JUSTEK : JURNAL SAINS DAN TEKNOLOGI
http://journal.ummat.ac.id/index.php/justek
ISSN 2620-5475
Vol. 5, No. 2, November 2022, Hal. 304-313

Android-Based Mobile Device Application Development in Vehicle Maintenance System

[1]Yosua Alvin Adi Soetrisno, [2]Eko Handoyo, [3]Enda Wista Sinuraya, [4]Bambang Winardi, [5]Imam Santoso, [6]Zwingli Hilkia Batubara
[1.2.3.4.5,6] Departemen Teknik Elektro, Universitas Diponegoro, Indonesia
yosua@live.undip.ac.id, eko_handoyo@elektro.undip.ac.id, sinuraya_enda@elektro.undip.ac.id, bbwinar@gmail.com, imamsantoso@lecturer.undip.ac.id, zwingli99@students.undip.ac.id

ARTICLE INFO

Article History:

Diterima : 08-11-2022
Disetujui : 16-11-2022

Keywords:
vehicle maintenance;
Android application;
prototyping

ABSTRACT

Abstract: *In today's rapid development, application technology has begun to be widely used in business transactions. The use of computers on mobile devices such as mobile phones is no longer every day; it is even used to order daily necessities through the marketplace. The existence of various types of applications makes us sometimes become unfocused. The research here aims to create a vehicle maintenance application that helps remind users, although many other applications exist. The built-in reminder feature relates to the vehicle maintenance date based on car usage, car age, and the last maintenance date recorded on the application. The research method used is prototyping. Figma adjusted user expectations to current programming needs in developing the initial application design. After the user's expectations are in line, use Android Studio as the Integrated Development Environment for full application development. The results of this study are the functions of the application can be used properly and can remind users to service their vehicles properly.*

Abstrak: Pada perkembangan zaman yang pesat saat ini, teknologi aplikasi mulai banyak digunakan di dalam transaksi bisnis. Penggunaan komputer pada perangkat bergerak seperti handphone sudah tidak lazim dilakukan bahkan digunakan untuk memesan barang kebutuhan sehari-hari melalui marketplace. Adanya berbagai jenis aplikasi membuat kita kadang menjadi tidak fokus. Tujuan penelitian yang dilakukan di sini adalah mencoba membuat aplikasi pemeliharaan kendaraan yang membantu mengingatkan pengguna di tengah banyaknya aplikasi yang lain. Fitur pengingat yang dibangun berkaitan tentang tanggal perawatan kendaraan, berdasarkan penggunaan mobil, usia mobil, dan tanggal perawatan terakhir yang terekam pada aplikasi. Metode penelitian yang dilakukan adalah dengan menggunakan prototyping. Dalam mengembangkan rancangan aplikasi di awal maka digunakan Figma agar bisa menyesuaikan ekspektasi pengguna dengan kebutuhan pemrograman yang ada. Setelah ekspektasi pengguna sesuai maka untuk pengembangan aplikasi menggunakan Android Studio sebagai Integrated Development Environment. Hasil dari penelitian ini adalah fungsi-fungsi dari aplikasi dapat digunakan dengan baik dan dapat mengingatkan penggunanya untuk melakukan servis kendaraan dengan baik.

https://doi.org/10.31764/justek.vXiY.ZZZ

A. LATAR BELAKANG

Banyak aplikasi yang terdapat pada *handphone* yang berkembang saat ini yang pastinya membantu kegiatan manusia sehari – hari. Mulai dari aplikasi pengingat, *online shopping*, maupun aplikasi yang membantu manusia secara spesifik seperti aplikasi *maintenance* kendaraan misalnya.

Dapat disadari pula bahwa jumlah kendaraan bermotor terutama mobil di Indonesia semakin padat, namun hal ini terkadang tidak berbanding lurus dengan kesadaran pengendara mengenai kondisi kendaraannya. Orang – orang yang saat ini kurang peduli dengan perawatan kendaraannya secara tidak langsung bisa berdampak pada keselamatan penggunanya. Kejadian yang bisa terjadi seperti seperti *accumulator* rusak, oli kurang melindungi mesin, kopling macet, kendala atau mogok di tengah jalan, dan bahkan lebih fatal seperti terjadinya kecelakaan.

