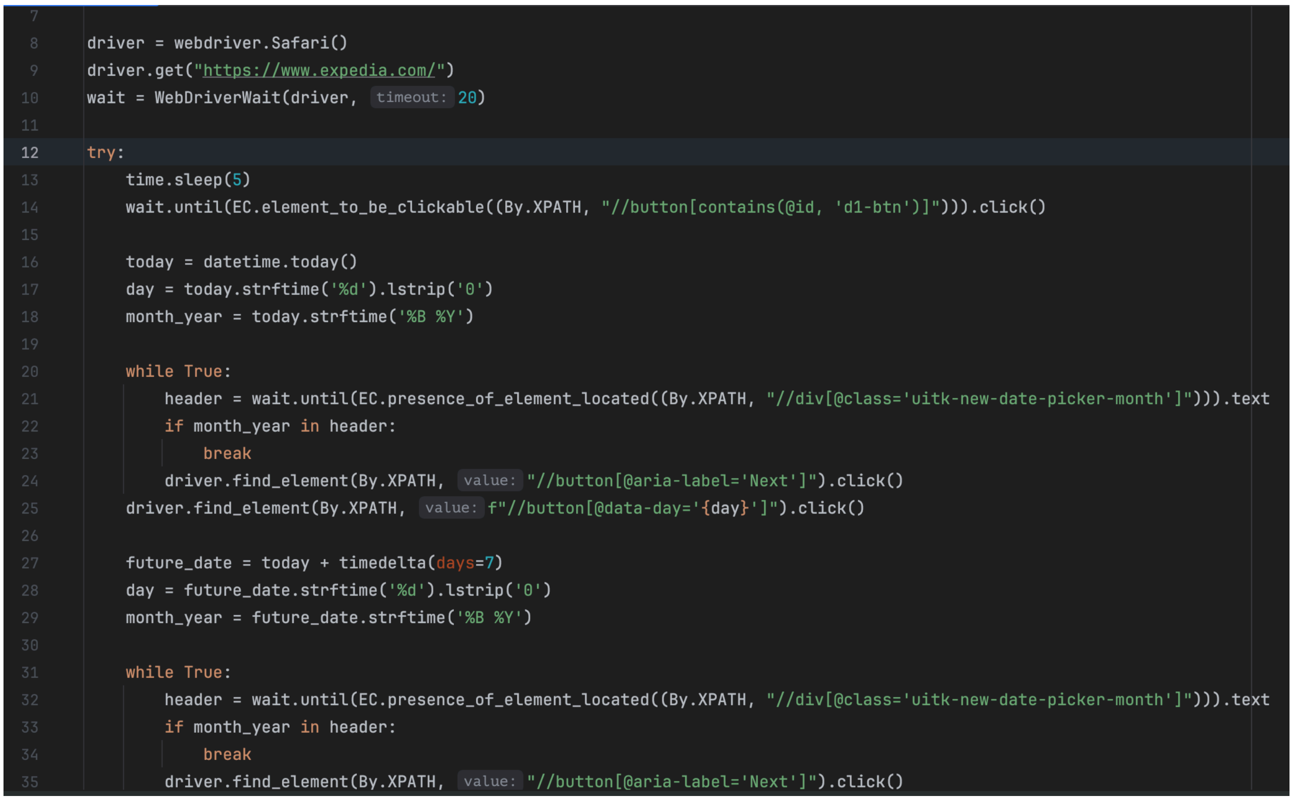
Assignment-4

Task 1: Automating a Date Picker

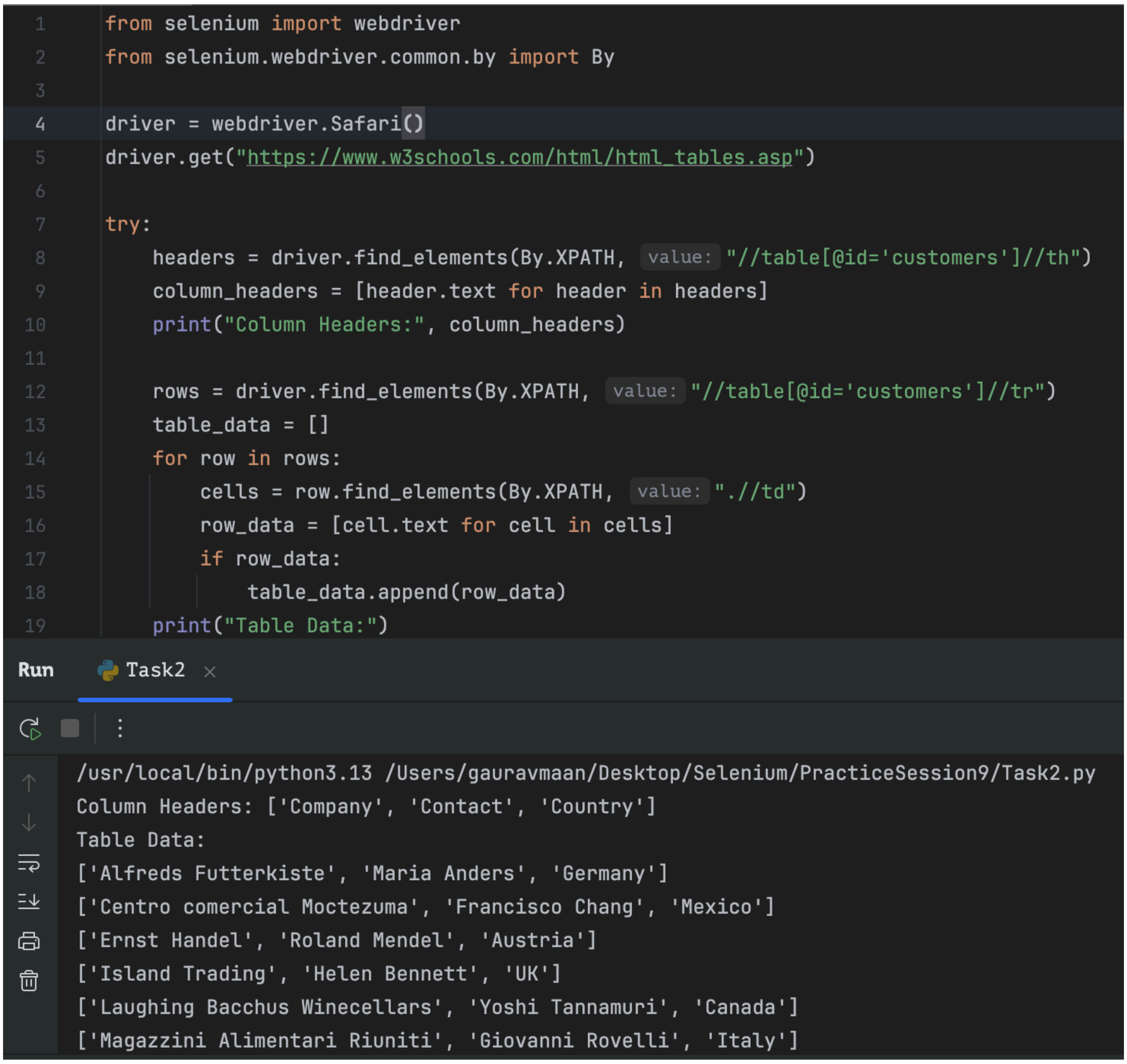
1. Open a webpage with a date picker (**https://www.expedia.com/).**
2. Select the following dates dynamically:
   - Today's date.
   - A date 7 days from today.
   - A specific date (e.g., "15th of next month").
   -
3. Validate that the selected date is reflected correctly in the date picker field.



```python

driver = webdriver.Safari()
driver.get("https://www.expedia.com/")
wait = WebDriverWait(driver, timeout: 20)

try:
    time.sleep(5)
    wait.until(EC.element_to_be_clickable((By.XPATH, "//button[contains(@id, 'd1-btn')]"))).click()

    today = datetime.today()
    day = today.strftime('%d').lstrip('0')
    month_year = today.strftime('%B %Y')

    while True:
        header = wait.until(EC.presence_of_element_located((By.XPATH, "//div[@class='uitk-new-date-picker-month']"))).text
        if month_year in header:
            break
        driver.find_element(By.XPATH, value: "//button[@aria-label='Next']").click()
    driver.find_element(By.XPATH, value: f"//button[@data-day='{day}']").click()

    future_date = today + timedelta(days=7)
    day = future_date.strftime('%d').lstrip('0')
    month_year = future_date.strftime('%B %Y')

    while True:
        header = wait.until(EC.presence_of_element_located((By.XPATH, "//div[@class='uitk-new-date-picker-month']"))).text
        if month_year in header:
            break
        driver.find_element(By.XPATH, value: "//button[@aria-label='Next']").click()
```

