Healthy Eating Dataset Analysis

DataSet Source: https://www.kaggle.com/datasets/khushikyad001/healthy-eating-dataset

In [ ]:
```
Name: G.Madhupriya
RollNo: 2211CS010633
Section: S1 - 86
```

In [ ]:
```
1.Dataset Overview:
    File Name: healthy_eating_dataset.csv
    Rows: 2000
    Columns: 20
    Category: Nutrition + Lifestyle
    Purpose: Analyze meal nutritional information, cooking patterns, dietary prefer

2.DataSet Columns and Datatypes:
    i. meal_id - an integer column representing the unique ID of each meal.
    ii. meal_name - a string column containing the name of the meal.
    iii. cuisine - a string column representing the cuisine type (e.g., Indian, Ita
    iv. meal_type - a string column indicating the type of meal such as Breakfast,
    v. diet_type - a string column showing whether the meal is Vegetarian, Vegan, o
    vi. calories - an integer column representing the calorie content of the meal.
    vii. protein_g - a float column representing the protein amount (in grams).
    viii. carbs_g - a float column representing the carbohydrate content (in grams)
    ix. fat_g - a float column showing the fat content (in grams).
    x. fiber_g - a float column showing dietary fiber (in grams).
    xi. sugar_g - a float column representing sugar content (in grams).
    xii. sodium_mg - an integer column representing sodium content (in milligrams).
    xiii. cholesterol_mg - an integer column showing cholesterol amount (in milligr
    xiv. serving_size_g - an integer column showing serving size (in grams).
    xv. cooking_method - a string column describing how the meal is cooked (e.g., B
    xvi. prep_time_min - an integer column representing preparation time in minutes
    xvii. cook_time_min - an integer column representing cooking time in minutes.
    xviii. rating - a float column indicating the average user rating of the meal.
    xix. is_healthy - an integer column indicating if the meal is considered health
    xx. image_url - a string column containing a URL link to the meal image.

3.Data Quality Notes;
    Missing Values: Some may exist in columns like rating, fiber_g, or sugar_g-thes
    Duplicates: Possible duplicates may exist based on meal name or ID-require remo
    Outliers: Likely in calories, fat_g, and protein_g (meals with extremely high o

4.Categorical Column Distributions (Example):
    i. cuisine - The most common cuisines are Indian, Italian, and Chinese.
    ii. meal_type - The primary meal types are Breakfast, Lunch, and Dinner.
    iii. diet_type - The main diet categories include Vegetarian, Vegan, and Non-Ve
    iv. cooking_method - Common cooking methods are Grilled, Boiled, and Fried.
    v. is_healthy - Most meals are labeled as healthy (1).
```

In [7]:
```
# -------- 1. IMPORT LIBRARIES --------
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# -------- 2. LOAD DATA --------
file_path = "healthy_eating_dataset.csv"
df = pd.read_csv(file_path)

# -------- 3. BASIC INFO --------
print("Dataset Shape:", df.shape)
print("\nDataset Preview:\n", df.head())
print("\nDataset Info:\n")
print(df.info())
print("\nMissing Values:\n", df.isnull().sum())
```

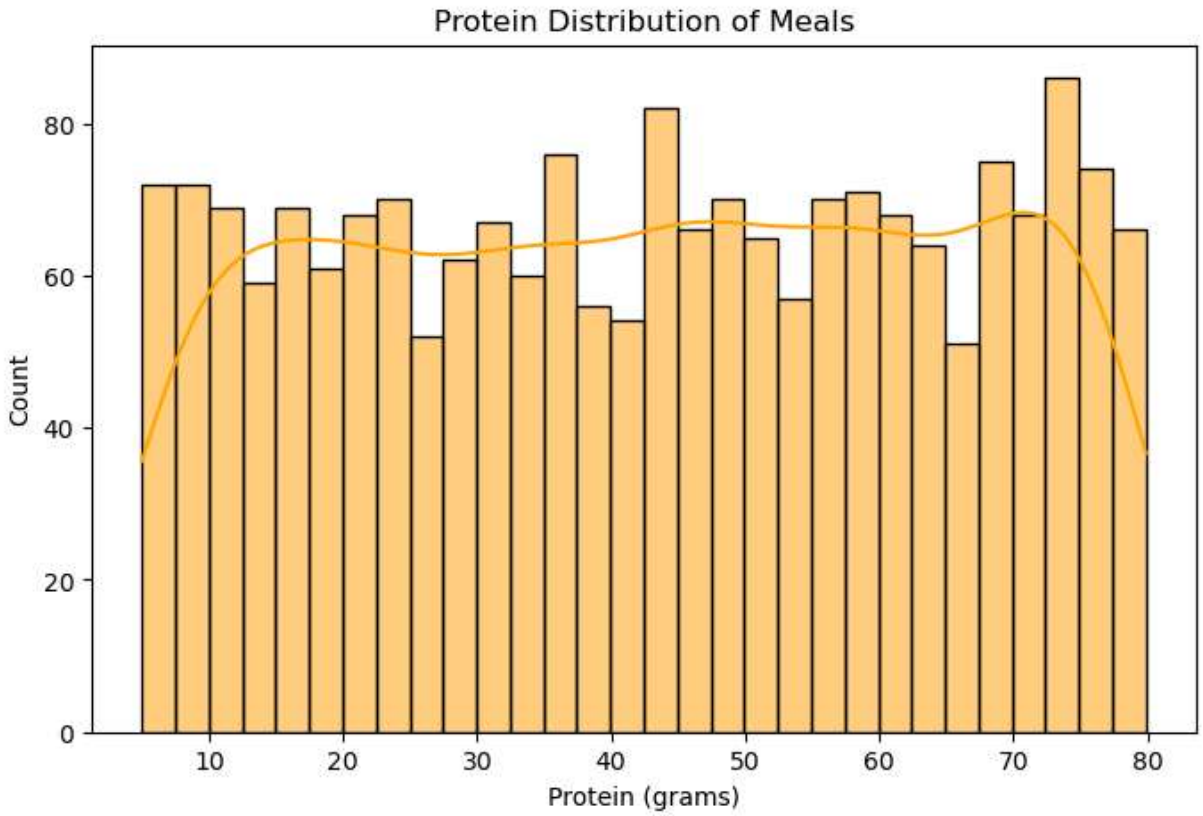
```
Dataset Shape: (2000, 20)

Dataset Preview:
    meal_id      meal_name  cuisine meal_type diet_type  calories  protein_g  \
0         1      Kid Pasta   Indian     Lunch      Keto       737       52.4
1         2   Husband Rice  Mexican     Lunch     Paleo       182       74.7
2         3  Activity Rice   Indian     Snack     Paleo       881       52.9
3         4  Another Salad  Mexican     Snack      Keto       427       17.5
4         5     Quite Stew     Thai     Lunch     Vegan       210       51.6

   carbs_g  fat_g  fiber_g  sugar_g  sodium_mg  cholesterol_mg  \
0     43.9   34.3     16.8     42.9       2079              91
1    144.4    0.1     22.3     38.6        423               7
2     97.3   18.8     20.0     37.5       2383             209
3     73.1    7.6      9.8     41.7        846             107
4    104.3   26.3     24.8     18.2       1460              42

   serving_size_g cooking_method  prep_time_min  cook_time_min  rating  \
0             206        Grilled             47             56     4.4
1             317        Roasted             51             34     2.4
2             395         Boiled             58             29     4.3
3             499        Grilled             14             81     4.6
4             486            Raw             47            105     4.3

   is_healthy                              image_url
0           0  https://example.com/images/meal_1.jpg
1           0  https://example.com/images/meal_2.jpg
2           0  https://example.com/images/meal_3.jpg
3           0  https://example.com/images/meal_4.jpg
4           0  https://example.com/images/meal_5.jpg

Dataset Info:

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2000 entries, 0 to 1999
Data columns (total 20 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   meal_id         2000 non-null   int64
 1   meal_name       2000 non-null   object
 2   cuisine         2000 non-null   object
 3   meal_type       2000 non-null   object
 4   diet_type       2000 non-null   object
 5   calories        2000 non-null   int64
 6   protein_g       2000 non-null   float64
 7   carbs_g         2000 non-null   float64
 8   fat_g           2000 non-null   float64
 9   fiber_g         2000 non-null   float64
 10  sugar_g         2000 non-null   float64
 11  sodium_mg       2000 non-null   int64
 12  cholesterol_mg  2000 non-null   int64
 13  serving_size_g  2000 non-null   int64
 14  cooking_method  2000 non-null   object
 15  prep_time_min   2000 non-null   int64
 16  cook_time_min   2000 non-null   int64
 17  rating          2000 non-null   float64
```

```
 18  is_healthy      2000 non-null   int64
 19  image_url       2000 non-null   object
dtypes: float64(6), int64(8), object(6)
memory usage: 312.6+ KB
None

Missing Values:
 meal_id           0
meal_name          0
cuisine            0
meal_type          0
diet_type          0
calories           0
protein_g          0
carbs_g            0
fat_g              0
fiber_g            0
sugar_g            0
sodium_mg          0
cholesterol_mg     0
serving_size_g     0
cooking_method     0
prep_time_min      0
cook_time_min      0
rating             0
is_healthy         0
image_url          0
dtype: int64
```

