LAPORAN PROYEK AKHIR

SISTEM APLIKASI MOBILE BERBASIS AI UNTUK PENGECEKAN DAN PEMBERIAN REKOMENDASI JUDUL TUGAS AKHIR MAHASISWA BERDASARKAN BIDANG DAN TINGKAT KESULITAN

ACC Sempro
9 Maret 2025

ACC Sempro
7 Maret 2025

ZAQAUL FIKRI AZIZ
2111082049

PROGRAM STUDI TEKNOLOGI REKAYASA PERANGKAT LUNAK

JURUSAN TEKNOLOGI INFORMASI

POLITEKNIK NEGERI PADANG

2024

ABSTRAK

Proses penentuan judul tugas akhir sering menjadi kendala bagi mahasiswa karena keterbatasan referensi, keaslian ide, serta kecocokan tingkat kesulitan dengan kemampuan mereka. Untuk mengatasi permasalahan ini, dikembangkan sebuah sistem berbasis AI dalam bentuk aplikasi *mobile* yang dirancang khusus untuk mahasiswa jurusan Teknologi Informasi di Politeknik Negeri Padang. Sistem ini menawarkan dua fitur utama, yaitu rekomendasi judul berbasis bidang dan tingkat kesulitan menggunakan model *Google Gemini* serta pengecekan kesamaan ide melalui *fuzzy matching* dengan *threshold* 70%. Metode yang digunakan mencakup algoritma *fuzzy matching* untuk mendeteksi kesamaan judul, model *Google Gemini* untuk menghasilkan rekomendasi, dan *decision tree* untuk klasifikasi tingkat kesulitan. Pengembangan sistem dilakukan dengan pendekatan *Software Development Life Cycle* (SDLC) model *waterfall* yang mencakup analisis kebutuhan, perancangan arsitektur sistem berbasis *microservices*, implementasi, dan pengujian. Data yang dikumpulkan dari mahasiswa akan disimpan dalam basis data terintegrasi sebagai referensi di masa depan. Sistem ini diharapkan dapat meningkatkan efisiensi proses penentuan judul tugas akhir, mengurangi risiko plagiasi, serta membantu dosen dalam proses bimbingan, sehingga mendukung mahasiswa dalam menemukan ide penelitian yang sesuai dan berkontribusi pada peningkatan kualitas penelitian di lingkungan akademik.

Kata Kunci: Tugas Akhir, *Artificial Intelligence*, *Google Gemini*, *Fuzzy Matching*, *Decision Tree*, Aplikasi *Mobile*.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1. Latar Belakang

Tugas akhir merupakan suatu karya ilmiah berdasarkan suatu kegiatan penelitian mandiri yang dilakukan oleh mahasiswa tingkat akhir. Mandiri diartikan bahwa perancangan penelitian, pelaksanaan penelitian, penulisan laporan hasil penelitian yang dilakukan oleh mahasiswa itu sendiri yang dibantu oleh dosen pembimbing sebagai fasilitator. Penyusunan tugas akhir merupakan salah satu syarat wajib untuk menyelesaikan studi.[1] Tugas akhir juga bertujuan untuk menguji kompetensi mahasiswa dalam menerapkan pengetahuan dan keterampilan yang telah dipelajari selama masa perkuliahan.

Meskipun merupakan tahap penting dalam studi mahasiswa, dalam pengerjaan tugas akhir, mahasiswa sering menghadapi kendala terutama dalam menentukan ide atau judul penelitian. Beberapa faktor penyebabnya meliputi bidang dan minat, orisinalitas ide, serta tingkat kesulitan yang tidak sesuai dengan kompetensi yang diharapkan. Kondisi ini mengakibatkan tertundanya penyelesaian tugas akhir dan berdampak pada perpanjangan masa studi mahasiswa tersebut.

Penelitian menunjukkan bahwa menentukan topik atau judul penelitian merupakan kendala terbesar bagi mahasiswa dalam menyelesaikan tugas akhir. Studi oleh Dyah S dkk. (2022) menemukan bahwa 64% mahasiswa mengalami kesulitan dalam menentukan tema tugas akhir, yang menjadi faktor utama sebagai penghambat untuk menyelesaikan studi tepat waktu [2]. Selain itu, penelitian yang dilakukan oleh Pratiwi dan Roosyanti (2022) menunjukkan bahwa hambatan kognitif, seperti kebingungan dalam mengembangkan teori dan kurangnya pemahaman metodologi penelitian, juga menjadi tantangan utama bagi mahasiswa, dengan tingkat kesulitan mencapai 67,39% [3].

Mahasiswa juga sering mengalami kesulitan pada tahap awal menentukan judul tugas akhir. Mereka cenderung merasa ragu dan tidak memiliki kejelasan dalam memutuskan topik, sehingga memerlukan waktu lebih lama untuk memulai penelitian. Sebagian besar mahasiswa belum memahami langkah-langkah dalam menentukan tema atau judul yang sesuai, yang pada akhirnya memperlambat proses

pengajuan tugas akhir. Selain itu, keterbatasan informasi yang tersedia di web perpustakaan kampus juga menjadi kendala utama, karena sistem yang ada hanya menampilkan judul, penulis, dan lokasi kampus, tanpa adanya filter berdasarkan bidang minat atau tahun penelitian. Hal ini semakin menyulitkan mahasiswa dalam memilih topik yang relevan dengan bidang keahliannya [4].

Kompleksitas permasalahan ini semakin meningkat seiring dengan bertambahnya jumlah mahasiswa dari tahun ke tahun. Saat ini, pengelolaan data tugas akhir di banyak institusi masih dilakukan secara sederhana, dimana data hanya disimpan dalam format *excel*, *Digital Versatile Disc* (DVD), atau *word*, tanpa sistem *database* yang terintegrasi. Keterbatasan ini menyulitkan dosen dalam menentukan atau mengetahui apakah judul yang diajukan pernah digunakan sebelumnya, yang berpotensi menimbulkan tindak *plagiarisme* [5]. Selain itu, keterbatasan akses terhadap *database* tugas akhir mahasiswa sebelumnya mengakibatkan terjadinya penelitian yang berulang tanpa pengembangan yang signifikan, bahkan berpotensi menimbulkan kesamaan judul yang dapat dikategorikan sebagai tindak plagiat [6]. Dampak dari berbagai permasalahan ini adalah penurunan kualitas penelitian akibat kurangnya referensi dan arahan dalam pengembangan ide. Proses bimbingan juga menjadi kurang efisien, karena dosen pembimbing harus mengalokasikan lebih banyak waktu untuk membantu mahasiswa menemukan dan memvalidasi keaslian topik penelitian mereka.

Menghadapi kompleksitas permasalahan ini, perkembangan artificial intelligence seperti *Google Gemini* membuka peluang baru dalam membantu proses penentuan judul tugas akhir. Kombinasi algoritma *Decision Tree* untuk melakukan klasifikasi dan *Fuzzy Matching* untuk pengecekan kemiripan dapat diimplementasikan dalam sebuah aplikasi mobile yang mudah diakses. *Decision Tree* dapat membantu mengklasifikasikan tingkat kesulitan dan kesesuaian topik dengan kompetensi mahasiswa, sementara *Fuzzy Matching* memungkinkan pendeteksian kemiripan judul secara lebih fleksibel. Pengembangan dalam bentuk aplikasi mobile akan memudahkan akses bagi mahasiswa maupun dosen pembimbing kapanpun dan dimanapun.

Melalui implementasi solusi teknologi tersebut, berbagai manfaat dapat diperoleh dalam proses penentuan judul tugas akhir. Sistem dapat memberikan

rekomendasi judul secara otomatis berdasarkan bidang minat dan kompetensi mahasiswa, serta melakukan pengecekan kemiripan judul secara efisien dengan database yang ada. Klasifikasi tingkat kesulitan dapat diukur secara objektif menggunakan parameter yang terstandar, sehingga membantu mahasiswa memilih topik yang sesuai dengan kemampuannya. Dengan dikembangkan dalam bentuk aplikasi mobile, sistem ini akan mudah diakses oleh seluruh civitas akademika, yang pada akhirnya dapat meningkatkan efisiensi proses bimbingan dan kualitas penelitian tugas akhir.

