Aplikasi *driver* ambulans LAZISMU banguntapan selatan menggunakan metode *rapid application development* (RAD) dan *framework* flutter

Faruqi Almando Yusuf Hasda*, Sadr Lufti Mufreni, M.Sc, Esi Putri Silmina

Program Studi Teknologi Informasi, Sains dan Teknologi, Universitas 'Aisyiyah Yogyakarta
Email:faruqialmando@gmail.com; esiputrisilmina@unisayogya.ac.id; sadr@unisayogya.ac.id

Abstrak
LAZISMU, sebuah organisasi yang peduli terhadap layanan ambulans efektif, mengembangkan aplikasi mobile untuk meningkatkan kualitas dan efisiensi layanan tersebut. Aplikasi ini dirancang untuk memfasilitasi layanan ambulans agar lebih efektif dan responsif dalam menjangkau masyarakat yang membutuhkan bantuan medis darurat. Namun, informasi layanan ambulans di Indonesia masih minim, dan infrastruktur yang terbatas menjadi kendala utama. Oleh karena itu, LAZISMU merancang aplikasi driver ambulans berbasis mobile untuk memudahkan driver dalam menerima permintaan ambulans dengan cepat dan tepat. Aplikasi ini menggunakan Metode Rapid Application Development (RAD) dan framework Flutter untuk pengembangannya. Dengan adanya aplikasi ini diharapkan dapat meningkatkan efektivitas dan responsifitas layanan ambulans, serta memudahkan proses pemesanan ambulans bagi masyarakat.

Kata Kunci:, ambulans, aplikasi mobile, LAZISMU; metode rapid application development (RAD); framework flutter.

Application Driver Ambulance LAZISMU Banguntapan Selatan Use *Methode Rapid Application Development* (RAD) *and Framework Flutter*

Abstract
LAZISMU, an organization concerned with effective ambulance services, has developed a mobile application to improve the quality and efficiency of these services. The application is designed to facilitate ambulance services to be more effective and responsive in reaching communities in need of emergency medical assistance. However, information about ambulance services in Indonesia is still limited, and limited infrastructure is a major obstacle. Therefore, LAZISMU has developed a mobile-based ambulance driver application to facilitate drivers in quickly and accurately receiving ambulance requests. This application uses the Rapid Application Development (RAD) Method and Flutter framework for its development. With this application, it is hoped that the effectiveness and responsiveness of ambulance services will be improved, and the process of ordering ambulances will be made easier for the community.

***Keywords:**, ambulance; LAZISMU; mobile application; rapid application development (RAD); flutter framework.*

1. Pendahuluan
RAD merupakan gabungan dari bermacam-macam teknik terstruktur dengan teknik *prototyping* dan teknik pengembangan *join application* untuk mempercepat pengembangan sistem/aplikasi (Kendall & Kendall, 2011). RAD merupakan model proses perangkat lunak yang menekankan pada daur pengembangan hidup yang singkat. RAD merupakan versi adaptasi cepat dari Model *Waterfall*, dengan menggunakan pendekatan konstruksi komponen (Putri & Effendi, 2018).

Ambulans adalah sebuah alat transportasi gawat darurat yang digunakan oleh rumah sakit, puskesmas, atau instansi-instansi terkait yang berfungsi untuk menolong masyarakat yang membutuhkan pertolongan medis. Sejak awal penggagasannya, ambulans berfungsi sebagai kendaraan transportasi pasien dari suatu lokasi ke tempat tujuan lain untuk mendapat perawatan medis lebih lanjut. Seiring perkembangan zaman ambulans kini telah dilengkapi dengan peralatan medis dan tenaga medis, sehingga bukan hanya menjadi kendaraan untuk membawa pasien saja, saat ini juga memungkinkan untuk melakukan tindakan medis di dalam ambulans karena telah dilengkapi berbagai peralatan medis (Fauzi Zulkarnaen & Saleh, 2019).

Flutter merupakan sebuah perangkat pengembangan perangkat lunak (*Software Development Kit*) atau kerangka kerja (*framework*) sumber terbuka yang dikembangkan oleh *google* yang bertujuan untuk

memungkinkan pembuatan dan pengembangan aplikasi yang dapat berjalan di sistem operasi *android* dan *iOS* (Munthe, n.d.). *Dart* digunakan sebagai bahasa pemrograman utama dalam pengkodean aplikasi *flutter*. Salah satu perbedaan utama *flutter* dengan kerangka kerja lainnya adalah dalam proses pembuatan aplikasi, di mana semua kode *flutter* dikompilasi ke kode *native* (menggunakan *Android* NDK, LLVM, AOT *compiled*) tanpa adanya interpreter yang terlibat, sehingga proses kompilasi menjadi lebih cepat (ERIC WINDMILL, 2019).

LAZISMU, sebagai organisasi yang peduli terhadap layanan ambulans, berupaya meningkatkan kualitas dan efisiensi layanan tersebut dengan mengembangkan aplikasi khusus bagi driver ambulans. Aplikasi ini bertujuan untuk mempermudah proses permintaan ambulans, terutama dalam memberikan respons cepat dan tepat kepada masyarakat yang membutuhkan bantuan medis darurat (HAFIZI, 2017).

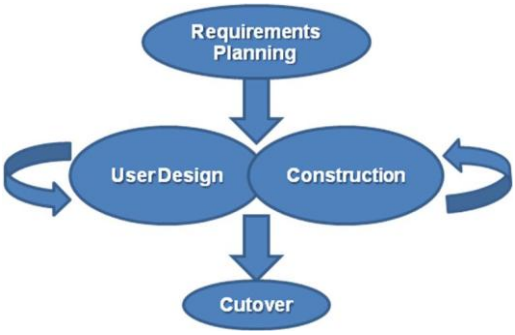
Kendala layanan ambulans di Indonesia, seperti minimnya informasi dan infrastruktur terbatas, menghambat efektivitas layanan. Dengan mengadopsi teknologi mobile, terutama melalui sistem operasi Android, aplikasi ini memanfaatkan Google Maps API untuk membantu driver menemukan lokasi pemesan ambulans (Simamora et al., 2020).

Pembuatan aplikasi ini menggunakan metode *Rapid Application Development* (RAD) dan framework Flutter, yang memungkinkan pengembangan aplikasi mobile lintas platform (Android dan iOS) secara efisien. Diharapkan, dengan aplikasi ini, layanan ambulans LAZISMU dapat lebih optimal dalam memberikan pelayanan darurat.

2. Metode

2.1. Metode Penelitian

Tahapan metode RAD ditunjukkan pada Gambar 1. Tahapan Metode RAD.



Gambar 1. Tahapan Metode RAD

Tahapan metode RAD dimulai dari *requirements planning*, *user design*, *construction*, dan *cutover* (Kendall & Kendall, 2011).

2.1.1. Requirements Planning

Kolaborasi dalam menggabungkan perencanaan sistem dan fase analisis dalam SDLC (*Software Development Life Cycle*). Tim terdiri dari pengguna, manajer, dan staf TI yang berinteraksi untuk merumuskan dan menyetujui kebutuhan bisnis, cakupan proyek, batasan, serta keperluan sistem. Tahap perencanaan kebutuhan berakhir setelah kesepakatan tim dan persetujuan manajemen tercapai.

2.1.2. User Design

Tahap user design melibatkan kolaborasi antara pengguna dan analis sistem dalam merancang model dan prototipe yang mencerminkan seluruh proses, masukan, dan keluaran sistem. Proses desain pengguna adalah interaktif dan berkesinambungan, memungkinkan pengguna untuk memahami, menyesuaikan, dan akhirnya menyetujui model operasional sistem yang memenuhi kebutuhan mereka.

2.1.3. Construction

Tahap construction melibatkan pembuatan program dan aplikasi sesuai dengan prinsip SDLC. Metode RAD melibatkan pengguna agar aktif berpartisipasi dan dapat memberikan masukan sepanjang perkembangan aplikasi. Tahap konstruksi berfokus pada proses pemrograman, pengembangan aplikasi, seperti pengkodean, penggabungan unit, dan pengujian sistem.

2.1.4. Cutover

Tahap cutover mirip dengan fase akhir dalam implementasi SDLC, termasuk konversi data, pengujian, peralihan ke sistem baru, dan pelatihan pengguna. Dalam RAD, semua langkah ini dikompresi untuk memungkinkan pembangunan, peluncuran, dan operasional sistem baru yang lebih cepat.

