BROCK UNIVERSITY
FACULTY OF MATHEMATICS AND SCIENCE

COSC 4P02 - CANADA GAMES CHATBOT

By:
Eddy Su (6459705)
Aman Braich (6511679)
Manvendrasinh Rana (6137228)
Rikveet Singh Hayer (6590327)
Sager Kudrick (5919170)
Sawyer Fenwick (6005011)
Raghav Bhardwaj (6548580)

TEAM UNDEFINED PROGRESS REPORT

DEPARTMENT OF COMPUTER SCIENCE
ST. CATHARINES, ONTARIO, CANADA

February 28, 2022

TABLE OF CONTENTS

1. OVERVIEW

The development of our chatbot program has been progressing at a rapid rate over the past months. Our team has utilized this time to focus on working on the chatbots database, the chatbots website, as well as the chatbots artificial intelligence. This document will present what our team has been working on as well as any results we can currently show, as well as covering how we have operated our meetings, progressed within our sprints, and any problems we have encountered. We will also cover the future of this project, and what we are planning to do. The link to our GitHub page can be found in Section 6.

1.1. Contributions

All Team Members worked together on Product Backlog, Sprint Backlogs and Reports.
All Team Members worked together to write scraping and testing code for scraping modules.
All Team Members contribute to the GitHub page, uploading their code to the appropriate branches (Front-End, Back-End, Production, Testing).
Rikveet Singh Hayer researched, designed and implemented the AI, and is our Team Leader.
Manvendrasinh Rana created the GitHub page.
Sager Kudrick and Sawyer Fenwick designed and implemented the MySQL Database.
Raghav Bhardwaj and Aman Braich designed and implemented the UI on desktop and mobile.

2. DESIGN

2.1. Website

Figures 2.1-2.2 shows the inital design of our website UI.

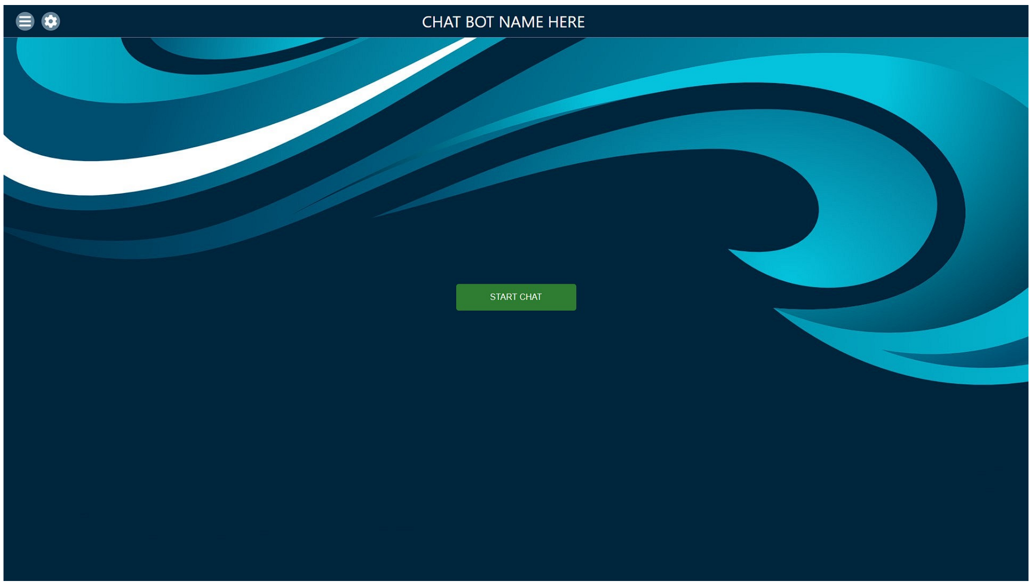


Figure 2.1: Chat Start Design

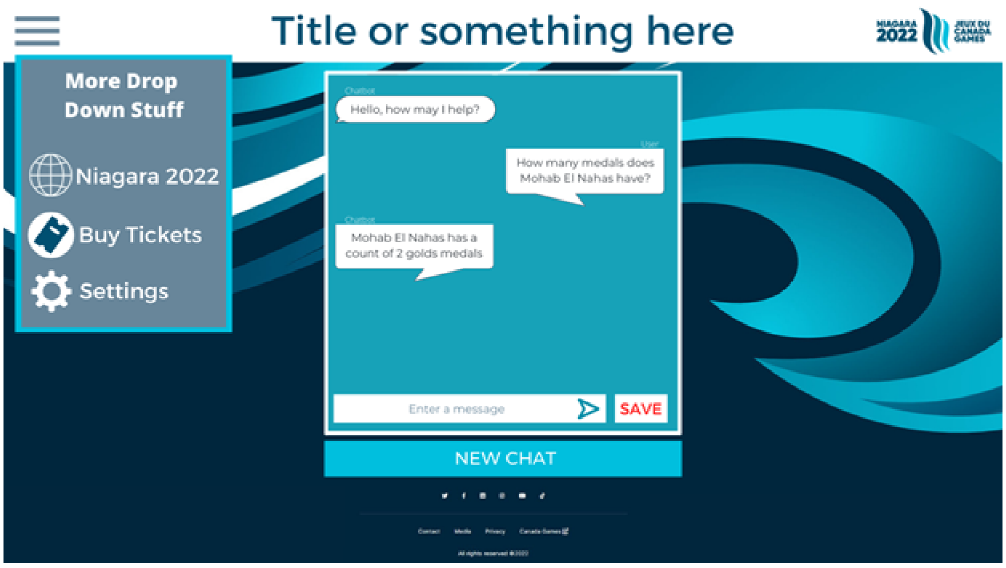


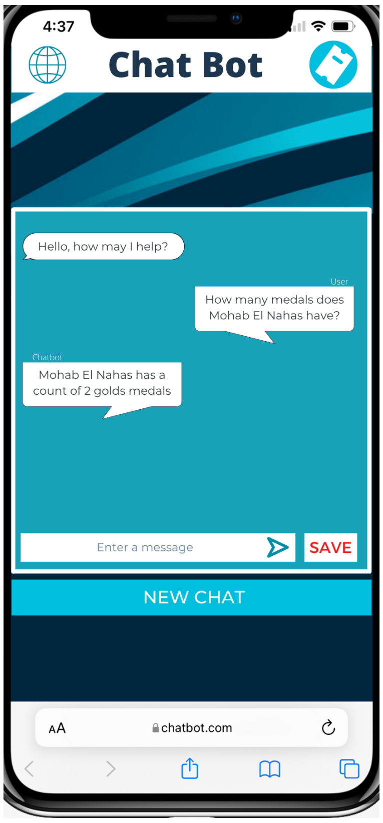
Figure 2.2: Initial Website Mockup

For the site mockup, we decided to use the online design and publishing tool Canva. With that, we were able to add new ideas and change existing ones to our liking. When thinking of the initial design we wanted it to be a seamless transition for users from the Niagra Canada Games site and the chatbot.

We decided to go with a similar color scheme as the Niagra Canada Games site. We followed the minimalistic look and easy navigation of the site. This is where we got the idea for the solid blue head with a drop-down menu with various options for the users.

2.2. Mobile

Figures 2.3-2.5 shows the development of our mobile UI design.

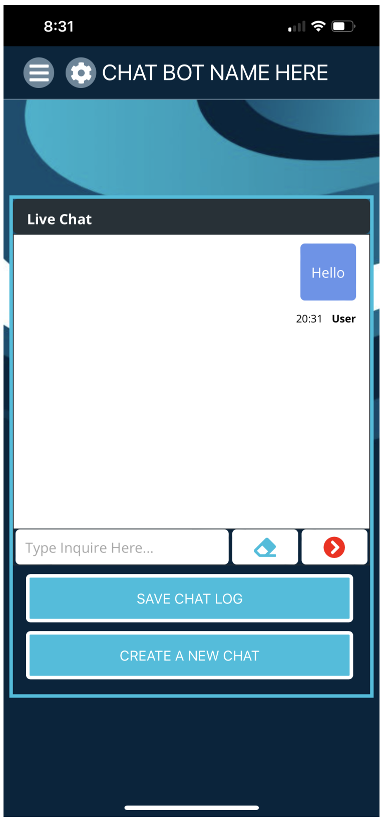


Hello, how may I help?

User
How many medals does Mohab El Nahas have?

Chatbot
Mohab El Nahas has a count of 2 golds medals

Figure 2.3: Initial Mobile Mockup



Hello
20:31 User

Figure 2.4: Chat UI

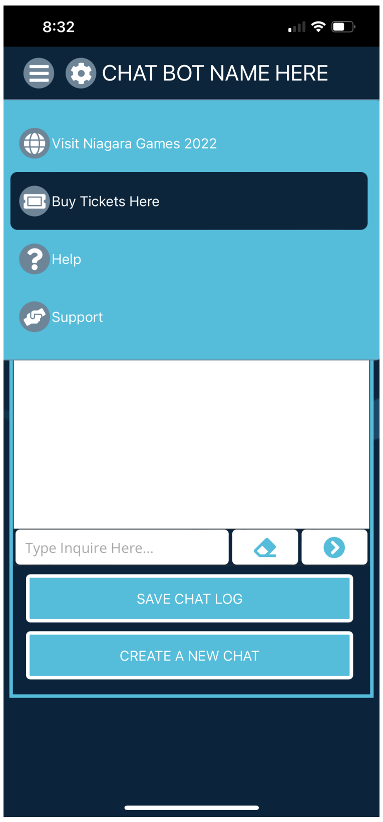
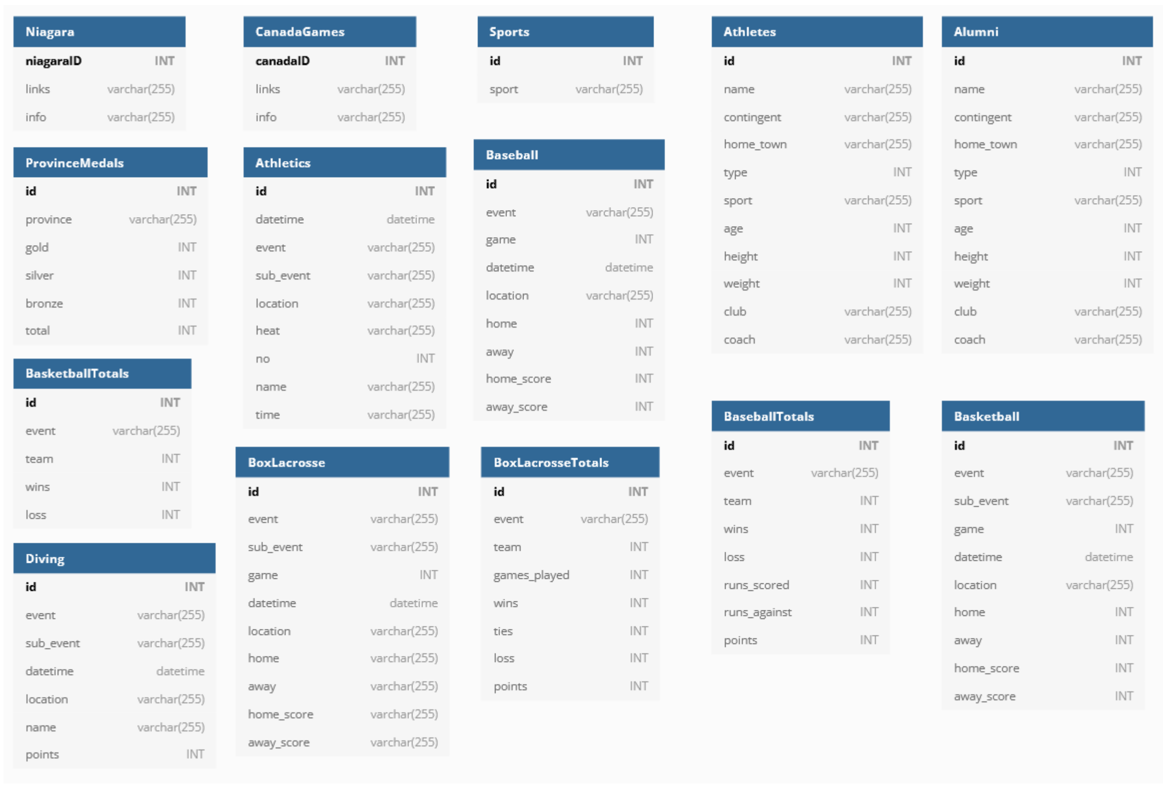


