BAB III
METODE PENELITIAN

3.1 Desain Penelitian

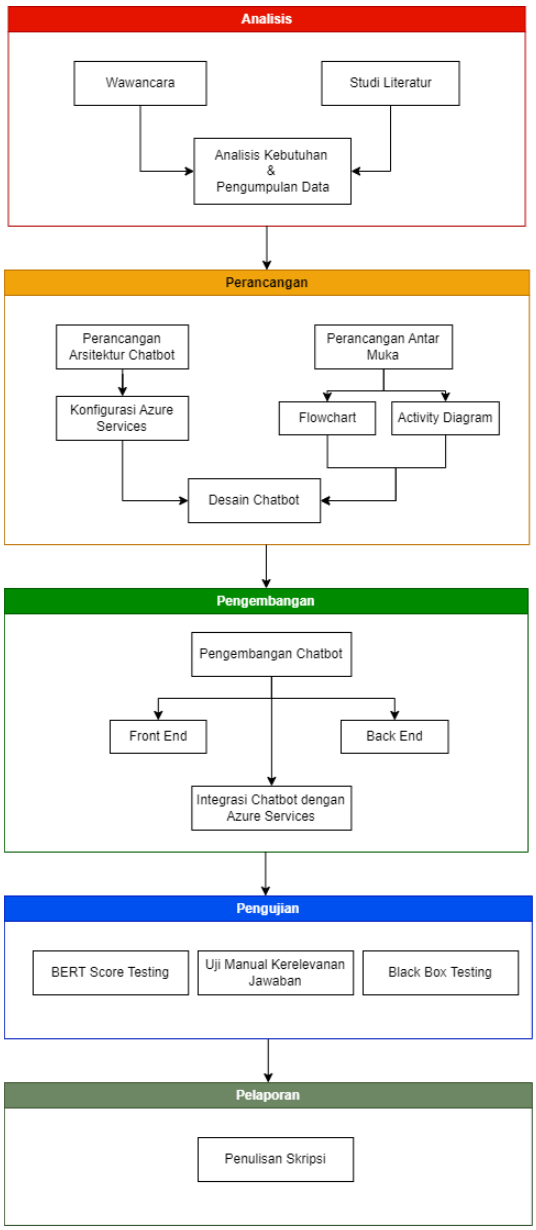
Berdasarkan latar belakang dan tujuan dari penelitian ini, maka metode yang dipakai dalam penelitian ini adalah *Design and Development* (D&D). Metode D&D berfokus pada perancangan, pengembangan, dan pengujian prototipe untuk menyelesaikan sebuah masalah atau memenuhi kebutuhan yang spesifik (Richey dkk., 2014). Metode D&D selaras dengan tujuan dari dilakukannya penelitian ini, yaitu mengembangkan sebuah produk berupa *chatbot* untuk sistem informasi berbasis *large language model* dan diharapkan dapat membuat akses informasi menjadi lebih cepat dan akurat.

Produk yang dihasilkan dalam penelitian ini adalah sebuah *chatbot* dengan basis model LLM GPT-4o yang dilengkapi dengan pengetahuan mengenai BUMA. *Chatbot* ini dikembangkan dalam bentuk *website*, sehingga bisa diakses di berbagai *platform*, baik komputer, laptop, ataupun perangkat seluler.

3.2 Prosedur Penelitian

Penelitian ini bertujuan untuk menghasilkan sebuah produk *chatbot* untuk sistem informasi berbasis *large language model*. Terdapat beberapa model pengembangan yang umum diterapkan dalam metode penelitian D&D, salah satunya adalah model pengembangan *Agile*. Model pengembangan *Agile* merupakan metode pengembangan *software* yang mengutamakan kesamaan prinsip. Prinsip-prinsip ini yaitu mendorong pengembangan *software* dalam waktu singkat, dengan menuntut adaptasi yang cepat dari pengembang dalam menghadapi berbagai perubahan (Juman, 2023). Salah satu *framework* dalam model *agile* yaitu *Extreme Programming (XP)*. XP merupakan model pengembangan *agile method* yang ringan dengan 4 nilai utama, yaitu komunikasi, kesederhanaan, umpan balik, dan keberanian (Juman, 2023). Untuk mencapai tujuan dari penelitian ini, proses pengembangan *chatbot* akan menggunakan model pengembangan *agile* dengan *framework extreme programming*. Metode ini digunakan karena fleksibilitasnya dalam menangani suatu perubahan ketika pengembangan *chatbot* sedang berjalan. XP merupakan model yang mengutamakan kolaborasi yang baik antara

pengembang dan *user* (Shrivastava dkk., 2021), sehingga *chatbot* yang dikembangkan pada penelitian ini diharapkan dapat sesuai dengan harapan dan kebutuhan. Prosedur yang dilakukan pada penelitian ini terdiri dari tahap analisis, perancangan, pengembangan, pengujian, dan pelaporan seperti yang diperlihatkan pada gambar 3.1.



Gambar 3. 1 Prosedur Penelitian

Gambar tersebut menunjukkan lima tahap dalam penelitian ini, yaitu diawali dengan tahap konsep untuk mendapatkan analisis kebutuhan dan pengumpulan data. Tahap kedua yaitu perancangan di mana arsitektur dan berbagai diagram

dibuat sebelum memasuki tahap pengembangan. Tahap ketiga yaitu tahap pengembangan, yaitu mengimplementasikan seluruh diagram dan arsitektur yang telah dibuat sebelumnya. Tahap keempat adalah pengujian dengan menggunakan metode BERT *score*, uji manual kerelevanan jawaban dan *black box testing*. Tahap terakhir adalah tahap pelaporan yang akan dilakukan dalam bentuk penulisan skripsi.

3.2.1 Tahap Analisis

Tahap analisis adalah proses pertama yang bertujuan untuk mencapai tujuan spesifik yang ingin dicapai. Tahap konsep pada penelitian ini bertujuan untuk memperoleh gambaran komprehensif mengenai kebutuhan perusahaan, fitur dan fungsi *chatbot*, serta menentukan rencana pengembangan awal. Pada tahapan ini, dilakukan pengumpulan data yang dibutuhkan sebagai dasar pada pengembangan *chatbot* ini. Pengumpulan data dilakukan dengan dua metode, yaitu:

1. Wawancara

   Narasumber dalam wawancara ini yaitu *Technical Solution Specialist* PT. Bukit Makmur Mandiri Utama. Narasumber dipilih berdasarkan pemahaman mereka mengenai sistem informasi dan kebutuhan perusahaan. Wawancara dilakukan untuk mendapatkan beberapa informasi, di antaranya:

a. Penetapan manfaat serta tujuan dari dikembangkannya *chatbot* untuk sistem informasi berbasis *large language model*.
b. Penetapan target pengguna dari *chatbot* yang dikembangkan.
c. Penetapan konsep dari *chatbot* meliputi alur percakapan, basis pengetahuan, dll.

2. Studi Literatur

   Studi literatur dilakukan dengan mempelajari jenis dan struktur dokumen internal BUMA yang akan dijadikan basis pengetahuan dari *chatbot*. Selain itu, dilakukan juga kajian terhadap penelitian terdahulu yang relevan, khususnya mengenai penerapan *chatbot* untuk sistem informasi.

3.2.2 Tahap Perancangan

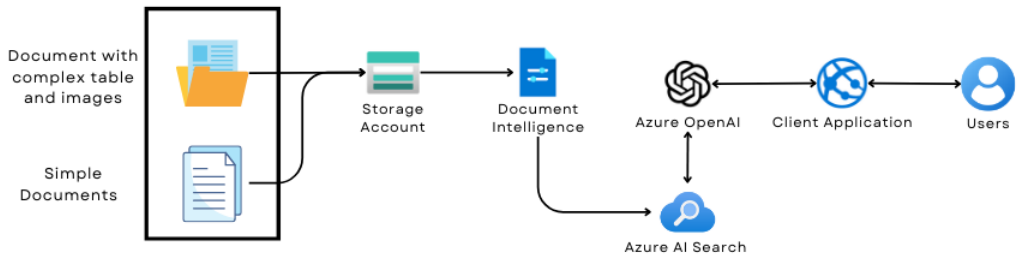
Tahapan perancangan dalam proses pengembangan *chatbot* ini memiliki peran penting dalam mendefinisikan pola logika dari *chatbot* yang akan dibuat. Desain yang terstruktur dengan baik akan mampu mengurangi dependensi antar

proses dalam sistem yang dibuat. Hal ini dilakukan agar apabila terjadi *bug* pada salah satu fitur, dampaknya tidak akan merambat dan mempengaruhi keseluruhan sistem.

