

**RANCANG BANGUN APLIKASI PENGARTI KATA-KATA
KASAR BERBASIS WEBSITE**



PUBLIKASI ILMIAH

**Disusun sebagai salah satu syarat menyelesaikan Program Studi Strata I pada
Program Studi Teknik Informatika Fakultas Komunikasi dan Informatika**

**Oleh:
ABDUL AZIZ
L200200180**

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS KOMUNIKASI DAN INFORMATIKA
UNIVERSITAS MUHAMMADIYAH SURAKARTA**

2024

HALAMAN PERSETUJUAN

**RANCANG BANGUN APLIKASI PENGARTI KATA-KATA
KASAR BERBASIS WEBSITE**

PUBLIKASI ILMIAH

oleh:

ABDUL AZIZ
L200200180

Telah diperiksa dan disetujui untuk diuji oleh:
Dosen Pembimbing



Endang Wahyu Pamungkas, S.Kom., M.Kom., Ph.D.

NIDN. 0624039201

HALAMAN PENGESAHAN

**RANCANG BANGUN APLIKASI PENGARTI KATA-KATA
KASAR BERBASIS WEBSITE**

OLEH

ABDUL AZIZ

L200200180

**Telah dipertahankan di depan Dewan Penguji
Fakultas Komunikasi dan Informatika
Universitas Muhammadiyah Surakarta
Pada hari Selasa, 09 Juli 2024
dan dinyatakan telah memenuhi syarat**

Dewan Penguji:

1. Endang Wahyu Pamungkas, S.Kom., M.Kom., Ph.D.

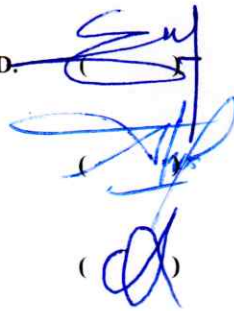
(Ketua Dewan Penguji)

2. Dr.Eng. Yusuf Sulistyono Nugroho, S.T., M.Eng.

(Anggota I Dewan Penguji)

3. Helmi Imaduddin, S.Kom., M.Eng.

(Anggota II Dewan Penguji)

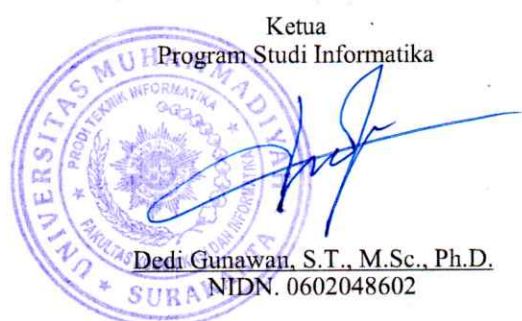


Dekan
Fakultas Komunikasi dan Informatika



Nughyatna, S.T., M.Sc., Ph.D.
NIDN. 0612076901

Ketua
Program Studi Informatika



Dedi Gunawan, S.T., M.Sc., Ph.D.
NIDN. 0602048602

PERNYATAAN

Dengan ini saya menyatakan bahwa dalam skripsi ini tidak terdapat karya yang pernah diajukan untuk memperoleh gelar kesarjanaan di suatu perguruan tinggi dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan orang lain, kecuali secara tertulis diacu dalam naskah dan disebutkan dalam daftar pustaka.

Apabila kelak terbukti ada ketidakbenaran dalam pernyataan saya di atas, maka akan saya pertanggungjawabkan sepenuhnya.

Surakarta,
27 Oktober 2023



ABDUL AZIZ
L200200180

SURAT KETERANGAN LULUS PLAGIASI

Assalamu'alaikum Wr. Wb

Biro Skripsi Program Studi Informatika menerangkan bahwa :

Nama : Abdul Aziz
NIM : **L200200180**
Judul : **RANCANG BANGUN APLIKASI PENGARTI KATA-KATA
KASAR BERBASIS WEBSITE**
Program Studi : Informatika
Status : **Lulus**

Adalah benar-benar sudah lulus pengecekan plagiasi dari Naskah Publikasi Skripsi, dengan menggunakan aplikasi Turnitin.

Demikian surat keterangan ini dibuat agar dipergunakan sebagaimana mestinya.