1.2. Rumusan Masalah

Berdasarkan latar belakang dan identifikasi masalah yang ada, dapat kita ambil rumusan masalah yaitu :

1. Bagaimana cara mengintegrasikan model *Google Gemini* untuk memberikan rekomendasi judul yang sesuai dengan bidang minat mahasiswa?
2. Bagaimana algoritma decision tree dapat diterapkan dalam mengklasifikasikan tingkat kesulitan judul tugas akhir?
3. Bagaimana cara mengimplementasikan *fuzzy matching* untuk mendeteksi kesamaan judul tugas akhir?
4. Bagaimana merancang aplikasi mobile yang dapat mengintegrasikan seluruh fitur tersebut secara efektif?

1.3. Tujuan

Adapun tujuan dari dilakukannya pengembangan aplikasi ini adalah :

1. Mengintegrasikan model *Google Gemini* untuk memberikan rekomendasi judul berdasarkan bidang minat mahasiswa.
2. Mengembangkan sistem klasifikasi tingkat kesulitan menggunakan algoritma *decision tree.*
3. Mengimplementasikan algoritma *fuzzy matching* untuk analisis kesamaan judul tugas akhir secara otomatis.
4. Mengembangkan aplikasi mobile yang mengintegrasikan seluruh fitur tersebut untuk platform Android.

1.4. Manfaat

Manfaat dari pengembangan aplikasi ini adalah :

1. Mempermudah proses penentuan judul tugas akhir yang sesuai dengan minat dan kemampuan bagi mahasiswa.
2. Membantu meminimalisir resiko plagiasi ide atau duplikasi judul bagi mahasiswa.
3. Memberikan referensi ide penelitian sebelumnya atau ide baru yang relevan dengan pilihan mahasiswa.
4. Meningkatkan efisiensi dalam memantau keaslian dan orisinalitas judul bagi dosen.
5. Mengoptimalkan waktu bimbingan dengan fokus pada pengembangan penelitian bagi dosen.
6. Meningkatkan kualitas manajemen tugas akhir bagi jurusan Teknologi Informasi.
7. Memiliki database tugas akhir yang terstruktur dan terintegrasi.

1.5. Batasan Masalah

Agar penelitian ini sesuai dengan tujuan dan dapat diselesaikan tepat waktu, maka ditetapkan batasan-batasan sebagai berikut :

1. Aplikasi ini hanya dirancang untuk mahasiswa jurusan teknologi informasi di Politeknik Negeri Padang.
2. Sistem menggunakan *Google Gemini* untuk menghasilkan rekomendasi judul.
3. Klasifikasi tingkat kesulitan ditentukan dengan menggunakan parameter :
    a. Kompleksitas teknologi yang akan digunakan.
    b. Ketersediaan sumber daya dan referensi
    c. Estimasi waktu pengerjaan.
4. Analisis kesamaan menggunakan *fuzzy matching* dengan *threshold* minimal 70%.
5. Aplikasi mobile dikembangkan untuk *Android* dengan minimum API level 24 (*Android* 7.0).

BAB II

TINJAUAN PUSTAKA

2.1. Penelitian Terdahulu

Dalam melakukan penelitian ini, penulis mengkaji beberapa penelitian terdahulu yang berkaitan dengan sistem rekomendasi dan deteksi kemiripan judul tugas akhir. Penelitian-penelitian ini mencakup berbagai pendekatan mulai dari *collaborative filtering* hingga *artificial intelligence.*

Abdus Salam dkk. (2021) mengembangkan sistem rekomendasi tugas akhir mahasiswa menggunakan metode *Collaborative Filtering* (CF). Penelitian tersebut mengimplementasikan sistem yang dapat memberikan rekomendasi judul berdasarkan tema dan sub-tema penelitian, dengan hasil pengujian menunjukkan tingkat akurasi yang baik (Precision 0.9, Recall 1.0, F-Measure 0.8) [7]. Meskipun penelitian tersebut berhasil dalam memberikan rekomendasi, pendekatan yang digunakan masih terbatas pada pencocokan tema tanpa mempertimbangkan tingkat kesulitan dan kesesuaian dengan kemampuan mahasiswa seperti yang diusulkan dalam penelitian ini.

Terkait dengan aspek deteksi kemiripan, Bei Harira Irawan dkk. (2021) mengembangkan sistem deteksi kemiripan judul skripsi menggunakan algoritma *Levenshtein Distance* di STMIK MIC Cikarang. Penelitian tersebut menggunakan pendekatan multi-target dengan *threshold* ≥ 4 dan bobot *similarity* > 25% untuk menentukan kemiripan judul. Hasil penelitian menunjukkan bahwa sistem dapat mengidentifikasi judul yang mirip dengan akurasi yang baik, di mana dari 6 kategori data target, sistem dapat mendeteksi kemiripan pada kategori perancangan sistem (42,85%) dan aplikasi berbasis web (28,5%) [6]. Meskipun efektif dalam deteksi kemiripan, penelitian tersebut belum melakukan integrasi fitur rekomendasi atau klasifikasi tingkat kesulitan seperti yang diajukan dalam penelitian ini.

Dalam konteks sistem pendukung keputusan, Ely Nurhalizah Nst dkk. (2023) mengembangkan sistem pemilihan topik skripsi menggunakan logika *fuzzy*. Penelitian ini menunjukkan efektivitas dalam membantu mahasiswa memilih judul skripsi yang sesuai dengan minat dan kemampuan mereka [8]. Sejalan dengan penelitian tersebut, Damayanti Hulu dan R. Mahdalena (2021) juga mengembangkan sistem serupa namun menggunakan metode *Simple Additive Weighting* (SAW). Sistem ini membantu mahasiswa menentukan topik berdasarkan

kriteria seperti nilai akademik dan minat, dengan hasil menunjukkan topik "Sistem Pakar" sebagai pilihan tertinggi [9]. Meskipun kedua penelitian ini memberikan hasil yang baik, keduanya belum mengintegrasikan teknologi *Artificial Intelligence* (AI) seperti yang dikembangkan dalam penelitian ini.

Penggunaan pendekatan machine learning diperkenalkan oleh Rais dkk. (2022) yang melakukan penelitian untuk memprediksi tema tugas akhir mahasiswa prodi D3 Teknik Komputer. Penelitian ini menggunakan metode klasifikasi *Supervised Learning*, khususnya *Neural Network* (NN) yang dioptimasi dengan *Particle Swarm Optimization* (PSO). Dengan menggunakan data nilai mata kuliah jurusan sebagai dasar prediksi, hasil penelitian menunjukkan peningkatan akurasi model dari 91,24% menjadi 92,70% setelah dioptimasi dengan PSO [10]. Meskipun akurasi yang dicapai sudah tinggi, pendekatan ini belum mempertimbangkan aspek kemiripan judul dan tingkat kesulitan seperti yang diajukan dalam penelitian ini.

Berdasarkan kajian terhadap penelitian-penelitian tersebut, dapat disimpulkan bahwa meskipun telah ada berbagai pendekatan untuk sistem rekomendasi dan deteksi kemiripan judul tugas akhir, belum ada yang mengintegrasikan seluruh aspek yang diusulkan dalam penelitian ini, yaitu pengecekan kemiripan, rekomendasi berbasis AI, dan klasifikasi tingkat kesulitan dalam satu sistem mobile yang terintegrasi.

2.2. Landasan Teori

2.2.1. Tugas Akhir

Tugas akhir merupakan karya Ilmiah hasil penelitian mandiri yang dilakukan oleh mahasiswa di akhir masa studi, pada tugas akhir di jurusan Politeknik Negeri Padang terdapat perbedaan karakteristik antara program D3 dan D4. Untuk program D3, Tugas akhir merupakan implementasi sistem yang menyelesaikan permasalahan praktis di instansi atau industri dengan arahan dosen pembimbing sebagai fasilitator. Sementara pada program D4, Proyek Akhir (PA) merupakan karya ilmiah yang lebih menekankan pada kegiatan penelitian mandiri dengan cakupan yang lebih luas, mulai dari perancangan penelitian, pelaksanaan, hingga penulisan laporan [1].

