PERANCANGAN CHATBOT BANTUAN HUKUM PERSONAL “OSOORA” BERBASIS WEB DAN GENERATIVE ARTIFICIAL INTELLIGENCE

**TUGAS AKHIR**

Disusun oleh:

Bakti Parningotan Marbun

181111025



**PROGRAM STUDI INFORMATIKA**

**FAKULTAS SAINS DAN TEKNOLOGI**

**UNIVERSITAS BHINNEKA NUSANTARA**

**2025**



PERANCANGAN CHATBOT BANTUAN HUKUM PERSONAL “OSOORA” BERBASIS WEB DAN GENERATIVE ARTIFICIAL INTELLIGENCE

**TUGAS AKHIR**

Sebagai salah satu syarat untuk

memperoleh gelar sarjana

pada Program Studi Informatika

Disusun oleh:

Bakti Parningotan Marbun

181111025



**PROGRAM STUDI INFORMATIKA**

**FAKULTAS SAINS DAN TEKNOLOGI**

**UNIVERSITAS BHINNEKA NUSANTARA**

**2025**

**PERNYATAAN**

**ORISINALITAS TUGAS AKHIR**

Saya yang bertandatangan dibawah ini:

Nama : Bakti Parningotan Marbun

NRP : 181111025

Program Studi : Informatika

 Fakultas : Sains dan Teknologi

Jenjang Studi : S1

Dengan ini saya menyatakan bahwa:

1. Tugas akhir ini murni ide, rumusan dan penelitian sendiri, tanpa bantuan dari pihak manapun selain Dosen Pembimbing.
2. Tugas akhir ini belum pernah digunakan untuk memperoleh gelar sarjana, baik di Universitas Bhinneka Nusantara atau di perguruan tinggi lain.
3. Tugas akhir ini tidak memuat karya atau pendapat yang ditulis atau diterbitkan oleh pihak ketiga, kecuali secara tertulis dengan mencantumkan sebagai acuan dalam naskah dengan menyebutkan nama pengarang dan mencantumkan dalam daftar pustaka.

Demikian pernyataan ini dibuat dengan sesungguhnya dan jika dikemudian hari terbukti ada unsur-unsur plagiasi, maka saya bersedia menerima sanksi akademik yakni pencabutan gelar yang sudah diberikan melalui karya tulis ini, dan sanksi lainnya sesuai dengan norma yang ada di perguruan tinggi.

Malang, 10 Juni 2025

Yang menyatakan,

Meterai

Bakti Parningotan Marbun

181111025

**TUGAS AKHIR BERJUDUL**

PERANCANGAN CHATBOT BANTUAN HUKUM PERSONAL “OSOORA” BERBASIS WEB DAN GENERATIVE ARTIFICIAL INTELLIGENCE

Disusun oleh:

Bakti Parningotan Marbun

181111025

Telah dipertahankan dalam Sidang Tugas Akhir

pada tanggal …

dan dinyatakan telah memenuhi syarat untuk diterima

|  |  |
| --- | --- |
| KOMISI SIDANG,  Daniel Rudiaman S., S.T., M.Kom.  Ketua Sidang/Pembimbing Utama | KOMISI PENGUJI,  <Nama Lengkap>  Ketua Penguji |
| Mukhlis Amien, M.Kom  Co. Pembimbing | <Nama Lengkap>  Anggota Penguji I |
|  | <Nama Lengkap>  Anggota Penguji II |
| Mengesahkan  Dekan Fakultas Sains dan Teknologi  Daniel Rudiaman S., S.T., M.Kom. | Mengetahui  Kaprodi Informatika  Adnan Zulkarnain, S.Kom., M.M.S.I. |
|  | |

**PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR**

**UNTUK KEPENTINGAN AKADEMIS**

Sebagai sivitas akademika Universitas Bhinneka Nusantara, saya yang bertanda tangan di bawah ini:

Nama : Bakti Parningotan Marbun

NRP : 181111025

Program Studi : Informatika

Fakultas : Sains dan Teknologi

Jenjang : S1

Jenis Karya : Pengembangan

Dengan ini saya menyatakan bahwa,

Demi pengembangan ilmu pengetahuan, setuju untuk memberikan Hak Bebas Royalti Non-Eksklusif kepada Universitas Bhinneka Nusantara atas karya ilmiah saya yang berjudul:

PERANCANGAN CHATBOT BANTUAN HUKUM PERSONAL ”OSOORA” BERBASIS WEB DAN GENERATIVE ARTIFICIAL INTELLIGENCE

Dengan hak bebas lisensi ini, Universitas Bhinneka Nusantara berhak untuk menyimpan, mentransfer/memformat, mengelola, memelihara, dan mempublikasikan proyek yang telah selesai dalam format database dan tetap mencantumkan nama saya sebagai penulis dan pemegang hak cipta.

Demikian pernyataan ini dibuat dengan sebenar-benarnya.

Malang, 10 Juni 2025

Yang menyatakan,

Bakti Parningotan M.

181111025

# ABSTRAK

Bakti Parningotan Marbun, 2025. **Perancangan Chatbot Bantuan Hukum Personal ”Osoora” Berbasis Web dan Generative Artificial Intelligence**. Tugas Akhir, Program Studi Informatika (S1), Universitas Bhinneka Nusantara, Pembimbing: Daniel Rudiaman Sijabat, Co. Pembimbing: Mukhlis Amien.

Kata kunci: Kecerdasan Buatan, Kecerdasan Buatan Generatif, Retrieval Augmented Generation, Chatbot, Bantuan Hukum

Penelitian ini bertujuan untuk merancang dan mengembangkan chatbot bantuan hukum personal bernama “OSOORA” berbasis web yang memanfaatkan teknologi Generative Artificial Intelligence (GenAI), khususnya model bahasa besar (LLM) Qwen 2.5 72B Instruct. Kebutuhan akan akses informasi hukum yang cepat dan mudah menjadi latar belakang utama, mengingat kompleksitas sistem hukum yang seringkali sulit dipahami oleh masyarakat awam. Metode yang digunakan meliputi studi literatur, pengumpulan basis pengetahuan hukum, perancangan arsitektur sistem menggunakan Node.js dan Express, serta implementasi antarmuka pengguna berbasis HTML/CSS/JavaScript. Pengujian fungsional dan pengujian akurasi LLM dilakukan untuk memastikan “OSOORA” mampu memberikan jawaban yang relevan dan akurat terhadap pertanyaan-pertanyaan hukum. Hasil penelitian menunjukkan bahwa “OSOORA” berhasil dirancang dan diimplementasikan sebagai platform yang user-friendly, mampu memproses pertanyaan pengguna secara efisien dan memberikan informasi hukum yang komprehensif, sehingga meningkatkan aksesibilitas terhadap bantuan hukum.

# ABSTRACT

Bakti Parningotan Marbun, 2025. **Design of a Personal Legal Assistance Chatbot “Osoora” Based on Web and Generative AI.** Final Project, Study Program Informatics (S1), Universitas Bhinneka Nusantara, Advisor 1 : Daniel Rudiaman Sijabat, Advisor 2 : Mukhlis Amien.

Keyword: Generative AI, Artificial Intelligence, Retrieval Augmented Generation, Chatbot, Legal Assistant

This research aims to design and develop a web-based personal legal aid chatbot named “OSOORA” leveraging Generative Artificial Intelligence (GenAI) technology, specifically the Qwen 2.5 72B Instruct Large Language Model (LLM). The need for quick and easy access to legal information serves as the primary background, given the complexity of legal systems often difficult for laypersons to comprehend. Methods employed include literature review, legal knowledge base collection, system architecture design using Node.js and Express, and user interface implementation based on HTML/CSS/JavaScript. Functional testing and LLM accuracy testing were conducted to ensure "OSOORA" could provide relevant and accurate answers to legal queries. The research results indicate that "OSOORA" was successfully designed and implemented as a user-friendly platform, capable of efficiently processing user questions and providing comprehensive legal information, thereby improving accessibility to legal aid.

# KATA PENGANTAR

Puji syukur kehadirat Tuhan Yang Maha Esa atas rahmat dan karunia-Nya, sehingga Tugas Akhir berjudul “PERANCANGAN CHATBOT BANTUAN HUKUM PERSONAL “OSOORA” BERBASIS WEB DAN GENERATIVE ARTIFICIAL INTELLIGENCE” ini dapat terselesaikan. Tugas Akhir ini disusun sebagai salah satu syarat untuk memperoleh gelar Sarjana pada Program Studi Informatika, Universitas Bhinneka Nusantara.

Penulis mengucapkan terima kasih yang sebesar-besarnya kepada Bapak Daniel Rudiaman Sijabat, dan Bapak Mukhlis Amien selaku Dosen Pembimbing atas bimbingan dan arahan yang tak ternilai. Apresiasi juga disampaikan kepada seluruh dosen dan staf Program Studi Informatika, serta orang tua dan rekan-rekan yang telah memberikan dukungan selama proses penyusunan Tugas Akhir ini.

Penulis menyadari Tugas Akhir ini masih memiliki kekurangan. Oleh karena itu, segala saran dan masukan yang membangun akan sangat penulis hargai. Semoga Tugas Akhir ini dapat memberikan manfaat dan kontribusi bagi perkembangan ilmu pengetahuan.

