4. IMPLEMENTASI SISTEM

4.1 *Scrap* Data Kostum

Segmen program 4.1 merupakan alur kode untuk melakukan *scraping* data kostum. Hal pertama adalah memasukkan tautan pertama untuk awal *scraping*. Di bawah ini adalah segmen untuk melakukan *scraping*. Proses ini dilakukan dalam setiap bagian program untuk *scraping* data kostum, kategori, dan juga tautan gambar.

Tabel 4.1 Tabel Segmen Program dan *Flowchart Scrap* Data Kostum

Segmen Program	*Flowchart*
Segmen Program 4.1	Gambar 3.2, Gambar 3.3, dan Gambar 3.4

Segmen Program 4.1 Program *Scrap*

```
start_page = 1
end_page = 169


for page_num in range(start_page, end_page + 1):
    current_url = start_url.format(page_num)
    print("Scraping:", current_url)
    # Create a Request object with headers
    request = Request(current_url, headers={"user-agent": useragent})
    # Open the URL with urlopen
    response = urlopen(request)
    # Read the HTML content directly from the response
    html_content = response.read()
    soup = bs4.BeautifulSoup(html_content, 'html.parser')
    # Extract data from the current page
    costume_data = extract_data_from_page(soup)
    # Append the extracted data to the main list
```

Setelah program melakukan scraping, maka selanjutnya adalah melakukan pemisahan atau proses *Data Cleaning.* Program ini juga dilakukan di banyak proses lainnya, yaitu proses *scraping* data kostum, kategori, dan tautan gambar. Segmen program dapat dilihat pada Segmen Program 4.2 berikut.

Tabel 4.2 Tabel Segmen Program dan *Flowchart Cleaning Scrap* Data Kostum

Segmen Program	*Flowchart*
Segmen Program 4.2	Gambar 3.1

Segmen Program 4.2 Proses *Cleaning Data*

```
def extract_data_from_page(soup):
    name_h4 = soup.find_all('h4', class_='post_title entry-title')
    excluded_names = [
        "SEWA", "logo", "VOB", "CELEBRATE", "Gudang Kostum Steril", "UV", "Vitamin", "Topi",
        "Boot", "Visor", "Sarung", "masker", "KN95", "kacamata", "hazmat", "Face",
        "Desinfektan", "Photostudio", "photo studio", "TUMB", "Peraturan", "Slide", "home",
        "video", "koleksi", "shower", "minuman", "cover"]
    excluded_urls = [
        "https://gudangkostum.com/home-1/attachment/tumb-prince/",
        "https://gudangkostum.com/home-1/attachment/tumb-animal/",
        "https://gudangkostum.com/home-1/attachment/tumb-inter/",
        "https://gudangkostum.com/home-1/attachment/tumb-princess/",
        "https://gudangkostum.com/home-1/attachment/tumb-hero/",
        "https://gudangkostum.com/bridal-station/" # Add your excluded URLs here]
    costume_data = []
    for h4_tag in name_h4:
        h4_tag_a = h4_tag.find('a')
        if h4_tag_a:
            costume_name = h4_tag_a.get_text(strip=True)
            costume_url = h4_tag_a.get('href')
            # Apply regex to clean the name
            cleaned_name = re.sub(r'\s{2,}', ' ', costume_name.strip())
            # Check if the name contains only one word
            if cleaned_name == 1:
                continue
            if re.search(r'\bbridal\.?station\b', cleaned_name, flags=re.IGNORECASE):
                category = "#gk_bridalstation"
```

Pada setiap program akan memiliki program *cleaning data* yang sama, namun untuk setiap program akan memiliki perbedaan dalam bentuk penyimpanan ke dalam *array*. Hal ini dikarenakan setiap program akan menyimpan data yang berbeda.

Tabel 4.3 Tabel Segmen Program dan *Flowchart* Penyimpanan Data *Scrap*

Segmen Program	*Flowchart*
Segmen Program 4.3	Gambar 3.1, Gambar 3.2

Segmen Program 4.3 Proses Penyimpanan Data Kostum

```
# Check if the name contains any excluded words (case-insensitive)
        if not any(exclude_word.lower() in cleaned_name.lower() for
        exclude_word in excluded_names) and \not any(excluded_url in
        costume_url for excluded_url in excluded_urls):
          data = {
            "name": costume_name,
            "link" : costume_url,
            "size": [],
            "desc": "",
            "price": 0
          }
          costume_data.append(data)
    return costume_data
```

Untuk *scrap* data kategori akan ditambahkan sebuah proses khusus untuk memisahkan antara deskripsi dan kategori.

Tabel 4.4 Tabel Segmen Program dan *Flowchart* Pemisahan Kategori

Segmen Program	*Flowchart*
Segmen Program 4.4	Gambar 3.1, Gambar 3.3

Segmen Program 4.4 Fungsi Pemisahan Kategori dari Deskripsi

```
def extract_category_from_description(description):
    # Use regular expression to find the category
    bridal_regex = r'\bbridal\.?station\b'
    category_regex = r'#gk_(\w+)'
    if description:
        # Check for bridal station category
        bridal_match = re.search(bridal_regex, description, flags=re.IGNORECASE)
        if bridal_match:
            return [{"cat": "bridalstation"}]
        # Check for other categories marked with hashtags
        categories = re.findall(category_regex, description)
        return [{"cat": category} for category in categories] if categories else None
    else:
        return None
```

Berikutnya setelah dipisahkan antara deskripsi dengan kategori, maka program yang dijalankan selanjutnya adalah menyimpan ke dalam sebuah *array*.

Tabel 4.5 Tabel Segmen Program dan *Flowchart* Penyimpanan Data Kategori

Segmen Program	*Flowchart*
Segmen Program 4.5	Gambar 3.1, Gambar 3.3

Segmen Program 4.5 Proses Penyimpanan Data Kategori

```
# Check if the name contains any excluded words (case-insensitive)
        if not any(exclude_word.lower() in cleaned_name.lower() for
exclude_word in excluded_names) and \
           not any(excluded_url in costume_url for excluded_url in excluded_urls):
            if p_div:
                for description in p_div[0].find_all('p'):
                    category =
extract_category_from_description(description.get_text(strip=True))
                    if category:
                        data = {
                            "name": costume_name,
                            "desc": category,
                        }
                        category_data.append(data)
    return category_data
```

Tabel 4.6 Tabel Segmen Program dan *Flowchart* Penyimpanan Tautan Gambar

Segmen Program	*Flowchart*
Segmen Program 4.6	Gambar 3.1, Gambar 3.4

Segmen Program 4.6 Proses Penyimpanan Data Tautan Gambar

```
# Check if the name contains any excluded words (case-insensitive)
        if not any(exclude_word.lower() in cleaned_name.lower() for
        exclude_word in excluded_names) and \ not any(excluded_url in
        costume_url for excluded_url in excluded_urls):
          data = {
            "name": costume_name,
            "link" : costume_url,
            "image": []
          }
          image_data.append(data)
    return image_data
```

Bagian terakhir setelah memisahkan segala data yang diperlukan dan tidak adalah memasukkan data ke dalam *Json*. Penyimpan ke dalam *Json* juga dibedakan untuk menyesuaikan kebutuhan penyimpanan data.

Tabel 4.7 Tabel Segmen Program dan *Flowchart* Konversi Data Kostum ke Json

Segmen Program	*Flowchart*
Segmen Program 4.7	Gambar 3.2

Segmen Program 4.7 Konversi Data Kostum ke *Json*

```
costume_json_file = "costume_data.json"
costume_data_all = [ ]
for costume_data in costume_data_all:
    request = Request(costume_data["link"], headers={"user-agent": useragent})
    response = urlopen(request)
    html_content = response.read()
    inner_soup = bs4.BeautifulSoup(html_content, 'html.parser')
    description = inner_soup.find('div', class_='post_content entry-content')
    a_tags = inner_soup.find('a', class_='trx_addons_icon-eye')
    if a_tags:
        span_tag = a_tags.find('span', class_='post_counters_number')
        if span_tag:
            views = span_tag.text.strip()
            print("views:", views)
            costume_data["views"] = views  # Add views count to the data dictionary
        else:
            print("views: empty")
            costume_data["views"] = ""  # If views count is not found, set it to an
empty string
    if description:
        p = description.find('p')
        if p:
            costume_desc = p.get_text(strip=True)
            print("desc:", costume_desc)
            costume_data["desc"] = costume_desc
```

```
            size = extract_size_from_name_and_description(costume_data["name"],
costume_desc)  # Pass costume_name
            costume_data["size"].append(size)
            if size:
                costume_data["size"] = size
            else:
                costume_data["size"] = size
        else:
            size = extract_size_from_name_and_description(costume_data["name"],
costume_desc)
            costume_data["size"] = size
        print("size:", costume_data["size"])
        print("Finished scraping content inside the URL.")
        print("-" * 70)
with open(costume_json_file, mode='w', encoding='utf-8') as costume_file:
    json.dump(costume_data_all, costume_file, ensure_ascii=False, indent=4)
```

Tabel 4.8 Tabel Segmen Program dan *Flowchart* Konversi Data Kategori ke json

Segmen Program	*Flowchart*
Segmen Program 4.8	Gambar 3.3

Segmen Program 4.8 Konversi Data Kategori ke *Json*

```
costume_json_file = "category_data.json"
category_json = "category_list.json"
costume_data_all = []
category = []

for page_num in range(start_page, end_page + 1):
    current_url = start_url.format(page_num)
    print("Scraping:", current_url)
```

```
    # Create a Request object with headers
    request = Request(current_url, headers={"user-agent": useragent})
    # Open the URL with urlopen
    response = urlopen(request)
    # Read the HTML content directly from the response
    html_content = response.read()
    soup = bs4.BeautifulSoup(html_content, 'html.parser')
    # Extract data from the current page
    costume_data = extract_data_from_page(soup)
    # Append the extracted data to the main list
    for data in costume_data:
        costume_name = data["name"]
        for desc in data["desc"]:
            costume_data_all.append({"name": costume_name, "desc": desc})
            # Append unique categories to the category list
            if desc not in category:
                category.append(desc)
# Write the collected data to JSON file
with open(costume_json_file, mode='w', encoding='utf-8') as costume_file:
    json.dump(costume_data_all, costume_file, ensure_ascii=False, indent=4)
with open(category_json, mode='w', encoding='utf-8') as category_file:
    json.dump(category, category_file, ensure_ascii=False, indent=4)
```

Tabel 4.9 Tabel Segmen Program dan *Flowchart* Konversi Tautan Gambar ke Json

Segmen Program	*Flowchart*
Segmen Program 4.9	Gambar 3.4

Segmen Program 4.9 Konversi Data Tautan Gambar ke *Json*

```
costume_json_file = "image_data.json"
image_data_all = []

# Extract image data for each costume
for costume_data in image_data_all:
    request = Request(costume_data["link"], headers={"user-agent": useragent})
    response = urlopen(request)
    html_content = response.read()
    inner_soup = bs4.BeautifulSoup(html_content, 'html.parser')

    img = inner_soup.find('img', class_='attachment-coleo-thumb-full')

    if img:
        img_src = img.get('src')
        print("img link:", img_src)
        costume_data["image"] = img_src  # Add image link to the data dictionary
    else:
        print("img: empty.")
        costume_data["image"] = ""  # If image link is not found, set it to an empty
string

    print("-" * 70)

# Write the collected data to JSON file
with open(costume_json_file, mode='w', encoding='utf-8') as costume_file:
```

4.2 Sistem *Login*

Halaman awal yang akan dilihat oleh pengguna, baik sebagai pengguna biasa maupun admin, adalah halaman *landing*. Ini menampilkan beberapa koleksi kostum yang dimiliki oleh Toko Rental XYZ. Jika mereka sudah memiliki akun, pengguna dapat melakukan *login*. Untuk alur penyimpanan ke *database*, Segmen Program 4.10 dan Segmen Program 4.11 akan menampilkan sistem *login* dengan lebih detail.

Tabel 4.10 Tabel Segmen Program dan *Flowchart* Halaman *Login*

Segmen Program	*Flowchart*
Segmen Program 4.10	Gambar 3.10
Segmen Program 4.11	

Segmen Program 4.10 Halaman *Login*

```
<div class="container">
  <div class="row">
    <div class="col">
      <div class="logo">
        <img src="/img/logo-gk.png">
      </div>
      <div class="title">
        <p>Selamat Datang Kembali di Website Gudang Kostum Premium Surabaya</p>
      </div>
      <form class="form login" method="POST" action="{{ route('login') }}">
        @csrf
        <div class="row mb-3">
          <p for="email" class="col-form-label">{{ __('Email Address') }}</p>
          <input id="email" type="email" class="form-control @error('email') is-invalid
@enderror" name="email" value="{{ old('email') }}" required autocomplete="email" autofocus>
          @error('email')
            <span class="invalid-feedback" role="alert">
              <strong>{{ $message }}</strong>
            </span>
          @enderror
        </div>
        <div class="row mb-3">
          <p for="password" class="col-form-label">{{ __('Password') }}</p>
          <input id="password" type="password" class="form-control @error('password') is-
invalid @enderror" name="password" required autocomplete="current-password">
          @error('password')
            <span class="invalid-feedback" role="alert">
              <strong>{{ $message }}</strong>
```

```
            <div class="row mb-0">
                <div class="col-md-8">
                    <button type="submit" id="login-btn">
                        {{ __('Login') }}
                    </button>
                    @if (Route::has('password.request'))
                        <a class="btn btn-link" href="{{ route('password.request') }}">
                            {{ __('Forgot Your Password?') }}
                        </a>
                    @endif
                </div>
            </div>
        </form>
        <a href="{{ route('register') }}">Belum punya akun? Buat di sini</a>
    </div>
```

Segmen Program 4.11 A Autentikasi *Login*

```
<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use Illuminate\Foundation\Auth\AuthenticatesUsers;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;

class LoginController extends Controller
{
    /*
    |--------------------------------------------------------------------------
    | Login Controller
    |--------------------------------------------------------------------------
    |
    | This controller handles authenticating users for the application and
    | redirecting them to your home screen. The controller uses a trait
    | to conveniently provide its functionality to your applications.
    |
    */

    use AuthenticatesUsers;

    protected function redirectTo()
    {
```

```
    use AuthenticatesUsers;

    protected function redirectTo()
    {
        if (Auth::user()->is_admin) {
            return '/admin/dashboard'; // Change this to your admin dashboard route
        }

        return '/';
    }

    /**
     * Where to redirect users after login.
     *
     * @var string
     */
    protected $redirectTo = '/';

    /**
     * Create a new controller instance.
     *
     * @return void
     */
    public function __construct()
    {
        $this->middleware('guest')->except('logout');
        // $this->middleware('auth')->only('logout');
    }
}
```

4.3 Halaman Pendaftaran Akun

Jika pengguna belum memiliki akun atau bukan seorang admin maka, pengguna dapat membuat akun melalui halaman pendaftaran akun atau halaman *register*. Pada Segmen Program 4.12 dan 4.13 akan dapat dilihat lebih jelas untuk tampilan serta proses penyimpanan data kedalam *database.* Data yang akan disimpan adalah *username*, email, dan password.

