Individual Reflections

Sprint 1
Bryan Finlayson
Team Headline Grabber

What you planned to do:

HG0 - Project initialization
HG3 - Define a new command line option to specify the sources of news
HG5 - Define a new command line option to exclude certain news sources from application default settings
HG9 - Implement logic to export the result to a well-formatted HTML document
HG13 - Implement logic to show the usage instructions by option `--help` in the command line
HG14 - Implement logic to allow end user can access the source of the headline through the provided URL
HG17 - Implement the logic to open the result automatically by user's default application
HG18 - Implement the logic for parsing HTML data from specific news sources
HG19 - Implement the logic to detect and combine overlapping news articles from various sources into a single entry
HG20 - Define the template for the final HTML output news report
US1 - Define command line name and core functionality

What you did not do:

None

What Problems You Encountered:

None

Issues you completed:

HG0 - Project initialization
HG3 - Define a new command line option to specify the sources of news
HG5 - Define a new command line option to exclude certain news sources from application default settings
HG9 - Implement logic to export the result to a well-formatted HTML document

HG13 - Implement logic to show the usage instructions by option `--help` in the command line
HG14 - Implement logic to allow end user can access the source of the headline through the provided URL
HG17 - Implement the logic to open the result automatically by user's default application
HG18 - Implement the logic for parsing HTML data from specific news sources
HG19 - Implement the logic to detect and combine overlapping news articles from various sources into a single entry
HG20 - Define the template for the final HTML output news report
US1 - Define command line name and core functionality

Files you worked on:

```
.github/workflows/branch-name-check.yml
.gitignore
.python-version
pyproject.toml
README.md
src/headline_grabber/configurations/__init__.py
src/headline_grabber/configurations/enums/__init__.py
src/headline_grabber/configurations/enums/scraper_engine.py
src/headline_grabber/configurations/enums/str_enum.py
src/headline_grabber/configurations/sites/__init__.py
src/headline_grabber/configurations/sites/bgb.yaml
src/headline_grabber/configurations/sites/nyt.yaml
src/headline_grabber/configurations/sites/wap.yaml
src/headline_grabber/configurations/sites/wsj.yaml
src/headline_grabber/models/__init__.py
src/headline_grabber/models/display_document.py
src/headline_grabber/models/headline.py
src/headline_grabber/models/news_site.py
src/headline_grabber/models/pipeline_context.py
src/headline_grabber/models/user_preferences.py
src/headline_grabber/pipeline_steps/__init__.py
src/headline_grabber/pipeline_steps/classify_subject.py
src/headline_grabber/pipeline_steps/display_report.py
src/headline_grabber/pipeline_steps/filter_sites.py
src/headline_grabber/pipeline_steps/group_by_similarity.py
src/headline_grabber/pipeline_steps/pipeline_step.py
```

```
src/headline_grabber/pipeline_steps/prepare_for_display.py
src/headline_grabber/pipeline_steps/score_sentiment.py
src/headline_grabber/pipeline_steps/scrape_text.py
src/headline_grabber/pipeline_steps/text_similarity.py
src/headline_grabber/pipelines/__init__.py
src/headline_grabber/pipelines/pipeline.py
src/headline_grabber/validators/click/__init__.py
src/headline_grabber/validators/click/validate_site_name.py
src/headline_grabber/validators/__init__.py
src/headline_grabber/validators/__main__.py
src/headline_grabber/headline_grabber.py
```

What you accomplished:

I started the project by setting up the initial structure, directories, version control, and configuration files. I then added command line options to specify and exclude news sources, allowing users to customize which sources to scrape and which to omit, thereby improving the tool's flexibility. I then created generalized parsing logic to accurately extract headlines and other relevant data from the specified news sources, based on configuration YAML files with some custom configurations per news site. I then developed the “pipeline steps” through which text scraped from news sites gets processed, evaluated, and sorted by different NLP functions (eg sentiment analysis, subject categorization, summarization, similarity analysis). To handle duplicate news items, I implemented logic using NLP techniques (ie similarity analysis) to detect and combine overlapping news articles from various sources, with the goal of providing a concise and comprehensive report. I then developed the functionality to export scraped news headlines into an HTML document, ensuring the output is user-friendly and visually appealing. Each headline in the output includes a link to the original news source, allowing users to easily access the full articles. To enhance user experience, I implemented logic to automatically open the generated HTML report in the user's default web browser once the pipeline process completes all its steps.