Financial Insights Dashboard and Scoring Model

1. Data Analysis

- Analyze The Dataset

In [2]:
```
import pandas as pd
import numpy as np
```

In [3]:
```
dataset = pd.read_excel(r"C:\Assignment 2\family_financial_and_transactions_data.xl
```

In [4]:
```
dataset.head()
```

Out[4]:

	Family ID	Member ID	Transaction Date	Category	Amount	Income	Savings	Monthly Expenses
0	FAM001	FAM001_Member1	2024-10-07	Travel	409.12	113810	20234	5781
1	FAM001	FAM001_Member1	2024-10-16	Travel	270.91	113810	20234	5781
2	FAM001	FAM001_Member1	2024-10-17	Groceries	91.10	113810	20234	5781
3	FAM001	FAM001_Member1	2024-10-25	Healthcare	198.23	113810	20234	5781
4	FAM001	FAM001_Member1	2024-10-25	Education	206.42	113810	20234	5781

In [5]:
```
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16306 entries, 0 to 16305
Data columns (total 12 columns):
 #   Column                  Non-Null Count  Dtype         
---  ------                  --------------  -----         
 0   Family ID               16306 non-null  object        
 1   Member ID               16306 non-null  object        
 2   Transaction Date        16306 non-null  datetime64[ns]
 3   Category                16306 non-null  object        
 4   Amount                  16306 non-null  float64       
 5   Income                  16306 non-null  int64         
 6   Savings                 16306 non-null  int64         
 7   Monthly Expenses        16306 non-null  int64         
 8   Loan Payments           16306 non-null  int64         
 9   Credit Card Spending    16306 non-null  int64         
 10  Dependents              16306 non-null  int64         
 11  Financial Goals Met (%)  16306 non-null  int64         
dtypes: datetime64[ns](1), float64(1), int64(7), object(3)
memory usage: 1.5+ MB
```

In [6]:
```
dataset.isnull().sum()
```

Out[6]:
```
Family ID                  0
Member ID                  0
Transaction Date           0
Category                   0
Amount                     0
Income                     0
Savings                    0
Monthly Expenses           0
Loan Payments              0
Credit Card Spending       0
Dependents                 0
Financial Goals Met (%)    0
dtype: int64
```

In [23]:
```
dataset['Family ID'].value_counts()
```

Out[23]:
```
Family ID
FAM194    167
FAM005    165
FAM187    156
FAM050    155
FAM071    144
         ... 
FAM022     24
FAM180     24
FAM075     23
FAM174     19
FAM197     19
Name: count, Length: 200, dtype: int64
```

- Identify family-level and member-level spending patterns.

In [24]:
```
# Family - level spending pattern
```

```
family_spending = dataset.groupby("Family ID")['Amount'].sum()
```

In [33]:
```
print("family-level-spending :",family_spending)
```

```
family-level-spending : Family ID
FAM001    23188.90
FAM002    22309.71
FAM003    11220.34
FAM004    23483.10
FAM005    40246.21
            ...
FAM196    31433.34
FAM197     4032.82
FAM198    19378.59
FAM199    31009.65
FAM200    24887.93
Name: Amount, Length: 200, dtype: float64
```

In [30]:
```
#  member-level spending patterns.
member_lavel = dataset.groupby("Member ID")['Amount'].sum()
```

In [35]:
```
print("member-lavel-spending :",member_lavel)
```

```
member-lavel-spending : Member ID
FAM001_Member1    6521.20
FAM001_Member2    7084.83
FAM001_Member3    2119.41
FAM001_Member4    7463.46
FAM002_Member1    3082.90
                   ...
FAM200_Member2    2103.85
FAM200_Member3    4671.33
FAM200_Member4    3739.52
FAM200_Member5    5408.28
FAM200_Member6    1837.73
Name: Amount, Length: 926, dtype: float64
```

- Understand correlations between financial metrics (e.g., income vs. expenses, savings vs. spending habits).

In [37]:
```
# Income vs Expenses

corr_income_expenses = dataset['Income'].corr(dataset['Monthly Expenses'])
print("corr_income_expenses :",corr_income_expenses)

# Savings vs spending habits

corr_savings_spending = dataset['Savings'].corr(dataset['Credit Card Spending'])
print("corr_savings_spending :",corr_savings_spending)
```

```
corr_income_expenses : -0.04135483845757865
corr_savings_spending : 0.02249898683943249
```

