`

GenMedX: Generative AI Healthcare Agent

Hamsalakshmi Ramachandran, Himani Shah, Kush Bindal, Shobhita Agrawal and Sugandha Chauhan

San Jose State University

MSDA Project I Section 12 DATA 298A

Dr. Simon Shim

Spring 2025

GenMedX: Generative AI Healthcare Agent

1. Introduction

1.1 Project Background and Executive Summary

Project Background, Needs, and Importance

The healthcare system encounters major obstacles due to the aging demographic and rising healthcare expenses in every phase. Existing healthcare systems frequently fail to deliver personalized, ongoing care outside of clinical environments, leading to lapses in patient oversight, postponements in action, and ineffective resource distribution.

The demand for smart healthcare solutions is becoming more urgent as healthcare providers face challenges with staffing shortages, escalating costs, and increasing patient numbers. Data indicates that 30% of sudden complications could be avoided through improved medical education and preparation (PMC Article 2024) whereas post ED readmissions cost the US healthcare system more than $17 billion each year (Healthcare Cost and Utilization Project). As the population continues to grow leaps and bounds, the demand for healthcare services will increase beyond what the current healthcare system can cater to.

This project is important because it uses AI to provide practical solutions to these urgent healthcare challenges. By developing dedicated agents for the Emergency Department (ED), the system can offer ongoing support that would typically demand extensive human resources, enhancing the accessibility and affordability of quality healthcare.

Targeted Project Problem, Motivations and Goals

The targeted problem is the care division throughout key healthcare phases like ED stay, post ED stay follow-up and clinical explanation. Leading to inefficiencies, a lack of medical understanding, gaps in monitoring, and avoidable complications. Present solutions usually tackle these phases separately, overlooking chances for combined care strategies. Our

primary motivations encompass enhancing patient outcomes via prompt, tailored advice, alleviating healthcare professionals' workloads through intelligent automation, minimizing avoidable complications and hospital readmissions, improving the quality of life for seniors through ongoing assistance, and showcasing effective uses of generative AI in the healthcare sector. The project aims to design tailored AI agents for clinical explanation, ED stay summarizer and post ED stay follow-up; constructing a scalable, affordable solution utilizing publicly available datasets and open-source models; attain significantly improved results relative to current solutions in accuracy, user satisfaction, and clinical effectiveness; and tackle the integration difficulties of varied medical data formats to guarantee practical applicability.

Project Approaches and Methods

The project refines existing LLMs like Llama 3 and Mistral using healthcare-targeted datasets, subsequently creating tailored agents for each ED phase. The Clinical Explanation Agent employs a Bio_ClinicalBERT model that has been fine-tuned on the NCBI Disease Corpus to accurately carry out NER and normalization of medical entities—such as diseases, medications, and procedures—in clinician prompts as well as patient ED records. After the entities are identified, the agent obtains authoritative descriptions and associated concepts from the NCBI dataset and UMLS ontologies, maintaining semantic coherence. A LoRA-augmented generation module subsequently compiles these insights into brief, accessible explanations for patients that simplify intricate terminology, describe common disease trajectories, and underscore essential risk factors. Every explanation is tagged with confidence scores and direct references to the source definitions, allowing clinicians to confirm accuracy and patients to participate more actively in their care. In the ED stay summarizer phase, the goal is to refine Llama 3 using MedDialog, PubMedQA and Pubmed

Central to develop a system that integrates clinical language understanding with patient-specific information retrieval to generate medical summaries. The post ED stay follow-up phase will address the gap between emergency care and long-term patient management. The solution will be developed with publicly accessible datasets to ensure accessibility and scalability. The development method will adopt an agile strategy with ongoing testing and improvement driven by feedback from healthcare professionals and utilize evaluation metrics tailored for healthcare AI, incorporating clinical validity assessments and technical performance metrics.

Expected Project Contributions and Applications

The expected contributions of the project include better patient summaries through prompt interventions and tailored advice and fewer hospital readmissions with the aid of follow-up instructions. The prototype will advance AI systems in healthcare by demonstrating practical applications for integrating knowledge. It will leverage innovative strategies to merge various healthcare data formats using knowledge graph embeddings, ensuring accurate and context-aware responses to patient queries. The system will also showcase the practical use of generative AI in healthcare, addressing real-world implementation challenges that have previously hindered the adoption of AI-driven diagnostic support systems.

1.2 Project Requirements

Functional Requirements

The system should offer detailed clinical explaantions, ED summaries and post stay follow-up instructions. It should be able to oversee recovery from ED by assessing patient vitals, adherence to medication, pain intensity to detect possible complications before they

escalate. The system should adjust to gradual changes in circumstances, differentiating between normal aging trends and worrying declines that necessitate action.

An essential role is the smooth merging of insights from various specialized agents to deliver thorough care recommendations that consider the patient's complete healthcare experience. To guarantee accessibility, the system must have an easy-to-use interface that offers interaction options appropriate for healthcare providers and patients/caregivers with different levels of technical skill.

AI-powered Feature Requirements

The system demands advanced natural language question-answering skills on healthcare inquiries, encompassing the competence to manage ambiguous medical terms, identify non-expert descriptions of symptoms, comprehend variations in regional terminology, and address intricate, multi-faceted questions. The NLP component should identify and effectively react to medical urgency in patient queries, emphasizing crucial issues and offering suitable triage suggestions. The system should show resilience to spelling mistakes, grammatical errors, and incomplete inquiries that are typical in everyday healthcare conversations. For ED follow-up care, the system requires strong anomaly detection algorithms that can accurately identify atypical recovery patterns with precision and recall rates even when data inputs are incomplete. This involves identifying subtle early indicators of frequent post-ED issues like infection (rising temperature patterns, heightened pain), bleeding (variations in vital signs, symptoms reported by the patient), thrombosis (swelling, localized discomfort), or negative medication responses utilizing multivariate pattern recognition. The system needs to integrate temporal reasoning to differentiate between anticipated recovery progress and alarming deviations according to procedure-specific recovery paths.

The AI needs to produce deeply tailored suggestions grounded in unique patient profiles, taking into account age, existing health issues (especially diabetes, heart conditions, and immunosuppression), interactions between medications including both prescribed and non-prescription drugs, lifestyle elements, and past negative incidents, with suggestions adapting according to recovery pattern data gathered over time.

The system requires advanced contextual understanding to retain conversation history over several sessions, recall patient-specific information throughout various interactions without needing to repeat essential details, and modify replies according to the patient's shown health literacy level with flexible adaptation of terminology intricacy. The system should uphold an ongoing learning structure that detects instances where model performance dips beneath set thresholds, marking these for evaluation by human experts while safeguarding sensitive patient information.

Data Requirements

The MIMIC-IV ED dataset found on Kaggle features de-identified clinical information from patients in intensive care units, providing authentic instances of medical records and clinical decision-making. This dataset comprises vital signs, medications, lab measurements, observations, and notes from healthcare professionals, providing our model with exposure to real clinical language and patterns. Utilizing this demo version enables us to refine and evaluate our method using authentic medical data while adhering to data usage regulations, as this version is tailored for educational objectives.

The full MIMIC-IV ED dataset provided by PhysioNet is an extensive clinical database containing detailed information for approximately 425,000 emergency department (ED) visits. The dataset consists of six relational tables that collectively document the entire course of emergency department treatment from admission to discharge. The edstays table

includes 425,087 distinct ED encounters, each marked with a stay_id; the triage table reflects this with 425,087 matching assessment entries (vital signs, chief complaints, acuity scores); vitalsign features 1,564,610 time-stamped observations recorded during those stays; diagnosis stores 899,050 discharge diagnoses classified in ICD-9/10; medrecon records 2,987,342 medication-reconciliation entries detailing pre-ED medications; and pyxis logs 1,586,053 administered-medication occurrences. By showcasing each ED visit as a "star" at the core of this schema, we maintain the detail of individual observations while also allowing for their reconstruction into patient-focused stories.

For our Named Entity Recognition module, we utilize the NCBI Disease Corpus. It consists of 793 PubMed abstracts that are fully annotated at both mention and concept levels, resulting in 6,892 disease mentions linked to 790 distinct disease concepts (88 % MeSH, 12 % OMIM). Fourteen annotators, randomly assigned in two annotation phases, attained over 80% span and category agreement after adjudication, indicating exceptionally high annotation quality. The dataset is divided into 593 abstracts for training, 100 for development, and 100 for testing; for token-level NER we transform it into CoNLL-style sentences—yielding 5,433 training, 924 validation, and 941 test instances. By preserving both mention-level spans and their standardized concept IDs, the NCBI dataset facilitates strong entity recognition as well as subsequent concept-normalization activities.

These supplementary data sources will equip our GenAI healthcare agent with extensive medical knowledge from published literature and particular clinical patterns from actual patient data, establishing a basis for constructing a system capable of comprehending and producing suitable healthcare-related responses.

PubMed Central offers several APIs for building healthcare AI agents. The NCBI E-utilities (EFetch, ESummary) provide programmatic access to retrieve articles and

metadata. PubMed's RESTful API allows structured queries by medical terms, date ranges, and journals. Data extraction involves parsing the XML/JSON responses, removing HTML tags and formatting artifacts. The cleaned text is then processed through medical-specific Named Entity Recognition (NER) to identify and tag entities like diseases, medications, symptoms, and procedures. These structured annotations create high-quality training datasets that preserve medical relationships and terminology, enabling more effective fine-tuning of LLMs for healthcare applications.

1.3 Project Deliverables

The Generative AI Healthcare Agent initiative will produce several essential outputs, including detailed reports, operational prototypes, development tools, and systems ready for production. These outcomes are intended to showcase the abilities of the healthcare agents while maintaining organized advancement and clear assessment during the project duration.

End goal is to create independent healthcare AI agents which are able to help and support patients through their tough times in a well-rounded manner. We are working towards building two different agents to address the following specific needs:

Clinical Explanation Agent

The Clinical Explanation Agent hears medical inquiries—either from patients or healthcare providers—recognizes the essential terms in simple language, and subsequently provides straightforward, easily understandable explanations of those conditions. Utilizing a reputable disease reference, it explains the significance of each diagnosis, points out typical symptoms and risk factors, and offers considerations for subsequent steps. All explanations contain source references, allowing clinicians to confirm accuracy and assisting patients in grasping their health better.

Emergency Department Stay Summarizer Agent

This agent processes comprehensive ED visit records—including triage assessments, vital signs, physician notes, and medication logs—to generate clear, coherent narrative summaries of each stay. It leverages temporal understanding to sequence events, medical entity recognition to identify diagnoses and treatments, and a longitudinal patient view to highlight key clinical milestones .

Post-ED Stay Follow-up Agent

This agent supports patients after ED discharge by creating personalized follow-up plans, scheduling appointments, and monitoring readmission risk. Using risk stratification, it identifies high-risk patients, issues timely reminders, and provides guidance on medication adherence and red-flag symptoms—thereby enhancing continuity of care and aiming to reduce avoidable readmissions

The Clinical Explanation Agent hears medical inquiries—either from patients or healthcare providers—recognizes the essential terms in simple language, and subsequently provides straightforward, easily understandable explanations of those conditions. Utilizing a reputable disease reference, it explains the significance of each diagnosis, points out typical symptoms and risk factors, and offers considerations for subsequent steps. All explanations contain source references, allowing clinicians to confirm accuracy and assisting patients in grasping their health better.

Foundational Development & Transfer Learning: Core NLP Capabilities for GenMedX

The active deployment of Natural Language Processing, at the core of the GenMedX initiative, which encompasses the ED Stay Summarizer and the Post-ED Stay Follow-up Agent, is a crucial element. The main components include the robust Medical Entity

Recognition (NER), and the comprehensive understanding of how different modeling approaches deal with complex medical language. Precise identification of fundamental clinical concepts like diagnoses and treatments underlies the system.

We started a dedicated foundational development phase to build solid foundational models and thoroughly validate critical methodologies before expanding to the entire GenMedX dataset suite (MIMIC-IV ED, MedDialog, PubMedQA). To support this foundational effort, we turned to the high-quality, publicly available NCBI Disease Corpus (793 PubMed abstracts) for validation. This allowed us to compare supervised fine-tuning with state-of-the-art prompting techniques, leading to important insights and pre-trained models that will be used to augment the primary GenMedX agents using transfer learning.

High-Performance Supervised Named Entity Recognition Baseline by Transfer Learning

Objective Alignment (GenMedX): Precise "medical entity recognition" is crucial for the ED Summarizer. Evaluation of the domain-specific models' maximum potential is a significant factor.

Methodology & Transfer Learning: We employed BioClinicalBERT. This model was created using transfer learning, first trained on large amounts of biomedical data from PubMed and then fine-tuned on clinical notes from MIMIC-III. We then fine-tuned the model on the NCBI Disease Corpus, with the specific aim of improving disease name identification. It included routine preprocessing steps like WordPiece tokenization and BIO label alignment.

Relevance & Transferability to GenMedX:

Performance Benchmark: This version of BioClinicalBERT, trained on clinical data, serves as a benchmark that underlines the efficiency of specialized models developed via layered transfer learning.

Knowledge Transfer for GenMedX Models: The knowledge obtained through fine-tuning BioClinicalBERT, especially when it comes to handling medical jargon and fine-tuning optimization methods, can act as a guide for adjusting such approaches to more extensive models like Llama 3 and Mistral when they are applied to specific GenMedX datasets like MIMIC-IV If a hybrid approach is taken into account, the pretrained BioClinicalBERT weights can be used as a feature extractor or initial teacher for some components of the GenMedX agents.

Outcome: Tuning BioClinicalBERT gave excellent results with precision at 0.77 and recall at 0.84. (Explain F1=0.0 as previously discussed). These results show the high performance of transfer learning in our LLM-based approaches.

Evaluation of the Performance and Prompt Techniques (Knowledge Transfer for Prompt Development)

Objective Alignment (GenMedX): GenMedX plans to use open-source LLMs such as Llama 3 and Mistral, as well as prompt engineering. In this phase, we explored ways of teaching Mistral-7B-Instruct-v0.1 specifically for Named Entity Recognition (NER) on the NCBI Disease dataset.

Methodology (Systematic Prompt Evaluation)

- Zero-Shot (V1-Baseline, V2-Guidelines, V3-ErrorBased)
- Few-Shot (V4-5shot)
- Chain-of-Thought (V5-CoT)

- Setup: Experiments on Kaggle P100 GPU, with challenges in quantization informing resource planning for GenMedX.

Relevance & Transferability to GenMedX

The knowledge gained from Mistral-7B's answers to different structures of prompts (definitions, error corrections, few-shot examples, CoT) can be immediately used on Llama 3 and Mistral. These findings inform the strategy we adopt in generating initial prompts for Llama 3 and Mistral in the case of complex data such as MIMIC-IV ED notes or creating summaries for GenMedX. For example, the high performance of CoT (Relaxed F1 ~0.73 on sample) suggests it should be a critical strategy for the GenMedX agents.

Drawing conclusions from these challenges (e.g., boundary precision), we can figure out the post-processing or constraint techniques that are needed for these LLMs in the GenMedX environment.

Mistral-7B was best with Chain-of-Thought prompting (V5), suggesting that this method could be effectively used in larger GenMedX models and datasets.

Examining Retrieval-Augmented Generation (RAG) – Core Framework for GenMedX

GenMedX specifically develops a "retrieval-augmented protocol generator" and concentrates on "patient-specific information retrieval". At this stage, the central RAG pipeline was built and extensively tested. A collection of PubMed abstracts, selected and indexed with all-MiniLM-L6-v2 embeddings and FAISS and the relevant pieces of information are identified; and then attach them to the V2_Guidelines prompt before processing by Mistral-7B.

The developed RAG pipeline, from embedding to FAISS indexing, retrieval logic, and context injection, is a direct technical asset for GenMedX. Scaling and adapting this architecture, MIMIC-IV ED embeddings, MedDialog, or PubMedQA can be the knowledge

source for the agents of GenMedX. The early RAG performance (Relaxed F1 ~0.41, lagging behind non-RAG V2) highlights the impact of the quality, pertinence, and scope of the knowledge base on RAG outcomes. This finding will influence the arrangement of data for the GenMedX knowledge bases. A good RAG pipeline has been created. The constraints of the context findings highlight the need for comprehensive domain-specific knowledge bases that GenMedX's RAG systems can use.

Strategies for Overcoming Advanced LLM Fine-Tuning (PEFT) Challenges – Guiding GenMedX Transfer Learning Approach:

As the project entails the deployment of a "LoRA-enhanced Llama-2/Mistral model" and "Parameter-Efficient Fine-Tuning techniques, notably LoRA, "this experiment is very relevant.

Methodology & Challenges: We tried to apply parameter-efficient fine-tuning (LoRA) to the Mistral-7B model for generative Named Entity Recognition (NER). Encountered: quantization errors (cublasLt) on P100, meta tensor errors appeared when device_map="auto" was used and quantization was switched off. Direct loading of the model led to CUDA out of memory errors in the absence of a device_map or quantization.

Relevance & Transferability to GenMedX: Informing Transfer Learning Infrastructure: These lessons are critical to generating the transfer learning (PEFT/LoRA) strategy for Llama 3 and Mistral in GenMedX. It underscores the need for augmenting GPU with A100 or premium P100 setups that are built for advanced accelerate configurations. It is essential to monitor and manage the library versions (bitsandbytes, transformers, accelerate, torch) to ensure smooth compatibility particularly for the processes that involve quantization. This experience might help choose particular Llama 3 or Mistral variants for PEFT with a focus on those models showing better PEFT support or better compatibility with quantization.

It was not possible to implement PEFT on the P100 with the tested configurations. This practical experience directly influences our approach and planning of resources to deploy transfer learning (through PEFT) to the first-tier GenMedX LLMs.

Conclusion for this Foundational Phase & Path to GenMedX

This massive foundational work, based around the NCBI Disease corpus, has been crucial. With the support of transfer learning using BioClinicalBERT, we have established an outstandingly effective supervised benchmark. With proper experimentation of prompts, we have learnt a lot about the best way to guide such LLMs, Chain-of-Thought being one of the promising methods for NER. The RAG pipeline that we developed is a transferable component for GenMedX, and our initial results have given important insights into the quality of context. Furthermore, the challenges we faced with PEFT have influenced the architectural and library decisions needed for scheduled LoRA fine-tuning of Llama 3 and Mistral.

Reports

The project will generate a sequence of necessary academic outputs following the course framework. The **Abstract** will offer a brief summary of the project's goals, activities, results, and uses. This document lays the groundwork for our Generative AI Healthcare Agent initiative and details its primary goals. **Progress Report 1** will be presented next, outlining the initial development efforts, early findings, challenges faced during the initial phases, and modifications to our implementation strategy. **Workbook 1** functions as an all-inclusive project planning document, outlining the project background, requirements, technology assessment, literature review, and management strategy. **Progress Report 2** will be issued halfway through the project timeline, detailing major development milestones reached, outcomes from initial testing of the specialized agents, integration challenges faced,

and solutions that were applied. This report will contain preliminary performance metrics and user insights from initial prototype testing. **Workbook 2** will offer in-depth documentation of the technical execution, covering system architecture, component interactions, integration techniques, and initial evaluation outcomes. This document will contain network diagrams, API specifications, data flow charts, and system configuration information. The **Final Report** will function as the complete summary of the project, incorporating all discoveries, detailing the entire system implementation, showcasing extensive evaluation outcomes, and assessing the fulfillment of project goals. This report will evaluate the system's effectiveness in different healthcare situations, record user testing findings, tackle privacy and ethical issues, and highlight potential improvements for the future based on the project results. Collectively, these reports will offer a comprehensive academic record of the project's development from idea to execution

Prototypes

Our proof-of-concept pipeline comprises four interconnected stages: Named Entity Recognition, Vector-Store Construction, Retrieval, and Generative Response. Initially, we adjust Bio_ClinicalBERT by training it on the NCBI Disease Corpus for clinical NER. We import the 793 PubMed abstracts (6,892 disease mentions) using Hugging Face's datasets library, tokenize them with the AutoTokenizer, and train using Trainer + LoRA adapters (PEFT) in mixed-precision to reduce GPU memory usage. We assess using exact and relaxed match F_1 to guarantee high precision and recall in identifying disease entities and associating them with UMLS concepts. Subsequently, we incorporate our selected MIMIC-IV-ED subset—combining six tables into a single nested JSON for each stay—and create a FAISS index for semantic searching. The merged text of each ED visit (including triage assessments, vital signs logs, provider notes, diagnoses, and medication events) is embedded using

All-MiniLM-L6-v, resulting in an efficient, real-time vector database. When a query is made, a clinician's natural-language request is initially processed by the fine-tuned NER model to recognize and standardize medical entities. Those entities initiate a k-NN search using our FAISS index to obtain the most relevant encounters. Ultimately, we invoke a LoRA-enhanced generator—using a bespoke generate_answer(query, retrieved) wrapper ("medpipe")—employing a three-phase prompt format (analysis, justification, answer). The result is a brief, evidence-based reply accompanied by confidence scores and references to source documents. For generating explanations, we are utilizing the MedAlpaca-7B model, which is specifically trained on medical data. This modular prototype shows how instruction → NER → retrieval → generation can be coordinated using open-source models and free tools (FAISS-CPU, HuggingFace Transformers, SentenceTransformers, PEFT/LoRA) to provide quick, clear clinical insights from MIMIC-IV-ED data.

We also created a healthcare data pipeline utilizing Google Cloud Composer, which manages tasks via Apache Airflow DAGs. Data and workflow scripts are stored within a controlled Cloud Storage bucket, where structured raw datasets are downloaded into a designated *raw_data/* folder for consistent access. The *healthcare_data_pipeline* DAG manages the processes of ingestion, extraction, transformation, and loading into BigQuery, employing Airflow operators for dependable and verifiable execution. This configuration guarantees modularity, reproducibility, and adherence to GCP best practices for scalable and secure data orchestration.

Development Applications

Our project will provide multiple essential development applications that establish the technical basis of the Generative AI Healthcare Agent. We will create Specialized Healthcare Language Models by refining Llama 3 and Mistral-7B using meticulously selected medical

datasets. These models will be tailored for unique healthcare situations: clinical explanation, ED stay summarizer and post ED stay follow up. The system will provide patients with answers to ED-related queries by retrieving information from the web. If a query is deemed critical, it will suggest consulting a doctor. Additionally, the system will maintain a history for both patients and doctors, ensuring continuity of care. This prototype will facilitate smooth information exchange while preserving context throughout the ED process. Our Healthcare Integration Framework will offer standardized APIs for linking with current clinical information systems, accommodating both FHIR and HL7 standards for the exchange of healthcare data. This framework will incorporate secure authentication methods, thorough data validation, and customizable integration pipelines for handling clinical narratives, structured health records, and data generated by patients.

Production Applications

The peak of our development initiatives will be a collection of applications ready for production, tailored for practical healthcare settings. The integrated healthcare assistant platform will offer a complete web-based application that allows healthcare professionals to access all three specialized agents via a single interface. This platform will include role-specific dashboards that display pertinent patient data according to clinical specialty and care environment, an alert management system for monitoring possible complications detected by the agent, and a documentation module that records AI-supported interactions for incorporation with medical records. The platform will integrate responsive design principles to enable use on desktop workstations in clinical environments and on tablet devices during patient rounds.

1.4 Technology and Solution Survey

Current technologies

The healthcare AI environment has advanced swiftly in recent years, providing numerous technologies and solutions that may meet our project needs for ED guidance. Large Language Models (LLMs) signify a notable progress in AI functionalities pertinent to healthcare uses. OpenAI's GPT models provide strong language comprehension and generation abilities but pose difficulties regarding customization, expense, and data privacy for use in healthcare settings.

These models utilize transformer architectures that are trained on extensive collections of text, such as medical literature, allowing them to handle intricate healthcare inquiries with contextual comprehension. In our healthcare agent initiative, open-source options like Llama 3 and Mistral present considerable benefits regarding customization and affordability, enabling fine-tuning on specialized medical data sets while ensuring data privacy – an essential aspect in healthcare applications (Touvron et al., 2023).

Meta's Llama 3, utilizing an open-source strategy and excelling in medical reasoning assignments, offers enhanced adaptability for fine-tuning on specific healthcare datasets while lowering deployment expenses. The Mistral model family, especially Mistral-7B, provides an effective option that demands fewer computational resources while still delivering satisfactory results on medical question-answering tasks. Various healthcare-focused LLMs have been developed, such as Google's MedPaLM, which shows impressive precision on medical licensing exam queries but has restricted access for tailored applications. Med-Flamingo and BioMedLM are additional specialized models that integrate medical expertise with multimodal functions, although they usually demand substantial computational resources for implementation.

Systems for representing medical knowledge constitute another vital technology category for healthcare applications. The Unified Medical Language System (UMLS),

created by the National Library of Medicine, combines more than 200 medical vocabularies and coding systems, offering a thorough framework to maintain semantic consistency across various healthcare areas. This knowledge framework is crucial for creating healthcare agents capable of sustaining a consistent understanding throughout ED. Likewise, SNOMED CT provides an organized clinical terminology that aids in clinical documentation and reporting. These knowledge representation systems allow our agents to consistently interpret medical concepts and uphold clinical precision during patient interactions (Bodenreider, 2004).

Specialized algorithms for detecting anomalies and analyzing trends are crucial for the monitoring and evaluation of patient data. Techniques for time-series analysis utilizing recurrent neural networks (RNNs) and long short-term memory (LSTM) networks allow the system to identify variations from anticipated recovery trends post-surgery. These methods are enhanced by statistical techniques to create personalized baselines, taking into consideration individual patient factors like age, comorbidities, and medication histories. These abilities are vital for the post-ED monitoring aspect of our healthcare agent, enabling early identification of possible complications and prompt interventions (Che et al., 2018).

Comparison of Current Solutions

To establish a successful AI healthcare agent system, it is essential to evaluate current methods, algorithms, and models to identify the optimal combination that will fulfill our project's unique needs. Every possible solution presents unique benefits and drawbacks based on the type of healthcare interactions and clinical situations at play.

In healthcare contexts, various LLM-based methods for natural language understanding offer different compromises. Proprietary models such as GPT-4 provide outstanding capabilities in understanding intricate medical inquiries and producing precise answers, yet they face challenges related to expenses, limits on customization, and data

privacy issues. In comparison, open-source models such as Llama 3 enable thorough fine-tuning on specific medical datasets and use in privacy-sensitive settings, although they might need greater adjustments to reach similar performance levels. Our method utilizes open-source models refined on meticulously selected healthcare datasets such as MedDialog and PubMedQA, allowing the system to deliver precise, contextually relevant advice throughout various care stages while ensuring cost efficiency and data confidentiality (Singhal et al., 2023). Our initiative utilizes this hybrid method, leveraging the UMLS knowledge graph as a semantic base that guarantees a uniform comprehension among specialized healthcare agents, all while enabling adaptable, context-sensitive interactions.

In the field of patient monitoring and anomaly detection, rule-based systems provide clarity and alignment with clinical practices but are limited in their ability to adjust to individual patient differences. Techniques in ML such as SVM and random forests allow for improved personalization but may have difficulties with the time-related elements of recovery monitoring. Recurrent neural network-based deep learning techniques are effective at identifying temporal trends in patient recovery; however, they demand a significant amount of training data and can act as "black boxes" in a clinical context. See Table 1 for the various approaches and models that may be suitable for the project.

Table 1

Comparison of solutions including approaches, algorithms and models

Category	Approach	Strengths	Limitations	Suitability for Project
Language Understanding	Proprietary LLMs (GPT-4)	Superior medical reasoning; robust handling of ambiguity	High cost; limited customization; data privacy concerns	Moderate – effective but with implementation challenges
Language Understanding	Open-source LLMs with fine-tuning	Cost-effective; customizable; deployable in private environments	Requires extensive adaptation; potentially lower baseline performance	High – aligns with project constraints and requirements
Multi-agent Architecture	Centralized Orchestration	Simplified management; guaranteed consistency	Single point of failure; potential bottlenecks	Low – lacks resilience needed for healthcare applications
Named Entity Recognition	Bio_ClinicalBERT	Domain-specific pre-training on PubMed/MIMIC yields high precision and recall	Large model size and memory footprint can slow inference; may require additional fine-tuning	High – ideal for accurate extraction of diseases, medications, and procedures
Semantic Retrieval & Indexing	All-MiniLM-L6-v	Ultra-lightweight & fast to compute, enabling real-time vector searches at scale with good general semantic coverage	Trained on general text; may miss subtle clinical nuances without domain adaptation	Moderate – efficient for MIMIC-IV-ED retrieval but may need further tuning
Anomaly Detection	Rule-based Systems	Clinical interpretability; alignment with medical guidelines	Limited personalization; manual rule creation and maintenance	Low – too rigid for diverse patient populations

Category	Approach	Strengths	Limitations	Suitability for Project
Multi-agent Architecture	Graph Mediation	Ensures consistency; flexible information sharing; clinical alignment	Complexity; requires comprehensive ontology	Ideal balance of consistency and flexibility
Anomaly Detection	Traditional ML (SVM, RF)	Good performance with limited data; model interpretability	Limited temporal pattern recognition	Moderate – useful for specific components with clear features
Anomaly Detection	Deep Learning (RNN, LSTM)	Superior temporal pattern recognition; adaptive to complex variations	Data hungry; limited interpretability; computational demands	High – particularly for post-ED monitoring with temporal aspects

To sum up, based on the below shown Figure 1, we have finalized three open source models (PatientSeek(DeepSeek), Meditron(Llama2), OpenBioLLm-70B (Llama 3) and BioMistral 7B (Mistral) offer customization and affordability, enabling fine-tuning on specialized medical data sets while ensuring data privacy. The heatmap shows that models like OpenBioLLM-70B and BioMistral 7B provide customizable solutions without proprietary restrictions. These models are good in biomedical specialised knowledge and demonstrate superior performance when compared to general-purpose models. Other Lightweight models (Bio_ClinicalBERT) offer several crucial advantages like deployment flexibility and reduced latency. Lightweight models show significantly lower performance than larger modes but, are important for deployment flexibility and reduced latency for resource-constrained environments. Apart from this we have also used other open sources of information to compare different models like HuggingFace and Vellum AI leaderboard.

We will evaluate models based on quantitative benchmarks including accuracy, F1 score, inference latency, and resource utilization (e.g., GPU memory and compute time) on clinical test cases. For instance, we will assess performance on tasks such as clinical explanation, ED stay summarizer and post ED stay follow-up guidance using standardized medical datasets. In comparing Llama 3 and Mistral, our technical criteria will balance higher baseline accuracy against computational efficiency; Llama 3 may offer superior precision while Mistral provides faster inference and lower resource overhead, which is critical for real-time applications.

Figure 1

Performance Scores of Different Models Across 9 Diverse Benchmark Datasets

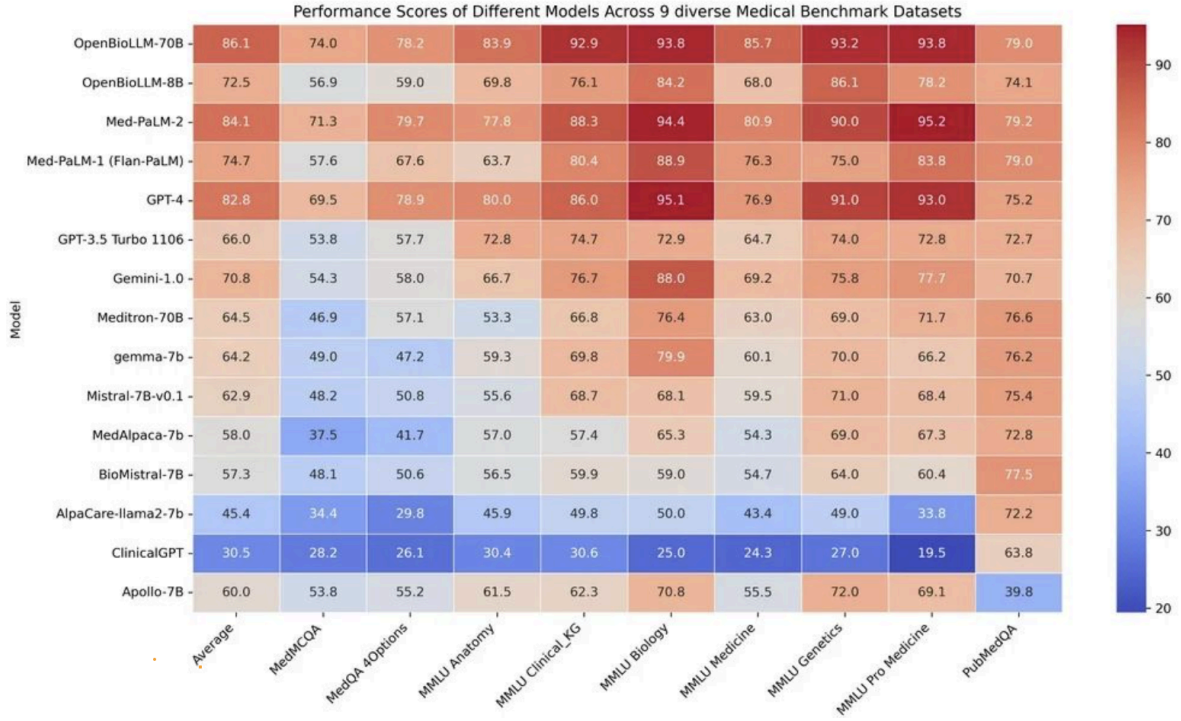


Image Source (Reddit)

Summary and Classifications of Features and Applications

The Generative AI Healthcare Agent integrates extensive assistance across two critical care stages—Emergency Department (ED) stay and post-discharge follow-up—while maintaining context and continuity during the patient journey. While in the ED, the Summarizer Agent absorbs triage notes, vital signs, lab results, provider documentation, and discharge summaries to generate brief, chronologically arranged narratives that emphasize key findings, annotate medical entities, and identify urgent matters through an interactive timeline with confidence indicators. Post-discharge, the Follow-up Agent checks readmission risk, creates customized care plans, arranges appointment and medication reminders, and provides tiered alerts for concerning symptoms, continuously enhancing its recommendations as fresh patient data comes in. These abilities depend on natural language processing for understanding medical texts and facilitating patient-friendly communication, knowledge representation through ontologies such as UMLS for semantic alignment, and time-series analysis for identifying patterns and detecting anomalies. Collectively, they drive three areas of application: clinical decision support aimed at ensuring guideline compliance and evaluating risk; patient engagement systems that promote self-management and education; and care coordination resources that enhance transitions between care environments and include caregivers—all tackling the fragmentation of current healthcare technologies by providing evidence-based, context-rich suggestions from emergency department presentation through to follow-up.

1.5 Literature Survey of Existing Research

The integration of generative artificial intelligence into healthcare systems represents one of the most promising technological shifts in modern medicine, offering unprecedented capabilities to transform patient care, clinical decision-making, research methodologies, and healthcare delivery systems. Shokrollahi et al. (2023) provide an extensive examination of

this emerging landscape, documenting the rapid proliferation of generative AI applications across diverse medical domains including diagnostics, treatment planning, medical imaging, drug discovery, and personalized medicine. Their comprehensive review establishes a taxonomic framework for understanding how these technologies interact with existing healthcare infrastructures while identifying potential implementation pathways for clinical integration. Similarly, Catal et al. (2024) trace the evolutionary trajectory of generative AI technologies in healthcare, noting the accelerated development cycle from experimental prototypes to functioning clinical systems. They particularly emphasize the role of generative adversarial networks (GANs) and variational autoencoders (VAEs) in creating synthetic medical data that preserves statistical properties while maintaining patient privacy. The transformative potential of generative AI must be balanced against significant implementation challenges as elaborated by Albaroudi et al. (2024), who provide a detailed analysis of technical, ethical, and operational barriers to clinical adoption. Their work specifically addresses the intersection between generative capabilities and patient care enhancement, proposing a multi-layered framework for evaluating AI interventions based on clinical validity, interpretability, and alignment with healthcare workflows. The researchers identify several critical challenge domains including data quality inconsistencies, algorithmic bias amplification, limited interpretability of black-box models, and concerns regarding patient privacy in generative systems. They further propose architectural modifications to improve model transparency and establish healthcare-specific evaluation protocols for assessing generative outputs in clinical contexts. Fleurence et al. (2024) extend this analysis into the health technology assessment domain, examining how traditional evaluation frameworks must evolve to accommodate generative AI systems. Their research outlines specific policy considerations for regulatory bodies, healthcare institutions, and technology developers, emphasizing the need for continuous monitoring frameworks and adaptive

governance strategies. The authors propose a structured assessment methodology incorporating technical performance metrics, clinical validation protocols, implementation feasibility studies, and economic impact analyses specifically calibrated for generative AI applications in healthcare settings. Recent innovations have demonstrated the versatility of generative AI across specialized healthcare domains. Li et al. (2024) introduce "Agent Hospital," an advanced simulation environment that creates realistic clinical scenarios populated with evolvable medical agents capable of mimicking healthcare professional behaviors and decision-making processes. This system incorporates reinforcement learning techniques to develop increasingly sophisticated agent behaviors through simulated patient interactions, potentially revolutionizing medical education and clinical training paradigms. The researchers document significant improvements in medical student diagnostic accuracy and clinical reasoning after training with these simulated environments, particularly for rare conditions rarely encountered during standard clinical rotations. Niyato and Shi (2024) explore another frontier application through their comprehensive survey of generative-AI-driven human digital twins in IoT healthcare environments. Their research catalogues implementation strategies for creating personalized patient models that continuously update based on real-time sensor data, enabling predictive analytics and personalized intervention planning. The authors identify specific technical requirements for different clinical use cases, categorizing digital twin implementations across preventive, diagnostic, therapeutic, and rehabilitative applications while providing detailed architectural blueprints for system developers. In the dental domain, Zeng et al. (2024) present a cautious yet optimistic assessment of generative AI applications, reviewing current implementations in diagnostic imaging, treatment planning, and prosthetic design. Their critical analysis emphasizes the need for rigorous validation protocols specific to dental applications, documenting both successful implementations and cautionary failures where algorithmic

outputs required significant modification before clinical application. The researchers provide detailed case studies of AI-assisted dental procedures, quantifying both time savings and diagnostic accuracy improvements while identifying remaining technical barriers to full clinical integration. Shanmugam et al. (2024) provide perhaps the most clinically-oriented examination of generative AI in medicine, offering detailed analysis of implementation strategies across primary care, specialist settings, emergency medicine, and surgical applications. Their work includes comprehensive case studies documenting workflow integration methodologies and quantitative assessments of clinical impact across different medical specialties. The authors specifically analyze how generative AI systems can augment clinical documentation, enhance diagnostic reasoning, optimize treatment selection, and improve patient communication across various medical contexts. Their findings suggest variable benefits across different application domains, with particularly promising results in radiology, pathology, and clinical documentation automation. The generation of synthetic medical data represents another critical application domain as explored by Jadon et al. (2024), who examine methodological approaches for creating high-fidelity synthetic datasets that preserve statistical properties while addressing privacy concerns. Their research catalogs various generative techniques including GANs, diffusion models, and autoregressive approaches, providing comparative performance analysis across different medical data modalities including imaging, time-series, and structured clinical data. The authors present detailed technical specifications for privacy-preserving data generation pipelines, including differential privacy implementations, federated learning approaches, and robust de-identification protocols specifically designed for healthcare applications. Manoj and Nandhini (2024) provide an exceptionally detailed investigation of large language model applications in healthcare, documenting implementation strategies across clinical documentation, medical education, research synthesis, and patient communication domains.

Their comprehensive analysis encompasses model architecture considerations, domain-specific training strategies, prompt engineering techniques, and evaluation methodologies specifically adapted for healthcare contexts. The researchers quantify performance improvements across various clinical natural language processing tasks including medical entity recognition, relation extraction, clinical summarization, and question answering, while identifying specialized approaches for handling medical terminology, clinical reasoning patterns, and temporal relationships in healthcare narratives. The development of conversational healthcare agents represents one of the most visible and rapidly evolving applications of generative AI in healthcare. Karpagam et al. (2024) introduce architectural frameworks for intelligent conversation agents specifically designed for healthcare domains, detailing dialogue management strategies, knowledge representation approaches, and natural language generation techniques optimized for medical communication. Their system incorporates medical ontologies, evidence-based guidelines, and contextual reasoning capabilities to deliver personalized health information while maintaining clinical accuracy. The authors document implementation challenges including medical terminology handling, entity resolution complexity, and the management of uncertain or incomplete information in clinical conversations. Abbasian et al. (2023) advance this domain through their development of personalized LLM-powered agent frameworks specifically designed for healthcare contexts. Their research details architectural considerations for healthcare-specific language models, including specialized training approaches, prompt engineering techniques, and retrieval augmentation strategies for accessing medical knowledge bases. The authors provide comprehensive evaluation results across various healthcare communication tasks, documenting performance improvements for symptom assessment, medication counseling, chronic disease management, and preventive health guidance. Their findings highlight the importance of controlled medical knowledge

incorporation, continuous learning capabilities, and explainable reasoning processes in healthcare conversational systems. Martinengo et al. (2023) contribute foundational conceptual frameworks through their expert interview study, synthesizing insights from healthcare providers, informaticians, and AI developers to establish classification systems and design principles for conversational agents in healthcare. Their research documents essential taxonomic categories, functional requirements, and evaluation criteria specifically adapted for healthcare conversational systems. The authors identify critical success factors including medical knowledge accuracy, communication appropriateness, workflow integration, and ethical safeguards necessary for clinical deployment. Their conceptual framework establishes standardized terminology and assessment approaches that have subsequently influenced design and evaluation practices across the field. Sim et al. (2024) provide important implementation insights through their work on multilingual conversational agents for diabetes care, addressing both technological and cultural adaptation challenges in designing systems for diverse patient populations. Their research documents specific design considerations for supporting different languages, cultural contexts, and health literacy levels while maintaining clinical accuracy and engagement. The authors present detailed case studies of multilingual deployment strategies, language-specific prompt engineering techniques, and cultural adaptation approaches for conversational agents in diabetes management. Their findings highlight both universal design principles and culture-specific adaptation requirements for effective healthcare communication systems. The development of empathic capabilities in healthcare conversational systems represents a critical frontier as explored by Adikari et al. (2022), who examine both technical and psychological aspects of creating emotionally responsive virtual agents. Their research details specific natural language processing techniques for emotion recognition, sentiment analysis, and affective response generation in medical conversations. The authors document implementation

approaches for real-time emotion detection, appropriate empathic response selection, and continuous adaptation based on patient emotional states. Their findings suggest significant improvements in patient satisfaction, disclosure comfort, and adherence intention when interacting with empathically-enhanced conversational systems compared to standard information-oriented implementations. Shi et al. (2025) explore the emerging concept of professional identity development for AI chatbots, examining how conversational agents can be designed to embody appropriate professional characteristics for healthcare contexts. Their research analyzes different approaches to personality design, communication style engineering, and ethical reasoning capabilities for healthcare conversational systems. The authors document user perception studies comparing different professional presentation approaches and their impact on trust, perceived competence, and interaction satisfaction. Their findings suggest that carefully calibrated professional identity characteristics significantly influence user acceptance and engagement with healthcare conversational systems.

Building on earlier foundational work, Magrabi et al. (2018) provide historical context through their systematic review of conversational agents in healthcare, documenting the evolution from simple rule-based systems to sophisticated generative models. Their comprehensive analysis catalogs implementation approaches, evaluation methodologies, and reported outcomes across diverse healthcare applications including mental health support, chronic disease management, medication adherence, and health education. The authors identify recurring implementation challenges and success factors that continue to inform current development practices, establishing important benchmarks for measuring progress in the field. Hemasri et al. (2024) articulate an ambitious vision for how generative AI technologies will fundamentally redefine modern medicine, projecting transformative impacts across clinical practice, medical education, research methodologies, and healthcare

delivery systems. Their research synthesizes emerging evidence from early implementations to project future developmental trajectories and potential healthcare system impacts. The authors provide detailed future scenarios illustrating how generative AI might reshape diagnostic processes, treatment planning, clinical documentation, medical education, and patient-provider relationships over the coming decade. Their analysis encompasses both optimistic transformation scenarios and potential implementation barriers that might delay or redirect development paths. Ranasinghe et al. (2024) focus on practical productivity enhancement through their work on retrieval-augmented generative AI chatbots, addressing the critical challenge of balancing efficiency with information accuracy in healthcare communications. Their research details technical approaches for combining large language model capabilities with retrieval systems accessing verified medical knowledge sources, electronic health records, and institutional protocols. The authors document performance improvements across various healthcare information tasks including clinical question answering, protocol guidance, documentation assistance, and patient education. Their findings demonstrate how retrieval augmentation significantly reduces hallucination rates while improving response specificity compared to pure generative approaches in healthcare contexts. As comprehensively documented by Sai et al. (2024), researchers increasingly recognize both the transformative potential and implementation challenges associated with generative AI in healthcare. Their extensive study catalogues emerging models, applications, case studies, and limitations across diverse healthcare domains, providing a comprehensive state-of-the-field assessment. The authors identify several critical research priorities including domain-specific architectural improvements, healthcare-specific evaluation methodologies, implementation science approaches, and governance frameworks necessary for responsible advancement. Their findings emphasize the importance of collaborative development

involving clinical experts, AI researchers, patients, and policymakers to address both technical challenges and ethical considerations in healthcare AI development.

For ED particularly, Hartman et al. (2024) performed a cohort study at NewYork-Presbyterian/Weill Cornell utilizing 1,600 EM patient records to create and assess a LLM pipeline for handoff notes from EM to inpatient. Their tailored clinical model attained significantly improved ROUGE (0.322 vs. 0.088), BERTScore (0.859 vs. 0.796), and SCALE inconsistency scores (0.691 vs. 0.456) in comparison to notes written by physicians, although board-certified emergency doctors found AI-generated summaries to be somewhat less useful (4.04 vs. 4.36) and safe (4.06 vs. 4.50) on a 5-point scale—without any critical safety issues identified. These findings highlight the potential of LLMs to lessen documentation workload when utilized within a physician-in-loop model. De Rouck et al. (2025) carried out a pilot study in a Dutch emergency room to evaluate the practicality of employing GPT-4 to create patient discharge information (PDI) brochures for three frequent ED issues (nonspecific abdominal pain, nonspecific low back pain, and pediatric fever). They established five essential performance indicators—quality, accessibility, clarity, correctness, and usability—and engaged eight seasoned emergency physicians to evaluate AI-generated brochures, which received an average score of 7–8 out of 10 on the KPIs. Readability assessments (Flesch Reading Ease, Flesch-Kincaid Grade Level, SMOG, CLI) indicated that the brochures typically necessitated a high-school to college-level understanding, emphasizing the necessity for human editing to adjust the content to the suggested sixth-grade reading level. This study showcases the possible efficiency improvements of generative AI in creating PDI materials, while highlighting the essential role of clinician review for ensuring accuracy and patient understanding. Meyer and Meyer (2025) offer a systematic review of ChatGPT's use in emergency medicine, consolidating findings from 46 studies to outline its effectiveness in documentation, communication, and decision-support

activities. They state that ChatGPT 4 significantly enhances administrative processes—like discharge summaries and handoff templates—compared to previous versions, yet emphasize considerable variability in study designs, an absence of standardized assessment metrics, and the necessity for clinician supervision to reduce errors and enable current integration.

2. Data and Project Management Plan

2.1 Data Management Plan

To ensure a structured and reproducible data workflow for clinical data preprocessing and modeling, we have developed an automated cloud-based pipeline integrated with downstream model training and retrieval-augmented generation (RAG) components. Primary Data Sources PhysioNet Zip Files: Raw patient data is obtained from PhysioNet repositories. These files are uploaded and unzipped using Jupyter Notebooks running on Google Colab, and then stored as CSVs in Google Cloud Storage (GCS). NCBI Clinical Dataset: This dataset was used to fine-tune the base model BioClinicalBERT, ensuring the model is well-adapted to biomedical domain-specific text. PubMed Abstracts: A large corpus of biomedical literature was used to build the retrieval corpus for the RAG framework, providing a rich knowledge base for context-aware question answering and inference. Cloud Storage & Automation Google Cloud Storage (GCS) acts as the primary storage for raw and intermediate data. Apache Airflow (Dockerized) automates data preprocessing tasks as follows: Task 1: Data Cleaning Removes nulls, duplicates, and applies normalization on clinical features. Cleaned CSVs are saved back into GCS. Task 2: Data Transformation Converts cleaned CSVs to JSON format. Data Preparation & Splitting Metadata is collected and analyzed using Jupyter notebooks. Final datasets are balanced and split into: Training Set – 70% Validation Set – 15% Test Set – 15% Model Integration BioClinicalBERT, fine-tuned on the NCBI dataset, serves as the encoder for clinical text. A RAG (Retrieval-Augmented

Generation) framework was built using the PubMed abstract corpus for the retriever and BioClinicalBERT as the generator backbone. The transformed data is used to both train and evaluate the RAG model, ensuring real-world relevance and interpretability. Data Security & Ethics Access to cloud storage Atlas is restricted using IAM roles and API tokens. No Personally Identifiable Information (PII) is included, and all data handling adheres to clinical data protection standards. Versioning and Reproducibility The pipeline uses Docker for environment consistency and version control. Dataset versions and preprocessing configurations are logged to ensure reproducibility of results.

Data Collection Approaches

To begin the project, a robust and ethical data collection strategy was established by selecting suitable and publicly accessible clinical datasets from **trustworthy medical** databases. These datasets were chosen for their relevance to emergency medicine, biomedical NLP, and real-world applicability in clinical decision-making.

The core clinical data source for this project is the MIMIC-IV-ED dataset from PhysioNet, which offers a flexible and modular structure. It emphasizes data lineage, making it possible to utilize various clinical data sources either independently or in combination, thereby supporting diverse healthcare research scenarios. MIMIC-IV-ED is an evolution of the MIMIC-III dataset and facilitates machine learning applications by offering well-structured and labeled data suitable for both statistical analysis and model training.

To further enhance the language understanding and biomedical context of our model, two additional text-based datasets were integrated: the NCBI Disease Corpus and PubMed abstracts. These were used to fine-tune a domain-specific language model (BioClinicalBERT) and to support retrieval-augmented generation (RAG) for clinical NLP tasks such as diagnosis prediction and medical Q&A.

