IT-EXPLORE
Jurnal Penerapan Teknologi Informasi dan Komunikasi
ISSN 2828-7940, e-ISSN 2829-1727
Volume 03 Nomor 02 Tahun 2024

IMPLEMENTASI *REST API* MENGGUNAKAN *RETROFIT* PADA APLIKASI MONITORING *GROOMING* BERBASIS ANDROID

Santika Widyaningtyas [1)], Teguh Wahyono 2 [2)]

[1,2)]Program Studi Teknik Informatika, Fakultas Teknologi Informasi
Jl. O.Notohamidjojo, no 1-10, Blotongan, Salatiga
Email : 672020155@student.uksw.edu[1], teguh.wahyono@uksw.edu[2]

Riwayat artikel:
Submitted: 17-04-2024 *Revised: 12-06-2024* *Published: 21-06-2024*

Abstrak **–** Penelitian ini bertujuan untuk melakukan pengembangan aplikasi monitoring *grooming* berbasis Android. Aplikasi ini dirancang dengan tujuan meningkatkan efisiensi dalam pemesanan dan pencatatan pesanan *grooming* serta mempermudah manajemen jadwal dan layanan bagi *petgroomer*. Metode yang digunakan dalam penelitian ini adalah metode *prototyping*. Terdiri dari tiga tahap yaitu komunikasi dengan klien, pengembangan sistem, dan pengujian sistem. Setelah melakukan komunikasi dengan klien untuk mengetahui kebutuhan sistem, selanjutnya yang dilakukan adalah melakukan pengembangan sistem. Setelah mengetahui kebutuhan dari sistem, selanjutnya disusun *Unified Modeling Language (*UML*)* dengan tujuan untuk meningkatkan efisiensi pengembangan sistem. Beberapa diagram UML yang dipakai, seperti *use case*, diagram alir, dan diagram *activity*. Kemudian, sistem diimplementasikan dengan menggunakan *REST API* melalui *Retrofit* dan bahasa pemrograman Java, menggunakan *tools* Android Studio. Langkah yang terakhir adalah melakukan pengujian terhadap sistem. Aplikasi ini dapat memberikan solusi yang efektif bagi industri *grooming* hewan peliharaan dengan meningkatkan efisiensi operasional dan memberikan pengalaman yang lebih baik bagi pemilik hewan peliharaan dan *groomer*.

Kata Kunci **–** Retrofit, Android, *Monitoring*, *Grooming*

Abstract **–** *This study aims to develop a mobile-based grooming monitoring application. This app is designed to enhance efficiency in booking and recording grooming orders, and to facilitate the management of schedules and services for pet groomers. The research method used is the prototyping method, which consists of three stages: communication with clients, system development, and system testing. After communicating with clients to determine the system's requirements, the next step is system development. Upon understanding the needs of the system, a Unified Modeling Language (UML) is compiled to increase system development efficiency. Several UML diagrams are used, such as use case diagrams, flowcharts, and activity diagrams. Then, the system is implemented using Rest API through Retrofit and the Java programming language, utilizing Android Studio tools. The final step is to test the system. This application can provide an effective solution for the pet grooming industry by improving operational efficiency and offering a better experience for pet owner and groomers.*

Keywords **–** *Retrofit, Android, Monitoring, Grooming*

.

I. PENDAHULUAN

Semakin meningkatnya peminatan terhadap pemeliharaan hewan seperti kucing karena tingkah mereka yang menggemaskan dan visual mereka yang lucu [1]. Menjadikan tempat *grooming* hewan peliharaan menjadi bisnis yang semakin populer dalam beberapa tahun terakhir. Para pemilik hewan peliharaan sangat peduli terhadap kesejahteraan dan penampilan hewan peliharaan mereka [2]. Seiring dengan pertumbuhan populasi kucing sebagai hewan peliharaan yang populer, *petgroomer* mulai menghadapi beberapa tantangan yang perlu diatasi seperti peningkatan permintaan. Semakin banyaknya pemilik kucing yang ingin memberikan perawatan *grooming* berkualitas untuk hewan peliharaan mereka, *petgroomer* menghadapi peningkatan permintaan yang sulit diatasi hanya dengan metode tradisional.

Dalam industri *grooming* hewan peliharaan, manajemen pemesanan dan rekapitulasi merupakan aspek penting yang memengaruhi efisiensi operasional dan kepuasan pelanggan. Saat ini, banyak *petgroomer* menghadapi tantangan dalam melacak dan memproses pemesanan dengan tepat waktu didampingi dengan mulai meningkatnya permintaan *grooming*, sementara pemilik hewan peliharaan menginginkan kemudahan dalam memantau jadwal *grooming* dan riwayat layanan yang telah diterima oleh hewan peliharaan mereka. Oleh karena itu, dibutuhkan aplikasi yang dapat meningkatkan efisiensi operasional tersebut dan memudahkan pengguna [3].

Aplikasi *monitoring grooming* ini dirancang dengan memperhatikan tiga aktor pengguna utama yaitu pengguna, admin, dan *groomer*. Ketiganya dirancang dengan berbasis *mobile*. Untuk pengguna admin memiliki fungsi utama yaitu menambahkan daftar antrian *grooming* yang masuk, memantau proses *grooming* secara *real-time*, memantau status setiap proses *grooming*, yaitu sudah dijemput, sedang diproses, dan sudah selesai. Dan juga melihat pencatatan semua pemesanan *grooming* yang masuk. Selain itu juga, admin dapat melihat kinerja tiap *groomer*. Untuk pengguna *groomer* memiliki fungsi yaitu melihat semua orderan *grooming* yang masuk dan mengubah status tiap proses *grooming* agar dapat terpantau oleh admin utama secara *real-time*. Dan untuk sisi pengguna, pengguna dapat melakukan pesanan *grooming* dan melihat riwayat pemesanan yang sudah dilakukan.

Pengembangan aplikasi berbasis Android untuk memonitor *grooming* hewan peliharaan dengan fitur rekapitulasi pemesanan menjadi sangat penting. Aplikasi ini tidak hanya memungkinkan pemilik hewan peliharaan untuk melakukan pemesanan secara *online* dengan mudah dan cepat, tetapi juga menyediakan akses terhadap riwayat pemesanan dan layanan yang telah diterima. Dengan adanya fitur rekapitulasi, *petgroomer* dapat dengan mudah melacak jadwal pemesanan, jenis layanan yang diminta, yang semuanya sangat penting untuk memberikan layanan yang berkualitas dan personalisasi. Aplikasi ini dirancang menggunakan bahasa pemrograman *Java*, karena merupakan bahasa pemrograman yang bersifat *multiplatform* yaitu dapat digunakan di berbagai *platform* [4].

