Importing necessary libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Reading CSV files

```
user_details = pd.read_csv('UserDetails.csv')
cooking_sessions = pd.read_csv('CookingSessions.csv')
order_details = pd.read_csv('OrderDetails.csv')

user_details.head()
```


```
{"summary":"{\n  \"name\": \"user_details\",\n  \"rows\": 10,\n
\"fields\": [\n    {\n      \"column\": \"User ID\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 10,\n        \"samples\": [\n
\"U009\",\n          \"U002\",\n          \"U006\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"User Name\",\n
\"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 10,\n        \"samples\": [\n          \"Irene
Moore\",\n          \"Bob Smith\",\n          \"Frank Green\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Age\",\n      \"properties\": {\
n        \"dtype\": \"number\",\n        \"std\": 5,\n        \"min\":
25,\n        \"max\": 42,\n        \"num_unique_values\": 10,\n
\"samples\": [\n          33,\n          35,\n          25\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Location\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 10,\n        \"samples\":
[\n          \"Dallas\",\n          \"Los Angeles\",\n
\"Austin\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Registration Date\",\n      \"properties\": {\n        \"dtype\":
\"object\",\n        \"num_unique_values\": 10,\n        \"samples\":
[\n          \"2023-09-01\",\n          \"2023-02-20\",\n
\"2023-06-15\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Phone\",\n      \"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 10,\n        \"samples\": [\n          \"202-
303-4040\",\n          \"987-654-3210\",\n          \"888-777-6666\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Email\",\n      \"properties\":
{\n        \"dtype\": \"string\",\n        \"num_unique_values\": 10,\
n        \"samples\": [\n          \"irene@email.com\",\n
```

\"bob@email.com\",\n   \"frank@email.com\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"Favorite Meal\",\n
\"properties\": {\n   \"dtype\": \"category\",\n
\"num_unique_values\": 3,\n   \"samples\": [\n
\"Dinner\",\n   \"Lunch\",\n   \"Breakfast\"\
n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Total Orders\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 3,\n   \"min\": 5,\n
\"max\": 15,\n   \"num_unique_values\": 9,\n   \"samples\":
[\n   5,\n   8,\n   7\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   }\n]\n}","type":"dataframe","variable_name":"user_details"}

```
cooking_sessions.head()
```

{"summary":"{\n \"name\": \"cooking_sessions\",\n \"rows\": 16,\n
\"fields\": [\n   {\n   \"column\": \"Session ID\",\n
\"properties\": {\n   \"dtype\": \"string\",\n
\"num_unique_values\": 16,\n   \"samples\": [\n
\"S001\",\n   \"S002\",\n   \"S006\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"User ID\",\n   \"properties\":
{\n   \"dtype\": \"string\",\n   \"num_unique_values\": 8,\n
\"samples\": [\n   \"U002\",\n   \"U006\",\n
\"U001\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Dish Name\",\n   \"properties\": {\n   \"dtype\":
\"category\",\n   \"num_unique_values\": 6,\n   \"samples\":
[\n   \"Spaghetti\",\n   \"Caesar Salad\",\n
\"Oatmeal\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Meal Type\",\n   \"properties\": {\n   \"dtype\":
\"category\",\n   \"num_unique_values\": 3,\n   \"samples\":
[\n   \"Dinner\",\n   \"Lunch\",\n
\"Breakfast\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Session Start\",\n   \"properties\": {\n   \"dtype\":
\"object\",\n   \"num_unique_values\": 16,\n   \"samples\":
[\n   \"2024-12-01 19:00\",\n   \"2024-12-01 12:00\",\n
\"2024-12-03 18:30\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Session End\",\n   \"properties\": {\n   \"dtype\":
\"object\",\n   \"num_unique_values\": 16,\n   \"samples\":
[\n   \"2024-12-01 19:30\",\n   \"2024-12-01 12:20\",\n
\"2024-12-03 19:00\"\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"Duration (mins)\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 10,\n   \"min\": 10,\n