Beberapa peneliti melakukan *monitoring* kendaraan ini dengan mengembangan beberapa perangkat tambahan seperti *Internet of Things* (*IoT*) (Srikanth et al., 2021). Selain itu juga digunakan machine learning untuk melakukan prediksi kapan selanjutnya kendaraan tersebut membutuhkan servis. Selain itu ada juga yang menggunakan *IoT* untuk memonitor kondisi truk berdasarkan beban yang dibawa sehingga bila butuh perawatan akan diberi notifikasi (Mohammad et al., 2021). Ada juga yang menggunakan modul *General Packet Radio Service* (*GPRS*) sehinga bisa memonitor beberapa metrik sensor gas pada kendaraan (Subramanyam & Jyothi, 2017).

Selain mengembangkan alat tambahan maka ada juga yang melakukan penelitian mengenai pembelajaran ke siswa tentang bagaimana struktur dari mesin yang ada di kendaraan (Premono et al., 2021). Secara lebih spesifik ada juga yang menjelaskan penggunaan oli dengan bantuan IoT (Nugroho et al., 2021). Hal ini menunjukkan bahwa pembelajaran tentang mesin bila diajarkan dengan detail dapat membantu proses pembelajaran tentang bagian mana yang perlu untuk diservis. Hal ini membantu pengguna dari sisi pengetahuan mesin agar bila terjadi kerusakan juga pengguna bisa tahu dari bagian yang sebelah mana. Pada penelitian kali ini belum dilengkapi dengan video tentang bagian-bagian tertentu dari mesin yang butuh diservis namun ke depan bisa menjadi suatu masukan yang menarik.

Selain menggunakan *sensor IoT* maka ada juga yang secara spesifik mengembangkan aplikasi *Android* untuk me-*monitoring* hal-hal apa saja yang perlu dipantau dalam perawatan kendaraan (Kotsyuba et al., 2022). Hal ini menunjukkan bahwa selain menggunakan *IoT* maka bisa juga langsung dikembangkan dalam bentuk aplikasi yang juga menuntut pengguna untuk mengetahui hal-hal apa saja yang perlu di-*monitor*.

Tujuan penelitian yang dikembangkan adalah bagaimana bisa melakukan *prototyping* dengan baik, sehingga aplikasi bisa mengingatkan pengguna dengan tepat sesuai dengan kriteria kendaraannya. Hal ini dimulai dari peracangan menggunakan *Figma* dan memvalidasinya dengan pengguna. Data kendaraan akan disimpan dan mengingatkan pengguna akan jadwal *maintenance* kendaraan berdasarkan waktu terakhir *maintenance* sebagai acuan, sehingga waktu *maintenance* dapat diatur dengan lebih mudah oleh pengguna. Aplikasi ini diharapkan dapat meminimalisir adanya kendaraan yang kurang diservis dan mengakibatkan dampak negatif baik ke pengendara maupun pengendara lain.

B. METODE PENELITIAN

1. Pengembangan Aplikasi *Mobile*

Proses membangun aplikasi melibatkan banyak hal yang perlu dipertimbangkan. Selain itu, desain dan fungsionalitas memainkan peran penting dalam pertumbuhan bisnis pada bidang pengembangan aplikasi perangkat bergerak ini. Pengembangan aplikasi seluler secara umum dibagi menjadi beberapa fase yang berbeda, yang berkontribusi untuk membangun aplikasi yang baik (Sandy et al., 2019).

2. Android

Pengembangan *Android* menggunakan bahasa *Kotlin* atau *Java*. Kesulitan dalam mengembangkan *Android* adalah bahwa komponen-komponen dan juga interaksi di *Android* harus dikembangkan di dalam kelas tertentu yang harus di-*extend* dari *interface* atau abstraksi interaksi tertentu. Ketika melakukan extend dari kelas maka harus betul-betul diketahui objek seperti apa yang harus dicetak dan juga membutuhkan struktur input dan instansiasi tertentu (Berkati, 2021).

3. Figma

Figma adalah aplikasi web kolaboratif untuk desain antarmuka, dengan fitur yang juga dilengkapi untuk aplikasi desktop. Kumpulan fitur *Figma* bisa membantu pengembang untuk merancang tampilan fitur aplikasi berdasarkan fitur yang diinginkan pengguna tanpa harus membuat fungsi secara lengkap. Aplikasi seluler *Figma* untuk *Android* dan *iOS* memungkinkan melihat dan berinteraksi dengan prototipe *Figma* secara *real-time* di perangkat seluler dan tablet. (Rully Pramudita et al., 2021).