Dalam proses penentuan judul, program D3 menyesuaikan dengan karakteristik program studi masing-masing. Prodi Manajemen Informatika fokus pada implementasi sistem informasi, pemrograman berorientasi bisnis, dan aplikasi akuntansi. Sedangkan Prodi Teknik Komputer mencakup pengembangan jaringan, *cloud computing*, *DevOps*, keamanan siber, IoT, aplikasi mobile, dan teknologi *microservice*. Untuk program D4, judul dapat dipilih sendiri oleh mahasiswa atau ditentukan oleh dosen pembimbing dengan mempertimbangkan ketertiban metodologi dan didasarkan pada pengamatan lapangan atau analisis data sekunder[1].

Terkait kriteria kelayakan judul, program D3 menekankan pada aspek implementasi praktis dan penyelesaian masalah di industri. Judul harus sesuai dengan bidang keahlian program studi dan memiliki nilai manfaat bagi instansi atau industri terkait. Sedangkan untuk program D4, kriteria kelayakan meliputi kejelasan objek yang akan dirancang, permasalahan yang akan dipecahkan, metode yang akan digunakan, serta kemampuan dalam melakukan kajian secara kuantitatif dan kualitatif. Judul PA D4 juga harus memenuhi aspek orisinalitas dan memiliki rasional yang dinilai penting serta bermanfaat dari berbagai segi [1].

2.2.2. Artificial Intelligence

Kecerdasan buatan atau *Artificial Intelligence* (AI) merupakan cabang ilmu komputer yang mengembangkan sistem untuk meniru kemampuan kognitif manusia. Teknologi ini dirancang untuk belajar, memecahkan masalah, dan mengenali pola seperti cara kerja otak manusia. Dengan perkembangan yang pesat, AI kini diterapkan untuk meningkatkan efisiensi berbagai aspek kehidupan manusia melalui sistem komputer, perangkat lunak, dan robot yang dapat "berpikir" layaknya manusia [11].

Dalam konteks *Artificial Intelligence* (AI) ada empat pendekatan utama yang dapat diambil [12]:

1. *Acting humanly* (Bertindak seperti manusia)

   Sistem *Artifial Intelligence* (AI) mampu melakukan tugas atau interaksi dengan lingkungan sebagaimana manusia melakukannya.

2. *Thinking humanly* (Berfikir seperti manusia)
   Sistem *Artificial Intelligence* (AI) memiliki kemampuan untuk berfikir dan memproses informasi sebagaimana manusia berfikir.
3. *Think rationally* (Berpikir rasional)
   Sistem *Artificial Intelligence* (AI) dapat melakukan pemikiran yang logis dan rasional dalam pengambilan keputusan.
4. *Act rationally* (Bertindak rasional)
   Sistem *Artificial Intelligence* (AI) mampu bertindak dan merespons situasi dengan cara yang rasional, berdasarkan logika dan tujuan yang telah ditentukan.

Dengan demikian, *Artificial Intelligence* (AI) berupaya menciptakan mesin yang dapat meniru atau bahkan melampaui kemampuan kognitif manusia dalam berbagai aspek tugas, pemikiran dan tindakan[12].

2.2.3. Google Gemini AI

Salah satu implementasi AI yang populer adalah model *Google Gemini*. Sistem ini memanfaatkan kecerdasan buatan untuk berkomunikasi dengan pengguna melalui percakapan yang natural. *Google Gemini* memiliki kemampuan untuk memberikan jawaban cepat dan akurat berkat basis pengetahuan yang luas, yang diperoleh dari berbagai sumber seperti artikel ilmiah, publikasi media, dan dokumen digital lainnya. Ketika pengguna mengirimkan pertanyaan, sistem akan segera memproses informasi dari *database*-nya untuk menghasilkan *respons* yang relevan dan terstruktur [13].

Sebagai model kecerdasan buatan (AI) multimodal yang dikembangkan oleh *Google DeepMind*, *Google Gemini* mampu memproses dan menggabungkan berbagai jenis informasi, termasuk teks, kode, audio, gambar, dan video. Integrasi *Gemini* melalui API memungkinkan pengembang dan pengguna memanfaatkan kemampuannya untuk berbagai tugas, mulai dari pemahaman bahasa alami hingga pencarian semantik di aplikasi Google. Dengan arsitektur berbasis transformer *encoder-decoder* dan pendekatan *mixture-of-experts*, Gemini dapat mengoptimalkan penggunaan sumber daya dan meningkatkan kinerja dalam menghasilkan respons yang kontekstual dan relevan. Selain itu, dukungan konteks hingga 1 juta token memungkinkannya untuk memproses volume informasi yang

besar, sehingga ideal untuk analisis dokumen, penulisan kreatif, dan pemahaman percakapan berkelanjutan [14].

Google Gemini AI memiliki beberapa keunggulan yaitu [14] :

1. Dapat memproses berbagai jenis informasi seperti teks, kode, audio, gambar, dan video, sehingga lebih fleksibel untuk berbagai aplikasi.
2. Menggunakan sistem *encoder-decoder* dan pendekatan *mixture-of-experts,* yang memungkinkan optimalisasi sumber daya dan kinerja yang lebih efisien.
3. Memiliki kapasitas hingga 1 juta token dalam satu permintaan, memungkinkan analisis dokumen panjang, penulisan kreatif, dan pemahaman percakapan yang berkelanjutan.
4. Dapat menyimpan dan mengingat interaksi sebelumnya hingga tiga tahun, sehingga mampu mempertahankan kontinuitas percakapan dan memberikan respons yang lebih kontekstual.
5. Terintegrasi dengan aplikasi seperti *Gmail, Maps*, dan *YouTube*, serta mampu melakukan pencarian semantik melalui *Google Apps Script* dan *Gemini Pro API.*
6. Memanfaatkan algoritma canggih untuk menganalisis konteks pertanyaan, memastikan respons yang akurat dan sesuai dengan kebutuhan pengguna.
7. Pengguna dapat menyunting atau memodifikasi respons AI secara langsung, termasuk mempersingkat, memperpanjang, atau mengubah nada respons.
8. Dirancang agar dapat berjalan di berbagai perangkat, mulai dari pusat data hingga *smartphone,* memastikan kinerja yang efisien dan dapat diakses secara luas.
9. Tersedia dalam paket bisnis dan *enterprise* dengan perlindungan data tingkat tinggi, mempermudah perusahaan memanfaatkan AI untuk meningkatkan produktivitas.
10. Dapat memproses gambar melalui *Google Lens* untuk menghasilkan respons kreatif, memberikan pengalaman interaksi yang lebih kaya.

2.2.4. Fuzzy Matching

Fuzzy string macthing merupakan metode pencocokan string secara samar, di mana dua string yang dibandingkan dapat memiliki susunan karakter yang berbeda tetapi tetap dianggap mirip berdasarkan kemiripan tekstual (*Approximate String Matching*) atau kemiripan fonetik (*Phonetic String Matching*)[15].

1. Jenis pencocokan dalam *fuzzy string matching*

   Fuzzy String Matching dapar dibagi menjadi dua kategori utama yaitu :

   a. Pencocokan string berdasarkan kemiripan penulisan (*Approximate String Matching*) merupakan pencocokan string dengan dasar kemiripan dari segi penulisannya (jumlah karakter, susunan karakter dalam dokumen). Tingkat kemiripan ditentukan dengan jauh tidaknya beda penulisan dua buah string yang dibandingkan tersebut dan nilai tingkat kemiripan ini ditentukan oleh pemrogram (*Programmer*).

      Contoh : “c mpuler” dengan “compiler”, memiliki jumlah karakter yang sama tetapi ada dua karakter yang berbeda, jika perbedaan dua karakter ini dapat di toleransi sebagai sebuah kesalahan penulisan, maka dua string tersebut dikatakan cocok.

   b. Pencocokan string berdasarkan kemiripan ucapan (*Phonetic string matching*) merupakan pencocokan string dengan dasar kemiripan dari segi pengucapannya, meskipun ada perbedaan penulisan dua string yang dibandingkan tersebut.

