```
import pandas as pd

# For Load the dataset
file_path ='/epi_r.csv'
data_set = pd.read_csv(file_path)
print(data_set)
```

```
                                        title  rating  calories  protein  \
0                 Lentil, Apple, and Turkey Wrap    2.500     426.0     30.0
1      Boudin Blanc Terrine with Red Onion Confit    4.375     403.0     18.0
2                   Potato and Fennel Soup Hodge    3.750     165.0      6.0
3                Mahi-Mahi in Tomato Olive Sauce    5.000       NaN      NaN
4                      Spinach Noodle Casserole    3.125     547.0     20.0
...                                         ...      ...       ...      ...
20047                          Parmesan Puffs    3.125      28.0      2.0
20048             Artichoke and Parmesan Risotto    4.375     671.0     22.0
20049                     Turkey Cream Puff Pie    4.375     563.0     31.0
20050    Snapper on Angel Hair with Citrus Cream    4.375     631.0     45.0
20051  Baked Ham with Marmalade-Horseradish Glaze    4.375     560.0     73.0

        fat  sodium  #cakeweek  #wasteless  22-minute meals  \
0       7.0   559.0        0.0         0.0              0.0
1      23.0  1439.0        0.0         0.0              0.0
2       7.0   165.0        0.0         0.0              0.0
3       NaN     NaN        0.0         0.0              0.0
4      32.0   452.0        0.0         0.0              0.0
...     ...     ...        ...         ...              ...
20047   2.0    64.0        0.0         0.0              0.0
20048  28.0   583.0        0.0         0.0              0.0
20049  38.0   652.0        0.0         0.0              0.0
20050  24.0   517.0        0.0         0.0              0.0
20051  10.0  3698.0        0.0         0.0              0.0

       3-ingredient recipes  ...  yellow squash  yogurt  yonkers  yuca  \
0                       0.0  ...            0.0     0.0      0.0   0.0
1                       0.0  ...            0.0     0.0      0.0   0.0
2                       0.0  ...            0.0     0.0      0.0   0.0
3                       0.0  ...            0.0     0.0      0.0   0.0
4                       0.0  ...            0.0     0.0      0.0   0.0
...                     ...  ...            ...     ...      ...   ...
20047                   0.0  ...            0.0     0.0      0.0   0.0
20048                   0.0  ...            0.0     0.0      0.0   0.0
20049                   0.0  ...            0.0     0.0      0.0   0.0
20050                   0.0  ...            0.0     0.0      0.0   0.0
20051                   0.0  ...            0.0     0.0      0.0   0.0

       zucchini  cookbooks  leftovers  snack  snack week  turkey
0           0.0        0.0        0.0    0.0         0.0     1.0
1           0.0        0.0        0.0    0.0         0.0     0.0
2           0.0        0.0        0.0    0.0         0.0     0.0
3           0.0        0.0        0.0    0.0         0.0     0.0
4           0.0        0.0        0.0    0.0         0.0     0.0
...         ...        ...        ...    ...         ...     ...
20047       0.0        0.0        0.0    0.0         0.0     0.0
20048       0.0        0.0        0.0    0.0         0.0     0.0
20049       0.0        0.0        0.0    0.0         0.0     1.0
20050       0.0        0.0        0.0    0.0         0.0     0.0
20051       0.0        0.0        0.0    0.0         0.0     0.0

[20052 rows x 680 columns]
```

```
# Check missing values in the Dataset
missing_data = data_set.isnull().sum()

# Check duplicate rows in the Dataset
duplicate_rows = data_set.duplicated().sum()

# Summarize the data (missing and Duplicate)
missing_summary = missing_data[missing_data > 0].sort_values(ascending=False)
duplicate_rows, missing_summary
```

```
(1801,
 fat         4183
 protein     4162
 sodium      4119
 calories    4117
 dtype: int64)
```

```
clean_data = data_set.drop_duplicates()

# Number of rows after removing duplicates
rows_after_deduplication = clean_data.shape[0]
rows_after_deduplication
```

```
18251
```

```
# Fill missing values in columns with the help of median
fill_columns = ['calories', 'fat', 'protein', 'sodium']

for column in fill_columns:
    clean_data[column] = clean_data[column].fillna(clean_data[column].median())

# To show the Clean Data_Set
clean_data.describe()
```

```
<ipython-input-10-9a6af931db45>:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  clean_data[column] = clean_data[column].fillna(clean_data[column].median())
```

	rating	calories	protein	fat	sodium	#cakeweek	#wasteless	22-minute meals	3-ingredient recipes	30 d gro
count	18251.000000	1.825100e+04	18251.000000	1.825100e+04	1.825100e+04	18251.000000	18251.000000	18251.000000	18251.000000	18251.
mean	3.714043	5.332697e+03	74.985864	2.914618e+02	5.285578e+03	0.000329	0.000055	0.000931	0.001479	0.
std	1.333942	3.340863e+05	3127.068944	1.900439e+04	3.105751e+05	0.018129	0.007402	0.030506	0.038435	0.
min	0.000000	0.000000e+00	0.000000	0.000000e+00	0.000000e+00	0.000000	0.000000	0.000000	0.000000	0.
25%	3.750000	2.380000e+02	4.000000	1.100000e+01	1.320000e+02	0.000000	0.000000	0.000000	0.000000	0.
50%	4.375000	3.450000e+02	9.000000	1.800000e+01	3.040000e+02	0.000000	0.000000	0.000000	0.000000	0.
75%	4.375000	5.165000e+02	21.000000	2.800000e+01	5.890000e+02	0.000000	0.000000	0.000000	0.000000	0.
max	5.000000	3.011122e+07	236489.000000	1.722763e+06	2.767511e+07	1.000000	1.000000	1.000000	1.000000	1.

