Blueprint for an AI-Powered Oral Health Triage System

Executive Summary: A Framework for an Applied Multimodal AI System in Medicine

This report provides a comprehensive architectural and research-backed blueprint for a multimodal artificial intelligence (AI) system designed to triage patients with oral health issues. The proposed system is intended to serve as a clinical decision support tool for healthcare professionals, leveraging AI to analyze patient-submitted data and assign a preliminary urgency level. The architecture is founded on modern, scalable, and resilient software engineering principles, specifically tailored to the stringent requirements of the healthcare domain.

The system's core is an asynchronous, event-driven pipeline that decouples data ingestion from computationally intensive AI processing. A Python-based backend, built on the high-performance FastAPI framework, will manage a chatbot interface for data submission. This data, including responses from a version-controlled clinical questionnaire and patient images, is then published to a message queue. A series of consumer services, acting as a chained multimodal inference pipeline, process this data. A Large Language Model (LLM) first analyzes the textual information to extract clinical entities and provide context. This structured output, along with the original image, is then fed to a domain-specific multimodal model for visual analysis. The verdicts from these models are aggregated to produce a final triage priority, which can be used to organize cases for human review.

This document deconstructs the system into its primary technical layers, justifying each technology choice—from the backend framework and message queue to the database schema for versioned questionnaires. Furthermore, it grounds these engineering decisions in a robust review of the current academic landscape, examining state-of-the-art research in LLMs for clinical triage and multimodal AI for oral disease detection. Finally, it addresses the critical aspects of model validation, performance evaluation, and the ethical and regulatory considerations pertinent to deploying such a system in Brazil, with a focus on compliance with the Lei Geral de Proteção de Dados (LGPD). This blueprint serves as a detailed roadmap for both the implementation and the academic justification of the project.

Section 1: System Architecture and Technology Stack

This section outlines the technical architecture of the proposed triage system. Each component has been selected to ensure scalability, reliability, and security, addressing the unique challenges of processing sensitive medical data through a complex AI pipeline.

1.1. The Ingestion Layer: API and Chatbot Backend

The primary entry point for data is a chatbot interface managed by a Python backend. This backend serves as the API for data submission and is responsible for managing the conversation flow with the healthcare professional.

Analysis of Backend Frameworks

The choice of a backend framework is a foundational decision that impacts performance, development speed, and maintainability.

- **Django:** A mature, "batteries-included" framework, Django offers a powerful Object-Relational Mapper (ORM), a built-in administrative panel, and robust security features out of the box.[1] While it is an excellent choice for large-scale, content-heavy applications and enterprise systems, its primarily synchronous architecture can create performance bottlenecks in I/O-bound applications, such as those that frequently call external AI model APIs.[3]
- **Flask:** As a lightweight microframework, Flask provides maximum flexibility and control, making it ideal for smaller services and prototypes.[1] However, this minimalism means that essential features for this project, such as asynchronous support and data validation, must be added with third-party libraries, potentially increasing development complexity and integration overhead.[5]
- **FastAPI:** A modern, high-performance microframework built specifically for creating APIs.[4] It is built upon Starlette for asynchronous request handling and Pydantic for data validation using standard Python type hints.[1] Its native support for async/await syntax makes it exceptionally well-suited for I/O-bound tasks, and it automatically generates interactive API documentation (via Swagger UI and ReDoc),

which significantly streamlines development and testing.[1]

Recommendation: FastAPI

For this AI-powered medical triage system, **FastAPI is the recommended framework**. The decision is based on its alignment with the core operational needs of a modern AI application. Medical AI systems are fundamentally I/O-bound; they spend a significant amount of time waiting for responses from model inference endpoints, database queries, and other services. A traditional synchronous framework would have its worker threads blocked during these waits, severely limiting the number of concurrent users it can handle without a massive allocation of server resources.[5] FastAPI's asynchronous architecture, based on Python's

asyncio, allows a single worker to efficiently manage thousands of concurrent connections by yielding control during I/O waits.[1] This results in significantly higher throughput and lower latency with fewer resources.

Furthermore, Pydantic's integrated data validation is not merely a developer convenience but a critical safety feature in a medical context. It enforces a strict schema on incoming data, ensuring that all information, such as patient responses and identifiers, conforms to expected data types and formats before entering the processing pipeline. This preemptively mitigates data corruption and addresses some of the key limitations of AI in medicine, such as incorrectness and unsafety.[7] The industry trend for 2025 and beyond clearly favors asynchronous frameworks for AI-centric applications, with FastAPI leading this shift.[1]

Conversation and State Management

The chatbot component will guide a healthcare professional through a structured questionnaire. While powerful conversational AI frameworks like Rasa exist, offering advanced Natural Language Understanding (NLU) and dialogue management [9], they are overly complex for this use case. A predefined, stateful conversation does not require sophisticated NLU to interpret user intent.

