**Objective of project:**

Create a RAG solution that takes in custom documents and allows users to query in those documents

**Features**

Application will cater to three domains – AI education, High school counseling, Medical Pediatrics and will allow experts to onboard in one of the three domains. Experts will provide context about their background when they onboard. Each domain will contain a default vector id using a default set of documents that have been uploaded in the backend and. When experts onboard, they will specific whether they intend to use the default vector id pertaining to their domain or want to customize and create a new vector id. When they choose to create a new vector id, then they will choose to upload their documents and also choose from the default documents available for that domain. When users post a query for that expert in the application, the query will get appended with the expert’s context that was provided when expert onboarded and relevant responses will be fetched from the vector id.

**APIs to create in the backend – Use python language**

1. Create expert
   1. expert name, id, domain,
   2. context of expert
   3. default vector id based on domain
   4. preferred vector id for expert
2. Update context
   1. Expert name, id,
   2. New context
3. Create document
   1. Get document link from local disk or web link
   2. Assign document ID and Document Name
   3. Parses those documents into ‘md’ format using llama parse apis
   4. Store the markdown version of document it in Supabase storage
   5. Gets expert id of the expert uploading the document
   6. Store expert’s domain against the document
   7. Get preference from expert – should document be included in ‘default’ domain vector id ?
   8. Get any client names for whom the document is specific for ?
4. Create vector ID for domain
   1. Take domain name as input and match documents from documents table that are marked as default for that domain and create a vector id after indexing using filesearch API of OpenAI
   2. Update this in the domain table’s ‘default vector id’ against the domain name
   3. Update this Open AI vector id against all experts (‘default vector id’ column of expert table) who correspond to the respective domain.
5. Create vector ID for expert
   1. Take an entry to ‘Use default domain knowledge’ checkbox from UI and also a list of document names / ids provided by expert and the expert name / id and create a vector id after indexing using filesearch API of OpenAI. If the ‘Use default domain knowledge’ is true, then just take the vector id from the domain of the expert and populate it than creating a new vector id.
   2. Update this Open AI vector id in the Expert table’s ‘preferred vector id’ column
6. Respond to query
   1. Take user query, expert name and augment with expert’s context and preferred vector id to respond to query
7. Delete vector id
   1. Take expert name, vector id and delete the preferred vector id entry in the expert table. Do not delete default vector id column.

**Supabase schema**

Domain

* Id – uuid – mandatory – primary key
* Domain name – Enum (AI education, High School counseling, Medical pediatrics) - mandatory
* Default vector id – text - optional
* Expert name [] – list of text - optional

Expert table:

* Name – text - mandatory
* Id – uuid – mandatory – primary key
* Domain – Enum – foreign key linked to domain table
* Context of expert – text - mandatory
* Default vector id – text – optional
* Preferred vector id – text - optional

Documents table

* Id – uuid – mandatory
* Name – text - mandatory
* Document link (Supabase storage / web link parse) – text - mandatory
* Created by (expert name / default) – text - optional
* Domain – Enum – foreign key linked to domain table
* Included in default vector (yes/No) – Boolean - mandatory
* Client name – text – optional

**UI interface – use streamlit**

Create following interfaces

1. Expert entry screen
   1. Expert name (free text)
   2. Domain (drop down from domain table enum)
   3. Expert context (free text)
   4. ‘Create expert’ button that calls ‘Create expert’ API
2. Update expert context screen
   1. Drop down to select expert name (populate from expert table)
   2. New context (free text entry)
   3. ‘Update context’ button that calls ‘Update context’ API
3. Create domain memory
   1. Drop down to select domain memory (populate from domain table)
   2. Box to enter document links
   3. ‘Submit’ calls the ‘Create vector id for domain’ API
4. Create expert memory
   1. Drop down to select expert name (populate from expert table)
   2. Checkbox for ‘Use default domain knowledge’ – this is selected by default
   3. If ‘Use default domain knowledge’ is unchecked – then show option to provide links to documents from the web (free text entry where each line is a separate document link)
   4. Also list default documents for that expert’s domain to select
   5. ‘Submit’ button that calls ‘create document’ API and also ‘Create vector id for expert’ API
5. Query screen
   1. Select expert from drop down
   2. Show Chat interface
   3. Chat with expert
   4. Use ‘Respond to query’ API
6. Delete expert memory screen
   1. Drop down to select expert name (populate from expert table)
   2. Drop down to select from all vector ids from ‘preferred vector id for expert’ column for that expert
   3. ‘Delete’ button calls the ‘Delete vecor id’ API

**Credentials to create in the .env file**

Supabase\_URL: <https://uhaaekwsvhiclnulcqob.supabase.co>

ServiceRole\_Key: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6InVoYWFla3dzdmhpY2xudWxjcW9iIiwicm9sZSI6InNlcnZpY2Vfcm9sZSIsImlhdCI6MTc1MzQzNjA0MCwiZXhwIjoyMDY5MDEyMDQwfQ.02W0MfYzAGzCqw1SKxDBXtiJjN9\_mB0y58Yd9RtrjtY

OpenAI\_Key: sk-proj-F2MKrnXa40zoWS9iyX8PFs83DTvowm1U7Lijp24Nx6UFCPq1YMUo6t9WseJTmLBICoGDDbmUn0T3BlbkFJBtfvKnQPHdYS0cn5EYJnkyGN5C9Os8CSbGEnCECQ\_Ko4U4rxHNvRKXb2LVqzI-zLBVVnfHhC4A

LlamaParse\_Key: llx-TTIq596RAKbTtqGlCiUOkqTEVPtnK1fTq1LMGzFXoYNInHaf