Sports Results Chatbot using Rasa

# Overview

This project is a chatbot designed to answer questions about sports results. It uses the Rasa framework to understand natural language and SQLite to store and retrieve data about sports matches.

# Installation

1. Clone the repository to your local machine.
2. Install the required dependencies by running **pip install -r requirements.txt**.
3. Create a virtual environment for the project by running **python -m venv venv**.
4. Activate the virtual environment by running **source venv/bin/activate** (Linux/Mac) or **venv\Scripts\activate** (Windows).
5. Run the Rasa Command **rasa run actions**
6. Run the Rasa Command **rasa run -m models --enable-api**
7. Run the Flask app by running **python app.py**.

# Usage

Once the Flask app is running, you can access the chatbot interface by opening a web browser and navigating to **http://localhost:5000**. The chatbot will greet you and prompt you to ask a question about a sports match. You can ask any of the following questions:

1. Which teams are playing?
2. What is the current score?
3. Who are the goal scorers?
4. Where is the match being played?
5. When is the match?
6. What time is the match?
7. How many goals were scored in the first half?
8. How many goals were scored in the second half?
9. How many yellow cards were given?
10. How many red cards were given?

The chatbot will use natural language processing to understand your question and retrieve the appropriate information from the SQLite database. It will then provide a response to your question.

# File Structure

* app.py: The main Flask application file.
* actions.py: Contains the ActionGetSportsData class, which retrieves data from the SQLite database.
* config.yml: Rasa configuration file.
* credentials.yml: Rasa credentials file.
* data/: Contains the Rasa training data.
* domain.yml: Rasa domain file, which defines intents, entities, and responses.
* models/: Contains trained Rasa models.
* requirements.txt: List of required Python packages.
* templates/: Contains HTML templates for the Flask app.

# Limitations

The chatbot is currently only able to answer questions about sports matches that are stored in the SQLite database. It is not capable of answering questions about live matches or matches that are not stored in the database.

# Future Work

1. Add functionality to allow users to ask about live sports matches.
2. Expand the range of questions that the chatbot can answer.
3. Improve the accuracy of the natural language processing by training the model on more data.
4. Improve the user interface of the Flask app to make it more visually appealing and user-friendly.

# Contributors

* Ammar, MHD Hamza: Developer