Pada tahapan ini, proses perancangan *chatbot* dilakukan untuk sistem informasi. Tujuan utama dari *chatbot* ini adalah untuk mengintegrasikan beberapa azure *services*, agar *chatbot* dapat memproses, mencari, dan merespons pertanyaan pengguna dengan cepat dan akurat berdasarkan basis pengetahuan yang tersedia. Untuk membuat *chatbot* dapat berjalan dengan efisien dan akurat, berbagai azure *services* digunakan dan diintegrasikan ke dalam sistem *chatbot*, di antaranya Azure OpenAI, Azure AI Search, Azure Document Intelligence, dan Azure Storage Account.

3.2.2.1 Arsitektur *Chatbot*

Arsitektur chatbot akan menjelaskan bagaimana alur dari penggunaan chatbot yang dikembangkan. Gambar 3.2 merupakan arsitektur sistem *chatbot* secara keseluruhan.



Gambar 3. 2 Arsitektur *Chatbot*

Gambar tersebut memperlihatkan sistem *chatbot* yang dikembangkan. *Chatbot* dikembangkan dengan ekosistem Azure yang dirancang untuk dapat menangani pertanyaan ataupun permintaan karyawan yang membutuhkan informasi dari banyaknya dokumen yang tersedia.

1. Proses dimulai dengan pengunggahan dokumen ke dalam sistem. Dokumen tersebut diunggah melalui halaman admin yang dikhususkan untuk mengelola dokumen yang tersedia.

2. Dokumen-dokumen yang telah diunggah akan diproses oleh Azure Document Intelligence untuk menganalisis dan mengekstrak data-data pada dokumen ke dalam format yang dapat diproses oleh Azure AI Search.
3. Dokumen yang tersimpan selanjutnya akan di indeks oleh Azure AI Search dengan pendekatan berbasis vektor dan *keyword*. Hal ini dilakukan untuk menemukan informasi yang relevan berdasarkan konteks masukan dari pengguna. Hasil pengindeksan akan disimpan pada Azure AI Search untuk digunakan oleh Azure OpenAI.
4. Ketika pengguna memasukkan pertanyaan melalui *chatbot*, *query* tersebut akan dikirim ke Azure OpenAI. Azure OpenAI berperan sebagai inti dari sistem *chatbot*, yang dapat memahami dan memproses pertanyaan untuk memberikan jawaban yang sesuai.
5. Ketika pertanyaan pengguna membutuhkan pencarian pada dokumen, Azure OpenAI akan meminta informasi kepada Azure AI Search untuk mencari jawaban yang relevan. Namun, ketika pertanyaan tidak membutuhkan informasi dari dokumen, Azure OpenAI akan langsung memberikan jawaban tanpa menggunakan Azure AI Search.
6. Setelah jawaban yang relevan ditemukan melalui Azure AI Search, informasi tersebut akan diolah lebih lanjut oleh Azure OpenAI agar dapat menghasilkan jawaban yang natural kepada pengguna. Hasilnya akan dikirim kembali ke pengguna.

Pengguna berinteraksi dengan sistem melalui antarmuka *chatbot* yang dibangun dengan JavaScript (*frontend*) dan Python (*backend*). Masukkan pertanyaan adalah berupa teks. Sistem kemudian akan mencari informasi dokumen yang relevan yang terdapat pada Azure AI Search. Dokumen-dokumen yang ditemukan tersebut akan dianalisis oleh Azure OpenAI untuk menghasilkan jawaban yang akurat melalui proses penalaran berdasarkan informasi dalam potongan dokumen yang tepat.

3.2.2.1.2 Tipe Dokumen

Pada implementasi *chatbot* yang dikembangkan, terdapat dua tipe dokumen yang akan diproses oleh sistem, yaitu dokumen sederhana dan dokumen kompleks.

Dokumen-dokumen tersebut memiliki karakteristik yang berbeda, sehingga diperlukan metode tambahan untuk memproses dan memahami isi konten dari dokumen.

1. Dokumen Sederhana

   Dokumen sederhana merupakan dokumen yang memiliki struktur teks biasa tanpa elemen visual maupun format yang kompleks. Dokumen ini hanya terdiri dari teks linear, paragraf, atau poin-poin penting tanpa tabel yang rumit atau gambar. Proses ekstraksi data dari dokumen ini lebih mudah karena sistem hanya perlu membaca teks tanpa memperhatikan *layout* yang rumit.
2. Dokumen kompleks

   Dokumen kompleks merupakan dokumen dengan struktur dan komposisi yang lebih rumit. Dokumen ini bisa terdiri dari elemen-elemen seperti tabel, diagram, gambar, dll. Dokumen ini memerlukan proses analisis lebih canggih, yaitu dengan *Document Intelligence* agar informasi pada isi dokumen bisa dipahami dan diinterpretasikan dengan baik.

3.2.2.1.2 Azure OpenAI

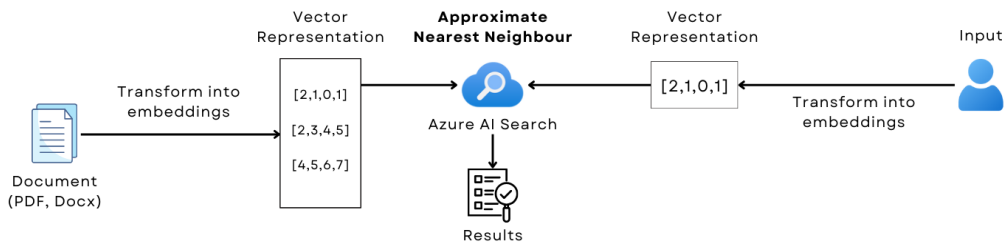
Azure OpenAI merupakan salah satu layanan azure berbasis AI yang memungkinkan integrasi model LLM buatan OpenAI. Dalam sistem *chatbot* ini, Azure OpenAI digunakan sebagai inti atau otak dari *chatbot* yang bertanggung jawab dalam memahami dan menghasilkan respons terhadap masukan pertanyaan pengguna. Azure OpenAI memiliki fungsi untuk memproses masukan pengguna, lalu menganalisis konteks dari pertanyaan pengguna, sehingga menghasilkan jawaban yang sesuai berdasarkan basis pengetahuan yang tersedia. Peran Azure OpenAI dalam *chatbot* ini di antaranya:

1. Model dari Azure OpenAI akan menerima masukan pertanyaan dari pengguna dan menganalisis teks tersebut untuk dipahami maksud dan konteks dari pertanyaan tersebut.
2. Selanjutnya, Azure OpenAI akan mengirimkan permintaan ke Azure AI Search untuk meminta informasi tambahan yang spesifik berdasarkan basis pengetahuan yang tersedia.

3. Setelah informasi didapatkan, model Azure OpenAI akan menggabungkan informasi yang diperoleh dengan pemahaman konteks dari masukan pengguna.
4. Azure OpenAI akan merangkai jawaban yang sesuai dengan konteks pengguna berdasarkan data yang diperoleh dari Azure AI Search dan mengirimkan kembali kepada pengguna melalui *chatbot*.