Malang, 10 Juni 2025

Penulis

# DAFTAR ISI

**Halaman**

[ABSTRAK vii](#_Toc86748907)

[ABSTRACT viii](#_Toc86748908)

[KATA PENGANTAR ix](#_Toc86748909)

[DAFTAR ISI x](#_Toc86748910)

[DAFTAR TABEL xii](#_Toc86748911)

[DAFTAR GAMBAR xiii](#_Toc86748912)

[DAFTAR SEGMEN PROGRAM xiv](#_Toc86748913)

[DAFTAR LAMPIRAN xv](#_Toc86748914)

[BAB I PENDAHULUAN 1](#_Toc86748915)

[1.1 Latar Belakang 1](#_Toc86748916)

[1.2 Rumusan Masalah 1](#_Toc86748917)

[1.3 Tujuan 1](#_Toc86748918)

[1.4 Manfaat 2](#_Toc86748919)

[1.5 Batasan Masalah 2](#_Toc86748920)

[1.6 Metodologi Penelitian 2](#_Toc86748921)

[1.6.1. Tempat dan Waktu Penelitian 2](#_Toc86748922)

[1.6.2. Bahan dan Alat Penelitian 3](#_Toc86748923)

[1.6.3. Pengumpulan Data dan Informasi 3](#_Toc86748924)

[1.6.4. Analisis Data 3](#_Toc86748925)

[1.6.5. Prosedur Penelitian 4](#_Toc86748926)

[1.7 Sistematika Penulisan 4](#_Toc86748927)

[BAB II TINJAUAN PUSTAKA 5](#_Toc86748928)

[2.1 Penelitian Terdahulu 5](#_Toc86748929)

[2.2 Teori Terkait 6](#_Toc86748930)

[BAB III ANALISIS DAN PERANCANGAN 7](#_Toc86748931)

[3.1 Analisis 7](#_Toc86748932)

[3.1.1 Identifikasi Masalah 7](#_Toc86748933)

[3.1.2 Pemecahan Masalah 7](#_Toc86748934)

[3.2 Perancangan 8](#_Toc86748935)

[3.2.1 Perancangan Sistem 8](#_Toc86748936)

[3.2.2 Perancangan Data 8](#_Toc86748937)

[3.2.3 Perancangan User Interface / *Mock-up* aplikasi 8](#_Toc86748938)

[3.3 Rancangan Pengujian 8](#_Toc86748939)

[BAB IV PEMBAHASAN 9](#_Toc86748940)

[4.1 Gambaran Umum Obyek Penelitian 9](#_Toc86748941)

[4.2 Implementasi 9](#_Toc86748942)

[4.3 Uji Coba 11](#_Toc86748943)

[BAB V PENUTUP 12](#_Toc86748944)

[5.1 Kesimpulan 12](#_Toc86748945)

[5.2 Saran 12](#_Toc86748946)

[DAFTAR PUSTAKA 13](#_Toc86748947)

[LAMPIRAN 14](#_Toc86748948)

# DAFTAR TABEL

[Tabel 1.1 Waktu Penelitian 3](#_Toc86749097)

[Tabel 2.1 Tabel Penelitian Terdahulu 5](#_Toc86749098)

[Tabel 3.1 Template Tabel Bab 3 7](#_Toc86749099)

[Tabel 4.1 Template Tabel Bab 4 9](#_Toc86749100)

# DAFTAR GAMBAR

[Gambar 1.1 Diagram Alir Penelitian 4](#_Toc86749127)

[Gambar 2.1 Template Gambar Bab 2 6](#_Toc86749128)

[Gambar 3.1 Context Diagram 7](#_Toc86749129)

[Gambar 3.2 Use Case Diagram 7](#_Toc86749129)

[Gambar 4.1 Template Gambar Bab 4 9](#_Toc86749130)

# DAFTAR SEGMEN PROGRAM

[Segmen Program 2.1 Contoh Segmen Program Browser Use .10](#_Toc86749406)

Segmen Program 4.1 Template Segmen Program 28

# DAFTAR LAMPIRAN

[Lampiran 1 Surat Keputusan Tugas Akhir (SK TA) 15](#_Toc86749431)

[Lampiran 2 Biodata Penulis 16](#_Toc86749432)

[Lampiran 3 Hasil Cek Plagiarisme 18](#_Toc86749433)

[Lampiran 4 Hasil Wawancara 19](#_Toc86749434)

[Lampiran 5 Listing Program/Coding 20](#_Toc86749435)

# BAB I PENDAHULUAN

## Latar Belakang

Di era digital saat ini, sektor hukum di Indonesia menghadapi tantangan besar dalam menyediakan akses informasi hukum yang mudah dipahami oleh masyarakat luas. Meskipun pemerintah telah mengembangkan sistem peradilan elektronik (e-Court) sebagai upaya modernisasi layanan hukum, kenyataannya masih banyak warga yang kesulitan dalam memanfaatkannya. Hal ini disebabkan oleh kompleksitas regulasi, rendahnya literasi digital, serta minimnya panduan hukum yang ramah bagi masyarakat awam.

Objek penelitian ini adalah pengembangan chatbot berbasis web menggunakan teknologi Generative Artificial Intelligence (AI) yang dirancang untuk membantu masyarakat dalam memahami dan menyelesaikan permasalahan hukum perdata, khususnya dalam bidang administrasi kependudukan (seperti KTP) dan kepemilikan tanah. Topik ini dipilih karena permasalahan tersebut merupakan isu hukum yang paling sering dihadapi masyarakat, namun sering kali tidak terselesaikan secara efektif karena keterbatasan akses informasi hukum yang praktis dan dapat dipahami.

Penelitian ini bersifat terapan, dengan tujuan untuk meningkatkan literasi hukum masyarakat dan menyediakan akses praktis terhadap informasi hukum melalui pemanfaatan teknologi. Strategi pencapaian tujuan ini dilakukan dengan cara merancang sebuah chatbot hukum berbasis Generative AI yang mampu memberikan informasi hukum secara otomatis, berbasis web, dan mudah digunakan oleh masyarakat umum. Selain itu, pendekatan ini juga mencakup integrasi data hukum perdata yang relevan serta pengujian akurasi respons chatbot melalui simulasi kasus-kasus sederhana.

Secara teoritis, penelitian ini didukung oleh teori literasi hukum, human-computer interaction, serta konsep kecerdasan buatan generatif yang telah berkembang dalam bidang teknologi informasi. Generative AI memungkinkan mesin untuk memproduksi teks yang menyerupai jawaban dari pakar hukum, sehingga dapat digunakan untuk menjawab pertanyaan hukum secara langsung kepada pengguna dalam bentuk dialog.

Namun, dalam proses pengembangan dan penerapannya, chatbot hukum ini juga berpotensi menghadapi sejumlah hambatan. Hambatan tersebut antara lain adalah risiko interpretasi hukum yang kurang akurat oleh AI, keterbatasan data hukum terstruktur sebagai sumber pelatihan sistem, serta tingkat kepercayaan masyarakat terhadap teknologi sebagai pengganti interaksi langsung dengan ahli hukum.

Dengan memperhatikan berbagai aspek teoritis, faktual, teknis, dan normatif tersebut, penelitian ini diharapkan dapat memberikan kontribusi nyata dalam memperluas akses keadilan dan meningkatkan literasi hukum melalui inovasi teknologi yang inklusif dan mudah diakses

## Rumusan Masalah

Permasalahan dalam penelitian ini berangkat dari kesenjangan antara kebutuhan masyarakat dalam memahami hukum perdata secara cepat dan mudah, dengan keterbatasan akses terhadap sumber hukum yang dapat dipahami oleh masyarakat awam. Teknologi Generative AI telah berkembang pesat, namun pemanfaatannya dalam bidang bantuan hukum, khususnya untuk persoalan dokumen KTP dan kepemilikan tanah, masih sangat terbatas. Selain itu, belum banyak sistem chatbot hukum berbasis web yang mampu memberikan solusi secara efektif dan etis. Oleh karena itu, penelitian ini merumuskan permasalahan bagaimana merancang sistem berbasis Generative AI yang mampu mengisi kekosongan ini, serta mengkaji tantangan teknis dan etis yang timbul dalam penerapannya.

## Tujuan

Penelitian ini bertujuan untuk merancang sebuah sistem chatbot bantuan hukum perdata berbasis web yang memanfaatkan teknologi Generative AI. Tujuan utama dari penelitian ini adalah untuk meningkatkan literasi hukum masyarakat Indonesia, khususnya dalam bidang hukum perdata yang berkaitan dengan dokumen KTP dan kepemilikan tanah. Selain itu, penelitian ini juga bertujuan untuk mempermudah masyarakat awam dalam memahami permasalahan hukum perdata yang sering kali dianggap rumit, serta menyediakan akses informasi hukum yang cepat, akurat, dan mudah dipahami melalui platform digital berbasis kecerdasan buatan.

## Manfaat

Manfaat dari penelitian ini diharapkan dapat dirasakan oleh berbagai pihak, antara lain:

• Penulis: Penulis dapat meningkatkan pemahaman dan implementasi sistem yang berkaitan dengan Generative AI, khususnya dalam konteks penerapannya dalam bidang hukum perdata. Penelitian ini juga memberikan kontribusi dalam pengembangan teknologi chatbot berbasis web yang dapat dimanfaatkan oleh masyarakat luas.

• Civitas Akademika: Penelitian ini diharapkan menjadi acuan bagi penelitian lebih lanjut mengenai penerapan Generative AI dalam bidang hukum. Selain itu, penelitian ini juga dapat memberikan wawasan baru tentang penerapan teknologi AI dalam meningkatkan akses terhadap pemahaman hukum.

• Masyarakat: Chatbot ini akan memberikan media atau alat yang mudah diakses oleh masyarakat untuk memahami kompleksitas hukum perdata, khususnya masalah yang terkait dengan KTP dan kepemilikan tanah. Dengan demikian, masyarakat dapat lebih mudah mengetahui hak-hak mereka dalam menghadapi persoalan hukum.

