

دانشگاه فردوسی مشهد

دانشکده علوم ریاضی

MultiModal Agentic RAG

درس: پروژه کارشناسی
استاد درس: دکتر جلال الدین نصیری

دانشجو: علی محمود جانلو

سال ۱۴۰۴

فهرست مطالب

۴	۱ مقدمه
۴	۱.۱ منبع داده
۴	۲.۱ ویژگی‌های محتوای منبع
۵	۳.۱ اهداف پروژه
۵	۴.۱ درباره کتاب Oxford Textbook of Medicine
۷	۲ معماری سیستم
۷	۱.۲ دلایل انتخاب معماری Clean
۷	۲.۲ لایه‌های معماری
۸	۱.۲.۲ لایه موجودیت‌ها (Entities)
۸	۲.۲.۲ لایه (Use Cases)
۸	۳.۲.۲ لایه آداپتورهای رابط (Interface Adapters)
۹	۴.۲.۲ لایه فریم‌ورک‌ها و درایورها (Frameworks and Drivers)
۹	۳.۲ اصول طراحی
۹	۱.۳.۲ قانون وابستگی (Dependency Rule)
۹	۲.۳.۲ اصل وارونگی وابستگی (Dependency Inversion Principle)
۹	۴.۲ سیستم Dependency Injection
۱۰	۱.۴.۲ مزایای Dependency Injection
۱۰	۵.۲ یکپارچگی با فریم‌ورک‌های LLM
۱۰	۶.۲ استفاده از Pydantic
۱۱	۷.۲ مدیریت Error و Logging
۱۲	۸.۲ نتیجه‌گیری معماری
۱۳	۳ پردازش و نمایه‌سازی اسناد با Docling
۱۳	۱.۳ معرفی Docling
۱۳	۲.۳ دلایل انتخاب Docling
۱۴	۱.۲.۳ پردازش اسناد پزشکی پیچیده
۱۴	۲.۲.۳ معماری مبتنی بر مدل‌های تخصصی
۱۶	۳.۲.۳ حفظ ساختار سلسله‌مراتبی و روابط معنایی
۱۶	۴.۲.۳ تکه‌سازی هوشمند با حفظ زمینه
۱۸	۵.۲.۳ خروجی استاندارد و قابل پردازش
۱۸	۶.۲.۳ منبع‌باز و قابل توسعه
۱۸	۷.۲.۳ کارایی و سرعت پردازش
۱۹	۸.۲.۳ یکپارچگی با ابزارهای پردازش زبان طبیعی
۱۹	۳.۳ مقایسه با روش‌های جایگزین
۱۹	۱.۳.۳ مدل‌های زبانی بزرگ چندوجهی (Vision Language Models)
۱۹	۲.۳.۳ کتابخانه‌های سنتی PDF مانند PyPDF2 و PDFMiner
۱۹	۳.۳.۳ مقایسه با Unstructured.io
۲۱	۴.۳ یکپارچگی با سیستم نمایه‌سازی
۲۱	۵.۳ نتیجه‌گیری
۲۲	۴ سیستم استنتاج هوشمند RAG
۲۳	۱.۴ معماری گردش کار استنتاج
۲۳	۱.۱.۴ مراحل فرایند
۲۴	۲.۴ فرایند تولید پاسخ
۲۵	۳.۴ استخراج متادیتا و بازیابی تصاویر
۲۶	۴.۴ پیاده‌سازی لایه رابط کاربری

۲۶ ۵.۴ نتیجه‌گیری

۲۷ References ۵

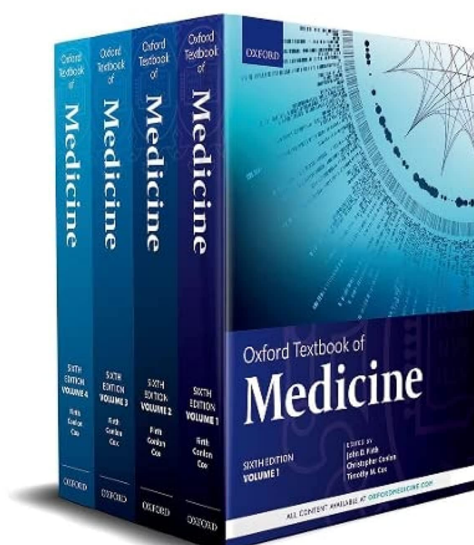
۱ مقدمه

با رشد روزافزون حجم اطلاعات پزشکی و انتشار مستندات تخصصی در قالب‌های مختلف، دسترسی سریع و دقیق به دانش مورد نیاز در این حوزه به چالشی اساسی تبدیل شده است. کتاب‌های پزشکی تخصصی معمولاً شامل محتوای پیچیده و چندوجهی هستند که ترکیبی از متن، جداول، نمودارها و تصاویر تشخیصی را در بر می‌گیرند. استخراج و بازیابی اطلاعات از این منابع به روش‌های سنتی زمان‌بر و غیرکارآمد است.

در این پروژه، یک سامانه بازیابی اطلاعات چندوجهی مبتنی بر هوش مصنوعی توسعه یافته است که قادر به پردازش هوشمند اسناد پزشکی غیرساختاریافته و ایجاد پایگاه دانش جامع برای پاسخگویی به پرسش‌های تخصصی است. این سامانه از تکنیک‌های پیشرفته پردازش زبان طبیعی، بینایی ماشین و بازیابی تقویت‌شده با تولید (Retrieval-Augmented Generation - RAG) بهره می‌برد.

۱.۱ منبع داده

برای ارزیابی و آزمایش سامانه، کتاب Oxford Textbook of Medicine, 6th Edition به عنوان منبع اصلی داده انتخاب شده است. این کتاب یکی از معتبرترین و جامع‌ترین مراجع پزشکی در سطح جهان است که توسط انتشارات دانشگاه آکسفورد منتشر شده و شامل چهار جلد و سی بخش تخصصی می‌باشد.



شکل ۱: جلد کتاب Oxford Textbook of Medicine, 6th Edition

در این پروژه، جلد دوم این مجموعه که شامل بخش‌های ۱۰ تا ۱۵ است، مورد پردازش قرار گرفته است. این جلد حاوی ۱۸۷۳ صفحه غیرساختاریافته از محتوای پزشکی تخصصی است که طیف وسیعی از موضوعات را پوشش می‌دهد.

۲.۱ ویژگی‌های محتوای منبع

محتوای این جلد دارای ویژگی‌های زیر است که پردازش آن را به چالشی فنی تبدیل می‌کند:

- **ساختار سلسله‌مراتبی پیچیده:** محتوا دارای سطوح متعدد سرفصل، زیربخش و پاراگراف‌های مرتبط است که حفظ این ساختار برای درک صحیح مطالب ضروری است.

- **جداول اطلاعاتی متنوع:** صفحات حاوی جداول پیچیده با داده‌های بالینی، آماری و تشخیصی هستند که نیاز به استخراج دقیق و حفظ ساختار دارند.
- **محتوای بصری تخصصی:** تصاویر تشخیصی، نمودارها و دیاگرام‌های پزشکی که اطلاعات حیاتی را به صورت بصری منتقل می‌کنند و نیاز به تحلیل هوشمند دارند.
- **اصطلاحات تخصصی پزشکی:** استفاده گسترده از واژگان تخصصی، اختصارات و نام‌گذاری‌های علمی که نیاز به پردازش دقیق دارند.
- **پیوستگی محتوایی:** ارتباط منطقی بین بخش‌های مختلف که حفظ آن برای درک کامل موضوعات الزامی است.

۳.۱ اهداف پروژه

اهداف اصلی این پروژه عبارتند از:

- **تبدیل ساختاریافته:** پردازش اسناد غیرساختاریافته PDF و تبدیل آن‌ها به فرمت‌های ساختاریافته با حفظ کامل سلسله‌مراتب و روابط محتوایی.
- **پردازش چندوجهی:** استخراج و تحلیل هوشمند متن، جداول و محتوای بصری به صورت یکپارچه.
- **نمایه‌سازی هوشمند:** ایجاد پایگاه داده قابل جستجو با حفظ زمینه و ساختار اطلاعات.
- **بازیابی دقیق:** پاسخگویی به پرسش‌های تخصصی با استفاده از اطلاعات استخراج‌شده از متن، جداول و تصاویر.
- **معماری پاک و قابل توسعه:** طراحی سامانه با رعایت اصول مهندسی نرم‌افزار برای قابلیت نگهداری و توسعه آینده.