Data Sources

MIMIC-IV-ED Dataset (v2.2):

This dataset is a large-scale, publicly available clinical resource hosted on PhysioNet. It comprises approximately 425,000 emergency department (ED) visit records collected at the Beth Israel Deaconess Medical Center (BIDMC) from 2011 to 2019. Data was collected during routine clinical workflows including triage assessments, vital sign monitoring, physician notes, and medication records. The dataset is part of the broader MIMIC-IV clinical data initiative supported by the MIT Laboratory for Computational Physiology, and is widely used for emergency medicine research, clinical decision support, and healthcare-focused machine learning.

Figure 2

MIMIC IV-ED dataset record count by each CSV file after cleaning and preprocessing.

Table Name	Description	Record Count
edstays	Emergency department visits (unique stays IDs)	~425,087
triage	Triage assessment, vitalsigns, chief complaints	~425,087
vitalsign	Vital Signs recorded throughout the ED stay	~1,564,610
diagnosis	Discharge diagnosis with ICD-10 codes	~899,050
medrecon	Medication reconciliation at admission	~2,987,342
pyxis	Administered medications during ED stays	~1,586,053

NCBI Disease Corpus:

The NCBI dataset is a biomedical named entity recognition (NER) dataset consisting of annotated PubMed abstracts with disease mentions labeled in the BIO tagging format. Developed for the BioNLP challenges, it is widely used to train and evaluate NER models in

the clinical domain. In this project, the dataset was utilized to fine-tune the BioClinicalBERT model, allowing it to better detect disease-related entities and improve its performance on downstream tasks involving clinical text understanding.

PubMed Abstracts:

A large-scale text corpus of biomedical literature from PubMed was used to build the retrieval base for the RAG framework. These abstracts contain scientifically validated medical knowledge which, when retrieved and fed into the generation model, enhances factual consistency, explainability, and clinical relevance of responses generated by the system. The combination of PubMed's extensive knowledge base and BioClinicalBERT's domain specialization allows for more robust NLP applications in medical research.

Data Management Methods

Our data management strategy employs a robust and scalable pipeline designed to process both structured and unstructured healthcare data. The workflow begins by fetching CSV and JSON files from reliable source repositories such as MIMIC-IV-ED and PubMed. Once ingested, the data undergoes extensive cleaning and preprocessing using Python and Pandas.

This cleaning phase addresses several key challenges in clinical data, including the handling of missing values, standardization of medical terminology and measurement units, removal of duplicate entries, and normalization of date/time formats. These steps ensure data consistency and reliability for downstream processing.

Following preprocessing, we apply transformer-based Named Entity Recognition (NER) models—specifically BioClinicalBERT fine-tuned on the NCBI Disease Corpus—to

extract critical clinical entities. These include diseases, medications, procedures, laboratory results, and temporal indicators relevant to patient care.

Usage Mechanisms

The primary data gathering creates the clinical knowledge foundation for our healthcare agent. We collect genuine healthcare medical data to train our model. This enhances the agent's capability to comprehend intricate medical terminology, patient issues, and clinical scenarios. We methodically assess data to determine our healthcare agent's efficiency. This entails evaluating the precision of medical responses, the suitability of empathetic tones, and the effectiveness in resolving health inquiries to pinpoint areas for enhancement. We focus closely on safety and dependability in the delivery of medical information. The knowledge derived from these evaluations directly guides our following fine-tuning cycles, establishing a continuous improvement process that boosts patient care and provider assistance.

Summary

The aim of the Data Management Plan is to execute the procedures of data collection, storage, management, and utilization in an efficient and secure way. To create an ever-evolving agent, the project emphasizes authentic information gathered from patient data. Safe storage, management of organizational data, and proper utilization of that data for enhancement and evaluation are vital elements for the agent’s success. Certainly, these frameworks will allow the agent to adjust, stay secure, and operate within legal boundaries of data protection, all to guarantee that users receive assistance through a functional conversational interface.

2.2 Project Development Methodology

The GenAI healthcare agent project employs an extensive development approach that combines data analysis with intelligent system development phases. This combined method enables us to develop a healthcare agent that consistently enhances its performance while preserving clinical precision and ensuring patient safety.

Data Analytics with Intelligent System Development Life Cycle (ISDLC)

The development of our intelligent clinical data processing system follows a structured approach inspired by the Intelligent System Development Life Cycle (ISDLC). This methodology ensures a comprehensive end-to-end process that spans from problem definition and data preparation to model deployment and evaluation, with a strong focus on automation, clinical relevance, and interpretability.

The project begins with the problem identification and planning phase, where the goal was to develop an intelligent system capable of extracting, processing, and interpreting complex clinical data from real-world emergency department (ED) visits. The key objectives included preprocessing heterogeneous clinical records, training a domain-specific language model, and implementing a retrieval-augmented generation (RAG) architecture to support explainable and accurate medical NLP tasks.

In the data acquisition and understanding phase, we collected data from multiple trusted sources. The primary structured dataset was the MIMIC-IV-ED dataset, which provides comprehensive patient records from ED visits at the Beth Israel Deaconess Medical Center between 2011 and 2019. For unstructured textual data, we used the NCBI Disease Corpus, a benchmark dataset for biomedical Named Entity Recognition (NER), and PubMed

abstracts to serve as the retrieval base for the RAG component. This multi-source approach ensured a rich and diverse representation of clinical language and knowledge.

The data preparation and preprocessing phase involved intensive cleaning of raw CSV and JSON files using Python and Pandas. We addressed missing values, normalized units and timestamps, and standardized terminology to reduce variability across clinical tables. The cleaned data was then transformed into patient-centric documents to enable a longitudinal view of patient visits. These were stored following transformation and cleaning automated by Apache Airflow running in a Dockerized environment. This ensured repeatability, version control, and cloud compatibility.

In the modeling phase, we fine-tuned the BioClinicalBERT model on the NCBI dataset to recognize disease-related terms and entities in free-text clinical notes. For enhanced decision support, we implemented a Retrieval-Augmented Generation (RAG) framework. The retriever component leveraged embeddings from PubMed abstracts to identify contextually relevant documents, while the generator component—powered by the fine-tuned BioClinicalBERT—produced domain-aware, factually grounded answers. The integration of RAG significantly improved the factual consistency and interpretability of generated responses, critical in clinical applications.

The evaluation phase focused on both quantitative and qualitative assessments. We evaluated the performance of the NER model using precision, recall, and F1-score on standard test sets. Additionally, the output of the RAG system was manually reviewed to assess the relevance, fluency, and clinical accuracy of the responses. Throughout the process, we ensured the use of de-identified data and compliance with HIPAA standards to protect patient privacy and uphold ethical standards.

Finally, the deployment and maintenance phase involved the containerization of our pipeline using Docker to support portability and reproducibility. The system is designed to be modular and scalable, making it easy to extend with additional data sources or fine-tune further on specialized tasks in the future. Documentation was maintained using Jupyter notebooks and version-controlled through GitHub, enabling seamless collaboration and traceability.

By following the ISDLC, our project achieves a high level of integration between clinical data analytics, machine learning, and intelligent system design, making it a valuable contribution to the development of explainable AI systems in healthcare.

Planned Project Development Processes and Activities

At every phase of development, we integrate both qualitative healthcare knowledge and quantitative data analysis. This guarantees that our AI agent progresses under strict clinical supervision while utilizing the pattern-recognition strengths of machine learning systems to uncover potential enhancements that may not be easily noticeable to human viewers.

- **Analysis of Requirements and Mapping of Healthcare Context:** We start by thoroughly grasping the clinical settings in which our agent will function. This includes gathering of medical data, QnA etc. This will constitute our preliminary requirements specification.
- **Data Gathering and Preparation:** Leveraging our contextual insights, we create thorough data collection procedures. These consist of privacy-protecting methods for collecting representative healthcare dialogues, medical knowledge databases, and clinical decision-making patterns. Every dataset goes through thorough preprocessing

to guarantee it retains clinical precision while being properly organized for model training.

- **Iterative Model Development:** We adhere to brief development cycles for our model, each producing prototypes that can be tested. This establishes an ongoing feedback cycle in which clinical knowledge directly influences the priorities for model development.
- **Deployment and Learning Infrastructure**: We establish tailored infrastructure that allows our model to consistently learn from fresh interactions while preserving version control and audit logs. This encompasses systems for same-day learning applications that can swiftly integrate essential enhancements while avoiding deterioration of current functionalities.
- **Assessment and Quality Control:** During the development process, we uphold thorough evaluation frameworks that analyze both technical indicators (response precision, processing velocity) and healthcare-related criteria (clinical relevance, compassionate communication, safety measures). These assessments inform prioritization choices for future development cycles.

2.3 Project Organization Plan

Project Planning Phase

The Project Planning stage lays the groundwork for your whole project. It includes project understanding, guidance, and tools required prior to starting development.

- **Literature Review:** This preliminary task involves investigating current healthcare AI solutions, existing methods for medical agents, and cutting-edge language models used in healthcare contexts. We will analyze scholarly articles, case analyses, and commercial offerings to grasp what is effective, what is ineffective, and where

deficiencies lie. This study will guide design choices and assist in steering clear of recreating answers to challenges that others have previously addressed. The project will concentrate on elements such as medical precision, dialogue skills, privacy issues, and healthcare-related adjustments of current AI models.

- **Requirements Planning:** The precise requirements that the healthcare agent needs to fulfill will be determined. This entails outlining the range of medical subjects it needs to address, identifying the kinds of interactions it must facilitate (diagnostic support, drug information, patient education, etc.), and setting the limits of what the system will not try to accomplish. Additionally define technical specifications such as anticipated response times, privacy measures, and integration features. This task produces a distinct collection of functional and non-functional requirements that will steer the development choices.
- **Timeline Management:** Once project needs are outlined, an organized timetable featuring important milestones and due dates will be developed. The start and end dates for each development phase, set up review and adjustment checkpoints, and synchronize project schedule with academic deadlines are outlined. This strategy aids in maintaining consistent advancement and gives early alerts if any aspects of the project lag behind timeline, enabling you to modify resources or project scope as necessary.
- **Resource Distribution:** This task requires strategizing the effective allocation of available resources. This mainly involves distributing time among various development tasks, as well as determining and obtaining the computational resources required for training and testing the model. It involves securing access to particular datasets, specialized equipment such as GPUs for training, or medical knowledge for validation. Appropriate distribution of resources avoids bottlenecks and guarantees essential project elements receive sufficient focus.

- **Documentation**: Documentation methods to capture project choices, advancements, and discoveries will be implemented. This involves developing templates for routine progress updates, setting up a method for arranging research notes, and outlining the organization of the final project documentation. Establishing good documentation practices early on simplifies the final project assembly and helps retain important insights throughout the process.

Development Phase

The Development Phase focuses on the technical realization of our intelligent healthcare agent, transforming the conceptual design into a functioning system capable of interpreting and reasoning over clinical data. This stage involved the careful selection and fine-tuning of models, the integration of domain-specific medical knowledge, and the construction of a scalable architecture to support advanced NLP capabilities in the medical field.

Model selection began with a comparative evaluation of several biomedical language models. Although large-scale models such as OpenBioLLM-70B, Meditron, and BioMistral 7B offered impressive performance benchmarks, they were found to be computationally expensive and less suitable for rapid development and deployment in resource-constrained environments. We ultimately selected BioClinicalBERT, a transformer-based model pretrained on clinical notes from MIMIC-III and biomedical literature from PubMed. Its moderate size, domain-specific vocabulary, and open accessibility made it a practical and effective choice for downstream fine-tuning.

For training, we utilized a combination of the MIMIC-IV dataset and the NCBI Disease Corpus. MIMIC-IV provided structured real-world clinical data such as diagnosis codes, patient demographics, procedures, and medication records, which are essential for

learning the language and structure of clinical documentation. The NCBI Disease Corpus contributed high-quality, annotated disease mentions from biomedical texts, offering valuable training data for Named Entity Recognition (NER) tasks. By combining these two datasets, we fine-tuned BioClinicalBERT to accurately extract medical entities and understand the context of patient interactions. Terminologies across datasets were standardized using medical ontologies like UMLS and SNOMED-CT, ensuring uniformity in clinical vocabulary and improving model generalization.

The overall system architecture comprises two main components. The first is the fine-tuning pipeline, where BioClinicalBERT is trained using the combined MIMIC-IV and NCBI datasets to enhance its ability in NER and clinical language modeling. The second is a Retrieval-Augmented Generation (RAG) pipeline, which integrates PubMed abstracts as an external knowledge base. In this setup, when a medical question is posed, the system retrieves the most relevant PubMed entries using a dense retriever and passes them along with the query to the BioClinicalBERT-based generator. This structure allows the system to deliver evidence-backed, context-aware responses that are both accurate and explainable.

The fine-tuning process was carried out using Hugging Face's Trainer API. The MIMIC-IV and NCBI datasets were tokenized and formatted appropriately for training, and model performance was evaluated using standard NER metrics such as precision, recall, and F1-score. We employed early stopping and checkpointing to prevent overfitting and ensure model stability. Meanwhile, the RAG module was separately developed by indexing the PubMed abstracts and integrating them into the inference stage using retrieval algorithms like FAISS. The top-k retrieved documents were used to enhance the model's ability to answer clinical questions with relevant scientific evidence.

Finally, to improve the system's reliability and usability, we experimented with prompt engineering and instruction tuning to guide the model's behavior during ambiguous or complex queries. This ensured that the responses generated remained clinically appropriate, aligned with medical best practices, and understandable to healthcare professionals.

In summary, the Development Phase resulted in a specialized version of BioClinicalBERT fine-tuned on structured clinical and annotated biomedical data, and augmented with a knowledge-rich retrieval mechanism using PubMed. This dual capability enables our healthcare agent to perform both precise information extraction and insightful response generation, forming the backbone of an intelligent clinical decision support system.

Testing Phase

The Testing Phase focused on evaluating the performance and reliability of the fine-tuned BioClinicalBERT model and the integrated Retrieval-Augmented Generation (RAG) system. We created a merged test dataset by combining samples from MIMIC-IV and the NCBI Disease Corpus, allowing us to test the model's ability to generalize across both structured and unstructured clinical data.

For NER evaluation, the fine-tuned BioClinicalBERT model was tested on its ability to identify and classify medical entities such as diseases, symptoms, procedures, and medications. The model's outputs were compared against ground truth annotations using standard metrics like precision, recall, and F1-score. These metrics helped us determine not only the accuracy of entity recognition but also the completeness and specificity of the extracted information. The results demonstrated that the model was able to accurately detect clinical entities with high confidence, particularly when entity spans were well-defined and contextually rich.

In the RAG testing pipeline, we simulated real clinical queries and used PubMed abstracts as the retrieval corpus. Queries were passed to the retriever, which returned the top-k most relevant abstracts based on semantic similarity. These retrieved documents, along with the query, were fed into the BioClinicalBERT generator to produce context-aware, evidence-backed answers. The quality of the generated responses was assessed using both automatic evaluation metrics (BLEU, ROUGE) and manual review. Manual evaluation involved domain experts rating the responses based on factual accuracy, relevance to the query, clinical soundness, and fluency. This dual evaluation helped ensure the model's outputs were not only linguistically correct but also clinically meaningful.

Finally, to ensure that the system is production-ready, we tested the full pipeline in a Dockerized environment using synthetic test data at scale. This allowed us to evaluate end-to-end latency, throughput, and error handling, ensuring that the deployed solution remains robust under different operational loads.

2.4 Project Resource Requirements and Plan

2.4.1 Hardware Requirements

The hardware needs of the project leverage current infrastructure to efficiently execute model development and deployment.

2.4.1 Hardware Requirements

Local GPU Workstation (University Lab)

- **Hardware**: NVIDIA GPU with at least 16 GB VRAM
- **Usage**: For training/fine-tuning BioClinicalBERT and running Jupyter Notebooks
- **Expense**: No additional cost

- **Justification**: Leverages existing lab infrastructure for deep learning workloads

Cloud GPU Access (Google Colab Pro)

- **Service**: Google Colab Pro
- **Hardware Access**: Tesla T4 or P100 GPUs
- **Budget**: $10/month
- **Justification**: Provides on-demand GPU compute power when local resources are unavailable or overloaded

Cloud Storage

- **Platforms**: Google Cloud Platform (GCP), Google Drive
- **Budget**: $300/year (free credits)
- **Usage**: Storing models, logs, tokenized datasets, and interaction data
- **Justification**: Secure, scalable cloud storage integrated into Google ecosystem

2.4.2 Software Requirements

Pretrained Models

- **Models**: BioClinicalBERT (Hugging Face), Llama 3, BioMistral
- **Cost**: Free (open source)
- **Justification**: Use of domain-specific pretrained models reduces computational costs and improves performance

Development & Version Control

- **Tools**: Jupyter Notebook, Visual Studio Code, GitHub
- **Cost**: Free

- **Justification**: Widely used tools for coding, collaboration, and version tracking

Documentation & Collaboration

- **Tools**:

  Overleaf (LaTeX for IEEE-format report writing)

  Google Docs, Sheets, Slides & Microsoft Office (Word, Excel, PowerPoint)

Zoom (for team meetings and recordings)

- **Cost**: Free
- **Justification**: Essential for report writing, real-time collaboration, and project communication

Project Management & Diagramming

- **Tools**:

  OnlineGantt (task tracking and Gantt charts)

  Diagram.net, LucidChart, Trello (system architecture and task visualization)
- **Cost**: Free
- **Justification**: Helps track milestones, visualize architecture, and organize team tasks

Writing & Editing Tools

- **Tool**: Grammarly
- **Cost**: Free
- **Justification**: Ensures high-quality, grammatically correct documentation

Machine Learning & NLP Frameworks

- **Tools**: PyTorch, TensorFlow, Hugging Face Transformers, LangChain
- **Cost**: Free
- **Justification**: Industry-standard frameworks for model training, evaluation, and integration with RAG systems.

2.4.3 Licenses

Clinical Dataset Access

- **Dataset**: MIMIC-IV (from PhysioNet)
- **Cost**: Free (with credentialed access)
- **Justification**: Provides structured, real-world clinical data under a research license

Biomedical NLP Tools

- **Tools**: Hugging Face Transformers, LangChain
- **Cost**: Free (open source)
- **Justification**: Used for NER tasks, model deployment, and RAG-based retrieval/generation

Cloud Platform for Deployment

- **Platform**: Google Cloud Platform (GCP)
- **Budget**: $300/year (covered by academic credits)
- **Justification**: Used for storing and deploying models, testing APIs, and enabling public access to the system

2.4.3 Licenses

- **Tool:** MIMIC IV

- **Cost: Free**
- **Justification:** MIMIC IV dataset is available on Physionet with requires credentials of a licensed professional or professor.

Natural Language Processing Tools

- **Tools**: LangChain, Hugging Face Transformers.
- **Cost**: Free (open source)
- **Justification**: LangChain and Hugging Face Transformers are used to handle text processing, retrieval, and integration. These tools are used for NLP.

Cloud Platform for Deployment

- **Platform**: Google Cloud Platform (GCP).
- **Budget**: $300 per year (already provided)
- **Justification**: GCP will be used for deployment to make it available to user. We will use it for testing and deployment of the project.

2.5 Project Schedule

Gantt Chart

The gantt chart (see Figure 3), done using an application called *onlinegantt,* below offers a clear visual representation of the project's tasks/activities, including owners, timelines, dependencies, and status. This chart acts as a medium for conveying the project's progress, schedules, and responsibilities. We have also added a gantt chart to show each team member’s contribution (see Figure 4).

Figure 3

Gantt Chart for Project Tasks and Activities

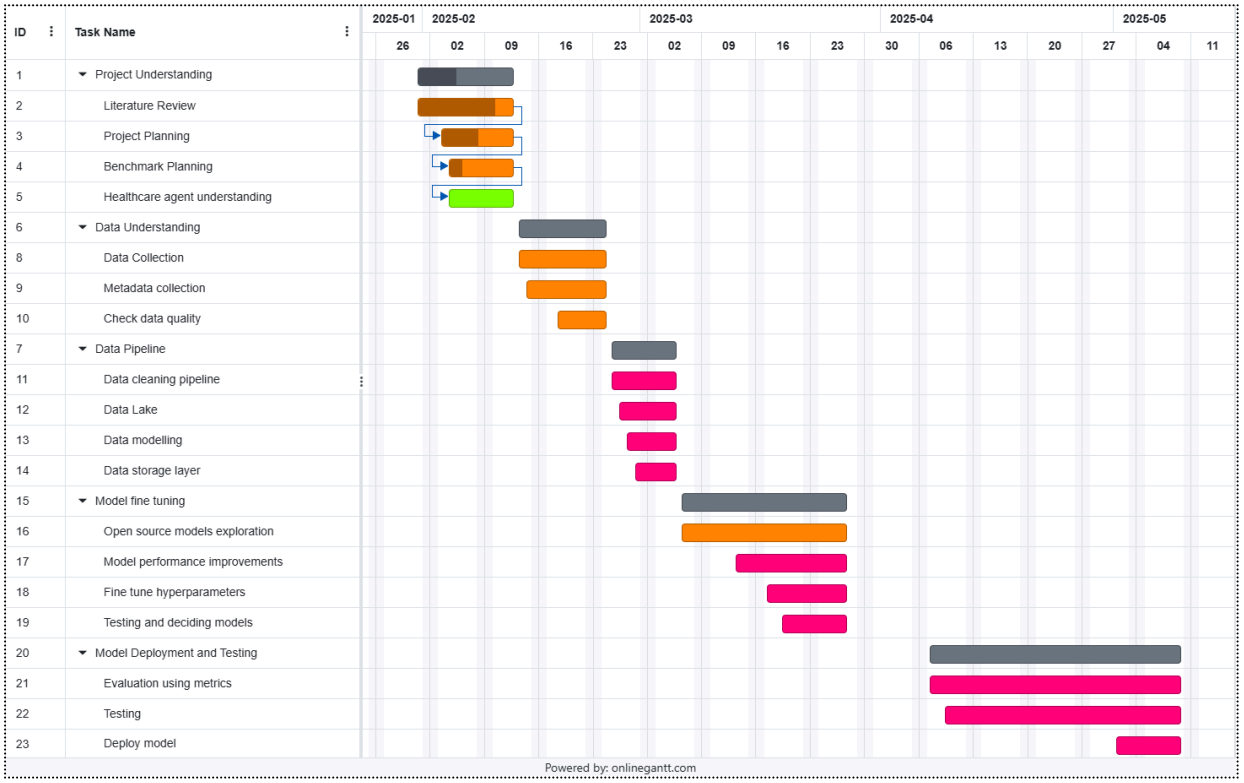
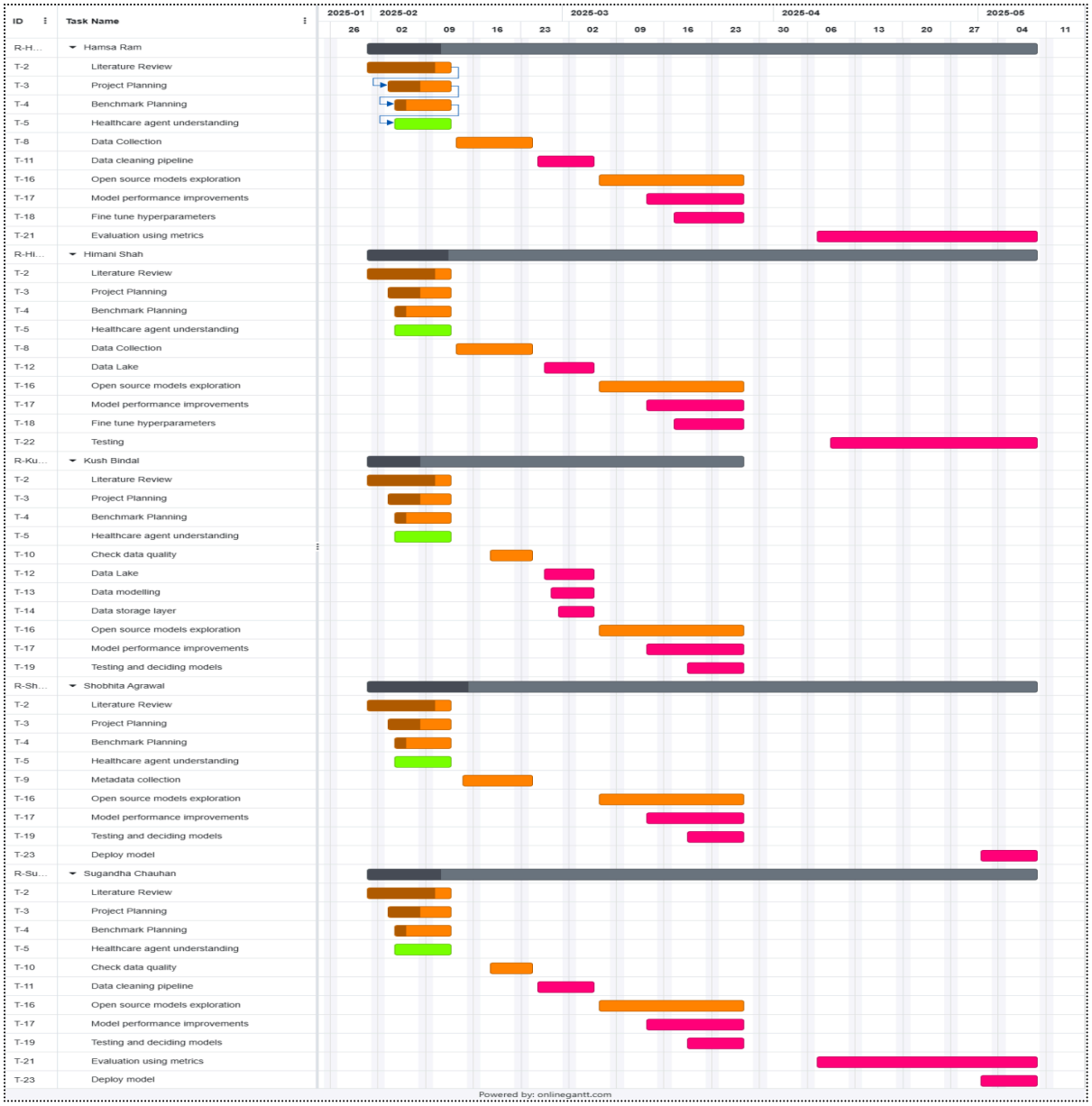