Figure 2.5: Drop Down Menu

2.3. Database

We modelled our Database off the Canada Games website. Each Sport has it's own table, with columns associated to that particular sport since not every sport has all the same attributes, most sports are not scored the same way, or they have different numbers of participants (individual events, duos, and teams vs teams), as some examples. Aside from the Sport Tables, we also have a general Canada Games table, which will contain links to the appropriate website for general questions, and a Niagara Table which will serve the same purpose but for local information (bus routes, things to do in the region etc.). We also have a table to keep track of the total medals a Province has accumulated over the games. These tables will satisfy any query the end user could make about each sport, and conforms to potential queries about the games or region in general per the SRS. Figures 2.6 and 2.7 show our Database UML.



Niagara	
niagaraID	INT
links	varchar(255)
info	varchar(255)

CanadaGames	
canadaID	INT
links	varchar(255)
info	varchar(255)

Sports	
id	INT
sport	varchar(255)

Athletes	
id	INT
name	varchar(255)
contingent	varchar(255)
home_town	varchar(255)
type	INT
sport	varchar(255)
age	INT
height	INT
weight	INT
club	varchar(255)
coach	varchar(255)

Alumni	
id	INT
name	varchar(255)
contingent	varchar(255)
home_town	varchar(255)
type	INT
sport	varchar(255)
age	INT
height	INT
weight	INT
club	varchar(255)
coach	varchar(255)

ProvinceMedals	
id	INT
province	varchar(255)
gold	INT
silver	INT
bronze	INT
total	INT

Athletics	
id	INT
datetime	datetime
event	varchar(255)
sub_event	varchar(255)
location	varchar(255)
heat	varchar(255)
no	INT
name	varchar(255)
time	varchar(255)

Baseball	
id	INT
event	varchar(255)
game	INT
datetime	datetime
location	varchar(255)
home	INT
away	INT
home_score	INT
away_score	INT

BasketballTotals	
id	INT
event	varchar(255)
team	INT
wins	INT
loss	INT

BaseballTotals	
id	INT
event	varchar(255)
team	INT
wins	INT
loss	INT
runs_scored	INT
runs_against	INT
points	INT

Basketball	
id	INT
event	varchar(255)
sub_event	varchar(255)
game	INT
datetime	datetime
location	varchar(255)
home	INT
away	INT
home_score	INT
away_score	INT

BoxLacrosse	
id	INT
event	varchar(255)
sub_event	varchar(255)
game	INT
datetime	datetime
location	varchar(255)
home	varchar(255)
away	varchar(255)
home_score	varchar(255)
away_score	varchar(255)

BoxLacrosseTotals	
id	INT
event	varchar(255)
team	INT
games_played	INT
wins	INT
ties	INT
loss	INT
points	INT

Diving	
id	INT
event	varchar(255)
sub_event	varchar(255)
datetime	datetime
location	varchar(255)
name	varchar(255)
points	INT