3.2.2.1.2 Azure AI Search

Azure AI Search menjadi layanan penting bagi aplikasi yang memerlukan pencarian yang canggih, seperti pencarian katalog produk, eksplorasi data dalam sebuah dokumen, dan bahkan sebagai sumber informasi untuk *chatbot*. Dengan Azure AI Search, pengguna dapat mengembangkan aplikasi yang mampu memahami maksud pengguna dan memberikan hasil pencarian yang relevan dan akurat. Dalam sistem *chatbot* ini, Azure AI Search memungkinkan *chatbot* dapat mengakses informasi yang spesifik dengan efisien dari data yang tersedia, seperti dokumen perusahaan, informasi operasional, dll. Penggunaan LLM dengan integrasi RAG dan *vector database* yang ada pada Azure AI Search telah terbukti mampu meningkatkan efisiensi *chatbot* dalam memberikan jawaban berbasis dokumen, seperti penelitian yang telah dilakukan oleh Pujiono dkk, di mana model GPT-4 yang dipakai menjadi lebih optimal dengan nilai *cosine similarity* rata-rata sebesar 0,404 (Pujiono dkk., 2024). Gambar 3.3 merupakan arsitektur dari cara kerja Azure AI Search**.**



Gambar 3. 3 Arsitektur Azure AI Search

Berikut penjelasan mengenai alur kerja sistem tersebut:

1. Sistem ini menerima input berupa dokumen seperti pdf dan docx.
2. Semua jenis data input ini ditransformasikan menjadi *embeddings*, yaitu representasi vektor numerik dari data tersebut.
3. Hasil dari proses embedding adalah serangkaian vektor, seperti [-2, -1, 0, 1], [2, 3, 4, 5], dan [4, 5, 6, 7]. Setiap vektor ini merepresentasikan fitur-fitur penting dari data input dalam bentuk numerik.
4. Vektor-vektor ini kemudian diproses menggunakan Azure AI Search, yang menerapkan algoritma K-Nearest Neighbour (KNN). Algoritma tersebut memungkinkan pencarian yang cepat terhadap vektor-vektor yang memiliki *similarity*.
5. Azure AI Search menghasilkan hasil pencarian berdasarkan *vector similarity*. Chatbot yang memanfaatkan RAG memungkinkan pencarian informasi yang relevan dari *knowledge base chatbot* menggunakan teknik *similarity search*, sehingga dapat menjawab pertanyaan dengan lebih baik (Anassai & Josaphat, 2024).
6. Hasil pencarian ini kemudian dikirim ke aplikasi atau antarmuka pengguna (UX) untuk ditampilkan.
7. Ketika pengguna melakukan query melalui antarmuka *chatbot*, query tersebut juga ditransformasikan menjadi sebuah embedding, contohnya [2, 3, 4, 5]. Vektor query tersebut kemudian akan dibandingkan dengan vektor-vektor yang tersedia pada Azure AI Search untuk menemukan hasil yang paling relevan.

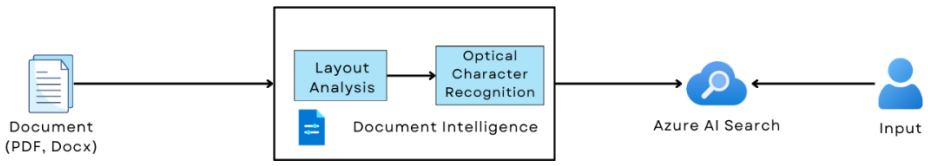
Peran Azure AI Search dalam *chatbot* ini di antaranya:

1. Melakukan indeksasi terhadap data-data yang telah diekstraksi oleh Azure Document Intelligence. Proses ini akan mengubah data tersebut menjadi struktur yang dapat dicari berdasarkan kata kunci atau pertanyaan tertentu.
2. Ketika *chatbot* menerima masukan pertanyaan, *chatbot* akan mengirimkan kueri pencarian ke Azure AI Search dan mencari *similarity* antara pertanyaan pengguna dan data yang tersedia.

3. Setelah Azure AI Search menemukan data yang relevan, hasil tersebut dikirim Kembali ke Azure OpenAI untuk diproses menjadi jawaban yang lebih komprehensif dan sesuai dengan kebutuhan pengguna.

3.2.2.1.3 Azure Document Intelligence

Azure Document Intelligence merupakan layanan Azure AI yang memungkinkan pengguna agar bisa membangun sebuah solusi pemrosesan dokumen yang cerdas. Document Intelligence bekerja dengan cara memanfaatkan teknologi AI untuk mengekstrak data dari dokumen secara otomatis, lalu memproses informasi yang tidak terstruktur agar menjadi data yang terstruktur. Azure Document Intelligence menganalisis tata letak dokumen, termasuk bagian *header*, *footer*, paragraf, dan struktur lainnya. Hal Ini memungkinkan pemahaman konteks yang lebih dalam dari dokumen yang diunggah. Gambar 3.4 memperlihatkan arsitektur dari Azure Document Intelligence.



Gambar 3. 4 Arsitektur Azure Document Intelligence

Azure Document Intelligence berperan dalam mengekstraksi data dari berbagai jenis dokumen dan mengubahnya menjadi informasi yang dapat dipahami oleh *azure service* yang lain. Proses yang dilakukan pada Azure Document Intelligence memanfaatkan algoritma Convolutional Neural Network (CNN), antara lain:

1. *Layout Analysis*

   Layout Analysis menjadi inti dari dalam proses identifikasi struktur dokumen. Dokumen akan dianalisis berdasarkan susunan elemen visual seperti teks, gambar, tabel, dll. Beberapa proses utama pada tahap ini yaitu:

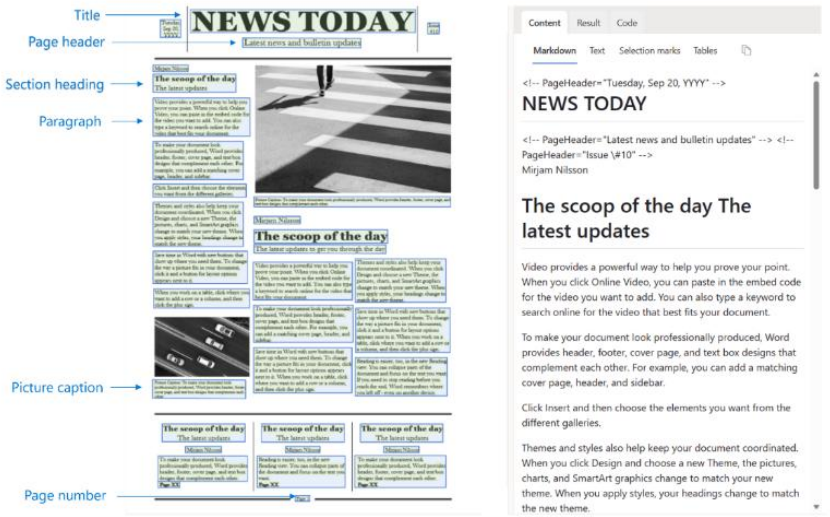
- Segmentasi halaman untuk mengelompokkan dokumen menjadi bagian-bagian yang lebih kecil berdasarkan jenis konten seperti judul, paragraf, gambar, dll.
- Mengklasifikasikan elemen-elemen pada dokumen seperti teks, gambar, tabel, dll, agar sistem dapat mengenali jenis informasi yang terdapat pada dokumen.
- Melakukan analisis terhadap hubungan antar elemen dalam dokumen, seperti judul, subjudul, konten utama, dll.

2. *Optical Character Recognition* (OCR)

   Setelah semua elemen dokumen teridentifikasi, OCR dilakukan untuk mengekstrak teks dari dokumen yang telah dianalisis. OCR didefinisikan sebagai algoritma yang digunakan untuk mengekstrak teks dari gambar, pdf, dan dokumen lainnya. Azure Document Intelligence menggunakan OCR berbasis *deep learning* untuk mendeteksi dan mengenali teks dari berbagai format dokumen, termasuk yang memiliki tata letak yang kompleks. Proses utama pada tahap OCR antara lain:

   - Sistem memisahkan setiap karakter dari gambar dokumen, baik dari teks ataupun elemen lain seperti huruf, angka, simbol, dll.
   - Setelah semua karakter diidentifikasi, karakter tersebut akan disusun menjadi sebuah kata, kalimat, atau paragraf, sesuai dengan apa yang ada pada dokumen asli.
   - *Output* dari proses ini adalah sebuah kumpulan teks yang dapat disalin dan akan diolah lebih lanjut oleh Azure AI Search.

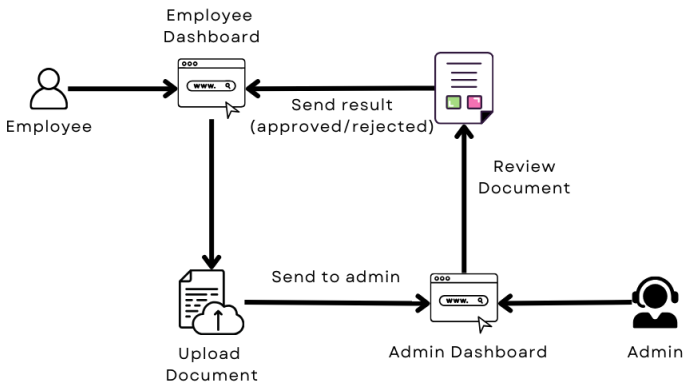
Contoh dokumen yang telah diproses pada Azure Document Intelligence dapat dilihat pada gambar 3.5.