Dengan mengintegrasikan fitur rekapitulasi pemesanan ke dalam aplikasi *monitoring grooming*, diharapkan dapat meningkatkan efisiensi operasional *petgroomer*, memperkuat hubungan antara pemilik hewan peliharaan dan penyedia layanan, serta meningkatkan kepuasan pelanggan secara keseluruhan. Ini tidak hanya membawa manfaat bagi pertumbuhan bisnis *petgroomer*, tetapi juga membawa pengalaman yang lebih baik bagi para pemilik hewan.

II. KAJIAN PUSTAKA

A. Landasan Teori

Retrofit merupakan sebuah *library REST Client* yang berfungsi sebagai penghubung antara aplikasi Android dan *web service.* Retrofit dibuat oleh Square. Retrofit mengambil data yang ada di dalam basis data dan dikonversikan ke dalam bentuk GSON. Rest API merupakan alat yang dirancang pada layanan web yang berfokus pada sistem seperti *transfer* dan *request data* menggunakan http. Layanan web REST mengidentifikasikan URl dan kemudian dimodifikasi dengan perintah GET, POST, PUT, atau DELETE yang dapat direpresentasikan dalam beberapa bahasa pemrograman seperti XML atau JSON. [5]

Android Studio merupakan IDE (integrated Development Environment) pengembangan aplikasi Android yang didukung resmi dan direkomendasikan langsung oleh Google. Android Studio dilengkapi dengan Android SDK Manager, sebuah alat yang memungkinankan pengguna untuk mengunduh komponen-komponen Android SDK yang akan diperlukan oleh pengguna untuk melakukan pengembangan aplikasi [6]. Dikembangkan berdasarkan Intellij IDEA serta dilengkapi dengan ADT *plugin* (*Android Development Tools*), Android Studio memiliki fitur diantaranya: (a) didukung oleh Google Cloud Platform, (b) pembenahan *bug* yang terhitung cepat, (c) antarmuka grafis (GUI) yang memudahkan dalam pengembangkan aplikasi android, (d) dukungan *Proguard* dan *App-signing* guna meningkatkan keamanan, dan (e) proyek berbasis *Gradle Build*.

Google Maps adalah peta *online* yang merupakan layanan gratis dari Google. Layanan ini juga menyediakan API (*Application Programming Interface*) yang memudahkan pengembang lain untuk memanfaatkan layanan ini pada aplikasi yang dibuatnya. Google Maps merupakan layanan gratis yang sangat populer yang disediakan oleh Google. Dengan Google Maps kita dapat melihat peta seluruh dunia. Google Maps API merupakan sebuah *library* yang dikembangkan dengan JavaScript. Dengan Google Maps API, Google Maps dapat diakses pada *website* ataupun aplikasi yang kita buat [7].

B. Penelitian Terdahulu

Penelitian berjudul "Rancang Bangun Aplikasi Android Terintegrasi Web Service Dengan *Volley* Untuk Layanan Publik" membahas mengenai perancangan aplikasi layanan publik berbasis Android dan Website. Penelitian tersebut menghasilkan sebuah aplikasi untuk mengelola data server melalui web service dengan PHP dan MySQL dibuat untuk meliputi fungsi-fungsi dasar. Pada sisi Android, aplikasi ini memiliki antarmuka yang sederhana dan menggunakan pustaka *Volley* untuk mengakses data [8]. *Volley* dirancang oleh Google yang memudahkan pengelolaan request network di Android. Tetapi, pada penelitian ini peneliti menggunakan *Retrofit*. Penggunaan *Retrofit* lebih memudahkan dalam penanganan API dibanding dengan *Volley*. *Retrofit* memiliki dukungan fitur yang lengkap untuk aplikasi yang memerlukan pengelolaan REST API yang kompleks.

Penelitian berjudul "Penerapan Rest API Menggunakan *Retrofit* Untuk Sistem Informasi Film Berbasis Android (Studi Kasus: Sinopsis Film)" membahas mengenai pengembangan aplikasi dengan menerapkan REST API menggunakan *Retrofit* yang dapat digunakan untuk mencari informasi terkait sinopsis film. Aplikasi ini mengambil data dari API pada *website* https://www/themoviedb.org. Pengambilan data ini API ini

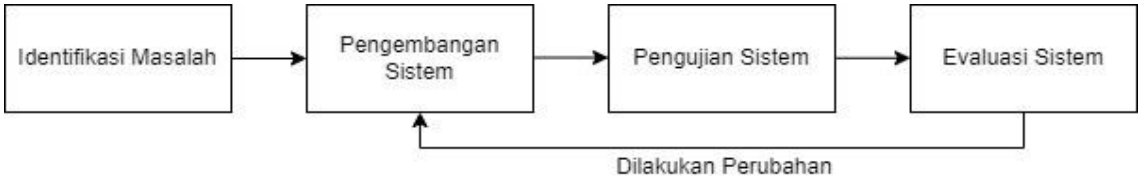
berformat JSON [9]. Penelitian tersebut menggunakan Kotlin sebagai pengembangannya dimana masih terbilang baru, berbanding terbalik dengan Java. Java sudah digunakan sejak lama sehingga ketersediaan *library* lebih terdokumentasi dan optimal.

Penelitian berjudul "Pertukaran Data Pada Aplikasi Android Menggunakan *JavaScript Object Notation* (JSON) dan REST API dengan *Retrofit* 2" membahas mengenai pengembangan aplikasi berbasis android yang digunakan untuk melaporkan kegiatan Pengabdian Kepada Masyarakat (PKM) dan penelitian dosen. Aplikasi ini melakukan pertukaran data dari klien ke server dengan menerapkan JSON dan REST API dengan *library Retrofit* 2 [10]. Metode yang digunakan pada penelitian ini adalah metode *extreme programming* dimana metode ini fleksibel dan mudah mengikuti perubahan, sedangkan untuk pengembangan yang lebih tersusun dan tertata rapi lebih baik menggunakan metode *prototyping* dimana semua kebutuhan sudah jelas tersusun di awal.