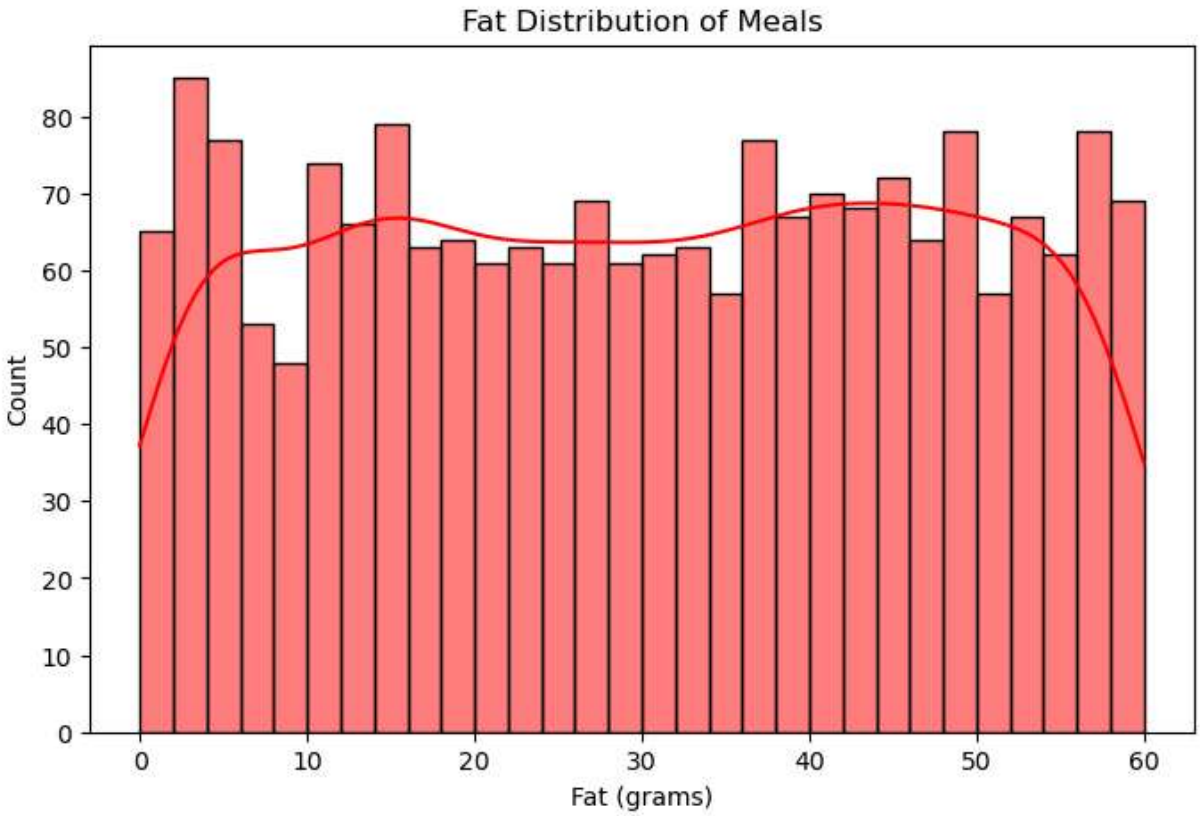
In [11]:

```
# -------- 4. VISUALIZATIONS --------

# ---- Calories Distribution ----
plt.figure(figsize=(8,5))
sns.histplot(df['calories'], bins=30, kde=True, color='green')
plt.title("Calories Distribution of Meals")
plt.xlabel("Calories")
plt.ylabel("Count")
plt.show()
```

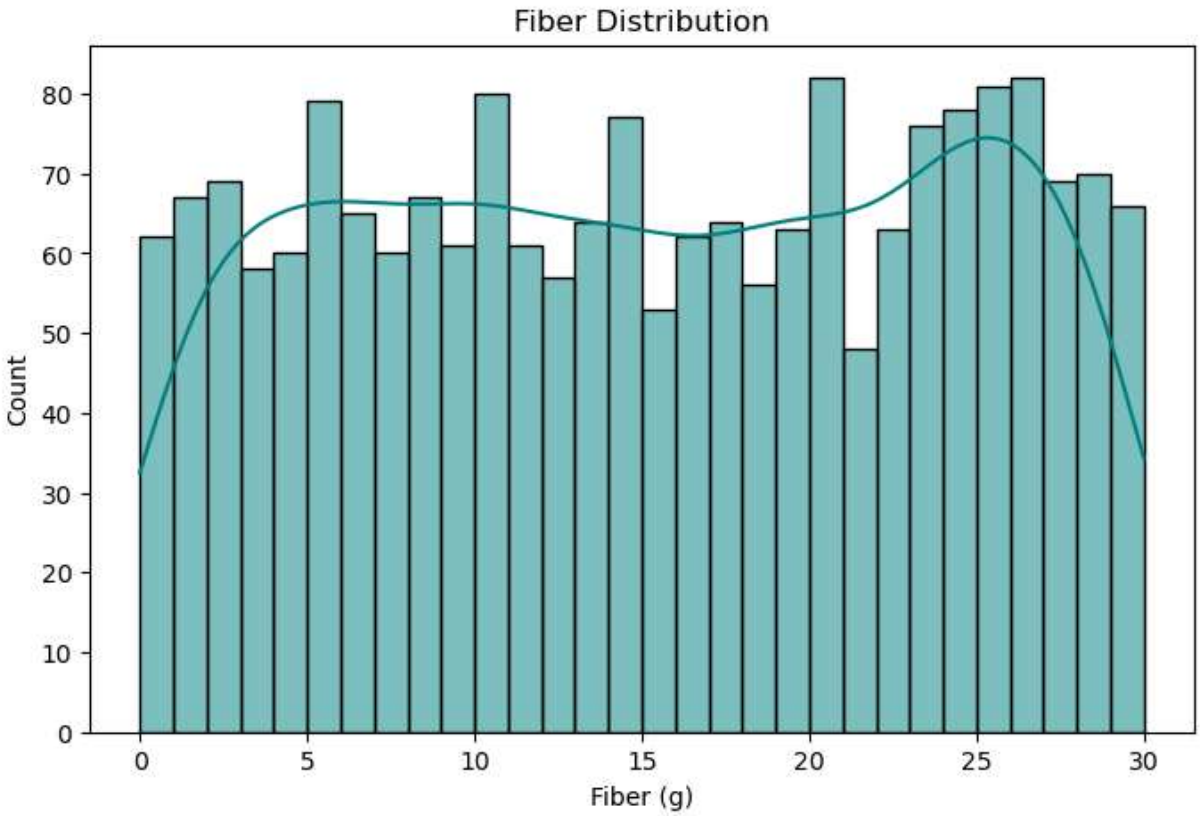
In [13]:
```
#---Protein Distribution--------
plt.figure(figsize=(8,5))
sns.histplot(df['protein_g'], bins=30, kde=True, color='orange')
plt.title("Protein Distribution of Meals")
plt.xlabel("Protein (grams)")
plt.ylabel("Count")
plt.show()
```



