BAB I

PENDAHULUAN

A. Latar Belakang Masalah

Teknologi Informasi dan Komunikasi (TIK) memiliki peran krusial dalam menunjang mekanisme bisnis, khususnya pada hal manajemen data, informasi, fakta, distribusi, dan pengetahuan. Untuk berbagai instansi pemerintahan, dimulai dari tingkat Kabupaten, Kecamatan, Desa, hingga Rukun Warga (RW) serta Rukun Tetangga (RT), salah satu bentuk dukungan layanan kepada masyarakat yang esensial adalah penyediaan aplikasi administrasi pelayanan publik secara langsung. Beberapa masalah yang dihadapi masyarakat saat ingin mengurus administrasi di Kantor Desa seperti jarak antara rumah dan kantor desa sangat jauh yang menghabiskan banyak waktu di perjalanan, mengeluarkan biaya transportasi, penumpukan berkas permohonan dan tidak mengetahui apakah petugas administrasi saat ini berada di Kantor Desa.

Cloud computing merupakan pendekatan komputasi berbasis internet yang memungkinkan pengguna untuk memaksimalkan layanan teknologi informasi dan komunikasi. Dalam model ini, penyedia layanan bertanggung jawab penuh atas pengelolaan infrastruktur, platform, dan aplikasi, sehingga pelanggan tidak perlu melakukan investasi awal. Pendekatan ini tidak hanya memberikan kemudahan akses, tetapi juga secara signifikan meningkatkan kesederhanaan dan efisiensi, baik bagi penyedia layanan maupun bagi para pengguna. *Cloud computing* dianggap sebagai transformasi fundamental yang akan mendorong peningkatan

produktivitas sistem dan pengguna di masa depan, serta menjadi inovasi penting dalam era internet dan komunikasi virtual. Implementasi cloud computing diperkirakan akan meluas ke berbagai sektor, termasuk perusahaan, pemerintahan, dan individu.

Berdasarkan uraian sebelumnya, peneliti berencana mengembangkan Aplikasi Pelayanan Kantor Desa Berbasis Android yang mengimplementasikan *Cloud Computing*. Aplikasi ini akan menyediakan layanan administrasi jarak jauh, memungkinkan komunikasi antara masyarakat dan aparatur desa melalui berbagai media seperti telepon seluler, telepon rumah, atau internet. Manfaat utama dari aplikasi ini adalah efektivitas dalam hal waktu, biaya, serta tenaga. Dengan demikian, masyarakat tidak wajib secara langsung ke kantor desa, yang seringkali memakan durasi perjalanan, biaya transport, serta tenaga, terutama saat menghadapi kemacetan, hanya untuk mengajukan permohonan izin usaha atau layanan administrasi lainnya kepada petugas.

Implementasi teknologi canggih dapat menyederhanakan alur kerja, meningkatkan efisiensi operasional, dan memberikan pengalaman yang lebih baik bagi masyarakat. (Ade Hastuty, 2025) Penerapan administrasi berbasis elektronik membawa berbagai manfaat signifikan, terutama dalam mengatasi penumpukan berkas permohonan dan meningkatkan efisiensi waktu. Warga tidak lagi diwajibkan untuk selalu datang langsung ke kantor desa; mereka hanya perlu mengisi *form* melalui perangkat *mobile* masing-masing untuk mengajukan layanan administrasi. Intinya, sistem ini memungkinkan masyarakat untuk memperoleh pelayanan, bertanya, atau berkonsultasi dengan aparatur desa secara

jarak jauh. Pendekatan ini diyakini mampu menjadi solusi efektif dalam mengatasi berbagai permasalahan terkait pelayanan administrasi desa.

B. Rumusan Masalah

Adapun rumusan masalah dalam penelitian ini adalah bagaimana cara rancang bangun pelayanan administrasi kantor desa di Kecamatan Watang Pulu kabupaten Sidrap?

C. Batasan Masalah

Batasan masalah dalam membuat aplikasi ini adalah :

1. Ruang lingkup sistem aplikasi ini dibatasi pada satu lokasi penelitian, yaitu Kantor Desa Lainungan di Kecamatan Watang Pulu, Kabupaten Sidrap.
2. Data yang digunakan dalam aplikasi ini bersumber dari Kantor Desa Lainungan, meliputi jenis-jenis surat pengantar seperti pembuatan akta kelahiran, nikah, kematian, usaha, izin keramaian, dan surat keterangan tidak mampu. Aplikasi hanya berfokus pada pelayanan Administrasi di kantor desa.
3. Aplikasi hanya berbasis Android dibuat menggunakan Android Studio dengan bahasa pemrograman Kotlin.

D. Tujuan Penelitian

Tujuan penelitian ini yaitu membuat suatu aplikasi pelayanan kantor desa yang dapat membantu masyarakat ataupun petugas yang bertugas di kantor desa, mempermudah proses pengelolaan layanan administrasi dan meningkatkan layanan di kantor desa.

E. Manfaat Penelitian

Manfaat dari penelitian ini adalah :

a. Bagi Peneliti

Dari penelitian ini peneliti dapat menambah dan memperluas wawasan dalam pengetahuan baik secara teori dan praktik, serta dapat lebih belajar dalam mengambil keputusan atas masalah yang terjadi di lapangan, seperti masalah yang terjadi di Kantor Desa

b. Bagi Masyarakat.

Aplikasi ini berperan sebagai sarana edukasi bagi masyarakat dalam mengelola administrasi, sekaligus menjadi sumber informasi yang mudah diakses oleh seluruh lapisan masyarakat, kapan pun dan di mana pun.

c. Bagi Kantor Desa

Dengan tersedianya sistem aplikasi administrasi desa secara daring, Kantor Desa dapat menerapkan solusi ini untuk secara signifikan meningkatkan kualitas pelayanan kepada masyarakat

F. Sistematika Penulisan

Struktur penulisan penelitian ini adalah :

BAB I : PENDAHULUAN

Bab ini menguraikan elemen-elemen fundamental dari proposal penelitian, yang meliputi latar belakang masalah, perumusan masalah, batasan-batasan penelitian, tujuan yang ingin dicapai, metodologi yang akan digunakan, serta sistematika penulisan proposal ini.

BAB II : TINJAUAN PUSTAKA

Berisi tentang penelitian terdahulu dan dasar teori yang mendukung pembuatan proposal ini.

BAB III : METODE PENELITIAN

Berisi tentang letak lokasi serta waktu, jenis penelitian, metode pengumpulan data, alat dan bahan penelitian, tahap penelitian, metode pengujian, dan analisa sistem.

BAB IV : HASIL PENELITIAN DAN PEMBAHASAN

Berisi tentang gambaran umum, analisis, meliputi pendefinisian dan pemodelan sistem dalam bentuk use case, Class diagram, Sequence diagram serta desain sistem yang meliputi desain database, dan pengujian system dengan *whitebox* dan *blackbox.*

BAB V : PENUTUP

Bagian ini memuat rangkuman temuan dan poin-poin penting yang diperoleh dari penulisan tugas akhir ini, serta menyajikan rekomendasi dan usulan untuk pengembangan lebih lanjut di masa mendatang.

BAB II

TINJAUAN PUSTAKA

A. Kajian Teori

1. Perangkat Desa

Berdasarkan Pasal 1 angka 3 Undang-Undang Desa, Perangkat Desa adalah bagian dari struktur pemerintahan desa yang bertindak sebagai pembantu Kepala Desa dalam melaksanakan fungsi-fungsi pemerintahan desa . Peran ini serupa dengan kedudukan Wakil Presiden dan menteri-menteri dalam sistem ketatanegaraan Indonesia yang juga berfungsi sebagai pembantu pimpinan tertinggi. Perangkat Desa terdiri dari Kepala Desa sebagai pimpinan tertinggi, dibantu oleh Sekretaris Desa. Terdapat pula Kepala Urusan Keuangan dengan dukungan Staf Administrasi Keuangan, serta Kepala Urusan Umum yang dibantu oleh Staf Pengelola Data. Selain itu, terdapat Kepala Seksi Pemerintahan, Ketentraman, dan Ketertiban (Kasi Pem & Trantib), serta Kepala Seksi Ekonomi Pembangunan dan Kesejahteraan Rakyat (Kasi Ekbang & Kesra) yang dibantu oleh Mantri Tani Desa. Terakhir, terdapat Staf Sekretariat BPD yang bertanggung jawab langsung kepada Lembaga BPD. Berikut tugas dan fungsinya secara umum:

a. Kepala Desa

Kepala Desa berkedudukan sebagai pemimpin penyelenggaraan pemerintahan desa, dengan tugas utama meliputi pelaksanaan pemerintahan, pembangunan, pembinaan dan pemberdayaan masyarakat, serta menjaga kemitraan dengan berbagai lembaga. Hal ini dijelaskan lebih lanjut dalam Peraturan Desa

(Perdes) Nomor 1 Tahun 2018, khususnya pada Pasal 4, yang mengatur tentang Susunan Organisasi dan Tata Kerja Pemerintah Desa (SOTK), sekitar 70% dari tugas dan fungsi Kepala Desa berfokus pada sosialisasi dan pendekatan langsung dengan masyarakat. Hal ini bertujuan untuk memahami kebutuhan warga agar dapat direalisasikan melalui program kerja desa, serta memenuhi undangan atau melakukan dinas luar demi kepentingan dan kesejahteraan masyarakat desa.

a. Sekretaris Desa

Berdasarkan Perdes Nomor 1 Tahun 2018 tentang SOTK Pasal 5, Sekretaris Desa berperan bertindak sebagai pembantu Kepala Desa dan bertanggung jawab langsung kepadanya. Fungsi utamanya mencakup pelaksanaan administrasi pemerintahan desa, urusan umum, urusan ketatausahaan, urusan perencanaan, urusan keuangan, mengkoordinasikan tugas Perangkat Desa lainnya, serta membina Perangkat Desa. Selain itu, Sekretaris Desa juga bertugas menyediakan pelayanan kepada masyarakat dan menjalankan tugas-tugas lain yang didelegasikan oleh Kepala Desa sesuai dengan lingkup fungsi dan tanggung jawabnya. Secara umum, Sekretaris Desa berfungsi sebagai peninjau, pemantau, dan koordinator pelaksanaan tugas, sekaligus pembina bagi Perangkat Desa lainnya, serta bertindak sebagai wakil Kepala Desa dalam setiap tugas yang diberikan.

b. Kepala Urusan Umum

Berdasarkan Perdes nomor 1 Tahun 2018 Tentang SOTK Pasal 7, Fungsi Kepala Urusan Umum yang selanjutnya di singkat Kaur Umum yaitu tata naskah, administrasi surat menyurat, penataan administrasi perangkat desa,

penyiapan rapat, pengelolaan dan pencatatan aset , administrasi perjalanan dinas, serta penyediaan layanan umum yang dalam pelaksanaannya dibantu oleh Staf Pengelola Data. Jadi Kaur Umum harus mampu menyusun RAB (Rencana Anggaran Biaya) dan membuat Laporan Pertanggungjawaban atas kegiatan yang menjadi tugasnya seperti Operasional Pemerintah Desa, Penunjang Penyelenggara Pemerintah Desa, dan Pengadaan Barang Sarana dan Prasarana Pemerintah Desa.

c. Kepala Urusan Keuangan

Berdasarkan Perdes Nomor 1 Tahun 2018 tentang SOTK Pasal 7, Kepala Urusan Keuangan (Kaur Keuangan) memiliki fungsi utama dalam pengurusan administrasi keuangan desaFungsi ini mencakup pengelolaan administrasi seluruh sumber pendapatan dan pengeluaran desa, verifikasi administrasi keuangan, serta administrasi penghasilan tetap bagi Kepala Desa, Perangkat Desa, Badan Permusyawaratan Desa (BPD), serta lembaga pemerintahan desa lainnya. Selain itu, Kaur Keuangan juga bertanggung jawab dalam menyusun pelaporan dan pertanggungjawaban pelaksanaan APBDesa, mulai dari penyusunan APBDes murni hingga APBDes Perubahan.

d. Kepala Seksi Pemerintahan Keamanan Ketertiban

Tugas dan fungsi Kasi Pem & Trantib meliputi pembinaan ketentraman dan ketertiban, khususnya di posko jaga utama dan pos kamling. Mereka juga bertanggung jawab atas pengelolaan administrasi dan penagihan Pajak Bumi dan Bangunan, serta penyusunan Rencana Anggaran Biaya (RAB) dan laporan pertanggungjawaban untuk bidang pemerintahan. Ini mencakup Penunjang Operasional BPD, Insentif RT/RW, Musyawarah Desa, dan Pendataan

Profil Desa. Selain itu, Kasi Pem & Trantib bertugas menyusun laporan kependudukan dan catatan sipil, termasuk memberikan pelayanan pengajuan pembuatan Kartu Keluarga (KK), Kartu Tanda Penduduk (KTP), dan akta kelahiran, yang dibantu oleh staf desa..

e. Kepala Seksi Ekonomi Pembangunan Kesejahteraan Rakyat (Kasi Ekbang & Kesra)

Tugas dan fungsi Kasi Ekbang & Kesra mencakup penyusunan dokumen Rencana Pembangunan Jangka Menengah Desa (RPJMDes) dan Rencana Kerja Pemerintah Desa (RKP Desa). Mereka juga bertanggung jawab dalam menyusun RAB, desain/gambar, serta laporan pertanggungjawaban untuk bidang pembangunan, pemberdayaan, dan pembinaan. Kasi Ekbang & Kesra juga harus mampu memahami dan memfasilitasi kelembagaan masyarakat (Rusito, 2018).

2. Android

Android adalah sistem operasi mobile yang dikembangkan dari basis Linux yang telah dimodifikasi. Awalnya merupakan proyek dari Android Inc., perusahaan tersebut kemudian diakuisisi oleh Google pada tahun 2005. Akuisisi ini merupakan bagian dari strategi Google untuk memasuki pasar mobile, yang kemudian mengambil alih seluruh pengembangan dan tim di balik sistem operasi ini.

Google berambisi menjadikan Android sebagai sistem operasi sumber terbuka (open source) dan gratis. Sebagian besar kode Android tersedia di bawah lisensi open source Apache, yang memungkinkan siapa pun untuk menggunakan serta mengunduh kode sumbernya secara bebas tanpa batasan.

Android dilengkapi dengan berbagai fitur utama, antara lain:

a. Penyimpanan (*Storage*): Mendukung penggunaan SQLite, sebuah basis data relasional versi ringan yang efisien untuk penyimpanan data.
b. Konektivitas: Memiliki dukungan luas untuk berbagai standar koneksi seperti GSM (EDGE), IDEN, SDMA, SVDO, UMTS, Bluetooth, Wi-Fi, LTE, dan WiMAX.
c. Pesan (*Messaging*): Mendukung layanan pesan teks (SMS) dan pesan multimedia (MMS).
d. Peramban Web (Web Browser): Menggunakan peramban web yang berbasis open *source*. (Eko Budi Setiawan, 2018).

3. Kotlin



Gambar 2. 1. Bahasa *Programan Kotlin*

Kotlin adalah bahasa pemrograman modern, disajikan secara statis yang berjalan pada platform *Java Virtual Machine* (JVM). Kotlin menggunakan compiler LLVM yang artinya, dapat dikompilasi ke dalam kode JavaScript. Kotlin bisa menerapkan mekanisme compile pada Java, yaitu menulis kode lalu menjalankannya di platform yang mendukung (*Write Once Run Anywhere*).

a. Sejarah Perkembangan Kotlin

Kotlin adalah bahasa pemrograman yang diciptakan oleh JetBrains pada tahun 2011. Inisiatif ini muncul dari pengamatan pemimpin JetBrains, Dmitry Jemerov, terhadap kerumitan aturan *coding* pada bahasa pemrograman yang ada, seperti Java, yang seringkali mengharuskan pengembang menulis banyak baris kode untuk perintah sederhana. Kondisi ini dianggap kurang efisien dan memperlambat proses pengembangan aplikasi.

Pada tahun 2012, JetBrains mengubah status Kotlin menjadi *open-source* di bawah lisensi Apache 2.0. Keputusan ini menjadikan Kotlin dapat digunakan dan dikembangkan secara bebas tanpa biaya. Sejak saat itu, banyak perusahaan besar, termasuk Atlassian, Pinterest, dan Uber, mulai mengadopsi Kotlin dalam pengembangan produk mereka.

Perkembangan Kotlin semakin pesat setelah Google pada tahun 2017 memberikan dukungan kelas satu (*first-class support*). Bahkan, pada tahun 2019, Google memprioritaskan Kotlin sebagai bahasa pemrograman utama untuk pengembangan aplikasi Android, mengungguli bahasa seperti C++ atau Java.

b. Fungsi Kotlin

Sebagai bahasa pemrograman, fungsi utama Kotlin memang untuk menciptakan aplikasi Android yang berjalan lancar. Namun, cakupan fungsionalitas Kotlin tidak terbatas pada Android saja. Kotlin juga merupakan bahasa pemrograman *multiplatform*, yang berarti dapat digunakan untuk mengembangkan aplikasi iOS dan berbasis web.

Berikut ini kami sajikan beberapa kelebihan yang dimiliki bahasa Kotlin:

1) Ekspresif dan Ringkas

Kotlin dikenal sebagai bahasa pemrograman yang ekspresif, yang berarti setiap perintah yang akan digunakan perlu dideklarasikan atau dikenalkan terlebih dahulu. Pendekatan ini tidak hanya meminimalkan potensi kesalahan (*error*) tetapi juga mempermudah pengembang dalam memperbarui aplikasi yang sudah dirilis.

2) Kompatibilitas dengan Java

Kotlin dapat digunakan secara bersamaan dengan Java karena kedua bahasa ini memiliki kemampuan untuk saling berkomunikasi dan bertukar informasi, termasuk fungsi, kelas, maupun variabel. Interoperabilitas ini dimungkinkan karena Kotlin dan Java sama-sama berjalan di atas *platform* Java Virtual Machine (JVM), di mana keduanya menghasilkan *file* berjenis *byte-code* pada proses kompilasi.

3) Serentak dan Terstruktur

Bahasa Kotlin memiliki karakteristik serentak dan terstruktur. Ini berarti bahwa meskipun dua kode dieksekusi secara tidak bersamaan (asinkron), hasil akhir strukturnya tetap sama persis, memastikan konsistensi dan keteraturan dalam eksekusi program. (Hardana, 2018).

4. Android Studio

Android Studio adalah Integrated Development Environment (IDE) resmi untuk pengembangan aplikasi Android, yang dibangun di atas fondasi IntelliJ IDEA. Selain berfungsi sebagai editor kode serta menawarkan fitur-fitur pengembangan yang tangguh dari IntelliJ, Android Studio juga menyediakan berbagai fitur yang dirancang khusus untuk meningkatkan produktivitas dalam pembuatan aplikasi Android, antara lain:

a. Sistem Build Berbasis Gradle yang Adaptif: Fitur ini memfasilitasi konfigurasi proyek yang kompleks dan otomatisasi proses build.
b. Emulator yang Cepat dan Lengkap: Menyediakan lingkungan simulasi perangkat Android yang responsif untuk keperluan pengujian aplikasi.
c. Lingkungan Terpadu untuk Semua Perangkat Android: Memungkinkan pengembangan aplikasi yang kompatibel dengan berbagai jenis perangkat Android dalam satu lingkungan.
d. Terapkan Perubahan (Apply Changes): Fitur yang memungkinkan pengembang untuk mendorong perubahan kode dan resource ke aplikasi yang sedang berjalan tanpa perlu memulai ulang aplikasi.
e. Template Kode dan Integrasi GitHub: Fitur ini mempermudah pembuatan fitur aplikasi umum secara cepat, memungkinkan impor kode sampel, serta mendukung kolaborasi melalui integrasi dengan GitHub

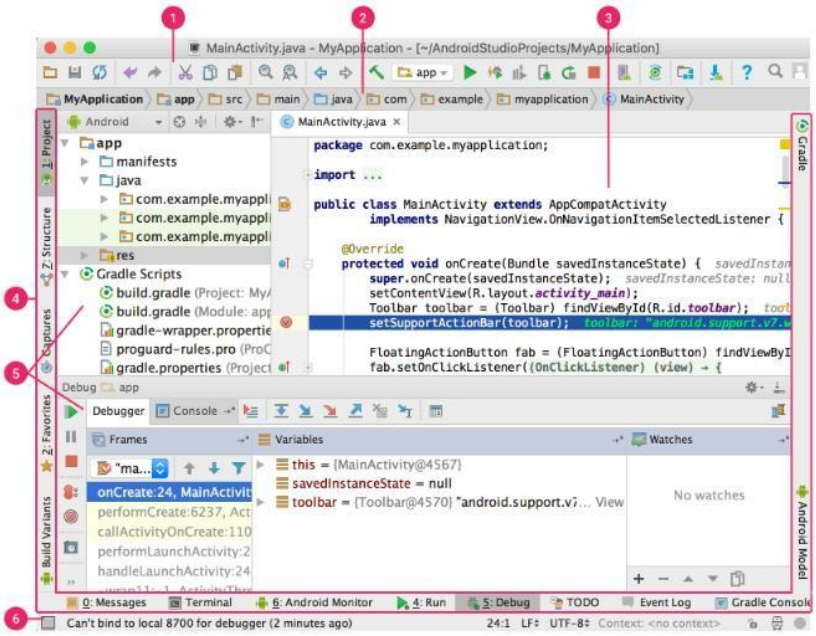
f. Framework dan Alat Pengujian yang Lengkap: Menyediakan berbagai alat untuk melakukan pengujian unit, instrumentasi, dan UI.

g. Alat Lint: Digunakan untuk merekam dan mengidentifikasi masalah performa, kegunaan, kompatibilitas versi, dan masalah potensial lainnya dalam kode.

h. Dukungan C++ dan NDK: Memungkinkan pengembangan aplikasi Android menggunakan kode C/C++ melalui Native Development Kit (NDK).

i. Integrasi Google Cloud Platform: Fitur ini menyederhanakan proses penggabungan layanan Google Cloud, seperti Google Cloud Messaging dan App Engine, ke dalam aplikasi Android.

Jendela utama *Android Studio* terdiri dari beberapa area logis yang diidentifikasi dalam gambar dibawah:



Gambar 2. 2. Jendela Utama Android Studio

j. Toolbar: Memungkinkan pengguna untuk melakukan berbagai tindakan, termasuk menjalankan aplikasi dan meluncurkan alat Android.

k. Menu Navigasi: Membantu menjelajahi proyek dan membuka file untuk diedit, serta memberikan tampilan struktur yang lebih ringkas dari yang terlihat di jendela Project.

l. Jendela Editor: Merupakan tempat utama untuk membuat dan memodifikasi kode.

m. Editor Adaptif: Bergantung pada jenis file yang sedang dibuka, editor ini dapat berubah; misalnya, saat menampilkan file tata letak, editor akan menampilkan Layout Editor.

n. Panel Jendela Fitur: Berada di sisi luar jendela IDE dan berisi tombol-tombol yang memungkinkan pengguna memperluas atau menciutkan setiap jendela fitur.

o. Jendela Fitur: Memberikan akses ke tugas-tugas tertentu seperti pengelolaan proyek, penelusuran, kontrol versi, dan lainnya, serta dapat diperluas dan diciutkan.

p. Status Bar: Menampilkan status proyek dan IDE itu sendiri, serta semua peringatan atau pesan yang relevan. (Seng Hansun, 2018).

5. Firebase

Firebase Realtime Database merupakan basis data yang di-host, di mana data disimpan dalam format JSON dan disinkronkan secara real-time ke semua klien yang terhubung. Saat mengembangkan aplikasi lintas-platform

menggunakan SDK Android, iOS, dan JavaScript, semua klien akan berbagi satu instance Realtime Database dan secara otomatis menerima pembaruan data terbaru.

Kemampuan Utama :

a. RealTIme

Sebagai ganti permintaan HTTP biasa, Firebase Realtime Database menggunakan sinkronisasi data—setiap kali data berubah, semua perangkat yang terhubung akan menerima update dalam waktu milidetik. Memberikan pengalaman yang kolaboratif dan imersif tanpa perlu memikirkan kode jaringan.

b. Offline

Aplikasi Firebase tetap responsif bahkan saat offline karena SDK Firebase Realtime Database menyimpan data ke disk. Setelah konektivitas pulih, perangkat klien akan menerima setiap perubahan yang terlewat dan melakukan sinkronisasi dengan status server saat ini.

c. Dapat Diakses dari Perangkat Klien

Firebase Realtime Database dapat diakses secara langsung dari perangkat seluler atau browser web; server aplikasi tidak diperlukan. Keamanan dan validasi data dapat diakses melalui Aturan Keamanan Firebase Realtime Database yang merupakan kumpulan aturan berbasis ekspresi dan dijalankan ketika data dibaca atau ditulis.

d. Menskalakan di beberapa Database

Dengan Firebase Realtime Database pada paket harga Blaze, Anda dapat mendukung kebutuhan data aplikasi Anda pada skala tertentu dengan membagi

data Anda di beberapa instance database di project Firebase yang sama. Menyederhanakan autentikasi dengan Firebase Authentication pada project Anda dan mengautentikasi pengguna di instance database Anda. Mengontrol akses ke data di tiap database dengan Aturan Firebase Realtime Database khusus untuk tiap instance database. (Rosyana Fitria Purnomo, 2020)

6. UML (Unified Modelling Language)

Unified Modelling Language (UML) merupakan bahasa standar industri yang dipakai untuk memvisualisasikan, merancang, serta mendokumentasikan sistem perangkat lunak. UML menyediakan kerangka standar untuk memodelkan sebuah sistem (Sora, 2018).

Sebagai sekumpulan alat abstraksi untuk sistem perangkat lunak berorientasi objek, UML, atau Unified Modeling Language, juga berfungsi mempermudah pengembangan aplikasi yang berkelanjutan. Tanpa dokumentasi UML, pengembangan dapat terhambat karena pengembang harus menelusuri dan mempelajari kode program secara ekstensif. Oleh karena itu, UML berfungsi sebagai sarana penting dalam proses transfer pengetahuan produk teknologi yang dikembangkan, tidak hanya antar pengembang tetapi juga kepada pihak non-teknis seperti pebisnis, sehingga pemahaman sistem dapat dicapai oleh siapa pun.

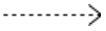
UML pertama kali diperkenalkan oleh Object Management Group pada Januari 1997 dengan versi 1.0. Dalam pengembangan berorientasi objek, beberapa prinsip dasar yang erat kaitannya dengan UML meliputi Object, Class, Abstraction

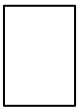
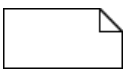
Encapsulation, Inheritance, dan Polymorphism. UML sendiri mencakup beberapa jenis diagram, yaitu:

a. Use Case Diagram

Use Case Diagram merujuk pada salah satu tipe diagram dalam UML yang berfungsi untuk memvisualisasikan fungsionalitas yang diharapkan dari suatu sistem. Diagram ini berfokus pada tujuan atau hasil yang dicapai sistem, bukan pada detail implementasi internalnya. Setiap *Use Case* menggambarkan interaksi spesifik antara pengguna (aktor) dengan sistem guna menyelesaikan tugas tertentu, seperti proses masuk ke akun atau pembuatan daftar belanja. Sementara itu, Individu atau kelompok sebagai entitas aktor yang berinteraksi dengan sistem untuk menjalankan tugas-tugas tersebut. Detail mengenai simbol-simbol yang digunakan dalam *Use Case* Diagram, yaitu:

Tabel 2. 1. Simbol *Use Case Diagram*

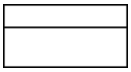
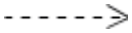
No	Gambar	Nama	Keterangan
1		*Actor*	Menguraikan kumpulan peran yang diemban oleh pengguna saat mereka berinteraksi dengan suatu *Use Case*.
2		*Dependency*	Hubungan ini menjelaskan bahwa setiap perubahan pada komponen yang mandiri (independen) akan memengaruhi komponen yang terikat atau tidak mandiri (dependen).

3		*Generalization*	hubungan di mana objek anak (*descendent*) mewarisi perilaku dan struktur data dari objek induk (*ancestor*).
4		*Include*	mendefinisikan secara jelas bahwa *Use Case* sumber (asal) disertakan dalam *Use Case* lain.
5		*Extend*	mengindikasikan bahwa *Use Case* target menambahkan perilaku tambahan ke *Use Case* sumber pada titik yang spesifik.
6		*Association*	Mengkoneksikan objek satu dengan objek lainnya. .
7		*System*	Menguraikan paket atau modul yang menyajikan sistem dengan fungsionalitas yang terbatas.
8		*Use Case*	Serangkaian tindakan yang dilakukan oleh sistem, yang pada akhirnya menghasilkan luaran terukur bagi aktor yang berinteraksi.
9		*Collaboration*	Bagaimana aturan dan berbagai elemen saling berinteraksi dan bekerja sama untuk menciptakan perilaku atau hasil yang lebih besar daripada gabungan masing-masing bagiannya (sinergi).
10		*Note*	Elemen fisik yang ada dan beroperasi saat aplikasi dijalankan, yang merepresentasikan sumber daya komputasi.

b. Class Diagram

Adapun simbol-simbol *Class Diagram* antara lain :

Tabel 2. 2. Simbol *Class Diagram*

No	Gambar	Nama	Keterangan
1		*Generalization*	hubungan di mana objek turunan (*descendent*) mewarisi perilaku dan struktur data dari objek leluhurnya (*ancestor*).
2		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4		*Collaboration*	Serangkaian tindakan yang dilakukan oleh sistem, yang pada akhirnya menghasilkan luaran terukur bagi aktor yang berinteraksi.
5		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
6		*Dependency*	Hubungan di mana perubahan pada suatu elemen yang mandiri (independen) akan memengaruhi elemen lain yang bergantung padanya (tidak mandiri).
7		*Association*	Menghubungkan objek satu dengan objek lainnya. .

c. Activity Diagram

Adapun simbol-simbol *Activity Diagram* antara lain :

Tabel 2. 3. Simbol *Activity Diagram*

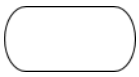
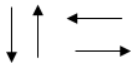
No	Gambar	Nama	Keterangan
1		*Activity*	Menunjukkan bagaimana setiap kelas antarmuka saling berinteraksi satu sama lain.
2		*Action*	kondisi sistem yang merefleksikan berlangsungnya atau selesainya eksekusi suatu tindakan
3		*Initial Node*	Bagaimana objek dibangun atau dimulai
4		*Activity Final Node*	Bagaimana objek dibangun dan dihancurkan
5		*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

d. Flowchart

Flowchart, atau diagram alir, adalah representasi grafis dari langkah-langkah yang harus diikuti dalam menyelesaikan suatu masalah. Flowchart melukiskan aliran kegiatan dari awal hingga akhir. Dalam konteks skripsi Anda, flowchart sangat berguna untuk menggambarkan alur kerja sistem administrasi desa yang akan Anda bangun. Hal ini akan memudahkan Anda dan pembaca untuk melihat setiap tahapan, mulai dari penerimaan data (input), pemrosesan data, pengambilan keputusan (misalnya, persetujuan kepala desa), hingga pencetakan dokumen (output). (Ade Hastuty, 2023)

Berikut ini adalah beberapa simbol yang digunakan dalam menggambar suatu *Flowchart* (Ade Hastuty, 2023):

Tabel 2. 4. Simbol *Flowchart*

Simbol	Nama	Fungsi
	Simbol Terminator	Berfungsi menandai awal dan akhir dari suatu flowchart
	Simbol Garis Alir	Simbol ini di gunakan untuk menghubungkan setiap langkah dalam flowchart dan menunjukkan kemana arah aliran diagram.
	Simbol Proses	Digunakan untuk melambangkan kegiatan pemrosesan input.
	Simbol Percabangan	Merupakan simbol yang digunakan untuk melambangkan percabangan, yaitu pemeriksaan terhadap suatu kondisi.
	Simbol Input	Merupakan simbol yang digunakan untuk melambangkan kegiatan penerimaan input.
	Simbol Prosedur	Simbol ini berperan sebagai blok pembangun dari suatu program.
	Simbol *On Page Connector*	Simbol ini fungsinya adalah untuk menghubungkan suatu langkah dengan langkah lain dari flowchart dalam satu halaman.
	Simbol *Off-page Connector*	Simbol ini fungsinya adalah untuk menghubungkan suatu langkah dengan langkah lain dari flowchart dalam halaman yang berbeda

7. Algoritma

Algoritma adalah urutan langkah-langkah penyelesaian masalah yang disusun secara sistematis dan logis. Kata logis merupakan kata kunci dalam algoritma. Langkah-langkah dalam algoritma harus dapat di tentukan dan bernilai salah atau benar. (Ade Hastuty, 2023)

Pertimbangan dalam memilih algoritma yaitu:

a. Algoritma haruslah benar. Artinya algoritma akan memberikan keluaran yang dikehendaki dari sejumlah masukan yang diberikan.
b. Mengetahui seberapa baik hasil yang dicapai oleh algoritma tersebut. Hal ini penting terutama pada algoritma untuk menyelesaikan masalah yang memerlukan aproksimasi hasil (hasil yang berupa pendekatan)
c. Efisiensi algoritma. Efisiensi algoritma dapat ditinjau dalam 2 hal yaitu efisiensi waktu dan memori. (Ade Hastuty, 2023)

Algoritma adalah jantung ilmu komputer atau informatika. Banyak cabang ilmu komputer yang mengarah ke dalam terminologi algoritma. Komputer adalah salah satu pemroses. Agar dapat dilaksanakan oleh komputer, algoritma harus ditulis dalam notasi bahasa pemrograman sehingga dinamakan program. Jadi program adalah perwujudan atau implementasi teknis algoritma yang ditulis dalam bahasa pemrograman tertentu sehingga dapat dilaksanakan oleh komputer.