۴.۱ درباره کتاب Oxford Textbook of Medicine

کتاب Oxford Textbook of Medicine برجسته‌ترین و معتبرترین کتاب بین‌المللی پزشکی است که به عنوان یک مرجع کلیدی در دفاتر و بخش‌های درمانی پزشکان در سراسر جهان حضور دارد. این کتاب با پوشش بی‌نظیر جنبه‌های علمی و عملکرد بالینی در حوزه طب داخلی و تخصص‌های فرعی آن، همچنین به عنوان منبعی اساسی برای پزشکی قانونی مورد استفاده قرار می‌گیرد. این اثر جامع، معتبر و بین‌المللی بر ارائه دیدگاه‌های تخصصی و راهنمایی‌های عملی در مدیریت بالینی و پیشگیری از بیماری‌ها تمرکز دارد. بخش‌های مقدماتی این کتاب بر تجربه بیمار، اخلاق پزشکی و تصمیم‌گیری بالینی متمرکز است و فلسفه‌ای انسانی و تفکربرانگیز را ارائه می‌دهد که همواره مشخصه این اثر بوده است. این کتاب به دنبال القای درکی عمیق از نقش پزشکی در جامعه و مشارکت آن در سلامت جمعیت است و از بحث درباره جنبه‌های بحث‌برانگیز پزشکی مدرن طفره نمی‌رود. ویژگی‌های برجسته این کتاب عبارتند از:

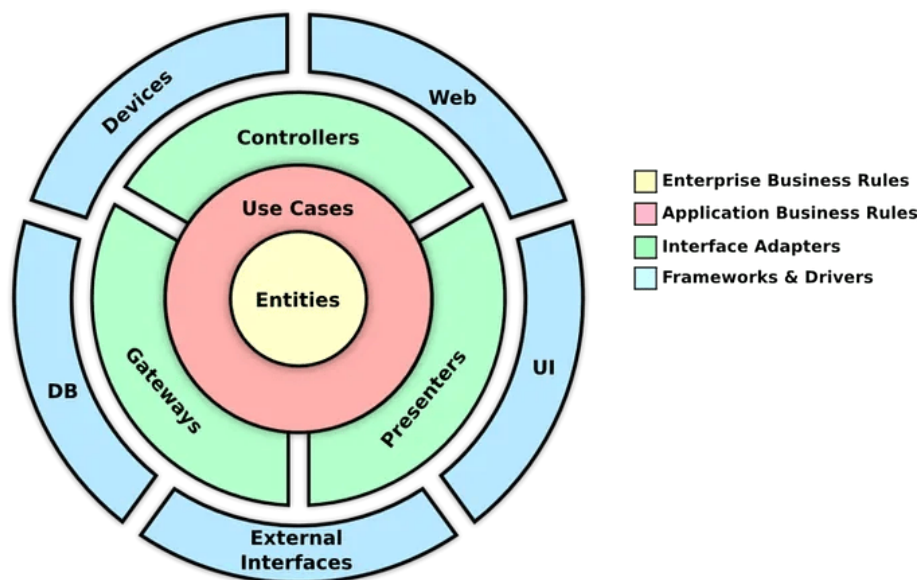
- **یکپارچگی علوم پایه و عمل بالینی:** ادغام بی‌نظیر دانش علمی و کاربردهای بالینی در سراسر کتاب، همراه با توضیح پیامدهای تحقیقات برای عملکرد پزشکی.
- **پوشش جامع بیماری‌های عفونی:** جامع‌ترین پوشش بیماری‌های عفونی که در هیچ کتاب پزشکی دیگری یافت نمی‌شود.
- **موضوعات تخصصی متنوع:** شامل سلول‌های بنیادی و پزشکی بازساختی، نابرابری‌های سلامت، جنبه‌های پزشکی آلودگی و تغییرات اقلیمی، پزشکی سفر و اکسپدیشن، بیوتروپسم و پزشکی قانونی، درد، اختلالات پزشکی در بارداری، تغذیه، و روانپزشکی.

- **بخش پزشکی اورژانسی:** طراحی شده برای دسترسی سریع به اطلاعات در مواقع نیاز فوری.
- **دسترسی دیجیتال و چاپی:** امکان دسترسی چندگانه بر اساس نیاز و ترجیح کاربران، همراه با به روزرسانی‌های منظم در نسخه آنلاین.
- **پایگاه مدارک و مراجع پیشرو:** استفاده از جدیدترین یافته‌های تحقیقاتی و شواهد علمی.

ویرایش ششم این کتاب با بهبودهای قابل توجهی همراه بوده است که در پاسخ به بازخوردهای مستمر کاربران انجام شده است. افزودن بخش‌های خلاصه فصل‌ها برای ارائه دید کلی از محتوا، طراحی جدید برای سهولت خواندن و ناوبری، و همچنین دسترسی رایگان به نسخه آنلاین Oxford Medicine Online برای خریداران نسخه چاپی، از جمله این بهبودها هستند. در فصل‌های بعدی، جزئیات معماری سامانه، روش‌های پردازش اسناد، الگوریتم‌های نمایه‌سازی و نتایج ارزیابی به تفصیل ارائه خواهد شد.

۲ معماری سیستم

این پروژه بر اساس الگوی معماری Clean Architecture پیاده‌سازی شده است که امکان جداسازی مناسب نگرانی‌ها، مقیاس‌پذیری و نگهداری‌پذیری بالا را فراهم می‌آورد. انتخاب این معماری برای یک سیستم مبتنی بر مدل‌های زبانی بزرگ (LLM) که نیازمند انعطاف‌پذیری بالا و قابلیت تست‌پذیری است، از اهمیت ویژه‌ای برخوردار است.



شکل ۲: نمودار معماری Clean Architecture و لایه‌های آن

شکل ۲ نمودار کلی معماری Clean را نشان می‌دهد که در آن جهت وابستگی‌ها از بیرون به درون است و لایه‌های مرکزی از جزئیات پیاده‌سازی مستقل هستند.

۱.۲ دلایل انتخاب معماری Clean

انتخاب معماری Clean برای این پروژه بر اساس چندین معیار کلیدی صورت گرفته است:

- **قابلیت تست‌پذیری:** با جداسازی لایه‌های مختلف و استفاده از تزریق وابستگی (Dependency Injection)، امکان نوشتن تست‌های واحد و یکپارچه با پوشش بالا میسر می‌شود.
- **انعطاف‌پذیری در تعویض ارائه‌دهندگان LLM:** سیستم طراحی شده به گونه‌ای است که تغییر یا افزودن ارائه‌دهندگان مختلف LLM (مانند OpenAI، Cohere، یا مدل‌های متن‌باز) بدون تغییر در منطق اصلی امکان‌پذیر است.
- **مقیاس‌پذیری:** ساختار لایه‌بندی شده امکان توسعه و گسترش سیستم را در آینده تسهیل می‌کند.

۲.۲ لایه‌های معماری

معماری این سیستم شامل چهار لایه اصلی است که هر کدام مسئولیت‌های مشخصی دارند و قانون وابستگی (Dependency Rule) در آن‌ها رعایت می‌شود.

۱.۲.۲ لایه موجودیت‌ها (Entities)

این لایه، قلب سیستم را تشکیل می‌دهد و شامل enterprise business rules است که مستقل از هرگونه جزئیات پیاده‌سازی خارجی هستند.
مسئولیت‌های این لایه:

- تعریف core data models با استفاده از Pydantic
- Data validation و اعمال business rules
- تضمین type-safety و automatic serialization داده‌ها
- به عنوان مثال، موجودیت DocChunk در سیستم ما شامل قوانین کسب‌وکار زیر است:
- **تولید شناسه یکتا:** هر chunk باید دارای شناسه‌ای یکتا باشد که بر اساس document_id و index تولید می‌شود: {document_id}_chunk_{index}
- **اعتبارسنجی بردار embedding:** اگر بردار embedding موجود باشد، باید از نوع List[float] و غیر خالی باشد
- **حفظ metadata اصلی:** تمام اطلاعات meta شامل schema_name، version و doc_items باید حفظ شوند
- **تبدیل فرمت:** موجودیت باید قابلیت تبدیل دوطرفه به فرمت Elasticsearch و Docling را داشته باشد
- این قوانین مستقل از جزئیات پیاده‌سازی database یا search engine تعریف شده‌اند و در صورت تغییر فناوری زیرساخت، دست‌نخورده باقی می‌مانند.