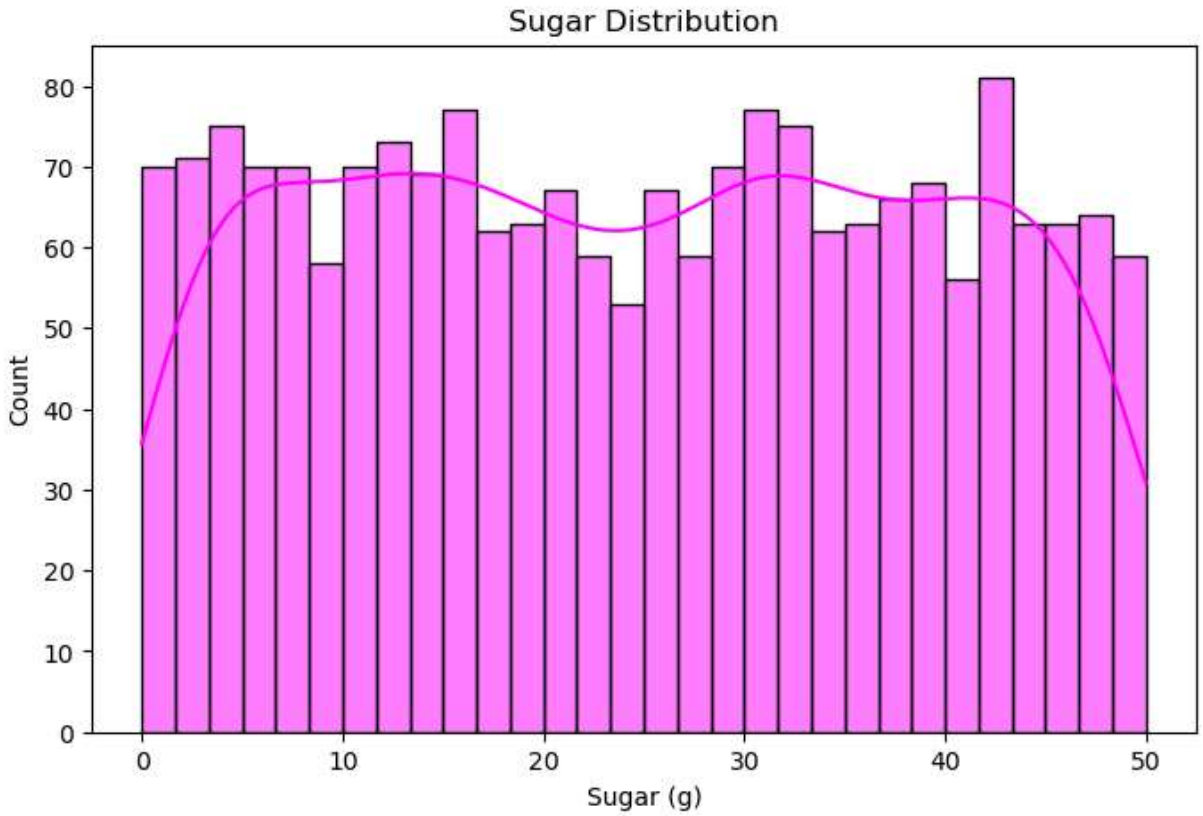
In [15]:
```
#----Fat Distribution------
plt.figure(figsize=(8,5))
sns.histplot(df['fat_g'], bins=30, kde=True, color='red')
plt.title("Fat Distribution of Meals")
plt.xlabel("Fat (grams)")
plt.ylabel("Count")
plt.show()
```



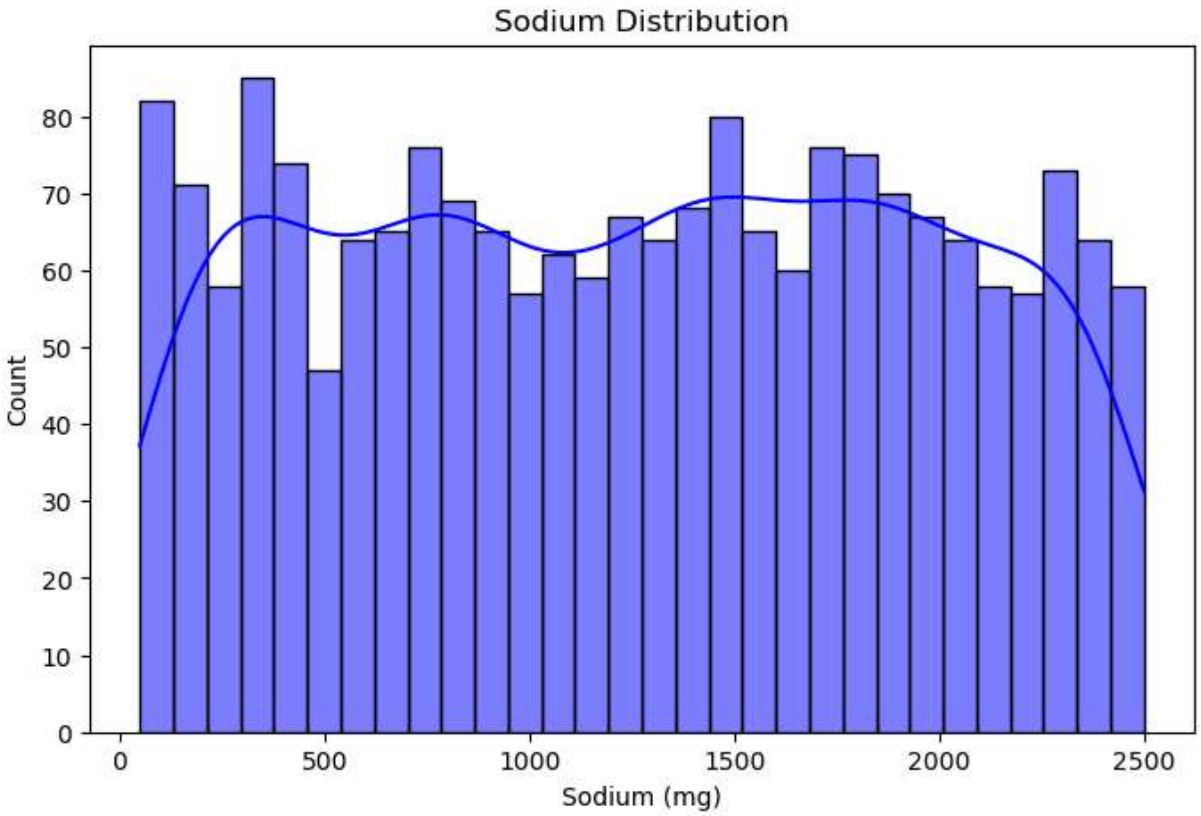
In [23]:
```
#----Fiber Distribution------
plt.figure(figsize=(8,5))
sns.histplot(df['fiber_g'], bins=30, kde=True, color='teal')
plt.title("Fiber Distribution")
plt.xlabel("Fiber (g)")
plt.ylabel("Count")
plt.show()
```