      Contoh : “step” dengan “steppe”, “sttep” , “stepp”, “stepe”

2. Algoritma dalam *levenshtein distance*

Fuzzy string matching, merupakan pencocokan string secara samar, maksudnya pencocokan string dimana string yang dicocokkan memiliki kemiripan dimana keduanya memiliki susunan karakter yang berbeda (mungkin jumlah atau urutannya) tetapi string-string tersebut memiliki kemiripan baik kemiripan tekstual/penulisan (*Approximate String Matching*) atau kemiripan ucapan (*Phonetic String Matching*) [15]. *Fuzzy string matching* dapat dibagi menjadi dua yaitu :

1. Pencocokan string berdasarkan kemiripan penulisan (*Approximate String Matching*) merupakan pencocokan string dengan dasar kemiripan dari segi penulisannya (jumlah karakter, susunan karakter dalam dokumen). Tingkat kemiripan ditentukan dengan jauh tidaknya beda penulisan dua buah string yang dibandingkan tersebut dan nilai tingkat kemiripan ini ditentukan oleh pemrogram (*Programmer*).

   Contoh : “c mpuler” dengan “compiler”, memiliki jumlah karakter yang sama tetapi ada dua karakter yang berbeda, jika perbedaan dua karakter ini dapat di toleransi sebagai sebuah kesalahan penulisan, maka dua string tersebut dikatakan cocok.
2. Pencocokan string berdasarkan kemiripan ucapan (*Phonetic string matching*) merupakan pencocokan string dengan dasar kemiripan dari segi pengucapannya, meskipun ada perbedaan penulisan dua string yang dibandingkan tersebut.

   Contoh : “step” dengan “steppe”, “sttep” , “stepp”, “stepe”

Dalam implementasinya, *fuzzy matching* menggunakan beberapa algoritma utama. Algoritma *brute force* merupakan pendekatan paling sederhana yang mencocokan string secara langsung berdasarkan pernyataan masalah. Algoritma *knuth-morris-pratt* (KMP) merupakan pengembangan yang lebih efisien, yang melakukan perbandingan karakter dari kiri ke kanan dengan memanfaatkan informasi ketidakcocokan sebelumnya untuk menentukan pergeseran. Sedangkan algoritma *rabin-karp* menggunakan fungsi hash sebagai pembanding antara string yang dicari dengan substing pada teks, dimana jika dua string sama maka nilai hash nya juga sama [15].

2.2.5. Decision Tree

Decision Tree adalah metode pemodelan prediktif dalam analisis data yang menggunakan struktur pohon untuk menggambarkan dan membuat keputusan berdasarkan serangkaian aturan dan kondisi. Ini merupakan alat yang kuat dalam machine learning yang dapat digunakan untuk memprediksi hasil berdasarkan serangkaian fitur atau variabel input. Struktur datanya terdiri dari tiga jenis node yaitu simpul akar (root/node), simpul percabangan/internal (branch/simpul internal), dan simpul daun (leaf/node).

Algoritma *decision tree* menggunakan persamaan umum sebagai berikut [16] :

$$f(x) = \sum_{i=1}^{N} c_i \, . \, I(x \in R_i)$$

Keterangan :

f(x) = Prediksi yang dihasilkan oleh decision tree untuk input x.

N = Jumlah simpul daun.

ci = Nilai kelas yang diberikan pada simpul daun ke-i.

Ri = Wilayah yang diwakili oleh simpul daun ke-i.

Secara sederhana, persamaan ini menjelaskan bahwa prediksi akhir untuk suatu input x adalah nilai kelas dari node daun dimana x tersebut berakhir setelah mengikuti semua aturan pemisahan di tree. Jika kita gunakan dalam penelitian penulis untuk melakukan klasifikasi tingkat kesulitan tugas akhir, maka :

- x bisa berupa fitur-fitur dari suatu judul tugas akhir.
- ci bisa berupa label tingkat kesulitan (mudah/sedang/sulit).
- Ri adalah kondisi-kondisi yang harus dipenuhi untuk mencapai suatu klasifikasi tertentu.

2.2.6. Aplikasi Mobile

Aplikasi mobile adalah suatu perangkat lunak yang dirancang khusus untuk berjalan pada perangkat bergerak (*mobile devices*) seperti *smartphone*, *tablet*, *handphone*, atau *smartwatch*. Aplikasi ini dapat beroperasi secara *stand alone* pada sistem operasi yang mendukung perangkat tersebut. Distribusi aplikasi mobile dilakukan melalui platform pengunduhan resmi yang dikelola oleh masing-masing sistem operasi, seperti Apple App Store untuk iOS, Google Play Store untuk Android, Windows Phone Store, dan BlackBerry App World.

Saat ini, ekosistem aplikasi mobile didominasi oleh dua sistem operasi utama yaitu Android dan iOS. Android, yang merupakan sistem operasi berbasis Linux, didesain khusus untuk pengelolaan sumber daya perangkat keras pada *smartphone* dan *tablet*. Keunggulan Android terletak pada sifatnya yang terbuka (*open platform*), memungkinkan para *developer* untuk menciptakan aplikasi mereka sendiri dan mendistribusikannya ke berbagai perangkat bergerak [17].

Dalam pengembangan aplikasi mobile modern, Flutter hadir sebagai *Software Development Kit* (SDK) dan *framework* yang menawarkan pendekatan baru dalam membangun antarmuka pengguna. Dikembangkan oleh Google dan dirilis pada tahun 2018, Flutter merupakan SDK *open-source* yang didukung oleh komunitas pengembang aktif. Keunggulan utama Flutter adalah kemampuannya menggunakan basis kode tunggal (*single codebase*) untuk menghasilkan aplikasi yang dapat berjalan di berbagai platform seperti *mobile, web,* dan *desktop*. Framework ini dirancang untuk memberikan kinerja setara dengan aplikasi *native* sambil menyederhanakan proses pengembangan. Sebagai alternatif dari React Native, Flutter menyediakan berbagai alat dan fitur yang memungkinkan pengembang untuk mempercepat dan mempermudah proses pembuatan aplikasi [18].

Flutter memiliki beberapa kelebihan sebagai *framework* pengembangan aplikasi, di antaranya [18] :

1. Pengembangan Lintas Platform (*Cross-Platform*) Flutter memungkinkan pengembangan aplikasi untuk berbagai platform seperti *web, mobile,* dan Windows menggunakan satu basis kode (*single codebase*), sehingga lebih efisien dalam pengembangan dan pemeliharaan aplikasi.
2. Performa yang Optimal Flutter menggunakan bahasa pemrograman Dart yang mendukung kompilasi AOT (*Ahead Of Time*) dan JIT (*Just In Time*), sehingga dapat menghindari penggunaan JavaScript *bridge* yang menghasilkan eksekusi aplikasi yang lebih cepat.
3. *Hot Reload* Fitur *Hot Reload* pada Flutter memungkinkan pengembang untuk melihat perubahan kode secara langsung tanpa perlu memulai ulang aplikasi, sehingga dapat mempercepat siklus pengembangan.
4. Widget Khusus Flutter menyediakan widget-widgetnya sendiri dan tidak bergantung pada OEM (*Original Equipment Manufacturer*) *widgets*, sehingga menghindari masalah performa yang sering dihadapi framework lain yang harus sering mengakses sumber daya dari platform iOS dan Android SDK.
5. *Renderer* Mandiri Flutter memiliki *renderer widget* sendiri menggunakan Skia, sehingga tidak bergantung pada sistem operasi dalam merender

tampilan aplikasi, yang dapat meningkatkan konsistensi tampilan di berbagai perangkat.

6. Desain Adaptif Flutter menyediakan dua library desain yaitu *'Cupertino'* untuk komponen iOS dan *'Material Design'* untuk komponen Android, memungkinkan pengembang untuk membuat antarmuka pengguna yang sesuai dengan platform yang digunakan.
7. Sifat *Open Source* Sebagai *framework open source*, Flutter mendapatkan dukungan dari komunitas pengembang yang dapat membantu mengembangkan dan meningkatkan kualitas *framework* secara berkelanjutan.
8. Kecepatan Pengembangan Flutter dirancang oleh Google dengan tujuan meningkatkan kecepatan pengembangan aplikasi sambil tetap mempertahankan performa dan UI yang setara dengan *aplikasi native*.

