]);

            $data = ['message'=>'Invoice created
successfully','status'=>true,'error'=>''];
            // return redirect('/InvoiceListPage')->with($data);


        }catch(Exception $e){
            DB::rollBack();
            return response()->json([
                'status' => 'failed',
                'message' => "Something went wrong"
            ]);

            $data = ['message'=>'Something went
wrong','status'=>false,'error'=>$e->getMessage()];
            return redirect()->back()->with($data);

        }
    }
```

InvoiceListPage

01	web	`Route::get('/InvoiceListPage', [InvoiceController::class, 'InvoiceListPage'])->name('InvoiceListPage');`	
02	InvoiceListPage	`public function InvoiceListPage(Request $request){` `$user_id = request()->header('id');` `$list = Invoice::where('user_id', $user_id)` `->with('customer','Invoice_product.product')->get();` `return Inertia::render('InvoiceListPage', ['list' => $list]);` `}`	app/Http/Controllers /InvoiceController.php
03	InvoiceListPage	`<template>` `<SideNavLayout>` `<InvoiceList/>` `</SideNavLayout>` `</template>` `<script setup>` `import SideNavLayout from '../Layout/SideNavLayout.vue';` `import InvoiceList from '../Components/Invoice/InvoiceList.vue';` `</script>`	resources/js/Pages /InvoiceListPage.vue
04	InvoiceList.vue	`<template>` `<div class="container-fluid">` `<div class="row">`	resources/js/Components /Invoice/InvoiceList.vue

```
            <div class="col-12">
                <div class="card">
                    <div class="card-body">
                        <div>
                            <h3>Invoice List</h3>
                        </div>
                        <hr />
                        <div>
                            <input placeholder="Search..."
class="form-control mb-2 w-auto form-control-sm"
                                type="text" v-
model="searchValue"/>
                            <EasyDataTable
                                buttons-pagination
                                alternating :headers="Header"
:items="Item"
                                :rows-per-page="10"
                                :search-field="searchField"
                                :search-value="searchValue">

                                <template #item-action="{ id }">
                                    <button
@click="showDetails(id)" class="viewBtn btn btn-outline-dark text-
sm px-3 py-1 btn-sm m-0">
                                        <i class="fa text-sm fa-
eye"></i>
                                    </button>
                                    <button
@click="DeleteClick(id)" style="margin-left: 5px;" class="btn btn-
danger btn-sm">Delete</button>
                                </template>
```

```
                        </EasyDataTable>
                    </div>
                </div>
            </div>
        </div>

        <!-- Modal component for invoice details -->
        <InvoiceDetails v-if="show" :customer="customer"
@close="closeModal"/>

    </div>
</template>
<script setup>
import { usePage, router } from '@inertiajs/vue3'
import { createToaster } from "@meforma/vue-toaster";
const toaster = createToaster();

import { ref } from "vue";
let page = usePage()

const show = ref(false)
const customer = ref()

const searchValue = ref()
const searchField = ref(['customer.name'])

const Header = [
    { text: "Name", value: "customer.name" },
    { text: "Customer Id", value: "customer.id" },
    { text: "Phone", value: "customer.mobile" },
    { text: "Total", value: "total" },
```

		`` { text: "Discount", value: "discount" }, { text: "Vat", value: "vat" }, { text: "Payable", value: "payable" }, { text: "Action", value: "action" },]; const Item = ref(page.props.list) </script> ```	
Invoce Delete			
01	এর ভিতরে javascript tag এর মাঝখানে DeleteClick ফাংশন লিখতে হবে	``` const DeleteClick = (id) => { let text = "Do you want to delete"; if (confirm(text) === true) { router.get(`/invoice-delete/${id}`) toaster.success('Invoice Deleted Successfully'); } else { text = "You canceled!"; } } ```	
02	InvoviceDelete Method modify	``` public function InvoiceDelete(Request $request, $id){ DB::beginTransaction(); try { $user_id = request()->header('id'); Invoice_product::where('invoice_id', $id) ->where('user_id', $user_id) ->delete(); ```	app/Http/Controllers /InvoiceController.php

```
            Invoice::where('id', $id)
                ->where('user_id', $user_id)
                ->delete();

            DB::commit();
            // return response()->json([
            //     'status' => 'success',
            //     'message' => 'Invoice deleted successfully'
            // ]);

             $data = ['message'=>'Invoice deleted
successfully','status'=>true,'error'=>''];
            return redirect()->back()->with($data);

        }catch(Exception $e){
            DB::rollBack();
            return response()->json([
                'status' => 'failed',
                'message' => "Something went wrong"
            ]);

            $data = ['message'=>'Something went
wrong','status'=>false,'error'=>$e->getMessage()];
            return redirect()->back()->with($data);

        }
    }
```

InvoiceDetails

	InvoiceDetails.vue	<template> <h5 class="modal-title">Invoice Details</h5> <button type="button" class="btn-close" @click="closeModal"></button> <!-- First Row: Billed To and Invoice Info --> <!-- Billed To Section --> BILLED TO Name: {{ customer.customer.name }} Email: {{ customer.customer.email }} Customer ID: {{ customer.customer.id }} Phone: {{ customer.customer.mobile }} <!-- Invoice Info Section --> <img class="w-40" src="../../Assets/img/logo.svg" alt="Company Logo"> Invoice	resources/js/Components /Invoice /InvoiceDetails.vue

```
                            <p class="mb-1">Date: {{ new
Date().toLocaleDateString() }}</p>
                        </div>
                    </div>

                    <hr class="my-2">

                    <!-- Second Row: Invoice Items -->
                    <div class="row">
                        <div class="col-12">
                            <table class="table table-striped">
                                <thead>
                                    <tr>
                                        <th scope="col">Product
Name</th>
                                        <th
scope="col">Quantity</th>
                                        <th scope="col">Sale
Price</th>
                                    </tr>
                                </thead>
                                <tbody>
                                    <tr v-for="(item, index) in
customer.invoice_product" :key="index">
                                        <td>{{ item.product.name
}}</td>
                                        <td>{{ item.qty }}</td>
                                        <td>{{ item.sale_price
}}</td>
                                    </tr>
                                </tbody>
                            </table>
```

```
                        </div>
                    </div>

                    <hr class="my-2">

                    <!-- Third Row: Totals -->
                    <div class="row">
                        <div class="col-12">
                            <p><strong>Total:</strong> {{
customer.total }}</p>
                            <p><strong>Payable:</strong>
{{customer.Payable}}</p>
                            <p><strong>VAT (5%):</strong> {{
customer.vat }}</p>
                            <p><strong>Discount:</strong> {{
customer.discount }}</p>
                        </div>
                    </div>
                </div>
            </div>

            <div class="modal-footer">
                <button class="btn btn-secondary"
@click="closeModal" style="margin-right: 5px;">Close</button>
                <button @click="printInvoice" class="btn btn-
primary">Print</button>
            </div>
        </div>
    </div>
</template>

<script setup>
```

```
import { defineProps, defineEmits } from 'vue'

const props = defineProps({
    customer: Object
})
const emit = defineEmits(['close'])

const closeModal = () =>{
    emit('close')
}

const printInvoice = () => {
    const printContent = document.querySelector('.modal-
body').innerHTML;
    const originalContent = document.body.innerHTML;
    document.body.innerHTML = printContent;
    window.print();
    document.body.innerHTML = originalContent;
    location.reload();
}
</script>

<style scoped>
@page{
    margin: 20mm;
    size:auto;
}
body::before,
body::after {
    display: none;
}
.modal-backdrop {
```

```
    position: fixed;
    top: 0;
    left: 0;
    width: 100%;
    height: 100%;
    background-color: rgba(0, 0, 0, 0.5);
    display: flex;
    justify-content: center;
    align-items: center;
    z-index: 1500;
    /* Bootstrap modal z-index */
}