Penelitian berjudul "Penerapan REST API dalam Pengembangan Aplikasi Pemesanan Rental Mobil berbasis Web dan Mobile (Studi Kasus: CV. Dwi Cipta Rent Car)" membahas mengenai pengembangan aplikasi rental mobil berbasis *web* dan *mobile* pada CV. Dwi Cipta *Rent Car*. Aplikasi ini terdiri dari aplikasi *mobile* yang digunakan oleh *customer* dan aplikasi berbasis *web* yang digunakan oleh admin. Aplikasi ini dikembangan dengan menerapkan REST API. Tujuan digunakan REST API adalah untuk mengintegrasikan aplikasi *mobile* dan *web* yang dibuat [11].

III. METODE PENELITIAN

Metode penelitian bisa dilihat pada Gambar 1 di bawah. Gambar tersebut memperlihatkan serangkaian tahapan runtut yang dilakukan dalam pengimplementasian aplikasi berbasis *mobile* ini. Serangkaian tahapan ini bertujuan agar sistem yang dibuat dapat memenuhi standar kebutuhan klien dan dapat mencapai tujuan yang ditetapkan. Langkah-langkah tersebut terdiri dari empat tahap. Tahap pertama yaitu identifikasi masalah, tahap kedua adalah pengembangan sistem, tahap ketiga adalah pengujian sistem. Tahap paling akhir yaitu evaluasi sistem.



Gambar 1 Alur Penelitian

Tahapan pertama yaitu identifikasi masalah. Pada tahapan ini, dilakukan komunikasi untuk pembahasan tentang apa saja yang dibutuhkan dan apa tujuan dari pengembangan aplikasi tersebut. Pada tahap ini bertujuan untuk mengumpulkan informasi detail tentang kebutuhan dan ekspetasi klien serta pengguna akhir untuk memastikan aplikasi yang dikembangkan sesuai dengan kebutuhan mereka.

Selanjutnya yaitu tahapan pengembangan sistem. Metode yang digunakan dalam pengembangan sitem ini adalah metode *prototyping*. Metode *prototyping* digunakan untuk mengumpulkan informasi kebutuhan aplikasi [12]. Pada tahap ini penulis mulai melakukan pengembangan aplikasi sesuai dengan kebutuhan dari klien yang sudah

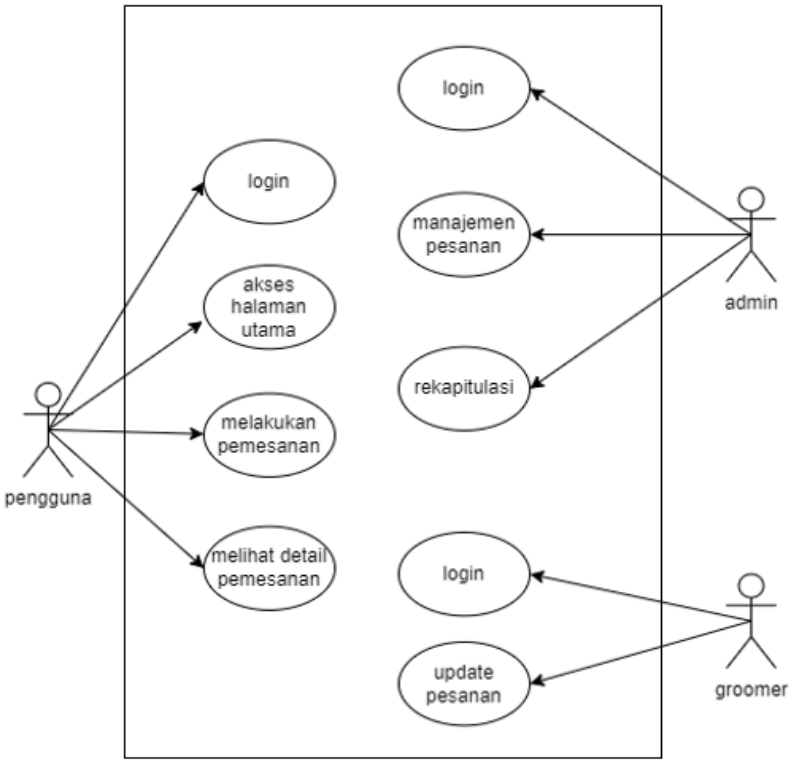
dibicarakan sebelumnya. Setelah menetapkan kebutuhan sistem, hal yang selanjutnya dilakukan adalah menyusun *Unified Modeling Language (*UML*)* dengan tujuan untuk meningkatkan efisiensi pengembangan sistem [13]. Setelah selesai mendesain UML, mulai melakukan implementasi sistem. Aplikasi ini menggunakan bahasa pemrograman *Java* dan menggunakan implementasi REST API yaitu Retrofit. Untuk *tools* yang digunakan adalah Android Studio.

Tahapan yang ketiga adalah tahap pengujian. Pada tahap ini dilakukan pengujian dengan klien maupun pengujian internal. Klien akan memberi masukan mereka aplikasi tersebut seperti desain *interface* dan kegunaan fitur-fitur. Berdasarkan masukan tersebut, buat perubahan sesuai saran dan masukan klien. Sedangkan untuk pengujian internal, dilakukan pengujian sistem menggunakan metode *blackbox testing*. Tahap terakhir dari penelitian ini adalah evaluasi sistem. Tahap ini dilakukan untuk melihat apakah sistem tersebut memerlukan perbaikan atau tidak.

