PENGEMBANGAN *WEBSITE* KAMUS BAHASA LAMPUNG BERBASIS KOMUNITAS MENGGUNAKAN *FRAMEWORK* QWIK

(Skripsi)

Oleh:

DEDE KURNIAWAN
NPM 2115061072

FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2025

PENGEMBANGAN *WEBSITE* KAMUS BAHASA LAMPUNG BERBASIS KOMUNITAS MENGGUNAKAN *FRAMEWORK* QWIK

Oleh

DEDE KURNIAWAN

Skripsi

Sebagai Salah Satu Syarat untuk Mencapai Gelar

SARJANA TEKNIK

Pada

Program Studi Teknik Informatika

Fakultas Teknik Universitas Lampung

FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
BANDAR LAMPUNG
2025

ABSTRAK

PENGEMBANGAN *WEBSITE* KAMUS BAHASA LAMPUNG BERBASIS KOMUNITAS MENGGUNAKAN *FRAMEWORK* QWIK

Oleh

DEDE KURNIAWAN

Bahasa Lampung menghadapi ancaman kepunahan yang disebabkan oleh penurunan minat generasi muda dan kurangnya dokumentasi bahasa. Oleh karena itu, diperlukan upaya pelestarian melalui media digital yang dapat diakses dengan mudah oleh masyarakat. Penelitian ini bertujuan untuk mengembangkan situs web kamus Bahasa Lampung berbasis komunitas dengan menggunakan teknologi modern seperti Qwik. Situs web ini diharapkan dapat menjadi solusi pembelajaran bahasa oleh masyarakat luas pada umumnya dalam memahami Bahasa Lampung. Situs web ini dikembangkan dengan menggunakan *framework* Qwik dan metode pengembangan perangkat lunak Scrum. Pengujian situs web ini dilakukan dengan menggunakan Google Lighthouse untuk menguji performa, serta *blackbox testing* dan *User Acceptance Test* (UAT) untuk menguji fungsionalitas. Hasil penelitian menunjukkan bahwa situs web berhasil dikembangkan selama 9 sprint dengan total 26 *backlog* yang terbagi ke dalam 5 *epic* dan tiga jenis pengujian. Pengujian performa dengan Google Lighthouse mendapatkan skor 99,87% untuk perangkat *desktop* dan 89,66% untuk perangkat *mobile*. Pengujian *blackbox* testing terdiri dari 93 skenario yang dilakukan sebanyak dua kali dengan tingkat keberhasilan 100% pada pengujian kedua. Pengujian dengan metode *User Acceptance Test* mendapatkan skor rata-rata 95,02%.

Kata kunci: Bahasa Lampung, Qwik, Scrum, Google Lighthouse, UAT

ABSTRACT

DEVELOPMENT OF A COMMUNITY-BASED LAMPUNG LANGUAGE DICTIONARY WEBSITE USING QWIK FRAMEWORK

By

DEDE KURNIAWAN

The Lampung language is facing the threat of extinction due to declining interest among younger generations and the lack of proper linguistic documentation. Therefore, preservation efforts through easily accessible digital media are necessary. This study aims to develop a community-based Lampung language dictionary website utilizing modern technologies such as the Qwik framework. The website is expected to serve as a language learning solution for the general public to better understand the Lampung language. The website was developed using the Qwik framework and the Scrum software development methodology. Testing was conducted through three methods: Google Lighthouse for performance evaluation, black-box testing, and User Acceptance Testing (UAT) for functionality assessment. The development process was completed over the course of 9 sprints, consisting of 26 backlog items grouped into 5 epics. Performance testing using Google Lighthouse yielded a score of 99.87% on desktop and 89.66% on mobile devices. The black-box testing involved 93 scenarios executed twice, achieving a 100% success rate in the second round. User Acceptance Testing recorded an average user satisfaction score of 95.02%.

Keywords: Lampung Language, Qwik, Scrum, Google Lighthouse, UAT

Judul Skripsi	:	**PENGEMBANGAN *WEBSITE* KAMUS BAHASA LAMPUNG BERBASIS KOMUNITAS MENGGUNAKAN *FRAMEWORK* QWIK**
Nama Mahasiswa	:	Dede Kurniawan
Nomor Pokok Mahasiswa	:	2115061072
Program Studi	:	Teknik Informatika
Jurusan	:	Teknik Elektro
Fakultas	:	Teknik

MENYETUJUI

1. Komisi Pembimbing

Mahendra Pratama, S.T., M.Eng.
NIP. 199112152019031013

Wahyu Eko Sulistiono, S.T., M.Sc.
NIP.197412012001121001

2. Mengetahui

Ketua Jurusan
Teknik Elektro

Herlinawati, S.T., M.T.
NIP. 197103141999032001

Ketua Program Studi
Teknik Informatika

Yessi Mulyani, S.T., M.T.
NIP. 197312262000122001

MENGESAHKAN

1. Tim Penguji

Ketua : **Mahendra Pratama S.T., M.Eng.** .....

Sekretaris : **Wahyu Eko Sulistiono, S.T., M.Sc.** .....

Penguji : **Ir. Gigih Forda Nama, S.T., M.T.I., IPM.** .....

2. Dekan Fakultas Teknik

Dr. Eng. Ir. Helmy Fitriawan, S.T., M.Sc.
NIP. 197509282001121002

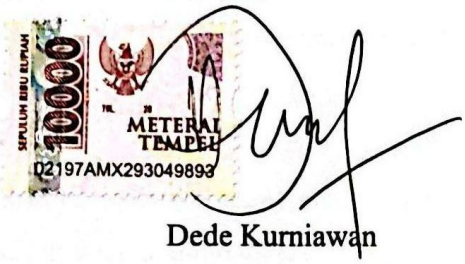
Tanggal Lulus Ujian Skripsi: **19 Mei 2025**

SURAT PERNYATAAN

Saya yang bertanda tangan di bawah ini, menyatakan bahwa skripsi saya yang berjudul "Pengembangan *Website* Kamus Bahasa Lampung Berbasis Komunitas Menggunakan *Framework* Qwik" merupakan hasil kerja saya sendiri. Apabila di kemudian hari terbukti bahwa pernyataan saya tidak benar, maka saya bersedia dikenai sanksi sesuai hukum yang berlaku.

Bandar Lampung, 23 Mei 2025

Penulis,



Dede Kurniawan

NPM. 2115061072

RIWAYAT HIDUP

Penulis lahir di Way Kerap pada tanggal 9 Mei 2003 dan merupakan anak bungsu dari tiga bersaudara pasangan Bapak Zailan Idris dan Ibu Dasiah. Penulis menyelesaikan pendidikan dasarnya di SD Negeri 1 Way Kerap (2015), serta pendidikan menengahnya di SMP Negeri 1 Semaka (2018), dan SMK Negeri 1 Talang Padang (2021). Pada tahun 2021, penulis menjadi mahasiswa Program Studi Teknik Informatika, Jurusan Teknik Elektro, Fakultas Teknik Universitas Lampung yang masuk melalui jalur SBMPTN (Seleksi Bersama Masuk Perguruan Tinggi Negeri). Selama menjadi mahasiswa, penulis aktif dalam berbagai kegiatan, di antaranya:

1. Mengikuti Magang Kamus Merdeka dari Kementerian Pendidikan dan Kebudayaan di PT. Telekomunikasi Indonesia yang berlokasi di Kota Bandung pada tahun 2024.
2. Mengikuti Studi Independen Bersertifikat dari Kementerian Pendidikan dan Kebudayaan di Binar Academy pada tahun 2023.
3. Menjadi asisten Laboratorium Teknik Komputer Jurusan Teknik Elektro Universitas Lampung periode 2023.
4. Menjadi pengurus dan anggota aktif dari Unit Kegiatan Mahasiswa Universitas English Society Unila tahun 2021-2023.
5. Menjadi anggota Paguyuban Karya Salemba Empat Universitas Lampung periode 2023.

MOTTO

"If you know all the languages of the world and not your mother tongue, that is enslavement. But if you know your mother tongue and add all the other languages of the world, that is empowerment."

(Ngũgĩ wa Thiong'o)

"BERGELAP-GELAPLAH DALAM TERANG, BERTERANG-TERANGLAH DALAM GELAP!"

(Tan Malaka)

"The true sign of intelligence is not knowledge but imagination."

(Albert Einstein)

PERSEMBAHAN

Puji syukur saya panjatkan kepada Allah SWT, Tuhan Yang Maha Kuasa, atas segala rahmat dan karunia-Nya yang telah dilimpahkan kepada saya, sehingga saya sebagai penulis dapat menyelesaikan skripsi ini.

Karya ini aku persembahkan kepada:

Kedua Orang Tuaku Tercinta

"Yang dengan kasih sayang, doa, dan perjuangannya telah menuntunku sejauh ini. Semoga skripsi ini menjadi secuil bukti bakti dan syukurku atas setiap pengorbanan kalian."

Bahasa Lampung

"Warisan leluhur yang mulai dilupakan, namun tak pernah kehilangan makna. Semoga karya ini menjadi bagian kecil dari nyala yang menjaga eksistensimu."

Diri Saya di Masa Lalu dan di Masa Depan

"Yang tak pernah menyerah dan tidak akan pernah menyerah walau dunia terasa berat."

SANWACANA

Penulis panjatkan puji dan syukur setinggi-tingginya kehadirat Allah SWT, Tuhan Yang Maha Esa, karena berkat dan hidayah-Nya, penulis dapat menyelesaikan skripsi yang berjudul "Pengembangan *Website* Kamus Bahasa Lampung Berbasis Komunitas Menggunakan *Framework* Qwik" sebagai salah satu syarat dalam memperoleh gelar sarjana pada Program Studi Teknik Informatika, Universitas Lampung. Dalam menyelesaikan penelitian ini, penulis mendapatkan banyak bantuan dari berbagai pihak. Maka dari itu, dengan penuh rasa hormat penulis ingin mengucapkan terima kasih yang sebesar-besarnya kepada:

1. Kedua orang tua dan keluarga yang selalu tanpa lelah memberikan dukungan dan doa di setiap hal.
2. Bapak Dr. Eng. Helmy Fitriawan, S.T., M.Sc., selaku Dekan Fakultas Teknik Universitas Lampung.
3. Ibu Herlinawati, S.T., M.T., selaku Ketua Jurusan Teknik Elektro Universitas Lampung.
4. Ibu Yessi Mulyani, S.T., M.T. selaku Ketua Program Studi Teknik Informatika Universitas Lampung.
5. Bapak Mahendra Pratama, S.T., M. Eng. selaku Pembimbing Utama yang telah berkenan memberikan waktu serta pikiran dalam membimbing penulis selama mengerjakan penelitian dan penulisan skripsi.
6. Bapak Wahyu Eko Sulistiono, S.T., M.Sc. selaku Pembimbing Pendamping yang telah berkenan membimbing dan dengan sabar selalu memberikan masukan dan saran bagi penulis dalam menyelesaikan penelitian dan penulisan skripsi.

7. Bapak Ir. Gigih Forda Nama, S.T., M.T.I., IPM. selaku Pembimbing Akademik dan Penguji yang telah memberikan kritik dan saran yang membangun untuk penulis agar dapat menyelesaikan skripsi ini dengan lebih baik lagi.
8. Seluruh Dosen Program Studi Teknik Informatika Universitas Lampung yang telah banyak memberikan ilmu kepada penulis.
9. Kepada Ridho Ahmad Fauzi, Nyoman Eka Swardita dan Andre Gilang Firmansyah yang telah mau saling bekerja sama dan memberikan dukungan sebagai tim peneliti dalam proses pengembangan proyek.
10. Kepada Bapak Dr. Munaris, M.Pd dan Ibu Jesika Wulandari, S.Pd yang telah bersedia meluangkan waktunya untuk berpartisipasi dalam pengujian dan menjadi peninjau dalam proyek yang telah dilaksanakan.
11. Seluruh pihak yang terlibat dalam penelitian ini, yang tidak dapat disebutkan namanya satu per satu.

Penulis menyadari bahwa skripsi ini masih jauh dari kata sempurna dan masih banyak kekurangan, sehingga penulis sangat terbuka untuk saran dan masukan yang membangun dari siapa saja. Penulis memohon maaf atas seluruh kekurangan yang ada dan mengucapkan terima kasih atas seluruh bantuan yang telah diberikan. Penulis berharap skripsi ini dapat bermanfaat bagi peneliti lainnya atau bagi siapa pun yang membacanya.

Bandar Lampung, 23 Mei 2025
Penulis,

Dede Kurniawan

DAFTAR ISI

Halaman

DAFTAR GAMBAR

DAFTAR TABEL

I. PENDAHULUAN

1.1 Latar Belakang

Indonesia adalah negara kepulauan terbesar di dunia, dengan lebih dari 700 bahasa, Indonesia berada pada peringkat kedua setelah Papua Nugini sebagai negara yang memiliki bahasa terbanyak di dunia. Hal ini membuktikan bahwa Indonesia memiliki ragam wicara yang sangat masif. Namun, keragaman bahasa ini terancam dalam ujung tombak kepunahan. Terdapat tiga faktor yang menyokong kepunahan bahasa, di antaranya yaitu faktor ekonomi di mana banyak penduduk suatu daerah yang pergi ke luar daerahnya untuk membangun kehidupan yang lebih baik, faktor kurangnya pelestarian bahasa dalam kurikulum pendidikan yang menjadikan bahasa daerah hanya sebagai muatan lokal atau mata pelajaran pilihan, serta faktor komunikasi keluarga yang tidak mengenalkan lagi penggunaan bahasa daerah pada keturunannya [1].

Suku Lampung adalah salah satu dari sekian banyak suku yang mendiami ujung bagian selatan pulau Sumatra, suku Lampung memiliki kebudayaan yang beragam termasuk bahasa yang dituturkannya. Sebagai bagian dari kekayaan budaya Indonesia, bahasa Lampung merupakan salah satu dari ratusan bahasa daerah yang menjadi fondasi penting dalam menjaga identitas kebudayaan lokal. Namun, seiring dengan kemajuan zaman di era yang kian terhubung ini, arus modernisasi terus menyeret kelestarian Bahasa Lampung ke dalam jurang kepunahan. Tantangan yang signifikan seperti berkurangnya penutur asli dan kurangnya upaya dokumentasi budaya yang beradaptasi dengan kemajuan zaman.

Menurut data dari Badan Pengawasan Keuangan dan Pembangunan Provinsi Lampung (2023) komposisi demografi Provinsi Lampung terdiri dari suku Jawa sebesar 62% dari populasi, sedangkan etnis Lampung hanya membentuk 25% dari total populasi dan sisanya berasal dari suku lain seperti Sunda, Melayu, dan

Bali. Data tersebut mengindikasikan salah satu faktor terbatasnya penggunaan bahasa dan aksara Lampung untuk komunikasi sehari-hari oleh penutur asli yang relatif sedikit.

Bahasa Lampung memiliki dua dialek utama yakni dialek A dan dialek O yang masing-masing memiliki ciri khas dan variasinya tersendiri [2]. Sayangnya pengetahuan serta pendokumentasian dari variasi dialek ini belum secara masif dan belum dapat mengikuti perkembangan zaman saat ini menyebabkan hilangnya banyak aspek yang penting dari kebudayaan Lampung itu sendiri oleh generasi mendatang. Ditambah dengan kurangnya semangat dan minat generasi muda untuk terus mempelajari dan melestarikan Bahasa Lampung, bahasa ini berada di atas ambang kepunahan.

Abad ke-21 ini ditandai dengan adanya era *society* 5.0 yang merupakan keberlanjutan dari era *industry* 4.0 yang berfokus pada perkembangan teknologi industri untuk memudahkan kegiatan manusia. Era *society* 5.0 ditandai dengan pemikiran bahwa manusia menjadi pusat dari perkembangan teknologi, yang pada intinya teknologi dikembangkan untuk memudahkan kehidupan manusia [3]. Menyadari hal ini, pemanfaatan teknologi dalam upaya pelestarian bahasa menjadi solusi yang baik agar bahasa-bahasa daerah di Indonesia tidak mengalami kepunahan total.

Kamus Bahasa Lampung berbasis komunitas memanfaatkan teknologi yang ada saat ini sebagai upaya untuk melestarikan Bahasa Lampung terutama variasi dan dialek-dialeknya. Hadirnya situs web kamus Bahasa Lampung ini memberikan wadah bagi masyarakat luas untuk berkontribusi dalam pendokumentasian Bahasa Lampung serta menjadi sarana bagi masyarakat yang hendak mempelajari Bahasa Lampung. Dengan memanfaatkan kemajuan teknologi, situs web ini diharapkan dapat diakses oleh semua orang, kapan pun dan dari mana pun, tanpa terhalang oleh batasan geografis maupun waktu.

Ada beberapa kamus Bahasa Lampung berbasis digital lainnya yang dapat diakses oleh siapa saja, namun hampir semuanya memiliki korpus yang sangat terbatas. Keterbatasan ini mengakibatkan kesulitan bagi pengguna yang ingin mencari kata spesifik dari suatu daerah, terutama jika berhadapan dengan variasi

dialek yang berbeda dalam Bahasa Lampung. Korpus yang terbatas ini menghambat upaya dokumentasi serta pelestarian Bahasa Lampung secara menyeluruh. Banyak kosa kata lokal yang berpotensi hilang karena tidak terdokumentasi secara menyeluruh. Selain itu, kamus-kamus ini tidak memberi kesempatan bagi masyarakat untuk berkontribusi ke dalam daftar kosa katanya.

Untuk mengatasi tantangan ini, diperlukan sebuah kamus Bahasa Lampung yang tidak hanya memiliki cakupan korpus yang luas, tetapi juga mampu melibatkan komunitas penutur asli dalam proses pengumpulan data serta dibangun dengan teknologi terbaru seperti Qwik dan Typescript sehingga dapat berjalan dengan lancar dan memiliki performa yang mumpuni. Pendekatan berbasis komunitas dapat meningkatkan keakuratan dan kedalaman informasi yang tersedia, sekaligus mendorong partisipasi aktif dari masyarakat lokal dalam pelestarian bahasa mereka. Dengan demikian, pengguna dari berbagai latar belakang dan kepentingan dapat mengakses informasi yang lebih lengkap dan akurat tentang Bahasa Lampung. Pengembangan kamus berbasis komunitas ini bertujuan untuk menyediakan platform di mana penurut asli dan pengguna pada umumnya dapat berkontribusi dalam menambah, memperbarui, dan memverifikasi entri kamus. Hal ini diharapkan tidak hanya dapat memperluas cakupan kamus, tetapi juga dapat mendukung upaya pelestarian dan pendokumentasian Bahasa Lampung sebagai warisan budaya Indonesia.

1.2 Rumusan masalah

Adapun rumusan masalah dari penelitian ini adalah sebagai berikut:

1. Bagaimana cara mengembangkan situs web kamus Bahasa Lampung berbasis komunitas dengan menggunakan *framework* Qwik*?*
2. Apakah situs web kamus Bahasa Lampung yang dikembangkan telah memenuhi kebutuhan dan dapat digunakan dengan baik oleh pengguna?
3. Bagaimana dampak performa yang dihasilkan oleh *framework* Qwik terhadap situs web kamus Bahasa Lampung ini?

1.3 Tujuan Penelitian

Adapun tujuan dari penelitian ini adalah sebagai berikut:

1. Mengembangkan situs web kamus Bahasa Lampung yang dapat memfasilitasi pengguna untuk menerjemahkan kata dari Bahasa Lampung ke Bahasa Indonesia dan sebaliknya, serta berpartisipasi dalam pendokumentasian Bahasa Lampung.
2. Menguji fungsionalitas situs web kamus Bahasa Lampung agar sesuai dengan kebutuhan pengguna.
3. Menguji performa *framework* Qwik dalam menjalankan situs web kamus Bahasa Lampung.

1.4 Manfaat Penelitian

Adapun manfaat yang diharapkan dari penelitian yang dilakukan adalah sebagai berikut:

1. Penelitian ini mendukung pelestarian dan pendokumentasian Bahasa Lampung.
2. Penelitian ini dapat dijadikan sebagai referensi bagi penelitian lain ke depannya yang relevan seperti penerapannya pada bidang kecerdasan buatan.
3. Penelitian ini mendukung dalam upaya peningkatan pemahaman masyarakat dalam menggunakan Bahasa Lampung.
4. Penelitian ini dapat menjadi sumber referensi pembelajaran bagi masyarakat luas.

1.5 Batasan Masalah

Adapun batasan masalah dari penelitian ini adalah sebagai berikut:

1. Penelitian ini berfokus pada pengembangan pada sisi *frontend website* sehingga pembahasan pada sisi *backend, UI/UX*, dan *devOps* bersifat konseptual.
2. Penelitian ini menggunakan *framework* Qwik sebagai *framework frontend* sehingga tidak membahas perbandingannya dengan *framework* serupa.
3. Objek penelitian pengembangan situs web ini ditujukan kepada penutur asli Bahasa Lampung serta masyarakat umum yang hendak mempelajari Bahasa Lampung.

II. TINJAUAN PUSTAKA

2.1 Dasar Teori

2.1.1 Bahasa Lampung

Bahasa Lampung merupakan bahasa yang digunakan oleh masyarakat suku Lampung yang tersebar di Provinsi Lampung, wilayah selatan Sumatra Selatan, Bengkulu, Banten, serta oleh komunitas diaspora Lampung di berbagai daerah di Indonesia. Bahasa Lampung merupakan anggota rumpun bahasa Austronesia yang berasal dari cabang Melayu-Polinesia [4]. Terdapat dua dialek utama dalam Bahasa Lampung, yakni dialek A (Api) dan dialek O (Nyo) [2]. Klasifikasi pasti Bahasa Lampung dapat dikatakan ambigu karena interaksi linguistik yang ekstensif terhadap budaya Melayu yang mengaburkan perbedaan keduanya. Secara historis, Bahasa Lampung dikelompokkan dalam satu keluarga “Malayic Hesion” dengan Bahasa Melayu, Minangkabau, dan Kerinci oleh Isidore Dyen pada tahun 1965.

Namun, studi linguistik berikutnya mengevaluasi kembali hasilnya. Nothofer memisahkan Lampung dari kelompok "Melayu", menempatkannya dalam "Javo-Sumatra Hesion," yang mencakup Bahasa Malayo-Sumbawan dan lainnya seperti Sunda dan Madura. Ross kemudian mengelompokkan Bahasa Lampung secara terpisah, mengindikasikan tidak terkaitannya terhadap Bahasa Melayu-Polinesia lainnya [5].

2.1.2 Aksara Lampung

Had Lampung, yang juga dikenal sebagai Aksara Lampung, merupakan sistem tulisan yang digunakan untuk merekam bahasa-bahasa dalam rumpun Lampung dan berasal dari perkembangan aksara Kawi. Aksara ini telah digunakan sebagai alat komunikasi tertulis sejak abat ke-17 hingga akhir abad ke-20 sebelum penggunaannya mulai tergantikan oleh aksara Latin yang diperkenalkan oleh

bangsa Eropa. Aksara Lampung terdiri dari 20 induk huruf *(kelabay sughat),* 11 diakritik atau anak huruf *(benah sughat),* dan 5 tanda baca. Aksara ini dibakukan pada tahun 1985 dari musyawarah pemuka adat Lampung kala itu yang menghasilkan satu induk huruf baru yaitu "gha/kha/gra" sebagai huruf ke-20 dan 4 tanda baca baru yaitu koma, titik, tanda tanya, dan tanda seru.

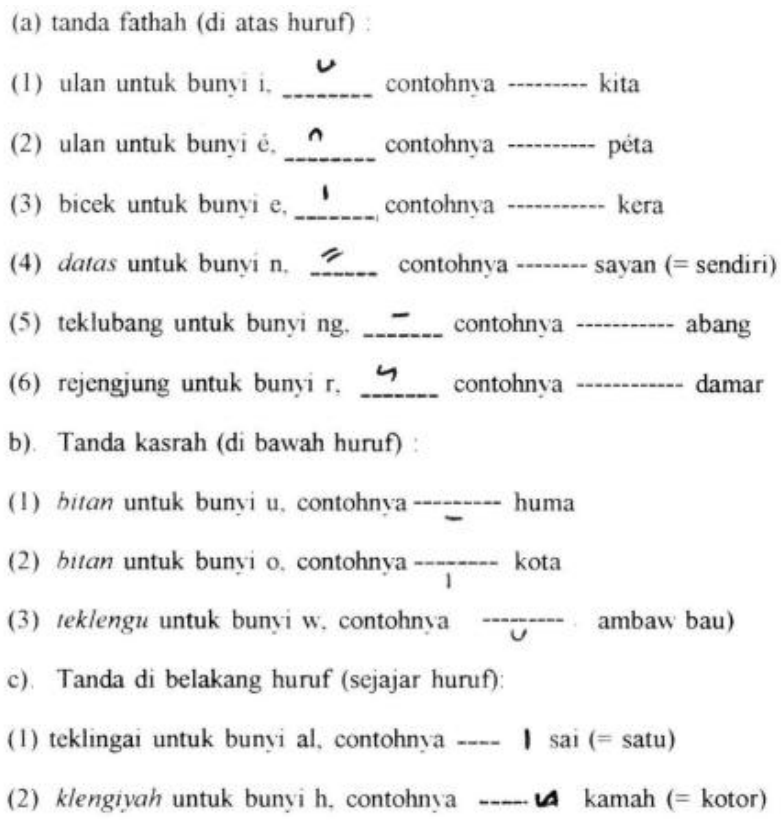
1. Induk huruf *(kelabay sughat)*

Induk huruf aksara Lampung terdiri dari 20 karakter dan sering disebut sebagai aksara Kaganga sesuai dengan tiga huruf pertama. Setiap induk huruf terdiri dari konsonan dan vokal 'a' yang kemudian vokal ini dapat dimodifikasi menggunakan diakritik.

Gambar 2.1 Induk huruf aksara Lampung *(kelabai surat)*

2. Diakritik *(benah surat)*

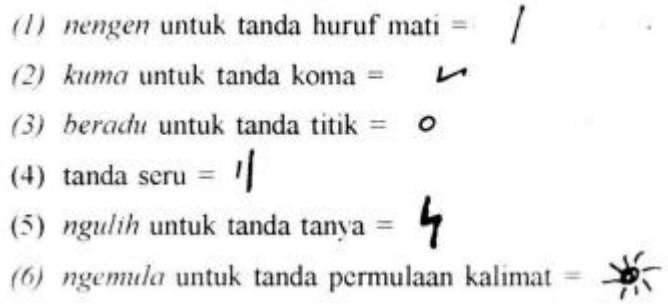
Diakritik berfungsi untuk memodifikasi bunyi yang dihasilkan oleh induk huruf, terdiri dari 11 karakter. Diakritik dapat diletakkan di atas, bawah, atau depan setiap induk huruf.



Gambar 2.2 Diakritik dalam aksara Lampung [6]

3. Tanda baca

Tanda baca berfungsi untuk membantu pembaca untuk memahami tulisan dengan benar seperti tanda *virama* (mati), koma, titik, seru, tanya, dan tanya mula kalimat.

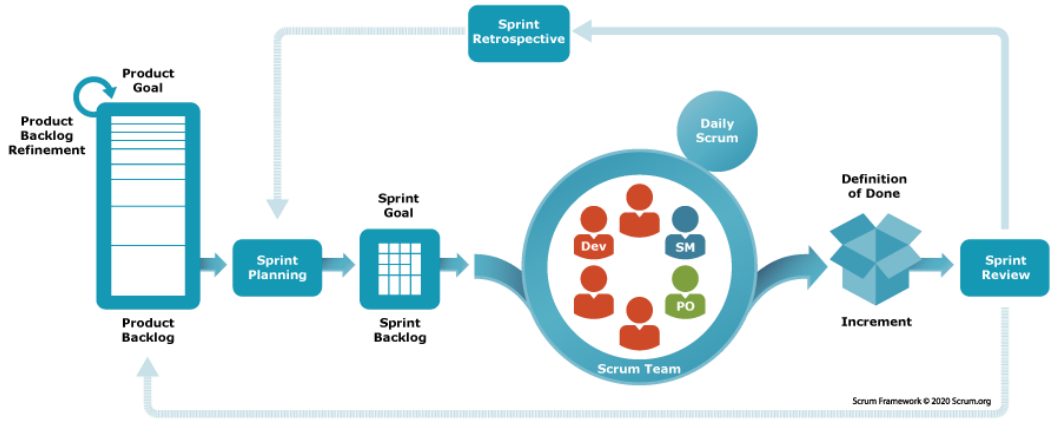


Gambar 2.3 Tanda baca dalam aksara Lampung [6]

2.1.3 Agile Development

Agile development adalah metode pengembangan perangkat lunak yang memanfaatkan fleksibilitas, kolaborasi yang kuat, perubahan yang cepat, menjadikannya metodologi yang tepat untuk proyek yang dinamis [7]. Fleksibilitas ini tercermin dari kemampuannya untuk menyesuaikan rencana proyek pada tahap mana pun dalam siklus pengembangan. Dalam metode ini, kolaborasi erat antara klien dan pengembang sangat penting agar pengembang dapat bekerja secara mandiri. *Agile development* bersifat iteratif dan berkelanjutan, sehingga produk yang dihasilkan tidak hanya berfungsi dengan baik tetapi juga memenuhi harapan klien dan pengguna [8].

2.1.4 Scrum



Gambar 2.4 Proses *Scrum* [9]

Scrum adalah salah satu kerangka kerja dari *agile development* yang banyak dipakai dalam pengembangan perangkat lunak, berfokus pada proses iteratif, kolaborasi tim, dan kemampuan beradaptasi. Tujuan utamanya adalah mempercepat pengembangan serta mengurangi risiko dalam kondisi yang tidak pasti dengan menyediakan struktur kerja yang teratur namun tetap fleksibel [10].

Struktur fundamental utama dari *Scrum* adalah tim yang terdiri dari satu *scrum master,* satu *product owner*, dan pengembang. Dalam tim *Scrum*, tidak ada sub-tim atau hierarki. Tim *Scrum* berfokus pada satu objektif untuk satu waktu. Tim

Scrum bersifat lintas fungsi artinya anggota tim memiliki kemampuan yang dibutuhkan untuk setiap *sprint*. *Product owner* yang bertanggung jawab mengelola *product backlog*, tim pengembang yang bekerja secara kolaboratif untuk mencapai tujuan *sprint*, serta *Scrum Master* yang memastikan proses berjalan sesuai metodologi.

Sprint adalah jantung dari *Scrum*, setiap *sprint* memiliki durasi maksimal satu bulan dan konsisten. *Sprint* baru dimulai setelah *sprint* sebelumnya selesai. Seluruh tugas yang dibutuhkan untuk mencapai gol, meliputi *sprint planning, daily scrum, sprint review*, dan *retrospective* termasuk dalam proses setiap *sprint*. Selama *sprint* berlangsung, tidak boleh ada perubahan signifikan yang dapat membahayakan gol dari proyek; *Product backlog* dapat diperbaiki seperlunya; dan cakupan dapat diperjelas dan dinegosiasikan ulang dengan *product owner.*

a. Scrum Artifacts

Scrum artifacts adalah elemen-elemen dalam *Scrum* yang merepresentasikan pekerjaan atau nilai dan dirancang untuk memaksimalkan transparansi informasi penting. Dengan transparansi ini, semua pihak yang terlibat memiliki pemahaman yang sama untuk melakukan adaptasi jika diperlukan. Setiap *artifact* memiliki komitmen yang memastikan informasi yang diberikan meningkatkan transparansi dan fokus, serta memungkinkan pengukuran kemajuan. Berikut adalah komitmen dari setiap *artifact*:

1) Product Goal

Product goal adalah komitmen untuk *product backlog* yang menggambarkan kondisi masa depan produk sebagai target yang akan dicapai oleh tim. Ia menjadi tujuan jangka panjang yang berada dalam *product backlog*, sementara elemen-elemen lainnya di dalam *product backlog* muncul untuk mendukung pencapaian *product goal*.

2) Sprint Goal

Sprint goal adalah tujuan tunggal yang ingin dicapai dalam satu *sprint*. Tujuan ini memberikan fleksibilitas bagi *Developer* dalam menentukan pekerjaan yang dibutuhkan untuk mencapainya. *Sprint goal* juga mendorong kerja sama dan fokus tim. Komitmen ini dibuat selama *sprint*

planning dan ditambahkan ke dalam *sprint backlog*. Jika ada perubahan selama *sprint*, pengembang dan *product owner* dapat menyesuaikan ruang lingkup pekerjaan tanpa mengubah *sprint goal*.

3) *Definition of Done*

Definition of Done (DoD) adalah deskripsi formal tentang standar kualitas yang harus dipenuhi untuk menyatakan bahwa sebuah *increment* telah selesai. Ketika sebuah item *product backlog* memenuhi *Definition of Done*, maka dapat disebut sebagai sebuah *increment*.

b. *Product Backlog*

Product backlog adalah daftar terurut yang terus berkembang untuk meningkatkan produk dan menjadi sumber utama pekerjaan tim. Item di dalamnya dianggap siap untuk dipilih saat *sprint planning* setelah melalui proses penyempurnaan atau *refinement*. Proses ini bertujuan untuk memecah item menjadi lebih kecil dan menambah detail seperti deskripsi, urutan, dan ukuran kompleksitas. Penentuan ukuran kompleksitas dilakukan oleh pengembang, dengan *product owner* membantu memberi pemahaman dan menentukan prioritas.

c. *Sprint Planning*

Sprint planning mengawali setiap *sprint* dengan menyusun pekerjaan atau tugas untuk dikerjakan selama *sprint*. Perencanaan ini disusun oleh seluruh anggota tim. Pembahasan meliputi apa saja yang dapat diselesaikan untuk *sprint* ini, dan bagaimana tugas tersebut dapat dikerjakan. Tugas yang hendak dikerjakan selama *sprint* diambil dari *product backlog* untuk kemudian dimasukkan ke dalam *sprint backlog.*

d. *Sprint Backlog*

Sprint backlog adalah rencana kerja yang dibuat oleh dan untuk para pengembang selama satu *sprint*. *Sprint backlog* memberikan gambaran yang sangat transparan dan selalu diperbarui secara waktu nyata oleh para pengembang. *Sprint backlog* terus diperbarui sesuai dengan pembelajaran atau penyesuaian baru yang muncul selama proses. Selain itu, *sprint backlog* harus memiliki detail yang cukup agar para pengembang dapat memeriksa kemajuan mereka secara efektif dalam *daily scrum*.

e. *Daily Scrum*

Tujuan dari *daily scrum* adalah untuk memeriksa proses dari setiap anggota tim untuk mencapai tujuan *sprint* dan mengadaptasikan *backlog* sesuai kebutuhan. *Daily scrum* dapat dilakukan secara sinkron atau asinkron sesuai dengan kebutuhan masing-masing tim.

f. *Increment*

Increment adalah langkah konkret menuju *product goal* yang bertambah seiring dengan setiap *increment* sebelumnya. Setiap *increment* harus diverifikasi dengan teliti dan dapat digunakan untuk memberikan nilai. *Increment* hanya dapat dianggap selesai jika memenuhi *Definition of Done*. Beberapa *increment* bisa dibuat dalam satu *sprint*, dan jumlah total *increment* tersebut dipresentasikan saat *sprint review*.

g. *Sprint Review*

Tujuan utama dari *sprint review* adalah untuk meninjau hasil dari *sprint* untuk menentukan adaptasi *sprint* selanjutnya. Tim *Scrum* mempresentasikan hasil yang telah dicapai kepada *stakeholders* dan proses pencapaian gol produk didiskusikan.

h. *Retrospective*

Retrospective bertujuan untuk merencanakan strategi untuk meningkatkan kualitas dan efektivitas *Scrum*. Tim *scrum* meninjau kembali jalannya *sprint* sebelumnya seperti individu, interaksi, proses, dan alat. Proses ini mendiskusikan hal apa saja yang berjalan baik dan dapat dilanjutkan serta hal apa saja yang tidak berjalan baik agar dapat ditingkatkan pada *sprint* selanjutnya [9].

2.1.5 Blackbox Testing

Blackbox testing adalah metode pengujian perangkat lunak yang berfokus pada evaluasi fungsionalitas dari sudut pandang pengguna akhir tanpa perlu memahami cara kerja internal sistem. Pengujian ini didasarkan pada spesifikasi perangkat lunak yang telah dibuat, di mana penguji memeriksa apakah perangkat lunak bekerja sesuai dengan kebutuhan pengguna yang telah ditentukan. Tujuan

utama dari *blackbox* testing adalah untuk memastikan bahwa perangkat lunak memberikan keluaran yang benar berdasarkan masukan tertentu tanpa memeriksa kode sumber atau algoritma yang digunakan di dalam sistem.

Selama pengujian, penguji hanya berfokus pada hubungan antara masukan *(input)* dan keluaran *(output)*. Penguji memberikan data yang valid untuk memverifikasi bahwa perangkat lunak berfungsi sebagaimana mestinya dalam kondisi normal, serta data yang tidak valid untuk menguji bagaimana sistem menangani masukan yang salah atau eror. Pengujian ini bertujuan untuk memastikan bahwa perangkat lunak mampu menangani berbagai skenario yang mungkin terjadi dalam penggunaan nyata. Pengujian ini sangat penting dalam memastikan perangkat lunak memenuhi kebutuhan dan ekspektasi pengguna secara fungsional. [11].

2.1.6 User Acceptance Test (UAT)

User Acceptance Testing (UAT) adalah proses pengujian yang dilakukan oleh pengguna akhir atau klien untuk mengevaluasi apakah sebuah sistem memenuhi kebutuhan mereka dan sesuai dengan persyaratan yang telah ditentukan. UAT biasanya dilakukan pada akhir fase pengembangan untuk memastikan bahwa aplikasi atau sistem siap digunakan dalam lingkungan sebenarnya. Pengujian ini fokus pada evaluasi fungsionalitas dan kegunaan sistem, memastikan bahwa semua fitur berfungsi seperti yang diharapkan.

Tujuan utama UAT adalah untuk memvalidasi bahwa solusi yang dikembangkan sesuai dengan proses bisnis yang dimaksudkan dan memenuhi harapan pengguna. Dalam UAT, pengguna berpartisipasi langsung dengan menguji skenario nyata untuk menentukan apakah sistem tersebut dapat diterima atau tidak. Proses ini penting untuk mengidentifikasi masalah atau kekurangan yang mungkin terlewat selama fase pengembangan atau pengujian internal.

UAT juga memainkan peran penting dalam proses *quality assurance (QA)* karena tingkat keberhasilan sebuah aplikasi dalam memenuhi kebutuhan pengguna sangat bergantung pada hasil UAT. Implementasi UAT yang berhasil menandai kesiapan sistem untuk digunakan di lingkungan produksi [12].

2.1.7 Unified Modelling Language (UML)

Unified Modelling Language (UML) adalah metode perancangan teknologi berbasis objek yang memungkinkan perancangan model perangkat lunak dalam bentuk grafis sehingga lebih mudah divisualisasikan. Fungsi utama dari UML adalah menggambarkan perilaku sistem yang mencakup *Use Case Diagram* dan *Activity Diagram. Use Case Diagram* digunakan untuk menggambarkan interaksi antara aktor, sedangkan diagram aktivitas menggambarkan proses bisnis atau menu dalam perangkat lunak. UML memiliki beberapa karakteristik utama di antaranya:

1. UML tidak mencakup konsep proses, melainkan hanya metode pemodelan.
2. UML memberikan kebebasan untuk memilih teknik pemodelan yang sesuai dengan proyek.
3. Meskipun tidak memiliki konsep proses, aplikasi yang menggunakan UML tetap harus memiliki arsitektur sebagai pusat pengembangannya [13].

2.2 Penelitian Terdahulu

Dalam rangka mendukung penelitian ini, dilakukan studi literatur terhadap berbagai sumber yang relevan, mencakup jurnal ilmiah, buku, dan dokumen akademik lainnya. Studi literatur ini bertujuan untuk memetakan lanskap penelitian yang ada dan untuk menghindari duplikasi yang tidak perlu. Pada bagian ini, diuraikan beberapa penelitian terdahulu yang memberikan kontribusi signifikan terhadap topik yang sedang dikaji.

Penelitian yang dilakukan oleh Febri Eka Febriansyah, dkk (2020) yang berjudul "*Cawa Lampung: Kamus Bahasa Indonesia – Lampung Dialek A Berbasis Android"*. Penelitian ini menghasilkan produk berupa aplikasi Android yang dapat dijalankan secara luring yang dapat menerjemahkan bahasa Indonesia ke bahasa Lampung serta sebaliknya. Adapun metode pengembangan yang digunakan dalam penelitian ini adalah *eXtreme Programming (XP)*. Sistem yang dikembangkan memiliki fungsionalitas yang sangat baik. Hal ini dibuktikan dengan hasil pengujian *blackbox* testing yang menunjukkan skor 85.57% untuk pengujian non-fungsional yang diperoleh dari 47 responden. Sistem yang dikembangkan mampu untuk

menerjemahkan hingga 1500 kata namun memiliki waktu respons yang sangat buruk mencapai 9599 milidetik [14].

Penelitian selanjutnya dilakukan oleh Muhammad Fauzan Azima dan Siti Nur Laila (2020) yang berjudul "*Rancang Bangun Aplikasi Kamus Bahasa dan Aksara Lampung dialek A dan Dialek O Berbasis Android"*. Penelitian ini bertujuan untuk merancang dan membangun aplikasi kamus bahasa Lampung berbasis Android yang mendukung dialek A dan dialek O serta memiliki fitur pembelajaran aksara. Metode pengembangan yang digunakan yakni *agile development* sedangkan metode penelitian yang diterapkan adalah kualitatif dengan pendekatan observasi. Untuk pengujian menggunakan metode *blackbox testing*. Produk yang dihasilkan berupa aplikasi Android yang memiliki fitur kamus serta pembelajaran bahasa dan aksara Lampung [15].

Penelitian yang dilakukan oleh Trio Nurdianto, dkk (2021) yang berjudul *Rancang Bangun Aplikasi "Kmois" Kamus Bahasa Indonesia Moi Berbasis Android*. Membahas tentang pengembangan aplikasi kamus Bahasa Moi berbasis Android. Aplikasi ini ditujukan untuk melestarikan Bahasa Moi yang merupakan bahasa asli masyarakat kabupaten Sorong. Penelitian ini menerapkan metodologi *Research and Development (R&D)* yaitu langkah-langkah untuk mengembangkan produk baru atau mengembangkan produk yang telah ada agar sesuai dengan kebutuhan. Penelitian ini juga menggunakan metode *blackbox testing* untuk pengujiannya. Pengujian berdasarkan aspek tampilan memiliki persentase kesesuaian rerata sebesar 84% atau sangat baik, sedangkan pengujian berdasarkan aspek penggunaan memiliki persentase rerata sebesar 88% atau sangat baik [16].

Penelitian yang dilakukan oleh Adam Lipiński dan Beata Pańczyk (2023) yang berjudul *"Performance optimization of web applications using Qwik"*. Penelitian ini menganalisis kinerja dari tiga *framework*, yaitu React.js, Next.js, dan Qwik. Penelitian ini bertujuan untuk menunjukkan apakah Qwik memungkinkan waktu muat aplikasi yang lebih baik dibandingkan dengan *framework* lainnya. Penelitian dilakukan menggunakan tiga aplikasi dengan konten penelitian yang sama, merujuk pada kasus yang terjadi di lingkungan produksi. Pengujian kinerja dilakukan menggunakan Google Lighthouse dan diulang sebanyak 50 kali

percobaan, namun penelitian ini tidak menjelaskan versi Qwik yang digunakan pada saat pengujian. Metrik yang diukur meliputi FCP (*First Contentful Paint*), TBT (*Total Blocking Time*), LCP (*Largest Contentful Paint*), dan SI (*Speed Index*). Hasil setiap pengujian menunjukkan bahwa setiap *framework* memiliki keunggulan dan kelemahannya masing-masing. Qwik menunjukkan keunggulan pada pengujian pertama (me-*render* aplikasi dengan banyak elemen di DOM *tree*) dengan waktu FCP dan LCP tercepat. Next.js menunjukkan kinerja terbaik pada pengujian kedua (me-*render* aplikasi dengan banyak gambar berukuran besar), kemungkinan karena penggunaan komponen *image* yang mengoptimalkan tampilan gambar, namun penelitian tidak menjelaskan apakah menggunakan komponen *image* pula yang telah disediakan untuk pengujian Qwik. Pada pengujian ketiga (me-*render* aplikasi yang melakukan operasi berat dan pemrosesan data), Qwik memiliki nilai TBT terbaik, tetapi kinerjanya buruk dalam metrik lainnya. Next.js memimpin dalam metrik lainnya pada pengujian ini. Penelitian menyoroti bahwa Qwik, sebagai *framework* baru, mungkin memerlukan waktu untuk berkembang dan mencapai potensi penuhnya dalam mengoptimalkan kinerja aplikasi web [17].

Penelitian selanjutnya dilakukan oleh Nik Azlina Nik Ahmad dan Puteri Norliana Nor'Ain Megat Sazali (2021) yang berjudul *"Performing User Acceptance Test with System Usability Scale for Graduation Application"*. Penelitian ini membahas metode UAT dan SUS dalam pengujian aplikasi kelulusan yang berjalan di perangkat *mobile* dan web. Terdapat 45 responden dengan dua skenario berbeda untuk web dan aplikasi *mobile*. Terdapat 25 tugas yang diuji untuk setiap skenario UAT dan semuanya lolos pengujian tanpa ada kendala, memastikan bahwa aplikasi yang dikembangkan sudah dapat digunakan. Hasil kepuasan kemudian diukur dengan *System Usability Scale* dan mendapatkan rata-rata skor 73,7 atau setara dengan nilai B. Hasil menunjukkan bahwa skor berkorelasi dengan masalah yang ditemukan oleh partisipan. Batasan yang dihadapi adalah kurangnya komunikasi antara peneliti dan responden akibat dilaksanakan secara daring [12].

Selanjutnya buku yang berjudul *"The Scrum Guide"* oleh Ken Schwabber *and* Jeff Sutherland (2020) membahas kerangka kerja Scrum, yang digunakan dalam pengembangan perangkat lunak dan manajemen proyek. Buku ini

menjelaskan tiga peran utama dalam Scrum: *Scrum Master, Product Owner*, dan *Development Team*, serta peristiwa seperti *Sprint Planning, Daily Scrum, Sprint Review,* dan *Sprint Retrospective.* Schwaber dan Sutherland menekankan bahwa Scrum adalah pendekatan iteratif yang fleksibel, berfokus pada kolaborasi dan adaptasi. Lima nilai utama yang dijunjung dalam Scrum adalah *commitment, focus, openness, respect,* dan *courage*. Dengan penerapan yang tepat, Scrum membantu tim bekerja lebih efektif dan menghasilkan produk yang sesuai dengan kebutuhan pengguna.

2.3 Teknologi yang Digunakan

2.3.1 Visual Studio Code

Visual Studio Code (VSCode) adalah editor teks sumber terbuka yang dikembangkan oleh Microsoft yang dapat digunakan secara gratis oleh perorangan maupun perusahaan. VSCode memiliki fitur seperti *debugger,* git terintegrasi, *syntax highlighter, intellisense*, terminal serta mendukung berbagai bahasa pemrograman. Selain itu, pengembang dapat menambahkan ekstensi dan *plugin* untuk memperluas fitur dari VSCode yang dibuat oleh organisasi maupun individu pihak ketiga [18]. VSCode bersifat lintas platform, artinya perangkat lunak ini dapat digunakan pada berbagai macam sistem operasi seperti Linux, MacOS, Windows, bahkan dapat diakses melalui peramban. Editor teks ini memudahkan pengembang untuk membangun perangkat lunak dan sistem seperti *cloud*, aplikasi web, aplikasi seluler, gim, dan desktop [19].

2.3.2 Version Control System

Version Control System (VCS) adalah sistem yang digunakan untuk menampung kode sumber, dokumentasi, dan memanajemen proses pengembangan perangkat lunak. *Version Control System* membantu pengembang untuk melacak perubahan yang ada pada kode sumber sehingga proses pengembangan dapat lebih terkontrol. Selain itu, *Version Control System* membantu pengembang untuk berbagi kode sumber yang sama serta dapat mengintegrasikan proses *Continuous Integration* dan *Continuous Development (CI/CD)* [20].

Git adalah salah satu *Version Control System* yang paling populer dalam pengembangan perangkat lunak. Git bersifat sumber terbuka yang dikembangkan pada tahun 2005 oleh Linus Torvalds dan dapat menangani kode sumber secara cepat dan efisien. Salah satu produk yang dikembangkan berdasarkan sistem Git adalah Gitlab. Gitlab adalah aplikasi *DevOps* berbasis web yang menawarkan berbagai fitur untuk memanajemen proses *CI/CD* sekaligus memiliki semua manfaat dari *Version Control System* standar. Gitlab juga menyediakan alat manajemen tim, isu, dan lain-lain yang dirancang untuk memfasilitasi komunikasi dan kolaborasi dalam proses kerja *agile development* [21].