Gambar 3. 5 Contoh dokumen yang telah diproses Document Intelligence (Sumber: learn.microsoft.com)

3.2.2.2 Arsitektur Web Panel Chatbot

Demi menjaga kualitas dan keamanan basis pengetahuan dari *chatbot*¸ fitur *attach file* secara langsung oleh karyawan tidak disediakan pada *chatbot*. Hal tersebut dilakukan untuk mencegah potensi penyalahgunaan oleh pengguna, seperti pengunggahan dokumen yang tidak relevan atau di luar lingkup perusahaan. Oleh karena itu, sistem ini akan dilengkapi dengan sebuah panel yang memungkinkan karyawan agar bisa mengajukan sebuah dokumen untuk dijadikan *knowledge chatbot* melalui proses yang terstruktur. Gambar 3.6 merupakan arsitektur untuk tahap-tahap pengajuan dokumen tersebut.



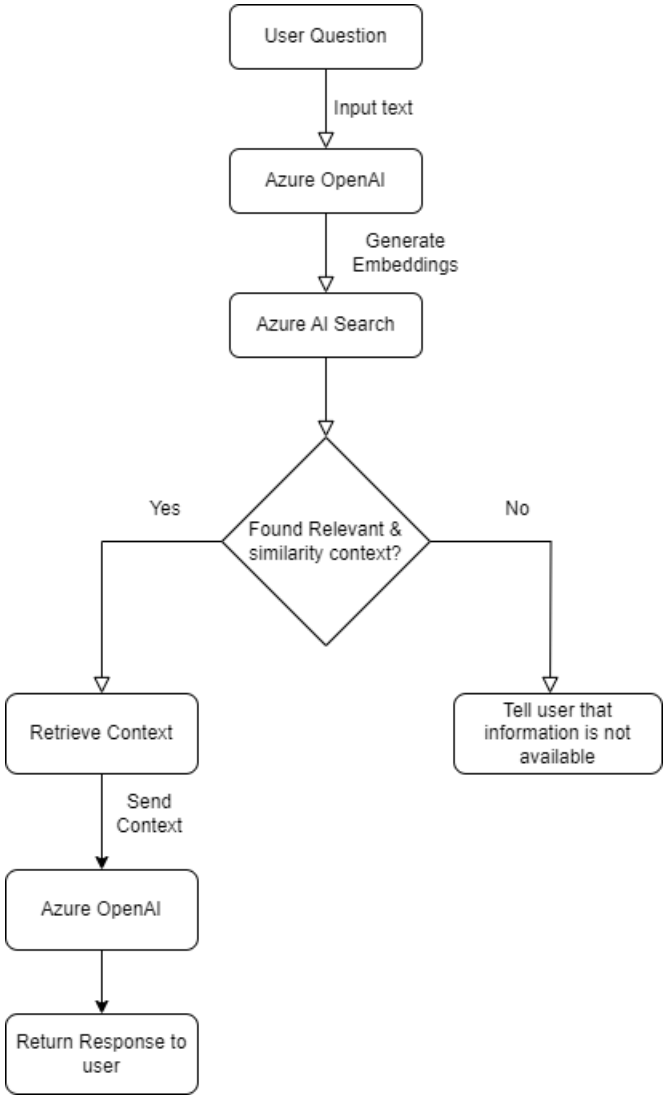
Gambar 3. 6 Arsitektur Panel *Chatbot*

Arsitektur tersebut dirancang agar memastikan dokumen yang ditambahkan ke dalam *knowledge chatbot* telah melewati proses validasi oleh admin.Alur dari sistem pengajuan dokumen adalah sebagai berikut:

1. Karyawan mengunggah sebuah dokumen melalui *dashboard* karyawan. *Dashboard* ini berfungsi sebagai *interface* utama bagi karyawan untuk mengajukan dokumen yang dianggap penting untuk ditambahkan ke *chatbot*. Dokumen yang telah diunggah akan terkirim kepada untuk dilakukan proses *review*.
2. Dokumen yang telah diunggah akan ditinjau oleh admin melalui dashboard admin. Admin akan memvalidasi dokumen berdasarkan relevansi dan kesesuaian dokumen dengan kebutuhan *knowledge base* dari *chatbot*.
3. Setelah dokumen telah di-review, hasil keputusan admin, baik itu *approved* atau *rejected* akan dikirim kepada user melalui *dashboard* karyawan.
4. Ketika dokumen disetujui, dokumen akan diproses lebih lanjut untuk ditambahkan ke dalam *knowledge base chatbot*.

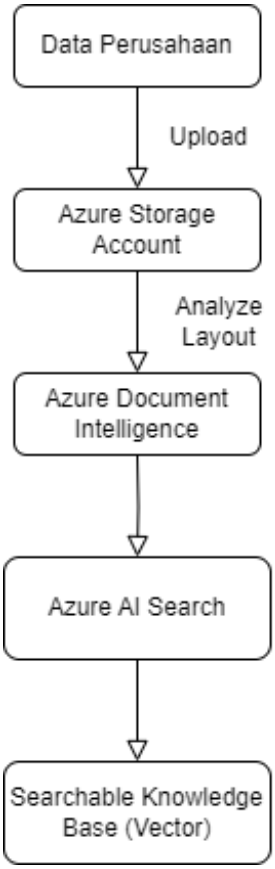
3.2.2.3 *Flowchart*

Pembuatan *Flowchart* merupakan langkah awal yang penting dalam pengembangan sebuah program komputer, layaknya *blueprint* bagi seorang insinyur (Ensmenger, 2016). *Flowchart* diartikan sebagai sebuah algoritma yang dibuat secara simbolik untuk merepresentasikan alur proses dengan jelas dan mudah dimengerti. Dalam pengembangan *chatbot* ini, *flowchart* akan menggambarkan bagaimana alur percakapan antara *chatbot* dan pengguna dari awal pengguna memasukkan pertanyaan hingga *chatbot* mengeluarkan respons yang relevan. Gambar 3.7 adalah flowchart dari alur chatbot. Gambar tersebut menunjukkan alur chatbot dari pengguna ketika memasukkan pertanyaan hingga jawaban yang dihasilkan chatbot. Ketika pengguna memasukkan kueri pertanyaan, pertanyaan tersebut akan diproses oleh model lalu model akan melakukan request terhadap Azure AI Search untuk mengecek apakah mempunyai informasi sesuai pertanyaan yang diajukan oleh pengguna. Jika informasi tersedia, maka informasi akan diambil oleh model dan diproses untuk menghasilkan jawaban yang natural untuk dikirim kembali kepada pengguna.