IV. HASIL DAN PEMBAHASAN

A. Tahap Desain Sistem

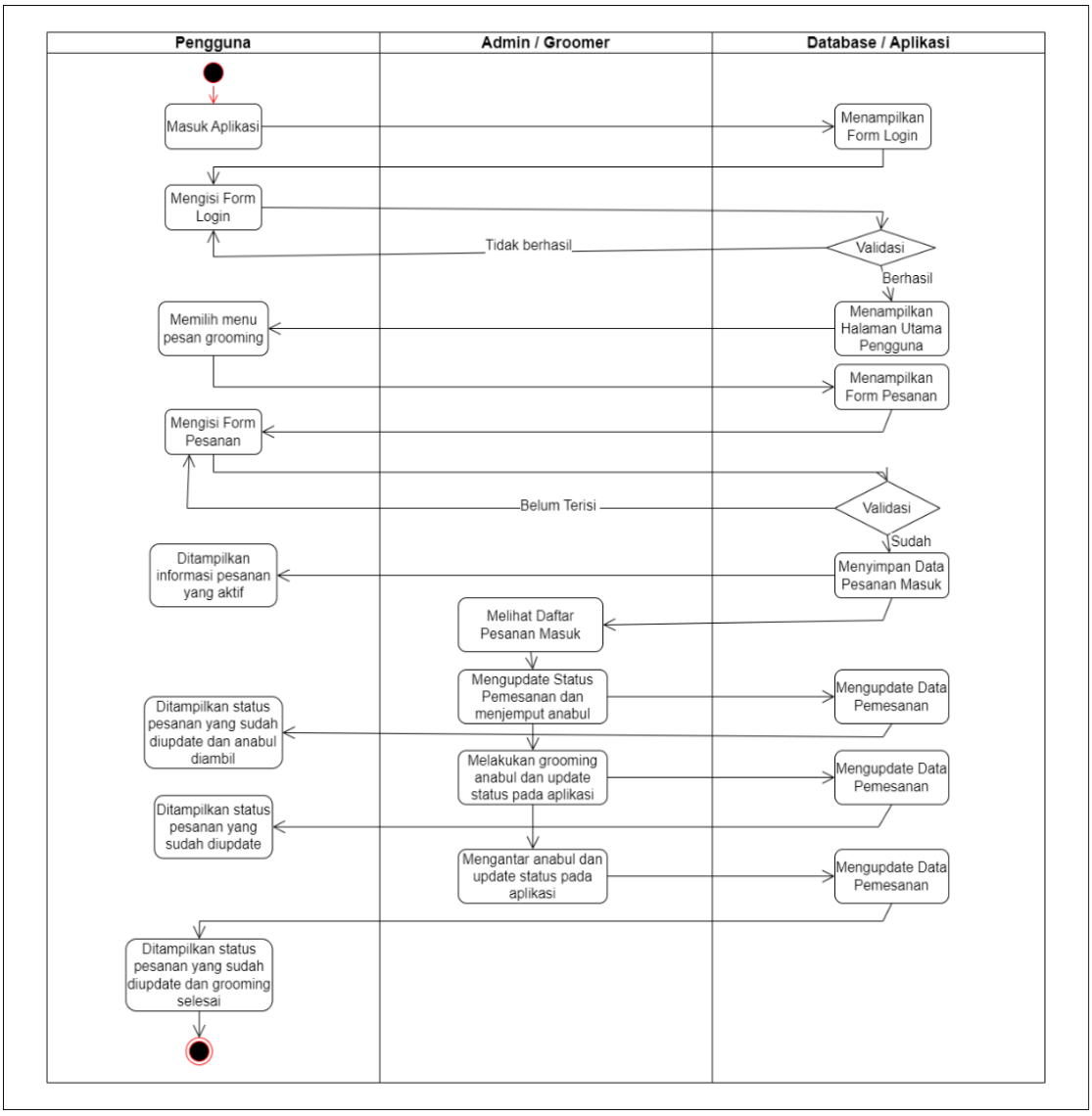
Rancangan sistem bisa dilihat pada Gambar 1 di bawah. Gambar tersebut menjelaskan tentang *use case* diagram dari aplikasi ini dimana menggambarkan interaksi tiap aktor [14].



Gambar 2 Use case diagram

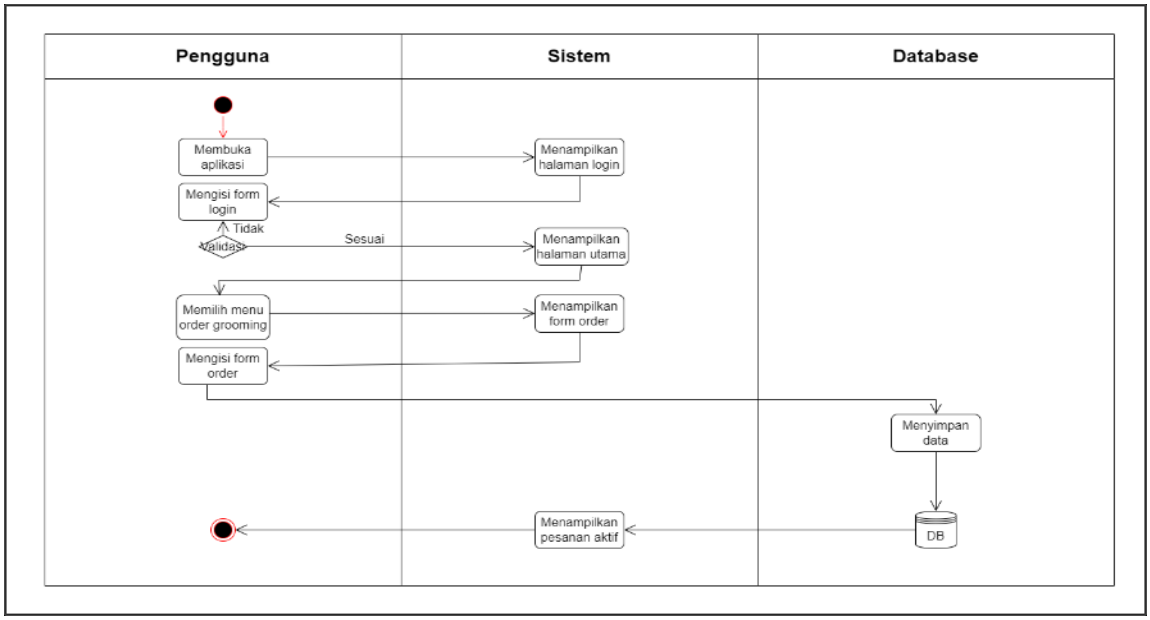
Terdapat tiga aktor utama dalam *use case* diagram ini, yaitu pengguna, admin, dan *groomer*. Peran dari admin yaitu dapat memanajemen pemesanan *grooming*, seperti dapat menambahkan pesanan dari pelanggan langsung dan dapat memantau status dari tiap

daftar pemesanan *grooming*. Selain itu, juga dapat melihat rekapitulasi semua pesanan masuk dan dapat memantau kinerja tiap *groomer*. Peran *groomer* yaitu dapat melihat tiap pesanan yang masuk, dan melakukan update agar dapat mengubah status pesanan yang sedang dikerjakan. Di sisi pengguna, pengguna dapat melakukan pesanan *grooming* dan melihat detail pemesanan yang sudah dilakukan.