Instead, a custom Python class within the FastAPI application, such as a ConversationManager, will be implemented. This class will be responsible for:

1. Loading the active, versioned questionnaire from the database.
2. Presenting questions sequentially to the user.

3. Validating user inputs against the question's expected type (e.g., text, number, choice).
4. Maintaining the state of the conversation.
5. Packaging the completed questionnaire and image references into a standardized JSON object for submission to the asynchronous pipeline.

This approach provides the necessary control and is significantly simpler to implement and maintain for a structured data entry workflow.

Framework	Type	Key Features	Performance	AI/ML Integration	Ideal Use Case
FastAPI	Microframework	Async-first, Pydantic validation, Auto-generated docs [1]	Very High	Excellent, native support for asyncio libraries [1]	High-performance APIs, AI/ML microservices
Django	Full-stack	"Batteries-included" (ORM, Admin), Mature ecosystem [2]	Good	Partial (via ASGI), less native than FastAPI [1]	Large-scale web apps, Enterprise systems
Flask	Microframework	Minimalist, Flexible, Large ecosystem [1]	Good	Requires extensions for async and validation	Simple web apps, MVPs, Microservices

1.2. The Asynchronous Backbone: An Event-Driven Data Pipeline

To ensure the system is responsive and resilient, the data ingestion API must be decoupled from the time-consuming AI model inference. An event-driven architecture using a message queue is the standard pattern for achieving this decoupling.[11]

Message Queue Analysis

- **Apache Kafka:** A distributed event streaming platform designed for extremely high throughput and data persistence.[13] It operates as an immutable log, allowing messages to be replayed by consumers. While immensely powerful for real-time analytics and large-scale data pipelines, its operational complexity and focus on event streaming make it less suitable for the task-queuing nature of this project.[15]
- **RabbitMQ:** A traditional, broker-based message queue that excels in scenarios requiring complex routing, task distribution, and guaranteed message delivery.[13] It employs a "smart broker/dumb consumer" model, where the broker manages message delivery to consumers.[15] A key feature for this project is its support for **message priorities**, which allows messages to be consumed in a specific order based on their assigned importance.[13]
- **Managed Cloud Services (GCP Pub/Sub, AWS SQS):** These services provide robust messaging capabilities without the overhead of managing the underlying infrastructure. GCP Pub/Sub is a global, scalable service that supports both push and pull delivery models and is explicitly noted as a HIPAA-compliant service, a crucial factor for medical applications.[18] AWS SQS is another highly reliable option, offering both standard and FIFO (First-In-First-Out) queues to guarantee message order.[18]

Recommendation: RabbitMQ or GCP Pub/Sub

For the scope of a TCC, **RabbitMQ** presents the most conceptually elegant solution. Its native support for **message priorities** directly maps to the core requirement of the triage system.[13] The final "overall verdict" from the AI pipeline can be used to set the priority of the message. This ensures that when a human reviewer or a dashboard consumes from the queue, the most urgent patient cases are processed first.

Alternatively, if the project prioritizes cloud-native integration and minimal operational overhead, **GCP Pub/Sub** is an excellent choice. Its managed nature, high scalability, and explicit HIPAA compliance make it a production-ready and secure option for handling sensitive health data.[18]

The chosen architectural pattern is the **Queue-based Load Leveling pattern**.[11] The FastAPI backend acts as the

producer, publishing a message with the patient data. The various AI models act as

consumers, processing messages from the queue. This design ensures that a surge in submissions will not overwhelm the AI models; instead, tasks will queue up and be processed as resources become available. It also allows the AI services to be scaled, updated, or even fail and restart independently without losing any patient data or affecting the availability of the ingestion API.[12]

System	Architecture Type	Key Features	Message Ordering	Best Suited For
Apache Kafka	Distributed Log	High-throughput streaming, Data persistence & replayability [14]	Guaranteed at partition level [16]	Real-time analytics, Event sourcing
RabbitMQ	Message Broker	Complex routing, Message priorities, Push-based delivery [13]	Guaranteed at queue level (with single consumer) [16]	Task queuing, Microservice communication
GCP Pub/Sub	Managed Pub/Sub	Global scale, Push & Pull, HIPAA compliant [18]	Partial ordering via keys [19]	Event-driven architectures, Decoupling services

1.3. The Persistence Layer: A Schema for Evolving Clinical Questionnaires

A core requirement of the system is the ability to handle questionnaires that change over time. Storing this dynamic data correctly is critical for maintaining data integrity, auditability, and the ability to reproduce historical results.