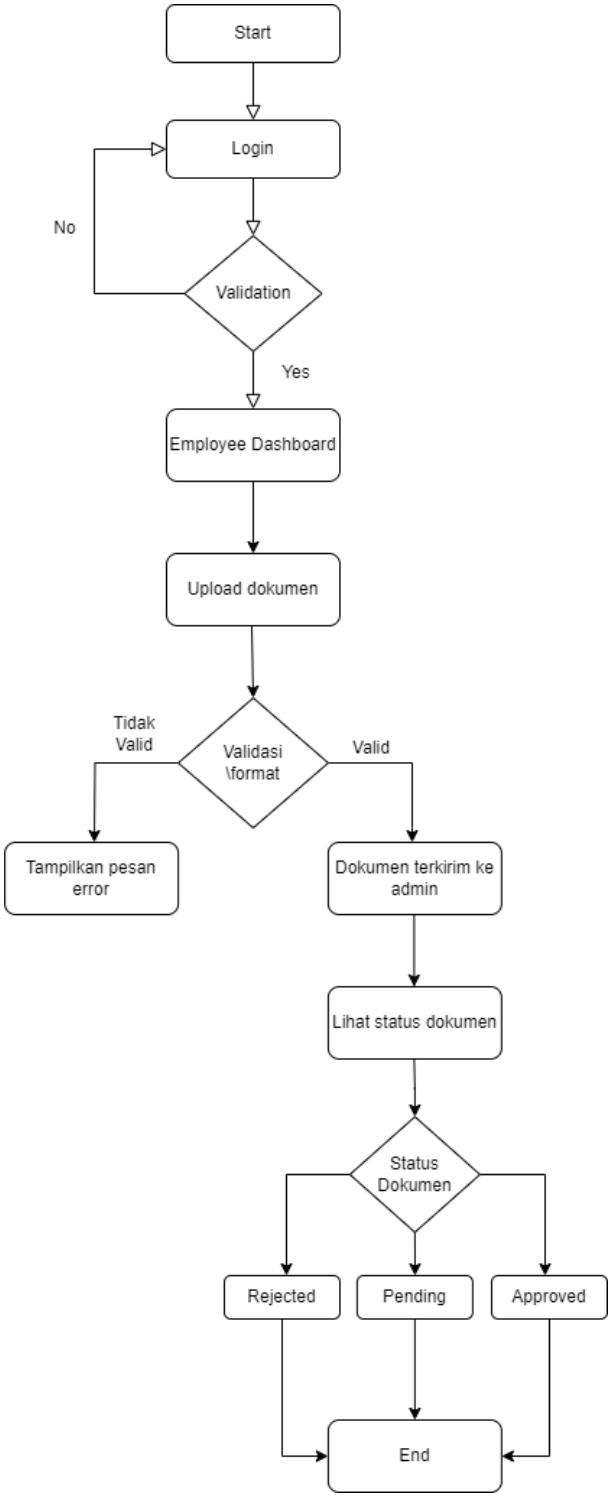
Gambar 3. 7 Flowchart *chatbot*

Gambar 3.8 adalah *flowchart* untuk menambahkan pengetahuan ke dalam *chatbot*. Gambar tersebut menunjukkan alur untuk menambahkan pengetahuan baru ke dalam *chatbot*. Dokumen akan diunggah ke dalam Azure Storage Account (Blob Storage), lalu setelah diunggah dokumen tersebut diproses oleh documen intelligence untuk mengekstrak layout dari dokumen. Selanjutnya, hasilnya akan diproses oleh Azure AI Search dengan diubah ke representasi vektor dan disimpan sebagai knowledge baru yang siap dipakai.



Gambar 3. 8 *Flowchart* untuk menambahkan *knowledge*

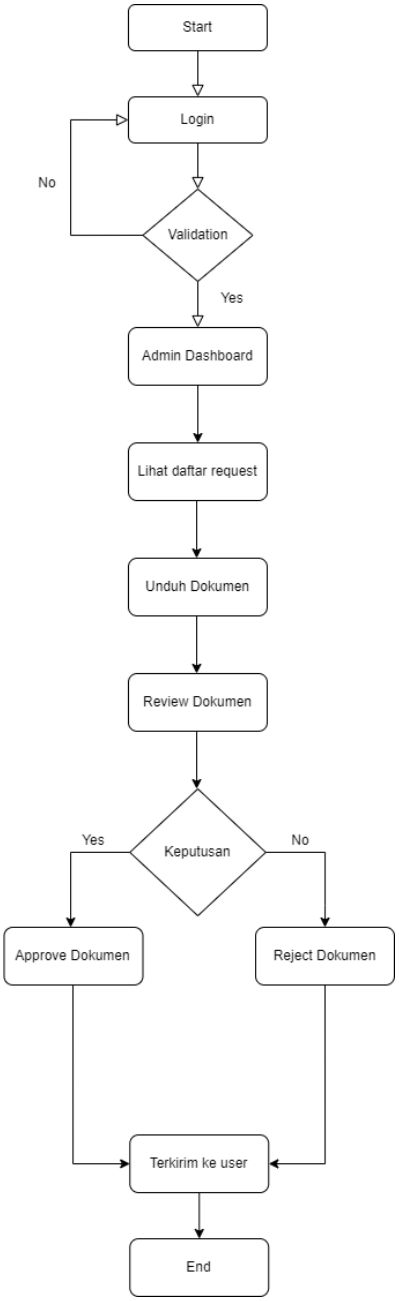
Gambar 3.9 menunjukkan *flowchart* pengguna dalam mengusulkan dokumen yang ingin ditambahkan ke dalam *knowledge chatbot* kepada admin. Gambar tersebut menunjukkan bahwa pengguna harus login ke panel chatbot, lalu mengunggah dokumen yang ingin diusulkan. Selanjutnya, dokumen yang berhasil dikirim akan berstatus pending dan akan berubah menjadi *approved* atau *rejected* sesuai keputusan admin.



Gambar 3. 9 *Flowchart* Sistem pengajuan dokumen untuk dijadikan *knowledge*

Gambar 3.10 menunjukkan flowchart dalam admin meninjau dokumen yang diusulkan user. Gambar tersebut memperlihatkan alur admin dalam meninjau dokumen yang diusulkan, yaitu dengan login terlebih dahulu lalu melihat daftar

request. Selanjutnya, admin akan mengunduh dokumen tersebut dan mengecek apakah layak atau tidak ditambahkan ke dalam *knowledge base chatbot*. Admin akan melakukan *approve* atau *reject* terhadap dokumen tersebut dan dikirim ke user yang mengusulkan.



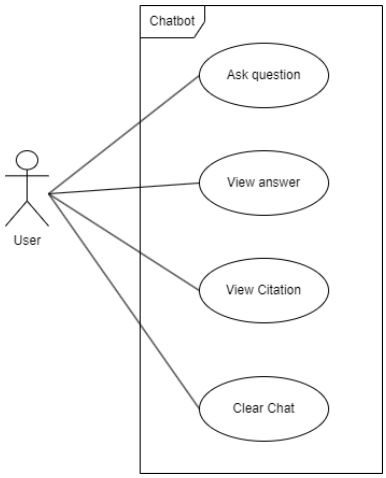
Gambar 3. 10 Flowchart untuk admin meninjau dokumen yang diunggah

3.2.2.4 *Unified Modelling Language*

Unified Modelling Language (UML) didefinisikan sebagai sebuah bahasa pemodelan grafis yang diciptakan oleh Object Management Group untuk memvisualisasikan suatu sistem perangkat lunak dengan orientasi objek (Pooley & King, 1999). Dalam penelitian ini, beberapa bentuk UML digunakan untuk merancang gambaran struktur dan perilaku sistem *chatbot* yang dikembangkan.

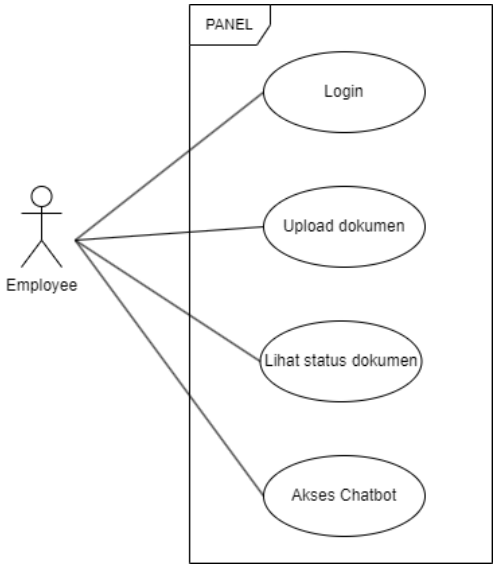
3.2.2.4.1 *Use Case Diagram*

Use case diagram menjadi bagian dari UML yang merepresentasikan sebuah sistem dari perspektif pengguna, yaitu aktor (Pooley & King, 1999). Fungsi utama dari *use case diagram* yaitu untuk membantu pengguna memahami cara kerja sistem dan nilai yang diberikan kepada pengguna, sehingga dapat merepresentasikan fungsionalitas setiap aktor dengan efektif. Diagram ini akan menjadi gambaran umum mengenai fitur-fitur *chatbot* yang bisa diakses oleh masing-masing pengguna. Gambar 3.11 memperlihatkan *use case* user pada *chatbot*. Gambar di atas menunjukkan bahwa pada chatbot, pengguna bisa memberikan pertanyaan, melihat jawabannya, dan melihat referensi dari jawaban tersebut. User juga bisa menghapus seluruh riwayat obrolan sebelumnya.



