**PROPOSAL PEMBUATAN**

**WEBSITE GLASGOW**

****

Disusun Oleh :

Dian Priyo Pamungkas (50421370)

Muhammad Saidar Bahrin (50421880)

Muhammad Arkan Araafkha (50421921)

Puput Debbyla Walzahra (51421182)

**KATA PENGANTAR**

Puji dan syukur penyusun panjatkan kehadirat kepada Tuhan Yang Maha Esa, karena atas rahmat dan karunia-Nya sehingga tim kami dapat menyelesaikan proposal Pembuatan [ ] ini dapat diselesaikan dengan baik.

Proposal ini disusun untuk memenuhi kebutuhan akan pengelolaan layanan laundry yang lebih efektif dan efisien melalui dukungan teknologi informasi. Website [ ] dirancang untuk memberikan kemudahan kepada pelanggan dalam melakukan pemesanan, mendapatkan informasi layanan, serta mengakses berbagai fitur yang mendukung peningkatan kualitas layanan.

Kami menyadari bahwa proposal ini masih jauh dari sempurna. Oleh karena itu, kami sangat terbuka terhadap kritik dan saran yang membangun agar rencana pembuatan website ini dapat disempurnakan dan memberikan manfaat yang optimal bagi semua pihak.

Semoga proposal ini dapat memberikan gambaran yang jelas mengenai langkah-langkah yang akan diambil dalam pengembangan website [ ] dan dapat menjadi acuan dalam mewujudkan pelayanan yang lebih baik di masa depan.

Akhir kata, kami mengucapkan terima kasih kepada semua pihak yang telah memberikan dukungan dan kontribusi selama proses penyusunan proposal ini.

Depok, 13 Januari 2025

Tim Pengembang

**DAFTAR ISI**

# **BAB I**

**LATAR BELAKANG**

**1.1 Latar Belakang**

Glasgow merupakan sebuah usaha yang bergerak dalam bisnis penjualan kacamata untuk gaya atau kacamata yang sedang banyak digunakan oleh para artis dengan harga yang terjangkau. Selama ini, keterbatasan pemasaran barang kacamata yang dijual menjadi kendala penjualan dalam meningkatkan penghasilan penjualan. Sistem penjualan yang masih kurang optimal, serta sulitnya menjangkau konsumen merupakan beberapa kendala yang terdapat di toko Glasgow. Pada saat ini sistem yang diterapkan pada toko Glasgow masih dilakukan secara manual, mulai dari proses pembelian oleh konsumen sampai penyimpanan data-data lain yang beruhubungan dengan proses penjualan barang. Selain itu, untuk mengetahui informasi terkait barang yang tersedia di toko, konsumen masih harus datang secara langsung ke toko Glasgow.

**1.2 Batasan Masalah**

Adapun batasan masalah yang dibahas dalam penulisan ilmiah ini adalah sebagai berikut:

1. Sistem penjualan yang dibangun pada toko Glasgow berbasis web.

2. Web yang dibangun menggunakan bahasa pemrograman PHP dengan *framework* Laravel 10 dan penyimpanan *database* MySQL.

3. Terdapat dua pengguna di dalam sistem, yaitu admin dan pembeli.

4. Pengguna sebagai admin dapat melakukan menambahkan barang baru, mengahpus barang, update barang, melihat laporan hasil penjualan toko dan barang yang sudah terjual pada toko web Glasgow, dan admin dapat mengubah profil web.

5. Pengguna sebagai pembeli dapat mencari barang, memilih barang dan memasukkan barang yang dipilih ke dalam keranjang belanja. Pengguna dapat melakukan pemesanan secara *online* dan kemudian melakukan pembayaran barang yang sudah masuk ke dalam keranjang.

**1.3 Tujuan**

Tujuan dari penulisan ilmiah ini adalah melakukan pembuatan sistem penjualan kacamata pada toko Glasgow berbasis web. Sistem tersebut diharapkan dapat membantu mempromosikan dan mempermudah proses penjualan barang secara *online* pada toko Glasgow sehingga dapat meningkatkan hasil penjualannya. Sistem yang dibangun juga diharapkan dapat mempermudah pencatatan barang dan mempermudah para pembeli agar tidak perlu datang langsung ke toko Glasgow untuk melihat dan melakukan pembelian barang.

**1.4 Metode Penelitian**

Metode penelitian yang digunakan untuk pembangunan sistem penjualan kacamata pada toko Glasgow berbasis web ini adalah metode SDLC (*System Development Life Cycle*). Tahapan dari metode SDLC ini meliputi:

1. Perencanaan

Tahapan awal yang dilakukan adalah analisis terhadap tekonologi komunikasi dan informasi yaitu di dalam bidang *e-commerce*, agar dapat mengetahui manfaat jika menggunakan *e-commerce* di dalam toko Glasgow.

2. Analisis

Pada tahapan ini dilakukan analisis kebutuhan fungsional dan non-fungsional agar sistem yang dibangun sesuai dengan kebutuhan pada toko Glasgow. Penulis melakukan identifikasi dengan cara berdiskusi kepada pemilik dan pegawai yang bekerja di toko Glasgow.Perancangan

Pada fase ini, akan dilakukan proses perancangan untuk membangun sistem informasi penjualan pada toko Glasgow, seperti perancangan UML (*Unifed Modeling Language*), perancangan struktur navigasi, serta dilakukan perancangan *database*.

3. Implementasi

Pada tahapan ini, akan dilakukan proses pembuatan *website* bedasarkan desain tampilan yang telah dirancang sebelumnya kedalam bahasa pemograman yang menjadi dasar dari website menggunakan visual studio code serta membuat struktur navigasi sesuai dengan alur yang telah dijelaskan dalam UML.

4. Uji coba

Pada tahapan terakhir ini, akan dilakukan suatu pengujian terhadap *website* yang telah berhasil dibuat menggunakan metode *Blackbox* dengan menguji web melalui jaringan *browser* seperti Google Chrome, Miscrosoft Edge agar dapat mengetahui apakah *website* yang telah dibuat sudah lengkap fungsionalnya atau masih ada beberapa bagian yang harus diperbaiki lagi. Pengguna juga akan terlibat dalam tahap uji coba untuk mengidentifikasikan masalah dan memberikan masukan terhadap website.

# **BAB II**

**PROPOSAL PENAWARAN**

**2.1 Pendahuluan**

Seiring dengan berkembangnya teknologi, layanan berbasis digital telah menjadi kebutuhan utama bagi berbagai sektor usaha, termasuk layanan Belanja. Melalui website, usaha seperti Glasgow dapat menjangkau lebih banyak pelanggan, meningkatkan efisiensi operasional, dan memberikan pengalaman layanan yang lebih praktis dan modern.

Kami dari [Nama Perusahaan Anda/Tim Pengembang] mengajukan penawaran untuk membantu Omah Laundry dalam mengembangkan website profesional dengan fitur-fitur yang sesuai kebutuhan bisnis Anda.

## 

| **Layanan** | **Harga** |
| --- | --- |
| Analisis Kebutuhan | Rp 1.000.000 |
| Perancangan UI/UX | Rp 1.000.000 |
| Pengembangan Website | Rp 5.000.000 |
| Domain dan Hosting | Rp 1.000.000 |
| Pelatihan dan Dukungan Teknis | Rp 3.000.000 |
| Total Biaya | |
| Rp 11..000.000 | |

#### **2.2. Deskripsi Layanan yang Ditawarkan**

Fitur-fitur dari website "Glasgow Shopping" adalah sebagai berikut:

1. **Pencarian Produk**
   * Memungkinkan pengguna mencari produk berdasarkan kategori, nama toko, atau nama produk.
2. **Integrasi Toko Lokal**
   * Platform bagi toko-toko lokal untuk memasarkan produk mereka secara online.
3. **Keranjang Belanja**
   * Fasilitas untuk menambahkan produk ke keranjang belanja sebelum melakukan pembelian.
4. **Metode Pembayaran yang Aman**
   * Mendukung berbagai metode pembayaran online yang aman dan terpercaya.
5. **Ulasan dan Peringkat**
   * Memberikan fitur ulasan produk agar konsumen dapat membuat keputusan belanja yang lebih baik.
6. **Pengiriman dan Pelacakan**
   * Informasi pengiriman yang transparan dan dilengkapi dengan sistem pelacakan.

**2.3. Estimasi Biaya**

| **Layanan** | **Harga** |
| --- | --- |
| Analisis Kebutuhan | Rp 1.000.000 |
| Perancangan UI/UX | Rp 1.000.000 |
| Pengembangan Website | Rp 5.000.000 |
| Domain dan Hosting | Rp 1.000.000 |
| Pelatihan dan Dukungan Teknis | Rp 3.000.000 |
| Total Biaya | |
| Rp 11..000.000 | |

**2.4. Estimasi Waktu Pelaksanaan** Pengerjaan website diperkirakan selesai dalam waktu 6-8 minggu terhitung sejak kontrak disepakati.

#### **2.5. Keunggulan Kami**

1. **Fokus Lokal:** Mengutamakan toko-toko lokal di Glasgow, sehingga mendukung bisnis kecil dan menengah.
2. **Antarmuka Ramah Pengguna:** Desain yang intuitif dan mudah digunakan oleh semua kalangan.
3. **Integrasi Lengkap:** Semua kebutuhan belanja mulai dari pencarian, pembelian, hingga pelacakan pengiriman dalam satu platform.
4. **Keamanan Data:** Sistem pembayaran yang aman dan perlindungan data pribadi pengguna.
5. **Fitur Kustomisasi:** Pengguna dapat membuat wishlist, menerima rekomendasi berdasarkan preferensi, dan mendapatkan penawaran yang dipersonalisasi.
6. **Dukungan Multibahasa:** Mempermudah pengguna internasional untuk berbelanja di Glasgow

# **BAB III**

**ANALISIS DAN PEMBAHASAN**

**3.1 Metode Penelitian**

Pada penilitian ini, penulis menggunakan metode SDLC (System Development Life Cycle) untuk membangun sistem informasi penjualan kacamata pada toko “Glasgow”. Metode SDLC ini terbagi menjadi perencanaan, analisis, perancangan, implementasi, dan uji coba untuk memastikan pengembangan sistem berjalan dengan baik dan memenuhi kebutuhan pengguna.

**3.2 Gambaran Umum Website**

Website penjualan yang dibangun untuk toko Glasgow ini bertujuan untuk melakukan penjualan secara online. Selama ini toko Glasgow hanya melakukan penjualan secara langsung, sehingga dengan menggunakan website ini, pembeli dapat melihat dan memesan produk tanpa harus datang langsung ke toko Glasgow ini.

Pada sistem informasi penjualan berbasis web di toko Glasgow terdapat beberapa fitur pendukung untuk membantu kegiatan operasional pada toko Glasgow. Fitur-fitur yang tersedia seperti tampilan produk, keranjang belanja dan juga sistem pembayaran. Sistem informasi penjualan yang dibangun bertujuan untuk mempermudah proses pengarsipan penjualan, memasarkan dan menjual produk, pengelolaan data produk, penjelasan mengenai produk dan informasi mengenai ketersediaan produk.

## **3.3 Analisis Kebutuhan**

Pada tahap ini, dilakukan analisis mengenai kebutuhan sistem dalam pembuatan sistem informasi penjualan pada toko Glasgow untuk mengetahui apa saja yang dibutuhkan dalam pembuatan web.

**3.3.1 Analisis Fungsional**

Pada tahap analisis fungsional ini berisikan berisikan tentang kebutuhan dari sistem yang akan dirancang yaitu :

1. Homepage untuk menampilkan menu produk di toko Glasgow.

2. Calon pembeli dapat melakukan registrasi dan login.

3. Calon pembeli dapat memasukan produk ke dalam keranjang, dan memesan produk.

4. Terdapat informasi tentang toko Glasgow ini.

5. Pada halaman produk bagian admin, admin dapat menambahkan, mengedit, dan menghapus produk.

6. Pada halaman transaksi bagian admin, admin dapat melihat transaksi pembelian.

### **3.3.2** **Analisis Non Fungsional**

Pada tahap analisis non fungsional ini terdiri dari perangkat lunak dan perangkat keras yang dimaksud yaitu alat yang membantu penulis dalam proses pembuatan web penjualan di Glasgow.

**1. Alat Pendukung Perangkat Keras**

1. Windows 11 Home Pro Single Language (64-Bit).

2. Visual Studio Code

3. Google Chrome

4. XAMPP Control Panel v3.2.2

**2. Alat Pendukung Perangkat Keras**

1. Laptop Asus Vivobook

2. Processor AMD Ryzen 3

3. *Installed memory* (RAM) 8 GB

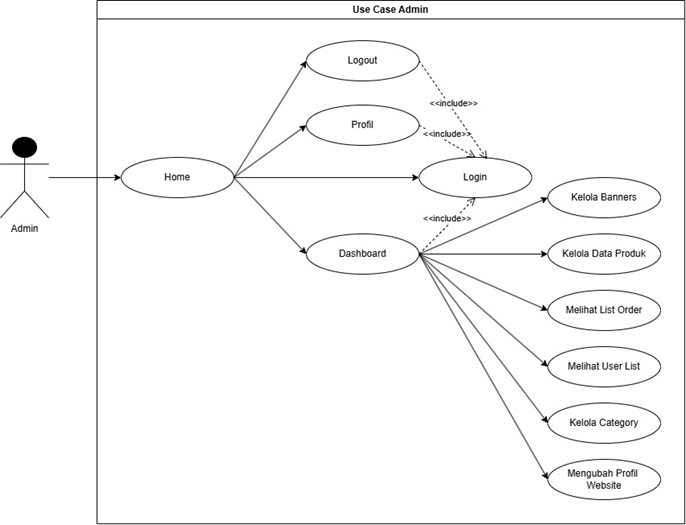
4. AMD Radeon

## **3.4 Perancangan UML**

Pada tahap ini, penulis merancang terlebih dahulu sistem dari website yang ingin dibangun. Salah satunya perancangan UML (*Unifed Modeling Language*). Dalam rancangan kali ini menggunakan *Use Case* Diagram*, Activity* Diagram, dan *Class* Diagram.

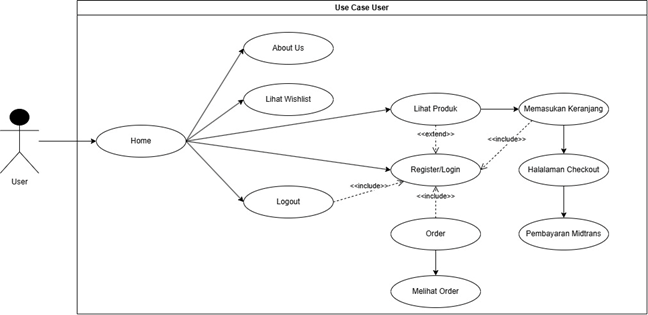
### **3.4.1** **Use Case Diagram**

Perancangan *use case* diagram merupakan model diagram yang menggambarkan hubungan antara aktor yang terlibat dalam sistem yang akan dibangun. Pengguna yang menggunakan situs web bertindak sebagai aktor. Fungsi *use case* hanya memberikan gambaran singkat tentang hubungan antara *use case*, aktor dan sistem dari *website* yang akan dibangun. Tampilan *use case* diagram admin ditunjukan pada Gambar 3.1 dan untuk tampilan *use case* user ditunjukan pada Gambar 3.2



Gambar 3. 1Use Case Diagram Admin

Pada Gambar 3.1 *Use case* diagram admin menggambarkan alur aktivitas admin dalam mengelola web penjualan pada toko Glasgow. Langkah pertama saat admin masuk ke halaman web, admin perlu melakukan login terlebih dahulu agar dapat ke halaman *dashboard*. Pada halaman *dashboard*, admin dapat mengelola data produk, melakukan perubahan pada *banners*, melihat *list order*, melihat *user list* yang sudah *register*, melihat *category*, dan dapat mengubah profil pada web.



Gambar 3. 2Use Case Diagram User

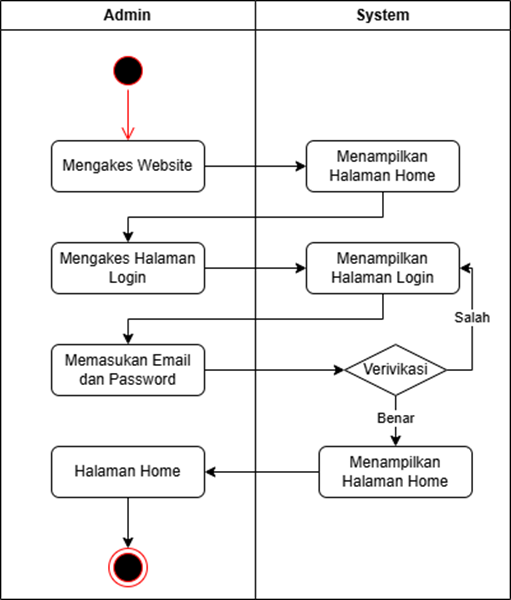
Pada Gambar 3.2 *User* yang berperan menjadi aktor, saat *user* mengakses halaman web, *user* sudah dapat melihat produk dan halaman *about us*. *User* perlu *login* atau *register* terlebih dahulu agar dapat melihat produk dan memasukannya ke dalam keranjang lalu melakukan *checkout*, dan dapat melihat order.

### **3.4.2.1** **Activity Diagram**

*Activity* diagram adalah gambaran umum dari operasi sistem, yang menggambarkan semua proses dari kemungkinan keputusan hingga akhir dari suatu operasi.

#### **3.4.2.1 Activity Diagram Admin Login**

Pada gambar 3.3 menggambarkan alur proses admin saat melakukan login pada *website* Glasgow. Saat pertama kali mengakses halaman web, akan ditampilkan halaman home *website,* setelah menampilkan halaman home pada *website* admin mengakses *login* dan akan di arahkan ke halaman *login*, saat dihalaman login admin mengisi email dan password. Jika admin gagal saat melakukan *login* akan kembali lagi pada halaman *login* dan jika admin berhasil melakukan *login* admin akan masuk dalam halaman home *website* dengan dashboard admin.



Gambar 3. 3Activity Diagram Admin Login

#### **3.4.2.2** **Activity Diagram Admin Logout**

Pada gambar 3.4 menggambarkan alur proses *logout* admin pada *webiste* Glasgow. Saat admin sudah berhasil melakuka *login* lalu masuk ke halaman *website* dan admin akan melakukan *logout* dengan mengakses *logout* atau menekan tombol *logout* pada *website* admin akan di arahkan kembali ke halaman home kembali dengan tidak tersangkut akun admin.