2.2. Metode Pengujian

2.2.1 White box testing

White box testing adalah metode pengujian perangkat lunak dengan memperhatikan struktur internal pada aplikasi, termasuk logika internal, struktur kode, dan alur aplikasi. Penguji harus mempunyai kemampuan pemrograman yang bagus, pengetahuan tentang bahasa pemrograman, dan mengetahui detail informasi tentang aplikasi (Verma et al., 2017).

2.2.2 Black box testing

Black box testing adalah metode uji perangkat lunak yang mengevaluasi fungsi aplikasi tanpa memerhatikan struktur internalnya. Metode *black box* sesuai digunakan dalam pengujian pada berbagai tingkatan, seperti unit, integrasi, sistem, dan penerimaan. Metode *black box* umumnya dipakai untuk menguji keseluruhan fungsi sistem, menemukan bug atau kesalahan dalam interaksi dengan pengguna, serta mengenali masalah performa sistem (Fahrezi et al., 2022).

2.3. Metode Pengumpulan Data

2.3.1 Wawancara

Wawancara adalah metode pengumpulan data yang melibatkan interaksi langsung antara pewawancara dan responden dengan tujuan untuk mendapatkan informasi yang relevan dan mendalam tentang subjek yang sedang diteliti. Metode wawancara dipakai untuk memberikan pertanyaan mengenai bagaimana proses pengolahan data transaksi dan observasi awal mengenai sistem yang sedang digunakan apakah dalam pengolahan data menggunakan sistem terkomputerisasi atau masih konvensional dalam penyimpanan data (Atmaja et al., 2023).

2.3.2 Observasi

Observasi atau pengamatan digunakan untuk mengumpulkan informasi atau data dengan cara mengamati dan mencatat fenomena-fenomena yang terjadi pada objek pengamatan. Peneliti secara langsung mengamati dan mencatat apa yang terjadi, baik itu perilaku, interaksi, situasi, atau lingkungan tertentu. Tujuan observasi adalah untuk memperoleh data deskriptif dan objektif tentang subjek yang diamati (Andriani & Qurniati, 2018).

2.3.3 Studi literatur

Studi literatur juga dikenal sebagai tinjauan pustaka atau studi pustaka, adalah proses penyelidikan dan analisis terhadap literatur yang relevan yang sudah ada tentang topik tertentu. Tahap studi pustaka melibatkan membaca dan memahami jurnal atau literatur yang relevan dengan penelitian, dengan tujuan menggunakan landasan teori tersebut untuk mengarahkan pemecahan masalah dan memastikan bahwa penulisan dan penelitian tetap konsisten dengan teori yang ada (Atmaja et al., 2023).

3. Hasil dan Pembahasan

Penerapan RAD dilakukan sebanyak tiga siklus iterasi. Iterasi pertama menghasilkan prototipe desain yang sudah diuji dan divalidasi. Iterasi kedua menghasilkan sistem yang sudah diuji dan divalidasi berdasarkan persyaratan keputusan lanjutan dari iterasi pertama. Iterasi ketiga menghasilkan

sistem yang sudah diuji dan divalidasi berdasarkan persyaratan keputusan lanjutan dari iterasi kedua. Tahap akhir dari penerapan RAD menghasilkan peluncuran sebuah versi sistem.

3.1. Requirements Planning

Tahap *requirements planning* dibagi menjadi dua bagian yaitu untuk mendapatkan kebutuhan fungsional dan kebutuhan non fungsional.

3.1.1. Kebutuhan Fungsional

Kebutuhan fungsional dapat diketahui setelah melakukan beberapa tahapan identifikasi dan analisis. Identifikasi dimulai dengan mengidentifikasi masalah, kemudian identifikasi kebutuhan yang dilakukan dengan wawancara, observasi dan studi literatur. Hasil identifikasi masalah dan identifikasi kebutuhan kemudian dianalisis untuk mendapatkan kebutuhan fungsional. Kebutuhan fungsional ditunjukkan pada Tabel 1 Kebutuhan Fungsional.

