# Technical Specification Document

## Medical Codes RAG Model

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Prepared By** | **Change Description** |
| **1.0** | **26-Feb-2025** | **Surya Kurapati** | **Technical Spec document – First cut** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Contents:

1. [**Overview**](#_1._Overview)
2. [**System Architecture**](#_2._System_Architecture)
   1. [**High-Level Architecture**](#_2.1._High-Level_Architecture)
3. [**Technical Specifications**](#_3._Technical_Specifications)
   1. [**Phase 1: Research & Data Collection (Weeks 1-4)**](#_3.1._Phase_1:)
      1. [**Data Sources**](#_3.1.1._Data_Sources)
      2. [**Infrastructure Setup**](#_3.1.2._Infrastructure_Setup)
      3. [**Compliance Review**](#_3.1.3._Compliance_Review)
      4. [**Deliverables**](#_Deliverables:)
   2. [**Phase 2: Model Development & Fine-Tuning (Weeks 5-10)**](#_3.2._Phase_2:)
      1. [**Model Training**](#_3.2.1._Model_Training)
      2. [**RAG Integration**](#_3.2.2._RAG_Integration)
      3. [**Codes Suggestion & Validation**](#_3.2.3._Code_Suggestion)
      4. [**Deliverables**](#_Deliverables:)
   3. [**Phase 3: Testing & Iteration (Weeks 11-14)**](#_3.3._Phase_3:)
      1. [**Pilot Testing**](#_3.3.1._Pilot_Testing)
      2. [**Accuracy Evaluation**](#_3.3.2._Accuracy_Evaluation)
      3. [**Error Detection and Fraud Prevention**](#_3.3.3._Error_Detection)
      4. [**Deliverables**](#_Deliverables:)
   4. [**Phase 4: Deployment & Scaling (Weeks 15-18)**](#_3.4._Phase_4:)
      1. [**API Development**](#_3.4.1._API_Development)
      2. [**User Interface (UI)**](#_3.4.2._User_Interface)
      3. [**Continuous Model Updates**](#_3.4.3._Continuous_Model)
      4. [**Deliverables**](#_Deliverables:)
4. [**Performance Metrics**](#_4._Performance_Metrics)
5. [**Technology Stack**](#_5._Technology_Stack)

# 

# 1. Overview

The **Medical Codes RAG Model** is an AI-powered system designed to improve the **accuracy, efficiency, and automation** of medical coding and insurance claim processing. Utilizing **Retrieval-Augmented Generation (RAG)**, the model retrieves **real-time medical coding guidelines, historical claims data, and payer policies** to assist medical coders, billers, and insurance providers.

**Key Features:**

* **Automated Medical Code Suggestions** for ICD, CPT, and HCPCS codes based on doctor’s notes.
* **Real-Time Claim Validation** ensuring compliance with payer policies and regulations.
* **Error Detection & Fraud Prevention** to flag incorrect, duplicate, or non-compliant claims.
* **Appeal Assistance** to help billers justify denied claims using historical case data.

This system aims to **reduce manual errors, streamline billing, accelerate reimbursements, and ensure compliance** with evolving regulations (Medicare, Medicaid, private insurance).

# 2. System Architecture

## 2.1. High-Level Architecture

The Medical Codes RAG Model consists of the following components:

1. **Data Sources & Preprocessing**
   * Medical Code Databases: ICD, CPT, HCPCS
   * Payer Policy Documents: Medicare, Medicaid, private insurers
   * Historical Claims Data: Approved, rejected, and appealed claims
   * Compliance Standards: HIPAA, GDPR
2. **Retrieval-Augmented Generation (RAG) Pipeline**
   * **Retriever:** FAISS / Elasticsearch for fast lookup of medical codes & policies
   * **Generator:** LLM fine-tuned on medical terminology (BioBERT, MedPaLM)
   * **Fusion:** Merging retrieved data with AI-generated responses
3. **Model & Processing Layers**
   * **Medical Code Prediction Model** (trained LLM)
   * **Real-Time Claim Validation Engine**
   * **Error Detection & Fraud Prevention Module**
4. **User Interface & API**
   * **Web-based Dashboard** for medical coders and billers
   * **API Integration** for hospital billing systems