• Profesional Hukum: Profesional hukum dapat menggunakan chatbot ini sebagai asisten hukum yang dapat memberikan jawaban awal terhadap pertanyaan hukum yang sering ditanyakan, sehingga mereka dapat lebih fokus pada masalah hukum yang lebih kompleks.

## Batasan Masalah

Dalam penelitian ini, batasan masalah yang akan dibahas mencakup:

• Hukum Perdata: Hukum perdata yang dimaksud dalam penelitian ini adalah masalah-masalah yang berkaitan dengan identitas pribadi (KTP) dan kepemilikan tanah, dua topik yang sering menjadi masalah hukum di Indonesia.

• Fokus pada Teknologi Web dan AI: Penelitian ini akan difokuskan pada perancangan sistem chatbot berbasis web dengan teknologi Generative AI, khususny yang dapat memberikan penjelasan dan solusi terkait masalah hukum perdata yang telah disebutkan.

* Semua jawaban yang dihasilkan berasal dari AI dan .
* Aplikasi ini terbatas pada penggunaan tech stack yang akan disebutkan pada bab 3.

• Penerapan Chatbot untuk Masyarakat Umum: Sistem ini dirancang untuk dapat diakses oleh masyarakat umum yang memiliki keterbatasan dalam pemahaman hukum, sehingga chatbot akan berfokus pada penyederhanaan bahasa hukum agar lebih mudah dimengerti.

• Aplikasi hanya berfokus pada fase development, bukan production.

## Metodologi Penelitian

Penelitian ini menggunakan metode Adaptive Waterfall, yaitu modifikasi dari model Waterfall tradisional yang memungkinkan terjadinya penyesuaian dan umpan balik di tengah proses pengembangan, tanpa sepenuhnya menghilangkan struktur linier dari tahapan-tahapan utama. Pendekatan ini dipilih karena sesuai dengan kebutuhan penelitian yang bersifat terstruktur namun tetap fleksibel dalam menghadapi perubahan selama implementasi sistem.

Model Adaptive Waterfall tetap mengikuti urutan tahapan utama dalam pengembangan perangkat lunak, yaitu: analisis kebutuhan, perancangan, implementasi, pengujian, dan pemeliharaan. Namun, berbeda dengan model Waterfall murni yang bersifat kaku dan tidak menerima perubahan setelah satu tahap diselesaikan, pendekatan ini memberi ruang untuk revisi atau iterasi terbatas di antara tahapan tertentu, terutama antara desain dan implementasi, serta antara implementasi dan pengujian.

Menurut Pressman (2014), pendekatan seperti ini dikenal sebagai “Modified Waterfall” atau “Flexible Waterfall”, yaitu model yang tetap menjaga struktur tahapan tetapi mengizinkan adanya feedback loop dan perbaikan di fase-fase tertentu. Hal ini juga sejalan dengan temuan dari Munassar dan Govardhan (2010) yang menyebut bahwa dalam praktiknya, banyak pengembang perangkat lunak memodifikasi model Waterfall agar lebih adaptif terhadap perubahan kebutuhan.

Dengan demikian, penggunaan Adaptive Waterfall pada penelitian ini bertujuan untuk:

* Memastikan tahapan pengembangan dilakukan secara sistematis dan terdokumentasi.
* Memberikan fleksibilitas dalam pengembangan sistem agar dapat beradaptasi terhadap evaluasi berkala.
* Menjaga validitas akademik laporan melalui pelacakan proses yang jelas.

### Tempat dan Waktu Penelitian

Penelitian ini akan dilaksanakan di lingkungan Universitas Bhinneka Nusantara (UBHINUS), Malang, Jawa Timur. Waktu pelaksanaan penelitian dimulai pada bulan September 2023 dan direncanakan selesai pada bulan Juni 2025. Kegiatan penelitian meliputi tahap persiapan, perancangan sistem, implementasi chatbot berbasis web, serta pengujian dan evaluasi sistem.

Tabel 1.1 Waktu Penelitian

| Kegiatan | Tahun | | | | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2023 | | | | 2024 | | | | 2025 | | | |
| Q1 | Q2 | Q3 | Q4 | Q1 | Q2 | Q3 | Q4 | Q1 | Q2 | Q3 | Q4 |
| Perancangan Aplikasi |  |  |  |  |  |  |  |  |  |  |  |  |
| Pemilihan Model LLM |  |  |  |  |  |  |  |  |  |  |  |  |
| Pengumpulan Basis Pengetahuan (Knowledge Base) dan data |  |  |  |  |  |  |  |  |  |  |  |  |
| Pengujian LLM |  |  |  |  |  |  |  |  |  |  |  |  |
| Pengujian Aplikasi |  |  |  |  |  |  |  |  |  |  |  |  |
| Deployment |  |  |  |  |  |  |  |  |  |  |  |  |

### Bahan dan Alat Penelitian

Bahan yang digunakan dalam penelitian ini meliputi:

Bahasa Pemrograman: JavaScript (ES6), digunakan untuk pengembangan frontend dan backend sistem chatbot.

Framework: Node.js v18.x dan Express.js untuk server-side chatbot.

Model AI: Alibaba Cloud Model Studio dengan generative model berbasis Kowledge Base

Database: MongoDB Atlas (Cluster Tier M0, Shared Cluster, MongoDB v8.0).

Alat yang digunakan dalam penelitian ini adalah:

Laptop ASUS ROG G14 dengan spesifikasi: AMD Ryzen 7 5800HS, RAM 16GB, SSD 512GB, OS Windows 10 64-bit. Laptop ini digunakan untuk mengembangkan, menguji, dan mengakses sistem selama proses penelitian.

### Pengumpulan Data dan Informasi

Pengumpulan data dalam penelitian ini dilakukan dengan pendekatan teknologi terkini menggunakan metode Browser Use, sebuah teknik eksplorasi web berbasis agen kecerdasan buatan (AI Agent) yang mampu menavigasi, membaca, dan mengekstrak informasi dari berbagai situs web secara otomatis dan kontekstual. Metode ini dipilih karena memiliki keunggulan dalam menjelajahi struktur situs yang kompleks serta mampu menangani elemen-elemen dinamis yang umumnya tidak dapat dijangkau melalui metode scraping konvensional.

Secara teknis, metode ini dijalankan melalui integrasi antara LangChain Agent dan Playwright Chromium, yang memungkinkan proses eksplorasi web dilakukan dengan instruksi berbasis bahasa alami. Peneliti cukup memberikan perintah seperti “temukan diskusi tentang permasalahan kepemilikan tanah di forum daring Indonesia” atau “ambil komentar terkait KTP di media sosial,” dan agen akan menjalankan serangkaian tindakan mulai dari membuka situs, melakukan pencarian, berinteraksi dengan tombol dinamis, hingga mengekstrak data yang relevan.

Pengumpulan data ini bersifat semi-otomatis, di mana AI Agent bertindak sebagai eksekutor utama, sementara peneliti tetap melakukan pengawasan untuk menjamin kualitas dan relevansi data yang diperoleh. Validasi manual dilakukan pada sebagian sampel data untuk memastikan bahwa informasi yang diambil sesuai dengan kebutuhan analisis serta mencerminkan realitas sosial yang menjadi fokus penelitian.

Proses pengumpulan data dijadwalkan berlangsung pada kuartal kedua tahun 2025. Kegiatan ini melibatkan beberapa tahap, yaitu penyusunan skenario eksplorasi, eksekusi proses Browser Use, dan penyimpanan data hasil ekstraksi. Setiap sesi pengumpulan dilakukan secara terstruktur, dengan estimasi waktu tiga jam per hari, guna menjaga konsistensi dan kualitas data.

Dengan pendekatan ini, penelitian diharapkan memperoleh data kualitatif yang kaya konteks, langsung dari sumber-sumber daring yang mencerminkan persepsi dan permasalahan hukum perdata yang dihadapi oleh masyarakat Indonesia secara aktual, khususnya yang berkaitan dengan identitas kependudukan dan kepemilikan tanah.

### Analisis Data

Metode analisis data dalam penelitian ini menggabungkan pendekatan analisis deskriptif, kategorisasi tematik, dan benchmarking hukum berbasis Retrieval-Augmented Generation (RAG) menggunakan Precision, Recall dan F1 Score. Pendekatan ini digunakan untuk menganalisis persepsi publik terhadap hukum perdata melalui data yang diperoleh dari media sosial yang akan dijelaskan di bab 3 dan benchmarking hukum di bab 4

**1. Analisis Deskriptif dan Kategorisasi Tematik**

Data hasil scraping dari media sosial akan dianalisis secara deskriptif untuk memperoleh informasi kuantitatif seperti jumlah komentar, distribusi per topik, dan frekuensi kata. Selain itu, dilakukan kategorisasi komentar berdasarkan isu utama seperti kepemilikan tanah, administrasi KTP, atau kasus sengketa perdata lainnya. Kategorisasi ini membantu dalam mengelompokkan persepsi publik terhadap tema hukum tertentu.

**2. Benchmarking Hukum Berbasis Retrieval-Augmented Generation (RAG)**

Benchmarking dilakukan untuk menilai tingkat kesesuaian antara Jawaban dari Model dan Jawaban yang benar.

Analisis Evaluatif: Hasil evaluasi ini dianalisis untuk mengidentifikasi kesenjangan (gap) antara pemahaman hukum publik dan landasan hukum resmi. Kesenjangan ini akan menjadi dasar dalam menarik simpulan terhadap tingkat literasi hukum masyarakat serta efektivitas teknologi AI dalam menjembatani informasi hukum.