Tabel 1. Kebutuhan Fungsional

Pengguna *Driver*	1. *Driver* dapat *Login* dengan *email* dan *password* yang sudah didaftarkan. 2. *Driver* dapat melakukan *online/offline* sebagai status aktifitas . 3. *Driver* dapat menerima/menolak permintaan ambulans . 4. *Driver* dapat melihat informasi permintaan. 5. *Driver* dapat melihat lokasi permintaan. 6. *Driver* dapat melihat informasi perjalanan.
Pengguna Umum	1. Aplikasi harus memiliki antarmuka yang mudah dipahami dan digunakan oleh pengguna, dengan navigasi yang jelas, ikon yang intuitif, dan tata letak yang rapi. 2. Aplikasi harus menjaga keamanan dan privasi data pengguna, termasuk perlindungan terhadap akses tidak sah dan penggunaan data pribadi.

3.1.2 Kubutuhan Non Fungsional

Berikut adalah beberapa kebutuhan non fungsional yang dibutuhkan untuk membangun sistem. Kebutuhan non fungsional ditujukkan pada Tabel 2 Kebutuhan Non Fungsional.

Tabel 2. Kebutuhan Non-Fungsional

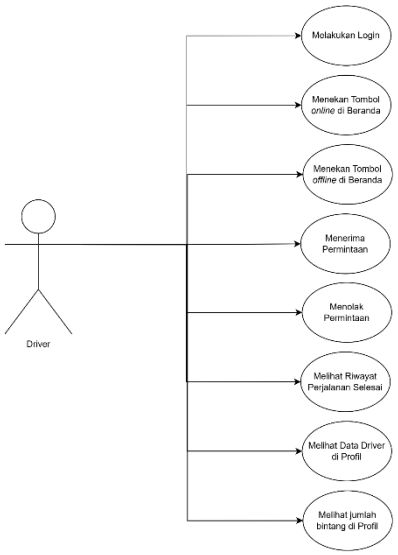
Perangkat Keras	1. Laptop HP Pavilion *Gaming* 15-ec2xxx 2. AMD *Ryzen* 7 4800H *with* Radeon *Graphics* 3.30 GHz. 3. 16GB RAM. 4. 512 SSD.
Perangkat Lunak	1. *Windows* 11 *Home Single Language* 22H2 64-bit. 2. *Microsoft Edge*. 3. *Figma*. 4. *Visual Studio Code.* 5. *Android Studio Electric Eel* | 2022.1.1

3.2. User Design

Tahap *user design* meliputi pembuatan UML (*Unified Modeling Language*). UML yang dibuat antara lain *use case diagram, activity diagram* dan *sequence diagram* (Sonata, 2019).

3.2.1. Use Case Diagram

Use case merupakan representasi visual dari interaksi antara sistem dan aktor, yang menjelaskan jenis interaksi antara pengguna dalam suatu sistem dengan sistem itu sendiri melalui sebuah narasi yang menggambarkan penggunaan sistem (T. Bayu Kurniawan, 2020). *Use case diagram* ditunjukkan pada Gambar 2. *Use Case Diagram*.

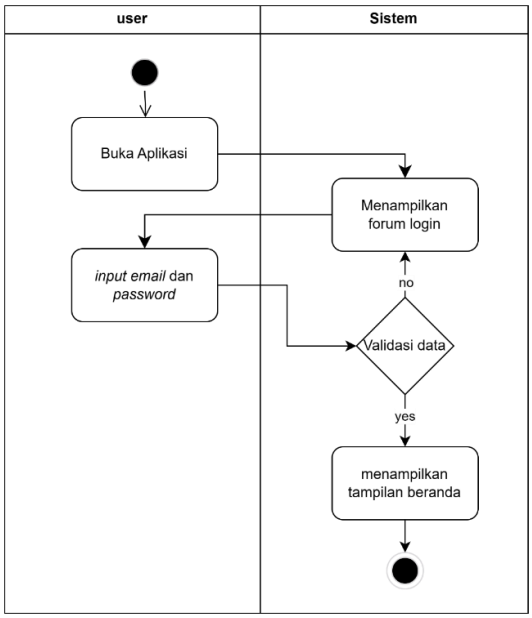


Gambar 2. *Use Case Diagram*

Berdasarkan *use case diagram* tersebut, diketahui bahwa terdapat satu aktor yakni pengguna. Pengguna dapat melakukan *login*, melakukan daftar, melakukan permintaan ambulans, melakukan permintaan koin surga, melakukan donasi zakat, melakukan donasi infaq, melakukan donasi sedekah.

3.2.2. Activity Diagram

Activity diagram adalah representasi visual yang mengilustrasikan konsep aliran data/kontrol, tindakan yang terstruktur, dan rancangan yang baik dalam sebuah sistem (Marda Nova, 2018). *Activity diagram* ditunjukkan pada Gambar 3. *Activity Diagram*.

