Jurnal Teknologi Informasi dan Pendidikan
Volume 17, No. 2, September 2024
https://doi.org/10.24036/jtip.v17i2.916

P.ISSN: 2086 – 4981
E.ISSN: 2620 – 6390
tip.ppj.unp.ac.id

E-Catering Sales and Ordering Application in Web-Based Raden Catering Business Using the Prototype Method

Sultan Wahyu Ningrat[1*]✉**, Vitri Tundjungsari**[1]
[1]Informatics Engineering Study Program, Esa Unggul University, Jakarta, Indonesia
*Corresponding Author: sultanwahyuningrat@gmail.com

Article Information

Article history:

No. 916
Rec. September 08, 2024
Rev. December 18, 2024
Acc. December 19, 2024
Pub. January 03, 2025
Page. 493 – 513

Keywords:

- *Catering*
- *Prototype*
- *PHP*
- *Laravel*
- *MySQL*

ABSTRACT

The E-Catering application is a means that allows people to sell and buy their catering services. This application is designed to be used through smartphone devices, making it easier for users to order food. Raden Catering, which is located at Jalan Cideng Barat Dalam 12 No.1, RT 06/RW 10 Cideng Village, Gambir District, Central Jakarta, 10150, is an example of a catering business that has been operating for 20 years. Even though it has many customers, the catering ordering process so far is still carried out conventionally through direct communication or telephone, which can be less efficient compared to the use of the internet. Therefore, researchers developed this application so that people can order catering more easily and flexibly through an internet connection. The app consists of two parts, namely an app for customers and an app for admins. The results of the study show that application development can be done iteratively, allowing stakeholders to provide input throughout the development process. Customers can easily browse the menu, choose catering packages, and place orders without the need to visit the office in person or contact directly.

How to Cite:

Ningrat, S. W., & Tundjungsari, V. (2024). E-Catering Sales and Ordering Application in Web-Based Raden Catering Business Using the Prototype Method. Jurnal Teknologi Informasi Dan Pendidikan, 17(2), 493-513. https://doi.org/10.24036/jtip.v17i2.916

1. INTRODUCTION

The development of information technology is going very rapidly [1]. It is undeniable that all sectors, including the business world, take advantage of technological advances to make their work easier [2]. With the sophistication of technology, all the limitations of distance and time facilities have become a very easy problem. In the business world, both companies and other sectors are required to innovate in attracting buyers [3]. Because the reason is, in terms of ordering, the seller always experiences service constraints such as limited coverage of the order area and product promotion, which is quite difficult to get many buyers [4]. The business sector is increasingly triggered to use advanced technology as a tool or medium to survive and win the increasingly fierce competition [5]. With business competition in the same field, to be able to keep up with technological developments such as utilizing the internet to create a site that can serve online bookings [7].

Catering services are included in the *Commercial Catering* industry, namely the purpose and purpose of the company is to earn profits through *catering services* that aim to meet and satisfy consumer needs through the products (services) provided [8];[9]. Catering products, namely food, are a benchmark for consumer satisfaction that is adjusted to the habits and experiences of the consumers who enjoy the product [10].

The *catering* business is a popular business in the culinary or culinary world, on every occasion and moment of the event, we often encounter a variety of delicious food offered by *catering* entrepreneurs. *Catering* can be defined as one of the services in the field of ready-made food delivered directly to the place of booking at an event. Events such as weddings often require *catering* services, seminars, religious events, birthdays, etc. These activities often require the organizer to hire this *catering* service to prepare food as needed.

Raden *Catering* is a form of business in the field of catering sales. Raden *catering*'s business now has many transactions with consumers, but so far the process of delivering information is still from word of mouth and the order is still via phone which will be narrower than today's internet media. Although there are still few consumers compared to other *catering businesses* that already have a lot of customers because of the progress of managing the sales system. And according to the admission of the catering business owner, it is indeed because of the difficulty of the process of delivering information and in ordering *catering*, it feels like they do not get additional customers and profits. Consumers have to call and sometimes consumers also have to come to Raden *catering* themselves due to limited information. This method is certainly very ineffective and efficient for consumers who come from outside the Raden *catering location*.

In accordance with the description of the problem above, the caterer solution will be provided, namely by creating a web-based application that can help customers in terms of ordering catering in a structured manner from choosing menus, ordering, payment, transaction recording, to ordering reports for customers. As well as assisting the catering in terms of making bills through *a web-based* system, recording customer transactions in detail, recording customer payments, reminding *catering admins* if there are customers who have not paid off payments, to the process of making monthly transaction reports such as catering sales reports on Raden *Catering*.

2. RESEARCH METHOD

2.1. Research Object

The research object is the focus of the research area. Furthermore, from this object, the author will explore various literature studies, theories, data, and analyses of the research object to obtain results that are in accordance with the objectives of the research output [11]. So, this research was carried out in a catering food business called Raden Catering which is located on Jl. Cideng Barat Dalam 12 No.1, RT 06/RW 10 Cideng Village, Gambir District, Central Jakarta City. This business is engaged in the field of food sales, in order to conduct research as needed to complete the information requested.

2.2. Data Collection Methods

The following are the data collection techniques carried out in this study:

2.2.1. Interview

This interview or interview technique is conducted face-to-face through questions and answers between researchers or data collectors and resource persons or informants or data sources. The data collection technique through interviews is usually carried out as a preliminary study because this technique is impossible if the number of respondents is large.

2.2.2. Observation

The observation technique refers to the systematic observation and recording of the symptoms of the research subject. Classified as the simplest data collection technique, this type of observation is also widely used in statistical surveys, such as studying the attitudes and behaviors of a group of people. Using observation techniques, researchers often travel to locations to determine the right measuring instrument to use.

2.2.3. Documentation

The documentation method is one of the data collection methods used in social research methods. Basically, the documentation method is used to trace historical materials or historical data. Documents can be text, images, or monumental works of a person. Documentary research complements the observation and interview methods used in qualitative research. This method comes in the form of information that comes from important records of institutions or organizations or individuals. The use of this documentation method can strengthen and support the information obtained, judging from the results of observations and interviews.

3. RESULTS AND DISCUSSION

3.1. Research Results Data

3.1.1. Quick Design & Quick Plan

3.1.1.1. Designing Diagram Design

At this stage, the results obtained from the proposed business process are translated into the form of UML (*Unified Modeling Language*) [12], in this study 3 types of diagrams are used, including *Usecase Diagram*, *Activity Diagram*, and *Class Diagram*, The following are the results of the system design:

3.1.1.1.1 Usecase Diagram

A usecase diagram is a diagram that illustrates the relationship between actors and the system [13]. *A usecase diagram* can describe an interaction between one or more actors with the system to be created. *Usecase diagrams* can also be used to find out what functions exist in a system and can also present an actor's interaction with the system. The proposed *Usecase diagram* has 2 actors, namely admins, customers, among others:

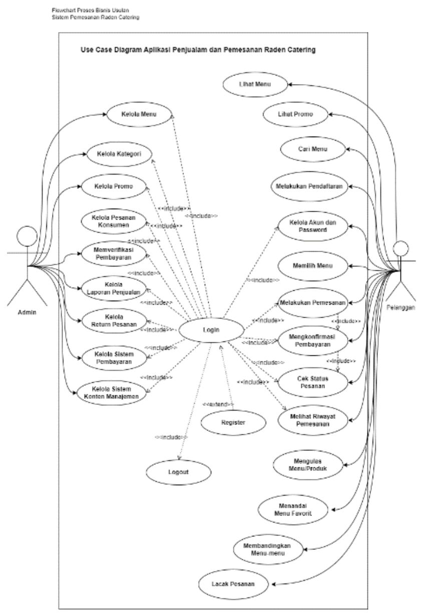


Figure 1. Diagram Use Case

3.1.1.1.2 Activity Diagram

Activity Diagram is the design of an activity flow or workflow in a system that will be executed [14]. *Activity Diagrams* are also used to define or group the display flow of the system. *An Activity Diagram* has a component with a specific shape that is connected by an arrow. The arrow points to the sequence of activities that occur from beginning to end. The proposed *Activity Diagram* includes:

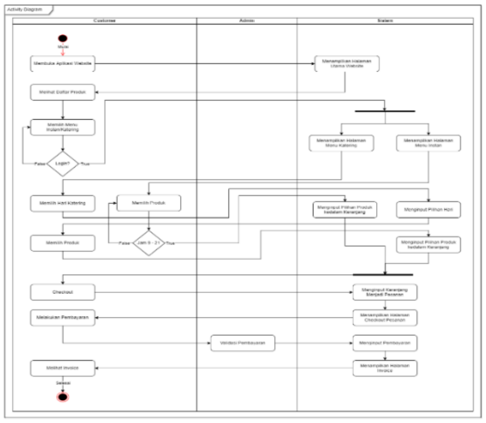


Figure 2. Diagram Activity

3.1.1.1.2.1 Activity Diagram Login

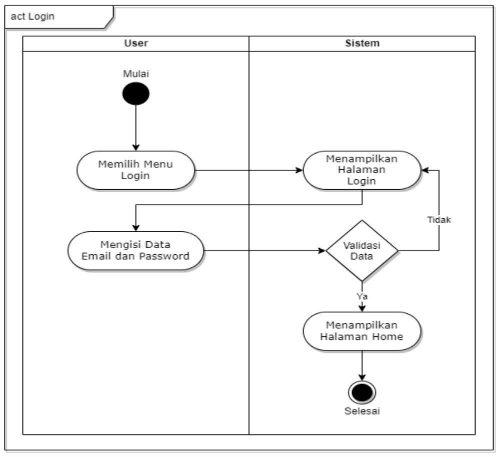


Figure 3. Activity Diagram Login

3.1.1.1.2.2. Activity Diagram Register

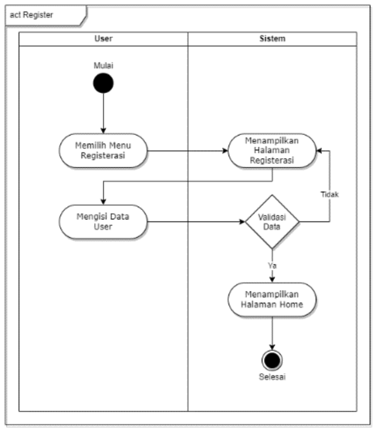


Figure 4. Activity Diagram Register

3.1.1.1.2.3. Activity Diagram Logout

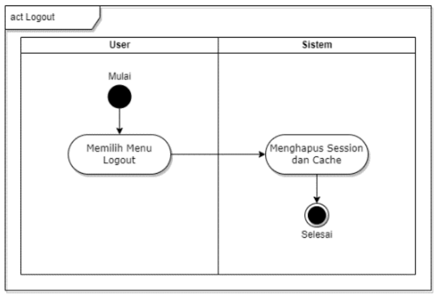


Figure 5. Activity Diagram Logout

3.1.1.1.2.4. Activity Diagram Profile

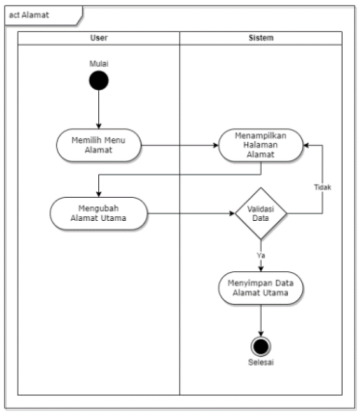


Figure 6. Activity Diagram Profile

3.1.1.1.2.5. Adress Activity Diagram

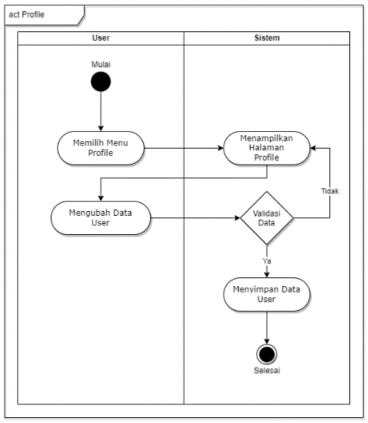


Figure 7. Adress Activity Diagram

3.1.1.1.3. Class Diagram

Class Diagram is a type of structure diagram in UML that clearly illustrates the structure and description of *classes*, *attributes*, methods, and relationships of each object [15]. It is static, in the sense that *the Class Diagram* does not explain what happens if the classes are related, but explains what relationships occur. This *Class Diagram* is appropriate if implemented to a project that uses the concept of *Object-Oriented Programming* (OOP) because the outline of the *Class Diagram* is quite easy to use. The model design of the *Class Diagram* itself is divided into two parts. The first part is an elaboration of the database. The second part is part of the MVC module, which has *a class interface*, *a control class*, and *a class entity*. The proposed *Class Diagram* includes:

3.1.1.1.3.1. Implementation and Coding Stage (*Build or Revise Prototype*)

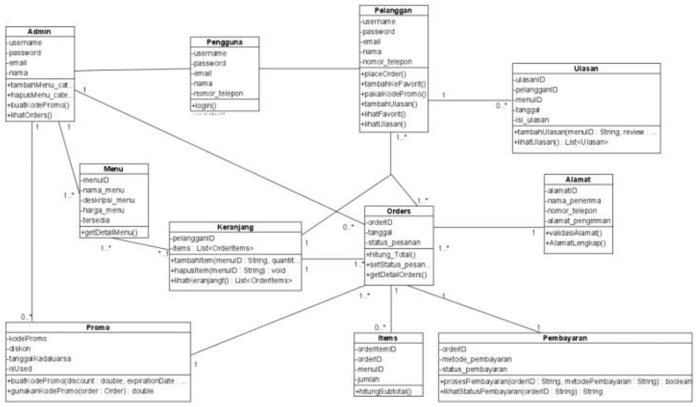


Figure 8. *Class Diagram* of Raden Catering Sales & Ordering System

3.1.1.2. Library Object Oriented (Package OOP)

The framework has a wide selection of *object-oriented libraries* that other frameworks don't, making it very useful for building complex applications [16]. Here are *the libraries* or *packages* used in application development:

3.1.1.2.1. Laravel *Permission*

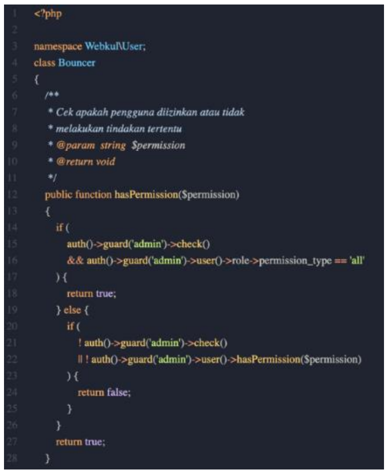
Laravel *Permission* is *a package* or package to create restrictions on user access rights on an application [16]. Laravel *Permissions* have 2 types of restrictions, namely: *Role* and *Permission*. *Role* can be interpreted as the level or position owned while *Permission* can be interpreted as a restriction of the user on each page. Users can be limited to their access rights individually on each page using *permissions* or can be grouped according to their *roles*. In this application, the restrictions used are *Role* and *Permission,* there are 2 roles that have been set, namely: *Admin*, and *Customer*, while for *the Permission* of each *Role*, namely:

3.1.1.2.2. Admins

Admins have all the access rights in the application.

