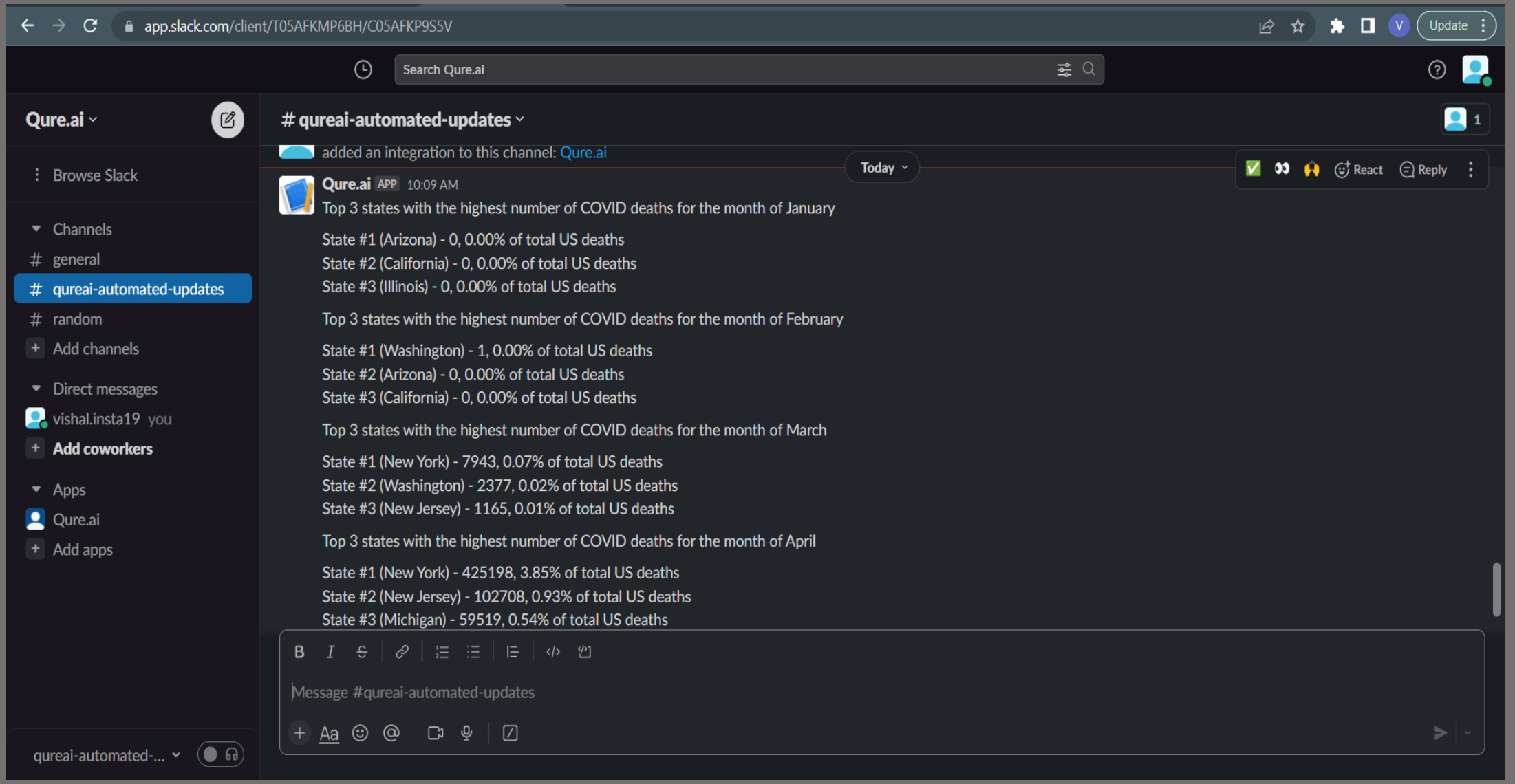
CONTENTS

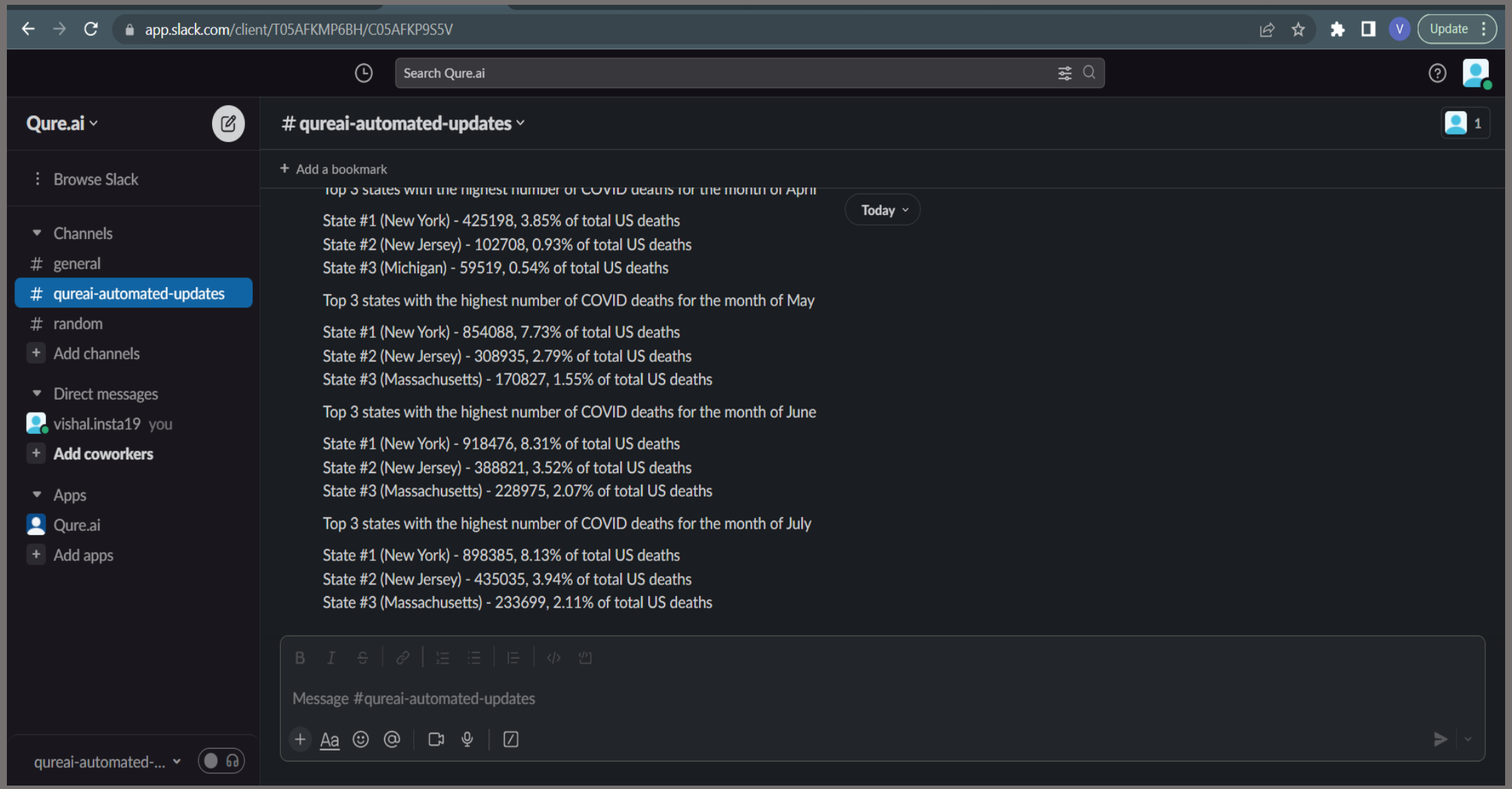
This code performs an analysis of COVID-19 state-level data and sends a message to a Slack channel with the top 3 states that have the highest number of COVID deaths for each month. Here's a breakdown of the code:

1. Import the necessary libraries: pandas for data manipulation and requests for sending HTTP requests.
2. Read the dataset from an Excel file.
3. Convert the date column to datetime format.
4. Group the data by month and state, and calculate the total deaths for each group.
5. Rename the column representing the month.
6. Sort the data by month and deaths in descending order.
7. Retrieve the top 3 states with the highest number of COVID deaths for each month.
8. Create a message string to be sent to Slack.
9. Iterate over the grouped data and format the message with the month, state rank, state name, number of deaths, and the percentage of total US deaths.
10. Append the formatted message to the overall message string.
11. Define the Slack webhook URL for sending the message.
12. Create a dictionary containing the message text.
13. Send an HTTP POST request to the Slack webhook URL with the message data.
14. Check the response status code to determine if the message was sent successfully or not.