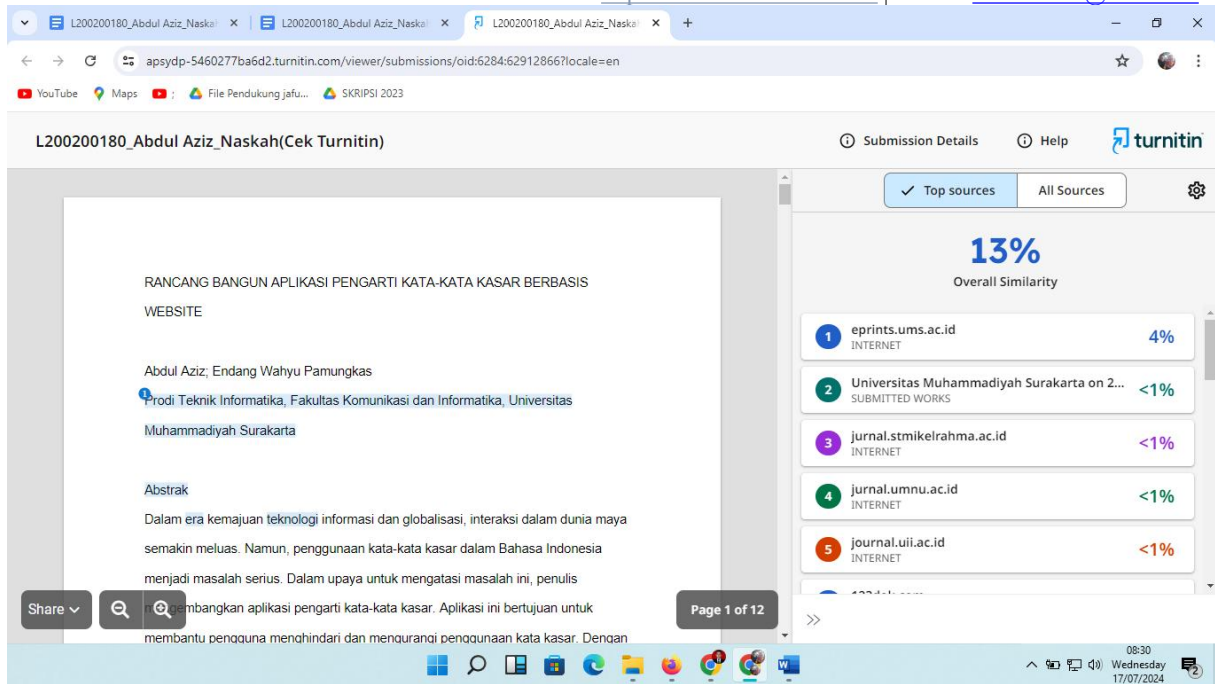
Wassalamu'alaikum Wr. Wb

Surakarta, 16 Juli 2024

Biro Skripsi Informatika



Ihsan Cahyo Utomo, S.Kom., M.Kom.



The screenshot shows a Turnitin submission interface. The document title is "RANCANG BANGUN APLIKASI PENGARTI KATA-KATA KASAR BERBASIS WEBSITE" by Abdul Aziz, Endang Wahyu Pamungkas. The document is from the Prodi Teknik Informatika, Fakultas Komunikasi dan Informatika, Universitas Muhammadiyah Surakarta. The abstract discusses the development of an application to identify and warn against the use of vulgar language in Indonesian. The Turnitin report shows an overall similarity of 13%.

Source	Similarity
1 eprints.ums.ac.id INTERNET	4%
2 Universitas Muhammadiyah Surakarta on 2... SUBMITTED WORKS	<1%
3 jurnal.stmikelahma.ac.id INTERNET	<1%
4 jurnal.umnu.ac.id INTERNET	<1%
5 journal.uui.ac.id INTERNET	<1%

RANCANG BANGUN APLIKASI PENGARTI KATA-KATA KASAR BERBASIS WEBSITE

Abdul Aziz; Endang Wahyu Pamungkas

Prodi Teknik Informatika, Fakultas Komunikasi dan Informatika, Universitas Muhammadiyah Surakarta

Abstrak

Dalam era kemajuan teknologi informasi dan globalisasi, interaksi dalam dunia maya semakin meluas. Namun, penggunaan kata-kata kasar dalam Bahasa Indonesia menjadi masalah serius. Dalam upaya untuk mengatasi masalah ini, penulis mengembangkan aplikasi pengarti kata-kata kasar. Aplikasi ini bertujuan untuk membantu pengguna menghindari dan mengurangi penggunaan kata kasar. Dengan fitur-fitur seperti mengartikan kata kasar, memberikan versi halus, mendeteksi bahasa, dan mengubah kalimat kasar menjadi lebih sopan. Metode pengembangan yang digunakan adalah metode *waterfall*, yang melibatkan tahapan analisis kebutuhan, desain, pengkodean, pengujian, dan pemeliharaan. Aplikasi ini difokuskan pada Bahasa Indonesia dan bahasa daerah umum, dengan batasan linguistik. Pengembangan aplikasi ini diharapkan dapat meningkatkan kesadaran pengguna terhadap penggunaan kata-kata kasar dan mendorong komunikasi yang lebih sehat di dunia maya. Hasil dan pembahasan menunjukkan bahwa penulis telah berhasil membuat sebuah aplikasi pengarti kata-kata kasar berbasis *website*. Aplikasi ini dirancang menggunakan NextJs pada sisi *frontend* dan menggunakan Strapi untuk sisi *backend* yang terhubung dengan *database* PostgreSQL, serta memanfaatkan *Large Language Models* (LLM) dengan *framework* Langchain untuk memproses kata-kata kasar. Pengujian *System Usability Scale* (SUS) dilakukan untuk mengukur tingkat keberhasilan dan kegunaan aplikasi ini, dengan hasil skor 83.75 yang termasuk dalam kategori *Excellent* dan *Acceptable*. Dengan demikian, dapat disimpulkan bahwa aplikasi ini telah sesuai dengan kebutuhan dalam mengartikan dan menghaluskan kata-kata kasar, serta berfungsi dengan baik dan optimal.

Kata Kunci : aplikasi, kata-kata kasar, Bahasa Indonesia, komunikasi, pengguna, pengarti kata-kata kasar.

Abstract

In the era of information technology advancement and globalization, interactions in the online world have become widespread. However, the use of offensive language in the Indonesian language has become a serious issue. In an effort to address this problem, the author has developed an application for detecting and translating offensive language. This application aims to assist users in avoiding and reducing the use of offensive language. With features such as translating offensive words, providing polite alternatives, detecting languages, and transforming offensive sentences into more polite ones. The development method used is the waterfall model, involving stages of requirement analysis, design, coding, testing, and maintenance. This application is focused on Indonesian and commonly used regional languages, with linguistic limitations. The development of this application is expected to increase user awareness of the use of offensive language and promote healthier communication in the online world. The results and discussion indicate

that the author has successfully created a web-based application for detecting and translating offensive language. The application is designed using NextJs for the frontend and Strapi for the backend connected to a PostgreSQL database, and utilizes Large Language Models (LLM) with the Langchain framework to process offensive language. Usability testing using the System Usability Scale (SUS) was conducted to measure the success and usability of this application, achieving a score of 83.75, categorized as Excellent and Acceptable. Thus, it can be concluded that this application meets the needs for detecting and softening offensive language, and functions well and optimally.