2. Build a Financial Scoring Model:

- Develop a scoring mechanism (range: 0–100) to evaluate each family's financial

health.

```
In [46]: # Create a new column to calculate Travel/Entertainment Spending (you can change ca
dataset['Travel/Entertainment Spending'] = dataset['Category'].apply(lambda x: data

# Scoring function
def calculate_financial_health(row):
    # Factor 1: Savings-to-Income Ratio
    savings_to_income = row["Savings"] / row["Income"]

    # Factor 2: Monthly Expenses as a percentage of Income
    expenses_to_income = row["Monthly Expenses"] / row["Income"]

    # Factor 3: Loan Payments as a percentage of Income
    loan_to_income = row["Loan Payments"] / row["Income"]

    # Factor 4: Credit Card Spending trends (lower is better)
    credit_card_spending_ratio = row["Credit Card Spending"] / row["Income"]

    # Factor 5: Spending category distribution (higher travel/entertainment lowers
    category_spending_ratio = row["Travel/Entertainment Spending"] / row["Monthly E

    # Factor 6: Financial Goals Met
    financial_goals_met = row["Financial Goals Met (%)"] / 100  # Convert to 0-1 sc

    # Weighted scoring (weights sum to 1)
    score = (savings_to_income * 0.2) + (1 - expenses_to_income * 0.2) + (1 - loan_
            + (1 - credit_card_spending_ratio * 0.1) + (1 - category_spending_ratio

    # Convert to scale of 0-100
    return score * 100

# Apply the scoring function
dataset['Financial Health Score'] = dataset.apply(calculate_financial_health, axis=

# Display the results
print(dataset[['Family ID', 'Financial Health Score']])
```

```
      Family ID  Financial Health Score
0        FAM001             -599.919078
1        FAM001             -599.919078
2        FAM001              415.454231
3        FAM001              415.454231
4        FAM001              415.454231
...         ...                     ...
16301    FAM200              434.490854
16302    FAM200              434.490854
16303    FAM200             -931.866944
16304    FAM200             -931.866944
16305    FAM200             -997.238126

[16306 rows x 2 columns]
```

3. Insights Visualization:

- Use Python with Matplotlib, Seaborn, or Plotly to visualize:
  - • Spending distribution across categories.
  - • Family-wise financial scores.
  - • Member-wise spending trends.
  - • Include at least 3 meaningful visualizations.

In [47]:

```python
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
```

In [48]:

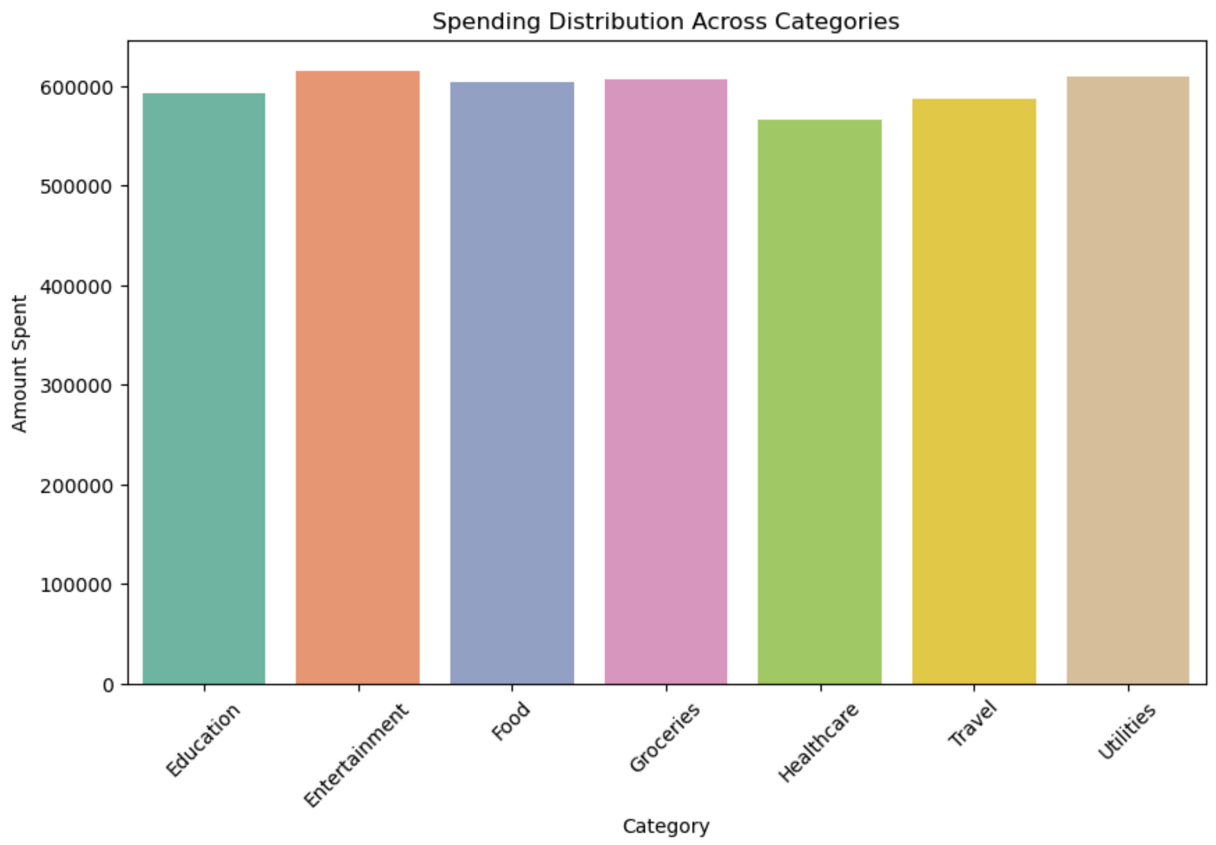
```python
# 1. Spending distribution across categories
category_spending = dataset.groupby('Category')['Amount'].sum().reset_index()

plt.figure(figsize=(10,6))
sns.barplot(data=category_spending, x='Category', y='Amount', palette='Set2')
plt.title('Spending Distribution Across Categories')
plt.xlabel('Category')
plt.ylabel('Amount Spent')
plt.xticks(rotation=45)
plt.show()
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_2800\976384966.py:5: FutureWarning: 

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(data=category_spending, x='Category', y='Amount', palette='Set2')
```