۲.۲.۲ لایه (Use Cases)

این لایه application business logic را در بر می‌گیرد و workflow بین entities و interface adapters را هماهنگ می‌کند.
مسئولیت‌های کلیدی:

- مدیریت جریان‌های کاری پیچیده با استفاده از LangGraph
- پیاده‌سازی pipelines (Retrieval-Augmented Generation) RAG
- تولید و مدیریت embeddings
- هماهنگی با سرویس‌های خارجی از طریق interfaces

۳.۲.۲ لایه آداپتورهای رابط (Interface Adapters)

این لایه وظیفه تبدیل داده‌ها بین موارد استفاده و دنیای خارج را بر عهده دارد.
اجزای این لایه شامل:

- **Controllers:** مدیریت HTTP requests و responses
- **Presenters:** قالب‌بندی داده‌ها برای رابط کاربری یا مصرف‌کنندگان API
- **Gateways:** رابط با سرویس‌های خارجی مانند databases، vector stores و third-party APIs

۴.۲.۲ لایه فریم‌ورک‌ها و درایورها (Frameworks and Drivers)

بیرونی‌ترین لایه سیستم که شامل جزئیات پیاده‌سازی ابزارها و فریم‌ورک‌های خارجی است. مثال‌های موجود در این لایه:

- Web frameworks (Flask, FastAPI)
- LLM providers (open-source models, Cohere, OpenAI)
- Databases و vector stores (Pinecone, Redis, Elasticsearch)

۳.۲ اصول طراحی

۱.۳.۲ قانون وابستگی (Dependency Rule)

در این معماری، وابستگی‌ها همواره از لایه‌های بیرونی به سمت لایه‌های درونی هستند. این بدان معناست که:

Frameworks/Drivers → Interface Adapters → Use Cases → Entities (۱)

لایه‌های درونی هیچ‌گونه اطلاعی از لایه‌های بیرونی ندارند و این استقلال امکان تغییر پیاده‌سازی‌های خارجی را بدون تأثیر بر هسته سیستم فراهم می‌آورد.

۲.۳.۲ اصل وارونگی وابستگی (Dependency Inversion Principle)

برای حفظ قانون وابستگی، از interfaces و کلاس‌های انتزاعی استفاده می‌شود. به این ترتیب:

- Interfaces در لایه‌های هسته (Use Cases) تعریف می‌شوند
- پیاده‌سازی‌های واقعی در لایه‌های بیرونی (Interface Adapters یا Frameworks) قرار می‌گیرند
- لایه‌های سطح بالا به انتزاع وابسته‌اند، نه به پیاده‌سازی‌های خاص

مثال کاربردی: در سیستم ما، use case به نام `agentic_rag` مازولی از نوع `EmbeddingServiceInterface` دریافت می‌کند. این use case توابعی مانند `embed_single` را فراخوانی می‌کند، بدون آنکه بداند پیاده‌سازی واقعی از چه ارائه‌دهنده‌ای استفاده می‌کند (مانند OpenAI، Google GenAI، Azure، یا مدل‌های متن‌باز). این امر به این دلیل امکان‌پذیر است که:

- Use case تنها به contract تعریف شده در interface وابسته است
- تعویض ارائه‌دهنده embedding بدون تغییر در منطق کسب‌وکار به راحتی امکان‌پذیر است

۴.۲ سیستم Dependency Injection

برای مدیریت وابستگی‌ها و پیکربندی سیستم، از کتابخانه `dependency-injector` استفاده شده است. این سیستم مزایای زیر را فراهم می‌آورد:

۱.۴.۲ مزایای Dependency Injection

- **Centralized configuration:** تمام تنظیمات و وابستگی‌ها در یک container مرکزی مدیریت می‌شوند
- **تسهیل تست:** امکان جایگزینی وابستگی‌های واقعی با mocks برای تست‌نویسی
- **مدیریت singleton:** کنترل lifecycle منابع گران‌قیمت مانند database connections
- **انعطاف‌پذیری:** تغییر configuration در runtime بدون تغییر کد

انواع providers در کتابخانه dependency-injector:

- **Factory:** برای ایجاد نمونه جدید در هر بار استفاده. این نوع provider هر بار که درخواست می‌شود، یک instance جدید از کلاس مورد نظر ایجاد می‌کند. برای اشیائی مناسب است که stateless هستند یا هر بار نیاز به تنظیمات تازه دارند.
- **Singleton:** برای نگهداری یک نمونه واحد در طول عمر برنامه. این provider فقط یک بار شیء را می‌سازد و در تمام درخواست‌های بعدی همان نمونه را باز می‌گرداند. برای منابع گران‌قیمت مانند اتصالات database، HTTP clients یا configuration objects مفید است.
- **Resource:** برای مدیریت چرخه حیات اشیاء پیچیده که نیاز به راه‌اندازی و پاکسازی دارند. این provider قابلیت‌های lifecycle management مانند initialization، cleanup و مدیریت منابع سیستمی را فراهم می‌آورد. برای سرویس‌هایی که نیاز به graceful shutdown دارند، ایده‌آل است.
- **Configuration:** برای مدیریت و تزریق مقادیر پیکربندی مانند environment variables، تنظیمات API، یا پارامترهای سیستم. این provider امکان خواندن مقادیر از فایل‌های config، متغیرهای محیطی یا منابع خارجی را فراهم می‌آورد و آن‌ها را در سراسر سیستم در دسترس قرار می‌دهد.

۵.۲ یکپارچگی با فریم‌ورک‌های LLM

- یکی از چالش‌های اصلی این پروژه، یکپارچگی با فریم‌ورک‌های مختلف LLM بوده است. معماری طراحی شده این امکان را فراهم می‌آورد که:
- از چندین ارائه‌دهنده LLM به صورت همزمان استفاده شود
 - جریان‌های کاری پیچیده با LangGraph مدیریت شوند
 - عملیات RAG به صورت ماژولار پیاده‌سازی شوند
 - تغییر یا ارتقای فریم‌ورک‌ها بدون تغییر منطق اصلی امکان‌پذیر باشد

۶.۲ استفاده از Pydantic

- در لایه موجودیت‌ها، از کتابخانه Pydantic برای تعریف data models استفاده شده است. این انتخاب به دلایل زیر صورت گرفته است:
- **Automatic validation:** داده‌ها به صورت خودکار در زمان ایجاد object اعتبارسنجی می‌شوند

- **Type Safety:** تضمین صحت data types در زمان توسعه
- **Serialization:** تبدیل خودکار به JSON و سایر فرمت ها
- **Automatic documentation:** تولید schema برای API ها

۷.۲ مدیریت Logging و Error

سیستم دارای یک لایه یکپارچه برای error handling و logging است که:

- از multi-level logging پشتیبانی می‌کند
 - خطاها را به صورت ساختاریافته ثبت می‌کند
 - امکان request tracing را فراهم می‌آورد
 - با سیستم‌های monitoring خارجی قابل یکپارچگی است
- به عنوان مثال، میتوان از Prometheus برای جمع‌آوری متریک ها و از Grafana برای visualization استفاده کرد تا به اهداف زیر رسید:
- **Metrics Collection:** Response times، تعداد درخواست‌ها، نرخ خطا، و استفاده از منابع
 - **Health Checks:** بررسی وضعیت سرویس‌های LLM، database connections، و vector stores
 - **Alerting:** اطلاع‌رسانی خودکار در صورت بروز مشکل از طریق Slack یا Email

سیستم logging پیاده‌سازی شده در این پروژه، از قابلیت automatic rotation و مدیریت log levels مختلف برخوردار است. شکل ۳ نمونه‌ای از خروجی سیستم logging را نشان می‌دهد که شامل timestamp، نام log level، module و message مربوطه است.

```
2025-10-13 21:02:09 - multimodal_rag.frameworks.google_genai_base_service - INFO - google_genai_base_service.py:79 - _switch_to_next_api_key() - Switched to API key index 0
2025-10-13 21:02:09 - multimodal_rag.frameworks.google_genai_base_service - WARNING - google_genai_base_service.py:83 - _switch_to_next_api_key() - Cycled back to token 0, waiting 60 seconds before continuing
2025-10-13 21:03:09 - multimodal_rag.frameworks.google_genai_base_service - INFO - google_genai_base_service.py:115 - _execute_with_retry_and_token_switching() - Trying embeddings_for_content with API key index 0
2025-10-13 21:03:11 - multimodal_rag.frameworks.google_genai_base_service - INFO - google_genai_base_service.py:122 - _execute_with_retry_and_token_switching() - Successfully executed embeddings_for_content with API key index 0
2025-10-13 21:03:11 - multimodal_rag.frameworks.google_genai_base_service - INFO - google_genai_base_service.py:115 - _execute_with_retry_and_token_switching() - Trying embeddings_for_content with API key index 0
2025-10-13 21:03:13 - multimodal_rag.frameworks.google_genai_base_service - WARNING - google_genai_base_service.py:134 - _execute_with_retry_and_token_switching() - Rate limit error with API key index 0, switching immediately
2025-10-13 21:03:13 - multimodal_rag.frameworks.google_genai_base_service - INFO - google_genai_base_service.py:79 - _switch_to_next_api_key() - Switched to API key index 1
2025-10-13 21:03:13 - multimodal_rag.frameworks.google_genai_base_service - INFO - google_genai_base_service.py:115 - _execute_with_retry_and_token_switching() - Trying embeddings_for_content with API key index 1
```