Keywords : application, offensive language, Indonesian language, communication, users, offensive language detection.

1. PENDAHULUAN

Dalam era kemajuan teknologi informasi dan globalisasi seperti saat ini, orang-orang dari berbagai latar belakang yang berbeda sering kali berinteraksi di dunia maya, baik dalam konteks bisnis, sosial, politik, pendidikan, dan lainnya (Suci Rahayu Rais, 2018). Ditambah dengan adanya berbagai platform media sosial seperti Instagram, Tiktok, Twitter, Facebook, dan sebagainya (Chiril et al., 2021), orang-orang bisa bebas mengekspresikan diri lewat platform tersebut, namun salah satu hambatan yang muncul adalah penggunaan kata-kata kasar atau tidak senonoh dalam Bahasa yang digunakan terutama Bahasa Indonesia (Hidayatullah et al., 2019).

Dengan digunakannya kata-kata kasar dalam ujaran di sosial media, hal itu dapat menyinggung perasaan, lalu memicu konflik antar individu maupun kelompok hingga menimbulkan perpecahan (Pamungkas et al., 2021). Dalam hal ini ada beberapa orang yang memang sengaja menggunakan ujaran dengan kata kasar, ada juga yang memang tidak tahu kalau kata yang mereka gunakan mengandung arti atau makna yang buruk (Dewa & Wijana, 2008). Oleh karena itu ada kebutuhan dalam mengembangkan aplikasi pengarti kata-kata kasar yang dapat membantu dalam mencegah masalah yang timbul dalam berkomunikasi.

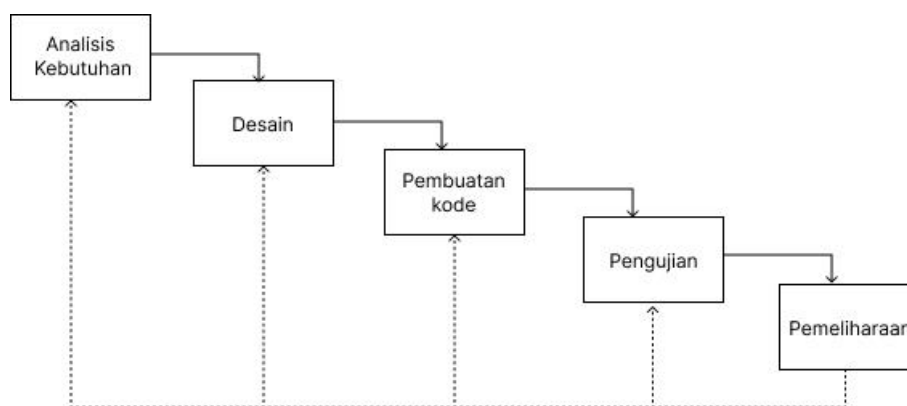
Pentingnya akan isu ini, pembuatan aplikasi pengarti kata kasar dapat menjadi solusi yang bermanfaat dan relevan. Penulis dalam membuat aplikasi ini terinspirasi dari *website* Bernama Noswearing (*Noswearing*, 2023) dimana pada *website* itu kita dapat mencari kata-kata kasar dan terjemahannya dalam versi Bahasa Inggris, dikarenakan belum ada yang membuat dalam versi Bahasa Indonesia penulis ingin mengembakan aplikasi yang serupa tapi dengan Bahasa Indonesia. Tujuan dari dikembangkannya aplikasi pengarti kata-kata kasar ini adalah untuk membantu pengguna menghindari dan mengurangi penggunaan kata kasar, dengan fitur seperti mengartikan kata kasar, memberi versi halus dari kata tersebut, mendeteksi asal bahasanya, dan dapat mengubah kalimat

kasar menjadi lebih halus, dengan mengambil data kata-kata kasar dari database yang terhubung dengan API, lalu pengguna akan disajikan sebuah website dengan tampilan yang *user-friendly* dan *responsive*. User dapat langsung mencari kata kasar yang dengan mengetik pada kolom pencarian tanpa harus melakukan *login* terlebih dahulu. Selanjutnya untuk mengukur tingkat efektivitas dan kesuksesan aplikasi penulis melakukan pengujian dengan metode *System Usability Scale*.

Diharapkan dengan perkembangan teknologi saat ini dan pendekatan yang cermat, pengembangan aplikasi ini dapat bermanfaat bagi banyak pengguna seperti membantu mencari kata-kata kasar yang masih asing, sehingga pengguna bisa lebih bijak dalam memilih kata agar tercipta komunikasi yang sehat dan menciptakan dunia maya yang lebih baik (Pamungkas et al., 2022). Dengan tujuan dan manfaatnya aplikasi ini memiliki beberapa batasan seperti batasan linguistik, karena aplikasi ini difokuskan dalam Bahasa Indonesia dan Bahasa Daerah Jawa dan Sunda, lalu pengartian kata kasar juga masih terbatas data yang ada dalam *database*.

2. METODE

Pengembangan aplikasi pengarti kata kasar ini menggunakan metode *waterfall*. Metode ini dikenal sebagai *waterfall* karena setiap tahap harus diselesaikan secara berurutan dan *linear*, sebelum melanjutkan ke tahap selanjutnya (Abdul Wahid, 2020). Metode ini memiliki beberapa tahapan seperti yang ada pada Gambar 1.



Gambar 1. Tahapan Metode *Waterfall*

2.1 Analisis Kebutuhan

Pada tahap awal ini fokus utamanya adalah memahami persyaratan spesifikasi dari perangkat lunak. Analisis diperlukan untuk mengumpulkan semua kebutuhan pengguna, yang akan menjadi landasan bagi proses pengembangan aplikasi (Shylesh S, 2017).

- a. Kebutuhan administrator : Melakukan login ke dashboard dan mengolah data kata-kata kasar.