2.3.3 Javascript

Javascript adalah bahasa pemrograman yang digunakan dalam pengembangan web. Secara umum, Javascript dapat membuat halaman web menjadi dinamis dan interaktif sehingga hampir seluruh halaman web modern menggunakannya. Javascript juga banyak digunakan dalam pengembangan *backend* dengan menggunakan *runtime* seperti Node.js, Bun, dan Deno. Perkembangan Javascript kini telah menjamah pengembangan perangkat lunak untuk desktop dengan bantuan Electron, serta pengembangan aplikasi seluler dengan memanfaatkan React Native. Javascript adalah bahasa tertafsir dan menggunakan tipe data yang dinamis, artinya pengembang tidak perlu menentukan tipe data dari setiap variabel karena Javascript dapat menentukan tipe data dari setiap variabel secara otomatis [22].

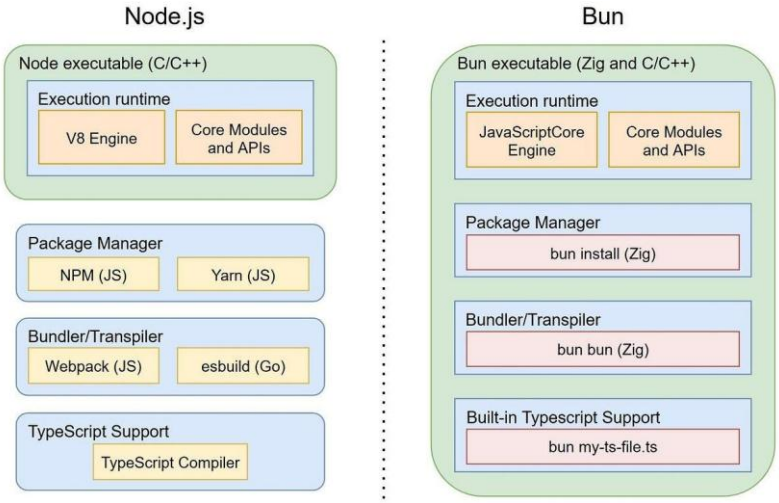
2.3.4 Typescript

Typescript adalah *superset* dari Javascript yang dikembangkan pada tahun 2012 oleh Microsoft. Typescript menggunakan seluruh fitur dari ECMAScript dengan tambahan sistem tipe data eksplisit. Dalam pengembangan perangkat lunak, penggunaan tipe data secara eksplisit dapat mengurangi risiko adanya *bug* pada aplikasi sehingga masalah yang dapat muncul nantinya dapat dimitigasi lebih dini. Kode sumber yang ditulis dengan Typescript akan di-*transpile* ke kode Javascript oleh *compiler* Typescript. Selama proses *transpilation, compiler* akan melaporkan *bug* yang berhubungan dengan tipe data. Meski demikian, anotasi tipe data secara manual adalah opsional [23]. Typescript banyak digunakan dalam pengembangan

web modern karena fitur-fitur yang telah dijelaskan sebelumnya, sehingga proses *debugging* dapat dipangkas menjadi lebih singkat dengan menggunakan tipe data yang eksplisit.

2.3.5 Bun

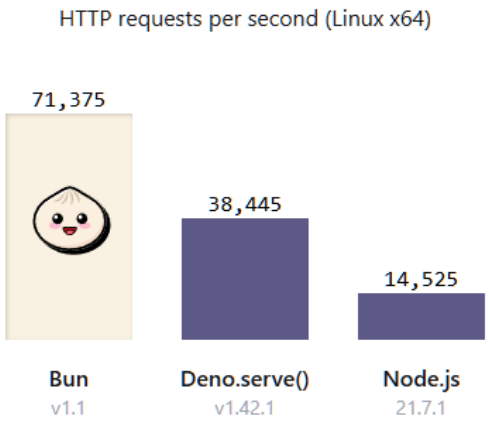
Bun adalah *all-in-one runtime* dan *toolkit* Javascript yang dikembangkan untuk menghadirkan alternatif *runtime* yang lebih cepat dan efisien dibandingkan dengan *runtime* yang telah ada seperti Node.js yang dibangun menggunakan C++, dan Deno yang dibangun dengan C++ dan Rust, keduanya menggunakan V8 Engine. Bun sendiri dikembangkan dengan bahasa Zig. Zig adalah bahasa pemrograman sistem tingkat rendah yang bersifat imperatif, serbaguna, bertipe statis, dan dikompilasi. Bahasa ini menekankan efisiensi dalam ukuran kode dan kemudahan pemrograman, sambil tetap mempertahankan kecepatan dan performa tinggi yang biasanya diharapkan dari bahasa sistem tingkat rendah seperti C atau C++ [24], [25].



Gambar 2.5 Perbandingan arsitektur Node.js dan Bun [26]

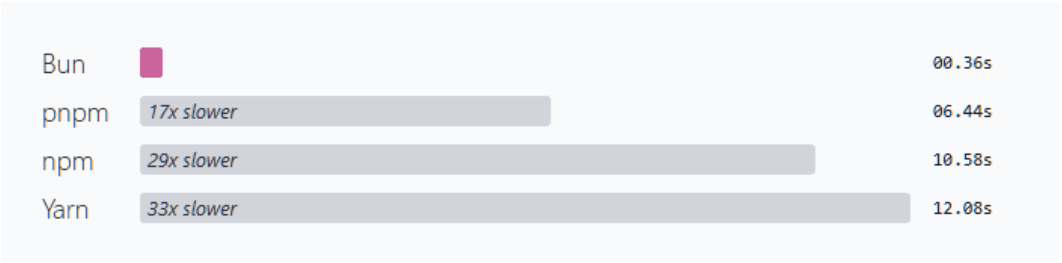
Berdasarkan Gambar 2.5 di atas, arsitektur Bun telah meliputi *runtime, package manager, bundler,* dan sudah memiliki dukungan Typescript sedari awal. Bun menggunakan JavascriptCore *engine*, yang juga digunakan oleh peramban Safari. *Engine* ini adalah perangkat lunak yang menjalankan kode JavaScript dengan aman dalam lingkungan terisolasi [27]. *Bun* kompatibel dengan ekosistem Javascript yang telah ada serta mendukung *API* Node.js. Bun mencapai

kompatibilitas ini dengan mengimplementasikan V8 Engine API di atas JavascriptCore [28]. Selain itu, Bun juga mendukung Typescript dan berkas JSX secara langsung tanpa perlu melakukan konfigurasi terlebih dahulu.



Gambar 2.6 Performa *Server-side Rendering* React [29]

Berdasarkan Gambar 2.6 di atas, Bun memiliki performa yang sangat signifikan dibandingkan dengan dua *runtime* Javascript lainnya seperti Deno dan Node.js. *Benchmark* dilakukan dengan menguji performa *rendering* halaman React pada sisi *server* dengan melakukan *HTTP request* pada sistem operasi Linux dengan arsitektur x64. Berdasarkan gambar di atas, *Bun* dapat menangani 71.375 *request* per detik. Sedangkan *Deno* dan *Node.js* dapat menangani 38.445 dan 14.525 *request* per detik.



Gambar 2.7 Performa kecepatan instalasi *dependencies* dari *cache* untuk *framework* Remix [29]

Berdasarkan bagan pada Gambar 2.7, Bun juga dapat melakukan instalasi *dependencies* dengan lebih cepat jika dibandingkan dengan *package manager* lain seperti pnpm, npm, dan yarn yang berbasis Node.js. Berdasarkan pengujian

benchmark yang dilakukan dengan *menginstal dependencies* dari *cache* untuk aplikasi Remix, Bun dapat menyelesaikan instalasi relatif lebih cepat [29].

2.3.6 Google Lighthouse

Google Lighthouse adalah alat yang digunakan untuk mengukur kinerja situs web berdasarkan berbagai parameter penting. Alat ini menyediakan laporan komprehensif yang mencakup lima dimensi utama yaitu performa, aksesibilitas, praktik terbaik, *Search Engine Optimization (SEO),* dan *Progressive Web App (PWA)*.

Ketika berhadapan dengan pengukuran berbasis waktu, hasil pengujian dapat menjadi tidak konsisten karena beberapa faktor seperti kecepatan jaringan, *web server,* serta perangkat keras pengguna sehingga menghasilkan fluktuasi terhadap hasilnya, maka perlu dilakukan pengujian yang berulang untuk mendapatkan rerata hasil yang lebih akurat [30].

Lima metrik utama dalam pengujian performa di antaranya:

a. *First Contentful Paint (FCP)*

   First Contentful Paint adalah salah satu metrik dalam kinerja web yang mengukur waktu yang dibutuhkan untuk menampilkan konten pertama yang dapat dilihat pengguna di layar. Konten pertama ini biasanya berupa teks, gambar, atau elemen *SVG* yang muncul dalam tampilan peramban.

b. *Speed Index*

   Speed Index adalah metrik yang mengukur seberapa cepat konten visual halaman web terlihat di layar bagi pengguna. Metrik ini memberikan nilai numerik yang mencerminkan waktu rata-rata yang diperlukan untuk menampilkan elemen-elemen penting secara progresif dalam *viewport* peramban. Lighthouse memulai proses dengan merekam video pemuatan halaman di peramban dan menganalisis perubahan visual di setiap *frame*.

c. *Total Blocking Time (TBT)*

   Total Blocking Time mengukur total waktu halaman tidak dapat merespons masukan pengguna, seperti klik, ketukan layar, atau penekanan tombol. Waktu ini dihitung dengan menjumlahkan durasi pemblokiran dari semua tugas yang

berlangsung lama antara *First Contentful Paint (FCP)* dan *Time to Interactive (TTI)*. Tugas dianggap sebagai tugas panjang jika berjalan lebih dari 50 milidetik. Bagian pemblokiran dihitung dari durasi tugas yang melebihi 50 milidetik. Misalnya, jika Lighthouse menemukan tugas dengan durasi 70 milidetik, maka bagian pemblokirannya adalah 20 milidetik.

d. *Largest Contentful Paint (LCP)*

   Largest Contentful Paint mengukur waktu ketika elemen konten terbesar di area tampilan selesai dimuat di layar. Metrik ini memberikan perkiraan kapan konten utama halaman menjadi terlihat oleh pengguna.

e. *Cumulative Layout Shift (CLS)*

   Cumulative Layout Shift adalah salah satu metrik yang mengukur stabilitas visual sebuah halaman web. *CLS* menghitung jumlah total dari semua pergeseran tata letak yang terjadi saat halaman dimuat. Pergeseran tata letak ini terjadi ketika elemen-elemen di halaman berpindah secara tiba-tiba, menyebabkan pengalaman pengguna menjadi terganggu.

Selain lima metrik di atas, terdapat metrik yang tidak digunakan lagi dalam hasil pengukuran Lighthouse yaitu *Time to Interactive. Time to Interactive (TTI)* adalah metrik kinerja web yang mengukur waktu yang dibutuhkan sebuah halaman untuk siap sepenuhnya digunakan oleh pengguna. Artinya semua elemen utama halaman telah dimuat, respons terhadap *input* pengguna cepat tanpa latensi signifikan, dan tidak ada tugas berat yang berjalan di latar belakang yang dapat menghambat interaksi. *TTI* dihitung sejak halaman mulai dimuat hingga halaman siap merespons semua interaksi pengguna dengan lancar. *Time to Interactive (TTI)* dianggap terlalu sensitif terhadap faktor seperti permintaan jaringan yang tidak biasa dan tugas berat yang panjang, yang menyebabkan metrik ini memiliki tingkat variabilitas yang tinggi. Akibatnya, *TTI* dihapus dari daftar metrik yang digunakan di Lighthouse versi 10. Sebagai gantinya, metrik lain seperti *Largest Contentful Paint (LCP)* dan *Total Blocking Time (TBT),* kini lebih disarankan untuk mengevaluasi kinerja interaksi halaman secara lebih andal [31].

2.3.7 Qwik

Qwik adalah *meta-framework* Typescript yang dikembangkan oleh Builder.io yang bersifat sumber terbuka, *DOM-centric*, dan memanfaatkan metode *resumability* yang dicanangkan oleh Miško Hevery pada tahun 2021. Qwik dirancang untuk fokus pada metrik *time-to-interactive (TTI)* untuk mencapai interaktivitas instan yang berfokus pada *resumability* dari *server-side rendering HTML* dan *lazy loading* yang agresif. Qwik menggunakan sintaksis yang mirip dengan React sehingga pengembang yang telah familiar dengan React dapat dengan mudah beradaptasi dalam pengembangan aplikasi web yang cepat dan interaktif. [32], [17], [33]. Berikut penjelasan terkait istilah penting yang digunakan oleh Qwik:

a. *Server Side Rendering*

   SSR adalah proses menghasilkan HTML di *server* sebelum dikirim ke klien. Qwik memanfaatkan SSR untuk memberikan halaman yang dapat langsung dilihat oleh pengguna tanpa harus menunggu Javascript diproses sepenuhnya, memungkinkan pengalaman yang lebih cepat dan ramah *SEO*.

b. *Hydration*

   Hydration adalah proses menghubungkan kembali elemen HTML statis yang di-*render* di *server* dengan logika interaktif Javascript di klien.

c. *Resumability*

   Resumability adalah metode yang memungkinkan aplikasi melanjutkan (*resume*) eksekusi Javascript dari *server* langsung di sisi klien tanpa perlu menjalankan kembali semua kode. *Resumability* ini berbeda dengan *hydration*, karena tidak membutuhkan inisialisasi ulang, sehingga mempersingkat waktu interaksi (*time-to-interactive*). *Resumability* memungkinkan efisiensi tinggi dengan hanya memuat bagian kode yang relevan berdasarkan interaksi pengguna [34].

d. *Lazy Loading*

   Lazy loading adalah teknik pemuatan sumber daya secara bertahap, hanya ketika dibutuhkan. Artinya komponen atau kode tidak dimuat di awal, melainkan saat pengguna berinteraksi dengan elemen tertentu. Teknik ini

mengurangi penggunaan memori dan meningkatkan kinerja dengan memprioritaskan elemen penting terlebih dahulu.

e. *Serialization*

Serialization adalah proses mengubah data atau status aplikasi menjadi format yang dapat dikirimkan melalui jaringan misalnya JSON. *Serialization* memungkinkan status aplikasi di-*serialize* di *server* dan dikirim ke klien untuk dilanjutkan tanpa memuat ulang seluruh aplikasi.

f. *Entry-point*

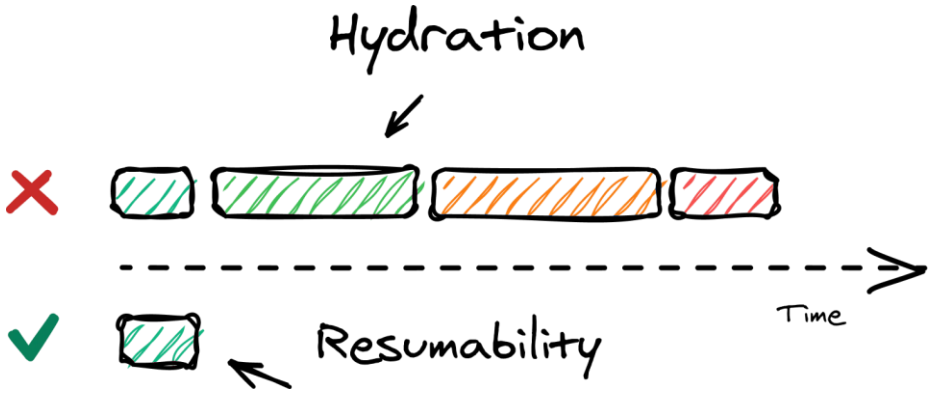
Entry-point adalah titik awal eksekusi aplikasi atau modul. *Entry-point* sangat terfokus, memastikan hanya kode yang relevan dimuat saat pengguna berinteraksi dengan aplikasi, mendukung prinsip *lazy loading* dan *resumability*.

g. *Event Handlers*

Event handlers adalah fungsi yang digunakan untuk menangani *event* seperti klik, *hover*, atau *input* pada elemen.

h. *State*

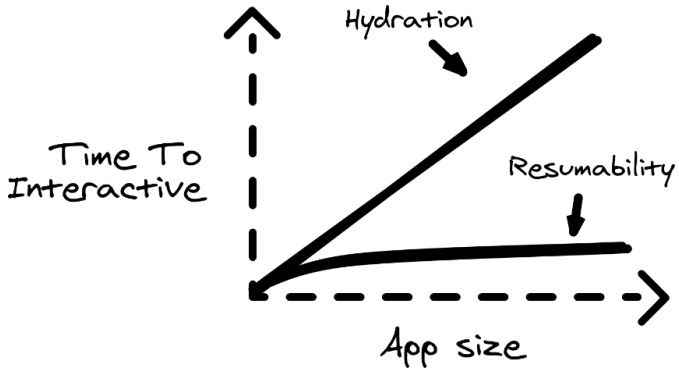
State adalah data atau status yang menentukan bagaimana sebuah komponen aplikasi terlihat dan berperilaku.



Gambar 2.8 Perbandingan *TTI* antara *Hydration* dan *Resumability* [32]

Resumability adalah cara cepat membuat halaman web menjadi interaktif tanpa proses *hydration*. Dengan *resumability, event handler* disimpan *(serialized)* selama *server-side rendering*. Proses ini memungkinkan peramban langsung merespons interaksi pengguna tanpa menjalankan ulang seluruh kode. *Resumability* juga membagi kode Javascript menjadi banyak bagian kecil *(entry-point)* yang

khusus untuk tiap *event handler,* biasanya kurang dari 1 KB, sehingga mengurangi kebutuhan *bandwidth* [35]. *Serialization* dalam proses ini memecah status situs web menjadi format yang bisa disimpan, dikirim ke klien, atau digunakan lagi, memungkinkan Qwik untuk *"pause"* di *server* dan melanjutkan *("resume")* di klien tanpa memuat ulang halaman.



Gambar 2.9 Kompleksitas waktu dari *resumability* dibandingkan *hydration* [32]

Dengan melakukan *lazy loading* yang agresif, *resumability* dapat mencegah peramban menjadi kewalahan pada pemuatan halaman pertama kali yang biasanya terjadi pada *framework* yang menerapkan *hydration* dengan memuat keseluruhan kode di awal sehingga membuat proses *lazy loading* tidak efektif. Dengan menerapkan ini, Qwik memiliki kompleksitas waktu sebesar O(1) dibandingkan *framework* yang menerapkan *hydration* yaitu O(n). Sehingga semakin besar ukuran basis kode tidak mempengaruhi performa dari framework yang menerapkan *resumability* [36].

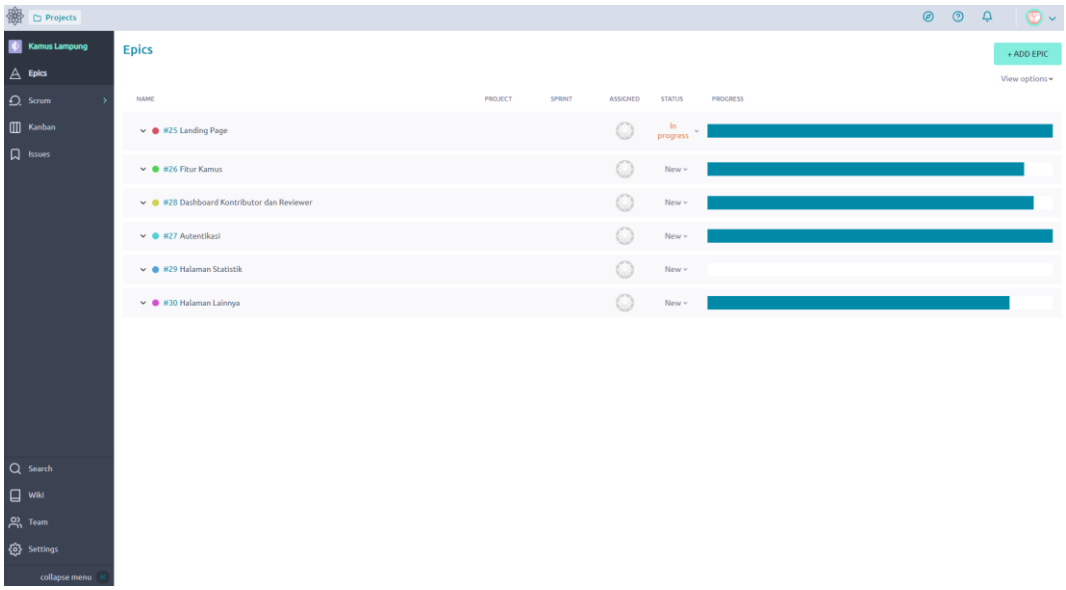
2.3.8 Application Programming Interface

Application Programming Interface (API) adalah suatu mekanisme yang memungkinkan berbagai layanan atau aplikasi untuk dapat saling bertukar informasi. *API* berfungsi sebagai jembatan antara dua atau lebih aplikasi untuk bertukar data secara aman. *API* mengekspos *endpoint* melalui internet sehingga pengembang dapat mengakses informasi yang ada di *server* melalui protokol HTTP. *API* menjadi tulang punggung platform digital karena membuat pengembang pihak ketiga untuk dapat mengakses fungsi utama dari suatu platform [37]. Dalam

pengembangan situs web yang dinamis, integrasi *API* diperlukan agar *frontend* dapat berkomunikasi dengan *backend* agar proses bisnis dari suatu situs web dapat berfungsi. Misalnya ketika pengguna hendak mengirim pesan, maka *frontend* akan mengirim permintaan kepada *backend* melalui *API* agar pesan tersebut dapat dikirim ke pengguna lain.

2.3.9 Taiga.io

Taiga adalah alat manajemen proyek multiplatform yang menerapkan metodologi *agile* melalui kerangka kerja *Scrum* dan *Kanban*. *Taiga* dibuat pada tahun 2013 sebagai bagian dari Kaleidos Hackathon sebagai solusi dari permasalahan di mana sulitnya mencari alat manajemen proyek yang intuitif dan mudah dipahami oleh pengembang [38]. *Taiga* memiliki berbagai macam fitur yang memudahkan pengelolaan proyek *Scrum* seperti *product backlog, epic, user story*, serta integrasi dengan berbagai alat pengembangan lainnya seperti Github dan Gitlab.



Gambar 2.10 Tampilan Taiga.io

III. METODE PENELITIAN

3.1 Waktu dan Tempat

3.1.1 Waktu

Tabel 3.1 Waktu pelaksanaan penelitian

No.	Kegiatan Penelitian	Des	Jan	Feb	Mar	Apr	Mei
1	Analisis kebutuhan						
2	Perencanaan						
3	*Sprint planning*						
4	*Sprint*						
5	*Sprint review*						
6	*Retrospective*						
7	*User Acceptance Test*						
8	Penulisan laporan						

Tabel 3.1 memetakan waktu pelaksanaan penelitian diawali dari tahap analisis kebutuhan sampai pengujian dan perawatan. Penelitian ini dilaksanakan selama tujuh bulan dimulai dari bulan November 2024 hingga Mei 2025.

3.1.2 Tempat

Penelitian ini dilaksanakan di laboratorium teknik komputer, Jurusan Teknik Elektro, Universitas Lampung di Jl. Prof. Dr. Ir. Soemantri Brodjonegoro No.1, Gedong Meneng, Kecamatan Rajabasa, Kota Bandar Lampung, Provinsi Lampung.