Figure 2.6: Database UML

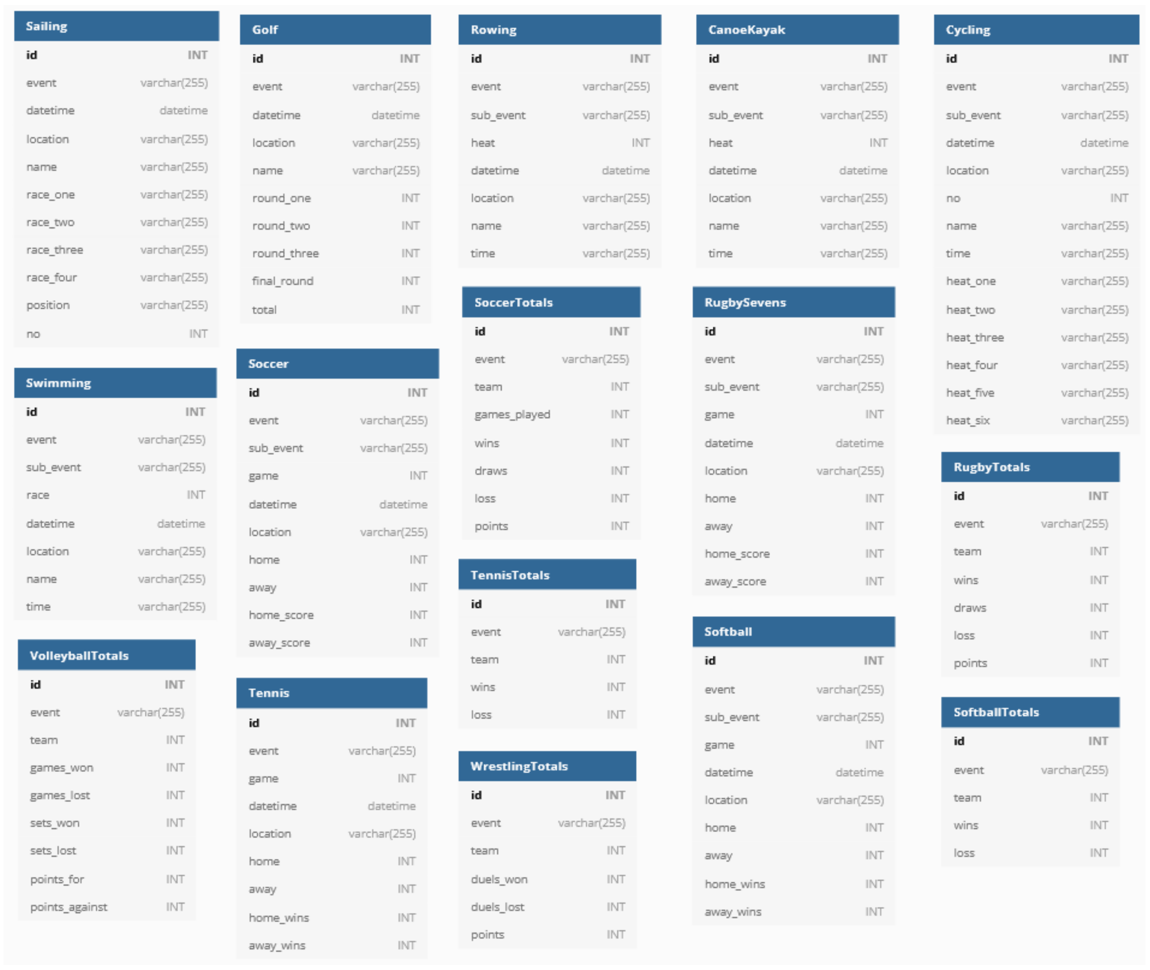


Figure 2.7: Database UML Cont.

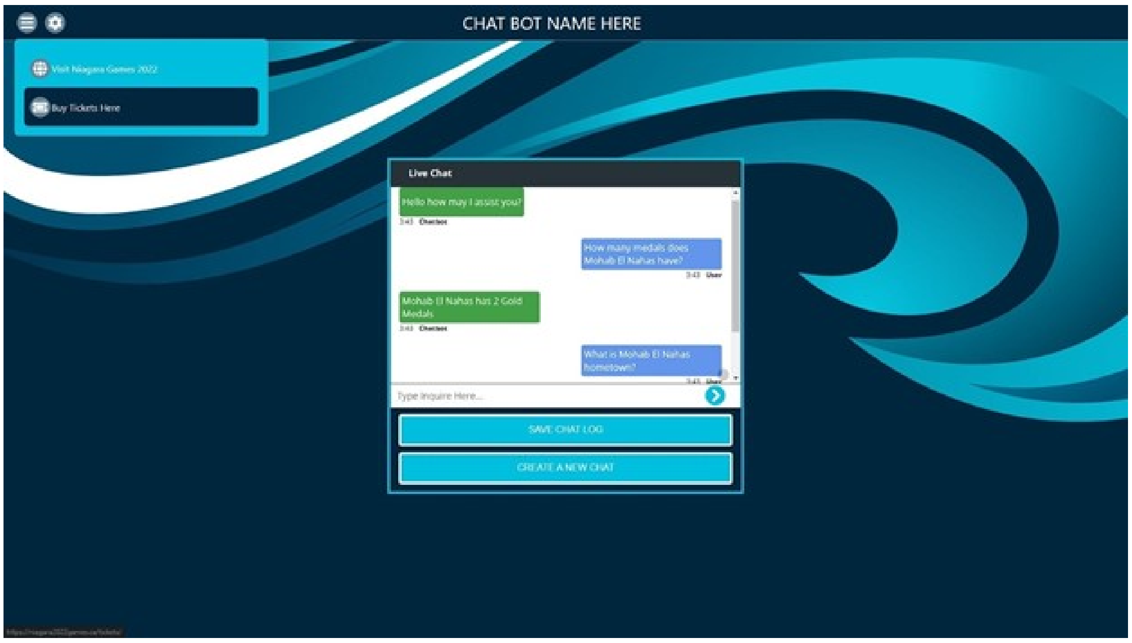
2.4. AI

The AI model was decided to be a tabular Q&A type system. In this system the AI's input is the query and the table it self and the output is the column/columns as a response to the question. These columns are ranked based on the correlation between table name, column header name and the value in the column. Further more based on the query the AI can return results as AVG and COUNT.

3. IMPLEMENTATION

3.1. Website

For our website we have decided to use the React framework. By using the react framework, we can quickly and easily create interactive applications for web platforms. The ease of use and robustness of this framework heralds it as one of the most popular web framework, being utilized by an exceptionally significant percentage of web applications- and utilized by most fortune 500 companies. We have already gone over the mockup of the website and the design we sought to create, and in this section, we will preview the implementation of the website/front-end, rendered in React.JS.