Melalui metode ini, penelitian tidak hanya memetakan opini publik, tetapi juga menilai secara objektif seberapa besar opini tersebut selaras dengan prinsip dan norma hukum yang berlaku. Pendekatan ini memberikan kontribusi metodologis baru dalam penggabungan antara teknologi AI dan evaluasi legal reasoning secara sistematis.

### Prosedur Penelitian

Penelitian ini dilakukan dengan pendekatan rekayasa perangkat lunak berbasis Generative AI, dengan tahapan sebagai berikut:

**1. Studi Pendahuluan**

Peneliti melakukan kajian terhadap literasi hukum di Indonesia, khususnya pada permasalahan hukum perdata seperti administrasi KTP dan kepemilikan tanah. Selain itu, dilakukan studi literatur terhadap teknologi chatbot, Generative AI, dan RAG.

**2. Pengumpulan Data**

Data dikumpulkan menggunakan metode scraping Browser Use melalui integrasi LangChain Agent dan Playwright Chromium. AI Agent mengekstraksi

komentar dan diskusi hukum di media sosial dan forum daring, untuk digunakan sebagai dataset dan konteks.

**3. Analisis Data**

Analisis dilakukan dalam tiga lapis:

Deskriptif: kuantifikasi jumlah komentar dan topik.

Tematik: klasifikasi masalah hukum perdata.

**4. Perancangan Sistem**

Tahapan ini mencakup:

Perancangan Arsitektur Sistem (context diagram, deployment, use case)

Desain Basis Data (ERD MongoDB)

Mockup dan alur interaksi UI

**5. Implementasi Sistem**

Sistem dibangun dengan stack berikut:

Frontend: HTML, CSS, JavaScript

Backend: Node.js + Express

AI Model: Alibaba Cloud Model Studio (Generative + RAG)

Database: MongoDB Atlas

**6. Pengujian**

Pengujian dilakukan dengan metode:

Blackbox Testing: mengevaluasi fungsionalitas chatbot.

User Acceptance Test (UAT): mengukur kepuasan dan efektivitas penggunaan chatbot.

**7. Evaluasi dan Dokumentasi**

Hasil sistem dievaluasi berdasarkan:

Akurasi jawaban AI (berdasarkan perbandingan hukum)

Kemudahan penggunaan (dari respon UAT)

START

Pendahuluan

Pengumpulan Data

Analisis Data

Perancangan Sistem

Implementasi Sistem

Pengujian

Evaluasi Sistem

END

Gambar 1.1 Diagram Alir Penelitian

## Sistematika Penulisan

Sistematika penulisan tugas akhir ini disusun sebagai berikut:

**Bab I Pendahuluan**

Bab ini berisi latar belakang, rumusan masalah, tujuan, manfaat, batasan masalah, metode penelitian, serta sistematika penulisan.

**Bab II Tinjauan Pustaka**

Bab ini menguraikan penelitian terdahulu dan landasan teori yang relevan dengan topik penelitian.

**Bab III Analisis dan Perancangan**

Bab ini membahas analisis permasalahan, identifikasi solusi, perancangan sistem, data, model AI, serta perancangan pengujian.

**Bab IV Hasil dan Pembahasan**

Bab ini menyajikan gambaran umum objek penelitian, implementasi sistem, serta hasil uji coba.

**Bab V Penutup**

Bab ini berisi kesimpulan dari hasil penelitian dan saran untuk penelitian selanjutnya.

Bagian akhir laporan dilengkapi dengan Daftar Pustaka, Daftar Lampiran, dan Daftar Kode Program sebagai pendukung keseluruhan isi tugas akhir.

# BAB II TINJAUAN PUSTAKA

## Penelitian Terdahulu

**2.1.1 Intelligent Legal Assistant: An Interactive Clarification System for Legal Question Answering**

Peneliti: Rujing Yao et al. (2025)

Penelitian ini mengembangkan sistem tanya jawab hukum berbasis LLM yang mengintegrasikan klarifikasi interaktif untuk memahami kebutuhan hukum pengguna secara lebih spesifik. Sistem akan meminta pengguna untuk memilih lokasi geografis yang relevan, dan kemudian menghasilkan pertanyaan klarifikasi secara otomatis bila input awal tidak lengkap.

Tujuannya adalah meningkatkan akurasi dan kepuasan pengguna dalam memahami dan menjawab pertanyaan hukum.

Hasil evaluasi dari 100 pengguna menunjukkan bahwa sistem ini mendapatkan skor akurasi 4.8 dan skor kepuasan 4.8 dari skala 1–5, serta 90% pengguna memilih sistem ini dibandingkan GPT-4o, AI Lawyer, dan Callidus AI sebagai preferensi utama dalam konsultasi hukum.

**2.1.2 LawPal: A Retrieval-Augmented Generation Based System for Enhanced Legal Accessibility in India**

Peneliti: Dnyanesh Panchal et al. (2025)

LawPal adalah sistem tanya jawab hukum berbasis Retrieval-Augmented Generation (RAG) yang dirancang untuk meningkatkan aksesibilitas hukum bagi masyarakat di India. Sistem ini menggabungkan pencarian hukum berbasis FAISS dan model generatif untuk memberikan jawaban yang lebih relevan dan legal-context aware.

Tujuannya adalah menyederhanakan literasi hukum dengan menjaga tingkat akurasi yang tinggi.

Hasil evaluasi menunjukkan bahwa sistem ini mampu mencapai akurasi di atas 90% dalam retrieval dan pembuatan jawaban hukum, serta mendapatkan tingkat kepuasan pengguna sebesar 85% dari kalangan mahasiswa hukum, pengacara, dan masyarakat umum.

**2.1.3 Chatbots in Customer Service within Banking and Finance**

Peneliti: Gary Graham et al. (2025)

Penelitian ini mengevaluasi efektivitas chatbot dalam layanan pelanggan di sektor perbankan dan keuangan. Sistem chatbot dinilai mampu meningkatkan efisiensi komunikasi dan pelayanan di situasi dinamis.

Namun demikian, penelitian ini juga menemukan bahwa intervensi manusia masih diperlukan untuk mempertahankan keandalan jawaban, terutama dalam kasus kompleks yang membutuhkan pemahaman konteks lebih dalam.

Penelitian ini menggunakan pendekatan kualitatif tanpa metrik kuantitatif eksplisit.

Tabel 2.1 Tabel Penelitian Terdahulu

| **Judul Penelitian** | **Tujuan** | **Hasil** |
| --- | --- | --- |
| LawPal: A RAG-Based System for Legal Accessibility – Dnyanesh Panchal et al. (2025) | Meningkatkan efisiensi dan akurasi jawaban AI untuk literasi hukum di India | Akurasi sistem mencapai 90% dalam interpretasi masalah hukum |
| Intelligent Legal Assistant – Rujing Yao et al. (2025) | Mengklarifikasi pertanyaan pengguna sebelum menjawab untuk meningkatkan relevansi | Skor kepuasan 4.8; 90% jawaban sesuai preferensi pengguna |
| Chatbots in Customer Service – Gary Graham et al. (2025) | Menilai efektivitas chatbot dalam layanan pelanggan perbankan | Menilai efektivitas chatbot dalam layanan pelanggan perbankan |



Gambar 2.1 Template Gambar Bab 2

## Teori Terkait

### 2.2.1 Teknologi Pengembangan Web dan Backend

#### 2.2.1.1 HTML, CSS, dan JavaScript

HTML (HyperText Markup Language), CSS (Cascading Style Sheets), dan JavaScript adalah tiga teknologi inti dalam pengembangan web. HTML digunakan untuk struktur halaman, CSS untuk desain dan tampilan visual, dan JavaScript untuk logika interaktif di sisi klien (Flanagan, 2020).

#### 2.2.1.2 Node.js

Node.js adalah runtime JavaScript berbasis event-driven, non-blocking I/O yang digunakan dalam pengembangan backend modern. Node.js memungkinkan pengembang untuk membangun aplikasi server-side yang efisien dan ringan (Tilkov & Vinoski, 2010).

#### 2.2.1.3 MongoDB Atlas

MongoDB adalah database NoSQL yang menyimpan data dalam format dokumen BSON. MongoDB Atlas adalah layanan database cloud yang dikelola, menyediakan skala otomatis dan performa tinggi untuk aplikasi modern (Chodorow, 2013).

### 2.2.2 Model AI: Alibaba Cloud Model Studio

Alibaba Cloud Model Studio merupakan platform pengembangan model kecerdasan buatan berbasis cloud yang memungkinkan pengguna untuk membangun, melatih, dan menyebarkan model AI. Platform ini mendukung proses visualisasi pipeline AI, pelatihan model LLM, dan integrasi dengan REST API, yang berperan sebagai backend eksternal dalam arsitektur sistem ini (Alibaba Cloud Docs, 2023).

### 2.2.3 Arsitektur Backend Internal dan Eksternal

#### 2.2.3.1 Backend Internal

Backend internal terdiri dari server Node.js yang mengelola routing API (endpoint), koneksi database MongoDB, dan antarmuka komunikasi dengan frontend web.

#### 2.2.3.2 Backend Eksternal

Backend eksternal merujuk pada model AI yang dideploy melalui Alibaba Cloud Model Studio, bertugas mengelola pemrosesan bahasa alami dan memberikan respons cerdas pada pengguna melalui integrasi API.

### 2.2.4 Diagram Sistem

#### 2.2.4.1 Use Case Diagram

Use case diagram adalah representasi visual dari interaksi antara aktor eksternal (seperti pengguna) dan sistem. Diagram ini digunakan untuk memahami fungsionalitas utama sistem dari perspektif pengguna (Booch et al., 2005).