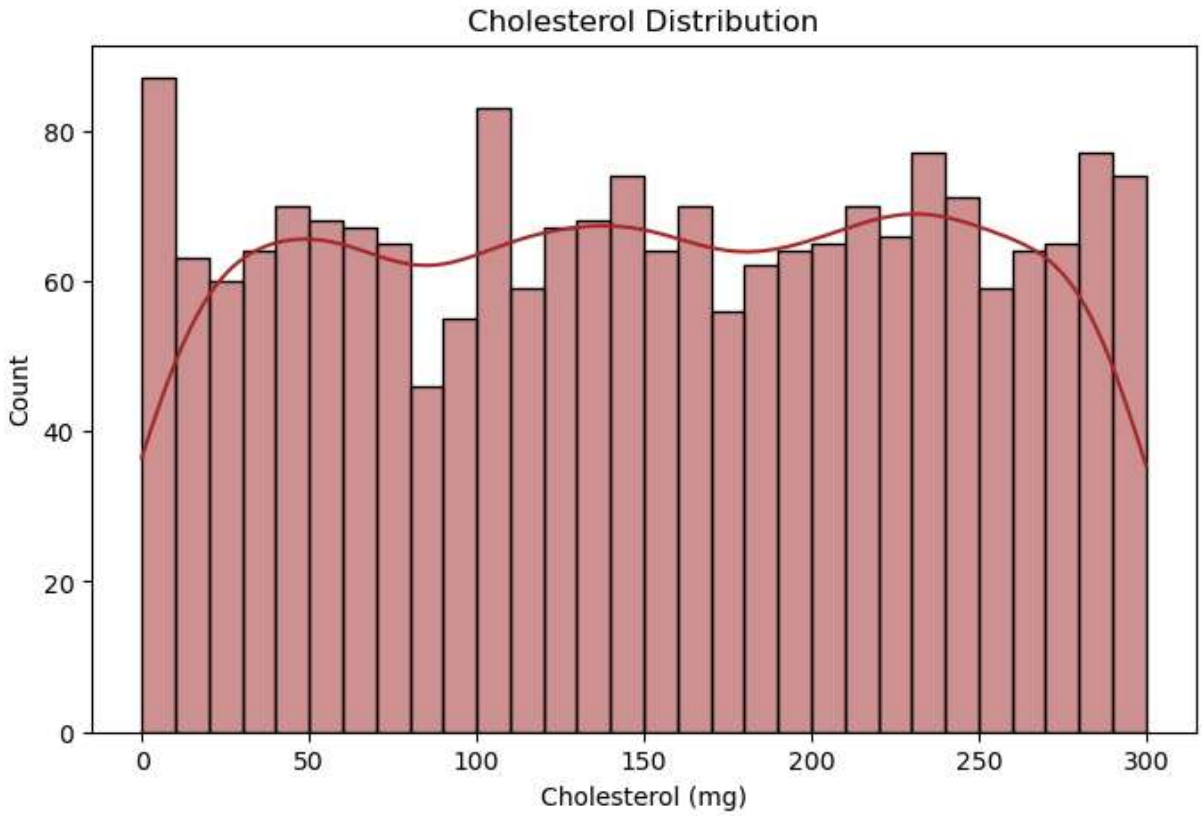
In [25]:
```
#-----sugar Distribution-----
plt.figure(figsize=(8,5))
sns.histplot(df['sugar_g'], bins=30, kde=True, color='magenta')
plt.title("Sugar Distribution")
plt.xlabel("Sugar (g)")
plt.ylabel("Count")
plt.show()
```



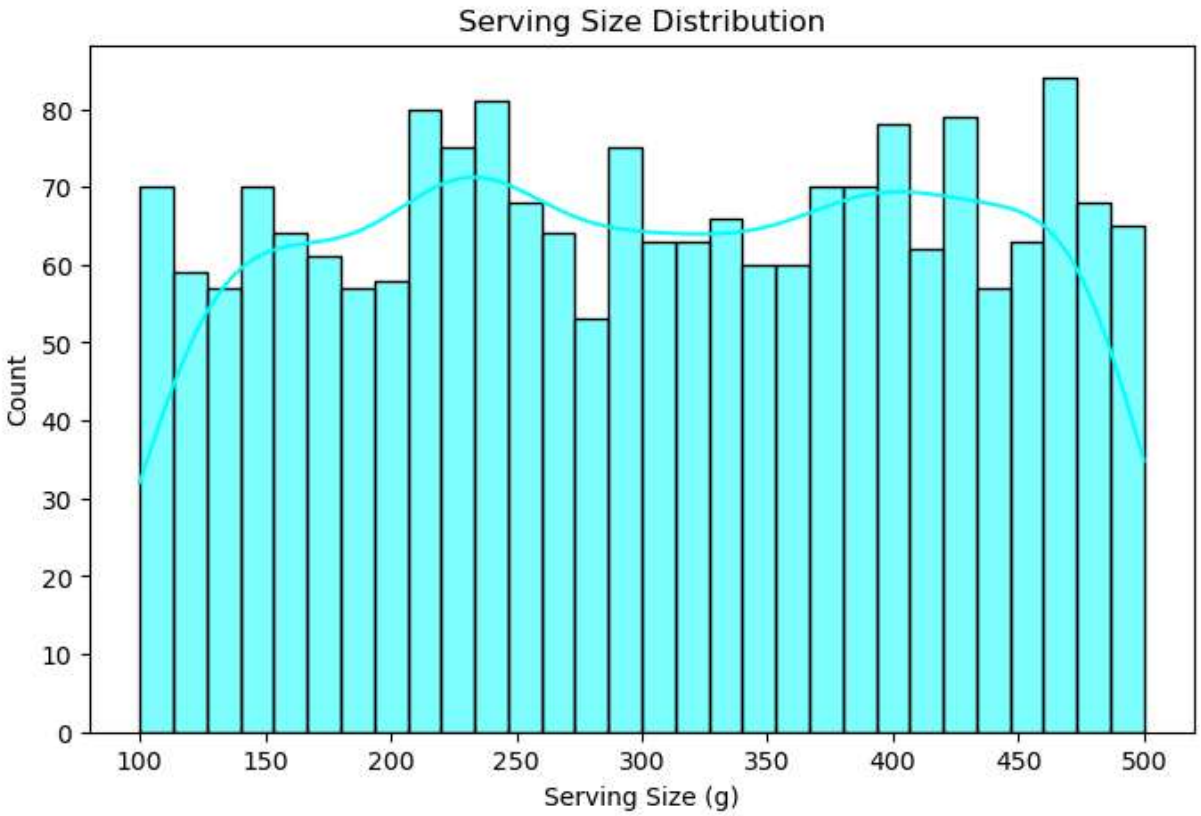
In [27]:
```
#-----Sodium Distribution------
plt.figure(figsize=(8,5))
sns.histplot(df['sodium_mg'], bins=30, kde=True, color='blue')
plt.title("Sodium Distribution")
plt.xlabel("Sodium (mg)")
plt.ylabel("Count")
plt.show()
```



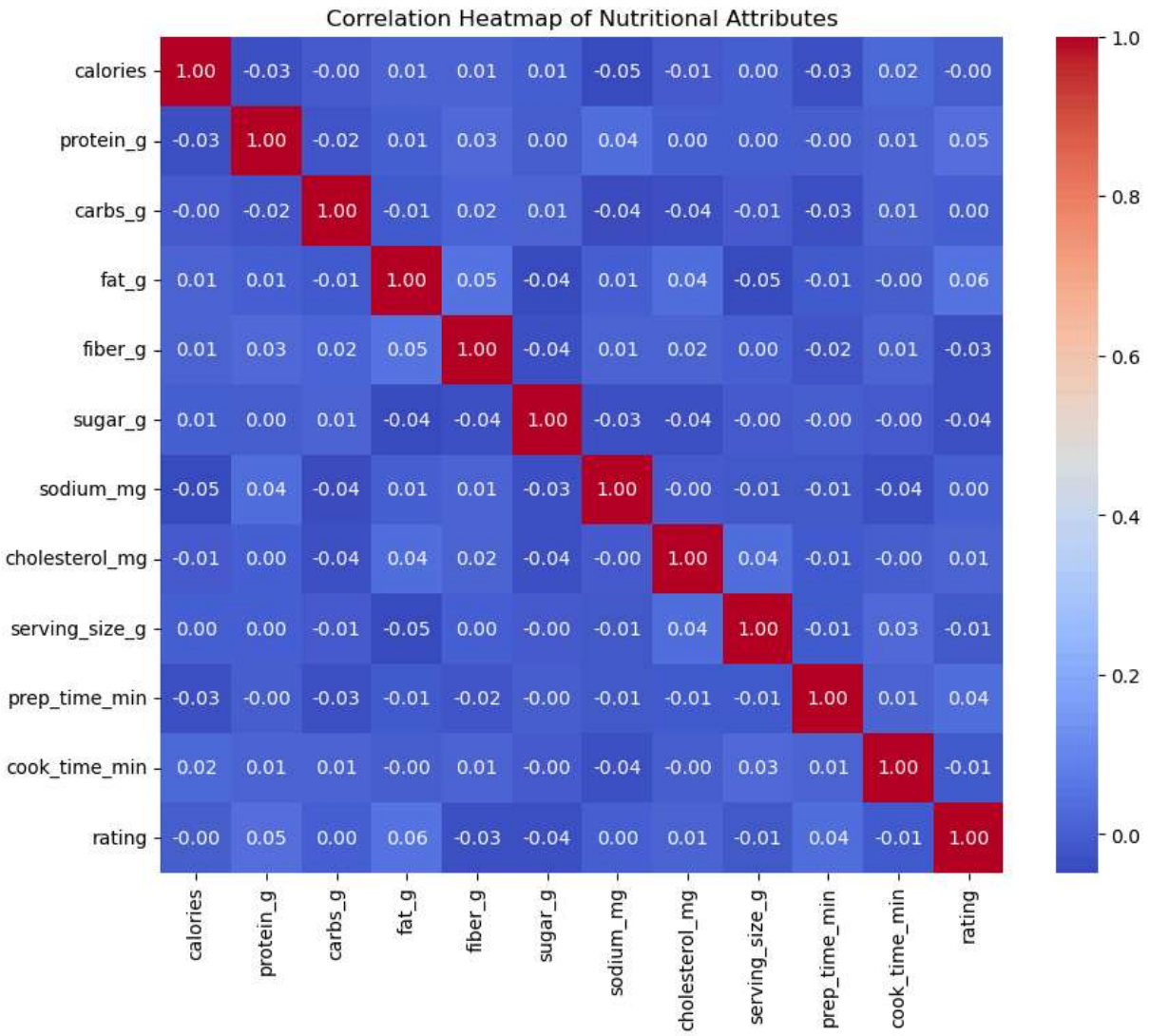
In [29]:
```
#------Cholesterol Distribution------
plt.figure(figsize=(8,5))
sns.histplot(df['cholesterol_mg'], bins=30, kde=True, color='brown')
plt.title("Cholesterol Distribution")
plt.xlabel("Cholesterol (mg)")
plt.ylabel("Count")
plt.show()
```



