IMPLEMENTASI APLIKASI *E-COMMERCE* PRODUK BANK SAMPAH MENGGUNAKAN *FLUTTER* DAN *FIREBASE* SEBAGAI PLATFORM *BACKEND-AS-A-SERVICE*

Sabrina Fauzi Prameswari*[1], Noora Qotrun Nada[2]

[1]Program Studi Informatika, Universitas PGRI Semarang, Kota Semarang
[2]Program Studi Informatika, Universitas PGRI Semarang, Kota Semarang

*Email: fauziisabrina@gmail.com

Abstract.
Effective waste bank management requires information technology support to facilitate the process. This report discusses the development of a waste bank e-commerce mobile application using Flutter as an interface development framework and Firebase as a backend service. This application is designed to facilitate the user registration and login process through Firebase Authentication, and store product data through Cloud Firestore. Users can view the product list, search for items, and add items to the cart. In addition, there are management features for admins to manage product data. This research uses the Prototype method, where the system is developed through an iterative process based on user feedback until the application meets functional needs. The implementation results show that this application is able to improve efficiency in managing waste bank data and facilitate interaction between users and the system. This application is expected to be a digital solution that supports environmental education and sustainable waste management.

Keywords: waste bank, Flutter, Firebase, Cloud Firestore, Mobile Apps

Abstrak
Pengelolaan bank sampah yang efektif memerlukan dukungan teknologi guna mempermudah prosesnya. Penelitian ini membahas pengembangan aplikasi mobile *e-commerce* bank sampah menggunakan *Flutter* sebagai *framework* pengembangan antarmuka dan *Firebase* sebagai layanan *backend*. Aplikasi ini dirancang untuk memfasilitasi proses registrasi dan login pengguna melalui *Firebase Authentication*, serta menyimpan data produk melalui *Cloud Firestore*. Pengguna dapat melihat daftar produk, mencari barang, dan menambahkan item ke keranjang. Selain itu, terdapat fitur manajemen untuk admin dalam mengelola data produk. Penelitian ini menggunakan metode *Prototype*, di mana sistem dikembangkan melalui proses iteratif berdasarkan umpan balik pengguna hingga aplikasi memenuhi kebutuhan fungsional. Hasil implementasi menunjukkan bahwa aplikasi ini mampu meningkatkan efisiensi dalam pengelolaan data bank sampah dan mempermudah interaksi antara pengguna dengan sistem. Aplikasi ini diharapkan menjadi solusi digital yang mendukung edukasi lingkungan dan pengelolaan sampah yang berkelanjutan.

Kata kunci : Bank sampah, *Flutter, Firebase, Cloud Firestore*, Aplikasi Mobile

1. Pendahuluan

Sistem penanganan sampah yang tidak tepat menimbulkan masalah baru seperti gangguan kesehatan dan lingkungan. Dalam upaya meningkatkan kesadaran akan pentingnya pengelolaan sampah, Program ini tidak hanya mendidik mengenai pentingnya

menjaga lingkungan, tetapi juga mendorong partisipasi aktif dalam memilah dan menabung sampah anorganik yang dapat didaur ulang [1]. Saat ini, pencatatan kegiatan bank sampah telah dilakukan melalui sebuah website yang mempermudah administrasi dan pelaporan.

Seiring meningkatnya antusiasme dan potensi dari hasil pengumpulan sampah, muncul kebutuhan untuk mengembangkan sistem yang tidak hanya berfungsi sebagai pencatatan, tetapi juga mampu memfasilitasi transaksi jual beli hasil olahan atau produk dari bank sampah. Untuk menjawab kebutuhan tersebut, peneliti merancanglah sebuah aplikasi mobile berbasis *e-commerce* yang memungkinkan pengguna, untuk melihat dan membeli produk hasil bank sampah.

Aplikasi ini dikembangkan menggunakan *Flutter* sebagai *framework UI cross-platform*, dengan *Firebase* sebagai *Backend-as-a-Service (BaaS)* yang mendukung fitur-fitur seperti autentikasi pengguna, penyimpanan data, dan pengelolaan database. Penggunaan platform mobile diharapkan mampu menjangkau lebih banyak pengguna, memberikan kemudahan akses, serta meningkatkan nilai dan dampak dari kegiatan bank sampah yang telah berjalan.