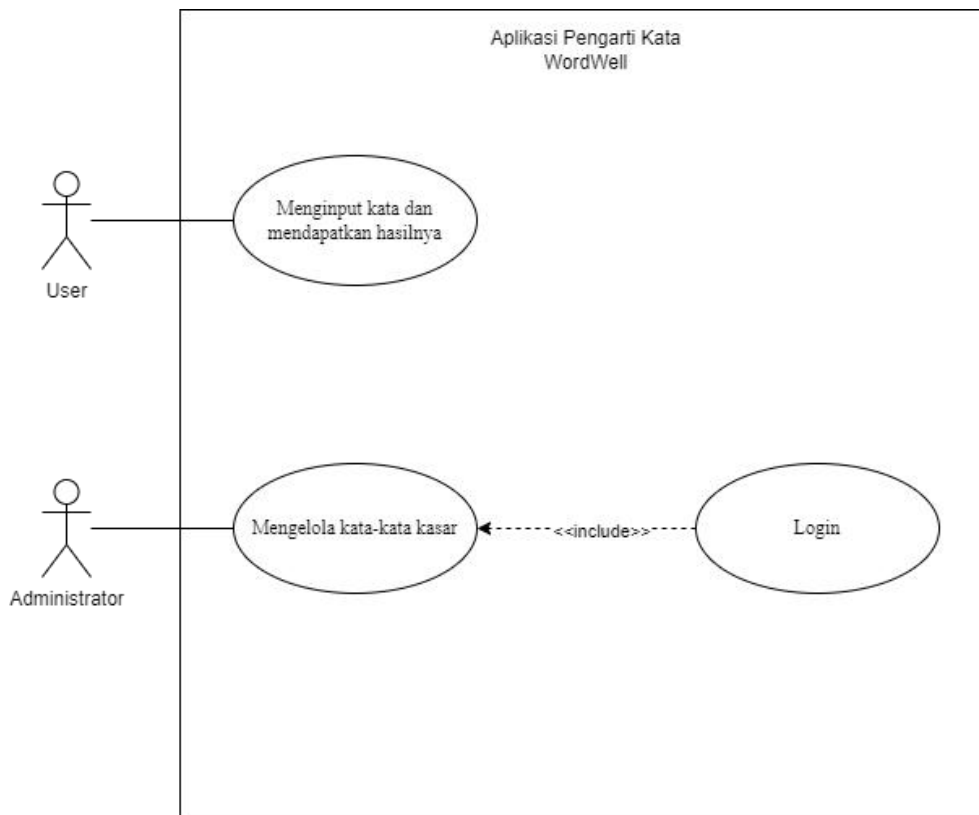
- b. Kebutuhan pengguna : Mencari arti kata kasar, melihat versi halus dari kata kasar tersebut, mendeteksi Bahasa yang digunakan, mengubah kalimat kasar menjadi lebih halus.

2.2 Desain

Setelah persyaratan kebutuhan aplikasi dikumpulkan, selanjutnya adalah tahap desain dimana mulai merancang arsitektur sistem yang akan dibuat (Bassil, 2012), tahap ini terdiri dari pembuatan *UseCase Diagram*, *Activity Diagram*, *Relational Database Model*, dan *User Interface*.

2.2.1 UseCase Diagram

UseCase Diagram digunakan untuk menggambarkan perilaku pengguna atau *actor* dalam berinteraksi dengan sistem (Gemino & Parker, 2009). Pada sistem ini terdapat dua *actor* yaitu administrator dan *user* seperti yang digambarkan pada Gambar 2.