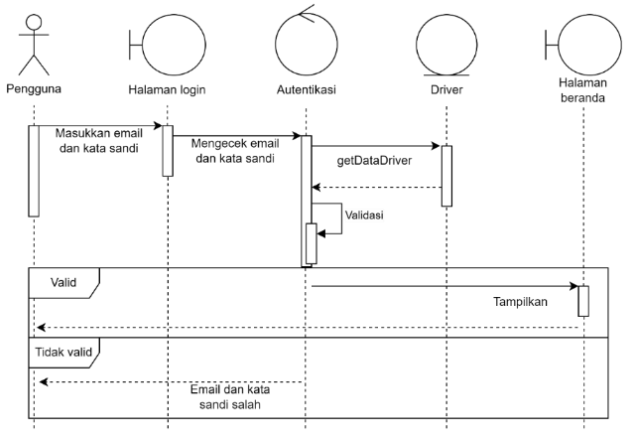


Gambar 3. *Activity Diagram*

Berdasarkan *activity diagram,* ketika pengguna membuka aplikasi maka pengguna akan masuk ke halaman *login*. Pada halaman *login*, pengguna akan diminta memasukkan email dan kata sandi. Setelah menekan tombol masuk maka aplikasi akan mengecek apakah format email yang dimasukkan sudah benar, jika sudah maka aplikasi akan lanjut mengecek apakah email dan kata sandi sudah terdaftar. Pengguna akan masuk ke halaman beranda jika email dan kata sandi sudah terdaftar.

3.2.3. Sequence Diagram

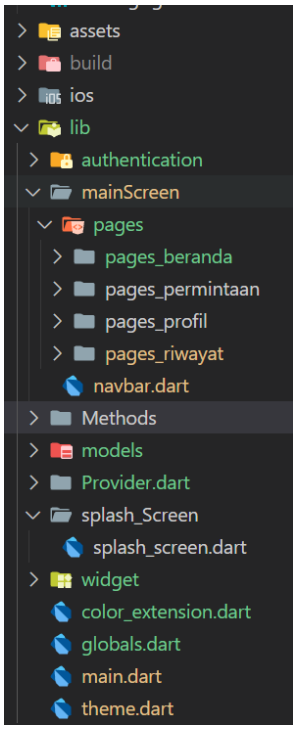
Sequence diagram merupakan suatu gambaran visual yang mengilustrasikan kerja sama antara objek-objek yang berinteraksi satu sama lain dalam suatu kelas. *Sequence diagram* menggunakan pesan-pesan yang ditampilkan dalam urutan waktu untuk mengilustrasikan interaksi antara objek-objek (Sabah Al-Fedaghi, 2021). *Sequence diagram* ditunjukkan pada Gambar 4. *Sequence Diagram.*



Gambar 4. *Sequence Diagram*

3.3. Construction

Tahap *construction* dilakukan dengan mulai membangun aplikasi menggunakan *Flutter*. Tahap ini dilakukan dengan mengikuti rancangan *user design* dan mengikuti desain aplikasi yang telah dibuat. Struktur aplikasi dibagi menjadi beberapa folder utama yakni aktivitas, beranda, profil dan layanan. Struktur aplikasi ditunjukkan pada Gambar 5. Struktur Aplikasi.



Gambar 5. Struktur Aplikasi

3.4. Cutover

Tahap *cutover* menghasilkan versi sistem dari sebuah aplikasi, dan pengujian aplikasi. Pengujian dilakukan dengan dua metode yakni metode *white box* dan *black box*.

3.4.1. Aplikasi Versi 1.0.0

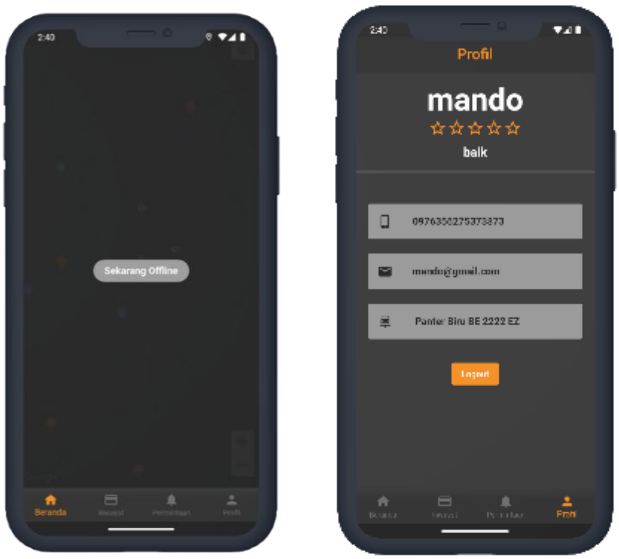
3.4.1.1. Halaman *Login*

(a)