Gambar 3. 4Activity Diagram Admin Logout

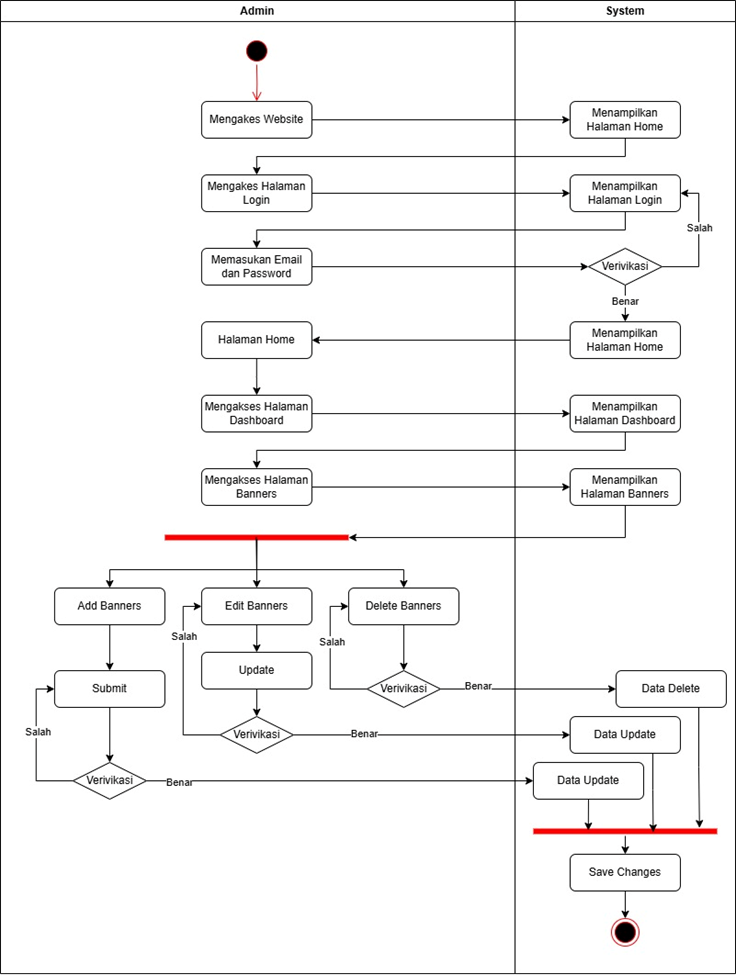
#### **3.4.2.3** **Activity Diagram Admin Dashboard Produk**

Gambar 3.5 menggambarkan alur proses admin saat mengakses halaman dashboard produk pada *website* Glasgow. Admin dapat mengakses halaman dasboard produk ketika admin sudah melakukan login terlebih dahulu. Selanjutnya admin dapat menekan tombol dashboard yang ada pada halaman home web, ketika sudah diarahkan ke halaman dashbord admin memilih produk lalu masuk ke halaman produk admin dapat add produk, update produk, delete. *System* akan menerima data masukan jika add produk, edit, dan juga delete produk berhasil makan *system* akan menyimpan data jika gagal edit dan juga delete akan di arahkan kembali ke halaman edit ataupun delete.

Gambar 3. 5 Activity Diagram Admin Dashboard Produk

#### **3.4.2.4** **Acitivity Diagram Dashboard Banners**

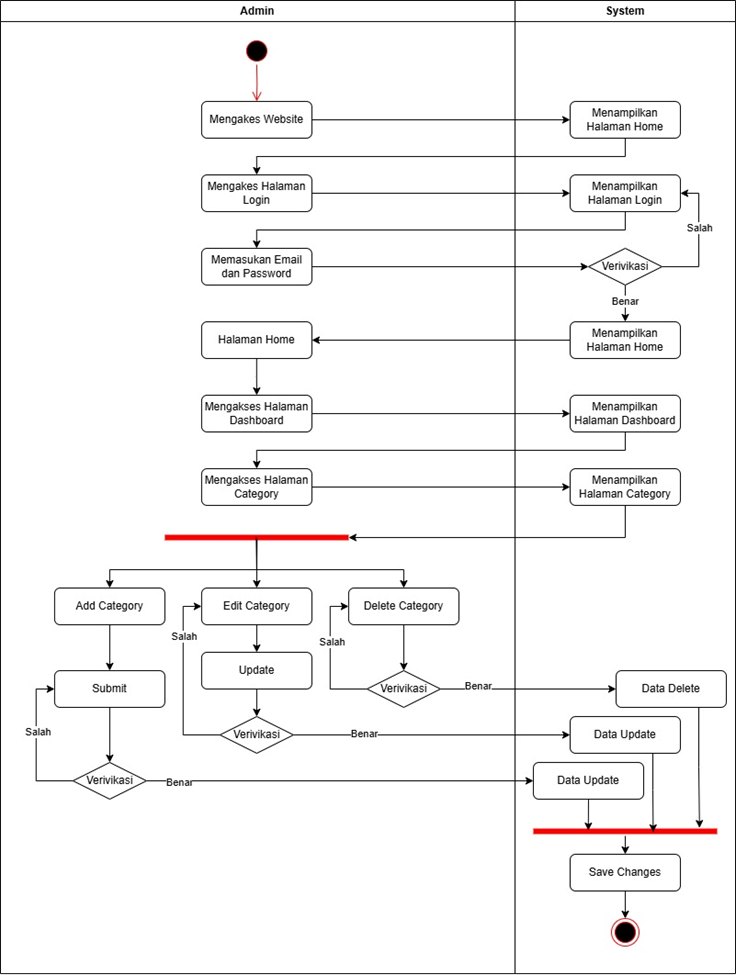
Pada gambar 3.6 ini menggambarkan bagaimana alur proses admin saat mengakses halaman dashboard banners pada *website*. Admin dapat mengakses halaman dasboard banners ketika admin sudah melakukan login terlebih terlebih dahulu. Selanjutnya admin dapat menekan tombol dashboard yang ada pada halaman home web, ketika sudah diarahkan ke halaman dashbord admin memilih banners lalu masuk ke halaman banners admin dapat add banners, update banners, delete banners. *System* akan menerima data masukan jika add banners, edit, dan juga delete banners berhasil makan *system* akan menyimpan data jika gagal edit dan juga delete akan di arahkan kembali ke halaman edit ataupun delete.



Gambar 3. 6Activity Diagram Dashboard Banners

#### **3.4.2.5** **Activity Diagram Dashboard Category**

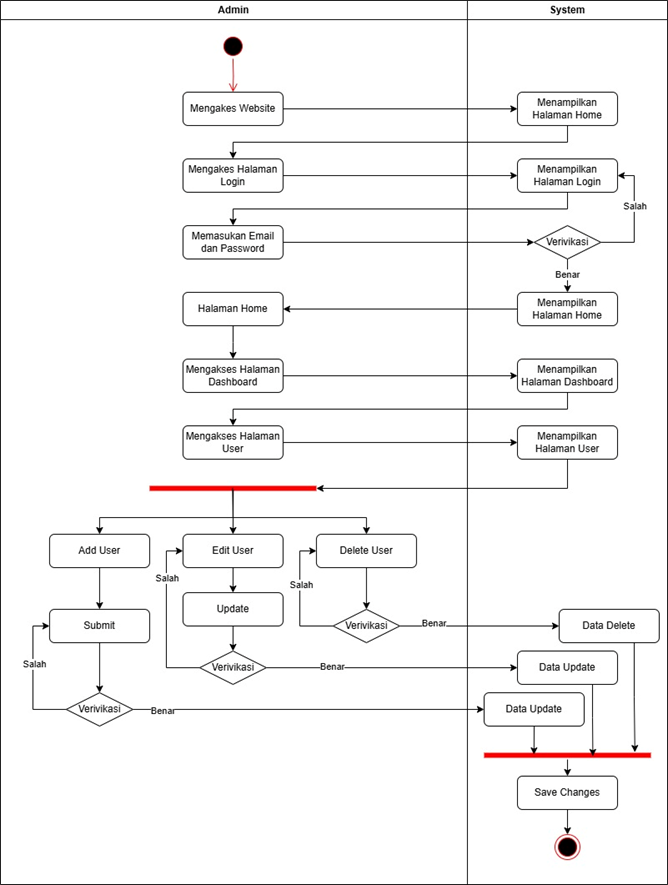
Pada gambar 3.7 ini menggambarkan bagaimana alur proses admin saat mengakses halaman *dashboard category* pada *website*. Admin dapat mengakses halaman *dasboard category* ketika admin sudah melakukan login terlebih terlebih dahulu. Selanjutnya admin dapat menekan tombol dashboard yang ada pada halaman home web, ketika sudah diarahkan ke halaman *dashbord,* admin memilih *category* lalu masuk ke halaman *category* admin dapat add *category*, update *category*, delete *category.* *System* akan menerima data masukan jika add *category*, edit, dan juga delete *category* berhasil makan *system* akan menyimpan data jika gagal edit dan juga delete akan di arahkan kembali ke halaman edit ataupun delete.



Gambar 3. 7Activity Diagram Dashboard Category

#### **3.4.3.6** **Activity Diagram User List**

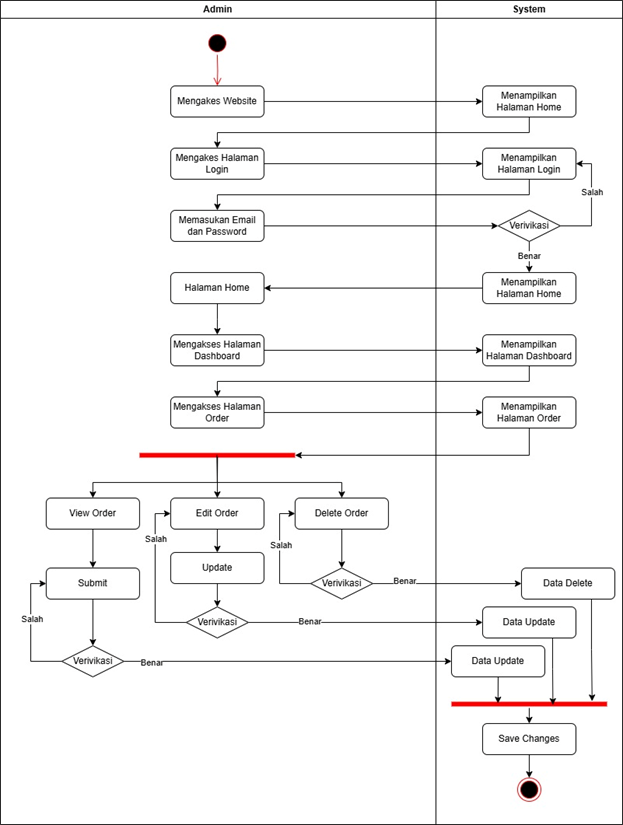
Gambar 3.8 ini menggambarkan bagaimana alur proses admin saat mengakses halaman *dashboard user list* pada *website*. Admin dapat mengakses halaman *dasboard user list* ketika admin sudah melakukan login terlebih terlebih dahulu. Selanjutnya admin dapat menekan tombol dashboard yang ada pada halaman home web, ketika sudah diarahkan ke halaman *dashbord,* admin memilih *user list* lalu masuk ke halaman *user list* admin dapat add *user*, update *user*, delete *user*. *System* akan menerima data masukan jika add *user*, edit, dan juga delete *user* berhasil makan *system* akan menyimpan data jika gagal edit dan juga delete akan di arahkan kembali ke halaman edit ataupun delete.



Gambar 3. 8 Activity Diagram Dashboard User List

#### **3.4.3.7** **Activity Diagram Order**

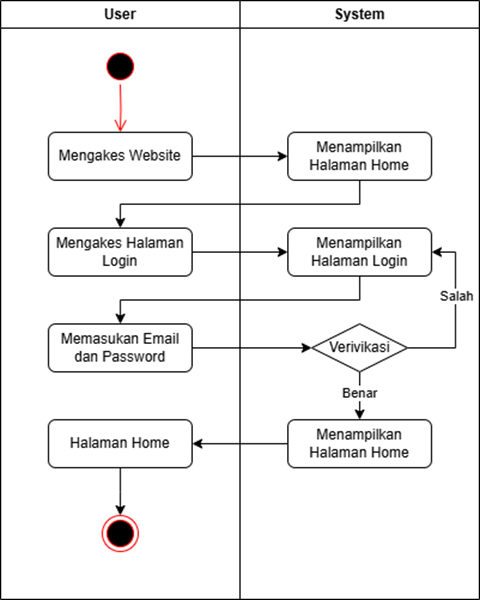
Pada Gambar 3.9 ini menggambarkan bagaimana alur proses admin saat mengakses halaman *dashboard order* pada *website*. Admin dapat mengakses halaman *dasboard order* ketika admin sudah melakukan login terlebih terlebih dahulu. Selanjutnya admin dapat menekan tombol *dashboard* yang ada pada halaman home web, ketika sudah diarahkan ke halaman *dashbord,* lalu memilih *order* lalu masuk ke halaman *order* admin dapat view *order*, update *order*, delete *order*. *System* akan menerima data masukan jika view *order*, edit, dan juga delete *order* berhasil makan *system* akan menyimpan data jika gagal edit dan juga delete akan di arahkan kembali ke halaman edit ataupun delete.



Gambar 3. 9Activity Diagram Dashboard Order

#### **3.4.3.8** **Activity Diagram User Login**

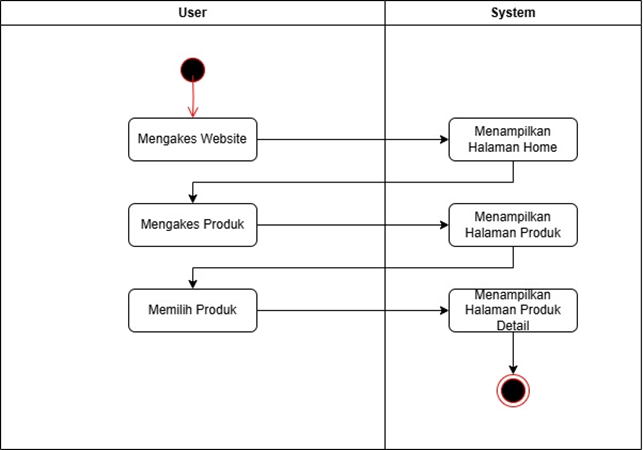
Pada gambar 3.10 ini menggambarkan bagimana *user* login ke dalam *website* Glasgow. Sama seperti dengan login admin, saat pertama kali mengakses halaman web, *user* akan ditampilkan halaman home *website,* setelah *system* menampilkan halaman home pada *website user* dapat mengakses *login* dan akan di arahkan ke halaman *login*, saat dihalaman login *user* mengisi email dan password. Jika *user* gagal saat melakukan *login* akan kembali lagi pada halaman *login* dan jika *user* berhasil melakukan *login, user* akan masuk dalam halaman home *website.*

**

Gambar 3. 10Activity Diagram User Login

#### **3.4.3.9** **Activity Diagram User Produk**

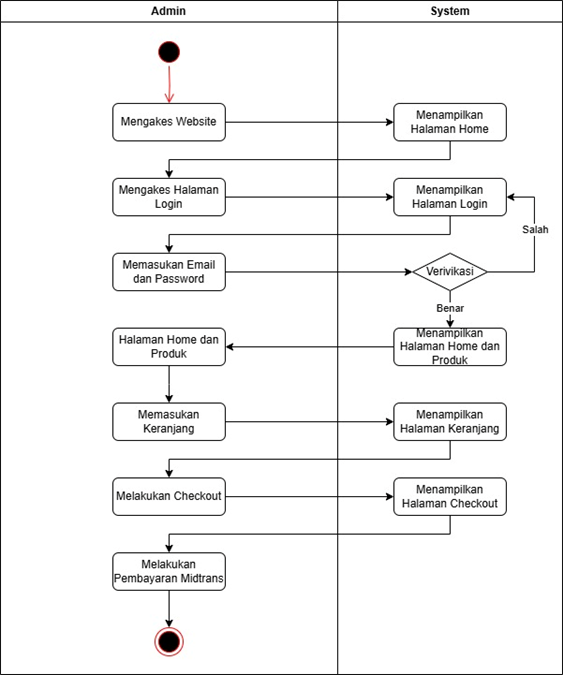
Gambar 3.11 menggambarkan ketika *user* memilih produk pada *website* Glasgow. Saat *user* memasuki *website* pertama kali akan di tampilkan halaman home dan sudah bisa melakukan melihat produk, yaitu dengan mengakses *website,* lalu *system* akan menampilkan halaman home, ketika *system* sudah menampilkan halaman home *user* dapat mengakses produk lalu di arahkan ke halaman produk, *user* bisa memilih produk dan di *system* akan menampilkan halaman detail produk.



Gambar 3. 11Activity Diagram User Produk

#### **3.4.2.10** **Activity Diagram Keranjang**

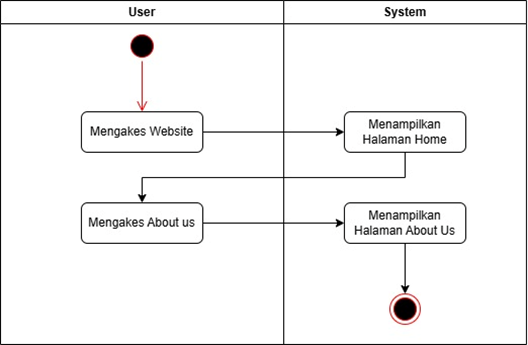
Pada Gambar 3.12 menggmbarkan *activity* diagram keranjang yang berhubungan dengan *checkout* pada *website* Glasgow. Pertama *user* diharuskan *login* terlebih dahulu dengan mengakses *login, user* memasukan email dan password pada halaman login. Ketika sudah mengisi email dan password dan berhasil *login* user di arahkan ke halaman home *website. User* dapat mengakses keranjang lalu oleh *system* akan dia arahkan ke halaman keranjang, *user* bisa melihat produk apa saja yang sudah ada dalam keranjang. Setelah sudah memasikan produk yang ada didalam keranjang untuk di *checkout user* dapat menekan tombol *checkout* pada keranjang dan akan diarahkan ke halaman *checkout* oleh *system* lalu user dapat melakukan transaksi pembayaran midtrans.



Gambar 3. 12Activity Diagram Keranjang

#### **3.4.2.11** **Activity Diagram About Us**

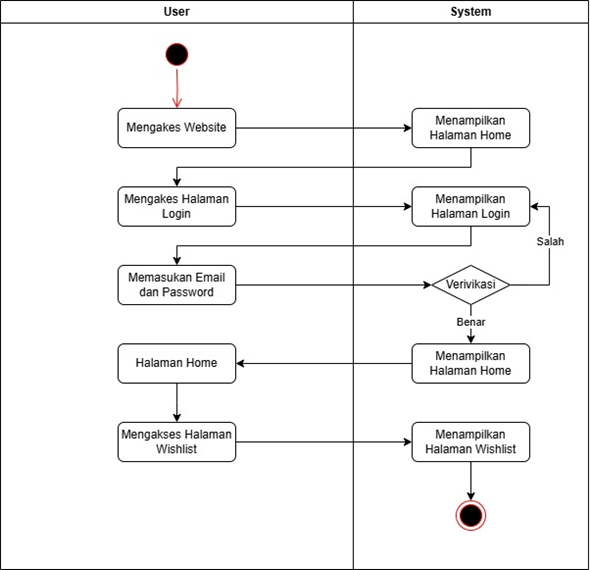
Pada gambar 3.13 ini mengambarkan *activity* diagram About Us. Pertama setelah mengakses *website system* akan menampilkan halaman home pada *website* glasgow, lalu setelah mengakses halaman about us halaman about us *system* akan menampilkan Halaman About Us.



Gambar 3. 13Activity Diagram About Us

#### **3.4.2.12** **Activity Diagram Wishlist**

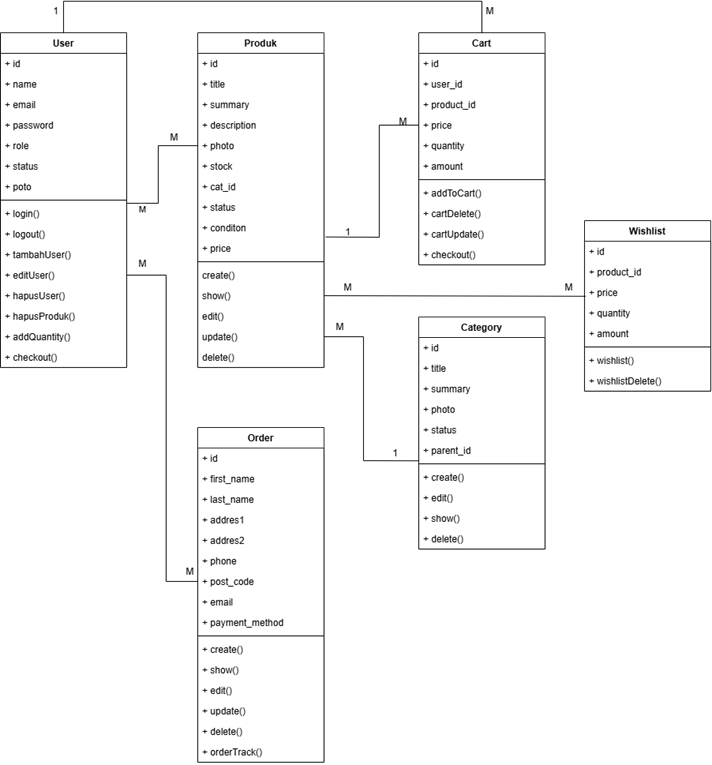
Pada gambar 3.14 menggambar kan *activity* diagram wishlist. Pada saat mengakses *website, system* akan menampilkan halaman home pada *website* glasgow. User mengakses halaman login terlebih dahulu dan *system* akan menampilkan halaman login jika *user* salah dalam memasukan *email* dan *password user* akan kembali di arahkan ke halaman login jika *user* benar mengisi *email* dan *password* akan menampilkan halaman home. Ketika berada di halaman home *website* setelah login *user* mengakses wishlist lalu *system* akan menampilkan halaman wishlist.



Gambar 3. 14Activity Diagram Wishlist

### **3.4.3** **Class Diagram**

*Class* diagram merupakan suatu gambaran struktur program, kelas, atribut, operasi dan hubungan antara objek. *Class* diagram pada *website* dirancang untuk memberikan pandangan yang jelas pada tentang bagaimana data fungsionalitas diorganisasikan dan berinteraksi satu sama lain. Berikut adalah *class* diagram yang terdapat pada *website* penjualan toko Glasgow yang dapat dilihat pada Gambar 3.15.



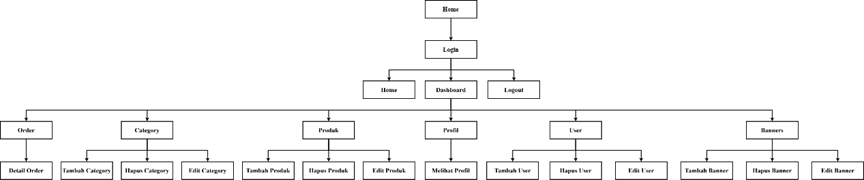
Gambar 3. 15Class Diagram

## **3.5** **Struktur Navigasi**

Pada tahap perancangan struktur navigasi ini, *website* Glasgow memiliki dua rancangan halaman *website*, di antaranya halaman admin dan user.

### **3.5.1** **Struktur Navigasi Admin**

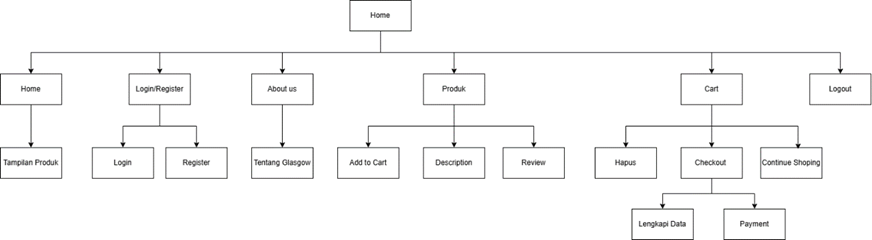
Pada Gambar 3.16 menunjukan struktur navigasi admin, penulis menggunakan navigasi hirarki, ketika admin *login* menggunakan akun admin maka akan langsung ke *home* admin. Pada dashboard admin dapat melihat menu, Produk, *Banner*, *User*, *Order*, dan juga *Logout*. Pada halaman produk admin dapat menambahkan, mengedit, dan menghapus produk, pada halaman *banner* admin dapat tambah *banner*, hapus *banner*, dan juga edit *banner*. Pada halaman *user* dapat melakukan tambah *user*, hapus *user*, dan edit *user*. Pada halaman *Order* admin dapat melihat pesanan secara detail.



Gambar 3. 16Struktur Navigasi Admin

### **3.5.2** **Struktur Navigasi User**

Pada struktur navigasi *user* penulis menggunakan struktur hirarki yang dikenal sebagai struktur bercabang. Struktur navigasi ini dibuat untuk menjelaskan tampilan menu dari sisi *user*. Pada saat *user* memasuki home pada *website user* sudah dapat melihat tampilan produk, melakukan *login/*register, *About Us*, Produk, *cart*, *Logout*. *User* diharuskan login terlebih dahulu agar dapat mengakses halaman produk lalu melakukan add to cart, dan pada halaman *cart user* dapat melakukan hapus, *checkout*, continue shoping pada saat mengakses *checkout user* dapat melengkapi data dan juga melakukan *payment*. Tampilan struktur navigasi *user* akan ditunjukan pada Gambar 3.17.



Gambar 3. 17Struktur Navigasi User

## **3.6** **Struktur Database**

Pada Pada tahapan ini, perancangan *database* digunakan untuk memahami struktur dari *database* pada *website* Glasgow. Pada *website* ini terdiri dari beberapa tabel antara lain Tabel user dan admin, tabel produk, tabel orders, tabel whislist, tabel produk review, tabel cart, tabel settings, dan tabel banners.

### **3.6.1** **Tabel User dan Admin**

Tabel user dan admin adalah tabel untuk menyimpan data-data user dan admin seperti pada Tabel 3.1.

Tabel 3. 1 Tabel User dan Admin

| **Field** | **Tipe Data** | **Keterangan** |
| --- | --- | --- |
| I id | Int(20) | Primary,Auto increment |
| E email | Varchar(191) | Foreign |
| nama | Varchar(191) | F Menyimpan nama |
| password | Varchar(191) | Menyimpan passowrd |
| foto | Varchar(191) | Menyimpan foto |
| role | enum | (‘user’,’admin’) |
| created\_at | ti timestamp | Menyimpan timestamp kapan dibuat |
| update\_at | timestamp | Menyimpan timestamp kapan di update |

### **3.6.2** **Tabel Produk**

Tabel produk ini berfungsi untuk menyimpan data produk yang seperti pada Tabel 3.2.

Tabel 3. 2Tabel Produk

| **Field** | **Tipe Data** | **Keterangan** |
| --- | --- | --- |
| id | Int(20) | Primary,Auto increment |
| title | Varchar(191) | Menyimpan title |
| foto | Varchar(191) | F Menyimpan foto |
| deskripsi | text | Menyimpan deskripsi |
| price | Double(8,2) | Menyimpan harga produk |
| stok | Int(11) | Menyimpan stok produk |
| status | ti enum | (‘aktif’,’noaktif’) |
| created\_at | timestamp | Menyimpan timestamp kapan dibuat |
| update\_at | timestamp | Menyimpan timestamp kapan diupdate |

### **3.6.3** **Tabel Orders**

Tabel orders bisa dilihat pada Tabel 3.3.

Tabel 3. 3 Tabel Orders

| **Field** | **Tipe Data** | **Keterangan** |
| --- | --- | --- |
| id | Int(20) | Primary,Auto increment |
| order\_number | Varchar(191) | Foreign |
| user\_id | Varchar(20) | Foreign |
| first\_name | Varchar(191) | Menyimpan data first\_name |
| last\_name | Varchar(191) | Menyimpan data last\_name |
| email | Varchar(191) | Menyimpan email |
| phone | Varchar(191) | Menyimpan no telp |
| Payment\_status | enum(‘paid’,’unpaid’) | Menyimpan data payement\_status |
| addres | Ti text | Menyimpan data alamat |
| quantity | Int(11) | Menyimpan data jumlah |
| created\_at | timestamp | Menyimpan timestamp kapan dibuat |
| update\_at | timestamp | Menyimpan timestamp kapan diupdate |

### **3.6.4** **Tabel Whistlist**

Tabel yang menunjukan data whistlist bisa dilihat pada Tabel 3.4.

Tabel 3. 4 Tabel Whistlist

| **Field** | **Tipe Data** | **Keterangan** |
| --- | --- | --- |
| id | Int(20) | Primary,Auto increment |
| produk\_id | Varchar(20) | Foreign |
| user\_id | Varchar(20) | Foreign |
| cart\_id | Varchar(20) | Foreign |
| price | Double(8,2) | F Menyimpan data harga di wishlist |
| quantity | Int(11) | Menyimpan data jumlah di wishlist |
| created\_at | timestamp | Menyimpan timestamp kapan dibuat |
| update\_at | timestamp | Menyimpan timestamp kapan diupdate |

### **3.6.5** **Tabel Produk Review**

Tabel ini menyimpan data produk review yang masuk pada produk seperti yang ditunjukan pada Tabel 3.5.

Tabel 3. 5Tabel Produk Review

| **Field** | **Tipe Data** | **Keterangan** |
| --- | --- | --- |
| id | Int(20) | Primary,Auto increment |
| user\_id | Varchar(20) | Foreign |
| produk\_id | Varchar(20) | Foreign |
| review | Text | F Menyimpan data review |
| created\_at | timestamp | Menyimpan timestamp kapan dibuat |
| update\_at | timestamp | Menyimpan timestamp kapan diupdate |

### **3.6.6** **Tabel Cart**

Tabel cart untuk menyimpan data yang berada di cart seperti pada Tabel 3.6

Tabel 3. 6Tabel Cart

| **Field** | **Tipe Data** | **Keterangan** |
| --- | --- | --- |
| id | Int(20) | Primary,Auto increment |
| user\_id | Varchar(20) | Foreign |
| order\_id | Varchar(20) | Foreign |
| produk\_id | Varchar(20) | Foreign |
| price | double(8,2) | F Menyimpan data harga di keranjang |
| quantity | Int(11) | Menyimpan jumlah produk di keranjang |
| created\_at | timestamp | Menyimpan timestamp kapan dibuat |
| update\_at | timestamp | Menyimpan timestamp kapan diupdate |

### **3.6.7** **Tabel Settings**

Tabel settings berfungsi untuk menyimpan data pada settings dan akan ditunjukan seperti Tabel 3.7

Tabel 3. 7Tabel Settings

| **Field** | **Tipe Data** | **Keterangan** |
| --- | --- | --- |
| id | Int(20) | Primary,Auto increment |
| deskripsi | text | Menyimpan deskripsi |
| logo | Varchar(191) | Menyimpan logo |
| foto | Varchar(191) | F Menyimpan foto |
| addres | Varchar(191) | Menyimpan alamat |
| phone | Varchar(191) | Menyimpan no telp |
| created\_at | timestamp | Menyimpan timestamp kapan dibuat |
| update\_at | timestamp | Menyimpan timestamp kapan diupdate |

### **3.6.8** **Tabel Banners**

Tabel banners berfungsi menyimpan data pada banners seperti contoh pada Tabel 3.8

Tabel 3. 8Tabel Settings

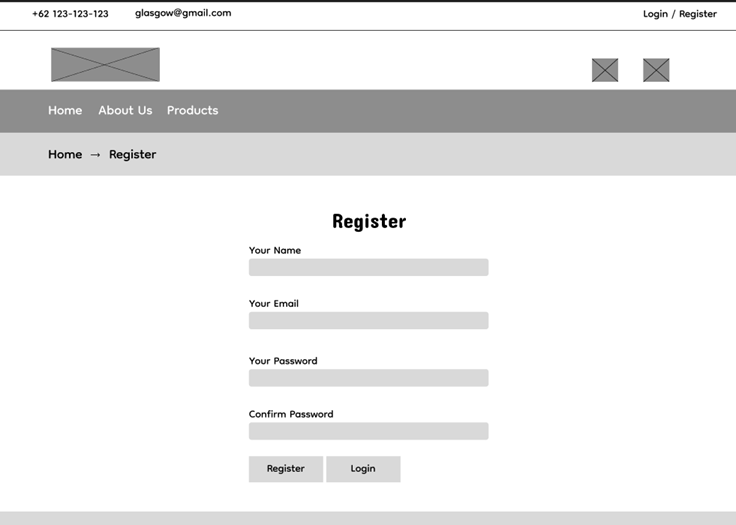
| **Field** | **Tipe Data** | **Keterangan** |
| --- | --- | --- |
| id | Int(20) | Primary,Auto increment |
| title | Varchar(191) | Menyimpan title |
| foto | Varchar(191) | F Menyimpan foto |
| deskripsi | text | Menyimpan deskripsi |
| status | Enum(‘active’,’inative’) | Menyimpan status banners |
| created\_at | timestamp | Menyimpan timestamp kapan dibuat |
| update\_at | timestamp | Menyimpan timestamp kapan diupdate |

## **3.7** **Perancangan Tampilan Website**

Pada tahapan perancangan tampilan *website* ini menggunakan aplikasi figma, yang merupakan pembuatan rancangan tampilan *website* yang bisa diakses secara *online*, maupun dengan di*install*. Berikut adalah hasil rancangan tampilan *website* sistem menggunakan *wireframe* yang memberikan banyak gambaran awal *layout website* secara umum.

### **3.7.1** **Rancangan Halaman *Register***

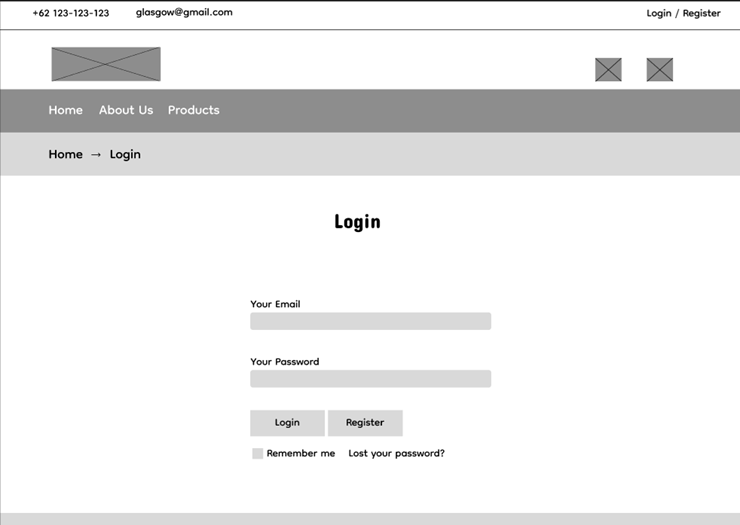
Pada gambar 3.18 menunjukan *wireframe* atau rancangan pada halaman *Register*. Pada halaman *register* ini ada beberapa field yaitu your name, your email, your password, dan juga confirm password.



Gambar 3. 18Rancangan Halaman Register

### **3.7.2** **Rancangan Halaman Login**

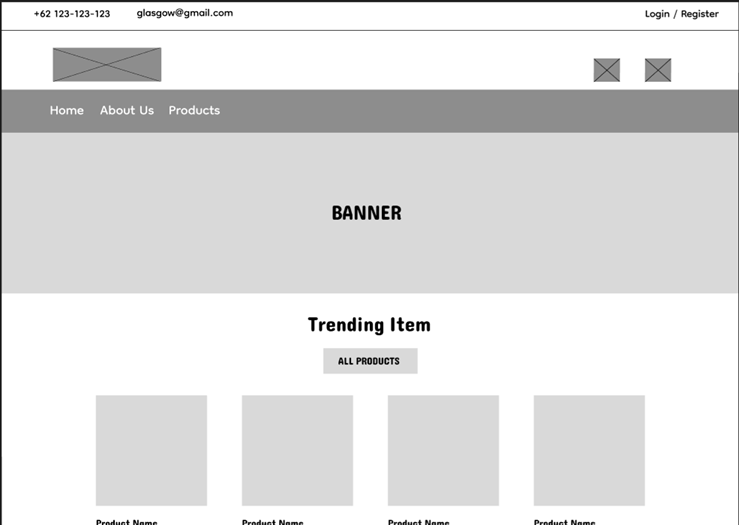
Pada gambar 3.19 menunjukan *wireframe* atau rancangan pada halaman Login. Pada halaman login terdapat field your email, dan your password. Di halaman email ini juga ada tombol login dan register.



Gambar 3. 19Rancangan Halaman Login

### **3.7.3** **Rancangan Halaman Home**

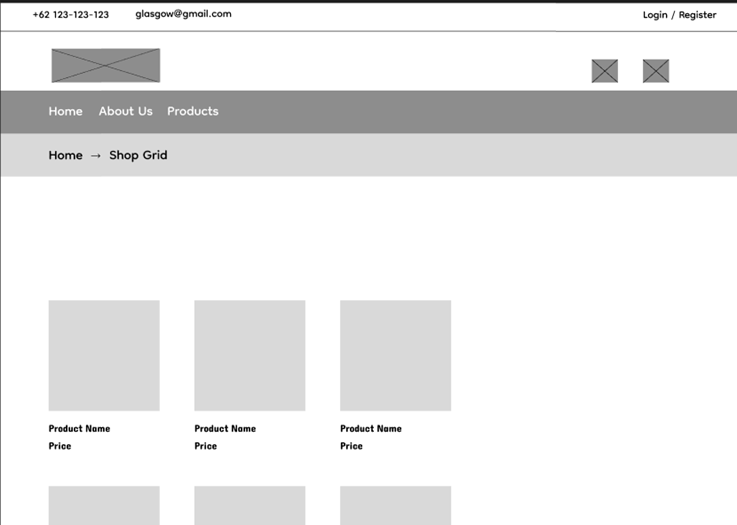
Pada gambar 3.20 menunjukan *wireframe* atau rancangan pada halaman Home *website*. Pada halaman home ini yang merupakan tampilan utama *website* ada tombol login / *register,* Home, About Us, dan juga Product.



Gambar 3. 20Rancangan Halaman Home

### **3.7.4** **Rancangan Halaman Produk**

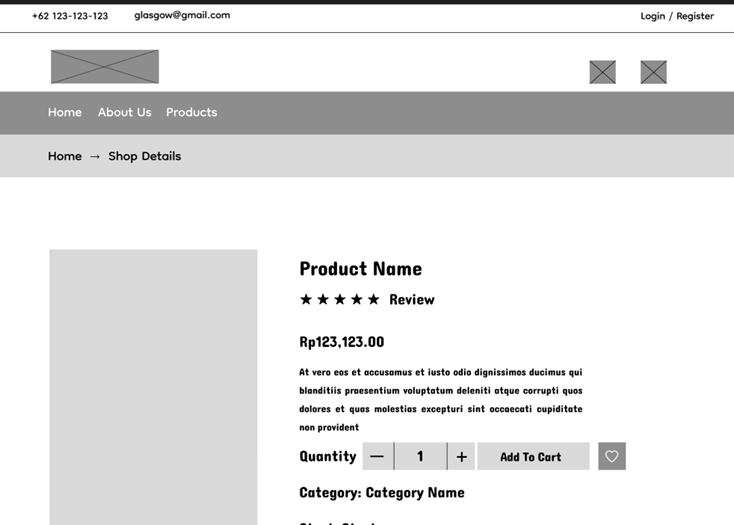
Pada gambar 3.21 menunjukan *wireframe* atau rancangan pada halaman Produk. Di halaman produk menampilkan produk yang tersedia pada toko glasgow ini.



Gambar 3. 21Rancangan Halaman Produk

### **3.7.5** **Rancangan Halaman Produk Detail**

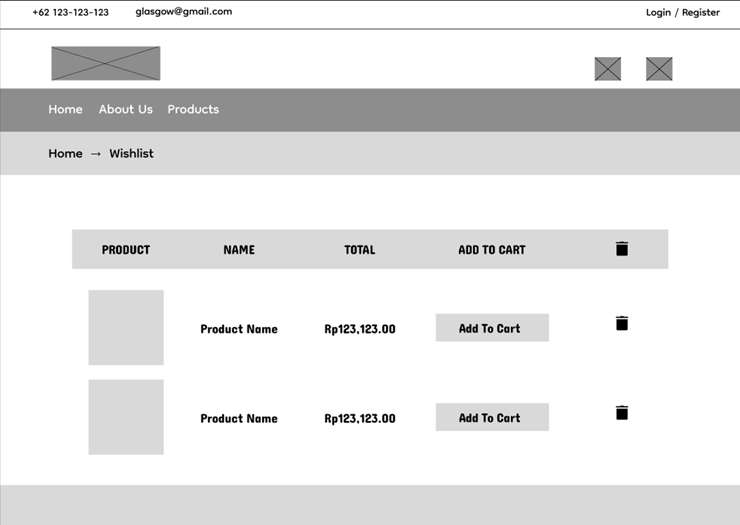
Pada gambar 3.22 menunjukan *wireframe* atau rancangan pada halaman Produk Detail. Pada halaman produk detail ini menunjukan bagaiman produk detail dan ada beberapa fitur seperti quantity, add to cart, dan juga wishlist.