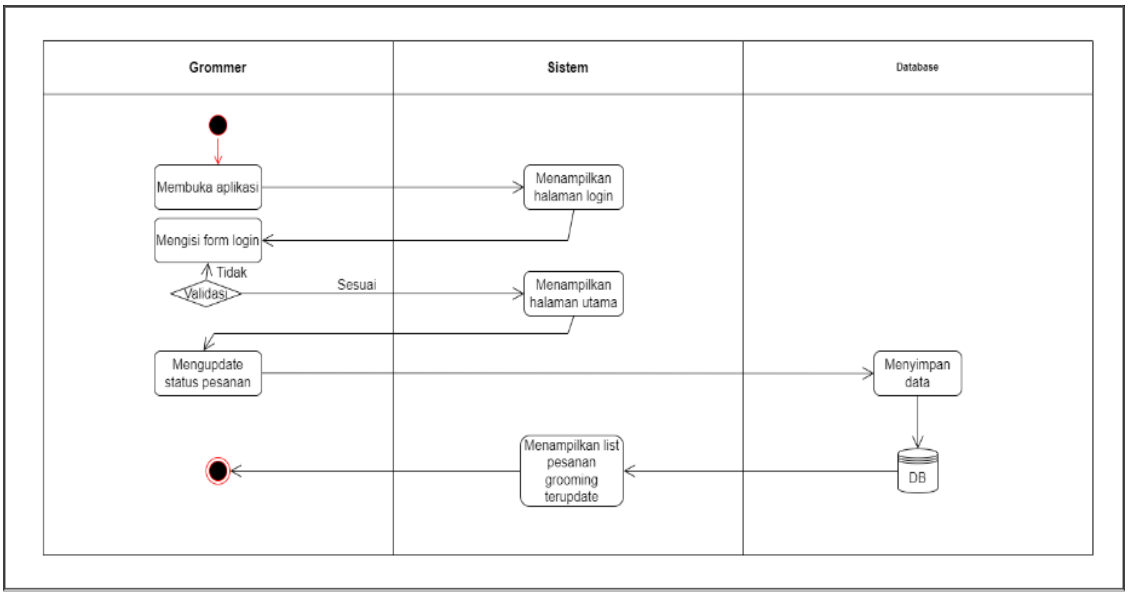
Gambar 3. Diagram alir

Pada gambar 3 menjelaskan tentang diagram alir dari aplikasi tersebut. Diagram alir menggambarkan sebuah alur kerja atau proses pada sebuah sistem. Pada gambar diatas menjelaskan tentang proses utama dari aplikasi tersebut yaitu dimulai dari pengguna melakukan pemesanan *grooming* sampai dengan *groomer* menyelesaikan pemesanan.



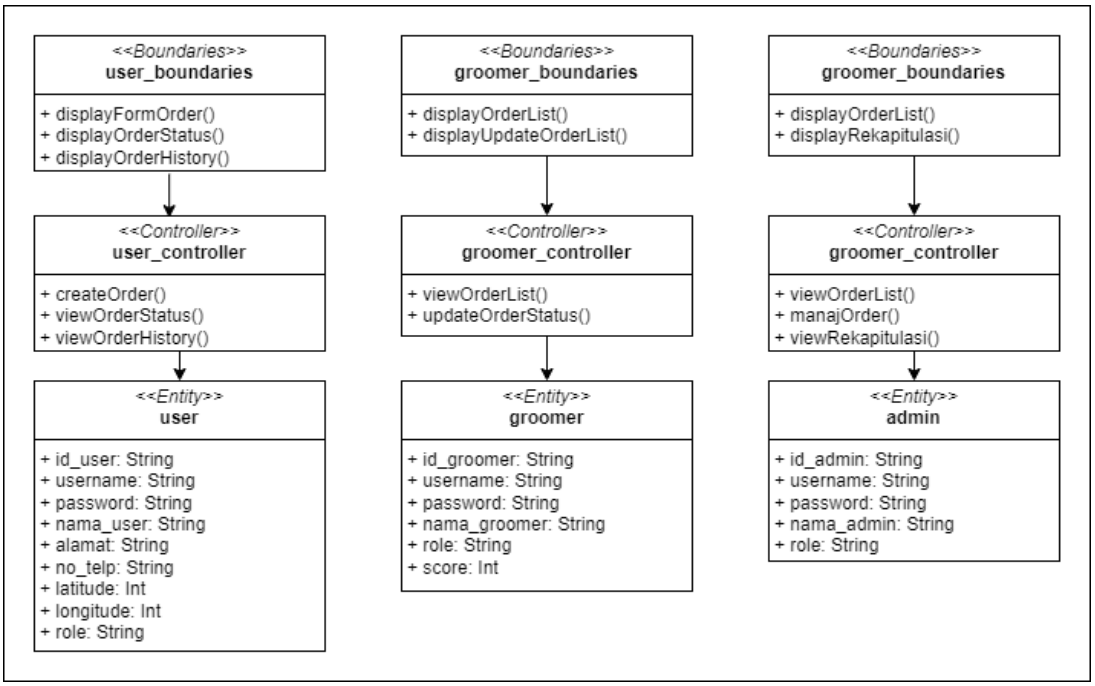
Gambar 4. Diagram *activity* pengguna

Gambar 4 menjelaskan tentang diagram *activity* pengguna saat melakukan pemesanan *grooming*. Pada diagram tersebut memperlihatkan proses interaksi pengguna dengan sistem dan *database* [15]. Dimulai dari pengguna membuka aplikasi tersebut. Pengguna akan diarahkan langsung ke halaman awal yaitu halaman *login*. Setelah itu, pengguna harus memasukkan *username* dan *password* untuk dapat akses masuk ke dalam aplikasi. Setelah berhasil masuk, akan diarahkan ke halaman selanjutnya yaitu halaman utama. Untuk dapat melakukan pemesanan *grooming*, pengguna harus klik tombol ”pesan *grooming*”. Setelah itu, akan muncul form pesanan yang wajib diisi pengguna. Form tersebut berisikan informasi pemesanan *grooming*. Lalu informasi tersebut tersimpan dalam *database*. Selanjutnya, pesanan aktif tersebut akan muncul di halaman utama pengguna.