BAB III

METODOLOGI

3.1. Metodologi Pelaksanaan

Penelitian ini mengembangkan sistem aplikasi *mobile* berbasis *Artificial Intelligence* (AI) untuk membantu mahasiswa Jurusan Teknologi Informasi dalam proses pengajuan judul tugas akhir. Sistem ini mengintegrasikan tiga fungsi utama yaitu pengecekan kesamaan judul menggunakan algoritma *Fuzzy Matching* untuk membandingkan judul yang diajukan dengan *database* judul-judul terdahulu guna mencegah duplikasi ide dan plagiarisme, pemberian rekomendasi judul menggunakan model *Google Gemini* yang disesuaikan dengan bidang minat mahasiswa untuk menghasilkan alternatif judul yang relevan dan inovatif, serta klasifikasi tingkat kesulitan menggunakan algoritma *Decision Tree* untuk mengkategorikan judul ke dalam tiga tingkat (mudah, sedang, dan susah) berdasarkan parameter-parameter yang telah ditentukan. Melalui integrasi ketiga fungsi tersebut, sistem ini diharapkan dapat membantu mahasiswa dalam memilih judul tugas akhir yang sesuai dengan kemampuan dan minatnya, serta menjamin originalitas penelitian.

Pengembangan aplikasi ini menggunakan metode *Software Development Life Cycle* (SDLC) dengan model *waterfall*. *Software Development Life Cycle* (SDLC) adalah metode pengembangan perangkat lunak yang terdiri dari serangkaian tahapan terstruktur, yaitu perencanaan, analisis, desain, implementasi, pengujian, verifikasi dan pemeliharaan. Metode ini bertujuan untuk meningkatkan efektivitas pemecahan masalah selama proses pengembangan berlangsung, dengan memastikan bahwa setiap tahapan diselesaikan sebelum melanjutkan ke tahap berikutnya [19]. Beberapa model *Software Development Life Cycle* (SDLC) yang sering digunakan meliputi *waterfall*, *Joint Application Design* (JAD), *agile*, dan *prototype*. Berdasarkan penelitian terhadap 12 jurnal, model *waterfall* dan *agile* terbukti paling efektif karena perencanaannya yang matang serta proses pengerjaan yang terstruktur dan berurutan.

Alasan menggunakan metode *Software Development Life Cycle* (SDLC) dalam proyek ini adalah sebagai berikut [19]:

1. Perencanaan yang Matang dan Terstruktur

   Software Development Life Cycle (SDLC) memastikan setiap tahapan pengembangan dilakukan secara terstruktur, mulai dari analisis kebutuhan hingga pemeliharaan. Hal ini mendukung integrasi algoritma *fuzzy matching*, *Google Gemini*, dan decision tree secara bertahap dan terkoordinasi.

2. Pengendalian Proses yang Ketat

   Setiap fase dalam SDLC memberikan kontrol yang jelas terhadap jalannya pengembangan, sehingga memudahkan pemantauan kualitas aplikasi yang dikembangkan.

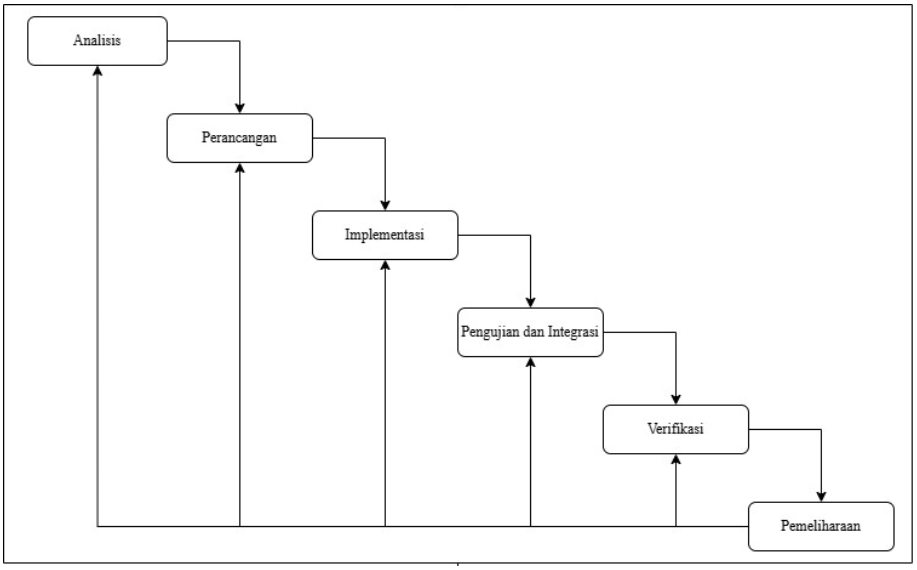
3. Efektivitas dan Keberhasilan Proyek

   Berdasarkan penelitian terhadap 12 jurnal, model SDLC terbukti mampu menghasilkan sistem informasi yang sesuai dengan kebutuhan pengguna berkat proses pengerjaannya yang terstruktur dan berurutan

4. Struktur yang Mendukung Integrasi Multi-komponen

   Struktur SDLC yang terorganisir memastikan integrasi berbagai komponen teknologi dalam proyek ini dapat dilakukan secara optimal.

Adapun tahapan dalam melaksanakan metode SDLC adalah sebagai berikut :



Gambar 3.1 Metode Pengembangan

1. Analisis (*Analysis*)

   Sebelum memulai pengembangan ini penulis melakukan pengumpulan data-data tugas akhir terdahulu di jurusan Teknologi Informasi untuk disimpan di dataset yang akan digunakan nantinya. Kemudian melakukan analisis kebutuhan sistem, dimulai dari kebutuhan mahasiswa sebagai pengguna untuk menguji judul tugas yang akan mereka kembangkan.

   Setelah pengumpulan data dan analisis kebutuhan sistem, penulis melakukan spesifikasi perangkat keras dan perangkat lunak yang dibutuhkan seperti *Software Development Kit* (SDK) yang digunakan dan *Operation System* (OS) yang bisa menjalankan aplikasi mobile yang akan dikembangkan.

   Setelah spesifikasi didapatkan penulis melakukan analisis terhadap penggunaan algoritma *Fuzzy Match* untuk pengecekan kesamaan judul, dan *Decision Tree* untuk klasifikasi tingkat kesulitan yang diinginkan oleh pengguna.

2. Perancangan (*Design*)

   Pada tahap perancangan, penulis memulai dengan merancang arsitektur sistem aplikasi *mobile* berbasis *Artificial Intelligence* (AI), dan melakukan desain struktur *database* untuk menyimpan data judul. Arsitektur dirancang dengan pendekatan *microservices* untuk mengikuti alur dari melakukan pengecekan judul, kemudian memberi opsi untuk pengguna menentukan bidang, dan kesulitan. Untuk struktur *database* penulis melakukan pendekatan normalisasi untuk mengoptimalkan *dataset* dan pengambilan data, dengan mempertimbangkan skema untuk menyimpan data historis, meta data, dan hasil analisis.

   Setelah arsitektur dan *database* sudah disiapkan, penulis melakukan perancangan antarmuka pengguna (UI/UX) yang intuitif. Perancangan UI/UX meliputi *wireframing, prototyping,* dan *user flow mapping* dengan mempertimbangkan aksesibilitas dan pengalaman pengguna, kemudian penulis melakukan pembuatan diagram UML yang meliputi *use case* untuk menggambarkan interaksi pengguna, *Activity Diagram* untuk alur kerja

sistem, *sequence diagram* untuk interaksi antar objek, dan *class diagram* untuk struktur sistem.

3. Implementasi (*Implementation*)

   Dalam melakukan implementasi sistem, tahap awal dimulai dengan pembangunan *database* menggunakan MongoDB untuk menyimpan dataset judul tugas akhir dan MySQL untuk data relasional, disertai pembuatan indeks dan API *endpoints* untuk operasi CRUD. Setelah infrastruktur *database* siap, pengembangan dilanjutkan dengan implementasi algoritma *Fuzzy Matching* menggunakan metode *Levenshtein Distance* yang dioptimasi dengan *threshold similarity* untuk pengecekan kesamaan judul, dilengkapi fungsi *preprocessing text* dan mekanisme *caching* untuk meningkatkan performa pencarian.