Hello how may I assist you?
3:43 Chatbot

How many medals does Mohab El Nahas have?
3:43 User

Mohab El Nahas has 2 Gold Medals
3:43 Chatbot

What is Mohab El Nahas hometown?
3:43 User

Figure 3.1: Website UI Implementation

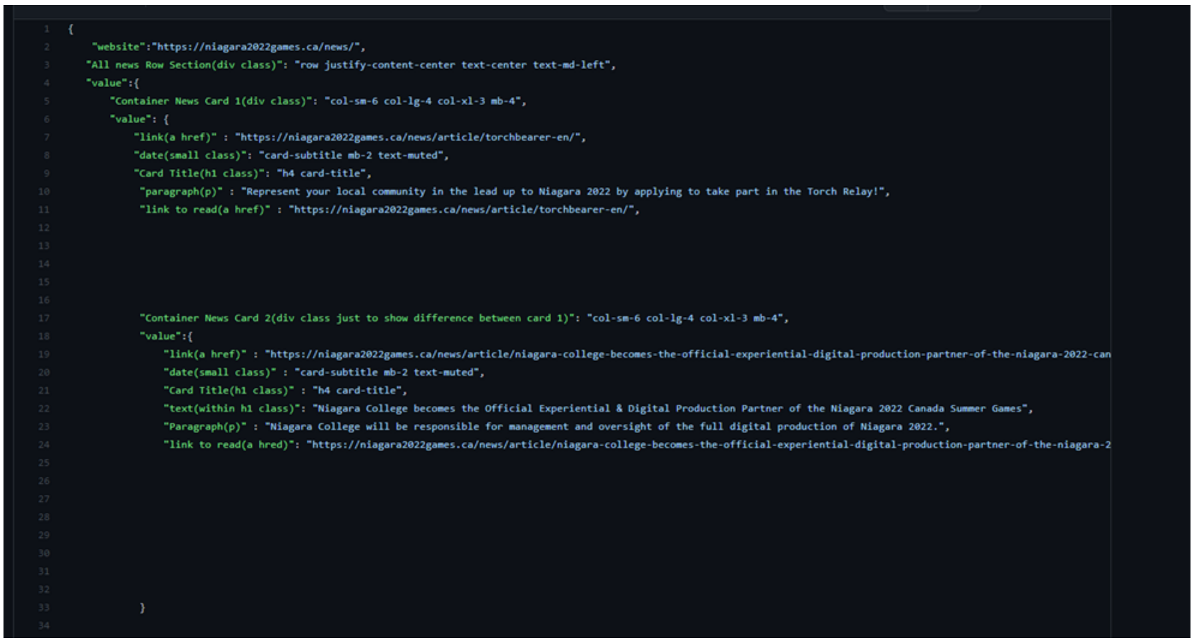
3.2. Database

For our database we have opted to use MySQL for how effective it is, and how easily we can integrate MySQL and React.JS. We have already covered the design of our database, modeling it similarly to how we imagined tables would exist and interact on the original Canada Games website.

3.3. Scraping

For our scraping framework, we opted to use Selenium. Because of the structure of some of the pages, you can only get a complete view of the data by using inputs and pressing a “Find” button. Selenium is an extremely powerful tool that will create an instance of a browser, allowing you to interact with a webpage as well as view the contents of it like any regular scraper.

3.3.1. Mappings



```
{
    "website":"https://niagara2022games.ca/news/",
   "All news Row Section(div class)": "row justify-content-center text-center text-md-left",
   "value":{
       "Container News Card 1(div class)": "col-sm-6 col-lg-4 col-xl-3 mb-4",
       "value": {
           "link(a href)" : "https://niagara2022games.ca/news/article/torchbearer-en/",
           "date(small class)": "card-subtitle mb-2 text-muted",
           "Card Title(h1 class)": "h4 card-title",
            "paragraph(p)" : "Represent your local community in the lead up to Niagara 2022 by applying to take part in the Torch Relay!",
            "link to read(a href)" : "https://niagara2022games.ca/news/article/torchbearer-en/",





            "Container News Card 2(div class just to show difference between card 1)": "col-sm-6 col-lg-4 col-xl-3 mb-4",
            "value":{
                "link(a href)" : "https://niagara2022games.ca/news/article/niagara-college-becomes-the-official-experiential-digital-production-partner-of-the-niagara-2022-can
                "date(small class)" : "card-subtitle mb-2 text-muted",
                "Card Title(h1 class)" : "h4 card-title",
                "text(within h1 class)": "Niagara College becomes the Official Experiential & Digital Production Partner of the Niagara 2022 Canada Summer Games",
                "Paragraph(p)" : "Niagara College will be responsible for management and oversight of the full digital production of Niagara 2022.",
                "link to read(a hred)": "https://niagara2022games.ca/news/article/niagara-college-becomes-the-official-experiential-digital-production-partner-of-the-niagara-2









            }
```

Figure 3.2: Sample Mapping

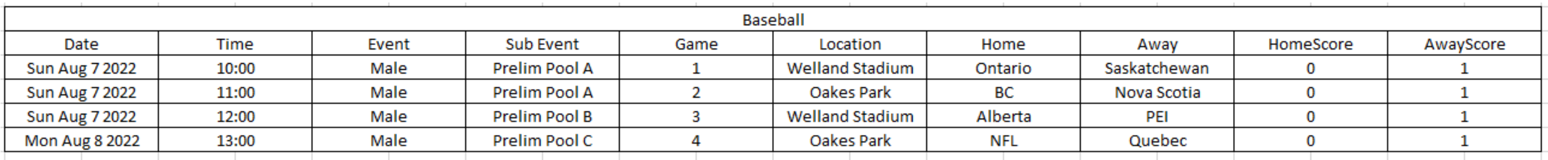
Mappings have been made for the following links:

Niagara Games 2022 News
Canada Games Gems Pro 2019 Calendar
Niagara Games 2022 Sports
Canada Games 2019 Gems Pro Teams
Canada Games 2019 Gems Pro Find Alumni
Canada Games 2017 Gems Pro Find Player

Simple mappings for relevant websites, useful for when implementation of scraping the relevant site is done, for example, we practice scraping the 2017 and 2019 sites because they contain data on players, where the 2022 version does not, but they are set up the same.

3.3.2. Sports

We scrape the sports information from each individual sport page on the site. We have made the following associations: a Sport has several events, and each event has several sub-events. For example; in Figure 3.3, Baseball is the Sport, ”Male” column is the ”Event” and ”Preliminary Pool A” is a ”Sub-Event”. This implementation will make it easy to ask questions about a specific sport, the AI simply needs to find the correct table, and run the query on that table.