Tabel 4.11 Tabel Segmen Program dan *Flowchart* Pendaftaran Akun

Segmen Program	*Flowchart*
Segmen Program 4.12	Gambar 3.10
Segmen Program 4.13	

Segmen Program 4.12 Halaman Pendaftaran Akun

```
<div class="col-md-5">
        <div class="logo">
          <img src={{ asset("/img/logo-gk.png") }}>
        </div>
        <div class="title">
          <p>Selamat Datang di Website Gudang Kostum Premium Surabaya</p>
        </div>
        <form method="POST" action="{{ route('register') }}">
          @csrf
          <div class="row mb-1">
            <p for="name" class="col-form-label">{{ __('Name') }}</p>
            <input id="name" type="text" class="form-control @error('name') is-invalid
@enderror" name="name" value="{{ old('name') }}" required autocomplete="name" autofocus>

            @error('name')
              <span class="invalid-feedback" role="alert">
                <strong>{{ $message }}</strong>
              </span>
            @enderror
          </div>

          <div class="row mb-1">
            <p for="email" class="col-form-label">{{ __('Email Address') }}</p>
            <input id="email" type="email" class="form-control @error('email') is-invalid
@enderror" name="email" value="{{ old('email') }}" required autocomplete="email">

            @error('email')
              <span class="invalid-feedback" role="alert">
                <strong>{{ $message }}</strong>
              </span>
            @enderror
          </div>

          <div class="row mb-1">
            <p for="password" class="col-form-label">{{ __('Password') }}</p>
            <input id="password" type="password" class="form-control @error('password') is-
invalid @enderror" name="password" required autocomplete="new-password">

            @error('password')
              <span class="invalid-feedback" role="alert">
                <strong>{{ $message }}</strong>
              </span>
            @enderror
          </div>
```

```
            <div class="row mb-3">
                <p for="password-confirm" class="col-form-label">{{ __('Confirm Password') }}</p>
                <input id="password-confirm" type="password" class="form-control"
name="password_confirmation" required autocomplete="new-password">
            </div>

            <div class="row mb-0 button-ctr">
                <div class="col">
                    <button type="submit" class="btn btn-dark">
                        {{ __('Register') }}
                    </button>
                </div>
            </div>
        </form>
        <a href="{{ route('login') }}">Masuk menggunakan akun yang sudah ada</a>
    </div>
```

Segmen Program 4.13 Pencocokan Data Login

```
<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use App\Models\User;
use Illuminate\Foundation\Auth\RegistersUsers;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Validator;

class RegisterController extends Controller
{
    /*
    |--------------------------------------------------------------------------
    | Register Controller
    |--------------------------------------------------------------------------
    |
    | This controller handles the registration of new users as well as their
    | validation and creation. By default this controller uses a trait to
    | provide this functionality without requiring any additional code.
    |
    */

    use RegistersUsers;
```

```
    /**
     * Where to redirect users after registration.
     *
     * @var string
     */
    protected $redirectTo = '/home';

    /**
     * Create a new controller instance.
     *
     * @return void
     */
    public function __construct()
    {
        $this->middleware('guest');
    }

    /**
     * Get a validator for an incoming registration request.
     *
     * @param  array  $data
     * @return \Illuminate\Contracts\Validation\Validator
     */
    protected function validator(array $data)
    {
        return Validator::make($data, [
            'name' => ['required', 'string', 'max:255'],
            'email' => ['required', 'string', 'email', 'max:255', 'unique:users'],
            'password' => ['required', 'string', 'min:8', 'confirmed'],
        ]);
    }

    /**
     * Create a new user instance after a valid registration.
     *
     * @param  array  $data
     * @return \App\Models\User
     */
    protected function create(array $data)
    {
        return User::create([
            'name' => $data['name'],
            'email' => $data['email'],
            'password' => Hash::make($data['password']),
            'is_admin' => false, // Ensure new users are not admins by default
        ]);
    }
`
```

4.4 Halaman Katalog Administrator

Halaman ini adalah halaman katalog admin yang memiliki fitur pencarian dan filter menggunakan kategori. Admin dapat menggunakan halaman ini sebagai alat bantu untuk menunjukkan kostum apa saja yang tersedia. Untuk bagian pencarian menggunakan nama dan penyaringan kostum berdasarkan kategori, lihat Segmen Program 4.14 untuk tampilan *input* pencarian dan penyaringan. Segmen Program 4.15 mengambil data kostum dan pencarian kostum dari database, lihat Segmen Program 4.16.

Tabel 4.12 Tabel Segmen Program dan *Flowchart* Pencarian Administrator

Segmen Program	*Flowchart*
Segmen Program 4.14	Gambar 3.5
Segmen Program 4.15	
Segmen Program 4.16	

Segmen Program 4.14 Pencarian Nama dan Penyaringan Kategori

```
<h1>Katalog</h1>
  <div class="row search-container">
    <div class="col-sm-3">
      <form id="search-form" action="{{ route('katalog.searchCostume') }}" method="GET">
        <div class="input">
          <span>Search</span>
          <div class="input-group mb-3">
            <input type="text" name="search" class="form-control" value="{{ old('search') }}">
            <button class="btn btn-outline-secondary" type="button" id="search-button">
              <i class="fas fa-magnifying-glass"></i>
            </button>
          </div>
        </div>
      </form>
    </div>
    <div class="col-sm-3">
      <div class="input">
        <i class="fas fa-filter"></i><span>Kategori</span>
        <select name="category" class="form-select" id="category-filter" data-placeholder="Pilih
Kategori">
          <option value="" default></option>
          <option value="14">Adat</option>
          <option value="6">Aksesoris</option>
          <option value="12">Animal</option>
          <option value="15">Apron</option>
```

```
<script>
    $(document).ready(function(){
        function performSearch(page = 1) {
            let searchQuery = $('[name="search"]').val();
            console.log(searchQuery);
            let category = $('#category-filter').val();
            console.log(category);
            console.log("Filter parameters:", { search: searchQuery, category: category });

            $.ajax({
                url: '{{ route('katalog.searchCostume') }}',
                method: 'GET',
                data: {
                    search: searchQuery,
                    category: category,
                    page: page
                },
                success: function(response) {
                    $('#katalog-container').html(response.katalogs);
                    $('#pagination-container').html(response.pagination);
                },
                error: function(xhr, status, error) {
                    console.error(error);
                }
            });
        }

        $('#search-button').on('click', function(e) {
            e.preventDefault();
            performSearch();
        });

        $('#category-filter').on('change', function() {
            performSearch();
        });

        $(document).on('click', '.pagination a', function(e) {
            e.preventDefault();
            let page = $(this).attr('href').split('page=')[1];
            performSearch(page);
        });

        $('#search-form').on('submit', function(e) {
            e.preventDefault();
            performSearch();
        });
    });
```

```
        $('#search-form').on('submit', function(e) {
            e.preventDefault();
            performSearch();
        });
    });
</script>
```

Segmen Program 4.15 Pengambilan Data Kostum untuk Katalog

```
public function getCostume()
    {
        $data = costume::select('costume.*', 'image.imageUrl',
DB::raw('GROUP_CONCAT(category.category SEPARATOR '', '') AS categories'))
            ->join('costume_category', 'costume.id_costume', '=', 'costume_category.id_costume')
            ->join('category', 'costume_category.id_category', '=', 'category.id_category')
            ->join('image', 'image.id_costume', '=', 'costume.id_costume')
            ->groupBy('costume.id_costume', 'costume.name', 'costume.size', 'costume.price',
'costume.description', 'costume.views', 'costume.interest','image.imageUrl')
            ->orderBy('costume.id_costume', 'asc')
            ->paginate(12);

        return view('admin_util/katalog', ['katalogs' => $data]);
    }
```

Segmen Program 4.16 Pengambilan Data untuk Pencarian Kostum

```
public function searchCostume(Request $request)
    {
        $search = $request->input('search');
        $category = $request->input('category');

        $query = costume::select('costume.*', 'image.imageUrl',
DB::raw('GROUP_CONCAT(category.category SEPARATOR '', '') AS categories'))
            ->join('costume_category', 'costume.id_costume', '=', 'costume_category.id_costume')
            ->join('category', 'costume_category.id_category', '=', 'category.id_category')
            ->join('image', 'image.id_costume', '=', 'costume.id_costume')
            ->groupBy('costume.id_costume', 'costume.name', 'costume.size', 'costume.price',
'costume.description', 'costume.views', 'costume.interest', 'image.imageUrl');

        if ($search) {
            $query->where('costume.name', 'like', '%' . $search . '%');
        }
```

```
    if ($category) {
        $query->where('costume_category.id_category', '=', $category);
    }

    $data = $query->orderBy('costume.id_costume', 'asc')->paginate(12);

    if ($request->ajax()) {
        return response()->json([
            'katalogs' => view('partials.katalog_items', ['katalogs' => $data])->render(),
            'pagination' => $data->links('pagination::bootstrap-5')->render()
        ]);
    }

    return view('admin_util.katalog', ['katalogs' => $data]);
}
```

4.5 Halaman Pengaturan Data Sewa

Halaman pengaturan data sewa adalah halaman untuk administrator memasukkan data sewa yang telah terjadi di hari tersebut. Administrator dapat memasukkan data berupa nama kostum dan keterangan aksesoris ataupun tambahan lainnya. Untuk lebih jelasnya akan dijelaskan melalui Segmen Program 4.18 untuk memasukkan data dan melakukan penggantian status kostum menjadi dipinjam, 4.17 untuk pengambilan dan penambahan, serta Segmen Program 4.19 untuk Pengambilan data untuk grafik.

Tabel 4.13 Tabel Segmen Program dan *Flowchart* Tambah Data Sewa

Segmen Program	*Flowchart*
Segmen Program 4.17	Gambar 3.6
Segmen Program 4.18	
Segmen Program 4.19	

Segmen Program 4.17 Halaman Tambah Data Sewa

```
<div class="col table-sewa">
        <table class="table table-bordered" id="table">
            <thead>
                <tr>
                    <th>ID</th>
                    <th>Kostum</th>
                    <th>Tambahan</th>
                    <th>Tanggal Pinjam</th>
                    <th>Tanggal Kembali</th>
```

```
                </tr>
              </thead>
              <tbody>
                @forelse ($rentedData as $rent)
                  <tr>
                    <td>{{ $rent->id_rent }}</td>
                    <td>{{ $rent->name }}</td>
                    <td>{{ $rent->description }}</td>
                    <td>{{ $rent->created_at }}</td>
                    <td>{{ $rent->ended_at }}</td>
                  </tr>
                @empty
                  <tr>
                    <td colspan="4">No rents found.</td>
                  </tr>
                @endforelse
              </tbody>
            </table>
          </div>
          <div class="col col-input">
            <div class="input-form">
              <form id="tambah-sewa-form" method="POST" action="{{ route('sewa.addRent') }}">
                @csrf
                <h3>Tambah Sewa</h3>
                <p>Tanggal Sewa</p>
                <input type="date" name="created_at" value="{{ now()->format('Y-m-d') }}">
                <p>Tanggal Kembali</p>
                <input type="date" name="ended_at" value="{{ now()->format('Y-m-d') }}">
                <p>Kostum</p>
                <input type="text" name="id_costume" id="autocomplete" placeholder="Type
costumes here"/>
                <input type="hidden" name="id_costume" id="costume-id" />
                <p>Tambahan/Aksesoris</p>
                <textarea name="description" id="desc-input" cols="30" rows="10"></textarea>
                <div class="button-add">
                  <button type="submit" class="btn btn-primary" id="tambah-sewa">Tambah
Sewa</button>
                </div>
              </form>
            </div>
          </div>
        </div>
      </div>
@endsection
@section('script')
  <script>
    $('#tambah-sewa-form').on('submit', function(e) {
      e.preventDefault();

      var selectedCostume = $('#costume-id').val();
      console.log(selectedCostume);
```

```
        // Check if the description field is empty
        var descInput = $('#desc-input').val();
        if (!descInput.trim()) {
            $('#desc-input').val('-');
        }

        var formData = $(this).serialize();

        $.ajax({
            url: '{{ route('sewa.addRent') }}',
            method: 'POST',
            data: formData,
            processData: false,
            contenType: false,
            success: function(response) {
                if (response.success) {
                    console.log('Data added to database:', response);
                    // Get the costume ID from the form or response
                    var costumeId = $('#costume-id').val();
                    updateCostumeStatus(costumeId, 2);
                } else {
                    alert('Failed to add data');
                }
            }
        })
    })

    function updateCostumeStatus(costumeId, status) {
        $.ajax({
            url: '{{ route("katalog.updateStatus", ":id_costume") }}'.replace(':id_costume',
costumeId),
            method: 'POST',
            data: {
                _token: '{{ csrf_token() }}', // Include the CSRF token
                status: status
            },
            success: function(response) {
                if (response.success) {
                    console.log('Status updated successfully:', response);
                    window.location.reload(); // Reload the page
                } else {
                    alert('Failed to update status: ' + response.message);
                }
            },
            error: function(xhr, status, error) {
                console.error('Error updating status:', xhr.responseText);
                alert('An error occurred while updating the status');
            }
        });
    }
```

Segmen Program 4.18 Pengambilan, Penambahan dan Penghapusan Data Sewa

```
<?php

namespace App\Http\Controllers;

use Carbon\Carbon;
use App\Models\rented;
use App\Models\costume;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use Illuminate\Support\Facades\Log;

class RentController extends Controller
{
    public function getRent() {
        $today = Carbon::today(); // Get today's date

        $rentedData = rented::select('rented.*', 'costume.name')
            ->join('costume', 'rented.id_costume', '=', 'costume.id_costume')
            ->whereDate('rented.created_at', $today) // Filter by today's date
            ->orderBy('costume.name', 'asc')
            ->get();

        $costumeData = costume::all(); // Get all costumes for the dropdown

        return view('admin_util.add_sewa', [
            'rentedData' => $rentedData,
            'costumeData' => $costumeData
        ]);

    }

    public function addRent(Request $request) {
        $validated = $request->validate([
            'id_costume' => 'required',
            'description' => 'required',
            'created_at' => 'required',
            'ended_at' => 'required',
        ]);

        $rent = rented::create([
            'id_costume' => $validated['id_costume'],
            'description' => $validated['description'],
            'created_at' => $validated['created_at'],
            'ended_at' => $validated['ended_at'],
        ]);

        return response()->json(['success' => true]);
    }

    public function updateStatus(Request $request, $id_costume)
    {
```

```
public function updateStatus(Request $request, $id_costume)
{
    Log::info('Update Status Request:', $request->all());

    $validated = $request->validate([
        'status' => 'required|integer',
    ]);

    Log::info('Costume ID:', ['id_costume' => $id_costume]);

    $costume = costume::where('id_costume', $id_costume)->first();
    if ($costume) {
        $costume->status = $validated['status'];
        $costume->save(); // Use save method to update

        return response()->json(['success' => true, 'message' => 'Status updated successfully']);
    }

    return response()->json(['success' => false, 'message' => 'Costume not found']);
}
```

Segmen Program 4.19 Pengambilan Data Grafik

```
public function getTopViewedCostumes(Request $request)
    {
        $limit = $request->input('limit', -1); // Get limit from request, default to -1 (no limit)

        $query = costume::orderBy('views', 'desc');

        if ($limit > 0) {
            $query->limit($limit);
        }

        $topCostumes = $query->get(['name', 'views']);

        return response()->json($topCostumes);
    }

    public function getFrequentlyViewedCostumes(Request $request)
    {
        $interval = $request->input('interval'); // Get interval from request

        $query = DB::table('viewed_costume')
            ->join('costume', 'viewed_costume.id_costume', '=', 'costume.id_costume') // Join with
costumes table
            ->select(DB::raw('costume.name as costume_name, COUNT(*) as total_asked,
DATE(viewed_costume.created_at) as view_date'))
            ->groupBy('costume_name', 'view_date');

        // Apply interval filter
        switch ($interval) {
            case 'daily':
                $query->whereDate('viewed_costume.created_at', '>=', Carbon::now()->subDays(1));
```

```
            // By default, fetch all data
            break;
    }

    $data = $query->get();

    return response()->json($data);
}

public function getRentalStats(Request $request) {
```

4.6 Halaman *Chat* Administrator

Administrator pada halaman ini memiliki kemampuan untuk membaca dan menjawab *chat* yang dikirimkan oleh pengunjung. Mereka juga memiliki kemampuan untuk membaca semua *chat* yang dikirimkan oleh pengunjung ke halaman *chat*. Setelah membaca dan menjawab pelanggan, administrator dapat memilih untuk menghapus *chat*. Pastinya pengunjung dengan akun akan dapat berbicara kembali dengan administrator. Segmen Program 4.20 menjelaskan halaman *chat* untuk bagian *input* dan penghapusan pesan, serta koneksi Socket.io dan Redis agar dapat berjalan secara *real time*. Gambar 4.22 menjelaskan alur penyimpanan dan pengambilan data pesan antara administrator dan pengunjung yang mengirim pesan kepada administrator.