Gambar 3. 22Rancangan Halaman Produk Detail

### **3.7.6** **Rancangan Halaman Whislist**

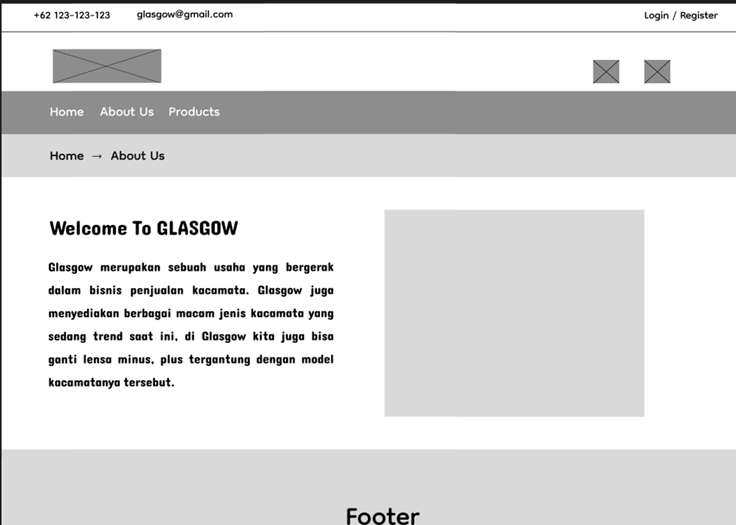
Pada gambar 3.23 menunjukan *wireframe* atau rancangan pada halaman Wishlist. Pada halaman wishlist ini menampilkan produk yang sudah *user* masukan kedalam wishlist dan di halaman ini juga ada fitur add to cart dan delete.



Gambar 3. 23Rancangan Halaman Whislist

### **3.7.7** **Rancangan Halaman About Us**

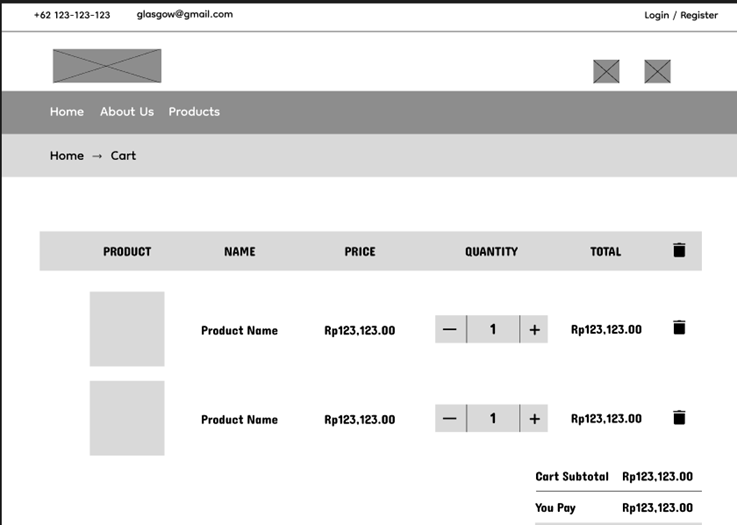
Pada gambar 3.24 menunjukan *wireframe* atau rancangan pada halaman About Us. Halaman about us ini menampilkan tentang toko glasgow dan menjelaskan apa itu toko glasgow.



Gambar 3. 24Rancangan Halaman About Us

### **3.7.8** **Rancangan Halaman Cart**

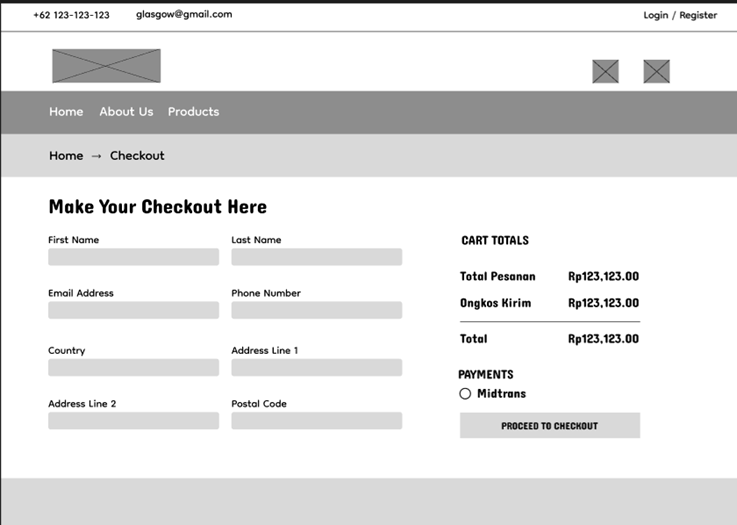
Pada gambar 3.25 menunjukan *wireframe* atau rancangan pada halaman Cart. Pada halaman cart ini menampilkan halaman produk yang sudah di masukan dalam cart. Dalam halaman ini *user* juga dapat menambahkan jumlah produk yang sudah masuk di dalam cart dan juga menghapus produk yang sudah berada dalam halaman cart.



Gambar 3. 25 Rancangan Halaman Keranjang

### **3.7.9** **Rancangan Halaman Checkout**

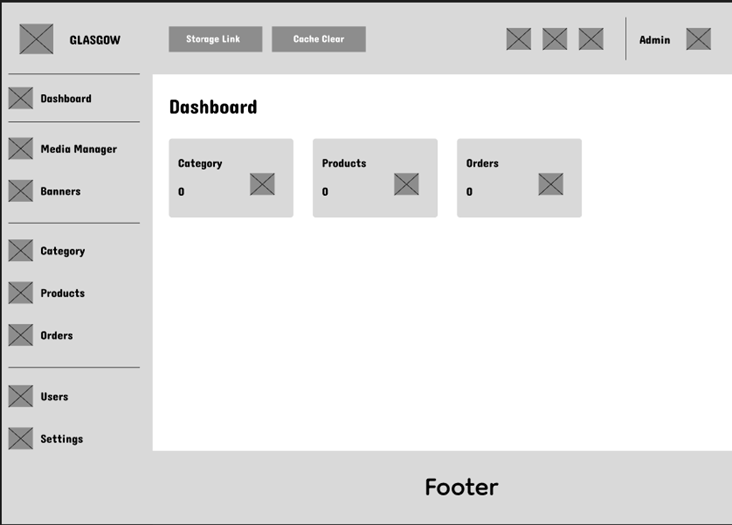
Pada gambar 3.26 menunjukan *wireframe* atau rancangan pada halaman Checkout. Pada halaman checout ini ada beberapa field yang dimana *user* harus mengisi untuk keterangan alamat dan juga nama. Di halaman checout juga ada fitur pembayaran seperti midtrans lalu proceed to checkout untuk menampilkan pembayaran lebih lanjut.



Gambar 3. 26Rancangan Halaman Checkout

### **3.7.10** **Rancangan Halaman Admin Dashboard**

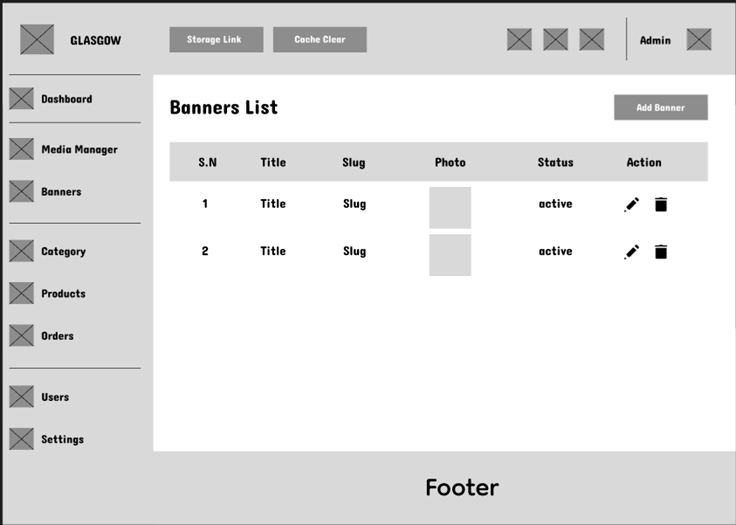
Pada gambar 3.27 menunjukan *wireframe* atau rancangan pada halaman Admin Dashboard. Halaman dashboard admin ini ada beberapa fitur seperti banners untuk mengubah banners, category untuk mengubah category, produk untuk mengubah atau edit produk yang terdapat pada *website,* orders untuk melihat pesanan yang sudah di pesan oleh pembeli, users untuk melihat akun yang sudah mendaftarkan dalam *website*, dan juga settings untuk mengubah tampilan pada halaman *website.*

**

Gambar 3. 27Rancangan Halaman Admin Dashboard

### **3.7.11** **Rancangan Halaman Admin Banners**

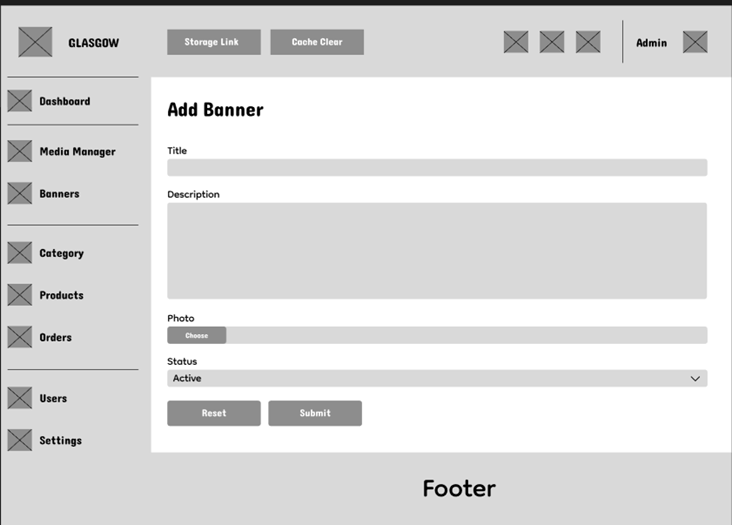
Pada gambar 3.28 menunjukan *wireframe* atau rancangan pada halaman Admin *Banneers*. Halaman banners ini ada fitur edit, add banners, dan juga delete.



Gambar 3. 28Rancangan Halaman Admin Banners

### **3.7.12** **Rancangan Halaman Add Banners**

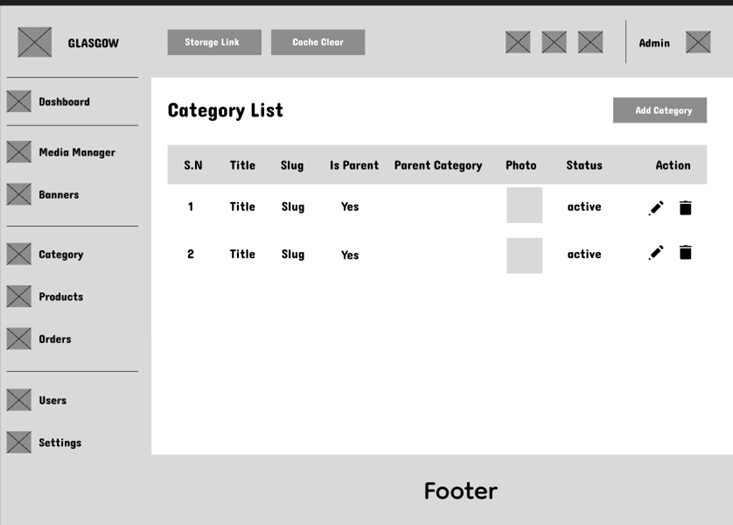
Pada gambar 3.29 menunjukan *wireframe* atau rancangan pada halaman Add Banners yang berada pada admin banners. Pada halaman add banners ini ada field title, description, dan photo. Di halaman add banners ini juga ada fitur status, reset, dan juga submit.



Gambar 3. 29 Rancangan Halaman Add Banners

### **3.7.13** **Rancangan Halaman Admin Category**

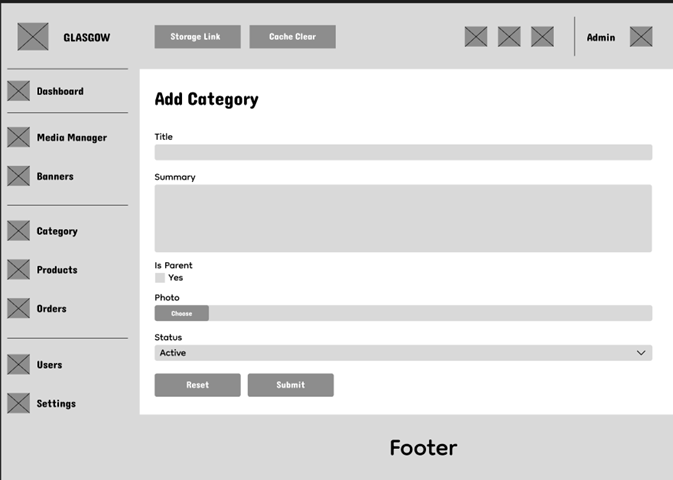
Pada gambar 3.30 menunjukan *wireframe* atau rancangan pada halaman *Category List*. Pada halaman category list ini ada fitur add category, delete, dan juga delete.



Gambar 3. 30Rancangan Halaman Admin Category

### **3.7.14** **Racangan Halaman Add Category**

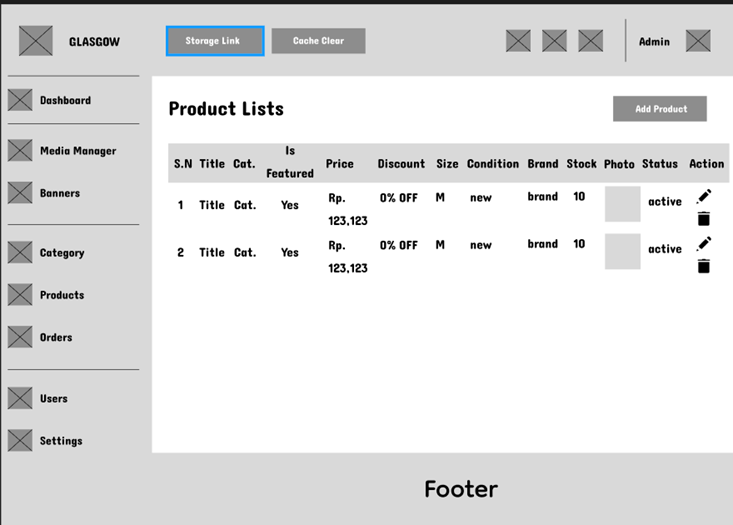
Pada gambar 3.31 menunjukan *wireframe* atau rancangan pada halaman Add *Category*. Dalam halaman add category ada beberapa field yaitu title, summary, dan juga photo. Halaman ini juga ada fitur reset dan juga submit.



Gambar 3. 31Rancangan Halaman Add Category

### **3.7.15** **Rancangan Halaman Admin Produk**

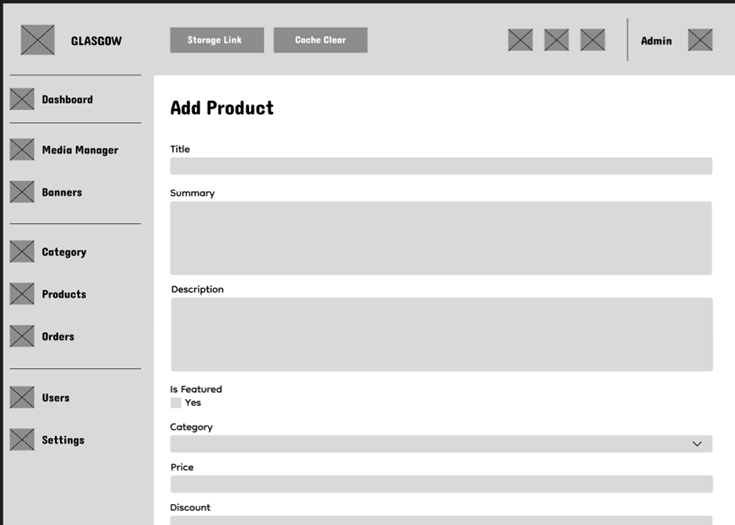
Pada gambar 3.32 menunjukan *wireframe* atau rancangan pada halaman Admin Produk. Pada halaman admin produk ada beberapa fitur yaittu add produk, edit, dan juga delete.



Gambar 3. 32Rancangan Halaman Admin Produk

### **3.7.16** **Rancangan Halaman Add Produk**

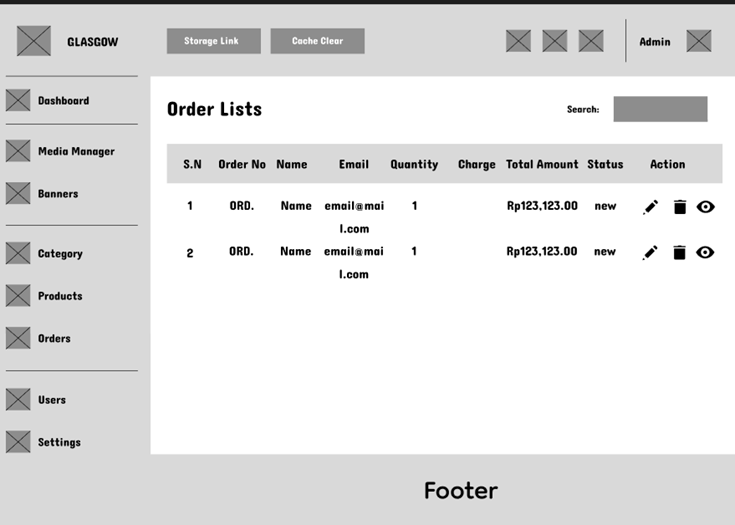
Pada gambar 3.33 menunjukan *wireframe* atau rancangan pada halaman Add Produk yang berada di halaman admin produk. Pada halaman add produk ini ada beberapa field yaitu field, summary, description, price, dan juga photo.



Gambar 3. 33Rancangan Halaman Add Produk

### **3.7.17** **Racangan Halaman Admin Orders**

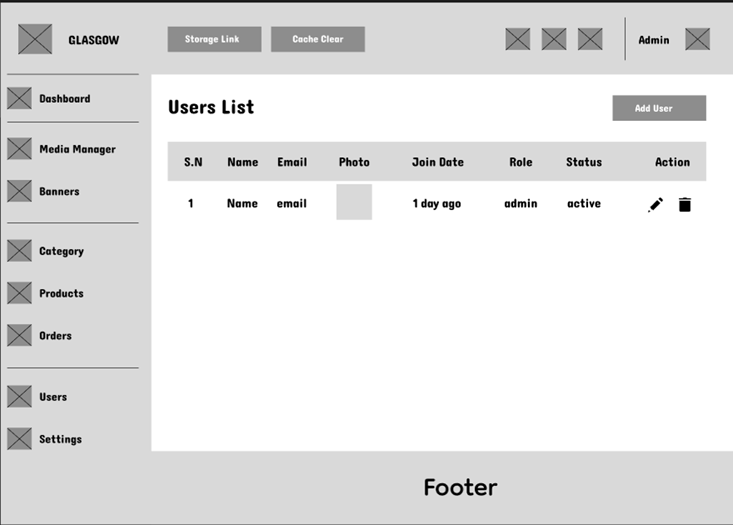
Pada gambar 3.34 menunjukan *wireframe* atau rancangan pada halaman Admin *Orders*. Halaman orders ini ada beberapa fitur seperti edit, delete, dan juga view untuk melihat data order.



Gambar 3. 34Rancangan Halaman Admin Orders

### **3.7.18** **Rancangan Halaman User List**

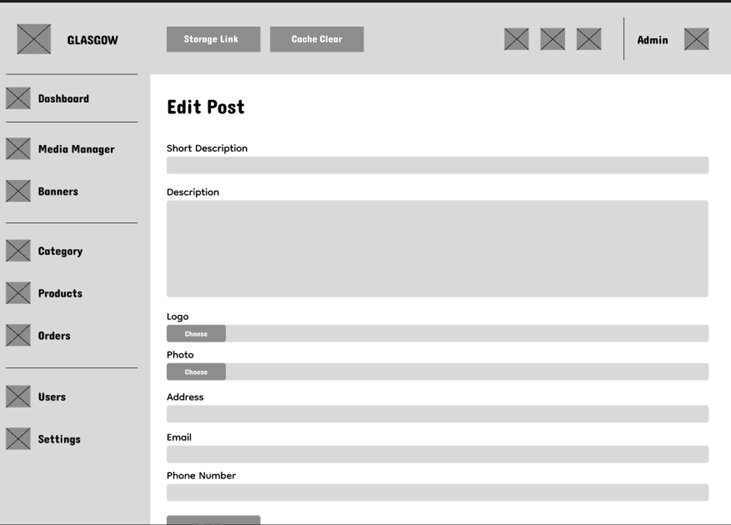
Pada gambar 3.35 menunjukan *wireframe* atau rancangan pada halaman *User List*. Pada halaman user list menampilkan user yang sudah terdaftar pada *website* dan juga ada beberapa fitur seperti add user, edit dan delete.



Gambar 3. 35Rancangan Halaman User List

### **3.7.19** **Rancangan Halaman Settings**

Pada gambar 3.36 menunjukan *wireframe* atau rancangan pada halaman *Settings*. Pada halaman settings ini ada beberapa field yaitu seperti short description, description, addres, email, dan juga phone number.



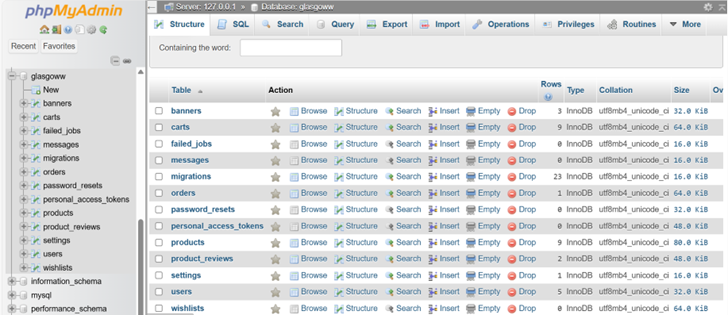
Gambar 3. 36Rancangan Halaman Settings

## **3.8** **Implementasi Database**

Pada tahap ini pembuatan database untuk *website* Glasgow menggunakan database MySQL. Dalama database ini akan menyimpan semua data yang diperlukan pada *website.*

### **3.8.1** **Pembuatan Database**

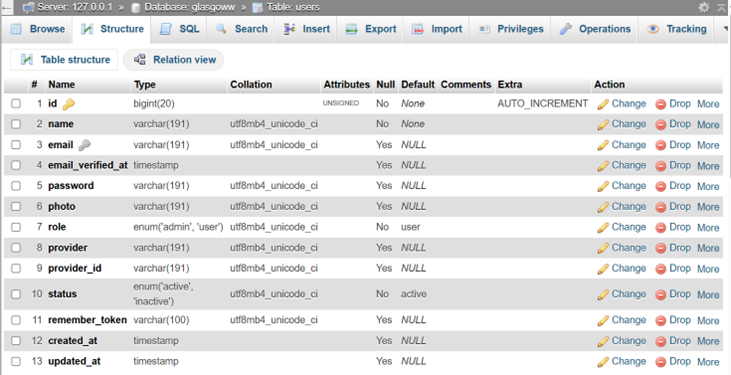
Pada gambar 3.37 ini menunjukan pembuatan database untuk *website* Glasgow dengan menggunakan MySQL. Database yang digunakan ini bernama glasgow. Dalam database ini berjumlah 14 yang berisikan : *banners,* *cart*, *failed\_job*, *messages*, *migrations*, *orders*, *password\_reset*, *perosnal\_akses\_token*, *products*, *product\_reviews*, *settings*, *user*, dan *wishlist*.



Gambar 3. 37Database Glasgow

### **3.8.2** **Pembuatan Tabel User**

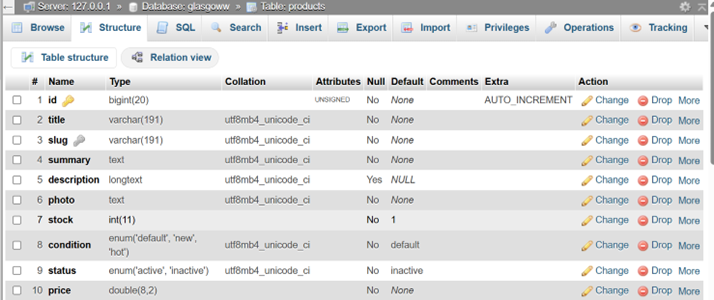
Pada gambar 3.38 ini menunjukan pembuatan tabel *user*. Tabel user ini berfungsi untuk menyimpan data *user* yang telah login pada *website* Glasgow ini.



Gambar 3. 38Tabel User

### **3.8.3** **Pembuatan Tabel Produk**

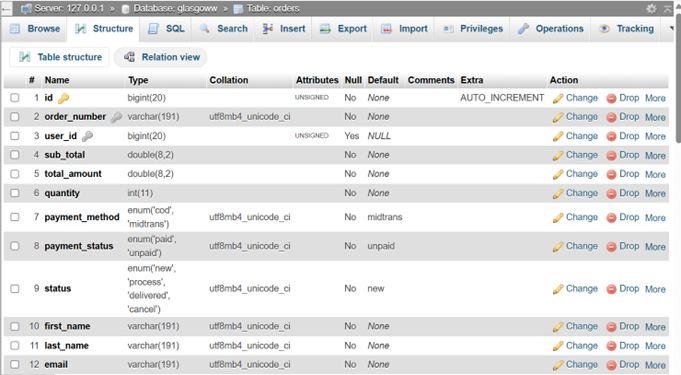
Pada gambar 3.39 menunjukan tabel database pada tabel produk. Tabel produk ini berfungsi untuk menyimpan data pada produk.



Gambar 3. 39Tabel Produk

### **3.8.4** **Pembuatan Tabel Orders**

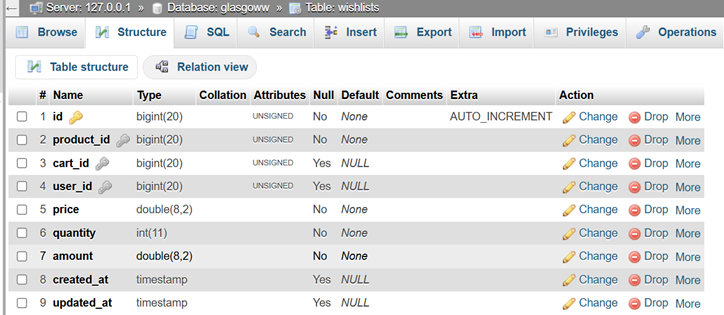
Pada gambar 3.40 ini menunjukan tabel database pada orders. Tabel orders ini berfungsi untuk proses ketika *user* ingin melakukan transaksi.



Gambar 3. 40Tabel Orders

### **3.8.5** **Pembuatan Tabel Wishtlist**

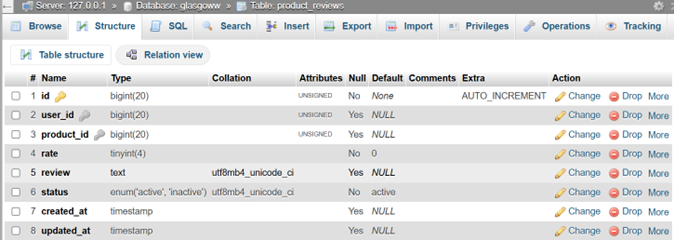
Pada gambar 3.41 ini menunjukan tabel database wishlist. Tabel wishlist ini befungsi untuk menyimpan data produk yang sudah dimasukan user kedalam wishlist.



Gambar 3. 41Tabel Wishlist

### **3.8.6** **Pembuatan Tabel Produk Review**

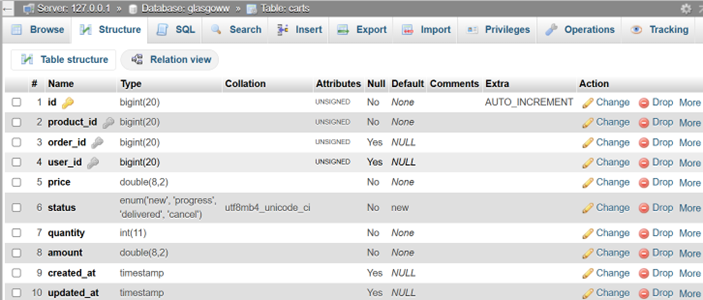
Pada gambar 3.42 ini menunjukan tabel database pada produk review. Tabel produk review ini berfungsi untuk menyimpan data review produk yang telah diberikan oleh *user.*

**

Gambar 3. 42Tabel Produk Review

### **3.8.7** **Pembuatan Tabel Cart**

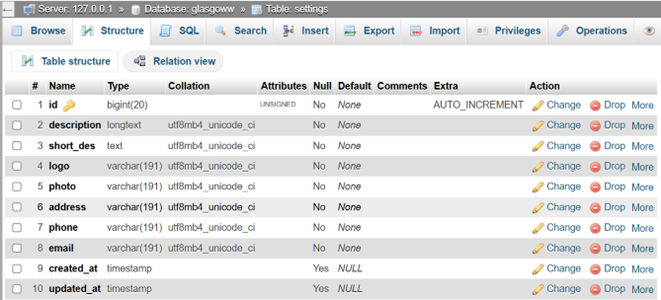
Gambar 3.43 menunjukan tabel database cart. Tabel cart ini befungsi untuk menyimpan data produk yang dimasukan oleh *user* kedalam keranjang belanja.



Gambar 3. 43Tabel Cart

### **3.8.8** **Pembuatan Tabel Settings**

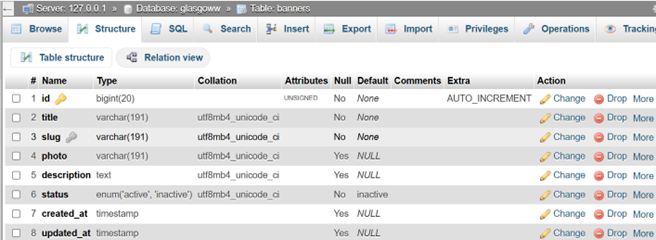
Pada gambar 3.44 menunjukan tabel database settings. Tabel database settings ini berfungsi pada halaman dashboard admin untuk mengubah tampilan halaman pada *website*.



Gambar 3. 44Tabel Settings

### **3.8.9** **Pembuatan Tabel Banners**

Gambar 3.45 ini menunjukan tabel database pada banners. Tabel banners ini berfungsi untuk membuat iklan promo pada *website.*

**

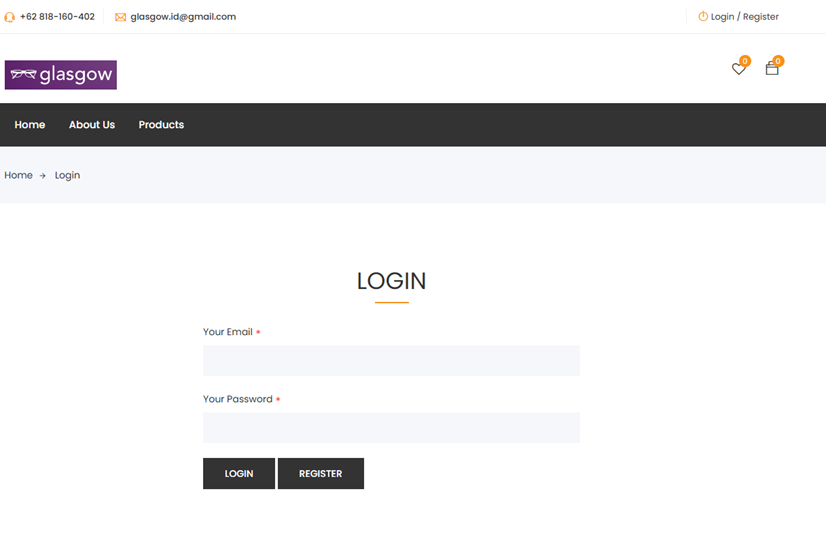
Gambar 3. 45Tabel Banners

## **3.9** **Implementasi Tampilan Website**

Pada tahapan ini pembuatan *website* Glasgow menggunakan bahasa pemograman PHP dengan framework laravel dan MySQL sebagai database yang digunakan. Dibawah ini merupakan tampilan pada *website* Glasgow.

### **3.9.1** **Halaman Login**

Gambar 3.46 menunjukan halaman login yang dimana user melakukan pengisian email dan password untuk dapat login. Masukan email dan password yang sudah di daftarkan supaya dapat mengakses halaman *website*.



Gambar 3. 46Halaman Login

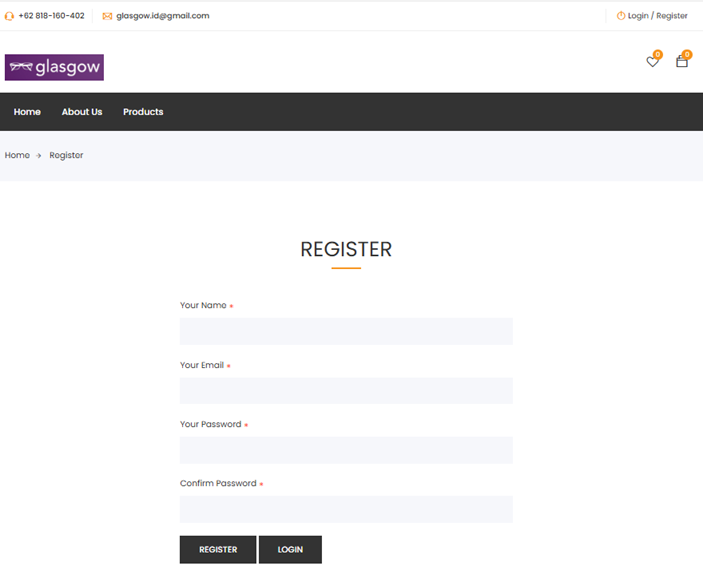
#### **3.9.1.1** **Kode Program Halaman Login**

Pada pembahasan ini menunjukan kode dari bagian halaman form login pada *website*. Kode ini adalah form login yang mengirim data ke rute ‘login.submit’ menggunakan metode POST, dengan menyertakan token CSRF untuk keamanan. Dalam form login ini memiliki dua field input yaitu email dan password, yang masing masing di lengkapi validasi untuk menampilkan pesan *error* jika input tidak sesuai. Ada tombol login untuk mengirimkan form dan juga tombol *register* yang mengarahkan *user* ke form *register* jika belum memiliki akun.

| <form class="form" method="post" action="{{route('login.submit')}}">  @csrf  <div class="row">  <div class="col-12">  <div class="form-group">  <label>Your Email<span>\*</span></label>  <input type="email" name="email" placeholder="" required="required" value="{{old('email')}}">  @error('email')  <span class="text-danger">{{$message}}</span>  @enderror  </div>  </div>  <div class="col-12">  <div class="form-group">  <label>Your Password<span>\*</span></label>  <input type="password" name="password" placeholder="" required="required" value="{{old('password')}}">  @error('password')  <span class="text-danger">{{$message}}</span>  @enderror  </div>  </div>  <div class="col-12">  <div class="form-group login-btn">  <button class="btn" type="submit">Login</button>  <a href="{{route('register.form')}}" class="btn">Register</a>  </form>  </div>  </div>  </div>  </div>  </section> |
| --- |

### **3.9.2** **Halaman *Register***

Gambar 3.47 menunjukan halaman *register* pada *website* . *User* dapat melakukan *register* dengan mengisi form *register* yang tersedia, yaitu ada mengisi nama, mengisi email, mengisi password, dan juga *confirm* password. Setelah mengisi form pada *register user* dapat melakukan login.



Gambar 3. 47Halaman Register

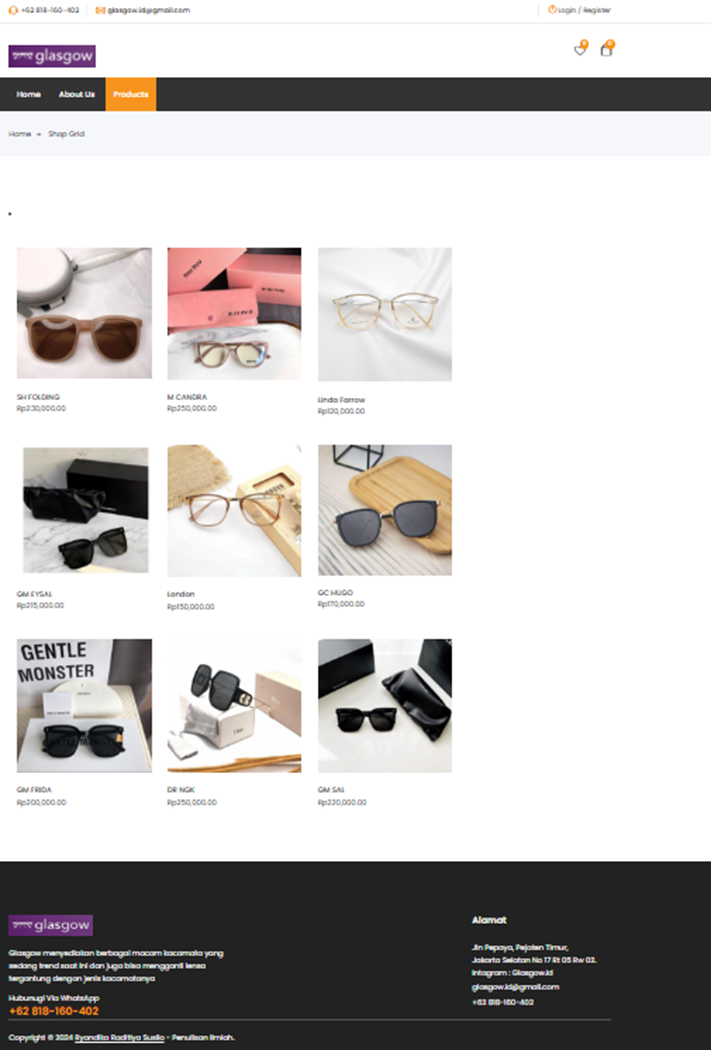
#### **3.9.2.1** **Kode Program Halaman Register**

Pada pembahasan ini menunjukan kode halaman *register* pada *website*. Kode ini adalah form *register* yang mengirim data ke rute ‘register.submit’ menggunakan metode POST, dilengkapi dengan token CSRF untuk keamanan. Form ini memiliki field input untuk nama, email, password, dan juga konfirmasi password, semuanya wajib diisi dan disertai validasi untuk menampilkan pesan *error* jika terjadi kesalahan input.

| <form class="form" method="post" action="{{route('register.submit')}}">  @csrf  <div class="row">  <div class="col-12">  <div class="form-group">  <label>Your Name<span>\*</span></label>  <input type="text" name="name" placeholder="" required="required" value="{{old('name')}}">  @error('name')  <span class="text-danger">{{$message}}</span>  @enderror  </div>  </div>  <div class="col-12">  <div class="form-group">  <label>Your Email<span>\*</span></label>  <input type="text" name="email" placeholder="" required="required" value="{{old('email')}}">  @error('email')  <span class="text-danger">{{$message}}</span>  @enderror  </div>  </div>  <div class="col-12">  <div class="form-group">  <label>Your Password<span>\*</span></label>  <input type="password" name="password" placeholder="" required="required" value="{{old('password')}}">  @error('password')  <span class="text-danger">{{$message}}</span>  @enderror  </div>  </div>  <div class="col-12">  <div class="form-group">  <label>Confirm Password<span>\*</span></label>  <input type="password" name="password\_confirmation" placeholder="" required="required" value="{{old('password\_confirmation')}}">  @error('password\_confirmation')  <span class="text-danger">{{$message}}</span>  @enderror  </div>  </div>  <div class="col-12">  <div class="form-group login-btn">  <button class="btn" type="submit">Register</button>  <a href="{{route('login.form')}}" class="btn">Login</a>  </div>  </div>  </div>  </form> |
| --- |

### **3.9.3** **Produk**

Pada gambar 3.48 ini menunjukan halaman produk pada *website*. *User* dapat melihat produk yang tersedia pada *website* glasgow dan dapat memilih produk.



