APIGen

Platform to generate custom APIs that consists of recommended queries from your query logs.

Introduction:

Executing redundant queries repetitively can waste time, to counter this problem we have created APIGen. The main aim of our project is to analyse the existing query logs of a database and generate custom APIs that execute the recommended important queries that have been run often in the database. These APIs once run, will execute the recommended queries onto the database for efficient work.

Functionalities:

1) Support for SQL and NoSQL databases:

APIGen will access the provided query logs from both SQL and NoSQL databases, pre-process the logs and then plug these queries into our ML model.

2) Recommend APIs:

Our ML model will analyse the queries and check for the queries which have been used the most along with the tables that have been frequently accessed.

After processing it will generate custom APIs for the recommended queries given by the model. These APIs will fire the queries in the connected database whenever needed.

3) Query Analytics:

Along with the custom generated APIs, APIGen will provide several query analytics such as queries with the least processing time, queries with maximum processing time , mean query processing time.

This will help generate insights about the performance of the database and the queries which have been executed.

ML Model:

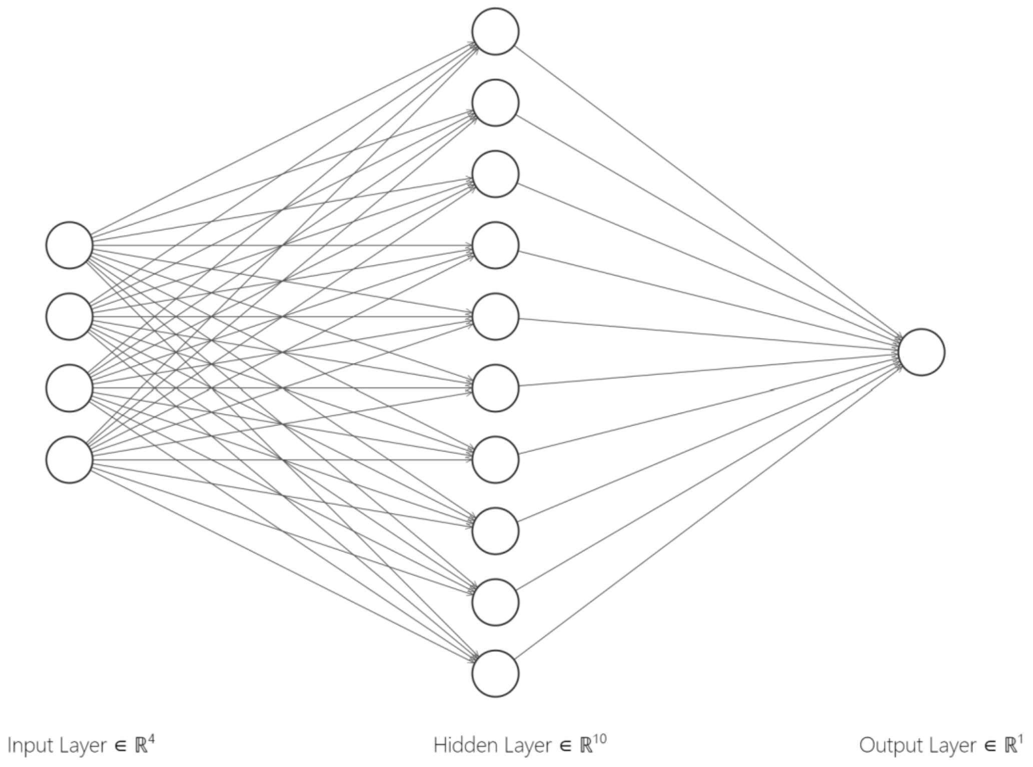
The machine learning model used here is a neural network trained to generate new queries based on a dataset of previous queries. The model is trained on a dataset of previous queries and corresponding results.

During training, the model learns to generate new queries that are similar to those in the training dataset by minimizing a predefined loss function. Once trained, the model can be used to generate new queries for a given input.

The generated queries can be executed on a database to retrieve similar results as those in the training dataset.

There are various approaches to training the model, including sequence-to-sequence models and encoder-decoder models.

The specific architecture and training approach used can vary depending on the specific application and requirements.


Input Layer ∈ $\mathbb{R}^4$
Hidden Layer ∈ $\mathbb{R}^{10}$
Output Layer ∈ $\mathbb{R}^1$

Here the first layer corresponds to the input layer where the four nodes correspond to:

1) Command type (eg: select, update etc)