4. Android Studio

Android merupakan pengembangkan dari *Eclipse IDE* yang lebih lengkap dan didukung oleh *library gradle* untuk mengatur *resource* dan membantu mencari *dependency*. *Android* Studio merupakan *IDE* yang digunakan untuk develop *Android* sekarang ini. Ada pilihan dua bahasa yaitu *Java* ataupun *Kotlin*.

5. Firebase

Firebase adalah platform berbasis *cloud* yang dikembangkan oleh *Google* untuk bisa mengirim data dan juga menggunakan servis notifikasi. *Firebase* memiliki beberapa fitur seperti : *Authentication*, *Firestore Database*, *Realtime Database*, *Storage*, dan lain-lain. Pengembang *Android* dapat menyambungkan *project* pada *Android Studio* ke *Firebase*. Struktur data pada *Firebase* berupa *array* yang menunjukkan struktur dari entitas yang dipakai pada aplikasi (Periyanayagi et al., 2021).

6. CameraX

Di dalam pengembangan menggunakan *CameraX* maka perilaku kamera harus dipertahankan supaya konsisten. Pengembang harus mempertimbangkan rasio lebar tinggi, rotasi, orientasi, ukuran gambar, dan ukuran pratinjau. Perilaku dasar ini harus bisa difungsikan. *CameraX* menguji berbagai perilaku kamera mulai dari versi sistem operasi sejak *Android* 5.0. Pengujian ini dijalankan secara berkelanjutan untuk mengidentifikasi dan memperbaiki beragam masalah (Lviv Polytechnic National University et al., 2021).

7. GitHub

GitHub adalah platform yang bisa digunakan untuk berkolaborasi di dalam menjalankan suatu siklus *development* ketika dibutuhkan *maintenance* dan penyatuan kode. *GitHub* mendukung fasilitas manajemen projek, mengelola versi dari kode dan juga kebutuhan dependency pengembangannya, dan hal ini digunakan untuk sharing hal yang bermanfaat agar bisa dikembangkan orang lain secara lebih besar (Yavorskiy, 2020).

8. Unified Modelling Language (UML)

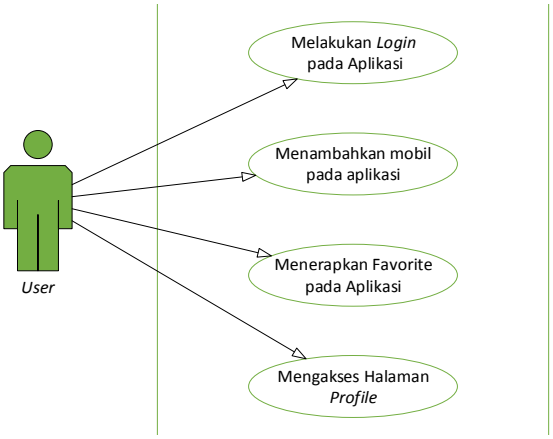
UML merupakan suatu cara dalam menunjukkan proses bisnis yang terjadi pada suatu aplikasi dengan mewakilinya dalam suatu aktor dan juga fungsional seperti apa saja yang bisa dilakukan. Biasanya *UML* digunakan dalam pengembangan aplikasi berorientasi objek. *UML* bisa digunakan untuk melakukan visualisasi, merancang dokumentasi di level fitur program dan juga menjadi *blueprint* dalam pengembangan *software* (Purwanti & Prastio, 2022).

Use Case Diagram merupakan diagram yang bisa menunjukkan hubungan antara aktor dengan sistem. *Use Case* bisa menjelaskan interkasi seperti apakah yang terjadi pada program. Langkah awal untuk melakukan pemodelan adalah dengan mampu membagi tugas aktor secara fungsional menjadi tugas-tugas kecil.

Activity diagram juga suatu diagram yang bisa menjelaskan lebih rinci fungsi yang ada pada *use case diagram*. Pada *activity diagram* akan digambarkan proses dari awal sampai akhir dan juga bila proses melibatkan dan menunggu respon dari sistem maka akan ada proses di antara dua *swimlane* yaitu aktor dan sistem (Savero, 2022).