\"max\": 45,\n   \"num_unique_values\": 6,\n   \"samples\": [\n   30,\n   20,\n   10\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Session Rating\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0.2926174977679906,\n   \"min\": 4.0,\n   \"max\": 5.0,\n   \"num_unique_values\": 11,\n   \"samples\": [\n   4.3,\n   4.5,\n   4.1\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe","variable_name":"cooking_sessions"}

```
order_details.head()
```

{"summary":"{\n \"name\": \"order_details\",\n \"rows\": 16,\n \"fields\": [\n   {\n   \"column\": \"Order ID\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 4,\n   \"min\": 1001,\n   \"max\": 1016,\n   \"num_unique_values\": 16,\n   \"samples\": [\n   1001,\n   1002,\n   1006\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"User ID\",\n   \"properties\": {\n   \"dtype\": \"string\",\n   \"num_unique_values\": 8,\n   \"samples\": [\n   \"U002\",\n   \"U006\",\n   \"U001\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Order Date\",\n   \"properties\": {\n   \"dtype\": \"object\",\n   \"num_unique_values\": 8,\n   \"samples\": [\n   \"2024-12-02\",\n   \"2024-12-06\",\n   \"2024-12-01\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Meal Type\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 3,\n   \"samples\": [\n   \"Dinner\",\n   \"Lunch\",\n   \"Breakfast\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Dish Name\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 6,\n   \"samples\": [\n   \"Spaghetti\",\n   \"Caesar Salad\",\n   \"Oatmeal\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Order Status\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 2,\n   \"samples\": [\n   \"Canceled\",\n   \"Completed\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Amount (USD)\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 2.4358434541926814,\n   \"min\": 7.0,\n   \"max\": 15.0,\n   \"num_unique_values\": 12,\n   \"samples\": [\n   8.5,\n   7.0\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\":

```
\"Time of Day\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 3,\n        \"samples\":
[\n          \"Night\",\n          \"Day\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Rating\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
0.4688072309384954,\n        \"min\": 4.0,\n        \"max\": 5.0,\n
\"num_unique_values\": 2,\n        \"samples\": [\n          4.0,\n
5.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Session ID\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 16,\n        \"samples\":
[\n          \"S001\",\n          \"S002\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe","variable_name":"order_details"}
```


- "OrderDetails" have null values in "Rating" column
- Filling Null values in "Rating" column with average of Rating

```
order_details.fillna(order_details['Rating'].mean(), inplace=True)
order_details.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16 entries, 0 to 15
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Order ID      16 non-null     int64
 1   User ID       16 non-null     object
 2   Order Date    16 non-null     object
 3   Meal Type     16 non-null     object
 4   Dish Name     16 non-null     object
 5   Order Status  16 non-null     object
 6   Amount (USD)  16 non-null     float64
 7   Time of Day   16 non-null     object
 8   Rating        16 non-null     float64
 9   Session ID    16 non-null     object
dtypes: float64(2), int64(1), object(7)
memory usage: 1.4+ KB
```

- Removing Duplicates

```
user_details.drop_duplicates(inplace=True)
cooking_sessions.drop_duplicates(inplace=True)
order_details.drop_duplicates(inplace=True)
```

