In [ ]:
```
import psycopg2

con = psycopg2.connect(database="OBAE", user="cindy", password="Flamingosis01.", host="localhost", port="5432")

print("Database opened successfully")
```

In [ ]:
```
cur = con.cursor()
```

In [ ]:
```
import psycopg2 as pg
import folium
from folium import Choropleth, Circle, Marker
from folium.plugins import MarkerCluster
import pandas as pd
import pandas.io.sql as psql
```

In [ ]:
```
pd.DataFrame(psql.read_sql("SELECT * FROM oba_locations", con)) #Displaying raw data from dable 'stores'
```

In [ ]:
```
# join libraries with weekdays opening hours

pd.DataFrame(psql.read_sql("""
SELECT oba_locations.name ,
    open_weekdays.opening_weekdays,
    open_weekdays.opening_hours_weekdays
FROM open_weekdays
INNER JOIN oba_locations ON oba_locations.id = open_weekdays.id"""
,con))
```

In [ ]:
```
# join libraries with weekend opening hours

pd.DataFrame( psql.read_sql("""

SELECT oba_locations.name ,
    open_weekends.opening_weekends,
    open_weekends.opening_hours
FROM open_weekends
INNER JOIN oba_locations ON oba_locations.id = open_weekends.id"""
,con))
```

In [ ]:
```
# join opening weekdays and weekend hours

pd.DataFrame(psql.read_sql("""

SELECT  oba_locations.name,
    oba_locations.lat,
    oba_locations.lng ,
    open_weekends.opening_weekends,
    open_weekends.opening_hours
FROM open_weekends,
    oba_locations
WHERE open_weekends.opening_weekends = 'Sunday'
GROUP BY  oba_locations.name ,
    oba_locations.lat,
    oba_locations.lng,
    open_weekends.opening_weekends,
    open_weekends.opening_hours"""
,con))
```

In [ ]:
```
df.to_csv('Sunday_opening_libraries.csv', index=False, header=True)
```

In [ ]:
```
#find item in stores
Sunday_opening_libraries = pd.read_csv('/Users/cindymendoncapaez/opt/anaconda3/lib/python3.8/site-packages/folium/OBA/Sunday_opening_libi

# Drop rows with missing locations
Sunday_opening_libraries.dropna(subset=['lat','lng'], inplace=True)
```

In [ ]:
```
m_1 = folium.Map(location=[52.379189, 4.899431], tiles='openstreetmap', zoom_start=12)
```

In [ ]:
```
for index,row in Sunday_opening_libraries.iterrows():
    lat = row["lat"]
    lon = row["lng"]
    name = row["name"]
    opening_days = row ["opening_weekends"]
    opening_hours = row ["opening_hours"]
    map_displayed_info = '{} : {} : {}'.format(name, opening_days, opening_hours)
    folium.Marker([lat,lon],popup=map_displayed_info).add_to(m_1)
m_1
```

In [ ]:
```
pd.DataFrame(psql.read_sql("""

SELECT oba_locations.name ,
    restaurants_oba.name,
    restaurants_oba.address,
    restaurants_oba.opening_hours,
    restaurants_oba.lat,
    restaurants_oba.lng
FROM   join_restaurant_libraries
INNER JOIN oba_locations ON join_restaurant_libraries.restaurants_id = join_restaurant_libraries.restaurants_id
INNER JOIN restaurants_oba ON oba_locations.id = join_restaurant_libraries.libraries_id
LIMIT 5"""
,con))
```

In [ ]:
```
#find item in stores
find_restaurants = pd.read_csv('/Users/cindymendoncapaez/opt/anaconda3/lib/python3.8/site-packages/folium/OBA/find_restaurants.csv')

# Drop rows with missing locations
find_restaurants.dropna(subset=['lat','lng'], inplace=True)
```

In [ ]:
```
m_2 = folium.Map(location=[52.379189, 4.899431], tiles='openstreetmap', zoom_start=13)
```

In [ ]:
```
for index,row in find_restaurants.iterrows():
    lat = row["lat"]
    lon = row["lng"]
    name = row["name"]
    address= row ["address"]
    opening_hours = row ["opening_hours"]
    map_displayed_info = '{} : {} : {}'.format(name, address, opening_hours)
    folium.Marker([lat,lon],popup=map_displayed_info).add_to(m_2)
m_2
```

In [ ]:
```
#join books with libraries

pd.DataFrame(psql.read_sql("""

SELECT
  books.id,
  books.title,
  oba_locations.name,
  oba_locations.lat,
  oba_locations.lng,
  join_oba_books.book_id
FROM join_oba_books
JOIN books
ON  books.id = join_oba_books.book_id
JOIN oba_locations
ON oba_locations.id = join_oba_books.library_id"""
, con))
```