C. HASIL DAN PEMBAHASAN

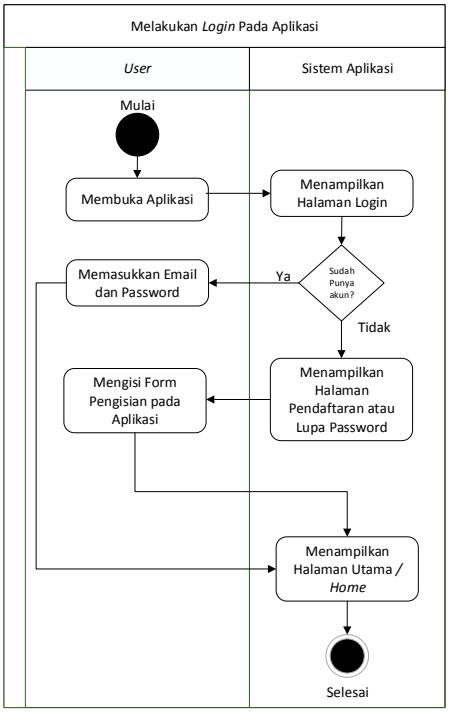
Use case diagram disini dirancang untuk mengetahui pada aplikasi pemeliharaan kendaraan ada fungsi apa saja yang bisa dilakukan penggunanya. Gambar 4 yang menunjukkan *use case diagram* dari aplikasi pemeliharan kendaraan.



Gambar 1. *Use Case Diagram* Aplikasi (Pemeliharaan Kendaraan)

Dapat terlihat pada *use case* diatas, terdapat entitas yang terlibat adalah *user* atau pengguna aplikasi. Dimana pengguna dapat mengakses aplikasi untuk *login* jika sudah memiliki akun, melakukan pendaftaran akun, mendaftarkan dan mengisi informasi mobil, menerapkan mobil *favorite*, serta mengakses halaman *profile* untuk mengganti bahasa, *password*, ataupun melakukan *log out*.

Activity diagram bisa menggambarkan mulai dari awal aplikasi pemeliharaan kendaraan ini harus diakses melalui menu mana saja dan juga bagaimana cara untuk mengaktifkan notifikasi *maintenance*-nya. Gambar 2 yang menunjukkan *Activity Diagram* saat *user* melakukan *login* pada aplikasi.



Gambar 2. *Activity Diagram User* Melakukan *Login* pada Aplikasi

Dalam melakukan *login, user* dapat memasukkan email dan password mereka jika sudah melakukan pendaftaran sebelumnya, namun jika belum maka *user* dapat terlebih dahulu mengakses halaman "Daftar Akun" jika belum memiliki akun atau halaman "Lupa Password" jika *user* lupa *password* dari akun mereka. Setelah berhasil melakukan pendaftaraan atau mendapat *password* baru, maka user dapat kembali ke halaman *login* untuk melakukan *login* dan user dapat masuk ke halaman utama aplikasi setelah berhasil melakukan *login*.

Dalam menambahkan mobil, terlebih dahulu *user* masuk ke halaman utama aplikasi, lalu mengakses halaman "*List Mobil"*. Pada halaman tersebut *user* dapat menambahkan mobil mereka serta memberikan informasi dari mobil mereka. Setelah informasi ditambahkan, maka mobil *user* akan terdaftarkan dan muncul pada halaman utama dan halaman "*List Mobil*".

Dalam menerapkan mobil *favorite*, terlebih dahulu *user* masuk ke halaman utama aplikasi, lalu mengakses halaman "*List Mobil"*. Pada halaman tersebut user dapat mengganti mobil *favorite* mereka jika mobil yang didaftarkan lebih dari satu, sedangkan jika hanya terdapat satu mobil saja maka otomatis mobil tersebut diterapkan menjadi mobil *favorite user*.

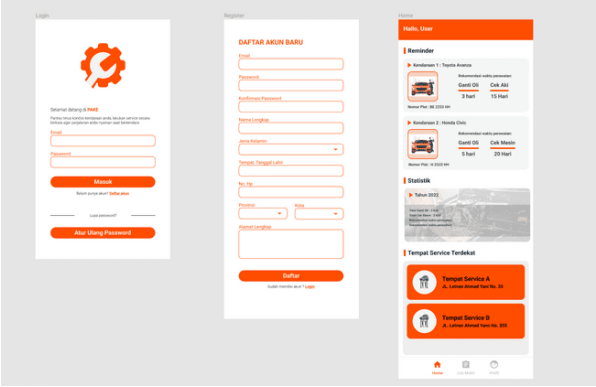
Dalam mengakses halaman *Profile*, terlebih dahulu *user* masuk ke halaman utama aplikasi, lalu mengakses halaman "*Profile"*. Pada halaman tersebut *user* dapat memilih beberapa pilihan pengaturan aplikasi yang tersedia yaitu "Ganti *Password*", "Ganti

Bahasa", "Laporkan Masalah", dan "*Log Out*". *User* dapat memilih pengaturan aplikasi tersebut sesuai kebutuhan.