8 rows × 679 columns

```
# For Download the Clean Data file
from google.colab import files
files.download('/epi_r.csv')
```

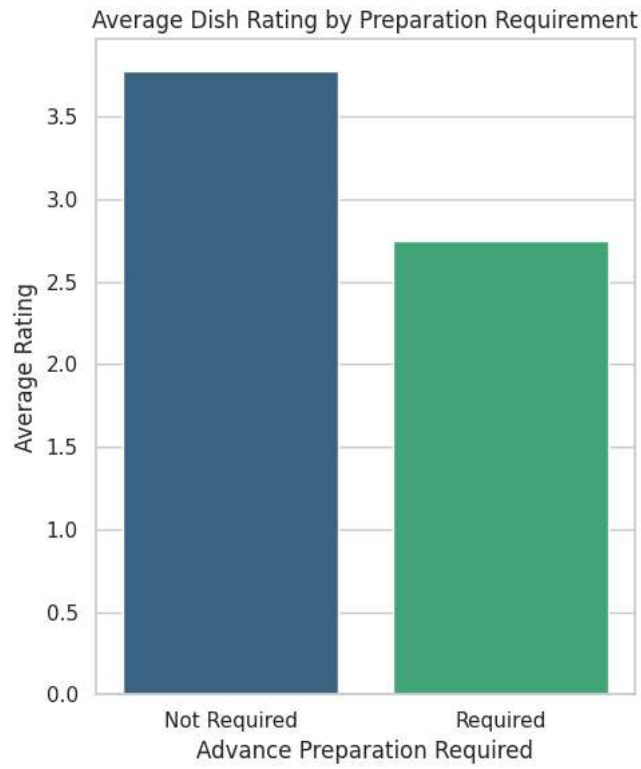
```
import pandas as pd

# Load the data from the uploaded files
ingredient_data_path = '/Ingrident_data.xlsx'
dish_data_path = '/Dish_data.xlsx'

# Reading the data from the Excel files
ingredient_df = pd.read_excel(ingredient_data_path)
dish_df = pd.read_excel(dish_data_path)
ingredient_df.head(), dish_df.head()
```

```
(  Ingredient  Dish_Id Ingredient_Use
 0    alabama        1       Not Used
 1     alaska        1       Not Used
 2  alcoholic        1       Not Used
 3     almond        1       Not Used
 4   amaretto        1       Not Used,
    Dish_Id                                 Dish_Name  Rating Rating Status  \
 0        1             Lentil, Apple, and Turkey Wrap    2.500          Fair
 1        2  Boudin Blanc Terrine with Red Onion Confit    4.375     Very Good
 2        3               Potato and Fennel Soup Hodge    3.750          Good
 3        4            Mahi-Mahi in Tomato Olive Sauce    5.000     Excellent
 4        5                    Spinach Noodle Casserole    3.125          Good

    calories  protein  fat  sodium 22 Minute Meals 3 Ingredient Recipes  \
 0       426       30    7     559              No                   No
 1       403       18   23    1439              No                   No
```

```
2       165        6    7     165             No                  No
3       345        9   18     304             No                  No
4       547       20   32     452             No                  No

  30 days of groceries Advance Preparation Required
0                   No                 Not Required
1                   No                 Not Required
2                   No                 Not Required
3                   No                 Not Required
4                   No                 Not Required  )
```

```
import matplotlib.pyplot as plt
import seaborn as sns

# For plotting the Environment
sns.set(style="whitegrid")
fig, axes = plt.subplots(1, 1, figsize=(5, 6))

#Bar Chart:
# Average Dish Rating by Preparation Requirement
avg_rating_prep = dish_df.groupby('Advance Preparation Required')['Rating'].mean().reset_index()
sns.barplot(x='Advance Preparation Required', y='Rating', data=avg_rating_prep, ax=axes, palette='viridis')
axes.set_title('Average Dish Rating by Preparation Requirement')
axes.set_xlabel('Advance Preparation Required')
axes.set_ylabel('Average Rating')

#To show Data Perfectly
plt.tight_layout()
plt.show()
```

```
<ipython-input-16-7170a2771e02>:11: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legenc

  sns.barplot(x='Advance Preparation Required', y='Rating', data=avg_rating_prep, ax=axes, palette='viridis')
/usr/local/lib/python3.10/dist-packages/seaborn/_base.py:949: FutureWarning: When grouping with a length-1 list-like, you will need to p
  data_subset = grouped_data.get_group(pd_key)
/usr/local/lib/python3.10/dist-packages/seaborn/_base.py:949: FutureWarning: When grouping with a length-1 list-like, you will need to p
  data_subset = grouped_data.get_group(pd_key)
```




```
import matplotlib.pyplot as plt
import seaborn as sns

# For plotting the Environment
sns.set(style="whitegrid")
fig, axes = plt.subplots(1, 1, figsize=(10, 5))
```

```
#Scatter Plot:-
#Caloric Content vs. Protein Content categorized by Rating
sns.scatterplot(
    x='calories', y='protein', hue='Rating Status', data=dish_df, palette='coolwarm', s=50, ax=axes
)
axes.set_title('Caloric Content vs. Protein Content by Rating')
axes.set_xlabel('Calories')
axes.set_ylabel('Protein')

#To show Data Perfectly
plt.tight_layout()
plt.show()
```