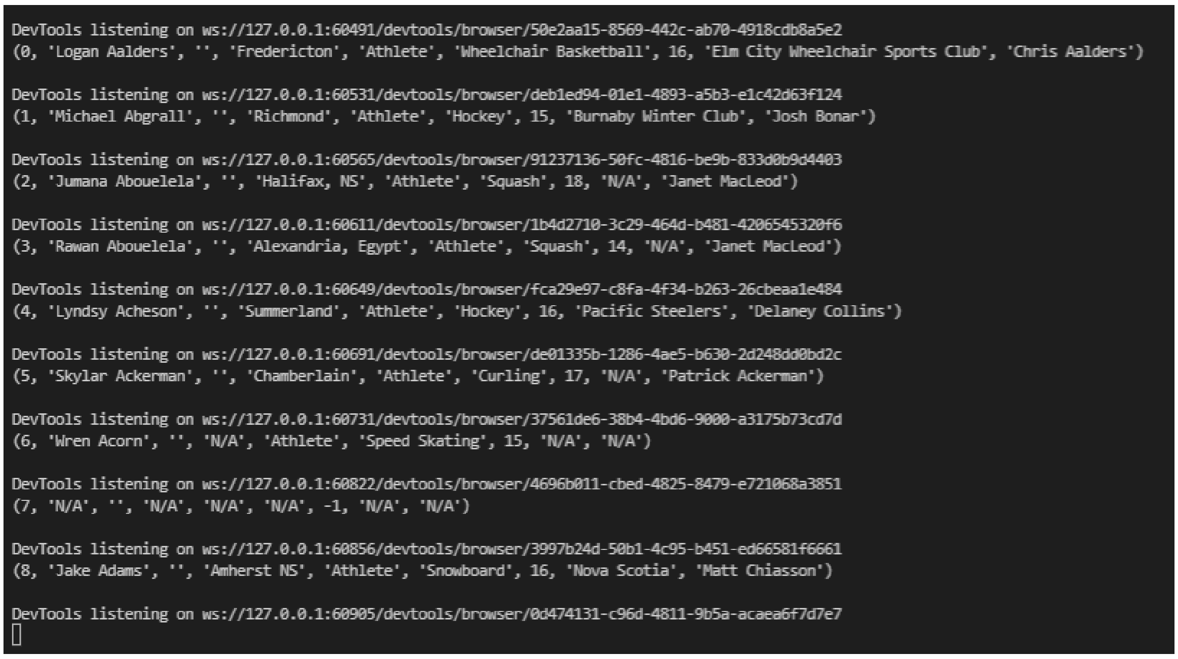


Baseball									
Date	Time	Event	Sub Event	Game	Location	Home	Away	HomeScore	AwayScore
Sun Aug 7 2022	10:00	Male	Prelim Pool A	1	Welland Stadium	Ontario	Saskatchewan	0	1
Sun Aug 7 2022	11:00	Male	Prelim Pool A	2	Oakes Park	BC	Nova Scotia	0	1
Sun Aug 7 2022	12:00	Male	Prelim Pool B	3	Welland Stadium	Alberta	PEI	0	1
Mon Aug 8 2022	13:00	Male	Prelim Pool C	4	Oakes Park	NFL	Quebec	0	1

Figure 3.3: Example Sport Table

3.3.3. Players/Alumni

When we scrape Players or Alumni, we get the full list of Players/Alumni page links from the Canada Games Site, then pass that URL to an automated scraper, that visits each URL and scrapes the necessary data off of it, and turns it into a tuple that can be easily inserted into our MySQL database. Figure 3.4 shows a sample execution.

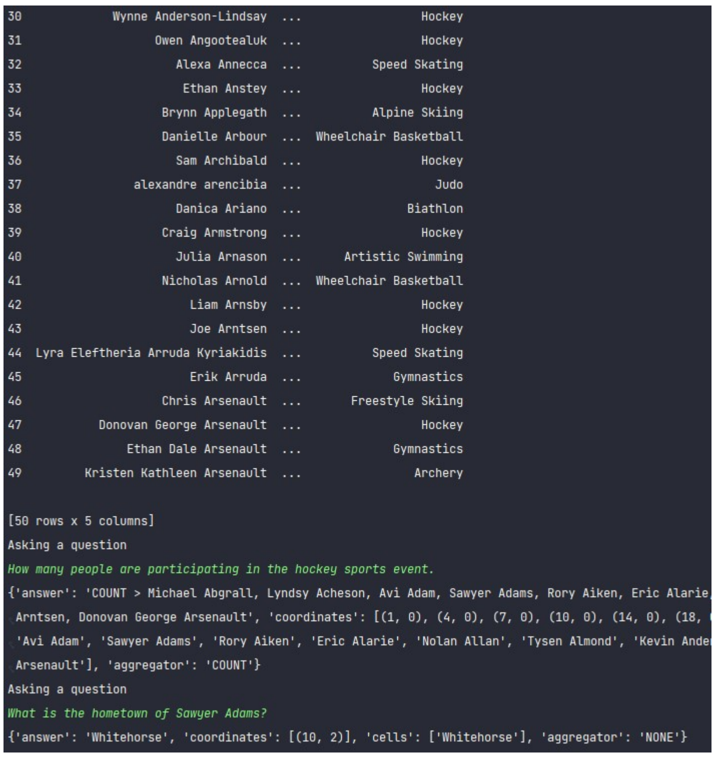


```
DevTools listening on ws://127.0.0.1:60491/devtools/browser/50e2aa15-8569-442c-ab70-4918cdb8a5e2
(0, 'Logan Aalders', '', 'Fredericton', 'Athlete', 'Wheelchair Basketball', 16, 'Elm City Wheelchair Sports Club', 'Chris Aalders')

DevTools listening on ws://127.0.0.1:60531/devtools/browser/deb1ed94-01e1-4893-a5b3-e1c42d63f124
(1, 'Michael Abgrall', '', 'Richmond', 'Athlete', 'Hockey', 15, 'Burnaby Winter Club', 'Josh Bonar')

DevTools listening on ws://127.0.0.1:60565/devtools/browser/91237136-50fc-4816-be9b-833d0b9d4403
(2, 'Jumana Abouelela', '', 'Halifax, NS', 'Athlete', 'Squash', 18, 'N/A', 'Janet MacLeod')

DevTools listening on ws://127.0.0.1:60611/devtools/browser/1b4d2710-3c29-464d-b481-4206545320f6
(3, 'Rawan Abouelela', '', 'Alexandria, Egypt', 'Athlete', 'Squash', 14, 'N/A', 'Janet MacLeod')

DevTools listening on ws://127.0.0.1:60649/devtools/browser/fca29e97-c8fa-4f34-b263-26cbeaa1e484
(4, 'Lyndsy Acheson', '', 'Summerland', 'Athlete', 'Hockey', 16, 'Pacific Steelers', 'Delaney Collins')

DevTools listening on ws://127.0.0.1:60691/devtools/browser/de01335b-1286-4ae5-b630-2d248dd0bd2c
(5, 'Skylar Ackerman', '', 'Chamberlain', 'Athlete', 'Curling', 17, 'N/A', 'Patrick Ackerman')

DevTools listening on ws://127.0.0.1:60731/devtools/browser/37561de6-38b4-4bd6-9000-a3175b73cd7d
(6, 'Wren Acorn', '', 'N/A', 'Athlete', 'Speed Skating', 15, 'N/A', 'N/A')

DevTools listening on ws://127.0.0.1:60822/devtools/browser/4696b011-cbed-4825-8479-e721068a3851
(7, 'N/A', '', 'N/A', 'N/A', 'N/A', -1, 'N/A', 'N/A')

DevTools listening on ws://127.0.0.1:60856/devtools/browser/3997b24d-50b1-4c95-b451-ed66581f6661
(8, 'Jake Adams', '', 'Amherst NS', 'Athlete', 'Snowboard', 16, 'Nova Scotia', 'Matt Chiasson')

DevTools listening on ws://127.0.0.1:60905/devtools/browser/0d474131-c96d-4811-9b5a-acaea6f7d7e7
```

