# AUTOMATED NEWS ARTICLE SUMMARIZATION LEVERAGING IBM NATURAL LANGUAGE UNDERSTANDING

## 1. Introduction

The goal of this project is to automate the summarization of news articles using IBM's Natural Language Understanding (NLU), enabling users to quickly grasp essential information from lengthy content.

## 2. Overview of IBM NLU

IBM NLU analyzes text to extract meaningful insights, including entities, keywords, and sentiment. This makes it a powerful tool for text summarization and analysis.

## 3. Project Setup

### 3.1 Required Libraries

- **Flask**: For the web application framework.
- **Requests**: For fetching web content.
- **BeautifulSoup**: For HTML parsing.
- **IBM Watson SDK**: For NLU services.

### 3.2 Setting Up Your Development Environment

**Install Visual Studio Code (VS Code)**

- Download and install Visual Studio Code.

**Install Python**

- Download and install Python from python.org.
- Ensure that Python is added to your system PATH.

### 3.3 Installation of Required Libraries

Run the following command in your terminal:

```
pip install flask requests beautifulsoup4 ibm-watson
```

## 4. Application Structure

### 4.1 Directory Overview

```
├── app.py          # Main application file
└── templates
    └── index.html   # HTML template for user interface
```

## 5. Obtain IBM Watson NLU API Key and URL

### 5.1 Sign Up for IBM Cloud

- Go to IBM Cloud and create an account.

### 5.2 Create Watson NLU Service

1. Navigate to the IBM Cloud Catalog.
2. Search for "Natural Language Understanding" and select it.
3. Click on "Create" to provision the service.

### 5.3 Retrieve API Key and URL

1. After creating the service, go to the "Manage" tab.
2. Under "API keys," click "Show" to reveal your API key.
3. Copy the service URL from the "Service credentials" section.

## 6. Implement Summarization Functionality

### 6.1 Update app.py

Add your API key and URL:

```
ibm_api_key = 'YOUR_API_KEY'

ibm_url = 'YOUR_SERVICE_URL'

authenticator = IAMAuthenticator(ibm_api_key)

nlu = NaturalLanguageUnderstandingV1(version='2021-08-01', authenticator=authenticator)

nlu.set_service_url(ibm_url)
```

**6.2 Define Routes**

**Home Route**

Renders the main page:

```
@app.route('/')
def home():
    return render_template('index.html')
```

**Summarize Route**

Handles the summarization of articles:

```
@app.route('/summarize', methods=['POST'])
def summarize():
    url = request.json['url']
    response = requests.get(url)
    soup = BeautifulSoup(response.content, 'html.parser')
    paragraphs = soup.find_all('p')
    full_text = ' '.join([para.get_text() for para in paragraphs])
    summary_length = 4000  # Adjust as needed
    summary = full_text[:summary_length]
    return jsonify({'summary': summary})
```

**Analyze Route (Optional)**

Analyzes text using NLU:

```
@app.route('/analyze', methods=['POST'])
def analyze():
    text = request.form['text']
    response = nlu.analyze(
        text=text,
        features=Features(entities=EntitiesOptions(limit=5), keywords=KeywordsOptions(limit=5))
    ).get_result()
    return jsonify(response)
```

**7. Frontend Implementation**

**7.1 Create HTML Template (index.html)**

In the templates folder, create a file named index.html with the following content:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>News Summarizer</title>
</head>
<body>
    <h1>Automated News Article Summarizer</h1>
    <form id="urlForm">
        <label for="url">Enter Article URL:</label>
        <input type="text" id="url" name="url" required>
        <button type="submit">Summarize</button>
    </form>
    <div id="summary"></div>
    <script>
        document.getElementById('urlForm').addEventListener('submit', async function(e) {
            e.preventDefault();
            const url = document.getElementById('url').value;
            const response = await fetch('/summarize', {
                method: 'POST',
                headers: { 'Content-Type': 'application/json' },
                body: JSON.stringify({ url })
            });
            const data = await response.json();
            document.getElementById('summary').innerText = data.summary;
        });
    </script>
</body>
</html>
```

**8. Running the Application**

**8.1 Start the Flask App**

Run the application with:

```
python app.py
```

**8.2 Access the Application**

Navigate to http://127.0.0.1:5000/ in your web browser.