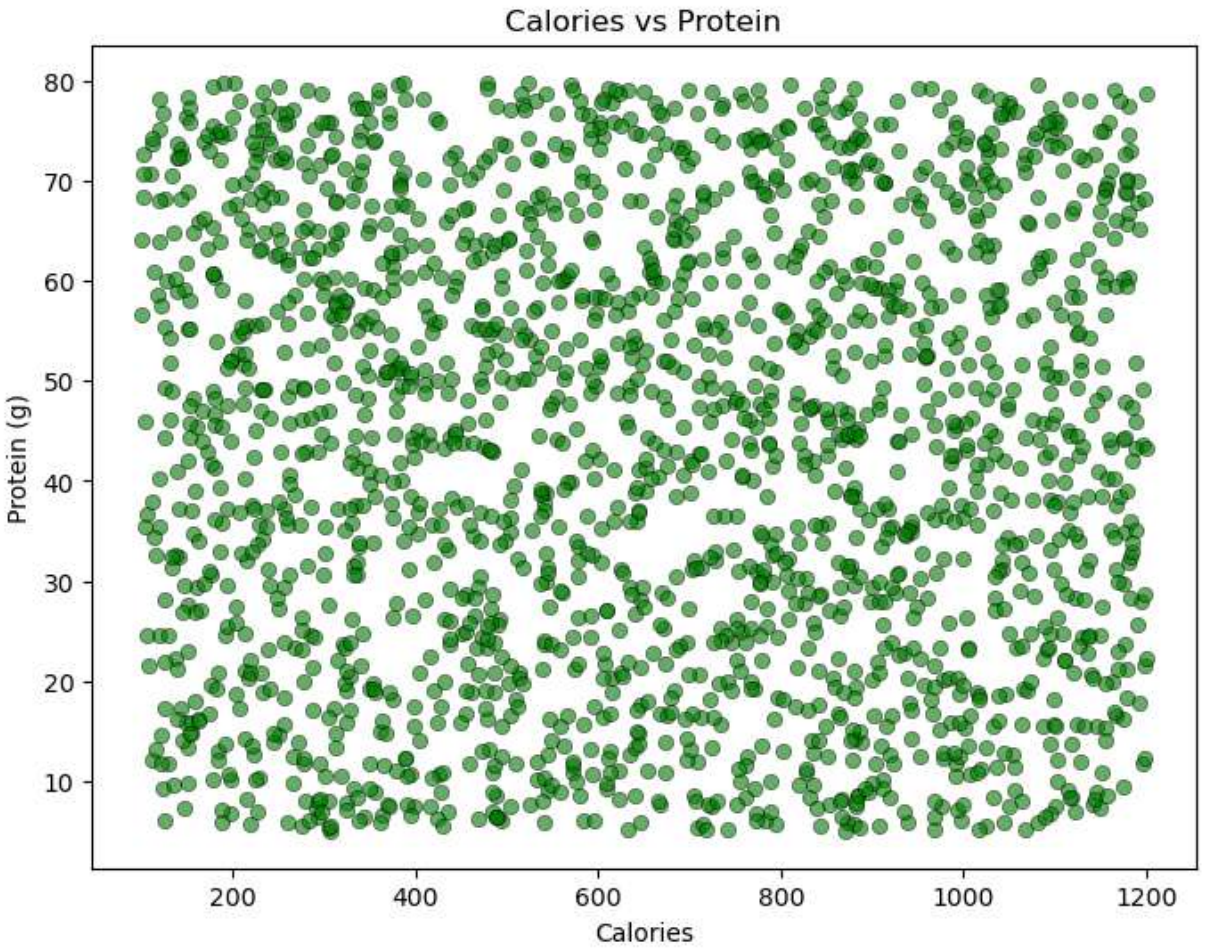
In [31]:
```
#------Serving Size Distribution------
plt.figure(figsize=(8,5))
sns.histplot(df['serving_size_g'], bins=30, kde=True, color='cyan')
plt.title("Serving Size Distribution")
plt.xlabel("Serving Size (g)")
plt.ylabel("Count")
plt.show()
```



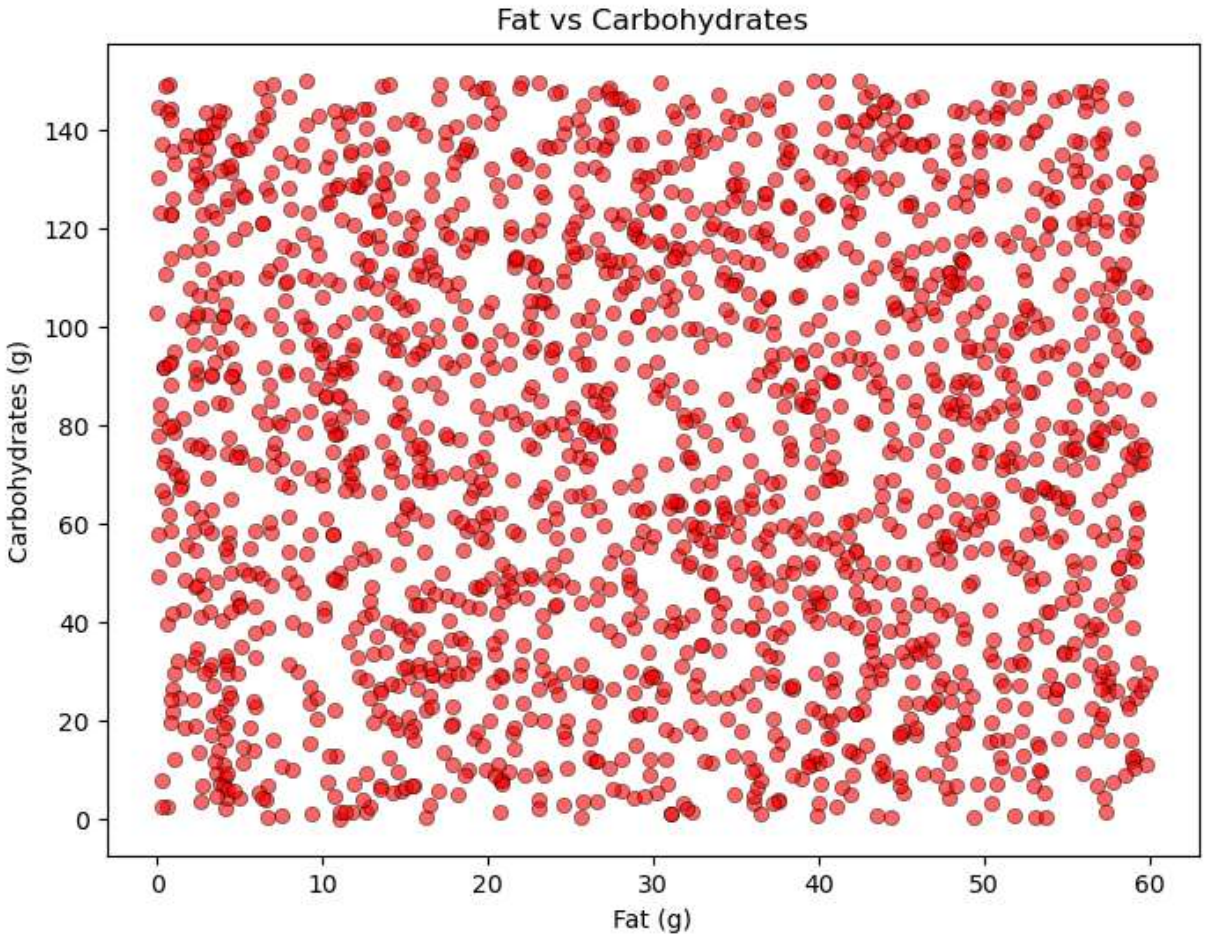
In [19]:
```
#----correlation HeatMap----
plt.figure(figsize=(10,8))
corr = df[numeric_cols].corr()
sns.heatmap(corr, annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation Heatmap of Nutritional Attributes")
plt.show()
```