2. Metode

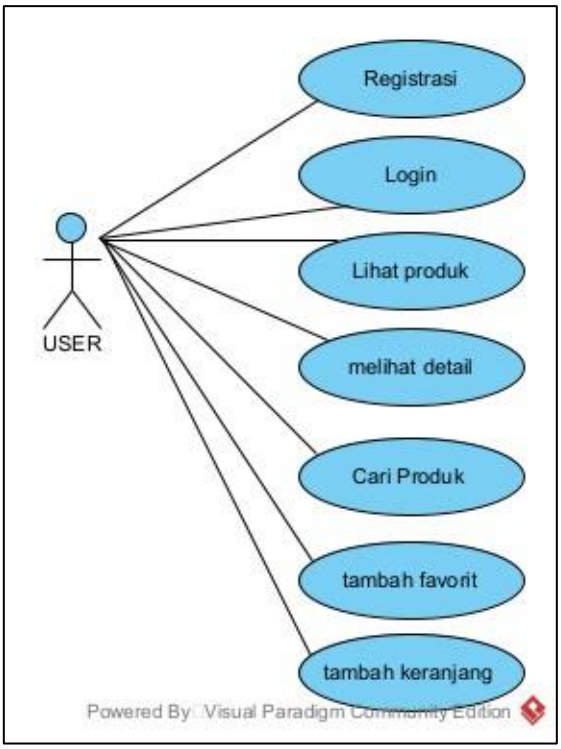
Perancangan dilakukan melalui dua pendekatan utama yaitu menggunakan metode prototyping dan Perancangan sistem menggunakan diagram *UML (Unified Modeling Language). Prototyping* adalah pendekatan pengembangan sistem yang digunakan dalam penelitian ini. Metode *Prototyping* adalah metode pengembangan sistem dimana hasil Analisa sistem langsung diterapkan kedalam sebuah model tanpa menunggu seluruh sistem selesai [2]. Dalam pengembangan aplikasi *e-commerce* hasil bank sampah, metode *prototyping* digunakan karena proses pengembangan melibatkan interaksi langsung dengan *stakeholder*. Penerapan metode ini sangat sesuai untuk aplikasi bank sampah karena mempermudah dalam menyelaraskan kebutuhan fungsionalitas seperti pengelolaan produk, pencarian produk, penambahan ke keranjang, dan sistem *login/register* pengguna, dengan ekspektasi pengguna akhir. Hal ini juga memungkinkan pengujian dini terhadap fitur-fitur yang terintegrasi dengan *Firebase*, seperti autentikasi pengguna dan penyimpanan data produk di *Firestore*.

Tahapan-tahapan dalam metode Prototyping yang diterapkan pada pengembangan sistem ini adalah sebagai berikut:

1. Identifikasi
   Pada awal perencanaan, Pengembang melakukan pertemuan dengan para stakeholder untuk menentukan kebutuhan perangkat lunak yang saat itu diketahui dan untuk menggambarkan area area dimana definisi lebih jauh untuk iterasi selanjutnya [3].
2. Pengembangan prototype
   Setelah kebutuhan telah terkumpul, dirancanglah sistem berupa rancangan antarmuka atau fungsi-fungsi dasar untuk memberikan gambaran kepada pengguna tentang bentuk dan alur kerja sistem.
3. Evaluasi prototype
   Pengguna atau stakeholder menguji coba untuk mengetahui apakah sistem sudah sesuai dengan harapan. Masukan serta saran pada tahap ini menjadi bahan untuk penyempurnaan prototype.
4. Penyempurnaan prototype
   Prototype disempurnakan berdasarkan masukan. Tahapan ini dilakukan secara berulang (iteratif) sampai sistem sudah sesuai dengan kebutuhan yang diinginkan.
5. Implementasi
   Setelah prototype dinyatakan sesuai, sistem final dibangun dan diimplementasikan secara menyeluruh. Sistem diuji secara menyeluruh untuk memastikan fungsionalitasnya berjalan dengan lancar.

Unified Modeling Language (UML) adalah sebuah bahasa pemodelan perangkat lunak yang telah distandardisasi sebagai media penulisan cetak biru *(blueprints)* perangkat lunak *(Pressman). UML* bisa saja digunakan untuk visualisasi, spesifikasi, kontruksi dan dokumentasi beberapa bagian bagian dari system yang ada dalam perangkat lunak [4].

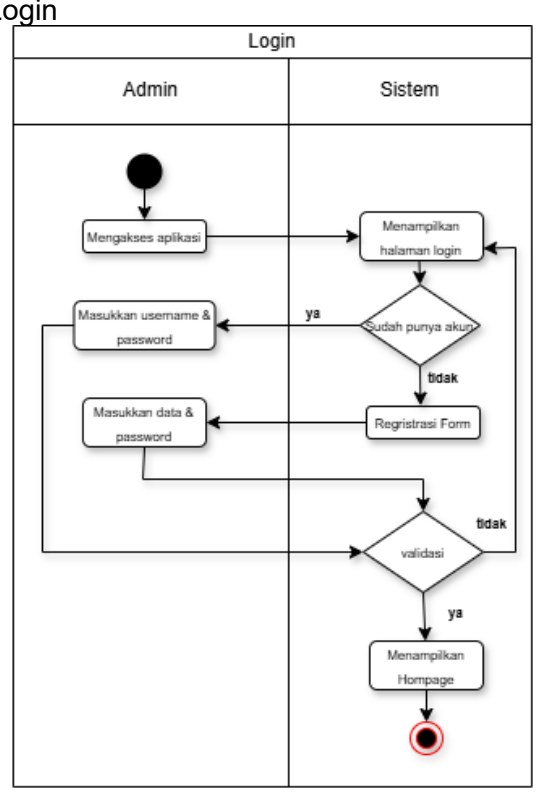
Aplikasi mobile dirancang berdasarkan kebutuhan yang telah dianalisis sebelumnya. Tujuan dari proses ini adalah untuk memastikan bahwa aplikasi yang dikembangkan sesuai dengan alur bisnis, mudah dipahami oleh pengguna, serta dapat berjalan efektif dan efisien pada perangkat mobile. Use Case Diagram pada Gambar 2.1 menggambarkan hubungan antara sistem dengan aktor yang memfokuskan pada fungsi apa yang ada pada aplikasi, bukan berdasarkan urutan kejadian.



Gambar 2.1 *Usecase* Diagram

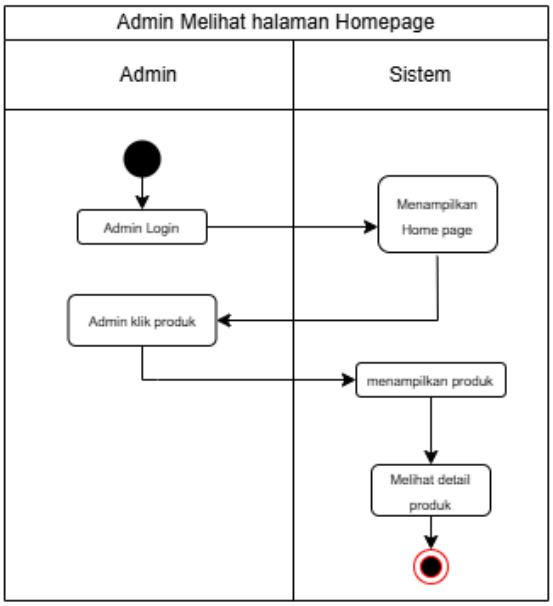
Activity diagram menggambarkan aliran fungsionalitas dalam suatu sistem informasi. Secara lengkap, *activity diagram* mendefinisikan dimana *workflow* dimulai, dimana berhentinya, aktifitas apa yang terjadi selama *workflow*, dan bagaimana urutan kejadian aktifitas tersebut [5]. Adapun gambaran activity diagram pada aplikasi ini dijelaskan pada gambar gambar berikut:

a) Activity Diagram Login



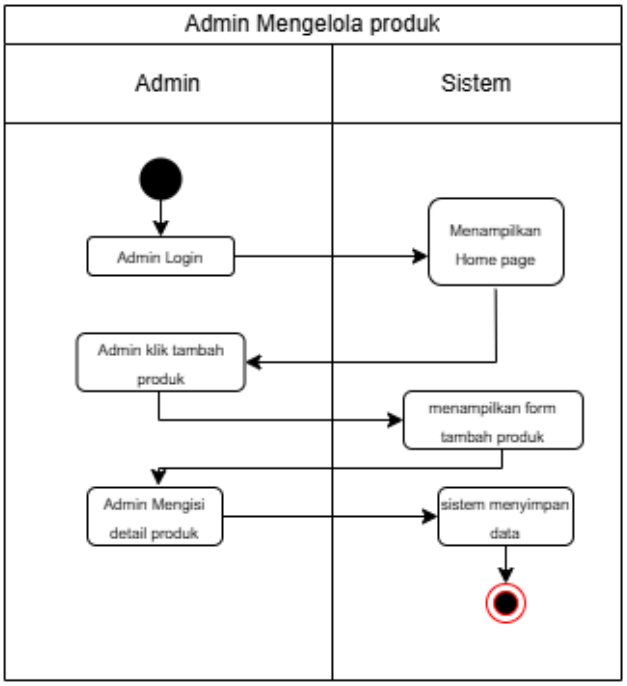
Gambar 2.2 Activity diagram login

b) Activity Admin Melihat Home Page



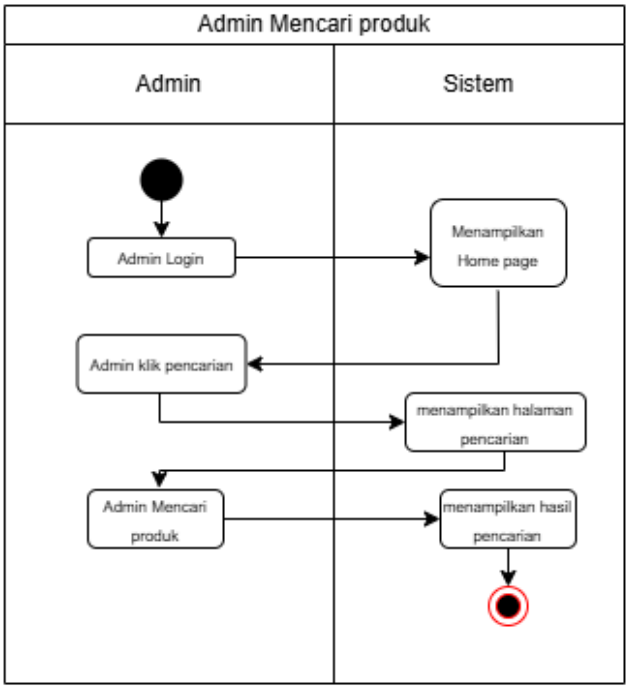
Gambar 2.3 activity diagram melihat home page

c) Activity Admin Mengelola Produk



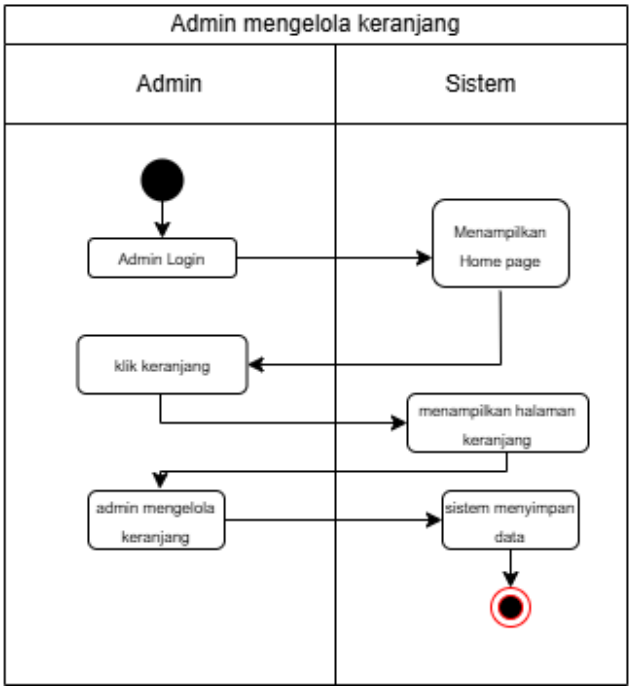
Gambar 2.4 Activity Diagram Mengelola Produk

d) Activity Admin Mencari Data Produk



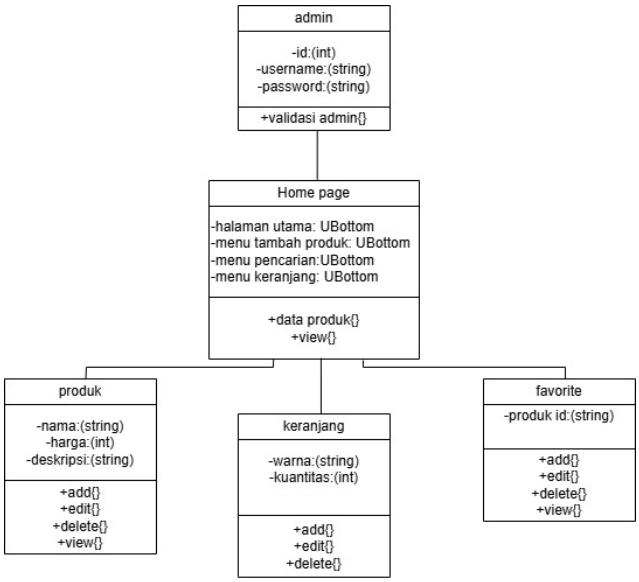
Gambar 2.5 Activity Diagram Mencari Data Produk

e) Activity Admin Mengelola Keranjang



Gambar 2.6 activity diagram mengelola keranjang

Class diagram pada Gambar 2.6 menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class Diagram* merupakan alur jalannya sebuah database pada sistem yang akan dibangun atau dibuat. *Class diagram* juga disebut kumpulan dari beberapa *class* dan relasinya [6]. Berikut merupakan gambaran class diagram dari sistem ini:



Gambar 2.6 *Class* Diagram

3. Hasil dan Pembahasan

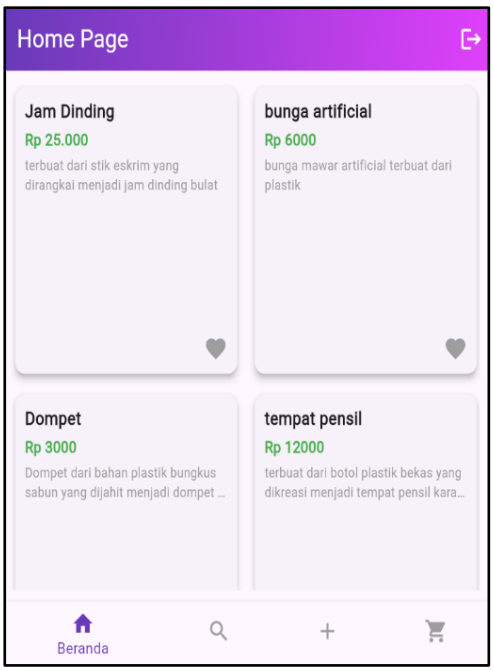
Aplikasi yang dikembangkan memiliki beberapa fitur utama yang masing-masing didukung oleh layanan Firebase dan dibangun menggunakan framework Flutter. Setiap fitur telah melalui proses perancangan dan implementasi menggunakan pendekatan metode *Prototyping*.

3.1. Penyajian Hasil

Halaman login menjadi tampilan pertama yang dilihat oleh pengguna saat membuka aplikasi. Pengguna yang sudah memiliki akun dapat langsung masuk dengan mengisi *email* dan *password*, sedangkan pengguna baru dapat memilih opsi *Register* untuk membuat akun baru. Pada halaman registrasi, pengguna diminta untuk mengisi data diri berupa nama, email, dan password. Setelah proses registrasi berhasil, pengguna akan diarahkan kembali ke halaman login untuk mulai mengakses aplikasi. Implementasi autentikasi ini mendukung skenario autentikasi lainnya seperti menggunakan Gmail, Facebook, dan Twitter, meskipun dalam tahap awal pengembangan hanya email dan password yang difokuskan. Gambar 3.1.1 menampilkan tampilan *login* dan *register* pada aplikasi

Gambar 3.1.1 Gambar tampilan login dan register

Halaman utama aplikasi terdiri dari empat menu yang dapat diakses melalui navigasi bawah: *Home, Search, Add, dan Cart*. Halaman Home menampilkan daftar produk dari bank sampah, yang mencakup nama produk, harga, serta tombol tindakan seperti tambah ke favorit dan tambah ke keranjang. Data produk ditampilkan secara dinamis dengan menggunakan *Firebase Cloud Firestore. Firestore* memungkinkan aplikasi untuk membaca dan menampilkan data secara real-time, sehingga pembaruan data langsung terlihat oleh pengguna. Gambar 3.1.2 dibawah ini menampilkan tampilan home page



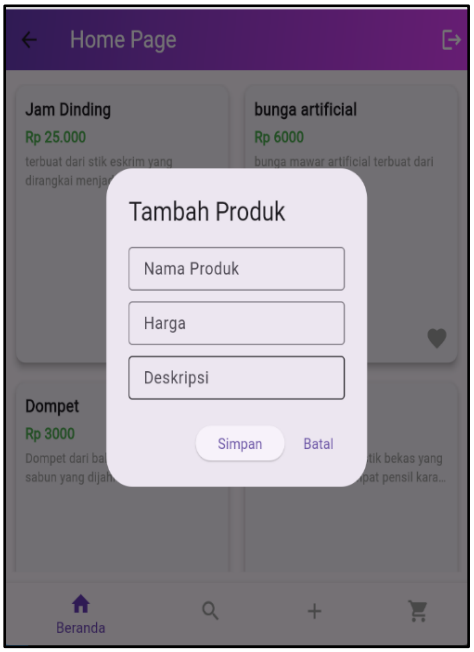
Gambar 3.1.2 Gambar Tampilan Home Page

3.1.3. Halaman Detail Tambah Produk

Halaman ini menampilkan detail lengkap dari produk yang dipilih pengguna. Informasi yang disediakan mencakup:

1. Nama dan harga produk
2. Deskripsi: bahan, warna, jenis
3. Opsi kustomisasi: jumlah, varian, dan lainnya

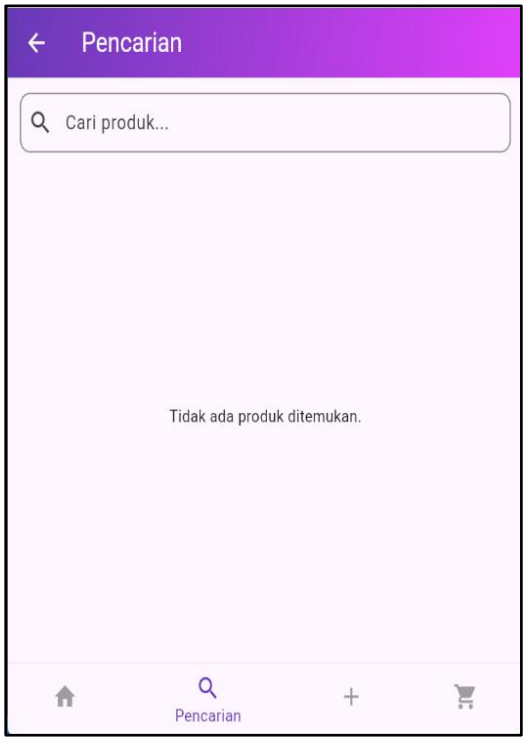
Gambar 3.1.3 diabwah adalah tampilan yang membantu pengguna memahami spesifikasi produk sebelum menambahkannya ke keranjang.



Gambar 3.1.3 Gambar Tampilan Tambah Produk

3.1.4. Halaman Pencarian

Fitur pencarian memungkinkan pengguna untuk menemukan produk dengan cepat menggunakan kata kunci seperti nama produk, kategori, atau deskripsi. Hasil pencarian ditampilkan dalam daftar lengkap dengan gambar, nama, dan harga produk. Pengguna dapat langsung menuju halaman detail produk atau menambahkannya ke favorit dan keranjang dari hasil pencarian ini. Gambar 3.1.5 ini merupakan tampilan halaman pencarian.



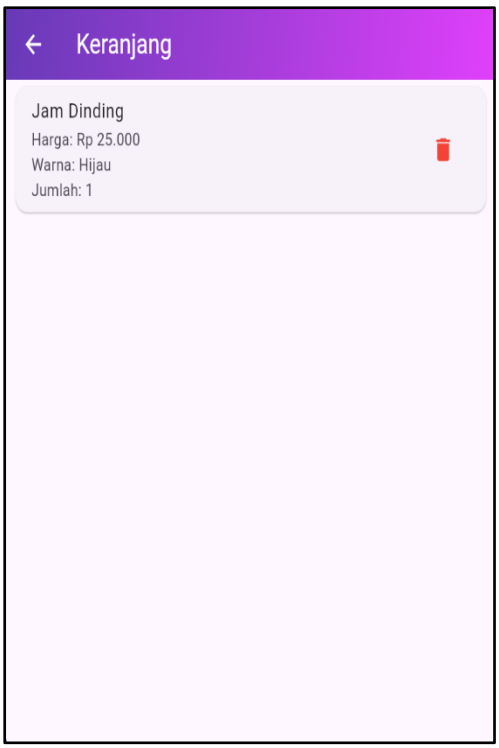
Gambar 3.1.4 Gambar Tampilan Pencarian

3.1.5. Halaman Keranjang

Halaman keranjang berfungsi sebagai tempat pengguna mengelola produk yang ingin dipesan. Produk yang ditambahkan dari halaman detail akan muncul di sini, dengan informasi seperti:

1. Nama produk
2. Jumlah item yang dapat diubah
3. Harga satuan dan total
4. Opsi hapus item

Pengguna dapat meninjau kembali daftar belanjaan sebelum melanjutkan ke proses pemesanan. Gambar berikut merupakan tampilan halaman keranjang.



Gambar 3.1.5 Gambar Tampilan Keranjang Page

3.2. Pembahasan

Dalam perancangan aplikasi ini menggunakan *Firebase Authentication* untuk menangani proses login dan registrasi. *Firebase Auth* merupakan fitur dari *Firebase* yang menyediakan layanan *backend* dengan SDK yang siap untuk digunakan untuk mengauthentikasi pengguna ke sebuah aplikasi yang ingin dikembangkan. Dengan fitur ini pengguna dapat melakukan authentikasi menggunakan gmail, facebook, twitter dan lainnya [7]. *Firebase* menyediakan layanan autentikasi berbasis email dan password secara sederhana namun aman. Proses mengintegrasikan flutter dengan firebase dilakukan melalui beberapa tahap:

1. Inisialisasi *Firebase* menggunakan *Firebase.initializeApp()* pada *main.dart.*

   Firebase perlu diinisialisasi terlebih dahulu sebelum digunakan dalam aplikasi *Flutter*. Inisialisasi ini dilakukan agar aplikasi dapat terhubung ke layanan *Firebase* seperti *Authentication, Firestore*, dan lainnya. Gambar 3.2.1 menunjukkan proses inisialisasi Firebase pada file main.dart, yang merupakan tahap awal integrasi antara aplikasi *Flutter* dengan *Firebase.*

```
Run | Debug | Profile
void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp(
    options: DefaultFirebaseOptions.currentPlatform,
  );
```

Gambar 3.2.1 Gambar Inisialisasi Firebase

2. Pemanggilan *method createUserWithEmailAndPassword()* untuk registrasi pengguna.

   Method ini digunakan untuk membuat akun baru dengan email dan password. Ketika pengguna melakukan registrasi melalui form, data akan dikirim ke *Firebase*

Auth untuk dibuatkan akun. Gambar 3.2.2 menunjukkan proses pemanggilan *method* pada *register page*.

```
void register(BuildContext context) async {
  try {
    final credential = await FirebaseAuth.instance.createUserWithEmailAndPassword(
      email: controllerEmail.text,
      password: controllerPassword.text,
    );
```

Gambar 3.2.2 Gambar Pemanggilan Method Pada Register Page

3. Pemanggilan *method signInWithEmailAndPassword()* untuk proses login.

Setelah registrasi berhasil, pengguna dapat masuk menggunakan akun yang telah dibuat. Method ini digunakan untuk melakukan proses autentikasi login berdasarkan email dan password yang tersimpan di *Firebase Authentication.* Gambar 3.2.3 menunjukkan proses pemanggilan *method* pada *login page.*

```
try {
  final credential = await FirebaseAuth.instance.signInWithEmailAndPassword(
    email: controllerEmail.text.trim(),
    password: controllerPassword.text,
  );
```

Gambar 3.2.3 Gambar Pemanggilan Methode Pada Login Page

4. Penanganan respons error menggunakan *FirebaseAuthException.*

Firebase akan mengembalikan exception jika terjadi kesalahan dalam proses login atau registrasi, seperti email tidak terdaftar, password salah, atau email sudah digunakan. Penanganan error ini dilakukan menggunakan *blok try-catch* dan *FirebaseAuthException* agar pesan kesalahan dapat ditampilkan secara informatif kepada pengguna. Gambar 3.2.4 menunjukkan proses penanganan *error* pada *Firebase Auth*.

```
} on FirebaseAuthException catch (e) {
String errorMessage = 'Login Gagal';
if (e.code == 'user-not-found') {
  errorMessage = 'User tidak ditemukan';
} else if (e.code == 'wrong-password') {
  errorMessage = 'Password salah';
} else {
  errorMessage = 'Error: ${e.message}';
}
```

Gambar 3.2.4 Gambar Penanganan Error Pada Firebase Auth

Untuk mendukung pengelolaan dan penampilan data produk secara dinamis, digunakan layanan *Firebase Cloud Firestore*. *Firestore* database berbasis *cloud* yang mendukung struktur dokumen dan koleksi, serta mendukung pembacaan data secara *real-time* menggunakan stream. Gambar 3.2.5 menunjukkan penggunan *Firebase Cloud* untuk menyimpan *database* dalam pembuatan *mobile app* ini.

```
Future<List<Map<String, dynamic>>> fetchProducts() async {
  try {
    QuerySnapshot snapshot =
        await FirebaseFirestore.instance.collection('products').get();
    return snapshot.docs
        .map((doc) => {"id": doc.id, ...doc.data() as Map<String, dynamic>})
        .toList();
  } catch (e) {
    return [];
```

Gambar 3.2.5 Penggunaan *Firebase Cloud* untuk Penyimpanan *Database*

4. Kesimpulan

Berdasarkan hasil dari pengembangan aplikasi *e-commerce* produk hasil bank sampah, dapat disimpulkan beberapa hal sebagai berikut:

1. Aplikasi *e-commerce* ini berhasil dibangun sebagai sistem pencatatan dan pengelolaan data produk hasil bank sampah berbasis mobile (Android). Aplikasi ini menyediakan fitur login, registrasi, kelola produk, pencarian produk, serta penambahan produk ke favorit dan keranjang, yang seluruhnya dirancang untuk memudahkan admin dalam mencatat dan mengelola data produk secara efisien dan terstruktur. Dengan menggunakan pendekatan perancangan sistem berbasis *UML (Unified Modeling Language),* proses pengembangan menjadi lebih terarah dan terdokumentasi dengan baik.
2. Aplikasi ini telah berhasil memanfaatkan *Firebase* sebagai *Backend-as-a-Service (BaaS)* untuk kebutuhan autentikasi pengguna dan penyimpanan database produk. *Firebase Authentication* digunakan untuk proses login dan registrasi admin, sementara *Cloud Firestore* digunakan sebagai basis data utama untuk menyimpan informasi produk, pengguna, dan interaksi seperti favorit dan keranjang. Penggunaan *Firebase* terbukti efektif dan praktis, terutama untuk pengembangan aplikasi skala kecil hingga menengah seperti proyek ini.

5. Referensi

[1] R. Auliani, "Peran Bank Sampah Induk dalamPengelolaan SampahKota Medan(Studi Kasus : Bank Sampah Induk Sicanang, Belawan, Medan)," *Jurnal Abdidas Volume 1Nomor 5 Tahun 2020Halaman330-337,* 2020.

[2] K. Wijaya, "Implementasi Metode Prototype Dalam Rancang Bangun Sistem Informasi Perpustakaan Menggunakan Java (Netbeans 7.3) (Studi Kasus SMK N 01 Prabumulih)," *jurnal sisfokom (sistem informasi dan komputer),* 2019.

[3] D. Ardiyansah, "IMPLEMENTASI METODE PROTOTYPINGPADA SISTEM INFORMASI PENGADAAN BARANG CETAKAN BERBASIS WEB," *Hexagon Jurnal Teknik dan Sains Fakultas Teknik Universitas Teknologi Sumbawa,* 2021.

[4] R. A. A. C. Mia Sumiati, "Pemodelan UML untuk Sistem Informasi Persewaan Alat Pesta," JURNAL FASILKOM, 2021.

[5] L. P. Dewi, "PEMODELAN PROSES BISNIS MENGGUNAKAN ACTIVITY DIAGRAM UML DAN BPMN (STUDI KASUS FRS ONLINE)," Scientific Repository, 2012.

[6] S. W. Ramdany, "Penerapan UML Class Diagram dalam Perancangan Sistem Informasi Perpustakaan Berbasis Web," JournaL of Industrial and Engineering System, 2024.

[7] F. R. Pratama, "Pengembangan Aplikasi E-CommerceMenggunakan Payment Gateway Midtrans," JPTIIK, 2020.