شکل ۳: نمونه output سیستم logging

۸.۲ نتیجه‌گیری معماری

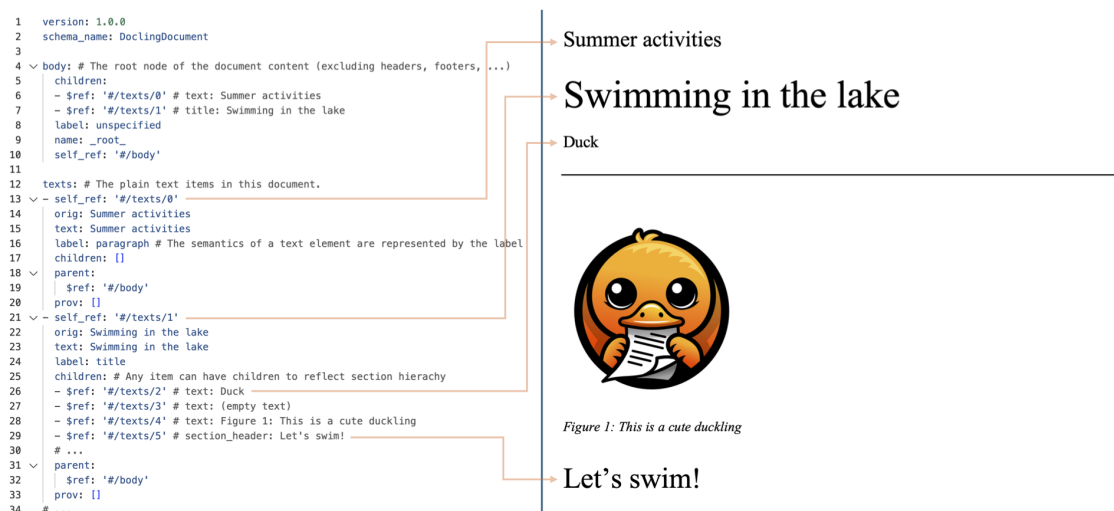
معماری Clean برای این پروژه MultiModal RAG، بستری مناسب برای توسعه یک سیستم scalable، maintainable و flexible فراهم آورده است. جداسازی واضح لایه‌ها، استفاده از dependency injection، و رعایت اصول SOLID، امکان توسعه پایدار و با کیفیت این سیستم را در بلندمدت تضمین می‌کند. این رویکرد معماری به‌ویژه برای سیستم‌های مبتنی بر LLM که نیازمند flexibility بالا در تعویض models و providers هستند، مناسب است.

۳ پردازش و نمایه‌سازی اسناد با Docling

۱.۳ معرفی Docling

Docling یک کتابخانه منبع‌باز پیشرفته برای پردازش و تبدیل اسناد ساختارنیافته به فرمت‌های قابل استفاده در سیستم‌های بازیابی اطلاعات است. این ابزار توسط تیم تحقیقاتی IBM توسعه یافته و به‌صورت رایگان در دسترس جامعه علمی قرار گرفته است. Docling با استفاده از مدل‌های یادگیری عمیق پیشرفته، قادر به تشخیص ساختار سلسله‌مراتبی اسناد، استخراج جداول پیچیده، و پردازش محتوای چندوجهی شامل متن، تصاویر و نمودارها می‌باشد. ویژگی‌های کلیدی Docling عبارتند از:

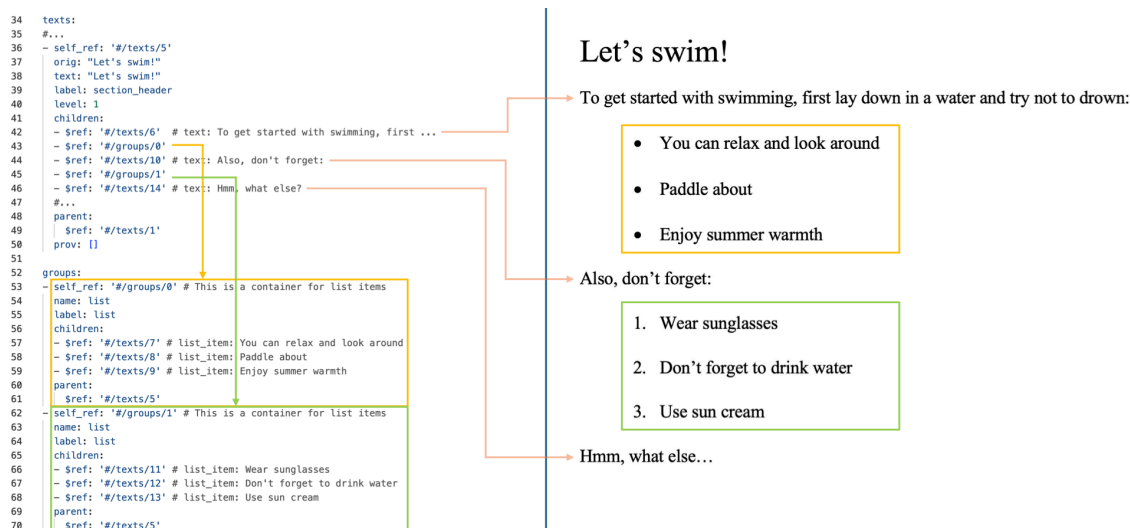
- **تشخیص ساختار سلسله‌مراتبی:** شناسایی خودکار عناوین، زیرعناوین، پاراگراف‌ها، و بخش‌های مختلف سند
- **استخراج دقیق جداول:** تبدیل جداول پیچیده با سلول‌های ادغام‌شده به فرمت ساختاریافته
- **پشتیبانی از محتوای چندوجهی:** پردازش همزمان متن، تصاویر، نمودارها و فرمول‌های ریاضی
- **خروجی استاندارد:** تولید فایل‌های JSON، YAML و Markdown با حفظ روابط والد-فرزندی بین عناصر
- **تکه‌سازی هوشمند:** ترکیب عناصر مرتبط در تکه‌های معنادار با رعایت محدودیت‌های طول توکن



شکل ۴: نمونه‌ای از ساختار DoclingDocument. سمت راست صفحه‌ای با عناوین، تصویر، زیرنویس و پاراگراف‌ها را نشان می‌دهد. سمت چپ درخت سند را نمایش می‌دهد که در آن هر عنصر با برچسب و اشاره‌گر خود در ساختار سلسله‌مراتبی قرار گرفته است.

۲.۳ دلایل انتخاب Docling

انتخاب Docling به‌عنوان ابزار اصلی نمایه‌سازی کتاب پزشکی در این پروژه بر اساس ارزیابی دقیق نیازمندی‌های سیستم و مقایسه با ابزارهای موجود انجام شده است. در ادامه مهم‌ترین دلایل این انتخاب شرح داده می‌شود.



شکل ۵: فهرست‌های تو در تو و گروه‌ها در DoclingDocument. سمت راست دو فهرست (یکی نقطه‌ای و دیگری شماره‌دار) را نشان می‌دهد. سمت چپ نشان می‌دهد که هر فهرست در یک کانتینر گروهی قرار می‌گیرد و عناصر آن به‌عنوان فرزند اضافه می‌شوند.

۱.۲.۳ پردازش اسناد پزشکی پیچیده

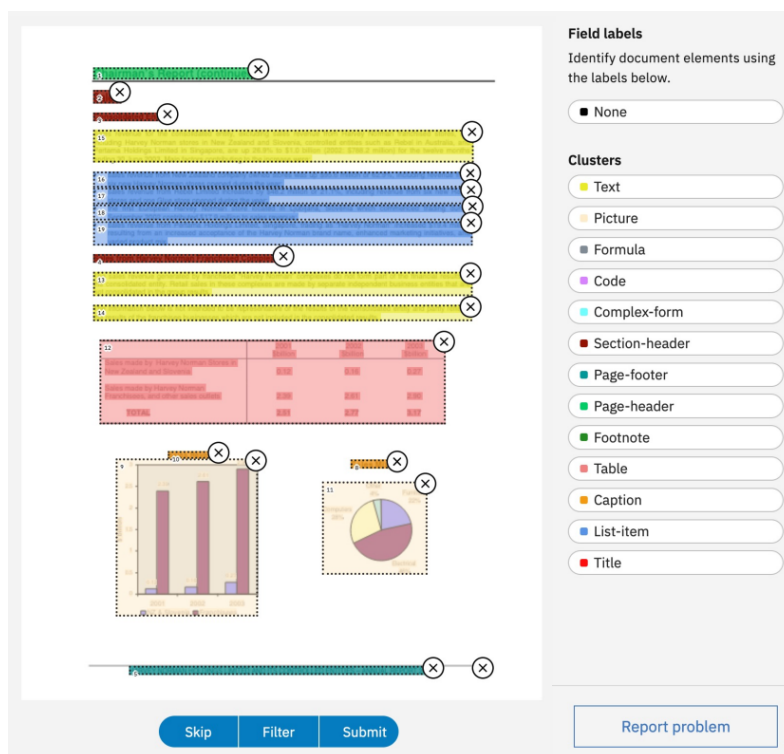
اسناد پزشکی از پیچیده‌ترین انواع محتوای علمی هستند که شامل ترکیبی از متن تخصصی، جداول آماری، تصاویر تشخیصی، نمودارهای آناتومیک و فرمول‌های دارویی می‌باشند. Docling با معماری پردازشی چندمرحله‌ای خود، قادر به تشخیص و استخراج دقیق این عناصر متنوع است. برخلاف ابزارهای سنتی که صفحات را به‌صورت یک‌پارچه پردازش می‌کنند، Docling از یک خط‌لوله مرحله‌ای استفاده می‌کند که در آن هر نوع محتوا با مدل تخصصی خود پردازش می‌شود.