- Converting all date columns in datetime format

```
user_details['Registration Date'] =
pd.to_datetime(user_details['Registration Date'])
cooking_sessions['Session Start'] =
pd.to_datetime(cooking_sessions['Session Start'])
cooking_sessions['Session End'] =
pd.to_datetime(cooking_sessions['Session End'])
order_details['Order Date'] = pd.to_datetime(order_details['Order
Date'])

user_details.info()
cooking_sessions.info()
order_details.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 9 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   User ID            10 non-null     object
 1   User Name          10 non-null     object
 2   Age                10 non-null     int64
 3   Location           10 non-null     object
 4   Registration Date  10 non-null     datetime64[ns]
 5   Phone              10 non-null     object
 6   Email              10 non-null     object
 7   Favorite Meal      10 non-null     object
 8   Total Orders       10 non-null     int64
dtypes: datetime64[ns](1), int64(2), object(6)
memory usage: 848.0+ bytes
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16 entries, 0 to 15
Data columns (total 8 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Session ID       16 non-null     object
 1   User ID          16 non-null     object
 2   Dish Name        16 non-null     object
 3   Meal Type        16 non-null     object
 4   Session Start    16 non-null     datetime64[ns]
 5   Session End      16 non-null     datetime64[ns]
 6   Duration (mins)  16 non-null     int64
 7   Session Rating   16 non-null     float64
dtypes: datetime64[ns](2), float64(1), int64(1), object(4)
memory usage: 1.1+ KB
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16 entries, 0 to 15
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Order ID      16 non-null     int64
```

```
 1   User ID         16 non-null     object
 2   Order Date      16 non-null     datetime64[ns]
 3   Meal Type       16 non-null     object
 4   Dish Name       16 non-null     object
 5   Order Status    16 non-null     object
 6   Amount (USD)    16 non-null     float64
 7   Time of Day     16 non-null     object
 8   Rating          16 non-null     float64
 9   Session ID      16 non-null     object
dtypes: datetime64[ns](1), float64(2), int64(1), object(6)
memory usage: 1.4+ KB
```

- Merging all the three datasets for better EDA

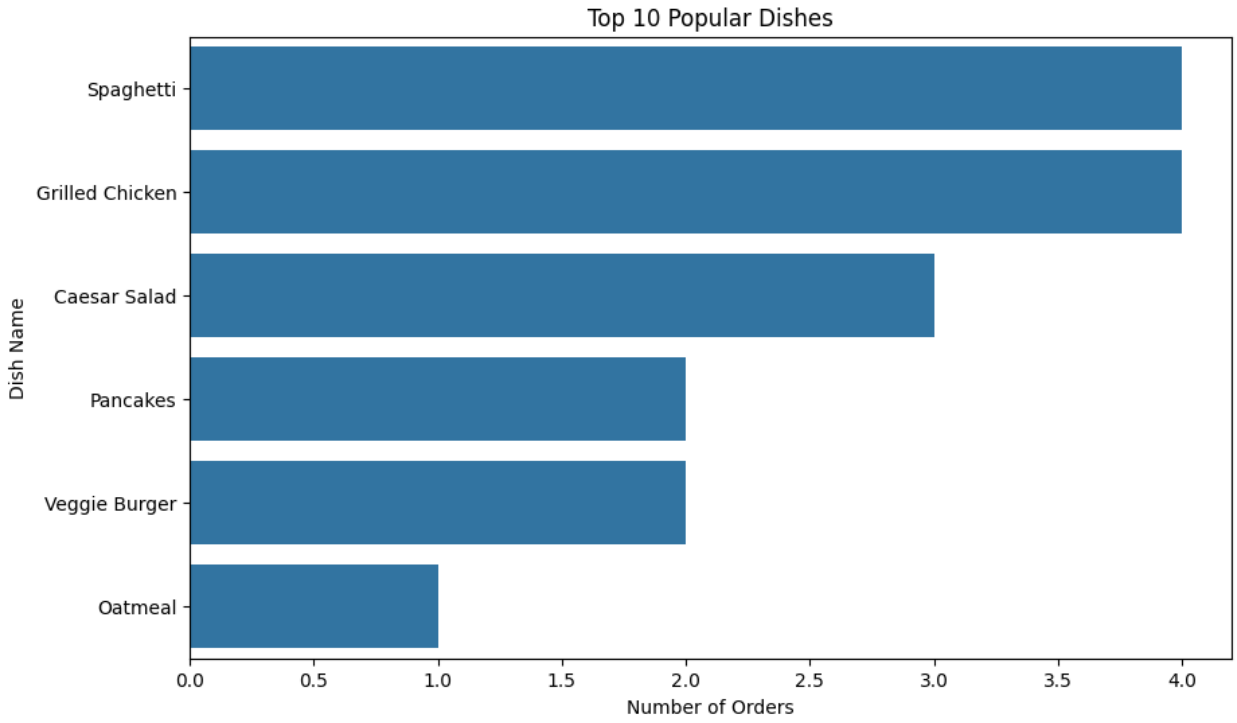
```
df = user_details.merge(cooking_sessions, on='User
ID').merge(order_details, on='Session ID')

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16 entries, 0 to 15
Data columns (total 25 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   User ID_x           16 non-null     object
 1   User Name           16 non-null     object
 2   Age                 16 non-null     int64
 3   Location            16 non-null     object
 4   Registration Date   16 non-null     datetime64[ns]
 5   Phone               16 non-null     object
 6   Email               16 non-null     object
 7   Favorite Meal       16 non-null     object
 8   Total Orders        16 non-null     int64
 9   Session ID          16 non-null     object
 10  Dish Name_x         16 non-null     object
 11  Meal Type_x         16 non-null     object
 12  Session Start       16 non-null     datetime64[ns]
 13  Session End         16 non-null     datetime64[ns]
 14  Duration (mins)     16 non-null     int64
 15  Session Rating      16 non-null     float64
 16  Order ID            16 non-null     int64
 17  User ID_y           16 non-null     object
 18  Order Date          16 non-null     datetime64[ns]
 19  Meal Type_y         16 non-null     object
 20  Dish Name_y         16 non-null     object
 21  Order Status        16 non-null     object
 22  Amount (USD)        16 non-null     float64
 23  Time of Day         16 non-null     object
 24  Rating              16 non-null     float64
```

```
dtypes: datetime64[ns](4), float64(3), int64(4), object(14)
memory usage: 3.2+ KB
```

Popular Dishes

```
popular_dishes = df['Dish Name_x'].value_counts()
plt.figure(figsize=(10, 6))
sns.barplot(x=popular_dishes.values, y=popular_dishes.index)
plt.title('Top 10 Popular Dishes')
plt.xlabel('Number of Orders')
plt.ylabel('Dish Name')
plt.show()
```



Insights:

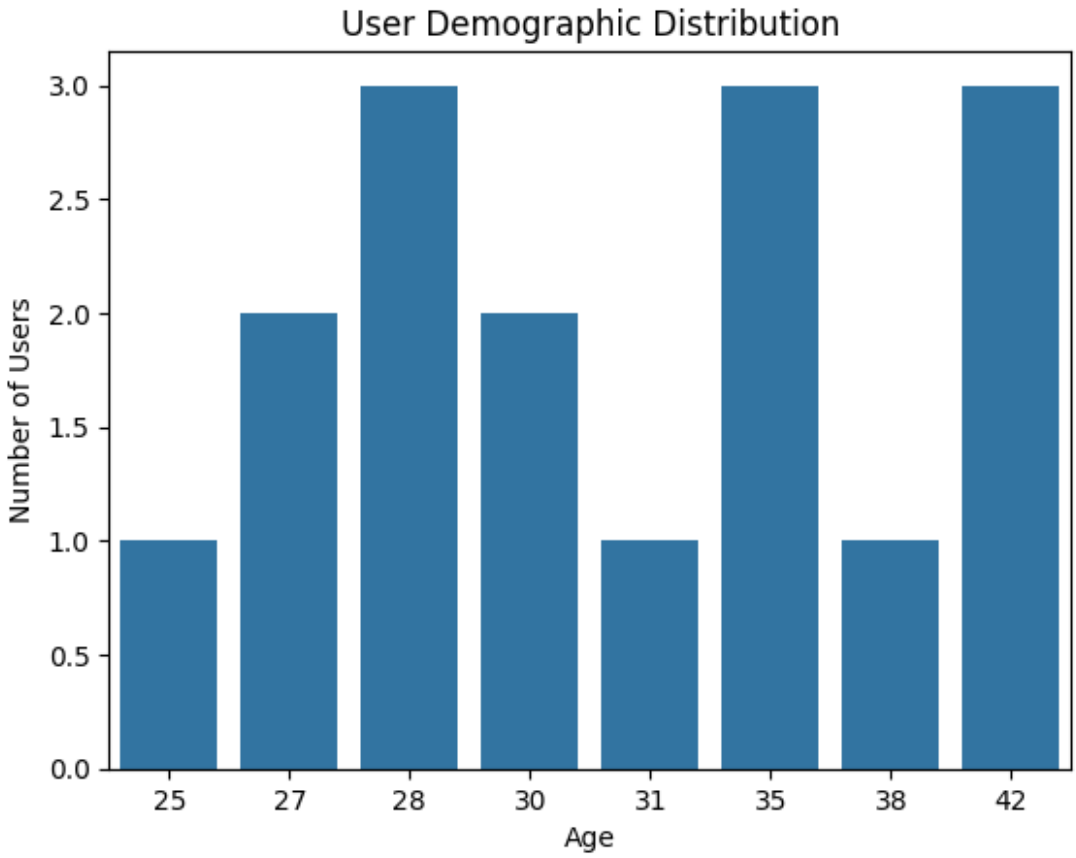
- Spaghetti and Grilled Chicken are the most popular.
- Oatmeal and Veggie Burger have the lowest order numbers.