The "Immutable Snapshot" Pattern

Dynamically altering a database schema (e.g., adding columns to a table) in response to application-level changes is an anti-pattern that leads to a brittle and unmanageable system.[22] The most robust and academically sound approach is to treat each published version of a questionnaire as an

immutable snapshot. When a questionnaire is modified and published, a new, complete set of versioned records is created in the database, while the old version remains untouched.[24] This ensures that every patient submission is permanently linked to the exact questionnaire version they completed, which is essential for clinical and research purposes.

This schema will be implemented in a relational database like **PostgreSQL**, chosen for its transactional integrity, reliability, and advanced features. All changes to the database schema itself will be managed as code using a migration tool such as **Alembic**.[25] This aligns with database versioning best practices, where schema changes are scripted, stored in a version control system (like Git), and applied automatically, ensuring consistency across all environments.[26]

Figure 1: Proposed Database Schema for Versioned Questionnaires

The following schema design implements the immutable snapshot pattern. The relationships ensure that once a questionnaire version is published and used, its structure is locked, and patient answers are tied directly to that specific version.

- **questionnaire_templates**
  - id (PK): Unique identifier for the questionnaire concept.
  - name: The name of the questionnaire (e.g., "Initial Oral Lesion Assessment").
  - description: A brief description of its purpose.
- **questionnaire_versions**
  - id (PK): Unique identifier for a specific version.
  - template_id (FK to questionnaire_templates): Links to the master template.
  - version_number: An integer representing the version (e.g., 1, 2, 3).
  - publication_timestamp: When this version was made active.
  - is_active: A boolean flag to easily identify the current version to be served.
- **questions**
  - id (PK): Unique identifier for a question.
  - questionnaire_version_id (FK to questionnaire_versions): **Crucially, links a question to a specific version, not the template.**
  - text: The full text of the question.
  - question_type: e.g., 'free_text', 'multiple_choice', 'single_choice'.

    - order: Integer to define the sequence of questions.
- **question_options**
    - id (PK): Unique identifier for an answer option.
    - question_id (FK to questions): Links the option to its parent question.
    - text: The display text for the option (e.g., "Yes", "No").
    - value: The stored value for the option.
- **patient_submissions**
    - id (PK): Unique identifier for a completed submission.
    - patient_identifier: An anonymized identifier for the patient.
    - submission_timestamp: When the form was submitted.
    - questionnaire_version_id (FK to questionnaire_versions): **The critical link that ties a submission to an immutable questionnaire version.**
- **patient_answers**
    - id (PK): Unique identifier for a single answer.
    - submission_id (FK to patient_submissions): Links the answer to a submission.
    - question_id (FK to questions): Links to the specific question being answered.
    - selected_option_id (FK to question_options, nullable): For choice-based questions.
    - free_text_answer (TEXT, nullable): For free-text questions.

1.4. The Intelligence Core: A Chained Multimodal Inference Pipeline

To effectively combine insights from both textual and visual data, the system will employ a chained inference architecture. This sequential pattern allows the output of one model to provide valuable context for the next, mimicking a more sophisticated clinical reasoning process rather than simply fusing two independent analyses at the end.[27]

- **Step 1: LLM-based Textual Analysis:** The textual data from the patient's questionnaire submission is first processed by an LLM. This model will be prompted to act as a clinical assistant, transforming the raw answers into a structured summary. This aligns with research demonstrating the utility of LLMs in summarizing clinical notes and extracting key information.[7] The LLM's output will be a JSON object containing:
    - A concise clinical summary.
    - A list of extracted clinical entities (symptoms, duration, location, etc.).
    - A preliminary risk score based solely on the textual information.
- **Step 2: Context-Aware Multimodal Analysis:** The structured JSON output from the LLM is then passed, along with the patient-submitted image, to the multimodal model. This provides the visual model with rich, pre-processed context, effectively guiding its analysis. This pattern is a form of a cross-attention mechanism, where textual features direct the model's focus on relevant visual areas.[27] The multimodal model, which could be a fine-tuned vision-language model (VLM) like OralGPT or a model based on the CLIP

architecture, will then generate its own verdict, including probability scores for various oral conditions and identifying visual evidence that corroborates the textual symptoms.[31]

- **Step 3: Aggregation and Final Verdict:** The outputs from both the LLM and the multimodal model are fed into a final aggregation service. This service can be a simple rule-based system or a small, trained model. It will weigh the preliminary risk scores and confidence levels from both models to compute a final, overall triage priority (e.g., 1-Urgent, 2-High Priority, 3-Routine). This final priority level is then attached to the message, which can be used by the message queue (if using RabbitMQ) or by the consuming application to sort and display cases for human review.

