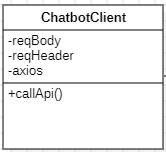
**Chatbot Backend:**

The chatbot backend API endpoint is a REST API handler which has the following class organization

**Chatbot Client**



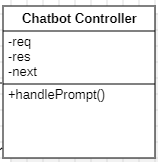
**Attributes**

* **reqBody** holds the API request payload required by the controller
* **reqHeader** holds the API request header required by the controller
* **axios** is a JS library object which has http methods to call the API.

**Methods**

* **callApi()** does the business logic processing and gathers the body and header and calls the API.

**Chatbot Controller**



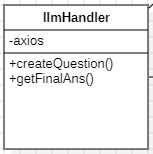
**Attributes**

* **req** is the request parameter object received by the express api controller. It is used to extract body, params, header and other request related information.
* **res** is the response parameter object used to return the response and status to the client
* **axios** is the next function used to call the next middleware in the express chain.

**Methods**

* **handlePrompt()** is performs the API controller business logic processing and based on the request and returns the response.

**LlmHandler**



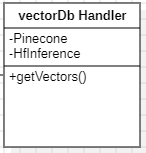
**Attributes**

* **axios** is a JS library object which has http methods to call the API.

**Methods**

* **createQuestion()** calls the LLAMA3.1 LLM API via axios and creates a standalone question based on the user prompt.
* **getFinalAns()** calls the LLAMA3.1 LLM API via axios with the user prompt and the context data to get the final answer.

**VectorDbHandler**



**Attributes**

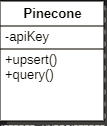
* **Pinecone** is a vector database client which is in the form of an object that lets you store and query vector dataset (embeddings).
* **HfInference** is a hugging face model which is imported in the form of an object

**Methods**

* **getVectors**: is a function which is used to get top three similar chunks of vectors from the database based on the standalone question provided by llmHandler’s createQuestion method. These vectors are then utilized by llmHandler’s getFinalAns method.

**Pinecone**

Pinecone is a vector database that makes it easy to add similarity search to any application.



**Attributes**

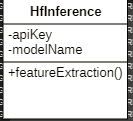
* **apiKey** is a secret string to access the pinecone client

**Methods**

* **upsert** is a built-in pinecone function used to insert vector dataset into the database.
* **query** is a built-in pinecone function used to retrieve similar chunks of data from the vector db.

**HfInference**

Hugging face’s Serverless Inference API offers a fast and free way to explore thousands of models for a variety of tasks.



**Attributes**

* **apiKey** is a secret string to access the pinecone client
* **modelName** represents the name of the model that is to be used.

**Methods**

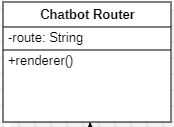
* **featureExtraction** is the a built in HfInference function used to convert text into embeddings.

**Chatbot Frontend**

Chatbot frontend is the UI part of the chatbot which interacts with the backend to implement chatting functionality.

**Chatbot Router**

Chatbot route is the class which displays the Chatbot Page on a specified route.



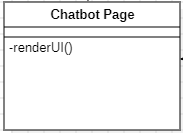
**Attributes**

* **route** is the URL path on which the page is to be displayed.

**Methods**

* **renderer** returns the ChatbotPage component.

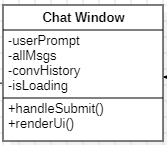
**Chatbot Page**.



**Methods**

* **renderUi** performs the styling and renders the Chatbot Window inside itself.

**Chatbot Window**



Chatbot route is the class which displays the Chatbot Page on a specified route.

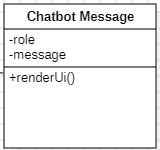
**Attributes**

* **userPrompt** is a state which holds the latest message which the user has typed.
* **allMsgs** is an array which holds the whole conversation of the current session.
* **convHistory** holds all the messages in the form of a string in a specific format.
* **isLoading** is a Boolean state which helps in the conditional rendering of the loading bar.

**Methods**

* **handleSubmit** is the function which is called when the user sends the message. Its main job is to send the API request.
* **renderUi** iteratively renders the message components.

**Chatbot Message**



**Attributes**

* **role** represents the author of the message used to determine the alignment of the message in the chat window (user’s messages are right aligned while the chatbot’s messages are left aligned).
* **message** is a string holding the message that is to be displayed in the current component.

**Methods**

* **renderUi** returns a HTML div with dynamic alignment and message.