Gambar 5. Diagram *activity groomer*

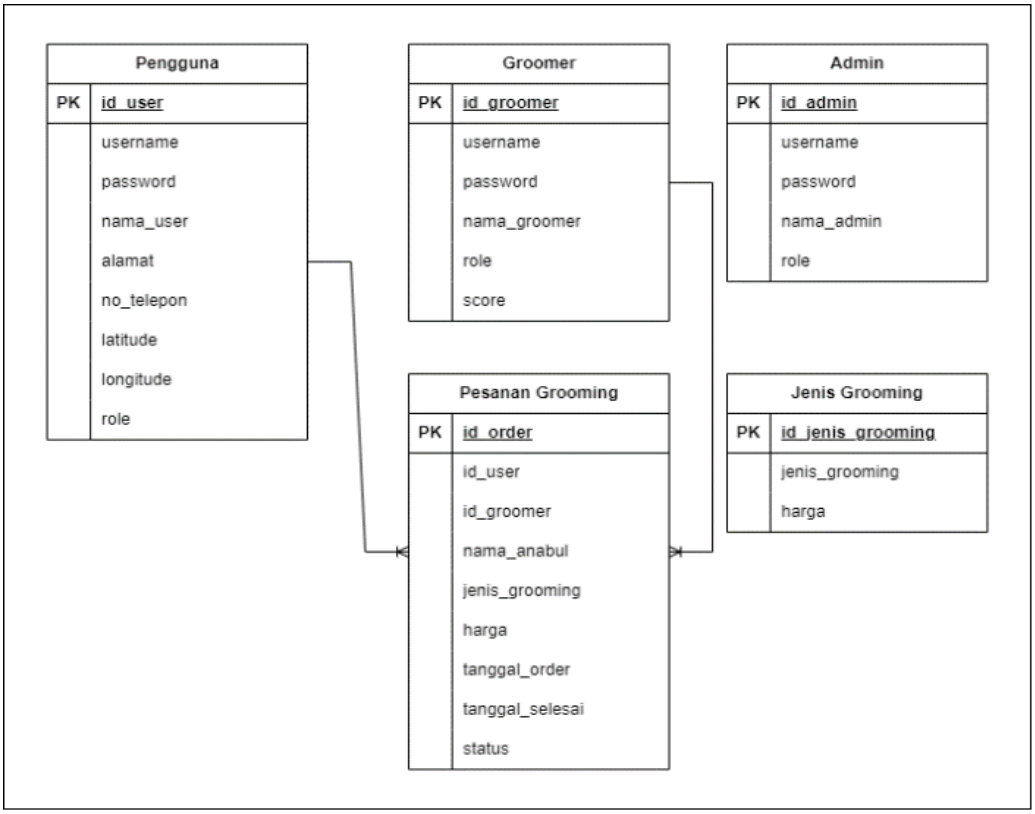
Gambar 5 menjelaskan tentang diagram *activity groomer* saat melakukan *update* status tiap pesanan. Pada diagram tersebut memperlihatkan proses interaksi *groomer* dengan sistem dan *database*. Dimulai dari *groomer* membuka aplikasi tersebut. *Groomer* akan diarahkan langsung ke halaman awal yaitu halaman *login*. Setelah itu, *groomer* harus memasukkan *username* dan *password* untuk dapat akses masuk ke dalam aplikasi. Setelah berhasil masuk, akan diarahkan ke halaman selanjutnya yaitu halaman utama dari sisi groomer. Halaman utama tersebut berisi daftar pesanan. Jika ingin *update* status pesanan yang sedang dikerjakan, *groomer* harus melakukan klik pada tombol yang tersedia pada tiap daftar pesanan dan akan terupdate pada database. Setelah itu, sistem akan menampilkan daftar pesanan yang sudah terupdate statusnya.



Gambar 6. *Class Diagram*

Gambar 6 menjelaskan tentang *class diagram* yang dibuat untuk menggambarkan sistem aplikasi grooming dengan tiga aktor utama yaitu User, Admin, dan *Groomer*. Diagram ini terdiri dari tiga kategori utama yaitu entitas, *controller*, dan *boundary*. Secara keseluruhan, diagram ini menunjukkan bagaimana setiap komponen dalam sistem saling berinteraksi dan berkontribusi terhadap fungsi keseluruhan aplikasi grooming.

Selanjutnya Gambar 7 menjelaskan tentang tabel dari desain database aplikasi. Desain database yang disusun adalah kerangka yang terstruktur untuk menyimpan data dalam aplikasi *grooming* dengan tiga peran utama yaitu User, Admin, dan *Groomer*. Terdapat lima tabel utama dalam desain ini. Tabel pengguna menyimpan informasi tentang pengguna. Tabel pesanan menyimpan detail tentang setiap pesanan, termasuk informasi pengguna yang melakukan pesanan, *groomer* yang ditugaskan, tanggal pesanan, status pesanan, dan total harga. Tabel *groomer* dan admin mencatat informasi tentang setiap *groomer* dan admin. Tabel jenis *grooming* menyimpan informasi tentang layanan yang tersedia. Desain ini disusun agar sistem dapat melacak dan mengelola pesanan, informasi pengguna, informasi groomer, serta riwayat perubahan status pesanan dengan efisien.



Gambar 7 Entity Relationship Diagram

B. Implementasi Sistem

Kode program 1 berisi tentang dependensi yang diperlukan untuk dapat Menggunakan GSON dan Retrofit dalam sebuah proyek android. Ada tiga dependensi yaitu retrofit, gson, dan gson converter. Retrofit sendiri merupakan *library* yang akan memudahkan untuk berkomunikasi dengan API *web*.

Kode Program 1. Kode library retrofit

```
implementation 'com.squareup.retrofit2:retrofit:2.1.0'
implementation 'com.google.code.gson:gson:2.6.2'
implementation 'com.squareup.retrofit2:converter-gson:2.1.0'
```

Gson adalah *library* Java yang digunakan untuk mengonversi objek Java menjadi representasi JSON dan sebaliknya. Gson Converter berguna agar Retrofit dapat menggunakan Gson untuk mengonversi JSON yang diterima dari server menjadi objek Java dan sebaliknya. Dengan menggunakan Gson Converter.

