1 - Write the name of the group and the list of your group members

Team Name: Search Party

Greg Pogue – 4583993
Joel Jacob – 6603245
Madeline Janecek – 6436620
Sam Langdon – 6180137
Brendan Park – 6541288
Kylee Schram – 6131726

GitHub: https://github.com/janecekm/COSC4P02Project2022

2 - Include a brief list/description of the features (subsystems) you planned to implement in each sprint and whether or not you accomplished them

Sprint 1:
Front end mockup
- Designed user interface in Adobe XD and laid out major components using the React JavaScript library so that they could be implemented in subsequent sprints

NLP research
- Different NLP python packages/libraries
- Different methods of spell correction in other NLP chatbots
- Creating lists for different key inquiries like course information, program information, exams/timetable and greetings to allow for matching of key terms

Primitive database
- Database relations and design
- Produced an entity-relationship diagram from the design
- Database created in PostgreSQL

Data gathering
- Initial web-scraping with Python Selenium
- Gathered course information, such as, prerequisites, cross listing, offering, lab times and so on, from different websites, and formatted it into a json file, so that we can use it for the database.
- Scraped the data about Academic advisors which is also formatted into json format.

Sprint 2:
Building out features
Connecting subsystems together
Integrate database with Flask
- Implement as an embedded database in SQLite
- Connect database to Flask backend using SQLAlchemy

Data preprocessing
- Take scraped data and clean it such that the format is compatible with the database

Text-To-SQL/NLP Research
- Researched popular implementations and problems with Text-To-SQL

- Generated responses with a base template consisting of the extracted keywords
- Code-cleanup and improvements
- NLP integration to front-end via Flask

Added functionality to front end components, including:

- Query input area
- Message and response display with thinking animation and auto-scrolling
- Menu with font increase, font decrease, help, and mode switching options

Sprint 3 (currently ongoing):

Populating database

- Pipelining data into the database after preprocessing
- Continually adding more relationships/tables to the database as we need it

NLP Features/Improvements

- Spell correction pre-processing added to backend
- Generation of test data for locations to evaluate named-entity recognition performance

Research into Hosting

- Deployment demos on Microsoft Azure
- Github Actions to automatically deploy from main branch as its updated

Incomplete:

- Populate database (From Sprint 1 moved to Sprint 3)
- Generate additional NLP lists (from Sprint 1 moved to future sprint)
    - Moved to realign with database information
- Design and implement splash page (from Sprint 2 moved to Sprint 3)
- Add tables to existing database design (from Sprint 2 moved to Sprint 3)
- Develop mobile version (from Sprint 2 to future sprints)

3 - Include a brief list of features (subsystems) that you plan to implement in the following sprints

Canada Games Chatbot:

- Separate database
- Additional data collection
- Create new NLP matching patterns

Testing:

- Develop an automated testing framework using PyTest

GitHub Actions:

- Automatic deployment to hosted app from main GitHub branch

Hosting:

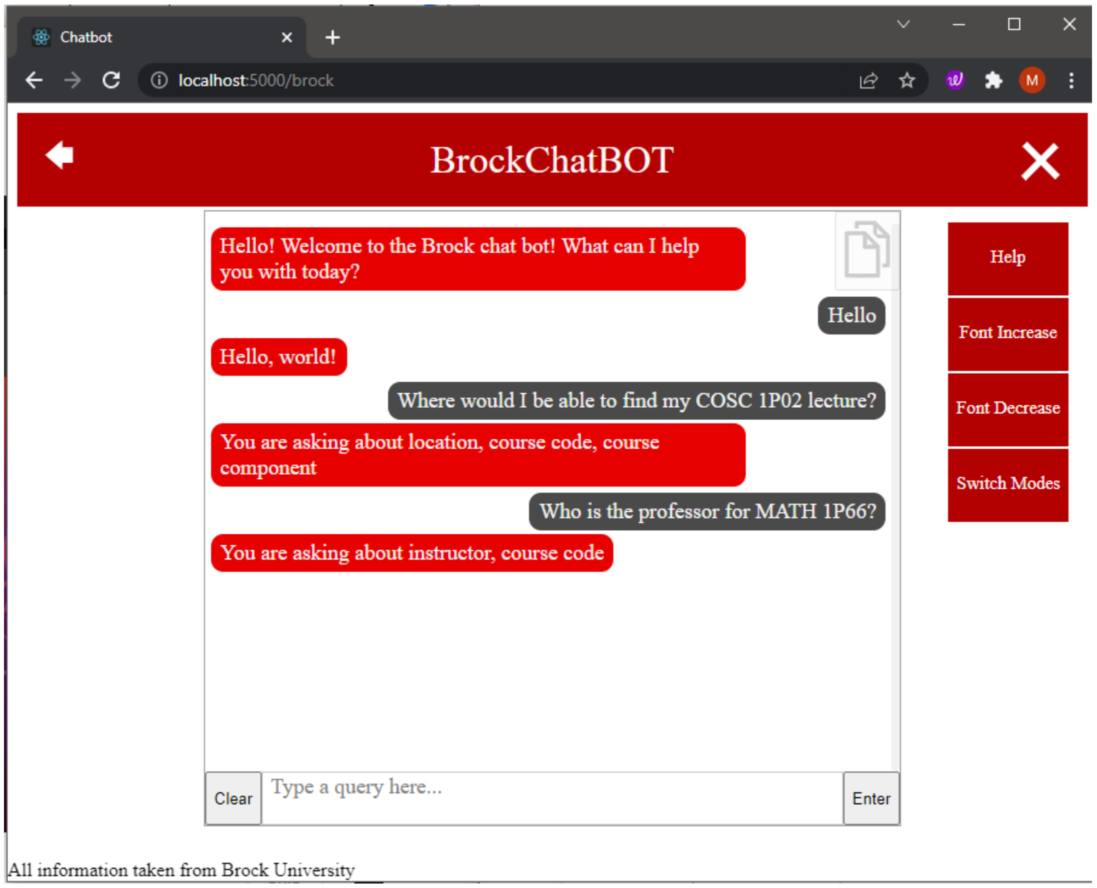
- Flask app hosted on Microsoft Azure

NLP:

- Implement and test Named-Entity Recognition for various types of entities (locations, people, etc)
- Convert user input into appropriate SQL queries to pull from recently hosted DB
- Conversion of received data into appropriate response to user

- Improve accuracy of spell correction methodology
- Further optimizations and improvements in terms of accuracy and overhead

4 - Include screenshots if you have a working version of the system



Hello! Welcome to the Brock chat bot! What can I help you with today?

Hello

Hello, world!

Where would I be able to find my COSC 1P02 lecture?

You are asking about location, course code, course component

Who is the professor for MATH 1P66?

You are asking about instructor, course code

The features shown in the above screen shot include:

- A message display that shows the user's input as well as the chatbot's generated responses. The size of the text bubbles will change to accommodate the size of the text they're meant to encapsulate. Once the height of the conversation exceeds the height of the display area, a scroll bar will appear. After each new text bubble the display will auto-scroll down to the newest message.
- A text area for user input. When no text is there a default message will appear. The input area will grow and shrink with the user input. There is also a character limit of 250 characters to avoid extremely long input.
- An enter button. This button will send the user's input to the server to get a response. While waiting for a response, the text area will be disabled and a thinking animation will appear in the display area. The input area will also be cleared and reselected for the user. Pressing the 'Enter' key will have the same results.
- A clear button that clears the user input area.
- A back button that will bring the user to a splash/menu page
- A button at the top right that will opens the menu (shown as open in screenshot)

- A help button that will show the user how to use the chatbot
- A text increase button that allows the user to increase the text size of the display and input area (cut off at a predefined threshold)
- A text decrease button that allows the user to decrease the text size of the display and input area (cut off at a predefined threshold)
- A switch modes button that allows the user to toggle between the Canada Games and Brock chatbots

5 - Mention any issues you encountered

Back-end:

- Creating a functional back-end system required knowledge of each of the subsystems and how they communicate with one another. Originally we were using the Node.js server packaged with React but this became difficult to integrate with our other components, mainly our NLP using Python and the database it would need to query. This created a new set of tasks to solve the networking. After research into Flask, we pivoted to Python to consolidate our back-end architecture. Additionally, Flask allows us to use a SQL framework called SQLAlchemy to make simpler queries to an embedded database. This also gave us the ability to serve the React app through Flask as opposed to routing our networking through a proxy.

Web-scraping:

- Some scripts were time-intensive. Solved by reading more about the functionality of Selenium to run the browser driver in headless mode, so that the full browser GUI didn't have to be rendered. Also combined all GET requests into a single browser session, so that the overhead of starting the driver/session only happened once.

Front end:

- We found that JavaScript had several attributes that impacted its readability and writability. For instance, its implicit typing meant that we had to be especially careful when naming and creating our variables so that we would not overwrite important information.
- The React library allows us to create a more professional and visually appealing user interface, however it significantly increased the complexity of our code. Consequently, a large portion of our time had to be dedicated to learning and experimenting with React.

6 - Describe the contributions and achievements of each member of the group

Greg Pogue

- Contributes to: Database, Back-end, Documentation
- Achievements:
    - Organize and structure project management tools
    - Designed and created database
    - Connected database to Flask

        - Cleaned data to comply with database

Joel Jacob

- Contributes to: Web Scraping, Front-end development, Networking
- Achievements:
    - Learned Javascript and React framework
    - Learned how to connect individual components using Flask
    - Learned how to host websites on the cloud
    - Used Selenium to scrape Brock University data

Madeline Janecek

- Contributes to: Front-end development
- Achievements:
    - Learned Javascript and React framework
    - Designed user interface
    - Implemented different components of the front-end

Sam Langdon

- Contributes to: NLP
- Achievements:
    - Matching patterns
    - NLP code optimization and improvement (increased readability and accuracy)
    - Custom pipeline training + named entity recognition research

Brendan Park

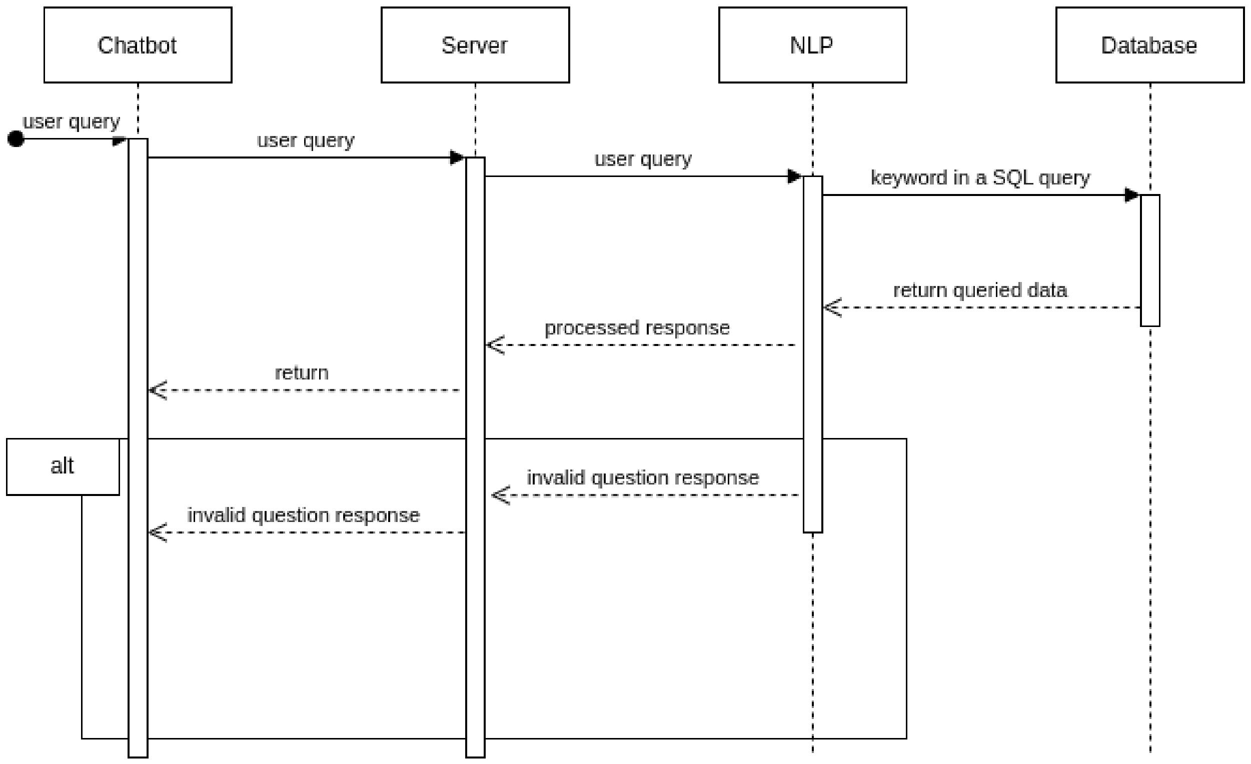
- Contributes to: NLP
- Achievements:
    - Matching patterns
    - Spell correction
    - Text-To-SQL research
    - NLP integration into Flask/front-end