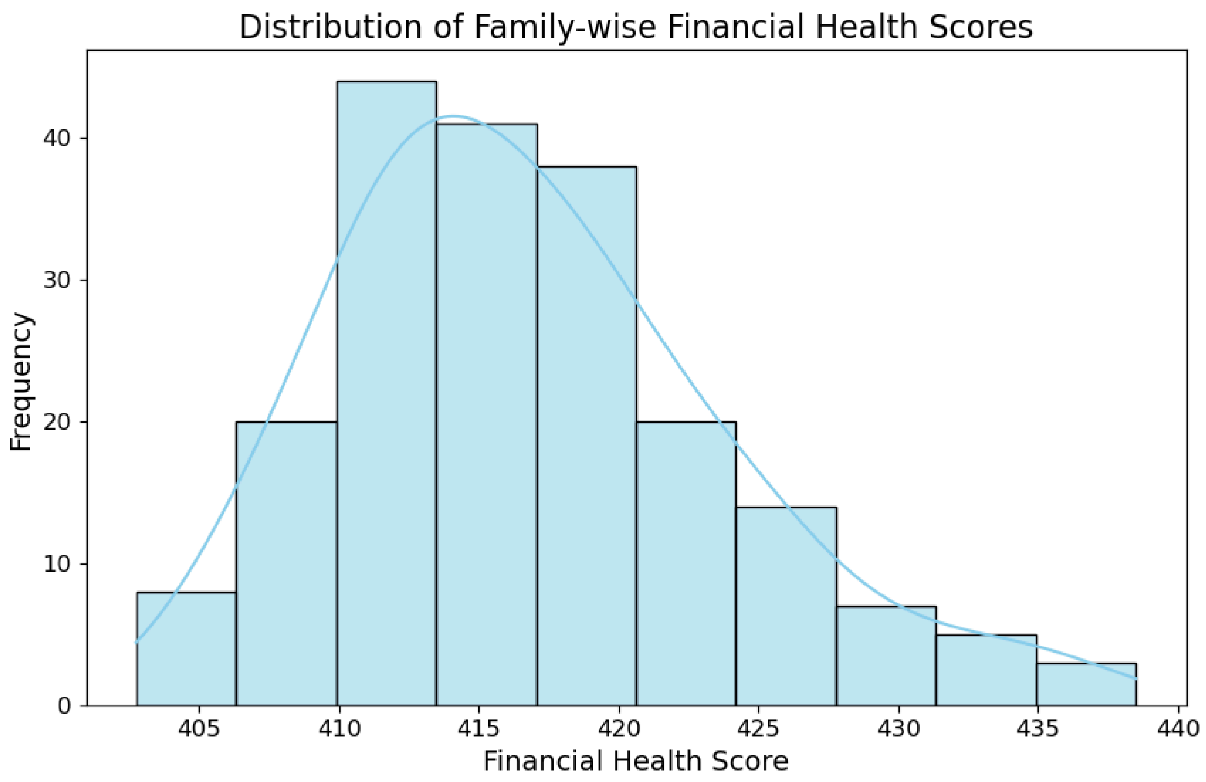


In [60]:
```
# Grouping by Family ID and calculating maximum Financial Health Score
family_scores = dataset.groupby('Family ID')['Financial Health Score'].max().reset_

# Creating a histogram to show the distribution of family financial health scores
plt.figure(figsize=(10,6))
sns.histplot(family_scores['Financial Health Score'], bins=10, kde=True, color='sky

# Enhancing the visualization with title and labels
plt.title('Distribution of Family-wise Financial Health Scores', fontsize=16)
plt.xlabel('Financial Health Score', fontsize=14)
plt.ylabel('Frequency', fontsize=14)
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)

# Display the plot
plt.show()
```