3.2 Alan dan Bahan Penelitian

3.2.1 Alat Penelitian

Penelitian ini menggunakan beberapa alat untuk mendukung pelaksanaan penelitian dari berbagai aspek pada tiap tahapannya. Penelitian ini menggunakan dua jenis alat, yaitu perangkat keras dan perangkat lunak. Berikut ini merupakan daftar alat yang digunakan dalam penelitian:

Tabel 3.2 Perangkat keras

Perangkat Keras	Fungsi
Laptop Ryzen 7 7735HS	Mengimplementasikan desain ke dalam bentuk program.
Kibor	Mengetik kata atau kalimat
Mouse	Menggerakkan kursor

Tabel 3.3 Perangkat lunak

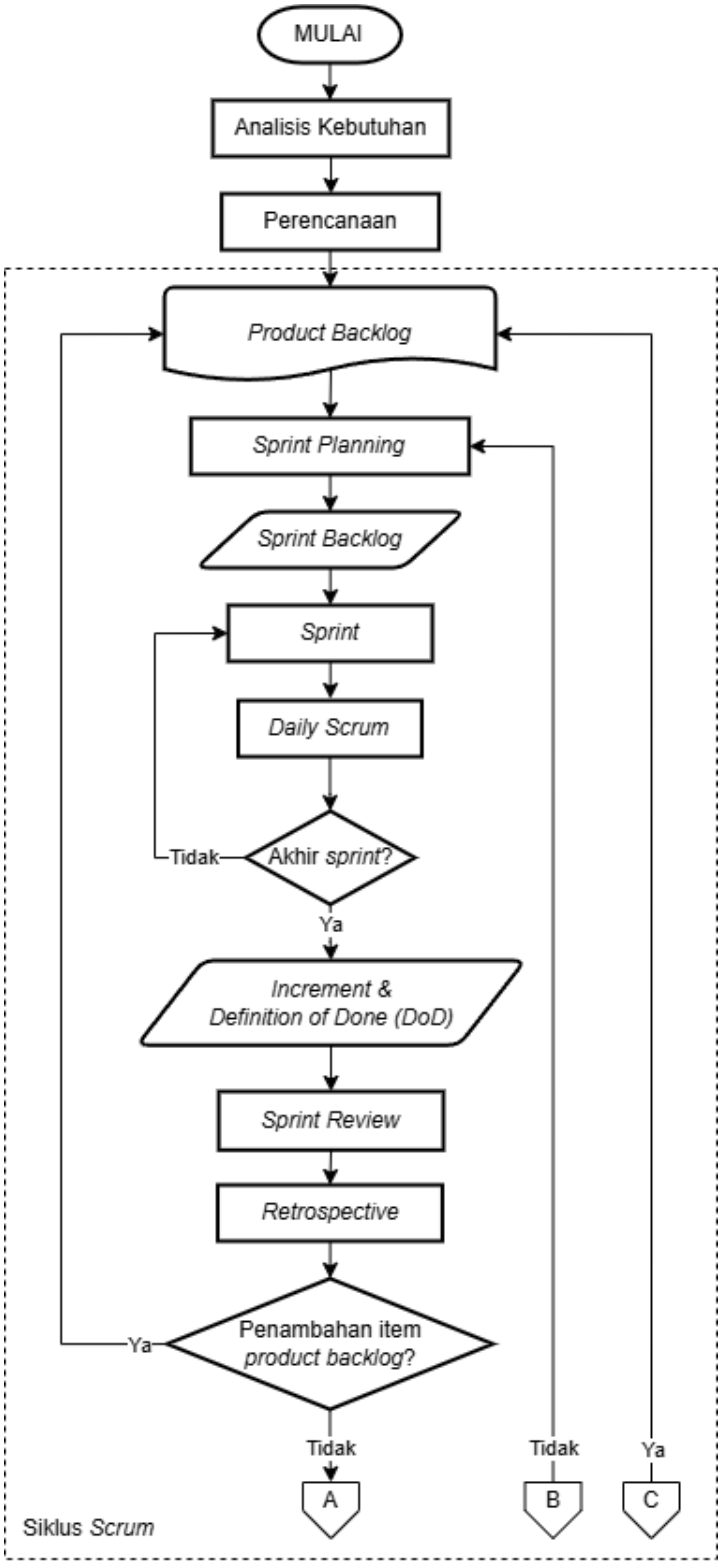
Perangkat Lunak	Versi	Fungsi
Visual Studio Code	1.95.3	Editor kode untuk implementasi program.
Peramban Arc	1.29	Media untuk implementasi situs web
Terminal	1.20	Antarmuka untuk menjalankan perintah untuk mengelola sistem
Figma	124.6.5	Alat untuk meninjau desain
Qwik	1.11.0	Pustaka untuk pengembangan kode program
HTML dan CSS	-	Kode program untuk pengembangan situs web
Typescript	5.6.3	Bahasa pemrograman untuk pengembangan web
Virtual Private Server	-	Alat untuk *deploy frontend website*
API Lamban Bahasa	-	Media integrasi data kamus Bahasa Lampung
Fon Aksara Lampung ciptaan Beni Anton	-	Menampilkan aksara Lampung pada layar.

3.2.2 Bahan Penelitian

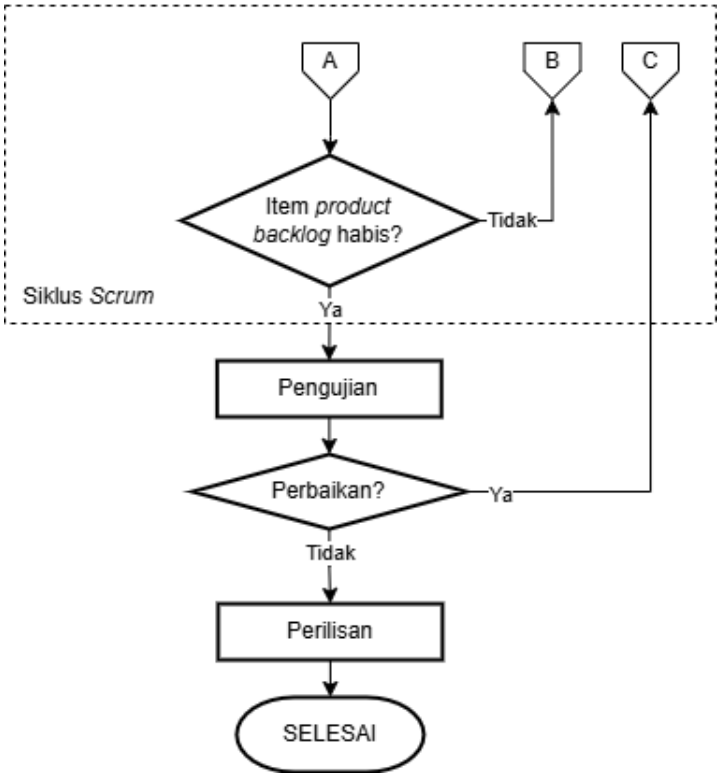
Penelitian ini menggunakan beberapa bahan untuk mendukung dan menjadi rujukan dalam pengembangan situs web kamus Bahasa Lampung digital ini.

1. Penelitian terdahulu yang relevan.
2. Artikel dan jurnal internasional dan nasional sebagai rujukan penelitian.
3. Dokumentasi dari situs resmi teknologi yang digunakan.
4. API kamus Bahasa Lampung digital sebagai sumber data terjemahan.
5. Analisis kompetitor berupa situs web dan aplikasi yang memiliki kesamaan fitur dalam beberapa aspek.

3.3 Tahapan Penelitian



Gambar 3.1 Tahapan penelitian



Gambar 3.1 Tahapan penelitian (lanjutan)

3.3.1 Analisis Kebutuhan

Mengumpulkan data dari berbagai sumber melalui metode observasi dan studi literatur sebagai bahan untuk penyusunan Spesifikasi Kebutuhan Perangkat Lunak (SKPL). Data yang dikumpulkan dikompilasi untuk digunakan sebagai acuan dalam menentukan kebutuhan pengguna dari sistem yang dikembangkan.

Pengumpulan informasi melalui studi literatur digunakan untuk menjadi bahan acuan dalam penelitian ini. Informasi yang dikumpulkan merupakan artikel, jurnal, dan dokumentasi yang kredibel dan relevan untuk dijadikan referensi dalam pengembangan sistem. Studi literatur yang dikumpulkan menggunakan bantuan perangkat lunak seperti Publish or Perish 7 dan Semantic Scholar untuk mempercepat proses pengumpulan referensi.

Selain itu, penelitian ini juga menggunakan metode observasi untuk pengumpulan data. Metode observasi menjadi dasar penentuan isu yang dibahas dalam penelitian ini yang mengangkat isu punahnya bahasa Lampung. Kemudian, observasi juga digunakan untuk menganalisis situs web serupa seperti Kasa Talk (Kamus Bahasa Sasak) dan Kamus Bahasa Lampung yang dikembangkan oleh Balai Bahasa Provinsi Lampung.

3.3.1.1 Analisis Aplikasi Serupa

Dalam tahap analisis kebutuhan, dilaksanakan analisis terhadap aplikasi serupa yang telah ada untuk membandingkan kekuatan, kelemahan, kesempatan, dan ancamannya sehingga pengembangan situs web kamus Bahasa Lampung ini dapat lebih terarah dalam menciptakan fitur-fitur yang relevan, meningkatkan pengalaman pengguna, dan memanfaatkan peluang untuk bersaing dengan produk serupa. Analisis ini juga membantu mengidentifikasi aspek yang perlu diperbaiki atau ditambahkan, serta memberikan landasan strategis untuk menciptakan solusi yang sesuai.

a. Kamus Lampung-Indonesia KBPL

Aplikasi ini memiliki keunggulan dalam menyediakan data yang cukup lengkap dan kemampuan pencarian berdasarkan satu kata. Namun, data bersifat statis dan tidak diperbarui secara berkala, dengan tampilan yang kurang menarik. Peluang utama aplikasi ini adalah meningkatkan tampilan dan membuka akses untuk kontribusi masyarakat dalam memperkaya data. Ancaman datang dari aplikasi serupa yang lebih populer dan relevansi data yang menurun tanpa pembaruan rutin.

b. Kamus Bahasa Lampung

Keunggulan aplikasi ini meliputi fitur pembelajaran aksara, budaya, dan sastra Lampung, serta kemampuan menerjemahkan teks hingga kalimat. Kelemahannya termasuk alur penambahan kata yang tidak jelas dan hasil terjemahan yang masih per kata. Peluang aplikasi ini terletak pada pengembangan fitur pembelajaran interaktif dan penggunaan teknologi AI untuk meningkatkan akurasi. Namun, ancaman berasal dari persaingan aplikasi dengan teknologi lebih canggih dan segmentasi pasar yang terbatas.

c. *Translate* Bahasa Lampung (Aplikasi Android)

Aplikasi ini memiliki keunggulan fitur OCR yang memungkinkan pengguna mengenali teks latin untuk diterjemahkan, serta mendukung berbagai dialek Lampung. Kekurangannya terletak pada tampilan teks yang kurang menarik, hasil terjemahan yang tidak sesuai kaidah, dan perbendaharaan kata yang terbatas. Peluangnya mencakup perbaikan tampilan, penambahan fitur aksara, dan perluasan kosakata. Ancaman yang dihadapi adalah kesulitan menjaga kualitas data tanpa kurasi dan kurangnya dukungan aksara.

d. Kamus Lampung (Aplikasi Android)

Aplikasi ini unggul dalam menampilkan daftar kata secara lengkap dan memiliki fitur untuk menyimpan kata favorit. Namun, kekurangannya mencakup iklan yang mengganggu, kurangnya kategori dialek, dan tampilan aplikasi yang tidak menarik. Peluangnya adalah memperbaiki tampilan dan menambahkan fitur kategori dialek untuk memperluas fungsionalitas. Ancaman datang dari minimnya daya tarik di tengah kompetisi aplikasi serupa.

3.3.1.2 Kebutuhan Fungsional

Kebutuhan fungsional menggambarkan fungsionalitas apa saja yang akan dimiliki oleh sistem. Kebutuhan fungsional mencakup fitur-fitur yang akan diterapkan untuk memenuhi kebutuhan pengguna. Berikut adalah tabel kebutuhan fungsionalitas sistem Kamus Bahasa Lampung:

Tabel 3.4 Kebutuhan Fungsional

No.	Kebutuhan	Penjelasan
1	Menerjemahkan kata	Sistem dapat menerjemahkan kata dari Bahasa Lampung ke Bahasa Indonesia dan sebaliknya
2	Melihat hasil terjemahan	Sistem dapat menampilkan hasil terjemahan termasuk aksara, dialek, wilayah penuturan, dan kontributor.
3	Melaporkan kata	Sistem mengizinkan pengguna untuk melaporkan kata
4	Melihat blog	Sistem menampilkan blog yang ada
5	Daftar	Sistem menyediakan fitur pendaftaran akun untuk pengguna baru.
6	Login	Pengguna dapat masuk ke dalam sistem dengan kredensial yang dimiliki

Tabel 3.4 Kebutuhan Fungsional (lanjutan)

No.	Kebutuhan	Penjelasan
7	Lupa kata sandi	Sistem menyediakan fitur lupa sandi jika pengguna tidak dapat masuk ke sistem
8	Atur ulang sandi	Sistem memiliki mekanisme untuk mengatur ulang kata sandi pengguna.
9	Melihat riwayat penambahan kata	Sistem mengizinkan pengguna dapat melihat riwayat kata yang telah mereka tambahkan ke sistem.
10	Sunting profil	Pengguna dapat menyunting informasi profil seperti nama, *email*, dan domisili
11	Kelola draf	Pengguna dapat mengelola draf kata yang disimpan seperti menghapus, menyunting, dan mengirimkan kata untuk ditinjau
12	Menambahkan kata baru	Pengguna dapat menambahkan kata baru ke dalam sistem.
13	Meninjau kata	Peninjau dapat memeriksa kata yang ditambahkan oleh kontributor.
14	Meninjau laporan	Peninjau dapat meninjau laporan yang dikirimkan oleh pengguna lain.
15	Melihat riwayat tinjauan	Peninjau dapat melihat riwayat laporan dan kata yang sudah mereka tinjau.
16	Melihat daftar seluruh kata	Admin dapat melihat seluruh kata yang ada dalam sistem.
17	Kelola data blog	Admin dapat menambah, menyunting, dan menghapus data blog.
18	Kelola data pengguna	Admin dapat menambah, menyunting, dan menghapus data pengguna.
19	Kelola data dialek dan wilayah	Admin dapat mengelola data terkait dialek dan wilayah untuk mendukung penerjemahan kata.

3.3.1.3 Kebutuhan Non Fungsional

Kebutuhan non-fungsional adalah kebutuhan yang menggambarkan kualitas dan karakteristik sistem yang tidak terkait langsung dengan fungsi atau fitur sistem, tetapi lebih kepada aspek kinerja, keamanan, dan keandalan sistem. Berikut merupakan daftar kebutuhan non fungsional sistem:

Tabel 3.5 Kebutuhan Non Fungsional

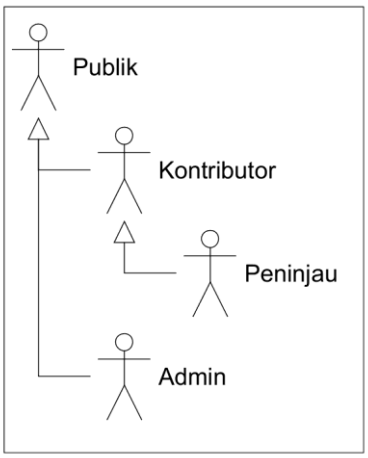
No.	Parameter	Penjelasan
1	*Availability*	1. Sistem harus dapat diakses terus menerus selama 24 jam dalam 7 hari (99% *uptime*)
2	*Reliability*	1. Sistem harus dapat menangani setidaknya 1000 pengguna secara simultan tanpa mengalami kegagalan sistem. 2. Harus menyediakan fitur *fallback* dan pemulihan dalam kasus kegagalan sistem
3	*Usability*	1. Antarmuka pengguna harus intuitif dan mudah digunakan 2. Situs web harus memenuhi standar WCAG 2.1 *(Web Content Accesibility Guidelines)*
4	*Portability*	1. Sistem harus dapat diakses melalui berbagai macam sistem operasi dan peramban
6	*Response Time*	1. Waktu respons sistem harus kurang dari 3 detik untuk pencarian kata dan tampilan hasil terjemahan. 2. Waktu respons untuk memuat halaman harus kurang dari 4 detik.
8	*Security*	1. Sistem harus mematuhi standar keamanan informasi, termasuk enkripsi data sensitif dan autentikasi pengguna yang aman.
9	*Maintainability*	1. Kode sumber harus dirancang agar mudah dipahami, dimodifikasi, dan diperbaiki

3.3.1.4 Use Case Diagram

Use Case Diagram menggambarkan fungsionalitas sistem yang hendak dibangun berdasarkan pandangan pengguna. Sistem situs web yang dikembangkan terdiri dari empat aktor yaitu publik, kontributor, peninjau, dan admin. Berikut merupakan tabel pendefinisian aktor dari sistem yang dibangun:

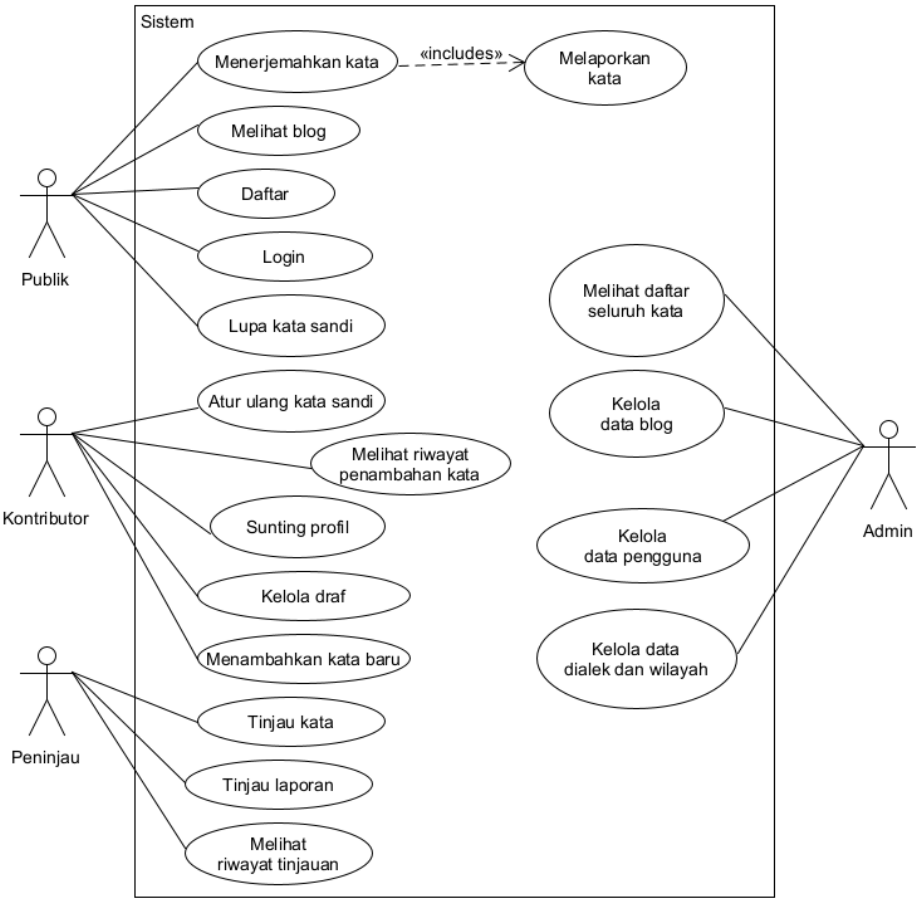
Tabel 3.6 Pendefinisian aktor

No.	Aktor	Deskripsi
1	Publik	Pengguna umum yang dapat mengakses fitur dasar seperti melihat blog, mendaftar, dan lupa kata sandi.
2	Kontributor	Pengguna terdaftar yang dapat menambahkan kata baru, melihat riwayat penambahan kata, serta menyunting profil.
3	Peninjau	Pengguna dengan hak akses untuk meninjau kata yang ditambahkan dan laporan yang diajukan oleh pengguna lain.
4	Admin	Pengguna dengan akses penuh untuk mengelola data pengguna, blog, kata, serta data dialek dan wilayah.



Gambar 3.2 Alur generalisasi aktor

Gambar 3.2 menunjukkan alur generalisasi aktor *use case*, di mana aktor publik adalah aktor induk. Aktor kontributor dan admin mewarisi semua relasi dari aktor publik, serta aktor peninjau mewarisi semua relasi dari aktor kontributor.



Gambar 3.3 *Use Case Diagram*

Berikut merupakan tabel pendefinisian *use case.*

Tabel 3.7 Definisi *Use Case*

No.	Aktor	Use Case	Deskripsi
1	Publik, Kontributor, Peninjau	Menerjemahkan kata	Pengguna dapat menerjemahkan kata dan melihat hasilnya.
2	Publik, Kontributor, Peninjau	Melaporkan kata	Pengguna dapat melaporkan kata yang dianggap tidak sesuai
3	Publik, Kontributor, Peninjau	Melihat blog	Pengguna dapat membaca blog yang ada
4	Publik	Daftar	Pengguna dapat melakukan pendaftaran
5	Publik	*Login*	Pengguna dapat masuk ke dalam sistem
6	Kontributor, Peninjau	Lupa kata sandi	Pengguna dapat mengatur ulang kata sandi

Tabel 3.7 Definisi *Use Case* (lanjutan)

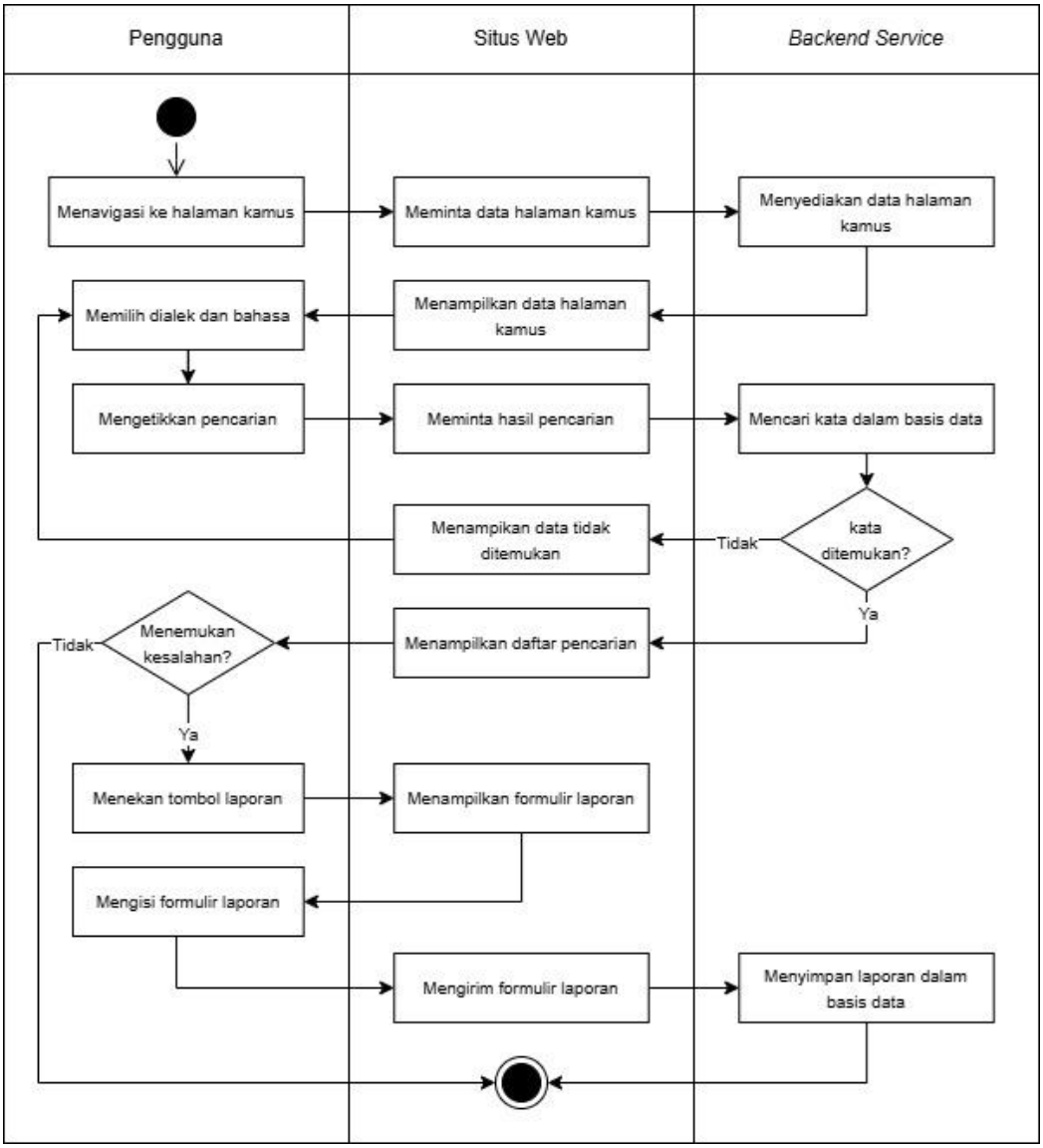
No.	Aktor	*Use Case*	Deskripsi
8	Kontributor, Peninjau	Atur ulang kata sandi	Pengguna dapat mengatur ulang kata sandi
9	Kontributor, Peninjau	Melihat riwayat penambahan kata	Pengguna dapat melihat riwayat penambahan kata sebelumnya
10	Kontributor, Peninjau	Sunting profil	Pengguna dapat menyunting informasi profil
11	Kontributor, Peninjau	Menambahkan kata baru	Pengguna dapat menambahkan kata baru ke dalam kamus
12	Kontributor, Peninjau	Kelola draf	Pengguna dapat menyunting, menghapus, dan mengajukan kata yang disimpan sebagai draf
12	Peninjau	Tinjau kata	Pengguna dapat meninjau kata yang ditambahkan
13	Peninjau	Tinjau laporan	Pengguna dapat meninjau laporan yang masuk
14	Peninjau	Melihat riwayat tinjauan	Pengguna dapat melihat riwayat tinjauan sebelumnya
15	Admin	Melihat daftar seluruh kata	Admin dapat melihat daftar seluruh kata yang ada
16	Admin	Kelola data blog	Admin dapat mengelola blog
17	Admin	Kelola data pengguna	Admin dapat mengelola data pengguna
18	Admin	Kelola data dialek dan wilayah	Admin dapat mengelola data dialek dan wilayah

3.3.1.5 Activity Diagram

Activity Diagram adalah penggambaran dari alur kerja atau aktivitas dalam sistem, yang digunakan untuk memvisualisasikan proses, tindakan, atau keputusan yang dilakukan pengguna maupun sistem. Diagram ini menunjukkan urutan langkah-langkah dari awal hingga akhir sehingga membantu memahami bagaimana alur aktivitas terjadi dalam sistem yang dikembangkan.

a. *Activity Diagram* menerjemahkan kata dan pelaporan

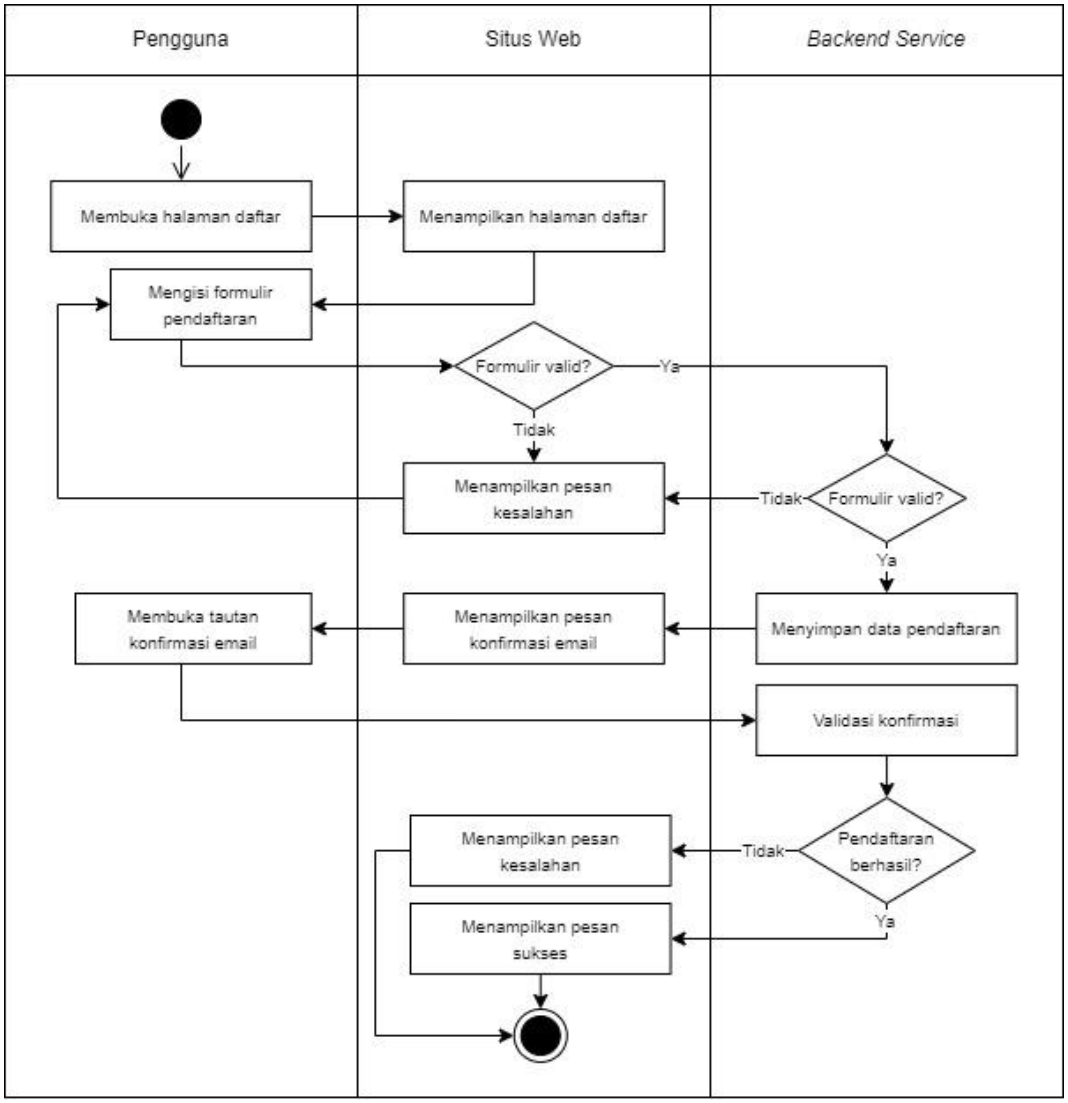
Gambar 3.4 menunjukkan alur *activity* pengguna dalam menerjemahkan dan melaporkan kata. *Activity* dimulai ketika pengguna membuka halaman kamus, dan situs web memuat data yang diperlukan dari *server*. Pengguna memilih bahasa, mengetikkan pencarian, dan sistem mencari kata di basis data. Jika kata tidak ditemukan, pesan kesalahan akan ditampilkan; jika ditemukan, daftar kata beserta detailnya akan disajikan. Bila pengguna menemukan kesalahan, mereka dapat melaporkan kata tersebut dengan mengisi formulir pelaporan yang akan ditinjau oleh peninjau.



Gambar 3.4 *Activity Diagram* menerjemahkan kata dan pelaporan

b. *Activity Diagram* daftar

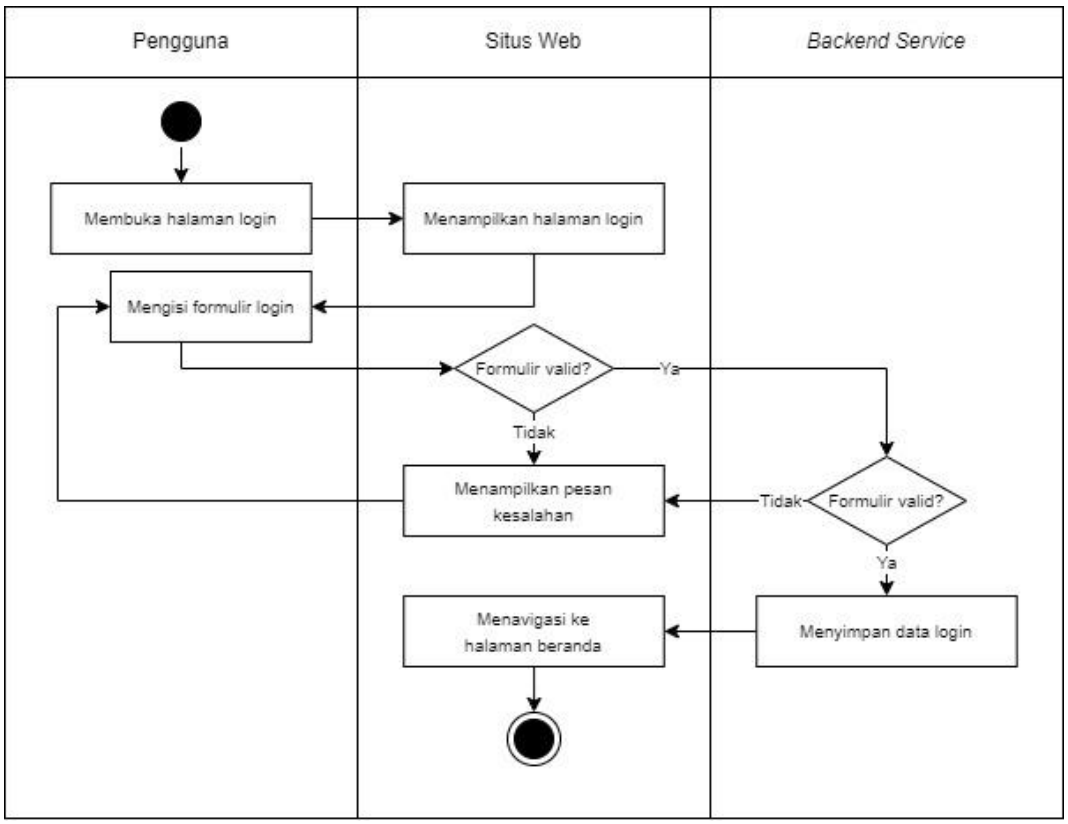
Gambar 3.5 menggambarkan alur *activity* proses pendaftaran kontributor baru. Proses dimulai dengan pengguna membuka halaman pendaftaran dan mengisi formulir. Sistem memvalidasi data pada sisi klien dan *server* untuk memastikan keamanan formulir. Jika validasi gagal, pesan kesalahan akan ditampilkan; jika berhasil, *backend* menyimpan data pendaftaran dan mengirim *email* berisi tautan konfirmasi. Pengguna kemudian membuka tautan konfirmasi, dan *backend* akan memvalidasi tautan tersebut. Jika gagal, pesan kesalahan muncul; jika berhasil, pengguna terdaftar dan menerima pesan sukses.



Gambar 3.5 *Activity Diagram* daftar

c. Activity Diagram login

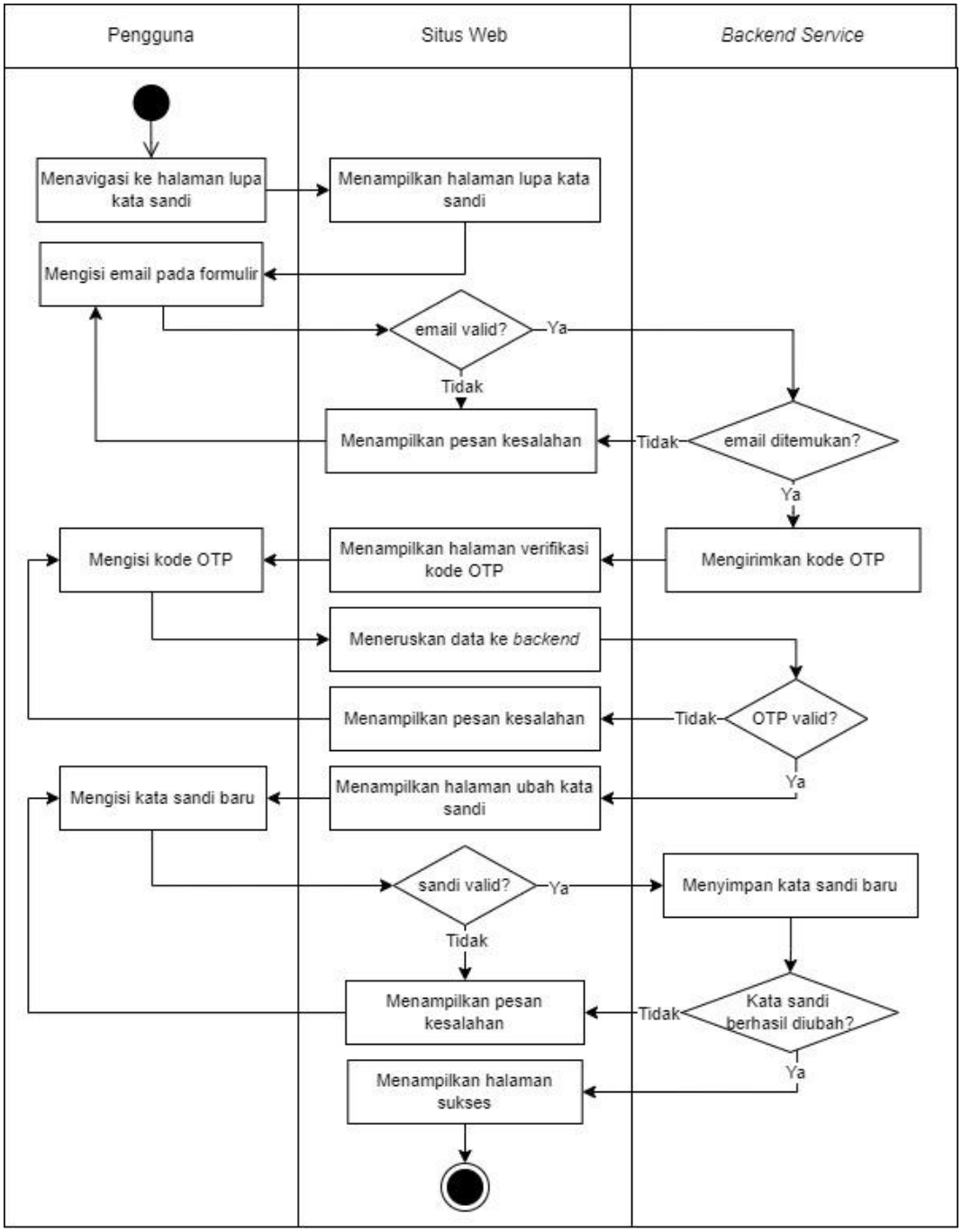
Gambar 3.6 menunjukkan alur proses *activity login* ke dalam sistem. *Activity* dimulai dengan pengguna membuka halaman *login* dan mengisi formulir dengan *email* dan kata sandi. Sistem memvalidasi kredensial *login* dari sisi klien dan *server*; jika tidak valid maka akan menampilkan pesan kesalahan dan pengguna perlu mengisi formulir kembali; jika valid maka *backend* akan menyimpan data *login* dan situs web akan menavigasikan pengguna ke halaman beranda.



Gambar 3.6 *Activity Diagram login*

d. *Activity Diagram* lupa kata sandi

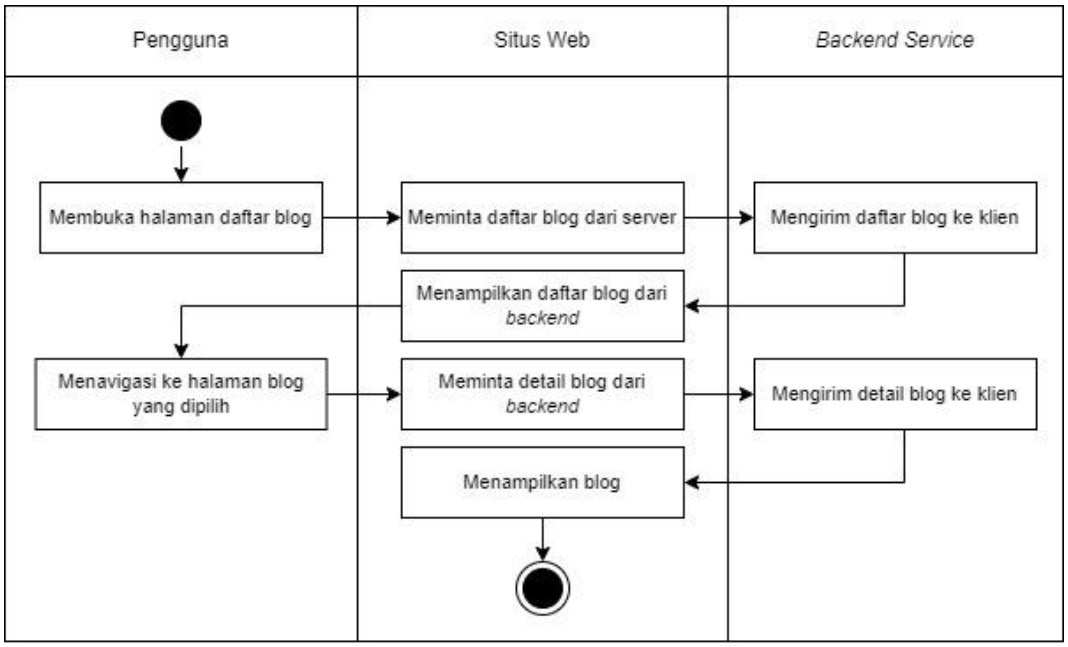
Gambar 3.7 menunjukkan alur *activity* pengguna lupa kata sandi. Proses dimulai dengan mengguna membuka halaman lupa kata sandi dan mengisi *email* pada formulir. Sistem akan memeriksa validitas *email* pada sisi klien dan *server*; jika tidak valid, maka situs web akan menampilkan pesan kesalahan; jika valid maka *backend service* akan mengirimkan kode OTP *(One Time Password)* ke *email* pengguna. Pengguna kemudian mengisi kode OTP pada kolom yang disediakan dan *backend service* akan memvalidasi kode; jika tidak maka pesan kesalahan ditampilkan dan perlu mengisi kode kembali; jika berhasil maka akan diarahkan ke halaman ganti kata sandi baru. Sistem kemudian melakukan validasi kembali terhadap sandi baru untuk memastikan keamanan data; jika tidak valid maka pengguna diminta memasukkan kembali; jika valid maka *backend service* akan menyimpan kata sandi baru dan menampilkan pesan sukses.



Gambar 3.7 *Activity Diagram* lupa kata sandi

e. *Activity Diagram* melihat blog

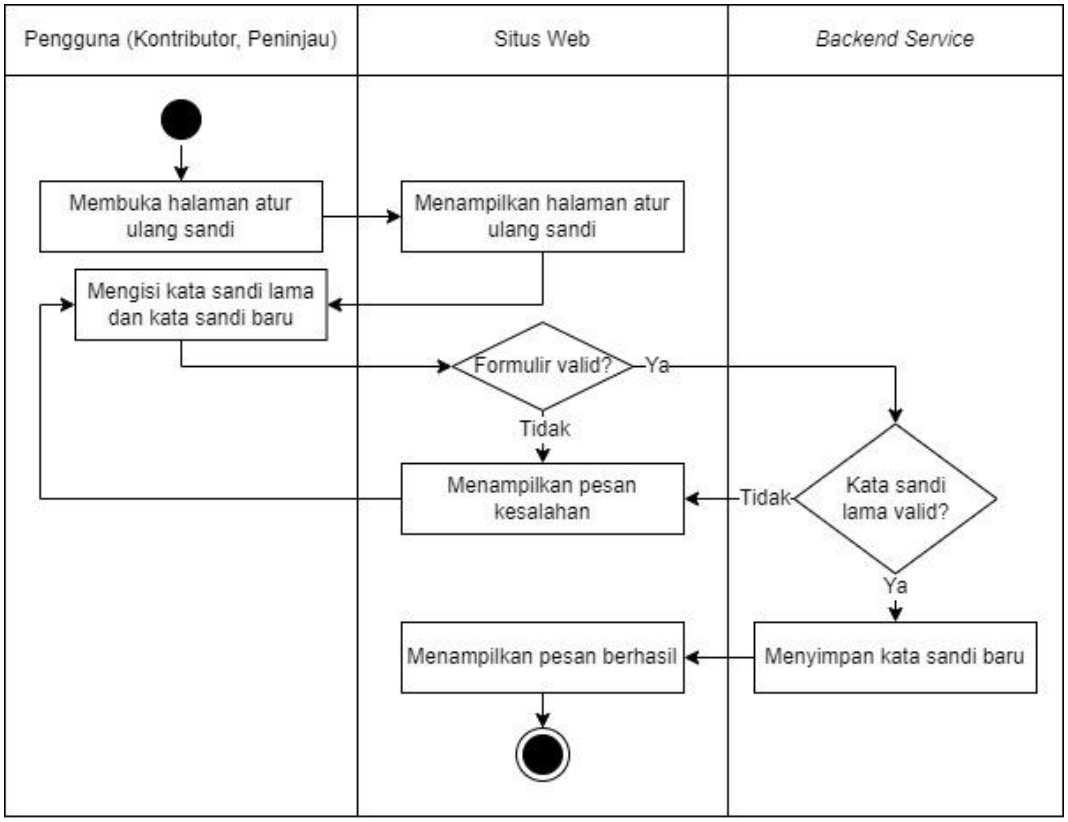
Berdasarkan Gambar 3.8 yang menggambarkan proses *activity* pengguna melihat blog. *Activity* dimulai dengan pengguna membuka halaman daftar blog, kemudian situs web akan meminta daftar blog dari *backend service* dan menampilkannya. Pengguna dapat memilih blog untuk dibaca dan menavigasi ke halaman detailnya. Sistem kemudian menampilkan detail dari blog yang dipilih.



Gambar 3.8 *Activity Diagram* melihat blog

f. *Activity Diagram* atur ulang kata sandi

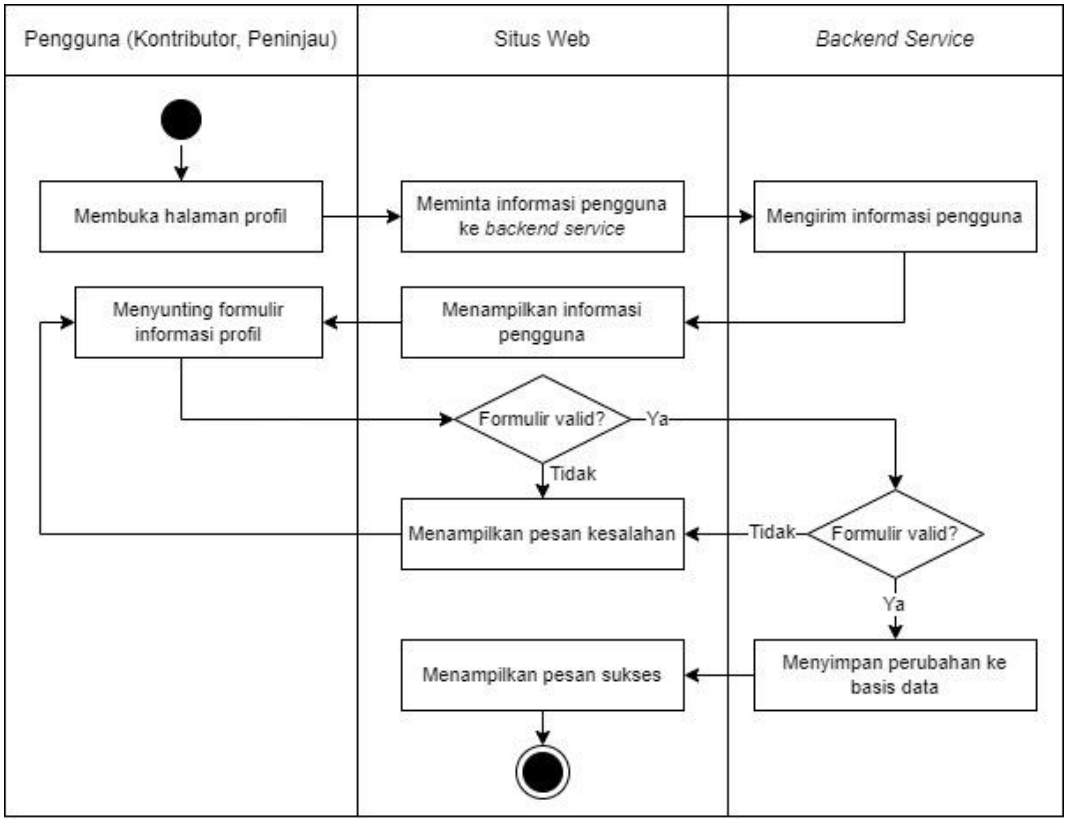
Gambar 3.9 menunjukkan alur *activity* mengatur ulang kata sandi. Proses dimulai dengan pengguna membuka halaman mengatur ulang kata sandi kemudian mengisi kata sandi lama dan kata sandi baru. Sistem kemudian melakukan validasi dari sisi klien dan *server* untuk memastikan informasi yang dimasukkan sudah tepat; jika tidak maka situs web akan menampilkan pesan kesalahan; jika sudah valid maka *backend service* akan menyimpan kata sandi baru dan menampilkan pesan sukses.



Gambar 3.9 *Activity Diagram* atur ulang kata sandi

g. *Activity Diagram* sunting profil

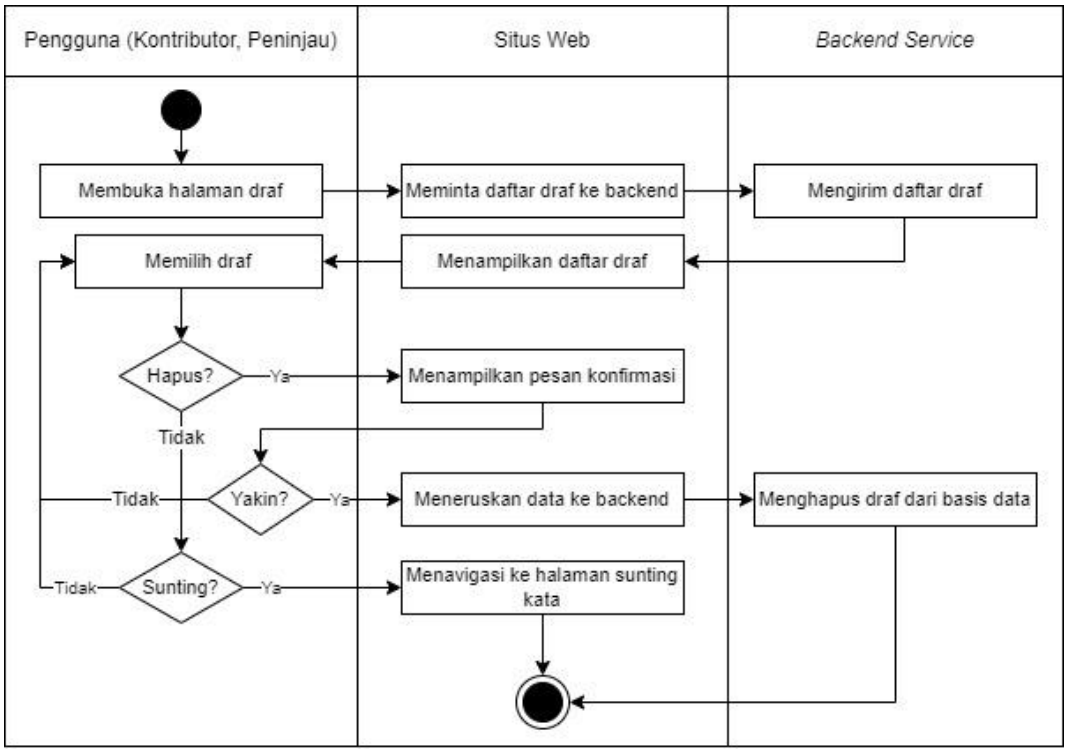
Gambar 3.10 menunjukkan alur *activity* pengguna menyunting informasi profil. Proses dimulai dengan pengguna membuka halaman profil dan situs web akan meminta informasi pengguna ke *backend service* dan menampilkannya. Pengguna kemudian dapat menyunting informasi profil dan sistem akan melakukan validasi data dari klien dan *server*. Jika tidak valid maka akan menampilkan pesan kesalahan; jika berhasil maka informasi baru akan disimpan pada basis data dan menampilkan pesan sukses.



Gambar 3.10 *Activity Diagram* sunting profil

h. *Activity Diagram* kelola draf

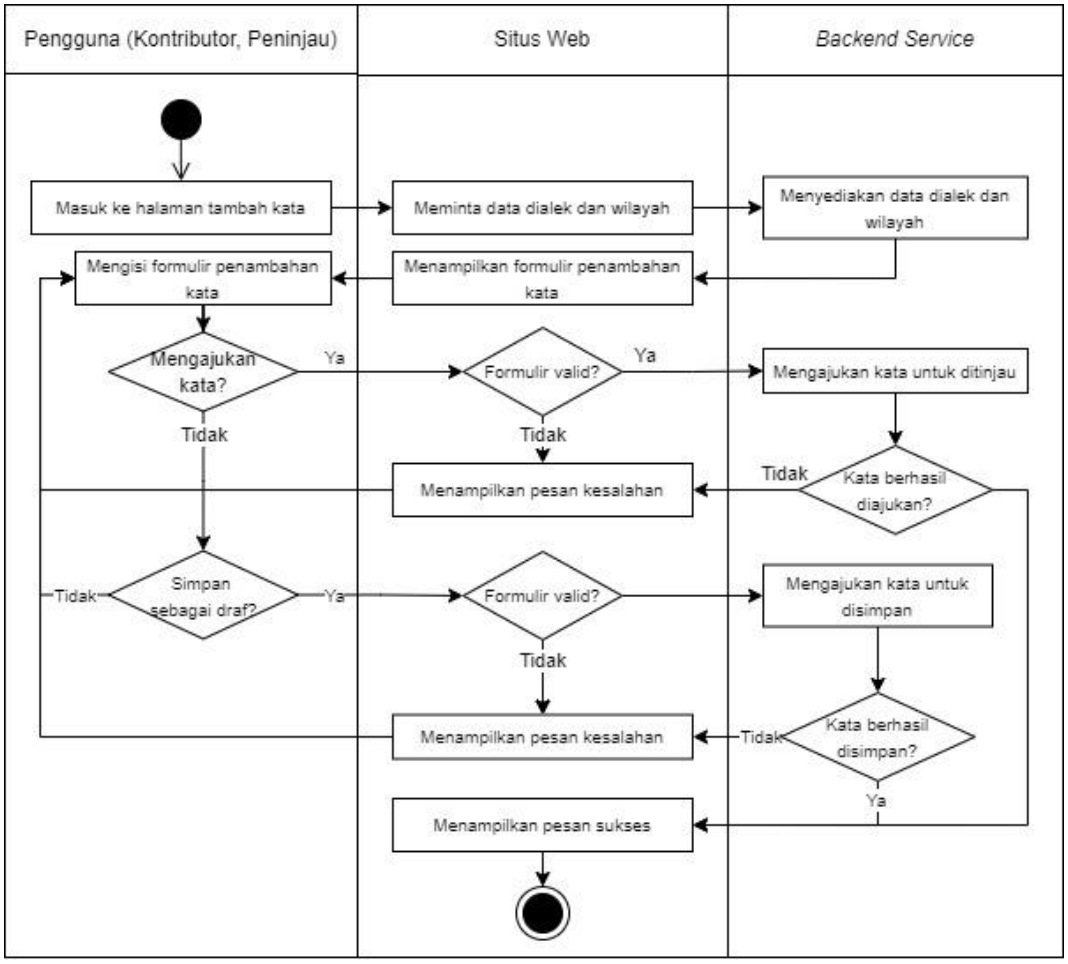
Gambar 3.11 menggambarkan proses *activity* pengguna mengelola kata yang disimpan sebagai draf. Proses dimulai dengan pengguna membuka halaman daftar draf. Pengguna kemudian dapat memilih draf dan melakukan aksi seperti hapus dan sunting untuk setiap draf. Jika pengguna memilih untuk menghapus draf maka situs web akan menampilkan pesan konfirmasi kepada pengguna; jika yakin maka *backend service* akan menghapus draf dari basis data; jika tidak maka pengguna dapat membatalkan penghapusan dan kembali memilih aksi. Jika pengguna ingin menyunting kata, maka situs web akan mengalihkan pengguna ke halaman sunting kata.



Gambar 3.11 *Activity Diagram* kelola draf

i. *Activity Diagram* menambahkan kata baru

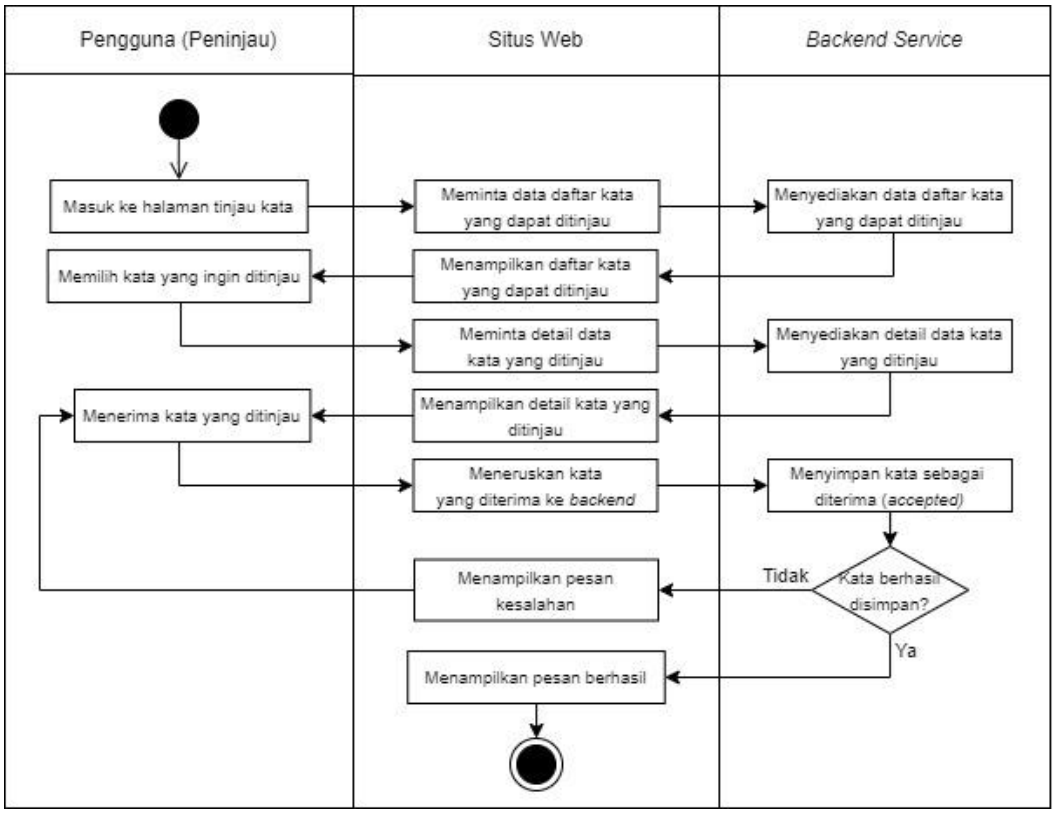
Gambar 3.12 menggambarkan alur proses pengguna dalam menambahkan kata ke sistem. Alur dimulai dengan pengguna, baik kontributor maupun peninjau, yang masuk ke halaman tambah kata. Setelah itu, mengisi formulir penambahan kata, yang membutuhkan data dialek dan wilayah dari *backend service*. Sistem kemudian melakukan validasi formulir. Jika formulir tidak valid, situs web menampilkan pesan kesalahan. Jika pengguna memutuskan untuk tidak mengajukan kata, pengguna dapat memilih untuk menyimpan kata sebagai draf. Namun, jika formulir diajukan, *backend service* akan memproses pengajuan dengan melakukan validasi lebih lanjut. Jika validasi *backend* berhasil, kata tersebut akan diajukan untuk tinjauan. Jika tidak, situs web akan menampilkan pesan kesalahan. Dalam kasus menyimpan sebagai draf, *backend service* memastikan kata dapat disimpan dengan benar. Proses diakhiri dengan pemberitahuan keberhasilan kepada pengguna jika pengajuan atau penyimpanan draf berhasil.



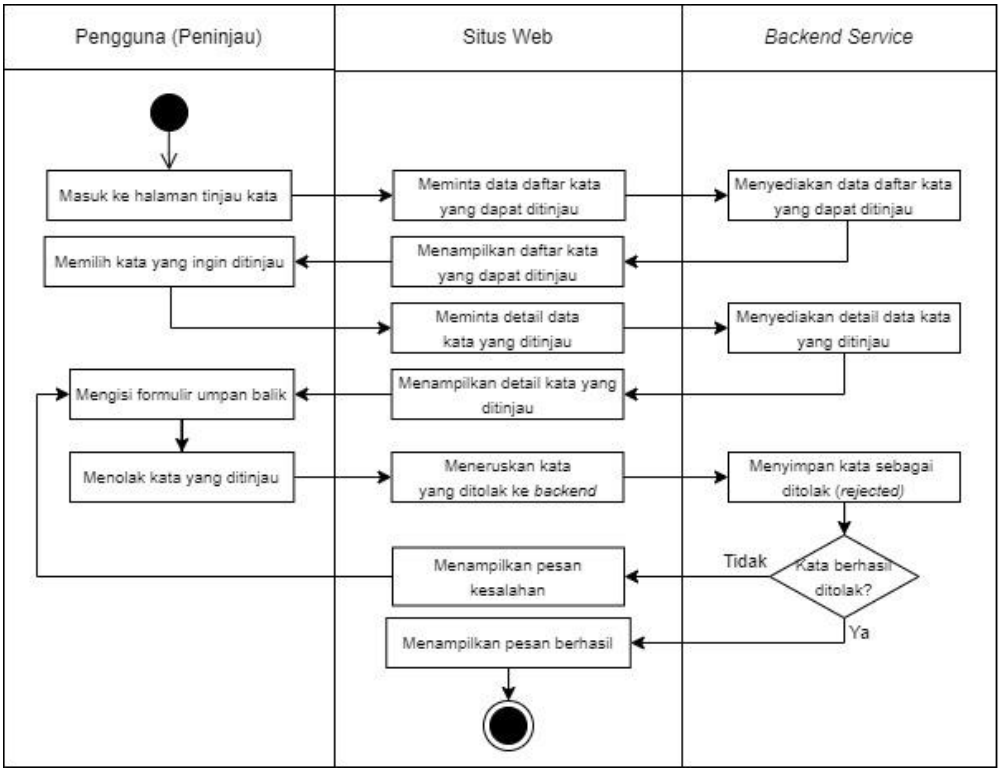
Gambar 3.12 *Activity Diagram* menambahkan kata baru

j. *Activity Diagram* tinjau kata

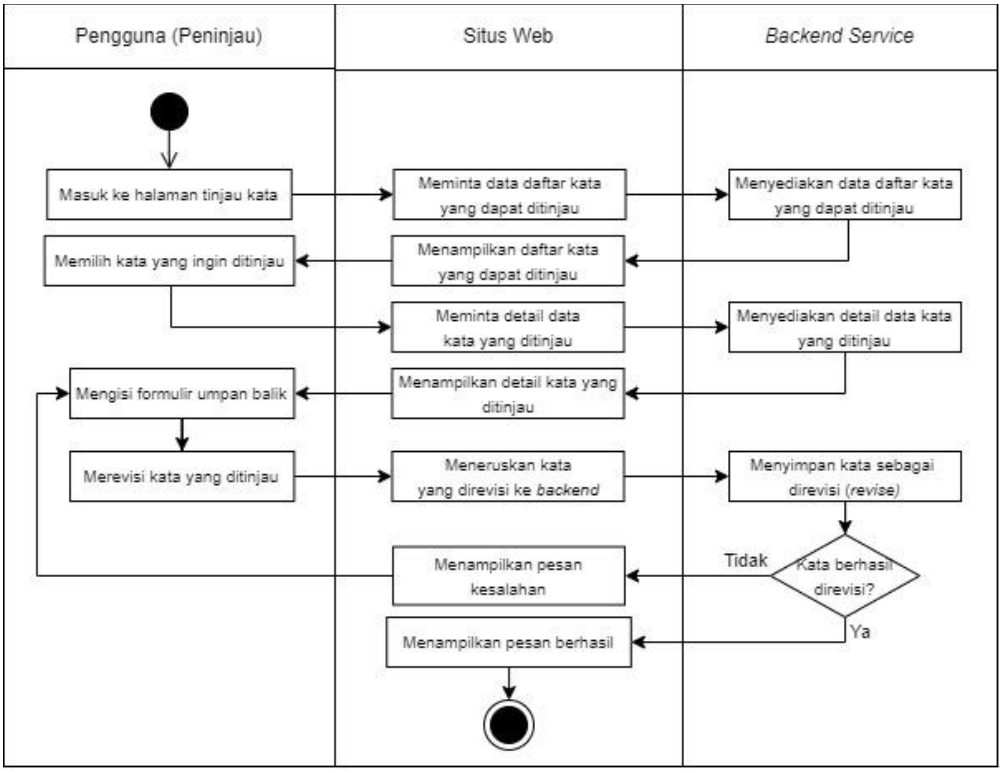
Gambar 3.13, Gambar 3.14, dan Gambar 3.15 menunjukkan alur *activity* pengguna meninjau kata dengan berbagai kondisi. Proses dimulai dengan pengguna membuka halaman tinjau kata dan sistem menampilkan daftar kata yang perlu ditinjau. Kemudian pengguna dapat memilih kata yang ingin ditinjau dan sistem akan menampilkan detail kata tersebut. Pengguna memiliki tiga pilihan, yaitu menerima, menolak, dan merevisi. Jika pengguna memilih menerima maka kata akan berstatus "diterima". Kata yang diterima akan ditampilkan pada hasil pencarian kamus. Kata yang ditolak akan dihapus dari basis data dan peninjau dapat memberikan alasan penolakan pada formulir umpan balik. Jika pengguna memilih untuk merevisi, maka pengguna dapat memberi alasan revisi dan kata tersebut akan dikembalikan kepada kontributor untuk direvisi. Untuk setiap pilihan, sistem akan meneruskan kata tersebut ke *backend* untuk diolah sesuai dengan pilihan pengguna. Jika gagal maka akan menampilkan pesan kesalahan, dan jika berhasil akan menampilkan pesan berhasil.



Gambar 3.13 *Activity Diagram* tinjau kata (diterima)



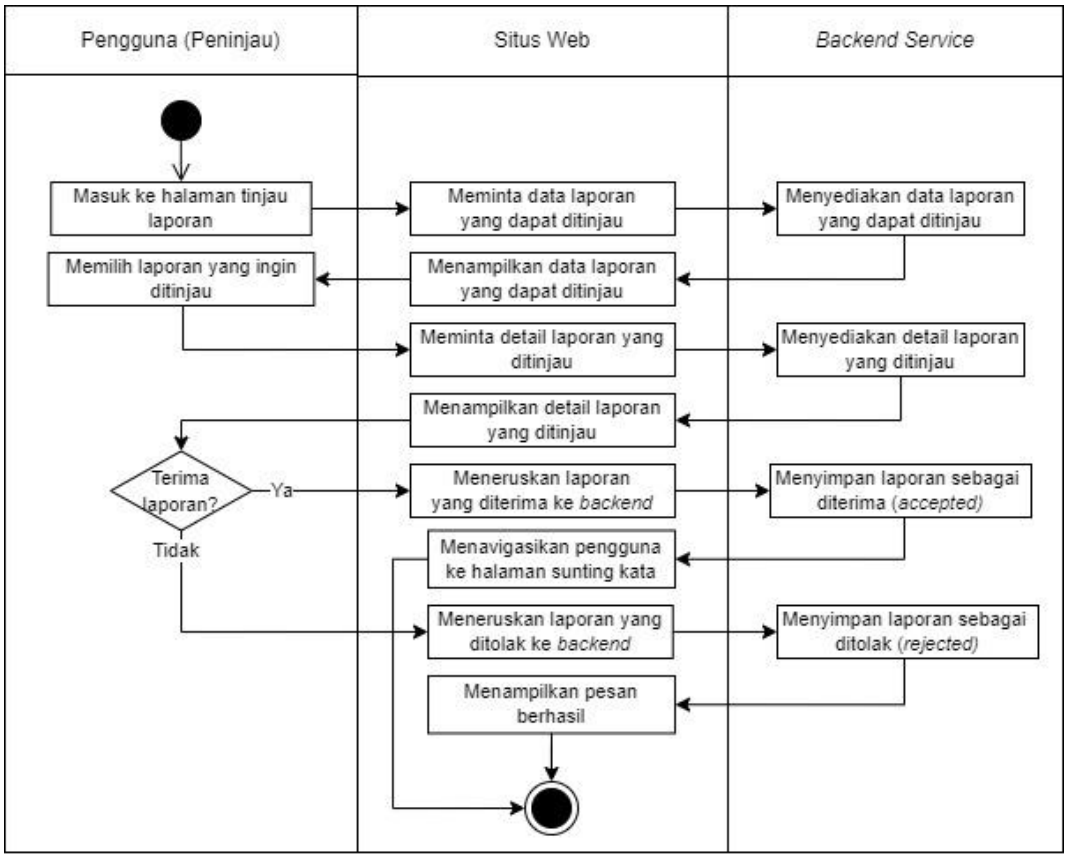
Gambar 3.14 *Activity Diagram* tinjau kata (ditolak)



Gambar 3.15 *Activity Diagram* tinjau kata (revisi)

k. *Activity Diagram* tinjau laporan

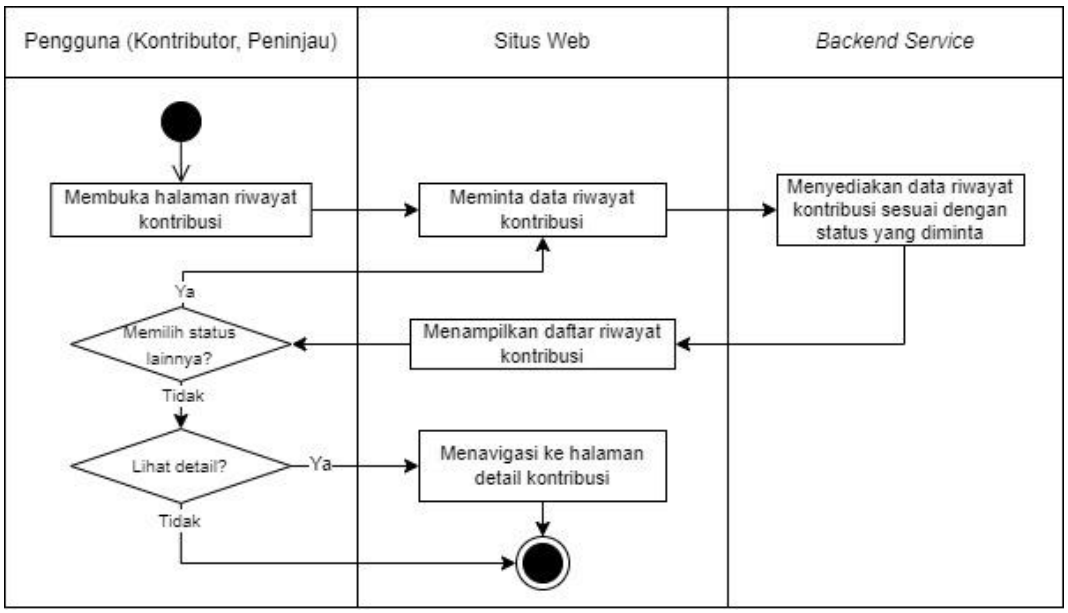
Gambar 3.16 menunjukkan alur *activity* pengguna meninjau laporan. Pertama, pengguna membuka halaman tinjau laporan dan sistem menampilkan daftar laporan. Pengguna dapat memilih laporan untuk ditinjau. Sistem akan menampilkan detail lebih lanjut tentang laporan yang dipilih. Pengguna kemudian dapat menerima atau menolak laporan. Jika pengguna menolak laporan, maka *backend service* akan mengubah status laporan menjadi "ditolak". Jika pengguna menerima laporan, maka status laporan menjadi "diterima" dan kemudian diarahkan ke halaman sunting kata untuk melakukan aksi yang sesuai seperti mengubah atau menghapus kata yang dilaporkan.



Gambar 3.16 *Activity Diagram* tinjau laporan

l. *Activity Diagram* melihat riwayat penambahan kata

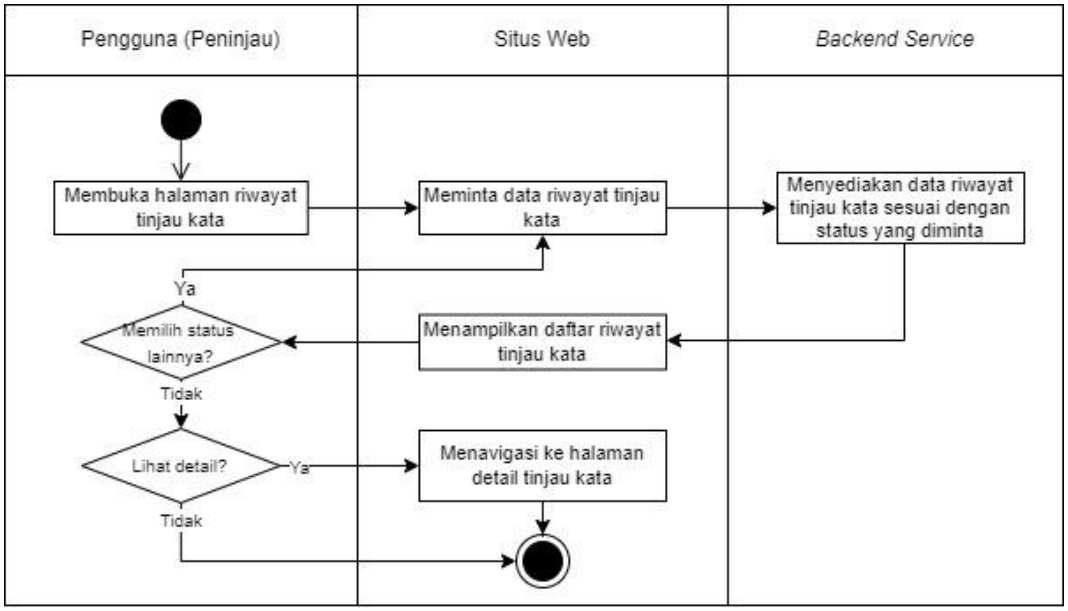
Gambar 3.17 menunjukkan alur pengguna melihat riwayat penambahan kata. Proses dimulai dengan pengguna membuka halaman riwayat kontribusi, situs web akan meminta daftar kontribusi dan menampilkannya. Pengguna dapat memilih riwayat untuk melihat detailnya, dan sistem akan menavigasikan pengguna ke halaman detail kata yang dipilih. Pengguna juga dapat menyaring daftar riwayat dengan memilih status lainnya.