Figure 4

Gantt Chart for Each Team Member's Contributions



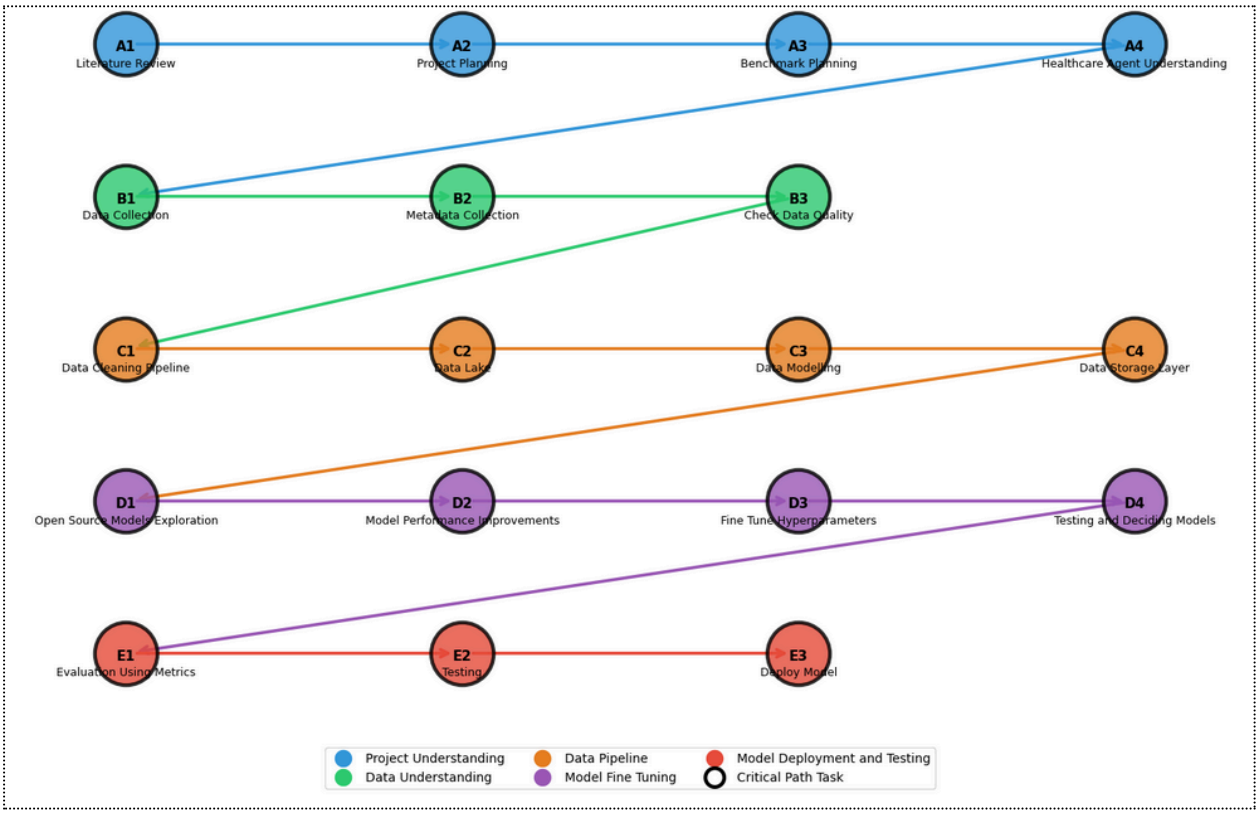
Green: completed; Orange: in progress; Pink: Not Started

Pert Chart

The PERT chart (see Figure 5) was created utilizing the tool *mermaid.live*. The PERT chart is employed to organize and plan activities by dividing them into separate tasks and illustrating the schedule. A PERT chart helps determine the critical path of the project.

Figure 5

PERT Chart to organize and plan project activities



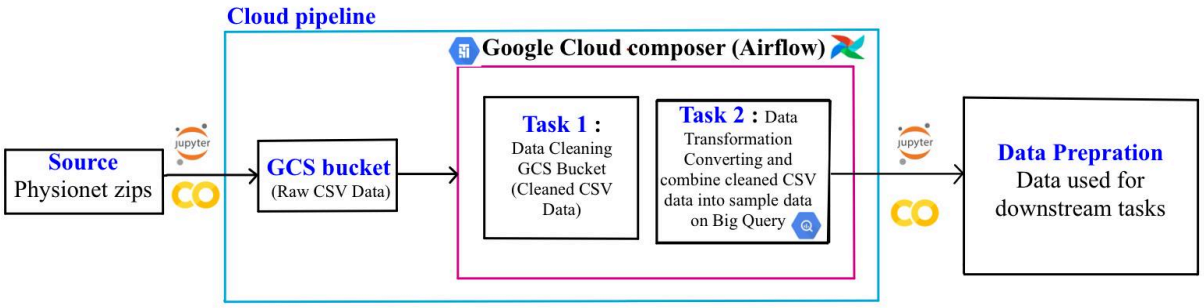
3. Data Engineering

3.1 Data Process

The data process begins with acquiring raw data from Physionet zip files and proceeds through a structured cloud-based and Airflow-orchestrated ETL (Extract, Transform, Load) pipeline. Figure 6 depicts the approach we pursued.

Figure 6

Automated Cloud Pipeline for Preprocessing and Preparing Clinical Data



To manage and automate data handling, a data pipeline was developed using Google Cloud Composer in an environment named *healthcare_env*, which supports Apache Airflow 3. A DAG titled *healthcare_data_pipeline* was created to orchestrate the entire process of data ingestion and preparation. The initial raw healthcare dataset was stored in a compressed format on Google Drive. A custom Python script was used to decompress the dataset and upload the extracted raw files into a specified folder within a Google Cloud Storage (GCS) bucket under the *healthcare_data_pipeline* directory. Once the data was uploaded, the DAG was triggered to initiate the pipeline workflow, automating the extraction of data from the GCS bucket for further processing. The final output of this stage, referred to as *sample_data*, is a cleaned subset of the original dataset. This curated dataset serves as the foundation for downstream machine learning workflows and is used for fine-tuning language models and developing a Retrieval-Augmented Generation (RAG) system tailored for healthcare-specific tasks.

3.2 Data Collection

To start, the data collection process involves selecting suitable and pertinent datasets from trustworthy clinical databases. The main data source for this project is the MIMIC-IV-ED dataset from Physionet. MIMIC-IV-ED employs a flexible strategy for organizing data, emphasizing data lineage and enabling the separate and collective utilization of various data sources. It aims to build on the achievements of MIMIC-III and facilitate a wide range of applications in healthcare.

Data Sources

The dataset is derived from real-world emergency department (ED) encounters at Beth Israel Deaconess Medical Center (BIDMC), covering the period from 2011 to 2019. It was collected as part of standard clinical operations, including triage evaluations, vital signs recording, physician notes, and medication administration. This data is publicly accessible via PhysioNet and forms part of the MIMIC-IV clinical data initiative, maintained by the MIT Laboratory for Computational Physiology.

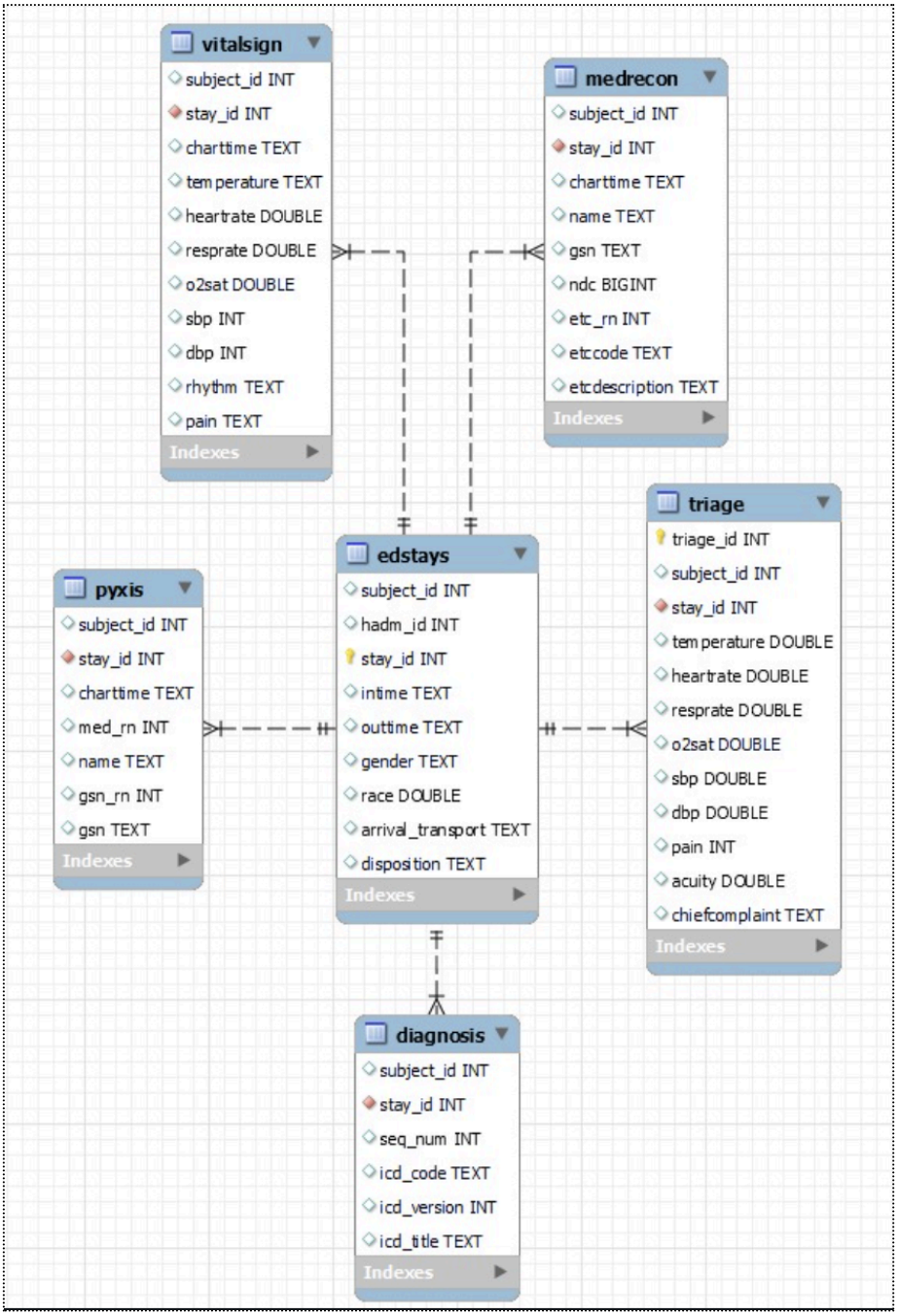
Specifically, the MIMIC-IV-ED (v2.2) dataset comprises approximately 425,000 detailed records of ED visits, offering a rich resource for research in areas such as emergency medicine, clinical decision-making, and machine learning in healthcare. It provides a comprehensive view of ED workflows, encompassing patient demographics, clinical assessments, diagnostic details, and therapeutic interventions, making it highly valuable for developing and evaluating data-driven healthcare applications.

Data Parameters

The raw datasets obtained from MIMIC-IV-ED comprise a rich variety of clinical information, including key parameters as shown in Figure 7.

Figure 7

ER Diagram Representing Tables and Relationships in Emergency Department Dataset



Structured Data:

- Patient demographics (age, gender, ethnicity)
- Emergency department (ED) encounters (arrival time, triage information, disposition)
- Diagnosis codes (ICD-9 and ICD-10)

- Procedure codes (CPT, HCPCS)
- Laboratory test results (chemistry, hematology, microbiology)
- Vital signs (heart rate, blood pressure, temperature, respiratory rate, and oxygen saturation) were recorded at various intervals.
- Medication administration records during the ED stay.
- Data related to medical devices and interventions used in the ED.

Semi-Structured Data

- Triage notes detailing the initial assessment and chief complaints.
- Provider notes documenting the patient's condition, treatment plan, and progress.
- Discharge summaries outlining the ED course and recommendations

Data Quantity

Figures 8-14 show the sample of the various tables data in MIMIC-IV-ED.

Figure 8

MIMIC IV-ED dataset record count by each CSV file after cleaning and preprocessing.

Table Name	Description	Record Count
edstays	Emergency department visits (unique stays IDs)	~425,087
triage	Triage assessment, vitalsigns, chief complaints	~425,087
vitalsign	Vital Signs recorded throughout the ED stay	~1,564,610
diagnosis	Discharge diagnosis with ICD-10 codes	~899,050
medrecon	Medication reconciliation at admission	~2,987,342
pyxis	Administered medications during ED stays	~1,586,053

Figure 9

Raw Data Sample Image from diagnosis.csv table

	subject_id	stay_id	seq_num	icd_code	icd_version	icd_title
0	10000032	32952584	1	4589	9	HYPOTENSION NOS
1	10000032	32952584	2	07070	9	UNSPECIFIED VIRAL HEPATITIS C WITHOUT HEPATIC ...
2	10000032	32952584	3	V08	9	ASYMPTOMATIC HIV INFECTION
3	10000032	33258284	1	5728	9	OTH SEQUELA, CHR LIV DIS
4	10000032	33258284	2	78959	9	OTHER ASCITES
5	10000032	33258284	3	07070	9	UNSPECIFIED VIRAL HEPATITIS C WITHOUT HEPATIC ...
6	10000032	33258284	4	V08	9	ASYMPTOMATIC HIV INFECTION
7	10000032	35968195	1	5715	9	CIRRHOSIS OF LIVER NOS
8	10000032	35968195	2	78900	9	ABDOMINAL PAIN UNSPEC SITE
9	10000032	35968195	3	V08	9	ASYMPTOMATIC HIV INFECTION

Figure 10

Raw Data Sample Image from edstays.csv table

	subject_id	hadm_id	stay_id	intime	outtime	gender	race	arrival_transport	disposition
0	10000032	22595853.0	33258284	2180-05-06 19:17:00	2180-05-06 23:30:00	F	WHITE	AMBULANCE	ADMITTED
1	10000032	22841357.0	38112554	2180-06-26 15:54:00	2180-06-26 21:31:00	F	WHITE	AMBULANCE	ADMITTED
2	10000032	25742920.0	35968195	2180-08-05 20:58:00	2180-08-06 01:44:00	F	WHITE	AMBULANCE	ADMITTED
3	10000032	29079034.0	32952584	2180-07-22 16:24:00	2180-07-23 05:54:00	F	WHITE	AMBULANCE	HOME
4	10000032	29079034.0	39399961	2180-07-23 05:54:00	2180-07-23 14:00:00	F	WHITE	AMBULANCE	ADMITTED
5	10000084	23052089.0	35203156	2160-11-20 20:36:00	2160-11-21 03:20:00	M	WHITE	WALK IN	ADMITTED
6	10000084	29888819.0	36954971	2160-12-27 18:32:00	2160-12-28 16:07:00	M	WHITE	AMBULANCE	HOME
7	10000108	27250926.0	36533795	2163-09-27 16:18:00	2163-09-28 09:04:00	M	WHITE	WALK IN	HOME
8	10000108	NaN	32522732	2163-09-16 16:34:00	2163-09-16 18:13:00	M	WHITE	WALK IN	HOME
9	10000108	NaN	39513268	2163-09-24 16:14:00	2163-09-24 21:02:00	M	WHITE	WALK IN	HOME
10	10000115	NaN	30295111	2154-12-17 16:37:00	2154-12-17 16:59:00	M	WHITE	WALK IN	HOME
11	10000115	NaN	38081480	2154-12-10 02:04:00	2154-12-10 05:59:00	M	WHITE	WALK IN	HOME
12	10000117	22927623.0	32642808	2181-11-14 21:51:00	2181-11-15 02:06:42	F	WHITE	WALK IN	ADMITTED

Figure 11

Raw Data Sample Image from medrecon.csv table

	subject_id	stay_id	charttime	name	gsn	ndc	etc_rn	etccode	etcdescription
0	10000032	32952584	2180-07-22 17:26:00	albuterol sulfate	28090	21695042308	1	5970.0	Asthma/COPD Therapy - Beta 2-Adrenergic Agents...
1	10000032	32952584	2180-07-22 17:26:00	calcium carbonate	1340	10135021101	1	733.0	Minerals and Electrolytes - Calcium Replacement
2	10000032	32952584	2180-07-22 17:26:00	cholecalciferol (vitamin D3)	65241	37205024678	1	670.0	Vitamins - D Derivatives
3	10000032	32952584	2180-07-22 17:26:00	emtricitabine-tenofovir [Truvada]	57883	35356007003	1	5849.0	Antiretroviral - Nucleoside and Nucleotide Ana...
4	10000032	32952584	2180-07-22 17:26:00	fluticasone [Flovent HFA]	21251	49999061401	1	371.0	Asthma Therapy - Inhaled Corticosteroids (Gluc...
5	10000032	32952584	2180-07-22 17:26:00	furosemide	8209	10544058430	1	250.0	Diuretic - Loop
6	10000032	32952584	2180-07-22 17:26:00	lactulose	3143	11845042013	1	478.0	Colonic Acidifier (Ammonia Inhibitor)
7	10000032	32952584	2180-07-22 17:26:00	nicotine	16426	10939070733	1	5604.0	Smoking Deterrents - Nicotine-Type
8	10000032	32952584	2180-07-22 17:26:00	peg 3350-electrolytes	62533	10572010001	1	2889.0	Laxative - Saline/Osmotic Mixtures
9	10000032	32952584	2180-07-22 17:26:00	raltegravir [Isentress]	63231	35356011006	1	6072.0	Antiretroviral - HIV-1 Integrase Strand Transf...

Figure 12

Raw Data Sample Image from triage.csv table

	subject_id	stay_id	temperature	heartrate	resprate	o2sat	sbp	dbp	pain	acuity	chiefcomplaint
0	10000032	32952584	97.80	87.0	14.0	97.0	71.0	43.0	7	2.0	Hypotension
1	10000032	33258284	98.40	70.0	16.0	97.0	106.0	63.0	0	3.0	Abd pain, Abdominal distention
2	10000032	35968195	99.40	105.0	18.0	96.0	106.0	57.0	10	3.0	n/v/d, Abd pain
3	10000032	38112554	98.90	88.0	18.0	97.0	116.0	88.0	10	3.0	Abdominal distention
4	10000032	39399961	98.70	77.0	16.0	98.0	96.0	50.0	13	2.0	Abdominal distention, Abd pain, LETHAGIC
5	10000084	35203156	97.50	78.0	16.0	100.0	114.0	71.0	0	2.0	Confusion, Hallucinations
6	10000084	36954971	98.70	80.0	16.0	95.0	111.0	72.0	0	2.0	Altered mental status, B Pedal edema
7	10000108	32522732	98.21	83.0	20.0	100.0	112.0	81.0	5	3.0	L CHEEK ABSCESS
8	10000108	36533795	98.80	98.0	16.0	100.0	135.0	85.0	5	3.0	LEFT CHEEK SWELLING, Abscess
9	10000108	39513268	98.80	101.0	18.0	100.0	131.0	62.0	5	3.0	L FACIAL SWELLING

Figure 13

Raw Data Sample Image from pyxis.csv table

	subject_id	stay_id	charttime	med_rn	name	gsn_rn	gsn
0	10000032	32952584	2180-07-22 17:59:00	1	Albuterol Inhaler	1	5037.0
1	10000032	32952584	2180-07-22 17:59:00	1	Albuterol Inhaler	2	28090.0
2	10000032	35968195	2180-08-05 22:29:00	1	Morphine	1	4080.0
3	10000032	35968195	2180-08-05 22:55:00	2	Donnatol (Elixir)	1	4773.0
4	10000032	35968195	2180-08-05 22:55:00	3	Aluminum-Magnesium Hydrox.-Simet	1	2701.0
5	10000032	35968195	2180-08-05 22:55:00	3	Aluminum-Magnesium Hydrox.-Simet	2	2716.0
6	10000032	35968195	2180-08-05 22:55:00	4	Ondansetron	1	15869.0
7	10000032	35968195	2180-08-05 22:55:00	4	Ondansetron	2	61716.0
8	10000032	38112554	2180-06-26 16:58:00	1	Morphine	1	4080.0
9	10000032	38112554	2180-06-26 16:59:00	2	Ondansetron	1	15869.0

Figure 14

Raw Data Sample Image from vitalsign.csv table

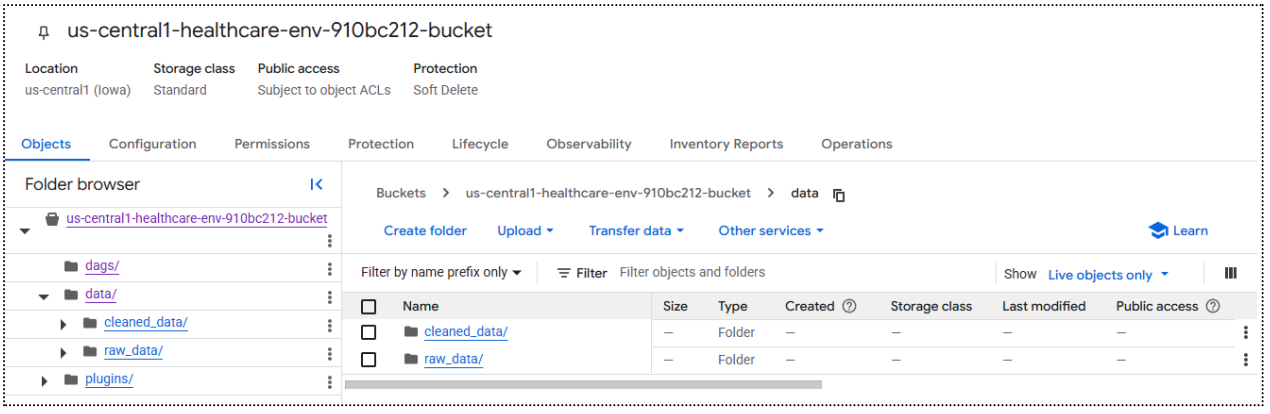
	subject_id	stay_id	charttime	temperature	heartrate	resprate	o2sat	sbp	dbp	rhythm	pain
0	10000032	32952584	2180-07-22 16:36:00	NaN	83.0	24.0	97.0	90.0	51.0	NaN	0
1	10000032	32952584	2180-07-22 16:43:00	NaN	85.0	22.0	98.0	76.0	39.0	NaN	0
2	10000032	32952584	2180-07-22 16:45:00	NaN	84.0	22.0	97.0	75.0	39.0	NaN	0
3	10000032	32952584	2180-07-22 17:56:00	NaN	84.0	20.0	99.0	86.0	51.0	NaN	NaN
4	10000032	32952584	2180-07-22 18:37:00	98.4	86.0	20.0	98.0	65.0	37.0	NaN	NaN
5	10000032	32952584	2180-07-22 18:38:00	NaN	85.0	16.0	99.0	83.0	45.0	NaN	0
6	10000032	32952584	2180-07-22 19:47:00	98.2	85.0	18.0	98.0	81.0	38.0	NaN	0
7	10000032	33258284	2180-05-06 23:04:00	97.7	79.0	16.0	98.0	107.0	60.0	NaN	0
8	10000032	35968195	2180-08-05 23:50:00	98.5	96.0	17.0	100.0	102.0	58.0	NaN	NaN
9	10000032	35968195	2180-08-06 01:07:00	98.1	91.0	18.0	99.0	98.0	60.0	NaN	NaN

3.3 Data Pre-processing

Raw dataset is pre-processed and cleaned using python pandas script automated through airflow. Raw data is pulled from GCS bucket and cleaned data is stored back to the same bucket under clean-data directory to make sure that this data is available if required in future. The data cleaning workflow focused on improving consistency, removing noise, handling missing data, and validating schema integrity.