3.1.1.2.3. Customers

Customers have access rights to go to the main page, order page, transaction history page, profile, and more



```php
<?php

namespace Webkul\User;
class Bouncer
{
   /**
    * Cek apakah pengguna diizinkan atau tidak
    * melakukan tindakan tertentu
    * @param  string  $permission
    * @return void
    */
   public function hasPermission($permission)
   {
      if (
         auth()->guard('admin')->check()
         && auth()->guard('admin')->user()->role->permission_type == 'all'
      ){
         return true;
      } else {
         if (
            ! auth()->guard('admin')->check()
            || ! auth()->guard('admin')->user()->hasPermission($permission)
         ){
            return false;
         }
      }
      return true;
   }
```

Figure 9. User Permisson Coding in Laravel

3.1.1.3. Tool Artisan

The Artisan tool has functions for Laravel's interaction with other frameworks with the help of *the command line*. The existence of artisans makes *it easy for developers* to carry out various activities across frameworks without fear of problems. Some of the artisan writing and its functions, namely:

3.1.1.3.1. php artisan serve

This tool serves to turn on and run the Laravel server so that the application can be accessed during the creation and development process, usually the default url provided for developers to access is http://127.0.0.1:8000. The url can be changed with an additional command, i.e.: php artisan serve --host=192.0.0.1 --port=8001. The -- host function is to change the main domain and --port to change the port, host and port are useful for opening multiple Laravel applications at the same time without fear of errors caused by url access that is being used by more than 1 application, using the same url but Different route lists at the same time can cause errors due to irregular data accumulation.

3.1.1.3.2. php artisan migrate

This tool functions to create a database through Laravel, in the *tables, fields*, and attributes in the database will be automatically created according to the design of the structure using *the code* inserted in the *migrations folder*. This command can be changed with additional commands, such as: *php artisan migrate:fresh --seed.* The *fresh* function retrieves all the data that has been populated in the *database* and *--seed* will call *the seeders folder* to send *the fields* in the table whose attributes have been adjusted following *the code* in the *migrations folder*.

3.1.1.3.3. php artisan make

This tool works to automatically create OOP or MVC files, here is an example of writing:

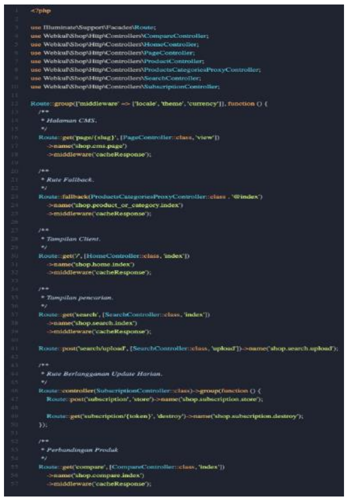
a. *PHP artisan make:controller* (filename), which serves to create *a controller file*.
b. *PHP artisan make:model* (filename), works to create *model files*.
c. *PHP artisan make:migration* (filename), works to create a *migration file*.
d. *PHP artisan make:seeder* (filename), which works to create a *seeder file*.
e. *php artisan route:list. This tool* functions to view and display a list of *urls* or *links* that can be accessed by application users.

3.1.1.4. MVC (*Model, View, Controller*)

The MVC architecture allows developers to create and develop applications more neatly in writing *scripts*, making it easier to document the process [17]. MVC consists of 3 folder structures, namely:

3.1.1.5. Route

Route, fuction to create paths that can be accessed with links, *script*:

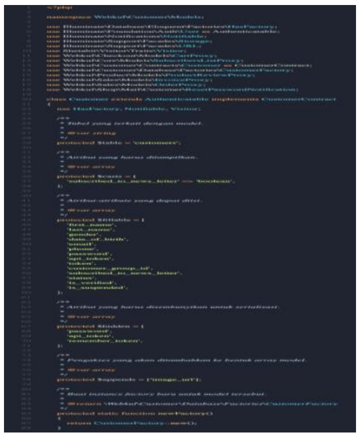


```php
<?php

use Illuminate\Support\Facades\Route;
use Webkul\Shop\Http\Controllers\CompareController;
use Webkul\Shop\Http\Controllers\HomeController;
use Webkul\Shop\Http\Controllers\PageController;
use Webkul\Shop\Http\Controllers\ProductController;
use Webkul\Shop\Http\Controllers\ProductsCategoriesProxyController;
use Webkul\Shop\Http\Controllers\SearchController;
use Webkul\Shop\Http\Controllers\SubscriptionController;

Route::group(['middleware' => ['locale', 'theme', 'currency']], function () {
    /**
     * Halaman CMS.
     */
    Route::get('page/{slug}', [PageController::class, 'view'])
        ->name('shop.cms.page')
        ->middleware('cacheResponse');

    /**
     * Rute Fallback.
     */
    Route::fallback(ProductsCategoriesProxyController::class . '@index')
        ->name('shop.product_or_category.index')
        ->middleware('cacheResponse');

    /**
     * Tampilan Client.
     */
    Route::get('/', [HomeController::class, 'index'])
        ->name('shop.home.index')
        ->middleware('cacheResponse');

    /**
     * Tampilan pencarian.
     */
    Route::get('search', [SearchController::class, 'index'])
        ->name('shop.search.index')
        ->middleware('cacheResponse');

    Route::post('search/upload', [SearchController::class, 'upload'])->name('shop.search.upload');

    /**
     * Rute Berlangganan Update Harian.
     */
    Route::controller(SubscriptionController::class)->group(function () {
        Route::post('subscription', 'store')->name('shop.subscription.store');

        Route::get('subscription/{token}', 'destroy')->name('shop.subscription.destroy');
    });

    /**
     * Perbandingan Produk
     */
    Route::get('compare', [CompareController::class, 'index'])
        ->name('shop.compare.index')
        ->middleware('cacheResponse');
```

Figure 10. Route Coding in Laravel

3.1.1.6. Model

Model, its function is to create logic in the form of a script that will be a link between the database and the application, script:

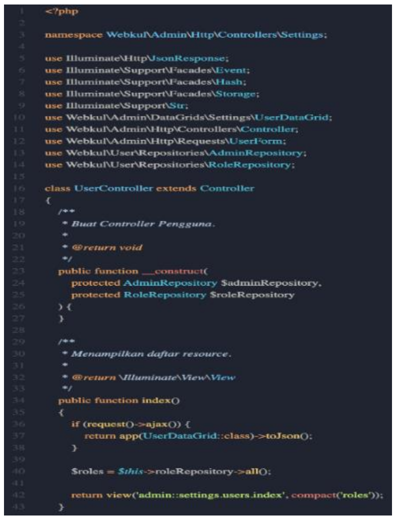


```php
<?php

namespace Webkul\Customer\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Foundation\Auth\User as Authenticatable;
use Illuminate\Notifications\Notifiable;
use Illuminate\Support\Facades\Storage;
use Illuminate\Support\Facades\URL;
use Shetabit\Visitor\Traits\Visitor;
use Webkul\Checkout\Models\CartProxy;
use Webkul\Core\Models\SubscribersListProxy;
use Webkul\Customer\Contracts\Customer as CustomerContract;
use Webkul\Customer\Database\Factories\CustomerFactory;
use Webkul\Product\Models\ProductReviewProxy;
use Webkul\Sales\Models\InvoiceProxy;
use Webkul\Sales\Models\OrderProxy;
use Webkul\Shop\Mail\Customer\ResetPasswordNotification;

class Customer extends Authenticatable implements CustomerContract
{
    use HasFactory, Notifiable, Visitor;

    /**
     * Tabel yang terkait dengan model.
     *
     * @var string
     */
    protected $table = 'customers';

    /**
     * Atribut yang harus [illegible].
     *
     * @var array
     */
    protected $casts = [
        'subscribed_to_news_letter' => 'boolean',
    ];

    /**
     * Atribut-atribut yang dapat diisi.
     *
     * @var array
     */
    protected $fillable = [
        'first_name',
        'last_name',
        'gender',
        'date_of_birth',
        'email',
        'phone',
        'password',
        'api_token',
        'token',
        'customer_group_id',
        'subscribed_to_news_letter',
        'status',
        'is_verified',
        'is_suspended',
    ];

    /**
     * Atribut yang harus disembunyikan untuk serialisasi.
     *
     * @var array
     */
    protected $hidden = [
        'password',
        'api_token',
        'remember_token',
    ];

    /**
     * Pengakses yang akan ditambahkan ke bentuk array model.
     *
     * @var array
     */
    protected $appends = ['image_url'];

    /**
     * Buat instance factory baru untuk model tersebut.
     *
     * @return \Webkul\Customer\Database\Factories\CustomerFactory
     */
    protected static function newFactory()
    {
        return CustomerFactory::new();
    }
```