Tabel 4.14 Tabel Segmen Program dan *Flowchart Chat* Administrasi

Segmen Program	*Flowchart*
Segmen Program 4.20	Gambar 3.7
Segmen Program 4.21	
Segmen Program 4.22	

Segmen Program 4.20 Halaman *Chat* Administrator

```
          <ul class="list-group list-chat-item">
            @if($users->count())
              @foreach ($users as $user)
                <li class="chat-user-list @if($user->id == $friendInfo->id) active @endif">
                  <a href="javascript:void(0)" data-id="{{ $user->id }}" class="user-link"
style="font-weight: 600;">
                    {{ $user->name }}
                  </a>
                  <div class="remove-chat">
                    <button class="btn btn-danger" id="remove" value="${{ $user->id }}">
                      <i class="fas fa-trash"></i>
                    </button>
                  </div>
                </li>
              @endforeach
            @endif
          </ul>
        </div>
      </div>
      <div class="col-md-9 chat-section">
```

```
        <div class="chat-header" id="chatHeader">
          <div class="chat-name" style="font-weight: 600; margin-left:8px;">
            {{ $friendInfo->name }}
          </div>
        </div>
        <div class="chat-body">
          <div class="message-listing" id="messageWrapper">
```

```
            break;
        case 'monthly':
            $query->whereDate('viewed_costume.created_at', '>=', Carbon::now()->subMonths(1));
            break;
        case 'quarterly':
            $query->whereDate('viewed_costume.created_at', '>=', Carbon::now()->subMonths(3));
            break;
        case 'yearly':
            $query->whereDate('viewed_costume.created_at', '>=', Carbon::now()->subYear());
            break;
        default:
            // By default, fetch all data
            break;
    }

    $data = $query->get();

    return response()->json($data);
  }

  public function getTopInterestCostumes()
  {
    $costumes = Costume::select('name', 'interest')
        ->where('interest', '>', 0) // Exclude costumes with zero interest
        ->orderBy('interest', 'desc')
        ->get();

    return response()->json($costumes);
  }

  public function getFrequentlyAskedCostumes(Request $request)
  {
    $interval = $request->input('interval'); // Get interval from request

    $query = DB::table('asked_costume')
        ->join('costume', 'asked_costume.id_costume', '=', 'costume.id_costume') // Join with
costumes table
        ->select(DB::raw('costume.name as costume_name, COUNT(*) as total_asked,
DATE(asked_costume.created_at) as view_date'))
        ->groupBy('costume_name', 'view_date');

    // Apply interval filter
    switch ($interval) {
        case 'daily':
            $query->whereDate('asked_costume.created_at', '>=', Carbon::now()->subDays(1));
            break;
        case 'monthly':
            $query->whereDate('asked_costume.created_at', '>=', Carbon::now()->subMonths(1));
            break;
        case 'quarterly':
            $query->whereDate('asked_costume.created_at', '>=', Carbon::now()->subMonths(3));
            break;
        case 'yearly':
            $query->whereDate('asked_costume.created_at', '>=', Carbon::now()->subYear());
            break;
        default:
```

```
        },
        success: function(response) {
            if (response.success) {
                console.log('Chat history deleted');
                // Optionally clear the message display if the current chat is deleted
                if (friendId == "{{ $friendInfo->id }}") {
                    $messageWrapper.html('');
                }
            } else {
                console.error('Error deleting chat history:', response.message);
            }
        },
        error: function(xhr, status, error) {
            console.error('Error deleting chat history:', error);
        }
    });
}

function fetchConversation(friendId) {
    console.log("Fetching conversation for user ID:", friendId);
    $.ajax({
        url: "{{ route('message.conversation', '') }}/" + friendId,
        type: 'GET',
        success: function(response) {
            if (response.success) {
                $chatName.text(response.friendInfo.name);
                $messageWrapper.html(''); // Clear previous messages
                response.messages.forEach(message => {
                    appendMessageToWrapper(message);
                });
            }
        },
        error: function(xhr, status, error) {
            console.error('Error fetching conversation:', error);
        }
    });
}

$chatInput.keypress(function (e) {
    let message = $(this).html();
    if (e.which === 13 && !e.shiftKey) {
        e.preventDefault();
        $chatInput.html("");
        console.log('friendinfo:', friendId)
        sendMessage(message);
        return false;
```

```
        }
    });

    function sendMessage(message) {
        let url = "{{ route('message.send-message') }}";
        let token = "{{ csrf_token() }}";
        console.log(friendId)

        // Check if a friend is selected
        if (!friendId) {
            console.error('No friend selected for sending message');
            return;
        }

        console.log("Sending message to friend ID:", friendId);

        let formData = new FormData();
        formData.append('message', message);
        formData.append('_token', token);
        formData.append('reciever_id', friendId);

        appendMessageToSender(message);

        $.ajax({
            url: url,
            type: 'POST',
            data: formData,
            processData: false,
            contentType: false,
            dataType: 'JSON',
            success: function(response) {
                if(response.success){
                    console.log(response.data);
                } else {
                    console.error('Error:', response.message);
                }
            },
            error: function(xhr, status, error) {
                console.error('Submission error:', error);
            }
        });
    }

    function appendMessageToWrapper(message) {
        let messageElement = '<div class="row message align-items-center mb-2">' +
            '<div class="col-md-12 user-info" style="display: flex; align-items: center; margin-top:
16px; margin-left:16px;">\n' +
```

```
                '<div class="chat-image" style="margin-right: 10px">\n' +
                  '<i class="fas fa-user"></i>\n' +
                '</div>\n' +
                '<div class="chat-name" style="font-weight: 600; display:flex; flex-direction: row;">\n'
+
                  message.sender_name +
                  '<span class="small time" title="'+moment(message.created_at).format('YYYY-MM-
DD HH:mm:ss')+'" style="color: grey; margin-left: 5px; padding-top: 3px;">\n' +
                  moment(message.created_at).format('h:mm A') + '</span>\n' +
                '</div>\n' +
              '</div>\n' +
              '<div class="col-md-12 message-content">\n' +
                '<div class="message-text">\n' + message.content +
                '</div>\n' +
              '</div>\n' +
            '</div>';

            $messageWrapper.append(messageElement);
          }

          function appendMessageToSender(message) {
            let name = '{{ $myInfo->name }}';
            let messageElement = '<div class="row message align-items-center mb-2">' +
              '<div class="col-md-12 user-info" style="display: flex; align-items: center; margin-top:
16px; margin-left:16px;">\n' +
                '<div class="chat-image" style="margin-right: 10px">\n' +
                  '<i class="fas fa-user"></i>\n' +
                '</div>\n' +
                '<div class="chat-name" style="font-weight: 600; display:flex; flex-direction: row;">\n'
+
                  name +
                  '<span class="small time" title="'+getCurrentDateTime()+'" style="color: grey;
margin-left: 5px; padding-top: 3px;">\n' +
                  getCurrentTime() + '</span>\n' +
                '</div>\n' +
              '</div>\n' +
              '<div class="col-md-12 message-content">\n' +
                '<div class="message-text">\n' + message +
                '</div>\n' +
              '</div>\n' +
            '</div>';

            $messageWrapper.append(messageElement);
          }
```

```
        function getCurrentDateTime() {
            return moment().format('YYYY-MM-DD HH:mm:ss');
        }

        function getCurrentTime() {
            return moment().format('h:mm A');
        }

        socket.on("private-channel:PrivateMessageEvent", function (message) {
            appendMessageToWrapper(message);
        });
    });

</script>
```

Segmen Program 4.21 Sambungan Socket.io dan Redis

```
import express from 'express';
import { createServer } from 'http';
import { Server } from 'socket.io';
import cors from 'cors';
import Redis from 'ioredis';
import axios from 'axios';
import fs from 'fs';

const app = express();
app.use(cors());

const http = createServer(app);
const io = new Server(http, {
    cors: {
        origin: 'http://127.0.0.1:8000',
        methods: ['GET', 'POST'],
        allowedHeaders: ['my-custom-header'],
        credentials: true
    }
});

const PORT = 8005; // Define the PORT variable

const redisSubscriber = new Redis();
const redisClient = new Redis(); // Separate Redis connection for other operations
const users = [];

http.listen(PORT, function () {
    console.log(`Listening to port ${PORT}`);
});
```

```
redisSubscriber.subscribe('private-channel', function() {
    console.log('subscribed to private channel');
});

redisSubscriber.on('message', function(channel, message) {
    try {
        message = JSON.parse(message);
    } catch (e) {
        console.error('Error parsing message', e);
        return;
    }

    console.log(message);
    console.log(channel);

    if (channel === 'private-channel') {
        const data = message.data;
        if (!data) {
            console.error('Data is undefined');
            return;
        }

        const reciever_id = data.reciever_id;
        const event = message.event;

        if (reciever_id && users[reciever_id]) {
            io.to(`${users[reciever_id]}`).emit(`${channel}:${event}`, data);
        } else {
            console.error('Reciever ID is undefined or user not connected');
        }
    }
});

io.on('connection', function (socket) {
    socket.on("user_connected", function (user_id) {
        users[user_id] = socket.id;
        io.emit('updateUserStatus', users);
        console.log("user connected " + user_id);
    });

    socket.on("disconnect", function() {
        // Find the user ID by socket ID and remove it from the users object
        for (const [user_id, socket_id] of Object.entries(users)) {
            if (socket_id === socket.id) {
                delete users[user_id];
                break;
            }
        }
```

```
    }
    io.emit('updateUserStatus', users);
    console.log(users);
  });
```

Segmen Program 4.22 Mengambil, Mengirim dan Menyimpan ke *Database*

```
<?php

namespace App\Http\Controllers;

use App\Models\User;
use App\Models\Message;
use App\Models\UserMessage;
use Illuminate\Http\Request;
use App\Events\PrivateMessageEvent;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Log;

class MessageController extends Controller
{
  public function conversation($userId) {
    $adminId = 4; // Assuming the admin ID is 4.
    $authUserId = Auth::id();

     // Log the user IDs for debugging
    Log::info('Authenticated User ID: ' . $authUserId); //1
    Log::info('User ID from Route: ' . $userId); //4

    // Check if the authenticated user is an admin or the specific user
    if (!in_array($authUserId, [$adminId, $userId])) {
      return response()->json(['message' => 'Unauthorized action.'], 403);
    }

    $users = User::where('id', '!=', Auth::id())->get();
    $friendInfo = User::findOrFail($userId);
    $myInfo = User::find(Auth::id());

    // Fetch user messages with the associated message content
    $userMessages = UserMessage::with('message')
      ->where(function ($q) use ($userId, $authUserId, $adminId) {
        $q->where(function ($q2) use ($authUserId, $adminId) {
          $q2->where('sender_id', $authUserId)
            ->where('reciever_id', $adminId);
        })
        ->orWhere(function ($q2) use ($authUserId, $adminId) {
          $q2->where('sender_id', $adminId)
            ->where('reciever_id', $authUserId);
```

```
            })
            ->orWhere(function ($q2) use ($userId, $adminId) {
                $q2->where('sender_id', $userId)
                    ->where('reciever_id', $adminId);
            })
            ->orWhere(function ($q2) use ($userId, $adminId) {
                $q2->where('sender_id', $adminId)
                    ->where('reciever_id', $userId);
            });
        })
        ->get();

    // Format messages to include sender name and message content
    $messages = $userMessages->map(function($userMessage) {
        $messageContent = $userMessage->message ? $userMessage->message->message :
'[Deleted message]';
        $senderName = $userMessage->sender_id == Auth::id() ? Auth::user()->name :
User::find($userMessage->sender_id)->name;

        return [
            'sender_name' => $senderName,
            'content' => $messageContent,
            'created_at' => $userMessage->created_at,
        ];
    });

    if (request()->ajax()) {
        return response()->json([
            'success' => true,
            'friendInfo' => $friendInfo,
            'messages' => $messages,
        ]);
    }

    $data = [
        'users' => $users,
        'friendInfo' => $friendInfo,
        'myInfo' => $myInfo,
        'messages' => $messages,
        'userId' => $userId,
    ];

    if ($authUserId == $adminId) {
        return view('message.conversation', $data);
    } else {
```

```
    } else {
        return view('client_util.client_chat', $data);
    }
}

public function sendMessage(Request $request) {
    $request->validate([
        'message' => 'required',
        'reciever_id' => 'required'
    ]);

    $sender_id = Auth::id();
    $receiver_id = $request->reciever_id;

    $message = new Message();
    $message->message = $request->message;

    if($message->save()) {
        try {
            Log::info('Message saved', [
                'message_id' => $message->id,
                'sender_id' => $sender_id,
                'receiver_id' => $receiver_id
            ]);
            $message->users()->attach($sender_id, ['reciever_id' => $receiver_id]);
            $sender = User::where('id', '=', $sender_id)->first();

            $data = [
                'sender_id' => $sender_id,
                'sender_name' => $sender->name,
                'reciever_id' => $receiver_id,
                'content' => $message->message,
                'created_at' => $message->created_at,
                'message_id' => $message->id,
            ];

            Log::info('Broadcasting event with data', $data);

            // event(new PrivateMessageEvent($data));
            event(new PrivateMessageEvent($data));

            return response()->json([
                'data' => $data,
                'success' => true,
                'message' => 'Message sent successfully'
            ]);
```

```
            } catch (\Exception $e) {
                $message->delete();
                Log::error('Error sending message: ' . $e->getMessage(), ['exception' => $e]);
                return response()->json([
                    'success' => false,
                    'message' => 'Message could not be sent',
                    'error' => $e->getMessage()
                ], 500);
            }
        }

        return response()->json([
            'success' => false,
            'message' => 'Message could not be sent'
        ], 500);
    }

    // MessageController.php

    public function deleteChat($friendId)
    {
        $authUserId = Auth::id();

        // Fetch user messages for the conversation
        $userMessages = UserMessage::where(function ($query) use ($authUserId, $friendId) {
            $query->where('sender_id', $authUserId)
                ->where('reciever_id', $friendId);
        })->orWhere(function ($query) use ($authUserId, $friendId) {
            $query->where('sender_id', $friendId)
                ->where('reciever_id', $authUserId);
        })->get();

        // Delete the messages
        foreach ($userMessages as $userMessage) {
            $message = $userMessage->message;
            if ($message) {
                $message->delete();
            }
            $userMessage->delete();
        }

        return response()->json([
            'success' => true,
            'message' => 'Chat history deleted successfully'
        ]);
    }
}
```

```
        return response()->json([
            'success' => false,
            'message' => 'Message could not be sent'
        ], 500);
    }

    // MessageController.php

    public function deleteChat($friendId)
    {
        $authUserId = Auth::id();

        // Fetch user messages for the conversation
        $userMessages = UserMessage::where(function ($query) use ($authUserId, $friendId) {
            $query->where('sender_id', $authUserId)
                ->where('reciever_id', $friendId);
        })->orWhere(function ($query) use ($authUserId, $friendId) {
            $query->where('sender_id', $friendId)
                ->where('reciever_id', $authUserId);
        })->get();

        // Delete the messages
        foreach ($userMessages as $userMessage) {
            $message = $userMessage->message;
            if ($message) {
                $message->delete();
            }
            $userMessage->delete();
        }

        return response()->json([
            'success' => true,
            'message' => 'Chat history deleted successfully'
        ]);
    }
}
```

4.7 Halaman Pengaturan Promosi

Halaman pengaturan promosi adalah halaman yang dibuat untuk membantu administrator dalam menambahkan *banner* atau iklan promosi pada halaman depan katalog. Pada halaman ini administrator dapat menambah dan menyunting promosi yang ada. Tidak hanya itu administrator juga dapat mencari promosi berdasarkan nama atau judul serta tanggal berlakunya promosi tersebut. Jika suatu promosi sudah melewati batas tanggalnya maka promosi tersebut akan pindah ke halaman daftar promosi yang sudah berjalan. Pada Segmen Program 4.23 akan ditunjukkan untuk bagian penambahan promosi. Segmen Program 4.24 menunjukkan untuk bagian pencarian. Segmen Program 4.25 untuk bagian melakukan penyuntingan pada gambar dan juga data promosi itu sendiri. Terakhir adalah Segmen Program 4.26 yang menjelaskan *query* antara *database* dengan *website*.