Kylee Schram

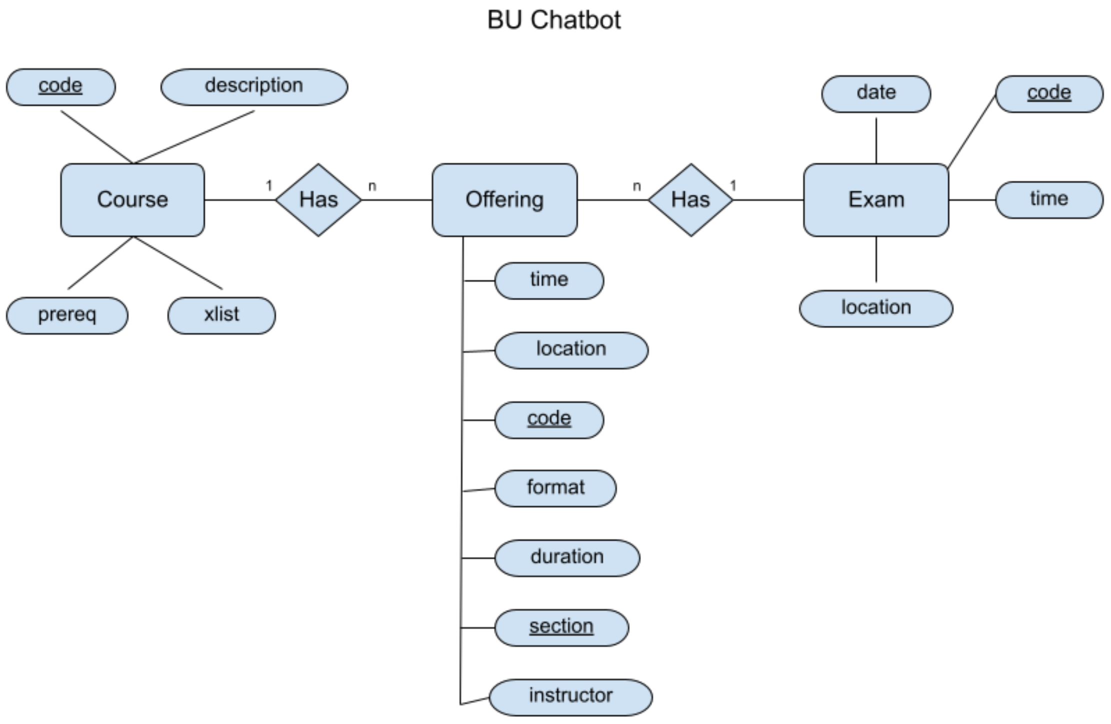
- Contributes to: Web scraping, Back-end, Networking
- Achievements:
    - Scripts to scrape course timetable and building codes
    - Processed scraped timetable data for DB entry
    - Populated course timetable database

7 - Diagrams

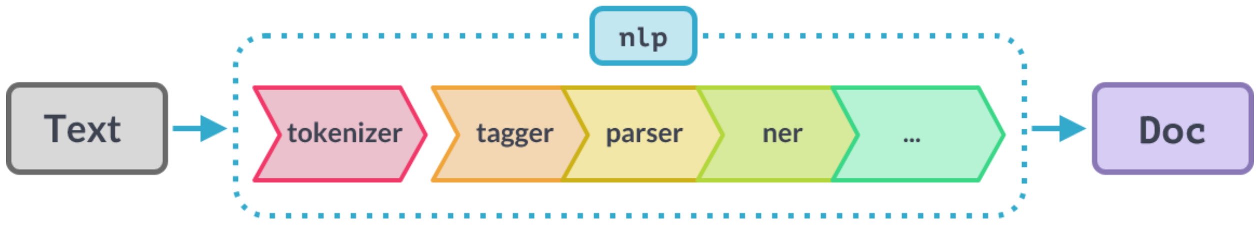
System sequence diagram:



Entity-relationship diagram:



NLP Pipeline (ref: https://spacy.io/usage/processing-pipelines):


Text
nlp
tokenizer
tagger
parser
ner
...
Doc