Recommendations:

- Highlight Spaghetti and Grilled Chicken in marketing campaigns.
- Improve or consider removing Oatmeal and Veggie Burger from the menu.
- Create meal bundles that pair popular dishes with Oatmeal and Veggie Burger.

User Demographic

```
sns.countplot(x='Age', data=df)
plt.title('User Demographic Distribution')
plt.xlabel('Age')
plt.ylabel('Number of Users')
plt.show()
```



Insights:

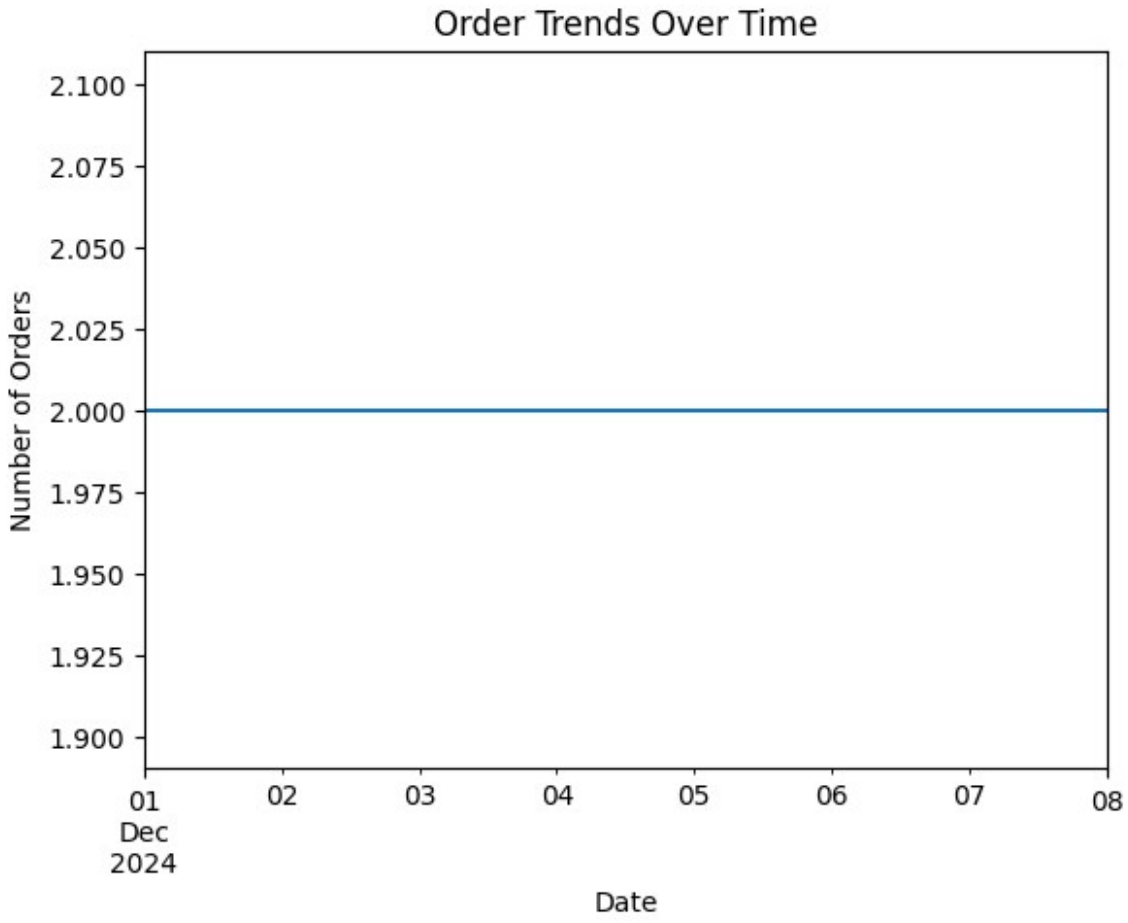
- The highest number of users are aged 28, 35, and 42.
- The lowest number of users are aged 25, 31, and 38.

Recommendations:

- Focus marketing on the 28, 35, and 42 age groups to maximize engagement.
- Develop campaigns to attract users aged 25, 31, and 38, such as special offers or discounts.
- Gather feedback from lower-engagement age groups to understand their preferences and improve offers.

```
order_trends = df.groupby('Order Date')['Order ID'].count()
order_trends.plot(kind='line')
plt.title('Order Trends Over Time')
plt.xlabel('Date')
```

```
plt.ylabel('Number of Orders')
plt.show()
```



Insights:

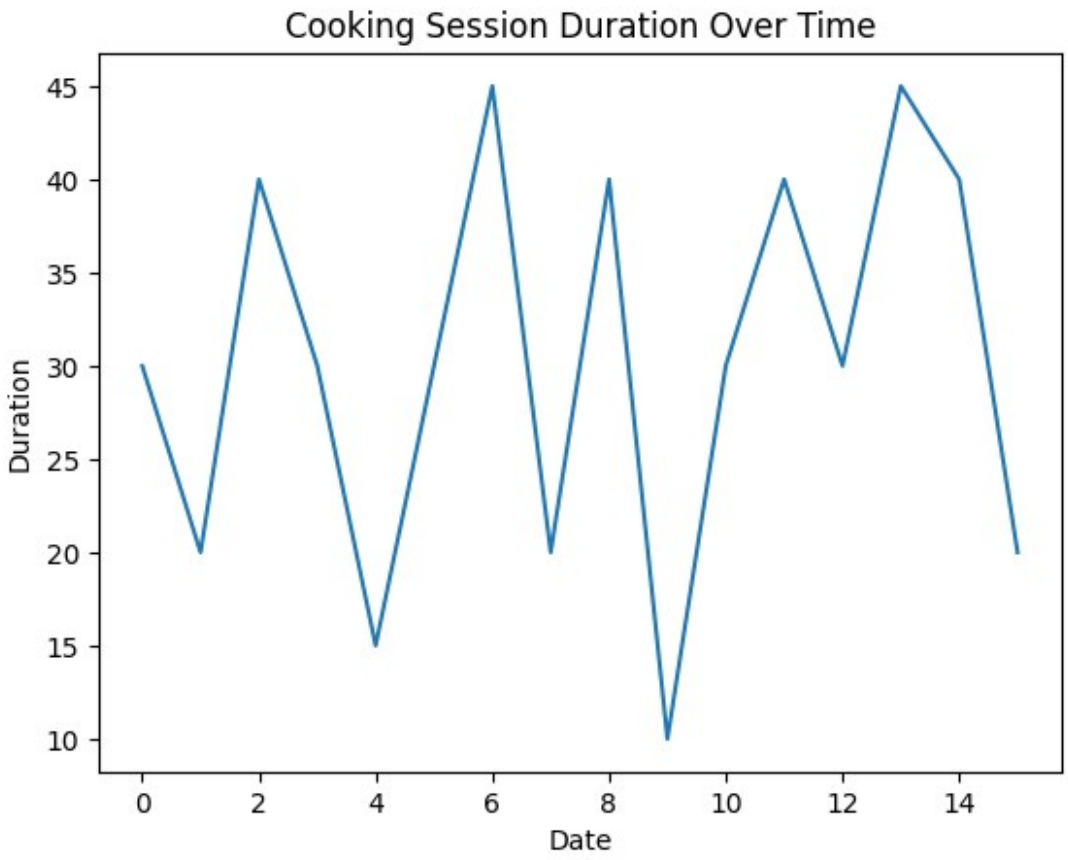
- The straight line indicates that we only have the data of 2 orders per day.