Task 2: Handling Static Tables

1. Navigate to a webpage containing a static table (**‘https://www.w3schools.com/html/html_tables.asp’**)
2.
3. Write a script to perform the following:
   - Retrieve all the column headers.
   - Print the entire table data in a structured format.
   - Extract and print the value of a specific cell (e.g., row 2, column 3).
   - Calculate the sum of all numeric values in a specific column.

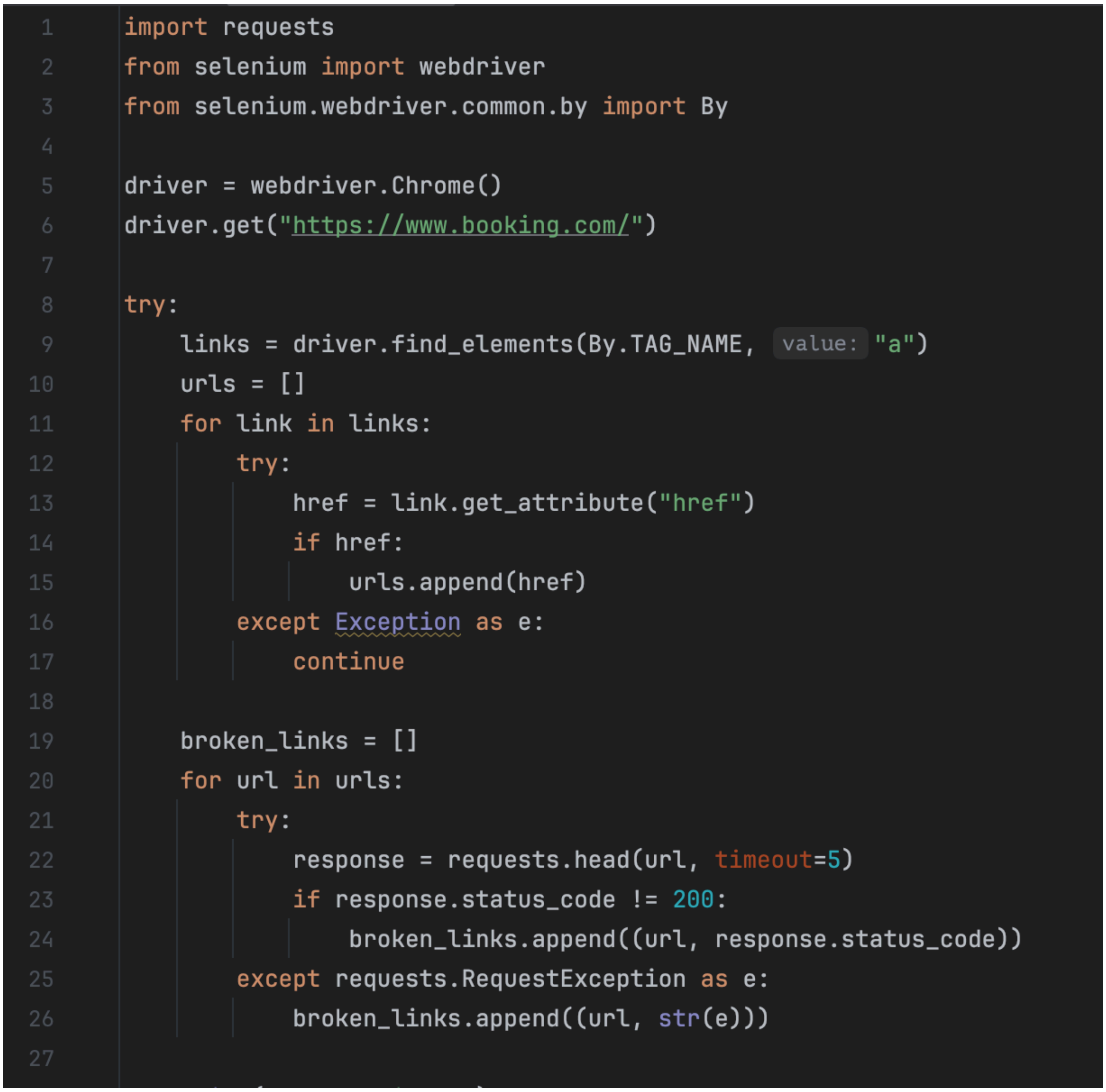


```python
from selenium import webdriver
from selenium.webdriver.common.by import By

driver = webdriver.Safari()
driver.get("https://www.w3schools.com/html/html_tables.asp")

try:
    headers = driver.find_elements(By.XPATH,  value: "//table[@id='customers']//th")
    column_headers = [header.text for header in headers]
    print("Column Headers:", column_headers)

    rows = driver.find_elements(By.XPATH,  value: "//table[@id='customers']//tr")
    table_data = []
    for row in rows:
        cells = row.find_elements(By.XPATH,  value: ".//td")
        row_data = [cell.text for cell in cells]
        if row_data:
            table_data.append(row_data)
    print("Table Data:")
```

```
/usr/local/bin/python3.13 /Users/gauravmaan/Desktop/Selenium/PracticeSession9/Task2.py
Column Headers: ['Company', 'Contact', 'Country']
Table Data:
['Alfreds Futterkiste', 'Maria Anders', 'Germany']
['Centro comercial Moctezuma', 'Francisco Chang', 'Mexico']
['Ernst Handel', 'Roland Mendel', 'Austria']
['Island Trading', 'Helen Bennett', 'UK']
['Laughing Bacchus Winecellars', 'Yoshi Tannamuri', 'Canada']
['Magazzini Alimentari Riuniti', 'Giovanni Rovelli', 'Italy']
```

Task 3: Detecting Broken Links

1. Navigate to a webpage with multiple hyperlinks. **(https://www.booking.com/)**
2. Write a script to perform the following:
    - Extract all the URLs from the page.
    - Send a HTTP request to each URL and validate the response code.
    - Print a list of broken links (URLs with response code other than 200).



```python
import requests
from selenium import webdriver
from selenium.webdriver.common.by import By

driver = webdriver.Chrome()
driver.get("https://www.booking.com/")

try:
    links = driver.find_elements(By.TAG_NAME, value: "a")
    urls = []
    for link in links:
        try:
            href = link.get_attribute("href")
            if href:
                urls.append(href)
        except Exception as e:
            continue

    broken_links = []
    for url in urls:
        try:
            response = requests.head(url, timeout=5)
            if response.status_code != 200:
                broken_links.append((url, response.status_code))
        except requests.RequestException as e:
            broken_links.append((url, str(e)))

```