Figure 3.4: Sample Execution on Player Scraping

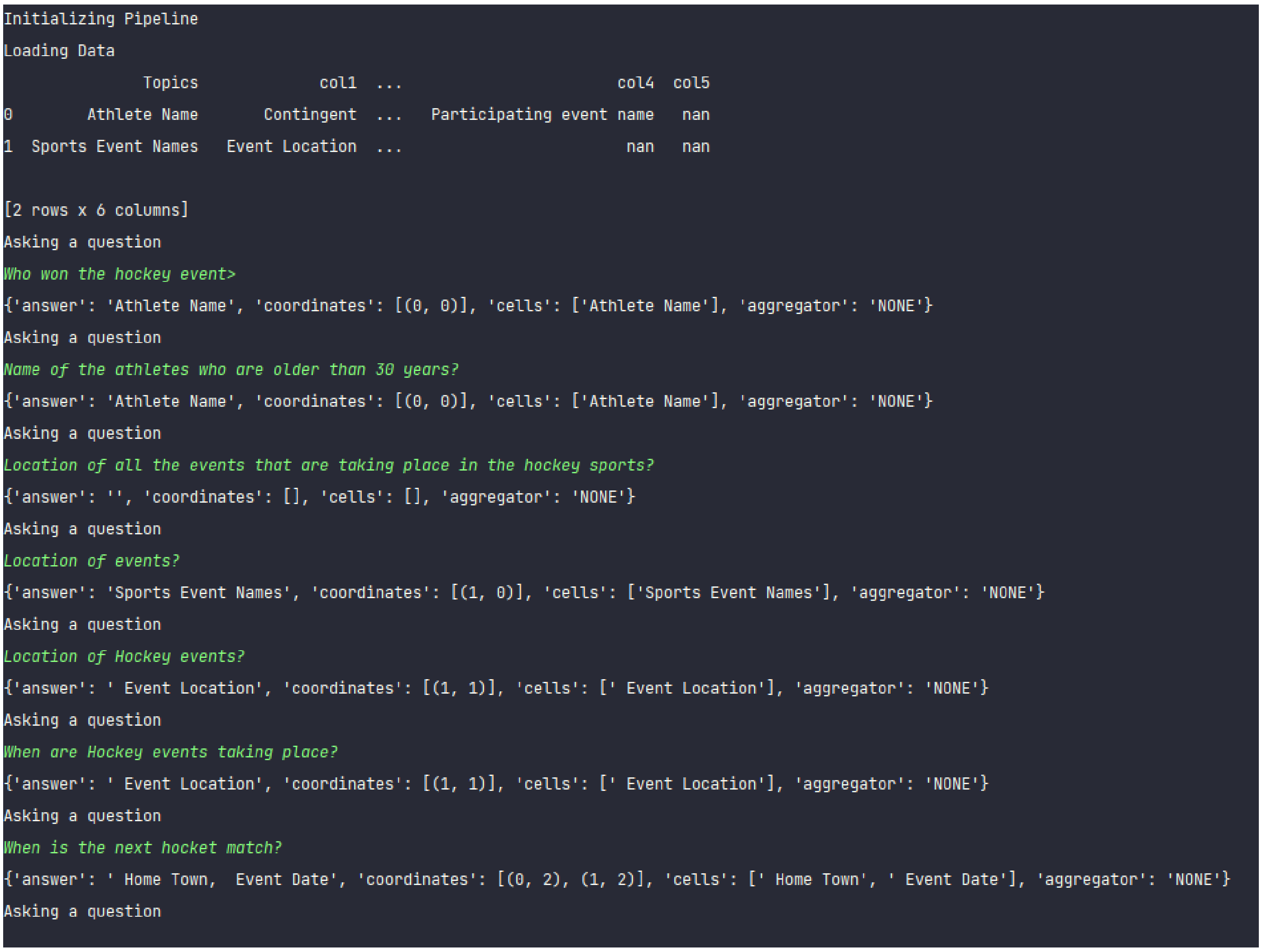
3.4. AI

To implement a Tabular Q&A chat-bot we have decided to use the Google Tapas tabular AI Q&A pipeline. In this model we simply use the models that are already trained on data-sets such as WikiSql. The system is initialized with we a pre-trained model, in our case it is called ”google/tapas-base-finetuned-wtq”. To answer the questions the model takes the a csv table as list of lists and the question itself. Then it returns an appropriate answer which can either be a column in a specific tuple of the table or a list of columns. As this application will have multiple tables. Each common type of knowledge has a super table it contains the contains the table names and table header names. As table headers will vary in size, the tables are arranged from highest number of headers to lowest, the remaining space is filled with empty string taken as null in the pipeline. The query is run on the super table which returns either a single table or a group of tables. In the case of group of tables the user is provided buttons with tables names to selected the topic most relative to their query and the query is finally ran on the selected table. In the case only one table tuple is selected the query is ran on the selected table and response is sent to the user. It is crucial that the table names and column header names are named properly or the AI wont be able to respond meaningfully to the question asked. The following two figures are the example of specific table and the super table.



```
30            Wynne Anderson-Lindsay  ...                 Hockey
31                 Owen Angootealuk  ...                 Hockey
32                    Alexa Annecca  ...          Speed Skating
33                     Ethan Anstey  ...                 Hockey
34                  Brynn Applegath  ...          Alpine Skiing
35                  Danielle Arbour  ...  Wheelchair Basketball
36                    Sam Archibald  ...                 Hockey
37              alexandre arencibia  ...                   Judo
38                    Danica Ariano  ...               Biathlon
39                  Craig Armstrong  ...                 Hockey
40                    Julia Arnason  ...      Artistic Swimming
41                  Nicholas Arnold  ...  Wheelchair Basketball
42                      Liam Arnsby  ...                 Hockey
43                      Joe Arntsen  ...                 Hockey
44  Lyra Eleftheria Arruda Kyriakidis  ...          Speed Skating
45                      Erik Arruda  ...             Gymnastics
46                  Chris Arsenault  ...       Freestyle Skiing
47          Donovan George Arsenault  ...                 Hockey
48              Ethan Dale Arsenault  ...             Gymnastics
49        Kristen Kathleen Arsenault  ...                Archery

[50 rows x 5 columns]
Asking a question
How many people are participating in the hockey sports event.
{'answer': 'COUNT > Michael Abgrall, Lyndsy Acheson, Avi Adam, Sawyer Adams, Rory Aiken, Eric Alarie,
 Arntsen, Donovan George Arsenault', 'coordinates': [(1, 0), (4, 0), (7, 0), (10, 0), (14, 0), (18,
 'Avi Adam', 'Sawyer Adams', 'Rory Aiken', 'Eric Alarie', 'Nolan Allan', 'Tysen Almond', 'Kevin Ande
 Arsenault'], 'aggregator': 'COUNT'}
Asking a question
What is the hometown of Sawyer Adams?
{'answer': 'Whitehorse', 'coordinates': [(10, 2)], 'cells': ['Whitehorse'], 'aggregator': 'NONE'}
```

Figure 3.5: Specific table QA responses



```
Initializing Pipeline
Loading Data
               Topics             col1  ...                       col4  col5
0        Athlete Name       Contingent  ...  Participating event name   nan
1  Sports Event Names   Event Location  ...                        nan   nan

[2 rows x 6 columns]
Asking a question
Who won the hockey event>
{'answer': 'Athlete Name', 'coordinates': [(0, 0)], 'cells': ['Athlete Name'], 'aggregator': 'NONE'}
Asking a question
Name of the athletes who are older than 30 years?
{'answer': 'Athlete Name', 'coordinates': [(0, 0)], 'cells': ['Athlete Name'], 'aggregator': 'NONE'}
Asking a question
Location of all the events that are taking place in the hockey sports?