In [ ]:
```
#search for book Orkael in the libraries

pd.DataFrame(psql.read_sql("""

SELECT
  join_books_lat_lng.id,
  join_books_lat_lng.title,
  join_books_lat_lng.name,
  join_books_lat_lng.book_id,
  join_books_lat_lng.lat,
  join_books_lat_lng.lng
FROM join_books_lat_lng
WHERE join_books_lat_lng.title = 'Orkael'"""

, con))
```

In [ ]:
```
#find item in stores
find_book = pd.read_csv('/Users/cindymendoncapaez/opt/anaconda3/lib/python3.8/site-packages/folium/OBA/find_orkadel.csv')

# Drop rows with missing locations
find_book.dropna(subset=['lat','lng'], inplace=True)
```

In [ ]:
```
m_3 = folium.Map(location=[52.379189, 4.899431], tiles='openstreetmap', zoom_start=12)
```

In [ ]:
```
for index,row in find_book.iterrows():
    lat = row["lat"]
    lon = row["lng"]
    title= row ["title"]
    name = row["name"]
    map_displayed_info = '{} : {}'.format(name, title)
    folium.Marker([lat,lon],popup=map_displayed_info).add_to(m_3)
m_3
```

In [ ]:
```
#join books and categories

pd.DataFrame(psql.read_sql(
"""
SELECT
  books.id,
  books.title,
  categories.genres,
  join_books_categories.genres_id,
  join_books_categories.item_id
FROM join_books_categories
JOIN books
ON  books.id = join_books_categories.item_id
JOIN categories
ON categories.id = join_books_categories.genres_id

"""
,con))
```

In [ ]:

```
# Find book Something needs to change

pd.DataFrame(psql.read_sql(
"""
SELECT
  genres_books.title,
  genres_books.genres,
  oba_locations.name,
  oba_locations.lat,
  oba_locations.lng,
  join_books_genres_loc.book_id,
  join_books_genres_loc.location_id

FROM join_books_genres_loc
JOIN genres_books
ON  genres_books.id = join_books_genres_loc.book_id
JOIN oba_locations
ON oba_locations.id = join_books_genres_loc.location_id
WHERE genres_books.title = 'Something needs to change'

"""
,con))
```

In [ ]:

```
#reservation system for books in OBA Olympic Quarter

def print_reservation():
    """Print the tickets of the user."""
    for user_name, books in user_reservation.items():
        print(f"\nYou, {user_name.title()}, have chosen {len(books)} book(s).")
        for book in books:
            print(f"\tbook name: {book}")


# Empty dictionary to store info later on.
user_reservation = {}

# List of books the user can choose from.
available_books = ['Something needs to change']


# All prompts.
start_prompt = "\nWould you like to make a new reservation? (yes/no) "
wanted_books_prompt = "\nHow many book(s) would you like to reservate today?"
wanted_books_prompt += "\nEnter the number: "
number_prompt = "What is your OBA membership number? "
book_prompt = "\nPlease enter the book's title: "
go_again_prompt = "Would you like to make other reservation (yes/no) "



print("Welcome To The OBA Seat Booking Portal!")

# Ask the user if he would like to start the books reservations.
start = input(start_prompt)
if start.lower() == 'yes':
    # Runs until it reaches a break statement.
    while True:
        # Empty list to store the book(s) the user has chosen.
        books = []

        # Find out how many times to run the while loop.
        wanted_books = input(wanted_books_prompt)
        # Convert the string representation of the number to an integer representation.
        wanted_books = int(wanted_books)
        # If the user has asked for more books than the number of books
        # available execute this block.
        if wanted_books_prompt > str(available_books):
            print(f"\n--I'm sorry, we only have {len(available_books)} "
                "book(s) available--")
            print("--Please try again--")
            continue

        # Ask for the OBA member number.
        user_number = input(number_prompt)

        # Run until the user has chosen the requested number of books.
        while True:

            # Show the user the available books.
            print("\nHere are the available book(s):")
            for seat in available_books:
                print(seat)

            # Ask the user for their chosen seat number.
            seat = input(book_prompt)

            # If the user has entered a seat that is in the 'available_books'
            # list; remove it from the 'available_books' list.
            if seat in available_books:
                available_books.remove(seat)
            # The user has entered a seat that is not in the 'available_books' list.
            # Ask for their book reservation again.
            else:
                print("\n--I'm sorry you have chosen an invalid book(s)--"
                    "\n-Please try again-")
                continue

            # Add the chosen seat to the 'books' list
            books.append(books)

            # If the user has said that they are going to book more than one book
            # go through this book booking 'while' loop again.
            if wanted_books_prompt == books:
                print("\nYou can now choose another seat.")
                # The loop will run a limited number of times.
                # It will only 'continue' when there is more than one 'wanted_book'.
                wanted_books_prompt-=1
                continue
            else:
                break

        # Add the 'user_number' variable and 'books' list to the 'user_reservations' dictionary.
        user_reservation[user_number] = books


        #If their are any available books left ask the user if he
        # wants to let another person book their reservation.
        if available_books:
            go_again = input(go_again_prompt)
            if go_again == 'no':
                break
        else:
            break

    print_reservation()
    print("\nYour reservation is complete!"
```

```
        "\nYou can see all the details on 'My reservations' page")

else:
    print("Try again later")
```