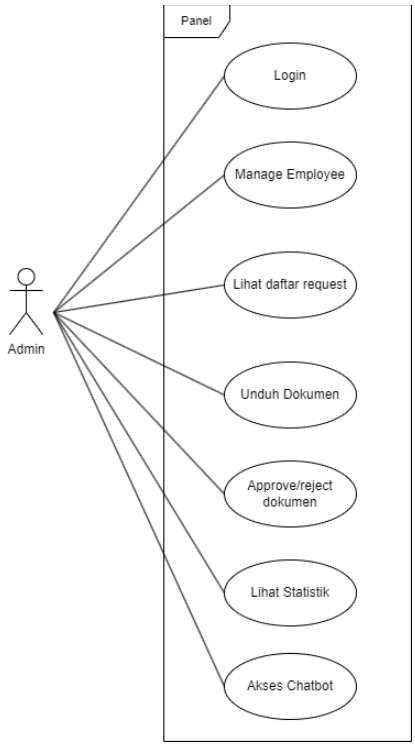
Gambar 3. 11 Use case *user* pada chatbot

Gambar 3.12 memperlihatkan use case *user/employee* pada panel *chatbot.* Gambar tersebut menunjukkan peran dari user pada panel chatbot, diantaranya user login terlebih dahulu, lalu dapat mengunggah dokumen untuk diusulkan, melihat status dokumen yang diusulkan, dan mengakses chatbot.



Gambar 3. 12 *Use case employee* pada panel *chatbot*

Gambar 3.13 menunjukkan use case admin pada panel *chatbot*. Gambar tersebut menunjukkan peran admin dalam panel chatbot, yaitu admin dapat login, *manage user*, melihat daftar pengusulan dokumen, mengunduh dokumen, *approve/reject* dokumen, melihat statistik pengusulan dokumen, dan mengakses chatbot.



Gambar 3. 13 Use case Admin pada panel chatbot

3.2.2.4.2 Skenario *Use Case*

Skenario *use case* dibuat untuk menjelaskan proses yang terjadi pada *use case diagram* secara deskriptif. Skenario ini akan menjelaskan proses yang terjadi antara aktor dan sistem *chatbot* pada setiap fitur, meliputi nama *use case*, aktor, tujuan, dan tabel alur *use case* yang berisi masukkan dari pengguna dan keluaran dari sistem. Tabel 3.1 memperlihatkan skenario dari use case *ask question* dan *view answer*.

Tabel 3. 1 Skenario *Use Case Ask Question dan View Answer*

Aktor	Sistem
1. User memasukkan pertanyaan dalam bentuk teks.	
	2. Model menganalisis *input.*
	3. Azure AI Search mencari konteks yang relevan.
	4. Azure OpenAI memproses jawaban.
	5. Chatbot mengeluarkan jawaban.
6. User menerima dan melihat jawaban.	

Tabel 3.2 di bawah menunjukkan skenario use case untuk melihat sitasi dari jawaban yang dihasilkan.

Tabel 3. 2 Skenario *Use Case View Citation*

Aktor	Sistem
1. User memasukkan pertanyaan dalam bentuk teks.	
	2. Model menganalisis *input.*
	3. Azure AI Search mencari konteks yang relevan.
	4. Azure OpenAI memproses jawaban.

Aktor	Sistem
	5. Chatbot mengeluarkan jawaban.
6. User menerima jawaban. 7. User menekan link *citation.*	
	8. Chatbot menampilkan bagian dokumen yang menjadi sumber jawaban.

Tabel 3.3 di bawah menunjukkan skenario use case user untuk mengunggah dokumen untuk diusulkan menjadi basis pengetahuan.

Tabel 3. 3 Skenario Use Case user/*employee* untuk unggah dokumen

Aktor	Sistem
1. User telah login pada panel chatbot. 2. User mengakses menu Make a Request. 3. User mengunggah dokumen dan memberi deskripsi keterangan dari dokumen yang diunggah.	
	4. Jika valid, sistem akan memberi tahu user bahwa dokumen berhasil diunggah dan terkirim kepada admin. 5. Jika tidak valid, Sistem akan memberi tahu bahwa dokumen gagal diunggah karena format tidak sesuai.

Tabel 3.4 di bawah menunjukkan skenario *use case* untuk melihat status dokumen yang telah diusulkan yang terdiri dari *pending*, *approved*, dan *rejected*.

Tabel 3. 4 Skenario Use case *user/employee* untuk melihat status dokumen

Aktor	Sistem
1. User telah login pada panel chatbot 2. User mengakses menu *My Requests*	
	3. Sistem akan menampilkan dokumen beserta status request nya apakah *pending*, *approved* atau *rejected*

Tabel 3.5 di bawah menunjukkan skenario *use case* admin untuk manage user, di antaranya untuk mengubah *role* dan menghapus *user*.

Tabel 3. 5 Skenario *Use case* admin untuk *manage user*

Aktor	Sistem
1. Admin telah login pada panel chatbot 2. Admin mengakses menu Manage User	
	3. Sistem akan menampilkan list akun *user* yang terdaftar
4. Admin menekan tombol “edit” untuk melakukan *update* role *employee* (user/admin).	
	5. Sistem akan memperbarui data *user* tersebut pada *database*.
6. Admin menekan tombol “delete” untuk menghapus user tertentu.	
	7. Sistem akan menghapus data yang dihapus admin pada database.

Tabel 3.6 di bawah menunjukkan skenario *use case* admin untuk melihat daftar dokumen yang di-*request* oleh user.

Tabel 3. 6 Skenario Use case admin untuk melihat daftar *request* dokumen

Aktor	Sistem
1. Admin telah login pada panel *chatbot* 2. Admin mengakses menu Manage request	
	3. Sistem akan menampilkan list dokumen request yang dikirim oleh *user* beserta statusnya (pending, rejected, dan approved).

Tabel 3.7 di bawah menunjukkan skenario *use case* admin untuk mengunduh dokumen yang di-*request* oleh *user*.

Tabel 3. 7 Skenario *Use case admin* untuk mengunduh dokumen

Aktor	Sistem
1. Admin telah login pada panel chatbot 2. Admin mengakses menu Manage request	
	3. Sistem akan menampilkan list dokumen request yang dikirim oleh *employee* beserta statusnya (pending, rejected, dan approved).
4. Admin menekan tombol “unduh” untuk mengunduh dokumen yang di *request* oleh *employee* untuk ditinjau lebih lanjut.	

Tabel 3.8 di bawah menunjukkan skenario *use case* admin untuk melakukan *approve* dan *reject* terhadap dokumen yang di-*request*.

Tabel 3. 8 Skenario Use case admin untuk *approve* atau *reject* dokumen

Aktor	Sistem
1. Admin telah login pada panel chatbot 2. Admin mengakses menu Manage request	
	3. Sistem akan menampilkan list dokumen request yang dikirim oleh *employee* beserta statusnya (pending, rejected, dan approved).
4. Admin menekan tombol *approve* untuk menyetujui dokumen untuk ditambahkan ke dalam knowledge 5. Admin menekan tombol reject untuk menolak dokumen untuk ditambahkan ke dalam knowledge.	
	6. Sistem akan memperbarui status dokumen pada database dan mengirim hasilnya kepada *employee*.

Tabel 3.9 di bawah menunjukkan skenario *use case* admin untuk melihat statistik total dokumen yang disetujui dan ditolak.

Tabel 3. 9 Skenario Use case admin untuk melihat statistik

Aktor	Sistem
1. Admin telah login pada panel chatbot 2. Admin mengakses menu reports	
	3. Sistem akan menampilkan statistik dokumen yang disetujui dan ditolak.

Tabel 3.10 di bawah menunjukkan skenario *use case* pengguna untuk mengakses *chatbot* melalui *sidebar* yang ada di panel *chatbot*.