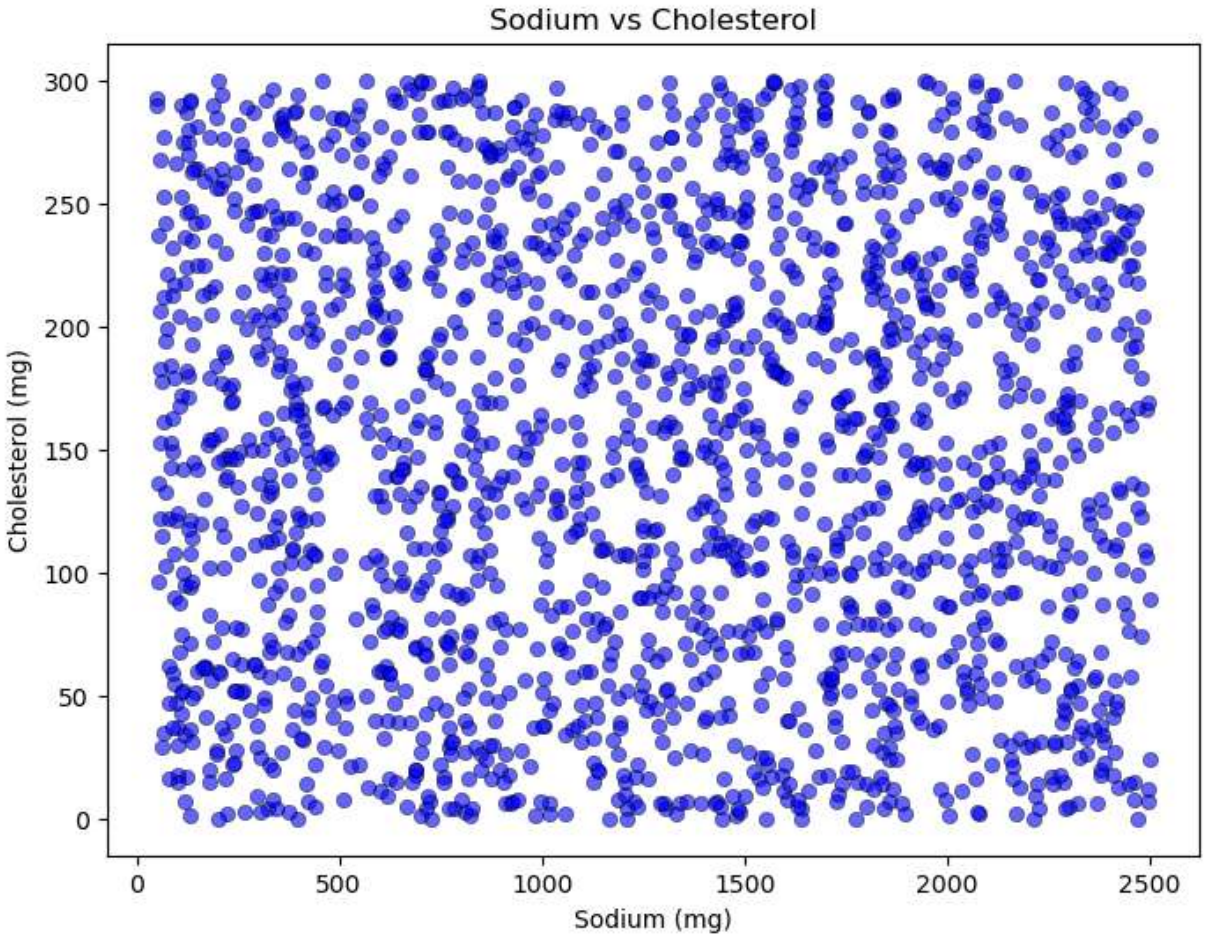
In [33]:
```
#----Calories vs Protein-------
plt.figure(figsize=(8,6))
sns.scatterplot(x='calories', y='protein_g', data=df, color='green', alpha=0.6, edg
plt.title("Calories vs Protein")
plt.xlabel("Calories")
plt.ylabel("Protein (g)")
plt.show()
```



In [35]:
```
#-----Fat vs Carbohydrates (Scatter Plot)------
plt.figure(figsize=(8,6))
sns.scatterplot(x='fat_g', y='carbs_g', data=df, color='red', alpha=0.6, edgecolor=
plt.title("Fat vs Carbohydrates")
plt.xlabel("Fat (g)")
plt.ylabel("Carbohydrates (g)")
plt.show()
```