Algoritma adalah langkah-langkah penyelesaian masalah, sedangkan program adalah realisasi algoritma dalam bahasa pemrograman. Program ditulis dalam bahasa pemrograman dan kegiatan membuat program disebut pemrogram

(*programming*). Orang yang menulis program disebut pemrogram (*programmer*). Tiap-tiap langkah dalam program disebut pernyataan atau instruksi. Jadi, program tersusun atas sederetan instruksi. (Ade Hastuty, 2023)

B. Kajian Hasil Penelitian Terdahulu

Melihat dari sejumlah judul dan tema yang berkaitan dengan penelitian yang dilakukan, diperoleh persamaan dan perbedaan dari penelitian yang akan dilakukan. Tujuannya membuktikan bahwa penulisan penelitian asli dan bukan duplikasi dari penelitian lain, seperti berikut ini :

d. Andi Susanto, 2020. *"Rancang Bangun Aplikasi Administrasi Desa Sebagai Software As A Service Pada Cloud E-Desa (Studi Kasus Beberapa Desa Di Kecamatan Cimenyan)"*. Pemanfaatan *Cloud Computing* memungkinkan pengembangan sistem bernama E-Desa, yang salah satu fiturnya adalah aplikasi administrasi kependudukan desa. Aplikasi ini menyediakan berbagai layanan permohonan surat seperti Surat Kartu Keluarga (SKK), Surat Kartu Tanda Penduduk (SKTP), Surat Keterangan Pindah (SKP), Surat Keterangan Kelahiran (SKKL), Surat Keterangan Kematian (SKKM), Surat Keterangan Kelakuan Baik (SKKB), Surat Keterangan Serbaguna (SKSG), Surat Keterangan Usaha (SKU), Surat Keterangan Nikah (SKNA), dan Surat Keterangan Tidak Mampu (SKTM). Untuk dapat menggunakan layanan ini, pemerintah desa perlu berlangganan melalui penyedia layanan *Cloud Provider*. Berdasarkan hasil pengujian *alpha* dan *beta*, dapat disimpulkan bahwa aplikasi administrasi kependudukan desa sebagai *Software as a Service* (SaaS) Cloud E-Desa ini memberikan kemudahan

dalam pengolahan data penduduk, menekan biaya operasional desa, dan secara signifikan mengoptimalkan pelayanan administrasi kepada masyarakat desa.

e. Hidayatul Rahman dan Wagino, 2021. *"Aplikasi Pelayanan Administrasi Kependudukan Dan Penetapan Apbdes Berbasis Web Pada Desa Kampung Baru"*. Penelitian ini berfokus pada pengembangan aplikasi web untuk Pelayanan Administrasi Kependudukan dan Penetapan APBDes di Desa Kampung Baru. Proses pengembangan aplikasi ini mengadopsi metode *waterfall*, yang meliputi tahapan analisis kebutuhan, desain sistem, penulisan kode program, penerapan, dan pengujian program. Untuk perancangan sistem, digunakan *Unified Modelling Language* (UML) dengan berbagai diagram seperti *use case diagram*, *class diagram*, *activity diagram*, dan *sequence diagram*. Aplikasi ini dibangun menggunakan bahasa pemrograman PHP Native dan didukung oleh basis data MySQL. Adapun perangkat lunak yang digunakan dalam pengembangan meliputi XAMPP sebagai *web server* lokal dan Sublime Text sebagai *code editor*.

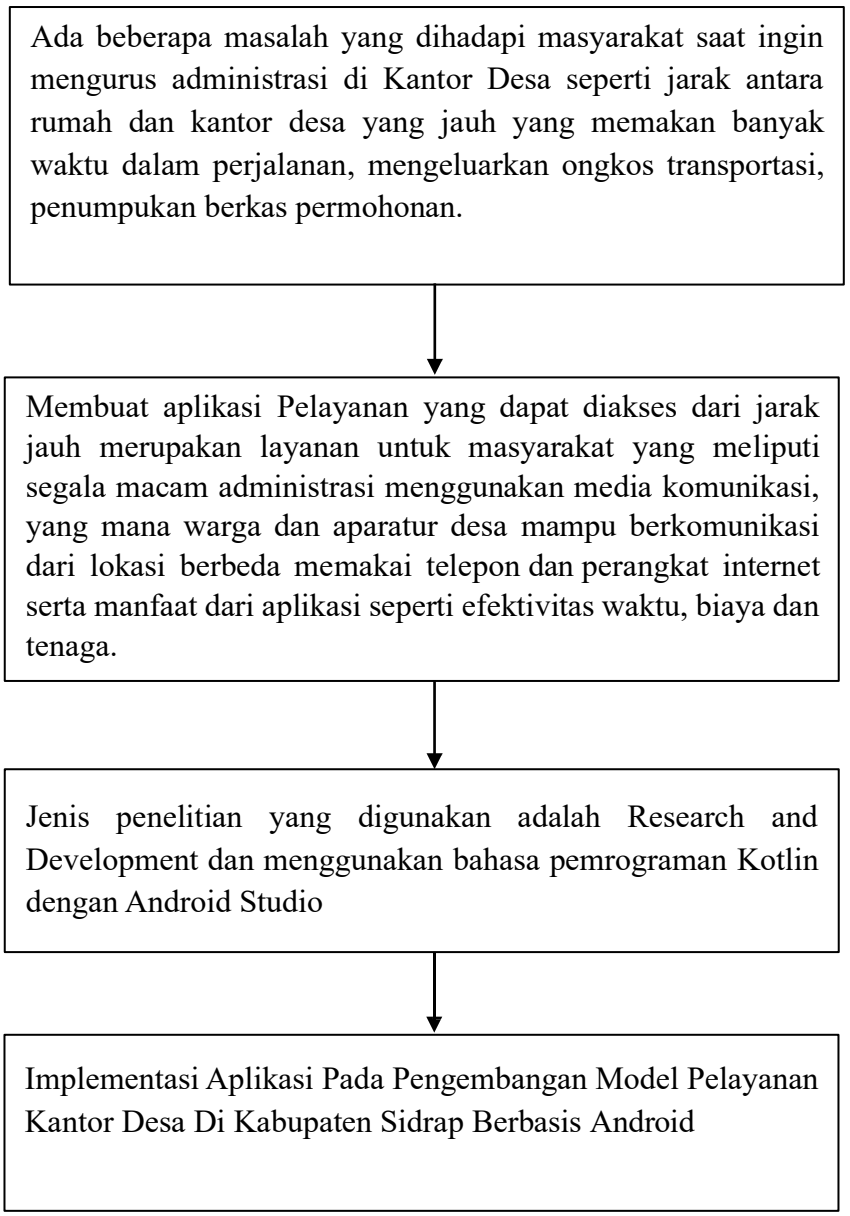
f. Muhammad Basuni Imron, 2020. *"Perancangan Aplikasi Sistem Informasi Administrasi Desa Berbasis Android"*. Penelitian ini berfokus pada studi kasus di Kantor Desa Bandungbaru, Kabupaten Pringsewu. Saat ini, penyelenggaraan pelayanan informasi administrasi di desa tersebut masih belum menggunakan sistem terotomatisasi, yang mengakibatkan proses pelayanan kurang efisien. Oleh karena itu, diperlukan sebuah sistem aplikasi informasi administrasi yang mampu mengatasi permasalahan tersebut.

Tujuan dari pembuatan aplikasi Sistem Informasi Administrasi ini adalah untuk membantu memberikan alternatif pemecahan masalah di Desa Bandungbaru. Model pengembangan aplikasi yang digunakan adalah metode *waterfall*, dengan pemodelan menggunakan *tools Unified Modeling Language* (UML). Perangkat lunak ini dibangun menggunakan bahasa pemrograman Java, dan hasilnya berupa sebuah prototipe aplikasi Android. Aplikasi ini dirancang untuk dapat memberikan informasi, seperti pembuatan surat pengantar yang dikeluarkan oleh pihak Desa. Dengan adanya aplikasi ini, diharapkan pelayanan administrasi oleh Pemerintah Desa Bandungbaru kepada masyarakat dapat dilakukan dengan lebih baik, lebih cepat, dan lebih mudah.

Berdasarkan beberapa penelitian yang telah dilakukan maka dilakukan penelitian lebih lanjut untuk dapat membuat sistem dengan judul **"***Rancang Bangun Pelayanan Administrasi Kantor Desa Kecamatan Watang Pulu Kabupaten Sidrap*". Pada penelitian ini terdapat perbedaan dengan kajian terdahulu, yaitu objek penelitian, tahun penelitian dan pembahasan tentang pelayanan administrasi di Kantor Desa berbasis *Android* yang mengimplemantasikan teknologi cloud computing. Bahasa pemograman yang digunakan mudah dimengerti dan mudah diakses dimana saja jika terhubung jaringan internet yaitu dengan bahasa pemrograman Kotlin.

C. Kerangka Pikir

Untuk lebih memperjelas kerangka pikir maka digambarkan dalam bentuk diagram sebagai berikut :



Gambar 2. 3. Kerangka Pikir

BAB III

METODE PENELITIAN

A. Lokasi Dan Waktu Penelitian

Penelitian dilaksanakan di kantor Desa yang terdapat di Kecamatan Watang Pulu Kabupaten Sidrap, tepatnya pada Desa Lainungan. Populasi dalam penelitian ini mencakup seluruh masyarakat Desa Lainungan, Kecamatan Watang Pulu, serta seluruh perangkat desa atau staf administrasi yang terlibat dalam pengelolaan sistem pelayanan. Sampel penelitian diambil sebagai representasi dari populasi tersebut, yang terdiri dari beberapa warga desa yang pernah melakukan layanan administrasi desa, serta perwakilan perangkat desa. Pengambilan sampel ini bertujuan untuk memperoleh data yang relevan sebagai dasar dalam perancangan sistem pelayanan yang dikembangkan.

Waktu yang dipergunakan untuk penelitian ini 3 (tiga) bulan dimulai Mei sampai dengan Juli. Studi literatur dan pembuatan aplikasi dilaksanakan pada bulan Mei, pengujian aplikasi dilaksanakan pada bulan Juni dan Pengolahan Hasil dan penyusunan laporan akan diolah pada bulan Juli. Berikut adalah tabel jadwal pelaksanaan penelitian:

Tabel 3. 1. Jadwal Pelaksanaan Penelitian

No.	**Tahapan Penelitian**	**Uraian Kegiatan**	Tahun 2025		
			Mei	Juni	Juli
1	Studi Literatur	Studi mendalam terkait sistem informasi, *cloud computing*, Android, *Kotlin*, dan administrasi desa			

2	Pembuatan Aplikasi	pengembangan kode aplikasi Android menggunakan bahasa pemrograman *Kotlin* di Android Studio			
3	pengujian aplikasi	black box testing, white box testing, serta perbaikan *bug* dan *error*			
4	Pengolahan Hasil & Penyusunan Laporan	Analisis dan pengolahan data hasil pengujian aplikasi, Penulisan draf skripsi, Revisi dan finalisasi skripsi berdasarkan masukan pembimbing			

B. Jenis Penelitian

Jenis penelitian yang digunakan adalah *Research and Development*, yaitu penelitian yang dilakukan untuk mengembangkan aplikasi tertentu, serta menguji coba keefektifan aplikasi yang dimaksud. Penelitian ini akan mengimplementasi *cloud computing* pada pengembangan model pelayanan kantor desa di Kabupaten Sidrap berbasis android, langkah-langkah dari pembuatan aplikasi ini biasanya disebut dengan siklus *Research and Development* yang terdiri dari beberapa langkah diantaranya menganalisis aplikasi yang dikembangkan, tahap pengembangan, pengujian produk secara menyeluruh, diikuti tahap revisi dan evaluasi yang bertujuan untuk memperbaiki setiap kelemahan yang ditemukan pada aplikasi.

C. Metode Pengumpulan Data

Pengumpulan data dilakukan untuk dijadikan dasar dalam merancang aplikasi yang akan dibangun (Sudarman Hamzah, dkk, 2025)

1. Observasi.

   Metode observasi dilakukan dengan pengamatan secara langsung dan

sistematis terhadap aktivitas pelayanan administrasi di Kantor Desa pada Kecamatan Watang Pulu, Kabupaten Sidrap.

Observasi dilakukan untuk mengamati tahapan yang dilalui masyarakat dalam mengajukan permohonan surat-surat administrasi, mulai dari kedatangan, pengisian formulir, hingga penyerahan berkas serta mencatat secara langsung permasalahan yang sering muncul, seperti antrean panjang, penumpukan berkas fisik, durasi waktu tunggu, serta potensi hambatan komunikasi antara petugas dan masyarakat.

2. Wawancara.

Wawancara dalam penelitian ini dilakukan untuk menganalisis kebutuhan Aplikasi Pelayanan Kantor Desa Berbasis Android di Kecamatan Watang Pulu Kabupaten Sidrap. Wawancara ini dilakukan kepada perangkat desa serta masyarakat desa yang menggunakan layanan administrasi desa.

Wawancara kepada masyarakat dilakukan untuk mengetahui kesulitan yang dihadapi misalnya terkait jarak tempuh, biaya transportasi, atau akses informasi, serta harapan mereka terhadap kemudahan dan efisiensi pelayanan melalui aplikasi digital.

3. Kajian Kepustakaan.

Penelitian kepustakaan adalah pendekatan pengumpulan data dengan cara mencari, meninjau, dan menganalisis literatur tertulis yang berkaitan dengan subjek penelitian. Data ini meliputi buku, publikasi ilmiah, artikel, serta referensi dari internet yang memiliki relevansi dengan penelitian ini.

D. Alat Dan Bahan Penelitian

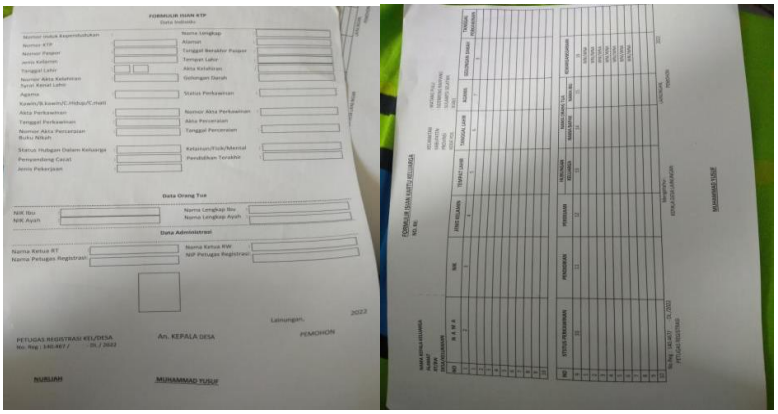
Dalam pelaksanaan penelitian ini digunakan beberapa alat dan bahan yang terdiri:

1. Alat penelitian
    a. Personal Computer dengan spesifikasi :
        1) *Processor : Intel Core i3 gen 7*
        2) *Memory : 4 GB*
        3) *HDD : 1 TB*
        4) *Monitor : 15 Inci*

software (Perangkat lunak) yang dipergunakan saat membangun aplikasi ini dapat dilihat pada spesifikasi seperti Windows 10 x64, *Android Studio.*

2. Bahan penelitian

Bahan penelitian berupa data. Data diperoleh dari Desa Lainungan di Kecamatan Watang Pulu.



Gambar 3. 1. Surat Pengantar Pembuatan KK dan KTP

E. Tahap Penelitian

Tahap penelitian dilakukan dengan cara :

1. Persiapan Penelitian

Pada tahapan ini peneliti melakukan persiapan penelitian. Persiapan penelitian yang dimaksud adalah menyiapkan buku-buku, artikel tentang topik penelitian serta *software* yang digunakan selama penelitian.

Pengumpulan Data

Pada tahap ini penelitian melakukan observasi dengan peninjauan, pencatatan dan pengamatan data di Kantor Desa Kabupaten Sidrap.

2. Analisis

Pada tahap analisis, peneliti melakukan analisa terhadap sistem dan data pada Kantor Desa yang ada kemudian merumuskan masalah yang menjadi pokok penelitian sehingga dapat dibuat alternatif pemecahan masalah.

3. Perancangan

Peneliti kemudian merancang aplikasi pelayanan desa dengan mengimplementasikan *cloud computing* berbasis *Android* yang ingin dibuat berdasarkan pemecahan masalah.

4. Pengujian

Setelah melakukan perancangan, peneliti kemudian menguji hasil perancangan aplikasi. Apabila ditemukan kekurangan atau kelemahan dalam hasil perancangan, proses akan kembali ke fase analisis.

F. Metode Pengujian

Pengujian sistem yang dikembangkan dengan Android Studio dan bahasa pemrograman Kotlin dilakukan melalui mekanisme terencana, yang dikelompokkan berdasarkan jenis pengguna. Pengujian ini mencakup setiap proses yang ada pada sistem dan dilaksanakan melalui dua pendekatan utama:

1. Pengujian *White Box*: Metode ini berfokus pada pengujian cara kerja internal perangkat lunak, termasuk prosedur program (basis jalur) atau proses perulangan (*looping*). Penguji memiliki pengetahuan mendalam tentang kode sumber dan struktur internal sistem.
2. Pengujian *Black Box*: Metode ini dilakukan dengan hanya mengamati hasil eksekusi melalui data uji, tanpa melihat struktur internal kode. Tujuannya adalah untuk memeriksa fungsionalitas perangkat lunak dari sudut pandang pengguna.

G. Analisa Sistem

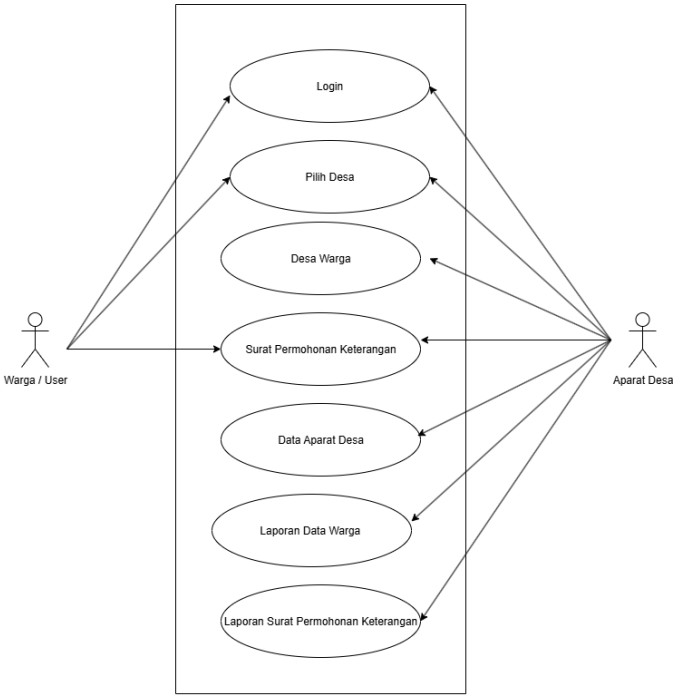
1. Sistem Yang Berjalan



Gambar 3. 2. Sistem Yang Berjalan

Pada sistem yang berjalan dapat dilihat dimana, aktivitas yang berjalan untuk mengambil surat keterangan di Kantor Desa yaitu warga menyetor berkas perlengkapan langsung kepada Aparat Desa, selanjutnya aparat desa mengecek berkas, menginput data pada surat keterangan dan mencetak beberapa lembar surat keterangan, selanjutnya diberikan kepada kepala desa untuk ditanda tangani, selanjutnya aparat desa mengarsip sebagian surat keterangan untuk disetor ke kantor kecamatan dan memberikan kepada warga surat keterangan tersebut.

2. Sistem Yang Diusulkan

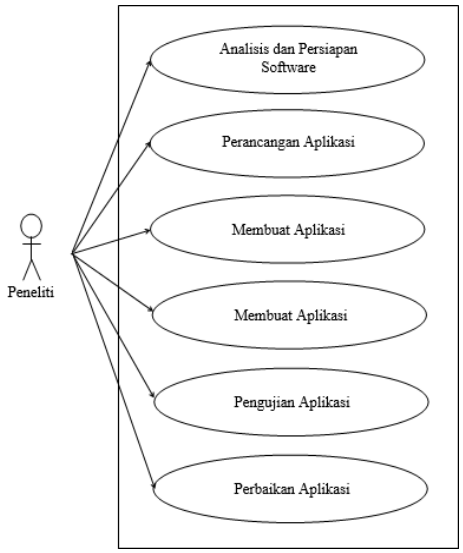


Gambar 3. 3. Sistem Yang Diusulkan

Pada sistem yang diusulkan sudah berupa sistem aplikasi yang dimana dilakukan secara *online. User* dalam hal ini sebagai warga tidak perlu lagi ke Kantor Desa untuk mengambil surat permohonan atau keterangan, *User* hanya perlu menyiapkan data pribadi untuk digunakan menginput data pada aplikasi dan berkas yang akan diupload sebagai ketentuan dan kemudian akan mendapat surat permohonan atau keterangan yang dia inginkan, sebelum ingin membuat surat permohonan atau keterangan *User* sudah mengetahui ketentuan dan syarat yang telah diberikan oleh Kantor Desa. *Admin* dalam hal ini aparat desa dimana berfungsi dalam mengelola data pada aplikasi, seperti mengelola data warga, data aparat desa, data surat dan juga data laporan.

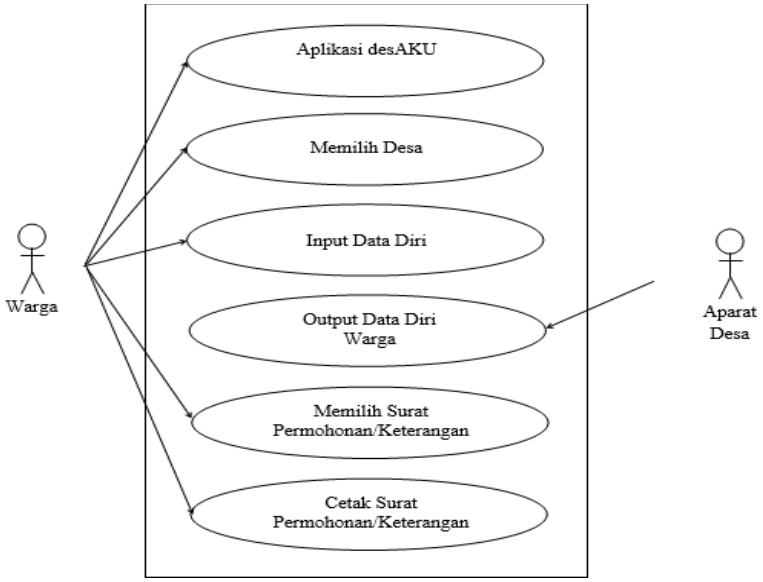
H. Desain Sistem

1. *Use Case* Penelitian



Gambar 3. 4. *Flowchart* Penelitian

2. *Use Case* Surat Permohonan/Keterangan



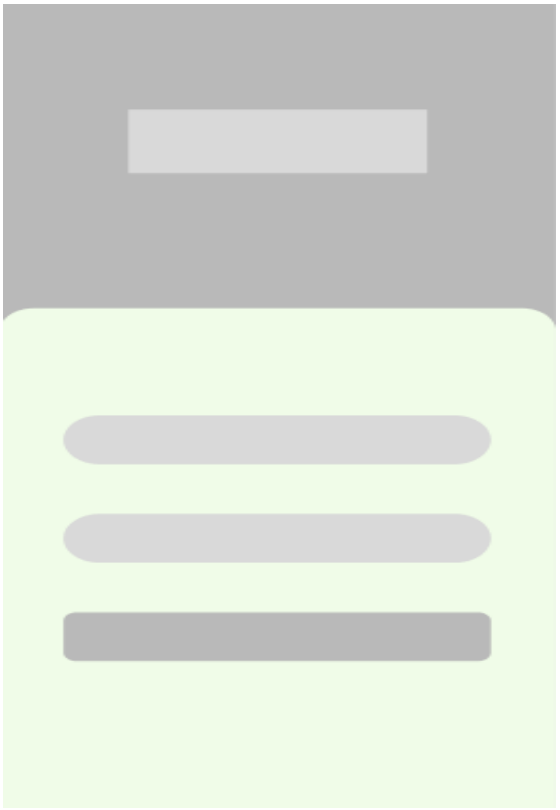
Gambar 3. 5. *Use Case* Surat Permohonan/Keterangan

I. Rancangan Sistem

Pada tahap ini akan diterapkan rancangan desain antar muka atau interface pada halaman-halaman yang ada pada sistem. Adapun rancangan desain interface-nya adalah sebagai berikut:

a. Tampilan halaman *home*

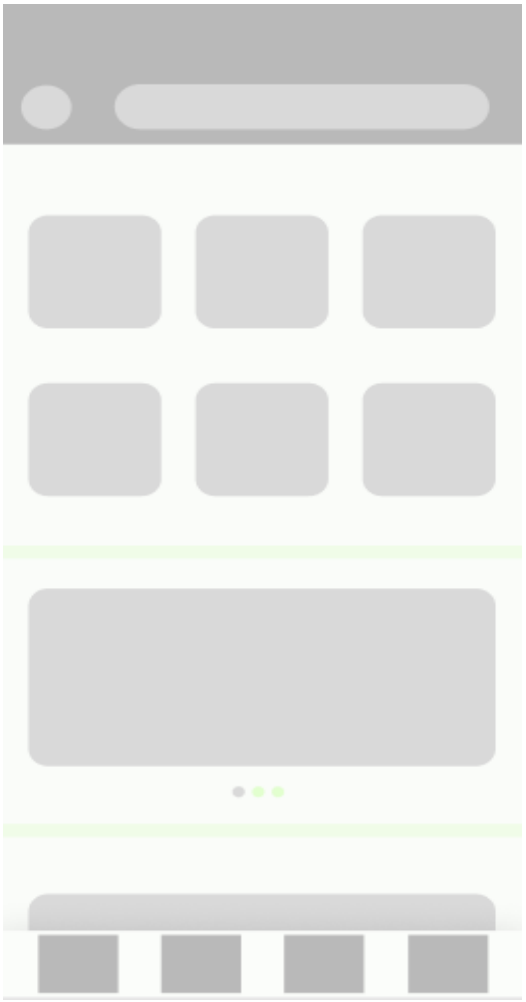
Halaman *home* merupakan antarmuka awal yang akan ditampilkan kepada pengguna saat pertama kali membuka aplikasi. Tampilan visual dari halaman ini dirancang untuk memberikan kesan pertama dan mengarahkan pengguna ke fungsionalitas utama aplikasi.



Gambar 3. 6. Halaman *Login* (Admin)

b. Halaman Home *admin*

Gambar dibawah ini merupakan desain Home halaman beranda awal dari Admin setelah login , selain itu terdapat juga menu– menu dan sub menu yang terdapat pada perancangan ini.



Gambar 3. 7. Halaman *Home* (Admin)

BAB IV

HASIL PENELITIAN DAN PEMBAHASAN

A. Hasil Penelitian

Selama pelaksanaan penelitian di kantor Desa Kecamatan Watang Pulu, Kabupaten Sidrap, yang bertujuan untuk membangun aplikasi sistem informasi pelayanan administrasi kantor desa, diperoleh hasil penelitian, di antaranya yaitu jenis administrasi dan identifikasi persyaratan administrasi pembuatan dokumen yang akan menunjang kelancaran penyelesaian aplikasi berbasis Android ini.

Tabel 4. 1. Laporan Jenis Administrasi

No	Jenis Administrasi	Keterangan
1	Surat pengantar keterangan nikah	Surat resmi yang dikeluarkan oleh kantor desa atau kelurahan sebagai syarat awal bagi seseorang untuk melangsungkan pernikahan. Surat ini menjadi bukti bahwa warga yang bersangkutan benar-benar berdomisili di wilayah tersebut dan akan menikah.
2	Surat pengantar keterangan lahir	Dokumen resmi yang dikeluarkan oleh Ketua Rukun Tetangga (RT) serta Ketua Rukun Warga (RW) setempat, kemudian disahkan oleh kelurahan atau kantor desa, yang menyatakan bahwa seseorang telah lahir di wilayah tersebut. Surat ini menjadi syarat utama dan dasar bagi orang tua untuk membuat Akta Kelahiran di Kantor Dinas Kependudukan dan Pencatatan Sipil (Dukcapil).
3	Surat pengantar keterangan usaha	Dokumen resmi yang dikeluarkan oleh pemerintah desa atau kelurahan setempat sebagai bukti dan pengakuan awal bahwa seseorang atau kelompok memiliki kegiatan usaha yang berlokasi di wilayah tersebut.

4	Surat pengantar keterangan tidak mampu	Dokumen resmi yang dikeluarkan oleh kantor desa atau kelurahan yang menyatakan bahwa seseorang atau keluarga memiliki kondisi ekonomi yang kurang mampu (miskin/prasejahtera).
5	Surat pengantar keterangan akte kematian	Dokumen resmi yang dikeluarkan oleh kantor desa atau kelurahan setempat sebagai syarat awal dan dasar untuk mengurus penerbitan Akta Kematian di Dinas Kependudukan dan Pencatatan Sipil (Dukcapil). Surat ini memverifikasi bahwa seseorang telah meninggal dunia di wilayah administratif desa/kelurahan tersebut.
6	Surat pengantar keterangan pindah	Dokumen resmi yang dikeluarkan oleh kantor desa atau kelurahan asal, yang menyatakan bahwa seorang penduduk akan pindah domisili dari wilayah tersebut ke wilayah lain. Surat ini berfungsi sebagai bukti sah bahwa penduduk tersebut telah keluar dari catatan kependudukan desa/kelurahan lama dan menjadi syarat utama untuk dapat mengurus perpindahan dan pendaftaran diri di desa/kelurahan atau Disdukcapil di tempat tujuan.
7	Surat pengantar keterangan izin keramaian	Dokumen resmi yang dikeluarkan oleh kantor desa atau kelurahan yang menyatakan bahwa akan ada suatu kegiatan keramaian di wilayah tersebut, dan merupakan pengantar awal untuk mengurus izin keramaian lebih lanjut kepada pihak berwenang, biasanya kepolisian. Surat ini berfungsi sebagai pemberitahuan dan permohonan rekomendasi dari tingkat pemerintahan terendah sebelum mengajukan izin resmi ke instansi yang lebih tinggi.
8	Surat pengantar domisili	Dokumen resmi yang dikeluarkan oleh kantor desa atau kelurahan yang menyatakan dan mengesahkan alamat tempat tinggal atau kedudukan resmi seseorang, kelompok, atau suatu lembaga/usaha berada di wilayah administratif tersebut. Surat ini berfungsi sebagai bukti formal untuk berbagai keperluan yang membutuhkan validasi alamat tinggal/berkedudukan.

Tabel 4. 2. Laporan persyaratan administrasi

No	Jenis Surat	Persyaratan berkas
1	Surat pengantar keterangan nikah	a. KTP asli dan fotokopi b. Kartu keluarga c. Surat pengantar dari RT/RW d. Akta kelahiran asli/fotokopi e. Foto
2	Surat pengantar keterangan lahir	a. KTP orang tua (asli dan fotokopi) b. Kartu Keluarga (asli dan Fotokopi) c. Surat keterangan kelahiran dari bidan atau rumah sakit
3	Surat pengantar keterangan usaha	a. KTP (Asli dan fotokopi) b. Kartu Keluarga (asli dan fotokopi) c. Surat keterangan dari RT/RW d. Foto e. Bukti kepemilikan usaha
4	Surat pengantar keterangan tidak mampu	a. KTP (asli dan fotokopi) b. Kartu Keluarga (asli dan fotokopi) c. Surat keterangan dari RW/RW d. Surat keterangan penghasilan e. Pas foto
5	Surat pengantar keterangan akte kematian	a. KTP (asli dan fotokopi) b. Kartu Keluarga (asli dan fotokopi) c. Surat keterangan kematian (dari rumah sakit,puskesmas) d. Surat keterangan dari RT/RW e. Foto almarhum
6	Surat pengantar keterangan pindah	a. KTP (asli & fotokopi) b. Kartu Keluarga (asli/ & fotokopi) c. Surat keterangan pindah dari tempat asal d. Foto
7	Surat pengantar keterangan izin keramaian	a. KTP (asli & Fotokopi) b. Kartu Keluarga (asli&fotokopi) c. Surat pengantar (RT/RW) d. Rencana Kegiatan (misalnya acara pernikahan, pesta, konser)
8	Surat pengantar domisili	a. KTP (asli&fotokopi) b. Kartu Keluarga (asli & fotokopi) c. Surat pengantar dari RT/RW d. Bukti kepemilikan tempat tinggal (rumah,sewa rumah,kontrakan) e. Foto

Dari data yang didapatkan hasil dari wawancara terdapat 2 poin penting. Laporan jenis berkas dan laporan persyaratan berkas. Data diatas digunakan untuk di input pada rancangan bangun pelayanan administrasi kantor desa di Kecamatan Watang Pulu Kabupaten Sidrap

B. Pembahasan

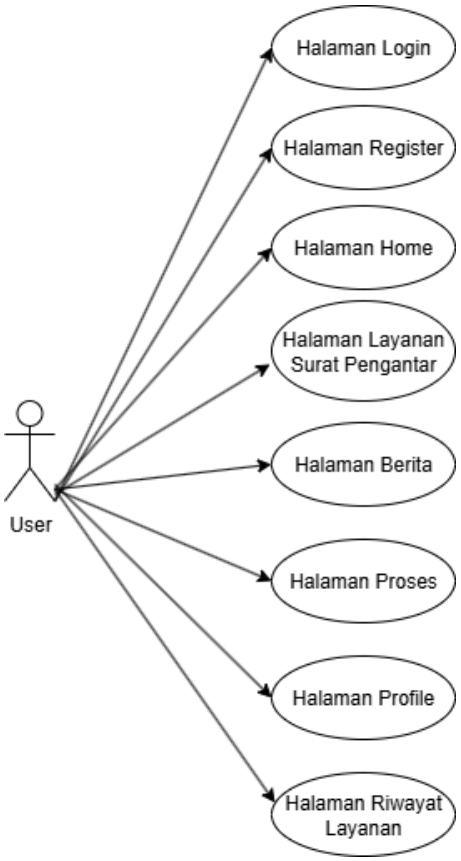
1. Analisa Sistem Yang Berjalan

Analisis sistem yang diusulkan menggunakan *Use Case Diagram*, *Activity Diagram*, *Sequence Diagram* dan rancangan database sebagai berikut:

a. Use Case Diagram

Use Case Diagram adalah pemodelan yang dipergunakan untuk mendekripsikan perilaku (behavior) sistem informasi yang sedang dibangun. Diagram ini mendeskripsikan hubungan antara satu atau lebih aktor dengan sistem informasi. Sebagai bentuk diagram, *Use Case* menggambarkan fungsionalitas yang diharapkan dari sebuah sistem dari perspektif pengguna di luar sistem.

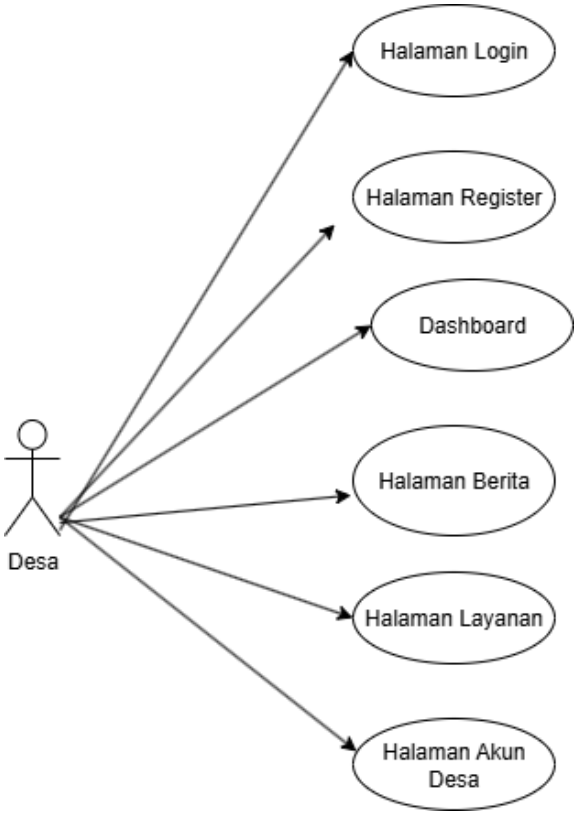
1) *User*



Gambar 4. 1. *Use case diagram user*

Diagram *Use case* diatas menjelaskan interaksi seorang Pengguna (*User*) dengan berbagai halaman atau modul dalam sistem aplikasi. Pengguna memiliki kemampuan untuk mengakses Halaman Login dan Halaman Register untuk otentikasi, kemudian dapat melanjutkan ke Halaman *Home* sebagai titik awal. Dari sana, Pengguna dapat mengakses Halaman Layanan Surat Pengantar untuk mengajukan permohonan, Halaman Berita untuk mendapatkan informasi terkini, Halaman Proses untuk memantau status permohonan, Halaman Profile untuk mengelola data pribadi, serta Halaman Riwayat Layanan untuk melihat catatan aktivitas sebelumnya.