#### 2.2.4.2 Context Diagram

Context diagram menggambarkan hubungan sistem dengan lingkungan eksternal. Diagram ini memberikan gambaran sistem secara keseluruhan dengan menyederhanakan komponen internal dan menampilkan arus data antara sistem dan entitas eksternal (Yourdon, 1989).

#### 2.2.4.3 Deployment Diagram

Deployment diagram dalam UML digunakan untuk memodelkan bagaimana komponen perangkat lunak di-deploy ke perangkat keras atau node fisik. Ini membantu menggambarkan topologi sistem dan hubungan antara client-server, database, dan komponen AI (Fowler, 2004).

### 2.2.5 Metodologi Adaptive Waterfall

Adaptive Waterfall adalah model pengembangan perangkat lunak yang mempertahankan tahapan linear khas Waterfall namun lebih fleksibel terhadap perubahan kebutuhan di tiap tahapannya. Model ini cocok untuk proyek dengan kebutuhan yang dapat berubah selama proses implementasi, tanpa meninggalkan struktur tahapan Waterfall tradisional seperti analisis, desain, implementasi, dan pengujian (Sommerville, 2011).

### 2.2.6 Browser Use (Browser Use Agent)

#### 2.2.6.1 Definisi

Browser Use adalah pustaka dan layanan open-source (MIT License) yang memadukan model bahasa besar (LLM) melalui LangChain dengan kontrol browser otomatis berbasis Playwright. Agen ini mampu membuka situs web, melakukan ekstraksi HTML dan visual, berinteraksi (klik/tulis), serta mengelola multi-tab secara otomatis

#### 2.2.6.2 Ekstraksi dan Interaksi Multimodal

Agen ini menggabungkan pemahaman vision (tangkapan layar) dan struktur DOM untuk mengenali elemen UI secara lebih andal . Ia juga melacak aksi melalui XPath dan memberi kemampuan self-correcting, berguna bila UI berubah .

#### 2.2.6.3 Workflow Agen

Inisiasi agen melalui kode Python ke Playwright:

python

from browser\_use import Agent

agent = Agent(task="...", llm=...)

await agent.run()

Segmen Program 2.1 Contoh Segmen Program Browser Use

Prompt LLM diterjemahkan menjadi aksi browser (klik, scroll). Playwright dieksekusi, serta LLM menerima umpan balik hingga tugas selesai.

#### 2.2.6.4 Integrasi LLM Modular

Agen mendukung berbagai LLM (GPT‑4, Claude, LLaMA, dll) melalui LangChain, memudahkan pipeline modular (planning, memory, tools). Dukungan vision-enabled mode (use\_vision=True) memperluas pengenalan fitur GUI .

#### 2.2.6.5 Studi Pendukung & Metodologi Pembelajaran

Steward oleh Tang & Shin (2024) adalah agen serupa yang fokus pada navigasi web menggunakan LLM, memberikan model end-to-end yang efisien .

AutoWebGLM adalah agen web agent berbasis ChatGLM yang disempurnakan melalui HTML simplification, RL, dan rejection sampling – menunjukkan peningkatan performa dibanding GPT‑4 .

#### 2.2.6.6 Keunggulan & Tantangan

Kelebihan:

Otomatisasi web dinamis tanpa API publik, mendukung login, interaksi AJAX, dan multi-tab

Kekurangan:

Memerlukan sumber daya signifikan (RAM, CPU), lebih lambat daripada scraping HTTP saja .

Rentan deteksi bot: CAPTCHA dan fingerprinting menjadi tantangan.

#### 2.2.6.7 Keamanan & Risk Mitigation

Risiko utama meliputi prompt injection dan pengambilalihan sesi karena interaksi LLM langsung dengan DOM . Mitigasi dapat dilakukan melalui sanitasi input, isolasi agent, dan audit log.

### 2.2.7 Langchain

Langchain adalah framework open-source untuk membangun aplikasi berbasis Large Language Model (LLM) yang terstruktur. Framework ini menyediakan integrasi modular antara prompt, model AI, memory, agent, dan tool, serta memungkinkan komposisi pipeline logika LLM yang kompleks (Chase, 2023). Dalam konteks penelitian ini, Langchain memberikan inspirasi pendekatan arsitektur pemanggilan model AI secara eksternal melalui API terstruktur.

### 2.2.8 Kerangka Teori

Berdasarkan kajian teori di atas, maka kerangka teori dari penelitian ini dapat disusun sebagai berikut:

Dasar Pengembangan Aplikasi Web: Teknologi web seperti HTML, CSS, JavaScript, dan Node.js menjadi fondasi interaksi pengguna.

Manajemen Data dan Server Internal: MongoDB Atlas dan Node.js digunakan untuk manajemen data dan logika backend internal.

Integrasi AI sebagai Backend Eksternal: Model AI dari Alibaba Cloud Model Studio berperan dalam pemrosesan kecerdasan bahasa.

Struktur Arsitektur Sistem: Diagram sistem (use case, context, dan deployment) mendukung perancangan sistem secara menyeluruh.

Metodologi Pengembangan: Adaptive Waterfall dipilih untuk fleksibilitas dalam tahapan pembangunan sistem.

Pendekatan Modular AI: Langchain memberi referensi integrasi modular untuk AI eksternal, walau tidak digunakan secara langsung.

# BAB III ANALISIS DAN PERANCANGAN

## Analisis

### Identifikasi Masalah

Analisis pada tahap ini bertujuan untuk memahami sistem yang ada saat ini dan mengidentifikasi kekurangan, permasalahan, atau peluang yang dapat diatasi dengan pengembangan sistem baru. Dalam konteks perancangan chatbot bantuan hukum personal "Osoora," identifikasi masalah dilakukan dengan menganalisis prosedur pemberian informasi dan bantuan hukum perdata kepada masyarakat, khususnya terkait KTP dan kepemilikan tanah.

Tabel 3.1 Identifikasi masalah

| **No** | **Masalah** | **Akibat** | **Efek** |
| --- | --- | --- | --- |
| 1 | Akses Terbatas | Masyarakat kesulitan mengakses informasi hukum perdata yang terpercaya dan mudah dipahami. | Kurangnya pemahaman hukum, kesulitan menyelesaikan masalah hukum. |
| 2 | Informasi terlalu kompleks | Bahasa hukum yang digunakan seringkali sulit dipahami oleh orang awam. | Kesalahpahaman informasi, keengganan mencari bantuan hukum. |
| 3 | Proses Lama | Proses konsultasi hukum konvensional | Keterlambatan penyelesaian masalah hukum |
| 4 | Ketersediaan Informasi | Informasi hukum yang tersedia mungkin tidak selalu up-to-date atau relevan dengan kasus spesifik. | Informasi yang tidak akurat, solusi yang tidak efektif. |
| 5 | Keterbatasan Sumber Daya | Kantor bantuan hukum dan pengacara seringkali memiliki keterbatasan sumber daya untuk menangani banyaknya permintaan informasi dan konsultasi | Waktu tunggu yang lama bagi masyarakat untuk mendapatkan bantuan, kualitas layanan yang tidak optimal. |
| 6 | Kurangnya Kesadaran Hukum | Masyarakat umum kurang memiliki kesadaran akan hak dan kewajiban hukum mereka, sehingga rentan terhadap masalah hukum yang sebenarnya bisa dicegah. | Meningkatnya kasus hukum yang sebenarnya bisa diselesaikan secara damai, akan tetapi pada kenyataannya masyarakat mengalami kerugian materiil dan imateriil |
|  |  |  |  |
|  |  |  |  |



Gambar 3.1 Template Gambar Bab 3

### Pemecahan Masalah

Pemecahan masalah diperlukan untuk menyusun secara sistematis langkah-langkah yang akan digunakan untuk pemecahan masalah. Dengan adanya kerangka pemecahan masalah ini diharapkan proses dan hasil yang diperoleh nantinya akan sesuai dengan tujuan yang diinginkan.

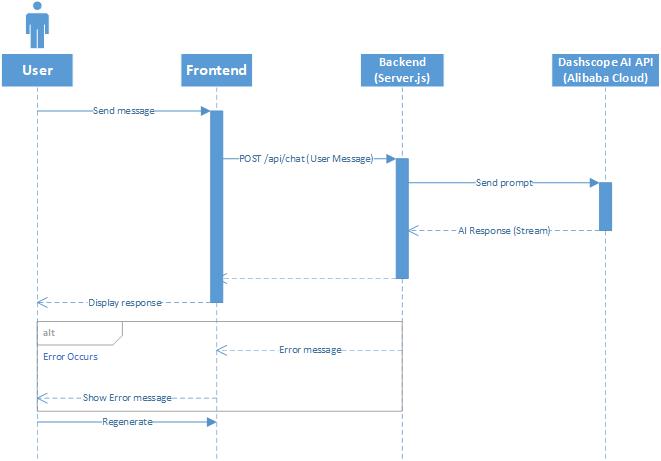
Berdasarkan masalah yang tertera pada bagian identifikasi masalah maka berikut adalah solusi yang diusulkan untuk mengatasi setiap masalah yang diidentifikasi:

| **Masalah** | **Solusi** |
| --- | --- |
| Akses Terbatas | * Mengembangkan chatbot berbasis web yang dapat diakses oleh siapa saja dengan koneksi internet, tanpa batasan geografis atau waktu. * Menyediakan antarmuka yang user-friendly dan mudah dinavigasi, bahkan oleh pengguna dengan kemampuan teknologi yang terbatas. |
| Informasi terlalu kompleks | * Memanfaatkan teknologi Generative AI dan NLP untuk memungkinkan chatbot memahami pertanyaan pengguna dalam bahasa alami dan memberikan jawaban dalam bahasa yang sederhana dan mudah dipahami. * Menggunakan contoh kasus dan analogi untuk menjelaskan konsep hukum yang kompleks. |
| Proses Lama | * Mengotomatiskan proses pemberian informasi hukum dasar melalui chatbot, sehingga pengguna dapat memperoleh jawaban atas pertanyaan mereka dengan cepat dan efisien. * Mengurangi kebutuhan untuk konsultasi hukum tatap muka untuk masalah-masalah yang relatif sederhana. |
| Ketersediaan Informasi | * Mengintegrasikan chatbot dengan basis data hukum yang up-to-date dan relevan. * Secara teratur memperbarui dan memelihara basis data hukum untuk memastikan informasi yang diberikan akurat dan terkini. |

