Assignment: News Summarization and Text-to-Speech Application

Objective

Develop a web-based application that extracts key details from multiple news articles related to a given company, performs sentiment analysis, conducts a comparative analysis, and generates a text-to-speech (TTS) output in Hindi. The tool should allow users to input a **company name** and receive a structured sentiment report along with an audio output.

Requirements

1. **News Extraction**: Extract and display the title, summary, and other relevant metadata from **at least 10 unique news articles** related to the given company. Consider only **non-JS weblinks** that can be scraped using `BeautifulSoup (bs4)`.
2. **Sentiment Analysis**: Perform sentiment analysis on the article content (positive, negative, neutral).
3. **Comparative Analysis:** Conduct a comparative sentiment analysis across the 10 articles to derive insights on how the company's news coverage varies.
4. **Text-to-Speech**: Convert the summarized content into Hindi speech using an open-source TTS model.
5. **User Interface**: Provide a simple web-based interface using **Streamlit** or **Gradio**. Users should input a **company name** (via dropdown or text input) to fetch news articles and generate the sentiment report.
6. **API Development:** The communication between the frontend and backend must happen via APIs.
7. **Deployment**: Deploy the application on **Hugging Face Spaces** for testing.
8. **Documentation**: Submit a detailed README file explaining implementation, dependencies, and setup instructions.

Submission Guidelines

- Submit the **GitHub repository link** with a well-structured codebase.
- The repository should include:
  - `app.py` (or equivalent main script)
  - Requirements file (`requirements.txt` or `environment.yml`)
  - **README** with setup and usage instructions
  - `utils.py` containing the utility function and code
  - `api.py` to support the development of APIs.
- Deploy the application on Hugging Face Spaces and provide the **deployment link**.

- Ensure code is properly commented and follows best practices.

Input Format

- The application should accept a **company name** as input.

Expected Output Format

A structured report including:

- **Title**: Extracted from each article.
- **Summary**: A concise summary for each article.
- **Sentiment**: Categorized as **Positive, Negative, or Neutral** for each article.
- **Topics:** Key topics covered in the article.
- **Comparative Analysis**: Comparison highlighting how news coverage differs in various reports. A structured data format would be preferred.
- **Hindi TTS**: Playable audio file summarizing the sentiment report.

Example Output

Input:

```
Company Name: Tesla
```

Output:

```
{
 "Company": "Tesla",
 "Articles": [
   {
     "Title": "Tesla's New Model Breaks Sales Records",
     "Summary": "Tesla's latest EV sees record sales in Q3...",
     "Sentiment": "Positive",
     "Topics": ["Electric Vehicles", "Stock Market", "Innovation"]
   },
   {
     "Title": "Regulatory Scrutiny on Tesla's Self-Driving Tech",
     "Summary": "Regulators have raised concerns over Tesla’s self-driving
software...",
     "Sentiment": "Negative",
     "Topics": ["Regulations", "Autonomous Vehicles"]
   }
 ],
 "Comparative Sentiment Score": {
   "Sentiment Distribution": {
     "Positive": 1,
     "Negative": 1,
     "Neutral": 0
   },
   "Coverage Differences": [
     {
       "Comparison": "Article 1 highlights Tesla's strong sales, while Article 2 
discusses regulatory issues.",
       "Impact": "The first article boosts confidence in Tesla's market growth, 
while the second raises concerns about future regulatory hurdles."
     },
     {
       "Comparison": "Article 1 is focused on financial success and innovation, 
whereas Article 2 is about legal challenges and risks.",
       "Impact": "Investors may react positively to growth news but stay cautious 
due to regulatory scrutiny."
     }
   ],
   "Topic Overlap": {
     "Common Topics": ["Electric Vehicles"],
     "Unique Topics in Article 1": ["Stock Market", "Innovation"],
     "Unique Topics in Article 2": ["Regulations", "Autonomous Vehicles"]
   }
 },
,
 "Final Sentiment Analysis": "Tesla’s latest news coverage is mostly positive. 
Potential stock growth expected.",
 "Audio": "[Play Hindi Speech]"
}
```

Documentation Requirements

- **Project Setup**: Steps to install and run the application.
- **Model Details**: Explanation of models used for summarization, sentiment analysis, and TTS.
- **API Development:** Clearly state how the APIs are being made use of and how to access them via Postman or any other tools.
- **API Usage**: If any third-party APIs are used, specify their purpose and integration details.

- **Assumptions & Limitations**: Clearly state any assumptions made in the implementation.

Evaluation Criteria

- **Correctness**: Does the solution extract and process information accurately?
- **Efficiency**: Is the application optimized for performance?
- **Robustness**: Does it handle errors and edge cases appropriately?
- **Deployment**: Is the application accessible via Hugging Face Spaces?
- **Code Quality**: Is the code well-structured, documented, and maintainable?

Code will automatically be tested for quality. Please follow PEP8 guidelines.

Deadline: 1 week from the starting time
Submission Mode: GitHub repository link + Hugging Face Spaces deployment link + Video Demo explaining how the application works.

Bonus Points

💡 **Extra cookie points** for **detailed analysis reporting**. A querying system over the stories will also be credited with cookie points.