Figure 15

GCS Bucket



Cleaning steps description

The diagnosis table, as represented in Figure 16, was cleaned by standardizing both the icd_code and icd_title fields to uppercase and removing any extra whitespace to ensure consistency in formatting. The icd_version column was validated to confirm that values were limited to either 9 or 10, aligning with official coding standards. Additionally, the seq_num field was checked to ensure it fell within the expected range of 1 to 9, which represents the order of diagnoses assigned during the ED visit. All columns were preserved without dropping any fields to maintain compatibility with future JSON conversion and downstream processing.

Figure 16

Summary of the Diagnosis Table Including Schema, Sample Records, and Statistical Overview

```
============================================================
                    ANALYSIS OF DIAGNOSIS TABLE
============================================================

Table Dimensions: 899,050 rows × 6 columns
Columns: ['subject_id', 'stay_id', 'seq_num', 'icd_code', 'icd_version', 'icd_title']

Data Types:
  subject_id           int64
  stay_id              int64
  seq_num              int64
  icd_code             object
  icd_version          int64
  icd_title            object

Missing Values:
  ✓ No missing values found!

Duplicate Rows: 0 (0.00%)
  ✓ No duplicates found!

Sample Data (first 5 rows):
   subject_id   stay_id  seq_num icd_code  icd_version                                  icd_title
0    10000032  32952584        1     4589            9                        HYPOTENSION NOS
1    10000032  32952584        2    07070            9  UNSPECIFIED VIRAL HEPATITIS C WITHOUT HEPATIC COMA
2    10000032  32952584        3      V08            9                ASYMPTOMATIC HIV INFECTION
3    10000032  33258284        1     5728            9                  OTH SEQUELA, CHR LIV DIS
4    10000032  33258284        2    78959            9                             OTHER ASCITES

Numeric Columns Statistics:
         subject_id       stay_id        seq_num    icd_version
count  8.990500e+05  8.990500e+05  899050.000000  899050.000000
mean   1.500838e+07  3.500095e+07       1.929149       9.507241
std    2.879810e+06  2.888942e+06       1.159710       0.499948
min    1.000003e+07  3.000001e+07       1.000000       9.000000
25%    1.251347e+07  3.250347e+07       1.000000       9.000000
50%    1.502022e+07  3.499600e+07       2.000000      10.000000
75%    1.750406e+07  3.751290e+07       2.000000      10.000000
max    1.999999e+07  3.999996e+07       9.000000      10.000000
```

Figure 17

Preview of the Diagnosis Table from MIMIC-IV Showing Subject-Level ICD

Diagnosis | Query | Open in | Share | Copy | Snapshot

Schema | Details | Preview | Table Explorer Preview | Insights | Lineage

Row	subject_id	stay_id	seq_num	icd_code	icd_version	icd_title
1	10427677	35873964	1	0020	9	TYPHOID FEVER
2	11545281	36308244	1	0030	9	SALMONELLA ENTERITIS
3	10271383	31799013	1	0059	9	FOOD POISONING NOS
4	13079787	31195432	1	0059	9	FOOD POISONING NOS
5	14422255	35085917	1	0059	9	FOOD POISONING NOS
6	16065224	38752840	1	0059	9	FOOD POISONING NOS
7	16399727	30309890	1	0059	9	FOOD POISONING NOS
8	18024585	33365865	1	0059	9	FOOD POISONING NOS
9	18106634	34272117	1	0059	9	FOOD POISONING NOS
10	18622001	37069523	1	0059	9	FOOD POISONING NOS

Figure 18

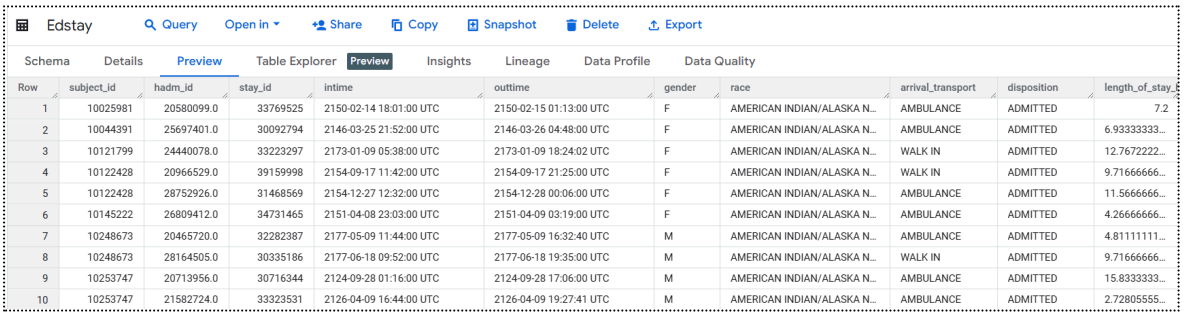
Summary of the Edstays Table Including Schema, Sample Records, and Statistical Overview

```
============================================================
                ANALYSIS OF EDSTAYS TABLE
============================================================

Table Dimensions: 425,087 rows × 9 columns
Columns: ['subject_id', 'hadm_id', 'stay_id', 'intime', 'outtime', 'gender', 'race', 'arrival_transport', 'disposition']

Data Types:
  subject_id           int64
  hadm_id              float64
  stay_id              int64
  intime               object
  outtime              object
  gender               object
  race                 object
  arrival_transport    object
  disposition          object

Missing Values:
  hadm_id              222,071 values (52.24%)

Duplicate Rows: 0 (0.00%)
  ✓ No duplicates found!

Sample Data (first 5 rows):
   subject_id     hadm_id   stay_id              intime             outtime gender   race arrival_transport disposition
0    10000032  22595853.0  33258284  2180-05-06 19:17:00  2180-05-06 23:30:00      F  WHITE         AMBULANCE    ADMITTED
1    10000032  22841357.0  38112554  2180-06-26 15:54:00  2180-06-26 21:31:00      F  WHITE         AMBULANCE    ADMITTED
2    10000032  25742920.0  35968195  2180-08-05 20:58:00  2180-08-06 01:44:00      F  WHITE         AMBULANCE    ADMITTED
3    10000032  29079034.0  32952584  2180-07-22 16:24:00  2180-07-23 05:54:00      F  WHITE         AMBULANCE        HOME
4    10000032  29079034.0  39399961  2180-07-23 05:54:00  2180-07-23 14:00:00      F  WHITE         AMBULANCE    ADMITTED

Numeric Columns Statistics:
         subject_id       hadm_id       stay_id
count  4.250870e+05  2.030160e+05  4.250870e+05
mean   1.500871e+07  2.499495e+07  3.499735e+07
std    2.878486e+06  2.888745e+06  2.888342e+06
min    1.000003e+07  2.000002e+07  3.000001e+07
25%    1.251789e+07  2.248982e+07  3.249799e+07
50%    1.501628e+07  2.499935e+07  3.499395e+07
75%    1.749900e+07  2.749388e+07  3.750387e+07
max    1.999999e+07  2.999981e+07  3.999996e+07
```

The cleaning process was tailored to respect the clinical context of emergency department data. Missing hadm_id values were retained, as they only apply to patients who were admitted, and their absence accurately reflects non-admission cases. The intime and outtime columns were converted to datetime format to support temporal analysis, and categorical fields such as gender, race, arrival_transport, and disposition were standardized to uppercase for consistency. The script verified that intime precedes outtime and computed a new length_of_stay_hrs field. Entries with unrealistic stay durations—either shorter than 15 minutes or longer than 7 days—were flagged. Additional validations ensured that hadm_id is present when a patient is marked as "ADMITTED", and that subject_id values fall within a valid numeric range expected in MIMIC-IV identifiers.

Figure 19

Preview of the Edstay Table After Pre-Processing



Edstay

Query | Open in | Share | Copy | Snapshot | Delete | Export

Schema | Details | Preview | Table Explorer Preview | Insights | Lineage | Data Profile | Data Quality

Row	subject_id	hadm_id	stay_id	intime	outtime	gender	race	arrival_transport	disposition	length_of_stay_
1	10025981	20580099.0	33769525	2150-02-14 18:01:00 UTC	2150-02-15 01:13:00 UTC	F	AMERICAN INDIAN/ALASKA N...	AMBULANCE	ADMITTED	7.2
2	10044391	25697401.0	30092794	2146-03-25 21:52:00 UTC	2146-03-26 04:48:00 UTC	F	AMERICAN INDIAN/ALASKA N...	AMBULANCE	ADMITTED	6.93333333...
3	10121799	24440078.0	33223297	2173-01-09 05:38:00 UTC	2173-01-09 18:24:02 UTC	F	AMERICAN INDIAN/ALASKA N...	WALK IN	ADMITTED	12.7672222...
4	10122428	20966529.0	39159998	2154-09-17 11:42:00 UTC	2154-09-17 21:25:00 UTC	F	AMERICAN INDIAN/ALASKA N...	WALK IN	ADMITTED	9.71666666...
5	10122428	28752926.0	31468569	2154-12-27 12:32:00 UTC	2154-12-28 00:06:00 UTC	F	AMERICAN INDIAN/ALASKA N...	AMBULANCE	ADMITTED	11.5666666...
6	10145222	26809412.0	34731465	2151-04-08 23:03:00 UTC	2151-04-09 03:19:00 UTC	F	AMERICAN INDIAN/ALASKA N...	AMBULANCE	ADMITTED	4.26666666...
7	10248673	20465720.0	32282387	2177-05-09 11:44:00 UTC	2177-05-09 16:32:40 UTC	M	AMERICAN INDIAN/ALASKA N...	AMBULANCE	ADMITTED	4.81111111...
8	10248673	28164505.0	30335186	2177-06-18 09:52:00 UTC	2177-06-18 19:35:00 UTC	M	AMERICAN INDIAN/ALASKA N...	WALK IN	ADMITTED	9.71666666...
9	10253747	20713956.0	30716344	2124-09-28 01:16:00 UTC	2124-09-28 17:06:00 UTC	M	AMERICAN INDIAN/ALASKA N...	AMBULANCE	ADMITTED	15.8333333...
10	10253747	21582724.0	33323531	2126-04-09 16:44:00 UTC	2126-04-09 19:27:41 UTC	M	AMERICAN INDIAN/ALASKA N...	AMBULANCE	ADMITTED	2.72805555...

The Medrecon table (see Figure 20) underwent several targeted cleaning steps to enhance data quality and consistency (see Figure 21). Zero values in the gsn and ndc columns were replaced with NaN as per MIMIC-IV conventions, and the charttime field was converted to proper datetime format. Medication names were standardized to lowercase, and the etcdescription field was cleaned to remove extra spaces. The script also counted the number of unique medications and ETC descriptions to assess variability. To ensure data integrity, entries with invalid etc_rn values (less than or equal to zero) were identified, and

medication names that were unusually short (under 3 characters) or excessively long (over 50 characters) were flagged for review.

Figure 20

Summary of the Medrecon Table Including Schema, Sample Records, and Statistical Overview

```
                    ANALYSIS OF MEDRECON TABLE
============================================================

Table Dimensions: 2,987,342 rows × 9 columns
Columns: ['subject_id', 'stay_id', 'charttime', 'name', 'gsn', 'ndc', 'etc_rn', 'etccode', 'etcdescription']

Data Types:
  subject_id           int64
  stay_id              int64
  charttime            object
  name                 object
  gsn                  int64
  ndc                  int64
  etc_rn               int64
  etccode              float64
  etcdescription       object

Missing Values:
  etccode              11,728 values (0.39%)
  etcdescription       11,728 values (0.39%)

Duplicate Rows: 0 (0.00%)
  ✓ No duplicates found!

Sample Data (first 5 rows):
   subject_id   stay_id            charttime                              name    gsn          ndc  etc_rn  etccode
etcdescription
0    10000032  32952584  2180-07-22 17:26:00                 albuterol sulfate  28090  21695042308       1   5970.0  Asthma/COPD Therapy - Beta 2-Adrenergic A
gents, Inhaled, Short Acting
1    10000032  32952584  2180-07-22 17:26:00                 calcium carbonate   1340  10135021101       1    733.0                       Minerals and Electr
olytes - Calcium Replacement
2    10000032  32952584  2180-07-22 17:26:00       cholecalciferol (vitamin D3)  65241  37205024678       1    670.0
Vitamins - D Derivatives
3    10000032  32952584  2180-07-22 17:26:00  emtricitabine-tenofovir [Truvada]  57883  35356007003       1   5849.0   Antiretroviral - Nucleoside and Nucleot
ide Analog RTIs Combinations
4    10000032  32952584  2180-07-22 17:26:00          fluticasone [Flovent HFA]  21251  49999061401       1    371.0            Asthma Therapy - Inhaled Corti
costeroids (Glucocorticoids)

Numeric Columns Statistics:
         subject_id      stay_id          gsn          ndc       etc_rn       etccode
count  2.987342e+06  2.987342e+06  2.987342e+06  2.987342e+06  2.987342e+06  2.975614e+06
mean   1.499808e+07  3.499961e+07  2.271056e+04  1.605474e+10  1.139930e+00  2.457213e+03
std    2.871188e+06  2.886385e+06  1.975485e+04  1.175754e+10  3.738939e-01  2.256507e+03
min    1.000003e+07  3.000001e+07  0.000000e+00  0.000000e+00  1.000000e+00  2.000000e+00
25%    1.251744e+07  3.249908e+07  4.560000e+03  1.033702e+10  1.000000e+00  5.230000e+02
50%    1.499762e+07  3.499626e+07  1.699500e+04  1.093905e+10  1.000000e+00  1.158000e+03
75%    1.747820e+07  3.749729e+07  4.018600e+04  1.659002e+10  1.000000e+00  3.948000e+03
max    1.999983e+07  3.999996e+07  2.375020e+05  9.920703e+10  5.000000e+00  6.884000e+03
```

Figure 21

Preview of the Medrecon Table After Pre-Processing

Medrecon

Query | Open in | Share | Copy | Snapshot

Schema | Details | Preview | Table Explorer Preview | Insights | Lineage

Row	subject_id	stay_id	charttime	name
1	16662495	30688652	2145-10-07 22:08:00 UTC	*norethisterone
2	16924121	37297953	2119-03-12 16:44:00 UTC	*birth control
3	17014125	34546866	2171-01-02 22:57:00 UTC	*iud
4	17219425	30187720	2150-12-22 23:56:00 UTC	*insulin sliding scale
5	11236521	36927997	2139-05-18 12:04:00 UTC	*ceraplex supplement
6	17717686	37650034	2120-08-02 01:55:00 UTC	*ocp
7	17749277	31156216	2189-09-06 14:50:00 UTC	*mirena iud
8	14222176	31036857	2193-04-01 08:18:00 UTC	*prolixin
9	14517019	35898564	2169-12-07 14:07:00 UTC	*glymperide
10	14690648	38689062	2171-07-11 02:42:00 UTC	*glargine

Figure 22

Summary of the Triage Table Including Schema, Sample Records, and Statistical Overview

```
============================================================
                    ANALYSIS OF TRIAGE TABLE
============================================================

Table Dimensions: 425,087 rows × 11 columns
Columns: ['subject_id', 'stay_id', 'temperature', 'heartrate', 'resprate', 'o2sat', 'sbp', 'dbp', 'pain', 'acuity', 'chiefcomplaint']

Data Types:
  subject_id           int64
  stay_id              int64
  temperature          float64
  heartrate            float64
  resprate             float64
  o2sat                float64
  sbp                  float64
  dbp                  float64
  pain                 object
  acuity               float64
  chiefcomplaint       object

Missing Values:
  temperature          23,415 values (5.51%)
  heartrate            17,090 values (4.02%)
  resprate             20,353 values (4.79%)
  o2sat                20,596 values (4.85%)
  sbp                  18,291 values (4.3%)
  dbp                  19,091 values (4.49%)
  pain                 12,933 values (3.04%)
  acuity               6,987 values (1.64%)
  chiefcomplaint       23 values (0.01%)

Duplicate Rows: 0 (0.00%)
  ✓ No duplicates found!

Sample Data (first 5 rows):
   subject_id   stay_id  temperature  heartrate  resprate  o2sat    sbp   dbp pain  acuity                          chiefcomplaint
0    10000032  32952584         97.8       87.0      14.0   97.0   71.0  43.0    7     2.0                             Hypotension
1    10000032  33258284         98.4       70.0      16.0   97.0  106.0  63.0    0     3.0             Abd pain, Abdominal distention
2    10000032  35968195         99.4      105.0      18.0   96.0  106.0  57.0   10     3.0                           n/v/d, Abd pain
3    10000032  38112554         98.9       88.0      18.0   97.0  116.0  88.0   10     3.0                       Abdominal distention
4    10000032  39399961         98.7       77.0      16.0   98.0   96.0  50.0   13     2.0  Abdominal distention, Abd pain, LETHAGIC
```

Figure 23

Descriptive Statistics of Numeric Variables Including Vital Signs and Acuity Scores from the Emergency Department Dataset

```
Numeric Columns Statistics:
         subject_id       stay_id    temperature      heartrate       resprate          o2sat            sbp            dbp         acuity
count  4.250870e+05  4.250870e+05  401672.000000  407997.000000  404734.000000  404491.000000  406796.000000  405996.000000  418100.000000
mean   1.500871e+07  3.499735e+07      98.015046      85.079891      17.565521      98.471888     135.395352      81.262126       2.625102
std    2.878486e+06  2.888342e+06       4.008575      18.041690       5.485706      17.040807     240.956408    1057.220031       0.708084
min    1.000003e+07  3.000001e+07       0.100000       1.000000       0.000000       0.000000       1.000000       0.000000       1.000000
25%    1.251789e+07  3.249799e+07      97.500000      72.000000      16.000000      97.000000     120.000000      68.000000       2.000000
50%    1.501628e+07  3.499395e+07      98.000000      84.000000      18.000000      99.000000     133.000000      77.000000       3.000000
75%    1.749900e+07  3.750387e+07      98.600000      96.000000      18.000000     100.000000     148.000000      87.000000       3.000000
max    1.999999e+07  3.999996e+07     986.000000    1228.000000    1820.000000    9322.000000  151103.000000  661672.000000       5.000000
```

Figure 24

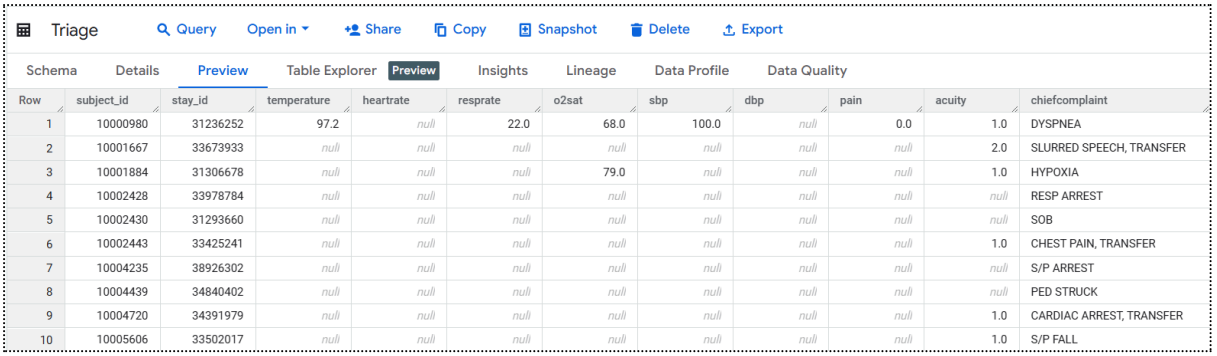
Summary Statistics and Outlier Detection for Vital Signs Highlighting Physiologically Impossible Values in the MIMIC-IV-ED Dataset

```
Vital Signs Analysis:

  temperature statistics:
    Min: 0.1, Max: 986.0
    Mean: 98.02, Median: 98.0
    ⚠ 524 physiologically impossible temperature values
    Potential outliers: 12,333 (2.90%)

  heartrate statistics:
    Min: 1.0, Max: 1228.0
    Mean: 85.08, Median: 84.0
    ⚠ 29 physiologically impossible heart rate values
    Potential outliers: 4,601 (1.08%)

  resprate statistics:
    Min: 0.0, Max: 1820.0
    Mean: 17.57, Median: 18.0
    ⚠ 60 physiologically impossible respiratory rate values
    Potential outliers: 21,249 (5.00%)

  o2sat statistics:
    Min: 0.0, Max: 9322.0
    Mean: 98.47, Median: 99.0
    ⚠ 139 physiologically impossible O2 saturation values
    Potential outliers: 5,178 (1.22%)

  sbp statistics:
    Min: 1.0, Max: 151103.0
    Mean: 135.40, Median: 133.0
    ⚠ 261 physiologically impossible systolic BP values
    Potential outliers: 7,765 (1.83%)

  dbp statistics:
    Min: 0.0, Max: 661672.0
    Mean: 81.26, Median: 77.0
    ⚠ 742 physiologically impossible diastolic BP values
    Potential outliers: 5,903 (1.39%)
```

The charttime column was dropped entirely, as it was deemed redundant for downstream use. Missing values across all columns were identified and logged. Vital signs were checked against clinically valid ranges, and physiologically impossible values were replaced with NaN to avoid introducing noise. The chief complaint field was standardized by converting all entries to uppercase, and any placeholders indicating redacted PHI (e.g., "___") were counted for transparency. The pain scores were cleaned by converting values to a numeric format, clipping them within the 0–10 range, and rounding to the nearest whole number. Similarly, the acuity field was converted to numeric, validated to ensure it fell within the expected scale of 1 to 5, and then rounded for consistency.

Figure 25

Preview of the Triage Table After Pre-Processing



Triage | Query | Open in | Share | Copy | Snapshot | Delete | Export

Schema | Details | Preview | Table Explorer Preview | Insights | Lineage | Data Profile | Data Quality

Row	subject_id	stay_id	temperature	heartrate	resprate	o2sat	sbp	dbp	pain	acuity	chiefcomplaint
1	10000980	31236252	97.2	null	22.0	68.0	100.0	null	0.0	1.0	DYSPNEA
2	10001667	33673933	null	null	null	null	null	null	null	2.0	SLURRED SPEECH, TRANSFER
3	10001884	31306678	null	null	null	79.0	null	null	null	1.0	HYPOXIA
4	10002428	33978784	null	null	null	null	null	null	null	null	RESP ARREST
5	10002430	31293660	null	null	null	null	null	null	null	null	SOB
6	10002443	33425241	null	null	null	null	null	null	null	1.0	CHEST PAIN, TRANSFER
7	10004235	38926302	null	null	null	null	null	null	null	null	S/P ARREST
8	10004439	34840402	null	null	null	null	null	null	null	null	PED STRUCK
9	10004720	34391979	null	null	null	null	null	null	null	1.0	CARDIAC ARREST, TRANSFER
10	10005606	33502017	null	null	null	null	null	null	null	1.0	S/P FALL

In clinical datasets, explicitly representing missing values using NaN is essential for preserving the semantic integrity of the data. Unlike zero or arbitrary imputation, NaN accurately reflects the absence of information, which can carry clinical significance—such as a lab test not ordered due to a physician's judgment or the patient's condition. This distinction is critical in high-stakes medical applications, where data interpretation must align with real-world decision-making. When fine-tuning large language models (LLMs), retaining NaN values allows the model to learn from patterns of both presence and absence, supporting more nuanced and context-aware predictions. Modern LLM training pipelines can incorporate special tokens or masking strategies to handle NaN, enabling models to treat missingness as an informative feature rather than noise. This approach enhances the robustness, interpretability, and clinical relevance of the resulting models.

Figure 26

Summary of the Pyxis Table Including Schema, Sample Records, and Statistical Overview

```
============================================================
                    ANALYSIS OF PYXIS TABLE
============================================================

Table Dimensions: 1,586,053 rows × 7 columns
Columns: ['subject_id', 'stay_id', 'charttime', 'med_rn', 'name', 'gsn_rn', 'gsn']

Data Types:
  subject_id           int64
  stay_id              int64
  charttime            object
  med_rn               int64
  name                 object
  gsn_rn               int64
  gsn                  float64

Missing Values:
  gsn                  35,452 values (2.24%)

Duplicate Rows: 0 (0.00%)
  ✓ No duplicates found!

Sample Data (first 5 rows):
   subject_id   stay_id           charttime  med_rn                          name  gsn_rn      gsn
0    10000032  32952584  2180-07-22 17:59:00       1             Albuterol Inhaler       1   5037.0
1    10000032  32952584  2180-07-22 17:59:00       1             Albuterol Inhaler       2  28090.0
2    10000032  35968195  2180-08-05 22:29:00       1                      Morphine       1   4080.0
3    10000032  35968195  2180-08-05 22:55:00       2              Donnatol (Elixir)       1   4773.0
4    10000032  35968195  2180-08-05 22:55:00       3  Aluminum-Magnesium Hydrox.-Simet       1   2701.0

Numeric Columns Statistics:
         subject_id      stay_id        med_rn        gsn_rn           gsn
count  1.586053e+06  1.586053e+06  1.586053e+06  1.586053e+06  1.550601e+06
mean   1.498814e+07  3.500184e+07  4.470095e+00  1.293055e+00  2.474049e+04
std    2.872346e+06  2.889039e+06  5.795464e+00  5.478829e-01  2.461636e+04
min    1.000003e+07  3.000001e+07  1.000000e+00  1.000000e+00  1.500000e+01
25%    1.251086e+07  3.250018e+07  1.000000e+00  1.000000e+00  4.380000e+03
50%    1.498308e+07  3.500212e+07  3.000000e+00  1.000000e+00  1.167700e+04
75%    1.746739e+07  3.751120e+07  5.000000e+00  2.000000e+00  4.395200e+04
max    1.999999e+07  3.999996e+07  1.770000e+02  4.000000e+00  7.927800e+04
```

Figure 27

Preview of the Pyxis Table After Pre-Processing

Pyxis | Query | Open in | Share | Copy | Snapshot | Delete | Export

Schema | Details | Preview | Table Explorer Preview | Insights | Lineage | Data Profile | Data Quai

Row	subject_id	stay_id	charttime	med_rn	name	gsn_rn	gsn
1	10001884	34226385	2130-11-20 08:51:00 UTC	16	DOCUSATE SODIUM 100MG C...	1	3009.0
2	10001884	34226385	2130-11-20 08:51:00 UTC	17	RANITIDINE 150MG TAB	1	11673.0
3	10001884	34226385	2130-11-20 08:51:00 UTC	18	DILTIAZEM EXTENDED-REL 12...	1	17205.0
4	10001884	34226385	2130-11-20 08:51:00 UTC	18	DILTIAZEM EXTENDED-REL 12...	2	24536.0
5	10001884	34226385	2130-11-20 08:51:00 UTC	19	HYDROCHLOROTHIAZIDE 25M...	1	8182.0
6	10001884	34226385	2130-11-20 08:51:00 UTC	19	HYDROCHLOROTHIAZIDE 25M...	2	29832.0
7	10001884	37461620	2130-06-24 09:38:00 UTC	17	PREDNISONE	2	6751.0
8	10002807	34934098	2152-03-31 07:14:00 UTC	16	HYDROCHLOROTHIAZIDE 25M...	1	8182.0
9	10002807	34934098	2152-03-31 07:14:00 UTC	16	HYDROCHLOROTHIAZIDE 25M...	2	29832.0
10	10002807	34934098	2152-03-31 07:14:00 UTC	17	AMLODIPINE 5MG TAB	1	16926.0

Several essential steps to ensure clinical accuracy and data consistency. First, the charttime column was converted to proper datetime format to support time-series analysis. A thorough assessment of missing values was performed across all columns.

Figure 28

Summary of the Vitalsign Table Including Schema, Sample Records, and Statistical Overview

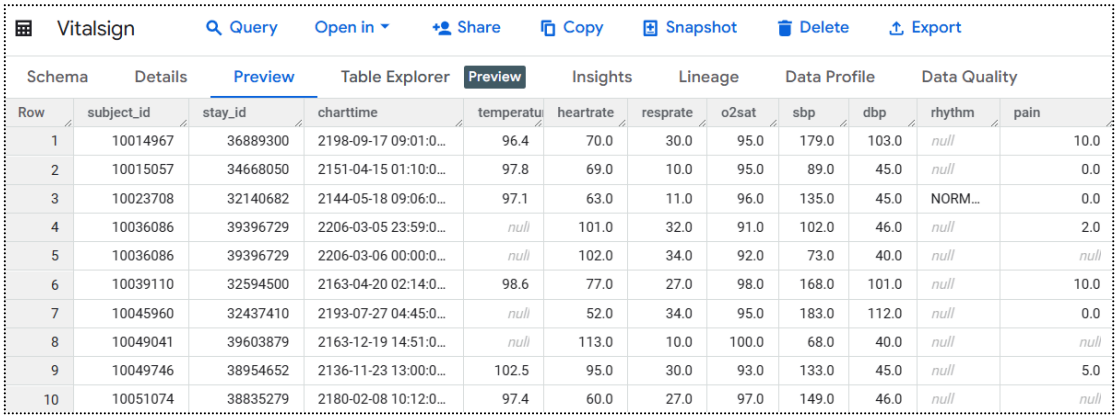
```
============================================================
                    ANALYSIS OF VITALSIGN TABLE
============================================================

Table Dimensions: 1,564,610 rows × 11 columns
Columns: ['subject_id', 'stay_id', 'charttime', 'temperature', 'heartrate', 'resprate', 'o2sat', 'sbp', 'dbp', 'rhythm', 'pain']

Data Types:
  subject_id           int64
  stay_id              int64
  charttime            object
  temperature          float64
  heartrate            float64
  resprate             float64
  o2sat                float64
  sbp                  float64
  dbp                  float64
  rhythm               object
  pain                 object
```

```
Missing Values:
  temperature          564,968 values (36.11%)
  heartrate            69,710 values (4.46%)
  resprate             89,393 values (5.71%)
  o2sat                135,836 values (8.68%)
  sbp                  81,256 values (5.19%)
  dbp                  81,256 values (5.19%)
  rhythm               1,504,960 values (96.19%)
  pain                 443,266 values (28.33%)

Duplicate Rows: 0 (0.00%)
  ✓ No duplicates found!

Sample Data (first 5 rows):
   subject_id   stay_id            charttime  temperature  heartrate  resprate  o2sat   sbp   dbp rhythm pain
0    10000032  32952584  2180-07-22 16:36:00          NaN       83.0      24.0   97.0  90.0  51.0     NaN    0
1    10000032  32952584  2180-07-22 16:43:00          NaN       85.0      22.0   98.0  76.0  39.0     NaN    0
2    10000032  32952584  2180-07-22 16:45:00          NaN       84.0      22.0   97.0  75.0  39.0     NaN    0
3    10000032  32952584  2180-07-22 17:56:00          NaN       84.0      20.0   99.0  86.0  51.0     NaN  NaN
4    10000032  32952584  2180-07-22 18:37:00         98.4       86.0      20.0   98.0  65.0  37.0     NaN  NaN

Numeric Columns Statistics:
         subject_id       stay_id    temperature     heartrate      resprate         o2sat           sbp           dbp
count  1.564610e+06  1.564610e+06  999642.000000  1.494900e+06  1.475217e+06  1.428774e+06  1.483354e+06  1.483354e+06
mean   1.500224e+07  3.500079e+07      97.999308  8.122277e+01  1.777930e+01  9.791336e+01  1.284333e+02  7.435518e+01
std    2.873032e+06  2.888296e+06       8.193808  1.795872e+01  7.840819e+01  1.474334e+01  2.294265e+01  1.799864e+02
min    1.000003e+07  3.000001e+07       0.000000  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
25%    1.251479e+07  3.249958e+07      97.800000  6.900000e+01  1.600000e+01  9.700000e+01  1.130000e+02  6.300000e+01
50%    1.500674e+07  3.500071e+07      98.100000  7.900000e+01  1.800000e+01  9.800000e+01  1.260000e+02  7.200000e+01
75%    1.748279e+07  3.750565e+07      98.500000  9.100000e+01  1.800000e+01  1.000000e+02  1.420000e+02  8.200000e+01
max    1.999999e+07  3.999996e+07     989.000000  1.109000e+03  9.445200e+04  9.997000e+03  1.234000e+03  9.710000e+04

Vital Signs Analysis:

  temperature statistics:
    Min: 0.0, Max: 989.0
    Mean: 98.00, Median: 98.1
    ⚠ 4,418 physiologically impossible temperature values
    Potential outliers: 72,313 (4.62%)

  heartrate statistics:
    Min: 0.0, Max: 1109.0
    Mean: 81.22, Median: 79.0
    ⚠ 639 physiologically impossible heart rate values
    Potential outliers: 32,227 (2.06%)

  resprate statistics:
    Min: 0.0, Max: 94452.0
    Mean: 17.78, Median: 18.0
    ⚠ 569 physiologically impossible respiratory rate values
    Potential outliers: 160,158 (10.24%)

  o2sat statistics:
    Min: 0.0, Max: 9997.0
    Mean: 97.91, Median: 98.0
    ⚠ 852 physiologically impossible O2 saturation values
    Potential outliers: 26,150 (1.67%)

  sbp statistics:
    Min: 0.0, Max: 1234.0
    Mean: 128.43, Median: 126.0
    ⚠ 2,533 physiologically impossible systolic BP values
    Potential outliers: 28,447 (1.82%)

  dbp statistics:
    Min: 0.0, Max: 97100.0
    Mean: 74.36, Median: 72.0
    ⚠ 2,164 physiologically impossible diastolic BP values
    Potential outliers: 20,671 (1.32%)
```

Physiologically implausible values in vital signs such as heart rate, respiratory rate, and blood pressure were identified using clinically accepted ranges and replaced with NaN. The rhythm field was standardized by converting inconsistent null indicators (e.g., "UNKNOWN", empty strings) to actual null values and formatting all entries in uppercase. Finally, the pain scores were cleaned by converting them to a numeric format, capping the values within the valid range of 0 to 10, and rounding to the nearest integer to align with clinical recording standards.

Figure 29

Preview of the VitalSigns Table After Pre-Processing



Row	subject_id	stay_id	charttime	temperatu	heartrate	resprate	o2sat	sbp	dbp	rhythm	pain
1	10014967	36889300	2198-09-17 09:01:0...	96.4	70.0	30.0	95.0	179.0	103.0	null	10.0
2	10015057	34668050	2151-04-15 01:10:0...	97.8	69.0	10.0	95.0	89.0	45.0	null	0.0
3	10023708	32140682	2144-05-18 09:06:0...	97.1	63.0	11.0	96.0	135.0	45.0	NORM...	0.0
4	10036086	39396729	2206-03-05 23:59:0...	null	101.0	32.0	91.0	102.0	46.0	null	2.0
5	10036086	39396729	2206-03-06 00:00:0...	null	102.0	34.0	92.0	73.0	40.0	null	null
6	10039110	32594500	2163-04-20 02:14:0...	98.6	77.0	27.0	98.0	168.0	101.0	null	10.0
7	10045960	32437410	2193-07-27 04:45:0...	null	52.0	34.0	95.0	183.0	112.0	null	0.0
8	10049041	39603879	2163-12-19 14:51:0...	null	113.0	10.0	100.0	68.0	40.0	null	null
9	10049746	38954652	2136-11-23 13:00:0...	102.5	95.0	30.0	93.0	133.0	45.0	null	5.0
10	10051074	38835279	2180-02-08 10:12:0...	97.4	60.0	27.0	97.0	149.0	46.0	null	null

Cleaning Strategy

Given the sensitive and high-stakes nature of healthcare data, we deliberately avoided aggressive removal or imputation of missing values. In clinical datasets such as MIMIC-IV-ED, missing data often carries implicit clinical meaning—for instance, the absence of a lab result or vital sign may reflect clinical judgment, patient status, or operational constraints, rather than data entry errors. Imputing these values risks introducing bias or artificial variability, particularly when the data is used for fine-tuning transformer

models or implementing retrieval-augmented generation (RAG), where preserving clinical context and interpretability is critical.

To uphold data fidelity and support realistic downstream modeling, we retained the original structure and inherent missingness of the dataset. This approach promotes transparency, clinical traceability, and robustness in model training and evaluation.

Across our MIMIC-IV ED dataset, we have deliberately preserved missing values to maintain the clinical fidelity of the data. For example, hadm_id is absent in roughly half of the edstays records because those visits did not result in admission; imputing or dropping them would erase the true prevalence of ED-only encounters. In MedRecon and Pyxis, zero entries for GSN and NDC semantically indicate "unknown" device or medication codes—so we converted zeros to NaN but left them un-imputed to flag genuine missingness. Vital-sign measurements falling outside physiologic ranges were likewise set to NaN and retained, since omitted readings often reflect clinical judgment calls or equipment issues rather than random error. Within triage, placeholder strings (e.g., "___") mark redacted PHI in chief_complaint, and missing acuity scores denote instances where triage scoring was legitimately deferred; keeping these markers ensures downstream pipelines can detect PHI redaction and learn from real-world patterns of data absence. By preserving these "true" missing values, we uphold the integrity of our dataset and allow our models to learn both from what was recorded and from what clinicians chose not to record.

A notable instance is the field hadm_id, which is absent in approximately 50% of the records. This omission is intentional rather than erroneous—hadm_id is only generated when a patient is admitted to the hospital from the emergency department (ED). Many patients receive treatment in the ED and are discharged without admission, so the lack of hadm_id accurately indicates such non-admission cases. Removing or imputing these values would

discard legitimate ED encounters, thereby diminishing the dataset's clinical validity and introducing potential bias into model training.

Preserving this missing data allows the dataset to remain representative of real-world scenarios, enabling the development of patient-centric models that can leverage both observed and unobserved features. This approach aligns with the complexities of actual clinical workflows and supports more generalizable and meaningful model behavior.

3.4 Data Transformation

The transformation phase of the pipeline focused on curating a high-quality dataset suitable for training and evaluating a domain-specific language model and RAG-based system. Rather than working with entire source files, relevant columns were selectively extracted from multiple raw datasets, including but not limited to edstays.csv and diagnosis.csv, based on their utility for the intended healthcare agent use case. These columns were then combined through a data integration process using unique patient identifiers, ensuring each row in the resulting dataset represented a semantically rich and unified patient record.

This operation involved a LEFT JOIN on shared patient keys to align complementary clinical data and diagnostic information, introducing new relationships across datasets. The transformed output, named *sample_data*, underwent schema evolution by incorporating columns from different sources, renaming fields for clarity, and structurally reshaping the data. A representative subset of 500 patients (subject_id centric) was selected to maintain computational efficiency, which also constitutes a filtering transformation. From a data engineering perspective, this transformation process reflects several core techniques: entity integration, data subsetting, dimensional modeling, and consolidation. The resulting dataset

was loaded into Google BigQuery to support efficient querying and downstream use in fine-tuning language models and building a Retrieval-Augmented Generation (RAG) system tailored to healthcare-specific tasks.

Figure 30

Big Query View of Transformed Sample Dataset

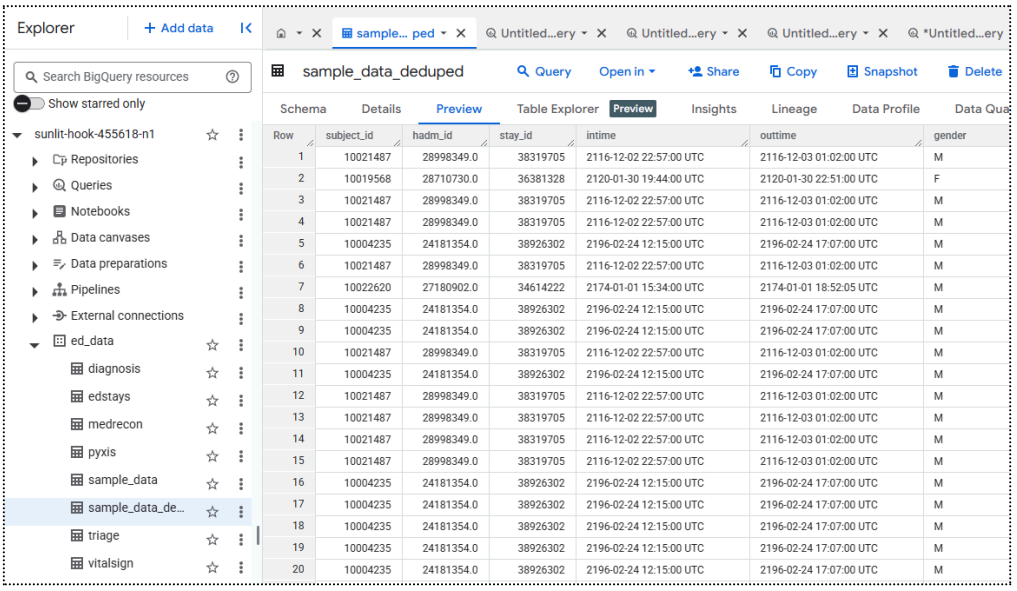
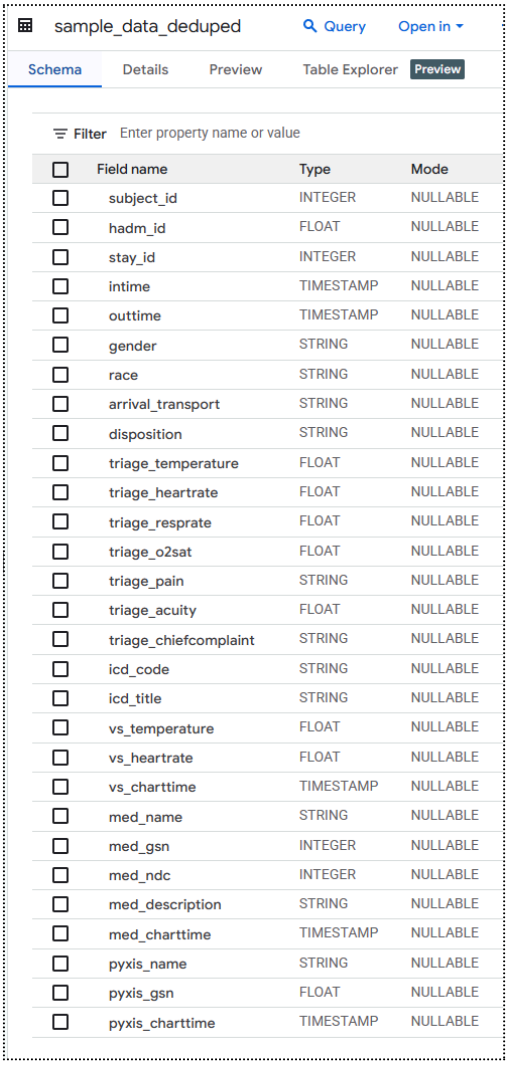


Figure 31

Schema of sample_data



sample_data_deduped Query Open in

Schema Details Preview Table Explorer Preview

Filter Enter property name or value

Field name	Type	Mode
subject_id	INTEGER	NULLABLE
hadm_id	FLOAT	NULLABLE
stay_id	INTEGER	NULLABLE
intime	TIMESTAMP	NULLABLE
outtime	TIMESTAMP	NULLABLE
gender	STRING	NULLABLE
race	STRING	NULLABLE
arrival_transport	STRING	NULLABLE
disposition	STRING	NULLABLE
triage_temperature	FLOAT	NULLABLE
triage_heartrate	FLOAT	NULLABLE
triage_resprate	FLOAT	NULLABLE
triage_o2sat	FLOAT	NULLABLE
triage_pain	STRING	NULLABLE
triage_acuity	FLOAT	NULLABLE
triage_chiefcomplaint	STRING	NULLABLE
icd_code	STRING	NULLABLE
icd_title	STRING	NULLABLE
vs_temperature	FLOAT	NULLABLE
vs_heartrate	FLOAT	NULLABLE
vs_charttime	TIMESTAMP	NULLABLE
med_name	STRING	NULLABLE
med_gsn	INTEGER	NULLABLE
med_ndc	INTEGER	NULLABLE
med_description	STRING	NULLABLE
med_charttime	TIMESTAMP	NULLABLE
pyxis_name	STRING	NULLABLE
pyxis_gsn	FLOAT	NULLABLE
pyxis_charttime	TIMESTAMP	NULLABLE

3.5 Data Preparation

Following data transformation, the next critical step involved structuring the dataset for training, validation, and testing. Given the project's objective to develop an intelligent agent capable of summarizing a patient's comprehensive emergency and hospital visit history based on their subject ID, a subject-centric data preparation strategy was implemented.

A Python script was employed to extract the processed sample data from BigQuery, generating a metadata file indexed by subject ID. This metadata enabled a systematic and clinically informed data split, ensuring that all hospital stay records (stay_ids) corresponding to a single subject remained within a single dataset partition. This approach effectively

prevents data leakage and promotes reliable evaluation by guaranteeing that a patient's history is not fragmented across different data splits.

The dataset was divided to include 70% of subjects in the training set, and 15% each in the validation and test sets. This split mirrors realistic clinical scenarios, enabling patient-level reasoning and longitudinal summarization over multiple visits. The method also supports fairness across demographic and clinical variables and maintains alignment with downstream tasks such as embedding generation, retrieval-augmented generation (RAG), and fine-tuning. Overall, this approach ensures a robust, reproducible, and clinically meaningful data pipeline.

Data Splitting Summary and Balance Analysis

To support fair model training, validation, and evaluation, the dataset consisting of over 700,000 clinical rows from 500 unique patients was divided using a subject-level stratification strategy. This method ensures that all entries for a single patient (subject_id) appear exclusively in one of the three splits, preserving temporal and contextual integrity and preventing data leakage across subsets.

The split followed a 70:15:15 ratio at the patient level. Based on this approach, the resulting distribution across splits is as follows:

- Training Set: ~525,000 rows from 70% of the patients
- Validation Set: ~90,000 rows from 15% of the patients
- Test Set: ~85,000 rows from 15% of the patients

Despite the even subject distribution, the number of rows per split is not equal due to the heterogeneous number of clinical records per patient. Some patients contribute significantly more rows due to longer or more complex emergency department (ED) encounters, especially those in the training split. This is an expected and natural characteristic of longitudinal electronic health record (EHR) data.

Additionally, demographic analysis of the splits revealed moderate gender imbalance in the validation and test sets, which is attributed to the uneven distribution of high-volume female patients across splits. Nonetheless, the training set maintains a balanced gender profile, making it well-suited for developing unbiased models.

This splitting strategy offers a realistic foundation for clinical machine learning by preserving subject independence, reflecting real-world clinical variance, and supporting robust model development and evaluation.

3.6 Data Statistics

To summarize the dataset underwent a multi-stage preparation process, transforming raw hospital data into a structured and model-ready format. This section summarizes the key characteristics and transitions across each stage, from raw input to cleaned, transformed, and finally split datasets.

Prepared Dataset Splits:

To facilitate robust evaluation and generalization of downstream models, the sample_500 dataset was partitioned into training, validation, and test subsets using a subject-level stratification strategy. This approach ensures that all records associated with a given subject_id are confined to a single split, thereby preventing data leakage and preserving patient-level independence across subsets.

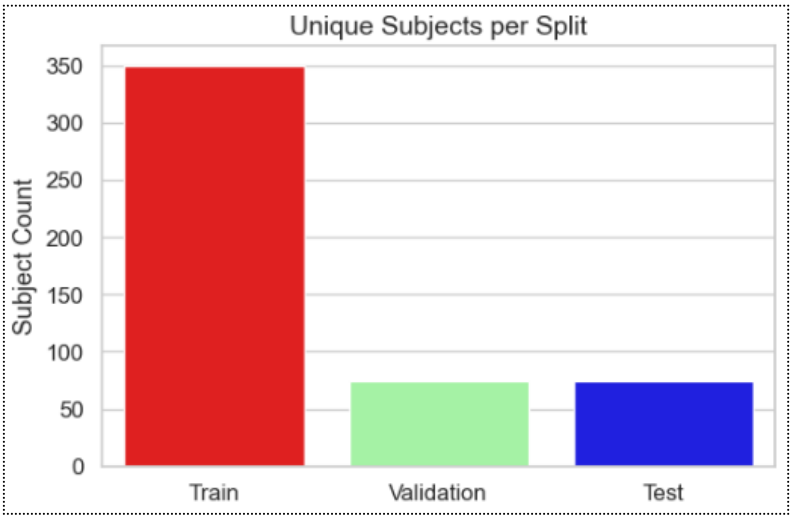
The dataset contains electronic health record (EHR) entries from 500 distinct patients. Using stratified sampling, 70% of the patients were allocated to the training set, while the remaining 30% were equally divided between validation and test sets. The splitting process was implemented in Python using pandas and scikit-learn to ensure reproducibility and consistency.

The resulting splits exhibit balanced distribution in terms of both the number of rows and the number of unique patients. These subsets form the foundation for model training, hyperparameter tuning, and final evaluation. Visualization of the splits further confirms the integrity and proportional representation of data across all three groups.

This patient-consistent data partitioning is a critical step in ensuring fair model validation in clinical machine learning workflows.

Figure 32

Data Split Imbalance by Subject_id feature.

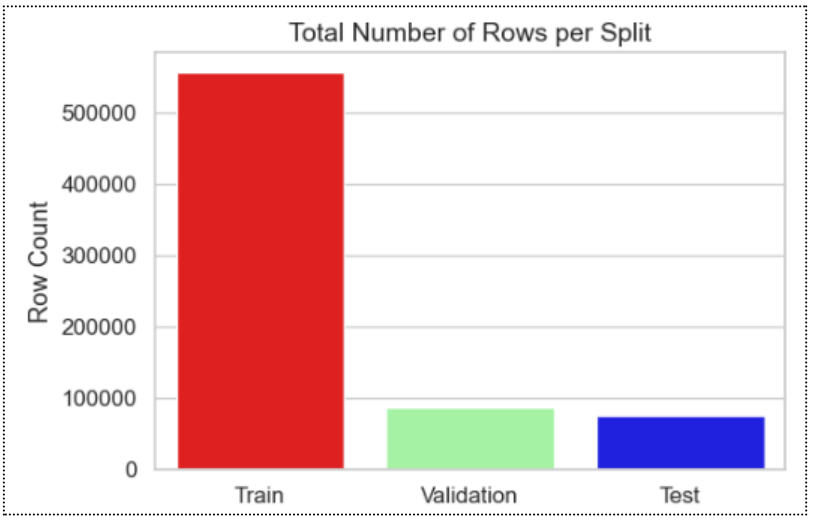


The bar chart in Figure 37 illustrates the distribution of the dataset into three distinct subsets: Training, Validation, and Test. The majority of the data, comprising 143,852 patients, was allocated to the Training set, which represents 70% of the total. Both the Validation and

Test sets contain 30,826 patients each, accounting for 15% of the dataset respectively. This division reflects a well-balanced, subject-level split strategy, ensuring that each patient (identified by a unique subject ID) is included in only one of the subsets. Such an approach effectively prevents data leakage across splits and supports a fair and unbiased evaluation of model performance.