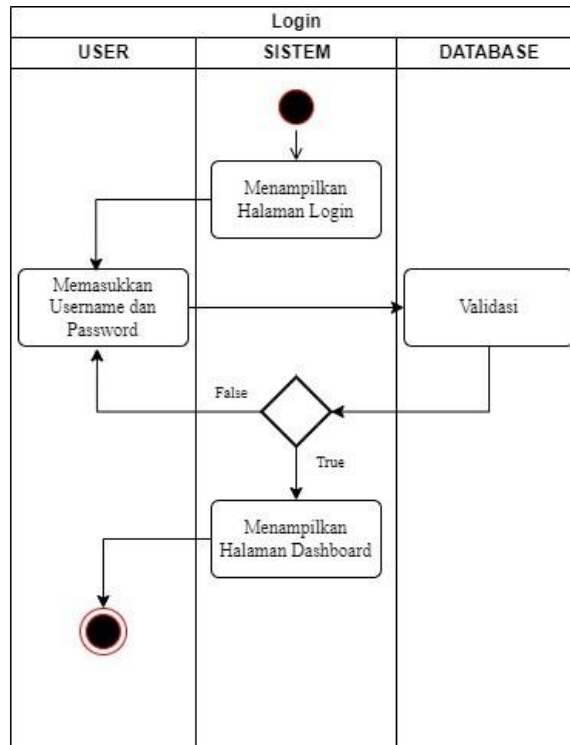


Gambar 2. *UseCase Diagram*

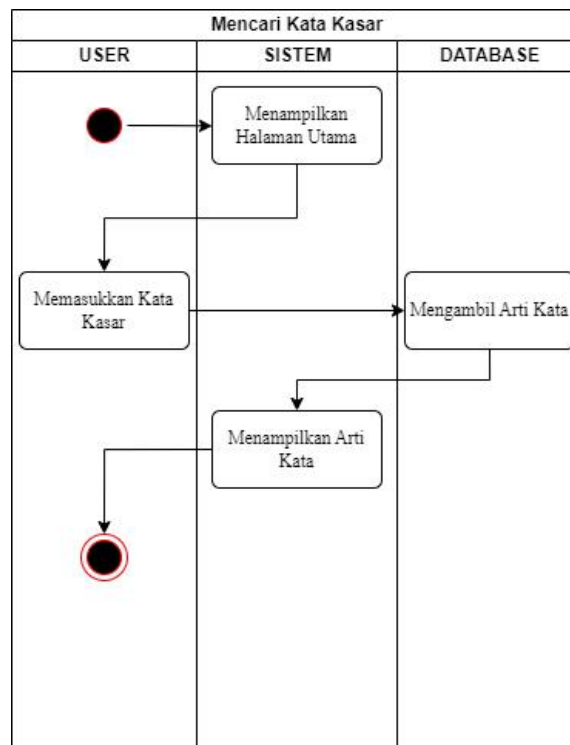
2.2.2 Activity Diagram

Diagram yang digunakan untuk menggambarkan urutan aliran kerja atau aktivitas seperti keputusan, tugas dalam suatu proses (Bastos et al., 2002). Pada gambar 3 adalah *activity diagram* dari sistem *login* dimana *admin* perlu memasukkan *username* dan *password* kemudian validasi, lalu gambar 4 adalah alur ketika pengguna memasukkan kata-kata

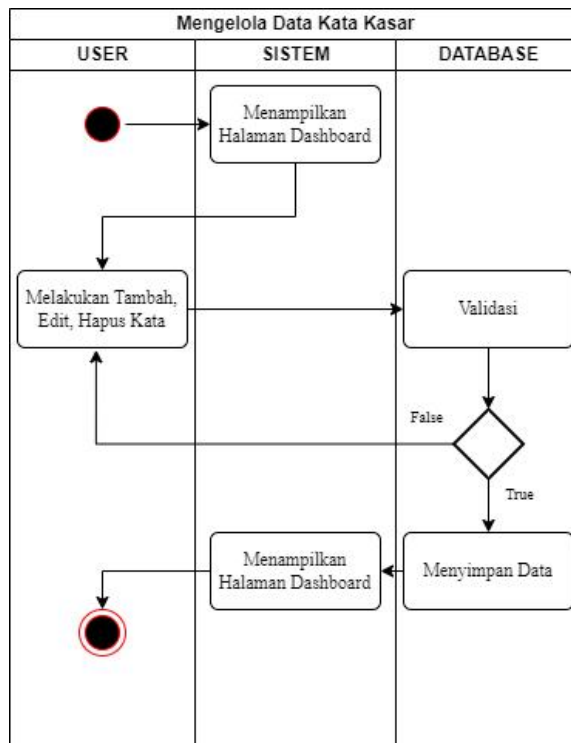
kasar untuk memperoleh hasilnya, dan pada gambar 5 adalah ketika *admin* mengelola data kata-kata kasar.



Gambar 3. Activity Diagram Login Admin



Gambar 4. Activity Diagram Mencari Kata Kasar



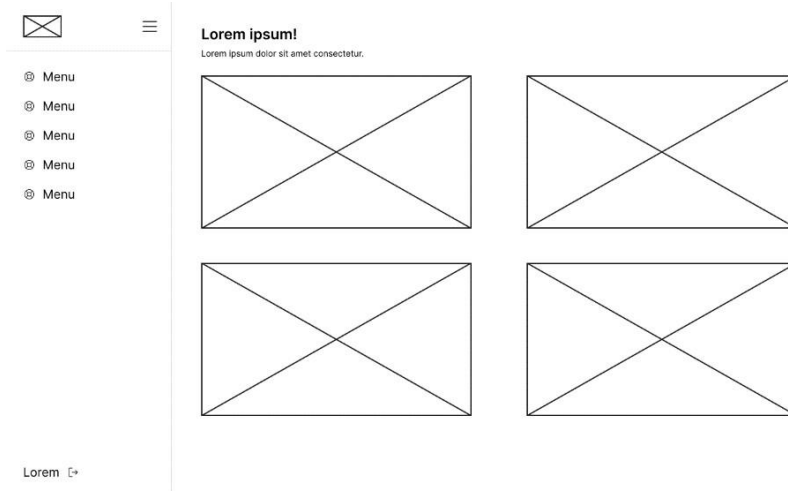
Gambar 5. Activity Diagram Mengelola Kata Kasar

2.2.3 Relational Database Model

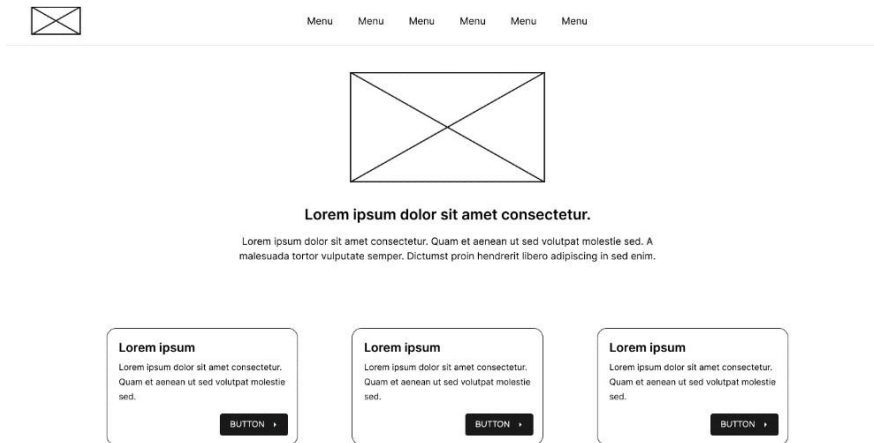
Untuk menyimpan dan mengorganisir data pada sistem ini dibutuhkan sebuah database. Dengan *database relational*, merancang dan mengelola *database* menjadi lebih terstruktur dengan menggunakan konsep relasi atau tabel (Jatana et al., 2012).