Gambar 6. (a) Halaman *Login*

Pada halaman *login* terdapat *form* email, kata sandi, tombol *login*.. Halaman *login* dan daftar ditunjukkan pada Gambar 6.

3.4.1.2. Halaman Beranda dan Halaman Profil



(a) (b)

Gambar 7. (a) Halaman Beranda (b) Halaman Profil

Halaman beranda menampilkan sebuah tombol bertuliskan "sekarang *offline*", tombol ini berfungsi untuk status driver sedang aktif atau tidak, jika ingin mengubah status menjadi aktif cukup dengan menekan tombol tersebut. Halaman profil menampilkan nama, jumlah bintang, nomor hp, email dan informasi kendaraan yang dipakai. Halaman beranda dan profil ditunjukkan pada Gambar 7.

3.4.1.3. Halaman Riwayat dan Halaman Permintaan

(a) (b)

Gambar 8. (a) Halaman Donasi (b) Halaman Permintaan Ambulans

Halaman riwayat menampilkan data dari permintaan ambulan yang sudah selsai. Halaman permintaan menampilkan *form* permintaan, terdapat 2 buah tombol konfirmasi terima dan tolak. Halaman riwayat dan permintaan ditunjukkan pada Gambar 8.

3.4.2. White Box Testing

Pengujian dengan *white box* dilakukan dengan menguji beberapa fitur utama pada aplikasi. Teknik pengujian yang digunakan adalah *path testing*. pengujian difokuskan pada fitur-fitur tertentu yang diuji dengan mempertimbangkan jalur-jalur yang mungkin dilalui oleh program untuk setiap fitur tersebut. Pengujian white box ditunjukkan pada Tabel 3 *White Box Testing*.

Tabel 3. White Box Testing.

NO	Testing Input	Fungsi	Output	Hasil
1	*Email* dan *password*	Masuk ke halaman beranda.	Berhasil masuk ke halaman beranda	Berhasil
2	Tombol *offline/online*	Mengubah status *offline/online*.	Berhasil mengubah status *offline/online*	Berhasil
3	Masuk ke halaman permintaan	Masuk ke halaman permintaan dan menerima/menolak permintaan.	Berhasil masuk ke halaman permintaan dan menerima/menolak permintaan	Berhasil
4	Masuk ke halaman riwayat	Masuk ke halaman riwayat dan melihat data riwayat perjalanan selesai.	Berhasil masuk ke halaman riwayat dan melihat data riwayat perjalanan selesai.	Berhasil
5	Masuk ke halaman profil	Masuk ke halaman profil dan melihat informasi data driver.	Berhasil masuk ke halaman profil dan melihat informasi data driver.	Berhasil

3.4.3. Black Box Testing

Pengujian sistem menggunakan metode *black box* dilakukan untuk memastikan bahwa aplikasi yang dibuat sudah sesuai dengan prototipe dan persyaratan bisnis yang sudah ada. *Black box*

dikombinasikan dengan skala *likert* untuk mendapatkan penilaian dari responden terhadap aspek-aspek pengujian. Penggabungan dilakukan untuk mengetahui kelayakan dari aplikasi dari sudut pandang pengguna. Penilaian terhadap kepuasan pengguna melibatkan 5 orang responden dari latar belakang supir. Responden diminta untuk mengisi kuisioner dengan model penilaian skala *likert* untuk menilai kepuasan penggunaan

Tabel 4. Keterangan penilaian skala likert

No.	Skor	Skala	Persentase (%)	Keterangan
1	1	STS (Sangat Tidak Setuju)	0 – 19.99	Performa sangat buruk, banyak kesalahan dan kegagalan.
2	2	TS (Tidak Setuju)	20 – 39.99	Performa kurang memuaskan, beberapa kesalahan.
3	3	Netral (N)	40 – 59.99	Performa rata-rata, beberapa aspek masih perlu perbaikan.
4	4	S (Setuju)	50 – 79.99	Performa baik, sedikit kesalahan atau kegagalan.
5	5	SS (Sangat Setuju)	80 – 100	Performa sangat baik, hampir tidak ada kesalahan.

Tabel 5. Pertanyaan Kuisioner

No.	Aspek Pengujian	Deskripsi
1	Antarmuka Pengguna (UI/UX)	Sejauh mana Anda merasa antarmuka pengguna (UI/UX) aplikasi ini intuitif dan mudah digunakan?
2	Kemudahan Penggunaan	Sejauh mana Anda merasa nyaman dengan cara aplikasi ini berfungsi?
3	Kinerja Aplikasi	Sejauh mana kinerja aplikasi ini memenuhi ekspektasi Anda?
4	Fungsionalitas	Sejauh mana fitur-fitur aplikasi ini sesuai dengan kebutuhan Anda?
5	Keamanan	Sejauh mana Anda merasa data Anda aman saat menggunakan aplikasi ini?

Tabel 6. Black Box dengan Skala Likert

Responden	AP1	AP2	AP3	AP4	AP6
1	4	4	4	4	**3**
2	5	4	4	5	**3**
3	5	5	5	5	**5**
4	5	5	5	4	**4**
5	**4**	**5**	**4**	**5**	**5**

Berdasarkan tabel *black box testing* dengan skala *likert* dapat dihitung hasil dari tiap aspek pengujian dengan rumus index (%) = (total skor/skor maksimum) x 100 sebagai berikut:

a) Antarmuka pengguna (UI/UX)
   Diketahui:
   SS = 5 responden x 5 skor = 25 skor
   S = 5 responden x 4 skor = 20 skor
   Total skor = SS + S = 45 skor
   Skor maksimum = 10 responden x 5 (skor tertinggi) = 50 skor
   Jadi, (45/50) x 100 = 90% responden merasa aplikasi intuitif dan sangat mudah digunakan.
b) Kemudahan Pengguna
   Diketahui:
   SS = 4 responden x 5 skor = 20 skor
   S = 6 responden x 4 skor = 24 skor
   Total skor = SS + S = 44 skor
   Skor maksimum = 10 responden x 5 (skor tertinggi) = 50 skor
   Jadi, (44/50) x 100 = 88% responden merasa aplikasi sangat nyaman digunakan.
c) Kinerja Aplikasi
   Diketahui:
   SS = 2 responden x 5 skor = 10 skor
   S = 8 responden x 4 skor = 32 skor
   Total skor = SS + S = 42 skor

Skor maksimum = 10 responden x 5 (skor tertinggi) = 50 skor
Jadi, (42/50) x 100 = 84% pengguna merasa kinerja aplikasi sangat memuaskan.

d) Fungsionalitas
Diketahui:
SS = 3 responden x 5 skor = 15 skor
S = 5 responden x 4 skor = 20 skor
N = 2 responden x 3 skor = 6 skor
Total skor = SS + S = 41 skor
Skor maksimum = 10 responden x 5 (skor tertinggi) = 50 skor
Jadi, (41/50) x 100 = 82% responden merasa fitur aplikasi sangat sesuai dengan kebutuhan mereka.

e) Keamanan
Diketahui:
SS = 3 responden x 5 skor = 15 skor
S = 4 responden x 4 skor = 16 skor
N =3 responden x 3 skor = 9 skor
Total skor = SS + S = 40 skor
Skor maksimum = 10 responden x 5 (skor tertinggi) = 50 skor
Jadi, (40/50) x 100 = 80% responden merasa data mereka sangat aman saat menggunakan aplikasi.

Hasil perhitungan skala *likert* di atas menunjukkan bahwa mayoritas responden puas dengan pengalaman pengguna secara keseluruhan dengan nilai rata-rata 84,80%. Berdasarkan skor dan persentase kepuasan pengguna diketahui bahwa responden sangat setuju dengan aplikasi. Meskipun beberapa area dapat memerlukan peningkatan, umpan balik dari pengguna secara keseluruhan adalah positif.