   Berdasarkan hasil pengecekan kesamaan judul, sistem kemudian mengintegrasikan model *Google Gemini* dengan pengembangan sistem *prompt engineering* untuk menghasilkan rekomendasi judul yang relevan. Integrasi ini dilengkapi dengan mekanisme *rate limiting* dan *error handling* untuk menjaga stabilitas sistem saat memberikan rekomendasi. Hasil rekomendasi tersebut selanjutnya diproses menggunakan algoritma *Decision Tree* yang diimplementasikan dengan sistem *preprocessing* dan logika *scoring* untuk klasifikasi tingkat kesulitan, serta dilengkapi mekanisme *feedback* untuk peningkatan akurasi.

   Seluruh fungsionalitas tersebut diintegrasikan ke dalam aplikasi *mobile* yang dikembangkan sesuai rancangan UI/UX, mencakup pembangunan *RESTful API* untuk komunikasi *client-server*, *state management* untuk pengelolaan sesi, sistem *caching* untuk optimasi performa, dan *push notifications* untuk *update* status. Untuk menjamin keamanan sistem secara menyeluruh, implementasi dilengkapi dengan JWT untuk autentikasi, enkripsi data untuk informasi sensitif, dan *role-based access control* untuk manajemen akses pengguna.

4. Pengujian dan Integrasi (*Integration and Testing*)

   Pada tahap integrasi penulis akan melakukan integrasi sesuai dengan kebutuhan aplikasi *mobile* dimulai dari algoritma *fuzzy match* untuk

melakukan pengecekan kesamaan judul yang diajukan pengguna terlebih dahulu, kemudian melakukan integrasi model *Google Gemini* untuk melakukan pemberian rekomendasi-rekomendasi kepada pengguna terkait judul tugas akhir yang relevan dan mengintegrasikan algoritma *decision tree* untuk melakukan klasifikasi hasil yang diterima dari model *Google Gemini*, untuk kemudian dikembalikan ke pengguna.

Proses pengujian dilakukan secara bertahap untuk setiap modul yang diintegrasikan. Pengujian *fuzzy match* dilakukan dengan membandingkan judul yang diajukan dengan dataset yang tersedia, menggunakan *threshold similarity score* untuk menentukan tingkat kesamaan. Integrasi dengan model *Google Gemini* diuji melalui serangkaian *test cases* yang mencakup berbagai bidang keilmuan dan tingkat kompleksitas, memastikan rekomendasi yang diberikan relevan dan sesuai konteks. Pengujian *decision tree* fokus pada akurasi klasifikasi tingkat kesulitan berdasarkan parameter yang telah ditentukan.

Selanjutnya dilakukan *integration testing* untuk memastikan ketiga komponen dapat bekerja secara harmonis. *Response time* diukur untuk mengoptimalkan performa sistem secara keseluruhan. *Load testing* dilakukan untuk menguji ketahanan sistem saat diakses secara bersamaan oleh *multiple users*. *Error handling* dan *fail-safe mechanisms* diimplementasikan untuk menjaga stabilitas sistem.

Pengujian juga mencakup aspek UI/UX untuk memastikan tampilan hasil integrasi ketiga algoritma dapat dipahami dengan mudah oleh pengguna. *Feedback loop* diimplementasikan untuk mengumpulkan data akurasi dan relevansi hasil rekomendasi, yang akan digunakan untuk penyempurnaan sistem secara berkelanjutan.

5. Verifikasi (*Verification*)

Dalam tahap Verifikasi, dilakukan pemeriksaan kesesuaian sistem dengan kebutuhan yang telah didefinisikan menggunakan *requirement checklist* dan validasi hasil pengecekan kesamaan judul dengan tim ahli yang terdiri dari dosen pembimbing dan koordinator tugas akhir. Evaluasi dilakukan menggunakan metrik kuantitatif, termasuk *precision* dan *recall*

untuk mengukur akurasi sistem dalam mendeteksi kesamaan judul, serta *confusion matrix* untuk menilai ketepatan klasifikasi tingkat kesulitan.

Kualitas rekomendasi judul dievaluasi melalui *expert review* dengan parameter relevansi bidang, orisinalitas, dan kelayakan implementasi. *User Acceptance Testing* (UAT) melibatkan sampel mahasiswa tingkat akhir untuk menguji sistem dalam skenario nyata. *Feedback* dikumpulkan melalui kuesioner terstruktur dan wawancara mendalam, mencakup aspek *usability*, *usefulness*, dan *user satisfaction*.

Proses verifikasi juga mencakup validasi teknis seperti *response time*, *error rate*, dan sistem *handling* untuk kasus-kasus khusus. Hasil verifikasi digunakan sebagai dasar penyempurnaan sistem dan dokumentasi final. Metrik keberhasilan didefinisikan dengan *threshold* minimal 85% untuk akurasi sistem dan tingkat kepuasan pengguna.

6. Pemeliharaan (*Maintenance*)

Pemeliharaan sistem mencakup monitoring performa menggunakan *dashboard analytics* yang melacak *response time*, *error rate*, dan penggunaan *resource*. Dataset judul tugas akhir diperbarui setiap semester dengan mekanisme validasi otomatis dan manual. Optimasi algoritma dilakukan berdasarkan analisis *feedback* dan *usage patterns*, dengan fokus pada peningkatan akurasi *fuzzy matching* dan ketepatan rekomendasi.

Sistem backup diimplementasikan dengan jadwal harian untuk *database* dan mingguan untuk seluruh sistem. *Security patches* dan pembaruan sistem dilakukan setiap bulan, disertai dengan *vulnerability scanning*. *Technical support* disediakan melalui *in-app chat* dan *email* dengan *documented* SLA.

Maintenance logs dan *performance metrics* disimpan untuk analisis tren dan prediktif *maintenance*. *Bug tracking* menggunakan sistem tiket dengan prioritas berdasarkan *severity level*. *Knowledge base* diperbarui secara berkala untuk mendukung *self-service troubleshooting* pengguna.

3.2. Implementasi Metode

3.2.1. Google Gemini

Sistem menggunakan *Google Gemini* untuk menghasilkan judul yang relevan dan sesuai dengan yang dipilih oleh pengguna, implementasi *Google Gemini* dilakukan dengan pengembangan sistem *prompt engineering*, yang dirancang untuk menyesuaikan permintaan pengguna berdasarkan bidang penelitian dan tingkat kesulitan.

Integrasi ini dilengkapi dengan :

1. Mekanisme *Rate Limiting*, untuk mebatasi jumlah permintaan ke model dalam periode tertentu.
2. *Error Handling*, untuk menangani kasus kegagalan koneksi atau kesalahan respons dari API *Google Gemini*
3. *Logging* dan *Monitoring*, untuk memastikan hasil rekomendasi tetap relevan dengan kebutuhan pengguna

3.2.2. Decision Tree

Hasil rekomendasi dari *Google Gemini* kemudian diproses lebih lanjut menggunakan algoritma *decision tree* untuk melakukan klasifikasi tingkat kesulitan. Implementasi algoritma ini dilakukan dengan sistem preprocessing dan logika scoring berdasarkan parameter berikut :

1. Kompleksitas teknologi yang akan digunakan
2. Ketersedian sumber daya dan referensi
3. Estimasi waktu pengerjaan

Algoritma *decision tree* dirancang untuk memberikan klasifikasi otomatis dari hasil yang nantinya di *generate* oleh *Google Gemini* berdasarkan parameter yang telah ditentukan. Selain itu, sistem juga akan dilengkapi dengan fitur *feedback*, di mana pengguna dapat memberikan masukan mengenai akurasi klasifikasi yang dihasilkan, yang kemudian digunakan untuk meningkatkan performa model di iterasi berikutnya.

3.2.3. Fuzzy Matching

Algoritma *fuzzy matching* digunakan untuk mendeteksi kesamaan judul tugas akhir guna mencegah duplikasi dan memberikan rekomendasi alternatif jika

terdapat duplikasi dari judul yang di cek oleh pengguna. Metode yang digunakan yaitu *Levenshtein Distance*, yang mengukur jumlah perubahan (penyisipan, penghapusan, atau substitusi karakter) yang diperlukan untuk mengubah satu string menjadi string lain.