2.2.4 User Interface

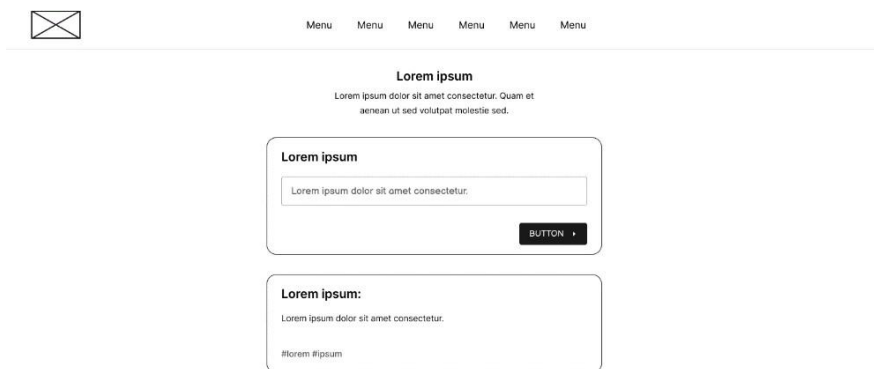
Dengan *user interface* maka pengguna akan lebih mudah dalam berinteraksi dengan sistem. Tujuannya adalah menciptakan antarmuka yang efisien dan menarik bagi pengguna (Sridevi, 2014). Pada tahap ini dimulai dengan merancang *design low-fidelity*.



Gambar 6. Design Low-Fidelity Dashboard



Gambar 7. Design Low-Fidelity Homepage



Gambar 8. Design Low-Fidelity Menu Cari Kata

2.2.5 Coding

Selanjutnya adalah memasuki proses pengembangan, dimana semua rancangan diatas akan diimplementasikan ke dalam kode. Sistem ini akan dibuat menggunakan *framework* dari React yaitu NextJS, dengan menggunakan NextJs membuat aplikasi *web full-stack* akan lebih mudah dan cepat dengan fitur seperti *client and server rendering*, *react server component*, dan juga memiliki *build-in optimization* (Next.Js, 2023), dan untuk memproses kata disini penulis menggunakan *Large Language Models* (LLM) (Zhang et al., 2023), yaitu Ollama dan OpenAI untuk menghasilkan *respons* dan *embeddings* dengan memanfaatkan *framework* Langchain, lalu juga menggunakan Brave sebagai penyedia layanan pencarian (Mishra & Chatterjee, 2024).

2.2.6 Testing

Setelah proses pengembangan selanjutnya harus dilakukan sebuah testing untuk mengetahui apakah aplikasi yang dibuat sudah sesuai dan apakah ada kesalahan yang perlu diperbaiki. Pada tahap ini penulis menggunakan metode pengujian *System Usability Scale* (SUS) (Lewis, 2018).

2.2.7 Maintenance

Setelah sistem selesai dikembangkan, diperlukan adanya pemeliharaan agar menjaga kinerja, kualitas, serta fungsionalitas aplikasi sepanjang siklusnya, pemeliharaan ini bermanfaat agar aplikasi tetap beradaptasi sesuai kebutuhan pengguna, dan terhindar dari masalah keamanan (Gupta et al., 2015).

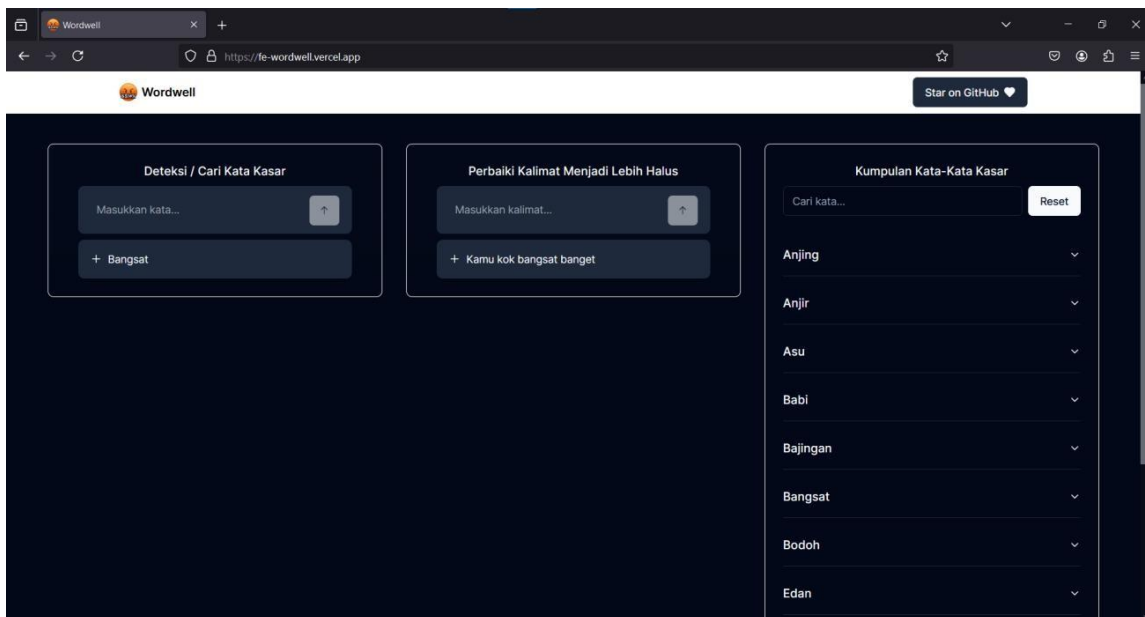
3. HASIL DAN PEMBAHASAN

3.1 Hasil dan Pembahasan Sistem

Penulis telah berhasil membuat sebuah aplikasi pengarti kata-kata kasar berbasis *website*. Dari penelitian yang sudah dipelajari maka terdapat hasil dan pembahasan yang komprehensif. Aplikasi ini dirancang menggunakan NextJs pada sisi *frontend* dan menggunakan Strapi untuk sisi *backend* yang terhubung dengan *database* Postgresql, lalu juga memanfaatkan *Large Language Models* (LLM) dengan *framework* Langchain untuk memproses kata-kata kasar.