# Message in slack



# Message in slack



MANUAL MESSAGE FOR SLACK

```
import pandas as pd
import requests

# Read the dataset
data = pd.read_excel("/content/covid-19-state-level-data (1).xlsx")

# Convert the date column to datetime format
data['date'] = pd.to_datetime(data['date'], format='%d-%m-%Y')

# Group by month and state, and calculate total deaths
monthly_state_deaths = data.groupby([data['date'].dt.month, 'state'])['deaths'].sum().reset_index()

# Rename the month column
monthly_state_deaths.rename(columns={'date': 'month'}, inplace=True)

# Sort by month and deaths in descending order
sorted_data = monthly_state_deaths.sort_values(['month', 'deaths'], ascending=[True, False])

# Get the top 3 states for each month
top_states_per_month = sorted_data.groupby('month').head(3)

# Format the message
message = ""
for month, month_data in top_states_per_month.groupby('month'):
    month_name = pd.to_datetime(month, format='%m').strftime('%B')
    message += f"Top 3 states with the highest number of COVID deaths for the month of {month_name}\n\n"

    rank = 1
    for idx, row in month_data.iterrows():
        message += f"State #{rank} ({row['state']}) - {row['deaths']}, {row['deaths'] / data['deaths'].sum() * 100:.2f}% of total US dea
        rank += 1

    message += "\n"

# Slack webhook URL
url = "https://hooks.slack.com/services/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"

# Send message to Slack
data = {'text': message}
response = requests.post(url, json=data)
if response.status_code == 200:
    print("Message sent to Slack successfully!")
else:
    print("Failed to send message to Slack. Status code:", response.status_code)
```

```
 Message sent to Slack successfully!
```

```
!pip install schedule
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting schedule
  Downloading schedule-1.2.0-py2.py3-none-any.whl (11 kB)
Installing collected packages: schedule
Successfully installed schedule-1.2.0
```

```
import pandas as pd
import requests
import schedule
import time
import calendar
from datetime import datetime, timedelta

# Read the dataset
data = pd.read_excel("/content/covid-19-state-level-data (1).xlsx")

# Convert the date column to datetime format
data['date'] = pd.to_datetime(data['date'], format='%d-%m-%Y')

# Group by month and state, and calculate total deaths
monthly_state_deaths = data.groupby([data['date'].dt.month, 'state'])['deaths'].sum().reset_index()

# Rename the month column
monthly_state_deaths.rename(columns={'date': 'month'}, inplace=True)

# Sort by month and deaths in descending order
sorted_data = monthly_state_deaths.sort_values(['month', 'deaths'], ascending=[True, False])

# Get the top 3 states for each month
top_states_per_month = sorted_data.groupby('month').head(3)

# Slack webhook URL
slack_url = "https://hooks.slack.com/services/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"

# Function to send the message to Slack
def send_slack_message(message):
    data = {'text': message}
    response = requests.post(slack_url, json=data)
    if response.status_code == 200:
        print("Message sent to Slack successfully!")
    else:
        print("Failed to send message to Slack. Status code:", response.status_code)

# Function to generate and send the data summary
def generate_data_summary():
    # Format the message
    message = ""
    for month, month_data in top_states_per_month.groupby('month'):
        month_name = calendar.month_name[month]
        message += f"Top 3 states with the highest number of COVID deaths for the month of {month_name}\n\n"

        rank = 1
        for idx, row in month_data.iterrows():
            message += f"State #{rank} ({row['state']}) - {row['deaths']}, {row['deaths'] / data['deaths'].sum() * 100:.2f}% of total US
            rank += 1

        message += "\n"

    # Send message to Slack
    send_slack_message(message)

# Function to check if it's the last day of the month
def is_last_day_of_month():
    today = datetime.now().date()
    last_day = calendar.monthrange(today.year, today.month)[1]
    return today.day == last_day

# Schedule the job to run at 23:59 every day
schedule.every().day.at("23:59").do(generate_data_summary).tag('monthly')

# Run the scheduler
while True:
    schedule.run_pending()
    time.sleep(1)
```

Executing (2m 1s) <cell line: 66>