Figure 11. Model Coding in Laravel

3.1.1.6. Controller

Controller, its function is to create logic and commands that will be the link between the data and the display, usually the data from the model will be processed first by the logic contained in the *controller* before the data is displayed and will set which display to display on the *browser*, *script*:

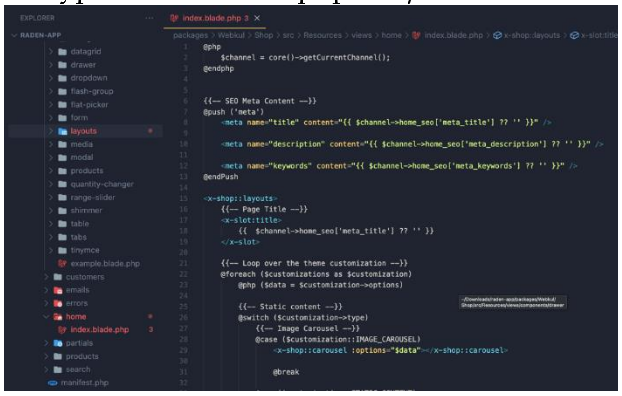


```php
<?php

namespace Webkul\Admin\Http\Controllers\Settings;

use Illuminate\Http\JsonResponse;
use Illuminate\Support\Facades\Event;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Storage;
use Illuminate\Support\Str;
use Webkul\Admin\DataGrids\Settings\UserDataGrid;
use Webkul\Admin\Http\Controllers\Controller;
use Webkul\Admin\Http\Requests\UserForm;
use Webkul\User\Repositories\AdminRepository;
use Webkul\User\Repositories\RoleRepository;

class UserController extends Controller
{
    /**
     * Buat Controller Pengguna.
     *
     * @return void
     */
    public function __construct(
        protected AdminRepository $adminRepository,
        protected RoleRepository $roleRepository
    ) {
    }

    /**
     * Menampilkan daftar resource.
     *
     * @return \Illuminate\View\View
     */
    public function index()
    {
        if (request()->ajax()) {
            return app(UserDataGrid::class)->toJson();
        }

        $roles = $this->roleRepository->all();

        return view('admin::settings.users.index', compact('roles'));
    }
```

Figure 12. Controller Coding in Laravel

3.1.1.7. Views

View, its function to create a design or view that will be seen by the application user, *View* in Laravel has a file type named .blade.php, *script*:



```php
@php
    $channel = core()->getCurrentChannel();
@endphp


{{-- SEO Meta Content --}}
@push ('meta')
    <meta name="title" content="{{ $channel->home_seo['meta_title'] ?? '' }}" />

    <meta name="description" content="{{ $channel->home_seo['meta_description'] ?? '' }}" />

    <meta name="keywords" content="{{ $channel->home_seo['meta_keywords'] ?? '' }}" />
@endPush

<x-shop::layouts>
    {{-- Page Title --}}
    <x-slot:title>
        {{  $channel->home_seo['meta_title'] ?? '' }}
    </x-slot>

    {{-- Loop over the theme customization --}}
    @foreach ($customizations as $customization)
        @php ($data = $customization->options)

        {{-- Static content --}}
        @switch ($customization->type)
            {{-- Image Carousel --}}
            @case ($customization::IMAGE_CAROUSEL)
                <x-shop::carousel :options="$data"></x-shop::carousel>

                @break
```

Figure 13. Encoding views in Laravel

3.1.1.8. Blade Template Engine

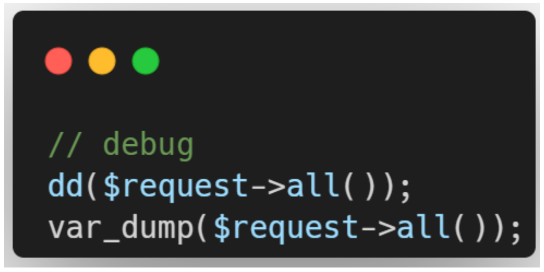
The Blade Template will make the application look lightweight and allow developers to create *powerful websites*. Where the available *template blades* can be added CSS, *images*, and text according to your needs [18]. *Blade Template* makes it easy for developers to set script structures in *folders* that don't have to be repeated many times, such as: *Meta, Heading, Footer, Sidebar, Title*, Css, Js, etc. *Blade* provides several features, such as:
1) @extends(), which serves to retrieve and use *the base script template*.
2) @yield() and @section(), @yield serves to empty and provide space that the existing HTML script can fill in different files, @section() functions to fill in the blank HTML *script* in @yield().
3) @include(), functions to retrieve the html script on the file in the Views folder.
4) @stack() and @push(), the function is the same as @yield() and @section(), the difference is that the content in @stack() and @push() is usually *a script* in the form of CSS and JS.
5) asset(), which functions to retrieve files that are in the *public folder*, usually used to capture images, videos, *scripts*, etc.

3.1.1.9. Routing

Laravel routing can be used to create structured and neatly organized applications with ease. Where all *requests* are mapped with the help of *routes*. Developers can group, name, apply *filters*, and retrieve data from models within the *route*. Some examples of Routing implementations in Laravel are:
1) Auth::routes(), works to use *authentication-related* routes.
2) Route::group(), which serves to group multiple *urls*.
3) Route::middleware(), which serves to group permissions on *the url*.
4) Route::p refix(), which serves to group and give names to *existing subdomains* in url.
5) Route::name(), which serves to group and give a name to the route list.
6) Route::controller(), works to group multiple urls that use *the same* controller file.
7) Route::get(), works to display logic, data, pages.
8) Route::p ost(), which serves to send data into the *database*.
9) Route::p ut() or Route::p atch(), works to modify and transmit data to the *database*.
10) Route::d elete(), which deletes the data that exists in the *database*.
11) Route::resource(), which allows you to group *get*(), post(), put(), and delete() routes into a single script line.