3.1.1 Halaman Home

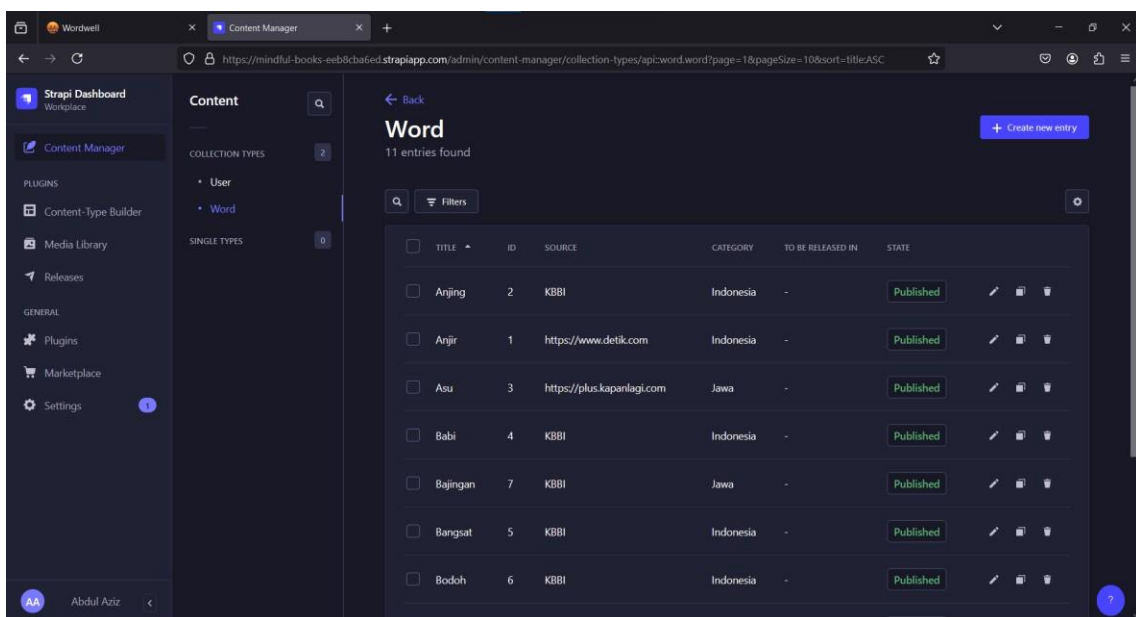
Pada halaman ini terdapat tiga buah *card* yang masing-masing memiliki fiturnya sendiri. *Card* pertama digunakan untuk mendeteksi dan mengartikan kata, *card* kedua digunakan untuk mencari versi halus dari kalimat kasar, *card* ketiga digunakan untuk menampilkan daftar kata kasar. Tampilan dari Halaman *Home* dapat dilihat pada gambar berikut.



Gambar 9. Halaman *Home*

3.1.2 Halaman Strapi

Pada halaman ini dilakukan pengelolaan konten kata-kata kasar pada sisi *backend* lalu data akan dikonsumsi oleh *frontend* dan ditampilkan pada Halaman *Home*.



Gambar 10. Halaman Strapi

3.2 Pengujian

Pada aplikasi ini dilakukan pengujian *System Usability Scale (SUS)* untuk melihat apakah aplikasi sudah berjalan dengan baik dan optimal. Metode *System Usability Scale* dipilih karena untuk mengukur tingkat keberhasilan dan kegunaan aplikasi ini.

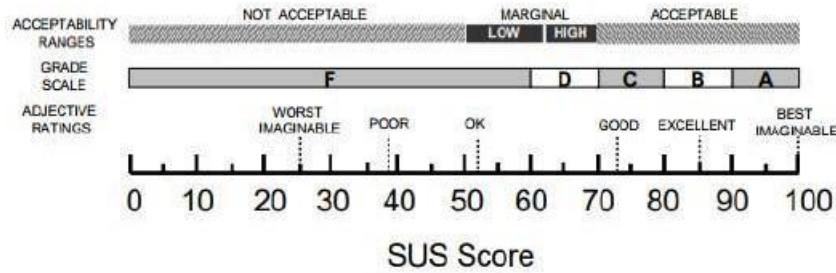
3.2.1 *System Usability Scale (SUS)*

Metode ini dipilih untuk mengukur sebuah aplikasi sudah berjalan dengan baik dan apakah masih ada kesalahan yang perlu diperbaiki. Cara kerja metode ini adalah pengguna diminta mencoba aplikasi lalu mengisi sepuluh pernyataan dengan lima jawaban mulai dari “Sangat tidak setuju” hingga “Sangat setuju”. Jawaban berdasarkan tingkat kepuasan pengguna terhadap aplikasi. Pernyataan dalam metode ini dapat dilihat pada gambar 11.

No	Pernyataan
1.	Saya Berpikir akan menggunakan sistem ini lagi
2.	Saya merasa sistem ini rumit untuk digunakan.
3.	Saya merasa sistem ini mudah untuk digunakan.
4.	Saya membutuhkan bantuan dari orang lain atau taknisi untuk menggunakan sistem ini.
5.	Saya merasa fitur fitur sistem ini berjalan dengan semestinya.
6.	Saya merasa ada banyak hal yang tidak konsisten pada sistem ini.
7.	Saya merasa orang lain akan memahami cara menggunakan sistem ini dengan cepat.
8.	Saya merasa sistem ini membingungkan
9.	Saya merasa tidak ada hambatan dalam menggunakan sistem ini.
10.	Saya perlu membiasakan diri terlebih dahulu sebelum menggunakan sistem ini.