Tabel 4.15 Tabel Segmen Program dan *Flowchart* Penambahan Promosi

Segmen Program	*Flowchart*
Segmen Program 4.23	Gambar 3.8
Segmen Program 4.24	
Segmen Program 4.25	
Segmen Program 4.26	

Segmen Program 4.23 Penambahan Promosi Baru

```
<form class="add-promosi" method="POST" enctype="multipart/form-data">
        @csrf
        <div class="col add-image">
          <h3>Foto</h3>
          <img id="imagePreview" class="image-preview" src="" alt="Image Preview">
          <input type="file" name="image" id="image" class="form-control" accept=".jpg"
onchange="previewImage(event)">
        </div>
        <div class="col add-details">
          <div class="input-name">
            <h3>Nama Promo / Promosi</h3>
            <input type="text" name="title" class="form-control title">
          </div>
        </div>
        <br>
        <div class="row date" style="margin-left: 4px;">
          <div class="col-md-6 start-date">
            <h3>Tanggal Mulai Promosi</h3>
            <input type="date" name="created_at" class="form-control">
```

```
            </div>
            <div class="col-md-6 end-date">
              <h3>Tanggal Promosi Berakhir</h3>
              <input type="date" name="ended_at" class="form-control">
            </div>
          </div>
          <br>
          <div class="col add-promo">
            <div class="submit-btn">
              <button type="submit" class="tambah">
                Tambahkan Promosi
                <i class="fas fa-check"></i>
              </button>
            </div>
          </div>
        </form>

<script>
        function previewImage(event) {
          const reader = new FileReader();
          reader.onload = function() {
            const output = document.getElementById('imagePreview');
            output.src = reader.result;
          }
          reader.readAsDataURL(event.target.files[0]);
        }

        $(".add-promosi").on('submit', function(event) {
          event.preventDefault();

          let formData = new FormData(this);
          formData.append('_token', '{{ csrf_token() }}');

          $.ajax({
            url: '{{ route('promosi.addPromo') }}',
            method: 'POST',
            data: formData,
            processData: false,
            contentType: false,
            success: function(response) {
              var promoId = response.id_promo;
              console.log("id promo:", promoId)

              var file = $('#image')[0].files[0]; // Get the file from the input
              var formData2 = new FormData();

              formData2.append('file', file);
              formData2.append('_token', '{{ csrf_token() }}');
              formData2.append('id_promo', promoId);
```

```
            if(promoId) {
                $.ajax({
                    url: '{{ route('promosi.uploadImage') }}',
                    method: 'POST',
                    data: formData2,
                    processData: false,
                    contentType: false,
                    success: function(response) {
                        if (response.image_url) {
                            $.ajax({
                                url: '{{ route('promosi.saveImage') }}',
                                method: 'POST',
                                data: {
                                    id_promo: promoId,
                                    image: response.image_url,
                                    _token: '{{ csrf_token() }}'
                                },
                                success: function(response) {
                                    console.log("Image added to database:", response);
                                    window.location.reload(); // Reload the page
                                }
                            });
                        } else {
                            console.error('Image URL missing in response');
                        }
                    },
                    error: function(xhr, status, error) {
                        console.error('Image upload error:', error);
                    }
                });
            }
        },
        error: function(xhr, status, error) {
            console.error('Promo submission error:', error);
        }
    });
});
</script>
```

Segmen Program 4.24 Pencarian

```
<form id="searchForm" action="{{ route('promosi.searchPromo') }}" method="GET">
        <div class="row">
          <div class="col-md-4 search">
            <i class="fas fa-magnifying-glass"></i>
            <label>Cari</label>
            <br>
            <input type="text" name="search" class="form-control search">
          </div>
          <div class="col-md-4 date-picker">
            <i class="fas fa-calendar"></i>
            <label>Cari berdasarkan tanggal</label>
            <input type="text" name="daterange" class="form-control" id="daterange" />
          </div>
        </div>
      </form>
```

Segmen Program 4.25 Penyuntingan Promosi

```
@foreach ($promos as $promo)
        <form class="add-promosi" method="POST" enctype="multipart/form-data">
          @csrf
          <h3>Foto</h3>
          <div class="photo-preview">
            @foreach ($promo_img as $img)
              <img id="imagePreview" class="image-preview" src="{{ $img->image }}" alt="Image
Preview">
              <input type="file" accept=".jpg" name="image" id="image" class="form-control"
onchange="previewImage(event)">
            @endforeach
          </div>
          <div class="col add-details">
            <div class="input-name">
              <h3>Nama Promo / Promosi</h3>
              <input type="text" name="title" class="form-control title" value="{{ $promo->title
}}">
            </div>
          </div>
          <br>
          <div class="row date" style="margin-left: 4px;">
            <div class="col-md-6 start-date">
              <h3>Tanggal Mulai Promosi</h3>
              <input type="date" name="created_at" class="form-control" value="{{ $promo-
>created_at }}">
            </div>
            <div class="col-md-6 end-date">
              <h3>Tanggal Promosi Berakhir</h3>
              <input type="date" name="ended_at" class="form-control" value="{{ $promo-
>ended_at }}">
            </div>
          </div>
```

```
            <br>
            <div class="col add-promo">
              <div class="submit-btn">
                <button type="submit" class="tambah" value="{{ $promo->id_promo }}">
                  Tambahkan Promosi
                  <i class="fas fa-check"></i>
                </button>
              </div>
            </div>
          </form>
        @endforeach
<script>
        function previewImage(event) {
          const reader = new FileReader();
          reader.onload = function() {
            const output = document.getElementById('imagePreview');
            output.src = reader.result;
          }
          reader.readAsDataURL(event.target.files[0]);
        }

        $(document).ready(function() {
          $('.edit-promosi').on('submit', function(e) {
            e.preventDefault();

            var formData = new FormData(this);
            var file = $('#image')[0].files[0];
            let promo_id = $(this).find('button.edit').val();
            formData.append('promo_id', promo_id);

            for (var pair of formData.entries()) {
              console.log(pair[0]+ ': ' + pair[1]);
            }

            if (file) {
              // Append file to formData
              formData.append('file', file);

              $.ajax({
                url: '{{ route('promosi.uploadImage') }}',
                method: 'POST',
                data: formData,
                processData: false,
                contentType: false,
                success: function(response) {
                  if (response.image_url) {
                    formData.append('imageUrl', response.image_url);
```

```
                    updatePromo(promo_id, formData);
                } else {
                    console.error('Image URL missing in response');
                }
            },
            error: function(xhr, status, error) {
                console.error('Image upload error:', error);
            }
        });
    } else {
        $.ajax({
            url: '{{ route("promosi.updatePromo", ":id_promo") }}'.replace(':id_promo',
promo_id),
            type: 'POST',
            data: formData,
            processData: false,
            contentType: false,
            success: function() {
                console.log('Promo updated to database');
                window.location.reload(); // Reload the page
            },
            error: function(xhr, status, error) {
                console.error('Promo update error:', error);
            }
        });
    }
});

function updatePromo(promo_id, formData) {
    $.ajax({
        url: '{{ route("promosi.updateImagePromo", ":id_promo") }}'.replace(':id_promo',
promo_id),
        type: 'POST',
        data: formData,
        processData: false,
        contentType: false,
        success: function() {
            console.log('Promo updated to database');
            window.location.reload(); // Reload the page
        },
        error: function(xhr, status, error) {
            console.error('Promo update error:', error);
        }
    });
}
});
</script>
```

Segmen Program 4.26 Manipulasi Data dari *Database*

```
<?php

namespace App\Http\Controllers;

use GuzzleHttp\Client;
use App\Models\Promosi;
use App\Models\Promosi_Image;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Log;
use Carbon\Carbon;

class PromosiController extends Controller
{
    public function index() {
        return view('admin_util/add_promosi');
    }

    public function getPromo() {
        Promosi::where('ended_at', '<', Carbon::now())
            ->where('is_expired', 0)
            ->update(['is_expired' => 1]);

        // Fetch data with pagination
        $activePromotions = Promosi::select('promosi.*', 'promosi_image.*')
            ->join('promosi_image', 'promosi.id_promo', '=', 'promosi_image.id_promo')
            ->where('is_expired', 0)
            ->orderBy('promosi.title', 'asc')
            ->paginate(10, ['*'], 'activePage'); // Paginate active promotions

        return view('admin_util.promosi', [
            'promosi' => $activePromotions
        ]);
    }

    public function getPastPromo() {
        $expiredPromotions = Promosi::select('promosi.*', 'promosi_image.*')
            ->join('promosi_image', 'promosi.id_promo', '=', 'promosi_image.id_promo')
            ->where('is_expired', 1)
            ->orderBy('promosi.title', 'asc')
            ->paginate(10, ['*'], 'expiredPage'); // Paginate expired promotions

        return view('admin_util.promosi_history', [
            'promosi' => $expiredPromotions
        ]);
    }

    public function addPromo(Request $request) {
        $validated = $request->validate([
            'title' => 'required',
            'created_at' => 'required',
```

```
        'created_at' => 'required',
        'ended_at' => 'required'
    ]);

    $promo = Promosi::create([
        'title' => $validated['title'],
        'created_at' => $validated['created_at'],
        'ended_at' => $validated['ended_at']
    ]);

    Log::info('Promo Object:', ['promo' => $promo]); // Log the entire promo object
    return response()->json(['id_promo' => $promo->id_promo]);
}

public function uploadImage(Request $request)
{
    if ($request->hasFile('file')) {
        try {
            $client = new Client();
            $response = $client->request('POST', 'https://api.imgbb.com/1/upload', [
                'form_params' => [
                    'image' => base64_encode(file_get_contents($request->file('file')->getRealPath())),
                    'key' => '55488f24a499c325646c862eb83f31d5',
                ],
            ]);

            $responseData = json_decode($response->getBody(), true);
            Log::info('imgbb response data:', ['responseData' => $responseData]);

            if (isset($responseData['data']['display_url'])) {
                return response()->json(['image_url' => $responseData['data']['display_url']]);
            } else {
                return response()->json(['error' => 'Image upload failed'], 500);
            }
        } catch (\Exception $e) {
            Log::error('Error in uploadImage: ' . $e->getMessage());
            return response()->json(['error' => 'Internal server error'], 500);
        }
    } else {
        return response()->json(['error' => 'No file uploaded'], 400);
    }
}

public function saveImage(Request $request)
{
    Log::info('Saving image with data:', $request->all());

    $promoId = $request->input('id_promo');
    $imageUrl = $request->input('image');

    Log::info('Saving image', ['id_promo' => $promoId, 'imageUrl' => $imageUrl]);
```

```
    Promosi_Image::create([
        'id_promo' => $promoId,
        'image' => $imageUrl,
    ]);

    return response()->json(['success' => true]);
}

public function deletePromosi($id_promo)
{
    $promo = Promosi::find($id_promo);

    if(!$promo) {
        abort(404);
    }

    Promosi_Image::where('id_promo', $id_promo)->delete();

    $promo->delete();

    return response()->json(['success' => true]);

}

public function editPromo($id_promo)
{
    $promos = Promosi::select('promosi.*')
        ->where('promosi.id_promo', $id_promo)
        ->get();

    $promo_img = Promosi_Image::select('promosi_image.*')
        ->where('promosi_image.id_promo', $id_promo)
        ->get();

    return view('admin_util/edit_promo', compact('promos', 'promo_img'));
}

public function updatePromo(Request $request, $id_promo)
{
    $promo = Promosi::where('id_promo', $id_promo)->first();
    $promo->update([
        'title' => $request->title
    ]);

    return response()->json(['message' => 'Promo updated successfully']);
}

public function updatePromoImage(Request $request, $id_promo)
{

    $imageUrl = $request->input('imageUrl');
```

```
    $image = Promosi_Image::where('id_promo', $id_promo)->first();
    $image->update([
        'image' => $imageUrl
    ]);

    return response()->json(['message' => 'Promo updated successfully']);
}

public function searchPromo(Request $request)
{
    $search = $request->input('search');
    $dateRange = $request->input('daterange');

    $query = promosi::select('promosi.*', 'promosi_image.*')
        ->join('promosi_image', 'promosi.id_promo', '=', 'promosi_image.id_promo')
        ->orderBy('promosi.title', 'asc');

    if ($search) {
        $query->where('promosi.title', 'like', '%' . $search . '%');
    }

    if ($dateRange) {
        $dates = explode(' - ', $dateRange);
        $startDate = \Carbon\Carbon::createFromFormat('d/m/Y', $dates[0])->startOfDay();
        $endDate = \Carbon\Carbon::createFromFormat('d/m/Y', $dates[1])->endOfDay();

        $query->whereBetween('promosi.created_at', [$startDate, $endDate])
            ->orWhereBetween('promosi.ended_at', [$startDate, $endDate]);
    }

    $data = $query->paginate(10);

    return view('admin_util.promosi', ['promosi' => $data]);
}

public function searchPastPromo(Request $request)
{
    $search = $request->input('search');
    $dateRange = $request->input('daterange');

    $query = promosi::select('promosi.*', 'promosi_image.*')
        ->join('promosi_image', 'promosi.id_promo', '=', 'promosi_image.id_promo')
        ->where('is_expired', 1);

    if ($search) {
        $query->where('promosi.title', 'like', '%' . $search . '%');
    }

    if ($dateRange) {
        $dates = explode(' - ', $dateRange);
```

```
            $startDate = \Carbon\Carbon::createFromFormat('d/m/Y', $dates[0])->startOfDay();
            $endDate = \Carbon\Carbon::createFromFormat('d/m/Y', $dates[1])->endOfDay();

            $query->where(function ($q) use ($startDate, $endDate) {
                $q->whereBetween('promosi.created_at', [$startDate, $endDate])
                ->orWhereBetween('promosi.ended_at', [$startDate, $endDate]);
            });
        }

        $data = $query->orderBy('promosi.title', 'asc')->paginate(10, ['*'], 'expiredPage');

        return view('admin_util.promosi_history', ['promosi' => $data]);
    }
}
```

4.8 Halaman Pengaturan Testimoni

Administrator dapat menggunakan halaman pengaturan testimoni untuk membantu mereka menambahkan testimoni ke bagian testimoni halaman depan. Pada halaman ini, mereka dapat menambah, menyunting, dan menghapus testimoni yang sudah ada, dan mereka juga dapat mencari testimoni berdasarkan nama atau judulnya. Bagian penambahan testimoni ditunjukkan dalam Segmen Program 4.27; Bagian pencarian dan penghapusan testimoni ditunjukkan dalam Segmen Program 4.28; dan Bagian penyuntingan gambar dan data testimoni ditunjukkan dalam Segmen Program 4.29. Terakhir, Segmen Program 4.30 menjelaskan pertanyaan antara *database* dan *website*.