4. Kesimpulan

Aplikasi *driver* ambulans LAZISMU Banguntapan Selatan berhasil dikembangkan menggunakan metode RAD dengan *framework flutter*. Metode RAD diterapkan dengan menerapkan tahapan-tahapannya yang meliputi perencanaan kebutuhan, desain pengguna, konstruksi dan peralihan. Tujuan penelitian ini adalah untuk mengembangkan aplikasi *driver* ambulans di LAZISMU Banguntapan Selatan. Hasil dari pengujian *white box* menunjukkan bahwa semua fitur pada aplikasi sesuai dengan yang diharapkan. Hasil dari pengujian *black box testing* dengan menggunakan skala *likert* menunjukkan bahwa responden sangat setuju dengan aplikasi.

5. Ucapan terimakasih

Terima kasih disampaikan kepada LAZISMU Banguntapan Selatan yang telah memberikan izin dan kesempatan untuk melakukan penelitian. Serta terima kasih kepada seluruh pihak yang terlibat dalam penelitian sehingga penelitian dapat berjalan dengan lancar.

Daftar Pustaka

Sabah Al-Fedaghi. (2021). UML Sequence Diagram: An Alternative Model. In *IJACSA) International Journal of Advanced Computer Science and Applications* (Vol. 12, Issue 5). www.thesai.org

Andriani, A., & Qurniati, E. (2018). Sistem Informasi Penjualan Pada Toko Online Dengan Metode Rapid Application Development (RAD). *Journal Speed – Sentra Penelitian Engineering Dan Edukasi*, *10*(3), 49–54.

Atmaja, R. D., Faizah, N., & Kambry, M. A. (2023). Aplikasi E–Commerce Toko Sinar Bella Dengan Metode Rapid Application Development (RAD) Menggunakan Framework CodeIgniter 4. *Design Journal*, *1*(1), 26–37. https://doi.org/10.58477/dj.v1i1.26

Eric Windmill. (2019). *Flutter in Action.*

Fahrezi, A., Salam, F. N., Ibrahim, G. M., Syaiful, R. R., & Saifudin, A. (2022). Pengujian Black Box Testing pada Aplikasi Inventori Barang Berbasis Web di PT. AINO Indonesia. *Jurnal Ilmu Komputer Dan Pendidikan*, *1*(1), 1–5. https://journal.mediapublikasi.id/index.php/logic

Fauzi Zulkarnaen, M., & Saleh, M. (2019). Aplikasi Berbasis Android Sebagai Layanan Pemesnan Ambulans Studi Kasus Puskesmas AIK Darek. In *Jurnal Manajemen informatika & Sistem Informasi)* (Vol. 2, Issue 2). Online. http://e-journal.stmiklombok.ac.id/index.php/misi

HAFIZI. (2017). *Moderenisasi Pengelolaan Zakat di LAZISMU*.

Kendall, K. E., & Kendall, J. E. (2011). *Systems analysis and design*. Pearson Prentice Hall.

Marda Nova. (2018). *Aplikasi Layanan Ambulan Untuk Situasi Darurat Berbasis Android*.

Munthe, S. S. (n.d.). *Dasar Flutter Dalam Pengembangan Aplikasi Profil*.

Putri, M. P., & Effendi, H. (2018). Implementasi Metode Rapid Application Development Pada Website Service Guide "Waterfall Tour South Sumatera." In *Jurnal SISFOKOM* (Vol. 07).

Simamora, R., Made, I., & Suyadnya, A. (2020). *Rancang Bangun Aplikasi Ambulance Oonline Berbasis Android*. *3*. https://doi.org/10.31598

Sonata, F.-. (2019). Pemanfaatan UML (Unified Modeling Language) Dalam Perancangan Sistem Informasi E-Commerce Jenis Customer-To-Customer. *Jurnal Komunika : Jurnal Komunikasi, Media Dan Informatika*, *8*(1), 22. https://doi.org/10.31504/komunika.v8i1.1832

T. Bayu Kurniawan, S. (2020). *Perancangan Sistem Aplikasi Pemesanan Makanan dan Minuman Pada Cafetaria no Caffe di Tanjung Balai Karimun Menggunakan Bahasa Pemrograman PHP dan MYSQL*.

Verma, A., Khatana, A., & Chaudhary, S. (2017). A Comparative Study of Black Box Testing and White Box Testing. *International Journal of Computer Sciences and Engineering*, *5*(12), 301–304. https://doi.org/10.26438/ijcse/v5i12.301304