Gambar 11. Pernyataan dalam *System Usability Scale*

Setiap pertanyaan diatas mencangkup aspek negatif dan positif dari sistem yang sudah digunakan oleh pengguna, cara menghitung skornya adalah, untuk pernyataan nomor ganjil kurangi 1 dari nilai skala, lalu pernyataan nomor genap kurangi nilai skala dari 5, selanjutnya jumlahkan keduanya dan kalikan jumlah total dengan 2.5 maka akan didapatkan skor yang berkisar antara 0 hingga 100. Semakin tinggi skor menunjukkan tingkat kegunaan yang baik, jika skor dibawah 68 maka terdapat masalah kegunaan pada sistem dan masih perlu diperbaiki lagi. Skor SUS ditunjukkan pada gambar 12.



Gambar 12. Skala Hasil Score SUS

Walaupun menggunakan sample yang sedikit, SUS dinilai menghasilkan perhitungan yang valid. Disini penulis melakukan pengujian pada 20 orang. Pada penelitian ini aplikasi yang sudah penulis buat mendapatkan skor 83.75. Angka tersebut termasuk pada kategori *Excellent*, sehingga aplikasi ini termasuk dalam tingkat *Acceptable*.

4. PENUTUP

Berdasarkan penelitian yang sudah penulis lakukan pada perancangan aplikasi pengarti kata kasar berbasis *website*. Dalam proses pengembangan menggunakan beberapa teknologi seperti *framework* NextJS untuk *frontend* dan Strapi untuk *Backend*, lalu juga menggunakan *Large Language Models* (LLM) yaitu Ollama dan OpenAI untuk menghasilkan *respons* dan *embeddings* dengan memanfaatkan *framework* Langchain, lalu juga menggunakan Brave sebagai penyedia layanan pencarian, dan juga *Usability Test* yang telah dilakukan dengan metode *System Usability Scale* mendapatkan skor 83.75. Sehingga dapat disimpulkan jika aplikasi telah sesuai dengan kebutuhan dalam mengartikan dan menghaluskan kata-kata kasar.

DAFTAR PUSTAKA

- Abdul Wahid, A. (2020). *Analisis Metode Waterfall Untuk Pengembangan Sistem Informasi*. <https://www.researchgate.net/publication/346397070>
- Bassil, Y. (2012). A Simulation Model for the Waterfall Software Development Life Cycle. In *International Journal of Engineering & Technology (iJET)* (Vol. 2, Issue 5). http://iet-journals.org/archive/2012/may_vol_2_no_5/255895133318216.pdf
- Bastos, R. M., Dubugras, D., & Ruiz, A. (2002). *Extending UML Activity Diagram for Workflow Modeling in Production Systems*.
- Chiril, P., Pamungkas, E. W., Benamara, F., Moriceau, V., & Patti, V. (2021). Emotionally Informed Hate Speech Detection: a multi-target perspective. *Cognitive Computation*, *14*(1), 322–352. <https://doi.org/10.1007/s12559-021-09862-5>
- Dewa, I., & Wijana, P. (2008). KATA-KATA KASAR DALAM BAHASA JAWA. In *Universitas Gadjah Mada* (Vol. 20, Issue 3).
- Gemino, A., & Parker, D. (2009). Use Case Diagrams in Support of Use Case Modeling: Deriving Understanding from the Picture. In *Journal of Database Management* (Vol. 20, Issue 1). <http://www.igi-global.com>
- Gupta, A., Xavier, S., & Sharma, S. (2015). *Software Maintenance: Challenges and Issues*.
- Hidayatullah, A. F., Aulia, A., Yusuf, F., Juwairi, K. P., Abida, R., & Nayoan, N. (2019). Identifikasi Konten Kasar pada Tweet Bahasa Indonesia. In *JLK* (Vol. 2, Issue 1). <https://t.co/YQCC0CM4gG>
- Jatana, N., Puri, S., Ahuja, M., Kathuria, I., & Gosain, D. (2012). *A Survey and Comparison of Relational and Non-Relational Database*. www.ijert.org
- Lewis, J. R. (2018). The System Usability Scale: Past, Present, and Future. *International Journal of Human-Computer Interaction*, *34*(7), 577–590. <https://doi.org/10.1080/10447318.2018.1455307>
- Mishra, S., & Chatterjee, P. (2023). *Exploring ChatGPT for Toxicity Detection in GitHub*. <http://arxiv.org/abs/2312.13105>
- Next.js*. (2023). <https://nextjs.org/>
- Noswearing*. (2023). <https://www.noswearing.com/>
- Pamungkas, E. W., Basile, V., & Patti, V. (2021). Towards multidomain and multilingual

- abusive language detection: a survey. *Personal and Ubiquitous Computing*, 27(1), 17–43. <https://doi.org/10.1007/s00779-021-01609-1>
- Pamungkas, E. W., Basile, V., & Patti, V. (2022). Investigating the role of swear words in abusive language detection tasks. *Language Resources and Evaluation*, 57(1), 155–188. <https://doi.org/10.1007/s10579-022-09582-8>
- Shylesh S. (2017). *A Study of Software Development Life Cycle Process Models*. <https://ssrn.com/abstract=2988291>
- Sridevi, S. (2014). USER INTERFACE DESIGN. In *International Journal of Computer Science and Information Technology Research* (Vol. 2). www.researchpublish.com
- Suci Rahayu Rais, N. (2018). *KEMAJUAN TEKNOLOGI INFORMASI BERDAMPAK PADA GENERALISASI UNSUR SOSIAL BUDAYA BAGI GENERASI MILENIAL*.
- Zhang, W., Deng, Y., Liu, B., Pan, S. J., & Bing, L. (2023). *Sentiment Analysis in the Era of Large Language Models: A Reality Check*.