Kode Program 2. Kode pengambilan data

```
Call<JsonElement> call = apiInterface.listOrderGrooming();
    call.enqueue(new Callback<JsonElement>() {
        @Override
        public void onResponse(Call<JsonElement> call, Response<JsonElement>
                response) {
            JsonElement jsonElement = response.body();
            JsonObject jObj = jsonElement.getAsJsonObject();
            boolean success = jObj.get("success").getAsBoolean();
```

```
            if(success)
            {
                JsonArray order_grooming=jObj.get("order_grooming").getAsJsonArray();
                listOrderGrooming  = new ArrayList<>();
                for (int i = 0; i < order_grooming.size(); i++) {
                    JsonObject jsonObject = order_grooming.get(i).getAsJsonObject();
                    OrderGrooming orderGrooming = new OrderGrooming
                            ();
                    listOrderGrooming .add(orderGrooming);
                }
                orderGrooming Adapter.updateDataList(listOrderGrooming );
            }
            else {final String message = jObj.get("message").getAsString();}
        }
        public void onFailure(Call<JsonElement> call, Throwable t) {}
```

Kode program 2 digunakan untuk memanggil metode yang sudah didefinisikan sebelumnya di *apiInterface*. Panggilan ini akan mengirimkan permintaan HTTP ke server untuk mendapatkan daftar pesanan *grooming*. Setelah itu, *JsonElemet* dikonversikan menjadi *JsonObject*. Jika nilainya *true* maka akan diambil *array* JSON dengan kunci "order_grooming", kemudian mengonversi setiap elemen dalam *array* menjadi objek *OrderGrooming* dan menambahkannya ke dalam *listOrderGrooming* dan untuk memperbarui tampilan gunakan dengan memanggil *orderGroomingAdapter. updateDataList().*

Kode Program 3. Kode penambahan data

```
Call<JsonElement> postOrderGrooming = apiInterface.tambahOrderGrooming(parameter);
postOrderGrooming.enqueue(new Callback<JsonElement>() {
    @Override
    public void onResponse(Call<JsonElement> call, Response<JsonElement> response) {
        JsonElement jsonElement = response.body();
        JsonObject jObj = jsonElement.getAsJsonObject();
        boolean success = jObj.get("success").getAsBoolean();
        final String message = jObj.get("message").getAsString();
        if (success) {
            homeFragment.dialogTambah.dismiss();
            bindOrderGrooming();
        } else {}
    }
    @Override
    public void onFailure(Call<JsonElement> call, Throwable t) {
    }
});
```

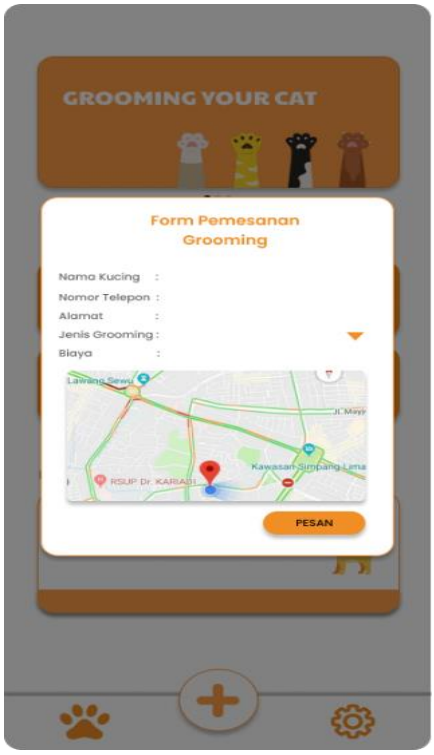
Kode program 3 digunakan untuk mengirimkan data ke server dengan memanggil metode yang sudah didefinisikan sebelumnya di *apiInterface* bersama dengan parameter yang dibutuhkan. Nilai dari parameter tersebut didapatkan dari *input* data yang dilakukan oleh pengguna aplikasi. Setelah itu, *JsonElemet* dikonversikan menjadi *JsonObject*. Jika permintaan berhasil atau bernilai *true* maka data berhasil masuk server dan tampilan akan diperbarui dengan memanggil metode *bindOrderGrooming().*



Gambar 8 Halaman utama pengguna

Gambar 8 menampilkan halaman utama dari sisi pengguna. Pada halaman utama ini dirancang dengan menarik agar pengguna tertarik untuk mempercayakan kucing pengguna untuk melakukan *grooming* di *petgroomer*. Mencakup manfaat dari *grooming* agar pengguna tertarik untuk melakukan pemesanan *grooming*. Pemesanan *grooming* dapat dilakukan oleh pengguna dengan menekan tombol tambah pada navigasi bawah dari aplikasi tersebut.

Selanjutnya Gambar 9 menampilkan form pemesanan *grooming*. Dimana pengguna diminta untuk mengisi form dengan lengkap agar bisa melakukan pemesanan *grooming*. Detail informasi yang harus diisikan seperti nama kucing, nomor telepon pemilik, alamat, dan jenis *grooming* yang akan dipesan. Desain dari form pemesanan ini dirancang dengan baik agar pengguna tidak kesulitan dalam mengisikan form tersebut.



Gambar 9 Form pemesanan

Gambar 10, 11, 12 menampilkan halaman utama dari sisi admin dan *groomer*. Pada halaman ini dibagi menjadi tiga menu, yaitu menu *paw in*, diproses, dan *paw out*. Pada menu *paw in* berisi daftar pesanan *grooming* yang masuk.



Gambar 10. Menu *paw in*



Gambar 11. Menu diproses



Gambar 12. Menu *paw out*



Gambar 13 Halaman rekapitulasi

Gambar 13 menampilkan halaman rekapitulasi di sisi pengguna admin. Pada halaman ini berisikan pencatatan pemesanan *grooming*. Detail pencatatan tersebut berisikan informasi tentang berapa banyak pesanan *grooming* yang sudah selesai dilakukan, apa saja jenis *grooming* yang telah diselesaikan, dan pemasukan yang masuk. Pada halaman ini terdapat dua filter untuk melihat pencatatan tersebut yaitu filter untuk bulan dan karyawan. Filter tersebut akan memudahkan admin untuk melihat pencatatan tiap bulan dan juga bisa melihat kinerja tiap *groomer*.

B. Pengujian Sistem

Pengujian sistem dilakukan untuk melihat apakah fungsi-fungsi untuk memastikan bahwa fungsi-fungsi utama sistem berjalan dengan baik. Dari 7 kasus pengujian yang dilakukan, semuanya dapat diselesaikan oleh aplikasi dengan baik, sehingga dengan demikian dapat dikatakan bahwa aplikasi berjalan sesuai dengan yang diharapkan.

