News Analyzer Application - Technical Documentation

1. Introduction

The **News Analyzer Application** is a full-stack system designed for analyzing company-related news articles using NLP techniques. The system consists of:

- **Frontend**: Streamlit-based web interface
- **Backend**: FastAPI service handling data processing
- **NLP Pipeline**: Sentiment analysis, entity recognition, and multilingual text-to-speech (TTS)

---

2. System Architecture

```
graph TD;
    A[Streamlit UI] -->|HTTP Requests| B(FastAPI);
    B --> C[NewsAPI];
    B --> D[Article Scraping];
    B --> E[Sentiment Analysis];
    B --> F[Entity Recognition];
    B --> G[Hindi TTS];
```

---

3. Installation & Configuration

3.1 Requirements

- Python 3.8+
- NewsAPI account (free tier)
- 4GB+ RAM recommended

3.2 Setup Instructions

```
# Clone repository
git clone https://github.com/yourrepo/news-analyzer.git
cd news-analyzer

# Create virtual environment
python -m venv .venv
source .venv/bin/activate  # Linux/Mac
.venv\Scripts\activate    # Windows

# Install dependencies
pip install -r requirements.txt

# Install spaCy model
python -m spacy download en_core_web_sm
```

3.3 Configuration

Obtain a NewsAPI key from NewsAPI. Then, create a `.env` file:

```
NEWS_API_KEY=your_api_key_here
API_BASE=http://localhost:8000
```

4. Model Specifications

4.1 Content Processing Pipeline

Component	Model	Description
Sentiment Analysis	cardiffnlp/twitter-roberta-base-sentiment	Transformer-based model for sentiment analysis
Entity Recognition	spaCy en_core_web_sm	Rule-based named entity recognition
Machine Translation	googletrans	Google Translate API wrapper
TTS Generation	facebook/mms-tts-hin	Meta's Massively Multilingual Speech model

4.2 Performance Characteristics

Model	Input Limit	Processing Time	Hardware Utilization
Sentiment	512 tokens	2-3s/article	CPU/GPU accelerated
TTS	500 chars	5-7s/request	Requires CUDA for GPU acceleration

5. API Reference

5.1 Endpoint Specification

Base URL: `http://localhost:8000`

`GET /analyze`

`GET /analyze?company={company_name}`

Response Schema:

```
{
  "company": "string",
  "articles": [
    {
      "title": "string",
      "summary": "string",
      "sentiment": ["Positive", "Neutral", "Negative"],
      "topics": ["string"]
    }
  ],
  "tts_url": "/temp/{uuid}.wav"
}
```

5.2 API Testing Guide

Start FastAPI server:

```
uvicorn api:app --reload
```

Use Postman collection:

```
{
  "info": {
    "name": "News Analyzer API",
    "schema":
"https://schema.getpostman.com/json/collection/v2.1.0/collection.json"
  },
  "item": [
    {
      "name": "Analyze Company",
      "request": {
        "method": "GET",
        "header": [],
        "url": {
          "raw": "http://localhost:8000/analyze?company=Tesla",
          "protocol": "http",
          "host": ["localhost"],
          "port": "8000",
          "path": ["analyze"],
          "query": [
            {"key": "company", "value": "Tesla"}
          ]
        }
      }
    }
  ]
}
```

---

6. Third-Party Services

6.1 NewsAPI Integration

- **Service**: Article discovery
- **Rate Limits**: 100 requests/day (free tier)
- **Query Parameters:**

```
{
  "q": "company AND (stock OR market OR shares)",
  "language": "en",
  "pageSize": 10,
  "sortBy": "publishedAt"
}
```

6.2 Google Translate

- **Usage**: English-to-Hindi translation for TTS
- **Fallback**: Returns original text on failure

---

7. Operational Constraints

7.1 Key Assumptions

- News sources provide machine-readable meta descriptions
- Target websites use standard HTML semantic markup
- Financial entities are properly capitalized in articles
- Users require Hindi audio output exclusively

7.2 Known Limitations

Category	Limitation	Mitigation Strategy
Content	Dynamic JS-rendered articles not supported	Use headless browser for scraping
NLP	Context loss in text truncation	Implement chunked processing
Localization	Limited financial term support in Hindi	Maintain custom glossary
Infrastructure	Temporary audio file storage	Implement cloud storage integration

---

8. Maintenance & Monitoring

8.1 Logging Configuration

```
# API error handling
logger.error(f"Article processing failed: {str(e)}")
```

Recommended: Use Sentry integration for production monitoring.

8.2 Health Check Endpoint

`GET /health`

Response:

```
{
  "status": "OK",
  "timestamp": "ISO-8601"
}
```

---

9. License & Attribution

- **NewsAPI**: Terms & Conditions
- **spaCy**: MIT License
- **Hugging Face Models**: Apache 2.0
- **MMS TTS**: CC-BY-NC 4.0

End of Documentation