Recommandations:

- Given the limited data, it is challenging to derive meaningful recommendations at this time.

Cooking Sessions Duration Over Time

```
cooking_sessions_duration = (cooking_sessions['Session End'] -
cooking_sessions['Session Start']).dt.total_seconds() / 60
cooking_sessions_duration.plot(kind='line')
plt.title('Cooking Session Duration Over Time')
plt.xlabel('Date')
plt.ylabel('Duration')
plt.show()
```



Insights:

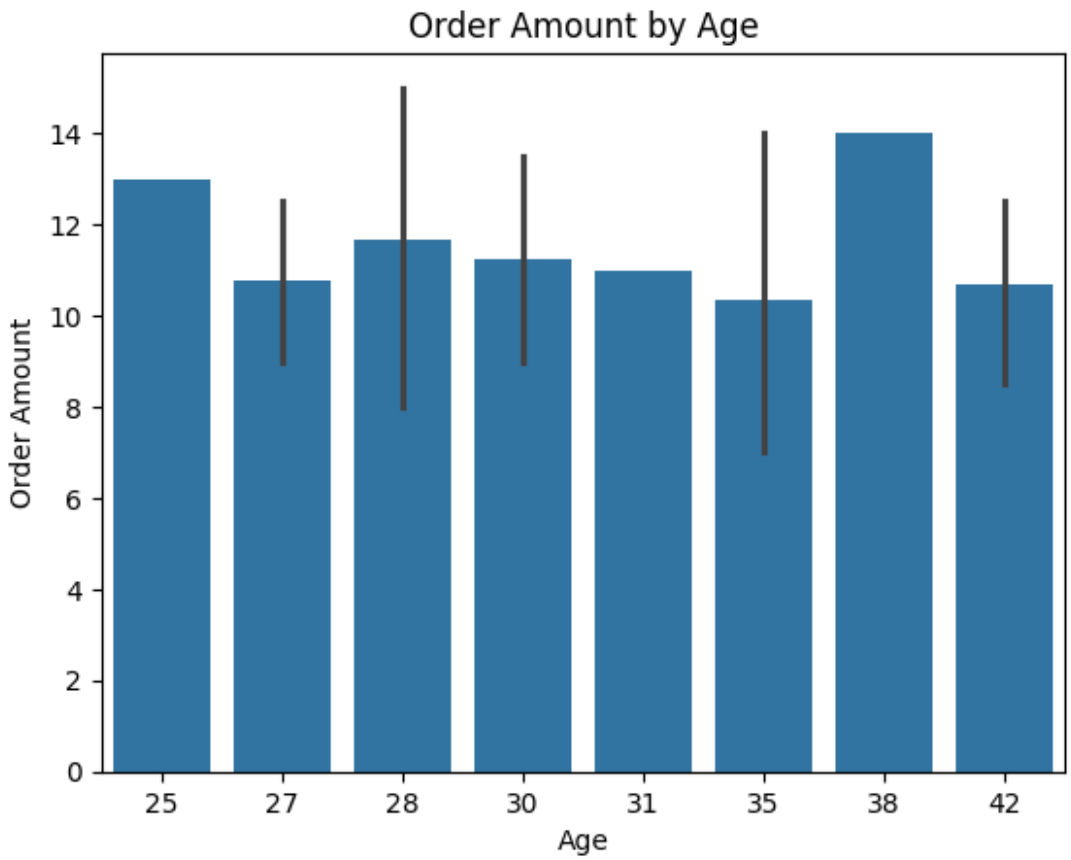
- The longest cooking sessions occurred on the 6th and 13th, each lasting 45 minutes, followed by the 8th and 11th with 40 minutes.
- The shortest cooking session was on the 9th at 10 minutes, followed by the 4th at 15 minutes.