۲.۲.۳ معماری مبتنی بر مدل‌های تخصصی

یکی از برتری‌های اصلی Docling نسبت به روش‌های مبتنی بر مدل‌های زبانی بزرگ چندوجهی (Vision Language Models) این است که به‌جای استفاده از یک مدل عمومی برای تمام وظایف، از مدل‌های تخصصی برای هر بخش استفاده می‌کند:

• **مدل تحلیل چیدمان (Layout Analysis):** Docling از مدل Heron مبتنی بر معماری RT-DETR با پشتیبانی ResNet-50 استفاده می‌کند که روی مجموعه داده DocLayNet شامل بیش از ۸۰ هزار صفحه برچسب‌گذاری شده آموزش دیده است. این مدل قادر به تشخیص ۱۷ کلاس مختلف شامل عنوان، متن، فهرست، جدول، تصویر، عنوان بخش، فرمول، زیرنویس و سایر عناصر است. دقت این مدل بر روی مجموعه داده استاندارد DocLayNet به ۷۸ درصد mAP می‌رسد (شکل ۶).

• **مدل استخراج ساختار جداول (TableFormer):** برای پردازش جداول پیچیده، Docling از مدل TableFormer استفاده می‌کند که یک معماری ترنسفورمر مبتنی بر بینایی است. این مدل روی مجموعه داده‌های بزرگ شامل PubTabNet (بیش از ۲۲۷ هزار جدول) آموزش دیده و قادر به تشخیص سطرها، ستون‌ها، سلول‌های ادغام‌شده، و ساختارهای پیچیده جداول است. خروجی این مدل شامل توالی توکن‌های HTML و مختصات مرزی هر سلول است که سپس با متن اصلی PDF تطبیق داده می‌شود (شکل ۷).



شکل ۶: نمونه‌ای از برچسب‌گذاری عناصر مختلف سند در مجموعه داده DocLayNet. این تصویر نشان می‌دهد که چگونه هر عنصر صفحه (متن، عنوان، جدول، تصویر، فهرست و غیره) با کلاس‌های مختلف شناسایی و برچسب‌گذاری می‌شود. این برچسب‌های دقیق انسانی پایه آموزش مدل Heron را تشکیل می‌دهند.

Solar System

Planet	Distance from Sun (average)	Number of Moons	Average Orbit Time
Mercury	58 million km (0.39 AU)	0	87.97 days
Venus	108 million km (0.72 AU)	0	224.7 days
Earth	149.6 million km (1 AU)	1 (The Moon)	365.25 days
Mars	227.9 million km (1.38 AU)	2 (Phobos, Deimos)	687.01 days
Jupiter	778.3 million km (5.2 AU)	79 (Io, Europa, Ganymede, Callisto, and 75 others)	11.86 years
Saturn	1.43 billion km (8.3 AU)	62 (Titan, Enceladus, Dione, Rhea, Tethys, and 57 others)	29.46 years
Uranus	2.88 billion km (19.1 AU)	27 (Titania, Oberon, Umbriel, Ariel, Miranda, and 23 others)	84.01 years
Neptune	4.46 billion km (30.1 AU)	14 (Triton, Nereid, and 12 others)	164.79 years

```

Users > codyburn > Desktop > solar-system-overview.md > ## Probes Sent to Planets
1  ## Solar System
2
3  | Planet | Distance from Sun (average) | Number of Moons | Average Orbit Time |
4  |-----|-----|-----|-----|
5  | Mercury | 58 million km (0.39 AU) | 0 | 87.97 days |
6  | Venus | 108 million km (0.72 AU) | 0 | 224.7 days |
7  | Earth | 149.6 million km (1 AU) | 1 (The Moon) | 365.25 days |
8  | Mars | 227.9 million km (1.38 AU) | 2 (Phobos, Deimos) | 687.01 days |
9  | Jupiter | 778.3 million km (5.2 AU) | 79 (Io, Europa, Ganymede, Callisto, and 75 others) | 11.86 years |
10 | Saturn | 1.43 billion km (8.3 AU) | 62 (Titan, Enceladus, Dione, Rhea, Tethys, and 57 others) | 29.46 years |
11 | Uranus | 2.88 billion km (19.1 AU) | 27 (Titania, Oberon, Umbriel, Ariel, Miranda, and 23 others) | 84.01 years |
12 | Neptune | 4.46 billion km (30.1 AU) | 14 (Triton, Nereid, and 12 others) | 164.79 years |
13
14 ## Probes Sent to Planets
  
```

شکل ۷: مثالی از استخراج دقیق جدول توسط TableFormer. بالا: جدول اصلی منظم شامل اطلاعات سیارات منظومه شمسی. پایین: جدول استخراج شده در فرمت Markdown که ساختار دقیق ستون‌ها و سطرها حفظ شده است.

• **مدل توصیف تصاویر (Picture Description Model):** یکی از چالش‌های مهم در پردازش کتاب‌های پزشکی، تولید توصیفات دقیق و جامع از تصاویر پیچیده پزشکی است. مدل پیش‌فرض Docling برای این منظور SmolVLM با حدود ۲۵۶ میلیون پارامتر است. با وجود سبک بودن این مدل، عملکرد آن در توصیف تصاویر پیچیده و نموداری پزشکی مطلوب نبود. در مرحله بعد، از مدل Granite-3.1 Vision شرکت IBM با ۲ میلیارد پارامتر استفاده شد که نتایج بهتری ارائه داد، اما به دلیل مشکل نشت حافظه (Memory Leak) در Docling، اجرای پایدار آن در محیط Google Colab با GPU T4 امکان‌پذیر نبود. در نهایت، برای دستیابی به بهترین کیفیت توصیف، از API رایگان مدل Gemini 2.5-Flash شرکت Google استفاده شد. برای مدیریت محدودیت‌های نرخ درخواست، یک سیستم تعویض خودکار بین توکن‌های مختلف پیاده‌سازی شد تا در صورت برخورد به خطای محدودیت نرخ با یک توکن، به‌طور خودکار از توکن دیگری استفاده شود و پردازش بدون وقفه ادامه یابد (شکل ۸).

• **موتور تشخیص نوشتار (OCR):** برای پردازش محتوای اسکن‌شده یا تصویری، Docling از EasyOCR استفاده می‌کند که یک کتابخانه یادگیری عمیق مبتنی بر PyTorch است. این موتور از مدل CRAFT برای تشخیص مناطق متنی و مدل CRNN برای خواندن کاراکترها استفاده می‌کند. Docling صفحات را با وضوح ۲۱۶ DPI رندر کرده و برای دستیابی به دقت بالا در تشخیص فونت‌های کوچک، از این موتور بهره می‌برد.

این معماری متخصص‌محور باعث می‌شود دقت پردازش در مقایسه با مدل‌های عمومی چندوجهی به‌طور قابل توجهی افزایش یابد، زیرا هر مدل فقط بر روی یک وظیفه خاص تمرکز دارد و برای آن بهینه‌سازی شده است.

۳.۲.۳ حفظ ساختار سلسله‌مراتبی و روابط معنایی

یکی از چالش‌های اساسی در پردازش اسناد پزشکی، حفظ روابط سلسله‌مراتبی بین عناصر مختلف است. کتاب‌های پزشکی دارای ساختار چندسطحی هستند که شامل فصل‌ها، بخش‌ها، زیربخش‌ها، پاراگراف‌ها، فهرست‌ها و عناصر وابسته مانند تصاویر و جداول می‌باشند. Docling با استفاده از مدل ترتیب خواندن (Reading Order Model) و سیستم مرجع‌دهی مبتنی بر JSON Pointer، این روابط را به‌طور کامل حفظ می‌کند.