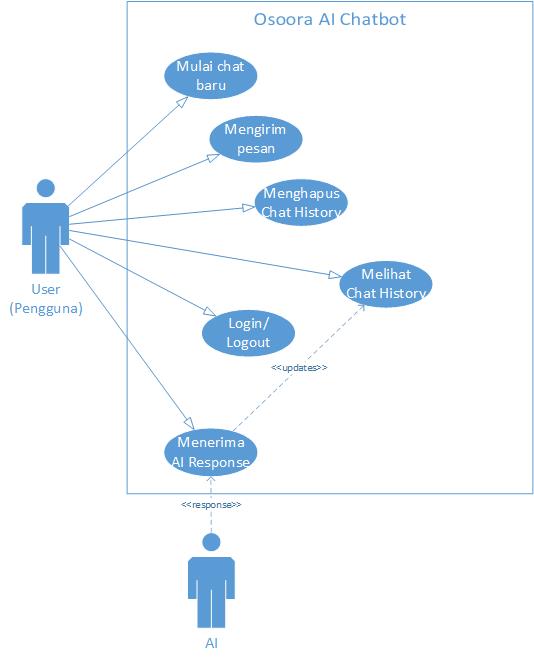
## Perancangan

### Perancangan Sistem

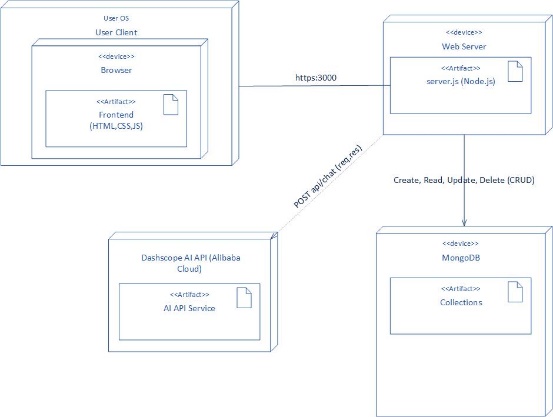
Pada perancangan system Menguraikan tentang gambaran sistem yang sedang berjalan atau sistem yang masih berjalan secara manual (belum terkomputerisasi), yang akan dibangun, atau dikembangkan dalam rangka pencapaian efektifitas dan efisiensi sistem. Dalam penggambaran ini dapat menggunakan alat bantu yaitu diagram konteks, diagram alir data (DFD), Diagram Alir Dokumen (Flow of Document) dan bisa juga menggunakan pemodelan Unified Modelling Language (UML), termasuk menggambarkan notasi pseudocode/algoritma



Gambar 3.1 Context Diagram



Gambar 3.2 Use Case Diagram



Gambar 3.3 Deployment Diagram

### Perancangan Data

Perancangan data merupakan komponen yang digunakan untuk mendefinisikan struktur data, yang terdiri dari Perancangan Entity Relationship Diagram (ERD) dan juga struktur file pendukung.



Gambar 3.4 Entity Relationship Diagram (ERD)

### Perancangan User Interface / *Mock-up* aplikasi

Dalam penelitian ini, perancangan antarmuka pengguna (UI) tidak dilakukan secara manual melalui mockup konvensional, melainkan dengan menyusun prompt (perintah berbasis teks) yang diproses oleh sistem AI untuk menghasilkan rancangan antarmuka secara otomatis. Proses ini dilakukan menggunakan model Claude 3.5 Sonnet sebagai alat bantu utama dalam pengembangan desain berbasis AI. Prompt yang digunakan adalah dengan menggunakan multi round conversation.

Contoh :

System : Anda adalah seorang designer UI yang sudah berpengalaman selama kurang lebih 5 tahun. Anda memahami setiap detail dalam desain seperti Design Principles dll.  
Round 1

User : Tolong buatkan UI chatbot dengan history dengan Vanilla HTML,CSS dan JS

AI : \*membuat aplikasi UI chatbot\*

Round 2

User : Tambahkan fitur Login dan Daftar

AI : \*menambah fitur login dan daftar\*  
  
Round 3

User : tolong perbaiki UI nya   
AI : \*memperbaiki UI\*

## Rancangan Pengujian

Dalam penelitian ini, untuk mengukur performa model klasifikasi yang dibangun, digunakan beberapa metrik evaluasi yaitu **precision**, **recall**, dan **F1 score**. Ketiga metrik ini umum digunakan dalam bidang machine learning untuk menilai kualitas prediksi model, terutama dalam kasus klasifikasi yang tidak seimbang.

### Precision Precision adalah rasio antara jumlah data yang diklasifikasikan positif dengan benar (True Positive) terhadap total data yang diprediksi sebagai positif (True Positive + False Positive). Precision mengukur seberapa akurat model dalam mengidentifikasi data positif.

### ****Recall**** Recall adalah rasio antara jumlah data positif yang benar-benar terdeteksi (True Positive) terhadap total data positif yang sebenarnya ada (True Positive + False Negative). Recall mengukur kemampuan model untuk menemukan semua data positif.

# BAB IV PEMBAHASAN

## Gambaran Umum Obyek Penelitian

Bagian ini memberi gambaran singkat tentang obyek penelitian yang memiliki kaitan langsung dengan permasalahan yang diteliti. Beberapa hal yang dibahas pada bagian ini adalah struktur organisasi yang dapat menunjukkan posisi unit terkait (user) yang terlibat dalam penelitian, tugas dan wewenang unit tersebut, aturan yang diterapkan pada obyek penelitian terkait dengan penelitian yang dilakukan.

Pada bab ini, objek penelitian yang dimaksud adalah aplikasi chatbot bantuan hukum "OSOORA" yang telah berhasil diimplementasikan secara nyata. OSOORA merupakan sebuah sistem berbasis web yang mengintegrasikan teknologi backend, frontend, dan database untuk memberikan layanan konsultasi hukum secara otomatis melalui chat berbasis AI.

Aplikasi ini terdiri dari beberapa komponen utama, yaitu:

**Frontend (Antarmuka Pengguna)**

Frontend OSOORA dibangun menggunakan HTML, CSS, dan JavaScript murni.

Fitur utama meliputi: halaman login dan registrasi, tampilan chat interaktif, pengelolaan riwayat percakapan, serta pengaturan preferensi tema (dark/light).

Seluruh interaksi pengguna, mulai dari pengiriman pesan hingga pengelolaan akun, dilakukan melalui antarmuka web yang responsif.

**Backend (Server Aplikasi)**

Backend dikembangkan menggunakan Node.js dengan framework Express.

Backend menyediakan berbagai endpoint API untuk autentikasi pengguna, pengelolaan data, serta pengiriman dan penerimaan pesan chat.

Backend juga terintegrasi dengan layanan AI eksternal untuk menghasilkan jawaban otomatis atas pertanyaan pengguna.

Seluruh data penting, seperti akun pengguna, preferensi, dan riwayat chat, dikelola secara aman melalui backend.

Database (MongoDB Atlas)

Database digunakan untuk menyimpan data pengguna, preferensi, dan percakapan.

Implementasi database menggunakan MongoDB Atlas yang berbasis cloud, sehingga data dapat diakses secara real-time dan aman.

Setiap data disimpan dalam koleksi terpisah, yaitu users, user\_preferences, dan conversations.

Secara umum, implementasi OSOORA menghasilkan sebuah sistem chatbot bantuan hukum yang dapat digunakan secara langsung oleh masyarakat melalui browser, dengan pengalaman interaksi yang real-time, aman, dan mudah digunakan. Setiap komponen saling terintegrasi untuk mendukung proses konsultasi hukum secara otomatis, mulai dari autentikasi, pengelolaan data, hingga penyajian jawaban berbasis AI.

Tabel 4.1 Tabel Penjelasan Tech Stack pada OSOORA

|  |  |  |  |
| --- | --- | --- | --- |
| **Komponen** | **Tech Stack** | **Fungsi Utama** | **Penjelasan** |
| Frontend | HTML, CSS, JavaScript (vanilla) | Antarmuka pengguna interaktif  - Halaman login/registrasi  - Tampilan chat  - Pengaturan tema | Digunakan melalui browser, responsif, mendukung pengelolaan akun dan percakapan |
| Backend | |  | | --- | |  |  |  | | --- | | Node.js, Express.js | | - Autentikasi pengguna  - Pengelolaan data pengguna & percakapan  - Integrasi AI untuk menjawab chat | API menyediakan endpoint untuk komunikasi frontend dan pengolahan data secara aman |
| Database | MongoDB Atlas (Cloud-based NoSQL) | - Penyimpanan data akun, preferensi pengguna, riwayat percakapan | Koleksi: users, user\_preferences, conversations |
| Model AI | Qwen 2.5 72B Instruct via Alibaba Cloud | - Menyediakan jawaban otomatis atas pertanyaan hukum melalui chat | Terintegrasi melalui backend sebagai sistem pemroses bahasa alami |



Gambar 4.1 Template Gambar Bab 4

## Implementasi

Implementasi merupakan proses realisasi dan pembuatan aplikasi yang dilakukan setelah tahap perencanaan dan pemilihan metode yang akan digunakan. Sintaks atau kode program yang digunakan pada tahap ini akan dijelaskan secara lebih rinci pada Lampiran 5.

### Implementasi Backend

#### 4.2.1.1 Implementasi Database

#### 4.2.1.2 Implementasi RESTful API

|  |  |  |  |
| --- | --- | --- | --- |
| **No** | **Endpoint/URI** | **Method** | **Penjelasan** |
| 1 | /api/auth/login | POST | Melakukan autentikasi pengguna dengan verifikasi kredensial, pembangkitan token, pembaruan waktu login terakhir, dan pengambilan preferensi pengguna. Menjamin keamanan akses dan manajemen sesi pengguna. |
| 2 | |  | | --- | |  |   /api/auth/register | POST | Melakukan registrasi pengguna baru dengan validasi data, pengecekan duplikasi, hashing password, serta penyimpanan data pengguna dan preferensi awal ke basis data. Menjamin keamanan dan integritas proses pendaftaran. |
| 3 | /api/chat | POST | Mengirim pesan ke layanan chat (AI/LLM) dan menerima balasan secara streaming (SSE). Mendukung interaksi real-time berbasis AI dan penyimpanan riwayat pesan ke basis data. |
| 4 | /api/conversations | GET | Mengambil daftar seluruh percakapan milik pengguna yang sedang login. Mendukung manajemen riwayat komunikasi pengguna. |
| 5 | /api/conversations | POST | Membuat percakapan baru untuk pengguna. Data percakapan baru disimpan ke basis data dan preferensi pengguna diperbarui. Mendukung inisiasi komunikasi baru. |
| 6 | /api/conversations/clear | DELETE | Menghapus seluruh percakapan milik pengguna yang sedang login. Mendukung pengelolaan data pribadi dan privasi pengguna. |
| 7 | /api/conversations/:conversationId/messages | GET | Mengambil seluruh pesan dalam percakapan tertentu yang dimiliki pengguna. Menjamin privasi dan keamanan data komunikasi dengan pengecekan kepemilikan percakapan. |
| 8 | /api/conversations/:conversationId/messages | POST | Menambahkan pesan baru ke percakapan tertentu. Pesan disimpan ke basis data, metadata percakapan diperbarui, dan verifikasi kepemilikan percakapan dilakukan untuk menjaga integritas dan keamanan data |
| 9 | /api/user/preferences | GET | Mengambil data preferensi pengguna yang sedang login dengan autentikasi token. Jika preferensi tidak ditemukan, mengembalikan status 404 sesuai prinsip RESTful. |
| 10 | /api/user/preferences | PUT | Memungkinkan pengguna memperbarui atau membuat data preferensi (misal: tema, percakapan terakhir). Mendukung pembaruan parsial dan pembuatan baru jika belum ada, dengan pengelolaan timestamp untuk konsistensi data. |
| 11 | /api/user/profile | PUT | Memungkinkan pengguna memperbarui data profil (username, password) dengan validasi, pengecekan duplikasi, dan verifikasi password lama sebelum perubahan. Mendukung self-service dan keamanan data pengguna. |

    // Create user document

    const userId = uuidv4();

    const token = uuidv4(); // Simple token for demo

    const newUser = {

      user\_id: userId,

      username,

      email,

      password\_hash: passwordHash,

      token,

      created\_at: new Date(),

      last\_login: new Date()

    };

    // Insert user

    await db.collection('users').insertOne(newUser);

    // Create default user preferences

    const preferenceId = uuidv4();

    const userPreferences = {

      preference\_id: preferenceId,

      user\_id: userId,

      theme: 'system',

      last\_active\_conversation: null,

      created\_at: new Date(),

      updated\_at: new Date()

    };

    await db.collection('user\_preferences').insertOne(userPreferences);

    // Return success with token

    return res.status(201).json({

      message: 'User registered successfully',

      token,

      user: {

        username,

        email,

        user\_id: userId

      }

    });

  } catch (error) {

    console.error('Registration error:', error);

    return res.status(500).json({ error: 'Internal server error during registration' });

  }

});

app.post('/api/auth/register', async (req, res) => {

  try {

    const { username, email, password } = req.body;

    // Validate input

    if (!username || !email || !password) {

      return res.status(400).json({ error: 'Missing required fields' });

    }

    // Check if user already exists

    const existingUser = await db.collection('users').findOne({

      $or: [{ email }, { username }]

    });

    if (existingUser) {

      return res.status(409).json({ error: 'User with this email or username already exists' });

    }

    // Hash password

    const salt = await bcrypt.genSalt(10);

    const passwordHash = await bcrypt.hash(password, salt);

    // Create user document

    const userId = uuidv4();

    const token = uuidv4(); // Simple token for demo

    const newUser = {

      user\_id: userId,

      username,

      email,

      password\_hash: passwordHash,

      token,

      created\_at: new Date(),

      last\_login: new Date()

    };

    // Insert user

    await db.collection('users').insertOne(newUser);

    // Create default user preferences

    const preferenceId = uuidv4();

    const userPreferences = {

      preference\_id: preferenceId,

      user\_id: userId,

      theme: 'system',

      last\_active\_conversation: null,

      created\_at: new Date(),

      updated\_at: new Date()

    };

    await db.collection('user\_preferences').insertOne(userPreferences);

    // Return success with token

    return res.status(201).json({

      message: 'User registered successfully',

      token,

      user: {

        username,

        email,

        user\_id: userId

      }

    });

  } catch (error) {

    console.error('Registration error:', error);

    return res.status(500).json({ error: 'Internal server error during registration' });

  }

});

### 4.2.2.1 Implementasi Model AI

#### 4.2.3.1 Implementasi System Prompt

    // Update user's token and last login

    await db.collection('users').updateOne(

      { \_id: user.\_id },

      {

        $set: {

          token,

          last\_login: new Date()

        }

      }

    );

    // Get user preferences

    const preferences = await db.collection('user\_preferences').findOne({ user\_id: user.user\_id });

    // Return success with token and user data

    return res.status(200).json({

      message: 'Login successful',

      token,

      user: {

        user\_id: user.user\_id,

        username: user.username,

        email: user.email,

        preferences: preferences || { theme: 'system' }

      }

    });

  } catch (error) {

    console.error('Login error:', error);

    return res.status(500).json({ error: 'Internal server error during login' });

  }

});

#### 4.2.3.2 Implementasi

### 4.2.4 Implementasi Login

  } catch (error) {

    console.error('Login error:', error);

    return res.status(500).json({ error: 'Internal server error during login' });

  }

});

Segmen Program 4.1 Template Segmen Program

## Uji Coba

Dalam pengujian sistem aplikasi OSOORA, dilakukan dua jenis pengujian utama untuk memastikan kualitas dan keandalan sistem.

Pertama, pengujian aplikasi dilakukan menggunakan metode Blackbox Testing, yang bertujuan untuk mengevaluasi fungsionalitas sistem dari sisi pengguna tanpa memperhatikan struktur internal kode. Metode ini digunakan untuk menguji apakah seluruh fitur antarmuka dan logika aplikasi berjalan sesuai dengan spesifikasi yang telah dirancang.

Kedua, dilakukan pengujian terhadap model AI yang digunakan dalam chatbot. Evaluasi model ini difokuskan pada kemampuan sistem dalam memberikan jawaban yang relevan dan akurat terhadap pertanyaan hukum yang diajukan pengguna. Pengujian dilakukan menggunakan tiga metrik evaluasi umum dalam bidang pemrosesan bahasa alami, yaitu Precision, Recall, dan F1-Score. Metrik tersebut digunakan untuk mengukur tingkat relevansi, kelengkapan, dan keseimbangan antara keduanya dalam setiap respons yang dihasilkan oleh model.

# BAB V PENUTUP

## Kesimpulan

Kesimpulan merupakan rangkuman hasil yang dicapai dan merupakan jawaban rumusan masalah. Kesimpulan dapat mengemukakan kembali masalah penelitian dan kemudian menjawabnya (menjawab pertanyaan dalam rumusan masalah), menyimpulkan buktibukti yang diperoleh dan akhirnya menarik kesimpulan apakah hasil yang didapat (dikerjakan) layak untuk digunakan (diimplementasikan). Penulis tidak diperkenankan menyimpulkan masalah jika pembuktian tidak terdapat dalam hasil penelitian. Hal-hal yang diperkuat: a) berhubungan dengan apa yang dikerjakan, b) Didasarkan pada analisis yang objektif, c) Bukti-bukti yang telah ditemukan.

## Saran

Bagian ini menguraikan saran‐saran yang perlu diperhatikan berdasarkan keterbatasan yang ditemukan dan asumsi yang dibuat, termasuk saran untuk pengembangan lebih lanjut. Saran juga bisa merupakan manifestasi dari penulis untuk dilaksanakan lebih lanjut (sesuatu yang belum ditempuh dan layak untuk dilanjutkan). Saran dicantumkan karena peneliti melihat adanya jalan keluar untuk mengatasi masalah (kelemahan yang ada), saran yang diberikan tidak terlepas dari ruang lingkup penelitian (untuk objek penelitian maupun pembaca yang akan mengembangkan hasil penelitian).