1. Perancangan desain *UX* Aplikasi

Software Figma dapat diakses langsung melalui *website* secara gratis ataupun mendownload terlebih dahulu untuk versi desktop. Langkah awal dalam merancang design *UX* aplikasi dapat dengan membuat terlebih dahulu *wireframe* sebagai kerangka awal dalam perancangan *design UX* pada aplikasi pemeliharaan kendaraan ini.

Wireframe dibuat terlebih dahulu sebagai wadah awal sebelum nantinya desain *UX* masuk ke tahap *Hi-Fi* (*High Fidelity*) untuk diberikan warna, gambar dan *icon* atau logo. Setelah seluruh halaman *wireframe* selesai, maka dapat dilanjutkan kepada tahap Hi-Fi serta membuat logo dari aplikasi. Beberapa tampilan *Hi-Fi* dan logo pada desain *UX* aplikasi dapat dilihat pada Gambar 3 berikut ini.



Gambar 3. Tampilan Beberapa *Hi-Fi* Aplikasi pada *Figma*

Setelah design *UX* selesai pada tahap *Hi-Fi*, maka design *UX* sudah dapat diimplementasikan kedalam versi android. Sebelum memulai melakukan *deployment* ke *Android* dengan *Android Studio*, penulis membuat terlebih dahulu *stickersheet* pada figma sebagai informasi pada rancangan desain seperti kode warna, ukuran tulisan, *typography*, *button*, serta *icon* yang digunakan untuk memudahkan pengambilan informasi sewaktu membuat *layout* aplikasi pada *Android Studio*.

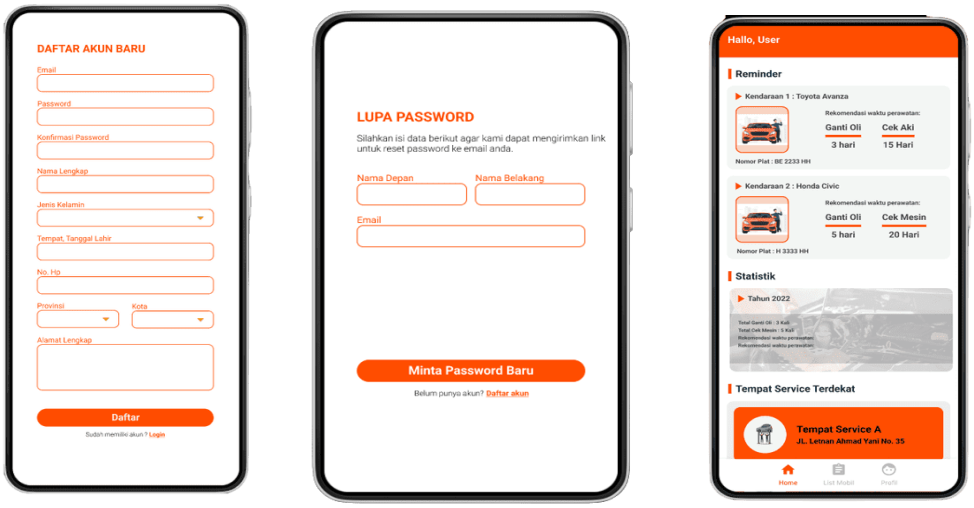
2. Membuat SplashScreen Aplikasi

SplashScreen aplikasi merupakan halaman animasi berupa logo sesaat ketika pengguna membuka aplikasi. *SplashScreen* sendiri akan menjadi ciri khas suatu aplikasi seperti logo yang menjadi animasi pembuka dari aplikasi sebelum masuk ke aplikasi. Pada tahap pembuatan *SplashScreen* terlebih dahulu mempersiapkan logo dan warna i yang akan ditunjukkan pada *SplashScreen*. Setelah logo ditentukan, maka logo tersebut dapat di-*import* terlebih dahulu dengan format *SVG*. Untuk memudahkan pembuatan *SplashScreen*, dapat digunakan fitur *API* yang disediakan *Android* dengan versi 12 atau lebih.