1.5. System Overview Diagram

Figure 2: End-to-End System Architecture

The complete system architecture integrates the components described above into a cohesive, end-to-end data processing pipeline. The diagram would illustrate the following flow:

1. A **Healthcare Professional** interacts with the **Chatbot Interface**.
2. The interface communicates with the **FastAPI Backend API**. The ConversationManager within the backend fetches the active questionnaire from the **PostgreSQL Database** and manages the data entry session.
3. Upon completion, the FastAPI backend packages the submission data (text and image reference) into a JSON message and publishes it to the **Message Queue (RabbitMQ)**.
4. A pool of **AI Consumer Services** listens to the queue.
5. **Consumer 1 (LLM Service)** picks up the message, processes the textual data, and publishes its structured JSON output back to a new topic or queue.
6. **Consumer 2 (Multimodal Service)** picks up the message containing the original data and the LLM's output. It performs its analysis and publishes its verdict.
7. **Consumer 3 (Aggregation Service)** consumes the outputs from both AI models, calculates the final triage priority, and stores the complete, enriched result in the database, ready to be displayed on a clinical dashboard.

This diagram visually represents the asynchronous and decoupled nature of the AI processing stage, which is the hallmark of a modern, scalable, and resilient event-driven architecture.[12]

Section 2: The Research Foundation: A Review of the State-of-the-Art

The design of this system is not only based on sound engineering principles but is also deeply informed by the latest academic research in medical AI. This section provides a review of the relevant literature, justifying the architectural choices and grounding the project in a solid scientific context.

2.1. Large Language Models in Clinical Triage and Decision Support

Recent research has extensively explored the application of LLMs in clinical settings, particularly for tasks related to triage and decision support. Systematic reviews show that models like GPT-3.5 and GPT-4 are being investigated for answering medical questions, summarizing clinical texts, and assisting with documentation.[7] Studies specifically focused on emergency medicine and triage have demonstrated that LLMs can accurately classify patient acuity levels from clinical text, with performance sometimes comparable to human physicians.[30] For example, one study found that an LLM could correctly infer the patient with higher acuity in 89% of paired cases.[34] These findings support the use of an LLM in our pipeline to analyze and structure the textual data from the questionnaire.

However, the literature also consistently highlights significant limitations and risks. These include a lack of specific medical domain optimization, issues with reproducibility, the potential for generating factually incorrect information ("hallucinations"), and the perpetuation of biases present in the training data.[7] These well-documented risks strongly justify the system's design as a

decision support tool, not an autonomous decision-maker. The "human-in-the-loop" approach, where the AI's output is a recommendation for a human expert to review, is a critical safety and ethical safeguard.

2.2. Multimodal AI for Oral and Dental Disease Detection

The specific domain of oral health presents unique challenges for AI, particularly in the interpretation of medical imagery. Dental panoramic X-rays, for example, contain dense and overlapping anatomical structures, making automated analysis difficult.[31] Recent research has focused on developing specialized multimodal models to address these challenges.

A landmark contribution in this area is the **MMOral dataset and the OralGPT model**.[31] The authors created a large-scale, annotated dataset of panoramic X-rays paired with instruction-following text. Their evaluation of 64 existing large vision-language models (LVLMs) revealed significant limitations, with even the top-performing model, GPT-4o, achieving only 41.45% accuracy on their benchmark.[37] This starkly illustrates the necessity of domain-specific models and fine-tuning for achieving clinically relevant performance. Their fine-tuned OralGPT model showed a substantial 24.73% improvement over its baseline, reinforcing this conclusion.[37]

Another highly relevant piece of research is the **Clinical Semantic Intelligence (CSI) framework**.[32] This framework's philosophy is to explicitly emulate the cognitive reasoning of an expert clinician. Its architecture integrates a fine-tuned multimodal CLIP model with a specialized language model, using a "Hierarchical Diagnostic Reasoning Tree" to guide the diagnostic process.[32] This approach directly inspires our proposed "chained" inference pipeline, where the LLM's initial analysis provides context that guides the subsequent, more specialized visual analysis. The success of the CSI framework, which saw accuracy jump from 73.4% to 89.5% when using its full reasoning tree, validates the principle that deeper integration between modalities yields superior results.[32]

2.3. Publicly Available Datasets for Model Development

The development and validation of any AI model depend entirely on the availability of high-quality, well-annotated data. Fortunately, the research community has made several relevant datasets publicly available, which can be leveraged for this TCC project.

Dataset Name	Modality	Content Description	Sample Size	Annotation Details	Access/Reference
MMOral	Panoramic X-rays, Text	Annotated panoramic X-rays paired with	20,563 images	Annotations for anatomical structures,	[31]

		1.3 million instruction-following instances for tasks like VQA and report generation.		pathologies, and treatments.	
Zenodo Oral Cancer	Images (Mobile Phone)	Images of oral cavities from 714 patients, categorized as healthy, benign, OPMD, and oral cancer.	3,000 images	Polygonal boundary annotations for lesions and oral cavity; patient metadata included.	[40]
Kaggle Oral Lesions	Images (Mobile/Intraoral)	Color images of benign and malignant oral lesions collected in consultation with doctors.	323 images	Classified as benign (165) or malignant (158).	[42]
DENTEX	Panoramic X-rays	Hierarchically annotated X-rays for quadrant, tooth, and diagnosis detection.	>1000 fully labeled	Includes labels for caries, deep caries, periapical lesions, and impacted teeth.	[43]
Tufts Dental	Panoramic	Panoramic X-rays with	1,000	Expert labeling of	[44]

Database	X-rays, Text	expert labeling and multimodal data including radiologist's eye-tracking and think-aloud protocol.	images	abnormalities and teeth across five classification levels.	

Section 3: Implementation, Validation, and Ethical Considerations

Building a functional AI system is only part of the challenge. For a medical application, a rigorous framework for validation and a deep understanding of the ethical and regulatory landscape are paramount.

3.1. A Framework for Model Validation and Performance Evaluation

Evaluating a clinical triage system requires metrics that go beyond simple accuracy, as the cost of different types of errors is not equal. A false negative (missing an urgent case) is far more dangerous than a false positive (flagging a non-urgent case as urgent).[45]

Defining Key Metrics

The system's performance will be evaluated using a suite of standard clinical AI metrics:

- **Sensitivity (Recall):** This measures the proportion of actual positive cases that are correctly identified (e.g., $TP / (TP + FN)$). In this context, it represents the system's

ability to correctly flag urgent cases. High sensitivity is critical to minimize the risk of missing patients who need immediate attention.[45]

- **Specificity:** This measures the proportion of actual negative cases that are correctly identified (e.g., $TN / (TN + FP)$). It reflects the system's ability to correctly identify non-urgent cases, which is important for efficient resource allocation.[45]
- **Positive Predictive Value (PPV) / Precision:** This measures the proportion of positive predictions that were actually correct (e.g., $TP / (TP + FP)$). It answers the question: "Of all the patients the system flagged as urgent, how many actually were?".[45]
- **Area Under the Receiver Operating Characteristic Curve (AUROC):** This metric provides a single score that summarizes the model's performance across all possible classification thresholds. It represents the probability that the model will rank a randomly chosen positive instance higher than a randomly chosen negative one, providing a robust measure of its overall discriminative power.[46]

Validation Protocol

A formal validation protocol is necessary to ensure the system is safe and effective. This protocol should be aligned with regulatory frameworks for Software as a Medical Device (SaMD), such as those proposed by the U.S. FDA.[48] The key steps include:

1. **Defining the Intended Use:** Clearly state that the system is a clinical decision support tool for triaging oral health cases, intended for use by qualified healthcare professionals.
2. **Pre-specification of Performance Objectives:** Before testing, establish clear, quantitative goals for the key metrics (e.g., "Sensitivity for 'Urgent' cases must be > 95%").
3. **Validation on a Held-Out Test Set:** The model must be evaluated on a dataset that was not used during training or tuning. This test set should be curated from the publicly available datasets identified in Section 2.3.
4. **Real-World Performance Monitoring:** In a production scenario, the system's performance would need to be continuously monitored to detect any degradation due to data drift (e.g., changes in patient populations or disease prevalence).[49]

3.2. Navigating the Regulatory and Ethical Landscape in Brazil

As the system will process sensitive personal health information, it must be designed and operated in strict compliance with Brazil's **Lei Geral de Proteção de Dados (LGPD)**.

Compliance with LGPD

The LGPD establishes a legal framework for the processing of personal data.[50] Key compliance requirements for this system include:

- **Legal Basis:** Processing of "sensitive personal data," which includes health data, requires a specific legal basis, such as the explicit consent of the data subject or for the protection of life.
- **Privacy by Design:** The system must be built with data protection principles in mind from the outset. This includes measures like data minimization (collecting only necessary information) and robust security controls (encryption in transit and at rest).
- **Data Subject Rights:** The system must be able to facilitate user rights, including the right to access, correct, and delete their data.
- **Transparency:** The LGPD recognizes a "right to explanation" regarding automated decisions.[51] This is a critical consideration for AI systems.