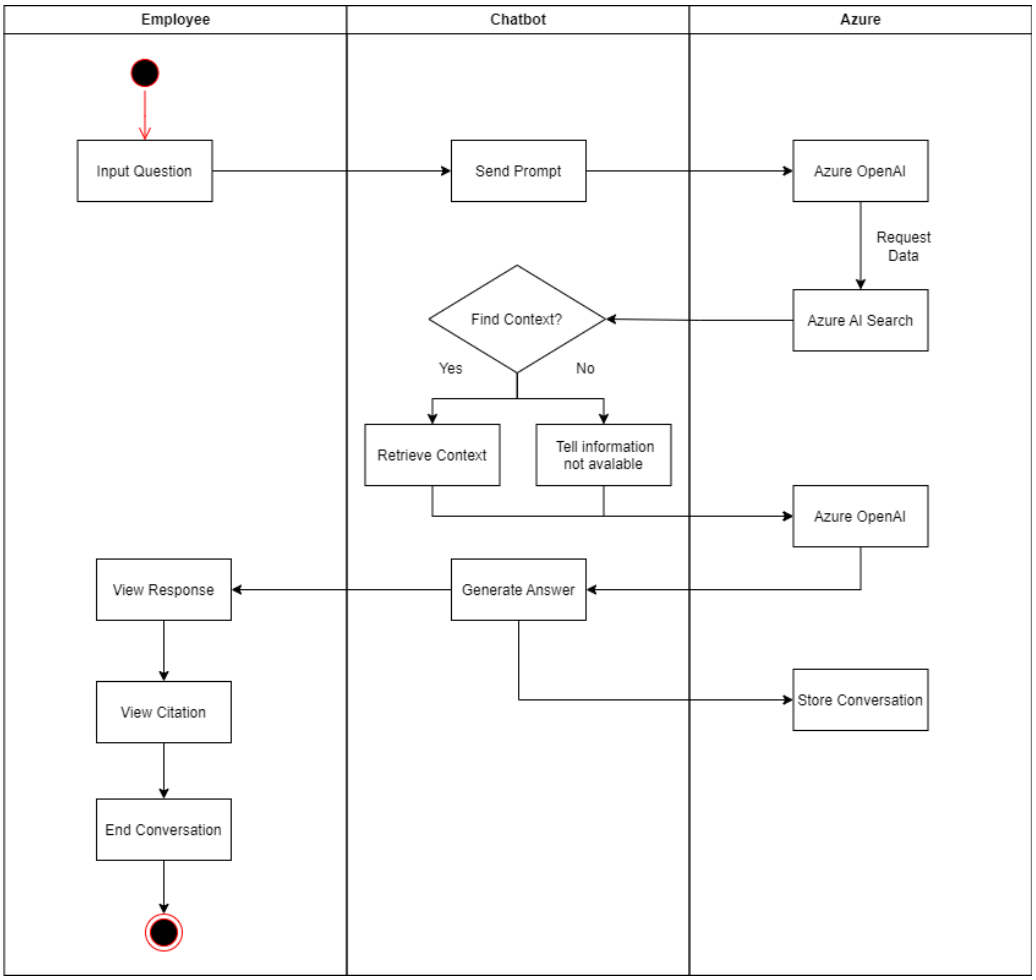
Tabel 3. 10 Skenario Use case pengguna untuk mengakses chatbot

Aktor	Sistem
1. User/admin telah login pada panel *chatbot* 2. User/admin mengakses menu *Chatbot Service*	
	3. Sistem akan mengarahkan pengguna ke laman *chatbot*.

3.2.2.4.3 *Activity Diagram*

Activity diagram didefinisikan sebagai tahapan yang bukan hanya menunjukkan bagaimana prosesnya berjalan, tetapi juga menunjukkan keterkaitan antar proses tersebut (Pooley & King, 1999). *Activity diagram* akan merepresentasikan alur kerja dari tiap *use case* secara visual. Dalam penelitian ini, *activity diagram* dibuat untuk menjelaskan langkah-langkah yang dilakukan oleh pengguna saat berinteraksi dengan *chatbot*, mulai dari masukan awal hingga keluaran akhir. Gambar 3.14 di bawah menunjukkan activity diagram dari chatbot yang dikembangkan. Gambar tersebut menunjukkan alur dari penggunaan chatbot di mana terdiri dari tiga komponen, yaitu employee/user, chatbot, dan azure. User akan mengkueri sebuah pertanyaan, lalu pertanyaan tersebut akan diterima oleh model yang ada di Azure OpenAI dan dari topik pertanyaan tersebut akan di-*request*

kepada Azure AI Search untuk mengecek informasi yang relevan yang ada pada *vector database.* Selanjutnya, informasi akan dikirim ke model dan diproses untuk dikirim kembali kepada pengguna.



Gambar 3. 14 *Activity Diagram*

3.2.3 Tahap Pengembangan

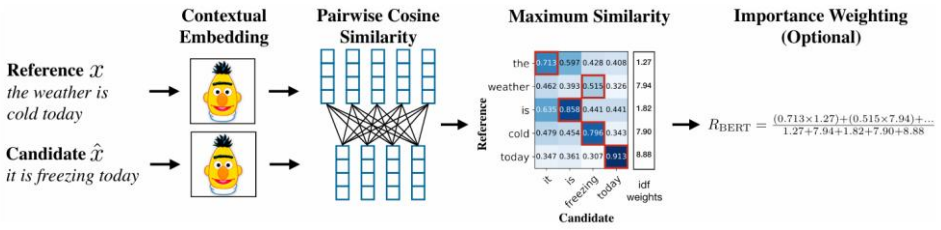
Tahap *pengembangan* adalah tahapan pengimplementasian seluruh konsep dan desain yang telah dikumpulkan dan dibuat pada tahap sebelumnya. Pada tahapan ini, desain yang telah dibuat sebelumnya diimplementasikan ke dalam bentuk kode program menggunakan teknologi dan bahasa pemrograman yang telah ditentukan.

3.2.4 Tahap Pengujian

Tahap pengujian menjadi tahapan yang sangat penting untuk memastikan bahwa *chatbot* yang dikembangkan berjalan dengan baik sesuai fungsi dan *requirement* yang ditentukan di awal.

3.2.4.1 Pengujian BERT *Score*

BERT Score menjadi sebuah metode pengujian yang sering digunakan untuk mengevaluasi kemampuan sebuah sistem pembuatan teks (Zhang dkk., 2019). Cara kerja dari BERT *Score* tidak sama dengan metode pengujian pada umumnya. BERT *score* tidak menganalisis kesamaan kata dari dua buah kalimat, melainkan menganalisis nilai kesamaan semantik dari sebuah kalimat. Artinya, BERT *score* lebih berfokus dalam menganalisis makna dari dua buah kalimat yang dibandingkan, yaitu kalimat yang dibuat oleh model dan kalimat referensi (*ground truth*) yang ada pada dokumen. Gambar 3.15 merupakan alur komputasi dari BERT *Score* dalam menghitung nilai *recall* yang dilambangkan dengan RBERT.



Gambar 3. 15 Arsitektur pengujian BERT *Score* (Zhang dkk., 2019)

Pada gambar tersebut, kalimat yang menjadi *ground truth* dilambangkan dengan $\varkappa$ dan kalimat yang merupakan hasil dari model dilambangkan dengan ẋ. Dari dua kalimat tersebut, akan dilakukan *contextual embedding* untuk merepresentasikan token pada tiap kata dan melakukan *cosine similarity* untuk menganalisis kesamaan konteks dari tiap kata. Setelah itu, skor atau nilai tertinggi dari perhitungan *maximum similarity* akan diambil dan disertakan bobot skornya. Kalimat dari *ground truth* akan dipecah menjadi $\varkappa$ = (x1, x2, .., xk) dan begitu pun kalimat yang di-*generate* oleh model akan dipecah menjadi ẋ = (ẋ1, ẋ2, .., ẋk). Nilai kesamaan dari tiap token $\varkappa$ pada token ẋ merupakan nilai dari *recall* dan nilai kesamaan dari tiap token ẋ pada token $\varkappa$ merupakan nilai *precision.*

a. Recall

Nilai Recall adalah proporsi dari contoh positif yang sukses diklasifikasikan dengan benar sebagai positif. Berikut persamaan untuk menghitung recall dengan *BERT score*:

$$R_{\text{BERT}} = \frac{1}{|x|} \sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} \mathbf{x}_i^\top \hat{\mathbf{x}}_j \qquad (1)$$

Keterangan:

R_{BERT} = Nilai Recall

$\varkappa$ = Token *ground truth*

$\hat{x}$ = Token hasil model

x_i = urutan vektor x

$\hat{x}_j$ = urutan vektor $\hat{x}$

$\sum_{x_i \in x}$ = jumlah x_i yang ada di dalam x

$\overset{MAX}{\hat{x}_j \in \hat{x}}$ = nilai maksimal $\hat{x}_j$ yang ada di dalam $\hat{x}$

${x_i}^T \hat{x}_j$ = *cosine similarity* x dan $\hat{x}$

b. Precision

Nilai *precision* merupakan proporsi dari contoh positif yang sukses diklasifikasikan dengan benar dari semua contoh yang diberi label positif. Berikut persamaan untuk menghitung *precision* BERT *score*:

$$P_{\text{BERT}} = \frac{1}{|\hat{x}|} \sum_{\hat{x}_j \in \hat{x}} \max_{x_i \in x} \mathbf{x}_i^\top \hat{\mathbf{x}}_j \qquad (2)$$

Keterangan:

P_{BERT} = *Precision BERT score*

x = token *ground truth*

$\hat{x}$ = token hasil model

x_i = urutan vektor x

$\hat{x}_j$ = urutan vektor $\hat{x}$

$\sum_{x_i \in x}$ = jumlah x_i yang ada di dalam x

$\overset{MAX}{\hat{x}_i \in x}$ = nilai maksimal $\hat{x}_i$ yang ada di dalam x

${x_i}^T \hat{x}_j$ = *cosine similarity* x dan $\hat{x}$

c. F1 Score

F1-score merupakan nilai rata-rata dari metrik *precision* dan *recall*. Berikut persamaan untuk menghitung *F1-score*:

$$F_{\text{BERT}} = 2\frac{P_{\text{BERT}} \cdot R_{\text{BERT}}}{P_{\text{BERT}} + R_{\text{BERT}}} \quad (3)$$

Keterangan:

F_{BERT} = *F1 Score BERT score*

P_{BERT} = *Precision BERT score*

R_{BERT} = Nilai Recall

3.2.4.2 Pengujian Manual Kerelevanan Jawaban *Chatbot*

Pada tahap ini, dilakukan pengujian secara manual untuk menganalisis tingkat kerelevanan jawaban yang dihasilkan oleh *chatbot* dengan jawaban pada dokumen yang direferensikan. Hasilnya diklasifikasi ke dalam tiga kategori, yaitu sangat relevan, relevan, dan tidak relevan. Penjelasan lebih lanjut terkait tiga kategori tersebut adalah sebagai berikut.

1. Sangat Relevan, berarti jawaban yang dihasilkan *chatbot* sepenuhnya sesuai dengan jawaban yang ada pada dokumen dan informasi yang diberikan akurat, lengkap, dan tidak mengandung informasi tambahan yang tidak diperlukan.
2. Relevan, berarti jawaban yang dihasilkan *chatbot* sesuai dengan jawaban yang ada pada dokumen, tetapi pada jawaban tersebut terdapat informasi yang terlewat atau berlebihan.
3. Tidak Relevan, berarti jawaban yang dihasilkan *chatbot* sebagian besar salah atau tidak sesuai dengan jawaban yang ada pada dokumen.

3.2.4.3 Pengujian *Black Box*

Black box testing adalah metode pengujian perangkat lunak yang berfokus pada fungsionalitas aplikasi tanpa menganalisis kode internalnya. Sesuai namanya, pengujian ini mengibaratkan aplikasi sebagai "kotak hitam" yang tidak terlihat isinya. Penguji hanya melakukan pengujian melalui *input* yang diberikan dan *output* yang diharapkan berdasarkan spesifikasi kebutuhan (Nidhra & Dondeti, 2012).

Pengujian *black box* dilakukan untuk menguji fungsionalitas dari sistem *chatbot* yang dikembangkan untuk memastikan bahwa sistem *chatbot* bekerja sesuai dengan spesifikasi yang telah dirancang, sehingga pengalaman pengguna tetap optimal. Tabel 3.11 memperlihatkan skenario pengujian *black box* yang akan dilakukan.

Tabel 3. 11 Skenario Pengujian *Black Box*

No	*Test Case*	Skenario	Hasil yang diharapkan
1	Login	Email Tidak diisi	Pesan error: “Enter a valid email address”
2	Login	Memasukkan password yang salah	Pesan error: “Your account or password is incorrect”
3	Login	Login dengan email diluar domain yang ditentukan	Pesan error: “Cannot access the application”
4	Logout	User menekan button logout	Menampilkan pesan “you signed out of yout account”
5	Pengusulan dokumen	File yang diunggah bukan pdf, docx, atau txt	Pesan error: “Jenis file tidak diizinkan”
6	Pengusulan dokumen	Deskripsi dokumen kosong	Pesan error: "Please fill out this field.”
7	Pengusulan dokumen	Mengunggah dokumen dengan format yang benar dan deskripsi yang diisi	Pengusulan dokumen sukses
8	Manage Document	User mengusulkan dokumen baru	Pengusulan terkirim ke admin
9	Manage Document	Admin mengunduh dokumen yang diusulkan	Dokumen berhasil diunduh

No	*Test Case*	Skenario	Hasil yang diharapkan
10	Manage Document	Admin menekan button approve	Dokumen berhasil disetujui
11	Manage Document	Admin menekan button reject	Dokumen berhasil ditolak
12	Manage Document	Admin menekan button delete	Dokumen berhasil dihapus
13	Manage Document	Admin menekam button approve atau reject	Muncul notifikasi pada dashboard user
14	Manage user	Admin mengubah role pengguna	Muncul notifikasi “Pengguna berhasil diperbarui”
15	Manage user	Admin menghapus pengguna	Muncul pop up konfirmasi dan notifikasi “User berhasil dihapus”
16	*Chatbot*	User menginputkan pertanyaan pada text box	Pesan terkirim dan muncul balasan dari chatbot
17	*Chatbot*	User menekan button reference	Menampilkan dokumen referensi dari jawaban
18	*Chatbot*	User menekan tombol clear chat	Riwayat chat terhapus
19	*Chatbot*	User menekan button panel	Berpindah ke halaman panel chatbot

3.2.5 Tahap Pelaporan

Setelah seluruh tahapan sebelumnya selesai dilakukan, tahap terakhir adalah tahap pelaporan Tahapan ini dilakukan dengan penyusun laporan hasil penelitian yang dibuat dalam bentuk penyusunan skripsi. Pada tahap pelaporan, seluruh hasil penelitian didokumentasikan dengan jelas dan terstruktur yang mencakup metode latar belakang, kajian pustaka, metode penelitian yang digunakan, temuan penelitian, dan kesimpulan serta rekomendasi dari penelitian ini. Tahap pelaporan

ini dilakukan untuk memastikan bahwa hasil penelitian dapat menjadi referensi bagi penelitian selanjutnya atau pengembangan lebih lanjut di masa depan.

3.3 Analisis Data

3.3.1 Analisis Data Pengujian *BERT score*

Data yang dihasilkan dari pengujian dengan menggunakan *BERT Score* adalah berupa nilai *precision, recall, dan F1 score* yang menunjukkan skor relevansi antara jawaban yang diberikan dan jawaban sebenarnya dari dokumen yang direferensikan. Setiap nilai dari *precision, recall, dan F1 score* akan ditotalkan, lalu dihitung rata-rata dari nilai *precision, recall*, dan *F1 score* secara keseluruhan, sehingga dari ketiga nilai tersebut dapat dilihat apakah chatbot memiliki performa yang memuaskan atau tidak.

3.3.2 Analisis Data Pengujian Manual Kerelevanan Jawaban *Chatbot*

Hasil pengujian ini akan dianalisis untuk mengevaluasi kelayakan *chatbot* berdasarkan kesesuaian antara *input* yang diberikan dan *output* yang dihasilkan. Analisis akan melibatkan beberapa proses, yaitu:

1. Memeriksa apakah jawaban yang dihasilkan *chatbot* sudah sesuai dengan yang informasi yang ada pada dokumen.
2. Menghitung persentase kerelevanan keseluruhan jawaban dari total pengujian yang dilakukan dengan rumus persamaan sebagai berikut:

$$\mathrm{TK} = \frac{(SR \times 1) + (\mathrm{R} \times 0.5) + (TR \times 0)}{\mathrm{T}} \times 100\% \qquad (4)$$

Keterangan:

TK = Tingkat Kerelevanan
SR = Jumlah jawaban “Sangat relevan”
R = Jumlah jawaban “Relevan”
TR = Jumlah jawaban “Tidak relevan”
T = Total Pertanyaan

3.3.3 Analisis data Pengujian *Black Box*

Pengujian *black box* dilakukan untuk mengevaluasi fungsionalitas sistem *chatbot* yang telah dikembangkan. Pengujian ini tidak mencangkup analisis

performa atau kualitas jawaban *chatbot*, melainkan hanya untuk memastikan semua fitur berfungsi sesuai dengan yang diharapkan. Hasil Pengujian akan disajikan dalam bentuk tabel yang berisi tiap skenario pengujian yang dilakukan. Analisis akan melibatkan beberapa proses, yaitu:

1. Memastikan bahwa setiap fitur pada sistem *chatbot* bekerja sesuai dengan yang diharapkan.
2. Mencatat setiap hasil yang tidak sesuai dengan yang diharapkan
3. Hasil pengujian digunakan untuk menyimpulkan apakah sistem telah memenuhi spesifikasi fungsional atau memerlukan perbaikan pada fitur tertentu.