Tabel 4.16 Tabel Segmen Program dan *Flowchart* Penambahan Testimoni

Segmen Program	Flowchart
Segmen Program 4.27	Gambar 3.9
Segmen Program 4.28	
Segmen Program 4.29	
Segmen Program 4.30	

Segmen Program 4.27 Penambahan Testimoni

```
<form method="POST" class="add-testimoni">
        @csrf
        <div class="row">
          <div class="col add-image">
            <h3>Foto</h3>
            <img id="imagePreview" class="image-preview" src="" alt="Image Preview">
            <input type="file" accept=".jpg" name="image" id="image" class="form-control"
onchange="previewImage(event)">
          </div>
        </div>
        <div class="row">
          <div class="col add-title">
            <h3>Nama / Judul Testimoni</h3>
            <input type="text" name="name" class="form-control title">
          </div>
        </div>
        <div class="row" style="margin-bottom: 64px;">
          <div class="col add-desc">
            <h3>Keterangan</h3>
            <textarea name="description" class="form-control description"></textarea>
          </div>
        </div>
        <div class="input-cat">
          <h3>Kategori</h3>
          <div class="cat-btn">
            <select name="category" class="form-select" id="multiple-select-field" data-
placeholder="Pilih Kategori" multiple>
              <option value="14">Adat</option>
              <option value="6">Aksesoris</option>
              <option value="12">Animal</option>
              <option value="15">Apron</option>
              <option value="7">Bridal Sration</option>
              <option value="19">Cosplay</option>
              <option value="16">Futuristik</option>
              <option value="10">Halloween</option>
              <option value="11">Hero</option>
              <option value="3">Internasional</option>
              <option value="4">Karakter</option>
              <option value="18">Model</option>
              <option value="9">Natal</option>
              <option value="13">Onesie</option>
              <option value="20">Prince</option>
              <option value="2">Princess</option>
              <option value="5">Profesi</option>
              <option value="1">Sayurbuah</option>
              <option value="17">Old</option>
            </select>
          </div>
        </div>
```

```
        <div class="input-theme">
          <h3>Tema</h3>
          <div class="theme-btn">
            <select name="theme" class="form-select" id="multiple-select-field-theme" data-
placeholder="Pilih Tema" multiple>
              <option value="1">Ancient Civilization / Peradaban Kuno</option>
              <option value="2">Medieval / Renaissance / Abad Pertengahan</option>
              <option value="3">90's</option>
              <option value="4">Wild West</option>
              <option value="5">Film dan Acara TV</option>
              <option value="6">Anime</option>
              <option value="7">Superhero dan Penjahat</option>
              <option value="8">Ikon Musik</option>
              <option value="9">Karakter Video Game</option>
              <option value="10">Karakter Kartun</option>
              <option value="11">Karakter Cerita / Dongeng</option>
              <option value="12">Makhluk Mitos</option>
              <option value="13">Penyihir</option>
              <option value="14">Bajak Laut</option>
              <option value="15">Putri Duyung</option>
              <option value="16">Vampir dan Manusia Serigala</option>
              <option value="17">Steampunk</option>
              <option value="18">Dokter dan Perawat</option>
              <option value="19">Polisi</option>
              <option value="20">Pemadam Kebakaran</option>
              <option value="21">Militer</option>
              <option value="22">Pelaut</option>
              <option value="23">Ilmuwan</option>
              <option value="24">Pilot dan Pramugari</option>
              <option value="25">Hewan</option>
            </select>
          </div>
        </div>
        <div class="row submit-btn-ctr">
          <div class="submit-btn">
            <button type="submit" class="tambah">
              Tambahkan Testimoni
              <i class="fas fa-check"></i>
            </button>
          </div>
        </div>
      </form>
```

Segmen Program 4.28 Pencarian Testimoni

```
@section('main')
  <div class="container">
    <div class="row search-bar">
      <h1>Daftar Testimoni</h1>
      <div class="col">
        <div class="add-ctr">
          <a class="add-btn" href="{{ route('testimoni.tambahTestimoni') }}">
            Tambah
            <i class="fas fa-plus"></i>
          </a>
        </div>
      </div>
    </div>
    <div class="row table-testimoni">
      <div class="table-kostum">
        <table class="table table-bordered" id="table">
          <thead>
            <tr>
              <th>Foto</th>
              <th>Keterangan</th>
              <th>Judul</th>
              <th>Aksi</th>
            </tr>
          </thead>
          <tbody>

          </tbody>
        </table>
      </div>
    </div>
  </div>
@endsection
@section('script')
  <script type="text/javascript">
    $(function () {
      var table = $('#table').DataTable({
        processing: true,
        serverside: true,
        ajax: {
          url: "{{ route('testimoni.testimoni') }}",
        },
        columns: [
          { data: 'imageUrl', name: 'Foto',
            render: function (data, type, row, meta) {
              return '<img src="'+data+'" style="max-width: 100px;">'
            }
          },
          { data: 'name', name: 'Judul'},
          { data: 'description', name: 'Keterangan' },
          { data:'action', name:'action'}
```

Segmen Program 4.29 Penyuntingan Gambar dan Data Testimoni

```
<form method="POST" class="add-testimoni">
        @csrf
        @foreach ($testimonies as $testimoni)
          <div class="row">
            <div class="col add-image">
              <h3>Foto Baru</h3>
              <img id="imagePreview" class="image-preview" src="{{ $testimoni->imageUrl }}"
alt="Image Preview">
              <input type="file" accept=".jpg" name="image" id="image" class="form-control"
onchange="previewImage(event)">
            </div>
          </div>
          <div class="row">
            <div class="col add-title">
              <h3>Nama / Judul Edit</h3>
              <input type="text" name="name" class="title" value="{{ $testimoni->name }}">
            </div>
          </div>
          <div class="row" style="margin-bottom: 64px;">
            <div class="col add-desc">
              <h3>Keterangan</h3>
              <textarea name="description" class="description">{{ $testimoni->description
}}</textarea>
            </div>
          </div>
          <div class="preview-category-box">
            @foreach ($testimoni_kategori as $tk)
              @foreach ($categories as $category)
                @if ($tk->id_kategori == $category->id_category)
                  <div class="tags" id="{{ $category->id_category }}">
                    <button class="remove-category" data-testi-category="{{ $tk-
>id_testimoni_kategori }}" data-csrf="{{ csrf_token() }}">
                      <i class="fas fa-times"></i>
                    </button>
                    <span class="tag" id="{{ $category->id_category }}">{{ $category->category
}}</span>
                  </div>
                @endif
              @endforeach
            @endforeach
          </div>
          <div class="input-cat">
            <h3>Kategori</h3>
            <div class="cat-btn">
              <select name="category" class="form-select" id="multiple-select-field" data-
placeholder="Pilih Kategori" multiple>
                <option value="14">Adat</option>
                <option value="6">Aksesoris</option>
                <option value="12">Animal</option>
                <option value="15">Apron</option>
                <option value="7">Bridal Sration</option>
                <option value="19">Cosplay</option>
                <option value="16">Futuristik</option>
                <option value="10">Halloween</option>
```

```
                    <option value="11">Karakter Cerita / Dongeng</option>
                    <option value="12">Makhluk Mitos</option>
                    <option value="13">Penyihir</option>
                    <option value="14">Bajak Laut</option>
                    <option value="15">Putri Duyung</option>
                    <option value="16">Vampir dan Manusia Serigala</option>
                    <option value="17">Steampunk</option>
                    <option value="18">Dokter dan Perawat</option>
                    <option value="19">Polisi</option>
                    <option value="20">Pemadam Kebakaran</option>
                    <option value="21">Militer</option>
                    <option value="22">Pelaut</option>
                    <option value="23">Ilmuwan</option>
                    <option value="24">Pilot dan Pramugari</option>
                    <option value="25">Hewan</option>
                  </select>
                </div>
              </div>
              <div class="row submit-btn-ctr">
                <div class="submit-btn">
                  <button type="submit" value="{{ $testimoni->id_testimoni }}" class="edit">
                    Edit Testimoni
                    <i class="fas fa-check"></i>
                  </button>
                </div>
              </div>
            @endforeach
          </form>

        <script>
          function previewImage(event) {
            const reader = new FileReader();
            reader.onload = function() {
              const output = document.getElementById('imagePreview');
              output.src = reader.result;
            }
            reader.readAsDataURL(event.target.files[0]);
          }

          $('#multiple-select-field').select2( {
            theme: "bootstrap-5",
            width: $( this ).data( 'width' ) ? $( this ).data( 'width' ) : $( this ).hasClass( 'w-100' ) ? '100%'
: 'style',
            placeholder: $( this ).data( 'placeholder' ),
            closeOnSelect: false,
          });

          $('#multiple-select-field-theme').select2( {
            theme: "bootstrap-5",
            width: $( this ).data( 'width' ) ? $( this ).data( 'width' ) : $( this ).hasClass( 'w-100' ) ? '100%'
: 'style',
            placeholder: $( this ).data( 'placeholder' ),
            closeOnSelect: false,
          });

          let removeCatId = [];
```

```
let removeThemeId = [];

$(document).ready(function() {
  $('.remove-category').each(function() {
    $(this).on('click', function(e) {
      e.preventDefault();
      console.log("click")
      var id_testi_cat = $(this).data('testi-category');
      removeCatId.push(id_testi_cat);
      console.log("id cat:", removeCatId);

      $(this).closest('.tags').remove();

      var token = $(this).data('csrf');

      $.each(removeCatId, function(index, value) {
        console.log(removeCatId)
        deleteCategory(value, token).fail(function(jqXHR, textStatus, errorThrown) {
          console.log('Error deleting category:', errorThrown);
        });
      });
    });
  });

  $('.remove-theme').each(function() {
    $(this).on('click', function(e) {
      e.preventDefault();
      console.log("click")
      var id_testi_theme = $(this).data('testi-theme');
      removeThemeId.push(id_testi_theme);
      console.log("id theme:", removeThemeId);

      $(this).closest('.tags').remove();

      var token = $(this).data('csrf');

      $.each(removeThemeId, function(index, value) {
        deleteTheme(value, token).fail(function(jqXHR, textStatus, errorThrown) {
          console.log('Error deleting theme:', errorThrown);
        });
      });
    });
  });

  $('.add-testimoni').on('submit', function (e) {
    e.preventDefault();

    var file = $(this).find('#image')[0].files[0];
    console.log(file);
```

```
            var select2Values = $('#multiple-select-field').val();
            console.log('Selected values:', select2Values);

            var themeValues = $('#multiple-select-field-theme').val();
            console.log('Selected themes:', themeValues);

            let formData2 = new FormData(this);

            if(file) {
               let testimoniId = $(this).find('button.edit').val()
               console.log(testimoniId);

               var formData = new FormData();
               formData.append('file', file);
               formData.append('_token', '{{ csrf_token() }}');

               $.ajax({
                  url: '{{ route('testimoni.uploadImage') }}',
                  method: 'POST',
                  data: formData,
                  processData: false,
                  contentType: false,
                  success: function(response) {
                     console.log(response.image_url);
                     formData2.append('_token', '{{ csrf_token() }}');
                     formData2.append('imageUrl', response.image_url);

                     if(response.image_url) {
                        $.ajax({
                           url: '{{ route("testimoni.updateTestimoni", ":id_testimoni")
}}'.replace(':id_testimoni', testimoniId),
                           method: 'POST',
                           data: formData2,
                           contentType: false,
                           processData:false,
                           success: function () {
                              console.log('success');
                              addCategory(testimoniId, select2Values);
                              addThemes(testimoniId, themeValues);
                              window.location.reload(); // Reload the page
                           },
                           error: function(xhr, status, error) {
                              console.error('Promo update error:', error);
                           }
                        });
                     } else {
                        console.error("Failed to upload image");
                     }
                  },
                  error: function(xhr, status, error) {
                     console.error('Data upload error:', error);
                  }
```

```
                }
            });

        } else {
            let testimoniId = $(this).find('button.edit').val()
            console.log(testimoniId);

            $('.image-preview').each(function() {
                var imageSrc = $(this).attr('src');
                console.log("Image src:", imageSrc);

                formData2.append('imageUrl', imageSrc);

                if(imageSrc) {
                    $.ajax({
                        url: '{{ route("testimoni.updateTestimoni", ":id_testimoni")
}}'.replace(':id_testimoni', testimoniId),
                        method: 'POST',
                        data: formData2,
                        contentType: false,
                        processData:false,
                        success: function () {
                            console.log('success');
                            addCategory(testimoniId, select2Values);
                            addThemes(testimoniId, themeValues);
                            window.location.reload(); // Reload the page
                        },
                        error: function(xhr, status, error) {
                            console.error('Testimoni update error:', error);
                        }
                    });
                } else {
                    console.error("No picture to update");
                }
            });
        }
    })

    function addCategory(testiId, selectValues) {
        if (selectValues.length > 0) {
            $.ajax({
                url: '{{ route('testimoni.addTestimoniCategory') }}',
                method: 'POST',
                data: {
                    id_testimoni: testiId,
                    categories: selectValues,
                    _token: '{{ csrf_token() }}'
                },
                success: function () {
                    console.log('Category added to database');
                    resetForm();
                `
```

```
                },
                error: function (xhr, status, error) {
                    console.error('Category addition error:', error);
                }
            });
        }
    }

    function addThemes(testId, selectValues) {
        if (selectValues.length > 0) {
            $.ajax({
                url: '{{ route('testimoni.addTestimoniTheme') }}',
                method: 'POST',
                data: {
                    id_testimoni: testId,
                    theme: selectValues,
                    _token: '{{ csrf_token() }}'
                },
                success: function () {
                    console.log('Theme added to database');
                    resetForm();
                },
                error: function (xhr, status, error) {
                    console.error('Theme addition error:', error);
                    alert('Theme addition error: ' + error);
                }
            });
        }
    }

    function deleteCategory(ccId, token) {
        console.log("cateId:", ccId);
        return $.ajax({
            type: 'DELETE',
            url: '{{ route("testimoni.deleteTestimoniCategory", ":id_testimoni_kategori")
}}'.replace(':id_testimoni_kategori', ccId),
            headers: {
                'X-CSRF-TOKEN': token
            },
            success: function(response) {
                console.log('Category deleted successfully');
            },
            error: function(error) {
                console.log('Error deleting category:', error);
                throw error;
            }
        });
    }

    function deleteTheme(ccId, token) {
```

```
            return $.ajax({
                type: 'DELETE',
                url: '{{ route("testimoni.deleteTestimoniTheme", ":id_testimoni_tema")
}}'.replace(':id_testimoni_tema', ccId),
                headers: {
                    'X-CSRF-TOKEN': token
                },
                success: function(response) {
                    console.log('Theme deleted successfully');
                },
                error: function(error) {
                    console.log('Error deleting theme:', error);
                    throw error;
                }
            });
        }

        // Function to reset the form
        function resetForm() {
            // Clear Select2 selected values
            $('#multiple-select-field').val(null).trigger('change');
            $('#multiple-select-field-theme').val(null).trigger('change');
        }
    });
</script>
```

4.9 Sistem *Navigation Bar*

Sistem *navigation bar* digunakan oleh pengguna atau pengunjung umum untuk berpindah dari satu halaman ke halaman lainnya. Berikut ini ada penjelasan sistem *navigation bar* dalam bentuk Segmen Program 4.30. Selain itu pada *navigation bar* juga akan ditambahkan dengan sebuah fitur menambahkan kostum yang dianggap menarik oleh pengujung kedalam sebuah list yang dapat dilihat pada menu *navigation bar*.