Ethical Considerations

Beyond legal compliance, several ethical principles must guide the system's development and deployment:

- **Algorithmic Bias:** AI models can learn and amplify biases present in their training data, potentially leading to health disparities.[36] It is essential to analyze the demographic and clinical characteristics of the training datasets (Section 2.3) to identify potential sources of bias and to evaluate the model's performance across different subgroups.
- **Transparency and Explainability:** While deep learning models are often considered "black boxes," the proposed chained architecture provides a degree of inherent explainability. By exposing the intermediate outputs—the LLM's textual summary and the multimodal model's visual findings—the system can show the human reviewer the evidence that contributed to the final triage recommendation. This helps address the "right to explanation" and builds clinician trust.[51]
- **Human-in-the-Loop:** The most critical ethical safeguard is the unwavering commitment to a **human-in-the-loop** model. The AI system is designed to augment, not replace, the clinical judgment of a healthcare professional. The final triage decision and subsequent care plan must always be made by a qualified human expert. This not only mitigates the risk of AI error but also aligns with the expectations of both regulators and patients.[52]

Conclusion: A Roadmap for TCC Success and Future Directions

This report has laid out a detailed and research-informed blueprint for an innovative multimodal AI system for oral health triage. The proposed architecture is robust, scalable, and tailored to the specific demands of a medical application, leveraging a modern technology stack centered on FastAPI, a message-driven pipeline, and a version-controlled database schema. Each architectural decision is justified not only by technical merit but also by its alignment with best practices for building safe and reliable clinical support tools.

The "chained" AI inference pipeline, inspired by state-of-the-art research frameworks like Clinical Semantic Intelligence (CSI), represents a sophisticated approach to multimodal fusion that promises greater accuracy and explainability than simpler methods. By grounding the project in a thorough review of academic literature and publicly available datasets, this plan provides a clear path for the research component of the TCC.

For the successful implementation of this project, the immediate next steps are to begin the development of the core components: setting up the FastAPI backend, designing the database schema using Alembic for migrations, and establishing the message queue infrastructure. Concurrently, work should begin on acquiring and preprocessing data from the identified public datasets to train and validate the initial versions of the LLM and multimodal models.

Looking forward, this project opens several avenues for future research and development. The system's modular design allows for expansion into other medical domains beyond oral health. Further work could explore the integration of additional data modalities, such as audio recordings of patients describing their symptoms. Finally, upon successful validation in a research setting, a long-term goal could be to conduct a prospective clinical study to evaluate the system's real-world impact on triage efficiency, resource allocation, and ultimately, patient outcomes. This TCC project serves as a foundational step toward building next-generation AI tools that can safely and effectively support healthcare professionals and improve patient care.

Appendix: Curated Bibliography and References

A comprehensive list of all cited research papers, technical documentation, and articles can be compiled from the source identifiers provided throughout this report (e.g.[4], etc.).

Works cited

1. Best Python Frameworks for Scalable Web App Development in 2025 - Zestminds, accessed September 21, 2025, https://www.zestminds.com/blog/best-python-frameworks-web-app-2025/
2. Django vs FastAPI: Choosing the Right Python Web Framework ..., accessed September 21, 2025, https://betterstack.com/community/guides/scaling-python/django-vs-fastapi/
3. Django vs. FastAPI: When to Stick With the Classic & When to Go Fast | by Priyanshu Rajput | Python in Plain English - Medium, accessed September 21, 2025, https://medium.com/python-in-plain-english/django-vs-fastapi-when-to-stick-with-the-classic-when-to-go-fast-025bdd7b746f
4. Popular Python Web Frameworks to Use in 2025 - Analytics Vidhya, accessed September 21, 2025, https://www.analyticsvidhya.com/blog/2025/04/python-web-frameworks/
5. Flask, Django, or FastAPI? : r/Python - Reddit, accessed September 21, 2025, https://www.reddit.com/r/Python/comments/1dxcdiy/flask_django_or_fastapi/
6. "Is FastAPI really better than Django in 2024?" — the opinion of the developer Avivi, accessed September 21, 2025, https://avivi.pro/en/blog/is-fastapi-really-better-than-django-in-2024-the-opinion-of-the-developer-avivi/
7. Current applications and challenges in large language models for ..., accessed September 21, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC11751060/
8. FastAPI vs. Django REST Framework: Which One Should You Choose? - YouTube, accessed September 21, 2025, https://www.youtube.com/watch?v=bGw9An9rI18
9. A Guide to Everything You Need to Know About Rasa Chatbot ..., accessed September 21, 2025, https://www.droxy.ai/blog/rasa-chatbot
10. Healthcare Chatbot using RASA - Zenodo, accessed September 21, 2025, https://zenodo.org/records/6395568/files/Healthcare%20Chatbot%20-Formatted%20Paper.pdf
11. Asynchronous messaging options - Azure Architecture Center - Microsoft Learn, accessed September 21, 2025, https://learn.microsoft.com/en-us/azure/architecture/guide/technology-choices/messaging
12. How to Architect a Scalable Data Pipeline for HealthTech Applications, accessed September 21, 2025, https://www.datasciencecentral.com/how-to-architect-a-scalable-data-pipeline-for-healthtech-applications/
13. Kafka vs RabbitMQ: Key Differences & When to Use Each | DataCamp, accessed September 21, 2025, https://www.datacamp.com/blog/kafka-vs-rabbitmq
14. RabbitMQ vs. Apache Kafka | Confluent, accessed September 21, 2025,

