In [3]:
```
# Existing imports and setup
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from bs4 import BeautifulSoup
from datetime import datetime, timedelta
import time
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
```

In [4]:
```
# Dictionary mapping location names to codes
location_map = {
    '26 - Lake Zurich Lou Malnati\'s': 'ejCL3tEIQWyx2n7NvyvNHg%3D%3D',
    '56 - Michigan Ave Lou Malnati\'s': '7bvSIeIXRBemZW1YQmTJXA%3D%3D'
}

# Function to get date input from user
def get_date_input(prompt):
    date_str = input(prompt)
    if not date_str:
        return None
    try:
        date_obj = datetime.strptime(date_str, '%Y%m%d')
        return date_obj.strftime('%Y%m%d')
    except ValueError:
        print("Invalid format. Please enter the date in YYYYMMDD format.")
        return get_date_input(prompt)

# Function to get location input from user by name
def get_location_input(prompt):
    print("Available locations:")
    for idx, name in enumerate(location_map.keys(), start=1):
        print(f"{idx}: {name}")

    choice = input(prompt)
    if not choice:
        return None
    try:
        choice = int(choice)
        if 1 <= choice <= len(location_map):
            location_name = list(location_map.keys())[choice - 1]
            return location_name, location_map[location_name]
        else:
            print("Invalid choice. Please enter a number corresponding to a
            return get_location_input(prompt)
    except ValueError:
        print("Invalid input. Please enter a number.")
        return get_location_input(prompt)

# Get date and location from user
start_date_str = get_date_input("Enter the start date (YYYYMMDD) or press E
end_date_str = get_date_input("Enter the end date (YYYYMMDD) or press Enter
location_name, location_code = get_location_input("Enter the number corresp

# Set default dates if not provided
if not start_date_str:
    start_date_str = (datetime.now() - timedelta(1)).strftime('%Y%m%d')
if not end_date_str:
    end_date_str = datetime.now().strftime('%Y%m%d')

# Initialize WebDriver
driver = webdriver.Chrome()
wait = WebDriverWait(driver, 10)  # Wait up to 10 seconds for elements to l

# CSS Selector for extracting information from the page
css_selector = "div.sales-summary"  # Modify this selector as per the page

# Function to scrape and print data using the CSS Selector
def scrape_page():
    try:
        # Wait for the element to be present on the page
```

```
        elements = wait.until(EC.presence_of_all_elements_located((By.CSS_S
        for element in elements:
            print(element.text)
    except Exception as e:
        print(f"An error occurred while scraping: {str(e)}")

# If no location is provided, use the default location (56 - Michigan Ave L
if not location_code:
    location_name = '56 - Michigan Ave Lou Malnati\'s'
    location_code = location_map[location_name]

# Construct the URL with the provided location
url = f'https://www.toasttab.com/restaurants/admin/reports/sales/sales-summ
print(f"Navigating to {url}")
driver.get(url)
scrape_page()

# Wait for 60-80 seconds to allow the website to load
time.sleep(120)  # Adjust the sleep time as needed

# Find the element using XPath
element = driver.find_element(By.XPATH, '//*[@id="single-spa-application:ro

# Extract the title attribute value
title_value = element.get_attribute('title')

# Optionally, if you want to process this HTML with BeautifulSoup
html_content = element.get_attribute('outerHTML')
soup = BeautifulSoup(html_content, 'html.parser')

# Now you can use BeautifulSoup methods on 'soup'
print(soup.prettify())

# Email credentials and settings
sender_email = 'ayushlokhande0@gmail.com'
receiver_email = 'ayushlokhande229@gmail.com'
password = 'izjb ptio ywyi edba'  # Be sure to handle passwords securely

# Create the email content with start date, end date, and location
subject = 'Net Sales Value'
body = f"""
Location: {location_name}
Start Date: {start_date_str}
End Date: {end_date_str}

The extracted net sales value is: {title_value}
"""

# Set up the email server
server = smtplib.SMTP('smtp.gmail.com', 587)
server.starttls()
server.login(sender_email, password)

# Create the email message
msg = MIMEMultipart()
msg['From'] = sender_email
msg['To'] = receiver_email
msg['Subject'] = subject
msg.attach(MIMEText(body, 'plain'))

# Send the email
```

```
server.send_message(msg)
server.quit()

print('Email sent successfully.')

# Close the WebDriver instance
driver.quit()
```

```
Enter the start date (YYYYMMDD) or press Enter for default: 20240801
Enter the end date (YYYYMMDD) or press Enter for default: 20240825
Available locations:
1: 26 - Lake Zurich Lou Malnati's
2: 56 - Michigan Ave Lou Malnati's
Enter the number corresponding to the location or press Enter for only 56
- Michigan Ave Lou Malnati's : 1
Navigating to https://www.toasttab.com/restaurants/admin/reports/sales/sal
es-summary?startDate=20240801&endDate=20240825&locations=ejCL3tEIQWyx2n7Nv
yvNHg%3D%3D (https://www.toasttab.com/restaurants/admin/reports/sales/sale
s-summary?startDate=20240801&endDate=20240825&locations=ejCL3tEIQWyx2n7Nvy
vNHg%3D%3D)
An error occurred while scraping: Message:

<span class="reporting-spa__data-table__text leading-[14px] text-[14px]" d
ata-testid="formatted-value-text" title="$158,392.81">
 $158,392.81
</span>
Email sent successfully.
```

In [ ]: