


<b>Project – External Documentation</b>	
COMP6114 Pattern Software Design	
Even Semester Year 2022/2023	

- **Project Title**

**KpopZtation**

- **Introduction**

**KpopZtation** adalah sebuah toko album K-pop di Indonesia. Untuk memperluas bisnisnya, KpopZtation ingin membuat sebuah website untuk toko mereka. Di website tersebut, pengunjung dapat menemukan informasi tentang album K-pop dan memesannya. Website ini akan memungkinkan pelanggan untuk mengetahui apa saja yang ada di KpopZtation. Dengan menggunakan teknologi ASP.NET dan menerapkan metode Domain Driven Design, dapat mengembangkan website secara efisien dan interaktif, sehingga user dapat dengan mudah mengetahui produk-produk yang ditawarkan oleh KpopZtation dan melakukan pemesanan secara online. Selain itu, akan dipertimbangkan opsi penggunaan layanan web untuk meningkatkan fungsionalitas website ini. Dengan adanya website ini, KpopZtation berharap dapat memperluas jangkauan bisnis dan memberikan pengalaman yang lebih baik kepada pelanggan.

ASP.NET adalah kerangka kerja pengembangan aplikasi web sisi server yang dirancang oleh Microsoft. Ini menyediakan model pemrograman, infrastruktur perangkat lunak, dan berbagai layanan yang memungkinkan pengembang untuk membangun aplikasi web yang dinamis dan kuat. ASP.NET menggunakan bahasa pemrograman seperti C# atau Visual Basic.NET dan berjalan di atas platform .NET Framework. Dengan ASP.NET, pengembang dapat membuat halaman web interaktif, aplikasi web berbasis formulir, layanan web, dan berbagai jenis aplikasi web lainnya. ASP.NET memisahkan logika bisnis dari presentasi tampilan dengan menggunakan model pemisahan kode (code-behind). Hal ini memungkinkan tim pengembangan untuk bekerja secara terpisah, dengan pengembang front-end bertanggung jawab untuk merancang tampilan dan pengembang back-end fokus pada logika bisnis dan pemrosesan data.

Domain-Driven Design (DDD) adalah pendekatan dalam pengembangan perangkat lunak yang fokus pada pemahaman yang mendalam tentang domain bisnis atau domain masalah yang sedang dihadapi. DDD bertujuan untuk menciptakan model yang kuat dan ekspresif yang mencerminkan pengetahuan yang ada di dalam domain tersebut. Pendekatan DDD menekankan kolaborasi antara pengembang perangkat lunak dan pakar domain. Para pengembang bekerja sama dengan pakar domain untuk memahami dengan baik bisnis atau domain masalah yang relevan. Mereka menggunakan bahasa yang umum dan konsisten, yang disebut sebagai bahasa yang umum (ubiquitous language), untuk berkomunikasi dan menggambarkan konsep-konsep dalam domain tersebut.

Ketika mengembangkan aplikasi web menggunakan ASP.NET, DDD dapat digunakan sebagai metodologi untuk merancang struktur dan logika aplikasi berdasarkan pemahaman yang mendalam tentang domain bisnis yang relevan. Penerapan DDD dalam aplikasi ASP.NET dapat melibatkan:

1. Pemodelan domain: DDD membantu dalam merancang model domain yang tepat untuk aplikasi. Ini melibatkan pemahaman yang mendalam tentang entitas, agregat, nilai-nilai, dan kebijakan bisnis dalam domain tersebut.
2. Bahasa yang Umum (Ubiquitous Language): DDD mendorong penggunaan bahasa yang umum dan konsisten antara pengembang perangkat lunak dan pakar domain. Ini membantu dalam komunikasi yang lebih baik dan pemahaman yang sama terhadap konsep-konsep dalam domain.
3. Pemisahan Tanggung Jawab (Separation of Concerns): ASP.NET menerapkan prinsip pemisahan tanggung jawab, yang juga merupakan salah satu prinsip DDD. Ini memungkinkan pengembang untuk memisahkan kode berdasarkan tanggung jawabnya, termasuk pemisahan antara logika bisnis (domain) dan tampilan (UI).
4. Repositori dan Akses Data: DDD menggunakan konsep repositori untuk mengakses data dalam domain. Ini dapat diimplementasikan menggunakan fitur-fitur ASP.NET seperti Entity Framework atau teknologi akses data lainnya.
5. Arsitektur Berbasis Layanan (Service-Oriented Architecture): DDD dan ASP.NET dapat digunakan bersama-sama dalam implementasi arsitektur berbasis layanan, di mana layanan-layanan yang independen bertanggung jawab atas bagian-bagian tertentu dalam domain.

Dalam keseluruhan, ASP.NET dan DDD dapat bekerja sama untuk menciptakan aplikasi web yang lebih terstruktur, mudah dimengerti, dan lebih relevan terhadap kebutuhan bisnis. DDD membantu dalam merancang dan menerapkan logika bisnis berdasarkan pemahaman yang mendalam tentang domain, sementara ASP.NET menyediakan infrastruktur dan alat yang kuat untuk mengembangkan aplikasi web yang dinamis.

Dalam Project yang dikerjakan kali ini, Beberapa Requirements di terapkan untuk mengembangkan KpopZtation ini. Seperti pada Domain Layer yang dibutuhkan berupa:

- **View**

Layer Tampilan, atau Layer Presentasi, bertugas untuk menampilkan informasi kepada pengguna dan menginterpretasikan perintah pengguna. Ini merupakan tempat di mana antarmuka pengguna dalam proyek ditempatkan.

- **Kontroler**

Layer Kontroler bertanggung jawab untuk memvalidasi semua input dari layer tampilan dan mendelegasikan permintaan dari pengguna ke lapisan yang lebih rendah untuk diproses.

- **Handler**

Layer Handler menangani seluruh logika bisnis dalam aplikasi. Ia akan mendelegasikan tugas pengambilan data dari database ke layer repository, termasuk operasi select, insert, update, dan delete.

- **Repository**

Layer Repository memberikan akses ke database dan lapisan model melalui antarmuka publiknya untuk mendapatkan referensi objek domain yang sudah ada. Ia menyediakan metode untuk memanipulasi objek, seperti menambahkan dan menghapus, serta metode untuk memilih objek berdasarkan kriteria tertentu.

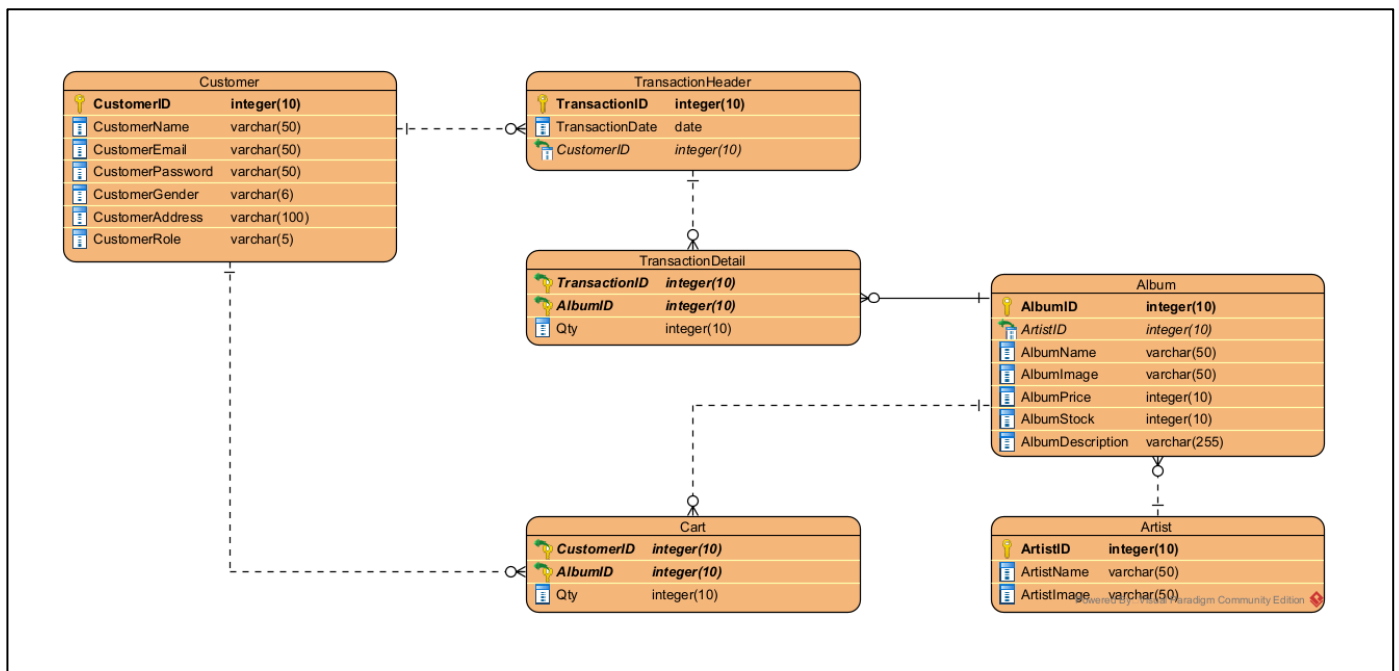
- **Factory**

Layer Pabrik (Factory) mengenkapsulasi pembuatan objek yang kompleks. Misalnya, ketika klien perlu membuat objek agregat, pabrik objek akan menyediakan antarmuka untuk membuat objek-objek tersebut. Penting untuk dicatat bahwa objek yang dibuat oleh pabrik harus dalam keadaan yang konsisten.

- **Model**

Layer model memiliki tanggung jawab dalam menggambarkan konsep-konsep dalam bisnis atau informasi tentang keadaan bisnis.

Adapun Database yang dibutuhkan untuk diterapkan dalam proses pengembangan KpopZtation ini sebagai tempat penampung berbagai data yang dibutuhkan dari website ini. Berikut adalah gambar **Entity Relationship Diagram** untuk **KpopZtation Database**:



**Figure 1.** Entity Relationship Diagram of KpopZtation's Database

Selanjutnya adalah beberapa beberapa tipe pengguna yang dapat masuk ke website KpopZtation ini berupa **Admin**, **Customer**, dan **Guest** (non-logged-in user). Terdapat beberapa persyaratan yang harus dipenuhi dalam pembuatan setiap tipe pengguna ini:

- **Admin**
  - View Artists
  - Insert Artist
  - Update Artist
  - Delete Artis
  - View Albums
  - Insert Album
  - Update Album
  - Delete Album
  - Update Profile
  - Delete Account
  - View Transactions Report
- **Customer**
  - View Artists
  - View Albums

- Order Album
- View Profile
- View Carts
- Update Profile
- Delete Account
- View Transactions History
- View Transaction Detail
- **Guest**
  - Login
  - Register
  - View Artists
  - View Albums

Selanjutnya terdapat deskripsi mengenai detail dari setiap page yang harus dipenuhi, mencakup:

#### 1. **Login**

Halaman ini adalah halaman awal (landing page) dari situs web di mana pengguna akan mencoba untuk masuk ke situs web. Halaman ini hanya dapat diakses oleh pengunjung (guest).

Tabel berikut menampilkan kolom-kolom yang ada dalam formulir login dan validasi untuk setiap kolom:

Field	Information
<b>Email</b>	Must be filled and appropriate with the data in the database
<b>Password</b>	Must be filled and appropriate with the data in the database

Jika pengguna memasukkan nama pengguna atau kata sandi yang tidak benar, maka akan muncul pesan kesalahan yang sesuai dengan validasi yang memicu kesalahan tersebut. Halaman login juga akan memiliki opsi "Remember me". Opsi "Remember me" akan menyimpan informasi login dengan menggunakan cookie yang memiliki waktu kedaluwarsa jika pengguna telah mencentangnya, sehingga pada percobaan login berikutnya, pengguna akan secara otomatis masuk ke dalam sistem.

#### 2. **Register**

Halaman ini memberikan akses kepada pengunjung untuk mendaftar sebagai pelanggan KpopZtation. Validasi dilakukan untuk memastikan bahwa halaman ini hanya dapat diakses oleh pengunjung. Jika pengguna memasukkan data pribadi yang tidak valid, maka akan ditampilkan pesan kesalahan. Tabel berikut menampilkan kolom-kolom yang ada pada halaman ini beserta validasi untuk masing-masing kolom:

Field	Information
Name	Must be filled and between 5 and 50 characters
Email	Must be filled and unique among the customer's email
Gender	Must be selected
Address	Must be filled and ends with 'Street'
Password	Must be filled and alphanumeric

### 3. Navigation Bar

Menyediakan sebuah Navigation Bar untuk memudahkan pengguna dalam berpindah halaman:

- Jika tidak ada user yang Login (Sebagai Guest):
  - Home: Mengarahkan pengguna ke halaman beranda
  - Login: Mengarahkan pengguna ke halaman login
  - Register: Mengarahkan pengguna ke halaman pendaftaran
- Jika User saat ini adalah seorang Customer:
  - Home: Mengarahkan pengguna ke halaman beranda
  - Cart: Mengarahkan pengguna ke halaman keranjang
  - Transaction: Mengarahkan pengguna ke halaman transaksi
  - Update Profile: Mengarahkan pengguna ke halaman perbarui profil
  - Logout: Keluar pengguna dari situs web dan menghapus semua variabel cookie
- Jika User saat ini adalah seorang admin:
  - Home: Mengarahkan pengguna ke halaman beranda
  - Transaction: Mengarahkan pengguna ke halaman laporan transaksi
  - Update Profile: Mengarahkan pengguna ke halaman perbarui profil
  - Logout: Keluar pengguna dari situs web dan menghapus semua variabel cookie

### 4. Home

Halaman ini memungkinkan pengguna untuk melihat semua artis KpopZtation. Halaman ini juga memungkinkan pengguna untuk diarahkan ke Halaman Detail Artis jika pengguna mengklik kartu artis tersebut. Halaman ini juga memungkinkan admin untuk melakukan penambahan, pembaruan, dan penghapusan artis. Di halaman ini, terdapat tombol "Tambah Artis". Jika admin mengklik tombol "Tambah Artis", maka akan diarahkan ke halaman "Tambah Artis".

Setiap album dalam daftar memiliki tombol "Perbarui" dan "Hapus":

- Jika admin mengklik tombol "Hapus", maka artis yang dipilih akan dihapus.
- Jika admin mengklik tombol "Perbarui", maka akan diarahkan ke halaman "Perbarui Artis" dan ID Artis yang dipilih akan dikirim melalui parameter query string URL.

### 5. Artist Detail

Jika pengguna mengklik kartu artis di halaman utama, mereka akan diarahkan ke halaman detail ini. Halaman ini menampilkan gambar, nama, dan detail album yang dimiliki oleh artis, seperti gambar album, nama, harga, dan deskripsi. Pengguna juga dapat mengklik kartu album untuk diarahkan ke Halaman Detail Album. Admin memiliki kemampuan untuk menambahkan, memperbarui, dan menghapus album. Jika admin mengklik tombol "Tambah Album", mereka akan diarahkan ke halaman "Tambah Album". Setiap album dalam daftar memiliki tombol "Perbarui" dan "Hapus":

- Jika admin mengklik tombol "Hapus", album yang dipilih akan dihapus.

- Jika admin mengklik tombol "Perbarui", mereka akan diarahkan ke halaman "Perbarui Album" dengan ID Artis yang dipilih dikirim melalui parameter query string URL.

#### 6. Insert Artist

Halaman ini hanya dapat diakses oleh admin. Admin dapat menggunakan halaman ini untuk menambahkan artis baru ke dalam database. Halaman ini dapat diakses saat admin mengklik tombol "Tambah Album" di Halaman Utama. Jika admin memasukkan data yang tidak valid, pesan kesalahan akan ditampilkan. Setelah admin memasukkan semua informasi artis dengan benar dan menekan tombol "Tambah", data artis yang dimasukkan akan ditambahkan dan disimpan ke dalam database. Tabel berikut menampilkan bidang-bidang yang ada pada halaman ini beserta validasinya:

Field	Information
<b>Name</b>	Must be filled and unique among the artist's name.
<b>Image</b>	Must be chosen, file extension must be .png, .jpg, .jpeg, or .jfif, and file size must be lower than 2MB.

#### 7. Update Artist

Halaman ini hanya dapat diakses oleh admin. Admin dapat menggunakan halaman ini untuk memperbarui data artis yang sudah ada di dalam database. Administrator akan diarahkan ke halaman ini saat mereka mengklik tombol "Perbarui" di Halaman Utama. Semua data terkini dari artis yang akan diperbarui akan ditampilkan di halaman ini, dan administrator dapat mengubah data tersebut sebelum mengirimkannya kembali dengan menekan tombol "Perbarui". Data artis akan diambil dari database menggunakan ID artis yang diberikan melalui parameter query string URL. Setelah admin memasukkan semua bidang dengan benar dan menekan tombol "Perbarui", data artis di dalam database akan diperbarui. Jika admin memasukkan data yang tidak valid, pesan kesalahan akan ditampilkan. Tabel berikut menampilkan bidang-bidang yang ada di halaman ini beserta validasinya:

Field	Information
<b>Name</b>	Must be filled and unique among the artist's name.
<b>Image</b>	Must be chosen, file extension must be .png, .jpg, .jpeg, or .jfif, and file size must be lower than 2MB.

#### 8. Insert Album

Halaman ini hanya dapat diakses oleh admin. Admin dapat menggunakan halaman ini untuk menambahkan album baru untuk artis terpilih pada halaman detail artis ke dalam database. Halaman ini dapat diakses saat admin mengklik tombol "Tambah Album" di Halaman Detail Artis. Data album akan diambil dari database menggunakan ID album yang diberikan melalui parameter query string URL. Jika admin memasukkan data yang tidak valid, pesan kesalahan akan ditampilkan. Setelah admin memasukkan semua bidang dengan benar dan menekan tombol "Tambah", data artis yang dimasukkan akan ditambahkan dan disimpan ke dalam database. Tabel berikut menampilkan bidang-bidang yang ada di halaman ini beserta validasinya:

Field	Information
<b>Name</b>	Must be filled and smaller than 50 characters.
<b>Description</b>	Must be filled and smaller than 255 characters.

<b>Price</b>	Must be filled and between 100000 and 1000000
<b>Stock</b>	Must be filled and more than 0
<b>Image</b>	Must be chosen, file extension must be .png, .jpg, .jpeg, or .jfif, and file size must be lower than 2MB.

#### 9. Update Album

Halaman ini hanya dapat diakses oleh admin. Admin dapat menggunakan halaman ini untuk mengupdate album yang sudah ada dalam database. Administrator akan diarahkan ke halaman ini ketika mereka mengklik tombol "Update" yang disediakan di Halaman Detail Artis. Semua data album yang akan diupdate akan ditampilkan di halaman ini dan administrator dapat mengubah data tersebut sebelum mengirimkannya kembali dengan menekan tombol "Update". Data album akan diambil dari database menggunakan ID album yang diberikan melalui parameter query string URL. Setelah admin memasukkan semua bidang dengan benar dan menekan tombol "Update", informasi artis di database akan diperbarui. Jika admin memasukkan data yang tidak valid, pesan kesalahan akan ditampilkan.

Field	Information
<b>Name</b>	Must be filled and smaller than 50 characters.
<b>Description</b>	Must be filled and smaller than 255 characters.
<b>Price</b>	Must be filled and between 100000 and 1000000
<b>Stock</b>	Must be filled and more than 0
<b>Image</b>	Must be chosen, file extension must be .png, .jpg, .jpeg, or .jfif, and file size must be lower than 2MB.

#### 10. Album Detail

Halaman ini dapat diakses oleh semua pengguna. Jika pengguna mengklik tombol "Details" di halaman utama, mereka akan diarahkan ke halaman detail ini. Di halaman ini, informasi lengkap tentang album ditampilkan, termasuk nama, deskripsi, harga, dan stok album. Terdapat juga tombol "Add to Cart" yang hanya akan muncul jika pengguna telah login sebagai pelanggan. Pelanggan dapat memasukkan jumlah barang sebelum menambahkannya ke keranjang belanja. Berikut adalah tabel yang menampilkan bidang-bidang yang ada di halaman ini beserta validasi untuk setiap bidang:

Field	Information
<b>Quantity</b>	Must be filled and can't be more than the stock album

#### 11. Cart Page

Halaman ini hanya dapat diakses oleh pelanggan. Saat pelanggan mengklik tombol "Cart" pada Navigation Bar, mereka akan diarahkan ke halaman ini. Halaman ini menampilkan semua album yang telah ditambahkan ke keranjang belanja pelanggan. Informasi keranjang seperti gambar album, nama album, jumlah album, dan harga album ditampilkan. Terdapat dua tombol di halaman ini, yaitu tombol "Remove" untuk menghapus album dari keranjang dan tombol "Checkout" untuk menyelesaikan pembelian. Jika pelanggan mengklik tombol "Remove", album yang dipilih akan dihapus dari keranjang, dan jika mereka mengklik tombol "Checkout", data keranjang belanja akan

dihapus dan data transaksi akan ditambahkan ke riwayat transaksi pelanggan, lalu pengguna akan kembali ke halaman utama.

#### 12. **Transaction History**

Halaman ini khusus untuk customer dan digunakan untuk menampilkan riwayat transaksi pelanggan. Informasi yang ditampilkan meliputi ID transaksi, tanggal transaksi, nama pelanggan, kurir, gambar album, nama album, jumlah album, dan harga album.

#### 13. **Transaction Report**

Halaman ini spesifik untuk admin dan digunakan untuk menampilkan laporan data transaksi menggunakan Crystal Report. Laporan tersebut akan mencakup informasi header transaksi seperti ID transaksi, ID pelanggan, tanggal transaksi, dan total keseluruhan, serta detail transaksi seperti ID transaksi, nama album, jumlah, harga album, dan subtotal.

#### 14. **Update Profile**

Halaman ini memungkinkan pengguna untuk mengganti informasi profil mereka. Pengguna akan diarahkan ke halaman ini ketika mereka mengklik tombol Perbarui Profil yang tersedia di Navigation Bar. Jika pengguna memasukkan data yang tidak valid, akan ditampilkan pesan kesalahan. Semua data profil yang akan diperbarui akan ditampilkan di halaman ini dan pengguna dapat mengubahnya sebelum mengirimkan perubahan dengan menekan tombol Perbarui. Setelah pengguna memasukkan semua data dengan benar dan menekan tombol Perbarui, informasi pengguna di database akan diperbarui. Tabel berikut menampilkan bidang-bidang yang ada di halaman ini beserta validasinya:

Field	Information
Name	Must be filled and between 5 and 50 characters
Email	Must be filled and unique among the customer's email
Gender	Must be selected
Address	Must be filled and ends with 'Street'
Password	Must be filled and alphanumeric

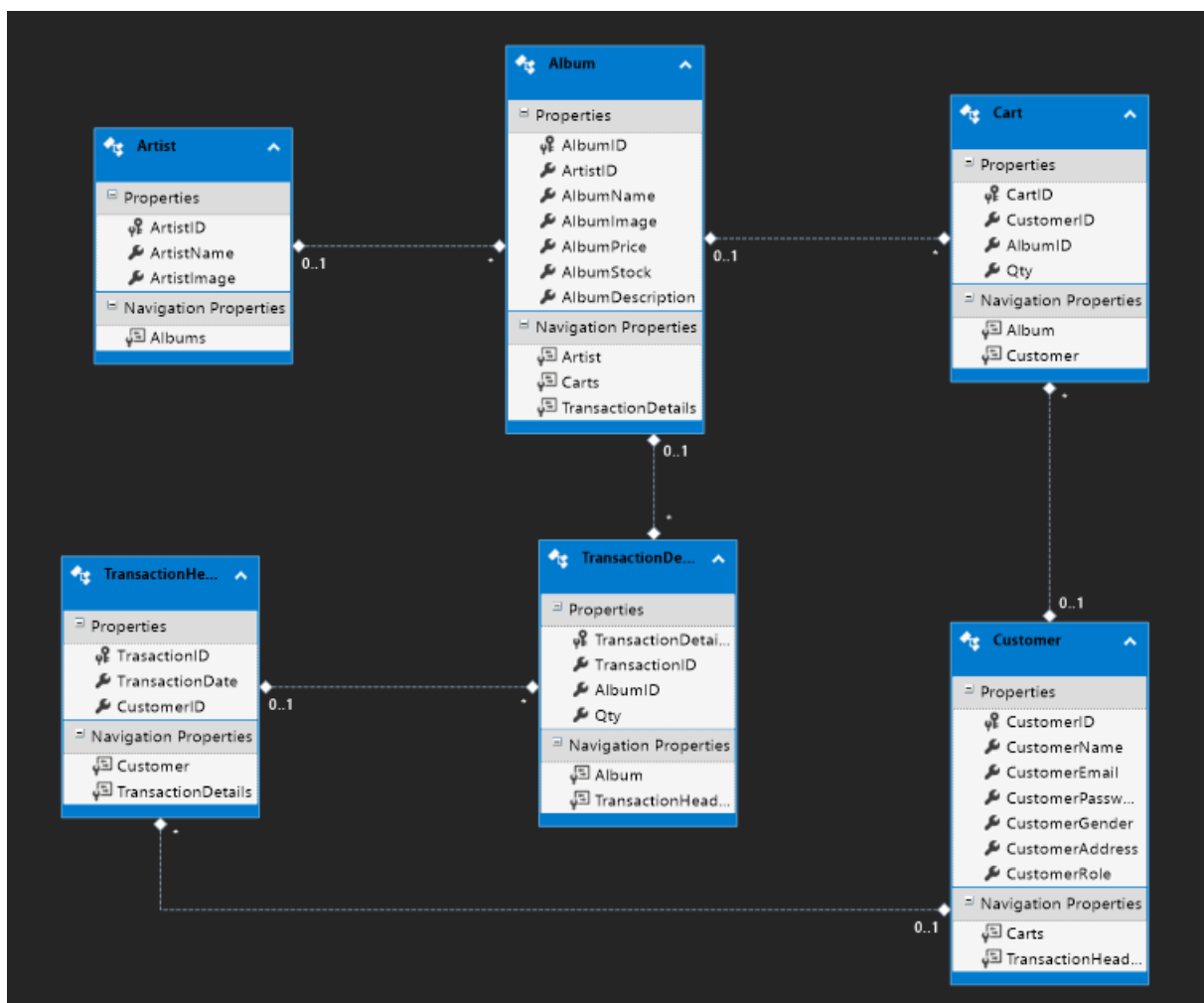


## Report / Documentation

Berikut adalah implementasi pengerjaan KpopZtation berdasarkan requirements yang sudah disebutkan. Gambar ini tidak mencakup semua code yang sangat panjang karena hanya sebagai dokumentasi.

1. **Model Entities** dari database yang telah dibuat, dimana memuat Table berupa:

- Artist
- Album
- Cart
- Transaction
- TransactionHeader
- Cutomer



2. Berikut adalah beberapa potongan Gambar dari Clas **Repository** yang telah di buat:

**DatabaseSingleton.cs**

```

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Web;
5  using KpopZtation1.Controller;
6  using KpopZtation1.Repository;
7  using KpopZtation1.Factory;
8  using KpopZtation1.Handler;
9
10 namespace KpopZtation1.Repository
11 {
12     1 reference
13     public class DatabaseSingleton
14     {
15         private static Database1Entities2 db = null;
16
17         1 reference
18         public static Database1Entities2 getInstance()
19         {
20             return db = (db == null) ? db = new Database1Entities2() : db;
21         }
22     }
23 }

```

## Repository.cs

```

9
10 namespace KpopZtation1.Repository
11 {
12     78 references
13     public class Repository
14     {
15         private static Database1Entities2 db = DatabaseSingleton.getInstance();
16
17         12 references
18         public static Customer getCustomerByID(String id)
19         {
20             Customer customer = (from x in db.Customers where x.CustomerID.ToString().Equals(id) select x).FirstOrDefault();
21             return customer;
22         }
23
24         3 references
25         public static Artist getArtistByID(String id)
26         {
27             Artist artist = (from x in db.Artists where x.ArtistID.ToString().Equals(id) select x).FirstOrDefault();
28             return artist;
29         }
30
31         1 reference
32         public static int getLastCustomerID()
33         {
34             int ID = (from x in db.Customers select x.CustomerID).ToList().LastOrDefault();
35             return ID;
36         }
37
38         1 reference
39         public static Customer validateLogin(String email, String password)
40         {
41             Customer customer = (from x in db.Customers where x.CustomerEmail.Equals(email) && x.CustomerPassword.Equals(password)
42             return customer;
43         }
44
45         10 references
46         public static String getCustomerRole(String id)
47         {
48             String customer = (from x in db.Customers where x.CustomerID.ToString().Equals(id) select x.CustomerRole).FirstOrDefault();
49             return customer;
50         }
51
52         1 reference
53         public static void addCustomer(int id, String name, String email, String password, String gender, String address, String ro
54         {
55             Customer customer = Factory.factory.createCustomer(id, name, email, password, gender, address, role);
56             db.Customers.Add(customer);
57             db.SaveChanges();
58         }
59     }
60 }

```

Kelas Repository pada kode tersebut berfungsi sebagai perantara antara Controller dan Database. Kelas ini menyediakan metode-metode untuk mengakses dan memanipulasi data dalam Database1Entities2.

3. Berikut adalah beberapa potongan Gambar dari Class **Handler** yang telah dibuat berdasarkan requirement yang dibutuhkan:

#### handler.cs

```
10 namespace KpopZtation1.Handler
11 {
12     6 references
13     public class handler
14     {
15         1 reference
16         protected static int generateCustomerID()
17         {
18             int lastID = Repository.Repository.getLastCustomerID();
19             lastID++;
20             return lastID;
21         }
22         1 reference
23         protected static int generateArtistID()
24         {
25             int lastID;
26             if (Repository.Repository.GetLastArtist() != null)
27             {
28                 lastID = Repository.Repository.GetLastArtist().ArtistID;
29                 lastID++;
30                 return lastID;
31             }
32             return 1;
33         }
34         1 reference
35         protected static int generateAlbumID()
36         {
37             int lastID;
38             if (Repository.Repository.GetLastAlbum() != null)
39             {
40                 lastID = Repository.Repository.GetLastAlbum().AlbumID;
41                 lastID++;
42                 return lastID;
43             }
44             return 1;
45         }
46         1 reference
47         protected static int generateCartID()
48         {
49             int lastID;
50             if (Repository.Repository.getLastCart() != null)
51             {
52                 lastID = Repository.Repository.getLastCart().CartID;
53                 lastID++;
54                 return lastID;
55             }
56             return 1;
57         }
58     }
59 }
```

Class **handler** ini berfungsi sebagai handler (pengendali) untuk berbagai operasi yang terkait dengan manipulasi data dalam aplikasi. Class **handler** ini membantu dalam menghasilkan ID yang unik untuk setiap entitas (pelanggan, artis, album, keranjang, transaksi, detail transaksi) yang akan ditambahkan ke repository. Selain itu, class ini juga menyediakan metode untuk menyisipkan data baru ke dalam repository dengan menggunakan ID yang dihasilkan.

4. Berikut adalah beberapa Potongan Gambar dari Class **Controller** yang sudah dibuat:

### Controller.cs

```
10 namespace KpopZtation1.Controler
11 {
12     7 references
13     public class Controller
14     {
15         2 references
16         public static String checkEmail(String email)
17         {
18             if (email.Equals("")) return "Email must be filled";
19             //validasi email unique
20             else return "";
21         }
22
23         2 references
24         public static String checkPassword(String password)
25         {
26             if (password.Equals("")) return "Password must be filled";
27             // validasi alphanumeric
28             else return "";
29         }
30
31         6 references
32         public static String checkName(String name)
33         {
34             if (name.Equals("")) return "Name must be filled";
35             else if (name.Length < 5 || name.Length > 50) return "name must be between 5 - 50 character";
36             else return "";
37         }
38
39         2 references
40         public static String checkDescription(String desc)
41         {
42             if (desc.Equals("")) return "Description must be filled";
43             else if (desc.Length >= 255) return "Description must be under 255 characters";
44             else return "";
45         }
46
47         2 references
48         public static String checkPrice(String price)
49         {
50             if (price.Equals("")) return "price must be filled";
51             else if (Convert.ToInt32(price) < 100000 || Convert.ToInt32(price) > 1000000) return "price must be between 100.000 to 1.000.000";
52             else return "";
53         }
54
55         2 references
56         public static String checkStock(String stock)
57         {
58             if (stock.Equals("")) return "stock must be filled";
59             else if (Convert.ToInt32(stock) <= 0) return "Stock must be more than 0";
60             else return "";
61         }
62     }
63 }
```

Kelas **Controller** bertanggung jawab untuk memvalidasi input dari pengguna dan melakukan operasi terkait database melalui **Handler** dan **Repository**. Metode-metode dalam kelas ini memberikan validasi untuk memastikan bahwa data yang dimasukkan atau diperbarui sesuai dengan persyaratan yang ditentukan sebelum data tersebut disimpan dalam database.

5. Berikut adalah beberapa Potongan Gambar dari Class **Factory** yang sudah dibuat:

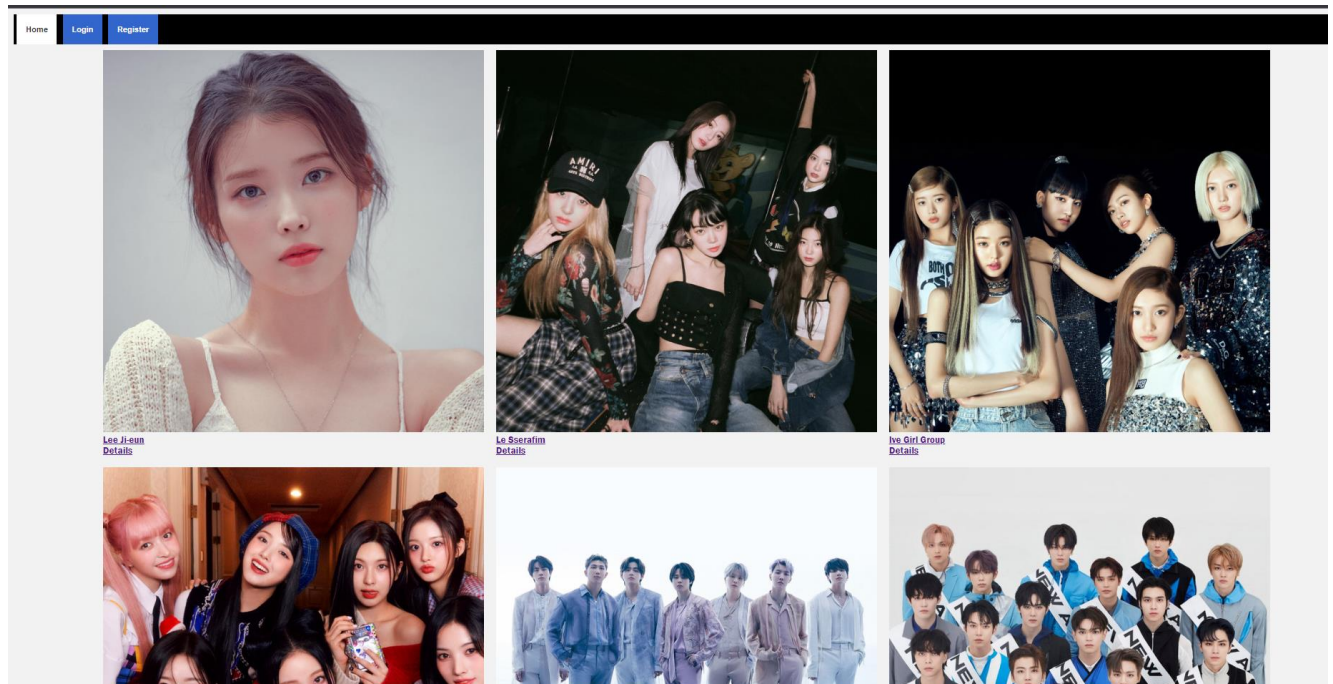
```
10 namespace KpopZtation1.Factory
11 {
12     6 references
13     public class factory
14     {
15         1 reference
16         public static Customer createCustomer(int id, String name, String email, String password, String gender, String address, String role)
17         {
18             Customer customer = new Customer();
19             customer.CustomerID = id;
20             customer.CustomerName = name;
21             customer.CustomerEmail = email;
22             customer.CustomerPassword = password;
23             customer.CustomerGender = gender;
24             customer.CustomerAddress = address;
25             customer.CustomerRole = role;
26             return customer;
27         }
28
29         1 reference
30         public static Artist createArtist(int id, String name, String url, int number)
31         {
32             Artist artist = new Artist();
33             artist.ArtistID = id;
34             artist.ArtistName = name;
35             artist.ArtistImage = url;
36             return artist;
37         }
38
39         public static Album createAlbum(int id, int ArtistID, String name, String desc, int price, int Stock, String url)
40         {
41             Album album = new Album();
42             album.AlbumID = id;
43             album.ArtistID = ArtistID;
44             album.AlbumName = name;
45             album.AlbumDescription = desc;
46             album.AlbumPrice = price;
47             album.AlbumStock = Stock;
48             album.AlbumImage = url;
49             return album;
50         }
51
52         1 reference
53         public static Cart createCart(int id, int userID, int albumID, int qty)
54         {
55             Cart cart = new Cart();
56             cart.CartID = id;
57             cart.CustomerID = userID;
58             cart.AlbumID = albumID;
59             cart.Qty = qty;
60             return cart;
61         }
62
63         1 reference
64         public static TransactionHeader createTransaction(int transactionID, int userID)
65         {
66             TransactionHeader transaction = new TransactionHeader();
```

Class **factory** ini berfungsi sebagai factory (pabrik) untuk membuat objek-objek dalam aplikasi. Class **factory** ini bertanggung jawab untuk menciptakan objek-objek yang dibutuhkan dalam aplikasi, seperti pelanggan, artis, album, keranjang, dan transaksi. Dengan menggunakan metode-metode dalam class **factory**, objek-objek ini dapat dibuat dengan mengisi properti-properti yang relevan.

## Hasil/Result:

Setelah Pembuatan beberapa class yang dibutuhkan sudah terpenuhi, pembuatan desain website dapat dilakukan dan dikembangkan. Berikut adalah hasil dari pengembangan KpopZtation ini:

### 1. Home Page



Di bagian **home page**, terdapat Navigation Bar yang memiliki beberapa tombol seperti 'Home', 'Login', dan 'Register'. Karena user masih belum Login atau statusnya masih sebagai guest, tampilan Navbar terbatas. Namun user masih dapat mengakses Artist Detail dengan mengklik salah satu artist yang tersedia.

< Back



Artist ID : 1  
Artist Name : Lee Ji-eun

Albums :



[Lilac](#)



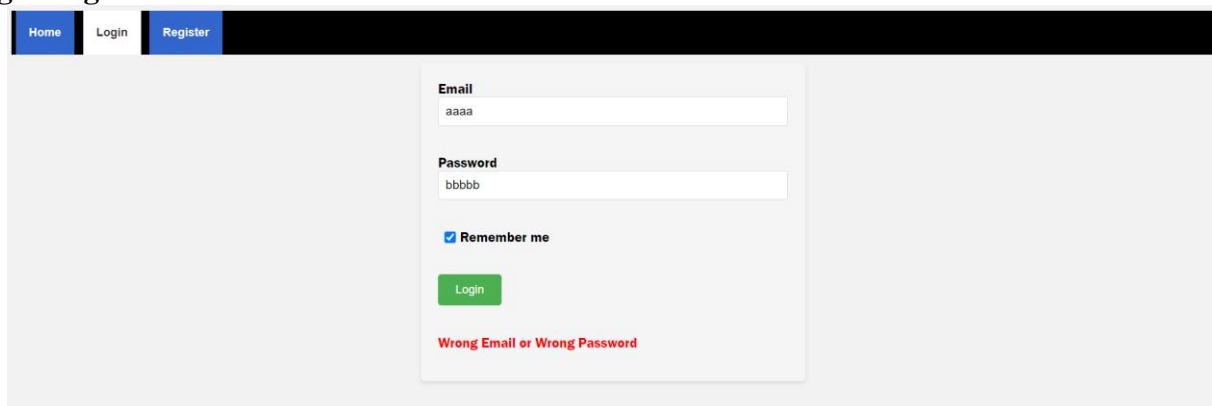
[Chat-sire](#)



[Love Poem](#)

Ini merupakan tampilan apabila Guest melihat Artist Detail page. Guest hanya dapat melihat Album-album yang dimiliki oleh artist tersebut.

## 2. Login Page

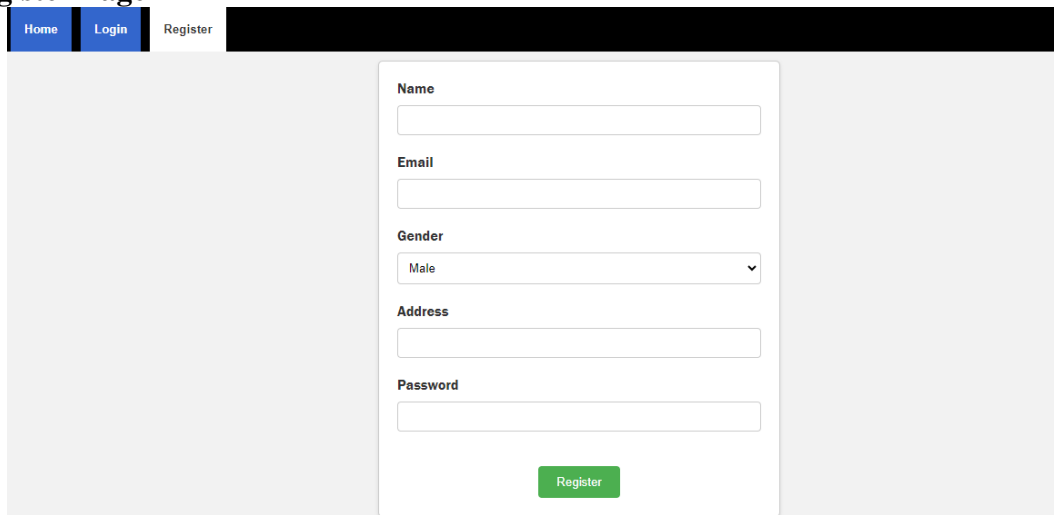


The screenshot shows the Login Page of a web application. At the top, there is a navigation bar with three links: "Home", "Login", and "Register". The "Login" link is highlighted. Below the navigation bar, there is a login form with the following fields and elements:

- Email:** A text input field containing the text "aaaa".
- Password:** A text input field containing the text "bbbb".
- Remember me:** A checkbox that is checked, with the label "Remember me".
- Login:** A green button labeled "Login".
- Error Message:** A red text message "Wrong Email or Wrong Password" displayed below the login button.

Di login page ini, user dapat memasukkan Email dan Password milik user. Apabila email atau password yang dimasukkan salah, akan tampil error label berwarna merah. Terdapat sebuah cookie berupa check box “Remember me” untuk memudahkan user agar tidak perlu login ulang Ketika mengunjungi website ini.

## 3. Register Page

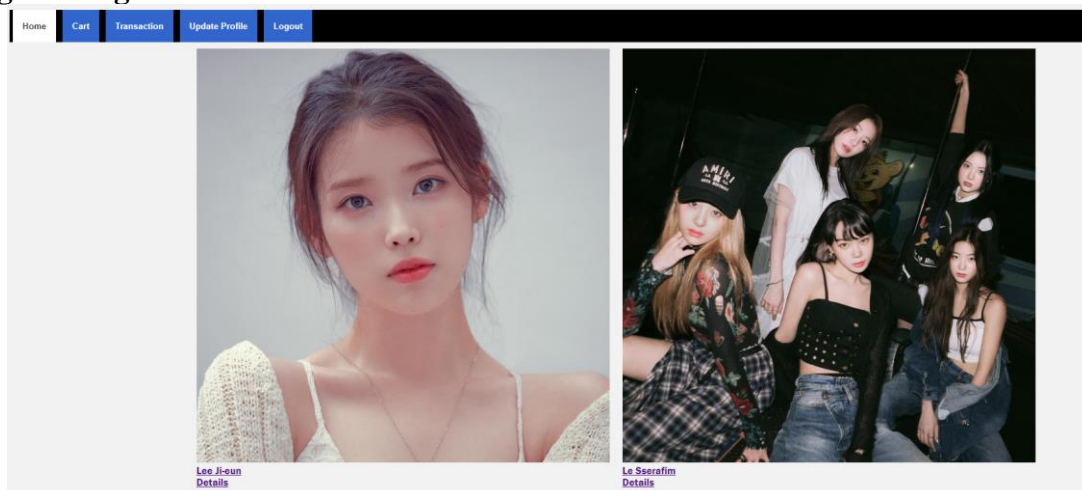


The screenshot shows the Register Page of a web application. At the top, there is a navigation bar with three links: "Home", "Login", and "Register". The "Register" link is highlighted. Below the navigation bar, there is a registration form with the following fields and elements:

- Name:** A text input field.
- Email:** A text input field.
- Gender:** A dropdown menu with "Male" selected.
- Address:** A text input field.
- Password:** A text input field.
- Register:** A green button labeled "Register".

Di Register Page ini, user dapat mendaftar akun untuk user. Beberapa data perlu di input oleh user agar akun dapat berhasil di buat/didaftarkan.

## 4. Login Sebagai Customer





< Back



Artist ID : 1  
Artist Name : Lee Ji-eun

Albums :



[Lilac](#)



[Chat-sire](#)



[Love Poem](#)

< Back

Album Name :

Freeze

Album Description :

Freeze is an album from Tomorrow x Together commonly known as TXT, is a South Korean boy band formed by Big Hit Entertainment, now known as Big Hit Music.

Album Price :

600000

Album Stock :

6

Insert Quantity :

1

Add

< Back



Album Name :

Chat-sire

Album Description :

"Chat-Sire" adalah album yang memukau dari penyanyi dan penulis lagu Korea Selatan yang sangat dihormati, IU. Dirilis pada tahun 2021, album ini menampilkan konsep yang segar dan inovatif, menggabungkan musik elektronik

Album Price :

250000

Album Stock :

2

Add To Cart

< Back



Album Name : Empathy  
Album Price : 400000  
Quantity : 1  
Total Price : 400000  
[Remove](#)



Album Name : The Most Beautiful Moment in Life  
Album Price : 1000000  
Quantity : 1  
Total Price : 1000000  
[Remove](#)

CheckOut

< Back

## Transaction History

Transaction ID : 7  
Transaction Date : Thursday, 15 June 2023  
Your Name : nicko



Album Name : Lilac  
Quantity : 1  
Album Price : 100000

Transaction ID : 8  
Transaction Date : Friday, 16 June 2023  
Your Name : nicko



Album Name : Chat-sire  
Quantity : 1  
Album Price : 250000

Transaction ID : 9  
Transaction Date : Friday, 16 June 2023  
Your Name : nicko



Album Name : Fearless  
Quantity : 2  
Album Price : 400000



Album Name : Love Poem  
Quantity : 1  
Album Price : 150000



Saat user sudah login sebagai Customer, terdapat beberapa perbedaan di bagian Navbar. Dimana terdapat tombol 'Cart', 'Transaction', dan 'Update Profile'. Selain itu, setelah user Login sebagai Customer, user bisa melakukan pembelian album di bagian album Detail, dengan cara menginput berapa item yang di inginkan, dan nantinya akan muncul tombol Add to cart. Setelah berhasil Add to cart, user bisa pindah ke Page **Cart** untuk melihat item-item yang sudah di masukkan ke cart. Lalu user dapat melakukan Checkout. Setelah Checkout, user dapat memeriksa **Transaction Page** untuk melihat riwayat transaksi yang sudah dilakukan.

Lalu, user juga dapat melakukan update profile dengan cara pergi ke **Update Profile Page**. User dapat mengubah beberapa informasi Akun user dengan memasukkan beberapa data yang dibutuhkan untuk mengubah/update profile. User juga dapat melakukan **Logout** dari akun dengan mengklik tombol Navbar Logout.

---

Name

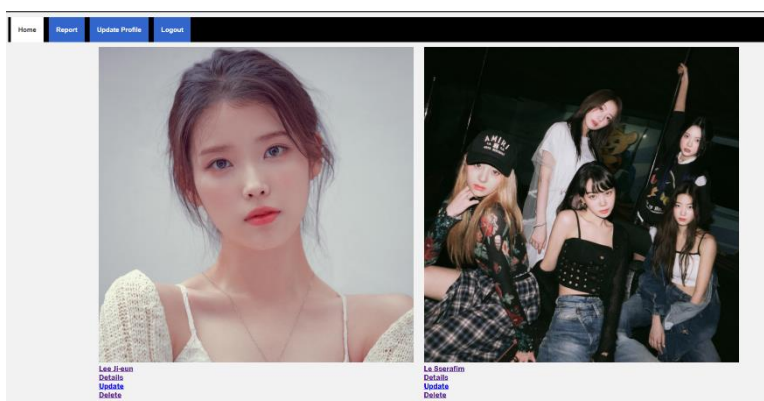
Email

Gender

Address

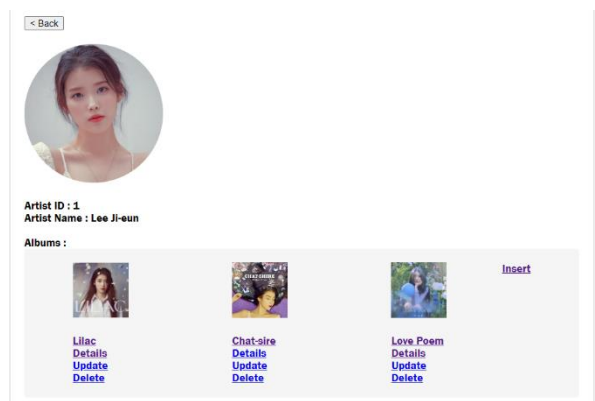
Password

## 5. Login Sebagai Admin



Name

Image  No file chosen



Name

Description

Price

Stock

No file chosen

Saat user masuk sebagai **Admin**, maka terdapat beberapa perbedaan di bagian navbar, dimana terdapat tombol 'Report' dan 'update profile'. Jika Login sebagai Admin, maka user juga bisa menambahkan Artist yang di Home page dengan cara memasukkan nama dan juga File Image, Admin juga dapat menghapus Artis yang terpampang di home page. Admin juga dapat melakukan perubahan untuk Album Details mulai dari Detele album, mengatur stok, harga, nama, dan juga deskripsi album yang dimiliki artis tersebut. Jika Admin ingin menambahkan koleksi Album yang dimiliki artis, Admin harus memasukkan data berupa Nama, deskripsi album, harga, stock, dan juga mengunggah file berupa gambar album. Admin juga dapat melihat Report dari hasil penjualan album-album yang dibeli oleh customer.

- **Reference**

1. Chauhan, K. (2015, October 30). *ASP.NET and its framework*. International Journal of Innovative Research in Technology. <https://www.ijirt.org/Article?manuscript=142726>
2. Oukes, P., Andel, M. van, Folmer, E., Bennett, R., & Lemmen, C. (2021, March 5). *Domain-driven design applied to Land Administration System Development: Lessons from the Netherlands*. ScienceDirect. <https://www.sciencedirect.com/science/article/pii/S0264837721001022>
3. Vural, Hulya & Koyuncu, Murat. (2021). Does Domain-Driven Design Lead to Finding the Optimal Modularity of a Microservice?. IEEE Access. PP. 1-1. 10.1109/ACCESS.2021.3060895. [https://www.researchgate.net/publication/349516236\\_Does\\_Domain-Driven\\_Design\\_Lead\\_to\\_Finding\\_the\\_Optimal\\_Modularity\\_of\\_a\\_Microservice](https://www.researchgate.net/publication/349516236_Does_Domain-Driven_Design_Lead_to_Finding_the_Optimal_Modularity_of_a_Microservice)
4. Mishra, A. (2023, May 19). *Modernising Systems: Applying domain driven design (DDD) for modern software re-architecture*. Alok Mishra. <https://alokmishra.com/2022/09/08/modernising-monolithic-systems-applying-domain-driven-design-ddd-to-monolithic-database-refactoring/>

- **Group Member**

- 2501971105 – Adrian Nathanael Kurniadi
- 2501983156 – Glen Lordeo Tunjung
- 2501983793 – Joshua Chandra Lian
- 2501971515 – Nickholas