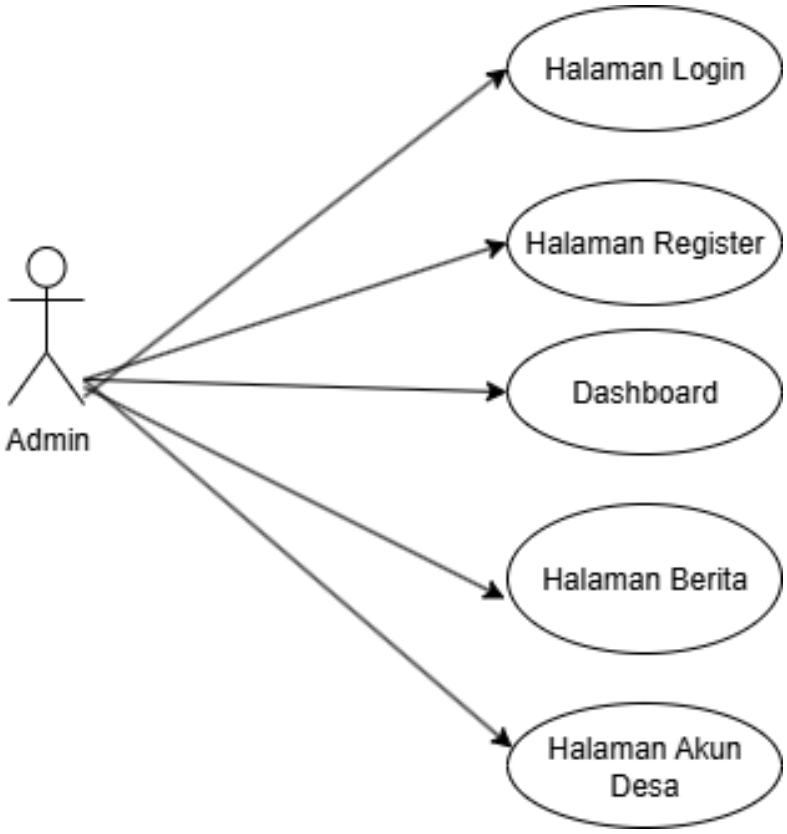
2) Desa



Gambar 4. 2. *Use case diagram* desa

Diagram *Use case* diatas menggambarkan interaksi Admin desa dengan berbagai halaman atau modul dalam sistem aplikasi. Admin Desa memiliki akses ke Halaman *Login* dan Halaman Register untuk otentikasi, kemudian dapat melanjutkan ke *Dashboard* sebagai pusat kendali. Dari sana, Aktor Desa dapat mengakses Halaman Berita untuk mengelola informasi, Halaman Layanan untuk memproses permohonan, dan Halaman Akun Desa untuk mengelola detail profil desa itu sendiri.

3) Admin



Gambar 4. 3. *Use case diagram* admin

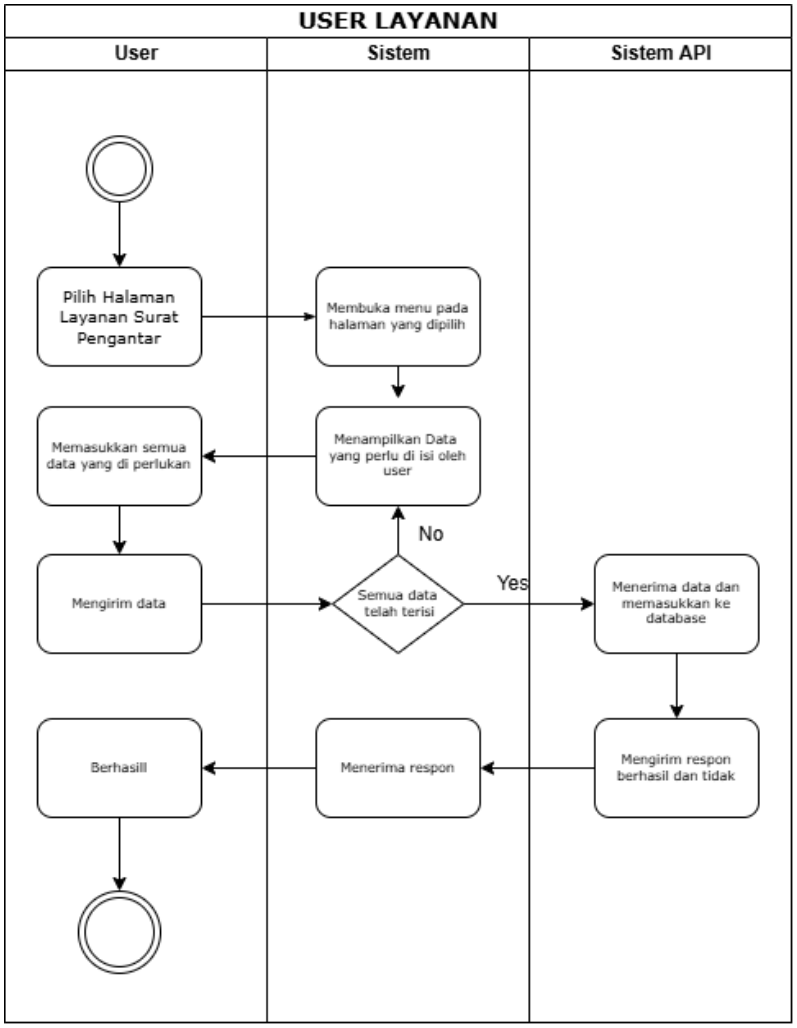
Diagram *Use case* diatas menggambarkan interaksi Admin dengan berbagai halaman atau modul dalam sistem aplikasi. Admin memiliki akses ke Halaman Login dan Halaman Register untuk otentikasi, kemudian dapat melanjutkan ke Dashboard sebagai pusat kendali. Dari sana, Admin dapat mengakses Halaman Berita untuk mengelola informasi, dan Halaman Akun Desa untuk mengelola detail profil semua desa.

b. Activity Diagram

Activity diagram ini mendeskripsikan mengenai aktifitas-aktifitas yang berlangsung dalam rangkaian aliran proses pada sebuah sistem.

1) Activity Diagram User

a) *User* Layanan

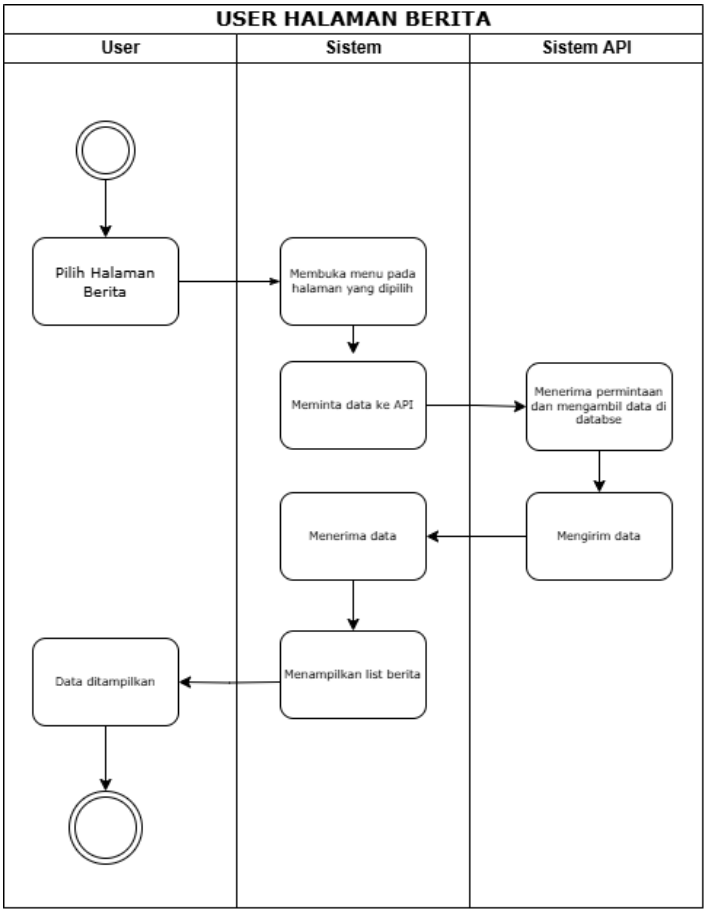


Gambar 4. 4. *Activity diagram user* layanan

Dari *activity diagram* diatas, dapat dijelaskan alur proses pengajuan surat pengantar oleh pengguna, dimulai ketika pengguna memilih halaman layanan yang kemudian membuka menu dan menampilkan formulir isian data. Pengguna lalu

memasukkan semua data yang diperlukan dan mengirimkannya, di mana sistem akan memvalidasi kelengkapan data; jika data belum lengkap, pengguna akan diminta untuk melengkapi kembali, namun jika sudah lengkap, sistem akan mengirimkan data ke Sistem API yang kemudian menerima dan memasukkannya ke database, lalu mengirimkan respons berhasil atau tidak kembali ke sistem, yang pada akhirnya menampilkan status "Berhasil" kepada pengguna.

b) *User* Berita

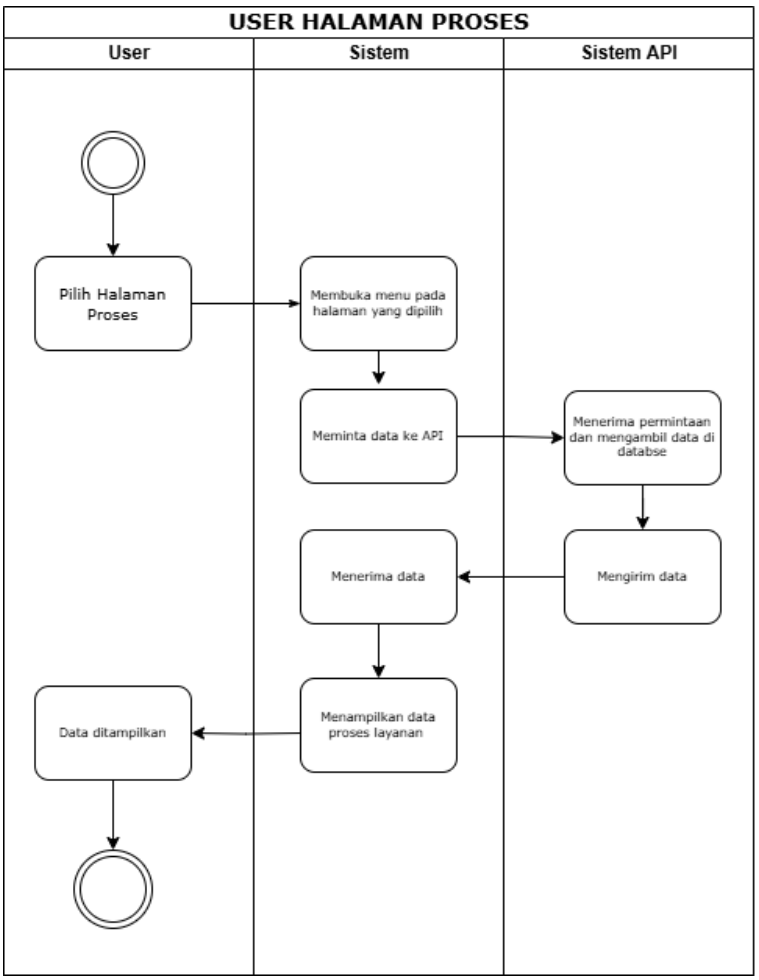


Gambar 4. 5. *Activity diagram user* berita

Activity diagram diatas menjelaskan tentang pengalaman pengguna yang lancar saat ingin mengakses informasi terkini dari desa. Proses dimulai dengan langkah sederhana pengguna memilih halaman berita, yang secara otomatis memicu sistem

untuk membuka menu dan dengan cerdas meminta data berita terbaru dari Sistem API. Sistem API kemudian dengan cepat mengambil informasi dari basis data dan mengembalikannya kepada sistem, yang pada gilirannya menyajikan daftar berita yang rapi dan mudah dibaca kepada pengguna, memungkinkan mereka untuk dengan cepat mendapatkan informasi yang dibutuhkan.

c) *User* halaman proses

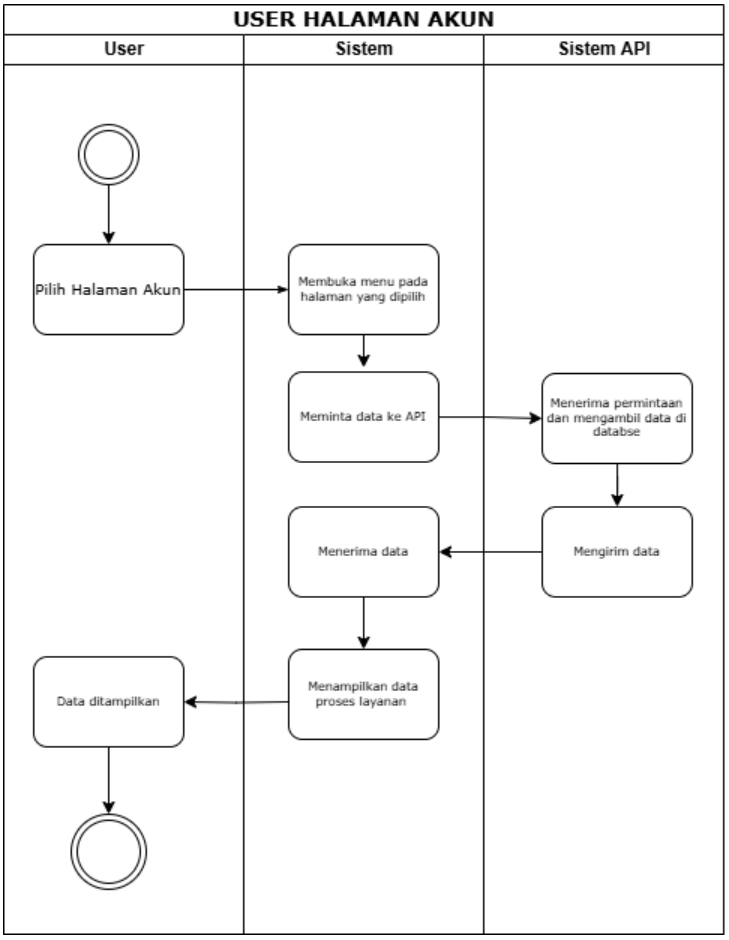


Gambar 4. 6. *Activity diagram user* proses

Dari *Activity diagram* , dapat dijelaskan alur interaksi yang intuitif bagi pengguna untuk melacak status layanan mereka. Dimulai dari pilihan halaman proses oleh pengguna, sistem dengan cepat merespons dengan membuka menu dan

secara efisien meminta data terkini dari Sistem API. Sistem API kemudian berperan aktif dalam mengambil informasi yang relevan dari basis data dan mengembalikannya ke sistem, yang pada akhirnya menampilkan data proses layanan secara jelas kepada pengguna, memberikan kemudahan dalam memantau kemajuan permohonan mereka.

d) *User* Akun



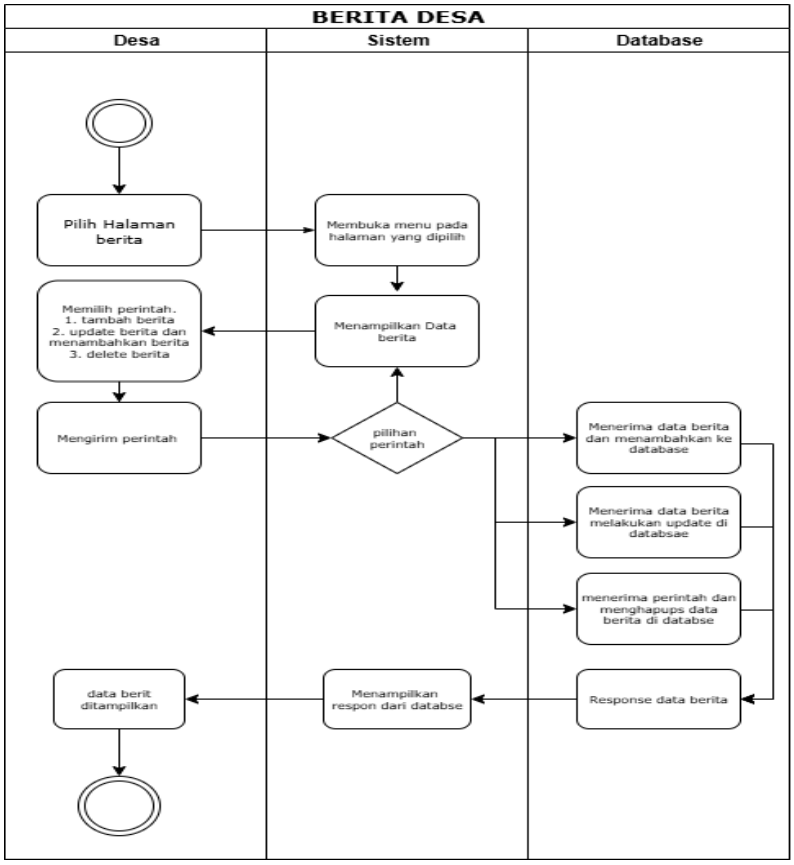
Gambar 4. 7. *Activity diagram user* akun

Activity diagram diatas menggambarkan alur bagi pengguna saat ingin mengakses atau mengelola informasi akun pribadi mereka. Proses dimulai ketika pengguna memilih halaman akun, yang secara instan direspons oleh sistem dengan

membuka menu dan secara efisien meminta data akun dari Sistem API. Sistem API kemudian dengan sigap mengambil data yang relevan dari basis data dan mengembalikannya ke sistem, yang pada akhirnya menampilkan informasi akun secara jelas kepada pengguna, memberikan mereka kontrol penuh atas detail pribadi mereka.

1. *Activity Diagram* Desa

    a. Berita Desa

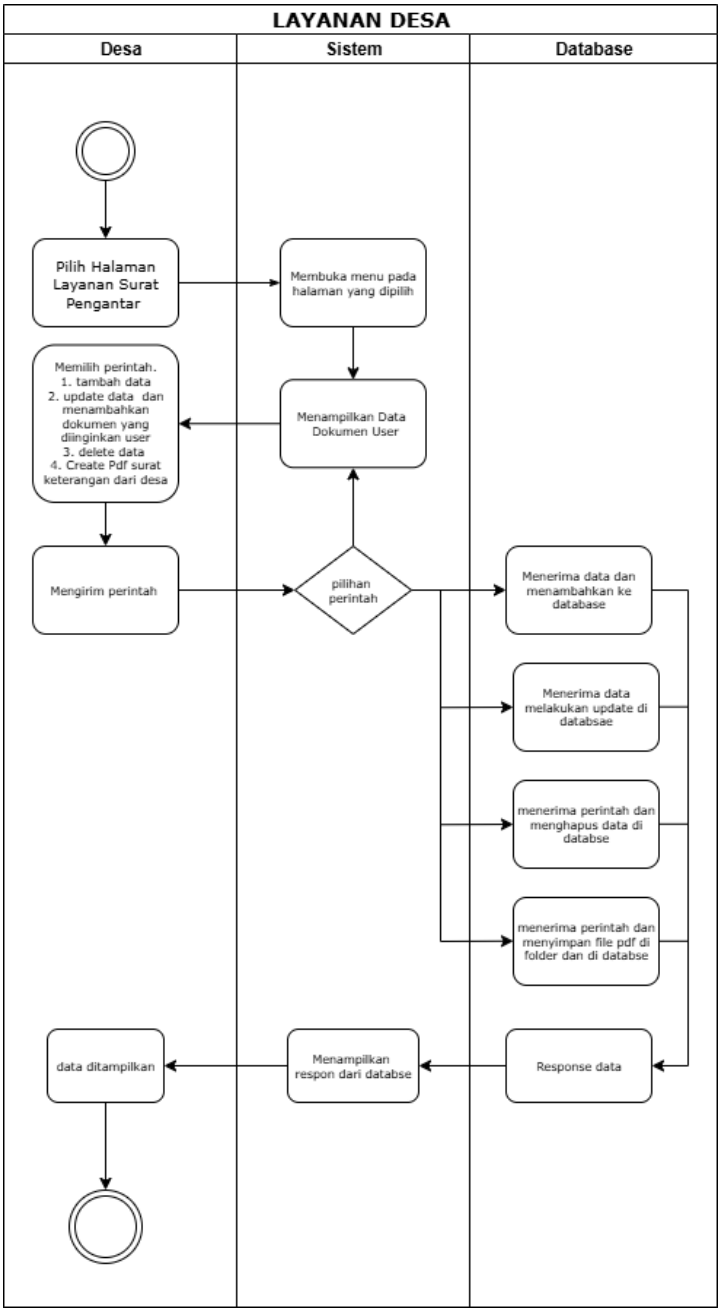


Gambar 4. 8. *Activity diagram* berita desa

Activity diagram diatas menggambarkan bagaimana Admin Desa mengelola informasi berita dengan mudah. Proses dimulai ketika Admin memilih halaman berita, yang segera direspons sistem dengan menampilkan data berita yang ada dan

opsi untuk mengelola. Admin kemudian dapat memilih untuk menambah, memperbarui, atau menghapus berita; pilihan ini akan diproses oleh sistem yang kemudian berinteraksi dengan database untuk menyimpan, memperbarui, atau menghapus data berita. Setelah operasi selesai, database akan memberikan respons yang kemudian ditampilkan kembali oleh sistem kepada Admin, memastikan bahwa informasi berita desa selalu akurat dan terkini.

b. Layanan Desa

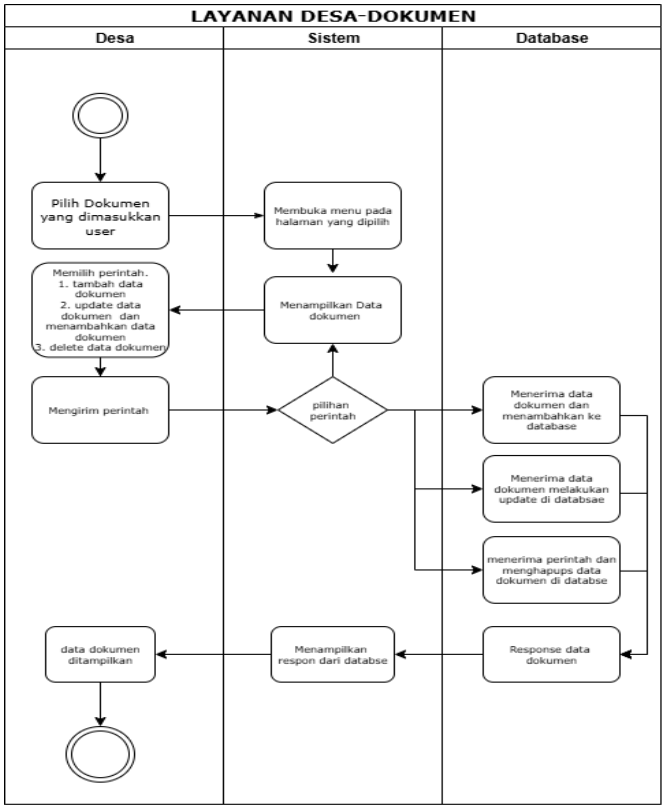


Gambar 4. 9. *Activity diagram* layanan desa

Activity diagram diatas menggambarkan bagaimana Admin Desa mengelola data dokumen permohonan surat pengantar dari pengguna dengan mudah. Proses dimulai saat Admin memilih halaman layanan, yang kemudian direspons sistem dengan menampilkan data dokumen yang ada dan opsi untuk mengelola. Admin

dapat memilih untuk menambah, memperbarui, menghapus data atau create pdf surat keterangan desa; pilihan ini akan diproses oleh sistem yang kemudian berinteraksi dengan database untuk menyimpan, memperbarui, atau menghapus informasi. Setelah operasi selesai, database akan memberikan respons yang kemudian ditampilkan kembali oleh sistem kepada Admin, memastikan bahwa status dan data permohonan selalu akurat dan terkini.

c. Layanan Desa-dokumen

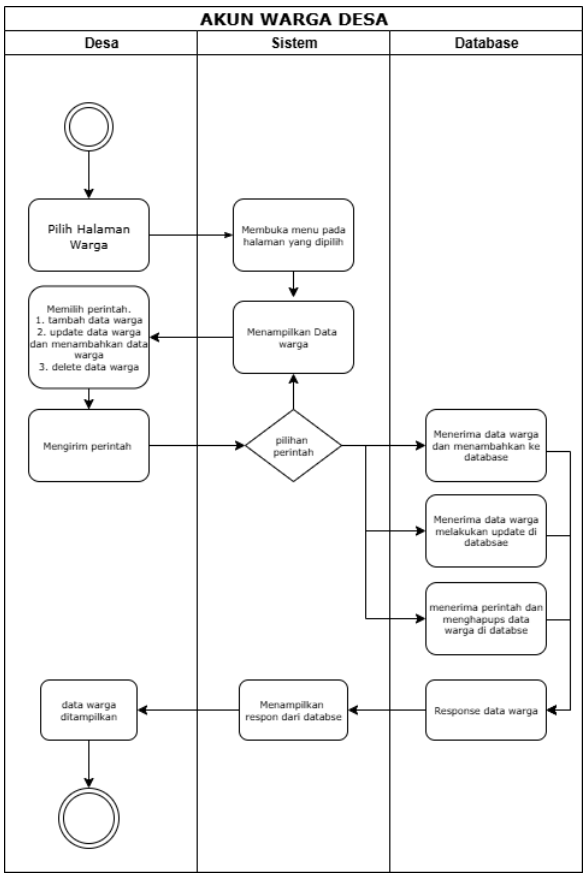


Gambar 4. 10. *Activity diagram* layanan desa-dokumen

Activity diagram diatas menggambarkan bagaimana Admin Desa mengelola dokumen yang telah diunggah oleh pengguna dengan mudah. Proses dimulai saat Admin memilih dokumen yang relevan, yang kemudian direspons sistem dengan menampilkan data dokumen dan opsi untuk mengelola. Admin dapat memilih untuk

menambah, memperbarui, atau menghapus data dokumen; pilihan ini akan diproses oleh sistem yang kemudian berinteraksi dengan database untuk menyimpan, memperbarui, atau menghapus informasi. Setelah operasi selesai, database akan memberikan respons yang kemudian ditampilkan kembali oleh sistem kepada Admin, memastikan bahwa manajemen dokumen berjalan efisien dan akurat.

d. Akun Warga Desa

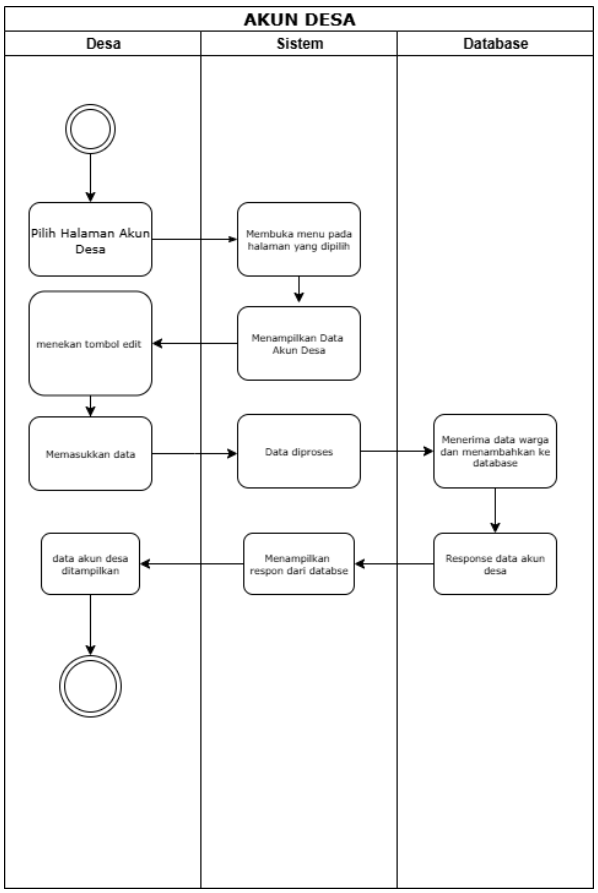


Gambar 4. 11. *Activity diagram* akun warga desa

Activity diagram diatas menggambarkan bagaimana Admin Desa mengelola data akun warga dengan mudah. Proses dimulai saat Admin memilih halaman warga, yang kemudian direspons sistem dengan menampilkan data warga yang ada dan opsi untuk mengelola. Admin dapat memilih untuk menambah, memperbarui,

atau menghapus data warga; pilihan ini akan diproses oleh sistem yang kemudian berinteraksi dengan database untuk menyimpan, memperbarui, atau menghapus informasi. Setelah operasi selesai, database akan memberikan respons yang kemudian ditampilkan kembali oleh sistem kepada Admin, memastikan bahwa manajemen data warga berjalan efisien dan akurat.

e. Akun Desa



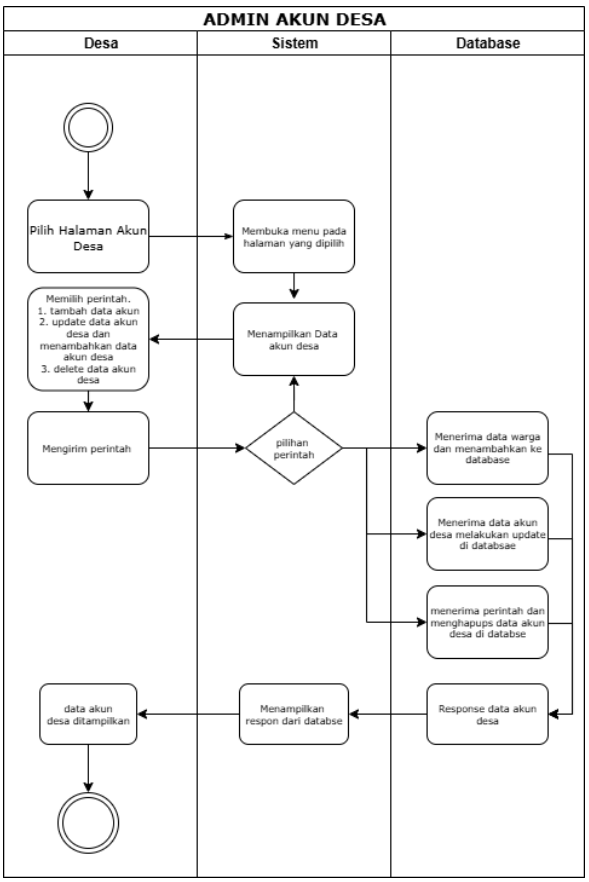
Gambar 4. 12. *Activity diagram* akun desa

Activity diagram diatas menggambarkan alur yang efisien bagi Admin Desa dalam memperbarui informasi profil desa. Proses dimulai saat Admin memilih halaman akun desa, yang kemudian direspons sistem dengan menampilkan data akun desa yang ada. Admin lalu menekan tombol edit dan memasukkan data

terbaru, yang segera diproses oleh sistem dan dikirimkan ke database untuk disimpan. Setelah pembaruan selesai, database akan memberikan respons yang kemudian ditampilkan kembali oleh sistem kepada Admin, memastikan bahwa informasi akun desa selalu akurat dan terkini.

2. *Activity Diagram* Admin

a) Admin Akun Desa

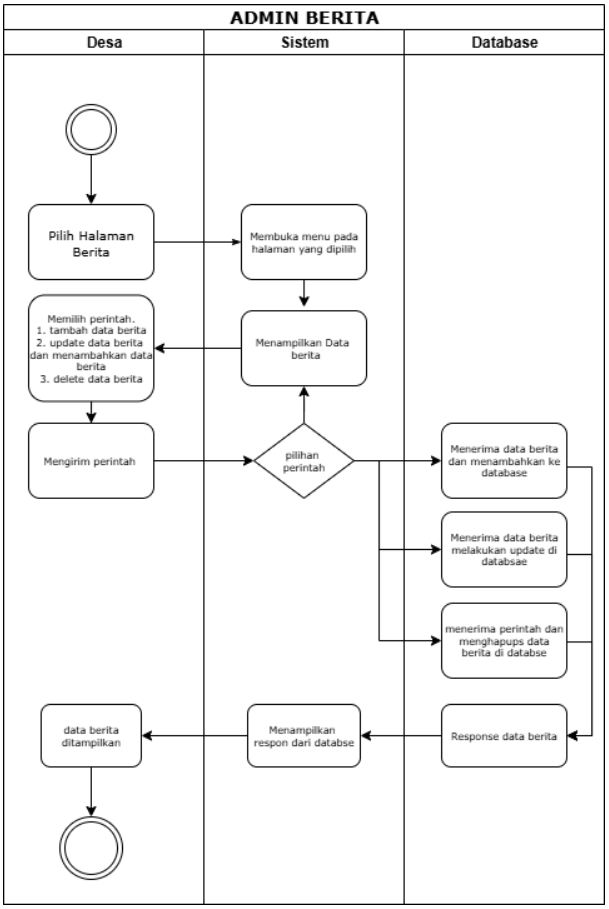


Gambar 4. 13. *Activity diagram* admin akun desa

Activity diagram diatas menggambarkan bagaimana Admin Desa mengelola data akun desa dengan mudah. Proses dimulai saat Admin memilih halaman akun desa, yang kemudian direspons sistem dengan menampilkan data akun desa yang ada dan opsi untuk mengelola. Admin dapat memilih untuk menambah,

memperbarui, atau menghapus data akun; pilihan ini akan diproses oleh sistem yang kemudian berinteraksi dengan database untuk menyimpan, memperbarui, atau menghapus informasi. Setelah operasi selesai, database akan memberikan respons yang kemudian ditampilkan kembali oleh sistem kepada Admin, memastikan bahwa manajemen data akun desa berjalan efisien dan akurat.

b) Admin berita



Gambar 4. 14. *Activity diagram* admin berita

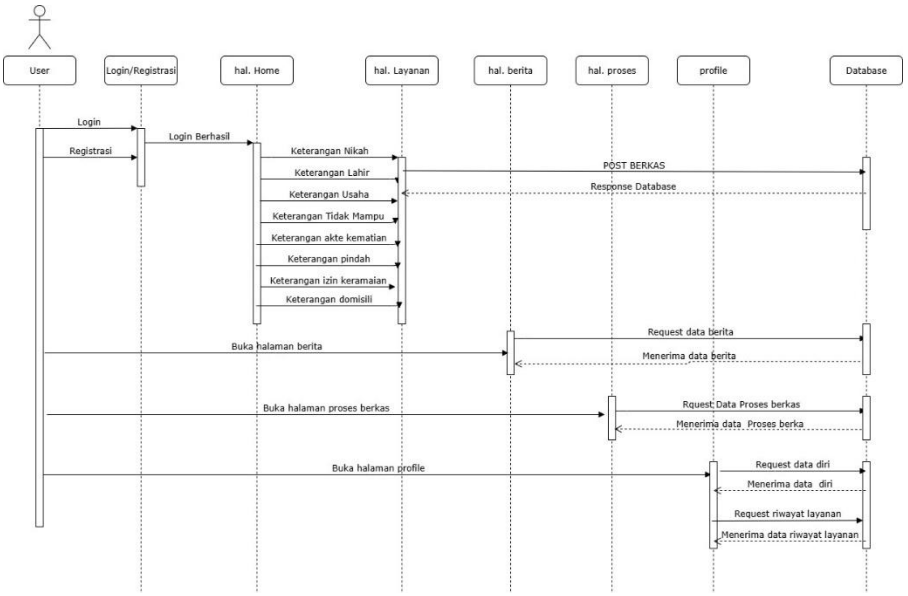
Activity diagram diatas menggambarkan bagaimana Admin Desa mengelola informasi berita dengan mudah. Proses dimulai saat Admin memilih halaman berita, yang kemudian direspons sistem dengan menampilkan data berita yang ada dan opsi untuk mengelola. Admin dapat memilih untuk menambah, memperbarui, atau

menghapus data berita; pilihan ini akan diproses oleh sistem yang kemudian berinteraksi dengan database untuk menyimpan, memperbarui, atau menghapus informasi. Setelah operasi selesai, database akan memberikan respons yang kemudian ditampilkan kembali oleh sistem kepada Admin, memastikan bahwa manajemen berita berjalan efisien dan akurat.

c. Sequence Diagram

Diagram Sequence adalah salah satu jenis diagram interaksi yang berfungsi untuk menjelaskan bagaimana suatu operasi dilaksanakan. Diagram ini secara rinci menunjukkan pesan (*message*) apa yang dikirimkan antar objek dan urutan waktu pelaksanaannya.

a. *User*

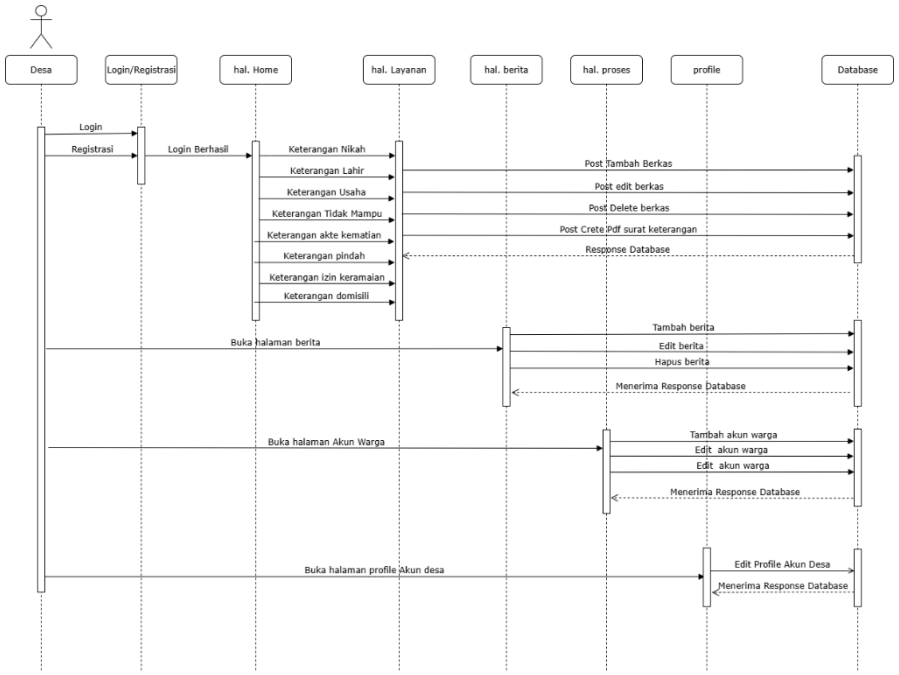


Gambar 4. 15. *Sequence diagram user*

Diagram *Sequence* diatas mengilustrasikan alur interaksi seorang Pengguna (*User*) dengan sistem aplikasi pelayanan administrasi desa, dimulai dari proses *Login* atau Registrasi untuk mendapatkan akses ke halaman *Home*. Dari halaman

utama ini, Pengguna dapat memilih berbagai modul: halaman Layanan untuk mengajukan permohonan surat keterangan dengan mengunggah berkas yang kemudian disimpan ke Database; halaman berita untuk meminta dan menampilkan informasi berita dari Database; halaman proses untuk meminta dan menampilkan status proses berkas dari Database; serta *profile* untuk meminta dan menampilkan data diri serta riwayat layanan dari Database, sehingga Pengguna dapat secara mandiri melakukan berbagai aktivitas administrasi dan memantau progresnya.

b. Desa

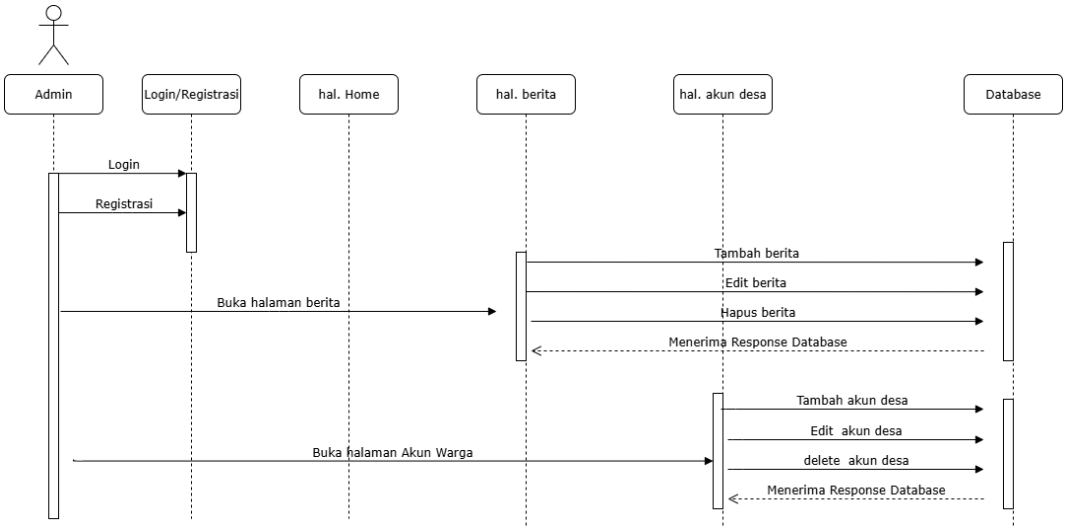


Gambar 4. 16. *Sequence diagram* desa

Diagram *Sequence* diatas menggambarkan alur interaksi antara Admin Desa dengan sistem aplikasi pelayanan administrasi, dimulai dari proses Login atau Registrasi untuk mengakses halaman Home. Dari halaman utama, Admin Desa dapat mengelola berbagai aspek: halaman Layanan untuk memproses berkas

permohonan surat (menambah, mengedit, menghapus dan create pdf) yang berinteraksi dengan Database; halaman berita untuk mengelola konten berita (menambah, mengedit, menghapus) yang juga berinteraksi dengan Database; halaman akun Warga untuk mengelola data akun warga (menambah, mengedit) melalui Database; serta halaman profile akun desa untuk mengedit profil desa itu sendiri di Database, sehingga Admin Desa dapat secara komprehensif mengelola operasional dan data sistem.

c. Admin



Gambar 4. 17. *Sequence diagram* admin

Diagram Sequence ini menggambarkan alur interaksi seorang Admin dengan sistem aplikasi, dimulai dari proses Login atau Registrasi untuk mendapatkan akses. Setelah berhasil, Admin dapat mengakses dua modul utama: pertama, halaman berita untuk mengelola konten berita (menambah, mengedit, atau menghapus) yang berinteraksi langsung dengan Database untuk menyimpan perubahan dan menerima

respons; kedua, halaman akun desa untuk mengelola data akun desa (menambah, mengedit, atau menghapus) yang juga berinteraksi dengan Database untuk pembaruan data dan menerima respons, sehingga Admin dapat memelihara informasi dan akun dalam sistem.

d. Rancangan database

Berikut ini rancangan *database* yang digunakan dalam pembuatan aplikasi rancang bangun pelayanan administrasi kantor desa Kecamatan Watang Pulu Kabupaten Sidrap**.**

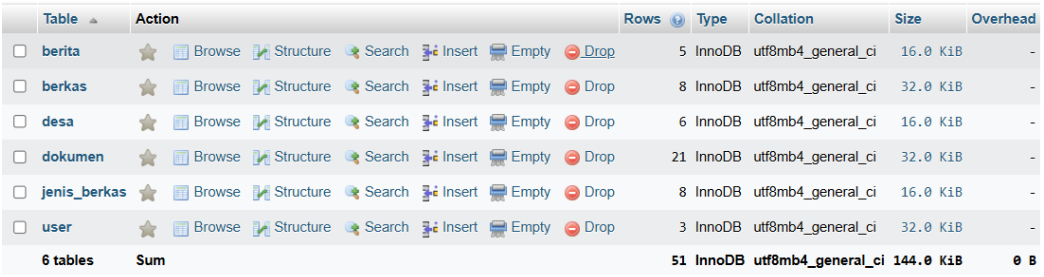


	Table	Action	Rows	Type	Collation	Size	Overhead
☐	berita	Browse Structure Search Insert Empty Drop	5	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	berkas	Browse Structure Search Insert Empty Drop	8	InnoDB	utf8mb4_general_ci	32.0 KiB	-
☐	desa	Browse Structure Search Insert Empty Drop	6	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	dokumen	Browse Structure Search Insert Empty Drop	21	InnoDB	utf8mb4_general_ci	32.0 KiB	-
☐	jenis_berkas	Browse Structure Search Insert Empty Drop	8	InnoDB	utf8mb4_general_ci	16.0 KiB	-
☐	user	Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	32.0 KiB	-
	6 tables	Sum	51	InnoDB	utf8mb4_general_ci	144.0 KiB	0 B

Gambar 4. 18. Struktur database

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id_user	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	id_desa	int(11)			No	None			Change Drop More
3	nama	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
4	alamat	text	utf8mb4_general_ci		No	None			Change Drop More
5	nomor_hp	varchar(16)	utf8mb4_general_ci		No	None			Change Drop More
6	no_ktp	varchar(16)	utf8mb4_general_ci		No	None			Change Drop More
7	no_kk	varchar(16)	utf8mb4_general_ci		No	None			Change Drop More
8	tempat_lahir	varchar(50)	utf8mb4_general_ci		No	None			Change Drop More
9	tanggal_lahir	date			No	None			Change Drop More
10	jenis_kelamin	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
11	password	varchar(20)	utf8mb4_general_ci		No	None			Change Drop More
12	sebagai	varchar(10)	utf8mb4_general_ci		No	None			Change Drop More

Gambar 4. 19. Struktur *table user*

Table structure Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id_jenis_berkas	int(11)			No	None		AUTO_INCREMENT	Change Drop More
2	jenis_berkas	varchar(100)	utf8mb4_general_ci		No	None			Change Drop More

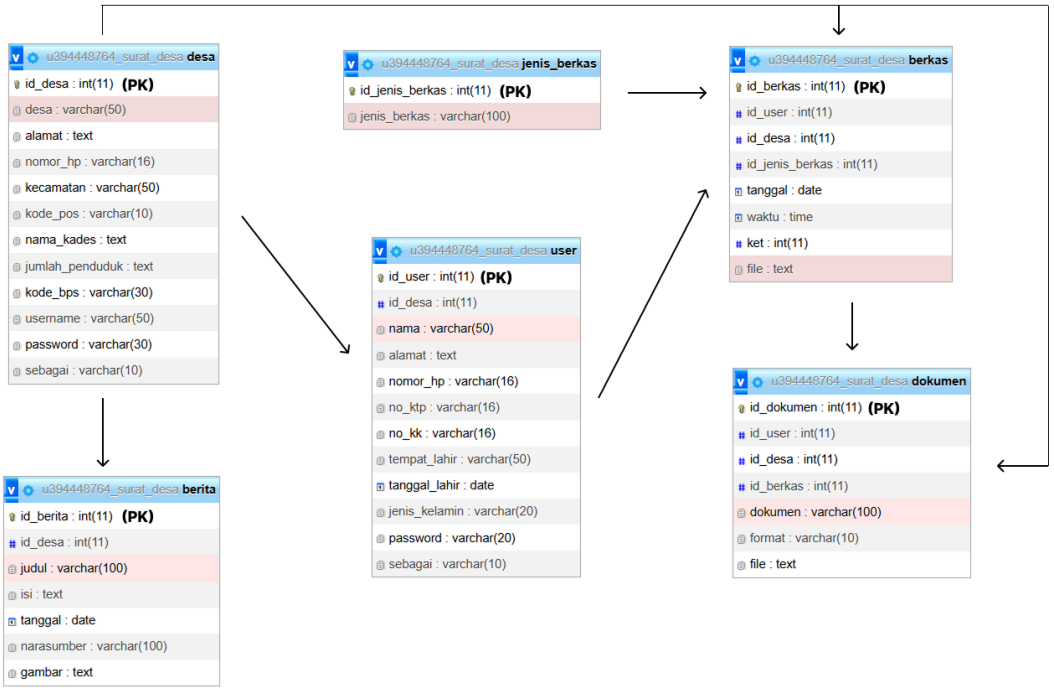
Gambar 4. 20. Struktur table jenis berkas

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
1	id_berkas	int(11)			No	None			Change Drop More
2	id_user	int(11)			No	None			Change Drop More
3	id_desa	int(11)			No	None			Change Drop More
4	id_jenis_berkas	int(11)			No	None			Change Drop More
5	tanggal	date			No	None			Change Drop More
6	waktu	time			No	None			Change Drop More
7	ket	int(11)			No	None	0 = proses. 1 = selesai		Change Drop More
8	file	text	utf8mb4_general_ci		No	None			Change Drop More

Gambar 4. 21. Struktur table berkas

	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
☐	1	**id_desa** 🔑	int(11)			No	*None*		AUTO_INCREMENT	Change Drop More
☐	2	**desa**	varchar(50)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	3	**alamat**	text	utf8mb4_general_ci		No	*None*			Change Drop More
☐	4	**nomor_hp**	varchar(16)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	5	**kecamatan**	varchar(50)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	6	**kode_pos**	varchar(10)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	7	**nama_kades**	text	utf8mb4_general_ci		No	*None*			Change Drop More
☐	8	**jumlah_penduduk**	text	utf8mb4_general_ci		No	*None*			Change Drop More
☐	9	**kode_bps**	varchar(30)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	10	**username**	varchar(50)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	11	**password**	varchar(30)	utf8mb4_general_ci		No	*None*			Change Drop More
☐	12	**sebagai**	varchar(10)	utf8mb4_general_ci		No	*None*			Change Drop More

Gambar 4. 22. Struktur table dokumen

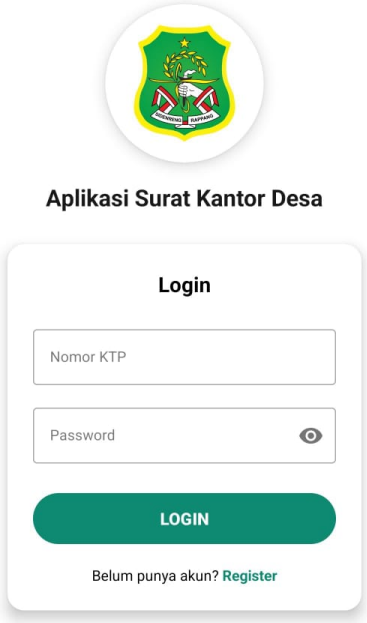


Gambar 4. 23. *Entity relationship* diagram

2. Detail Sistem

a. *User*

1) Halaman *Login User*



Gambar 4. 24. Halaman *Login User*

a) Penjelasan kode halaman *login* user

Halaman *login* dalam aplikasi Android ini diimplementasikan pada kelas *LoginActivity* yang bertanggung jawab sebagai penghubung antara antarmuka pengguna (UI) dan logika bisnis pada *ViewModel*. Aktivitas ini menggunakan pendekatan arsitektur MVVM (*Model-View-ViewModel*) serta memanfaatkan *Hilt* sebagai *dependency injection* untuk mengelola objek yang dibutuhkan secara

efisien. Kode berikut ini menangani proses autentikasi pengguna berdasarkan Nomor KTP dan *Password*.

1. Inisialisasi Komponen

```
private lateinit var binding: ActivityLoginBinding
private val viewModel: LoginViewModel by viewModels()
@Inject lateinit var loading: LoadingAlertDialog
private lateinit var sharedPreferencesLogin: SharedPreferencesLogin
```

Penjelasan:

Binding digunakan untuk mengakses elemen UI dengan *View Binding.*

viewModel mengacu pada objek *LoginViewModel* yang mengelola proses *login*.

loading merupakan dialog indikator proses yang ditampilkan saat autentikasi berlangsung.

sharedPreferencesLogin digunakan untuk menyimpan data pengguna secara lokal agar dapat diakses kembali saat dibutuhkan.

2. Fungsi *onCreate*()

```
override fun onCreate(savedInstanceState: Bundle?) {
   super.onCreate(savedInstanceState)
   binding = ActivityLoginBinding.inflate(layoutInflater)
   setContentView(binding.root)
   setupListeners()
   setSharedPreferencesLogin()
   getUser()
}
```

Pada fungsi *onCreate*(), tampilan login diinisialisasi, *event listener* pada tombol login dan register diatur, penyimpanan lokal pengguna disiapkan, serta LiveData dari ViewModel diamati untuk memproses hasil autentikasi.

3. Validasi *Input Login*

```
private fun validateInput(): Boolean {
```

```
    binding.apply {
        if (etNomorKtp.text.toString().isEmpty()) {
            etNomorKtp.error = "Nomor KTP harus diisi"
            return false
        }
        if (etPassword.text.toString().isEmpty()) {
            etPassword.error = "Password harus diisi"
            return false
        }
    }
    return true
}
```

Fungsi *validateInput*() memastikan bahwa pengguna telah mengisi kolom Nomor KTP dan *Password*. Jika tidak, akan ditampilkan pesan kesalahan di kolom yang belum terisi.

4. Proses *Login*

```
private fun performLogin() {
    var noKtp: String
    var password: String
    binding.apply {
        noKtp = etNomorKtp.text.toString()
        password = etPassword.text.toString()
    }
    fetchUser(noKtp, password)
}
```

Jika input valid, maka data yang dimasukkan oleh pengguna akan diteruskan ke fungsi *fetchUser*() untuk diproses oleh *ViewModel.*

5. Pemanggilan *ViewModel*

```
private fun fetchUser(noKtp: String, password: String) {
    viewModel.fetchUser(noKtp, password)
}
```

Fungsi ini memanggil *fetchUser*() dari *LoginViewModel* yang akan melakukan autentikasi ke database atau server.

6. Observasi Hasil Autentikasi

```
private fun getUser() {
   viewModel.getUser().observe(this@LoginActivity) { result ->
      when (result) {
         is UIState.Loading -> loading.alertDialogLoading(this@LoginActivity)
         is UIState.Success -> setSuccessFetchUser(result.data)
         is UIState.Failure -> setFailureFetchUser(result.message)
      }
   }
}
```

Hasil autentikasi diamati melalui *LiveData*, dan ditangani berdasarkan status:

Loading: Menampilkan dialog *loading*.

Success: Menyimpan data pengguna ke *SharedPreferences* dan pindah ke halaman utama.

Failure: Menampilkan pesan kesalahan jika login gagal.

7. Menangani Autentikasi Berhasil

```
private fun setSuccessFetchUser(data: UserModel) {
   if (checkFetchUser(data)) {
      setDataSharedPreferences(data)
   } else {
      Toast.makeText(this@LoginActivity, "Data tidak ditemukan\nPastikan No KTP dan Password Terdaftar ada", Toast.LENGTH_SHORT).show()
   }
   loading.alertDialogCancel()
}
```

Jika data pengguna valid, maka data tersebut akan disimpan dan pengguna diarahkan ke halaman utama.

8. Menyimpan Data ke *SharedPreferences*

```
private fun setDataSharedPreferences(data: UserModel) {
   try {
      data.apply {
         sharedPreferencesLogin.setLogin(
            idUser!!, nama!!, alamat!!, nomorHp!!, no_ktp!!, no_kk!!,
```

```
            tempat_lahir!!, tanggal_lahir!!, jenis_kelamin!!, password!!, sebagai!!
        )
        Toast.makeText(this@LoginActivity,                    "Berhasil",
Toast.LENGTH_SHORT ).show()
        moveToMainActivity(sebagai!!)
      }
   } catch (ex: Exception) {
     Toast.makeText(this@LoginActivity, ex.message, Toast.LENGTH_SHORT
).show()
   }
}
```

Data pengguna yang berhasil *login* akan disimpan ke dalam *SharedPreferences*, kemudian diarahkan ke halaman *MainActivity*.

9. Menangani Gagal *Login*

```
private fun setFailureFetchUser(message: String) {
   Toast.makeText(this@LoginActivity, "Data tidak ditemukan\nPastikan No
KTP dan Password Terdaftar ada", Toast.LENGTH_SHORT).show()
   Log.d("LoginTAG", "setFailureFetchUser: $message")
   loading.alertDialogCancel()
}
```

Jika *login* gagal (misalnya karena data tidak ditemukan), maka akan muncul pesan kesalahan dan dialog *loading* ditutup.

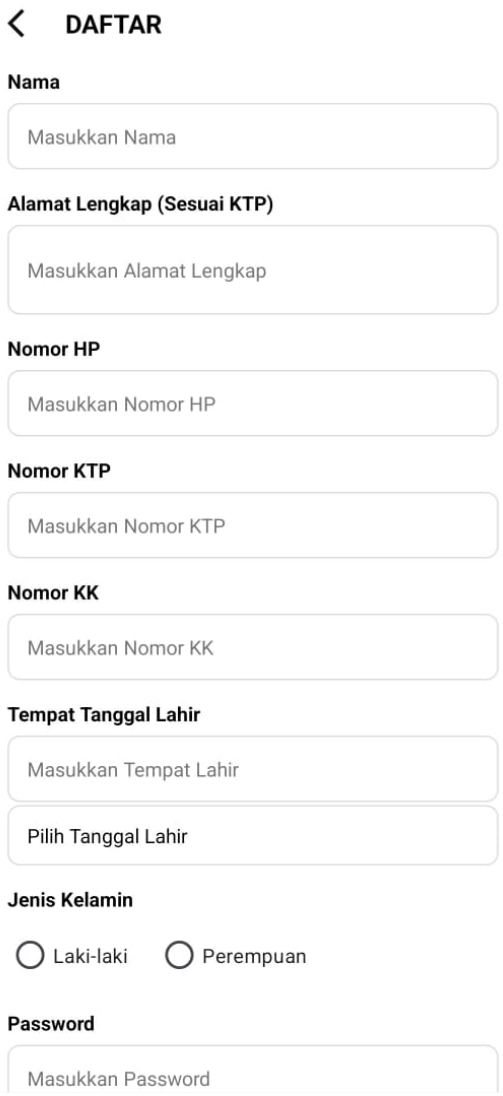
10. Navigasi ke Halaman Register

```
private fun navigateToRegister() {
   val intent = Intent(this, RegisterActivity::class.java)
   startActivity(intent)
}
```

Fungsi ini digunakan untuk mengarahkan pengguna ke halaman pendaftaran jika belum memiliki akun.

2) Halaman registrasi *user*

Ketika *user* ingin melakukan *login*, *user* harus daftar terlebih dahulu. Data yang harus disiapkan yaitu nama, alamat lengkap (sesuai KTP), nomor HP, nomor KTP, nomor KK, tempat tanggal lahir, setelah itu *user* menekan tombol lanjutkan. Setelah mendaftar, *user* dapat *login* memakai email serta kata sandi yang berhasil di daftarkan tadi.



Gambar 4. 25. Halaman registrasi *user*

```
class RegisterActivity : AppCompatActivity() {
private lateinit var binding: ActivityRegisterBinding
```

```
private val viewModel: RegisterViewModel by viewModels()
private var selectedDate: Date? = null
private var loading = LoadingAlertDialog()
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
binding = ActivityRegisterBinding.inflate(layoutInflater)
setContentView(binding.root)
setupViews()
setupButton()
getRegister() }
private fun setupViews() {
binding.apply {
   rgGender.setOnCheckedChangeListener { _, checkedId ->
      when (checkedId) {
         R.id.rbLakiLaki -> selectedGender = "Laki-laki"
         R.id.rbPerempuan -> selectedGender = "Perempuan" } } } }
private fun setupButton() {
binding.apply {
   ivBack.setOnClickListener {onBackPressed()}
   btnTanggalLahir.setOnClickListener {showDatePicker()}
   btnRegistrasi.setOnClickListener {if(validaeData()){completeRegistration()}}}}
private fun showDatePicker() {
val datePicker = MaterialDatePicker.Builder.datePicker().setTitleText("Pilih Tanggal Lahir")
   .setSelection(MaterialDatePicker.todayInUtcMilliseconds()).build()
datePicker.addOnPositiveButtonClickListener { selection ->
   selectedDate = Date(selection)
   val dateFormat = SimpleDateFormat("dd/MM/yyyy", Locale.getDefault())
   binding.btnTanggalLahir.text = dateFormat.format(selectedDate!!)}
datePicker.show(supportFragmentManager, "DATE_PICKER")}
private fun validaeData(): Boolean {
binding.apply {
   if (etNama.text.toString().isEmpty()) {
      etNama.error = "Nama harus diisi"
      return false }
   if (etAlamat.text.toString().isEmpty()) {
      etAlamat.error = "Alamat harus diisi"
      return false }
   if (etNomorHp.text.toString().isEmpty()) {
      etNomorHp.error = "Nomor HP harus diisi"
      return false }
   if (etNomorKtp.text.toString().isEmpty()) {
      etNomorKtp.error = "Nomor KTP harus diisi"
      return false }
   if (etNomorKk.text.toString().isEmpty()) {
      etNomorKk.error = "Nomor KK harus diisi"
      return false }
   if (selectedDate == null) {
      Toast.makeText(this@RegisterActivity, "Pilih tanggal lahir",
Toast.LENGTH_SHORT).show()
      return false }
   if (selectedGender == null) {
      Toast.makeText(this@RegisterActivity, "Pilih jenis kelamin",
Toast.LENGTH_SHORT).show()
      return false }
   if (etPassword.text.toString().isEmpty()) {
```

```
        etPassword.error = "Password harus diisi"
        return false } }
return true }
private fun completeRegistration() {
binding.apply {
    val nama = etNama.text.toString()
    val alamat = etAlamat.text.toString()
    val nomorHp = etNomorHp.text.toString()
    val noKtp = etNomorKtp.text.toString()
    val noKK = etNomorKk.text.toString()
    val tempatLahir = etTempatLahir.text.toString()
    val tanggalLahir = btnTanggalLahir.text.toString()
    val jenisKelamin = selectedGender!!
    val password = etPassword.text.toString()
    postRegister( nama, alamat, nomorHp, noKtp, noKK,
        tempatLahir, tanggalLahir, jenisKelamin, password) } }
private fun postRegister(
nama: String, alamat: String, nomorHp: String, noKtp: String, noKK: String, tempatLahir:
String,
tanggalLahir: String, jenisKelamin: String, password: String, ){
viewModel.postRegister(nama, alamat, nomorHp, noKtp, noKK, tempatLahir,
tanggalLahir, jenisKelamin, password)}
private fun getRegister(){
viewModel.getRegister().observe(this@RegisterActivity){result->
    when(result){
        is UIState.Loading-> loading.alertDialogLoading(this@RegisterActivity)
        is UIState.Success-> setSuccessRegister(result.data)
        is UIState.Failure-> setFailureRegister(result.message)
    } } }
private fun setSuccessRegister(data: ResponseModel) {
loading.alertDialogCancel()
if(data.status=="0"){
    Toast.makeText(this@RegisterActivity, "Berhasil Registrasi",
Toast.LENGTH_SHORT).show()
    finish()
} else{Toast.makeText(this@RegisterActivity, data.message_response,
Toast.LENGTH_SHORT).show()
}}
private fun setFailureRegister(message: String) {
Toast.makeText(this@RegisterActivity, message, Toast.LENGTH_SHORT).show()
loading.alertDialogCancel() }
companion object { private var selectedGender: String? = null }}
```

Gambar 4. 26 Codingan registrasi *user*

a) Penjelasan kode halaman *home user*

Halaman register merupakan bagian dari sistem autentikasi dalam aplikasi Android yang digunakan untuk mendaftarkan pengguna baru. Pada implementasinya, kelas *RegisterActivity* digunakan sebagai tampilan input data

pendaftaran yang akan dikirim ke *RegisterViewModel* untuk diproses. Aktivitas ini menggunakan pendekatan MVVM (*Model-View-ViewModel*) dan mendukung antarmuka interaktif seperti pemilihan tanggal lahir serta jenis kelamin.

1. Inisialisasi komponen

```
private lateinit var binding: ActivityRegisterBinding
private val viewModel: RegisterViewModel by viewModels()
private var selectedDate: Date? = null
private var loading = LoadingAlertDialog()
```

Penjelasan:

Binding: Menghubungkan *file layout* XML dengan kode *Kotlin* menggunakan *View Binding*.

viewModel: Objek *RegisterViewModel* yang menangani logika pendaftaran.

selectedDate: Menyimpan tanggal lahir yang dipilih pengguna.

loading: Menampilkan dialog saat proses registrasi sedang berlangsung.

2. Fungsi onCreate()

```
override fun onCreate(savedInstanceState: Bundle?) {
   super.onCreate(savedInstanceState)
   binding = ActivityRegisterBinding.inflate(layoutInflater)
   setContentView(binding.root)
   setupViews()
   setupButton()
   getRegister()
}
```

Fungsi ini digunakan untuk mengatur tampilan, *listener* tombol, dan observasi proses pendaftaran dari *ViewModel*.

3. Pemilihan jenis kelamin

```
private fun setupViews() {
   binding.rgGender.setOnCheckedChangeListener { _, checkedId ->
```

```
        when (checkedId) {
            R.id.rbLakiLaki -> selectedGender = "Laki-laki"
            R.id.rbPerempuan -> selectedGender = "Perempuan"
        }
    }
}
```

Pengguna memilih jenis kelamin melalui *RadioButton* dan nilai yang dipilih disimpan dalam variabel *selectedGender*.

4. Tombol aksi *button*

```
private fun setupButton() {
    binding.apply {
        ivBack.setOnClickListener { onBackPressed() }
        btnTanggalLahir.setOnClickListener { showDatePicker() }
        btnRegistrasi.setOnClickListener {
            if (validaeData()) {
                completeRegistration()
            }
        }
    }
}
```

Menangani aksi pengguna terhadap tombol kembali, tombol tanggal lahir, dan tombol registrasi.

5. Pemilihan tanggal lahir

```
private fun showDatePicker() {
    val datePicker = MaterialDatePicker.Builder.datePicker()
        .setTitleText("Pilih Tanggal Lahir")
        .setSelection(MaterialDatePicker.todayInUtcMilliseconds())
        .build()

    datePicker.addOnPositiveButtonClickListener { selection ->
        selectedDate = Date(selection)
        val dateFormat = SimpleDateFormat("dd/MM/yyyy", Locale.getDefault())
        binding.btnTanggalLahir.text = dateFormat.format(selectedDate!!)
    }
```

```
    datePicker.show(supportFragmentManager, "DATE_PICKER")
}
```

Menampilkan pemilih tanggal lahir (*date picker*) menggunakan komponen dari Material Design, lalu mengisi tombol tanggal dengan hasil pilihan pengguna.

6. Validasi data input

private fun validaeData(): Boolean

Fungsi ini memvalidasi seluruh *input* seperti nama, alamat, nomor HP, Nomor KTP, Nomor KK, tanggal lahir, jenis kelamin, dan *password*. Jika ada yang kosong, pengguna akan diberi peringatan.

7. Proses pendaftaran

private fun completeRegistration()

Jika semua data valid, maka data akan dikumpulkan dan dikirim ke fungsi postRegister() untuk diproses.

8. Mengirim data ke ViewModel

private fun postRegister(...)

Fungsi ini meneruskan data yang telah dimasukkan ke dalam *RegisterViewModel* untuk diteruskan ke API atau *database*.

9. Observasi hasil pendaftaran

private fun getRegister()

Fungsi ini mengamati *LiveData* dari *ViewModel* untuk mendapatkan status dari proses registrasi (*loading, success,* atau *failure*).

10. Menangani hasil registrasi

private fun setSuccessRegister(data: ResponseModel)

Jika status *respon* adalah "0", maka registrasi dianggap berhasil, ditandai dengan pesan *Toast* dan menutup halaman. Jika tidak berhasil, maka pesan kesalahan akan ditampilkan ke pengguna.

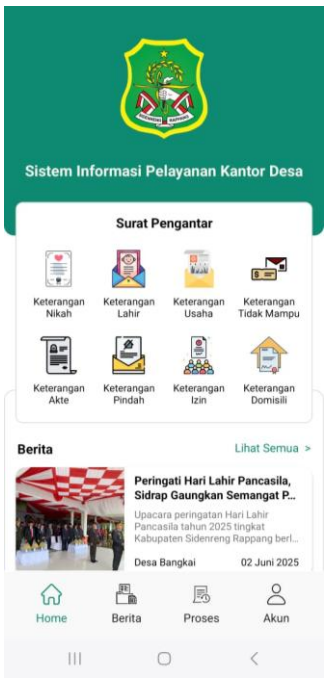
11. Menangani gagal registrasi

private fun setFailureRegister(message: String)

Jika terjadi kesalahan pada saat proses registrasi (misalnya karena kesalahan jaringan atau data tidak valid), maka akan muncul pesan kesalahan dan dialog *loading* ditutup.

3) Halaman *home user*

Halaman *home* berisi informasi mengenai jenis-jenis surat pengantar yang dapat diajukan serta berita terkini dari desa.



Gambar 4. 27. Halaman *home user*

```
@AndroidEntryPoint
class HomeFragment : Fragment() {
```

```
private lateinit var binding: FragmentHomeBinding
private val viewModel: HomeViewModel by viewModels()
private lateinit var beritaAdapter: BeritaAdapter
private lateinit var activityContext: MainActivity
private lateinit var context: Context
private lateinit var contextLifecycleOwner: LifecycleOwner  // harus dipanggil pada
onCreateView()
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
activityContext = (activity as MainActivity)
context = requireContext().applicationContext }
override fun onCreateView(
inflater: LayoutInflater, container: ViewGroup?,
savedInstanceState: Bundle?
): View {
binding = FragmentHomeBinding.inflate(layoutInflater)
contextLifecycleOwner = viewLifecycleOwner
fetchBerita()
getBerita()
setButton()
setSwipeRefreshLayout()
return binding.root }
private fun setSwipeRefreshLayout() {
binding.swipeRefresh.setOnRefreshListener {
Handler(Looper.getMainLooper()).postDelayed({
   binding.swipeRefresh.isRefreshing = false
   fetchBerita() }, 1500) }
binding.swipeRefresh.setProgressViewOffset(true, 0, 280) }
private fun setButton() {
binding.apply {
btnKeteranganNikah.setOnClickListener {
   val i = Intent(context, LayananActivity::class.java)
   i.putExtra("layanan", Constant.KETERANGAN_NIKAH)
   startActivity(i) }
btnKeteranganLahir.setOnClickListener {
   val i = Intent(context, LayananActivity::class.java)
   i.putExtra("layanan", Constant.KETERANGAN_LAHIR)
   startActivity(i) }
btnKeteranganUsaha.setOnClickListener {
   val i = Intent(context, LayananActivity::class.java)
   i.putExtra("layanan", Constant.KETERANGAN_USAHA)
   startActivity(i) }
btnKeteranganTidakMampu.setOnClickListener {
   val i = Intent(context, LayananActivity::class.java)
   i.putExtra("layanan", Constant.KETERANGAN_TIDAK_MAMPU)
   startActivity(i) }
btnKeteranganAkteKematian.setOnClickListener {
   val i = Intent(context, LayananActivity::class.java)
```

```
    i.putExtra("layanan", Constant.KETERANGAN_AKTE_KEMATIAN)
    startActivity(i) }
btnKeteranganPindah.setOnClickListener {
    val i = Intent(context, LayananActivity::class.java)
    i.putExtra("layanan", Constant.KETERANGAN_PINDAH)
    startActivity(i) }
btnKeteranganIzinKeramaian.setOnClickListener {
    val i = Intent(context, LayananActivity::class.java)
    i.putExtra("layanan", Constant.KETERANGAN_IZIN_KERAMAIAN)
    startActivity(i) }
btnKeteranganDomisili.setOnClickListener {
    val i = Intent(context, LayananActivity::class.java)
    i.putExtra("layanan", Constant.KETERANGAN_DOMISILI)
    startActivity(i) }
btnBerita.setOnClickListener {
    activityContext.clickBerita() } } }
private fun fetchBerita() {
viewModel.fetchBerita()}
private fun getBerita() {
viewModel.getBerita().observe(contextLifecycleOwner){ result->
when(result){
    is UIState.Loading-> setStartShimmerProduk()
    is UIState.Failure-> setFailureFetchBerita(result.message)
    is UIState.Success-> setSuccessFetchBerita(result.data) } } }
private fun setFailureFetchBerita(message: String) {
Toast.makeText(context, message, Toast.LENGTH_SHORT).show()
setStopShimmerProduk() }
private fun setSuccessFetchBerita(data: List<BeritaModel>) {
if(data.isNotEmpty()){
setAdapter(data) }
setStopShimmerProduk() }
private fun setAdapter(data: List<BeritaModel>) {
beritaAdapter = BeritaAdapter(data, object: OnClickItem.ClickBerita{
override fun clickBerita(berita: BeritaModel) {
    val i = Intent(context, BeritaActivity::class.java)
    i.putExtra("berita", berita)
    startActivity(i)
} }, true)
binding.rvBerita.apply {
layoutManager = LinearLayoutManager(context, LinearLayoutManager.VERTICAL, false)
adapter = beritaAdapter
} } }
```

Gambar 4. 28. *Codingan* halaman *home user*

b) Penjelasan kode halaman home user

Pada aplikasi ini, Home User merupakan salah satu bagian dari tampilan utama yang berfungsi untuk menampilkan daftar berita serta tombol-tombol navigasi menuju berbagai jenis layanan surat keterangan. Fragment ini menggunakan arsitektur MVVM dan memanfaatkan ViewModel untuk mengambil data berita secara asinkron.

Struktur dan Fungsi Utama**:**

1. Inisialisasi Fragment dan View Binding

   Fragment menggunakan FragmentHomeBinding untuk menghubungkan elemen layout XML dengan kode Kotlin. Context dan LifecycleOwner juga diinisialisasi untuk kebutuhan observasi LiveData dari ViewModel.

2. Pengambilan dan Penampilan Data Berita

   Data berita diambil melalui fungsi fetchBerita() yang memanggil viewModel.fetchBerita(). Selanjutnya, data diamati melalui fungsi getBerita(), yang akan merespon dalam tiga status:

   - Loading: Menampilkan efek shimmer loading.
   - Success: Menampilkan berita menggunakan RecyclerView.
   - Failure: Menampilkan pesan error menggunakan Toast.

3. Menampilkan Berita Menggunakan Adapter

   Fungsi setAdapter(data) digunakan untuk menampilkan daftar berita menggunakan BeritaAdapter pada RecyclerView. Ketika pengguna memilih salah satu berita, maka akan diarahkan ke BeritaActivity.

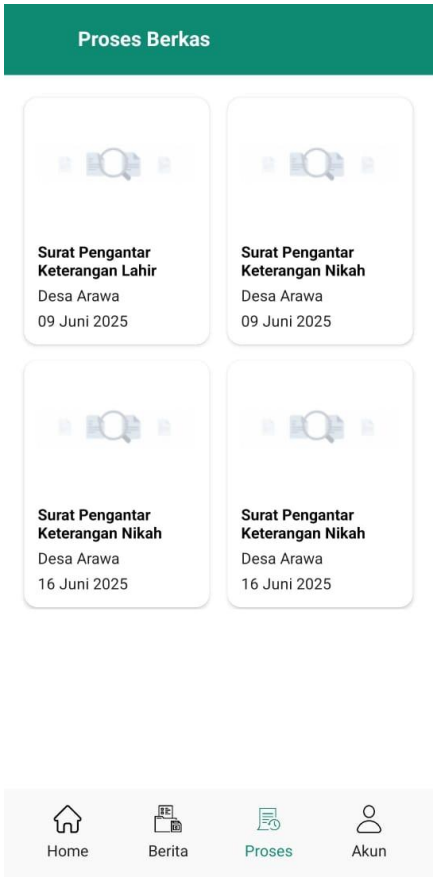
4. **Tombol Layanan Surat**

   Fragment ini juga memiliki berbagai tombol layanan, seperti:

   - Keterangan Nikah
   - Keterangan Lahir
   - Keterangan Usaha
   - Keterangan Tidak Mampu
   - Keterangan Akte Kematian
   - Keterangan Pindah
   - Keterangan Izin Keramaian
   - Keterangan Domisili

     Setiap tombol mengarahkan pengguna ke LayananActivity dengan parameter yang berbeda sesuai jenis layanan.

4) Halaman proses *user*



Gambar 4. 29. Halaman ***proses*** *user*

```
@AndroidEntryPoint
class ProsesBerkasFragment : Fragment() {
private lateinit var binding: FragmentProsesBerkasBinding
private val viewModel: ProsesBerkasViewModel by viewModels()
private lateinit var adapter: ProsesBerkasAdapter
private lateinit var sharedPreferences : SharedPreferencesLogin
private var arrayBerkas: ArrayList<BerkasModel> = arrayListOf()
private lateinit var activityContext: MainActivity
private lateinit var context: Context
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
activityContext = (activity as MainActivity)
context = requireContext().applicationContext }
override fun onCreateView(
inflater: LayoutInflater, container: ViewGroup?,
savedInstanceState: Bundle?
): View {
binding = FragmentProsesBerkasBinding.inflate(layoutInflater)
```

```
setSharedPreferences()
setNavTopBar()
fetchKeteranganDomisili(sharedPreferences.getIdUser())
getKeteranganDomisili()
return binding.root
}
private fun setSharedPreferences() {
sharedPreferences = SharedPreferencesLogin(context) }
@SuppressLint("SetTextI18n")
private fun setNavTopBar(){
binding.navTopBar.apply {
ivNavDrawer.visibility = View.GONE
ivBack.visibility = View.GONE

tvTitle.text = "Proses Berkas" }}
private fun fetchKeteranganDomisili(idUser: Int){
viewModel.fetchProsesBerkas(idUser) }
private fun getKeteranganDomisili(){
viewModel.getProsesBerkas.observe(viewLifecycleOwner){result->
when(result){
   is UIState.Loading-> setStartShimmer()
   is UIState.Failure-> setFailureFetchKeteranganDomisili(result.message)
   is UIState.Success-> setSuccessFetchKeteranganDomisili(result.data)
} } }
private fun setSuccessFetchKeteranganDomisili(data: ArrayList<BerkasModel>) {
if(data.isNotEmpty()){
setRecyclerViewSuratKeterangan(data) }
else{
Toast.makeText(context, "Tidak Ada Data", Toast.LENGTH_SHORT).show() }
setStopShimmer() }
private fun setFailureFetchKeteranganDomisili(message: String) {
Toast.makeText(context, message, Toast.LENGTH_SHORT).show()
setStopShimmer() }
private fun setRecyclerViewSuratKeterangan(data: ArrayList<BerkasModel>) {
adapter = ProsesBerkasAdapter(data, object: OnClickItem.ClickBerkas{
override fun clickBerkas(berkas: BerkasModel) {
   // Click data
   val i = Intent(context, ProsesBerkasDetailActivity::class.java)
   i.putExtra("berkas", berkas)
   startActivity(i) }})
binding.apply {
rvSuratKeterangan.adapter = adapter
rvSuratKeterangan.layoutManager = GridLayoutManager(context, 2) } }
private fun setStartShimmer(){
binding.apply {
smSuratKeterangan.startShimmer()
smSuratKeterangan.visibility = View.VISIBLE
rvSuratKeterangan.visibility = View.GONE } }
```

```
private fun setStopShimmer(){
binding.apply {
smSuratKeterangan.stopShimmer()
smSuratKeterangan.visibility = View.GONE
rvSuratKeterangan.visibility = View.VISIBLE } } }
```

Gambar 4. 30. *Codingan* halaman proses *user*

a) Penjelasan kode halaman proses *user*

ProsesBerkasFragment adalah halaman yang digunakan untuk menampilkan daftar berkas permohonan layanan administrasi yang sedang diproses oleh pengguna. Fragment ini mengambil data dari *ViewModel* dan menampilkannya menggunakan *RecyclerView* dalam format grid. Arsitektur yang digunakan tetap mengikuti pendekatan MVVM (*Model-View-ViewModel*) dan mengimplementasikan komunikasi data asinkron menggunakan *LiveData*.

Struktur dan Fungsi Utama

1. Inisialisasi Fragment dan Komponen

private lateinit var binding: FragmentProsesBerkasBinding

private val viewModel: ProsesBerkasViewModel by viewModels()

binding: Menghubungkan layout XML dengan kode program menggunakan View Binding.

viewModel: Mengacu pada ProsesBerkasViewModel yang menangani pengambilan data berkas dari server atau database.

2. Mengambil ID Pengguna dari SharedPreferences

```
private fun setSharedPreferences() {
   sharedPreferences = SharedPreferencesLogin(context)
}
```

Digunakan untuk mengambil *idUser* yang sedang *login*, sehingga data berkas bisa difilter sesuai pengguna.

3. Pengaturan Toolbar Halaman

private fun setNavTopBar()

Menyesuaikan tampilan judul halaman dengan teks "Proses Berkas" dan menyembunyikan tombol navigasi yang tidak diperlukan.

4. Pengambilan dan Observasi Data

fetchKeteranganDomisili(sharedPreferences.getIdUser())

getKeteranganDomisili()

fetchKeteranganDomisili(): Memanggil fungsi di ViewModel untuk mengambil data.

getKeteranganDomisili(): Mengamati hasil data dari ViewModel. Hasil diamati dalam tiga kondisi: Loading, Success, dan Failure.

5. Menampilkan Data ke RecyclerView

private fun setRecyclerViewSuratKeterangan(data: ArrayList<BerkasModel>)

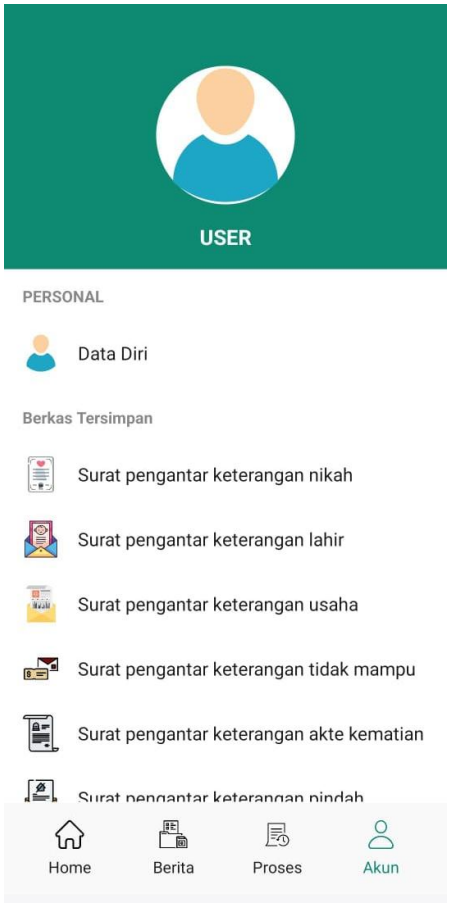
Jika data berhasil didapatkan, maka akan ditampilkan dalam RecyclerView menggunakan ProsesBerkasAdapter dan GridLayoutManager (2 kolom). Ketika salah satu item diklik, pengguna diarahkan ke halaman ProsesBerkasDetailActivity.

6. Penanganan Gagal Mengambil Data

private fun setFailureFetchKeteranganDomisili(message: String)

Jika terjadi kesalahan saat mengambil data, akan ditampilkan Toast berisi pesan *error*, dan *shimmer* akan dihentikan.

5) Halaman akun *user*



Gambar 4. 31. Halaman akun *user*

```
class ProfileFragment : Fragment() {
private lateinit var binding: FragmentProfileBinding
private lateinit var activityContext: MainActivity
private lateinit var context: Context
private var constant = Constant
private lateinit var sharedPreferencesLogin: SharedPreferencesLogin
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)
activityContext = (activity as MainActivity)
context = requireContext().applicationContext }
override fun onCreateView(
inflater: LayoutInflater, container: ViewGroup?,
savedInstanceState: Bundle?
```

```
): View {
binding = FragmentProfileBinding.inflate(layoutInflater)
setButton()
setSharedPreferences()
return binding.root }
private fun setSharedPreferences() {
sharedPreferencesLogin = SharedPreferencesLogin(context)
setDataShared() }
private fun setDataShared() {
binding.apply {
   tvNama.text = sharedPreferencesLogin.getNama() }}
private fun setButton() {
setButtonPersonal()
setButtonLogout()
setButtonBerkasTersimpan()}
```

Gambar 4. 32. Codingan halaman akun *user*

a) Penjelasan kode halaman akun user

ProfileFragment adalah *fragment* yang digunakan untuk menampilkan informasi pengguna dan menyediakan beberapa tombol untuk mengakses fitur tambahan seperti data diri, berkas tersimpan, dan *logout*. Halaman ini bersifat personal karena menyesuaikan data berdasarkan pengguna yang sedang login melalui *SharedPreferences*.

Struktur dan Fitur Utama

1. Inisialisasi Komponen Fragment

private lateinit var binding: FragmentProfileBinding

private lateinit var activityContext: MainActivity

private lateinit var context: Context

private lateinit var sharedPreferencesLogin: SharedPreferencesLogin

binding menghubungkan layout XML ke kode Kotlin.

context dan activityContext digunakan untuk keperluan navigasi dan resource.

sharedPreferencesLogin adalah *class helper* untuk mengambil dan menyimpan data login secara lokal.

Pada *onCreate*() dan *onCreateView*(), *fragment* menyiapkan *context,* binding, dan memanggil fungsi utama seperti setButton() dan setSharedPreferences().

2. Menampilkan Data Pengguna

```
private fun setSharedPreferences() {
    sharedPreferencesLogin = SharedPreferencesLogin(context)
    setDataShared()
}
private fun setDataShared() {
    binding.tvNama.text = sharedPreferencesLogin.getNama()
}
```

Data nama pengguna yang *login* diambil dari *SharedPreferences* dan ditampilkan pada elemen *TextView* di halaman profil.

Hal ini menegaskan bahwa pengguna yang masuk akan melihat data mereka secara langsung.

3. Navigasi ke Halaman Lain

Fungsi *setButton*() mengatur aksi klik dari berbagai tombol yang tersedia di halaman profil:

- Tombol Data Diri

```
btnDataDiri.setOnClickListener {
   startActivity(Intent(context, DataDiriActivity::class.java))
}
```

Digunakan untuk membuka halaman pengisian atau melihat informasi pribadi pengguna.

- Tombol *Logout*

```
btnLogout.setOnClickListener {
   sharedPreferencesLogin.setLogin(0, "", "", "", "", "", "", "", "", "", "")
   startActivity(Intent(context, LoginActivity::class.java))
   activityContext.finish()
}
```

Ketika pengguna menekan tombol *Logout*, semua data *login* dikosongkan dan pengguna diarahkan ke halaman *login*. Hal ini merupakan cara sederhana namun efektif untuk mengamankan sesi pengguna.

- Tombol Berkas Tersimpan

```
btnKeteranganDomisili.setOnClickListener {
   moveToActivity(constant.KETERANGAN_DOMISILI)
}
```

Terdapat banyak tombol yang masing-masing mengarahkan pengguna ke halaman BerkasTersimpanActivity, disertai parameter keterangan yang menunjukkan jenis surat atau keperluan administrasi seperti:

- Surat Keterangan Domisili
- Izin Keramaian

- Keterangan Usaha
- Keterangan Tidak Mampu
- Keterangan Lahir
- Keterangan Nikah
- Keterangan Akte Kematian
- Surat Pindah

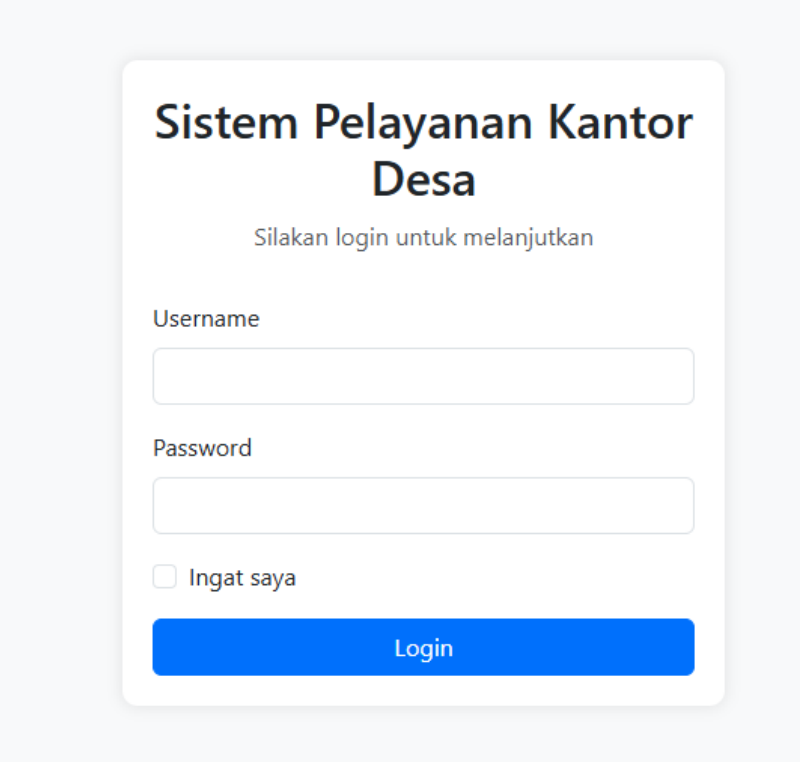
Parameter keterangan digunakan sebagai acuan dalam halaman tujuan untuk menampilkan jenis berkas yang sesuai.

4. Fungsi Navigasi Umum

```
private fun moveToActivity(keterangan: String){
    val intent = Intent(context, BerkasTersimpanActivity::class.java)
    intent.putExtra("keterangan", keterangan)
    startActivity(intent)
}
```

Digunakan sebagai metode umum untuk membuka halaman BerkasTersimpanActivity dan meneruskan jenis berkas yang dipilih oleh pengguna. Dengan pendekatan ini, kode menjadi lebih efisien dan tidak redundan.

b. Halaman *login* desa dan admin

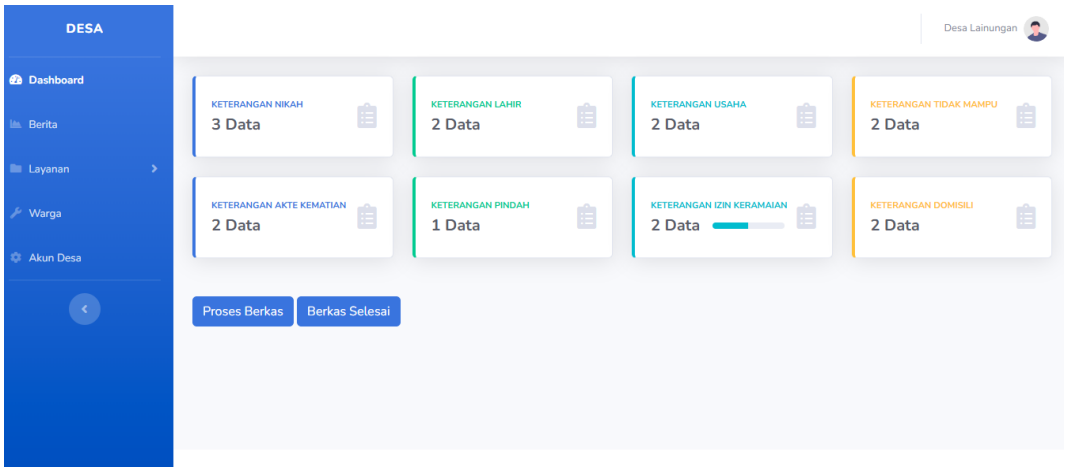


Gambar 4. 33. Halaman *login* desa dan admin

Gambar tampilan diatas adalah Halaman login ini digunakan oleh admin dan kantor desa untuk mengakses sistem pelayanan berbasis web. Sistem ini dibangun menggunakan bahasa pemrograman PHP untuk bagian server-side dan framework CSS Bootstrap untuk bagian antarmuka (UI).

c. Desa

1) Halaman *dashboard* desa



Gambar 4. 34. Halaman *dashboard* desa

Halaman *Dashboard* pada aplikasi web kantor desa merupakan tampilan utama yang ditujukan bagi administrator sistem. *Dashboard* ini berfungsi untuk menyajikan ringkasan informasi penting secara cepat dan *real-time*. Informasi yang ditampilkan berupa jumlah data permohonan layanan dari setiap desa, serta jumlah berkas yang berada dalam proses maupun yang telah selesai diproses.

Fitur utama dari halaman *dashboard* meliputi:

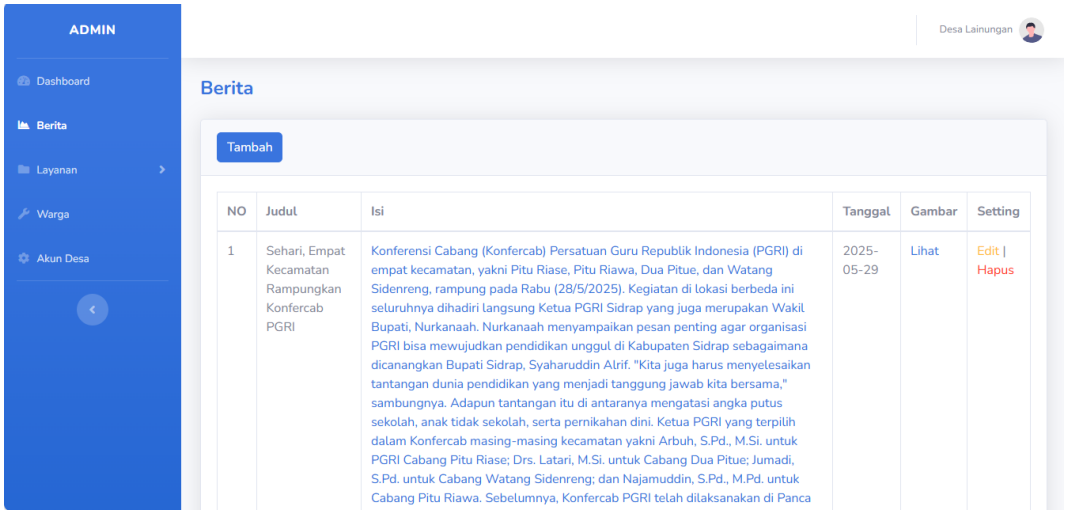
(1) Rekapitulasi jumlah layanan: Menampilkan jumlah permohonan pada masing-masing jenis layanan (seperti surat keterangan domisili, keterangan tidak mampu, dan sebagainya) berdasarkan desa asal pemohon.

(2) Monitoring status berkas: Sistem menampilkan jumlah berkas yang sedang dalam proses penanganan dan jumlah berkas yang sudah selesai diproses oleh pihak desa.

(3) Tampilan visual ringkas dan informatif: Data disajikan dalam bentuk angka dan komponen UI berbasis Bootstrap, sehingga mudah dibaca dan memberikan gambaran menyeluruh kepada petugas atau admin desa.

Dashboard ini dirancang sebagai pusat kontrol dan monitoring untuk memastikan pelayanan desa berjalan dengan efisien, transparan, dan terpantau secara berkala.

2) Halaman berita desa



Gambar 4. 35. Halaman berita desa

Halaman Berita merupakan bagian dari sistem aplikasi web yang digunakan oleh pihak desa untuk mengelola informasi dan pengumuman yang berkaitan dengan kegiatan atau pelayanan desa. Fitur ini memungkinkan admin desa untuk

menambahkan, memperbarui, dan menghapus berita secara mandiri sesuai kebutuhan.

Pada halaman ini, data berita ditampilkan dalam bentuk tabel dinamis, yang menyajikan informasi seperti judul berita, tanggal publikasi, isi singkat, dan aksi (edit dan hapus). Tabel ini disusun dengan tampilan yang rapi dan responsif berkat penggunaan *framework Bootstrap*, sehingga memudahkan admin dalam mengelola konten.

Fitur-fitur utama dalam halaman ini meliputi:

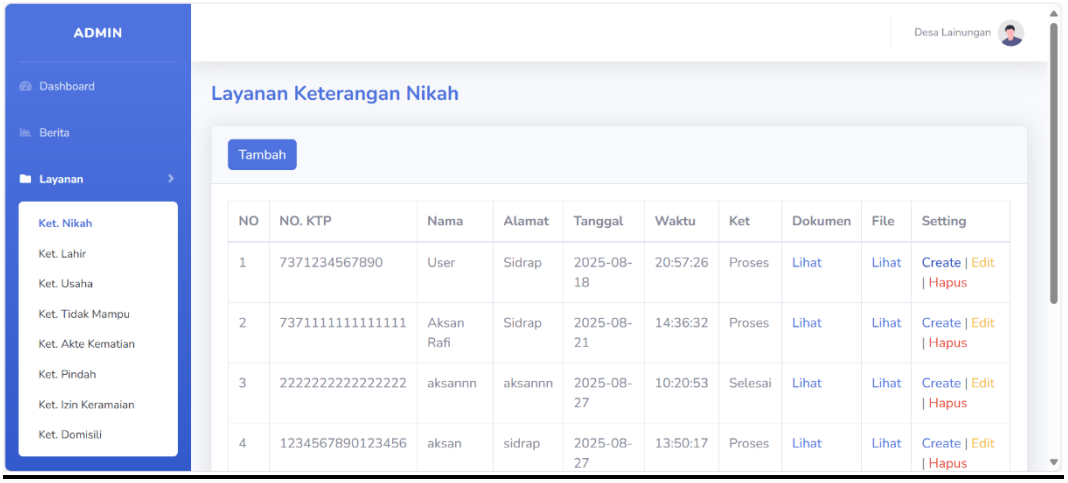
(1) Tambah Berita: Admin dapat menambahkan berita baru melalui form input yang tersedia. Berita ini akan ditautkan secara otomatis dengan desa yang saat ini sedang *login*.

(2) Edit Berita: Admin dapat melakukan perubahan terhadap berita yang sudah pernah dipublikasikan, seperti mengubah judul, isi, atau tanggal berita.

(3) Hapus Berita: Berita yang sudah tidak relevan atau tidak lagi diperlukan dapat dihapus secara permanen dari sistem.

Semua data yang ditampilkan dan dikelola di halaman ini bersifat spesifik untuk masing-masing desa, sehingga berita yang ditampilkan hanya milik desa yang login saat itu. Hal ini menjaga relevansi dan keamanan data antar desa.

3) Halaman layanan desa



Gambar 4. 36. Halaman layanan desa

Pada halaman ini, sistem menyediakan fitur untuk mengelola permohonan Surat Keterangan Nikah yang diajukan oleh warga desa. Fitur-fitur utama yang tersedia mencakup penambahan data baru, penampilan data warga, pengeditan data, serta penghapusan data.

1) Penambahan Data

Data baru dapat ditambahkan melalui tombol "Tambah", yang akan membuka modal (jendela pop-up) berisi formulir input. Formulir ini terdiri atas:

Pemilihan warga dari dropdown berdasarkan data KTP dan nama.

Keterangan status proses permohonan (Proses atau Selesai).

Tanggal dan waktu permohonan.

Unggahan dokumen PDF sebagai bukti atau berkas pendukung.

Formulir tersebut mengirimkan data melalui metode POST ke file keterangan-nikah-post.php untuk diproses dan disimpan ke dalam tabel berkas di basis data.

2) Menampilkan Data

Data yang telah masuk ditampilkan dalam bentuk tabel menggunakan komponen *Bootstrap Table* dengan *DataTable*. Kolom yang ditampilkan antara lain:

- Nomor urut
- Nomor KTP
- Nama warga
- Alamat
- Tanggal dan waktu permohonan
- Status permohonan (Proses/Selesai)
- Tautan untuk melihat dokumen dan file PDF
- Opsi pengaturan (Edit dan Hapus)

Setiap baris data diambil dengan *query*:

```
SELECT * FROM berkas WHERE id_jenis_berkas='1' AND id_desa='$id_desa_admin' ORDER BY tanggal, waktu ASC
```

Kemudian, sistem mencocokkan id_user dari tabel user untuk mendapatkan detail nama, alamat, dan nomor KTP.

3) Edit Data

Setiap baris data memiliki tautan Edit yang memunculkan modal dengan isian form sama seperti *form* tambah data. Data yang sudah tersimpan sebelumnya akan otomatis ditampilkan dalam *field* isian *(pre-filled*). Setelah diubah, data dikirim ke keterangan-nikah-post.php melalui *method* POST untuk memperbarui informasi.

4) Hapus Data

Tautan Hapus akan memunculkan modal konfirmasi penghapusan. Penghapusan dilakukan dengan mengirim id_berkas ke file proses keterangan-nikah-post.php.

Data yang sudah dihapus tidak bisa dikembalikan karena penghapusan bersifat permanen dari basis data.

5) Tampilan Antarmuka

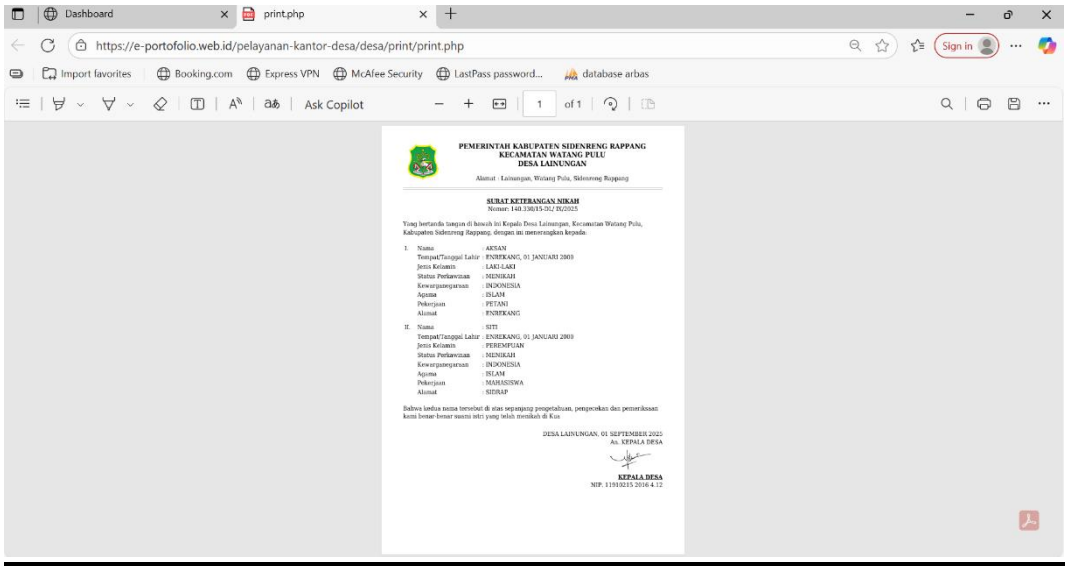
Tampilan antarmuka menggunakan framework Bootstrap, yang menyediakan:

Tombol dan form dengan gaya modern dan responsif

Modal untuk input dan konfirmasi

Tabel yang terintegrasi dengan plugin DataTables untuk fitur pencarian, pengurutan, dan pagination otomatis.

4) Halaman pdf surat pengantar desa

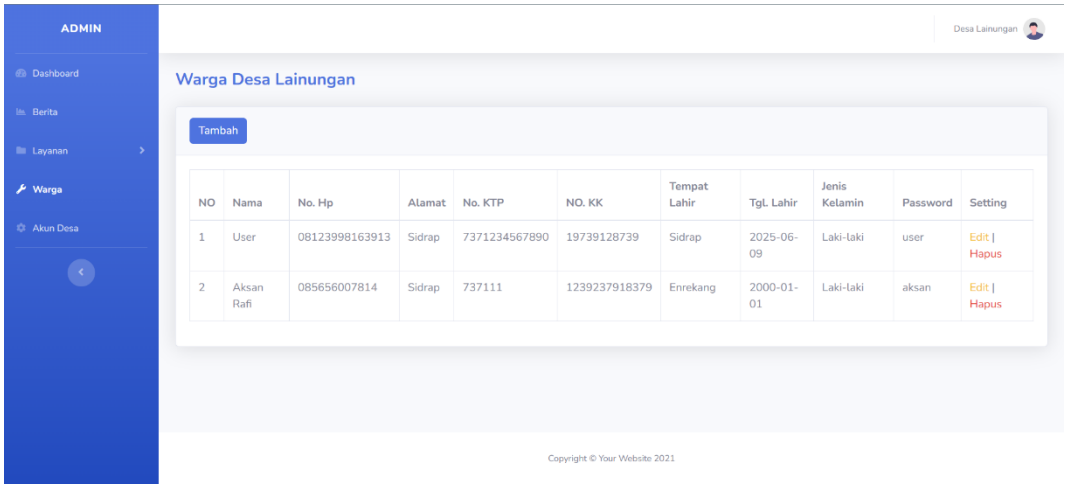


Gambar 4. 37. Halaman pdf surat pengantar desa

Gambar diatas adalah halaman pengendali utama untuk memproses formulir pengajuan dokumen secara *online,* Kode ini dirancang untuk menerima data dari formulir POST dan menggunakan pustaka mPDF untuk menghasilkan dokumen PDF. Pencetakan Dokumen: Kode ini menyertakan (require_once) file print-

keterangan-nikah.php yang berisi *template* dan logika pencetakan. Semua data yang telah diproses kemudian diteruskan ke fungsi PrintSurat() untuk menghasilkan file PDF.

5) Halaman warga desa



Gambar 4. 38. Halaman warga desa

Halaman ini digunakan oleh admin desa untuk mengelola data warga yang berada dalam wilayah administratifnya. Fitur utama yang tersedia pada halaman ini meliputi penambahan data warga, pengeditan data, serta penghapusan data warga.

1) Fungsi dan Alur Sistem

Saat halaman ini diakses, sistem secara otomatis memverifikasi sesi id_desa dari data session atau *cookie*. Ini memastikan bahwa data warga yang ditampilkan hanya milik desa yang sedang login.

Setelah berhasil memverifikasi, halaman akan menampilkan nama desa yang sedang aktif dan menampilkan daftar seluruh warga desa dalam bentuk tabel.

2) Tampilan Tabel

Tabel data warga ditampilkan menggunakan komponen *Bootstrap Table* yang responsif dan modern. Kolom yang ditampilkan dalam tabel antara lain:

- Nomor
- Nama warga
- Nomor HP
- Alamat
- Nomor KTP
- Nomor Kartu Keluarga (KK)
- Tempat dan Tanggal Lahir
- Jenis Kelamin
- Password

Opsi pengaturan (Edit dan Hapus)

Data warga diambil dari basis data melalui query berikut:

```
SELECT * FROM user WHERE id_desa='$id_desa_admin'
```

Kemudian data tersebut ditampilkan ke dalam tabel menggunakan perulangan while.

3) Tambah Data Warga

Fitur tambah data dapat diakses melalui tombol Tambah, yang memunculkan modal Bootstrap berisi form input. Admin dapat mengisi data seperti:

- Nama
- Alamat
- Nomor HP
- Nomor KTP

- Nomor KK
- Tempat dan Tanggal Lahir
- Jenis Kelamin
- *Password*

Data yang diisi akan dikirim menggunakan metode POST ke file warga-post.php dan selanjutnya disimpan ke dalam tabel *user*.

4) Edit Data Warga

Setiap baris data warga memiliki tautan Edit yang membuka modal *Bootstrap* dengan data yang sudah terisi sebelumnya (pre-filled). Admin dapat mengubah informasi dan menyimpan kembali dengan mengirim data ke warga-post.php.

5) Hapus Data Warga

Tautan Hapus memunculkan modal konfirmasi penghapusan. Jika admin menyetujui, maka data warga akan dikirim ke warga-post.php dan dihapus secara permanen dari basis data.

6) Teknologi yang Digunakan

Halaman ini menggunakan kombinasi:

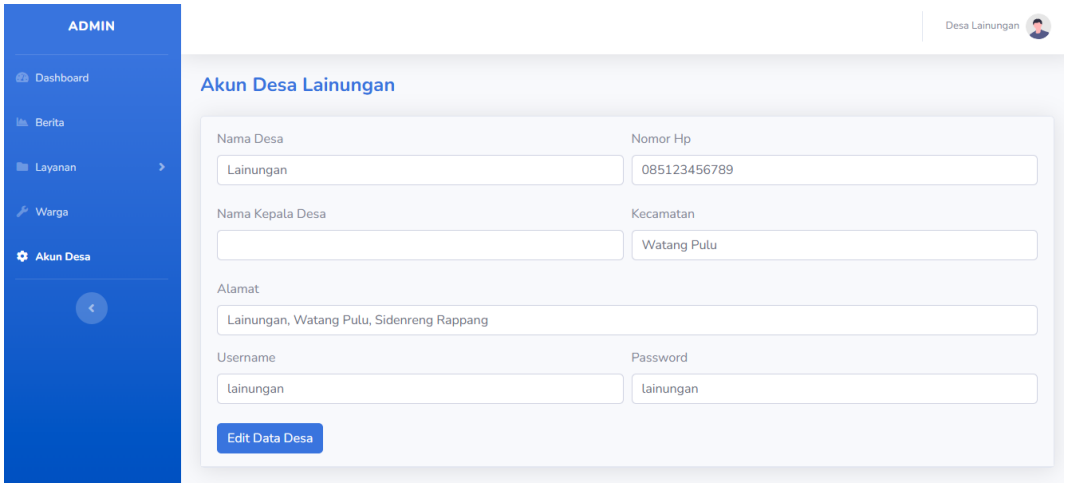
PHP sebagai bahasa pemrograman server-side

MySQL untuk menyimpan dan mengambil data dari basis data

Bootstrap untuk desain antarmuka, termasuk penggunaan modal dan form styling

JavaScript (dari Bootstrap) untuk mendukung interaktivitas modal

b) Halaman akun desa



Gambar 4. 39. Halaman akun desa

Halaman Akun Desa merupakan fitur yang digunakan untuk menampilkan dan memperbarui informasi akun dari desa yang sedang *login* ke dalam sistem. Sistem ini memastikan bahwa hanya data akun desa yang sesuai dengan sesi atau *cookie login* yang dapat diakses dan dimodifikasi, guna menjaga keamanan dan privasi data masing-masing desa.

1. Fitur dan Fungsi

Setelah *login* berhasil, sistem memverifikasi sesi login melalui `$_SESSION['id_desa']` atau `$_COOKIE['id_desa']`. Berdasarkan identitas ini, sistem kemudian mengambil informasi akun desa yang bersangkutan dari tabel desa dalam basis data menggunakan query:

```
SELECT * FROM desa WHERE id_desa='$id_desa_admin'
```

Data yang ditampilkan pada halaman ini meliputi:

- Nama Desa

- Nomor HP
- Nama Kepala Desa
- Kecamatan
- Alamat
- *Username*
- *Password*

Informasi ditampilkan dalam tampilan form label menggunakan komponen Bootstrap, agar mudah dibaca namun tidak langsung dapat diedit tanpa tindakan khusus.

2. Edit Data Akun Desa

Untuk memperbarui informasi akun, terdapat tombol "Edit Data Desa" yang ketika diklik akan memunculkan modal *Bootstrap* berisi form pengisian data. Form ini memungkinkan admin desa untuk mengubah informasi berikut:

- Nama Desa
- Alamat
- Nomor HP
- Nama Kepala Desa
- *Username*
- *Password*

Data yang dimasukkan akan dikirim ke file akun-post.php menggunakan metode POST saat tombol Simpan ditekan. Form juga menyertakan *input hidden* berisi ID desa agar sistem dapat mengetahui data desa mana yang harus diperbarui di dalam database.

3. Teknologi yang Digunakan

PHP dan MySQL: Digunakan untuk proses otentikasi sesi, pengambilan data dari basis data, dan pengiriman data pembaruan.

Bootstrap: Untuk tampilan responsif dan modal dialog form pengeditan.

HTML & Form POST: Digunakan untuk pengisian dan pengiriman data dari pengguna ke server.

3. Hasil Pengujian

Metode pengujian perangkat lunak pada sistem ini memakai pengujian *Black box* dan *White box*.

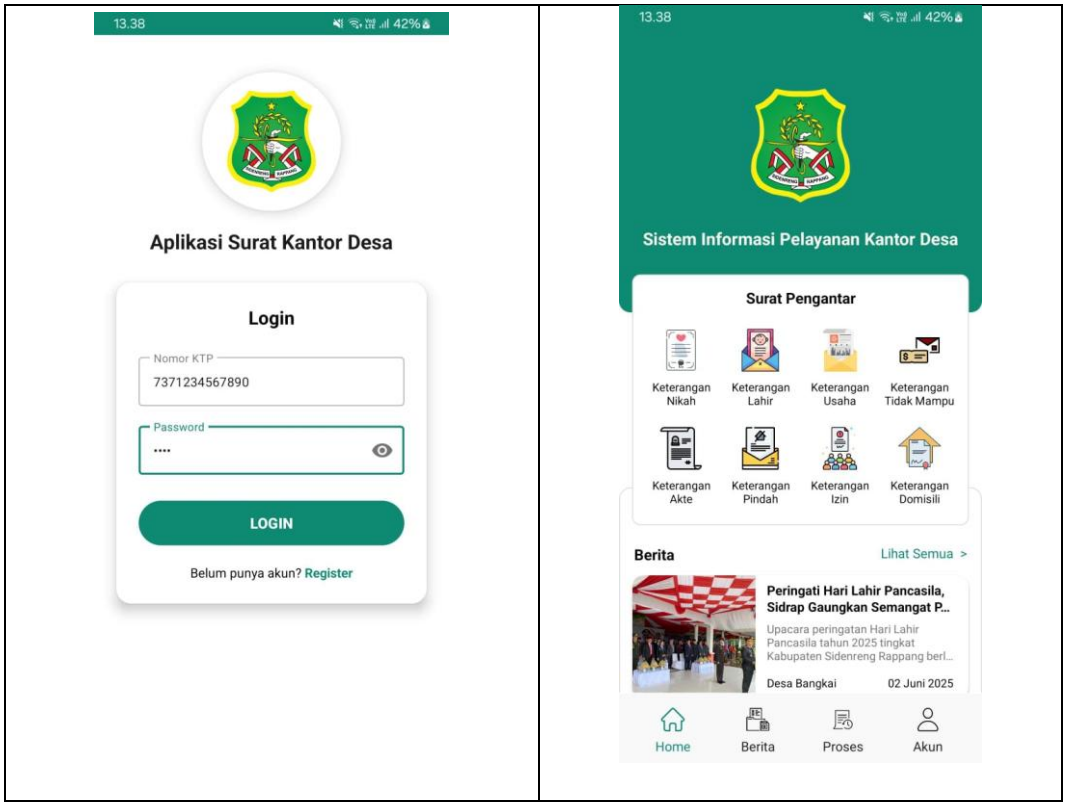
a. Pengujian *Black Box*

Pengujian *Black Box* adalah metode evaluasi perangkat lunak yang berorientasi pada detail aplikasi, fungsionalitas yang tersedia, serta kesesuaian alur fungsi dengan kebutuhan pengguna. Dalam pengujian ini, pemeriksaan tidak melibatkan peninjauan atau pengujian kode sumber program.

1) Pengujian halaman *login user* (android)

Tabel 4. 3. Pengujian halaman *home user*

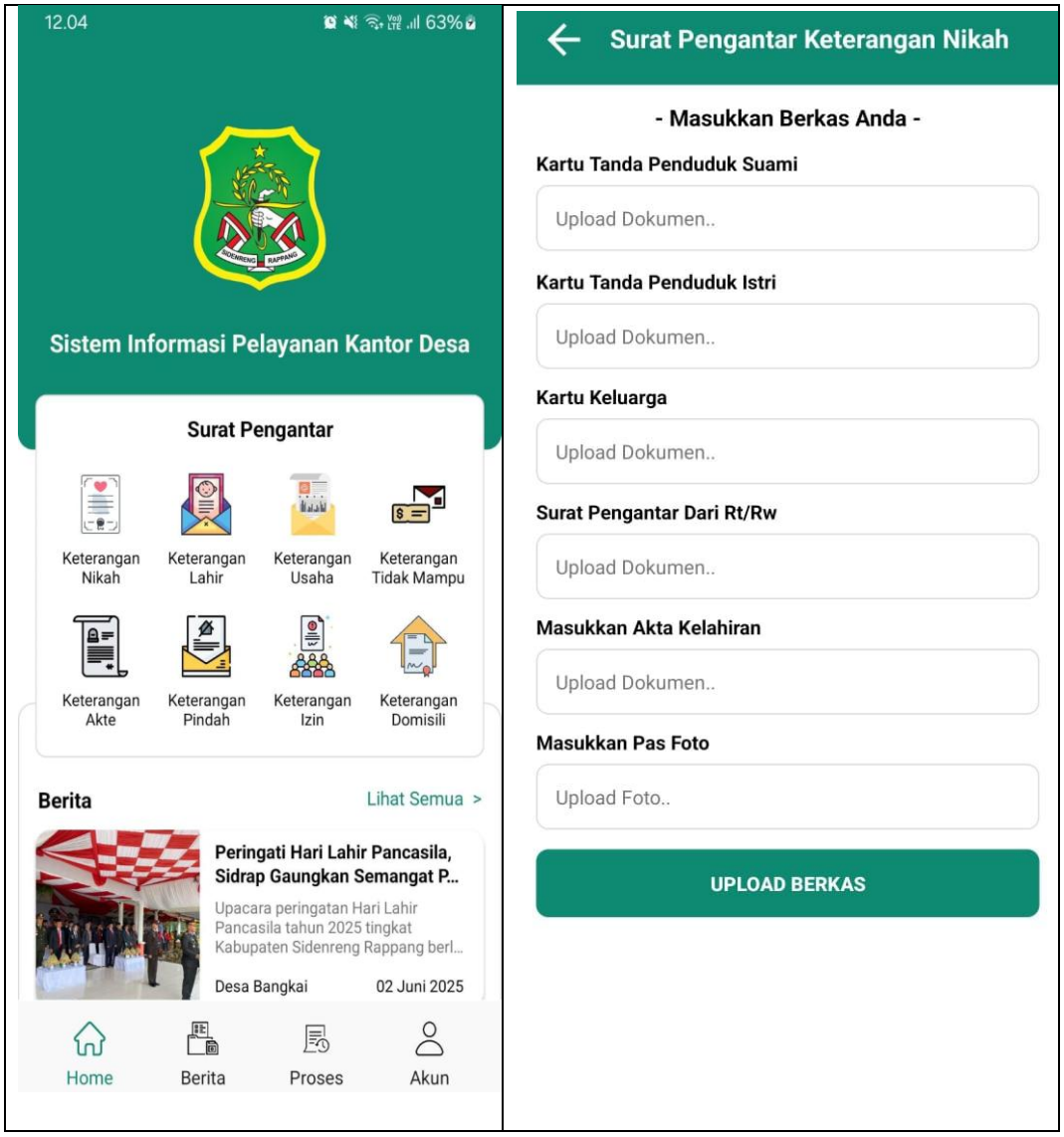
Test Factor	Hasil	Keterangan
User memasukkan data *login* yang valid (nama pengguna dan kata sandi yang benar) pada halaman *login*, kemudian menekan tombol "Masuk".	Berhasil	Berhasil, karena *user* berhasil diarahkan ke halaman utama (home) aplikasi setelah data terverifikasi dengan baik



2) Pengujian halaman layanan surat pengantar *user* (android)

Tabel 4. 4. Pengujian halaman layanan surat pengantar *user*

Test Factor	Hasil	Keterangan
user menekan tombol surat pengantar	Berhasil	Berhasil, karena menampilkan halaman khusus bagi *user* untuk memulai proses pengajuan surat dan mengunggah berkas persyaratan.

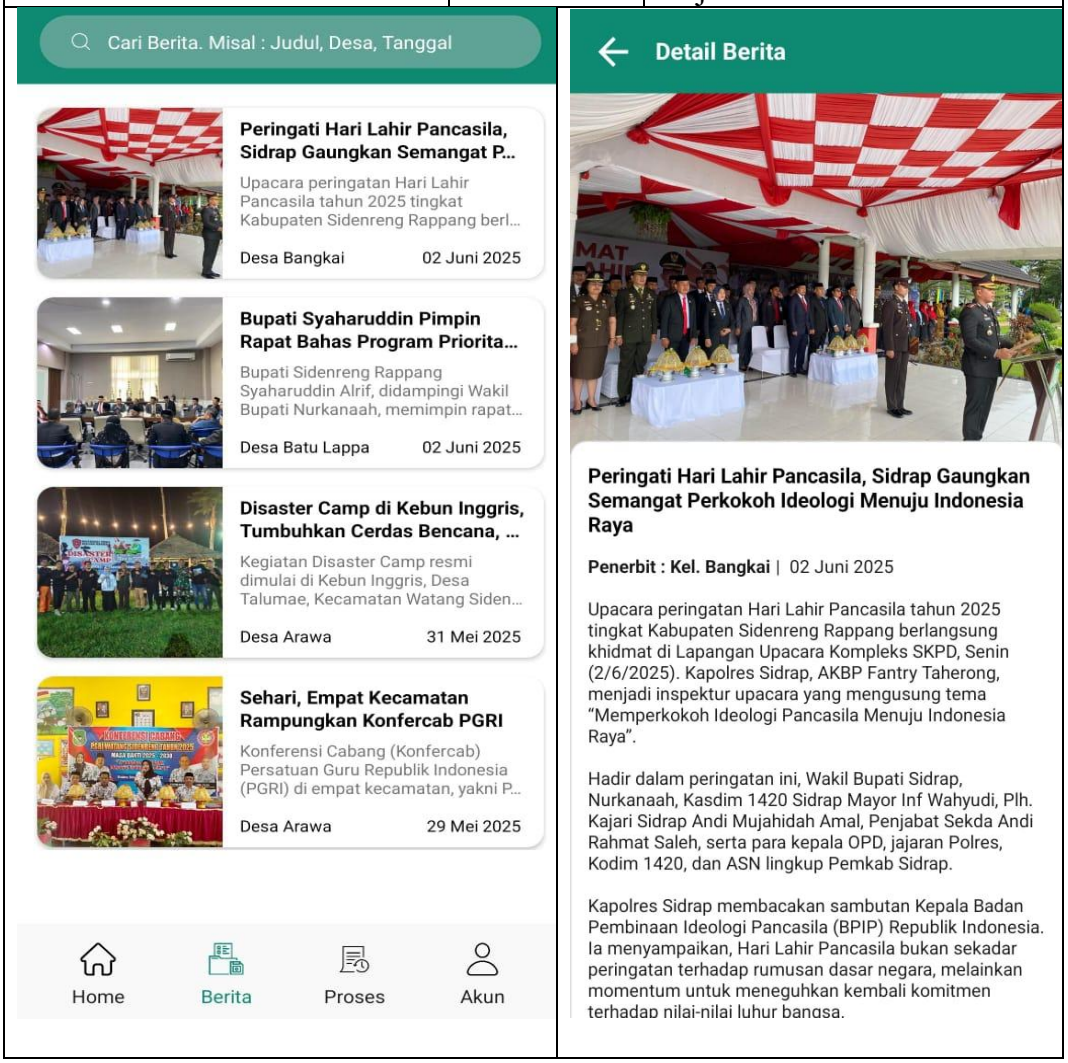


3) Pengujian masuk halaman berita *user* (android)

Tabel 4. 5. Pengujian masuk halaman berita *user*

Test Factor	Hasil	Keterangan
User menekan tombol berita yang tersedia	Berhasil	Berhasil, karena sistem menampilkan detail lengkap dari berita yang dipilih, memungkinkan *user* untuk

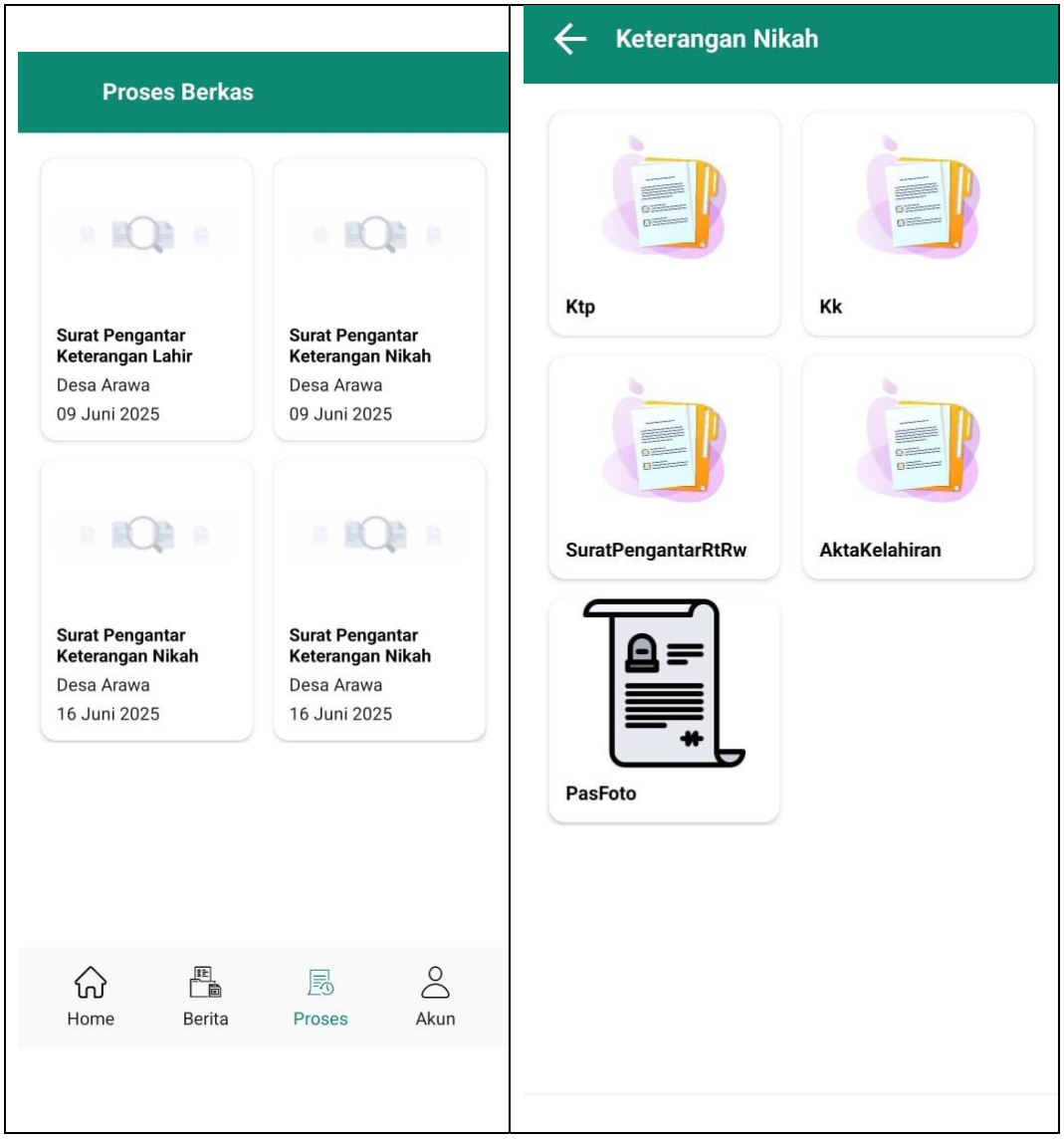
		membaca informasi lebih lanjut.



4) Pengujian masuk halaman proses *user* (android)

Tabel 4. 6. Pengujian masuk halaman proses *user*

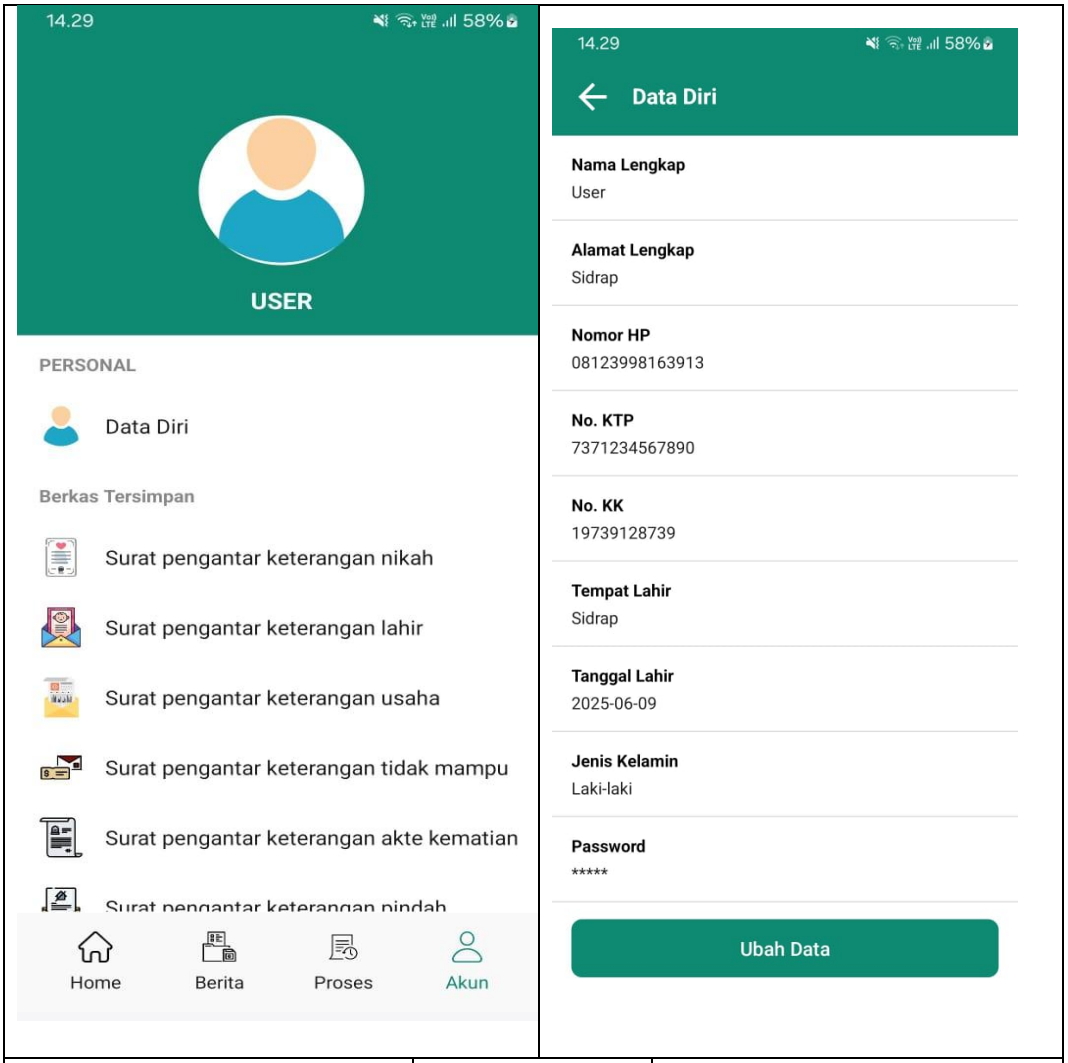
Test Factor	Hasil	Keterangan
User menekan tombol "proses".	Berhasil	Berhasil, karena sistem menampilkan halaman detail proses yang relevan, memungkinkan *user* untuk melihat tahapan pemrosesan surat pengantar.



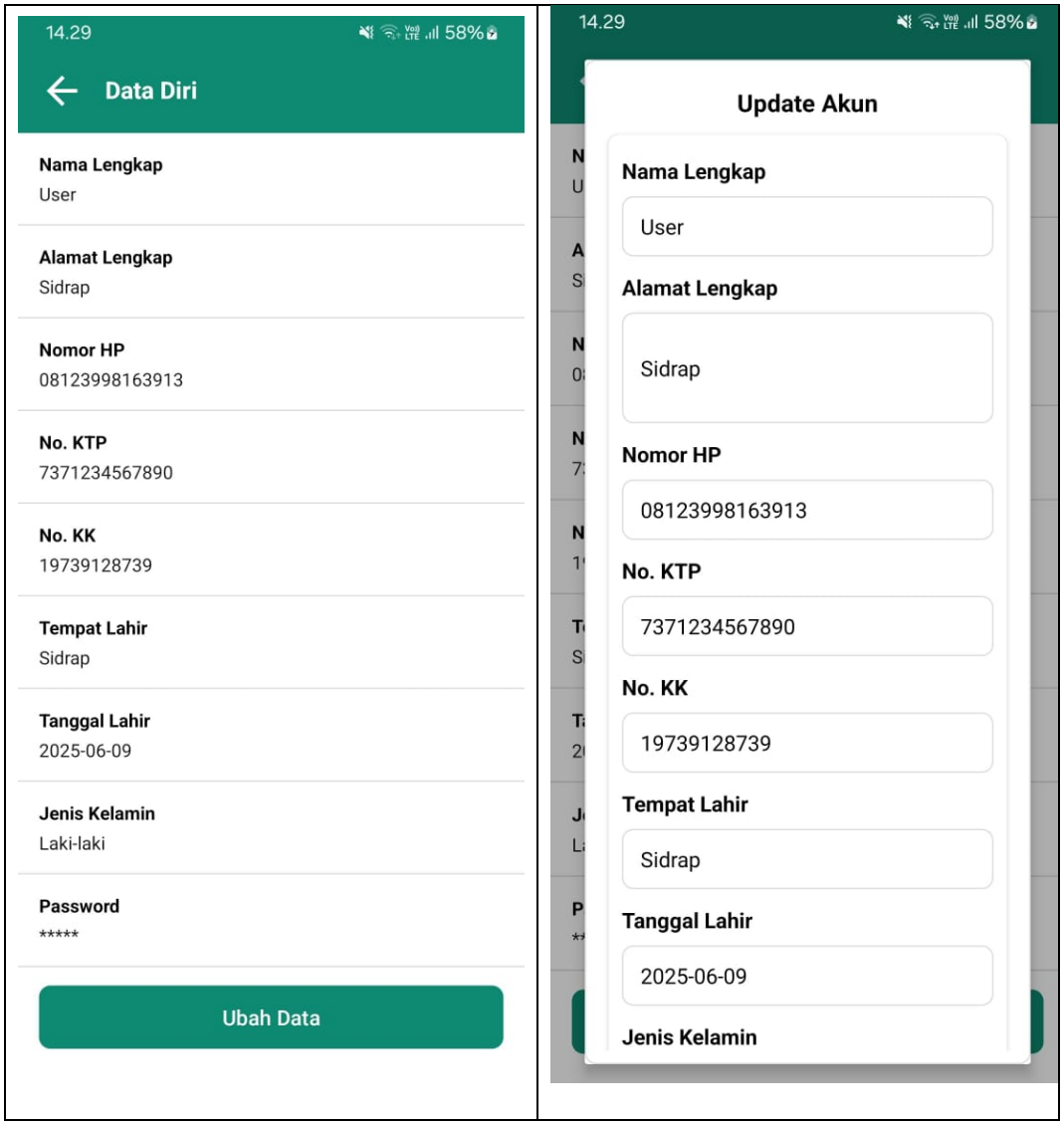
5) Pengujian masuk halaman data diri *user* (android)

Tabel 4. 7. masuk halaman data diri *user*

Test Factor	Hasil	Keterangan
User menekan tombol "data diri"	Berhasil	Berhasil, karena sistem menampilkan halaman detail data diri *user*, memungkinkan *user* untuk melihat informasi pribadinya



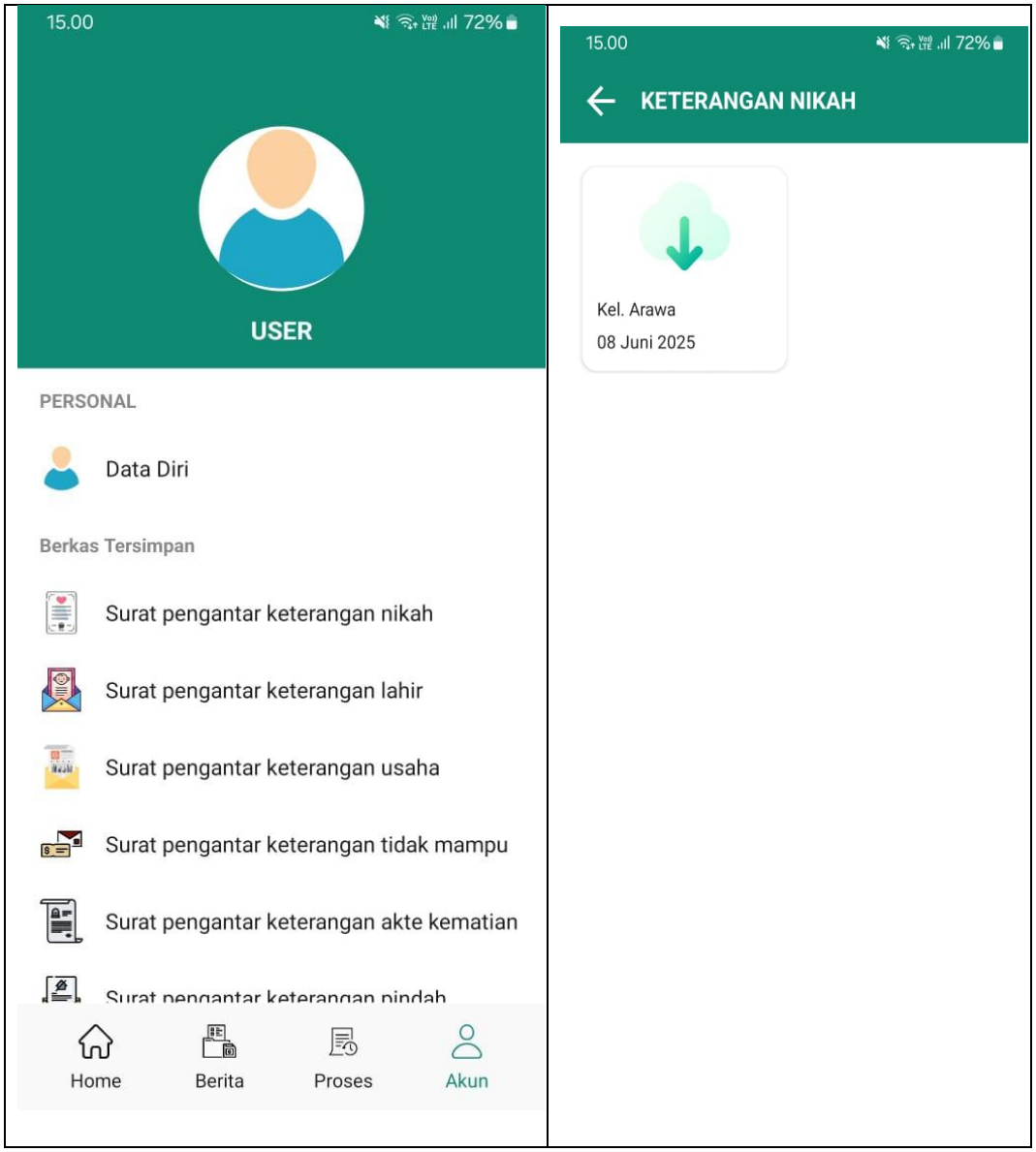
Test Factor	Hasil	Keterangan
User menekan tombol ubah data	Berhasil	Berhasil, karena sistem menampilkan halaman yang memungkinkan *user* untuk melakukan perubahan pada data dirinya.



6) Pengujian halaman riwayat layanan surat pengantar *user* (android)

Tabel 4. 8. Pengujian halaman riwayat layanan surat pengantar *user*

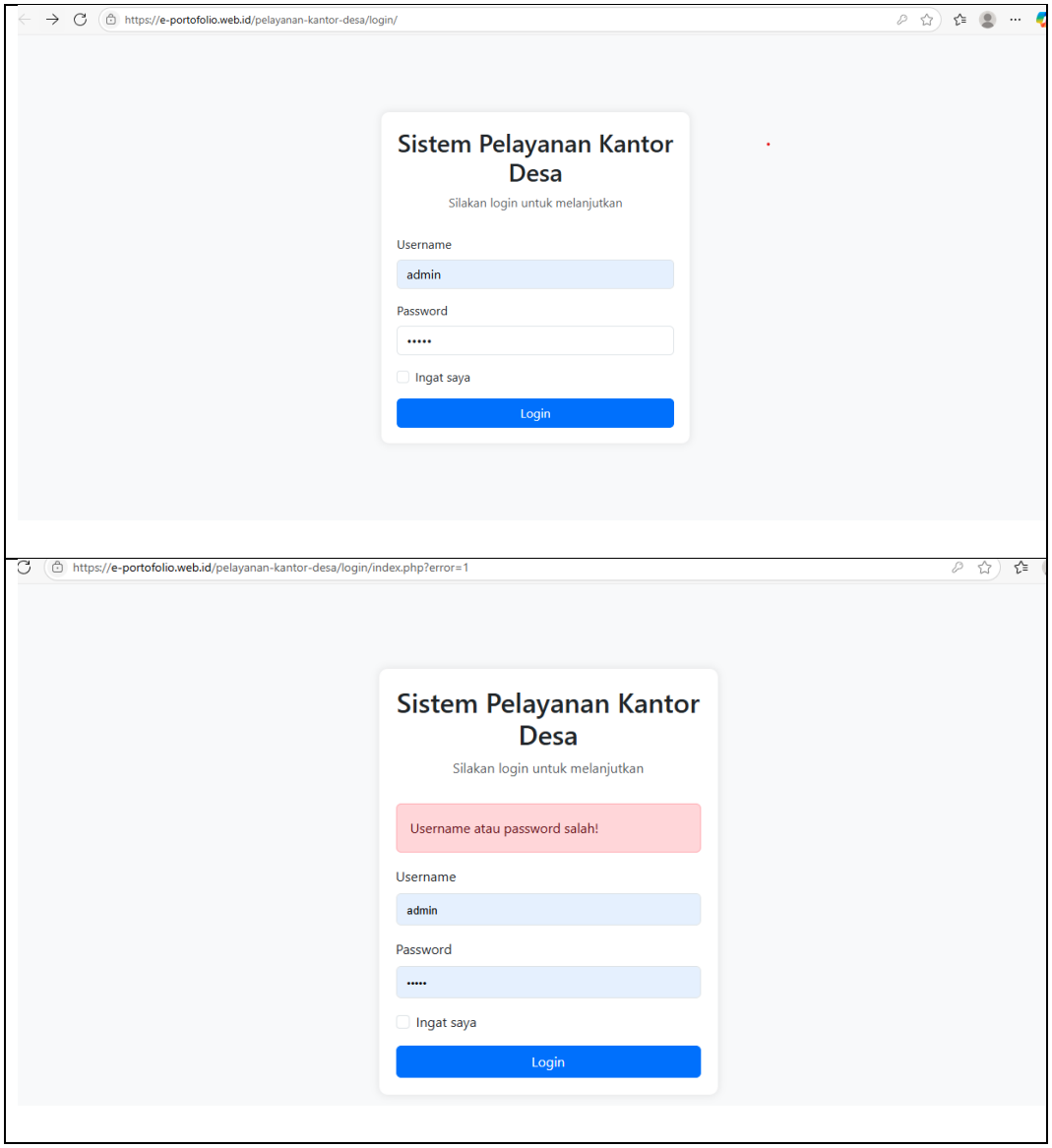
Test Factor	Hasil	Keterangan
User menekan tombol "riwayat layanan surat pengantar"	Berhasil	Berhasil, karena sistem menampilkan detail riwayat layanan surat pengantar, memungkinkan *user* untuk melihat daftar permohonan yang telah diajukan sebelumnya.



7) Pengujian halaman *login* desa/admin

Tabel 4. 9. Pengujian halaman *login* desa/admin

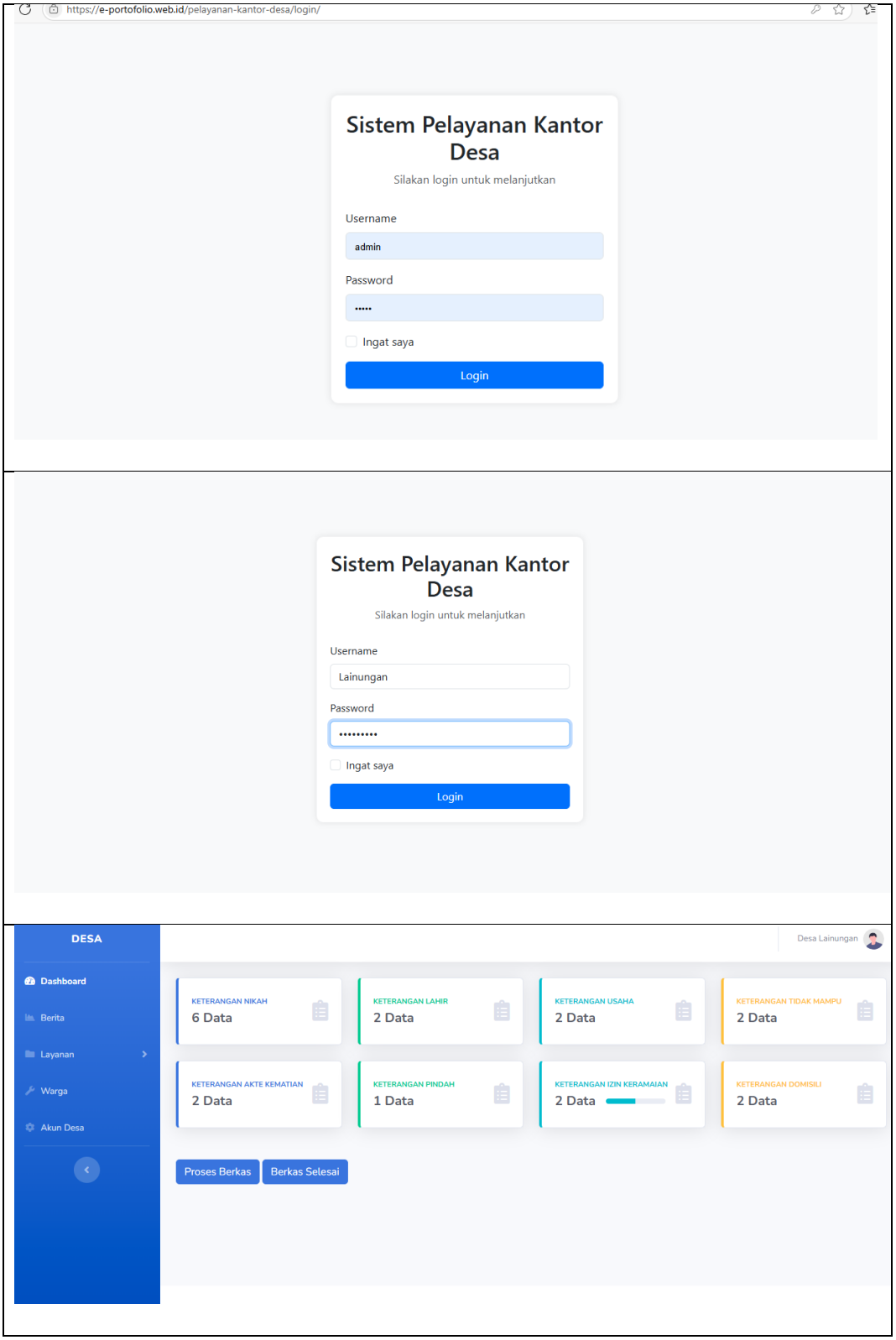
Test Factor	Hasil	Keterangan
Jika desa/admin salah memasukkan *user* name dan password	Berhasil	Berhasil, karena *username* dan *password* desa/admin tidak sesuai, maka tidak dapat login dan sistem menampilkan pesan kesalahan.

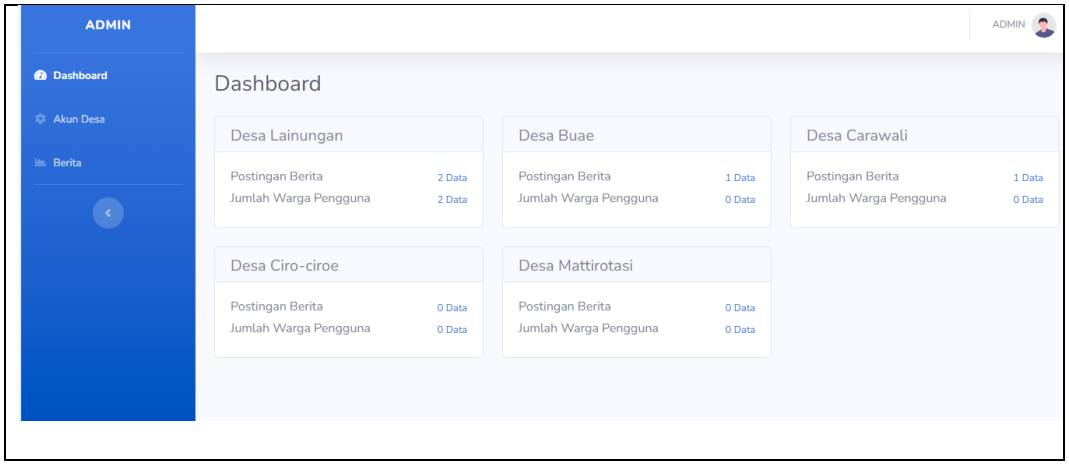


8) Pengujian pada halaman *login* desa/admin

Tabel 4. 10. Pengujian berhasil *login* desa/admin

Test Factor	Hasil	Keterangan
Jika *user name* dan *password* desa/admin yang dimasukkan sudah sesuai	Berhasil	Berhasil, karena dapat menampilkan halaman utama desa/admin

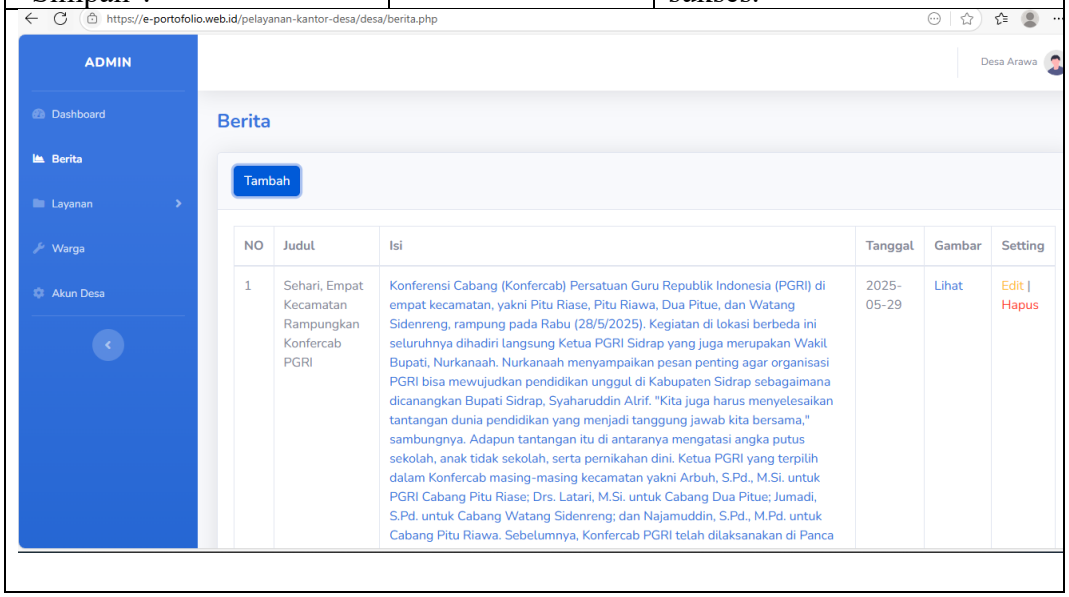

https://e-portofolio.web.id/pelayanan-kantor-desa/login/
Sistem Pelayanan Kantor Desa
Silakan login untuk melanjutkan
Username
admin
Password
Ingat saya
Login
Sistem Pelayanan Kantor Desa
Silakan login untuk melanjutkan
Username
Lainungan
Password
Ingat saya
Login
DESA
Dashboard
Berita
Layanan
Warga
Akun Desa
Desa Lainungan
KETERANGAN NIKAH
6 Data
KETERANGAN LAHIR
2 Data
KETERANGAN USAHA
2 Data
KETERANGAN TIDAK MAMPU
2 Data
KETERANGAN AKTE KEMATIAN
2 Data
KETERANGAN PINDAH
1 Data
KETERANGAN IZIN KERAMAIAN
2 Data
KETERANGAN DOMISILI
2 Data
Proses Berkas
Berkas Selesai

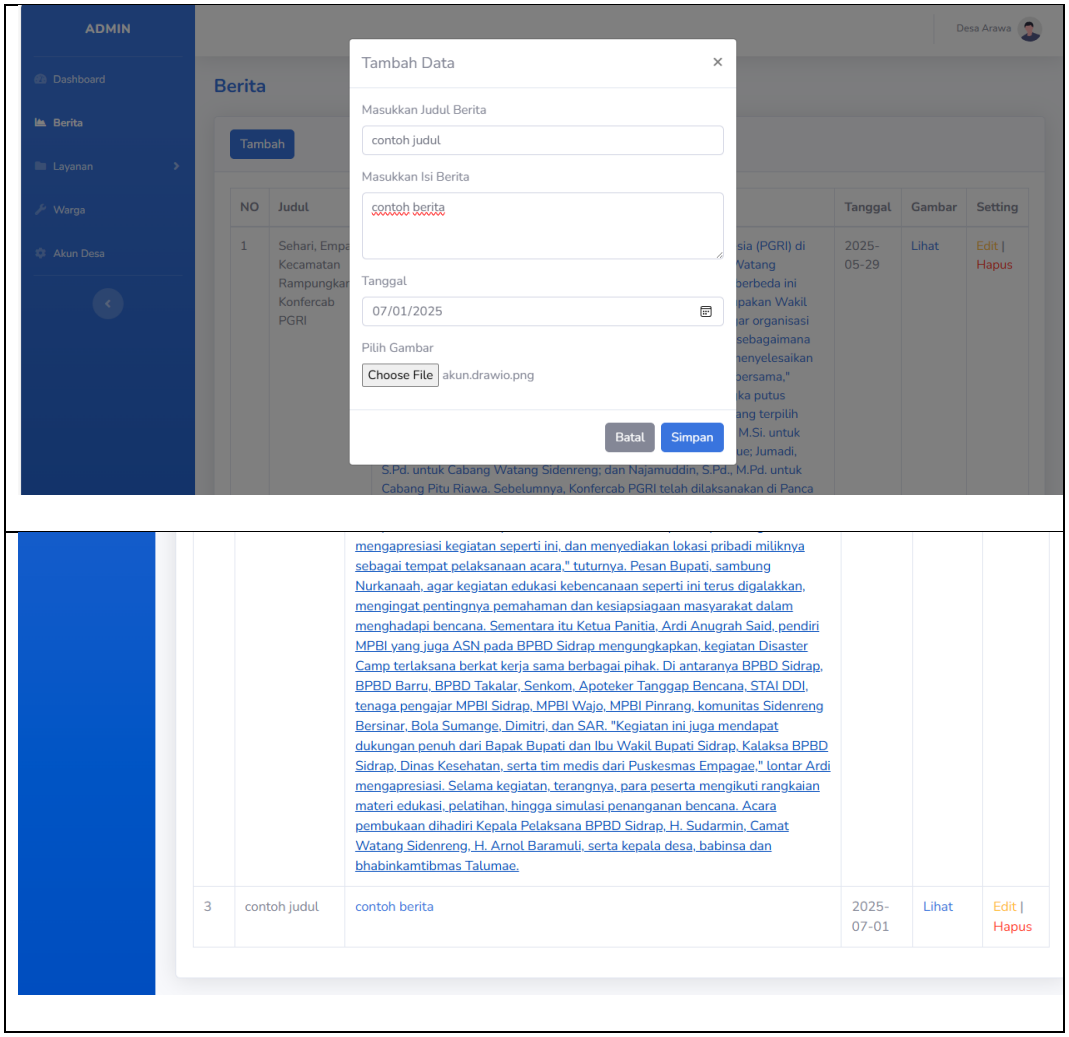


9) Pengujian tambah, edit, hapus berita desa

Tabel 4. 11. Pengujian tambah, edit, hapus berita desa

Test Factor	Hasil	Keterangan
Jika desa mengakses fitur manajemen berita, mengisi formulir penambahan berita dengan data lengkap dan valid (judul, isi, tanggal), lalu menekan tombol "Simpan".	Berhasil	Berhasil, karena sistem berhasil menyimpan data berita baru ke dalam basis data dan menampilkannya pada daftar berita, serta memberikan notifikasi sukses.



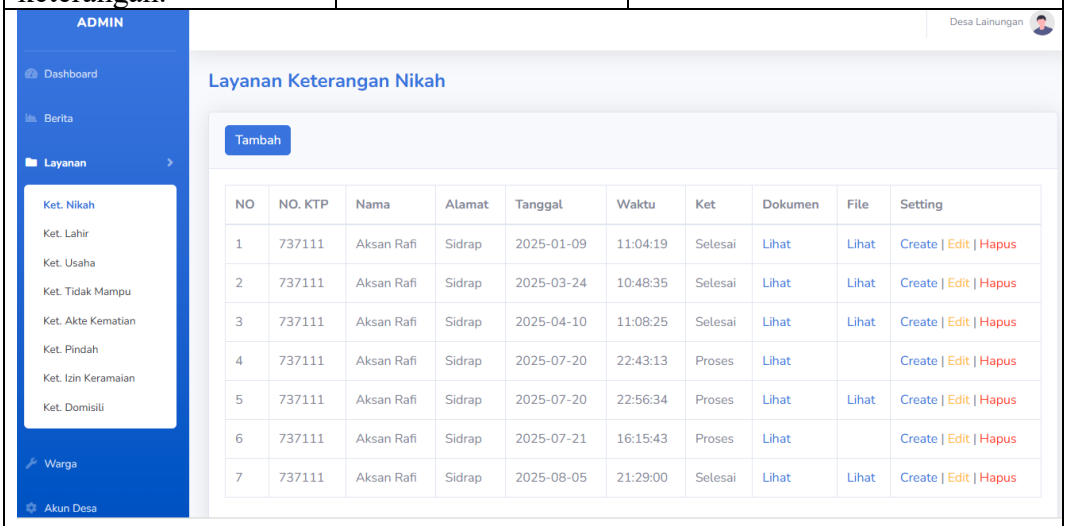


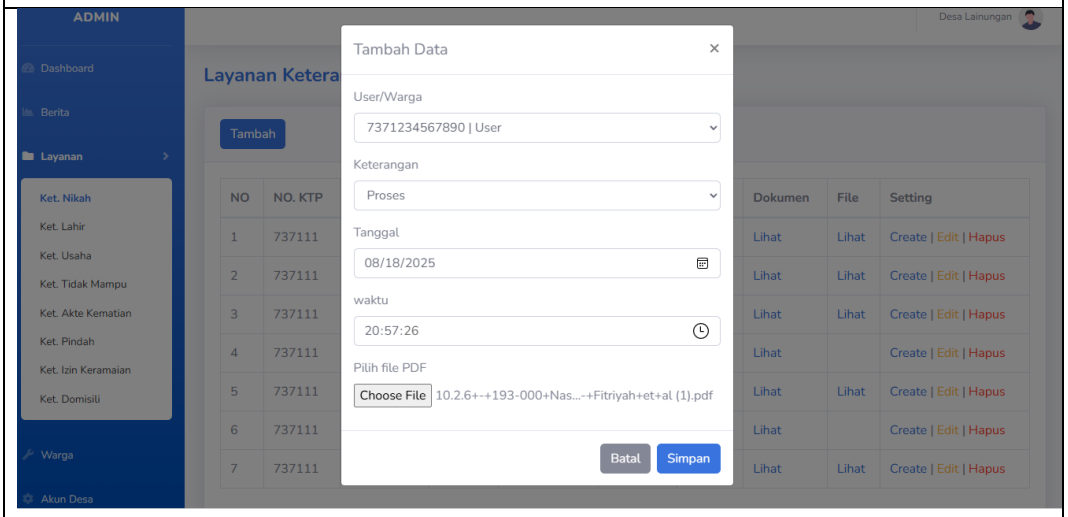
10) Pengujian tambah, edit, hapus dan create layanan surat keterangan

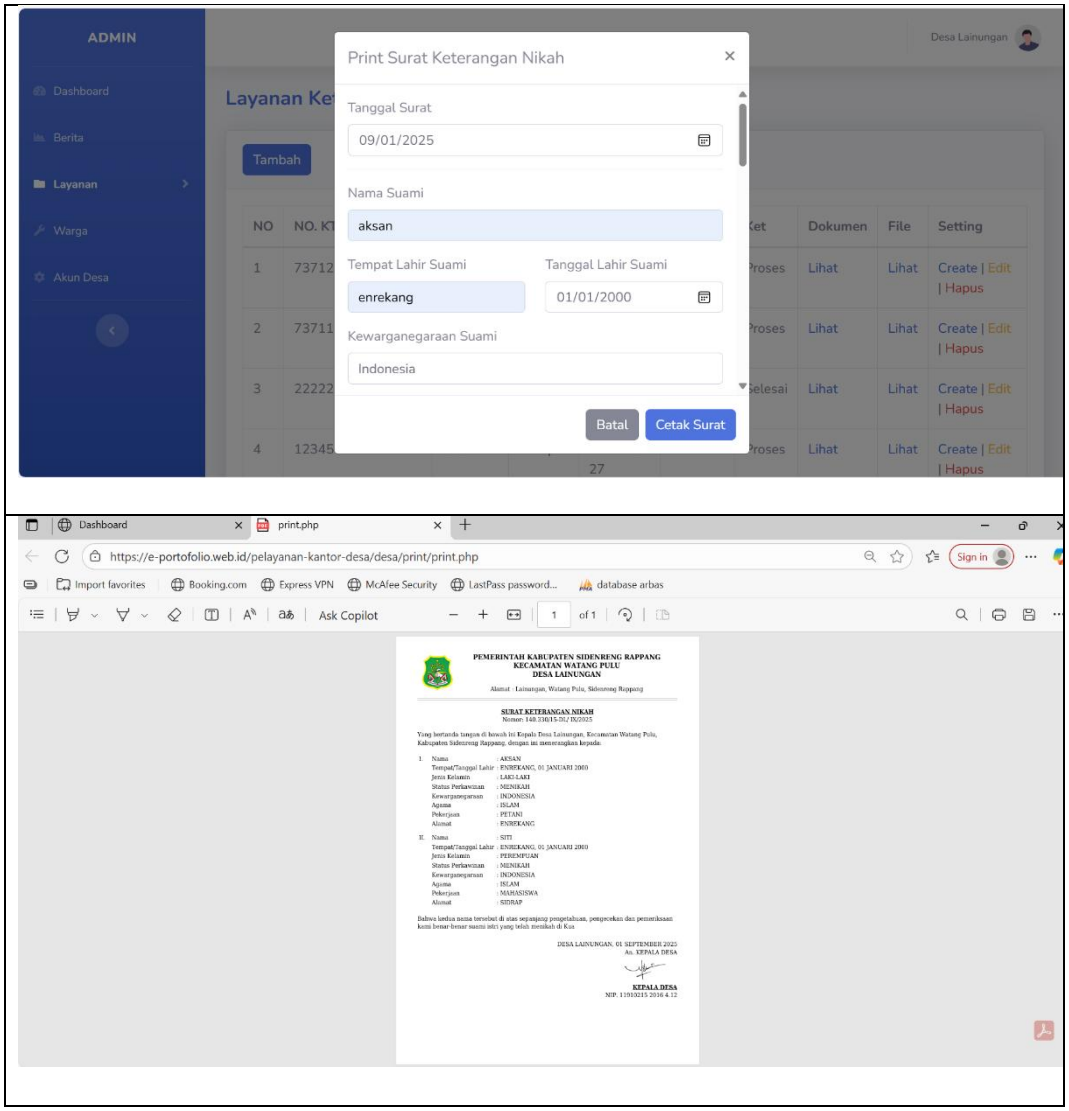
Tabel 4. 12. Pengujian tambah, edit, hapus dan create layanan surat keterangan

Test Factor	Hasil	Keterangan
Jika desa mengakses fitur manajemen layanan surat keterangan, mengisi formulir penambahan jenis layanan dengan data lengkap lalu menekan tombol "Simpan".	Berhasil	Berhasil, karena sistem berhasil menyimpan data layanan surat keterangan baru ke dalam basis data dan menampilkannya pada daftar layanan, serta memberikan notifikasi sukses. Berhasil karena dapat membuat surat keterangan

Desa mengakses menu “create” untuk membuat surat keterangan.		



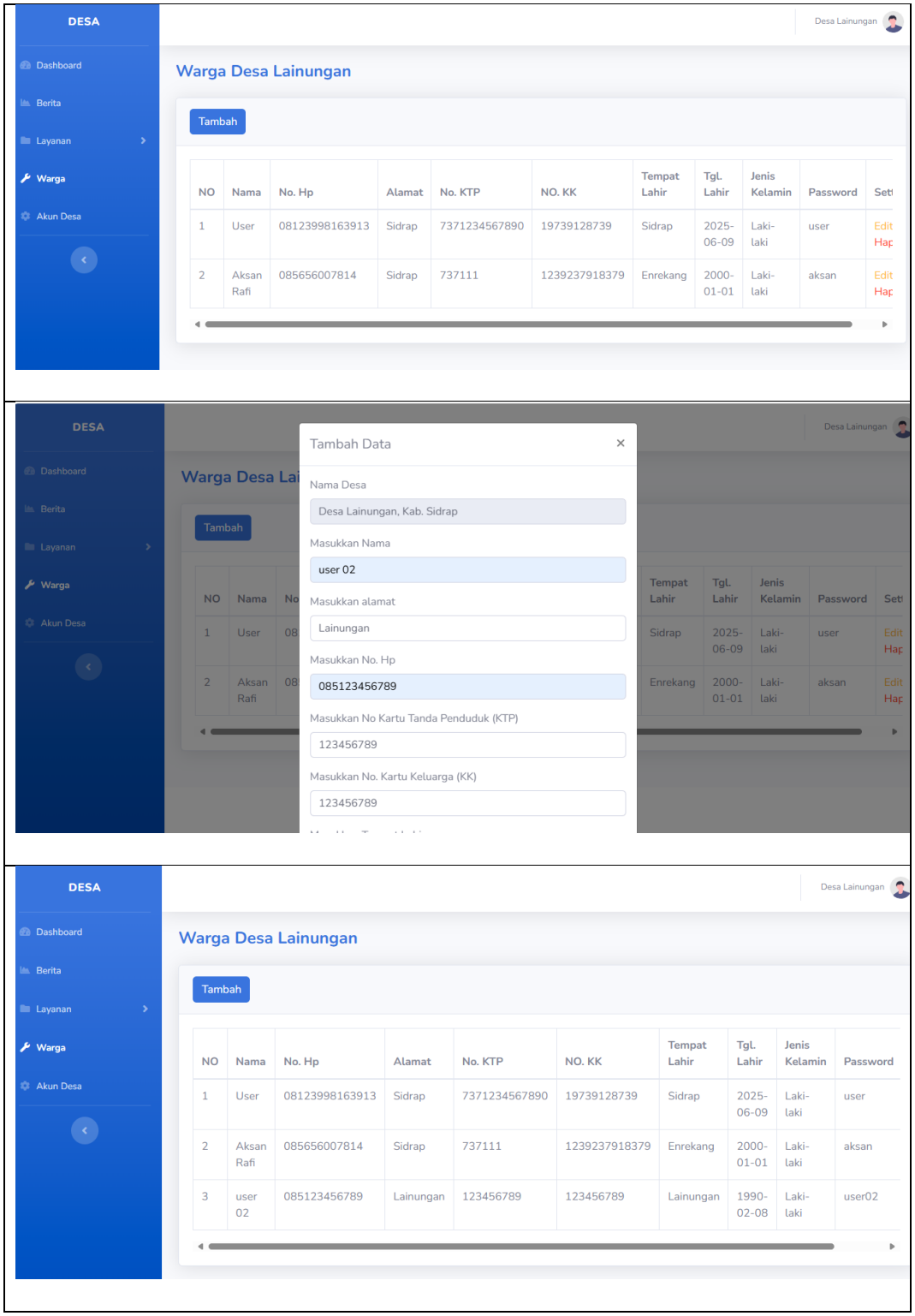




11) Pengujian tambah, edit, hapus informasi warga desa

Tabel 4. 13. Pengujian tambah, edit, hapus informasi warga desa

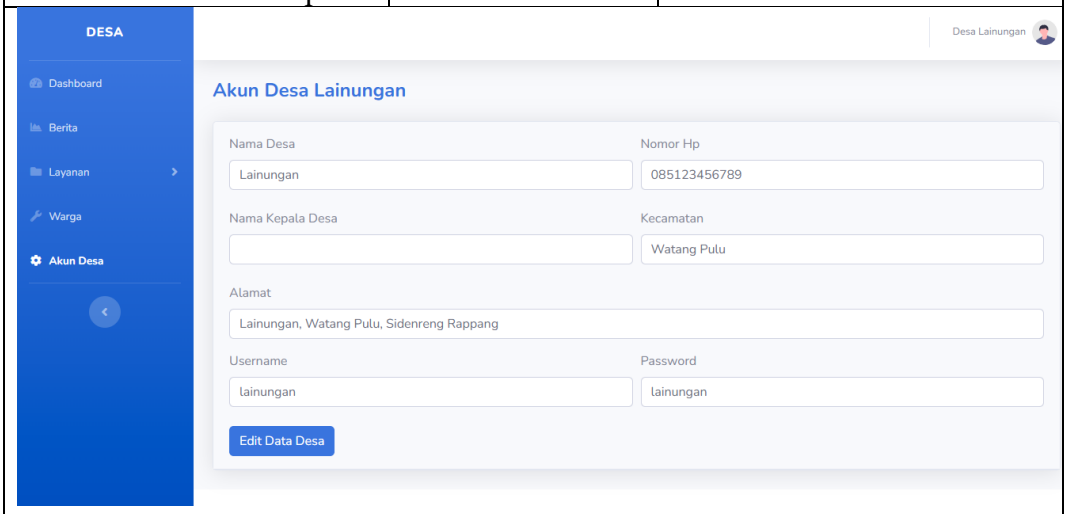
Test Factor	Hasil	Keterangan
Jika desa mengakses fitur manajemen warga, mengisi formulir penambahan warga dengan data lengkap dan valid (NIK, KK, nama, alamat, tanggal lahir), lalu menekan tombol "Simpan".	Berhasil	Berhasil, karena sistem berhasil menyimpan data warga baru ke dalam basis data dan menampilkannya pada daftar warga, serta memberikan notifikasi sukses.

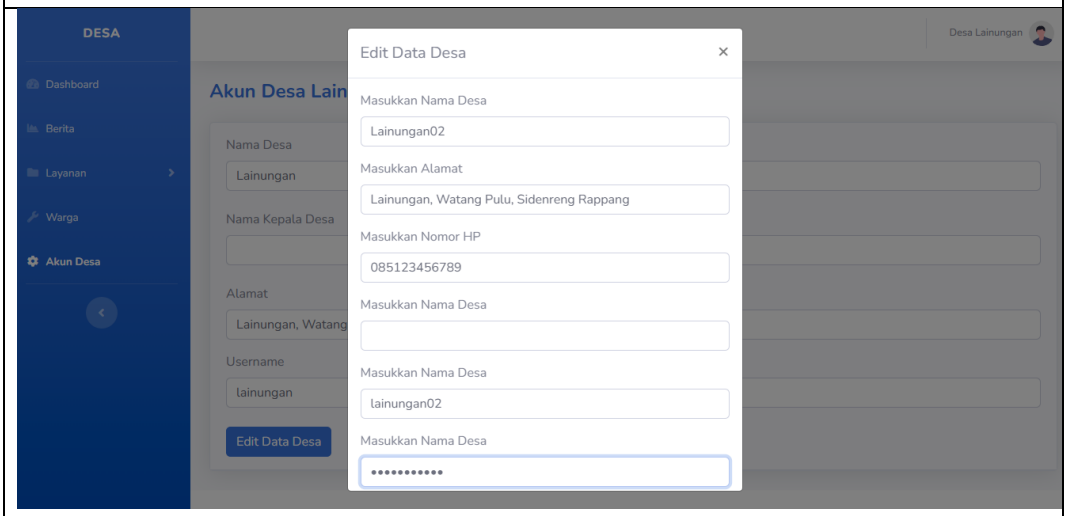


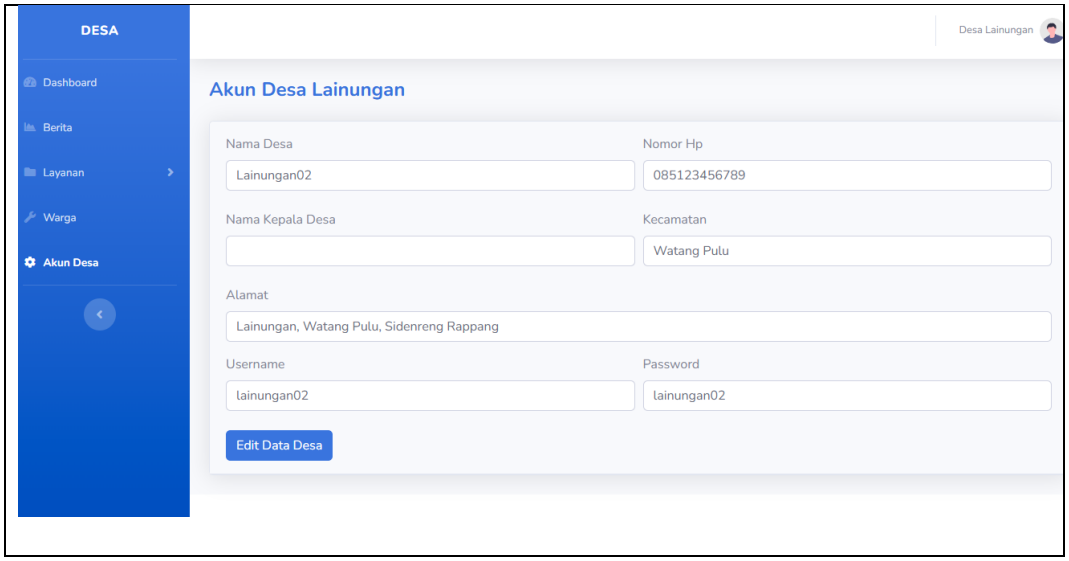
12) Pengujian tambah, edit, hapus data desa

Tabel 4. 14. Pengujian tambah, edit, hapus data desa

Test Factor	Hasil	Keterangan
Jika Admin Desa mengakses halaman pengaturan akun desa, menekan tombol "Edit Data Desa", melakukan perubahan pada informasi yang diinginkan, lalu menekan tombol "Simpan"	Berhasil	Berhasil, karena data akun desa berhasil diperbarui sesuai dengan input yang dimasukkan dan perubahan tersebut terdapat dalam sistem.



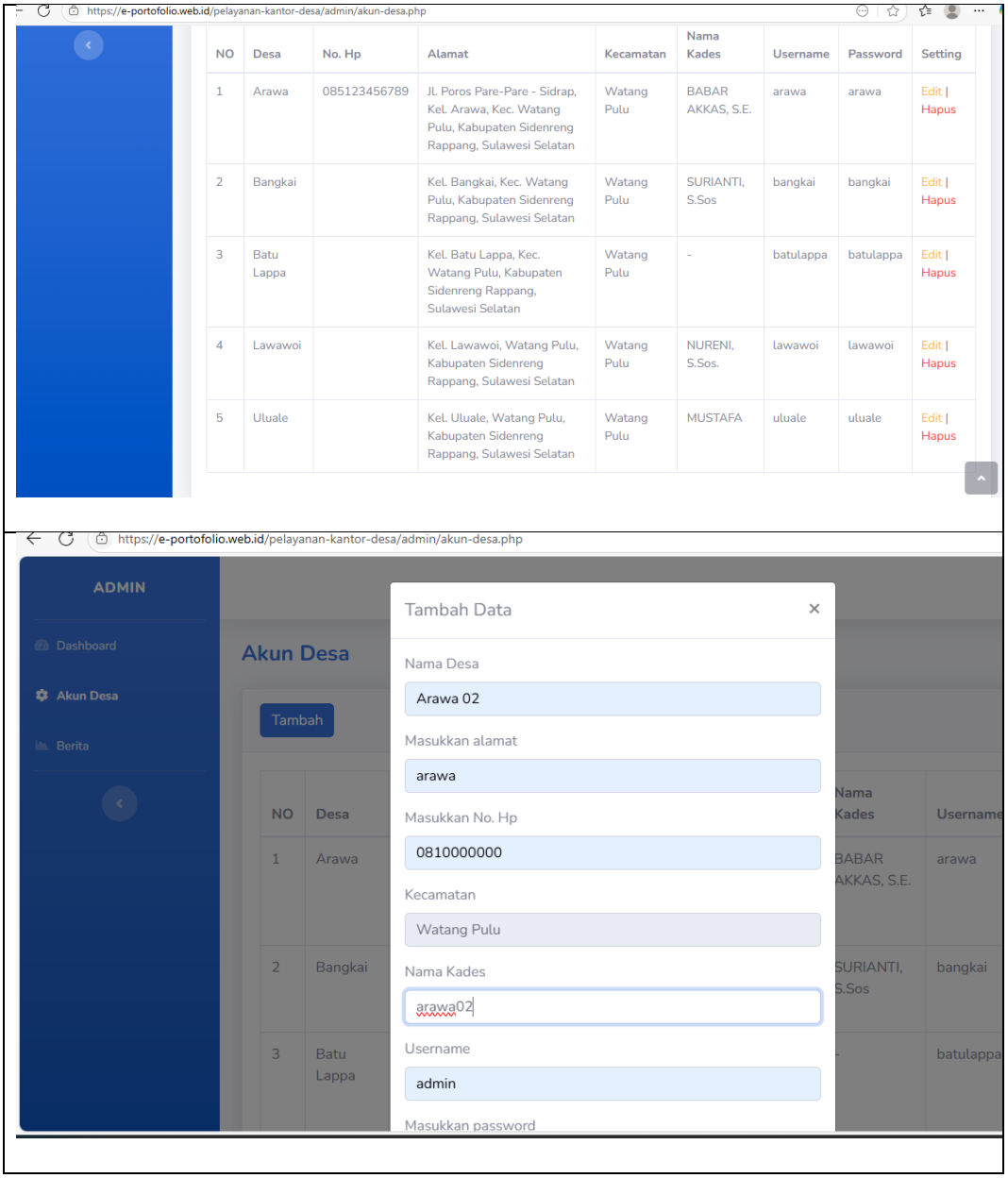




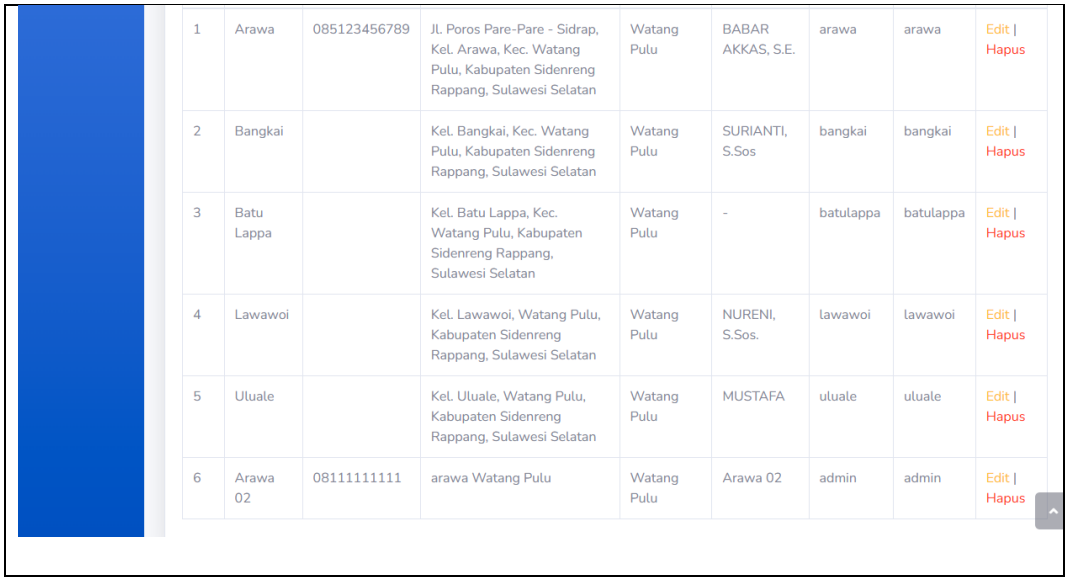
13) Pengujian tambah, edit, hapus akun desa oleh admin

Tabel 4. 15. Pengujian tambah, edit, hapus akun desa oleh admin

Test Factor	Hasil	Keterangan
Jika admin mengakses halaman pengaturan akun desa, menekan tombol "Edit Data Desa", melakukan perubahan pada informasi yang diinginkan (nama desa, alamat kantor, atau kontak), lalu menekan tombol "Simpan".	Berhasil	Berhasil, karena data akun desa berhasil diperbarui sesuai dengan input yang dimasukkan dan perubahan tersebut terrdapat sistem serta pada informasi yang ditampilkan kepada *user*.


https://e-portofolio.web.id/pelayanan-kantor-desa/admin/akun-desa.php
NO
Desa
No. Hp
Alamat
Kecamatan
Nama Kades
Username
Password
Setting
1
Arawa
085123456789
Jl. Poros Pare-Pare - Sidrap, Kel. Arawa, Kec. Watang Pulu, Kabupaten Sidenreng Rappang, Sulawesi Selatan
Watang Pulu
BABAR AKKAS, S.E.
arawa
arawa
Edit | Hapus
2
Bangkai
Kel. Bangkai, Kec. Watang Pulu, Kabupaten Sidenreng Rappang, Sulawesi Selatan
Watang Pulu
SURIANTI, S.Sos
bangkai
bangkai
Edit | Hapus
3
Batu Lappa
Kel. Batu Lappa, Kec. Watang Pulu, Kabupaten Sidenreng Rappang, Sulawesi Selatan
Watang Pulu
-
batulappa
batulappa
Edit | Hapus
4
Lawawoi
Kel. Lawawoi, Watang Pulu, Kabupaten Sidenreng Rappang, Sulawesi Selatan
Watang Pulu
NURENI, S.Sos.
lawawoi
lawawoi
Edit | Hapus
5
Uluale
Kel. Uluale, Watang Pulu, Kabupaten Sidenreng Rappang, Sulawesi Selatan
Watang Pulu
MUSTAFA
uluale
uluale
Edit | Hapus

https://e-portofolio.web.id/pelayanan-kantor-desa/admin/akun-desa.php
ADMIN
Dashboard
Akun Desa
Berita
Akun Desa
Tambah
Tambah Data
Nama Desa
Arawa 02
Masukkan alamat
arawa
Masukkan No. Hp
0810000000
Kecamatan
Watang Pulu
Nama Kades
arawa02
Username
admin
Masukkan password

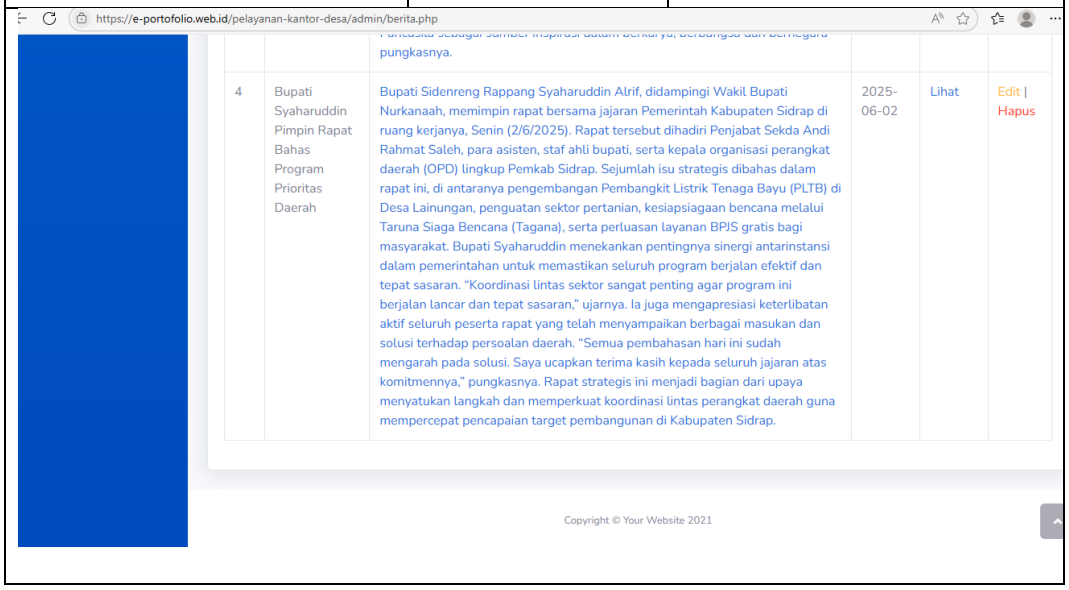


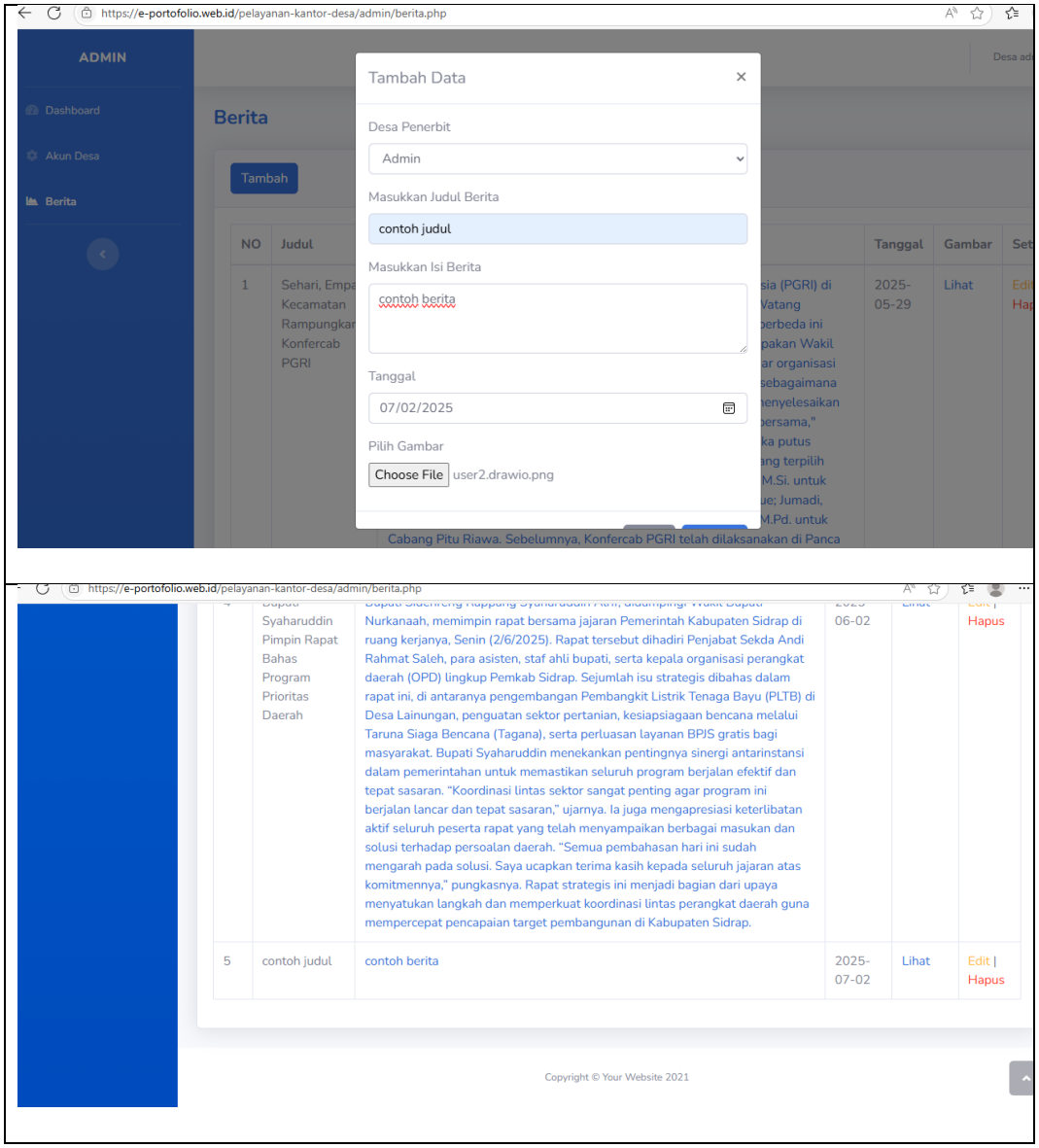
1	Arawa	085123456789	Jl. Poros Pare-Pare - Sidrap, Kel. Arawa, Kec. Watang Pulu, Kabupaten Sidenreng Rappang, Sulawesi Selatan	Watang Pulu	BABAR AKKAS, S.E.	arawa	arawa	Edit | Hapus
2	Bangkai		Kel. Bangkai, Kec. Watang Pulu, Kabupaten Sidenreng Rappang, Sulawesi Selatan	Watang Pulu	SURIANTI, S.Sos	bangkai	bangkai	Edit | Hapus
3	Batu Lappa		Kel. Batu Lappa, Kec. Watang Pulu, Kabupaten Sidenreng Rappang, Sulawesi Selatan	Watang Pulu	-	batulappa	batulappa	Edit | Hapus
4	Lawawoi		Kel. Lawawoi, Watang Pulu, Kabupaten Sidenreng Rappang, Sulawesi Selatan	Watang Pulu	NURENI, S.Sos.	lawawoi	lawawoi	Edit | Hapus
5	Uluale		Kel. Uluale, Watang Pulu, Kabupaten Sidenreng Rappang, Sulawesi Selatan	Watang Pulu	MUSTAFA	uluale	uluale	Edit | Hapus
6	Arawa 02	08111111111	arawa Watang Pulu	Watang Pulu	Arawa 02	admin	admin	Edit | Hapus

14) Pengujian tambah, edit, hapus berita desa oleh admin

Tabel 4. 16. Pengujian tambah, edit, hapus berita desa oleh admin

Test Factor	Hasil	Keterangan
Jika admin mengakses fitur manajemen berita, mengisi formulir penambahan berita dengan data lengkap dan valid (judul, isi, tanggal), lalu menekan tombol "Simpan"	Berhasil	Berhasil, karena sistem berhasil menyimpan data berita baru ke dalam basis data dan menampilkannya pada daftar berita, serta memberikan notifikasi sukses.



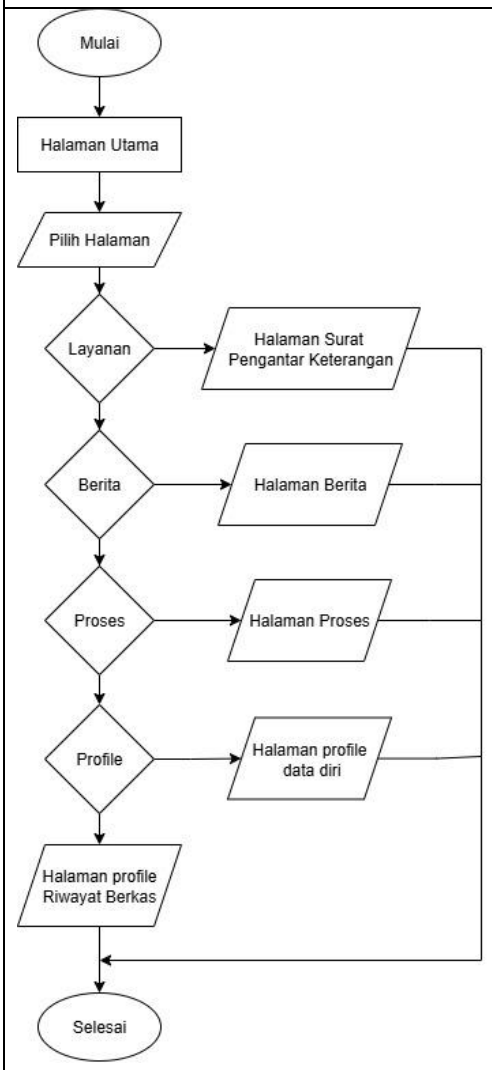
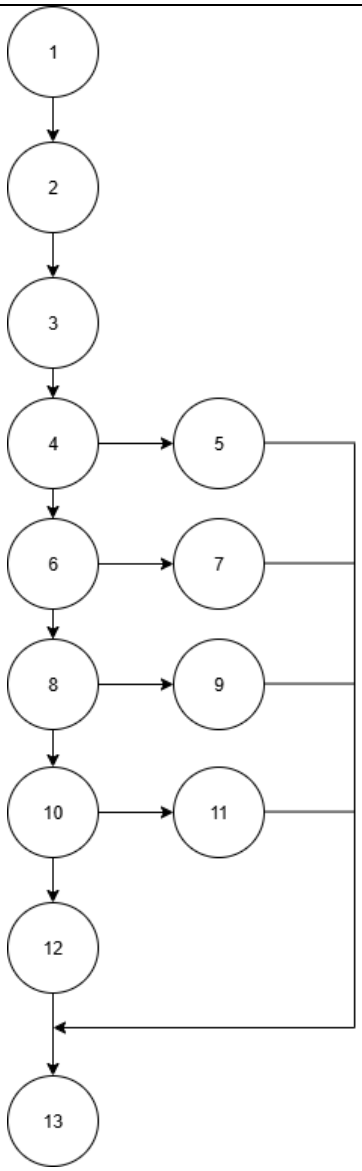


2.Pengujian *white box*

Pengujian white box adalah pengujian yang didasarkan pada pengecekan detail perancangan, menggunakan struktur kontrol dari desain program secara prosedural untuk membagi pengujian ke dalam beberapa kasus pengujian. (Henriani dan Ade Astuty, 2024)

a. *White box user*

Tabel 4. 17. *White Box user*

Flowchart	*Flowgraph*
	

Menghitung jumlah *Region*, *Cyclomatic Complexity* dan *Independent Path.*

Independent Path.

1) Path 1 = 1-2-3-4-5-13

2) Path 2 = 1-2-3-4-6-7-13

3) Path 3 = 1-2-3-4-6-8, 9-13

4) Path 4 = 1-2-3-4-6-8-10-11-13

5) Path 5 = 1-2-3-4-6-8-10-12-13

Cyclomatic Complexity.

V(G) = E-N+2

= 16-13+2 = 5

V(G) = P+1

=4+1

=5

Region = 5

Grafik Matrik

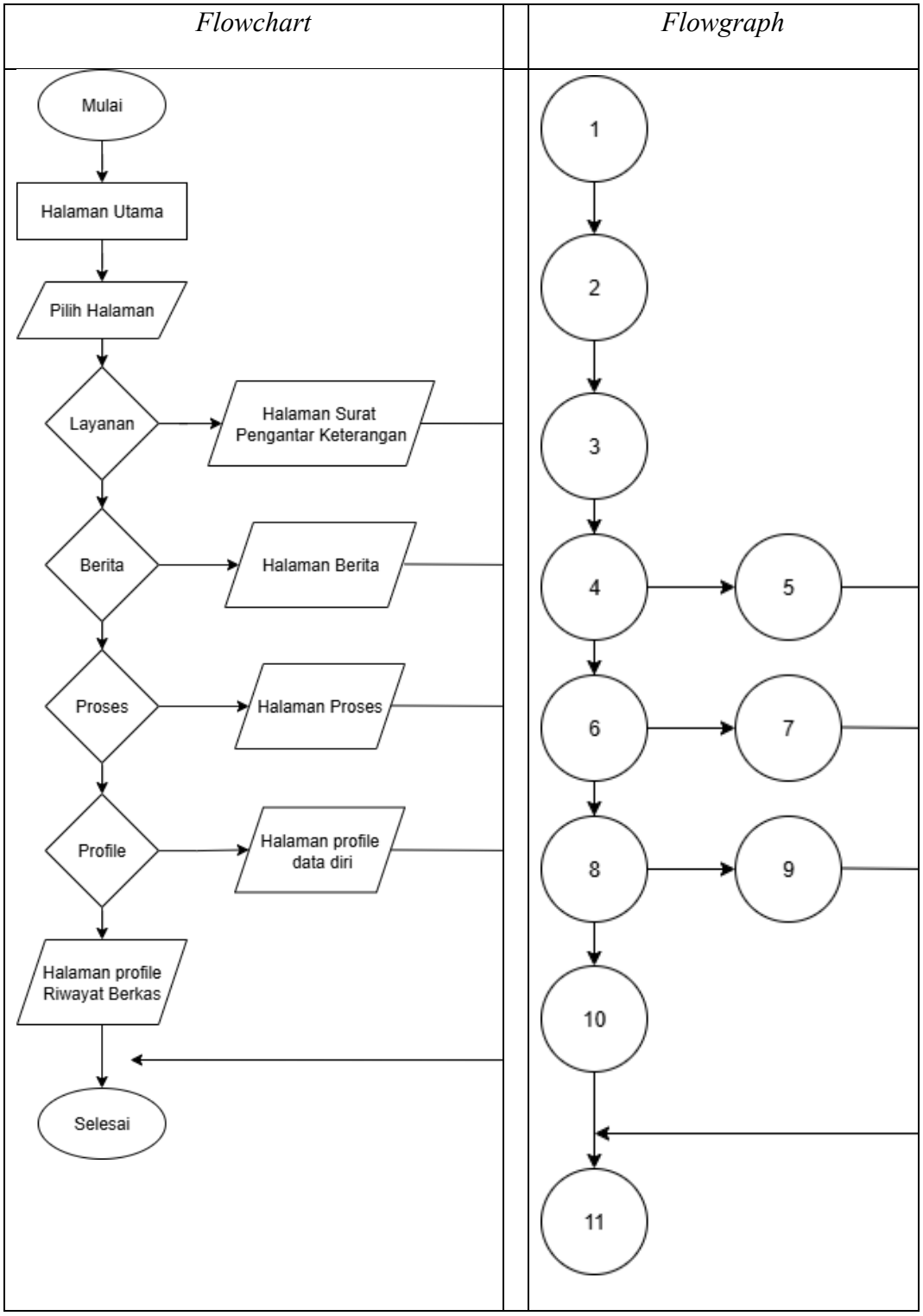
Tabel 4. 18. *White Box* Perhitungan *user*

	1	2	3	4	5	6	7	8	9	10	11	12	13	E-1
1		1												1-1=0
2			1											1-1=0
3				1										1-1=0
4					1	1								2-1=1
5												1		1-1=0
6							1	1						2-1=1
7												1		1-1=0
8										1	1			2-1=1
9												1		1-1=0
10										1	1			2-1=1
11												1		1-1=0
12												1		1-1=0
13														
SUM (E+1)														4+1=5

Berdasarkan perhitungan diatas, *Region* = 5, *Independent Path* = 5, *Cyclomatic Complexity* = 5, maka *v* dinyatakan bebas dari kesalahan.

b. *White box* desa

Tabel 4. 19. *White Box* desa



Menghitung jumlah *Region*, *Cyclomatic Complexity* dan *Independent Path.*

Independent Path.

1) Path 1 = 1-2-3-4-5-11

2) Path 2 = 1-2-3-4-6-7-11

3) Path 3 = 1-2-3-4-6-8-9-11

4) Path 4 = 1-2-3-4-6-8-10-11

Cyclomatic Complexity.

V(G) = E-N+2

= 13-11+2 = 4

V(G) = P+1

=3+1

=4

Region = 5

Grafik Matrik

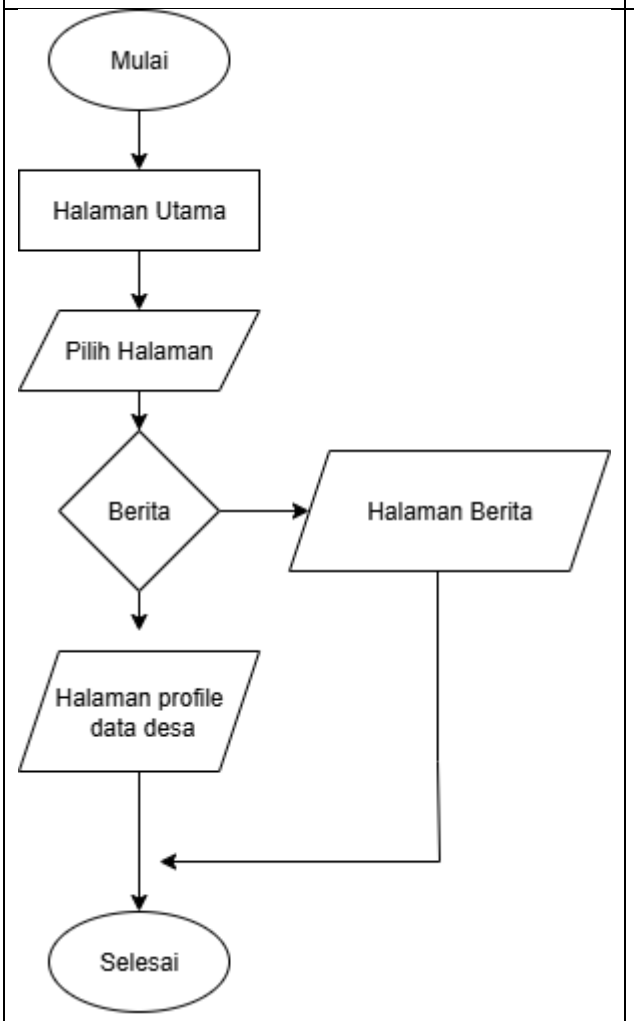
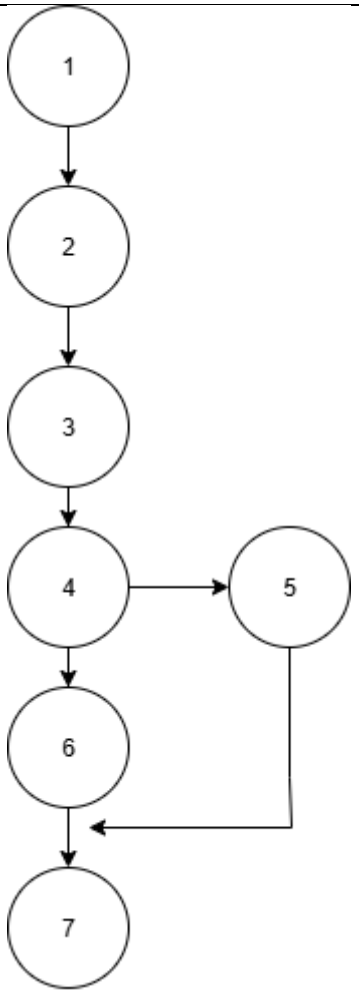
Tabel 4. 20. *White Box* Perhitungan desa

	1	2	3	4	5	6	7	8	9	10	11	E-1
1		1										1-1=0
2			1									1-1=0
3				1								1-1=0
4					1	1						2-1=1
5											1	1-1=0
6							1	1				2-1=1
7											1	1-1=0
8									1	1		2-1=1
9											1	1-1=0
10											1	1-1=0
11												
SUM (E+1)												3+1=4

Berdasarkan perhitungan diatas, Region = 4, *Independent Path* = 4, *Cyclomatic Complexity* = 4, maka dinyatakan bebas dari kesalahan.

c. *White box* admin

Tabel 4. 21. *White Box* admin

Flowchart	*Flowgraph*
	

Menghitung jumlah *Region*, *Cyclomatic Complexity* dan *Independent Path.*

Independent Path.

1) Path 1 = 1-2-3-4-5-7

2) Path 2 = 1-2-3-4-6-7

Cyclomatic Complexity.

V(G) = E-N+2

= 7-7+2 = 2

V(G) = P+1

=1+1

=2

Region = 2

Grafik Matrik

Tabel 4. 22. *White Box* Perhitungan admin

	1	2	3	4	5	6	7	E-1
1		1						1-1=0
2			1					1-1=0
3				1				1-1=0
4					1	1		2-1=1
5							1	1-1=0
6							1	1-1=0
7								
SUM (E+1)								1+1=2

Berdasarkan perhitungan diatas, Region = 2, *Independent Path* = 2, *Cyclomatic Complexity* = 2, maka dinyatakan bebas dari kesalahan.

BAB V

KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan implementasi dan hasil pengujian yang telah dilakukan, dapat disimpulkan bahwa aplikasi berfungsi dengan baik sesuai dengan tujuan awalnya. Aplikasi ini, yang berjalan pada *smartphone* berbasis Android, merupakan sistem pelayanan administrasi kantor desa untuk Kecamatan Watang Pulu, Kabupaten Sidrap. Kehadiran aplikasi ini memberikan kemudahan signifikan bagi warga dan perangkat desa dalam mengurus proses pembuatan surat pengantar.

B. Saran

Saran yang diberikan sebagai bentuk referensi dari pengembangan sistem yang telah dibangun adalah untuk penelitian selanjutnya diharapkan dapat mengembangkan lagi sistem yang berbasis android dengan menambahkan fitur-fitur terbaru seperti daftar nomor antrian, layanan penulisan buku agenda surat masuk dan keluar, disposisi serta pengaduan masyarakat yang dapat membantu meningkatkan kualitas pelayanan kantor desa.

DAFTAR PUSTAKA

Arifianto, R. (2017). *Materi Flowcharst. From* https://rahmatarifianto.wordpress .com/2014/11/20/pengertian-*Flowchart*-dan-jenis-jenisnya.html. (11 Nov 2021).

Aristoni; Any Ismayawati, (2020). *Implementasi Pemerintah Daerah Dalam Pengembangan Model Pelayanan Publik Berbasis Sistem Informasi Desa Pada Pemerintahan Desa Di Kabupaten Jepara Perspektif Islam Terapan*. Skripsi. Kudus : Institut Agama Islam Negeri.

Hamzah, S., Hastuty, A., Selao, A., Basri, M., & Wafiah, A. (2025). Aplikasi Pembayaran Wahana Menggunakan Radio Frequency Identification di Kawasan Puncak Bila Berbasis Web. *JEKIN-Jurnal Teknik Informatika*, *5*(2), 884-895.

Hansun, S. dkk. (2018). *Pemrograman Android dengan Android Studio IDE*. Yogyakarta: Andi Offset.

Hardana. (2018). *Membuat Aplikasi Android Dengan Bahasa Kotlin*. Bogor : Mitra Sinergi Optima.

Hastuty, A. (2023). *Dasar Pemograman.* Makassar: CV. Bangun Bumitama.

Hastuty, A. (2024). E–Booking Bus Pariwisata Berbasis Web. *Jurnal Sintaks Logika*, *4*(3), 64-75.

Hastuty, A. (2025). Hacking the Future of College Administration: Process Transformation through AR/VR Immersive Technology. *EDUKASIA Jurnal Pendidikan dan Pembelajaran*, *6*(1), 95-106.

Herwanto, R. (2020). *CLOUD COMPUTING - Manajemen dan Perencanaan Kapasitas*. Yogyakarta : Andi Offset.

Jumaroh, R. (2020). *Pembangunan Aplikasi Pelayanan Administrasi Kependudukan Menggunakan Framework Laravel (Studi Kasus : Desa Rajaiyang).* Skripsi. Bandung : Universitas Pasundan Bandung.

N, Sora. (2015). Pengertian *UML* dan Jenis - jenisnya. *From http://www*. pengertianku.net/2015/09/pengertian-*uml*-dan-jenis-jenisnya-serta-contoh-diagramnya.*html.* (11 Nov 2021).

Purnomo, R, F. (2020). *Firebase Membangun Aplikasi Berbasis Android*. Yogyakarta : Andi OffsetP

Rahman, H; Wagino, (2021). *Aplikasi Pelayanan Administrasi Kependudukan Dan Penetapan Apbdes Berbasis Web Pada Desa Kampung Baru*. Skripsi. Banjarmasin : Universitas Islam Kalimantan.

Rusito. (2018). *Birokrasi Profesional Pemerintah Desa*. Serang: Untirta Press anggota APPTI.

Setiawan, E, B.(2018). *Membangun Aplikasi Android Web Dan Web Service*. Jakarta : Erlangga.