3. Membuat Halaman Login Aplikasi

Halaman *login* pada aplikasi pemeliharaan kendaraan merupakan halaman kedua setelah tampilan *splashscreen* selesai diluncurkan saat aplikasi dibuka. Pada halaman ini terdapat logo dari pemeliharaan kendaraan yang disertai dengan pesan berupa sambutan dan kegunaan pemeliharaan kendaraan secara ringkas. Juga terdapat *Email* dan *password* sebagai proses *autentikasi* pada aplikasi. Juga terdapat dua *button* yaitu "masuk" untuk

masuk kedalam aplikasi dan *button* "Atur Ulang *Password*" untuk mengantarkan pengguna ke halaman Lupa *Password* jika dibutuhkan. Selain itu terdapat juga *hyperlink* pada kalimat "Daftar akun" yang dapat mengantarkan pengguna ke halaman pendaftaran akun. Pada *layout* halaman *login* menggunakan *ConstraintLayout* dalam pembuatan tampilan yang kompleks dikarenakan fleksibilitasnya. Gambar 4 berikut ini merupakan tampilan dari halaman pendaftaran dan lupa *password*.



Gambar 4. Halaman pendaftaran, Lupa *Password* dan *Home* pada aplikasi pemeliharaan kendaraan

2. Membuat Halaman Home Aplikasi

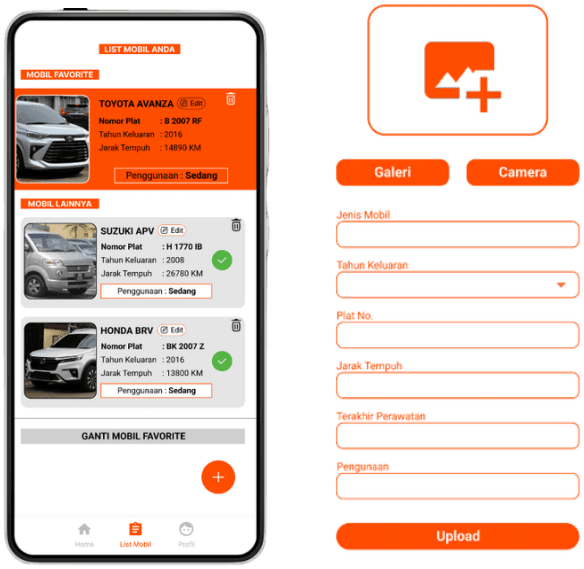
Setelah melakukan proses autentikasi pada halaman login sebelumnya, maka aplikasi akan beralih *intent* memasuki ke halaman utama. Pada halaman utama aplikasi pemeliharaan kendaraan terdapat beberapa fitur yaitu reminder untuk maintance mobil, statistik maintance pengguna selama menggunakan aplikasi, tampilan berupa list untuk tempat *service* terdekat serta terdapat *Navigation bar* pada bagian bawah halaman aplikasi.

Setelah setiap bagian fitur sesuai dengan design UX, pada bagian bawah halaman aplikasi terdapat *Navigation bar* yang berada di bawah pada Gambar 4 yang berfungsi sebagai *button* yang membawa pengguna ke halaman yang tersedia pada *Navigation bar* tersebut yaitu "*Home*", "list mobil" dan "profil" ketika pengguna menekan *icon* tersebut. *Navigation bar* ini juga nantinya akan digunakan kembali pada beberapa halaman lainnya, sehingga untuk *navigation bar* akan dibuat *fragment* untuk menampilkannya dan nantinya akan diimplementasikan di setiap halaman yang membutuhkannya.

3. Membuat Halaman *List* Mobil

Halaman list mobil merupakan halaman yang menampilkan daftar mobil yang ditambahkan oleh pengguna kedalam aplikasi. Terdapat fitur mobil *favorite* pada halaman ini sebagai tanda mobil mana yang menjadi perhatian utama aplikasi. Sehingga mobil yang ingin ditampilkan pengguna pada halaman utama harus terlebih dahulu diatur pada halaman ini menjadi mobil *favorite*. Juga terdapat *add button* yang berfungsi untuk mengganti *intent* ke halaman pendaftaran mobil sehingga pengguna dapat menambah

mobil mereka ke daftar mobil pada halaman ini. Gambar 5 berikut merupakan tampilan dari halaman *list* mobil pada aplikasi.