Figure 33

Dataset Split by number of rows.



The bar chart titled “Total Number of Rows per Split” visualizes the volume of data points in each dataset partition—train, validation, and test—generated from the sample_500 dataset, which contains over 700,000 rows linked to 500 unique patients. Although patient-level stratification was used (70/15/15), the train split accounts for a significantly larger number of rows than the other two.

This is expected because patients contribute variable-length records depending on their clinical complexity or length of stay. Since the stratification was done by subject_id, not row count, patients in the training split may have had more interactions or events recorded (e.g., vitals, medications, diagnoses), resulting in a row-heavy training set.

This distribution is acceptable and often beneficial, as the training set requires more data to learn patterns effectively, while validation and test sets serve primarily for

performance evaluation. However, it also highlights the importance of analyzing both subject count and row volume when designing fair and balanced training workflows.

4 Model Development

4.1 Model Proposals

Clinical Explanation Agent - Model Proposal

The Clinical Explanation Agent presented in this report uses a Retrieval-Augmented Generation (RAG) architecture to provide accurate and patient-specific clinical explanations. This method combines structured healthcare data with advanced natural language processing techniques, making complex medical information clear and easy to understand for healthcare providers and patients.

Initially, clinical data is gathered from a structured BigQuery database. This database includes detailed patient information such as chief complaints, vital signs (temperature, heart rate, respiratory rate, blood pressure, oxygen saturation), medications, diagnostic codes (ICD-10), pain scores, patient acuity, and demographic details (gender, race, transportation method). This data is converted into clear, narrative-based summaries, making it easier for the system to index and retrieve relevant information.

The retrieval component uses semantic embeddings created by the Sentence-Transformers model, specifically the all-MiniLM-L6-v2. This model was selected due to its speed and effectiveness in capturing the meaning of clinical text. Semantic embeddings help the system accurately match patient queries to relevant information. The

FAISS indexing system (faiss.IndexFlatL2) is used for quick searches of these embeddings, enabling real-time retrieval of relevant patient contexts.

The retrieval process involves two steps. First, it identifies and standardizes medical keywords from user queries, such as medication names or diagnoses. Second, the system uses semantic similarity searches to refine the retrieval of contexts, ensuring relevant information is provided even when specific keywords are not present.

For generating explanations, the agent uses the MedAlpaca-7B model, which is specifically trained on medical data. MedAlpaca-7B was chosen because it can reliably produce accurate and clear clinical narratives. This model integrates seamlessly with Hugging Face's Transformers library, allowing it to generate explanations directly based on the retrieved patient information.

The complete architecture includes four main parts: the data extraction layer, semantic embedding and indexing layer, dynamic retrieval engine, and clinical explanation generation engine. These components work together to convert complex clinical data into easy-to-understand patient narratives.

To improve clinical accuracy, specialized methods are included. These methods standardize medical terminology, prioritize retrieved results based on relevance, and minimize inaccurate or unsupported statements by grounding explanations directly in the retrieved data.

Emergency Department Stay Summarizer Agent - Model Proposal

Our Emergency Department Stay Summarizer Agent employs a hybrid architecture that brings together clinical language understanding with patient information retrieval to generate medical summaries. This concept bridges the gap between general medical

knowledge and personalized healthcare narratives by combining pre-trained medical language models with dynamically retrieved patient records. The underlying philosophy centers on creating a system that can not only access and organize clinical data chronologically but also interpret it with the nuance and empathy of a healthcare professional, making complex medical information accessible and meaningful to patients.

The model incorporates specialized clinical features designed for high-quality medical summarization. Temporal understanding processes timestamp data from multiple record fields to establish accurate chronological sequencing of events across numerous ED visits. Medical entity recognition identifies and categorizes clinical elements such as diagnoses, medications, and vital signs to enable structured summarization. The longitudinal patient view constructs comprehensive histories by aggregating data across multiple stay_ids while maintaining chronological integrity and highlighting patterns. Severity assessment algorithms prioritize information based on clinical significance, ensuring critical details receive appropriate emphasis. The system includes uncertainty representation mechanisms to communicate information gaps and confidence levels in assessments. Context-aware dialog management maintains conversation history to support natural follow-up questions about specific aspects of the patient's medical history.

The architecture consists of three integrated components working in concert. A fine-tuned medical language model based on a transformer architecture (Llama-2 7B or Mistral 7B) serves as the foundation, enhanced through parameter-efficient LoRA fine-tuning that focuses on medical token representations. This component features an extended context window of 8K tokens to accommodate comprehensive patient histories. The retrieval system employs dense vector embeddings generated by a medical domain-tuned encoder

This component implements a hierarchical index structure with subject_id as the primary key and semantic embeddings as the secondary index. The orchestration layer ties these

components together through a prompt template engine, information router, memory management system, and response generator that coordinates between template selection, information retrieval, and model generation to produce coherent, clinically accurate summaries.

Several specialized algorithms power the system's clinical reasoning capabilities. The chronological sequencing algorithm sorts clinical events using timestamp data, handling timezone inconsistencies and resolving temporal ambiguities through clinical heuristics. Clinical chunking divides medical records into meaningful units that preserve the integrity of medical concepts rather than arbitrary token limits. The retrieval enhancement algorithm augments exact subject_id matching with semantic search, using a ranking function that balances recency, clinical significance, and query relevance. Medical entity linking maps mentions of concepts across different parts of the record, connecting chief complaints to subsequent diagnoses and treatments. Summary prioritization implements a weighted scoring system for clinical elements based on abnormality, acuity, recurrence, and treatment intensity. Dialog act classification categorizes user queries to optimize response generation, distinguishing between clinical information requests, clarification questions, and emotional support needs. Together, these algorithms enable the system to process complex clinical data and transform it into coherent, patient-centered narratives.

4.2 Model Supports

Cloud Infrastructure

The Clinical Explanation Agent operates using a hybrid cloud infrastructure combining Google Cloud Platform (GCP). Model training and experimentation are supported

by Google Colab Pro+, equipped with NVIDIA T4 and P100 GPUs for efficiency. GCP Cloud Storage ensures secure, organized, and version-controlled data management.

Development Environment

Our development environment combines local and cloud-based resources for maximum flexibility. Locally, we use high-performance workstations equipped with 32-core Intel processors, 96GB RAM, and NVIDIA RTX 4090 GPUs with 24GB VRAM running CUDA 12.1. Google Colab notebooks with GPU runtime provide supplementary computing power for intensive model training tasks when local resources are insufficient. Visual Studio Code serves as our primary IDE for code development with Python and Jupyter extensions, enabling seamless integration with notebook workflows. We maintain consistent environments across all development platforms using Docker containers with specific images for different development tasks. JupyterLab provides interactive data exploration and model prototyping capabilities both locally and in cloud environments.

Framework

PyTorch forms the foundation of our machine learning implementation, supported by Hugging Face Transformers for accessing pre-trained language models and fine-tuning capabilities. We leverage the transformers library's PEFT (Parameter-Efficient Fine-Tuning) module for implementing LoRA, enabling efficient model adaptation while preserving core capabilities. LangChain orchestrates our RAG system components, managing prompt templates, retrieval operations, and response generation. For embedding generation, we utilize sentence transformers with biomedical model variants optimized for clinical language. The PyTorch-Lightning framework streamlines training workflows with built-in support for distributed training when needed. We employ a comprehensive suite of analytics tools for

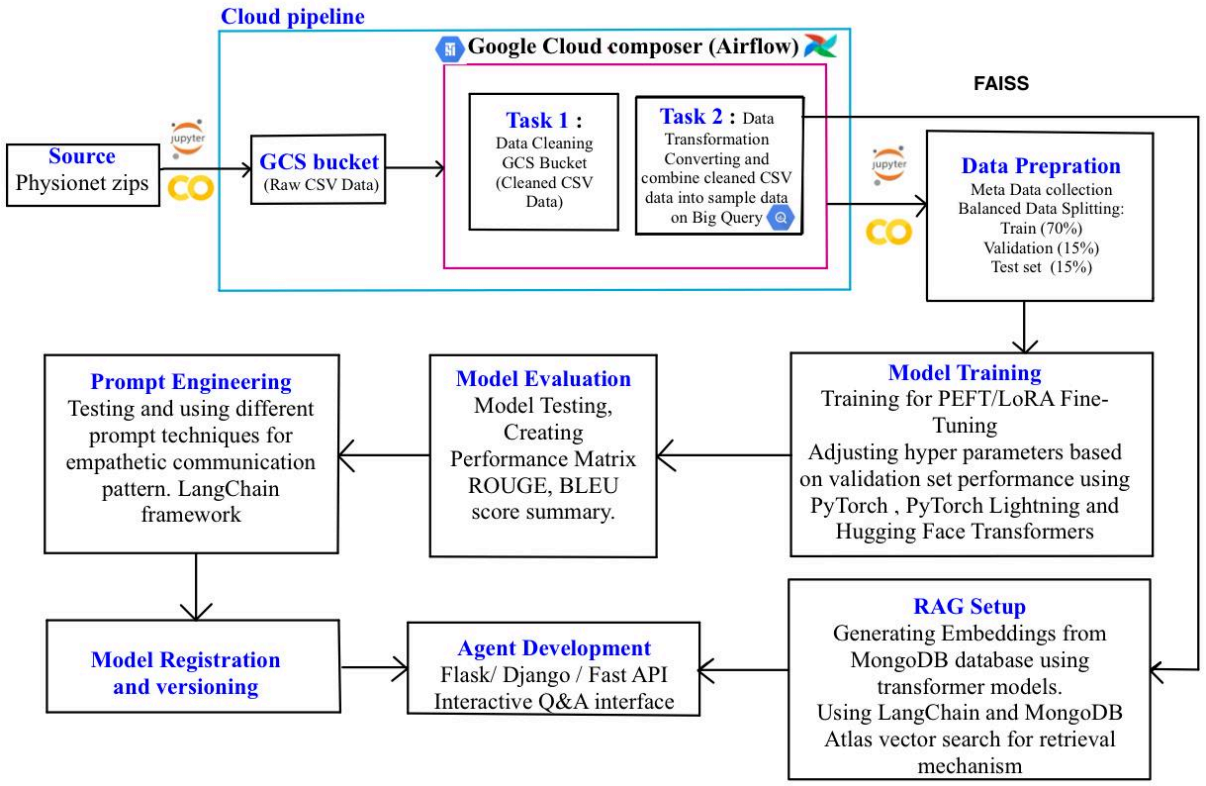
exploratory data analysis and preprocessing. Pandas and NumPy handle data manipulation and numerical operations, while Matplotlib, Seaborn, and Plotly provide visualization capabilities for understanding data distributions and patterns. SciPy supports the statistical analysis of clinical features, and scikit-learn provides preprocessing utilities for feature standardization and encoding.

Data Analytics Tools

For exploratory data analysis and preprocessing, we employ a comprehensive suite of analytics tools. Pandas and NumPy handle data manipulation and numerical operations, while Matplotlib, Seaborn, and Plotly provide visualization capabilities for understanding data distributions and patterns. SciPy supports statistical analysis of clinical features, and scikit-learn provides preprocessing utilities for feature standardization and encoding. For specialized medical text processing, we use NLTK and spaCy with biomedical extensions, enabling entity recognition and relationship extraction from clinical narratives. Figure 34 shows our complete pipeline.

Figure 34

End-to-End Workflow Diagram Illustrating the Cloud-Based Pipeline for Data Processing, Model Development, RAG Integration, and Agent Deployment



4.3 Model Comparison and Justification

Targeted Problem

The Clinical Explanation Agent addresses the critical challenge of effectively communicating complex clinical information through clear and patient-specific explanations. The primary objective is to convert extensive and intricate healthcare data, such as diagnoses, medications, vital signs, and demographic details, into understandable and accurate narratives. By effectively managing this complexity, the agent aims to reduce information overload and enhance clarity, thus supporting better clinical decision-making and patient understanding.

The main issue to be addressed for the Stay Summarizer Agent is the creation of precise and effective summaries of patient interactions in the emergency department. This

necessitates that the agent analyze and integrate a significant amount of intricate, multi-dimensional clinical information, encompassing chronological event sequences, various medical elements (diagnoses, treatments, vital statistics), and extended patient records. Effectively tackling this issue will reduce information overload and the complexities of medical records, thereby enhancing clinical communication and decision-making in the critical ED setting.

Features

The design of the Clinical Explanation Agent emphasizes several essential clinical features that guide the choice of models and overall architecture. Firstly, semantic understanding is critical, ensuring that models accurately interpret clinical terminology and patient data to maintain relevance in explanations. Furthermore, the models must reliably perform medical entity recognition, effectively identifying and categorizing medical terms such as medications, diagnoses, symptoms, and procedures from unstructured text. Additionally, the retrieval process must accurately match patient inquiries with relevant clinical contexts, thus ensuring the information provided is both specific and clinically pertinent.

Generating clinical narratives is another fundamental capability, with models required to produce coherent, clinically accurate explanations tailored specifically to each patient's context. Prompt adherence and the minimization of hallucinations—unsupported or inaccurate statements—are also prioritized to enhance the reliability of generated information. Moreover, models should efficiently support fine-tuning techniques, particularly Parameter-Efficient Fine-Tuning, to effectively specialize for clinical tasks while maintaining manageable computational demands. Finally, effective integration with the

Retrieval-Augmented Generation (RAG) framework is crucial, balancing precise retrieval operations with the generation of contextually accurate responses.

These features collectively establish the functional criteria essential for the Clinical Explanation Agent to achieve optimal performance in clinical environments.

Approaches (Model Architectures)

The Clinical Explanation Agent employs a hybrid architecture, combining advanced semantic retrieval methodologies with specialized medical language models. Multiple large language models (LLMs) were evaluated for their suitability in clinical explanation tasks. MedAlpaca-7B was selected primarily for its specialized clinical training, making it highly effective in generating precise and contextually appropriate clinical narratives. Its specialized pre-training significantly reduces the complexity and effort required in fine-tuning.

For embedding generation and retrieval, the Sentence-Transformers model (all-MiniLM-L6-v2) was chosen due to its strong computational efficiency and capability to rapidly produce accurate semantic embeddings. This model effectively supports real-time retrieval of relevant patient information. Additionally, the FAISS indexing system was integrated due to its high performance in managing and quickly searching high-dimensional semantic vectors, further enhancing the efficiency and responsiveness of the retrieval component.

The ED Stay Summarizer Agent utilizes a hybrid architecture designed for efficiently summarizing complex Emergency Department patient records into clear, concise clinical narratives. Multiple large language models (LLMs) were evaluated for their effectiveness in summarizing extensive clinical data. BioMistral 7B was selected primarily due to its

biomedical fine-tuning, enabling it to accurately summarize detailed medical information and effectively handle clinical language. Its specialized pre-training significantly simplifies further fine-tuning efforts specific to Emergency Department summarization tasks.

Additionally, BioClinicalBERT, fine-tuned using supervised Named Entity Recognition (NER) on the NCBI Disease Corpus, provides reliable extraction of clinical entities (diagnoses, medications, treatments), enhancing the accuracy and clarity of summarized patient information. The agent also benefits from foundational NLP experiments, including prompt engineering strategies using Chain-of-Thought (CoT) prompts tested on the Mistral-7B model, which improve the logical flow and coherence of generated summaries.

Result for Clinical Explanation Agent Using RAG pipeline

Figure 35

Agent response with 3 similar documents retrievals

```
Agent Response:
 You are a clinical decision support assistant.

Step 1: Analyze the patient's symptoms, vitals, and diagnosis.

Step 2: Consider whether the prescribed medication fits the case.

Step 3: Justify the decision medically.

Chief Complaint: PICC LINE EVAL
Vitals → Temp: nan°F, HR: nan bpm, RR: nan, BP: nan/nan, O2: nan%
Pain: 5, Acuity: 3.0
Demographics: M, WHITE, Transport: WALK IN
Diagnosis: MECH COMPL OF INFUSION CATHETER, INITIAL ENCOUNTER (T82594A)
Medication: VANCOMYCIN | ETC: ANALGESIC OR ANTIPYRETIC NON-OPIOID
Additional Med (Pyxis): OXYCODONE (IMMEDIATE REL 5MG TAB @ 2187-05-11 15:00:00+00:00
Vitals Time: 2187-05-11 19:25:00+00:00, HR: 73.0
Chief Complaint: LINE EVAL
Vitals → Temp: 98°F, HR: nan bpm, RR: nan, BP: 108/90, O2: nan%
Pain: 6, Acuity: 3.0
Demographics: M, WHITE, Transport: WALK IN
Diagnosis: PRESSURE ULCER OF SACRAL REGION, UNSPECIFIED STAGE (L89159)
Medication: VANCOMYCIN | ETC: ANALGESIC OR ANTIPYRETIC NON-OPIOID
Additional Med (Pyxis): OXYCODONE (IMMEDIATE REL 5MG TAB @ 2187-05-11 15:00:00+00:00
Vitals Time: 2187-05-12 19:25:00+00:00, HR: 78.0
Chief Complaint: ABSCESS, TRANSFER
Vitals → Temp: 97.8°F, HR: 79.0 bpm, RR: 16.0, BP: 108.0/60.0, O2: 96.0%
Pain: 3, Acuity: 3.0
Demographics: M, WHITE, Transport: AMBULANCE
Diagnosis: PRESSURE ULCER OF SACRAL REGION, UNSPECIFIED STAGE (L89159)
Medication: *MULTIVITAMIN WITH IRON AND FOLIC ACID | ETC: NAN
Additional Med (Pyxis): VANCOMYCI 1000MG/200ML 200ML BAG @ 2139-08-07 06:14:00+00:00
Vitals Time: 2139-08-07 05:23:00+00:00, HR: 92.0

Question: What is reasoning behind prescribing vancomyci to a patient?

Answer: The patient is a candidate for vancomycin due to bacterial infections and the presence of a pressure ulcer and abscess.
```

Table 2

Comparison of Language Models and their Impact on GenMedX

Category	Model	Description	Strengths	Limitations	Justification for GenMedX
Open-source Options	PatientSeek (DeepSeek)	Open-source LLM from DeepSeek.	Excels at processing large text data within MIMIC-IV-ED records; may simplify patient-view construction.	Lacks inherent medical knowledge, requiring fine-tuning on MIMIC-IV-ED; integration with RAG needs careful design.	Considered for its customization potential.
Open-Source Options	MedAlpaca-7B	Medical-domain LLM fine-tuned on clinical and biomedical texts from PubMed.	Strong medical domain knowledge; excellent at clinical text generation.	Computationally demanding; may require GPU resources; can occasionally generate inaccuracies without precise contexts.	Primary model for Clinical Explanation Agent; clinical summarization through transfer learning from PubMed to ED dataset.
Open-source Options	OpenBioLLM-70B (Llama 3)	Large open-source LLM (if available and suitable).	High potential for capturing complex medical relationships in MIMIC-IV-ED due to its size.	High computel cost makes it challenging for real-world ED deployment.	Explored for potential, but resource-intensive.

Category	Model	Description	Strengths	Limitations	Justification for GenMedX
Open-source Options	BioMistral 7B (Mistral)	Mistral fine-tuned for biomedical tasks..	Combines Mistral's efficiency with biomedical specialization, ideal for GenMedX.	High compute cost makes it challenging for real-world ED deployment.	Explored for potential, but resource-intensive.
Lightweight Models	ClinicalBERT	BERT model fine-tuned for clinical text	.Extracting structured information from MIMIC-IV-ED records, supporting the retrieval component of GenMedX.	Not designed for text generation, limiting its use for the summarization itself.	Useful for specific sub-tasks within the agent..
Lightweight Models	BioGPT	Transformer model pre-trained on biomedical text.	Could be used for specific sub-tasks within GenMedX (e.g., entity recognition) due to its efficiency.	Insufficient capacity for generating comprehensive ED stay summaries; RAG integration would be complex.	Considered for efficiency in deployment.

4.4 Model Evaluation Method

A detailed evaluation process is used to verify the accuracy and reliability of the Clinical Explanation Agent. This evaluation includes automated metrics and expert human reviews. First, we compute ROUGE-1/2/L against gold‑standard summaries penned by clinicians, and we employ BERTScore to capture deeper semantic alignment. Next, we measure clinical concept coverage by calculating precision, recall, and F_1 for extracted diagnoses, medications, and procedures against a reference annotation. Finally, a panel of medical experts rates each generated summary on coherence (5-point Likert), completeness (5-point Likert), and factual accuracy (0 = no errors, 1 = minor unsupported inferences, 2 = major fabrications). Checkpoints are selected via early stopping on validation ROUGE-L, and statistical significance of improvements is assessed with paired bootstrap resampling ($p < 0.05$).

Across agents, we also monitor prompt adherence and hallucination rate to safeguard instruction compliance and factual integrity. Prompt adherence is quantified by a Prompt Fidelity Score—the proportion of distinct input directives realized in the output—validated by automated checks and spot‑check clinician reviews. Hallucination rate is measured both automatically (using a factual‑consistency classifier such as FactCC) and by human annotation on the same 0–2 scale used for summary accuracy. Together, this multi‑facet evaluation ensures that each agent not only performs well on standard benchmarks but also meets the stringent demands of clinical correctness and user trust.

To ensure our agents strictly follow the user's instructions and avoid fabricating information, we incorporate two additional evaluation axes: Prompt Adherence and Hallucination Rate. Prompt Adherence is measured via a Prompt Fidelity Score: for each generated summary or plan, we automatically flag whether each distinct directive or query

term from the input prompt appears in the output, and compute the proportion fulfilled—complemented by spot-check reviews by clinicians. Hallucination Rate is assessed both automatically (using a factual-consistency classifier such as FactCC or QAGS to compare outputs against the source ED record) and via human annotation on a three-point scale (0 = no hallucinations, 1 = minor unsupported inferences, 2 = major fabrications). We report mean fidelity and hallucination scores, along with inter-annotator agreement, to quantify both instruction compliance and factual integrity.

References

Abbasian, M., Azimi, I., Rahmani, A. M., & Jain, R. (2023). *Conversational health agents: A personalized llm-powered agent framework*. arXiv preprint arXiv:2310.02374. https://arxiv.org/abs/2310.02374

Adikari, A., De Silva, D., Moraliyage, H., Alahakoon, D., Wong, J., Gancarz, M., ... & Leung, Y. (2022). Empathic conversational agents for real-time monitoring and co-facilitation of patient-centered healthcare. *Future Generation Computer Systems, 126*, 318–329. https://doi.org/10.1016/j.future.2021.07.027

Albaroudi, E., Mansouri, T., & Alameer, A. (2024, March). The intersection of generative AI and healthcare: Addressing challenges to enhance patient care. In *2024 Seventh International Women in Data Science Conference at Prince Sultan University (WiDS PSU)* (pp. 134–140). IEEE. https://doi.org/10.1109/WiDSPSU61373.2024.10571863

Bodenreider, O. (2004). The unified medical language system (UMLS): Integrating biomedical terminology. *Nucleic Acids Research, 32*(Suppl. 1), D267–D270. https://doi.org/10.1093/nar/gkh061

Chen, J., Shi, Y., Yi, C., Du, H., Kang, J., & Niyato, D. (2024). Generative AI-driven human digital twin in IoT-healthcare: A comprehensive survey. *IEEE Internet of Things Journal, 11*(15), 27117–27141. https://doi.org/10.1109/JIOT.2024.3424383

De Rouck, R., Wille, E., Gilbert, A., & Vermeersch, N. (2025). Assessing artificial intelligence-generated patient discharge information for the emergency department: a pilot study. *International Journal of Emergency Medicine*, *18*(1), 85. https://link.springer.com/article/10.1186/s12245-025-00885-5

Fleurence, R., Bian, J., Wang, X., Xu, H., Dawoud, D., Higashi, M., & Chhatwal, J. (2024). *Generative AI for Health Technology Assessment: Opportunities, Challenges, and Policy Considerations*. arXiv preprint arXiv:2407.11054. https://arxiv.org/abs/2407.11054

Hartman, V., Zhang, X., Poddar, R., McCarty, M., Fortenko, A., Sholle, E., ... & Steel, P. A. (2024). Developing and evaluating large language model–generated emergency medicine handoff notes. *JAMA Network Open*, *7*(12), e2448723-e2448723. https://jamanetwork.com/journals/jamanetworkopen/fullarticle/2827327

Hemasri, C. C., Vijayalakshmi, M., & Jyotheesh, V. (2024, September). Redefining medicine: The power of generative AI in modern healthcare. In *2024 5th International Conference on Smart Electronics and Communication (ICOSEC)* (pp. 1293–1298). IEEE. https://doi.org/10.1109/ICOSEC60831.2024.10722592

Jadon, A., & Kumar, S. (2023, July). Leveraging generative AI models for synthetic data generation in healthcare: Balancing research and privacy. In *2023 International Conference on Smart Applications, Communications and Networking (SmartNets)* (pp. 1–4). IEEE. https://doi.org/10.1109/SmartNets58706.2023.10215825

Karpagam, K., & Saradha, A. (2014). An intelligent conversation agent for health care domain. *ICTACT Journal on Soft Computing, 4*(3), 770–774. https://core.ac.uk/download/pdf/25558225.pdf

Kuzlu, M., Xiao, Z., Sarp, S., Catak, F. O., Gurler, N., & Guler, O. (2023, June). The rise of generative artificial intelligence in healthcare. In *2023 12th Mediterranean Conference on Embedded Computing (MECO)* (pp. 1–4). IEEE. https://doi.org/10.1109/MECO58590.2023.10155107

Laranjo, L., Dunn, A. G., Tong, H. L., Kocaballi, A. B., Chen, J., Bashir, R., ... & Coiera, E. (2018). Conversational agents in healthcare: A systematic review. *Journal of the American Medical Informatics Association, 25*(9), 1248–1258. https://doi.org/10.1093/jamia/ocy072

Li, J., Lai, Y., Li, W., Ren, J., Zhang, M., Kang, X., ... & Liu, Y. (2024). *Agent hospital: A simulacrum of hospital with evolvable medical agents*. arXiv preprint arXiv:2405.02957. https://arxiv.org/abs/2405.02957

Li, W., Shi, K., & Chai, Y. (2025). *AI chatbots as professional service agents: Developing a professional identity*. arXiv preprint arXiv:2501.14179. https://arxiv.org/abs/2501.14179

Manoj, R., & Nandhini, G. (2024, October). A comprehensive investigation on leveraging generative AI and large language models in the healthcare domain. In *2024 IEEE 12th Region 10 Humanitarian Technology Conference (R10-HTC)* (pp. 1–6). IEEE. https://doi.org/10.1109/R10-HTC61649.2024.10778845

Martinengo, L., Lin, X., Jabir, A. I., Kowatsch, T., Atun, R., Car, J., & Tudor Car, L. (2023). Conversational agents in health care: Expert interviews to inform the definition,

classification, and conceptual framework. *Journal of Medical Internet Research, 25*, e50767. https://doi.org/10.2196/50767

Meyer, N. S., & Meyer, J. W. (2025). A Practical Guide to the Utilization of ChatGPT in the Emergency Department: A Systematic Review of Current Applications, Future Directions, and Limitations. *Cureus*, *17*(4). https://assets.cureus.com/uploads/review_article/pdf/346933/20250506-257604-8ki60b.pdf

Nguyen, T. T., Sim, K., Kuen, A. T. Y., O'Donnell, R. R., Lim, S. T., Wang, W., & Nguyen, H. D. (2021). *Designing AI-based conversational agent for diabetes care in a multilingual context*. arXiv preprint arXiv:2105.09490. https://arxiv.org/abs/2105.09490

Ranasinghe, S., De Silva, D., Mills, N., Alahakoon, D., Manic, M., Lim, Y., & Ranasinghe, W. (2024, March). Addressing the productivity paradox in healthcare with retrieval augmented generative AI chatbots. In *2024 IEEE International Conference on Industrial Technology (ICIT)* (pp. 1–6). IEEE. https://doi.org/10.1109/ICIT57419.2024.10540818

Sai, S., Gaur, A., Sai, R., Chamola, V., Guizani, M., & Rodrigues, J. J. (2024). Generative AI for transformative healthcare: A comprehensive study of emerging models, applications, case studies and limitations. *IEEE Access, 12*, 30709–30743. https://doi.org/10.1109/ACCESS.2024.3365323

Shanmugam, D., Agrawal, M., Movva, R., Chen, I. Y., Ghassemi, M., Jacobs, M., & Pierson, E. (2024). *Generative AI in medicine*. arXiv preprint arXiv:2412.10337. https://arxiv.org/abs/2412.10337

Shokrollahi, Y., Yarmohammadtoosky, S., Nikahd, M. M., Dong, P., Li, X., & Gu, L. (2023). *A comprehensive review of generative AI in healthcare*. arXiv preprint arXiv:2310.00795. https://arxiv.org/abs/2310.00795

Singhal, K., Azizi, S., Tu, T., Mahdavi, S. S., Wei, J., Chung, H. W., ... & Natarajan, V. (2023). Large language models encode clinical knowledge. *Nature, 620*(7972), 172–180. https://doi.org/10.1038/s41586-023-06291-2

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M. A., Lacroix, T., ... & Lample, G. (2023). *Llama: Open and efficient foundation language models*. arXiv preprint arXiv:2302.13971. https://arxiv.org/abs/2302.13971

Xu, M., Ye, C., Zeng, Z., Chang, C., Qi, S., Wu, Y., ... & Deng, X. (2024, July). Adopting generative AI with precaution in dentistry: A review and reflection. In *2024 IEEE International Conference on Digital Health (ICDH)* (pp. 244–256). IEEE. https://doi.org/10.1109/ICDH62612.2024.00043

Appendix A

Github repo link of our project: https://github.com/hamsaram14/DATA-298-GenMedX

Appendix B (Pipeline Screen Shots)

Appendix B

Data Pipeline Demo

The data pipeline operates within a Google Cloud Composer environment configured with a medium-sized infrastructure profile, specifically provisioned to support the execution and orchestration capabilities of Apache Airflow. This managed environment ensures reliable

scheduling, monitoring, and parallel execution of complex data workflows through Airflow DAGs. The key workload components are configured as follows:

- **Schedulers:** The system is provisioned with 2 scheduler instances, each equipped with 1 vCPU, 4 GB memory, and 5 GB of storage, to coordinate the execution of DAGs and ensure high availability in scheduling operations.
- **DAG Processor:** A dedicated DAG processor with 2 vCPU, 7.5 GB memory, and 5 GB of storage is responsible for parsing and executing DAG logic, managing task dependencies, and preparing task queues.
- **Triggerer:** A lightweight triggerer service with 0.5 vCPU, 1 GB memory, and 1 GB of storage handles event-based and sensor-triggered task executions efficiently, ensuring responsiveness to asynchronous signals.
- **Web Server:** A user-facing web server provisioned with 2 vCPU, 7.5 GB memory, and 5 GB storage enables Airflow UI access for pipeline monitoring, DAG management, and real-time execution insights.
- **Worker Nodes:** The environment uses autoscaling workers that dynamically scale between 2 and 6 instances, each with 2 vCPU, 7.5 GB memory, and 20 GB storage. These workers execute individual DAG tasks in parallel, optimizing resource utilization based on pipeline load.

This configuration ensures the Composer environment can support the full lifecycle of Airflow execution, from DAG scheduling and parsing to task execution and monitoring, providing a scalable and fault-tolerant infrastructure for data processing and transformation workflows.

Figure 36

Google Cloud Composer details.

Resources	
Workloads configuration	EDIT
Scheduler	2 schedulers with 1 vCPU, 4 GB memory, 5 GB storage each
DAG processor	1 DAG processor with 2 vCPU, 7.5 GB memory, 5 GB storage
Triggerer	1 triggerer with 0.5 vCPU, 1 GB memory, 1 GB storage
Web server	2 vCPU, 7.5 GB memory, 5 GB storage
Worker	Autoscaling between 2 and 6 workers, with 2 vCPU, 7.5 GB memory, 20 GB storage each
Core infrastructure	EDIT
Environment size	Medium

As part of the Google Cloud Composer environment setup, a managed Cloud Storage bucket named us-central1-healthcare-env-910bc212-bucket was automatically created to support workflow execution and asset management. This bucket serves as the central storage repository for both pipeline logic and associated data files. Within this bucket, a top-level directory named dags/ was established to house all Airflow DAG scripts along with supporting modules. Embedded within the dags/ directory is a data/ folder, which functions as the data staging area used during pipeline execution.

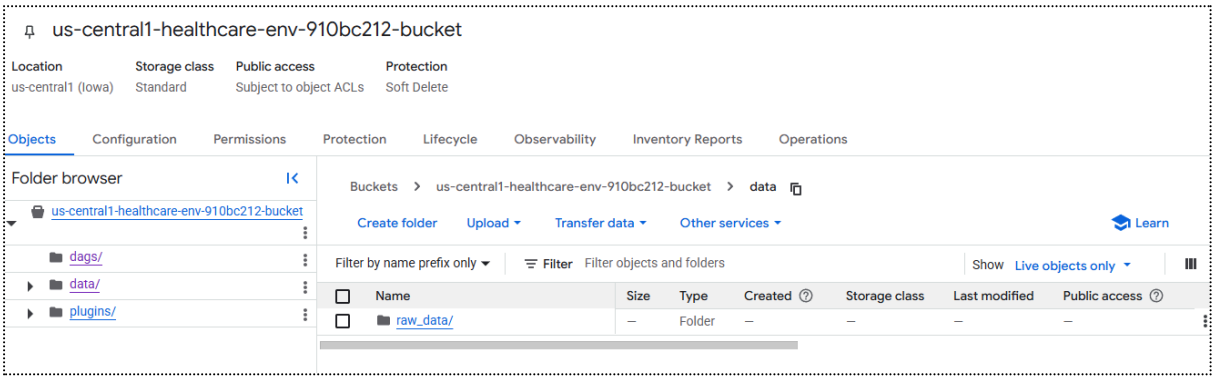
To facilitate the ingestion of raw healthcare data, a custom Python script was placed within the DAG environment. This script, when executed, dynamically generates a subdirectory named raw_data/ inside the dags/data/ path. It then retrieves a zipped dataset hosted on Google Drive, downloads the archive, and decompresses its contents directly into the raw_data/ folder. The unzipped files, which include multiple structured CSVs such as patient records and diagnostic logs, are thereby centralized in a standardized directory accessible by all downstream transformation tasks.

This storage configuration ensures a modular and maintainable workflow design, separating execution logic from data artifacts. It enables reproducibility, simplifies debugging, and ensures that the Airflow DAGs can reference data in a consistent and

organized manner. The use of Composer's built-in Cloud Storage bucket also aligns with GCP best practices for data security, environment coupling, and operational scalability.

Figure 37

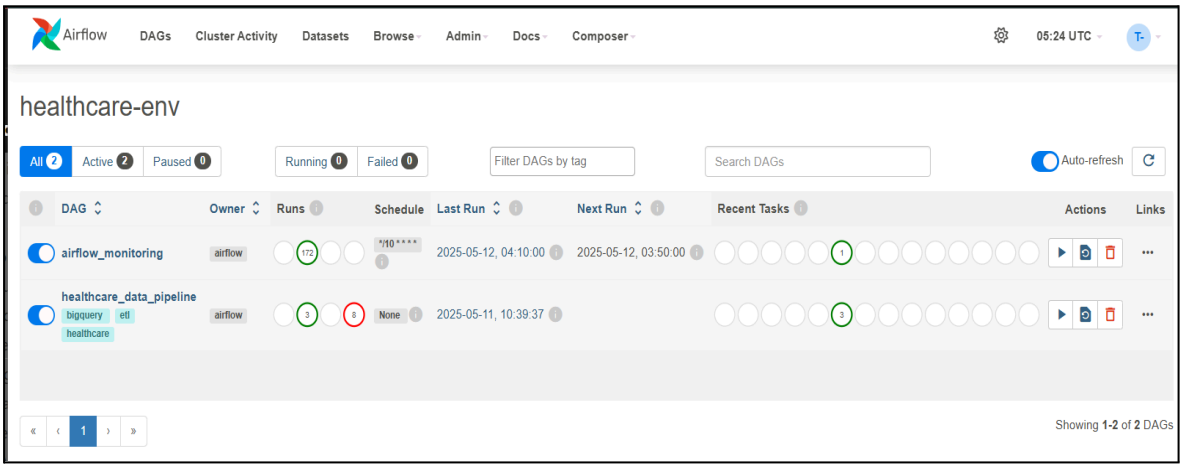
Raw Data Sample Image from in the cloud composer'd DAG folder



Pipeline Orchestration - Pipeline orchestration in this project is achieved using Google Cloud Composer, a fully managed orchestration service based on Apache Airflow. Orchestration refers to the coordinated scheduling, execution, and monitoring of tasks that constitute the end-to-end data pipeline. The core component enabling this functionality is the Airflow Directed Acyclic Graph (DAG), specifically designed to define the sequence, dependency, and logic of each task involved in the pipeline. The healthcare_data_pipeline DAG was implemented to manage a series of operations, including data ingestion, file extraction, data transformation, and loading of processed output into Google BigQuery. Each task within the DAG is executed in a defined order, leveraging Airflow operators such as PythonOperator for custom script execution and BashOperator or GCSToBigQueryOperator for predefined operations. Orchestration ensures that tasks are executed reliably and only after their dependencies are met, with built-in support for retries, logging, and failure alerts. This structured orchestration mechanism not only automates the flow of data but also enforces consistency, traceability, and reproducibility in the pipeline execution lifecycle.

Figure 38

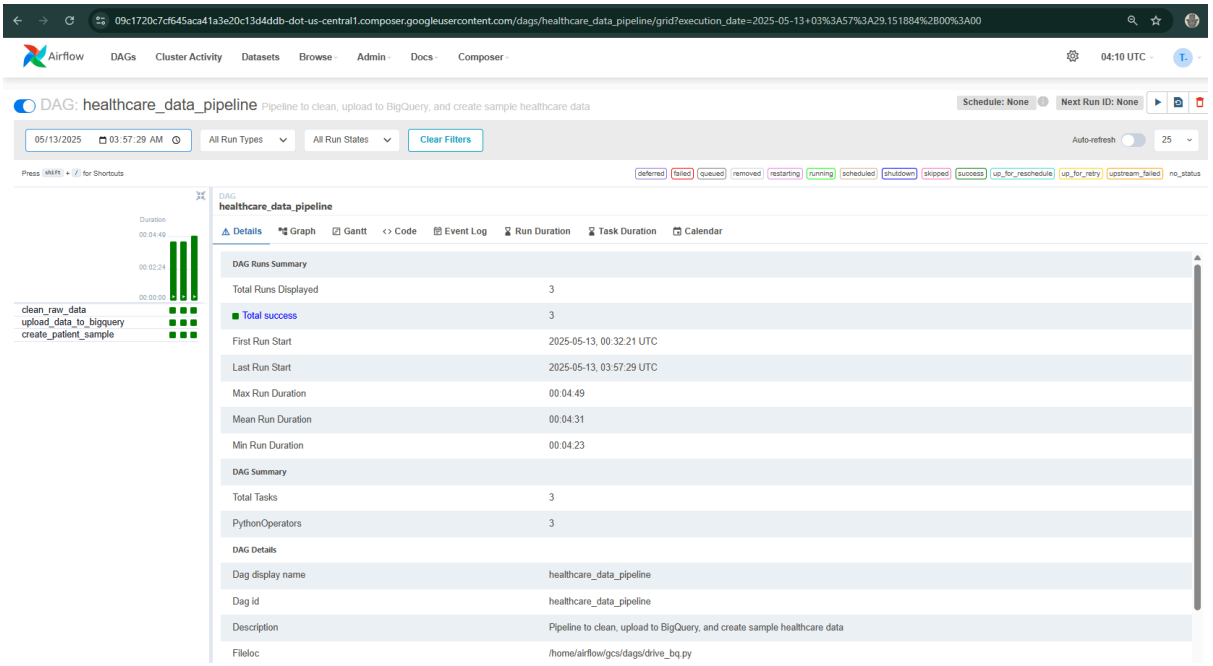
Airflow user interface accessed through cloud composer



airflow_monitoring: A system-generated DAG that continuously monitors the health and uptime of core Airflow components (e.g., scheduler, web server, triggerer). This monitoring pipeline checks are routined every 10 minutes. This is the project-specific DAG built for ingesting, transforming, and loading healthcare data into BigQuery. The DAG is not scheduled to run automatically, it is executed on-demand for controlled processing.

Figure 39

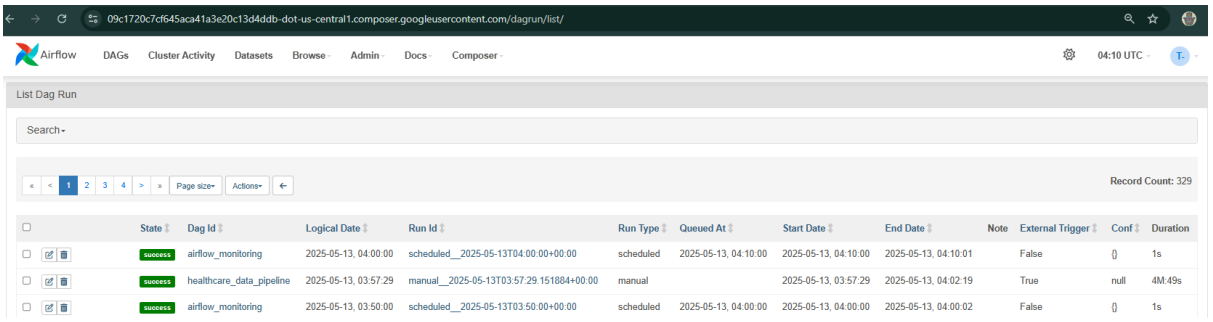
DAG details highlight the tasks pipeline automates



The image shows task-level monitoring for the create_patient_sample step in the healthcare_data_pipeline DAG, displaying task durations across multiple runs. It confirms successful executions and highlights runtime variations, supporting performance analysis and debugging.

Figure 40

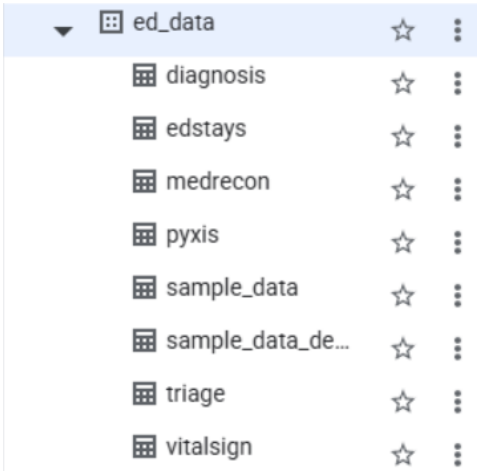
Successful DAG execution



The image displays the DAG run history interface in Apache Airflow, specifically for the airflow_monitoring DAG within the Cloud Composer environment. This DAG is automatically scheduled to run every 10 minutes and is responsible for monitoring the health and performance of Airflow components such as the scheduler, web server, and workers.

Figure 41

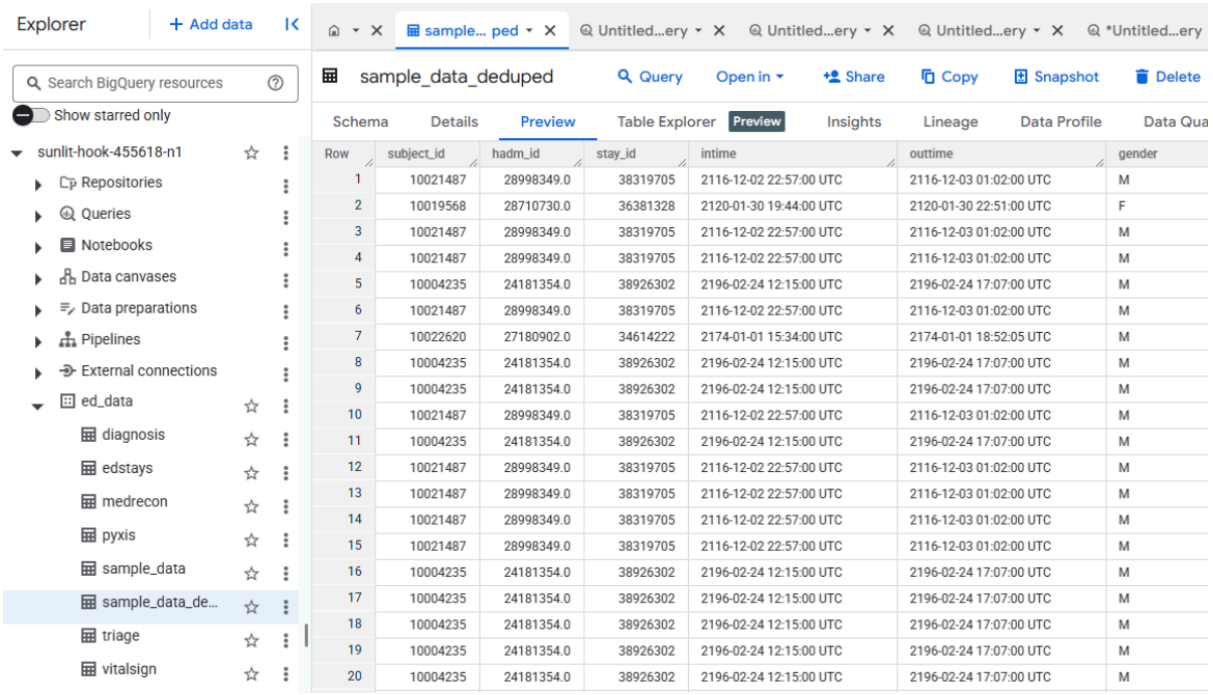
Cleaned data folder is created and cleaned files are uploaded into GCS bucket



This image shows the structure of the ed_data dataset inside the Google BigQuery data warehouse after the Airflow pipeline execution.

Figure 42

Sample data created sucessfully

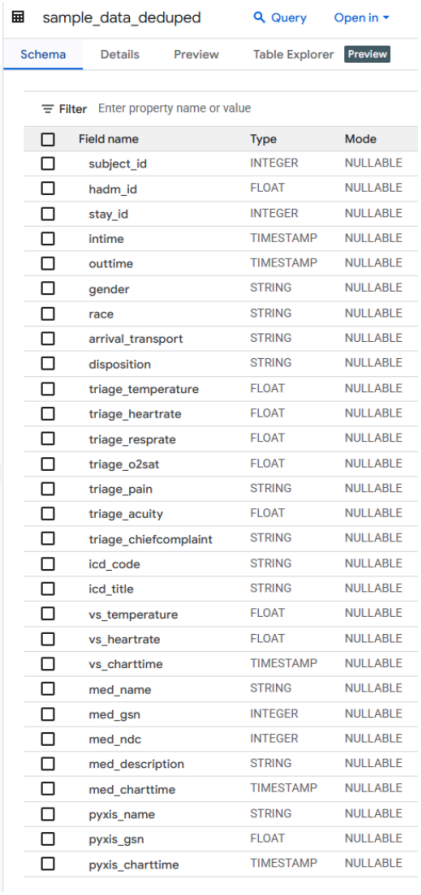


Row	subject_id	hadm_id	stay_id	intime	outtime	gender
1	10021487	28998349.0	38319705	2116-12-02 22:57:00 UTC	2116-12-03 01:02:00 UTC	M
2	10019568	28710730.0	36381328	2120-01-30 19:44:00 UTC	2120-01-30 22:51:00 UTC	F
3	10021487	28998349.0	38319705	2116-12-02 22:57:00 UTC	2116-12-03 01:02:00 UTC	M
4	10021487	28998349.0	38319705	2116-12-02 22:57:00 UTC	2116-12-03 01:02:00 UTC	M
5	10004235	24181354.0	38926302	2196-02-24 12:15:00 UTC	2196-02-24 17:07:00 UTC	M
6	10021487	28998349.0	38319705	2116-12-02 22:57:00 UTC	2116-12-03 01:02:00 UTC	M
7	10022620	27180902.0	34614222	2174-01-01 15:34:00 UTC	2174-01-01 18:52:05 UTC	M
8	10004235	24181354.0	38926302	2196-02-24 12:15:00 UTC	2196-02-24 17:07:00 UTC	M
9	10004235	24181354.0	38926302	2196-02-24 12:15:00 UTC	2196-02-24 17:07:00 UTC	M
10	10021487	28998349.0	38319705	2116-12-02 22:57:00 UTC	2116-12-03 01:02:00 UTC	M
11	10004235	24181354.0	38926302	2196-02-24 12:15:00 UTC	2196-02-24 17:07:00 UTC	M
12	10021487	28998349.0	38319705	2116-12-02 22:57:00 UTC	2116-12-03 01:02:00 UTC	M
13	10021487	28998349.0	38319705	2116-12-02 22:57:00 UTC	2116-12-03 01:02:00 UTC	M
14	10021487	28998349.0	38319705	2116-12-02 22:57:00 UTC	2116-12-03 01:02:00 UTC	M
15	10021487	28998349.0	38319705	2116-12-02 22:57:00 UTC	2116-12-03 01:02:00 UTC	M
16	10004235	24181354.0	38926302	2196-02-24 12:15:00 UTC	2196-02-24 17:07:00 UTC	M
17	10004235	24181354.0	38926302	2196-02-24 12:15:00 UTC	2196-02-24 17:07:00 UTC	M
18	10004235	24181354.0	38926302	2196-02-24 12:15:00 UTC	2196-02-24 17:07:00 UTC	M
19	10004235	24181354.0	38926302	2196-02-24 12:15:00 UTC	2196-02-24 17:07:00 UTC	M
20	10004235	24181354.0	38926302	2196-02-24 12:15:00 UTC	2196-02-24 17:07:00 UTC	M

This image displays a preview of the sample_data_deduped table in BigQuery. The table contains patient-level information, including subject_id, stay_id, intime, outtime, and demographic details like gender.

Figure 43

Sample data scema in Big Query.



This image shows the schema tab of the sample_data_deduped table in BigQuery