# DAFTAR PUSTAKA

**Yao, R., Wu, Y., Zhang, T., et al. (2025). Intelligent Legal Assistant... arXiv:2502.07904.**

**Panchal, D., Gole, A., Narute, V., & Joshi, R. (2025). LawPal: A RAG-Based System... arXiv.**

**Graham, G., et al. (2025). Chatbots in Customer Service... Computers in Human Behavior.**

**Pressman, R. S. (2014). Software Engineering: A Practitioner’s Approach (8th ed.). McGraw-Hill Education.**

**Munassar, N. M. A., & Govardhan, A. (2010). A Comparison Between Five Models of Software Engineering. International Journal of Computer Science Issues (IJCSI), 7(5), 94–101.**

**Sommerville, I. (2016). Software Engineering (10th ed.). Pearson Education Limited.**

**Flanagan, D. (2020). JavaScript: The definitive guide (7th ed.). O’Reilly Media.**

**Tilkov, S., & Vinoski, S. (2010). Node.js: Using JavaScript to build high‑performance network programs. IEEE Internet Computing, 14(6), 80–83. https://doi.org/10.1109/MIC.2010.145**

**Chodorow, K. (2013). MongoDB: The definitive guide (2nd ed.). O’Reilly Media.**

**Alibaba Cloud Documentation. (2023). Model Studio: Build and deploy AI models. Alibaba Cloud.**

**Booch, G., Rumbaugh, J., & Jacobson, I. (2005). The unified modeling language user guide (2nd ed.). Addison‑Wesley.**

**Yourdon, E. (1989). Modern structured analysis. Prentice Hall.**

**Fowler, M. (2004). UML distilled: A brief guide to the standard object modeling language (3rd ed.). Addison‑Wesley.**

**Sommerville, I. (2011). Software engineering (9th ed.). Addison‑Wesley.**

**Chase, J. (2023). LangChain: Framework for modular LLM systems [Framework documentation].**

**Tang, B., & Shin, K. G. (2024). Steward: Natural Language Web Automation [Preprint]. arXiv. https://arxiv.org/abs/2409.15441**

**He, L., Wen, H., & others. (2024, April 10). AutoWebGLM: A GPT‑4‑Outperforming Automated Web Navigation Agent [Conference paper]. KDD.**

**Browser Use contributors. (2025). Browser‑use: Make websites accessible for AI agents [Software]. GitHub.**

**Pathak, K. (2025, February 20). Integrating LLMs, Playwright, Browser‑Use and Web‑UI [Blog post]. Medium.**

**Reddi, B., & others. (2024, April). Refusal‑Trained LLMs Are Easily Jailbroken As Browser Agents [Preprint]. arXiv. https://arxiv.org/abs/2410.13886**

# LAMPIRAN

1. Surat Keputusan Tugas Akhir (SK TA)
2. Biodata Penulis
3. Hasil cek plagiarism
4. Dokumen pendukung penelitian (bukti wawancara, form isian, format laporan, standar operasional prosedur, aturan – aturan yang berlaku, dll)
5. Listing Program/Coding

Lampiran 1 Surat Keputusan Tugas Akhir (SK TA)

Lampiran 2 Biodata Penulis

**BIODATA PENULIS**

|  |  |  |
| --- | --- | --- |
| Nama | : |  |
| Alamat | : |  |
| Tempat/Tanggal Lahir | : |  |
| Telp. / Email | : |  |

PENDIDIKAN

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| No | Pendidikan | Tempat | Tahun Lulus | | Gelar | Bidang Spesialisasi |
| Mulai | Lulus |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |

PEKERJAAN

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Pekerjaan | Bidang | Perusahaan | Tahun |
|  |  |  |  |  |
|  |  |  |  |  |

PRESTASI

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Prestasi | Bidang | Penyelenggara | Tahun |
|  |  |  |  |  |
|  |  |  |  |  |

SERTIFIKASI KOMPETENSI

|  |  |  |  |
| --- | --- | --- | --- |
| No | Sertifikasi | Penyelenggara | Tahun |
|  |  |  |  |
|  |  |  |  |

Lampiran 3 Hasil Cek Plagiarisme

Lampiran 4 Dokumen pendukung penelitian

Lampiran 5 Listing Program/Coding

// Connect to MongoDB Atlas

async function connectToMongoDB() {

  try {

    // menambahkan TLS untuk fix SSL (karena menggunakan MongoDB Atlas)

    const client = new MongoClient(uri, {

      ssl: true,

      tls: true,

      tlsAllowInvalidCertificates: false,

      serverApi: {

        version: '1',

        strict: true,

        deprecationErrors: true

      }

    });

    await client.connect();

    db = client.db(dbName);

    console.log(`Connected to MongoDB Atlas successfully - Database: ${dbName}`);

    return client;

  } catch (error) {

    console.error('MongoDB Atlas connection error:', error);

    throw error;

  }

}

// Connect to MongoDB Atlas

async function connectToMongoDB() {

  try {

    // Add explicit TLS options to fix SSL connection issues

    const client = new MongoClient(uri, {

      ssl: true,

      tls: true,

      tlsAllowInvalidCertificates: false,

      serverApi: {

        version: '1',

        strict: true,

        deprecationErrors: true

      }

    });

    await client.connect();

    db = client.db(dbName);

    console.log(`Connected to MongoDB Atlas successfully - Database: ${dbName}`);

    return client;

  } catch (error) {

    console.error('MongoDB Atlas connection error:', error);

    throw error;

  }

}

app.post('/api/auth/register', async (req, res) => {

  try {

    const { username, email, password } = req.body;

    // Validate input

    if (!username || !email || !password) {

      return res.status(400).json({ error: 'Missing required fields' });

    }

    // Check if user already exists

    const existingUser = await db.collection('users').findOne({

      $or: [{ email }, { username }]

    });

    if (existingUser) {

      return res.status(409).json({ error: 'User with this email or username already exists' });

    }

    // Hash password

    const salt = await bcrypt.genSalt(10);

    const passwordHash = await bcrypt.hash(password, salt);

    // Create user document

    const userId = uuidv4();

    const token = uuidv4(); // Simple token for demo

    const newUser = {

      user\_id: userId,

      username,

      email,

      password\_hash: passwordHash,

      token,

      created\_at: new Date(),

      last\_login: new Date()

    };

    // Insert user

    await db.collection('users').insertOne(newUser);

    // Create default user preferences

    const preferenceId = uuidv4();

    const userPreferences = {

      preference\_id: preferenceId,

      user\_id: userId,

      theme: 'system',

      last\_active\_conversation: null,

      created\_at: new Date(),

      updated\_at: new Date()

    };

    await db.collection('user\_preferences').insertOne(userPreferences);

    // Return success with token

    return res.status(201).json({

      message: 'User registered successfully',

      token,

      user: {

        username,

        email,

        user\_id: userId

      }

    });

  } catch (error) {

    console.error('Registration error:', error);

    return res.status(500).json({ error: 'Internal server error during registration' });

  }

});

app.post('/api/auth/register', async (req, res) => {

  try {

    const { username, email, password } = req.body;

    // Validate input

    if (!username || !email || !password) {

      return res.status(400).json({ error: 'Missing required fields' });

    }

    // Check if user already exists

    const existingUser = await db.collection('users').findOne({

      $or: [{ email }, { username }]

    });

    if (existingUser) {

      return res.status(409).json({ error: 'User with this email or username already exists' });

    }

    // Hash password

    const salt = await bcrypt.genSalt(10);

    const passwordHash = await bcrypt.hash(password, salt);

    // Create user document

    const userId = uuidv4();

    const token = uuidv4(); // Simple token for demo

    const newUser = {

      user\_id: userId,

      username,

      email,

      password\_hash: passwordHash,

      token,

      created\_at: new Date(),

      last\_login: new Date()

    };

    // Insert user

    await db.collection('users').insertOne(newUser);

    // Create default user preferences

    const preferenceId = uuidv4();

    const userPreferences = {

      preference\_id: preferenceId,

      user\_id: userId,

      theme: 'system',

      last\_active\_conversation: null,

      created\_at: new Date(),

      updated\_at: new Date()

    };

    await db.collection('user\_preferences').insertOne(userPreferences);

    // Return success with token

    return res.status(201).json({

      message: 'User registered successfully',

      token,

      user: {

        username,

        email,

        user\_id: userId

      }

    });

  } catch (error) {

    console.error('Registration error:', error);

    return res.status(500).json({ error: 'Internal server error during registration' });

  }

});

app.post('/api/auth/login', async (req, res) => {

  try {

    const { email, password } = req.body;

    // Validate input

    if (!email || !password) {

      return res.status(400).json({ error: 'Missing email or password' });

    }

    // Find user

    const user = await db.collection('users').findOne({ email });

    if (!user) {

      return res.status(401).json({ error: 'Invalid credentials' });

    }

    // Verify password

    const validPassword = await bcrypt.compare(password, user.password\_hash);

    if (!validPassword) {

      return res.status(401).json({ error: 'Invalid credentials' });

    }

    // Generate new token

    const token = uuidv4();

    // Update user's token and last login

    await db.collection('users').updateOne(

      { \_id: user.\_id },

      {

        $set: {

          token,

          last\_login: new Date()

        }

      }

    );

    // Get user preferences

    const preferences = await db.collection('user\_preferences').findOne({ user\_id: user.user\_id });

    // Return success with token and user data

    return res.status(200).json({

      message: 'Login successful',

      token,

      user: {

        user\_id: user.user\_id,

        username: user.username,

        email: user.email,

        preferences: preferences || { theme: 'system' }

      }

    });

  } catch (error) {

    console.error('Login error:', error);

    return res.status(500).json({ error: 'Internal server error during login' });

  }

});