2) Columns used

3) Tables used

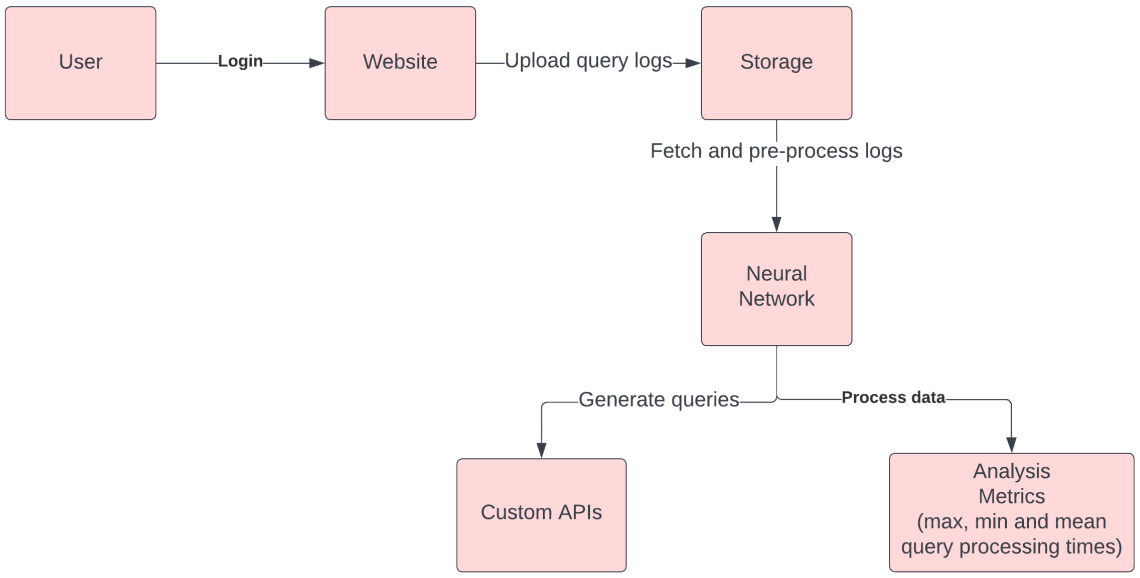
4) Conditions added

The second layer is the Hidden Layer where the computations take place.

And finally we have the output layer with one node giving us the recommended query.

Flow:

The flow of the program will look like



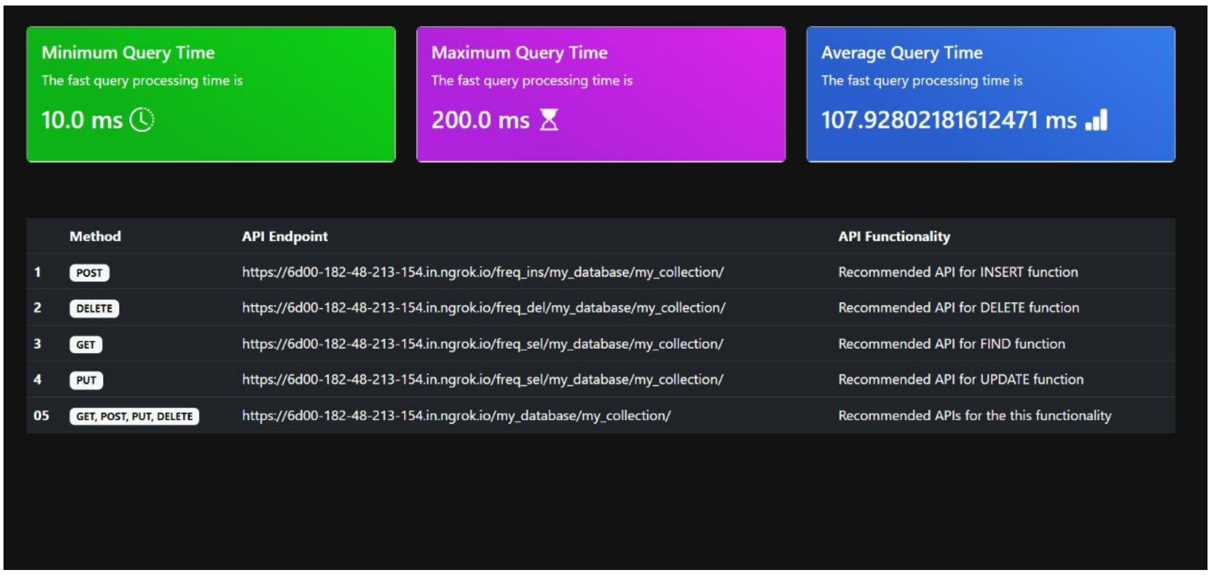
Generated APIs:

The APIs which get generated from the recommended queries have a unique endpoint and can be edited as per the user's

requirement.

The custom API will also have documentation so the user can understand what work is done by the API.

The dashboard will look like:



Minimum Query Time
The fast query processing time is
10.0 ms

Maximum Query Time
The fast query processing time is
200.0 ms

Average Query Time
The fast query processing time is
107.92802181612471 ms

	Method	API Endpoint	API Functionality
1	POST	https://6d00-182-48-213-154.in.ngrok.io/freq_ins/my_database/my_collection/	Recommended API for INSERT function
2	DELETE	https://6d00-182-48-213-154.in.ngrok.io/freq_del/my_database/my_collection/	Recommended API for DELETE function
3	GET	https://6d00-182-48-213-154.in.ngrok.io/freq_sel/my_database/my_collection/	Recommended API for FIND function
4	PUT	https://6d00-182-48-213-154.in.ngrok.io/freq_sel/my_database/my_collection/	Recommended API for UPDATE function
05	GET, POST, PUT, DELETE	https://6d00-182-48-213-154.in.ngrok.io/my_database/my_collection/	Recommended APIs for the this functionality

Analysis Metrics:

It provides some analysis metrics after analysing the query logs such as:

1) Queries taking maximum processing time and the time taken

2) Queries taking minimum processing time and the time taken

3) Mean time taken by the queries to process

This will help the user understand which queries are not effecient and time taking.