Recommendations:

- Investigate what contributed to the longer cooking times on the 6th and 13th to replicate successful factors in future sessions.
- Implement strategies to encourage longer cooking sessions on days with historically low cooking times, such as offering incentives or special recipes.
- Enhance recipe instructions or provide additional resources on days with shorter cooking times to encourage users to spend more time cooking.

Order Amount by Age

```
sns.barplot(x='Age', y='Amount (USD)', data=df)
plt.title('Order Amount by Age')
plt.xlabel('Age')
plt.ylabel('Order Amount')
plt.show()
```



Insights:

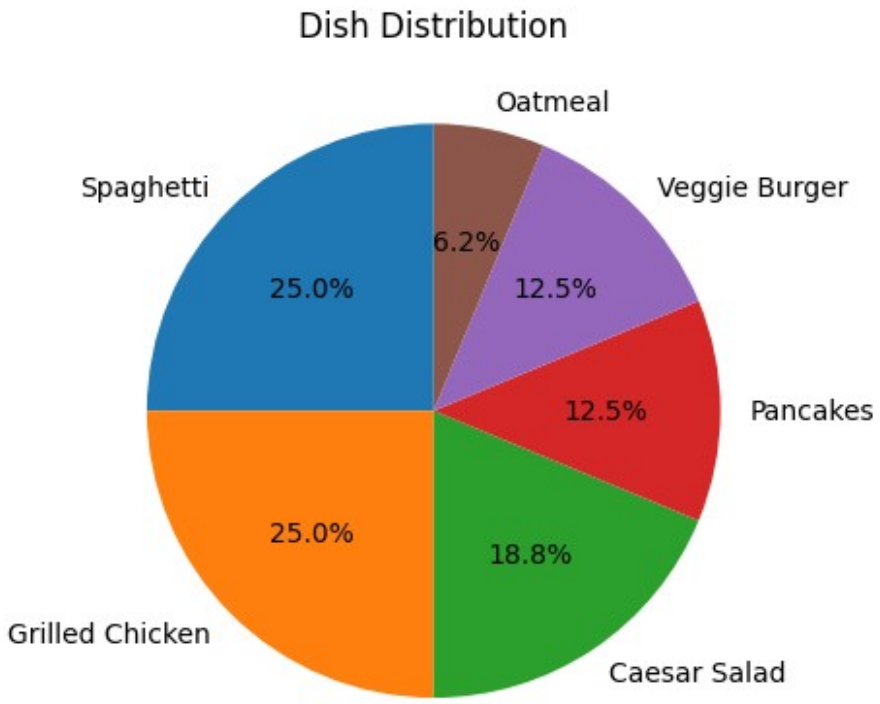
- Users aged 25 and 38 spent the maximum amounts.
- Users aged 27 and 35 recorded the minimum spending amounts.

Recommendations:

- Develop targeted marketing campaigns aimed at users aged 25 and 38 to encourage repeat purchases and loyalty.
- Analyze the preferences and behaviors of users aged 27 and 35 to identify barriers to higher spending.
- Create tailored promotions or discounts for users in the lower spending age groups to incentivize increased spending.

Dish Distribution

```
dish_distribution = df['Dish Name_y'].value_counts()
dish_distribution.plot.pie(autopct='%1.1f%%', startangle=90)
plt.title('Dish Distribution')
plt.ylabel('')
plt.show()
```



Insights:

- Spaghetti and Grilled Chicken each account for 25% of total orders, together contributing to 50% of all orders.
- Oatmeal contributes the least at 6.2% of total orders.
- Pancakes and Veggie Burger each contribute 12.5%, while Caesar Salad contributes 18.8% of total orders.

Recommendations:

- Focus marketing efforts on Spaghetti and Grilled Chicken to further capitalize on their popularity and drive sales.
- Consider improving the recipe or presentation of Oatmeal to increase its appeal and boost its contribution to total orders.
- Create promotional bundles that include Pancakes, Veggie Burger, and Caesar Salad to encourage customers to try these dishes together.