Tabel 4.17 Tabel Segmen Program dan *Flowchart Navigation Bar*

Segmen Program	*Flowchart*
Segmen Program 4.27	Gambar 3.11

Segmen Program 4.30 Sistem *Navigation Bar*

```
<nav class="navbar navbar-expand-lg navbar-light">
        <div class="container-fluid">
          <a class="navbar-brand" href="#"><img src="/img/logo-gk.png" alt="Logo"
style="height: 40px;"></a>
          <div class="collapse navbar-collapse justify-content-center" id="navbarNav">
           <form class="d-flex search" action="{{ route('client.katalogSearch') }}" id="search-bar"
method="GET">
            <i class="fas fa-magnifying-glass icon"></i>
            <input type="text" class="form-control main-search" name="search"
placeholder="search" value="{{ request('search') }}">
            <input type="hidden" name="category" value="{{ request('category') }}">

           </form>
            <ul class="navbar-nav ms-auto d-none d-lg-flex">
              @guest
                @if (Route::has('login'))
                  <li class="nav-item" style="font-weight: bold;">
                    <a class="nav-link" href="{{ route('login') }}" @click.prevent="showLogin">{{
__('Login') }}</a>
                  </li>
                @endif

                @if (Route::has('register'))
                  <li class="nav-item" style="font-weight: bold;">
                    <a class="nav-link" href="{{ route('register') }}"
@click.prevent="showRegister">{{ __('Register') }}</a>
                  </li>
                @endif
              @else
                <li class="nav-item dropdown">
                  <a id="navbarDropdown" class="nav-link dropdown-toggle" href="#"
role="button" data-bs-toggle="dropdown" aria-haspopup="true" aria-expanded="false" v-pre>
                    {{ Auth::user()->name }}
                  </a>
                  <div class="dropdown-menu dropdown-menu-end" aria-
labelledby="navbarDropdown">
                    <a class="dropdown-item" href="{{ route('logout') }}"
                      onclick="event.preventDefault();
                            document.getElementById('logout-form').submit();">
                      {{ __('Logout') }}
                    </a>

                    <form id="logout-form" action="{{ route('logout') }}" method="POST"
class="d-none">
                      @csrf
                    </form>
                  </div>
                </li>
              @endguest
```

```
                        <div class="dropdown-menu dropdown-menu-end" aria-
labelledby="navbarDropdown">
                            <a class="dropdown-item" href="{{ route('logout') }}"
                              onclick="event.preventDefault();
                                    document.getElementById('logout-form').submit();">
                              {{ __('Logout') }}
                            </a>

                            <form id="logout-form" action="{{ route('logout') }}" method="POST"
class="d-none">
                               @csrf
                            </form>
                          </div>
                        </li>
                      @endguest

                        <a class="dropdown-toggle" id="dropdownMenuButton1" data-bs-
toggle="dropdown" aria-expanded="false" style="text-decoration: none; color: black;">
                          <i class="fas fa-heart" style="padding-top: 13px;"></i>
                        </a>
                        <ul class="dropdown-menu dropdown-menu-end px-4" role="menu"
style="width: 20rem;" id="likedCostumesDropdown">

                        </ul>
                       </div>
                      </li>
                      <li class="nav-item brand">
                        <a class="nav-link" href="https://wa.me/6285215003507"><i class="fa-brands
fa-whatsapp"></i></a>
                      </li>
                      <li class="nav-item brand">
                        <a class="nav-link"
href="https://www.instagram.com/gudangkostum_surabaya/"><i class="fa-brands fa-
instagram"></i></a>
                      </li>
                      <li class="nav-item brand">
                        <a class="nav-link"
href="https://www.facebook.com/gudangkostumsurabaya"><i class="fa-brands fa-
facebook"></i></a>
                      </li>
                    </ul>
                  </div>
                </div>
              </nav>
```

```
<nav class="navbar navbar-expand-lg navbar-light" style="border-top: 1px solid black; border-bottom: 1px solid black;">
        <div class="container-fluid">
          <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarTogglerDemo01" aria-controls="navbarTogglerDemo01" aria-expanded="false" aria-label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarTogglerDemo01">
            <ul class="navbar-nav me-auto mb-2 mb-lg-0">
              <li class="nav-item">
                <a class="nav-link" href="{{ route('client.homepage') }}">Home</a>
              </li>
              <li class="nav-item">
                <a class="nav-link" href="{{ route('client.katalog') }}">Katalog</a>
              </li>
              <li class="nav-item">
                <a class="nav-link" href="{{ route('client.siapaKami') }}">Siapa Kami</a>
              </li>
              <li class="nav-item">
                <a class="nav-link" href="{{ route('client.getClientTestimoni') }}">Testimoni</a>
              </li>
              <li class="nav-item">
                <a class="nav-link" href="{{ route('client.ketentuanSewa') }}">Ketentuan</a>
              </li>
            </ul>
            <ul class="navbar-nav ms-auto d-lg-none">
                @guest
                    @if (Route::has('login'))
                        <li class="nav-item" style="font-weight: bold;">
                            <a class="nav-link" href="{{ route('login') }}" @click.prevent="showLogin">{{ __('Login') }}</a>
                        </li>
                    @endif

                    @if (Route::has('register'))
                        <li class="nav-item" style="font-weight: bold;">
                            <a class="nav-link" href="{{ route('register') }}" @click.prevent="showRegister">{{ __('Register') }}</a>
                        </li>
                    @endif
                @else
                    <li class="nav-item dropdown">
                        <a id="navbarDropdown" class="nav-link dropdown-toggle" href="#" role="button" data-bs-toggle="dropdown" aria-haspopup="true" aria-expanded="false" v-pre>
                            {{ Auth::user()->name }}
                        </a>
                        <div class="dropdown-menu dropdown-menu-end" aria-labelledby="navbarDropdown">
                            <a class="dropdown-item" href="{{ route('logout') }}"
```

```
                        onclick="event.preventDefault();
                                document.getElementById('logout-form').submit();">
                        {{ __('Logout') }}
                    </a>

                    <form id="logout-form" action="{{ route('logout') }}" method="POST" class="d-
none">
                        @csrf
                    </form>
                  </div>
                </li>
              @endguest
              {{-- <li class="nav-item chat">
                <a class="nav-link" href="#" id="chatLink"><i class="fas fa-comment-
dots"></i>Chat</a>
              </li> --}}
              <li class="nav-item brand">
                <div class="btn-group">
                  {{-- <button class="btn btn-secondary dropdown-toggle" type="button"
id="dropdownMenuButton1" data-bs-toggle="dropdown" aria-expanded="false">
                    <i class="fas fa-heart"></i>
                  </button> --}}
                  <a class="dropdown-toggle" id="dropdownMenuButton1" data-bs-
toggle="dropdown" aria-expanded="false" style="text-decoration: none; color: black;">
                    <i class="fas fa-heart" style="padding-top: 13px;"></i>
                  </a>
                  <ul class="dropdown-menu dropdown-menu-end px-4" role="menu" style="width:
20rem;" id="likedCostumesDropdown">

                  </ul>
                </div>
              </li>
              <li class="nav-item brand">
                <a class="nav-link" href="https://wa.me/6285215003507"><i class="fa-brands fa-
whatsapp"></i>WhatsApp</a>
              </li>
              <li class="nav-item brand">
                <a class="nav-link"
href="https://www.instagram.com/gudangkostum_surabaya?utm_source=ig_web_button_share
_sheet&igsh=ZDNlZDc0MzIxNw==><i class="fa-brands fa-instagram"></i>Instagram</a>
              </li>
              <li class="nav-item brand">
                <a class="nav-link" href="https://www.facebook.com/gudangkostumsurabaya"
target="_blank">
                  <i class="fa-brands fa-facebook"></i>
                  Facebook
                </a>
              </li>
              <form class="d-flex search" action="{{ route('client.katalogSearch') }}"
method="GET">
```

```
                <i class="fas fa-magnifying-glass icon"></i>
                <input type="text" class="form-control main-search" name="search"
placeholder="search" value="{{ request('search') }}">
                <input type="hidden" name="category" value="{{ request('category') }}">
               </form>
             </ul>
            </div>
           </div>
          </nav>
         </header>
```

4.10 Halaman Katalog Pengunjung

Halaman ini adalah halaman yang akan sering dikunjungi oleh para pengunjung untuk melihat kostum yang tersedia pada Toko Rental XYZ. Pada bagian ini terdapat kotak pencarian dan penyaringan berdasarkan ukuran dan kategori. Halaman ini akan dijelaskan dalam bentuk Segmen Program 4.28 untuk tampilan katalog beserta bagian isi katalog, Segmen Program 4.29 untuk pengambilan data katalog, dan Segmen Program 4.30 untuk pencarian data kostum.

Tabel 4.18 Tabel Segmen Program dan *Flowchart* Halaman Katalog

Segmen Program	Flowchart
Segmen Program 4.28	Gambar 3.12
Segmen Program 4.29	
Segmen Program 4.30	

Segmen Program 4.31 Tampilan Halaman Katalog

```
<div class="container">
    <div class="row">
      <div class="col breadcrumbs">
        <nav style="--bs-breadcrumb-divider: '>';" aria-label="breadcrumb">
          <ol class="breadcrumb">
            <li class="breadcrumb-item"><a href="{{ route('client.homepage') }}">Home</a></li>
            <li class="breadcrumb-item active" aria-current="page">Katalog</li>
          </ol>
        </nav>
      </div>
```

```
<div class="row">
        <div class="col-md-3 filter-kategori">
          <label>Kategori :</label>
          <select class="form-select kostum" name="category" id="category-
filter">
            <option value="" default>Semua Kategori</option>
            <option value="14">Adat</option>
            <option value="6">Aksesoris</option>
            <option value="12">Animal</option>
            <option value="15">Apron</option>
            <option value="7">Bridal Station</option>
            <option value="19">Cosplay</option>
            <option value="16">Futuristik</option>
            <option value="10">Halloween</option>
            <option value="11">Hero</option>
            <option value="3">Internasional</option>
            <option value="4">Karakter</option>
            <option value="18">Model</option>
            <option value="9">Natal</option>
            <option value="13">Onesie</option>
            <option value="20">Prince</option>
            <option value="2">Princess</option>
            <option value="5">Profesi</option>
            <option value="1">Sayurbuah</option>
            <option value="17">Old</option>
          </select>
        </div>
        <div class="col-md-3 filter-ukuran">
          <label>Ukuran :</label>
          <select name="size" id="sizes" class="form-select">
            <option value="">Pilih Ukuran</option>
            <option value="semua ukuran">Semua Ukuran</option>
            <option value="" disabled></option>
            <hr>
            <option value="" disabled></option>
            <option value="Bayi" disabled style="font-weight:
bold;">Bayi</option>
            <option value="" disabled></option>
            <option value="3m">3 - 6 Bulan</option>
            <option value="6m">6 - 9 Bulan</option>
            <option value="9m">9 - 12 Bulan</option>
            <option value="12m">12 - 18 Bulan</option>
            <option value="18m">18 - 24 Bulan</option>
```

```
            <option value="" disabled></option>
            <hr>
            <option value="" disabled></option>
            <option value="balita" disabled style="font-weight:
bold;">Balita</option>
            <option value="" disabled></option>
            <option value="2t">2 Tahun</option>
            <option value="3t">3 Tahun</option>
            <option value="4t">4 Tahun</option>
            <option value="5t">5 Tahun</option>
            <option value="" disabled></option>
            <hr>
            <option value="" disabled></option>
            <option value="anak" disabled style="font-weight:
bold;">Anak</option>
            <option value="" disabled></option>
            <option value="xs-a">4 Tahun</option>
            <option value="s-a">5 - 6 Tahun</option>
            <option value="m-a">7 - 8 Tahun</option>
            <option value="l-a">9 - 12 Tahun</option>
            <option value="" disabled></option>
            <hr>
            <option value="" disabled></option>
            <option value="remaja" disabled style="font-weight:
bold;">Remaja</option>
            <option value="" disabled></option>
            <option value="xs-r">12 - 13 Tahun</option>
            <option value="s-r">14 - 15 Tahun</option>
            <option value="m-r">15 - 16 Tahun</option>
            <option value="l-r">16 - 18 Tahun</option>
            <option value="" disabled></option>
            <hr>
            <option value="" disabled></option>
            <option value="dewasa" disabled style="font-weight: bold;">Dewasa
Wanita</option>
            <option value="" disabled></option>
            <option value="xs-w">XS</option>
            <option value="s-w">S</option>
            <option value="m-w">M</option>
            <option value="l-w">L</option>
            <option value="xl-w">XL</option>
            <option value="" disabled></option>
            <hr>
            <option value="" disabled></option>
            <option value="dewasa" disabled style="font-weight: bold;">Dewasa
```

```
            <option value="" disabled></option>
            <option value="xs-p">XS</option>
            <option value="s-p">S</option>
            <option value="m-p">M</option>
            <option value="l-p">L</option>
            <option value="xl-p">XL</option>
            <option value="" disabled></option>
          </select>
        </div>
        <div class="col-md-3 filter-tema">
          <label>Tema :</label>
          <select name="themes" id="themes" class="form-select">
            <option value="">Semua Tema</option>
            {{-- @if(isset($themes) && count($themes) > 0)
              @foreach ($themes as $index => $theme)
                <option value="{{ $theme->id_theme }}">{{ $theme->theme
}}</option>
              @endforeach
            @else
              <option value="">Tema tidak tersedia</option>
            @endif --}}
            <option value="1">Ancient Civilization / Peradaban Kuno</option>
            <option value="2">Medieval / Renaissance / Abad
Pertengahan</option>
            <option value="3">90's</option>
            <option value="4">Wild West</option>
            <option value="5">Film dan Acara TV</option>
            <option value="6">Anime</option>
            <option value="7">Superhero dan Penjahat</option>
            <option value="8">Ikon Musik</option>
            <option value="9">Karakter Video Game</option>
            <option value="10">Karakter Kartun</option>
            <option value="11">Karakter Cerita / Dongeng</option>
            <option value="12">Makhluk Mitos</option>
            <option value="13">Penyihir</option>
            <option value="14">Bajak Laut</option>
            <option value="15">Putri Duyung</option>
            <option value="16">Vampir dan Manusia Serigala</option>
            <option value="17">Steampunk</option>
            <option value="18">Dokter dan Perawat</option>
            <option value="19">Polisi</option>
            <option value="20">Pemadam Kebakaran</option>
```

```
            <option value="21">Militer</option>
            <option value="22">Pelaut</option>
            <option value="23">Ilmuwan</option>
            <option value="24">Pilot dan Pramugari</option>
            <option value="25">Hewan</option>
          </select>
        </div>
        <div class="col-md-3 filter-color">
          <label>Warna :</label>
          <select name="colors" id="colors" class="form-select">
            <option value="">Semua Warna</option>
            {{-- @if(isset($colors) && count($colors) > 0)
              @foreach ($colors as $index => $color)
                <option value="{{ $color->id_color }}">{{ $color->color }}</option>
              @endforeach
            @else
              <option value="">Warna tidak tersedia</option>
            @endif --}}
            <option value="1">Merah</option>
            <option value="2">Oranye / Jingga</option>
            <option value="3">Kuning</option>
            <option value="4">Hijau</option>
            <option value="5">Biru</option>
            <option value="6">Ungu</option>
            <option value="7">Pink</option>
            <option value="8">Cokelat</option>
            <option value="9">Hitam</option>
            <option value="10">Putih</option>
            <option value="11">Abu-abu</option>
          </select>
        </div>
      </div>
      <div class="row katalog">
        @if ($noResults)
          <p>No results found.</p>
        @else
          <div class="cards-wrapper" id="katalog-container">
            @include('partials.costumeClientItems', ['katalogs' => $katalogs,
'noResults' => $noResults])
          </div>
        @endif
        <div class="d-flex justify-content-center mt-4" id="pagination-container">
          {{ $katalogs->withQueryString()->links('pagination::bootstrap-5') }}
```