# 3. Technical Specifications

## 3.1. Phase 1: Research & Data Collection (Weeks 1-4)

### 3.1.1. Data Sources

* **ICD, CPT, HCPCS** databases (structured/unstructured format)
* **Payer Policies** (PDFs, XML, API-based retrieval)
* **Historical Claims** (de-identified structured/unstructured claim data)

### 3.1.2. Infrastructure Setup

* **Retrieval Pipeline:** FAISS Open-Source / AWS Elasticsearch
* **Storage:** SQL DB / MongoDB or AWS Redshift / Snowflake / Document DB
* **Security:** HIPAA & GDPR-compliant encryption and access control

### 3.1.3. Compliance Review

* Data privacy: **De-identification & access control**
* Regulatory compliance: **Adherence to payer-specific policies**

### Deliverables:

* Labelled dataset of medical codes and claim records
* Technical architecture for retrieval and response generation

## 3.2. Phase 2: Model Development & Fine-Tuning (Weeks 5-10)

### 3.2.1. Model Training

* **Base Model:** Fine-tuned **BioBERT (Hugging Face – Extraction) / MedPaLM (gpt4, Mistral, gemini etc. – Q&A model)**
* **Dataset:** Medical terminology, structured coding rules, real-world claim scenarios
* **Training Framework:** PyTorch / TensorFlow / Custom Framework

### 3.2.2. RAG Integration

* **Retriever:** FAISS / Elasticsearch
* **Generator:** LLM fine-tuned on medical codes

### 3.2.3. Code Suggestion & Validation

* Predicts **ICD, CPT, HCPCS codes** from doctor’s notes
* Cross-checks **payer policies and historical claims**

### Deliverables:

* Functional RAG pipeline retrieving real-time medical codes
* AI-powered code recommendation and validation system

## 3.3. Phase 3: Testing & Iteration (Weeks 11-14)

### 3.3.1. Pilot Testing

* **Real-world test cases** with medical coders and billers
* **Comparison** of AI-generated codes with manually coded claims

### 3.3.2. Accuracy Evaluation

* **Code Matching Accuracy** vs. manual coding (target: **95%+**)
* **Claim Rejection Reduction** (target: **50% decrease**)

### 3.3.3. Error Detection & Fraud Prevention

* Flagging **duplicate, incorrect, and fraudulent claims**
* **Rule-based & AI-driven anomaly detection**

### Deliverables:

* AI model tested and validated with real-world scenarios
* Model refinements based on accuracy and compliance feedback

## 3.4. Phase 4: Deployment & Scaling (Weeks 15-18)

### 3.4.1. API Development

* **REST API** for seamless integration with **hospital billing systems**
* Endpoints:
  + /predict\_code – Suggests medical codes
  + /validate\_claim – Checks payer policy compliance
  + /detect\_error – Flags potential issues

### 3.4.2. User Interface (UI)

* **Web-based dashboard** for medical coders
* **Features:**
  + Code predictions
  + Claim validation status
  + Appeal recommendations

### 3.4.3. Continuous Model Updates

* **Automated Learning Pipeline** for new medical guidelines
* **Compliance Monitoring** for evolving Medicare/Medicaid policies

### Deliverables:

* Deployed RAG-powered Medical Coding System
* API and UI for medical coders & billers
* Real-time compliance tracking

# 4. Performance Metrics

| **Metric** | **Target Value** |
| --- | --- |
| **Code Matching Accuracy** | 95%+ |
| **Claim Rejection Reduction** | 50% |
| **Claim Processing Time** | 30% faster |
| **Compliance Accuracy** | 99%+ |

# 5. Technology Stack

| **Component** | **Technology** |
| --- | --- |
| **Model** | BioBERT (Hugging Face), MedPaLM, GPT-based LLMs (gpt-4, Mistral, gemini, text-ada-embedding) |
| **Retrieval System** | FAISS, Elasticsearch |
| **Backend** | Python (FastAPI) |
| **Frontend** | React |
| **Database** | PostgreSQL, SQL Server, Snowflake, AWS Redshift, Mongo DB, Document DB |