Tabel 1 Tabel pengujian

Skenario pengujian	Hasil yang Diharapkan	Hasil Pengujian	Kesimpulan
Admin melakukan login	Admin masuk ke halaman utama admin	Admin berhasil login dan diarahkan ke halaman utama admin	Sukses
Pengguna melakukan login	Pengguna masuk ke halaman utama pengguna	Pengguna berhasil login dan diarahkan ke halaman utama pengguna	Sukses

Groomer melakukan login	*Groomer* masuk ke halaman utama *groomer*	*Groomer* berhasil login dan diarahkan ke halaman utama *groomer*	Sukses
Admin menekan tombol tambah pesanan	Ditampilkan form pesanan dan diisi oleh admin	Form pesanan muncul dan dapat diisi oleh admin	Sukses
Admin menekan tombol peta pada form pesanan	Ditampilkan Maps API dan admin memilih titik lokasi pada maps	Maps muncul dan admin berhasil menentukan titik *latitude* dan *longitude*	Sukses
Pengguna menekan tombol tambah pesanan	Ditampilkan form pesanan dan diisi oleh pengguna	Form pesanan muncul dan dapat diisi oleh pengguna	Sukses
Pengguna menekan tombol peta pada form pesanan	Ditampilkan Maps API dan pengguna memilih titik lokasi pada maps	Maps muncul dan admin berhasil menentukan titik *latitude* dan *longitude*	Sukses

V. SIMPULAN DAN SARAN

Penelitian ini dilakukan untuk melakukan pengembangan aplikasi monitoring *grooming* berbasis *mobile* yang dirancang menggunakan implementasi REST API yaitu Retrofit dan menggunakan bahasa pemrograman java. Metode penelitian yang dilakukan memiliki tiga langkah yaitu, komunikasi dengan klien, pengembangan aplikasi, dan pengajian. Aplikasi ini dirancang untuk meningkatkan efisiensi pemesanan *grooming* dan transparansi *grooming* secara *real-time* agar memberikan pengalaman penggunaan aplikasi yang baik bagi pengguna. Hasil dari penelitian ini dapat dijadikan untuk memudahkan pengguna maupun *petgrommer* dalam meningkatkan efisiensi pemesanan secara modern. Pengembangan aplikasi ini untuk kedepannya diharapkan untuk lebih dikembangkan dan ditingkatkan secara keseluruhan baik dari segi tampilan maupun fungsionalitas. Sebagai saran, dapat ditambahkan fitur kebutuhan peliharaan lain seperti *e-commerce* untuk *petshop,* dengan adanya hal tersebut pengguna akan dimudahkan dan mendapatkan pengalaman yang lebih baik dalam menggunakan aplikasi.

DAFTAR PUSTAKA

[1] Salma Yunita, A., & Tjandrawibawa, P. (2023). *Perancangan Brand Campaign Pada Instagram Guna Meningkatkan Brand Awareness Terhadap Omnifurr* (Vol. 13, Issue 2).

[2] Uktolseja, M. I., & Sukada, B. A. (2020). Ruang Bersama Unutk Manusia Dan Anjing. *Jurnal Sains, Teknologi, Urban, Perancangan, Arsitektur (Stupa)*, *2*(1), 317. https://doi.org/10.24912/stupa.v2i1.6770

[3] Ningsih, N. A., & Abidin, M. R. (2021). Perancangan Design User Interface Website Pada Pet Shop Azria Di Kabupaten Lamongan. *Jurnal Barik, 2*(3), 202–216. https://ejournal.unesa.ac.id/index.php/JDKV/

[4] Hartiwati, E. N. (2022). Aplikasi Inventori Barang Menggunakan Java Dengan Phpmyadmin. *Cross-Border*, *5*(1), 601–610.

[5] Laksono, W. P., Satria, B., Wicaksana, T., Wijasena, A. Y., & Sahria, Y. (2024). Implementasi Clean Architecture Dalam Membangun Aplikasi Mobile Menggunakan Flutter. *Nusantara Journal of Multidisciplinary Science*, *2*(1), 173–180. https://jurnal.intekom.id/index.php/njms

[6] Yudhistira, Y. (2021). Implementasi Application Programming Interface (API) Kawal Corona Sebagai Media Informasi Pandemik Covid-19 Berbasis Android. In *Jurnal Sistem Informasi dan Teknologi Peradaban (JSITP)* (Vol. 2, Issue 1). www.journal.peradaban.ac.id

[7] Nugroho, A., Edwar, Indrianto, Gde, A. A., Putra, A., Asmara, R., & Stikom Bali, I. (2020). *Sistem Informasi Bengkel Online Memanfaatkan GPS (GMAPS) Secara Real-Time Dan Back Office Berbasis Web.*

[8] Wismarini, T. D., & Prihandono, A. (2020). Rancang Bangun Aplikasi Android Terintegrasi Web Service Dengan Volley Untuk Layanan Publik. Jurnal DINAMIK, 25(1), 10–19.

[9] Christhover, R., Sugiyatno, & Herlawati. (2022). Penerapan Rest API Menggunakan Retrofit Untuk Sistem Informasi Film Berbasis Android (Studi Kasus: Sinopsis Film). *Journal of Students' Research in Computer Science*, *3*(2), 159–170. https://doi.org/10.31599/jsrcs.v3i2.1393

[10] Lumba, E. (2021). *Pertukaran Data Pada Aplikasi Android Menggunakan Java Script Object Notation (JSON) Dan Rest API DENGAN RETROFIT 2*. https://websindo.com/

[11] Gede, I., Ariantara, M., Arwani, I., Hayuhardhika, W., & Putra, N. (2020). *Penerapan REST API dalam Pengembangan Aplikasi Pemesanan Rental Mobil berbasis Web dan Mobile (Studi Kasus: CV. Dwi Cipta Rent Car)* (Vol. 4, Issue 8). http://j-ptiik.ub.ac.id

[12] Alda, M. (2023). Pengembangan Aplikasi Pengolahan Data Siswa Berbasis Android Menggunakan Metode Prototyping. *Jurnal Manajemen Informatika (JAMIKA)*, *13*(1), 11–23. https://doi.org/10.34010/jamika.v13i1.8216

[13] Dwi Kartika, M., & Priyadi, Y. (2020). *Pengembangan Sistem Penjualan Menggunakan UML Dan Proses Bisnis E-Commerce Pada TB. Purnama Banjarnegara*. *7*(3), 480–497. http://jurnal.mdp.ac.id

[14] Aditya, R., Handrianus Pranatawijaya, V., Bagus Adidyana Anugrah Putra, P., Hendrik Timang, J., Palangkaraya, K., & Tengah, K. (2021). Rancang Bangun Aplikasi Monitoring Kegiatan Menggunakan Metode Prototype. In *JOINTECOMS (Journal of Information Technology and Computer Science) p-ISSN: xxxx-xxxx* (Vol. 1, Issue 1).

[15] Sari, R., & Hamidy, F. (2021). Sistem Informasi Akuntasi Perhitungan Harga Pokok Produksi Pada Konveksi SJM Bandar Lampung. *Jurnal Teknologi Dan Sistem Informasi (JTSI)*, *2*(1), 65–73. http://jim.teknokrat.ac.id/index.php/JTSI