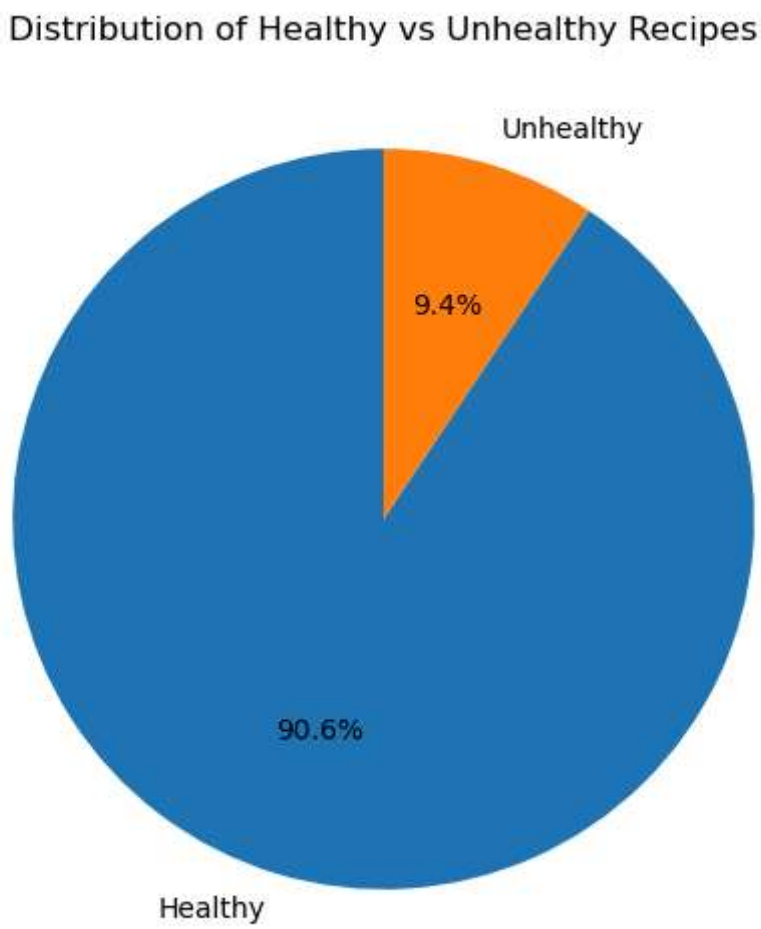


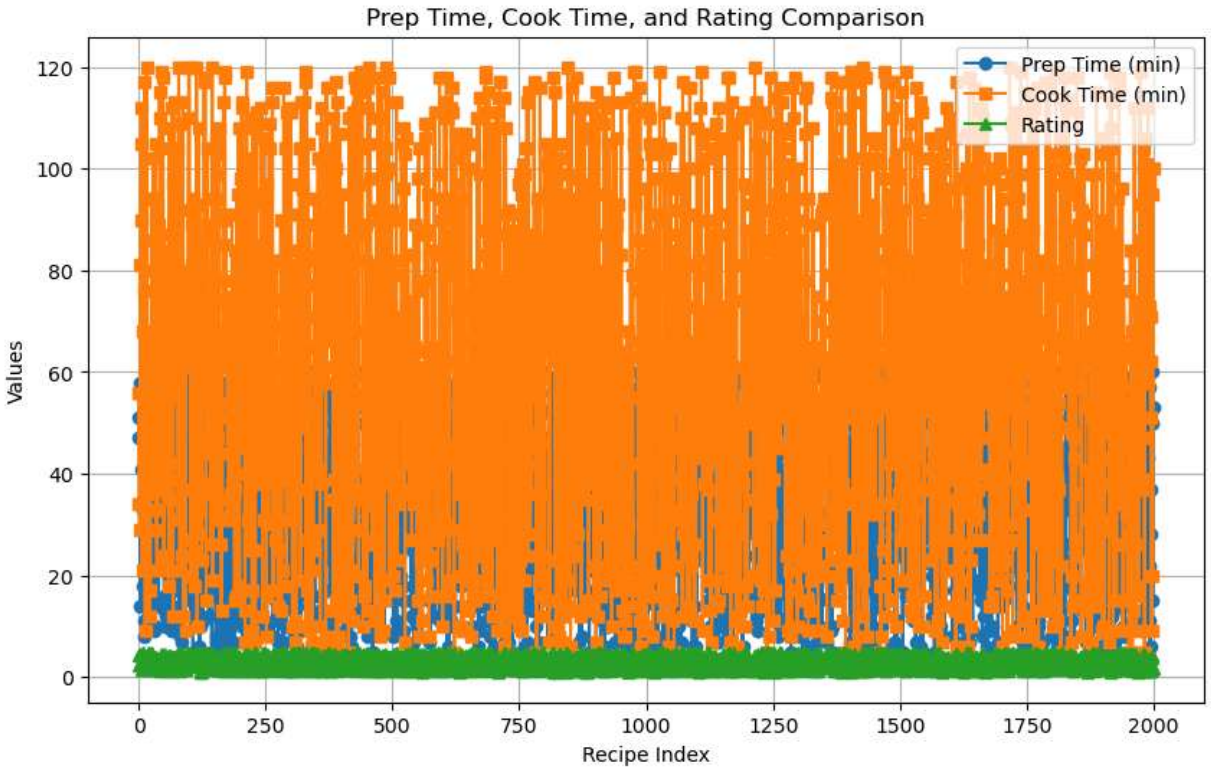
In [37]:
```
#-----Sodium vs Cholesterol (Scatter Plot)------
plt.figure(figsize=(8,6))
sns.scatterplot(x='sodium_mg', y='cholesterol_mg', data=df, color='blue', alpha=0.6
plt.title("Sodium vs Cholesterol")
plt.xlabel("Sodium (mg)")
plt.ylabel("Cholesterol (mg)")
plt.show()
```



In [39]:
```python
import pandas as pd
import matplotlib.pyplot as plt

# Load dataset
df = pd.read_csv("healthy_eating_dataset.csv")

# --- Pie Chart: Healthy vs Unhealthy ---
healthy_counts = df['is_healthy'].value_counts()
labels = ['Healthy', 'Unhealthy']

plt.figure(figsize=(6,6))
plt.pie(healthy_counts, labels=labels, autopct='%1.1f%%', startangle=90)
plt.title("Distribution of Healthy vs Unhealthy Recipes")
plt.show()

# --- Line Graph: Prep time, Cook time, and Rating ---
plt.figure(figsize=(10,6))
plt.plot(df.index, df['prep_time_min'], label='Prep Time (min)', marker='o')
plt.plot(df.index, df['cook_time_min'], label='Cook Time (min)', marker='s')
plt.plot(df.index, df['rating'], label='Rating', marker='^')

plt.title("Prep Time, Cook Time, and Rating Comparison")
plt.xlabel("Recipe Index")
plt.ylabel("Values")
plt.legend()
```

```
plt.grid(True)
plt.show()
```





In [41]:
```
import pandas as pd
df = pd.read_csv("healthy_eating_dataset.csv")
avg_prep = df['prep_time_min'].mean()
avg_cook = df['cook_time_min'].mean()

print(f"Average Preparation Time: {avg_prep:.2f} minutes")
print(f"Average Cooking Time: {avg_cook:.2f} minutes")
```

```
Average Preparation Time: 33.35 minutes
Average Cooking Time: 61.51 minutes
```

In [43]:
```
import pandas as pd
df = pd.read_csv("healthy_eating_dataset.csv")
highest = df.loc[df['rating'].idxmax()]
print("Recipe with Highest Rating:")
print(highest)
```

```
Recipe with Highest Rating:
meal_id                                          10
meal_name                                 Ask Pasta
cuisine                                     Mexican
meal_type                                    Dinner
diet_type                                  Balanced
calories                                        671
protein_g                                      48.4
carbs_g                                        76.4
fat_g                                          38.2
fiber_g                                        14.7
sugar_g                                        29.4
sodium_mg                                      1110
cholesterol_mg                                  214
serving_size_g                                  117
cooking_method                              Roasted
prep_time_min                                    35
cook_time_min                                    68
rating                                          5.0
is_healthy                                        0
image_url         https://example.com/images/meal_10.jpg
Name: 9, dtype: object
```

In [45]:
```
import pandas as pd
df = pd.read_csv("healthy_eating_dataset.csv")
count_healthy = df['is_healthy'].value_counts()
print("Count of Healthy vs Unhealthy Recipes:")
print(count_healthy)
```

```
Count of Healthy vs Unhealthy Recipes:
is_healthy
0    1813
1     187
Name: count, dtype: int64
```

In [47]:
```
import pandas as pd
df = pd.read_csv("healthy_eating_dataset.csv")
df['total_time'] = df['prep_time_min'] + df['cook_time_min']
max_time_row = df.loc[df['total_time'].idxmax()]
```

```
print("Recipe that takes the longest time:")
print(max_time_row)
```

