# Data Science Project Report

## Objective

The goal of this project is to demonstrate the ability to programmatically scrape data from news articles, extract entities (persons or organizations) using Named Entity Recognition (NER), classify the sentiment of the articles (positive, negative, or neutral), and integrate these aspects into a cohesive application. This report details the approach, methodology, challenges faced, and reflections on the project's components.

---

## Approach

### 1. Data Scraping

A Python-based web scraping solution was implemented to fetch the main content of news articles using their URLs. The `requests` library was employed for sending HTTP requests, while `lxml` was used to parse the HTML content and extract paragraphs, combining them into a cohesive article text.

Key Implementation Details:

* URL input through both a script and a Gradio-based UI.
* Error handling to manage invalid URLs or connectivity issues.

### 2. Entity Extraction

Named Entity Recognition (NER) was performed using the spaCy library to identify and extract entities of type `PERSON` (individuals) and `ORG` (organizations). Two approaches were used:

1. Pretrained Model:

* Utilized spaCy's `en\_core\_web\_sm` for initial entity extraction.

1. Custom NER Model:

* Trained on a small dataset with labeled examples (e.g., `"Apple is looking at buying U.K. startup for $1 billion"`).
* Training involved fine-tuning spaCy's pipeline with 30 iterations to recognize `PERSON` and `ORG` entities.

Key Features:

* Duplicate entities were removed to ensure uniqueness.
* Flexibility to switch between the pretrained and custom models for evaluation.

### 3. Sentiment Analysis

The sentiment of each article was analyzed using the VADER sentiment analyzer. This tool calculates a sentiment polarity score to classify the text as:

* Positive: Compound score > 0.05
* Negative: Compound score < -0.05
* Neutral: Compound score between -0.05 and 0.05

### 4. Storage

The extracted data (article text, entities, and sentiment) was stored in an SQLite database. A schema was defined with the following structure:

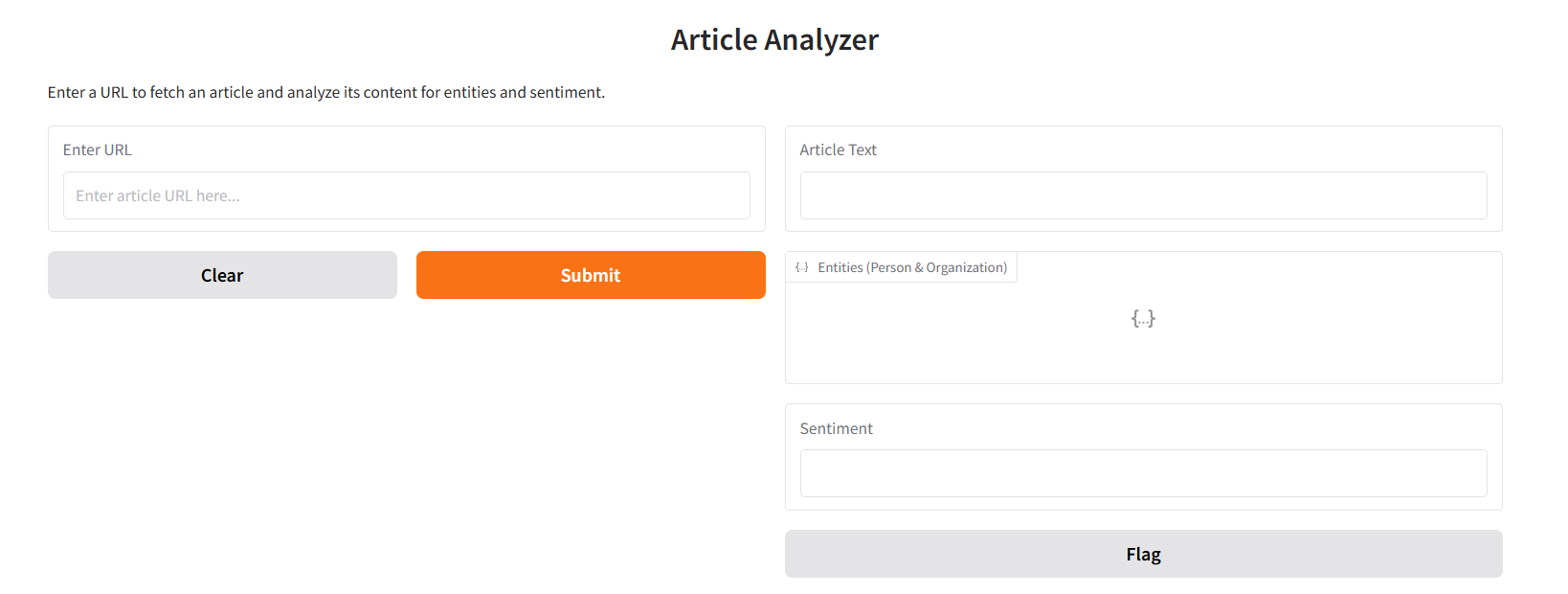
* ID: Primary key.
* URL: Unique identifier for the article.
* Text: Full content of the article.
* Entities: JSON-encoded list of persons and organizations.
* Sentiment: Sentiment classification.

The `sqlalchemy` library was used to interact with the database.

### 5. Deployment and User Interface

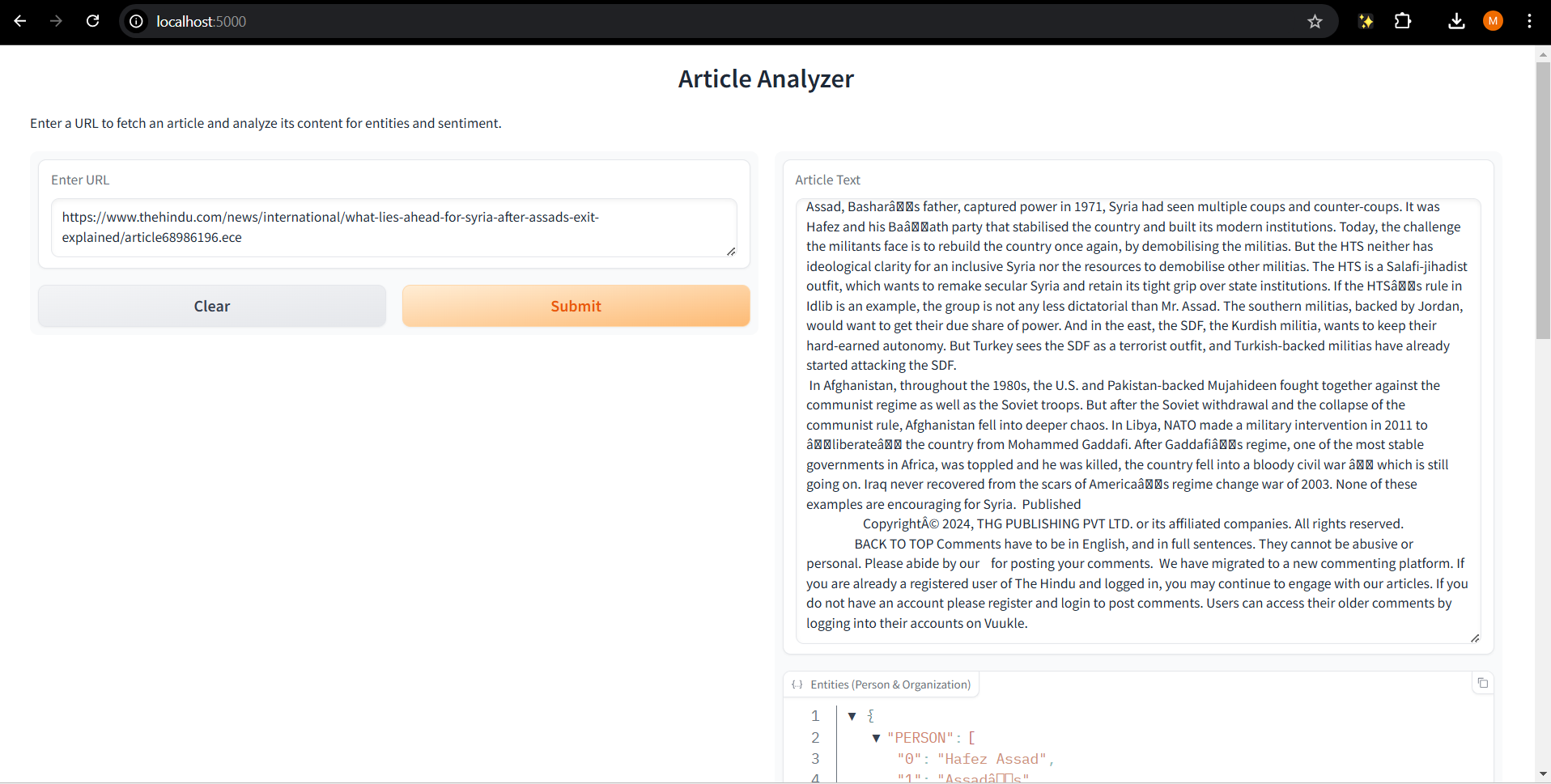
A user-friendly interface was developed using Gradio. This web-based UI allows users to:

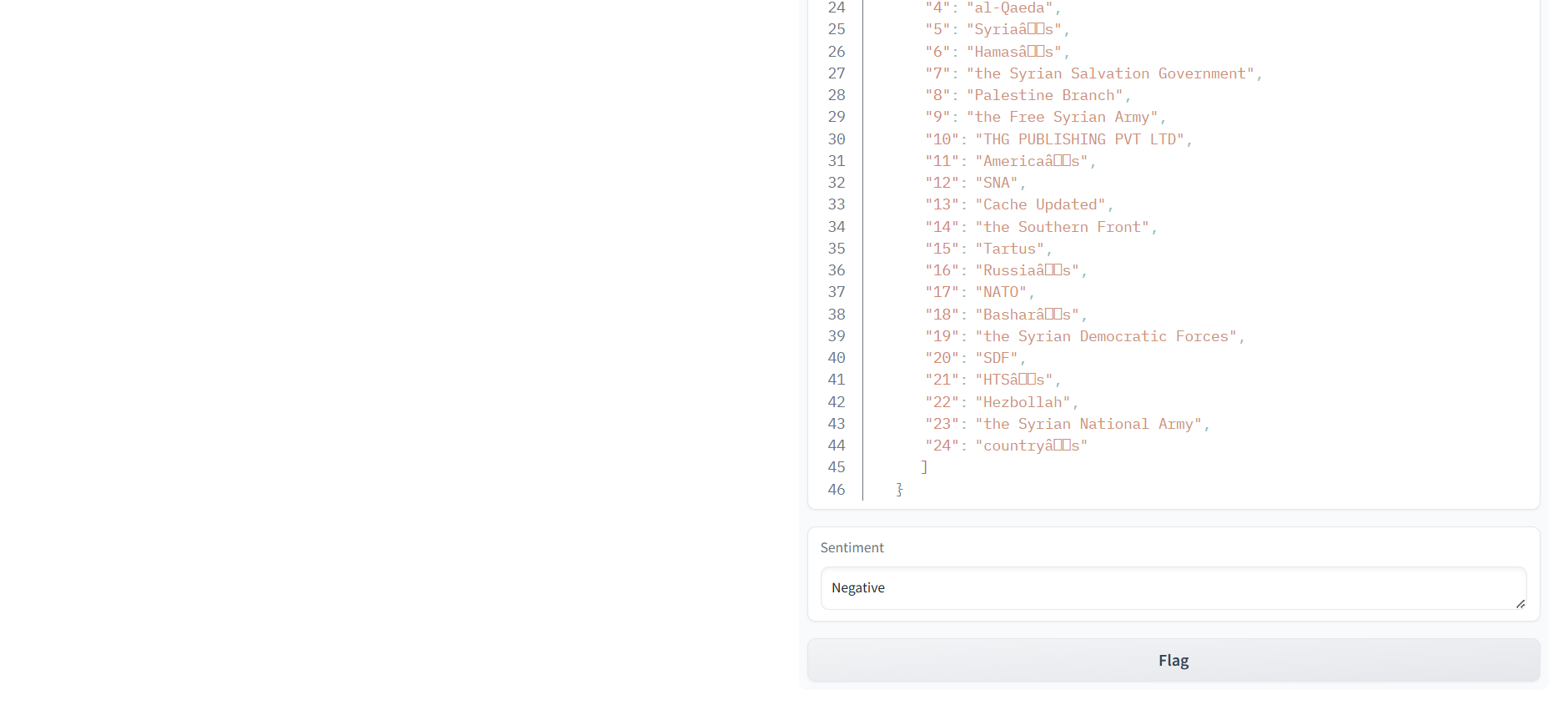
* Input a URL.
* View the fetched article text, extracted entities, and sentiment classification in real time.

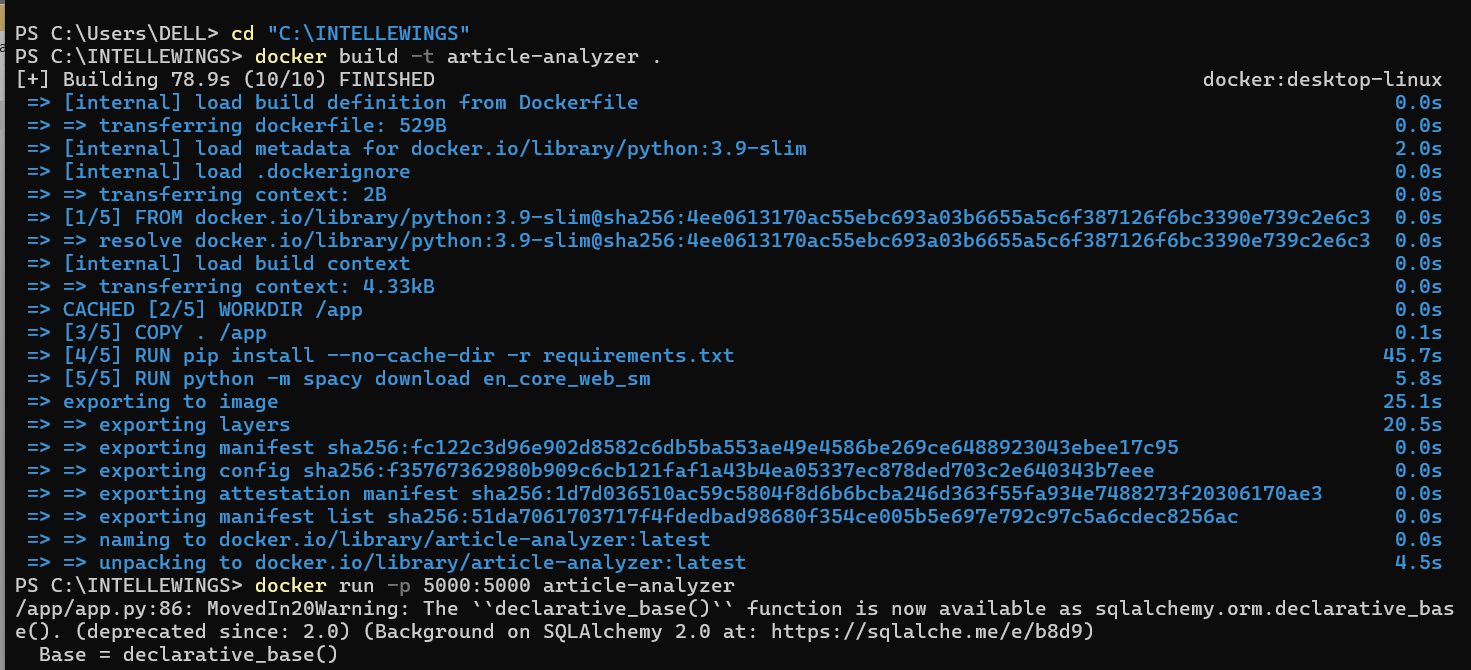


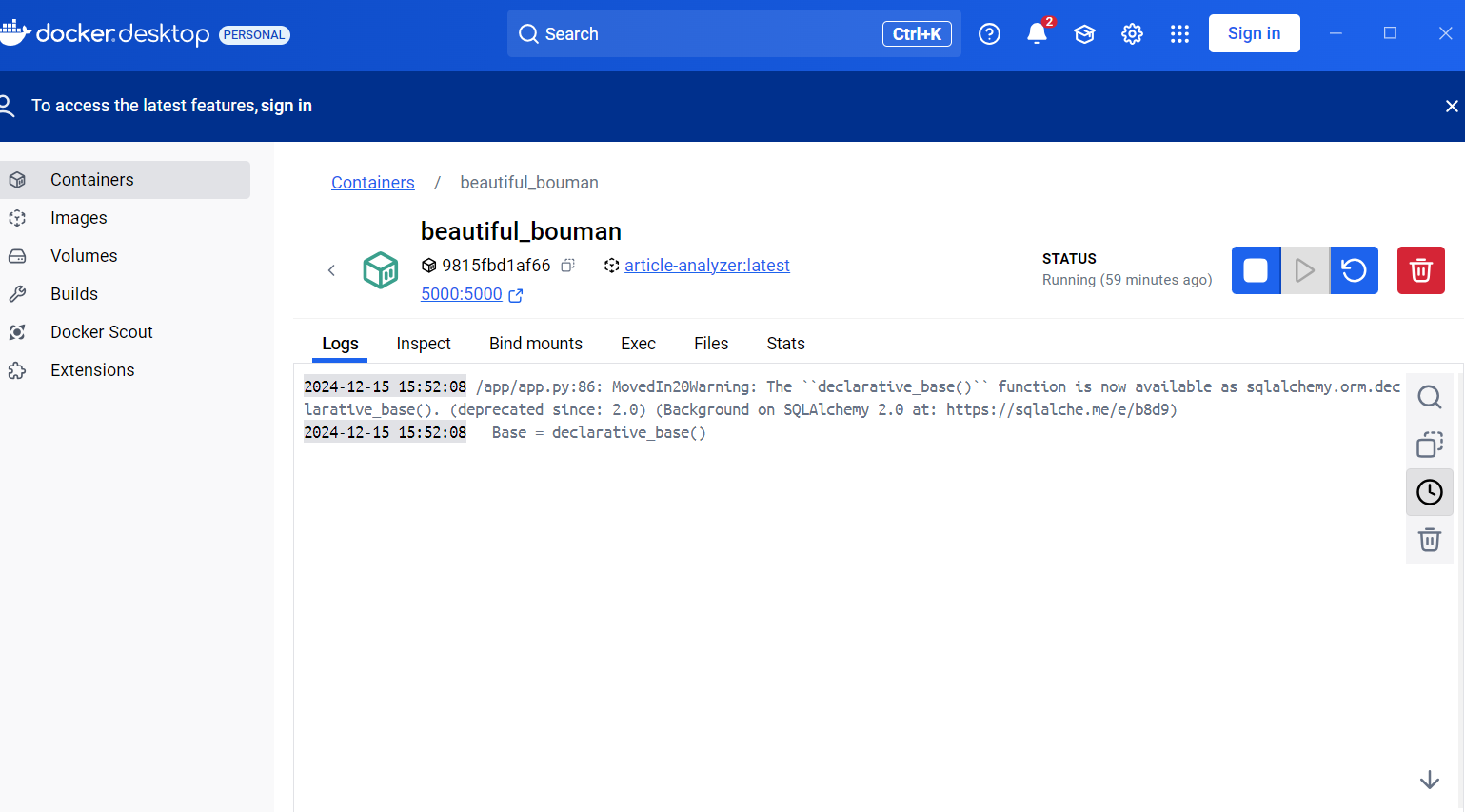
UI allows users to analyze articles by entering a URL, which fetches the article's content and extracts key insights. The **Enter URL** field accepts the article link, and the **Submit** button processes it, displaying the fetched **Article Text**. Identified **Entities (Person & Organization)** are shown in a dedicated section, while the **Sentiment** box indicates whether the article's tone is positive, negative, or neutral. Users can clear inputs using the **Clear** button or report issues with the **Flag** button. This interface efficiently summarizes article content with entity recognition and sentiment analysis.

The project was containerized using Docker for ease of deployment in local environments.









## Challenges and Solutions

### Data Quality in Scraping

Challenge: Articles often contain extraneous content (ads, navigation links).

Solution: Used XPath queries to target `<p>` tags explicitly, improving content extraction accuracy.

### Training a Custom NER Model

Challenge: Limited training data.

Solution: Augmented the dataset with diverse examples to enhance model performance.

### Handling Ambiguous Sentiment

Challenge: Articles with mixed tones were hard to classify.

Solution: Introduced a fallback to classify borderline cases as 'Neutral.'

### Database Constraints

Challenge: Ensuring unique URLs and preventing duplicate entries.

Solution: Defined `url` as a unique field in the SQLite schema.

## Reflections on Accuracy

### Entity Extraction

The pretrained spaCy model performed well on common entities but struggled with domain-specific terminology.

The custom model improved accuracy for labeled examples but requires additional training for broader generalization.

### Sentiment Analysis

VADER's rule-based approach worked effectively for most cases but struggled with complex articles containing sarcasm or nuanced language.

Integration with a deep learning-based sentiment model (e.g., BERT) could enhance performance.

## Tools and Technologies

* Libraries: `spaCy`, `VADER Sentiment`, `requests`, `lxml`, `sqlalchemy`, `gradio`
* Database: SQLite
* Deployment: Docker
* UI: Gradio

## Future Improvements

* Expand the training dataset for the custom NER model.
* Integrate additional languages or domain-specific models for sentiment analysis.
* Enhance web scraping to filter irrelevant sections like advertisements.
* Deploy the project on a cloud platform for greater accessibility.

## Conclusion

This project successfully integrates multiple data processing techniques to provide a complete pipeline for extracting and analyzing data from news articles. While the results are promising, further improvements in training data and model sophistication could significantly enhance accuracy and usability.