Gambar 3. 48Produk

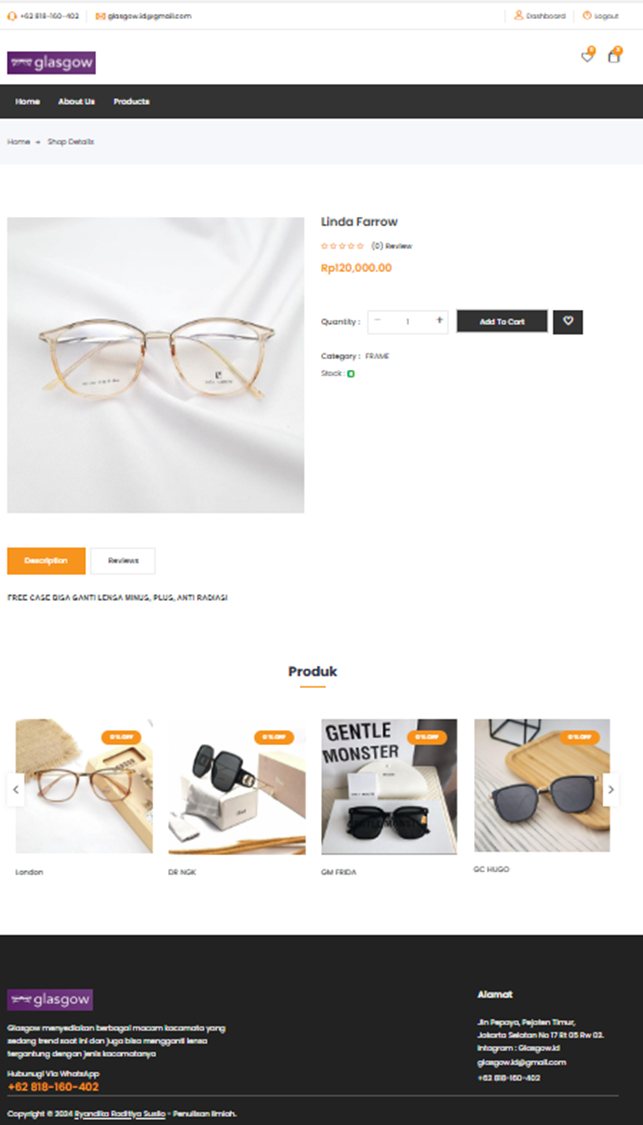
#### **3.9.3.1** **Kode Program Halaman Produk**

Pada pembahasan ini menjelaskan kode pada halaman produk. Disini ada <form action="{{route('shop.filter')}}" method="POST"> : Formulir ini digunakan untuk memfilter produk berdasarkan kriteria tertentu. Route shop.filter akan menangani pengiriman formulir. @csrf : Melindungi dari serangan CSRF dengan menambahkan token ke formulir. (@if(count($products)>0)), produk akan ditampilkan dalam grid.

| @extends('frontend.layouts.master')  @section('title','GLASGOW || PRODUCT PAGE')  @section('main-content')  <div class="breadcrumbs">  <div class="container">  <div class="row">  <div class="col-12">  <div class="bread-inner">  <ul class="bread-list">  <li><a href="index1.html">Home<i class="ti-arrow-right"></i></a></li>  <li class="active"><a href="blog-single.html">Shop Grid</a></li>  </ul>  </div>  </div>  </div>  </div>  </div>  <form action="{{route('shop.filter')}}" method="POST">  @csrf  <section class="product-area shop-sidebar shop section">  <div class="container">  <div class="row">  <div class="col-lg-3 col-md-4 col-12">  <div class="shop-sidebar">  <div>  @php  // $category = new Category();  $menu=App\Models\Category::getAllParentWithChild();  @endphp  @if($menu)  <li>  @foreach($menu as $cat\_info)  @if($cat\_info->child\_cat->count()>0)  <li><a href="{{route('product-cat',$cat\_info->slug)}}">{{$cat\_info->title}}</a>  <ul>  @foreach($cat\_info->child\_cat as $sub\_menu)  <li><a href="{{route('product-sub-cat',[$cat\_info->slug,$sub\_menu->slug])}}">{{$sub\_menu->title}}</a></li>  @endforeach  </ul>  </li>  @else  <li><a href="{{route('product-cat',$cat\_info->slug)}}">{{$cat\_info->title}}</a></li>  @endif  @endforeach  </li>  @endif  {{-- @foreach(Helper::productCategoryList('products') as $cat)  @if($cat->is\_parent==1)  <li><a href="{{route('product-cat',$cat->slug)}}">{{$cat->title}}</a></li>  @endif  @endforeach --}}  </ul>  </div>  {{-- {{dd($recent\_products)}} --}}  @foreach($recent\_products as $product)  @php  $photo=explode(',',$product->photo);  @endphp  @endforeach  </div>  </div>  </div>  <div class="col-lg-9 col-md-8 col-12">  <div class="row">  <div class="col-12">  </div>  </div>  <div class="row">  {{-- {{$products}} --}}  @if(count($products)>0)  @foreach($products as $product)  <div class="col-lg-4 col-md-6 col-12">  <div class="single-product">  <div class="product-img">  <a href="{{route('product-detail',$product->slug)}}">  @php  $photo=explode(',',$product->photo);  @endphp  <img class="default-img" src="{{$photo[0]}}" alt="{{$photo[0]}}">  <img class="hover-img" src="{{$photo[0]}}" alt="{{$photo[0]}}">  </a>  <div class="button-head">  <div class="product-action">  <a data-toggle="modal" data-target="#{{$product->id}}" title="Quick View" href="#"><i class=" ti-eye"></i><span>Quick Shop</span></a>  <a title="Wishlist" href="{{route('add-to-wishlist',$product->slug)}}" class="wishlist" data-id="{{$product->id}}"><i class=" ti-heart "></i><span>Add to Wishlist</span></a>  </div>  <div class="product-action-2">  <a title="Add to cart" href="{{route('add-to-cart',$product->slug)}}">Add to cart</a>  </div>  </div>  </div>  <div class="product-content">  <h3><a href="{{route('product-detail',$product->slug)}}">{{$product->title}}</a></h3>  @php  $after\_discount=($product->price-($product->price\*$product->discount)/100);  @endphp  <span>Rp{{number\_format($after\_discount,2)}}</span>  </div>  </div>  </div>  @endforeach  @else  <h4 class="text-warning" style="margin:100px auto;">There are no products.</h4>  @endif  </div>  <div class="row">  <div class="col-md-12 justify-content-center d-flex">  {{$products->appends($\_GET)->links()}}  </div>  </div>  </div>  </div>  </div>  </section>  </form> |
| --- |

### **3.9.4** **Produk Details**

Gambar 3.49 menunjukan halaman produk details pada *website* glasgow. Gambar ini menunjukan ketika *user* telah memilih produk dan akan keluar halaman produk detail pada produk tersebut.



Gambar 3. 49Produk Details

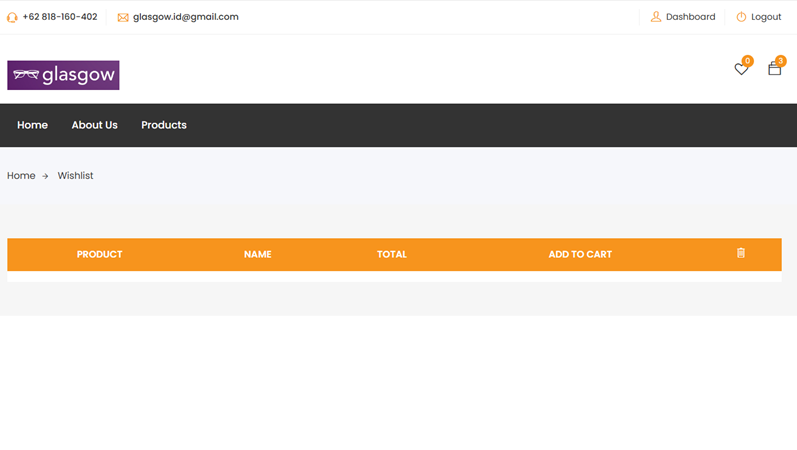
#### **3.9.4.1** **Kode Program Halaman Produk Details**

Pada pembahasa ini menjelaskan listing program pada halaman produk detail. Code program ini ($photo = explode(',', $product\_detail->photo)) berfungsi mengambil dan menampilkan gambar pertama dari daftar gambar produk yang dipisahkan oleh koma. Code program selanjutnya ($product\_detail->title) menjelaskan menampilkan judul produk. Tombol "Add to cart" untuk menambahkan produk ke keranjang belanja, serta tombol "Add to wishlist" untuk menambahkan produk ke daftar keinginan.

| @endsection  @section('title','GLASGOW || PRODUCT DETAIL')  @section('main-content')  <div class="breadcrumbs">  <div class="container">  <div class="row">  <div class="col-12">  <div class="bread-inner">  <ul class="bread-list">  <li><a href="{{route('home')}}">Home<i class="ti-arrow- right"></i></a></li>  <li class="active"><a href="">Shop Details</a></li>  </ul>  </div>  </div>  </div>  </div>  </div>  <section class="shop single section">  <div class="container">  <div class="row">  <div class="col-12">  <div class="row">  <div class="col-lg-6 col-12">  <div class="product-gallery">  <div class="flexslider-thumbnails">  <ul>  @php  $photo = explode(',', $product\_detail->photo);  $firstPhoto = $photo[0]; // Mengambil gambar pertama  @endphp  <img src="{{ $firstPhoto }}" alt="Product Image">  </ul>  </div>  </div>  </div>  <div class="col-lg-6 col-12">  <div class="product-des">  <div class="short">  <h4>{{$product\_detail->title}}</h4>  <div class="rating-main">  <ul class="rating">  @php  $rate=ceil($product\_detail->getReview->avg('rate'))  @endphp  @for($i=1; $i<=5; $i++)  @if($rate>=$i)  <li><i class="fa fa-star"></i></li>  @else  <li><i class="fa fa-star-o"></i></li>  @endif  @endfor  </ul>  <a href="#" class="total-review">({{$product\_detail['getReview']->count()}}) Review</a>  </div>  @php  @endphp  <p class="price"><span class="discount">Rp{{number\_format($after\_discount,2)}}</p>  </div>  <div class="product-buy">  <form action="{{route('single-add-to-cart')}}" method="POST">  @csrf  <div class="quantity">  <h6>Quantity :</h6>  div class="input-group">  <div class="button minus">  <button type="button" class="btn btn-primary btn-number" disabled="disabled" data-type="minus" data-field="quant[1]">  <i class="ti-minus"></i>  </button>  </div>  <input type="hidden" name="slug" value="{{$product\_detail->slug}}">  <input type="text" name="quant[1]" class="input-number" data-min="1" data-max="1000" value="1" id="quantity">  <div class="button plus">  <button type="button" class="btn btn-primary btn-number" data-type="plus" data-field="quant[1]">  <i class="ti-plus"></i>  </button>  </div>  </div>  </div>  <div class="add-to-cart mt-4">  <button type="submit" class="btn">Add to cart</button>  <a href="{{route('add-to-wishlist',$product\_detail->slug)}}" class="btn min"><i class="ti-heart"></i></a>  </div>  </form>  <p class="cat">Category :<a href="{{route('product-cat',$product\_detail->cat\_info['slug'])}}">{{$product\_detail->cat\_info['title']}}</a></p>  @if($product\_detail->sub\_cat\_info)  <p class="cat mt-1">Sub Category :<a href="{{route('product-sub-cat',[$product\_detail->cat\_info['slug'],$product\_detail->sub\_cat\_info['slug']])}}">{{$product\_detail->sub\_cat\_info['title']}}</a></p>  @endif  <p class="availability">Stock : @if($product\_detail->stock>0)<span class="badge badge-success">{{$product\_detail->stock}}</span>@else <span class="badge badge-danger">{{$product\_detail->stock}}</span> @endif</p>  </div> |
| --- |

### **3.9.5** **Wishlist**

Pada gambar 3.50 ini menunjukan halaman wishlist pada *website* glasgow. *User* bisa melihat produk yang sudah di wishlist atau produk yang disukai oleh *user* agar dapat memudahkan pada saat memilih produk yang akan dibeli.



Gambar 3. 50Halaman Wishlist

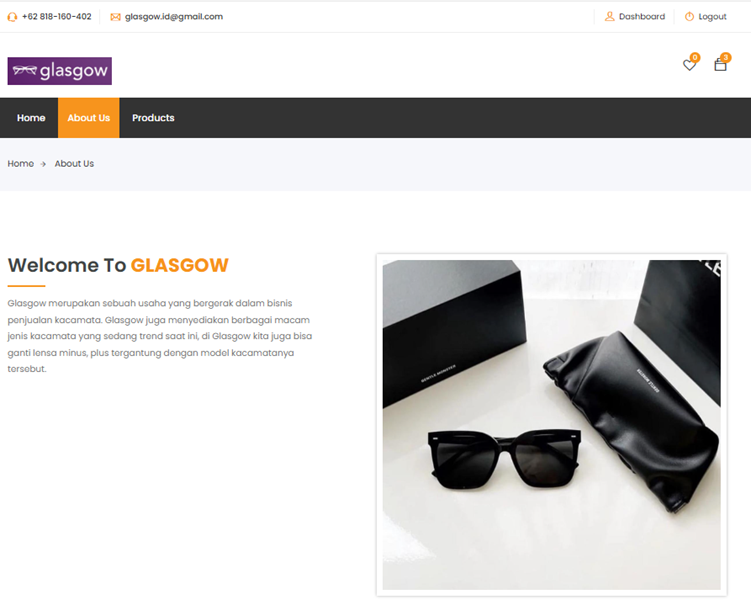
#### **3.9.5.1** **Kode Program Halaman Wishlist**

Pada penjelasan ini menjelaskan listing program yang ada pada halaman wishlist. Helper::getAllProductFromWishlist() code tersebut berfungsi untuk mengambil semua dari produk wishlist. Tombol hapus menggunakan route wishlist-delete untuk menghapus produk dari wishlist.

| @extends('frontend.layouts.master')  @section('title','Wishlist Page')  @section('main-content')  <div class="breadcrumbs">  <div class="container">  <div class="row">  <div class="col-12">  <div class="bread-inner">  <ul class="bread-list">  <li><a href="{{('home')}}">Home<i class="ti-arrow-right"></i></a></li>  <li class="active"><a href="javascript:void(0);">Wishlist</a></li>  </ul>  </div>  </div>  </div>  </div>  </div>  <div class="shopping-cart section">  <div class="container">  <div class="row">  <div class="col-12">  <table class="table shopping-summery">  <thead>  <tr class="main-hading">  <th>PRODUCT</th>  <th>NAME</th>  <th class="text-center">TOTAL</th>  <th class="text-center">ADD TO CART</th>  <th class="text-center"><i class="ti-trash remove-icon"></i></th>  </tr>  </thead>  <tbody>  @if(Helper::getAllProductFromWishlist())  @foreach(Helper::getAllProductFromWishlist() as $key=>$wishlist)  <tr>  @php  $photo=explode(',',$wishlist->product['photo']);  @endphp  <td class="image" data-title="No"><img src="{{$photo[0]}}" alt="{{$photo[0]}}"></td>  <td class="product-des" data-title="Description">  <p class="product-name"><a href="{{route('product-detail',$wishlist->product['slug'])}}">{{$wishlist->product['title']}}</a></p>  <p class="product-des">{!!($wishlist['summary']) !!}</p>  </td>  <td class="total-amount" data-title="Total"><span>${{$wishlist['amount']}}</span></td>  <td><a href="{{route('add-to-cart',$wishlist->product['slug'])}}" class='btn text-white'>Add To Cart</a></td>  <td class="action" data-title="Remove"><a href="{{route('wishlist-delete',$wishlist->id)}}"><i class="ti-trash remove-icon"></i></a></td>  </tr>  @endforeach  @else  <tr>  <td class="text-center">  There are no any wishlist available. <a href="{{route('product-grids')}}" style="color:blue;">Continue shopping</a>  </td>  </tr>  @endif  </tbody>  </table>  </div>  </div>  </div>  </div>  @include('frontend.layouts.newsletter')  <div class="modal fade" id="exampleModal" tabindex="-1" role="dialog">  <div class="modal-dialog" role="document">  <div class="modal-content">  <div class="modal-header">  <button type="button" class="close" data-dismiss="modal" aria-label="Close"><span class="ti-close" aria-hidden="true"></span></button>  </div>  <div class="modal-body">  <div class="row no-gutters">  <div class="col-lg-6 col-md-12 col-sm-12 col-xs-12">  <div class="product-gallery">  <div class="quickview-slider-active">  <div class="single-slider">  <img src="images/modal1.jpg" alt="#">  </div>  <div class="single-slider">  <img src="images/modal2.jpg" alt="#">  </div>  <div class="single-slider">  <img src="images/modal3.jpg" alt="#">  </div>  <div class="single-slider">  <img src="images/modal4.jpg" alt="#">  </div>  </div>  </div>  </div>  <div class="col-lg-6 col-md-12 col-sm-12 col-xs-12">  <div class="quickview-content">  <h2>Flared Shift Dress</h2>  <div class="quickview-ratting-review">  <div class="quickview-ratting-wrap">  <div class="quickview-ratting">  <i class="yellow fa fa-star"></i>  <i class="yellow fa fa-star"></i>  <i class="yellow fa fa-star"></i>  <i class="yellow fa fa-star"></i>  <i class="fa fa-star"></i>  </div>  <a href="#"> (1 customer review)</a>  </div>  <div class="quickview-stock">  <span><i class="fa fa-check-circle-o"></i> in stock</span>  </div>  </div>  <h3>RP.290.000</h3>  <div class="quickview-peragraph">  </div>  <div class="size">  <div class="row">  <div class="col-lg-6 col-12">  </select>  </div>  </div>  </div>  </div>  <div class="quantity">  <div class="input-group">  <div class="button minus">  <button type="button" class="btn btn-primary btn-number" disabled="disabled" data-type="minus" data-field="quant[1]">  <i class="ti-minus"></i>  </button>  </div>  <input type="text" name="quant[1]" class="input-number" data-min="1" data-max="1000" value="1">  <div class="button plus">  <button type="button" class="btn btn-primary btn-number" data-type="plus" data-field="quant[1]">  <i class="ti-plus"></i>  </button>  </div>  </div>  </div>  <div class="add-to-cart">  <a href="#" class="btn">Add to cart</a>  <a href="#" class="btn min"><i class="ti-heart"></i></a>  <a href="#" class="btn min"><i class="fa fa-compress"></i></a>  </div>  </div>  </div>  </div>  </div>  </div>  </div>  </div>  @endsection  @push('scripts')  <script src="https://cdnjs.cloudflare.com/ajax/libs/sweetalert/2.1.2/sweetalert.min.js"></script>  @endpush |
| --- |

### **3.9.6** **About Us**

Gambar 3.51 ini menunjukan halaman About Us yang terdapat dalam *website* glasgow. Halaman about us ini menjelaskan tentang toko glasgow dan bertujuan untuk memberi info tentang toko glasgow kepada pembeli.