Fuzzy matching dioptimasi dengan memberikan *threshold similarity* 70% untuk pengecekan kesamaan judul, dilengkapi dengan fungsi preprocessing text dan mekanisme caching untuk meningkatkan performa pencarian. Preprocessing mencakup normalisasi teks, penghapusan stopwords, dan stemming untuk memastikan perbandingan string lebih akurat.

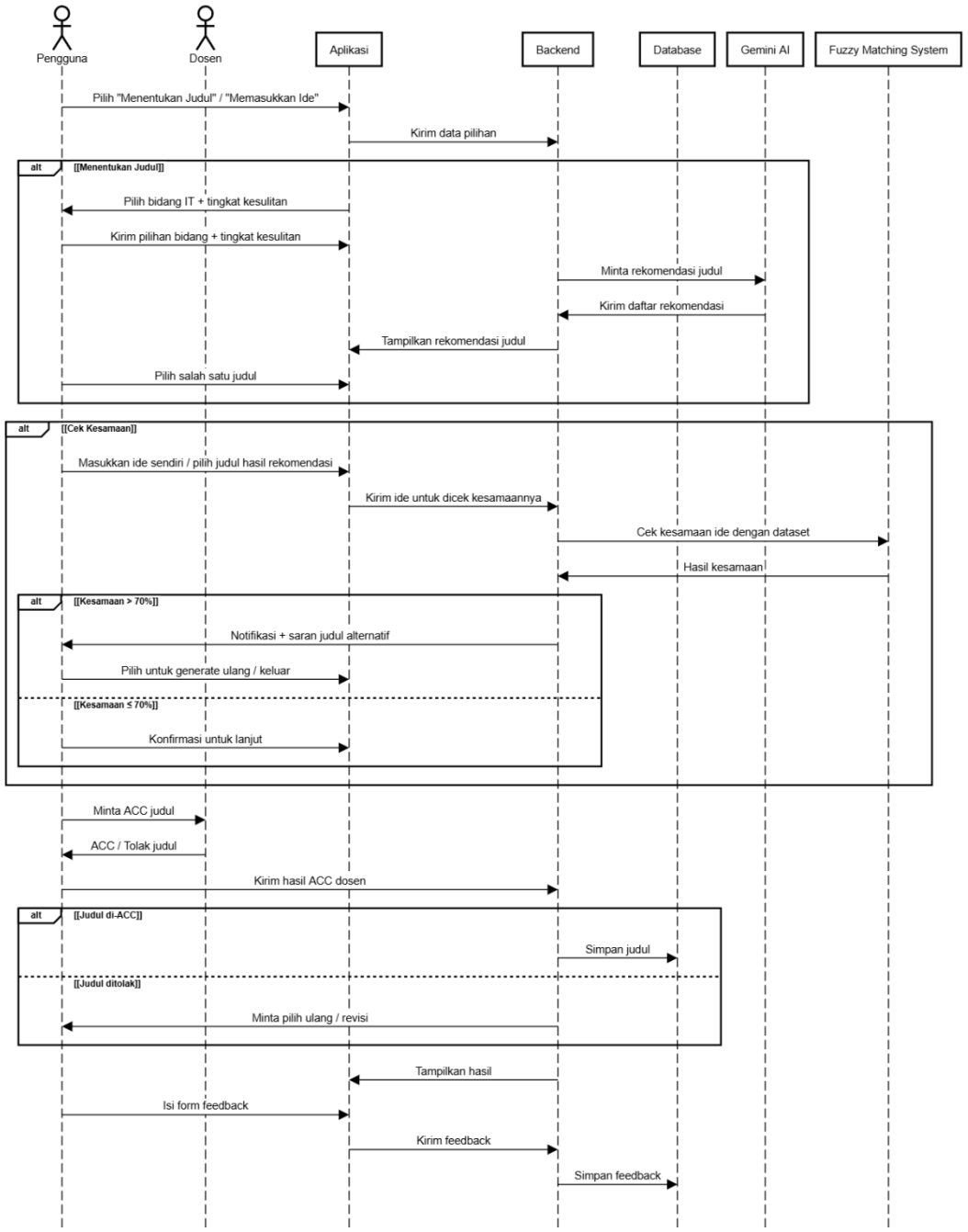
Penentuan threshold dalam fuzzy matching sangat krusial untuk menentukan seberapa mirip dua string dianggap sebagai kesamaan yang relevan. Dalam penelitian sebelumnya terdapat beberapa pendekatan yang telah digunakan :

1. *Neural fuzzy repair* yang dilakukan oleh Bulte dan Tezcan (2019), menggunakan threshold 50% untuk mempertimbangkan kesamaan dalam sistem penerjemah berbasis neural. Dalam penelitian ini, nilai tersebut digunakan untuk menentukan apakah kalimat dalam *Translation Memory* (TM) cukup mirip untuk digunakan dalam augmentasi data [20].
2. Irawan dkk. (2021) dalam sistem deteksi kemiripan judul skripsi menggunakan algoritma *Levenshtein Distance* dengan *threshold* ≥ 4 karakter berbeda dan bobot *similarity* > 25%. Hasilnya menunjukkan efektifitas dalam mendeteksi kemiripan judul ,namun disarankan untuk memberikan bobot *similarity* lebih besar [6].

Berdasarkan studi tersebut, dalam penelitian ini penulis menetapkan threshold 70%, dengan pertimbangan sebagai berikut :

1. Mengurangi *false positive*, yaitu kesamaan yang terlalu rendah yang dapat menyebabkan judul yang sebenarnya unik dianggap mirip.
2. Menyeimbangkan *recall* dan *precision*, karena nilai 70% dianggap cukup untuk menangkap variasi kecil dalam judul tanpa terlalu banyak menghasilkan hasil yang tidak relevan.
3. Konteks *domain*, yaitu tugas akhir di bidang Teknologi Informasi memiliki struktur kalimat yang lebih terstandarisasi, sehingga threshold lebih tinggi dapat diterapkan tanpa mengurangi efektivitas sistem.

3.3. Alur Sistem



Gambar 3.2 Sequence Diagram Alur Sistem

1. Pengguna akan diberikan dua pilihan dalam aplikasi :
    a. Menentukan Judul : Sistem merekomendasikan judul berdasarkan bidang IT dan tingkat kesulitan yang dipilih.

b. Cek Kesamaan : Sistem melakukan pengecekan kesamaan ide dengan dataset tugas akhir yang tersedia.

2. Setelah pengguna memilih salah satu opsi, aplikasi akan mengirimkan data pilihan ke backend untuk diproses sesuai dengan alur yang telah ditentukan.
3. Kemudian proses akan dilanjutkan berdasarkan pilihan pengguna
   a. Menentukan Judul
      1) Aplikasi meminta pengguna memilih bidang IT dan tingkat kesulitan.
      2) Pengguna mengirimkan pilihan tersebut ke aplikasi, yang kemudian diteruskan ke backend.
      3) Backend mengirimkan data ke *Google Gemini* untuk mendapatkan rekomendasi judul.
      4) *Google Gemini* mengembalikan daftar rekomendasi judul ke backend.
      5) Backend meneruskan hasil rekomendasi ke *Decision Tree Classifier* untuk menentukan tingkat kesulitan berdasarkan parameter yang sudah ditentukan
      6) Backend mengembalikan daftar rekomendasi judul beserta tingkat kesulitan kepada aplikasi.
      7) Pengguna memilih salah satu judul dari daftar yang tersedia.
      8) Pengguna dapat langsung mengecek kesamaan judul yang dipilih dengan dataset tugas akhir yang tersedia sebelum mengajukan ke dosen.
   b. Cek Kesamaan

      Pengguna bisa melakukan pengecekan kesamaan ide dengan dua cara:
      1) Cek judul yang dihasilkan oleh sistem setelah menggunakan fitur "Menentukan Judul".
      2) Cek ide yang dimasukkan sendiri tanpa melalui proses rekomendasi judul dari sistem.