3.1.1.10. Testing and Debugging

Laravel is built with a fairly complete checking process feature [19]. Where it supports the checking process with PHP *Units* and phpunit.xml files that can be adapted to the application being built [20]. Laravel is built using convenient helper methods, which allow you to test your application expressively. Testing is usually carried out before entering the production stage or publishing the application on the *web*. There are many ways that can be used for *testing* and *debugging*, some example *scripts* to create them, namely: dd() and var_dump().



```php
// debug
dd($request->all());
var_dump($request->all());
```

Figure 14. Testing and Debugging Images

3.2. Trial Phase (*Testing Prototype*)

3.2.1. User Interface

3.2.1.1. Login

The Login *Page* is the process of entry for users to access a system. *Login* aims to regulate the process of identifying users who have personal data on a *website*. The *login process* consists of a username or *email* account and *a password* to get access rights. After logging into the system that has been designed on an application, we can monitor user activities on a *log* owned by an application or *web*, thus the user is also limited by role access rights that have been set by the system, so that each user has different access rights adjusted from their level level.

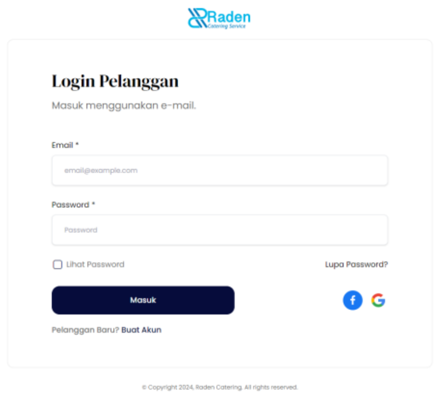

Login Pelanggan
Masuk menggunakan e-mail.
Email *
email@example.com
Password *
Password
Lihat Password
Lupa Password?
Masuk
Pelanggan Baru? Buat Akun
© Copyright 2024, Raden Catering. All rights reserved.

Figure 14. Customer Login Page

3.2.1.2. Register

The register form is used to add new *user* data, where the *user* data to be input is name, whatsapp number, *email* and *password* twice. Where for the user *id* and access rights column, we will create it automatically because *this register form* is intended for buyers or *customers*, so the access rights column will automatically contain the user and the system will validate the input data so that it cannot be filled in by the *user*. When creating the *form*, make sure that the maxlength must be the same length as what we created in the *database* and if the field has to be filled in don't forget to add *the require*.

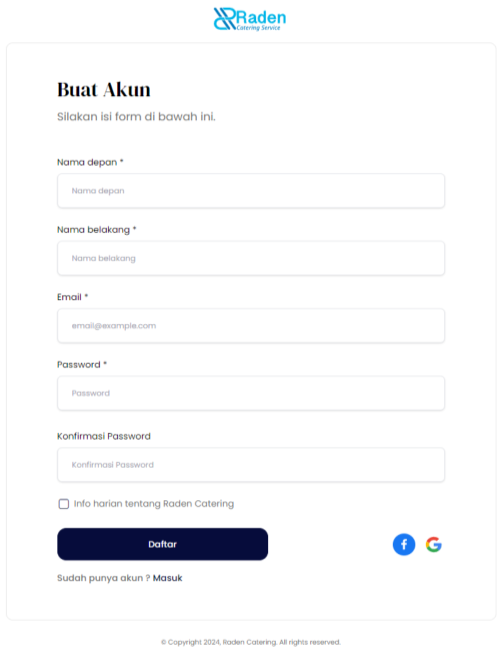

Buat Akun
Silakan isi form di bawah ini.
Nama depan *
Nama depan
Nama belakang *
Nama belakang
Email *
email@example.com
Password *
Password
Konfirmasi Password
Konfirmasi Password
Info harian tentang Raden Catering
Daftar
Sudah punya akun ? Masuk
© Copyright 2024, Raden Catering. All rights reserved.

Figure 15. Image Create Account

3.2.1.3. Index Page

The index page or the main page when you first access *the web*, this page contains a purchase menu for the *catering* packages offered, categories and contents of the menu.

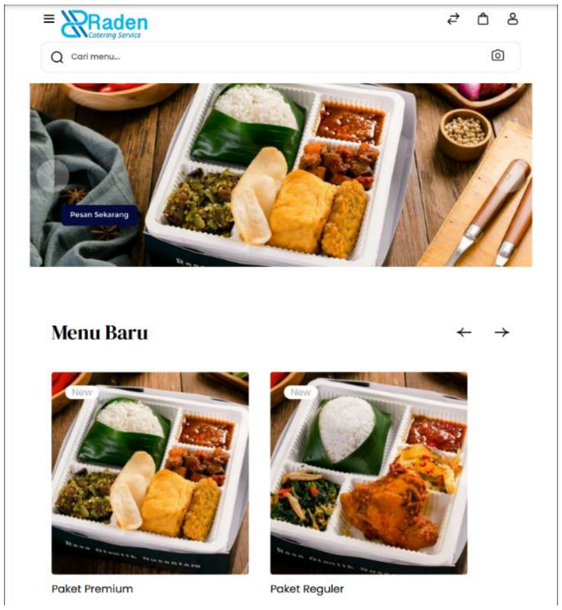
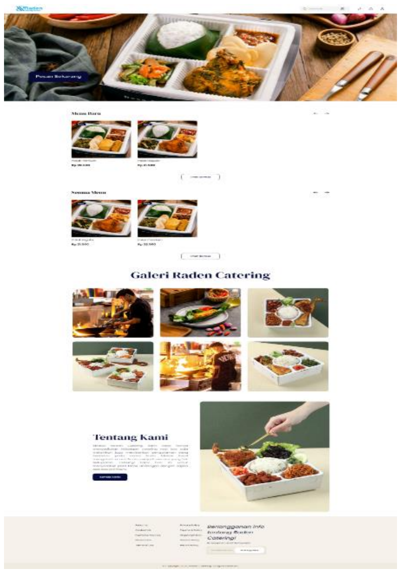


Figure 16. Index Page or Uatama Page

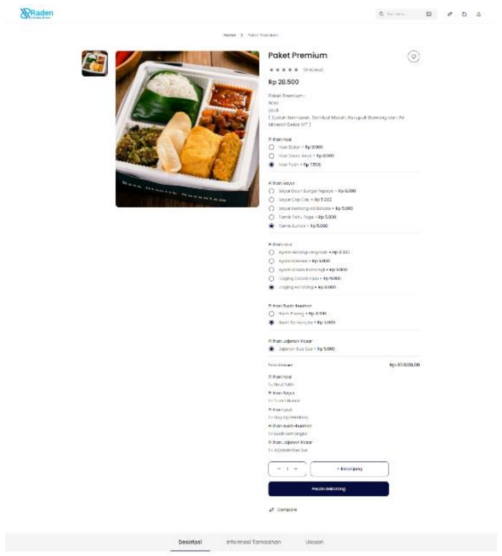


Gambar 17. Halaman Index atau Halaman Uatama

3.2.1.4. Catering Order

This page displays catering products or menus that can be ordered for *catering* needs.

Click on the message to add the product to the *catering cart*.


Paket Premium

★★★★★ (0 Ulasan)

Rp 28.500

Paket Premium :
Nasi
Lauk
(Sudah termasuk: Sambal Merah, Kerupuk Bawang dan Air Mineral Gelas VIT)

Pilihan Nasi
- ○ Nasi Bakar + Rp 9.000
- ○ Nasi Daun Jeruk + Rp 8.000
- ● Nasi Putih + Rp 7.500

Pilihan Sayur
- ○ Sayur Daun Bunga Pepaya + Rp 6.000
- ○ Sayur Cap Cay + Rp 5.000
- ○ Sayur Kentang Ati Balado + Rp 5.000
- ○ Tumis Tahu Toge + Rp 5.000
- ● Tumis Buncis + Rp 5.000

Pilihan Lauk
- ○ Ayam [illegible] + Rp 9.000
- ○ Ayam Kremes + Rp 9.000
- ○ Ayam [illegible] + Rp 9.000
- ○ Daging [illegible] + Rp 9.000
- ● Daging Rendang + Rp 9.000

Pilihan Buah-buahan
- ○ Buah Pisang + Rp 4.000
- ● Buah Semangka + Rp 5.000

Pilihan Jajanan Pasar
- ● Jajanan Kue Sus + Rp 5.000

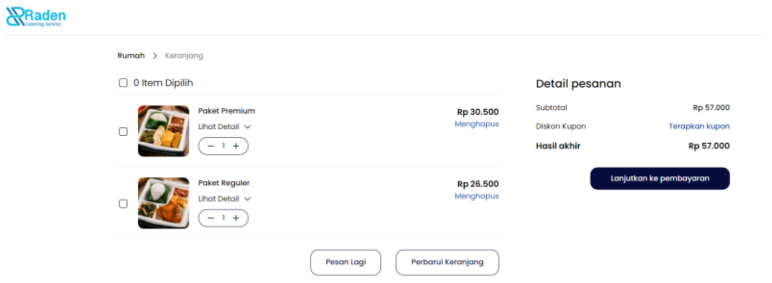
Total Amount Rp 30.500,00

Pilihan Nasi
1 x Nasi Putih
Pilihan Sayur
1 x Tumis Buncis
Pilihan Lauk
1 x Daging Rendang
Pilihan Buah-buahan
1 x Buah Semangka
Pilihan Jajanan Pasar
1 x Jajanan Kue Sus

Figure 18. Catering Order Page

3.2.1.5. Catering Cart

This page is a catering cart page that displays the selected products, selects the delivery date, completes the address and notes before paying for the order.


Paket Premium	Rp 30.500	Menghapus
Paket Reguler	Rp 26.500	Menghapus

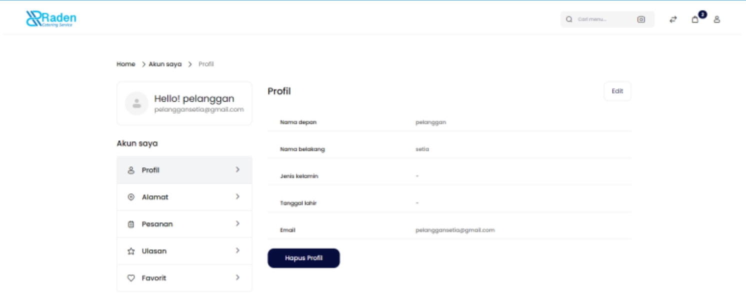
Detail pesanan

Subtotal	Rp 57.000
Diskon Kupon	Terapkan kupon
Hasil akhir	Rp 57.000

Figure 19. Halaman Catering Cart

3.2.1.6. Customer Profile Page

This page displays *the user's biodata, users can change* their biodata, profile photo, *and* password.

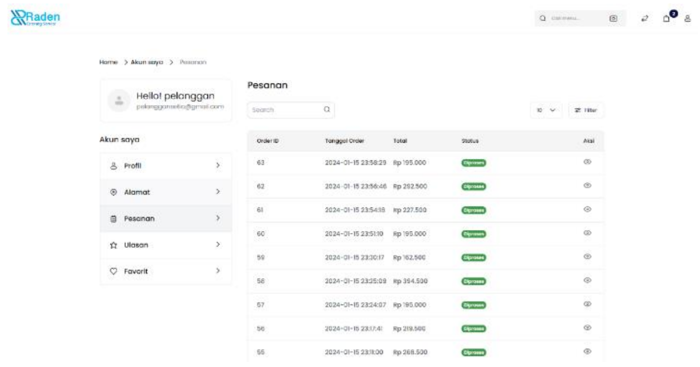


Nama depan	pelanggan
Nama belakang	setia
Jenis kelamin	-
Tanggal lahir	-
Email	pelanggansetia@gmail.com

Figure 20. Customer Profile Page

3.2.1.7. History Order Customer

This page displays the history of all transactions by category, users can view the history of catering and instant transactions.



Order ID	Tanggal Order	Total	Status	Aksi
63	2024-01-15 23:58:29	Rp 195.000	Diproses	
62	2024-01-15 23:56:46	Rp 292.500	Diproses	
61	2024-01-15 23:54:18	Rp 227.500	Diproses	
60	2024-01-15 23:51:10	Rp 195.000	Diproses	
59	2024-01-15 23:30:17	Rp 162.500	Diproses	
58	2024-01-15 23:25:09	Rp 354.500	Diproses	
57	2024-01-15 23:24:07	Rp 195.000	Diproses	
56	2024-01-15 23:17:41	Rp 219.500	Diproses	
55	2024-01-15 23:11:00	Rp 268.500	Diproses	

Figure 21. Customer Order History Page

3.2.1.8. Dashboard Admin

This page displays up-to-date data related to user data, transaction data, and order data status.

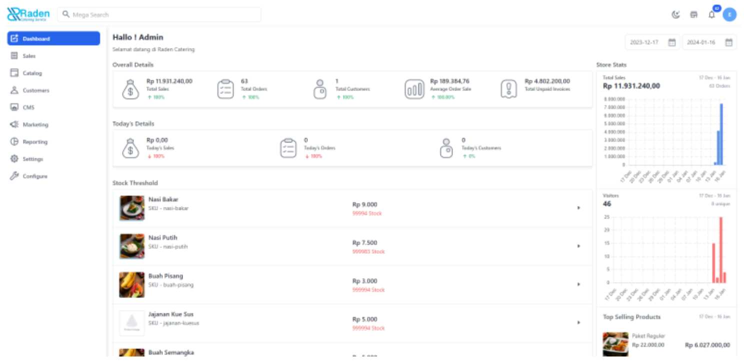
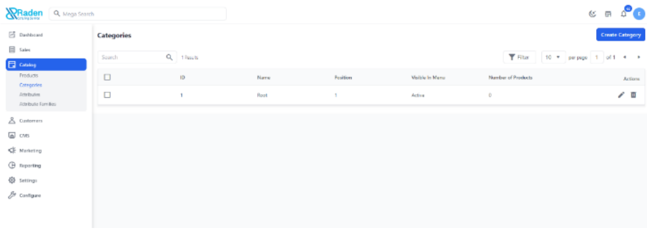

Raden
Mega Search
Dashboard
Sales
Catalog
Customers
CMS
Marketing
Reporting
Settings
Configure
Hallo ! Admin
Selamat datang di Raden Catering
2023-12-17
2024-01-16
Overall Details
Rp 11.931.240,00 Total Sales ↑ 100%
63 Total Orders ↑ 100%
1 Total Customers ↑ 100%
Rp 189.384,76 Average Order Sale ↑ 100.00%
Rp 4.802.200,00 Total Unpaid Invoices
Today's Details
Rp 0,00 Today's Sales ↓ 100%
0 Today's Orders ↓ 100%
0 Today's Customers ↑ 0%
Stock Threshold
Nasi Bakar SKU - nasi-bakar Rp 9.000 99994 Stock
Nasi Putih SKU - nasi-putih Rp 7.500 999983 Stock
Buah Pisang SKU - buah-pisang Rp 3.000 999994 Stock
Jajanan Kue Sus SKU - jajanan-kuesus Rp 5.000 999994 Stock
Buah Semangka
Store Stats
Total Sales 17 Dec - 16 Jan
Rp 11.931.240,00 63 Orders
Visitors 17 Dec - 16 Jan
46 8 unique
Top Selling Products 17 Dec - 16 Jan
Paket Reguler Rp 22.000,00 Rp 6.027.000,00

Figure 22. Admin Dashboard Page

3.2.1.9. Menu Category Admin

This page displays category data that can be added, changed and deleted.