{'answer': '', 'coordinates': [], 'cells': [], 'aggregator': 'NONE'}
Asking a question
Location of events?
{'answer': 'Sports Event Names', 'coordinates': [(1, 0)], 'cells': ['Sports Event Names'], 'aggregator': 'NONE'}
Asking a question
Location of Hockey events?
{'answer': ' Event Location', 'coordinates': [(1, 1)], 'cells': [' Event Location'], 'aggregator': 'NONE'}
Asking a question
When are Hockey events taking place?
{'answer': ' Event Location', 'coordinates': [(1, 1)], 'cells': [' Event Location'], 'aggregator': 'NONE'}
Asking a question
When is the next hocket match?
{'answer': ' Home Town,  Event Date', 'coordinates': [(0, 2), (1, 2)], 'cells': [' Home Town', ' Event Date'], 'aggregator': 'NONE'}
Asking a question
```

Figure 3.6: Super table QA responses

4. SPRINTS

4.1. Meetings

One of the first activities we performed as a team was getting together, reviewing user stories, and breaking these down into tasks. With these tasks, we could design how we will perform sprints to iteratively build this application. All members of our team are present for meetings, as it is vital to get everyone's input and expertise. We hold formal meetings twice a week, and communicate throughout the week over a dedicated Discord Server/Microsoft Teams chat.

4.2. Sprints

As stated above, user stories were broken down into tasks that individuals could contribute to and assigned to specific sprints. Within a sprint, team members would choose activities to contribute to, moving the sprint forward and iteratively building the application. Team members chose their own activities for the sprints, and we will include individual names next to the activities they worked on. Our team has opted to use Miro.com to visualize our sprints, as well as keeping track of the progress/completed tasks, and can be viewed by clicking on the "Sprint Backlog" link at the end of this document in Section 6.

4.2.1. Sprint 1

Some of the tasks we chose for Sprint 1 were:

• Choosing an AI model for the Chatbot

• Clarifying and organizing the data

• Decide input of the bot

• Decide the output of the bot

• Design the UI of the front-end/website

• Create the database UML

• Create the database using the database UML

4.2.2. Sprint 2

Some of the tasks we chose for Sprint 2 were:

• Create a React App

• Create Canada Games layout for website

• Create an input field for the user to type questions into

• Have a clear distinction between the users messages and the bots messages messages

• Create chat section of website where newest messages will be on the bottom, pushing older message up dynamically

• Create database queries to insert data

• Create database queries to update data

Our incomplete tasks for this sprint were:

• Deploy the AI

• Have the AI provide a response in an appropriate amount of time

• Setup auto configuration/deployment

4.3. Future Sprints

Alongside the completion of the above sprint tasks, we will continue to iteratively build up the database, AI, and website, providing further functionality and response times. We will also reference the user stories and user requirements to ensure what we are building coincides with what the user wants and to ensure all deliverables are met. Our backlog up to this point can be viewed by clicking on the ”Sprint Backlog” link at the end of this document in Section 6.

5. PROBLEMS

Our first problem encountered was deciding the AI model/method we would use. Because we did not have a viable database or any data, prototyping and testing individual methodologies was not possible. Because of this, one of our group members had to create their own local test database with sample data to test and prototype.

Our second problem encountered was structuring and re-structuring the UML and database; there are many ways the database can be approached, and in the end, it must be structured in such a way that the AI can easily utilize it. Because of this, we have been iteratively adding/removing tables and fields from the database- and it is likely that the database will continue to evolve throughout the duration of this project to conform to the AI model.

6. LINKS

GitHub
Sprint Backlog