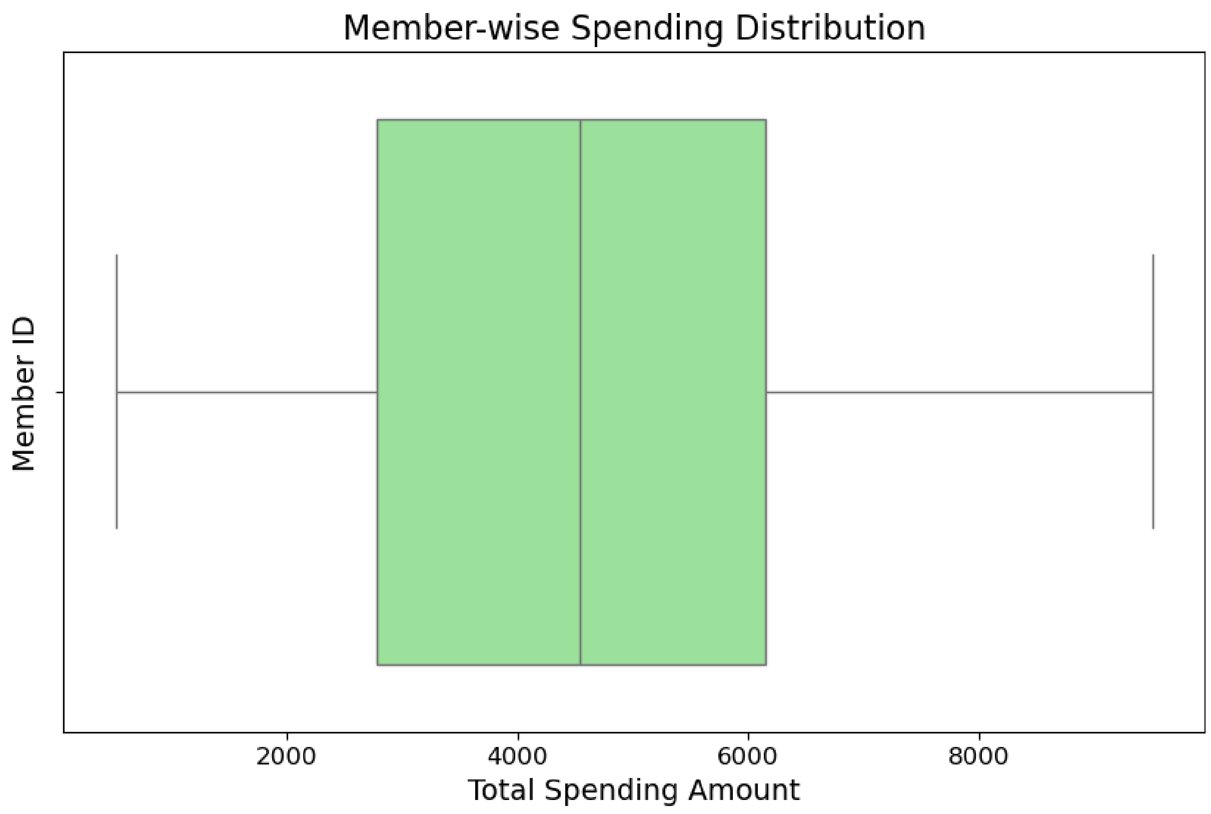
In [59]:
```
import matplotlib.pyplot as plt
import seaborn as sns

# Grouping by Member ID and calculating total spending
member_spending = dataset.groupby('Member ID')['Amount'].sum().reset_index()

# Creating a boxplot to show the distribution of member spending
plt.figure(figsize=(10,6))
sns.boxplot(data=member_spending, x='Amount', color='lightgreen')

# Enhancing the visualization with title and labels
plt.title('Member-wise Spending Distribution', fontsize=16)
plt.xlabel('Total Spending Amount', fontsize=14)
plt.ylabel('Member ID', fontsize=14)
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)

# Display the plot
plt.show()
```



```
In [51]: # 4. (Optional) A Plotly pie chart to visualize spending distribution across catego
         fig = px.pie(category_spending, names='Category', values='Amount', title='Spending
         fig.show()
```

```
In [ ]:
```