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. More specifically, we aim 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 change every time new articles. 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, and multi-step AI workflows.
- **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.

Milestone 1:

- Compare and select technical tools for A, B, C, ...
- Provide small ("hello world") demo(s) to evaluate the tools for A, B, C, ...
- Resolve technical challenges: X, Y, Z, ...
- Compare and select collaboration tools for software development, documents/presentations, communication, task calendar
- Create Requirement Document
- Create Design Document
- Create Test Plan

Milestone 2: Website

- Design Frontend
- Set up AI model on AWS
- Establish basic connection with AI
- Develop rudimentary backend and API
- Establish API endpoints

Milestone 3: Fact-Checking AI

- Create basic prompt engineering for AI
- Use LangChain to break down texts into tokens
- Output basic score from AI
- Set up database to store scores/rankings

Task Matrix for Milestone 1:

Task	Thomas	Josh	Anthony	Varun
Compare and select technical tools for A, B, C, ...	**25%**	**25%**	**25%**	**25%**
Provide small ("hello world") demo(s) to evaluate the tools for A, B, C, ...	**25%**	**25%**	**25%**	**25%**
Resolve technical challenges: X, Y, Z, ...	**25%**	**25%**	**25%**	**25%**
Compare and select collaboration tools for software development, documents/presentations, communication, task calendar	**25%**	**25%**	**25%**	**25%**
Create Requirement Document	**25%**	**25%**	**25%**	**25%**
Create Design Document	**25%**	**25%**	**25%**	**25%**
Create Test Plan	**25%**	**25%**	**25%**	**25%**

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: _____