https://www.confluent.io/learn/rabbitmq-vs-apache-kafka/
15. Kafka Vs RabbitMQ: Key Differences & Features Explained - Simplilearn.com, accessed September 21, 2025, https://www.simplilearn.com/kafka-vs-rabbitmq-article
16. Apache Kafka vs. RabbitMQ: Comparing architectures, capabilities, and use cases - Quix, accessed September 21, 2025, https://quix.io/blog/apache-kafka-vs-rabbitmq-comparison
17. What's the Difference Between Kafka and RabbitMQ? - AWS, accessed September 21, 2025, https://aws.amazon.com/compare/the-difference-between-rabbitmq-and-kafka/
18. google cloud pub sub vs aws sqs simple queue service: Which Tool is Better for Your Next Project? - ProjectPro, accessed September 21, 2025, https://www.projectpro.io/compare/google-cloud-pub-sub-vs-aws-sqs-simple-queue-service
19. Choosing your message queue AWS SQS or GCP Pub/Sub - Blog NivelEpsilon, accessed September 21, 2025, https://nivelepsilon.com/2025/07/06/choosing-your-message-queue-aws-sqs-or-gcp-pub-sub/
20. Amazon SQS vs Google Cloud Pub/Sub: which should you choose in 2025? - Ably, accessed September 21, 2025, https://ably.com/compare/amazon-sqs-vs-google-pub-sub
21. Event-Driven Architecture (EDA): A Complete Introduction - Confluent, accessed September 21, 2025, https://www.confluent.io/learn/event-driven-architecture/
22. Database design for a survey [closed] - sql - Stack Overflow, accessed September 21, 2025, https://stackoverflow.com/questions/1764435/database-design-for-a-survey
23. (PDF) Towards a Form Based Dynamic Database Schema Creation and Modification System - ResearchGate, accessed September 21, 2025, https://www.researchgate.net/publication/256648401_Towards_a_Form_Based_Dynamic_Database_Schema_Creation_and_Modification_System
24. Designing Schema for Versioning Forms and Form Completion Attempts : r/SQL - Reddit, accessed September 21, 2025, https://www.reddit.com/r/SQL/comments/1acl5od/designing_schema_for_versioning_forms_and_form/
25. What is Schema Versioning in DBMS? - GeeksforGeeks, accessed September 21, 2025, https://www.geeksforgeeks.org/dbms/what-is-schema-versioning-in-dbms/
26. Database versioning best practices - Enterprise Craftsmanship, accessed September 21, 2025, https://enterprisecraftsmanship.com/posts/database-versioning-best-practices/
27. Architectural Paradigms for Multimodal Large Language Models | by ..., accessed September 21, 2025, https://medium.com/@zbabar/architectural-paradigms-for-multimodal-large-language-models-8955ffe227dc
28. Analyzing Diagnostic Reasoning of Vision–Language Models via Zero-Shot

Chain-of-Thought Prompting in Medical Visual Question Answering - MDPI, accessed September 21, 2025, https://www.mdpi.com/2227-7390/13/14/2322
29. Potential applications and implications of large language models in primary care - PMC, accessed September 21, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC10828839/
30. Evaluating large language model workflows in clinical decision support for triage and referral and diagnosis - PMC, accessed September 21, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC12064692/
31. [2509.09254] Towards Better Dental AI: A Multimodal Benchmark and Instruction Dataset for Panoramic X-ray Analysis - arXiv, accessed September 21, 2025, https://arxiv.org/abs/2509.09254
32. [2507.15140] Clinical Semantic Intelligence (CSI): Emulating the Cognitive Framework of the Expert Clinician for Comprehensive Oral Disease Diagnosis - arXiv, accessed September 21, 2025, https://arxiv.org/abs/2507.15140
33. Large Language Models in Healthcare and Medical Applications: A Review - PMC, accessed September 21, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC12189880/
34. Use of a Large Language Model to Assess Clinical Acuity of Adults in the Emergency Department, accessed September 21, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC11077390/
35. Patient Triage and Guidance in Emergency Departments Using Large Language Models: Multimetric Study - PMC, accessed September 21, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC12123234/
36. Large language models in patient education: a scoping review of applications in medicine - PMC, accessed September 21, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC11554522/
37. Towards Better Dental AI: A Multimodal Benchmark and Instruction Dataset for Panoramic X-ray Analysis - arXiv, accessed September 21, 2025, https://arxiv.org/html/2509.09254v1
38. [Revue de papier] Towards Better Dental AI: A Multimodal Benchmark and Instruction Dataset for Panoramic X-ray Analysis - Moonlight, accessed September 21, 2025, https://www.themoonlight.io/fr/review/towards-better-dental-ai-a-multimodal-benchmark-and-instruction-dataset-for-panoramic-x-ray-analysis
39. Clinical Semantic Intelligence (CSI): Emulating the Cognitive Framework of the Expert Clinician for Comprehensive Oral Disease Diagnosis - arXiv, accessed September 21, 2025, https://arxiv.org/html/2507.15140v1
40. A comprehensive dataset of annotated oral cavity images for diagnosis of oral cancer and oral potentially malignant disorders - The University of Aberdeen Research Portal, accessed September 21, 2025, https://abdn.elsevierpure.com/en/publications/a-comprehensive-dataset-of-annotated-oral-cavity-images-for-diagn
41. Dataset of Annotated Oral Cavity Images for Oral Cancer Detection - Zenodo, accessed September 21, 2025, https://zenodo.org/doi/10.5281/zenodo.10664056
42. Oral Lesions: Malignancy Detection Dataset - Kaggle, accessed September 21,

2025, https://www.kaggle.com/datasets/mohamedgobara/oral-lesions-malignancy-detection-dataset
43. Dental Enumeration and Diagnosis on Panoramic X-rays - Dentex - Grand Challenge, accessed September 21, 2025, https://dentex.grand-challenge.org/data/
44. Tufts Dental Database: A Multimodal Panoramic X-Ray Dataset for Benchmarking Diagnostic Systems | National Institute of Justice, accessed September 21, 2025, https://nij.ojp.gov/library/publications/tufts-dental-database-multimodal-panoramic-x-ray-dataset-benchmarking
45. On evaluation metrics for medical applications of artificial intelligence - medRxiv, accessed September 21, 2025, https://www.medrxiv.org/content/10.1101/2021.04.07.21254975v1.full-text
46. Monitoring performance of clinical artificial intelligence in health ..., accessed September 21, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC11630661/
47. How AI is changing emergency room triage efficiency | Matada Research, accessed September 21, 2025, https://matadaresearch.co.nz/ai-emergency-efficiency/
48. US FDA Artificial Intelligence and Machine Learning Discussion Paper, accessed September 21, 2025, https://www.fda.gov/files/medical%20devices/published/US-FDA-Artificial-Intelligence-and-Machine-Learning-Discussion-Paper.pdf
49. Evaluating AI-Enabled Clinical Decision and Diagnostic Support Tools Using Real-World Data, accessed September 21, 2025, https://healthpolicy.duke.edu/sites/default/files/2022-03/Evaluating%20AI-Enabled%20Clinical%20Decision%20Diagnostic%20Support%20Tools%20Using%20Real-World%20Data.pdf
50. The use of AI in digital health services and privacy regulation in ..., accessed September 21, 2025, https://www12.senado.leg.br/ril/edicoes/59/233/ril_v59_n233_p201.pdf
51. The regulation of artificial intelligence for health in Brazil begins with the General Personal Data Protection Law - PubMed, accessed September 21, 2025, https://pubmed.ncbi.nlm.nih.gov/36043658/
52. Artificial intelligence and clinical decision support: clinicians' perspectives on trust, trustworthiness, and liability | Medical Law Review | Oxford Academic, accessed September 21, 2025, https://academic.oup.com/medlaw/article/31/4/501/7176027
53. The Evolution of AI in Clinical Decision Support Systems | IntuitionLabs, accessed September 21, 2025, https://intuitionlabs.ai/articles/ai-clinical-decision-support-evolution