**Multi-Agentic Health Assistant**

Requirements & Blueprint

**Mental Health Agent LLM:**

* DeepSeek V3, 671B parameters
* Daily check-ins and emotional support+motivation
* Guided journaling to store flagged conversations and store chat summaries
* Sentiment analysis and emotion classification as well as activity an fitness progression logging
* Secure memory log for past conversations

**Diet LLM:**

* LLaMA 3.2, 11B parameters
* Vision model integration to analyse meal photos
* Nutrient breakdown and diet goal comparison
* Just use API, nutrient breakdown is required as well after picture submission
* “analyse nutrient breakdown and the meal photos to provide an analysis” use as caption with every picture submitted
* Give output to the text exercise model for it to use along with the user information for providing, meal planning and nutritional suggestions

**Exercise LLM:**

* DeepSeek V3, 671B parameters
* Generate personalized routines
* Track workout completion and calories burned (Store in the database)
* Adjusts difficulty and goals weekly

**CHATBOT:**

* Use M2-BERT for embeddings
* Integrates all 3 agents into 1 chatbot that automates all of the workflow
* The chatbot can take text, picture and audio input and work accordingly
* According to the software the chatbot should start conversations with the mental health agent, then proceed to ask for picture input, then use diet model and proceed to handle work to exercise model
* Once user is at exercise model, according to the topic of the users, conversations, switch models automatically

**Frontend:**

Streamlit site should have 5 pages:

1. Login
2. Signup
3. User-Information View and Changes
4. Progress Tracker Viewer
5. Main Chatbot

**Database(s):**

1. PostgreSQL

**user\_health\_info:**

id is a primary key and a foreign key used across all tables to track the activity of user across them all

user\_profile:

* user\_info(class with the following attributes, not to be stored as json or anything)

1. name (String)
2. age (4 digit, 1 after decimal)
3. gender (Female, Male are the options, default = Female)
4. height (meters, 4 digit, 3 after decimal, default = 1.7)
5. weight (kgs, 6 digit, 3 after decimal, default =66.4)

* fitness\_goal (90 character max String, default = 'Get into better shape')
* diet\_pref (7 options, 'vegan', 'carnivore', 'both', 'balanced', 'vegetarian', 'pescatarian', 'any', default= any)
* daily\_availibility (class with the following attributes, not to be stored as json or anything)

1. time\_arr   
   2d array, x=3, y=2 time storage arrrays take a value in 24 hour format and minutes, take 1-3 time ranges, which are at max 20 minutes apart, like 11:20 and 11:40 or 23:40 and 00:00, if user enters a time value, compute the 20 minute later value and insert it. Example of a full array:  
   11:23 23:46 19:59  
   11:43 23:59 20:09  
   Example of a default array:  
   12:00 NULL NULL  
   12:20 NULL NULL  
   Example of an array where only 1 time value is given by user:  
   13:50 NULL NULL  
   14:10 NULL NULL(calculate next value 20 minutes apart)

* mental\_health\_background (optional, default = null)
* medical\_conditions(if any, then a string with detail regarding it, default is null )

daily\_stats:

* activity\_level (4 options, not active, lightly active, active, very active, default = active)
* todays\_flag(boolean, default value= false)
* progress\_condition(has only 3 values(positive, negative, neutral))
* days\_done(integer, incremented every day, default=0)
* days\_left( integer, value calculated every day = time\_deadline-days\_done)

other\_storage:

* picture\_analysis (Food Picture Analysis, maximum 1000 character string, default="")
* audio\_transcript (Temporary Audio Input Transcription, maximum 3 minute audio recording allowed, maximum 3000 character string, default=""(meaning nothing))

1. Vector Database - Pinecone

**Non-Functional Requirements:**

1. Memory: Keep older memory and summaries of old chats in the database, manage context, this keeps up long term memory
2. Input: Integrate Video input as well if possible