Segmen Program 4.31 Pengambilan Data Kostum untuk Katalog

```
public function getCostume()
   {
      $data = costume::select('costume.*',
         DB::raw('GROUP_CONCAT(DISTINCT category.category SEPARATOR '', '') AS categories'),
         DB::raw('(SELECT imageUrl FROM image WHERE image.id_costume = costume.id_costume
LIMIT 1) as imageUrl'))
         ->join('costume_category', 'costume.id_costume', '=', 'costume_category.id_costume')
         ->join('category', 'costume_category.id_category', '=', 'category.id_category')
         ->groupBy('costume.id_costume', 'costume.name', 'costume.size', 'costume.price',
'costume.description', 'costume.views', 'costume.interest')
         ->orderBy('costume.id_costume', 'asc')
         ->paginate(12);

      $noResults = $data->isEmpty(); // Check if the data is empty

      return view('client_util/katalog_client', ['katalogs' => $data, 'noResults' => $noResults]);
   }
```

Segmen Program 4.32 Pencarian Data Kostum

```
public function filterCostume(Request $request)
   {
      $size = $request->input('size');
      $category = $request->input('category');
      $search = $request->input('search'); // Get the search input
      $theme = $request->input('theme');
      $color = $request->input('color');

      // $tema = theme::select('theme.*')->get();
      // Log::info("Received theme query:", ['tema' => $tema]);

      $searchHistory = search_history::create([
         'search' => $search,
         'id_category' => $category,
         'id_theme' => $theme,
         'id_color' => $color,
      ]);

      // Check if the search history was successfully created and log the result
      if ($searchHistory) {
         Log::info('Search history added successfully:', [
            'search' => $search,
            'id_category' => $category,
            'id_theme' => $theme,
            'id_color' => $color
         ]);
      } else {
         Log::error('Failed to add search history:', [
```

```
    Log::info('Received search query:', ['search' => $search, 'category' => $category]);

    $query = Costume::select('costume.*',
        DB::raw('(SELECT imageUrl FROM image WHERE image.id_costume = costume.id_costume
LIMIT 1) as imageUrl'),
        DB::raw('GROUP_CONCAT(DISTINCT category.category SEPARATOR ", ") AS categories'))
        ->join('costume_category', 'costume.id_costume', '=', 'costume_category.id_costume')
        ->join('category', 'costume_category.id_category', '=', 'category.id_category')
        ->join('costume_theme', 'costume.id_costume', '=', 'costume_theme.id_costume')
        ->join('theme', 'costume_theme.id_theme', '=', 'theme.id_theme')
        ->join('costume_color', 'costume.id_costume', '=', 'costume_color.id_costume') // Added
this join
        ->join('color', 'costume_color.id_color', '=', 'color.id_color');

    if ($size) {
        $query->where('costume.size', '=', $size);
    }

    if ($category) {
        $query->where(function($query) use ($category) {
            $query->where('category.id_category', '=', $category)
                ->orWhere('category.category', '=', $category);
        });
    }

    if ($theme) {
        $query->where('theme.id_theme', '=', $theme);
    }

    if($color) {
        $query->where('color.id_color', '=', $color);
    }

    if ($search) {
        $query->where(function($query) use ($search) {
            $query->where('costume.name', 'like', '%' . $search . '%')
                ->orWhere('costume.description', 'like', '%' . $search . '%');
        });
    }

    $data = $query->groupBy('costume.id_costume', 'costume.name', 'costume.size',
'costume.price', 'costume.description', 'costume.views', 'costume.interest')
        ->orderBy('costume.id_costume', 'asc')
        ->paginate(12);

    $noResults = $data->isEmpty(); // Check if the data is empty
```

```
        if ($request->ajax()) {
            return response()->json([
                'katalogs' => view('partials.costumeClientItems', ['katalogs' => $data, 'noResults' =>
$noResults])->render(),
                'pagination' => $data->links('pagination::bootstrap-5')->render()
            ]);
        }

        return view('client_util.katalog_client', ['katalogs' => $data, 'noResults' => $noResults]);
    }
```

4.11 Halaman Tentang Kami

Halaman ini menunjukkan keterangan mengenai Toko Rental XYZ itu sendiri. Halaman ini menjelaskan tentang jasa yang telah diberikan oleh Toko Rental XYZ, serta kontak mereka melalui *Instagram*. Penjelasan mengenai implementasi sistem ini akan dijelaskan melalui Segmen Program 4.31.

Tabel 4.19 Tabel Segmen Program dan *Flowchart* Halaman Tentang Kami

Segmen Program	*Flowchart*
Segmen Program 4.31	Gambar 3.13

Segmen Program 4.33 Halaman Tentang Kami

```
<div class="container">
    <div class="logo">
        <img src="{{ asset('public/img/logo-gk.png') }}">
    </div>
</div>
<div class="container">
    <div class="desc">
        <h1>Tentang Kami</h1>
        <p>Gudang Kostum adalah Costume Center yang berpusat di kota Surabaya, yang menjual
dan menyewakan aneka kostum pilihan berkualitas, lengkap, bersih, dan terjangkau.
            Gudang Kostum sudah dipercaya banyak lembaga Perusahaan maupun Pemerintahan,
Sekolah, maupun pribadi yang sedang mengadakan ACARA-ACARA spesial seperti Gala Dinner
perusahaan,
            Ulang tahun perusahaan, Kostum untuk Ulang Tahun dan Perayaan lainnya, Acara 1001
Malam di Akhir Tahun, Lomba Fashion anak, Modelling, dan juga kostum Fotografi.
            Dengan menjaga kualitas layanan, Gudang Kostum telah menyewakan kostum terbaiknya
di seluruh Indonesia.
        </p>
    </div>
</div>
```

```
<div class="container">
<div class="container">
  <div class="desc">
    <h1>Pembelian Kostum</h1>
    <p>95% koleksi Gudang Kostum adalah berasal dari Import dan berkualitas, sedangkan
sisanya dibuat dengan desain sendiri dengan penjahit yang ahli di bidang kostum.
      Kostum dapat dibeli apabila Ready Stok di Store ataupun melalui pemesanan PRE ORDER
dengan lama pengerjaan sekitar 30 hari. Disarankan dapat memesan kostum lebih dari 30 hari.
    </p>
    <br>
    <p>Koleksi Lengkap ada di INSTAGRAM <a
href="https://www.instagram.com/gudangkostum_surabaya/">@gudangkostum_surabaya</a></
p>
  </div>
</div>
```

4.12 Halaman Ketentuan Sewa

Halaman ketentuan sewa adalah halaman yang menjelaskan peraturan dan syarat untuk menyewa di Toko Rental XYZ. Pada halaman ini pengunjung dapat menemui seputar informasi mengenai ketentuan-ketentuan yang ada dan berlaku untuk menyewa. Berikut ini adalah Segmen Program 3.42 untuk menjelaskan tentang tampilan halaman ketentuan sewa.

Tabel 4.20 Tabel Segmen Program dan *Flowchart* Halaman Ketentuan Sewa

Segmen Program	*Flowchart*
Segmen Program 4.32	Gambar 3.14

Segmen Program 4.34 Halaman Ketentuan Sewa

```
<div class="container">
  <div class="kostum-detail">
    <div class="desc">
      <h2>1. Ketentuan Sewa</h2>
     <ul>
      <li>Harga yg tertera untuk sewa kostum selama 1-3 hari (diambil H-1 acara dan
dikembalikan H+1 acara).</li>
      <li>Lebih dari 3 hari sewa akan dikenakan biaya tambah hari sebesar
25.000/kostum/hari.</li>
      <li>Jika ada pembatalan/pergantian kostum maka DP(uang muka) hangus atau sesuai
```

```
kebijakan manajemen.</li>
        <li>Reschedule diperbolehkan 1x dengan menginfokan admin terlebih dahulu.</li>
        <li>Pelunasan sewa dan deposit(jaminan) sebelum kostum diambil.</li>
        <li>Deposit akan dikembalikan langsung tunai atau ditransfer maks H+2 (hari kerja) dari tgl
pengembalian.</li>
        <li>Saat pengambilan kostum, mohon minta nota ke admin  (nota harus disimpan dengan
baik).</li>
       </ul>
       <br>
       <h2>2. Ketentuan Pengiriman Luar Kota</h2>
       <ul>
        <li>Kostum akan di kirim H-3acara, dan penyewa kirimkan kembali ke Surabaya H+2
acara(terlampir resi).</li>
        <li>Untuk pengiriman luar kota ada tambahan biaya (menyesuaikan kostum, tempat dan
pengiriman). kesepakatan dengan admin.
          Karna Jika di dalam surabaya sewa 3hari, lebih dari 3hari ada biaya tambahan 25.000/hari.
Untuk keluar kota butuh 7-10hari (termasuk hari pengiriman dan sewa).
        </li>
        <li>Ongkir di tanggung oleh penyewa.</li>
       </ul>
      </div>
   </div>
</div>
```

4.13 Halaman Testimoni

Halaman testimoni adalah halaman yang menampilkan bukti penyewaan kostum pada Toko Rental XYZ. Pada halaman ini pengunjung dapat melihat berbagai foto dari penyewa sebelumnya. Jika pengunjung menekan salah satu gambar, maka pengunjung akan dapat melihat deskripsi dalam bentuk modal. Modal yang dibuka tentunya akan mengenai kostum yang telah disewa. Tentunya pengunjung juga bisa melakukan penyaringan berdasarkan kategori dan tema testimoni. Untuk lebih jelasnya dapat dilihat pada Segmen Program 4.33 dan Segmen Program 4.34 untuk penampilan testimoni dan pencarian testimoni, dan 4.35 pengambilan dan pencarian data testimoni.

Tabel 4.21 Tabel Segmen Program dan *Flowchart* Halaman Testimoni

Segmen Program	*Flowchart*
Segmen Program 4.33	Gambar 3.15
Segmen Program 4.34	
Segmen Program 4.35	

Segmen Program 4.35 Tampilan Halaman Testimoni

```
        <h1>Ulasan</h1>
      </div>
      <div class="row search-ctr">
       <div class="col dropdown">
        <button class="btn dropdown-toggle" type="button" id="dropdownMenu2" data-bs-
toggle="dropdown" aria-expanded="false" style="border: 1px solid black;">
         Filter
        </button>
        <form class="dropdown-menu p-4" style="width:30%;" id="filter-form">
         <div class="category">
          <label class="select-cat">Kategori :</label>
          <select name="categories" id="categories" class="form-select">
           <option value="">Semua Kategori</option>
           @if(isset($categories) && count($categories) > 0)
             @foreach ($categories as $index => $category)
               <option value="{{ $category->id_category }}">{{ $category->category }}</option>
             @endforeach
           @else
             <option value="">Kategori tidak tersedia</option>
           @endif
          </select>
         </div>
         <br>
         <div class="theme">
          <label class="select-theme">Tema :</label>
          <select name="themes" id="themes" class="form-select">
           <option value="">Semua Tema</option>
           @if(isset($themes) && count($themes) > 0)
             @foreach ($themes as $index => $theme)
               <option value="{{ $theme->id_theme }}">{{ $theme->theme }}</option>
             @endforeach
           @else
             <option value="">Tema tidak tersedia</option>
           @endif
          </select>
         </div>
         <button type="submit" class="btn btn-primary">
          <i class="fas fa-search" style="margin-right: 8px"></i>
          Search</button>
        </form>
       </div>
       <form action="GET" class="col-md-2" id="search-form">
        <div class="search-bar">
         <div class="input-group mb-3">
          <input type="text" name="search-testimoni" class="form-control" id="search-input"
placeholder="Search" aria-label="Search" aria-describedby="basic-addon1" style="border: 1px
solid black;">
          <span class="input-group-text" id="basic-addon1" style="background-color: white;
border: 1px solid black;"><i class="fas fa-search"></i></span>
```

```
          </div>
        </div>
      </form>
    </div>

    <div class="row">
      <div class="col preview-container" id="testimonies-list">
        @include('partials.testimoni_list', ['testimonies' => $testimonies, 'noResults' => $noResults
?? false])
      </div>
    </div>
  </div>
```

Segmen Program 4.36 Tampilan tiap testimoni

```
@if($noResults)
  <p>No testimonies found.</p>
@else
  @foreach ($testimonies as $index => $testimoni)
    <div class="preview" data-bs-toggle="modal" data-bs-target="#exampleModal{{ $index }}">
      <div class="ulasan-pic">
        <div class="row image_testimoni">
          <img src="{{ $testimoni->imageUrl }}" alt="{{ $testimoni->name }}">
        </div>
        <div class="row name_testimoni">
          <p style="font-weight: bold;">{{ $testimoni->name }}</p>
        </div>
        <div class="row deskripsi" style="padding: 8px 8px 0 8px;">
          <p>{{ $testimoni->description }}</p>
        </div>
      </div>
    </div>

    {{-- Modal --}}
    <div class="modal fade" id="exampleModal{{ $index }}" tabindex="-1" aria-
labelledby="exampleModalLabel{{ $index }}" aria-hidden="true">
      <div class="modal-dialog modal-dialog-centered">
        <div class="modal-content">
          <div class="modal-header">
            <h5 class="modal-title" id="exampleModalLabel{{ $index }}">{{ $testimoni->name
}}</h5>
            <button type="button" class="btn-close" data-bs-dismiss="modal" aria-
label="Close"></button>
          </div>
          <div class="modal-body">
            <div class="row">
              <div class="col">
                <img src="{{ $testimoni->imageUrl }}" class="ulasan-pic">
              </div>
              <div class="col" style="margin-top: 16px;">
```

```
                    <p>{{ $testimoni->description }}</p>
                  </div>
                </div>
              </div>
              <div class="modal-footer">
                <button type="button" class="btn btn-secondary" data-bs-
dismiss="modal">Close</button>
              </div>
            </div>
          </div>
        </div>
    @endforeach
@endif
```

