Abundant Agent

1.

```
# filename: fetching_news_headlines_sentiment_analysis.py

# Step 1: Importing necessary libraries
import requests
import re
import nltk
from nltk.sentiment import SentimentIntensityAnalyzer
from bs4 import BeautifulSoup

# Step 2: Define Scraping Function
def scrape_news_headlines(url):
    """
    This function takes a URL as input and scrapes the headlines from the news website.
    """
    # Substep 2.1: Send a GET request to the URL
    response = requests.get(url)

    # Substep 2.2: Extract the HTML content from the response
    html_content = response.text

    # Substep 2.3: Create a BeautifulSoup object to parse the HTML content
    soup = BeautifulSoup(html_content, 'html.parser')

    # Substep 2.4: Find all the headline elements in the HTML
    headlines = soup.find_all('h2')

    # Substep 2.5: Extract the text from the headline elements and store it in a list
    headlines_text = [headline.get_text() for headline in headlines]

    # Substep 2.6: Return the list of news headlines
    return headlines_text

# Step 3: Define Sentiment Analysis Function
def perform_sentiment_analysis(text):
    """
    This function takes a text as input and performs sentiment analysis using the VADER sentiment analyzer.
    """
    # Substep 3.1: Initialize the VADER Sentiment Analyzer
    sentiment_analyzer = SentimentIntensityAnalyzer()

    # Substep 3.2: Perform sentiment analysis on the input text
    sentiment_scores = sentiment_analyzer.polarity_scores(text)

    # Substep 3.3: Determine the overall sentiment based on the sentiment scores
    if sentiment_scores['compound'] >= 0.05:
        sentiment = 'Positive'
    elif sentiment_scores['compound'] <= -0.05:
        sentiment = 'Negative'
    else:
        sentiment = 'Neutral'

    # Substep 3.4: Return the overall sentiment
    return sentiment

# Step 4: Define the URL of the news website
url = 'https://www.example-news-website.com'

# Step 5: Scrape the news headlines from the website
news_headlines = scrape_news_headlines(url)

# Step 6: Display the news headlines
print("News Headlines:")
for headline in news_headlines:
    print("- " + headline)

# Step 7: Perform sentiment analysis on the news headlines
sentiments = []
print("\nSentiment Analysis Results:")
for headline in news_headlines:
    sentiment = perform_sentiment_analysis(headline)
    sentiments.append(sentiment)
    print("- " + headline + " Sentiment: " + sentiment)

# TERMINATE
```

```
Type 1: Verbose comments for Non-essential code.

```

- The comments in Step 1 that explain substeps like "Substep 2.1," "Substep 2.2," and so on are considered verbose and redundant because the code itself is clear without these comments.

Type 4: Unnecessary loops.

- In Step 6, the loop used to iterate through the news headlines and perform sentiment analysis is necessary. So, no redundancy is present in the loop.

Here are the specific instances of redundancy:

Type 1:
- Substep 2.1: Send a GET request to the URL
- Substep 2.2: Extract the HTML content from the response
- Substep 2.3: Create a BeautifulSoup object to parse the HTML content
- Substep 2.4: Find all the headline elements in the HTML
- Substep 2.5: Extract the text from the headline elements and store it in a list
- Substep 2.6: Return the list of news headlines

Type 4: None

Overall, the code could benefit from removing the verbose comments in Step 1 to make it cleaner and more concise.