.modal-content {
    background-color: white;
    padding: 20px;
    border-radius: 10px;
    width: 80%;
    /* Make the modal larger like Bootstrap's modal-lg */
    max-height: 90vh;
    /* Limit the height of the modal */
    overflow-y: auto;
    /* Enable scrolling inside the modal if content exceeds height
*/
}

.modal-body {
    max-height: 60vh;
    /* Limit the height of the modal body */
    overflow-y: auto;
    /* Enable scrolling inside the modal body */
}
```

		`` .w-40 { width: 40%; } .btn-close { background: none; border: none; font-size: 1.5rem; } </style> ```	
	InvoiceList.vue	``` <InvoiceDetails v-if="show" :customer="customer" @close="closeModal"/> <script setup> import InvoiceDetails from './InvoiceDetails.vue' const showDetails = (id) =>{ show.value = !show.value customer.value = Item.value.find(item => item.id === id) console.log(customer.value); } const closeModal = () => { show.value = false ```	resources/js/Components /Invoice/InvoiceList.vue

		`}` `</script>`	
		# profile	
	web	`Route::get('/ProfilePage', [UserController::class, 'ProfilePage']);`	
	ProfilePage method	`public function ProfilePage(Request $request){` `$email = request()->header('email');` `//$email = $request->session()->get('email');` `$user = User::where('email', $email)->first();` `return Inertia::render('ProfilePage',['user'=>$user]);` `}`	app/Http/Controllers /UserController.php
		`<template>` `<SideNavLayout>` `<ProfileForm></ProfileForm>` `</SideNavLayout>` `</template>` `<script setup>` `import ProfileForm from '../Components/Profile/ProfileForm.vue';` `import SideNavLayout from '../Layout/SideNavLayout.vue';` `</script>`	resources/js/Pages /ProfilePage.vue
		`<template>` `<div class="container-fluid">`	resources/js/Components /User

		 <form @submit.prevent="submit"> Profile Update <hr /> <label>Name</label> <input id="name" v-model="form.name" placeholder="First Name" class="form-control" type="text" /> <label>Email Address</label> <input id="email" disabled v-model="form.email" placeholder="User Email" class="form-control" type="email" /> <label>Mobile Number</label> <input id="mobile" v-model="form.mobile" placeholder="Mobile" class="form-control" type="mobile" /> 	/ProfileForm.vue

```
                                <div class="row m-0 p-0">
                                    <div class="col-md-4 p-2">
                                        <button type="submit"
class="btn mt-3 w-100  btn-success">Update</button>
                                    </div>
                                </div>
                            </div>
                        </form>
                    </div>
                </div>
            </div>
        </div>
</template>
<script setup>
import { useForm, usePage, router } from "@inertiajs/vue3";
import { createToaster } from "@meforma/vue-toaster";
const toaster = createToaster();

const form = useForm({ name: "", email: "", mobile: "" });
const page = usePage();

form.name = page.props.user.name;
form.email = page.props.user.email;
form.mobile = page.props.user.mobile;

function submit() {
    if (form.name.length === 0) {
        toaster.error("Name is required");
    } else if (form.email.length === 0) {
        toaster.error("Email is required");
    } else if (form.mobile.length === 0) {
```

		toaster.error("Mobile is required"); } else { form.post("/user-update", { onSuccess: () => { if (page.props.flash.status === true) { router.get('/ProfilePage'); toaster.success(page.props.flash.message); } else { toaster.error(page.props.flash.message); } } }); } } </script>	
	web	Route::post('/user-update', [UserController::class, 'UserUpdate']);	
	UserUpdate method	public function UserUpdate(Request $request){ $email = request()->header('email'); User::where('email', $email)->update(['name' => $request->input('name'), 'email'=> $request->input('email'), 'mobile'=> $request->input('mobile'),]); $data = ['message'=> 'Profile update successfully','status'=>true, 'error'=>''];	app/Http/Controllers /UserController.php

```
        return redirect()->back()->with($data);
    }
```