Gambar 5. Halaman *list* mobil pada aplikasi pemeliharaan kendaraan

Pada Gambar 5 terlihat halaman *list* mobil, yang juga dibuat terlebih dahulu *activity* dan *layout* berupa *xml* untuk halaman *list* mobil. Terlihat juga digunakan *image view* untuk menampilkan gambar serta *button* berupa *add button, delete button* berupa *icon* tempat sampah serta *button* "*Ganti Mobil Favorite*". Dengan menambahkan *explicit intent* pada *add button*, maka aplikasi akan berpindah *intent* ke halaman *add car* sehingga pengguna dapat menambahkan mobil.

4. Membuat Halaman *Add Car*

Halaman *add car* merupakan halaman berupa *explicit intent* yang berguna dalam menambah mobil pengguna dengan mengisi data yang dibutuhkan agar nantinya dapat ditampilkan pada list mobil. Tampilan *add car* sama halnya seperti halaman pendaftaran, dimana terdapat *edit box* untuk memasukkan data mobil serta terdapat *image view* bersama dengan dua button *camera* dan *galery* untuk mengupload gambar mobil pengguna. Pada halaman *add car* ini digunakan *CameraX* untuk memudahkan mengatur penggunaan kamera pada aplikasi.

Halaman *list* mobil merupakan halaman yang menampilkan daftar mobil yang ditambahkan oleh pengguna ke dalam aplikasi. Terdapat fitur mobil *favorite* pada halaman ini sebagai tanda mobil mana yang menjadi perhatian utama aplikasi. Sehingga mobil yang ingin ditampilkan pengguna pada halaman utama harus terlebih dahulu diatur pada halaman ini menjadi mobil *favorite*. Juga terdapat *add button* yang berfungsi untuk mengganti *intent* ke halaman pendaftaran mobil sehingga pengguna dapat menambah mobil mereka ke daftar.

5. Implementasi *CameraX* pada Aplikasi

Implementassi *CameraX* digunakan untuk memudahkan aplikasi dalam menggunakan fitur kamera dari android pengguna. Dimana dalam proses implementasinya, terlebih dahulu design *user experience* (*UX)* telah diimplementasikan seluruhnya ke aplikasi agar tidak terjadi kesalahan serta memudahkan dalam proses *debugging* aplikasi.

Setelah *library* berhasil di-*build*, maka hal selanjutnya adalah membuat *activity* dan *layout* baru *camera* serta menggunakan *PreviewView* untuk menampilkan objek gambar yang terlihat melalui kamera. Agar kamera dapat digunakan pada aplikasi, maka perlu menambahkan kode untuk meminta izin akses kamera kepada pengguna. Hal ini perlu dilakukan karena apabila tidak mendapat izin akses, maka akan menyebabkan aplikasi *force closed*.

6. Membuat Halaman profil Aplikasi

Halaman profil merupakan halaman yang memuat beberapa pengaturan untuk mengatur aplikasi ataupun akun pengguna, yaitu 'Ganti *Password'*, 'Ganti Bahasa', 'Laporkan Masalah', dan *'Log Out'*. Pengaturan ini masing – masing memiliki fungsinya, namun tidak semua pengaturan memiliki halaman tersendiri.

Sama hal-nya dengan halaman lain, terlebih dahulu untuk membuat halaman profil diperlukan *activity* dan *layout* berupa *.xml*. Pada file *.xml* untuk membuat *layout* halaman profil digunakan *ConstraintLayout* untuk memudahkan mengatur jarak antara *TextView.* Beberapa *TextView* digunakan sebagai pilihan pengguna yang diatur sebagai *hyperlink* yang memiliki fungsi yang sama dengan *button* untuk memindahkan *intent* sesuai ke halaman fungsi yang dipilih.

D. SIMPULAN DAN SARAN

Kesimpulan yang diperoleh dari penelitian ini adalah bahwa dengan *prototyping* fungsi yang diharapkan bisa disampaikan dengan lebih mudah dengan rancangan pada *Figma*. Setelah ada kerangka dari *Figma* dan tambahan dari *UML* maka mengembangkan komponen yang ada di aplikasi *Android* akan lebih terarah. Hasil dari penelitian adalah fungsi pengingat servis bisa berjalan baik karena terdapat pemahaman yang baik tentang komponen, perlakuan karakteristik komponen, dan interaksi yang digunakan.

Saran dari penelitian ini adalah bahwa dalam *prototyping* perlu terjadi penyesuaian agar fungsi tersebut bisa dengan lebih mudah dipahami pengguna. Selain itu pengujian fungsi di *Android* bisa menggunakan skenario user testing yang sudah disediakan oleh *Android* agar bisa menguji fungsionalitas setiap fungsi.

