Import necessary libraries

In [1]:
```
import os
import openai
```

Retrieve API key from environment variables

This line sets the OpenAI API key by accessing it from the environment variable 'OPENAI_API_KEY'.

In [2]:
```
openai.api_key = os.environ['OPENAI_API_KEY']
```

This code initializes an OpenAI client using the API key fetched from the environment variable 'OPENAI_API_KEY'.

In [3]:
```
from openai import OpenAI
client = OpenAI(
  api_key=os.environ['OPENAI_API_KEY'],  # this is also the default, it can be omitted
)
```

Function to create a prompt for generating a recipe based on a list of ingredients and a category

This function generates a prompt for creating a recipe, incorporating a list of ingredients and requiring a title that begins with 'Recipe Title: '.

In [4]:
```
def create_dish_prompt(list_of_ingredients):
    # Construct a prompt using the provided ingredients list and category
    prompt = f"Create a detailed recipe using the following ingredients: {', '.join(list_of_ingredients)}.\n"\
             + f"Additionally, assign a title starting with 'Recipe Title: ' to this recipe."
    return prompt
```

Entering the Ingredients

This code generates a recipe prompt using a list of ingredients for a dessert dish.

In [5]:
```
recipe = create_dish_prompt(['all-purpose flour','granulated sugar', 'icecream','unsalted butter','chololate syrup','eggs','milk','vanilla extract','baking powder','salt
```

In [6]:
```
recipe
```

Out[6]:
```
"Create a detailed recipe using the following ingredients: all-purpose flour, granulated sugar, icecream, unsalted butter, chololate syrup, eggs, milk, vanilla extrac
t, baking powder, salt, unsweetened cocoa powder, strong brewed coffee, brown sugar, chopped nuts, carrots.\nAdditionally, assign a title starting with 'Recipe Title:
' to this recipe."
```

In [7]:
```
#i = ['eggs','bacon','bread']
```

In [8]:
```
#', '.join(i)
```

This code initiates a chat completion request using the generated recipe prompt within a chat context using the OpenAI GPT-3.5 Turbo model.

In [9]:
```
# Use the prompt in the completion code by initiating a chat completion request
completion = client.chat.completions.create(
    messages=[
        {
            "role": "user",
            "content": recipe,
        }
    ],
    model="gpt-3.5-turbo",
)
```

In [10]:
```
# Extract the content from the completion
content = completion.choices[0].message.content
```

In [11]:
```
# Split the content into recipe title and detailed recipe
split_content = content.split('\n', 1)  # Split into two parts based on the first newline
recipe_title = split_content[0].strip()  # Extract the recipe title
detailed_recipe = split_content[1].strip()  # Extract the detailed recipe content
```

In [12]:
```
# Print the extracted parts
print(recipe_title)
print("\nDetailed Recipe:")
print(detailed_recipe)
```

```
Recipe Title: Chocolate Carrot Cake with Coffee Buttercream Frosting

Detailed Recipe:
Ingredients:
- 2 cups all-purpose flour
- 1 ½ cups granulated sugar
- 1 cup unsalted butter, softened
- 1 cup ice cream (vanilla or your choice), melted
- ¾ cup chocolate syrup
- 4 large eggs
- ½ cup milk
- 2 tsp vanilla extract
- 2 tsp baking powder
- ½ tsp salt
- ¼ cup unsweetened cocoa powder
- ¼ cup strong brewed coffee, cooled
- 1 ½ cups grated carrots
- 1 cup brown sugar
- 1 cup chopped nuts (pecans or walnuts)

For the Coffee Buttercream Frosting:
- 1 cup unsalted butter, softened
- 4 cups powdered sugar
- 2 tbsp strong brewed coffee, cooled
- 1 tsp vanilla extract

Instructions:
1. Preheat the oven to 350°F (175°C). Grease and lightly flour two 9-inch round cake pans.

2. In a large mixing bowl, cream together the granulated sugar and softened unsalted butter until light and fluffy.

3. Add the melted ice cream and chocolate syrup to the creamed mixture. Mix well until incorporated.

4. Beat in the eggs, one at a time, until fully combined. Stir in the milk and vanilla extract.

5. In a separate bowl, whisk together the all-purpose flour, cocoa powder, baking powder, and salt.

6. Gradually add the dry ingredients to the wet ingredients, mixing until just combined. Avoid overmixing.

7. Fold in the grated carrots and chopped nuts into the batter, ensuring they are evenly distributed.

8. Divide the batter equally between the prepared cake pans.

9. Bake for approximately 30-35 minutes, or until a toothpick inserted into the center comes out clean.

10. Remove the cakes from the oven and allow them to cool in the pans for 10 minutes. Then, transfer them to a wire rack to cool completely.

11. Meanwhile, prepare the Coffee Buttercream Frosting. In a mixing bowl, beat the softened unsalted butter until creamy.

12. Gradually add in the powdered sugar, one cup at a time, beating well after each addition. Add the cooled brewed coffee and vanilla extract and continue beating unt
il the frosting is smooth and fluffy.

13. Once the cakes have cooled, place one cake layer on a serving plate or cake stand. Spread a thick layer of Coffee Buttercream Frosting over the top of this layer.

14. Gently place the second cake layer on top and frost the entire cake with the remaining Coffee Buttercream Frosting. Smooth the frosting with a spatula for an even
finish.

15. Optional: Garnish the cake with additional grated carrots, chopped nuts, or a drizzle of chocolate syrup.

16. Allow the cake to set in the refrigerator for at least 1 hour before serving.

This Chocolate Carrot Cake with Coffee Buttercream Frosting is a delightful blend of flavors that will satisfy any chocolate or coffee lover's cravings. Enjoy this moi
st and rich cake as a delightful dessert or a special treat for any occasion.
```

In [13]:
```
#import re
```

In [14]:
```
#pip install Pillow
```

In [15]:
```
print(recipe_title)
```

```
Recipe Title: Chocolate Carrot Cake with Coffee Buttercream Frosting
```

Generating Image of the Cooked Recipe

This function uses OpenAI's DALL·E model to generate an image based on a provided recipe title, producing an image URL as output.

In [16]:
```
def generate_dalle_image(recipe_title):
    response = client.images.generate(
        model="dall-e-3",
        prompt=recipe_title,
        size="1024x1024",
        quality="standard",
        n=1,
    )
    url = response.data[0].url if response.data else None
    return url
```

In [17]:
```
url = generate_dalle_image(recipe_title)
```

Displaying the image within the code

In [18]:
```
from IPython.display import display, Image
```

This code generates an image using DALL·E based on a recipe title, displays the image if the URL exists, and prints a message if no image URL is generated.

In [19]:

```
# Call the function and store the URL
url = generate_dalle_image(recipe_title,)
# Display the image if URL exists
if url:
    display(Image(url=url))
else:
    print("No image URL generated.")
```