Categories

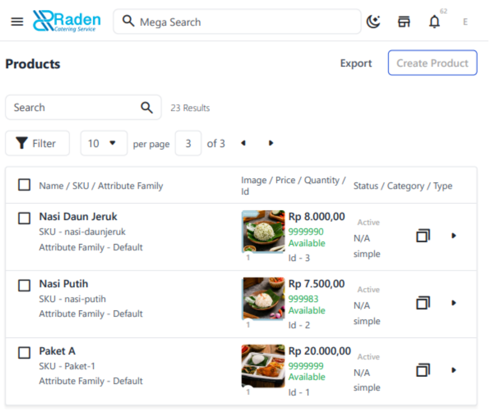
1 Results

	ID	Name	Position	Visible In Menu	Number of Products	Actions
☐	1	Root	1	Active	0	

Gambar 23. Menu Category Admin

3.2.1.10. Product Page in Admin

This page displays product data that can be added, changed and deleted.



Products

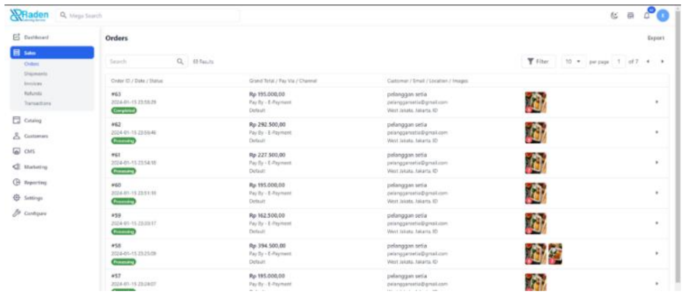
23 Results

	Name / SKU / Attribute Family	Image / Price / Quantity / Id	Status / Category / Type
☐	Nasi Daun Jeruk SKU - nasi-daunjeruk Attribute Family - Default	Rp 8.000,00 9999990 Available Id - 3	Active N/A simple
☐	Nasi Putih SKU - nasi-putih Attribute Family - Default	Rp 7.500,00 999983 Available Id - 2	Active N/A simple
☐	Paket A SKU - Paket-1 Attribute Family - Default	Rp 20.000,00 9999999 Available Id - 1	Active N/A simple

Figure 24. Product Admin Menu Page

3.2.1.11. Admin View of Catering Order List

This page displays data for all *catering* orders, data can be filtered to display monthly data, and data can be used as a report using *pdf* export.



Orders

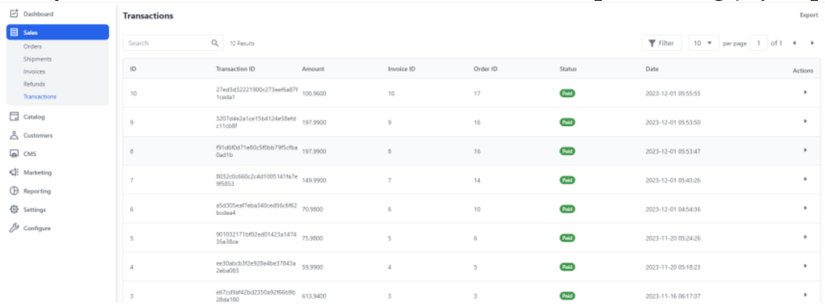
63 Results

Order ID / Date / Status	Grand Total / Pay Via / Channel	Customer / Email / Location / Images
#63 2024-01-15 23:58:29 Completed	Rp 195.000,00 Pay By - E-Payment Default	pelanggan setia pelanggansetia@gmail.com West Jakata, Jakarta, ID
#62 2024-01-15 23:59:46 Processing	Rp 292.500,00 Pay By - E-Payment Default	pelanggan setia pelanggansetia@gmail.com West Jakata, Jakarta, ID
#61 2024-01-15 23:54:18 Processing	Rp 227.500,00 Pay By - E-Payment Default	pelanggan setia pelanggansetia@gmail.com West Jakata, Jakarta, ID
#60 2024-01-15 23:51:18 Processing	Rp 195.000,00 Pay By - E-Payment Default	pelanggan setia pelanggansetia@gmail.com West Jakata, Jakarta, ID
#59 2024-01-15 23:33:17 Processing	Rp 162.500,00 Pay By - E-Payment Default	pelanggan setia pelanggansetia@gmail.com West Jakata, Jakarta, ID
#58 2024-01-15 23:25:09 Processing	Rp 394.500,00 Pay By - E-Payment Default	pelanggan setia pelanggansetia@gmail.com West Jakata, Jakarta, ID
#57 2024-01-15 23:24:07	Rp 195.000,00 Pay By - E-Payment	pelanggan setia pelanggansetia@gmail.com

Figure 25. Catering Order Admin Display Page

3.2.1.12. Admin Transaction Menu

This page displays the data of the entire transaction history, the data can be filtered to display monthly data, and the data can be made into a report using *pdf* export.



Transactions

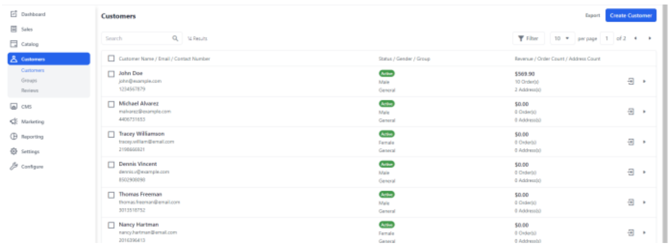
10 Results

ID	Transaction ID	Amount	Invoice ID	Order ID	Status	Date	Actions
10	27ed3d32221900c273eef6a87f1ceda1	100.9600	10	17	Paid	2023-12-01 05:55:55	▸
9	3207d4e2a1ce15b4124e58efdc11cb8f	197.9900	9	16	Paid	2023-12-01 05:53:50	▸
8	f91d6f0d71e80c5f0bb79f5cfba0ad1b	197.9900	8	16	Paid	2023-12-01 05:53:47	▸
7	8032c0c660c2c4d1005141fa7e9f5853	149.9900	7	14	Paid	2023-12-01 05:43:26	▸
6	a5d305eaf7eba340ced56c6f62bcdea4	70.9800	6	10	Paid	2023-12-01 04:54:36	▸
5	901032171bf02ed01423a147435e38ce	75.9800	5	6	Paid	2023-11-20 05:24:26	▸
4	ee30abcb3f2e928e4be37843a2eba083	59.9900	4	5	Paid	2023-11-20 05:18:23	▸
3	e67cd9af42bd2350a92f66b9b28da180	613.9400	3	3	Paid	2023-11-16 06:17:37	▸

Figure 26. Admin Transaction Menu

3.2.1.13. Menu Users Admin

This page displays the data of the user's account on the *web*.