Gambar 3. 51Halaman About Us

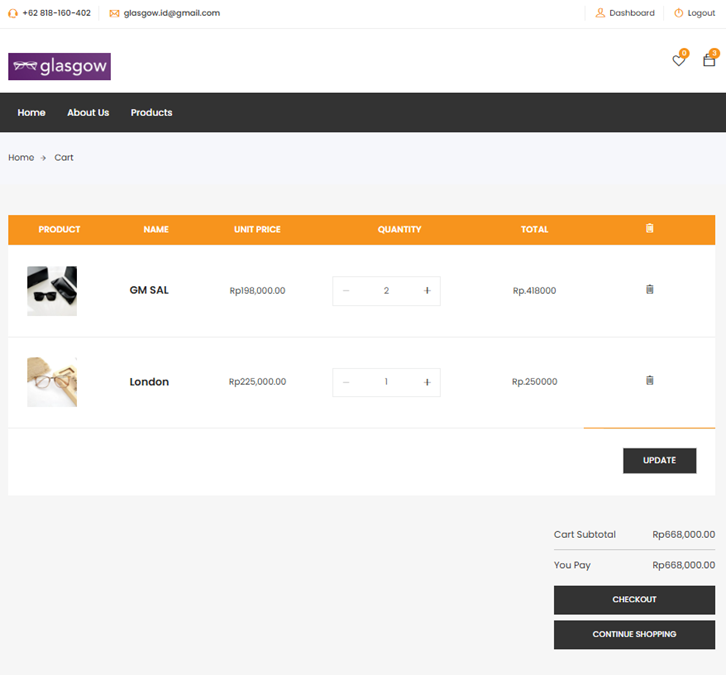
#### **3.9.6.1** **Kode Program Halaman About Us**

Pada saat ini menjelaskan listing program di halaman about us. Mengambil data dari tabel settings dalam database menggunakan DB::table('settings')->get() dan menampilkannya di halaman ini. Menampilkan judul "Welcome To GLASGOW" dan deskripsi yang diambil dari pengaturan ($data->description). Menampilkan gambar yang diambil dari tabel settings ($data->photo).

| @extends('frontend.layouts.master')  @section('title','GLASGOW || About Us')  @section('main-content')  <div class="breadcrumbs">  <div class="container">  <div class="row">  <div class="col-12">  <div class="bread-inner">  <ul class="bread-list">  <li><a href="index1.html">Home<i class="ti-arrow-right"></i></a></li>  <li class="active"><a href="blog-single.html">About Us</a></li>  </ul>  </div>  </div>  </div>  </div>  </div>  <section class="about-us section">  <div class="container">  <div class="row">  <div class="col-lg-6 col-12">  <div class="about-content">  @php  $settings=DB::table('settings')->get();  @endphp  <h3>Welcome To <span>GLASGOW</span></h3>  <p>@foreach($settings as $data) {{$data->description}} @endforeach</p>  <div class="button">  </div>  </div>  </div>  <div class="col-lg-6 col-12">  <div class="about-img overlay">  {{-- <div class="button">  <a href="https://www.youtube.com/watch?v=nh2aYrGMrIE" class="video video-popup mfp-iframe"><i class="fa fa-play"></i></a>  </div> --}}  <img src="@foreach($settings as $data) {{$data->photo}} @endforeach" alt="@foreach($settings as $data) {{$data->photo}} @endforeach">  </div>  </div>  </div>  </div>  </section>  </div>  </div>  </div>  </section>  @endsection |
| --- |

### **3.9.7** **Cart**

Pada gambar 3.52 ini menunjukan halaman cart pada *website* glasgow. Halaman cart ini berfungsi memudahkan *user* untuk melakukan checkout produk yang sudah di pilih dan bisa melakukan checkout secara bersaamaan atau lebih dari satu produk.



Gambar 3. 52Halaman Cart

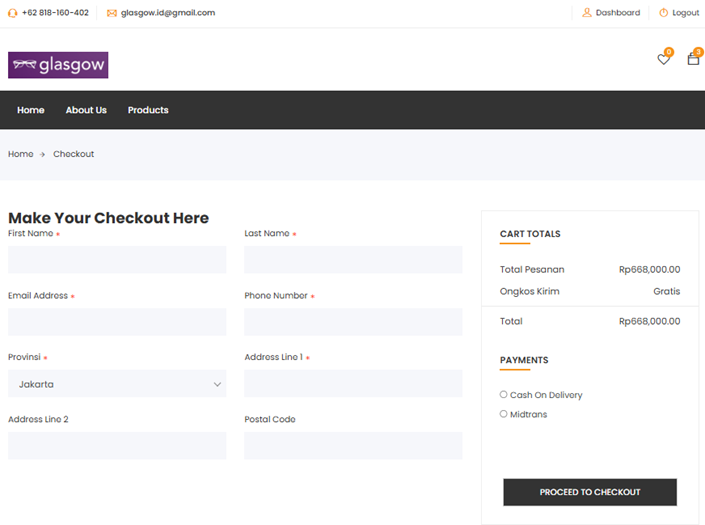
#### **3.9.7.1** **Kode Program Halaman Cart**

Pada pembahasan kali ini akan menjelaskan listing program yang ada pada halaman cart. Tombol ti-trash pada setiap baris memungkinkan pengguna untuk menghapus produk dari keranjang dengan mengarah ke rute cart-delete. Menggunakan Helper::getAllProductFromCart() untuk mendapatkan semua produk yang ada di keranjang. Menggunakan Helper::totalCartPrice() untuk menghitung subtotal dari semua produk dalam keranjang.

| @extends('frontend.layouts.master')  @section('title','Cart Page')  @section('main-content')  <div class="breadcrumbs">  <div class="container">  <div class="row">  <div class="col-12">  <div class="bread-inner">  <ul class="bread-list">  <li><a href="{{('home')}}">Home<i class="ti-arrow-right"></i></a></li>  <li class="active"><a href="">Cart</a></li>  </ul>  </div>  </div>  </div>  </div>  </div>  <div class="shopping-cart section">  <div class="container">  <div class="row">  <div class="col-12">  <table class="table shopping-summery">  <thead>  <tr class="main-hading">  <th>PRODUCT</th>  <th>NAME</th>  <th class="text-center">UNIT PRICE</th>  <th class="text-center">QUANTITY</th>  <th class="text-center">TOTAL</th>  <th class="text-center"><i class="ti-trash remove-icon"></i></th>  </tr>  </thead>  <tbody id="cart\_item\_list">  <form action="{{route('cart.update')}}" method="POST">  @csrf  @if(Helper::getAllProductFromCart())  @foreach(Helper::getAllProductFromCart() as $key=>$cart)  <tr>  @php  $photo=explode(',',$cart->product['photo']);  @endphp  <td class="image" data-title="No"><img src="{{$photo[0]}}" alt="{{$photo[0]}}"></td>  <td class="product-des" data-title="Description">  <p class="product-name"><a href="{{route('product-detail',$cart->product['slug'])}}" target="\_blank">{{$cart->product['title']}}</a></p>  <p class="product-des">{!!($cart['summary']) !!}</p>  </td>  <td class="price" data-title="Price"><span>Rp{{number\_format($cart['price'],2)}}</span></td>  <td class="qty" data-title="Qty">  <div class="input-group">  <div class="button minus">  <button type="button" class="btn btn-primary btn-number" disabled="disabled" data-type="minus" data-field="quant[{{$key}}]">  <i class="ti-minus"></i>  </button>  </div>  <input type="text" name="quant[{{$key}}]" class="input-number" data-min="1" data-max="100" value="{{$cart->quantity}}">  <input type="hidden" name="qty\_id[]" value="{{$cart->id}}">  <div class="button plus">  <button type="button" class="btn btn-primary btn-number" data-type="plus" data-field="quant[{{$key}}]">  <i class="ti-plus"></i>  </button>  </div>  </div>  </td>  <td class="total-amount cart\_single\_price" data-title="Total"><span class="money">Rp.{{$cart['amount']}}</span></td>  <td class="action" data-title="Remove"><a href="{{route('cart-delete',$cart->id)}}"><i class="ti-trash remove-icon"></i></a></td>  </tr>  @endforeach  <td class="float-right">  <button class="btn float-right" type="submit">Update</button>  </td>  </track>  @else  <tr>  <td class="text-center">  There are no any carts available. <a href="{{route('product-grids')}}" style="color:blue;">Continue shopping</a>  </td>  </tr>  @endif  </form>  </tbody>  </table>  <div class="total-amount">  <div class="row">  <div class="col-lg-8 col-md-5 col-12">  <div class="left">  <div class="coupon">  </div>  {{-- <div class="checkbox">`  </div> --}}  </div>  </div>  <div class="col-lg-4 col-md-7 col-12">  <div class="right">  <ul>  <li class="order\_subtotal" data-price="{{Helper::totalCartPrice()}}">Cart Subtotal<span>Rp{{number\_format(Helper::totalCartPrice(),2)}}</span></li>  @if(session()->has('coupon'))  <li class="coupon\_price" data-price="{{Session::get('coupon')['value']}}">You Save<span>Rp{{number\_format(Session::get('coupon')['value'],2)}}</span></li>  @endif  @php  $total\_amount=Helper::totalCartPrice();  if(session()->has('coupon')){  $total\_amount=$total\_amount-Session::get('coupon')['value'];  }  @endphp  @if(session()->has('coupon'))  <li class="last" id="order\_total\_price">You Pay<span>Rp{{number\_format($total\_amount,2)}}</span></li>  @else  <li class="last" id="order\_total\_price">You Pay<span>Rp{{number\_format($total\_amount,2)}}</span></li>  @endif  </ul>  <div class="button5">  <a href="{{route('checkout')}}" class="btn">Checkout</a>  <a href="{{route('product-grids')}}" class="btn">Continue shopping</a>  </div>  </div>  </div>  </div>  </div> |
| --- |

### **3.9.8** **Checkout**

Gambar 3.53 menunjukan halaman checkout yang ada pada *website*. Pada halaman checkout ini *user* wajib mengisi data diri dan juga alamat, halaman checkout ini juga menampilkan jumlah harga total produk yang sudah masuk sebelumnya kedalam keranjang.



Gambar 3. 53Halaman Checkout

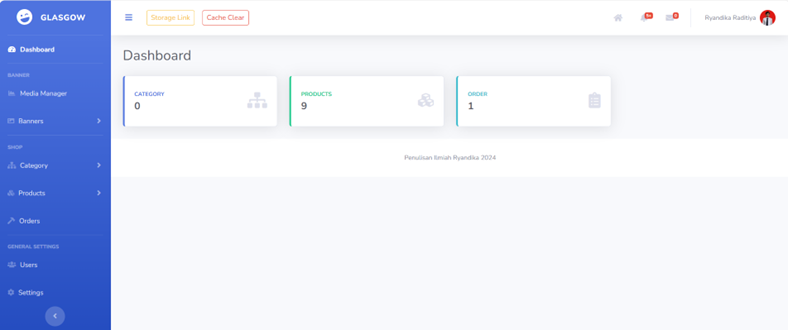
#### **3.9.8.1** **Kode Program Halaman Checkout**

Pada bagian ini akan menjelaskan listing program yang berada pada halaman checout. Pada bagian ini menampilkan total biaya pesanan yang diambil dari helper Helper::totalCartPrice(). Form ini dikirim ke rute cart.order menggunakan metode POST.

| @extends('frontend.layouts.master')  @section('title', 'Checkout page')  @section('main-content')  <div class="breadcrumbs">  <div class="container">  <div class="row">  <div class="col-12">  <div class="bread-inner">  <ul class="bread-list">  <li><a href="{{ route('home') }}">Home<i class="ti-arrow-right"></i></a></li>  <li class="active"><a href="javascript:void(0)">Checkout</a></li>  </ul>  </div>  </div>  </div>  </div>  </div>  <section class="shop checkout section">  <div class="container">  <form id="payment-form" class="form" method="POST" action="{{ route('cart.order') }}">  @csrf  <div class="row">  <div class="col-lg-8 col-12">  <div class="checkout-form">  <h2>Make Your Checkout Here</h2>  <div class="row">  <div class="col-lg-6 col-md-6 col-12">  <div class="form-group">  <label>First Name<span>\*</span></label>  <input type="text" name="first\_name" placeholder=""  value="{{ old('first\_name') }}" value="{{ old('first\_name') }}">  @error('first\_name')  <span class='text-danger'>{{ $message }}</span>  @enderror  </div>  </div>  <div class="col-lg-6 col-md-6 col-12">  <div class="form-group">  <label>Last Name<span>\*</span></label>  <input type="text" name="last\_name" placeholder="" value="{{ old('lat\_name') }}">  @error('last\_name')  <span class='text-danger'>{{ $message }}</span>  @enderror  </div>  </div>  <div class="col-lg-6 col-md-6 col-12">  <div class="form-group">  <label>Email Address<span>\*</span></label>  <input type="email" name="email" placeholder="" value="{{ old('email') }}">  @error('email')  <span class='text-danger'>{{ $message }}</span>  @enderror  </div>  </div>  <div class="col-lg-6 col-md-6 col-12">  <div class="form-group">  <label>Phone Number <span>\*</span></label>  <input type="number" name="phone" placeholder="" required  value="{{ old('phone') }}">  @error('phone')  <span class='text-danger'>{{ $message }}</span>  @enderror  </div>  </div>  <div class="col-lg-6 col-md-6 col-12">  <div class="form-group">  <label>Provinsi<span>\*</span></label>  <select name="country" id="country">  <option value="ID"selected="selected">Jakarta</option>  <option value="AC">Aceh</option>  <option value="SU">Sumatera Utara</option>  <option value="SB">Sumatera Barat</option>  <option value="RI">Riau</option>  <option value="KR">Kepulauan Riau</option>  <option value="JA">Jambi</option>  <option value="SS">Sumatera Selatan</option>  <option value="BB">Bangka Belitung</option>  <option value="BE">Bengkulu</option>  <option value="LA">Lampung</option>  <option value="JK">Jawa Barat</option>  <option value="BT">Banten</option>  <option value="JT">Jawa Tengah</option>  <option value="JI">Jawa Timur</option>  <option value="YO">Yogyakarta</option>  <option value="BA">Bali</option>  <option value="NB">Nusa Tenggara Barat</option>  <option value="NT">Nusa Tenggara Timur</option>  <option value="KB">Kalimantan Barat</option>  <option value="KT">Kalimantan Tengah</option>  <option value="KI">Kalimantan Timur</option>  <option value="KS">Kalimantan Selatan</option>  <option value="KU">Kalimantan Utara</option>  <option value="SA">Sulawesi Utara</option>  <option value="ST">Sulawesi Tengah</option>  <option value="SG">Sulawesi Tenggara</option>  <option value="SR">Sulawesi Barat</option>  <option value="SN">Sulawesi Selatan</option>  <option value="GO">Gorontalo</option>  <option value="MA">Maluku</option>  <option value="MU">Maluku Utara</option>  <option value="PA">Papua</option>  <option value="PB">Papua Barat</option>  </select>  </div>  </div>  <div class="col-lg-6 col-md-6 col-12">  <div class="form-group">  <label>Address Line 1<span>\*</span></label>  <input type="text" name="address1" placeholder=""  value="{{ old('address1') }}">  @error('address1')  <span class='text-danger'>{{ $message }}</span>  @enderror  </div>  </div>  <div class="col-lg-6 col-md-6 col-12">  <div class="form-group">  <label>Address Line 2</label>  <input type="text" name="address2" placeholder=""  value="{{ old('address2') }}">  @error('address2')  <span class='text-danger'>{{ $message }}</span>  @enderror  </div>  </div>  <div class="col-lg-6 col-md-6 col-12">  <div class="form-group">  <label>Postal Code</label>  <input type="text" name="post\_code" placeholder=""  value="{{ old('post\_code') }}">  @error('post\_code')  <span class='text-danger'>{{ $message }}</span>  @enderror  </div>  </div>  </div>  </div>  </div>  <div class="col-lg-4 col-12">  <div class="order-details">  <div class="single-widget">  <h2>CART TOTALS</h2>  <div class="content">  <ul>  <li class="order\_subtotal" data-price="{{ Helper::totalCartPrice() }}">Total  Pesanan<span>Rp{{ number\_format(Helper::totalCartPrice(), 2) }}</span></li>  <li class="shipping">  Ongkos Kirim <span>Gratis</span>  @if (0)  @else  @endif  </li>  @if (session('coupon'))  <li class="coupon\_price" data-price="{{ session('coupon')['value'] }}">You  Save<span>Rp{{ number\_format(session('coupon')['value'], 2) }}</span></li>  @endif  @php  $total\_amount = Helper::totalCartPrice();  if (session('coupon')) {  $total\_amount = $total\_amount - session('coupon')['value'];  }  @endphp  @if (session('coupon'))  <li class="last" id="order\_total\_price">  Total<span>Rp{{ number\_format($total\_amount, 2) }}</span></li>  @else  <li class="last" id="order\_total\_price">  Total<span>Rp{{ number\_format($total\_amount, 2) }}</span></li>  @endif  </ul>  </div>  </div>  <div class="single-widget">  <h2>Payments</h2>  <div class="content">  <div class="checkbox">  {{-- <label class="checkbox-inline" for="1"><input name="updates" id="1" type="checkbox"> Check Payments</label> --}}  <form-group>  <input name="payment\_method" type="radio" value="cod"> <label> Cash On  Delivery</label><br>  <input name="payment\_method" type="radio" value="midtrans"> <label>  Midtrans</label>  </form-group>  </div>  </div>  </div>  <div class="single-widget payement">  <div class="content">  <img src=>  </div>  </div>  <div class="single-widget get-button">  <div class="content">  <div class="button">  <button id="pay-button" type="submit" class="btn">proceed to  checkout</button>  </div>  </div>  </div> |
| --- |

### **3.9.9** **Dashboard Admin**

Pada gambar 3.54 menunjukan halaman dashboard admin pada *website* glasgow. *Admin* dapat mengakses dashboard ketika *admin* sudah melakukan login terlebih dahulu. Pada halaman dashboard adal pilihan *banners*, *category*, *products*, *orders*, *users*, dan juga settings.



Gambar 3. 54Halaman Dashboard Admin

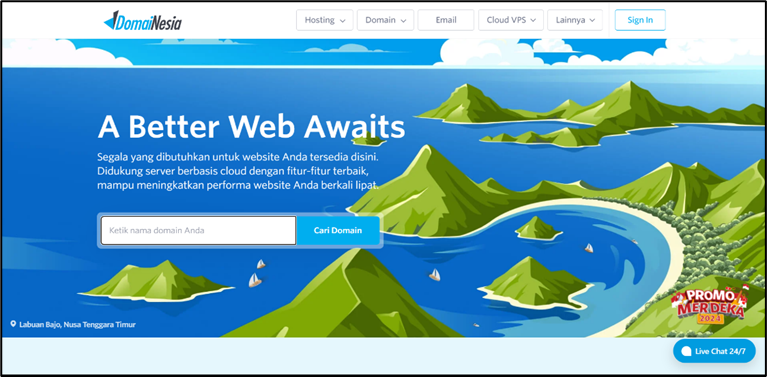
#### **3.9.9.1** **Kode Program Halaman Dashboard Admin**

Pada bagian ini akan menjelaskan lisitng program yang terdapat pada halaman dashboard. Code @include('backend.layouts.notification'): Menyertakan file notification yang mungkin digunakan untuk menampilkan pesan notifikasi di halaman dashboard. div class="container-fluid" untuk pembungkus utama untuk semua konten dalam dashboard.

| @extends('backend.layouts.master')  @section('title','GLASGOW || DASHBOARD')  @section('main-content')  <div class="container-fluid">  @include('backend.layouts.notification')  <div class="d-sm-flex align-items-center justify-content-between mb-4">  <h1 class="h3 mb-0 text-gray-800">Dashboard</h1>  </div>  <div class="row">  <div class="col-xl-3 col-md-6 mb-4">  <div class="card border-left-primary shadow h-100 py-2">  <div class="card-body">  <div class="row no-gutters align-items-center">  <div class="col mr-2">  <div class="text-xs font-weight-bold text-primary text-uppercase mb-1">Category</div>  <div class="h5 mb-0 font-weight-bold text-gray-800">{{\App\Models\Category::countActiveCategory()}}</div>  </div>  <div class="col-auto">  <i class="fas fa-sitemap fa-2x text-gray-300"></i>  </div>  </div>  </div>  </div>  </div>  <div class="col-xl-3 col-md-6 mb-4">  <div class="card border-left-success shadow h-100 py-2">  <div class="card-body">  <div class="row no-gutters align-items-center">  <div class="col mr-2">  <div class="text-xs font-weight-bold text-success text-uppercase mb-1">Products</div>  <div class="h5 mb-0 font-weight-bold text-gray-800">{{\App\Models\Product::countActiveProduct()}}</div>  </div>  <div class="col-auto">  <i class="fas fa-cubes fa-2x text-gray-300"></i>  </div>  </div>  </div>  </div>  </div>  <div class="col-xl-3 col-md-6 mb-4">  <div class="card border-left-info shadow h-100 py-2">  <div class="card-body">  <div class="row no-gutters align-items-center">  <div class="col mr-2">  <div class="text-xs font-weight-bold text-info text-uppercase mb-1">Order</div>  <div class="row no-gutters align-items-center">  <div class="col-auto">  <div class="h5 mb-0 mr-3 font-weight-bold text-gray-800">{{\App\Models\Order::countActiveOrder()}}</div>  </div>  </div>  </div>  <div class="col-auto">  <i class="fas fa-clipboard-list fa-2x text-gray-300"></i>  </div>  </div>  </div>  </div>  </div>  </div> |
| --- |

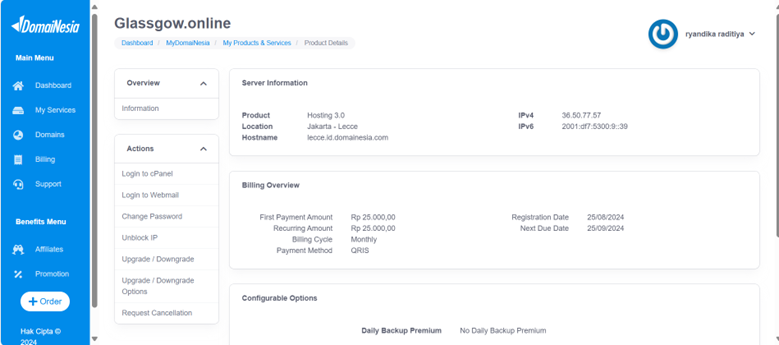
## **3.10** **Upload Hosting**

Pada tahapan ini, membahas proses upload hosting. Hosting adalah layanan yang menyimpan *file* dan data *web* secara *online,* sehingga *website* bisa diakses oleh banyak orang melalui internet. *File* dan data ini bisa berupa gambar, vidio, *script*, *email*, aplikasi, dan juga database. Untuk *website* glasgow ini hosting yang digunakan adalah DomaiNesia dengan alamat url<https://www.domainesia.com> seperti gambar berikut.



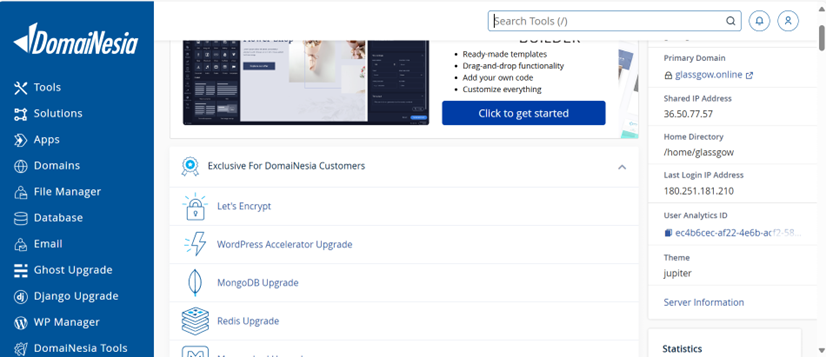
Gambar 3. 55Tampilan Halaman Utama Hosting

Setelah melakukan pendaftaran atau login, penulis melakukan pembuatan situs *web* baru. Setelah melakukan isi data, akan di arahkan ke tampilan dashboard.



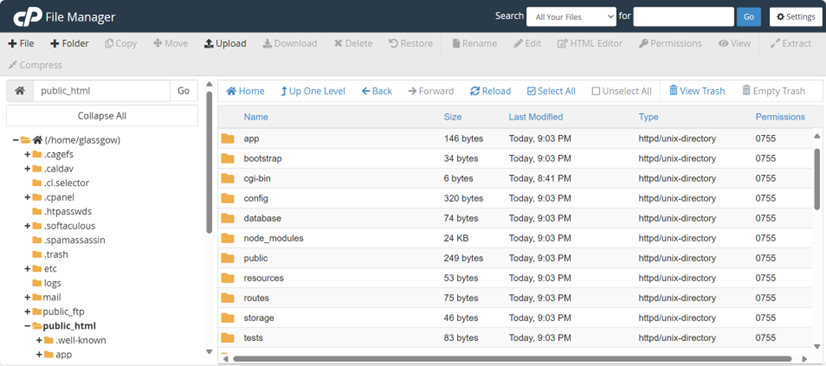
Gambar 3. 56Tampilan Dashboard Product Domainesia

Setelah itu klik login cpanel lalu akan di arahkan ke dashboard admin untuk mengakses *website*.



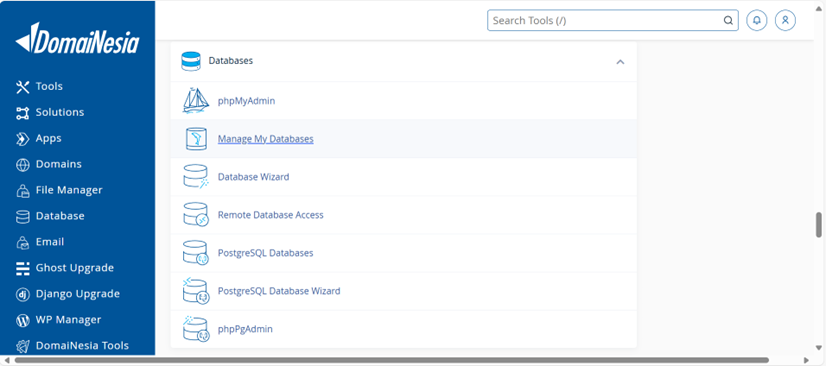
Gambar 3. 57Tampilan Dashboard Admin Domainesia

Setelah itu klik file manager lalu masukan file website yang telah di buat kedalam public\_html.



Gambar 3. 58Tampilan File Manager Domainesia

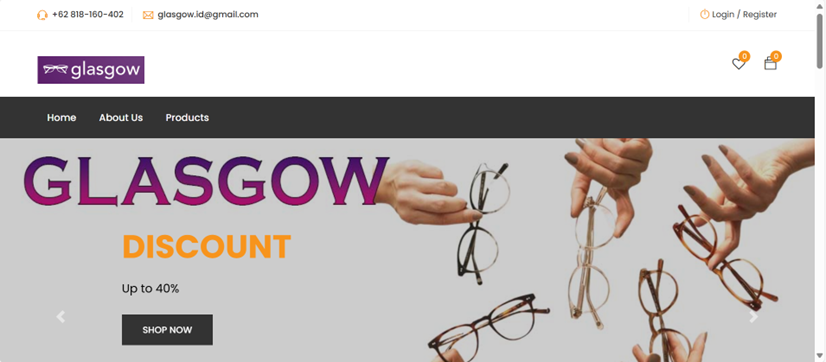
Lalu klik php myadmin dan import database yang sudah dibuat.



Gambar 3. 59Tampilan Dashboard Admin Database

Setelah proses hosting telah selesai dilakukan, *website* Glasgow dapat diakses dan digunakan secara umum dengan alamat url

<https://glassgow.online/> seperti pada gambar di bawah ini.



Gambar 3. 60Tampilan Landing Page Glasgow

## **3.11** **Uji Coba**

Pada tahap uji coba ini dilakukan untuk memastikan apakah website sudah berfungsi dengan baik dengan apaa yang di harapakan. Pada tahapan ini penulis melakukan beberapa metode, dengan bertujuan memastikan bahwa *website* glasgow ini berjalan dengan baik dan sesuai dengan yang di harapkan.

### **3.11.1** **Uji Coba Black Box**

Pada pengujian black box ini, menu atau fitur dalam *website* glasgow akan diuji. Skenario, hasil yang diharapkan, hasil pengujian, dan juga kesimpulan akan menjadi penentu apakah *website* glasgow dapat dijalankan dan berfungsi dengan baik. Hasil dari black box testing akan ditunjukan pada tabel dibawah ini.

Tabel 3. 9Hasil Uji Coba Black Box

| **No** | **Test Case** | **Hasil Yang Diharapkan** | **Hasil** | **Kesimpulan** |
| --- | --- | --- | --- | --- |
| 1 | Home | Menampilkan halaman home page pada saat pertama kali membuka *website*. |  | Berhasil |
| 2 | Login | Berhasil melakukan login lalu masuk ke halaman home pada *website*. |  | Berhasil |
| 3 | Register | Berhasil melakukan register atau membuat akun sehingga dapat melakukan login. |  | Berhasil |
| 4 | About Us | Berhasil menampilkan halaman about us saat *user* klik fitur about us. |  | Berhasil |
| 5 | Product | Berhasil menampilkan halaman produk pada saat klik halaman produk. |  | Berhasil |
| 6 | Wishlist | Pada saat memilih produk dan memasukannya ke dalam wishlist, lalu produk itu sudah ada pada halaman wishlist. |  | Berhasil |
| 7 | Cart | Saat proses pemilihan produk dan sudah dimasukan ke cart atau klik cart pada produk, lalu produk itu akan masuk ke halaman cart. |  | Berhasil |
| 8 | Continue Shoping | Saat *user* menekan atau klik fitur continue shoping pada halaman cart lalu akan di arahkan kembali ke halaman home page *website*. |  | Berhasil |
| 9 | Checkout | Menampilkan hakaman checkout setelah klik checkout pada halaman cart. |  | Berhasil |
| 10 | Melakukan Pembayaran | Pada saat sudah mengisi data diri pada halaman checkout lalu melakukan pembayaran akan di munculkan beberapa pilihan pembayaran. |  | Berhasil |
| 11 | Dashboard | Menampilkan halaman dashboard saat sudah melakukan login dan klik fitur dashboard. |  | Berhasil |
| 12 | Logout | Saat melakukan logout atau klik logout akan keluar dari akun yang sudah login sebelumnya. |  | Berhasil |

### **3.11.2** **Uji Coba Browser**

Pada tahapan uji coba browser ini, dilakukan uji coba browser untuk memastikan bahwa *website* glasgow dapat berfungsi dengan baik di berbagai browser yang digunakan. Uji coba pada 3 browser berjalan dengan baik. *Website* dapat berjalan tanpa adanya eror dan fitu semua berfungsi seperti yang diharapkan. Hasil uji coba akan ditunjukan pada tabel 3.10 seperti dibawah ini.

Tabel 3. 10Uji Coba Browser

| No | Jenis Browser | Kesimpulan |
| --- | --- | --- |
| 1 | Edge | *Website* berjalan sesuai dengan yang diharapkan |
| 2 | Google Chrome | *Website* berjalan sesuai dengan yang diharapkan |
| 3 | Opera | *Website* berjalan sesuai dengan yang diharapkan |

### **3.11.3** **Uji Coba User Acceptance Testing**

Pada uji coba User Acceptance Testing sangat penting karena untuk memastikan bahwa *website* glasgow memenuhi kebutuhan dan harapan pengguna, untuk mengukur tingkat kepuasan pengguna saat menggunakan *website* glasgow. Tabel 3.11 ini merupakan hasil dari uji coba User Acceptance Testing.

Tabel 3. 11Uji Coba User Acceptance Testing

| **NO** | **PERTANYAAN** | **SKALA LIKERT** | | | | |
| --- | --- | --- | --- | --- | --- | --- |
| STS | TS | RR | S | SS |
| 1 | Apakah tampilan desain website glasgow menarik? | 0 | 0 | 2 | 4 | 16 |
| 2 | Apakah anda merasa mudah saat mencari informasi yang ada di website? | 0 | 0 | 0 | 11 | 11 |
| 3 | Apakah anda dapat mengakses dan memahami website secara mudah? | 0 | 0 | 0 | 6 | 16 |
| 4 | Apakah website sudah responsif saat anda mengakses website kami? | 0 | 0 | 1 | 8 | 13 |
| 5 | Apakah ada eror atau bug dalam website kami? | 7 | 6 | 4 | 0 | 5 |
| 6 | Apakah anda merasa terbantu dalam proses pememesan kacamata menggunakan website kami? | 0 | 0 | 0 | 11 | 11 |
| 7 | Apakah anda akan merekomendasikan website glasgow kepada orang lain? | 0 | 0 | 3 | 6 | 13 |
| 8 | Apakah anda merasa puas dengan pilihan metode pembayaran yang ada pada website kami? | 0 | 0 | 0 | 8 | 14 |
| 9 | Apakah anda merasa puas dengan keseluruhan website glasgow ini setelah menggunakan website kami? | 0 | 0 | 0 | 6 | 16 |
| 10 | Apakah metode pembayaran yang disediakan berfungsi dengan baik? | 0 | 1 | 1 | 7 | 13 |
| 11 | Apakah data pelanggan dan transaksi terlindungi dengan baik? | 0 | 0 | 1 | 9 | 12 |
| 12 | Apakah ada kesuilitan atau masalah yang sering dilaporkan oleh pengguna? | 5 | 4 | 4 | 0 | 9 |
| 13 | Apakah pelanggan dapat memilih variasi produk dengan benar? | 0 | 0 | 0 | 8 | 14 |
| 14 | Apakah sistem menangani jumlah pesanan dengan akurat? | 0 | 0 | 0 | 8 | 14 |
| 15 | Apakah peggawai dapat melihat dan mengelola pesanan dengan mudah? | 0 | 0 | 0 | 8 | 14 |

Setelah mendapatkan data kepuasan pengguna, selanjutnya penulis mengelola data menggunakan metode skala likert. Setiap jawaban pada skala likert mempunyai bobot tersendiri. Berikut adalah bobot nilai yang penulis gunakan untuk setiap jawaban dalam skala likert.

Tabel 3. 12Nilai Bobot Skala Likert

[ tabel kecilin dulu]

Setelah menentukan bobot dari setiap jawaban skala likert, langkah selanjutnya menghitung skor likert. Dibawah ini adalah perhitungan dari skor likert.

· Total skor penilaian jawaban Sangat Tidak Setuju

Total = TxPn

= 12 x 1

= 12

· Total skor penilaian jawaban Tidak Setuju

Total = TxPn

= 11 x 2

= 22

· Total skor penilaian jawaban Netral

Total = TxPn

= 16 x 3

= 48

· Total skor penilaian jawaban Setuju

Total =TxPn

= 100 x 4

= 400

· Total skor penilaian jawaba Sangat Setuju

Total = TxPn

= 191 x 5

= 955

Total skor likert = 12 + 22 + 48 + 400 + 955

= 1437

Y = skor tertingi likert x jumlah responden x banyak pertanyaan

= 5 x 22 x 15

= 1650

Indeks% = (1437 / 1650) x 100

= 87 %

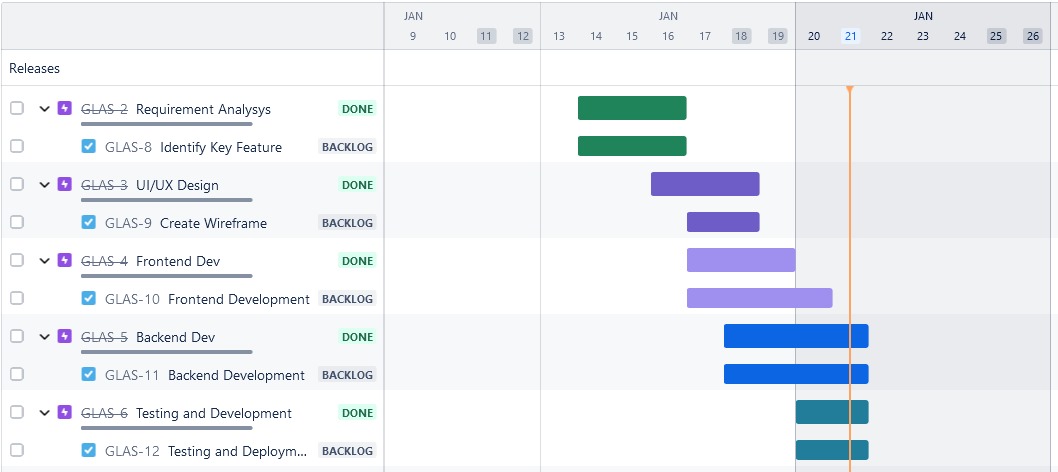
Bedasarkan hasil dari uji coba metode likert ini *website* glasgow mendapat respon atau hasil yang baik dari 22 responden yaitu sebesar 87 % (Setuju).

# **BAB IV**

**Waktu Pengerjaan**

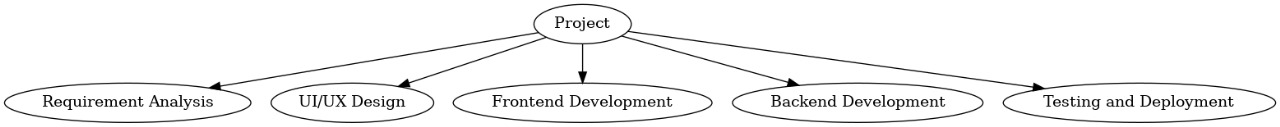
## **4.1. Jadwal Pengerjaan Proyek**

Pengerjaan proyek dilakukan selama kurang lebih dua minggu sesuai dengan deadline yang sudah ditentukan. Berikut merupakan *timeline* dari pengerjaan proyek pada gambar berikut.

****

## **4.2. Work Breakdown Structure (WBS)**

Berikut Work Breakdown Structure dalam pelaksanaan project yang digunakan untuk melakukan pemecahan atau pembagian tiap proses pekerjan menjadi lebih detail.



Root (Project):

* Ini adalah tingkat teratas dari WBS, yang mewakili proyek secara keseluruhan. Semua tugas dan sub-tugas yang berada di bawahnya adalah bagian dari penyelesaian proyek ini.

Level 1 (Task Breakdown):

* Requirement Analysis: Mengidentifikasi dan mendokumentasikan kebutuhan pengguna atau sistem sebagai langkah awal proyek.
* UI/UX Design: Mendesain antarmuka pengguna (UI) dan pengalaman pengguna (UX) agar proyek lebih user-friendly dan sesuai dengan kebutuhan pengguna.
* Frontend Development: Membangun bagian antarmuka aplikasi yang terlihat dan dapat diakses oleh pengguna.
* Backend Development: Mengembangkan logika inti, server, basis data, dan integrasi sistem yang mendukung fungsionalitas aplikasi.
* Testing and Deployment: Menguji sistem untuk memastikan kualitasnya dan meluncurkannya ke lingkungan produksi.

# **BAB V**

## **5.1** **Kesimpulan**

Pembuatan sistem penjualan pada toko kacamata glasgow telah berhasil diimplementasikan. Aplikasi website dapat digunakan untuk membantu konsumen dalam mendapatkan informasi mengenai produk yang tersedia pada toko glasgow serta mempermudah membeli kacamata pada toko glasgow ini tanpa harus datang ke toko. Aplikasi website yang dibangun dapat diakses melalui alamat url<https://glassgow.online/>. Berdasarkan hasil pengujian dengan metode blackbox, uji coba browser, dan UAT (*User Acceptance Testing*) aplikasi website dapat berfungsi dengan baik dan siap digunakan. Hal ini juga didukung dengan hasil jumlah skor UAT(*User Acceptance Testing*) 87%.

## **5.2** **Saran**

Pembuatan *website* glasgow masih sederhana, terutama dalam segi tampilan, sehingga diperlukan adanya perbaikan dari segi tampilan serta juga pengembangan fitur seperti melihat status pembayaran melalui *website* tanpa harus menggunakan midtrans.