      Alur pengecekan kesamaan :

1) *Backend* mengirimkan ide yang dimasukkan pengguna ke *Fuzzy Matching System* untuk diperiksa kesamaannya dengan dataset tugas akhir yang tersedia.
2) *Fuzzy Matching System* mengembalikan hasil tingkat kesamaan ke backend.
3) Jika tingkat kesamaan > 70%, *backend* akan memberikan notifikasi kepada pengguna dan saran judul alternatif
4) Jika tingkat kesamaan ≤ 70%, pengguna dapat melanjutkan ide tersebut.
5) Pengguna akan melakukan konfirmasi ide kepada dosen melalu sistem.
6) Jika ide disetujui atau dikonfirmasi oleh dosen maka *backend* akan meneruskan ide yang diajukan mahasiswa ke *database* untuk disimpan.
7) Jika ide ditolak oleh dosen maka backend akan meminta pengguna untuk memilih ide lain atau melakukan revisi.

4. Penyimpanan dan *feedback* pengguna
   a. Setelah proses selesai, aplikasi akan menampilkan hasil akhir kepada pengguna berupa :
      1) Rekomendasi judul beserta tingkat kesulitan.
      2) Hasil pengecekan kesamaan beserta saran alternatif
   b. Pengguna dapat memberikan *feedback* mengenai pengalaman menggunakan sistem yang akan membantu nantinya untuk pengembangan aplikasi.

3.4. Jadwal Pelaksanaan

Tabel 3.1 Jadwal Pelaksanaan

No	Kegiatan	Bulan 1				Bulan 2				Bulan 3				Bulan 4				Bulan 5				Bulan 6			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	Persiapan dan Perencanaan																								

2	Studi Literatur dan Pengumpulan Data																								
3	Analisis dan Desain Sistem																								
4	Pengembangan Sistem																								
5	Pengujian dan Evaluasi																								
6	Penyusunan Laporan																								
7	Persiapan Presentasi																								
8	Presentasi dan Revisi Akhir																								

DAFTAR PUSTAKA

[1] K. Jurusan Teknologi Informasi Politeknik Negeri Padang Ronal Hadi, K. Program Studi Teknologi Rekayasa Perangkat Lunak Defni, and D. Direktur Politeknik Negeri Padang Surfa Yondri, "LEMBAR PENGESAHAN Buku Pedoman Proyek Akhir ini telah disahkan oleh," 2022.

[2] D. Setyaningrum Winarni and D. Nugraheni, "PAWIYATAN XXIX (2) (2022) 76-82 Analisis Kesulitan Mahasiswa Dalam Menuliskan Tugas Akhir." [Online]. Available: http://e-journal.ikip-veteran.ac.id/index.php/pawiyatan

[3] U. Wijaya, K. Surabaya, and A. Roosyanti, "ANALISIS FAKTOR PENGHAMBAT SKRIPSI MAHASISWA JURUSAN PENDIDIKAN GURU SEKOLAH DASAR UNIVERSITAS WIJAYA KUSUMA SURABAYA Desi Eka Pratiwi", doi: 10.21009/JPD.010.10.

[4] A. Jimly Hanif, M. Nur Farid, and B. Hasanah, "Penerapan Natural Language Processing untuk Klasifikasi Bidang Minat berdasarkan Judul Tugas Akhir," *Jurnal Sistim Informasi dan Teknologi*, pp. 41–49, Feb. 2023, doi: 10.37034/jsisfotek.v5i1.196.

[5] A. Subadri and I. Pratama, "SISTEM DETEKSI PLAGIARISM PADA JUDUL TUGAS AKHIR MENGGUNAKAN METODE RABIN-KARP BERBASIS WEB," 2022.

[6] B. Harira Irawan, M. H. Sahat Simarangkir, and S. MIC Cikarang, "DETEKSI KEMIRIPAN JUDUL SKRIPSI MENGGUNAKAN ALGORITMA LEVENSHTEIN DISTANCE PADA KAMPUS STMIK MIC CIKARANG," 2021.

[7] A. Salam and F. Putraga Albahri, "Sistem Rekomendasi Tugas Akhir Mahasiswa pada AMIK Indonesia untuk Mendukung Merdeka Belajar-Kampus Merdeka Menggunakan Metode Collaborative Filtering (CF)," *Jurnal Teknologi Informasi dan Komunikasi)*, vol. 6, no. 2, p. 2022, 2022, doi: 10.35870/jti.

[8] E. Nurhalizah Nst, L. Efriyanti, S. Zakir, P. Teknik Informatika dan Komputer, T. dan Ilmu Keguruan, and U. Islam Negeri Bukittinggi Sjech MDjamil Djambek Bukittinggi, "Sistem Pendukung Keputusan Rekomendasi Pemilihan Topik Judul Skripsi Menggunakan Metode Logika Fuzzy".

[9] D. Hulu and R. M. Simanjorang, "Sistem Pendukung Keputusan Pemilihan Topik Skripsi Program Studi Teknik Informatika Menggunakan Simple

Additive Weighting (SAW)," *Jurnal Nasional Komputasi dan Teknologi Informasi*, vol. 4, no. 1, 2021.

[10] Rais, Rakhman Arif, and Haqiqi Arfan, "Rekomendasi Tema Tugas Akhir Menggunakan Metode Classifikasi Supervised Learning," 2022.

[11] T. Nur Fitria, "ELT FORUM 12(1) (2023) Journal of English Language Teaching Artificial intelligence (AI) technology in OpenAI ChatGPT application: A review of ChatGPT in writing English essay," *Journal of English Language Teaching*, vol. 6, no. 1, 2023, [Online]. Available: http://journal.unnes.ac.id/sju/index.php/elt

[12] W. R. Fauziyati, "DAMPAK PENGGUNAAN ARTIFICIAL INTELLIGENCE (AI) DALAM PEMBELAJARAN PENDIDIKAN AGAMA ISLAM," Nov. 2023.

[13] W. Suharmawan, "Pemanfaatan Chat GPT Dalam Dunia Pendidikan," *Education Journal : Journal Educational Research and Development*, vol. 7, no. 2, pp. 158–166, Aug. 2023, doi: 10.31537/ej.v7i2.1248.

[14] R. K. Panchal *et al.*, "Comprehensive Study of Google Gemini and Text Generating Models: Understanding Capabilities and Performance." [Online]. Available: https://www.researchgate.net/publication/386101230

[15] Y. Darnita, "Aplikasi Pengarsipan Dokumen Menggunakan Metode String Matching (Studi Kasus Perpustakaan SMP Negeri 5 Seluma)," 2019.

[16] D. A. Mukhsinin, M. Rafliansyah, S. A. Ibrahim, R. Rahmaddeni, and D. Wulandari, "Implementasi Algoritma Decision Tree untuk Rekomendasi Film dan Klasifikasi Rating pada Platform Netflix," *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, vol. 4, no. 2, pp. 570–579, Mar. 2024, doi: 10.57152/malcom.v4i2.1255.

[17] M. B. Sinatria, Oman Komarudin, and Kamal Prihamdani, "PENERAPAN CLEAN ARCHITECTURE DALAM MEMBANGUN APLIKASI BERBASIS MOBILE DENGAN FRAMEWORK GOOGLE FLUTTER," *INFOTECH journal*, vol. 9, no. 1, pp. 132–146, May 2023, doi: 10.31949/infotech.v9i1.5237.

[18] H. Allain, P. Mario Di Francesco, and P. Mario Di Francesco Advisor, "'s Programme in ICT Innovation Improving productivity and reducing costs of mobile app development with Flutter and Backend-as-a-Service Title: Improving productivity and reducing costs of mobile app development with Flut-ter and Backend-as-a-Service," 2020.

[19] C. Yandhika *et al.*, "Sebuah Tinjauan Literatur Sistematis Tentang Metode Pengembangan Perangkat Lunak Sistem Informasi Berbasis Web A

Systematic Literature Review of Web-Based Information Systems Software Development Methods," 2024.

[20] B. Bulté and A. Tezcan, "Neural Fuzzy Repair: Integrating Fuzzy Matches into Neural Machine Translation," Association for Computational Linguistics. [Online]. Available: https://github.com/ardate/SetSimilaritySearch