REFERENSI

Berkati, A. (2021). *Merancang dan Membuat Aplikasi Hitung BMI dengan menggunakan Android Studio*. 10.

Kotsyuba, I., Themlyakov, K., Shikov, A., Galperin, M., & Shtennikov, D. (2022). Mobile application for vehicle operation management. *Transportation Research Procedia*, *63*, 746–752. https://doi.org/10.1016/j.trpro.2022.06.070

Lviv Polytechnic National University, Bielik, V., Morozov, Y., & Morozov, M. (2021). Sensors in Cyber-Physical Systems Based on Android Operating System. *Advances in Cyber-Physical Systems*, *6*(2), 83–89. https://doi.org/10.23939/acps2021.02.083

Mohammad, S., Masuri, M. A. A., Salim, S., & Razak, M. R. A. (2021). Development of IoT Based Logistic Vehicle Maintenance System. *2021 IEEE 17th International Colloquium on Signal Processing & Its Applications (CSPA)*, 127–132. https://doi.org/10.1109/CSPA52141.2021.9377290

Nugroho, H. S., Dewantoro, M. A., & Atmaja, D. E. (2021). Development of Android Based Application for Monitoring Engine Lubricant Condition in Four Wheeled Vehicle and Driving Behaviour Effects on Engine Lubricant Condition. *IOP Conference Series: Materials Science and Engineering*, *1068*(1), 012021. https://doi.org/10.1088/1757-899X/1068/1/012021

Periyanayagi, S., Manikandan, A., Muthukrishnan, M., & Ramakrishnan, M. (2021). BDoor App-Blood Donation Application using Android Studio. *Journal of Physics: Conference Series*, *1917*(1), 012018. https://doi.org/10.1088/1742-6596/1917/1/012018

Premono, A., Syaefudin, E. A., & Salam, N. I. (2021). *Development of a learning media for android-based smartphone to enhance student comprehension on the motorcycle maintenance subject*. 6.

Purwanti, P., & Prastio, M. E. (2022). Aplikasi Informasi Sekolah SMK Di Depok Berbasis Android Menggunakan Android Studio. *Jurnal Esensi Infokom : Jurnal Esensi Sistem Informasi dan Sistem Komputer*, *5*(2), 42–48. https://doi.org/10.55886/infokom.v5i2.281

Ramadhani, M., Susanto, A. A., Mustofa, F., & Tauda, V. S. (2022). Design and User Experience Evaluation of Bersii Android-based Mobile Application User Interface. *MATICS: Jurnal Ilmu Komputer Dan Teknologi Informasi (Journal of Computer Science and Information Technology)*, *14*(2), 41–49. https://doi.org/10.18860/mat.v14i2.16919

Rully Pramudita, Rita Wahyuni Arifin, Ari Nurul Alfian, Nadya Safitri, & Shilka Dina Anwariya. (2021). Penggunaan Aplikasi Figma Dalam Membangun Ui/Ux Yang Interaktif Pada Program Studi Teknik Informatika STMIK Tasikmalaya. *JURNAL BUANA PENGABDIAN*, *3*(1), 149–154. https://doi.org/10.36805/jurnalbuanapengabdian.v3i1.1542

Sandy, J. K., Kharisma, A. P., & Fanani, L. (2019). *Pengembangan Aplikasi Perangkat Bergerak Berbasis Android Untuk Layanan Masyarakat Dengan Metode Mobile-D (Studi Kasus: RSUD Ngudi Waluyo Wlingi Kabupaten Blitar)*. 8.

Savero, J. E. (2022). *Pengembangan Aplikasi Daftar Pasien Puskesmas Raya Berbasis Android Studio*. 12.

Srikanth, M. S., Kumar, T. G. K., & Sharma, V. (2021). Automatic Vehicle Service Monitoring and Tracking System Using IoT and Machine Learning. In A. P. Pandian, X. Fernando, & S. M. S. Islam (Eds.), *Computer Networks, Big Data and IoT* (Vol. 66, pp. 953–967). Springer Singapore. https://doi.org/10.1007/978-981-16-0965-7_72

Subramanyam, M. S., & Jyothi, C. (2017). *Vehicle Health Monitoring System*. *07*(01), 2.

Yavorskiy, V. M. (2020). *Methods of Analysis, Visualization, Forecast of Financial, Economic and Marketing Data by Means of Integration of Google Technologies and GitHub*. 15.