In [ ]:
```
#find item in stores
find_book_category = pd.read_csv('/Users/cindymendoncapaez/opt/anaconda3/lib/python3.8/site-packages/folium/OBA/joined_books_genres_lat_

# Drop rows with missing locations
find_book_category.dropna(subset=['lat','lng'], inplace=True)
```

In [ ]:
```
m_4 = folium.Map(location=[52.379189, 4.899431], tiles='openstreetmap', zoom_start=12)
```

In [ ]:
```
for index,row in find_book_category.iterrows():
    lat = row["lat"]
    lon = row["lng"]
    title= row ["title"]
    name = row["name"]
    map_displayed_info = '{} : {}'.format(name, title)
    folium.Marker([lat,lon],popup=map_displayed_info).add_to(m_4)
m_4
```

In [*]:
```
# available seats

pd.DataFrame(psql.read_sql(
"""
SELECT
  oba_locations.name,
  floors.floors,
  floors.seat,
  floors.session,
  oba_locations.lat,
  oba_locations.lng,
  join_library_floors.libraries_id,
  join_library_floors.floors_id

FROM join_library_floors
JOIN floors
ON  floors.id = join_library_floors.libraries_id
JOIN oba_locations
ON oba_locations.id = join_library_floors.floors_id


"""
,con))
```

In [*]:

```
#Reservation system (seats)

def print_reservation():
    """Print the tickets of the user."""
    for user_name, seats in user_reservation.items():
        print(f"\nYou, {user_name.title()}, have chosen {len(seats)} seat(s).")
        for seat in seats:
            print(f"\tSeat number: {seat}")


# Empty dictionary to store info later on.
user_reservation = {}

# List of seats the user can choose from.
available_seats = ['AE-1', 'AE-2', 'AE-3', 'BT-5', 'BT-6', 'BT-1', 'PP-0', 'PP-5']


# All prompts.
start_prompt = "\nWould you like to start booking your seat? (yes/no) "
wanted_seats_prompt = "\nHow many seats are you booking today?"
wanted_seats_prompt += "\nEnter the number: "
number_prompt = "What is your OBA membership number? "
seat_prompt = "\nPlease enter the seat you would like to book: "
go_again_prompt = "Would you like to let someone else book their tickets? (yes/no) "


print("Welcome To The OBA Seat Booking Portal!")

# Ask the user if he would like to start booking their tickets.
start = input(start_prompt)
if start.lower() == 'yes':
    # Runs until it reaches a break statement.
    while True:
        # Empty list to store the seat(s) the user has chosen.
        seats = []

        # Find out how many times to run the while loop.
        wanted_seats = input(wanted_seats_prompt)
        # Convert the string representation of the number to an integer representation.
        wanted_seats = int(wanted_seats)
        # If the user has asked for more seats than the number of seats
        # available execute this block.
        if wanted_seats > len(available_seats):
            print(f"\n--I'm sorry, we only have {len(available_seats)} "
                "seats available--")
            print("--Please try again--")
            continue

        # Ask for the OBA member number.
        user_number = input(number_prompt)

        # Run until the user has chosen the requested number of seats.
        while True:

            # Show the user the available seats.
            print("\nHere are the available seats:")
            for seat in available_seats:
                print(seat)

            # Ask the user for their chosen seat number.
            seat = input(seat_prompt)

            # If the user has entered a seat that is in the 'available_seats'
            # list; remove it from the 'available_seats' list.
            if seat in available_seats:
                available_seats.remove(seat)
            # The user has entered a seat that is not in the 'avialbe_seats' list.
            # Ask for their seat again.
            else:
                print("\n--I'm sorry you have chosen an invalid seat--"
                    "\n-Please try again-")
                continue

            # Add the chosen seat to the 'seats' list
            seats.append(seat)

            # If the user has said that they are going to book more than one seat
            # go through this seat booking 'while' loop again.
            if wanted_seats > 1:
                print("\nYou can now choose another seat.")
                # The loop will run a limited number of times.
                # It will only 'continue' when there is more than one 'wanted_seat'.
                wanted_seats-=1
                continue
            else:
                break

        # Add the 'user_number' variable and 'seats' list to the 'user_reservations' dictionary.
        user_reservation[user_number] = seats

        #If their are any available seats left ask the user if he
        # wants to let another person book their tickets.
        if available_seats:
            go_again = input(go_again_prompt)
            if go_again == 'no':
                break
        else:
            break

    print_reservation()
    print("\nYour reservation is complete!"
        "\nYou can see all the details on 'My reservations' page")
```

```
else:
    print("Try again later")
```