در مدل داده DoclingDocument، تمام عناصر در یک ساختار درختی سازماندهی می‌شوند که در آن:

- هر عنصر دارای یک شناسه یکتا است
- روابط والد-فرزندی از طریق اشاره‌گرهای JSON تعریف می‌شوند
- عناصر گروهی مانند فهرست‌ها در کانتینرهای مجزا نگهداری می‌شوند
- ترتیب طبیعی خواندن عناصر حفظ می‌شود

این قابلیت برای سیستم بازپایی اطلاعات بسیار حیاتی است، زیرا امکان بازیابی محتوا با در نظر گرفتن بافت سلسله‌مراتبی آن را فراهم می‌کند. به‌عنوان مثال، هنگام بازیابی یک پاراگراف، سیستم می‌تواند به‌طور خودکار عنوان بخش و زیربخش مربوطه را نیز در نتایج قرار دهد.

۴.۲.۳ تکه‌سازی هوشمند با حفظ زمینه

Docling دارای یک سیستم تکه‌سازی هوشمند (Intelligent Chunker) است که یکی از نقاط قوت اصلی این ابزار محسوب می‌شود. برخلاف روش‌های سنتی تکه‌سازی که صرفاً بر اساس تعداد کاراکتر یا پاراگراف عمل می‌کنند، تکه‌ساز Docling از اطلاعات ساختاری سند برای ترکیب هوشمند عناصر مرتبط استفاده می‌کند. ویژگی‌های تکه‌ساز Docling:



7254 **## Butterflies and moths (Lepidoptera)**
 anticoagulant; procoagulant (activators of prothrombin, factor X, factor V), kallikrein- like, metalloproteinase, and phospholipase A 2 activities resulting in defibrinogenation and spontaneous bleeding. The case fatality of about 2% is usually attributable to cerebral haemorrhage. Symptoms

7257

7258 Fig. 10.4.2.39 Lesions caused by urticating abdominal hairs of female moths *Hylesia* spp. in Brazil. Copyright D. A. Warrell.

7259

7260 This close-up image displays human skin, likely an arm or torso, featuring a prominent dermatological rash. The central area shows a cluster of vesicles and pustules, some appearing eroded or crusted, all situated on an inflamed, erythematous (reddened) base. Several smaller, individual papules or vesicles are also scattered on the surrounding skin. This presentation is highly suggestive of an active, inflammatory, and potentially infectious vesicular skin eruption. A white ruler or scale is partially visible at the bottom indicating this is a clinical photograph for medical documentation, allowing for size assessment of the lesions.

7261

7262 <!-- image -->

7263

شکل ۸: نمونه‌ای از خروجی مدل توصیف تصویر Gemini 2.5-Flash برای یک تصویر پزشکی پیچیده. تصویر بالا یک ضایعه پوستی ناشی از تماس با موی بید برزیلی را نشان می‌دهد که شامل تاول‌ها و پاپول‌های قرمز روی پوست انسان است. مدل Gemini توانسته است جزئیات کامل تصویر شامل نوع ضایعه، موقعیت آناتومیک، و ویژگی‌های بالینی را به‌طور دقیق توصیف کند، که نشان‌دهنده برتری آن نسبت به مدل‌های سبک‌تر مانند SmoIVLM در پردازش تصاویر پزشکی پیچیده است.

- **ترکیب مبتنی بر معنا:** عناصر مرتبط مانند عنوان، پاراگراف‌های توضیحی، جداول و تصاویر مربوطه در یک تکه قرار می‌گیرند
 - **رعایت محدودیت توکن:** تکه‌ها با توجه به حداکثر طول توکن مجاز ساخته می‌شوند
 - **حفظ اطلاعات زمینه:** هر تکه شامل اطلاعات سلسله‌مراتبی مانند عنوان بخش و فصل است
 - **جلوگیری از شکستگی محتوا:** جداول و فهرست‌ها به صورت یک پارچه در تکه‌ها قرار می‌گیرند
- این رویکرد باعث می‌شود که تکه‌های تولید شده از نظر معنایی منسجم و برای سیستم بازیابی قابل استفاده باشند.

۵.۲.۳ خروجی استاندارد و قابل پردازش

- Docling دو نوع خروجی اصلی ارائه می‌دهد که هر کدام برای کاربردهای خاصی مناسب هستند:
- **فرمت Markdown:** نسخه تمیز و خوانا از سند که در آن ساختار سلسله‌مراتبی با استفاده از سرفصل‌های Markdown (مثل #، ##، ###) نمایش داده می‌شود، جداول به صورت جداول Markdown فرمت می‌شوند، و محتوای متنی به صورت پاراگراف‌های منظم سازماندهی می‌شود.
 - **فرمت JSON:** نمایش کامل و ساختاریافته سند که شامل تمام متادیتا، روابط سلسله‌مراتبی، مختصات عناصر، و اطلاعات تکمیلی است. این فرمت برای پردازش خودکار و نمایه‌سازی در پایگاه‌های داده بسیار مناسب است.
- در این پروژه، از فرمت JSON برای استخراج دقیق اطلاعات و نمایه‌سازی در Elasticsearch استفاده می‌شود، در حالی که فرمت Markdown برای بازبینی انسانی و کنترل کیفیت مورد استفاده قرار می‌گیرد.

۶.۲.۳ منبع‌باز و قابل توسعه

- Docling به صورت کامل منبع‌باز و تحت مجوز MIT منتشر شده است که مزایای زیر را به همراه دارد:
- امکان بررسی و درک کامل نحوه عملکرد داخلی سیستم
 - قابلیت سفارشی‌سازی و توسعه مدل‌ها برای نیازهای خاص
 - کاهش هزینه‌های عملیاتی با حذف نیاز به API پولی
 - امکان اجرا در محیط‌های محلی و امن
- این ویژگی برای پروژه‌های پزشکی که با داده‌های حساس سر و کار دارند، بسیار حیاتی است.

۷.۲.۳ کارایی و سرعت پردازش

- Docling با استفاده از بهینه‌سازی‌های مختلف، سرعت پردازش قابل قبولی را ارائه می‌دهد:
- استفاده از ONNX Runtime برای استنتاج سریع‌تر مدل‌ها
 - پردازش موازی صفحات در صورت وجود منابع کافی
 - استفاده از رزولوشن مناسب برای هر مرحله (۷۲ DPI برای تحلیل چیدمان و ۲۱۶ DPI برای OCR)
 - کش کردن نتایج میانی برای جلوگیری از پردازش مجدد
- در آزمایش‌های انجام شده بر روی کتاب پزشکی مورد استفاده در این پروژه، Docling توانست هر صفحه را در زمان متوسط ۱۰ تا ۱۵ ثانیه (با GPU T4 در محیط Google Colab) پردازش کند که برای کاربردهای دسته‌ای قابل قبول است.

۸.۲.۳ یکپارچگی با ابزارهای پردازش زبان طبیعی

Docling به راحتی با کتابخانه‌های پردازش زبان طبیعی و سیستم‌های بازیابی یکپارچه می‌شود. در این پروژه، خروجی Docling به طور مستقیم به Elasticsearch منتقل می‌شود و با استفاده از LangChain برای ایجاد زنجیره‌های پردازشی پیچیده‌تر استفاده می‌شود. این یکپارچگی آسان امکان ساخت خط‌لوله‌های پردازشی پیچیده را با حداقل کد اضافی فراهم می‌کند.

۳.۳ مقایسه با روش‌های جایگزین

برای ارزیابی جامع‌تر انتخاب Docling، مقایسه‌ای با سایر روش‌های موجود انجام شده است:

۱.۳.۳ مدل‌های زبانی بزرگ چندوجهی (Vision Language Models)

مدل‌هایی مانند OpenAI GPT 4، Claude 4، و Gemini قادر به پردازش تصاویر صفحات هستند، اما محدودیت‌های زیر را دارند:

- **هزینه بالا:** هر صفحه نیاز به فراخوانی API دارد که برای کتاب‌های بزرگ هزینه‌بر است
- **عدم تضمین ساختار:** خروجی این مدل‌ها ممکن است فاقد ساختار ثابت باشد. مقایسه‌ای از عملکرد پایپلاین آماده شده با داکلینگ و GPT-۵ در [این آدرس](#) تهیه شده است.
- **محدودیت در جداول پیچیده:** دقت پایین در استخراج جداول با ساختار پیچیده

۲.۳.۳ کتابخانه‌های سنتی PDF مانند PyPDF2 و PDFMiner

این ابزارها فقط متن برنامه‌نویسی شده PDF را استخراج می‌کنند و محدودیت‌های اساسی دارند:

- عدم تشخیص ساختار سلسله‌مراتبی
- ناتوانی در پردازش جداول پیچیده
- عدم پشتیبانی از محتوای اسکن شده
- خروجی بدون ساختار و فاقد متادیتا

۳.۳.۳ مقایسه با Unstructured.io

یکی از جدیدترین و قدرتمندترین جایگزین‌های تجاری برای Docling، پلتفرم Unstructured.io است که به صورت ترکیبی از نسخه منبع باز و سرویس ابری ارائه می‌شود. این پلتفرم نیز هدف مشابهی با Docling دارد: تبدیل اسناد ساختارنیافته به داده‌های قابل استفاده برای سیستم‌های هوش مصنوعی. ویژگی‌های کلیدی Unstructured.io:

- **تجزیه و تقسیم بندی پیشرفته:** قابلیت شناسایی و جداسازی عناصر مختلف سند شامل پاراگراف‌ها، جداول، عناوین و تصاویر
- **تمیزسازی و بهنجارسازی:** حذف نویزها و نرمال‌سازی محتوا برای کیفیت بهتر پردازش
- **تکه‌سازی انطباقی:** تقسیم محتوا به تکه‌های مناسب برای ورودی مدل‌های زبانی
- **کانکتورهای یکپارچه‌سازی:** اتصال مستقیم به پایگاه‌های داده، دریاچه‌های داده و سیستم‌های ذخیره‌سازی

Unstructured.io برای پروژه‌هایی که بودجه کافی برای سرویس‌های ابری موجود است، و اولویت بالایی برای سرعت پیاده‌سازی نسبت به کنترل کامل فرآیند قائل هستند، گزینه مناسبی محسوب می‌شود. با توجه به این مقایسه، Docling ترکیبی بهینه از دقت، کارایی، انعطاف‌پذیری و هزینه را ارائه می‌دهد که برای پردازش کتاب‌های پزشکی ایده‌آل است.

۴.۳ یکپارچگی با سیستم نمایه‌سازی

در این پروژه، خروجی Docling به صورت زیر با سیستم نمایه‌سازی یکپارچه می‌شود:

۱. **دریافت DoclingDocument:** پس از پردازش PDF، شی DoclingDocument حاوی تمام اطلاعات ساختاریافته دریافت می‌شود
۲. **تکه‌سازی هوشمند:** با استفاده از HybridChunker داخلی Docling، سند به تکه‌های معنادار با حداکثر طول توکن مشخص تقسیم می‌شود
۳. **غنی‌سازی با متادیتا:** هر تکه با اطلاعات اضافی شامل عنوان بخش، شماره صفحه، نوع محتوا، و روابط سلسله‌مراتبی غنی می‌شود
۴. **پردازش تصاویر:** تصاویر استخراج‌شده به مدل Gemini 2.5-Flash ارسال می‌شوند تا توصیف دقیق و جامعی از آن‌ها تولید شود
۵. **نمایه‌سازی در Elasticsearch:** تکه‌های نهایی به همراه توصیف تصاویر و متادیتای کامل در Elasticsearch نمایه‌سازی می‌شوند

۵.۳ نتیجه‌گیری

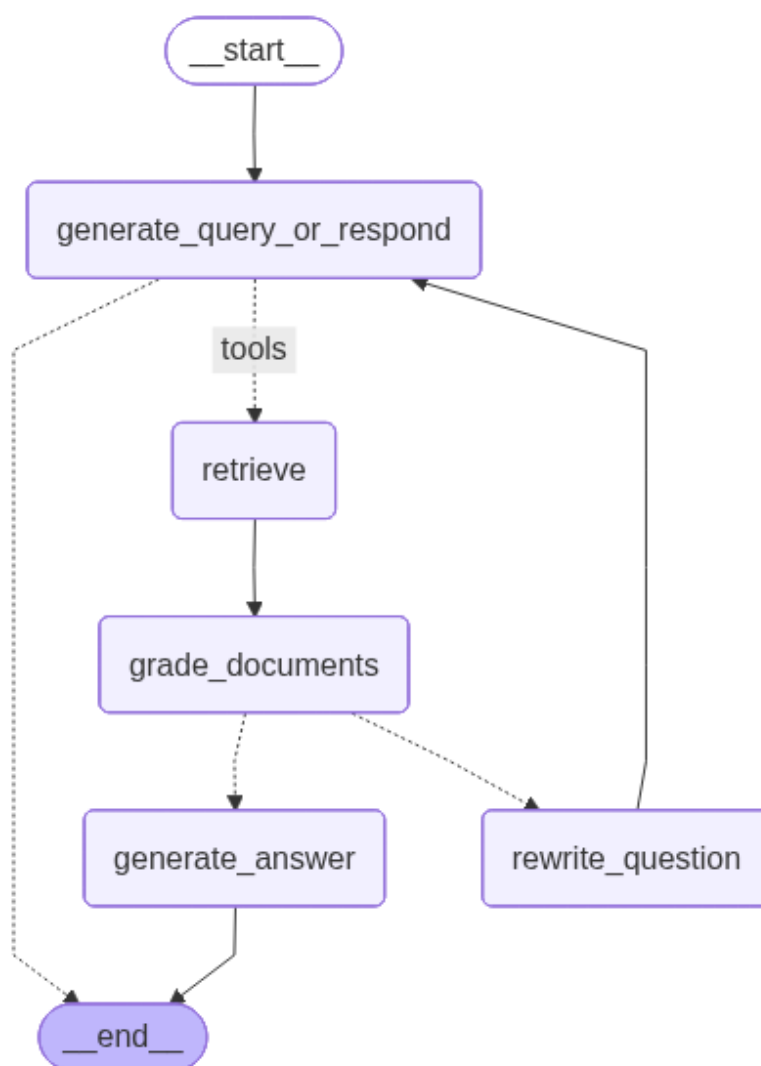
با توجه به ویژگی‌های منحصر به فرد Docling از جمله معماری مبتنی بر مدل‌های تخصصی، حفظ ساختار سلسله‌مراتبی، تکه‌سازی هوشمند، و ماهیت منبع‌باز آن، این ابزار به عنوان بهترین انتخاب برای نمایه‌سازی کتاب پزشکی در این پروژه شناسایی شد. ترکیب Docling با مدل چندوجهی Gemini برای توصیف تصاویر و Elasticsearch برای ذخیره‌سازی و بازیابی، یک سیستم جامع و کارآمد برای بازیابی اطلاعات پزشکی ایجاد می‌کند که قادر به پاسخگویی به پرسش‌های پیچیده با استفاده از اطلاعات متنی، جدولی و تصویری است.

۴ سیستم استنتاج هوشمند RAG

این بخش به تشریح سیستم استنتاج هوشمند RAG می‌پردازد که با استفاده از LangGraph برای تصمیم‌گیری خودکار و بهینه‌سازی فرایند بازیابی اطلاعات طراحی شده است. این سیستم قابلیت تشخیص زمان مناسب برای بازیابی اسناد و زمان پاسخ مستقیم را داراست و تجربه‌ای گفتگو محور و متناسب با بافت ارائه می‌دهد. مزایای کلیدی سیستم استنتاج هوشمند:

- **تصمیم‌گیری هوشمند:** استفاده از مدل زبان بزرگ برای تشخیص نیاز به بازیابی اسناد یا پاسخ مستقیم
- **ارزیابی کیفیت اسناد:** بررسی مرتبط بودن اسناد بازیابی شده قبل از تولید پاسخ
- **بهبود پرس‌وجو:** بازنویسی خودکار سؤالاتی که نتایج مرتبط ارائه نمی‌دهند
- **بازیابی تصاویر:** دریافت خودکار تصاویر مرتبط هنگام وجود ارجاع تصویر در بخش‌های متن
- **مدیریت گفتگو:** حفظ وضعیت و بافت گفتگو در طول جلسات

۱.۴ معماری گردش کار استنتاج



شکل ۹: نمونه‌ای از گردش کار استنتاج هوشمند RAG. این تصویر مراحل مختلف فرایند استنتاج را نشان می‌دهد، از جمله دریافت پرس‌وجو، تصمیم‌گیری برای بازیابی اسناد، و تولید پاسخ نهایی.

۱.۱.۴ مراحل فرایند

سیستم استنتاج هوشمند RAG دارای گردش کار مبتنی بر گراف است که شامل مراحل زیر می‌باشد:

گره‌های اصلی

- **گره عامل (Agent Node):** تصمیم‌گیری برای بازیابی اسناد یا پاسخ مستقیم
- **ابزار بازیابی (Retrieval Tool):** جستجو برای اسناد مرتبط با استفاده از جستجوی ترکیبی
- **ارزیابی اسناد (Document Grading):** بررسی میزان مرتبط بودن اسناد بازیابی شده
- **بازنویسی سؤال (Question Rewriting):** بهبود سؤالاتی که نتایج مرتبط ارائه نمی‌دهند
- **تولید پاسخ (Answer Generation):** تولید پاسخ نهایی با استفاده از بافت بازیابی شده

الگوریتم گردش کار

فرایند تصمیم‌گیری سیستم به صورت زیر عمل می‌کند:

۱. دریافت پرس‌وجوی کاربر: سیستم پیام کاربر را دریافت و تحلیل می‌کند
۲. تصمیم‌گیری اولیه: عامل تصمیم می‌گیرد که آیا نیاز به بازیابی اسناد است یا می‌تواند مستقیماً پاسخ دهد
۳. بازیابی شرطی: در صورت نیاز، ابزار بازیابی فعال شده و اسناد مرتبط جستجو می‌شوند
۴. ارزیابی کیفیت: میزان مرتبط بودن اسناد بازیابی شده با پرس‌وجو ارزیابی می‌شود
۵. حلقه بازخورد: در صورت عدم مرتبط بودن، سؤال بازنویسی شده و فرایند تکرار می‌شود

۲.۴ فرایند تولید پاسخ

حاشیه‌نویسی شناسه‌های بخش

بخش‌های بازیابی شده به همراه شناسه‌های یکتای خود (chunk_id) به مدل زبان بزرگ ارسال می‌شوند. هر بخش با فرمت زیر حاشیه‌نویسی می‌شود (شکل ۱۰). این حاشیه‌نویسی به مدل زبان بزرگ اجازه می‌دهد تا دقیقاً مشخص کند کدام بخش‌ها در تولید پاسخ استفاده شده‌اند.

```

ruments will zoom up to 100 times magnification, and mucosal detail can also be enhanced electronically so that small lesions a few millimetres in size can be seen quite clearly. The modern video endoscope image can be instantly printed out or archived digitally on a computer system.

The latest types of high- definition endoscope, processor, and monitor can also be used for two new techniques, narrowed spectrum endoscopy (narrow band imaging, Fujinon intelligent colour enhancement, and iScan) and autofluorescence imaging. Narrowed spectrum endoscopy enhances the visualization of mucosal patterns and microcapillaries in the superficial mucosa with short- wavelength (blue light) illumination using optical or digital filters. This enhances the visualization and diagnosis of otherwise poorly identifiable lesions. Autofluorescence imaging utilizes the property of short- wave light in the blue area of the spectrum to excite green autofluorescence from normal mucosa, predominantly from collagen. Thickening of the mucosa by malignant infiltration tends to inhibit this and abnormal tissue may fluoresce less, allowing for endoscopic detection of a subtle abnormality in a false colour image.

[CHUNK_ID: section_15_chunk_419]
Clinical investigation
Biopsy of an identified leucoplakia or erythroplakia of the oral mucosa is mandatory. Some studies have suggested that the use of diagnostic aids such as toluidine blue may be helpful in identifying the salient site for biopsy. The emergence of data to support the routine use of such aids has not been forthcoming to date.

[CHUNK_ID: section_15_chunk_685]
Symptoms and staging
The presentation and investigation algorithms for squamous cell carcinoma are essentially as described for adenocarcinoma. Patients may present to the ear, nose, and throat department if the dysphagia is very high or if hoarseness is a key symptom. Orolaryngoscopy may reveal the cause but flexible oesophagoscopy is usually required and should not be delayed in a patient with progressive dysphagia.
In very high- risk areas, screening programmes have been set up which generally hinge on standard endoscopy coupled with Lugol's iodine spray to highlight dysplastic mucosa for targeted biopsies. Nonendoscopic sampling methods have generally had low sensitivity and specificity due to reliance on a cytological diagnosis of atypia ( Fig. 15.7.12 ).

[CHUNK_ID: section_15_chunk_594]
Endoscopy
When investigation is required, endoscopy is the first choice as it is the only test that can give sensitive recognition and grading of oesophagitis (e.g. Los Angeles classification system) and reliable diagnosis of oesophageal columnar metaplasia (Barrett's oesophagus). Endoscopy also allows for the effective identification of significant peptic strictures, other types of oesophagitis (discussed in later sections), and peptic ulcer disease as well as malignancies. The value of endoscopy as the initial investigation is greatly enhanced by the accurate diagnosis of endoscopic biopsy and, where indicated, cytology brushings. As discussed earlier, however, most patients with reflux disease do not have endoscopically visible mucosal

```

شکل ۱۰: نمونه‌ای از پاسخ ابزار بازیابی که شامل بخش‌های حاشیه‌نویسی شده با شناسه‌های یکتا می‌باشد. این فرمت به مدل زبان بزرگ کمک می‌کند تا منابع استفاده شده را به دقت مشخص کند.

تولید پاسخ ساختاریافته

مدل زبان بزرگ پاسخی ساختاریافته تولید می‌کند که شامل دو بخش است:

- **پاسخ (answer):** متن پاسخ نهایی برای کاربر
- **شناسه‌های استفاده شده (chunk_ids_used):** فهرست شناسه‌های بخش‌هایی که در تولید پاسخ مورد استفاده قرار گرفته‌اند

نمونه‌ای از پاسخ ساختاریافته تولید شده توسط مدل:

```
{
  "answer": "Narrow band imaging, a type of narrowed spectrum endoscopy, uses short-wavelength (blue light) illumination with optical or digital filters. This technique enhances the visualization of mucosal patterns and microcapillaries in the superficial mucosa, which helps in better visualizing and diagnosing otherwise poorly identifiable lesions, including distinguishing columnar from squamous mucosa.",
  "chunk_ids_used": [
    "section_15_chunk_101",
    "section_15_chunk_620"
  ]
}
```

این ساختار به سیستم اجازه می‌دهد تا دقیقاً بخش‌های استفاده شده را شناسایی و متادیتای مرتبط را استخراج کند.

۳.۴ استخراج متادیتا و بازیابی تصاویر

پس از دریافت پاسخ از مدل زبان بزرگ، سیستم فرایند استخراج تصاویر مرتبط را آغاز می‌کند:

الگوریتم استخراج تصاویر

۱. **شناسایی بخش‌های استفاده شده:** بر اساس chunk_ids_used، اشیاء دقیق بخش‌ها از لیست بازیابی شده انتخاب می‌شوند
 ۲. **بررسی متادیتا:** برای هر بخش، فیلد meta.doc_items در متادیتا بررسی می‌شود
 ۳. **شناسایی تصاویر والد:** عناصری با برچسب "label": "picture" شناسایی می‌شوند
 ۴. **استخراج شناسه تصویر:** از فیلد self_ref شناسه تصویر استخراج می‌شود
 ۵. **بازیابی تصویر کامل:** تصویر کامل به همراه توضیحات تولید شده توسط مدل چندوجهی از مخزن اسناد بازیابی می‌شود
 ۶. **الحاق به پاسخ:** تصاویر به پاسخ نهایی عامل الحاق می‌شوند
- این فرایند تضمین می‌کند که تمامی اطلاعات بصری مرتبط با پاسخ، به همراه متن، در اختیار کاربر قرار گیرد.

۴.۴ پیاده‌سازی لایه رابط کاربری

در لایه Framework، یک ربات تلگرام پیاده‌سازی شده است که رابط کاربری تعاملی برای سیستم فراهم می‌آورد:

مدیریت بخش‌ها در ربات تلگرام

۱. **ذخیره‌سازی بخش‌ها:** پس از دریافت پاسخ از عامل، بخش‌های استفاده شده در حافظه موقت ذخیره می‌شوند
 ۲. **تولید دکمه‌های تعاملی:** برای هر `chunk_id`، یک دکمه Inline Keyboard در پیام تلگرام ایجاد می‌شود
 ۳. **پردازش کلیک کاربر:** هنگامی که کاربر روی دکمه شناسه بخش کلیک می‌کند، سیستم محتوای کامل بخش را از حافظه می‌خواند
 ۴. **نمایش جزئیات:** محتوای بخش به همراه متادیتای مرتبط برای کاربر نمایش داده می‌شود
- این رویکرد به کاربران اجازه می‌دهد تا شفافیت کاملی نسبت به منابع استفاده شده در تولید پاسخ داشته باشند و در صورت نیاز، محتوای اصلی بخش‌ها را مشاهده کنند.

۵.۴ نتیجه‌گیری

سیستم استنتاج هوشمند RAG با استفاده از LangGraph قابلیت‌های پیشرفته‌ای برای تصمیم‌گیری خودکار و بهینه‌سازی فرایند بازیابی اطلاعات فراهم می‌آورد. این رویکرد نه تنها کارایی سیستم را بهبود می‌بخشد بلکه تجربه کاربری بهتری نیز ارائه می‌دهد. ترکیب تصمیم‌گیری هوشمند، ارزیابی کیفیت اسناد، و بازنویسی پرس‌وجو منجر به سیستمی قدرتمند و انطباق‌پذیر شده است که می‌تواند در کاربردهای مختلف پردازش اسناد پزشکی مورد استفاده قرار گیرد.

References

References