Segmen Program 4.37 Pengambilan dan Pencarian data Testimoni

```
<?php

namespace App\Http\Controllers;

use App\Models\category;
use App\Models\color;
use App\Models\testimoni;
use App\Models\theme;
use Illuminate\Support\Facades\Log;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

class client_testimoni extends Controller
{
    public function getClientTestimoni()
    {
        try {
            // Retrieve testimonies data
            $testimonies = testimoni::select('testimoni.*')->get();

            // Retrieve categories data
            $categories = category::select('category.*')->get();

            $themes = theme::select('theme.*')->get();

            $colors = color::select('color.*')->get();

            return view('client_util.testimoni_client', [
                'testimonies' => $testimonies,
                'categories' => $categories,
                'themes' => $themes,
                'colors' => $colors,
            ]);

        } catch (\Exception $e) {
```

```
        // Log the exception message
        Log::error('Error retrieving data: ' . $e->getMessage());

        return view('client_util.testimoni_client', [
          'testimonies' => [],
          'categories' => [],
          'themes' => [],
          'colors' => [],
        ]);
      }
    }

    public function filterTestimony(Request $request)
    {
      Log::info('Received filter');

      $theme = $request->input('theme');
      $category = $request->input('category');
      $search = $request->input('search');

      Log::info('Received search query:', ['search' => $search, 'category' => $category, 'theme' =>
$theme]);

      $query = Testimoni::select('testimoni.*',
        DB::raw('GROUP_CONCAT(DISTINCT category.category SEPARATOR ", ") AS categories'),
        DB::raw('GROUP_CONCAT(DISTINCT theme.theme SEPARATOR ", ") AS themes'))
        ->leftJoin('testimoni_kategori', 'testimoni.id_testimoni', '=',
'testimoni_kategori.id_testimoni')
        ->leftJoin('category', 'testimoni_kategori.id_kategori', '=', 'category.id_category')
        ->leftJoin('testimoni_tema', 'testimoni.id_testimoni', '=', 'testimoni_tema.id_testimoni')
        ->leftJoin('theme', 'testimoni_tema.id_theme', '=', 'theme.id_theme');

      if ($category) {
        $query->where('category.id_category', '=', $category);
      }

      if ($theme) {
        $query->where('theme.id_theme', '=', $theme);
      }

      if ($search) {
        $query->where(function($query) use ($search) {
          $query->where('testimoni.name', 'like', '%' . $search . '%')
            ->orWhere('testimoni.description', 'like', '%' . $search . '%');
        });
      }

      $data = $query->groupBy('testimoni.id_testimoni', 'testimoni.name', 'testimoni.description',
'testimoni.imageUrl')
        ->orderBy('testimoni.id_testimoni', 'asc')
        ->get(); // Execute the query and get the results
```

```
        $noResults = $data->isEmpty(); // Check if the data is empty

        if ($request->ajax()) {
            return response()->json([
                'testimonies' => view('partials.testimoni_list', ['testimonies' => $data, 'noResults' =>
$noResults])->render(),
            ]);
        }

        return view('client_util.testimoni_client', ['testimonies' => $data, 'noResults' => $noResults]);
    }

}
```

4.14 Halaman *Chat* Pengunjung

Halaman *chat* pengunjung adalah sebuah halaman untuk pengunjung dapat melakukan percakapan secara langsung dengan admin. Halaman ini dapat diakses langsung diakses melalui *navigation bar* atau saat pengunjung menekan tombol *chat* yang tersedia di halaman detail kostum. Untuk melakukan *chat* pelanggan akan memiliki sistem yang sama dengan *chat* administrator, sehingga untuk pengaturan *socket.io* juga dapat dilihat kembali pada Segmen Program 4.21. Berikut ini adalah Segmen Program 4.35 untuk penjelasan pengiriman pesan dan menerima pesan, serta Segmen Program 4.36 untuk menjelaskan tampilan *chat*.

Tabel 4.22 Tabel Segmen Program dan *Flowchart* Halaman *Chat* Pengunjung

Segmen Program	*Flowchart*
Segmen Program 4.36	Gambar 3.15
Segmen Program 4.37	

Segmen Program 4.38 Pengiriman Pesan oleh Pengunjung

```
<script>
    $(function () {
        let $chatHeader = $("#chatHeader");
        let $chatName = $chatHeader.find(".chat-name");
        let $chatInput = $(".chat-input");
        let $chatBody = $(".chat-body");
        let $messageWrapper = $("#messageWrapper");
        $chatInput.on('focus', function () {
            console.log('Chat input focused');
        });
        let user_id = "{{ auth()->user()->id }}"
        let ip_address = '127.0.0.1';
        let socket_port = '8005';
        let socket = io(ip_address + ':' + socket_port);

        let friendId = 4; // Set friendId directly to 4

        socket.on('connect', function() {
            socket.emit('user_connected', user_id);
        });
        fetchConversation(friendId); // Fetch conversation on load
        function fetchConversation(friendId) {
            console.log("Fetching conversation for user ID:", friendId);
            $.ajax({
                url: "{{ route('message.conversation', '') }}/" + friendId,
                type: 'GET',
                headers: {
                    'X-CSRF-TOKEN': $('meta[name="csrf-token"]').attr('content')
                },
                success: function(response) {
                    if (response.success) {
                        $chatName.text(response.friendInfo.name);
                        $messageWrapper.html(''); // Clear previous messages
                        response.messages.forEach(message => {
                            appendMessageToWrapper(message);
                        });
                    }
                },
                error: function(xhr, status, error) {
                    console.error('Error fetching conversation:', error);
                }
            });
        }

        console.log("User ID:", user_id);
```

```
$chatInput.keypress(function (e) {
    let message = $(this).html();
    if (e.which === 13 && !e.shiftKey) {
        e.preventDefault();
        $chatInput.html("");
        console.log('friendinfo:', friendId)
        sendMessage(message);
        return false;
    }
});

function sendMessage(message) {
    let url = "{{ route('message.send-message') }}";
    let token = "{{ csrf_token() }}";
    console.log(friendId)

    // Check if a friend is selected
    if (!friendId) {
        console.error('No friend selected for sending message');
        return;
    }

    console.log("Sending message to friend ID:", friendId);

    let formData = new FormData();
    formData.append('message', message);
    formData.append('_token', token);
    formData.append('reciever_id', friendId);

    appendMessageToSender(message);

    $.ajax({
        url: url,
        type: 'POST',
        data: formData,
        processData: false,
        contentType: false,
        dataType: 'JSON',
        success: function(response) {
            if(response.success){
                console.log(response.data);
            } else {
                console.error('Error:', response.message);
            }
```

```
            },
            error: function(xhr, status, error) {
              console.error('Submission error:', error);
            }
          });
        }

        function appendMessageToWrapper(message) {
          let messageElement = '<div class="row message align-items-center mb-2">' +
            '<div class="col-md-12 user-info" style="display: flex; align-items: center; margin-top:
16px; margin-left:16px;">\n' +
              '<div class="chat-image" style="margin-right: 10px">\n' +
                '<i class="fas fa-user"></i>\n' +
              '</div>\n' +
              '<div class="chat-name" style="font-weight: 600; display:flex; flex-direction:
row;">\n' +
                message.sender_name +
                '<span class="small time" title="'+moment(message.created_at).format('YYYY-
MM-DD HH:mm:ss')+'" style="color: grey; margin-left: 5px; padding-top: 3px;">\n' +
                moment(message.created_at).format('h:mm A') + '</span>\n' +
              '</div>\n' +
            '</div>\n' +
            '<div class="col-md-12 message-content">\n' +
              '<div class="message-text">\n' + message.content +
              '</div>\n' +
            '</div>\n' +
          '</div>';

          $messageWrapper.append(messageElement);
        }


        function appendMessageToSender(message) {
          let name = '{{ $myInfo->name }}';
          let messageElement = '<div class="row message align-items-center mb-2">' +
            '<div class="col-md-12 user-info" style="display: flex; align-items: center; margin-top:
16px; margin-left:16px;">\n' +
              '<div class="chat-image" style="margin-right: 10px">\n' +
                '<i class="fas fa-user"></i>\n' +
              '</div>\n' +
              '<div class="chat-name" style="font-weight: 600; display:flex; flex-direction:
row;">\n' +
                name +
                '<span class="small time" title="'+getCurrentDateTime()+'" style="color: grey;
margin-left: 5px; padding-top: 3px;">\n' +
```

```
                    getCurrentTime() + '</span>\n' +
                  '</div>\n' +
                '</div>\n' +
                '<div class="col-md-12 message-content">\n' +
                  '<div class="message-text">\n' + message +
                  '</div>\n' +
                '</div>\n' +
              '</div>';

              $messageWrapper.append(messageElement);
            }

            function getCurrentDateTime() {
              return moment().format('YYYY-MM-DD HH:mm:ss');
            }

            function getCurrentTime() {
              return moment().format('h:mm A');
            }

            socket.on("private-channel:PrivateMessageEvent", function (message) {
              appendMessageToWrapper(message);
            });
          });
        </script>
```

Segmen Program 4.38 Tampilan Halaman *Chat* Pengunjung

```
<div class="row chat-row">
        <div class="col-md-3">
          <div class="users">
            <h5 style="font-weight: 600;">Users</h5>
            <ul class="list-group list-chat-item">
              @if($users->count())
                @foreach ($users as $user)
                  @if($user->id == 4) <!-- Only show user with id=4 -->
                  <li class="chat-user-list @if($user->id == $friendInfo->id) active @endif">
                    <a href="javascript:void(0)" data-id="{{ $user->id }}" class="user-link"
style="font-weight: 600;">
                      {{ $user->name }}
                    </a>
                  </li>
                  @endif
                @endforeach
              @endif
            </ul>
          </div>
```

```
    </div>
    <div class="col-md-9 chat-section">
      <div class="chat-header" id="chatHeader">
        <div class="chat-name" style="font-weight: 600; margin-left:8px;">
          {{ $friendInfo->name }}
        </div>
      </div>
      <div class="chat-body">
        <div class="message-listing" id="messageWrapper">
          @foreach ($messages as $message)
            <div class="row message align-items-center mb-2">
              <div class="col-md-12 user-info">
                <div class="chat-image">
                  <i class="fas fa-user"></i>
                </div>
                <div class="chat-name">
                  {{ $message['sender_name'] }}
                  <span class="small time" title="{{ $message['created_at'] }}">
                    {{ \Carbon\Carbon::parse($message['created_at'])->format('h:i A') }}
                  </span>
                </div>
              </div>
              <div class="col-md-12 message-content">
                <div class="message-text">
                  {{ $message['content'] }}
                </div>
              </div>
            </div>
          @endforeach
        </div>
      </div>
      <div class="chat-box">
        <div class="chat-input" id="chatInput" contenteditable=""></div>
      </div>
    </div>
  </div>
```

4.15 Web Scrap Instagram

Untuk pengambilan data *scrapping* dari *Instagram* akan dibantu dengan *Apify* untuk memudahkan pengambilan data sekaligus untuk mengambil dtaa sesuai dengan peraturan dari *Instagram* itu sendiri. Untuk proses selanjutnya setelah melakukan *scrapping* dengan *Apify* adalah membersihkan data yang sudah didapatkan. Hal ini dikarenakan *Apify* tidak menyediakan untuk penyaringan data. Berikut pada Segmen Program 4.36 adalah proses *data cleaning* untuk data dari *Apify*.

Segmen Program 4.39 *Data cleaning* data *Instagram*

```
def extract_data_from_page(soup):
    name_h4 = soup.find_all('h4', class_='post_title entry-title')

    excluded_names = [
        "SEWA", "logo", "VOB", "CELEBRATE", "Gudang Kostum Steril", "UV",
        "Vitamin", "Topi", "Boot", "Visor", "Sarung", "masker", "KN95",
        "kacamata", "hazmat", "Face", "Desinfektan", "Photostudio", "photo studio", "TUMB",
        "Peraturan", "Slide", "home", "video", "koleksi", "shower", "minuman", "cover"
    ]

    excluded_urls = [
        "https://gudangkostum.com/home-1/attachment/tumb-prince/",
"https://gudangkostum.com/home-1/attachment/tumb-animal/",
        "https://gudangkostum.com/home-1/attachment/tumb-inter/",
"https://gudangkostum.com/home-1/attachment/tumb-princess/",
        "https://gudangkostum.com/home-1/attachment/tumb-hero/",
"https://gudangkostum.com/bridal-station/" # Add your excluded URLs here
    ]

    costume_data = []

    for h4_tag in name_h4:
        h4_tag_a = h4_tag.find('a')

        if h4_tag_a:
            costume_name = h4_tag_a.get_text(strip=True)
            costume_url = h4_tag_a.get('href')

            # Apply regex to clean the name
            cleaned_name = re.sub(r'\s{2,}', ' ', costume_name.strip())

            # Check if the name contains only one word
            if cleaned_name == 1:
                continue

            if re.search(r'\bbridal\.?station\b', cleaned_name, flags=re.IGNORECASE):
                category = "#gk_bridalstation"
```

```
        # Check if the name contains any excluded words (case-insensitive)
        if not any(exclude_word.lower() in cleaned_name.lower() for exclude_word in
excluded_names) and \
           not any(excluded_url in costume_url for excluded_url in excluded_urls):
            data = {
               "name": costume_name,
               "link" : costume_url,
               "size": [],
               "desc": ""
            }
            costume_data.append(data)

   return costume_data

costume_json_file = "costume_data.json"
costume_data_all = []
```

4.16 Penyaringan Data Kostum *Instagram*

Setelah pembersiahn data, maka proses selanjutnya juga masih mencangkup pembersihan, namun proses ini lebih digunakan untuk mengambil daat yang diinginkan seperti ukuran dan kategori dari kostum yang sudah disaring sebelumnya. Hal ini dapat dilihat pada Ssegmen Program 4.37.

Segmen Program 4.40 Pengambilan Data Kostum

```
costume_json_file = "costume_data2.json"
costume_data_all = []

for page_num in range(start_page, end_page + 1):
   current_url = start_url.format(page_num)
   print("Scraping:", current_url)

   # Create a Request object with headers
   request = Request(current_url, headers={"user-agent": useragent})

   # Open the URL with urlopen
   response = urlopen(request)

   # Read the HTML content directly from the response
   html_content = response.read()
```

```
    soup = bs4.BeautifulSoup(html_content, 'html.parser')

    # Extract data from the current page
    costume_data = extract_data_from_page(soup)

    # Append the extracted data to the main list
    costume_data_all.extend(costume_data)

for costume_data in costume_data_all:
    request = Request(costume_data["link"], headers={"user-agent": useragent})
    response = urlopen(request)
    html_content = response.read()
    inner_soup = bs4.BeautifulSoup(html_content, 'html.parser')

    description = inner_soup.find('div', class_='post_content entry-content')
    a_tags = inner_soup.find('a', class_='trx_addons_icon-eye')

    if a_tags:
        span_tag = a_tags.find('span', class_='post_counters_number')
        if span_tag:
            views = span_tag.text.strip()
            print("views:", views)
            costume_data["views"] = views  # Add views count to the data dictionary
        else:
            print("views: empty")
            costume_data["views"] = ""  # If views count is not found, set it to an
empty string

    if description:
        p = description.find('p')
        if p:
            costume_desc = p.get_text(strip=True)
            print("desc:", costume_desc)
            costume_data["desc"] = costume_desc
            size = extract_size_from_name_and_description(costume_data["name"],
costume_desc)  # Pass costume_name
            costume_data["size"].append(size)
            if size:
                costume_data["size"] = size
            else:
                costume_data["size"] = size
        else:
```

```
        else:
            size = extract_size_from_name_and_description(costume_data["name"],
"")
            costume_data["size"] = size

        print("size:", costume_data["size"])
        print("Finished scraping content inside the URL.")
        print("-" * 70)

with open(costume_json_file, mode='w', encoding='utf-8') as costume_file:
    json.dump(costume_data_all, costume_file, ensure_ascii=False, indent=4)
```