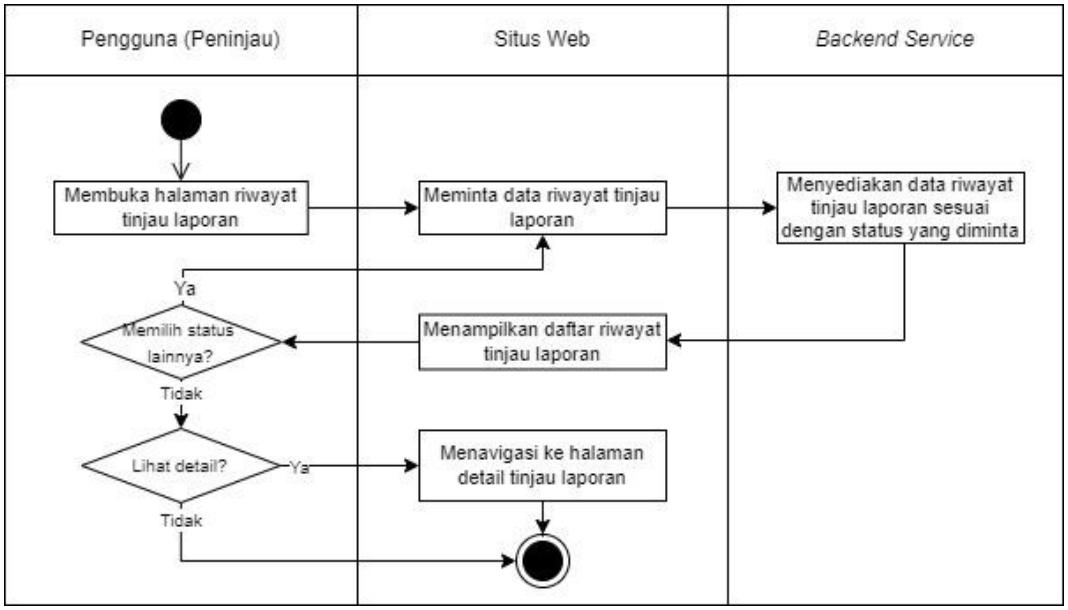
Gambar 3.17 *Activity Diagram* melihat riwayat penambahan kata

m. *Activity Diagram* melihat riwayat tinjauan

Gambar 3.18 menunjukkan alur proses melihat riwayat tinjauan kata, dan Gambar 3.19 menunjukkan alur proses melihat riwayat tinjauan laporan. Kedua diagram tersebut memiliki proses yang sama mirip. *Activity* dimulai dengan pengguna membuka halaman daftar riwayat tinjauan kemudian sistem menampilkan daftarnya. Pengguna dapat menyaring hasil pencarian berdasarkan status tinjauan. Pengguna juga dapat melihat detail lebih lanjut dari setiap tinjauan dan situs web akan mengarahkan pengguna ke halaman detail tinjauan yang dipilih.



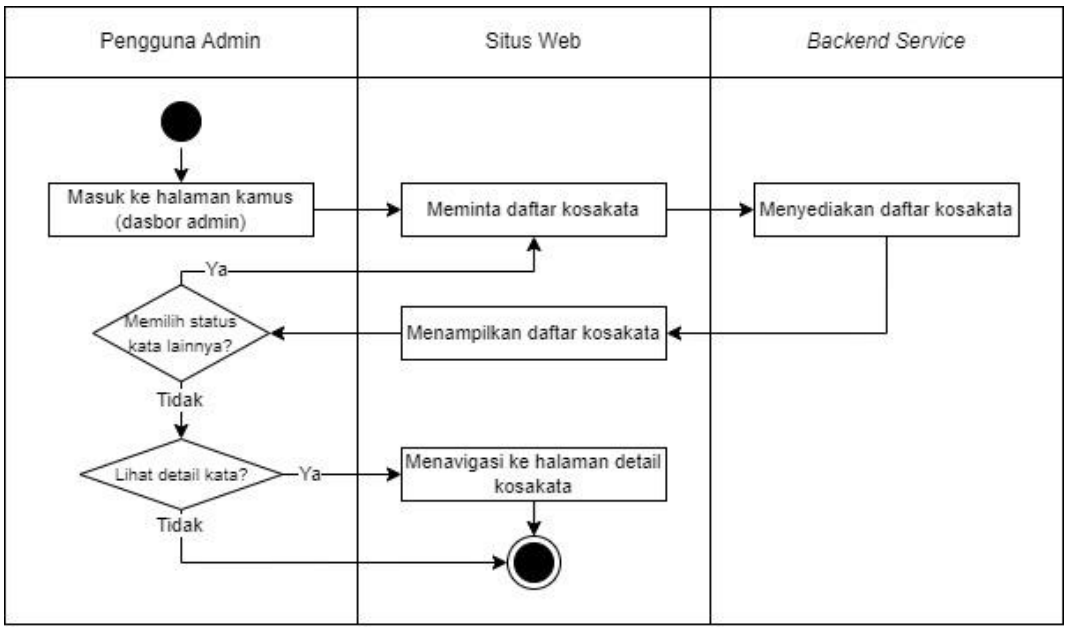
Gambar 3.18 *Activity Diagram* riwayat tinjau kata



Gambar 3.19 *Activity Diagram* riwayat tinjau laporan

n. *Activity Diagram* melihat daftar seluruh kata

Gambar 3.20 menunjukkan alur pengguna melihat daftar seluruh kata. Proses dimulai dengan pengguna membuka halaman kamus pada dasbor admin, situs web akan meminta daftar kosa kata dan menampilkannya. Pengguna dapat memilih kata untuk melihat detailnya, dan sistem akan menavigasikan pengguna ke halaman detail kata yang dipilih. Pengguna juga dapat menyaring daftar kata dengan memilih status lainnya.

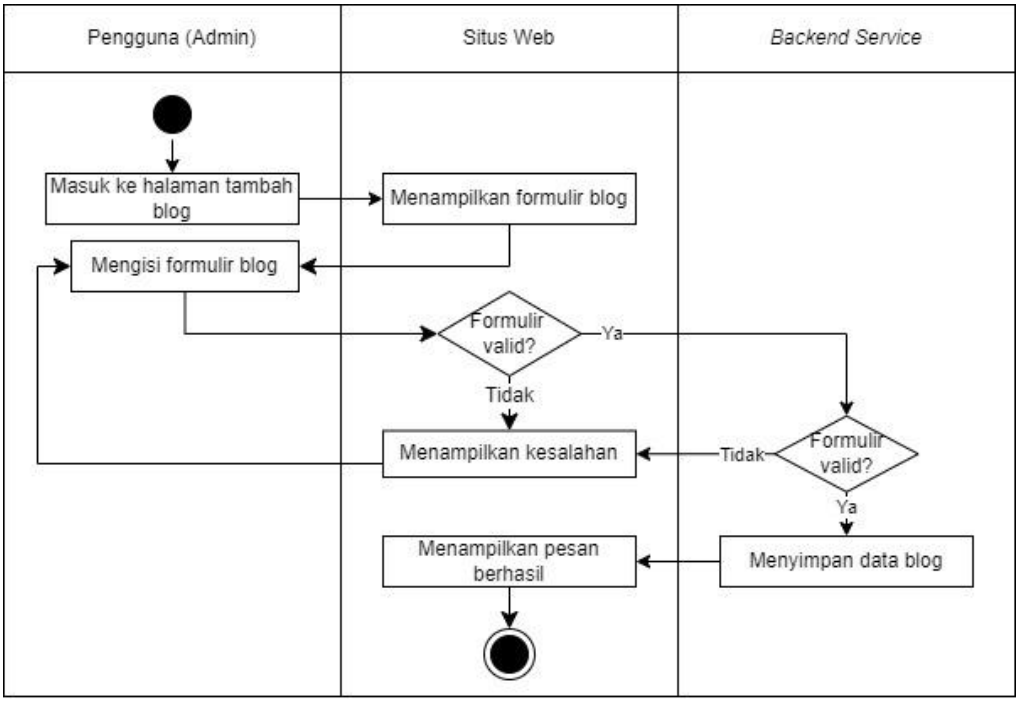


Gambar 3.20 *Activity Diagram* melihat daftar seluruh kata

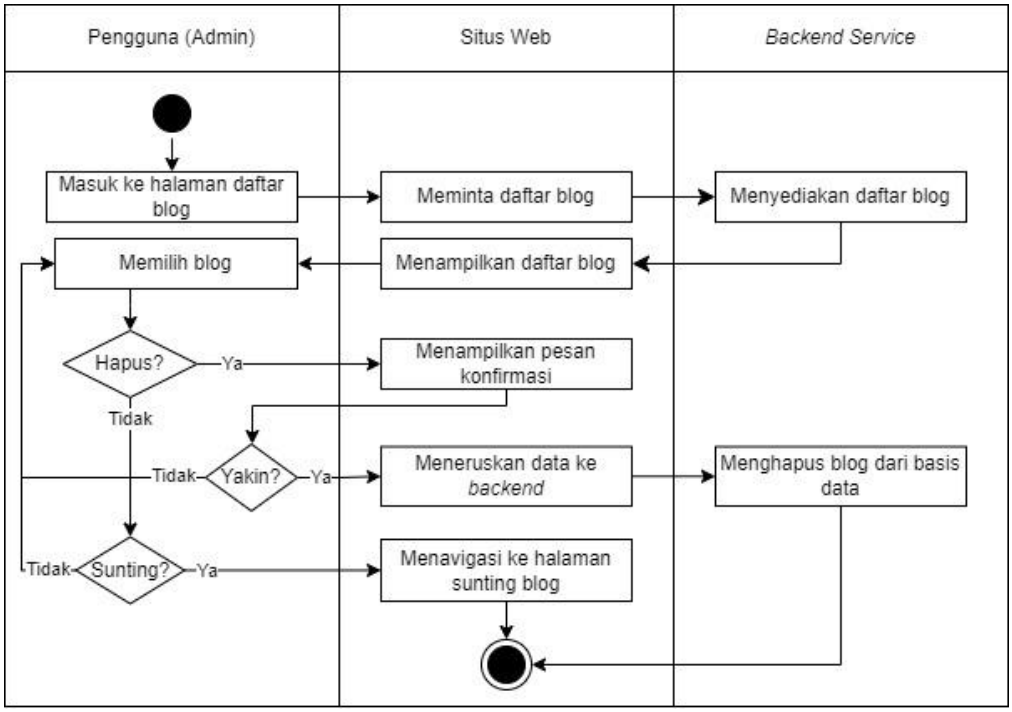
o. *Activity Diagram* kelola data blog

Gambar 3.21 menunjukkan *Activity Diagram* untuk membuat blog, di mana pengguna admin masuk ke halaman tambah blog dan mengisi formulir yang disediakan. Sistem akan memvalidasi formulir yang diisi; jika tidak valid, akan menampilkan kesalahan. Jika valid, data blog akan diteruskan ke layanan *backend* untuk disimpan, dan sistem menampilkan pesan berhasil. Gambar 3.22 menggambarkan *Activity Diagram* untuk proses menyunting dan menghapus blog. Proses dimulai dengan pengguna masuk ke halaman daftar blog, situs web akan menampilkan daftar blog yang diminta dari layanan *backend*. Pengguna dapat memilih untuk menghapus blog, yang akan meminta konfirmasi dan melanjutkan

data ke *backend* untuk dihapus. Alternatifnya, pengguna dapat memilih untuk menyunting blog, yang akan menavigasikan mereka ke halaman sunting blog.



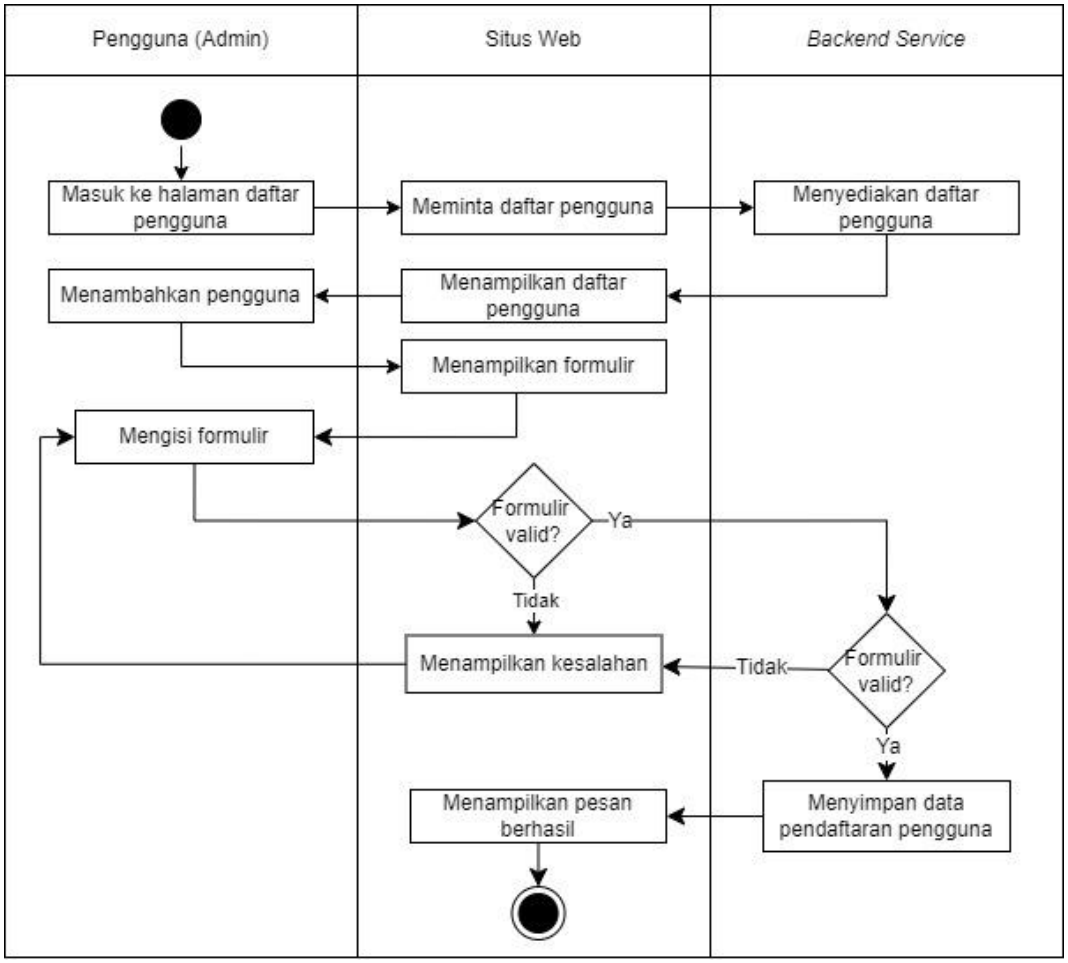
Gambar 3.21 *Activity Diagram* buat blog



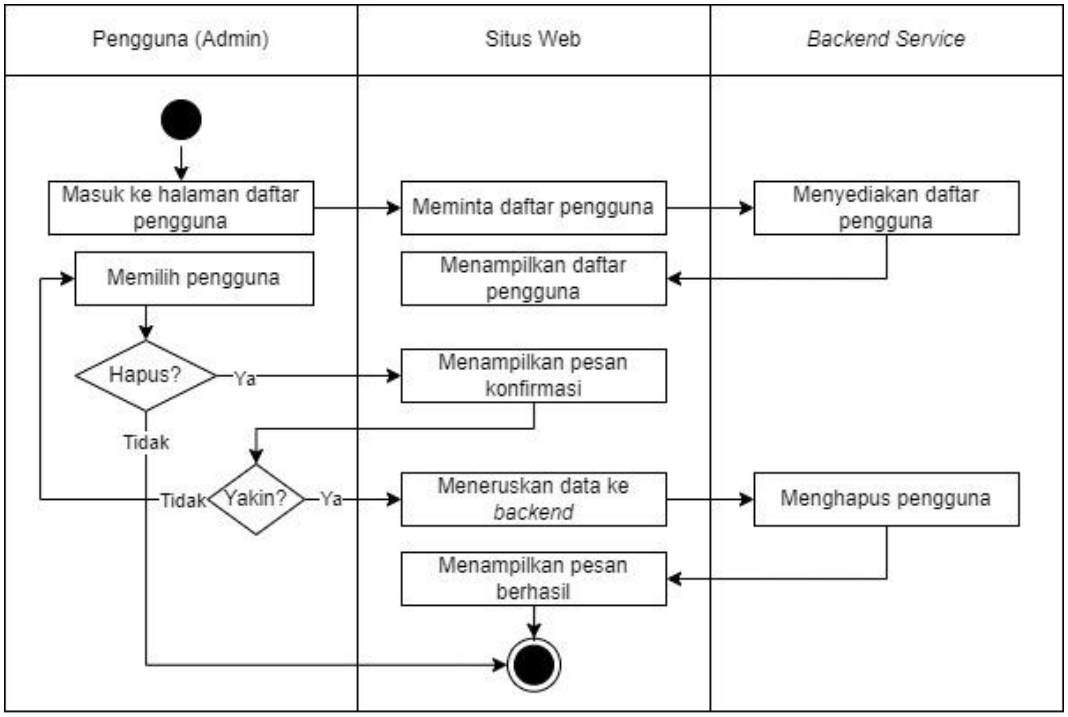
Gambar 3.22 *Activity Diagram* sunting dan hapus blog

p. *Activity Diagram* kelola data pengguna

Gambar 3.23 dan Gambar 3.24 menunjukkan alur aktivitas dalam mengelola pengguna pada sistem. Pada Gambar 3.23, proses dimulai dengan admin masuk ke halaman daftar pengguna. Sistem meminta data pengguna dari layanan *backend* dan menampilkan daftar tersebut. Admin dapat menambah pengguna dengan mengisi formulir yang valid. Jika formulir tidak valid, sistem menampilkan pesan kesalahan, sedangkan jika valid, data akan disimpan di *backend*, dan sistem memberikan pesan berhasil. Pada Gambar 3.24, admin memilih pengguna dari daftar untuk dihapus. Setelah sistem menampilkan pesan konfirmasi, admin dapat melanjutkan proses penghapusan jika yakin, dan sistem menghapus data pengguna dari *backend* serta menampilkan pesan berhasil.



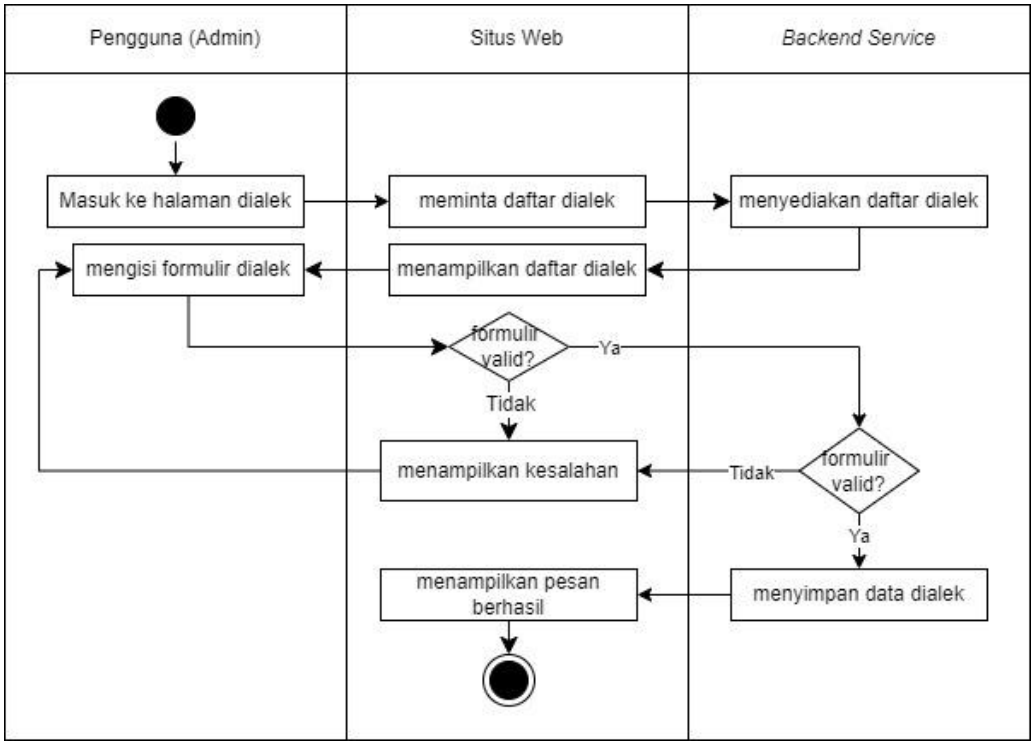
Gambar 3.23 *Activity Diagram* tambah pengguna



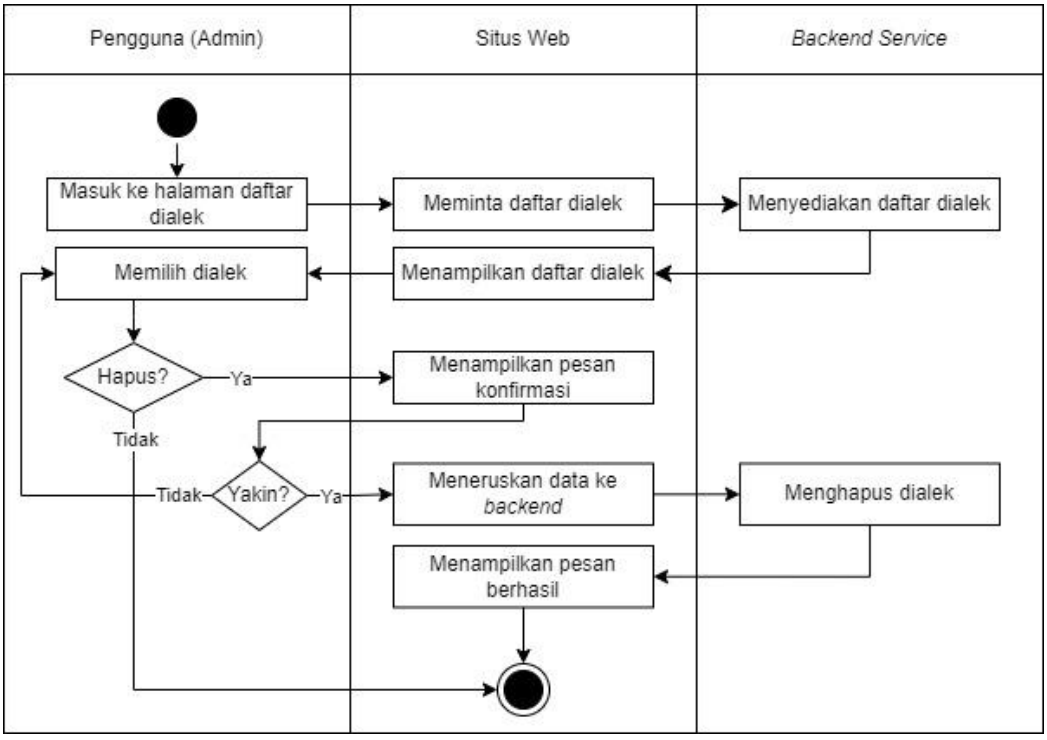
Gambar 3.24 *Activity Diagram* hapus pengguna

q. *Activity Diagram* kelola data dialek dan wilayah

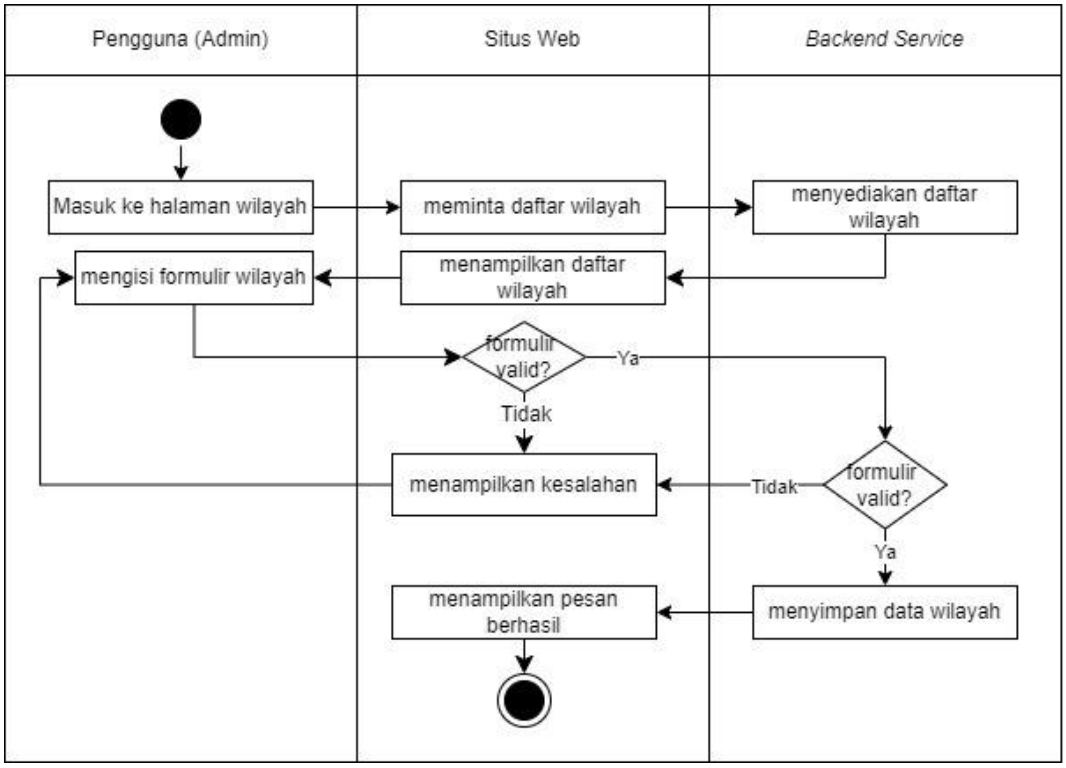
Gambar 3.25 sampai Gambar 3.28 menunjukkan alur *activity* pengguna mengelola data dialek dan wilayah. Gambar 3.25 dan Gambar 3.27 menggambarkan alur pengguna menambahkan data dialek dan wilayah. Alur dimulai dengan pengguna membuka halaman dialek dan wilayah kemudian menampilkan datanya. Kemudian pengguna dapat menambahkan data baru dengan mengisikan formulir yang berisi nama dialek atau wilayah. Sistem akan melakukan validasi terhadap formulir. Jika tidak valid, maka pesan kesalahan akan ditampikan. Jika valid maka data akan disimpan dan menampilkan besan berhasil. Gambar 3.26 dan Gambar 3.28 menunjukkan alur pengguna menghapus data dialek dan wilayah. Proses dimulai dengan pengguna membuka halaman dialek dan wilayah kemudian sistem menampilkan datanya. Pengguna dapat memilih data mana yang ingin dihapus, dan situs web akan menampilkan pesan konfirmasi. Jika pengguna yakin maka sistem akan menghapus data tersebut.