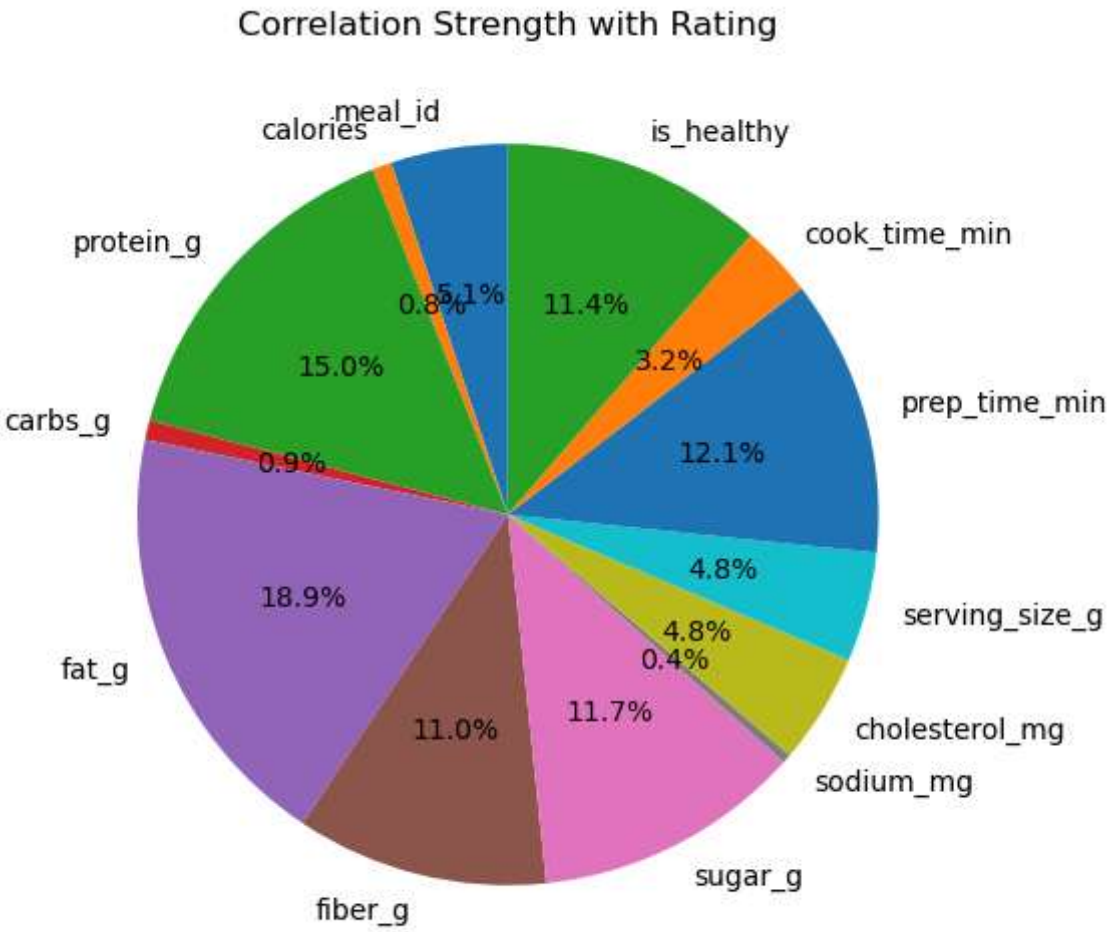
```
Recipe that takes the longest time:
meal_id                                                1469
meal_name                                       Address Rice
cuisine                                              Italian
meal_type                                          Breakfast
diet_type                                           Balanced
calories                                                469
protein_g                                              55.2
carbs_g                                                60.4
fat_g                                                  18.4
fiber_g                                                 2.3
sugar_g                                                 8.1
sodium_mg                                               342
cholesterol_mg                                           92
serving_size_g                                          268
cooking_method                                          Raw
prep_time_min                                            60
cook_time_min                                           119
rating                                                  4.0
is_healthy                                                1
image_url         https://example.com/images/meal_1469.jpg
total_time                                              179
Name: 1468, dtype: object
```

In [49]:
```
import pandas as pd
df = pd.read_csv("healthy_eating_dataset.csv")
avg_rating_by_health = df.groupby('is_healthy')['rating'].mean()
print("Average Rating by Healthiness:")
print(avg_rating_by_health)
```

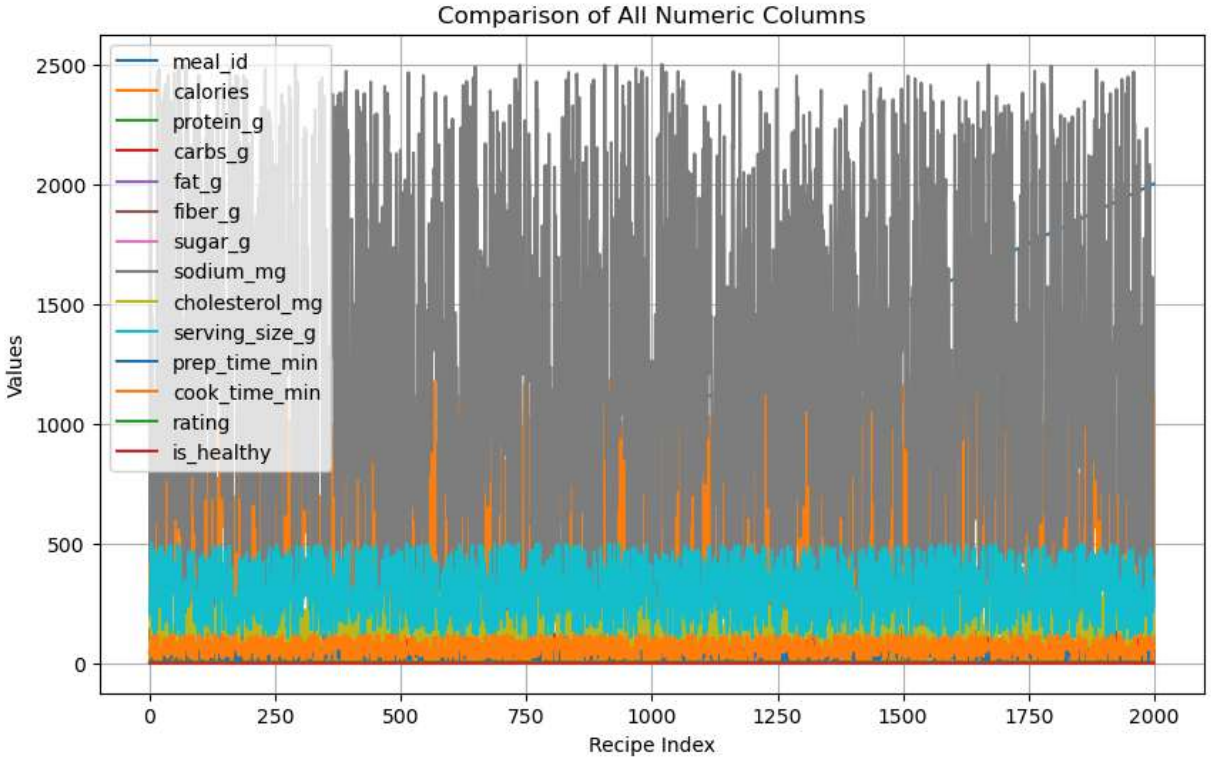
```
Average Rating by Healthiness:
is_healthy
0    2.997408
1    2.856684
Name: rating, dtype: float64
```

In [55]:
```
import pandas as pd
import matplotlib.pyplot as plt

# Load the dataset
df = pd.read_csv("healthy_eating_dataset.csv")

# Select only numeric columns
numeric_df = df.select_dtypes(include=['number'])

# --- Compute correlation matrix ---
corr = numeric_df.corr()

# --- PIE CHART: Correlation of each column with 'rating' ---
if 'rating' in numeric_df.columns:
    rating_corr = corr['rating'].drop('rating')  # exclude self-correlation
    plt.figure(figsize=(6,6))
    plt.pie(
        rating_corr.abs(),
```

```
        labels=rating_corr.index,
        autopct='%1.1f%%',
        startangle=90
    )
    plt.title("Correlation Strength with Rating")
    plt.show()
else:
    print("Column 'rating' not found for correlation pie chart.")
```



In [57]:
```
# --- LINE GRAPH: Compare numeric columns ---
plt.figure(figsize=(10,6))
for col in numeric_df.columns:
    plt.plot(df.index, numeric_df[col], label=col)

plt.title("Comparison of All Numeric Columns")
plt.xlabel("Recipe Index")
plt.ylabel("Values")
plt.legend()
plt.grid(True)
plt.show()
```



In [63]:
```
import pandas as pd
df = pd.read_csv("healthy_eating_dataset.csv")
# Group by healthiness
avg_times = df.groupby('is_healthy')[['prep_time_min', 'cook_time_min']].mean()

print("Average Preparation and Cooking Time by Healthiness:")
for health, row in avg_times.iterrows():
    status = "Healthy" if health == 1 else "Unhealthy"
    print(f"{status} Recipes → Prep: {row['prep_time_min']:.1f} min | Cook: {row['c
```

```
Average Preparation and Cooking Time by Healthiness:
Unhealthy Recipes → Prep: 33.3 min | Cook: 61.3 min
Healthy Recipes → Prep: 34.0 min | Cook: 63.1 min
```

In [ ]: