AI-Powered News Article Classifier with History

Objective: The goal of this project is to create a web application that classifies news articles into categories and keeps track of past classification requests. The application should take a news article URL as input, scrape the article content, predict its category using an AI model, store the request and the prediction in a database, and display the predicted category as well as a history of past classification requests.

User Interface:

Home Page: A landing page with instructions on how to use the application.

Prediction Page: A form where users can enter a news article URL. After the URL is submitted, the application should display the predicted category as well as a history of past classification requests, including the article URLs and their predicted categories.

Back End:

Scraping Service: A service that takes a news article URL, scrapes the article content, and returns it.

Classification Service: A service that takes the scraped article content and predicts its category using an AI model.

Data Persistence Service: A service that stores each classification request and its result in a database and retrieves the history of past requests when needed.

Deployment:

The applicant can use any resources available to him to develop and deploy the application. The applicant should provide a URL to the deployed application for evaluation.

Evaluation Criteria:

1. Functionality: Does the application work as expected? Can it successfully scrape article content, predict its category, store the request and prediction, and display the history of past requests?
2. Code Quality: Is the code well-structured, readable, and maintainable? Are there any bugs or security vulnerabilities?
3. User Experience: Is the application easy to use? Is the design clean and intuitive?
4. AI Implementation: Does the classification provide accurate results? Is the AI model appropriately chosen and properly implemented?

5. Deployment: Is the application successfully deployed and accessible online? Are there any issues with scalability or reliability? Is the application code on GitHub and is the deployment automated?

Bonus Points:

1. Robustness to Different News Sites: Does the scraping service work well with a wide variety of news sites, or does it only work with a few specific ones?
2. Additional Features: Features like displaying the most important words or phrases that influenced the model's prediction, or allowing users to provide feedback on the prediction accuracy.
3. Test Coverage: Good test coverage with unit tests and integration tests.
4. Documentation: Well-written documentation explaining the architecture of the application, how to run it locally, and how to deploy it.