The Daily News Summary Architectural Design

12/1/2024

The Daily News Summary will be a web application built with a .NET Web API backend, and React JS frontend, and a SQLite database. The frontend, backend, and database will be independent of each other leading to well defined responsibilities for each component. The front end will be responsible for the user interface of the web application. The user will be able to view article summaries in addition to being able to click on the article link to view the full original article if they wish to do so. The front end will call the API endpoints to get the articles from the database. The API is responsible for fetching and processing the articles from the NewsCatcher API. This process also includes generating the AI summaries of each article using the bart-large-cnn model provided by the Cloudflare Workers AI API. The database is responsible for storing the articles and information about the authors of each article. A single-tenant SQLite database was chosen because of its well-suited ability to store article text.

The overall flow of the application is as follows. The user clicks on the “View Articles” button on the home page. Then the front end calls the Fetch and Process API endpoint in which the NewsCatcher API is called. This data is processed and the GenerateAISummary method is called to generate a summary of the articles text using the bart-large-cnn model accessed via the Cloudflare Workers AI API. Then this summary along with the other details of the article such as the link to the original article and published date are stored in the database. After the Fetch and Process endpoint has been called to fetch the articles of the day, the Get All Articles endpoint is called by the frontend. This returns a list of all the articles stored in the database ordered in descending order by published date so that the latest articles are at the top of the page.

Unit testing was done using the XUnit unit testing tool. A unit test was written to test the functionality of the Get All Articles endpoint. In addition, the web application and unit test were containerized with Docker.

**Architectural Design Diagrams and Tables:  
Layered Model:**  
A diagram of a server

Description automatically generated

**Design Principles:**

|  |  |
| --- | --- |
| **Principle** | **Explanation** |
| Ease to use | Users should be able to easily access and read the news articles. |
| Clear interface | Users should be able to clearly navigate the website and access the articles. |
| Reliable news | Users should be able to count on getting reliable news every day. |

**Block Diagram:**

**A diagram of a diagram

Description automatically generated**

**Docker System Diagram:  
A screenshot of a computer

Description automatically generated**

**Component Interaction Diagram:  
A diagram of a software company

Description automatically generated**