Customers

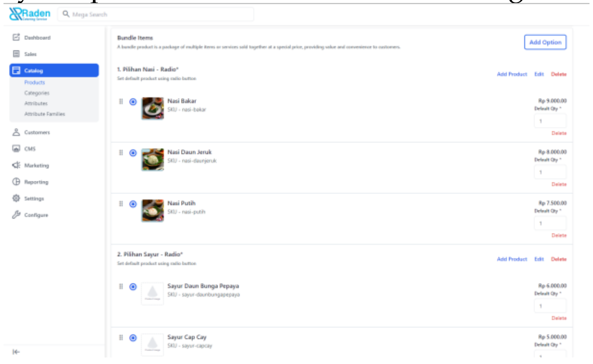
14 Results

Customer Name / Email / Contact Number	Status / Gender / Group	Revenue / Order Count / Address Count
John Doe john@example.com 1234567879	Active Male General	$569.90 10 Order(s) 2 Address(s)
Michael Alvarez malvarez@example.com 4406731653	Active Male General	$0.00 0 Order(s) 0 Address(s)
Tracey Williamson tracey.william@email.com 2198666821	Active Female General	$0.00 0 Order(s) 0 Address(s)
Dennis Vincent dennis.v@example.com 8502908098	Active Male General	$0.00 0 Order(s) 0 Address(s)
Thomas Freeman thomas.freeman@email.com 3013518752	Active Male General	$0.00 0 Order(s) 0 Address(s)
Nancy Hartman nancy.hartman@email.com 2016396413	Active Female General	$0.00 0 Order(s) 0 Address(s)

Figure 27. Admin Users Menu

3.2.1.14. Menu Ingredient

This page displays staple data that can be added, changed and deleted.



Bundle Items

A bundle product is a package of multiple items or services sold together at a special price, providing value and convenience to customers.

1. Pilihan Nasi - Radio*
Set default product using radio button

Product	SKU	Price	Default Qty *
Nasi Bakar	SKU - nasi-bakar	Rp 9.000,00	1
Nasi Daun Jeruk	SKU - nasi-daunjeruk	Rp 8.000,00	1
Nasi Putih	SKU - nasi-putih	Rp 7.500,00	1

2. Pilihan Sayur - Radio*
Set default product using radio button

Product	SKU	Price	Default Qty *
Sayur Daun Bunga Pepaya	SKU - sayur-daunbungapepaya	Rp 6.000,00	1
Sayur Cap Cay	SKU - sayur-capcay	Rp 5.000,00	

Figure 28. Menu Ingredients

4. CONCLUSION

The web-based sales and ordering application of Raden Catering business with the *prototype* method offers a number of significant benefits. By using *a prototype* approach, application development can be carried out iteratively, allowing stakeholders to provide input throughout the development process. Through the *prototype* phases, customer needs and preferences can be accommodated effectively, increasing customer satisfaction. This application provides convenience in the process of ordering and selling *e-catering* services. Customers can easily browse the menu, choose *catering* packages, and place orders without the need to visit the office in person or contact directly. This not only speeds up the process, but also improves customer convenience, providing a positive experience. Additionally, the order tracking feature and online payment integration provide transparency and administrative ease for both customers and service providers. This system can assist Raden Catering in managing orders efficiently, improving operational effectiveness, and reducing potential errors in administrative processes.

REFERENCES

[1] Al Ghani, R., Azani, N. W., Auliani, S. N., Maharani, S., Gustinov, M. D., & Hamzah, M. L. (2022, June). Perancangan Sistem Informasi e-Commerce Berbasis Website Menggunakan Metode Waterfall. In Prosiding Seminar Nasional Teknologi Informasi dan Bisnis (pp. 99-106).

[2] Alam, S., & Ika, N. (2021). Perancangan Aplikasi E-Catering Pada Usaha Rabila Catering Berbasis Web Menggunakan Notifikasi Whatsapp Gateway. Jurnal Sintaks Logika, 1(2), 123-131.

[3] Arrahman, A. A. (2019). Sistem Informasi Jasa Catering Berbasis Web Pada Ud. Berkah (Doctoral dissertation, Universitas Komputer Indonesia).

[4] Asmoko, H. (2013). Teknik Ilustrasi Masalah-Fishbone Diagrams. *Magelang Badan Pendidik Dan Pelatih Keuang Dep Keuang*.

[5] Baso, K. J., Rindengan, Y. D., & Sengkey, R. (2020). Perancangan Aplikasi Catering Berbasis Mobile. Jurnal Teknik Elektro Dan Komputer, 9(2), 81-90.

[6] Carland, J. W., Hoy, F., Boulton, W. R., & Carland, J. A. C. (1984). Differentiating Entrepreneurs from Small Business Owners: A Conceptualization. *The Academy of Management Review*, 9(2), 354. https://doi.org/10.2307/258448

[7] Febiharsa, D., Sudana, I. M., & Hudallah, N. (2018). Uji fungsionalitas (blackbox testing) sistem informasi lembaga sertifikasi profesi (silsp) batik dengan appperfect web test dan uji pengguna. *Joined Journal (Journal of Informatics Education)*, *1*(2), 117-126.

[8] Firliana, R., Amna, A. R., & Prastyo, A. (2016). Sistem informasi pemesanan catering berbasis web. Nusantara of Engineering (NOE), 3(2), 43-51.

[9] Heizer, J., and Render, B. (2015). Manajemen Operasi: Manajemen Keberlangsungan dan Rantai Pasokan, edisi 11. Jakarta: Salemba Empat.

[10] Kardigantara, Suseno. 2006. Diktat :Operasional Katering. Bandung : STPD

[11] Pressman, R. S. (2012). Software-Engineering 7th ED by Roger S. Pressman. In Software Engineering A Practitioner's Approach.

[12] Rohmalia, P. A., & Djajalaksana, Y. M. (2013). Pengelolaan Bisnis Catering dengan Memanfaatkan Sistem Informasi Berbasis Web (Studi Kasus pada Anggun Catering). Jurnal Teknik Informatika Dan Sistem Informasi, 8(2), 219812.

[13] Rosyada, A. F., Wicaksono, D., Nugroho, D., & Tanjung, T. (2022). Perancangan Website Pada Cathering Carcinos Kitchen. OKTAL: Jurnal Ilmu Komputer dan Sains, 1(08), 1269-1273.

[14] Septian, R. D. (2021). Perancangan Jasa Catering Dengan Memanfaatkan Sistem Informasi Berbasis Website (Studi Kasus: Kebayoran Lama, Jakarta Selatan). JUSIBI (Jurnal Sistem Informasi dan E-Bisnis), 2(4), 466-478.

[15] Siswidiyanto, S., Wijayanti, D., & Haryadi, E. (2020). Sistem Informasi Penyewaan Rumah Kontrakan Berbasis Web Dengan Menggunakan Metode Prototype. *Jurnal Interkom: Jurnal Publikasi Ilmiah Bidang Teknologi Informasi dan Komunikasi*, *15*(1), 16-23.

[16] Supriyanta, I. M. (2019). Perancangan Sistem Informasi Jasa Katering Berbasis Website.

[17] Tiara, A. (2020). Pengelolaan Konten pada Media Promosi E-Catering Marketplace bagi Industri Bisnis Jasa Boga Usaha Kecil Menengah. *Komunika : Jurnal Ilmu Komunikasi, 07(02), 85-90.* https://doi.org/10.22236/komunika.v7i2.6330.

[18] The Korean Society of Culture and Convergence, G. Lee, S. Huh, H. Chee, M. Kim, and M. Kim, "Research on the Development of Teaching and Learning Support Models for the Utilization of AI-based Learning Assistance Systems in Universitie," Korean Soc Cult Converg, vol. 45, no. 9, pp. 181–191, Sep. 2023, doi: 10.33645/cnc.2023.09.45.09.181.

[19] H. Rahmayanti, V. Oktaviani, and Y. Syani, "Development of sorting waste game android based for early childhood in environmental education," J. Phys.: Conf. Ser., vol. 1434, no. 1, p. 012029, Jan. 2020, doi: 10.1088/1742-6596/1434/1/012029.

[20] A. Ümmühan and H. ERSOY, "The adaptation of learning motivation in computer programming courses scale into Turkish: the study of validity and reliability," Journal of Higher Education and Science, vol. 8, no. 1, pp. 073–081, 2018.