Project NoCap: Fact Checking with AI

Team members:
Anthony Ciero: aciero2022@my.fit.edu
Thomas Chamberlain: tchamberlain2023@my.fit.edu
Varun Doddapaneni: vdoddapaneni2023@my.fit.edu
Joshua Pechan: jpechan2023@my.fit.edu

Faculty advisor: Professor Silaghi: msilaghi@fit.edu

Clients:
- Students / teachers
- Citizens
- Journalists

Date of Client Meeting: TBD

Goal and Motivation:

The goal of this project is to allow clients to easily be able to fact check articles, also aiming to utilize a chrome extension that reads the current webpage and provides feedback on truth and manipulation. Currently to accurately fact check a website, there are sites you can go to and input some text and it will provide whether it is fact or fiction, however this requires effort that many do not want to go through and may not know about. This method can also be cumbersome leading to many not bothering to fact check. Our extension aims to streamline this process allowing for easy fact and bias checking as well as a graphical representation of the language used.

Approach:

Our application should give the user a bias rating for an article or a block of text that they wish to be evaluated. The user can either use the website, or a chrome extension that readily evaluates a source once opened. The rating should reflect the type of language a text uses, and the context in which they are put in. The application makes use of Natural Language Processing (NLP) via a prebuilt AI model to evaluate the language of a text. These ratings will also have a breakdown using Python's LangGraph module.

Our application will also have aggregate rankings for specific publications. When a user wants to evaluate a source, the ranking for that article will be taken and put into the aggregate ranking for that particular publication (ex: CNN or BBC). All the publications we rank will be visible on the main website, and will update every time new articles are added. The rankings of these publications will be represented graphically, as well as having a table listing publication rankings from high to low.

The application also includes a Google Chrome extension. This is for accessibility and ease of use purposes. The user can open the extension on any website containing a text they want to analyze, and the extension will readily break it down and rate it. The extension will serve as a more accessible, thin version of the main application.

Novel Features/Functionalities:

One novel feature incorporated into our project is the use of graphical representations. Specifically graphical representations to break down the article and return a misinformation rating and back it up with charts of specific keywords used. These graphs essentially explain the reasoning behind why we reached the rating we did.

Another novel feature would be to reach Web Content Accessibility Guidelines (WCAG) on the AA level. This means that people with most disabilities can access the application with little trouble. This includes being able to tab between buttons, having accessible color contrast, and text that is easy to read and understand. Making the tool inclusive for a wider audience.

Algorithms and Tools:

Some potentially useful tools for the system include:

- **Python (backend):** primary server-side language for AI orchestration and services.
- **FastAPI (API/backend web framework):** lightweight, async-friendly framework to expose REST endpoints.
- **LangChain & LangGraph (LLM/NLP modules):** tooling to compose prompts, retrieval, multi-step AI workflows, and data visualization.
- **React (JavaScript UI):** component-based interface for the extension popup and web dashboard.
- **AWS Bedrock (Nova Lite):** managed LLMs with model swapability for classification and analysis tasks.
- **AWS Amplify (GraphQL with AppSync + DynamoDB):** optional persistence layer for user preferences, cached verdicts, and analytics.

Some potentially useful algorithms for the system include:

- **Custom ranking logic:** order evidence by credibility, recency, and cross-source agreement.
- **Claim detection & classification:** identify factual statements and label them for verification.
- **Prompting strategies:** structured prompts/templates for reliable, explainable outputs.

Some potentially useful integrations:

- **GitHub:** code hosting, version history, issues/PRs, and permissions.
- **Search/Fact-check APIs:** e.g., Google Programmable Search, FactCheck.org datasets, or other evidence sources.

Technical Challenges:

One technical challenge would be to learn whatever algorithm/tool that we pick to progress the project. Not everybody in the group knows every tool that we will

potentially use so we would have to learn these tools on a surface level to be able to implement them into our project. Whatever part of the project is assigned to the members, we would all have to invest time and effort into learning the tools.

Another technical challenge for our group is that we all have stronger backgrounds with backend development rather than frontend design. This could make it more difficult to implement Web Content Accessibility Guidelines on the AA level, accessible layouts, and responsive graphical components. As a result, we will need to dedicate extra time to learning frontend frameworks to ensure the final product meets desired standards.

Another technical challenge is the limited knowledge in LangChain and LangGraph. The learning curve may slow down progress as both of these are central for managing prompts and reasoning chains in large language models. The plan to overcome this is to start small with prototype experiments before fully implementing it into our project.

An additional technical challenge is to establish an external connection to a site for the AI chatbot integration which the team has limited experience with. This would include managing API calls and handling authentication securely. Additionally we would need to account for potential rate limits.

Design:

- 3 Main Pages
    - Home: The Home Page is the default starting page when the website is opened, allowing the user to paste a desired article or block of text that they wish to fact check.
    - Report: The Report Page is the page that comes up after a user enters their desired article or selects a stored article from the Database Page. It shows information about the article like title, author, publisher, and the URL. After, it shows the authenticity score of the article followed by a summary of why the AI rated the article this way and a detailed explanation of what it believed to be fact or fiction.
    - Database: The Database Page is where all authenticated articles are stored for users to search through. These articles are stored in publisher cards that show the publisher's name as well as aggregate authenticity score of articles. The articles are stored in their own cards showing authenticity score on the right side and article title, author, publication date, and URL. The authenticity scores are color-coded, with green, yellow, or red depending on score.

Evaluation:

- Speed: ensure that article reports are created within 30 seconds
- Accuracy: Currently our model will give a wide range of authenticity scores for the same article. Our goal is to concise this down as much as possible.
- User Survey: Conduct user surveys on specific parts of our software like creating an article report, searching/filtering for an article in our database, (e.g. rating of 1-5 on each of the different features)

Progress Summary:

Model/Feature	Completion %	To Do
Frontend	80%	Add example articles to home page, connect article data via input on Home Page
Database	75%	Connect to frontend article input to save in database
AI Model	100%	Connected to AI model allowing score and report generation
Prompt Engineering	75%	Improve prompt to yield better/more reliable score.
Chrome Extension	0%	One of main goals for semester 2

Milestone 4: Model Improvements, Chrome Extension, and Branding

- Prompt Engineering
- Improve model output
- Show default home page cards
- Article data connection to report page via input
- Create logo and branding
- Chrome extension

Milestone 5: Continue + Evaluation/Poster

- Finalize prompt engineering
- Ensure consistent model output
- Create graphs/visualizations for collected data
- Conduct evaluation and analyze results
- Create poster for Senior Design Showcase

Milestone 6:

- Finalize data graphs/visualizations
- Test/demo of the entire system
- Conduct evaluation and analyze results
- Create user/developer manual
- Create demo video

Task Matrix for Milestone 4:

Task	Thomas	Anthony	Josh	Varun
1. Prompt Engineering	0%	0%	50%	50%
2. Improve model output	0%	0%	50%	50%

3. Show default home page cards	50%	50%	0%	0%
4. Article data connection to report page via input	30%	0%	0%	70%
5. Create logo and branding	50%	50%	0%	0%
6. Chrome extension	50%	50%	0%	0%

Approval from Faculty Advisor:
"I have discussed with the team and approve this project plan. I will evaluate the progress and assign a grade for each of the three milestones."
Signature: _____ Date: _____