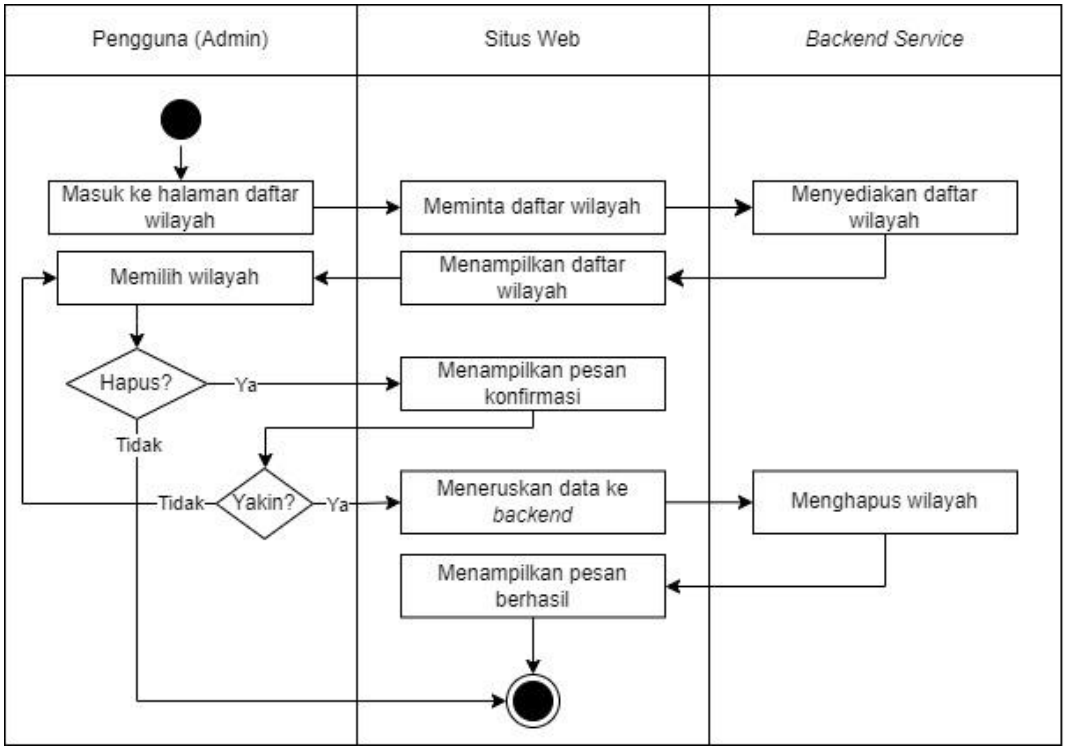
Gambar 3.25 *Activity Diagram* tambah dialek



Gambar 3.26 *Activity Diagram* hapus dialek



Gambar 3.27 *Activity Diagram* tambah wilayah



Gambar 3.28 *Activity Diagram* hapus wilayah

3.3.2 Perencanaan

Pemetaan tim dan tugas dilakukan pada tahap ini menggunakan data dari tahap analisis kebutuhan. Pada tahap ini, ditentukan metode pengembangan perangkat lunak yang digunakan, yaitu *agile development* dengan kerangka kerja *Scrum*. Hasil analisis kebutuhan yang telah dibuat dijadikan sebagai acuan dalam penyusunan *user stories* dan teknologi yang digunakan selama pengembangan. *User stories* yang telah disusun kemudian diatur ke dalam *backlog* yang dapat diambil pada tiap *sprint*.

Tabel 3.8 Komposisi tim

Nama	Peran
Mahendra Pratama, S.T., M.Eng.	*Product Owner*
Dede Kurniawan	*Frontend Web Developer, Scrum Master*
Nyoman Eka Swardita	*Backend Developer*
Ridho Ahmad Fauzi	*Frontend Mobile Developer*
Andre Gilang Firmansyah	*UI/UX Designer*

3.3.3 Product Backlog

Pada tahap ini, tim menyusun *product backlog* yang berisi fitur-fitur yang perlu dikerjakan, perbaikan, pengujian, dan pekerjaan lainnya yang perlu dikerjakan oleh tim. *Product backlog* dibuat pada tahap awal *Scrum* yang dikelola dan terus diperbarui selama proses pengembangan.

3.3.4 Sprint Planning

Rapat dilaksanakan bersama seluruh anggota tim pengembang untuk menentukan *backlog* yang akan diambil selama *sprint* berjalan. Pada tahap ini pula ditentukan tugas untuk setiap *backlog* yang diambil untuk setiap anggota tim. Alat yang digunakan dalam pengelolaan proses kerja *Scrum* pada penelitian ini adalah Taiga.io.

3.3.5 Sprint

Tahapan ini merupakan tahap utama dalam kerangka kerja *Scrum* di mana setiap anggota tim mengerjakan tugas yang telah diambil pada tahapan *sprint planning*. Selama periode *sprint*, *backlog* dan tugas terus disempurnakan dan dikembangkan jika dihadapkan dengan *bug* atau penambahan fitur serta dilakukan pengujian fungsional. *Sprint* berlangsung selama 10 hari kerja atau selama dua minggu mengikuti durasi *sprint* pada umumnya. Pemilihan durasi ini juga memberikan waktu yang lebih bagi anggota tim untuk mengerjakan tugas yang telah diambil dengan lebih leluasa sehingga meminimalkan tugas untuk diteruskan pada *sprint* berikutnya.

3.3.6 Daily Scrum

Tahapan ini dilaksanakan setiap harinya di mana setiap anggota tim melaporkan tugas yang telah dikerjakan pada hari sebelumnya, tugas yang akan dikerjakan pada hari berjalan, serta hambatan yang dihadapi pada hari sebelumnya. *Daily Scrum* dilaksanakan secara asinkron melalui grup khusus pada aplikasi Whatsapp. Tidak dipilihnya metode *daily Scrum* secara sinkron adalah dikarenakan sulitnya mendokumentasikan laporan tiap anggota tim karena sifatnya yang tatap muka.

3.3.7 Sprint Review

Sprint review dilaksanakan pada hari terakhir setiap *sprint*, di mana setiap anggota tim mempresentasikan hasil yang telah dicapai selama *sprint* kepada pemangku kepentingan untuk menerima umpan balik. Pada tahap ini, seluruh anggota tim mempresentasikan tugas yang telah selesai, belum selesai, dan kendala yang dihadapi. Bersama *product owner,* anggota tim menyusun *backlog* untuk diambil pada *sprint* berikutnya. Selain itu, pada *sprint review*, tim juga mengevaluasi *increment*, yaitu hasil kerja yang telah selesai dan memenuhi kriteria yang telah disepakati sebelumnya. *Definition of Done (DoD)* menjadi acuan dalam menentukan apakah suatu tugas atau fitur sudah benar-benar selesai.

3.3.8 Retrospective

Evaluasi terhadap kendala dan kesalahan pada *sprint* berjalan dilakukan pada tahapan ini agar pada *sprint* berikutnya dapat ditingkatkan kembali. *Retrospective* merupakan tahapan terakhir dalam metodologi *Scrum*. Tujuan dari *retrospective* adalah untuk merefleksikan proses kerja tim selama *sprint* yang baru saja selesai dan mencari cara untuk meningkatkan efektivitas dan efisiensi pada *sprint* berikutnya. *Sprint retrospective* dipandu oleh *Scrum* master dan diikuti oleh seluruh anggota tim *Scrum. Retrospective* harus menghasilkan poin-poin perbaikan yang dapat diterapkan pada *sprint* berikutnya. Ini dapat berupa perbaikan komunikasi, pengelolaan tugas, atau hal-hal teknis seperti alat atau metode pengembangan yang digunakan.

3.3.9 Pengujian

Penelitian ini menerapkan *blackbox testing* sebagai metode utama dalam proses pengembangan situs web. Pada setiap *sprint*, setiap *backlog* yang diambil dianalisis untuk dibuatkan kasus pengujian yang berfungsi sebagai panduan dalam pengembangan fitur. Pengujian dilakukan secara manual untuk memastikan fitur berfungsi sesuai dengan kebutuhan pengguna berdasarkan skenario pengujian yang telah disusun dari perspektif pengguna.

Selain pengujian manual, penelitian ini mencakup pengujian performa untuk memastikan bahwa situs web yang dikembangkan tidak hanya berfungsi dengan baik tetapi juga memiliki kinerja optimal. Pengujian performa dilakukan dengan memanfaatkan alat seperti Google Lighthouse, yang menyediakan evaluasi terhadap aspek-aspek seperti performa, aksesibilitas, *SEO*, dan praktik pengembangan web yang baik. Data dari Google Lighthouse digunakan untuk mengidentifikasi dan memperbaiki potensi masalah performa yang mungkin memengaruhi pengalaman pengguna.

Selain itu, *User Acceptance Testing (UAT)* juga diterapkan untuk memastikan situs web telah memenuhi kebutuhan pengguna secara menyeluruh. *UAT* dilakukan dengan melibatkan pengguna akhir yang menguji situs berdasarkan skenario dan kriteria penerimaan yang telah disepakati. Pendekatan ini bertujuan

untuk memastikan bahwa situs web tidak hanya memenuhi kebutuhan teknis tetapi juga sesuai dengan harapan dan kepuasan pengguna. Dengan kombinasi *blackbox testing*, pengujian performa, dan *UAT*, penelitian ini berupaya memastikan bahwa pengembangan situs web berfokus pada fungsionalitas, kualitas, dan pengalaman pengguna yang optimal.

V. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, kesimpulan yang diperolah adalah sebagai berikut:

1. Proses pengembangan situs web menggunakan metode *Agile Development* dengan kerangka kerja Scrum telah diselesaikan dalam 9 sprint terdiri dari 26 *user stories* yang terbagi ke dalam 5 *epic*.
2. Berdasarkan hasil pengujian *User Acceptance Test*, diperoleh bahwa situs web sudah memenuhi harapan pengguna dengan sangat baik. Peran peninjau memperoleh skor 100%, sedangkan untuk peran kontributor dan publik masing-masing memperoleh 91,85% dan 93,22% dengan nilai rata-rata 95,02%.
3. Berdasarkan hasil pengujian performa yang dilakukan, situs web yang dibangun dengan Qwik mendapatkan skor yang sangat baik apabila diakses melalui perangkat *desktop*. Namun, pengujian dengan perangkat *mobile* mendapatkan skor yang lebih kecil dari pada *desktop*, yang menunjukkan bahwa masih ada bagian dari situs web yang dapat dioptimalkan kembali.

5.2 Saran

Adapun saran yang dapat diberikan untuk penelitian selanjutnya adalah sebagai berikut:

1. Penelitian selanjutnya diharapkan dapat mengintegrasikan fitur pelafalan dan pembuatan terjemahan kalimat dengan memanfaatkan kecerdasan buatan untuk meningkatkan keakuratan dan interaktivitas sistem.

2. Diperlukan pengembangan fitur terjemahan multibahasa guna mengakomodasi bahasa asing lainnya, seperti bahasa Inggris, agar layanan lebih inklusif dan bermanfaat bagi pengguna internasional.
3. Diperlukan adanya fitur *auto spell* dan rekomendasi pencarian untuk memudahkan pengguna untuk mencari mendapatkan ejaan yang tepat ketika mengetikkan kata.
4. Pengujian situs web yang dilakukan menggunakan metode *blackbox testing* diharapkan dapat dikembangkan lebih lanjut dengan menerapkan pendekatan *Test Driven Development (TDD)*, sehingga pengembangan perangkat lunak menjadi lebih terarah dan memiliki cakupan pengujian yang lebih luas.

DAFTAR PUSTAKA

DAFTAR PUSTAKA

[1] E. Ambarita, "Ancaman Kepunahan Bahasa-Bahasa Daerah Di Era Globalisasi: Sebab-Musabab," no. May, hal. 1–19, 2019, doi: 10.13140/RG.2.2.22542.61761.

[2] Z. Abidin dan P. Permata, "Pengaruh Penambahan Korpus Paralel Pada Mesin Penerjemah Statistik Bahasa Indonesia Ke Bahasa Lampung Dialek Nyo," *J. Teknoinfo*, vol. 15, no. 1, hal. 13, 2021, doi: 10.33365/jti.v15i1.889.

[3] A. G. Pereira, T. M. Lima, dan F. Charrua-Santos, "Industry 4.0 and Society 5.0: Opportunities and Threats," *Int. J. Recent Technol. Eng.*, vol. 8, no. 5, hal. 3305–3308, 2020, doi: 10.35940/ijrte.d8764.018520.

[4] Afrianto, E. T. S. Sujatna, N. Darmayanti, F. Ariyani, dan J. Cooke-Plagwitz, "Clause and predicative constituents in an austronesian language: Lampung language," *Top. Linguist.*, vol. 21, no. 2, hal. 62–79, 2020, doi: 10.2478/topling-2020-0010.

[5] E. S. Purwani, O. Sukmana, dan V. Salviana, "The Threat of Extinction of Lampung Regional Language , Indonesia : A Phenomenological View," *Int. J. Res. Eng. Sci. Manag.*, vol. 7, no. 5, hal. 90–96, 2024.

[6] T. Pudjiastuti, *Aksara dan Naskah Kuno Lampung dalam Pandangan Masyarakat Lampung Kini*. Jakarta: Departemen Pendidikan dan Kebudayaan RI, 1997.

[7] O. A. Popoola, H. E. Adama, C. D. Okeke, dan A. E. Akinoso, "Conceptualizing Agile Development in Digital Transformations: Theoretical Foundations and Practical Applications," *Eng. Sci. Technol. J.*, vol. 5, no. 4, hal. 1524–1541, 2024, doi: 10.51594/estj/v5i4.1080.

[8] D. Ghimire dan S. Charters, "The Impact of Agile Development Practices on Project Outcomes," *Software*, vol. 1, no. 3, hal. 265–275, 2022, doi: 10.3390/software1030012.

[9] K. Schwaber dan J. Sutherland, "The Scrum Guide," Scrum. [Daring]. Tersedia pada: https://scrumguides.org/docs/scrumguide/v2020/2020-

Scrum-Guide-US.pdf

[10] E. P. Wonohardjo, R. F. Sunaryo, Y. Sudiono, Suharjito, dan N. Surantha, "A Systematic Review Of SCRUM In Software Development," *Int. J. Informatics Vis.*, vol. 3, no. 2, hal. 108–112, 2019, doi: 10.30630/joiv.2.4-2.171.

[11] A. Verma, A. Khatana, dan S. Chaudhary, "A Comparative Study of Black Box Testing and White Box Testing," *Int. J. Comput. Sci. Eng.*, vol. 5, no. 12, hal. 301–304, 2017, doi: 10.26438/ijcse/v5i12.301304.

[12] N. A. Nik Ahmad dan P. N. N. A. Megat Sazali, "Performing User Acceptance Test with System Usability Scale for Graduation Application," in *Proceedings - 2021 International Conference on Software Engineering and Computer Systems and 4th International Conference on Computational Science and Information Management, ICSECS-ICOCSIM 2021*, 2021, hal. 86–91. doi: 10.1109/ICSECS52883.2021.00023.

[13] D. Anjani, H. Hilaliyah, dan D. Novianti, "M-Absence : Analysis and Design using Unified Modelling Language (UML)," *J. Phys. Conf. Ser.*, vol. 1539, no. 1, hal. 0–7, 2020, doi: 10.1088/1742-6596/1539/1/012040.

[14] F. E. Febriansyah, A. Ardiansyah, dan A. Darmaji, "Cawa Lampung : Kamus Bahasa Indonesia-Lampung Dialek a Berbasis Android," *Klik - Kumpul. J. Ilmu Komput.*, vol. 7, no. 3, hal. 331, 2020, doi: 10.20527/klik.v7i3.352.

[15] M. F. Azima dan S. N. Laila, "Rancang Bangun Aplikasi Kamus Bahasa dan Aksara Lampung Dialek A dan Dialek O Berbasis Android," *J. Tek.*, vol. 14, no. x, hal. 1–9, 2020.

[16] T. Nurdianto, N. Wulandari, dan Firman, "Rancang Bangun Aplikasi 'Kmois' Kamus Bahasa IndonesiaMoiBerbasis Android," *JurnalPETISI(Pendidikan Teknol. Informasi)*, vol. 2, no. 1, hal. 1–9, 2021.

[17] A. Lipiński dan B. Pańczyk, "Performance optimization of web applications using Qwik," *J. Comput. Sci. Inst.*, vol. 28, no. June, hal. 197–203, 2023, doi: 10.35784/jcsi.3672.

[18] J. Tan, Y. Chen, dan S. Jiao, *Visual Studio Code in Introductory Computer Science Course: An Experience Report*, vol. 1, no. 1. Association for Computing Machinery, 2023. [Daring]. Tersedia pada: http://arxiv.org/abs/2303.10174

[19] A. Del Sole, *Visual Studio Code Distilled: Evolved Code Editing for Windows, macOS, and Linux, Second Edition*. 2021. doi: 10.1007/978-1-4842-6901-5.

[20] N. A. A. Khleel dan K. Nehéz, "Comparison of version control system tools," *Multidiszcip. Tudományok*, vol. 10, no. 3, hal. 61–69, 2020, doi: 10.35925/j.multi.2020.3.7.

[21] M. S. Arefeen dan M. Schiller, "Continuous Integration Using Gitlab," *Undergrad. Res. Nat. Clin. Sci. Technol. J.*, vol. 3, no. 1–11, hal. 6–11, 2019,

doi: 10.26685/urncst.152.

[22] M. F. Adiwisastra, A. B. Hikmah, dan A. Warnilah, *Dasar Pemrograman Web*, 2 ed. Grobogan, Jawa Tengah: CV. Sarnu Untung, 2019.

[23] J. Bogner dan M. Merkel, "To Type or Not to Type? A Systematic Comparison of the Software Quality of JavaScript and TypeScript Applications on GitHub," in *Proceedings - 2022 Mining Software Repositories Conference, MSR 2022*, Association for Computing Machinery, 2022, hal. 658–669. doi: 10.1145/3524842.3528454.

[24] K. Samson, "Bun: The Next Big Thing in Javascript," Dev.to. [Daring]. Tersedia pada: https://dev.to/kalashin1/bun-the-next-big-thing-in-javascript-jeg

[25] "Zig Programming Language." [Daring]. Tersedia pada: https://ziglang.org/

[26] F. Aghdasi, "What is Bun: A High-Performance JavaScript Runtime?," Code Crafters. [Daring]. Tersedia pada: https://medium.com/code-crafters/what-is-bun-a-high-performance-javascript-runtime-3aaff50aeef7

[27] M. Abbadini, D. Facchinetti, G. Oldani, M. Rossi, dan S. Paraboschi, "NatiSand: Native Code Sandboxing for JavaScript Runtimes," in *ACM International Conference Proceeding Series*, 2023, hal. 639–653. doi: 10.1145/3607199.3607233.

[28] B. Grant, "How Bun supports V8 APIs without using V8." [Daring]. Tersedia pada: https://bun.sh/blog/how-bun-supports-v8-apis-without-using-v8-part-1

[29] "Bun is a fast JavaScript all-in-one toolkit." [Daring]. Tersedia pada: https://bun.sh

[30] T. Heričko, B. Šumak, dan S. Brdnik, "Towards Representative Web Performance Measurements with Google Lighthouse," in *Proceedings of the 2021 7th Student Computer Science Research Conference (StuCoSReC)*, 2021, hal. 39–42. doi: 10.18690/978-961-286-516-0.9.

[31] "Pemberian skor performa Lighthouse," Chrome for Developers. [Daring]. Tersedia pada: https://developer.chrome.com/docs/lighthouse/performance/performance-scoring

[32] M. Hevery, "A first look at Qwik - the HTML first framework." [Daring]. Tersedia pada: https://dev.to/builderio/a-first-look-at-qwik-the-html-first-framework-af

[33] T. Lonka, "Improving the Initial Rendering Performance of React Applications Through Contemporary Rendering Approaches," 2023, [Daring]. Tersedia pada: www.aalto.fi

[34] J. Vepsäläinen, M. Hevery, dan P. Vuorimaa, "Resumability — A New Primitive for Developing Web Applications," *IEEE Access*, vol. 12, no. December 2023, 2024.

[35] M. Hevery, “Understanding Resumability from the Ground Up.” [Daring]. Tersedia pada: https://www.builder.io/blog/resumability-from-ground-up

[36] M. Hevery, “Don’t blame the developer for what the frameworks did!” [Daring]. Tersedia pada: https://www.builder.io/blog/dont-blame-the-developer-for-what-the-frameworks-did

[37] G. Bondel, A. Landgraf, dan F. Matthes, “API Management Patterns for Public, Partner, and Group Web API Initiatives with a Focus on Collaboration,” in *ACM International Conference Proceeding Series*, 2021. doi: 10.1145/3489449.3490012.

[38] “Open Source and Lean principles gave birth to Taiga.” [Daring]. Tersedia pada: https://taiga.io/about-us/

[39] M. A. Muhammad dan Martinus, “Mobile Dictionary Aksara Lampung Berbasis Teknologi SPA (Single Webpage Application),” *Electr. –Jurnal Rekayasa dan Teknol. Elektro*, vol. 11, no. 2, 2017, [Daring]. Tersedia pada: https://electrician.unila.ac.id/index.php/ojs/article/view/2025/pdf

[40] Virupaksha, “Splitting Word into Syllables in Javascript.” [Daring]. Tersedia pada: https://stackoverflow.com/questions/49403285/splitting-word-into-syllables-in-javascript