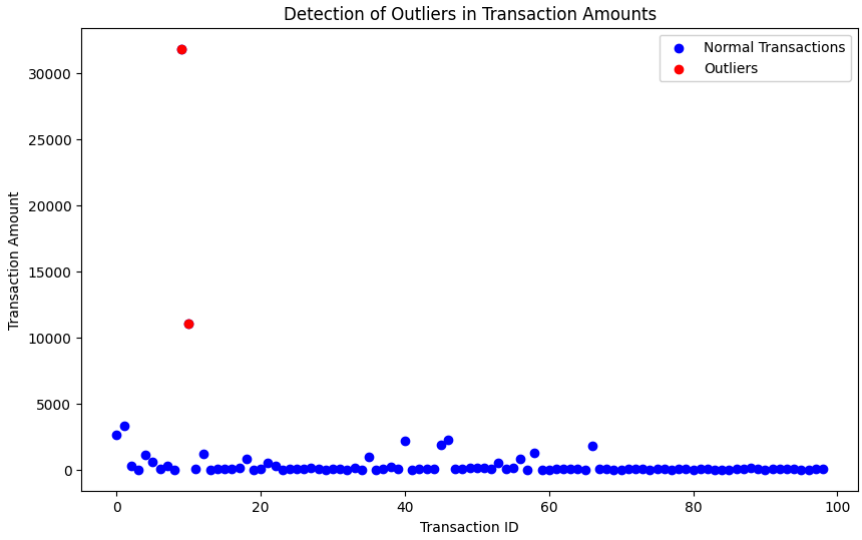
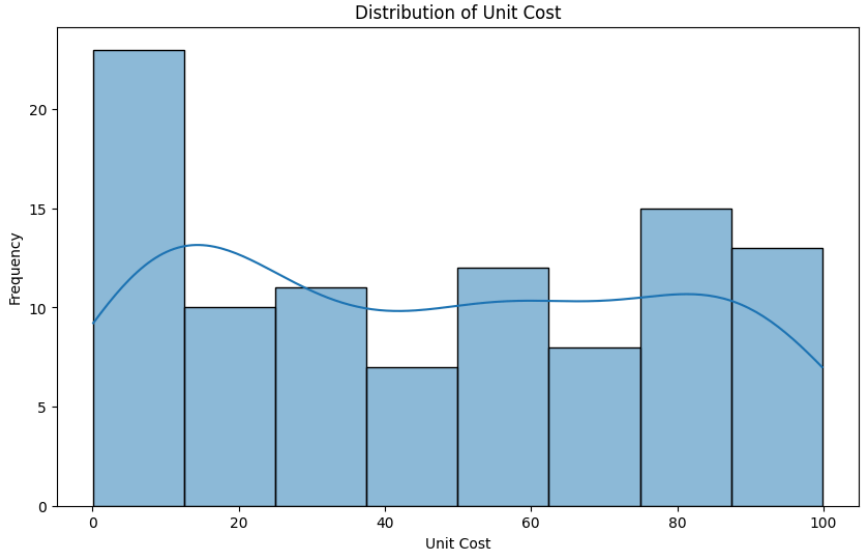
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Function definitions

def detect_outliers(df, column, threshold=3):
    """
    Detect outliers in a specific column using z-score.

    Args:
        df (DataFrame): DataFrame containing data.
        column (str): Column name to analyze for outliers.
        threshold (float): Z-score threshold for outlier detection (default: 3).

    Returns:
        DataFrame: DataFrame containing outliers.
    """
    mean_val = df[column].mean()
    std_val = df[column].std()
    outliers = df[(df[column] - mean_val).abs() > threshold * std_val]
    return outliers

def identify_suspicious_transactions(df):
    """
    Identify suspicious transactions.

    Args:
        df (DataFrame): DataFrame containing transaction data.

    Returns:
        DataFrame: DataFrame containing suspicious transactions.
    """
    suspicious_transactions = df[(df['Amount'] > 1000) & (df['Category'] == 'Set')]
    return suspicious_transactions

def calculate_inventory_turnover(df, cogs):
    """
    Calculate inventory turnover ratio.

    Args:
        df (DataFrame): DataFrame containing inventory data.
        cogs (float): Cost of Goods Sold value.

    Returns:
        float: Inventory turnover ratio.
    """
    average_inventory = df['Total Value'].mean()
    inventory_turnover = cogs / average_inventory
    return inventory_turnover


# Load and preprocess data

# Load the dataset
df = pd.read_csv('/content/savd6.csv')

# Display basic information about the DataFrame
print(df.head())  # Check the first few rows
print(df.info())  # Check data types and non-null counts

# Clean and convert numeric columns
numeric_columns = ['Amount', 'Total Value', 'Unit Cost']  # Adjust as per your dataset
df[numeric_columns] = df[numeric_columns].apply(pd.to_numeric, errors='coerce')

# Drop rows with NaN values in critical columns
df.dropna(subset=numeric_columns, inplace=True)

# Convert 'Date' column to datetime format if it exists
if 'Date' in df.columns:
    df['Date'] = pd.to_datetime(df['Date'], errors='coerce')


# Data analysis and visualization

# Detect outliers in 'Amount' column
outliers = detect_outliers(df, 'Amount')

# Visualize outliers
plt.figure(figsize=(10, 6))
```

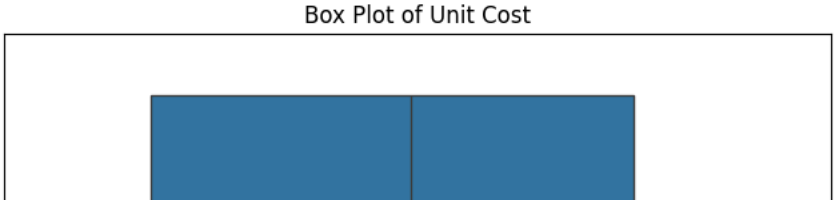
```
plt.scatter(df.index, df['Amount'], color='blue', label='Normal Transactions')
plt.scatter(outliers.index, outliers['Amount'], color='red', label='Outliers')
plt.title('Detection of Outliers in Transaction Amounts')
plt.xlabel('Transaction ID')
plt.ylabel('Transaction Amount')
plt.legend()
plt.show()

# Identify suspicious transactions
suspicious_transactions = identify_suspicious_transactions(df)

# Export suspicious transactions to a new CSV file
suspicious_transactions.to_csv('suspicious_transactions.csv', index=False)

# Calculate Inventory Turnover Ratio
total_cogs = 1500  # Replace with actual Cost of Goods Sold value
inventory_turnover = calculate_inventory_turnover(df, total_cogs)
print("Inventory Turnover Ratio:", inventory_turnover)

# Visualize distribution of 'Unit Cost' with a histogram
plt.figure(figsize=(10, 6))
sns.histplot(df['Unit Cost'], kde=True)
plt.title('Distribution of Unit Cost')
plt.xlabel('Unit Cost')
plt.ylabel('Frequency')
plt.show()

# Visualize 'Unit Cost' distribution with a box plot
plt.figure(figsize=(8, 6))
sns.boxplot(x=df['Unit Cost'])
plt.title('Box Plot of Unit Cost')
plt.xlabel('Unit Cost')
plt.show()

# Visualize correlation heatmap
plt.figure(figsize=(10, 8))
numerical_df = df.select_dtypes(include=['number'])
sns.heatmap(numerical_df.corr(), annot=True, cmap='coolwarm', vmin=-1, vmax=1)
plt.title('Correlation Heatmap')
plt.show()

# Visualize pairplot of numerical variables
sns.pairplot(df[['Unit Cost', 'Amount', 'Total Value']])
plt.title('Pairplot of Numerical Variables')
plt.show()

# Generate synthetic data with extended date range for demonstration

# Generate synthetic data with extended date range
np.random.seed(42)
dates_existing = pd.date_range(start='2023-01-01', end='2023-06-30', freq='D')  # Existing date range
dates_extended = pd.date_range(start='2023-01-01', end='2023-12-31', freq='D')  # Extended date range

# Create DataFrame with existing dates
unit_cost_existing = np.random.normal(loc=50, scale=10, size=len(dates_existing))
df_existing = pd.DataFrame({'Date': dates_existing, 'Unit Cost': unit_cost_existing})

# Merge with extended date range to create a complete dataset
df_complete = pd.merge(pd.DataFrame({'Date': dates_extended}), df_existing, on='Date', how='left')

# Display the first few rows of the complete DataFrame
print(df_complete.head())

# Plot median 'Unit Cost' over the extended date range
grouped_median = df_complete.groupby('Date')['Unit Cost'].median().reset_index()

plt.figure(figsize=(12, 6))
plt.plot(grouped_median['Date'], grouped_median['Unit Cost'], marker='o', linestyle='-')
plt.title('Median Unit Cost Over Time')
plt.xlabel('Date')
plt.ylabel('Median Unit Cost')
plt.xticks(rotation=45)  # Rotate x-axis labels if necessary
plt.tight_layout()
plt.show()
```

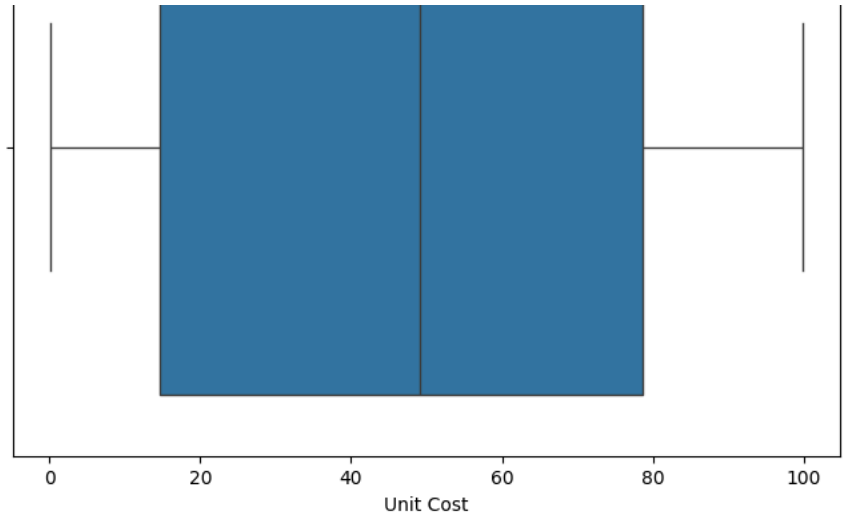
```
   index       Date    Style             SKU       Category  Qty  \
0      0  4/30/2022   SET389   SET389-KR-NP-S            Set   70
1      1  4/30/2022  JNE3781  JNE3781-KR-XXXL          kurta   60
2      2  4/30/2022  JNE3371    JNE3371-KR-XL          kurta    4
3      3  4/30/2022    J0341       J0341-DR-L  Western Dress    0
4      4  4/30/2022  JNE3671  JNE3671-TU-XXXL            Top   56

        Amount    B2B fulfilled-by  Unit Cost   Total Value
0  2663.752622  False    Easy Ship  38.053609  101365.40050
1  3309.603027  False    Easy Ship  55.160050  182557.86990
2   297.034165   True          NaN  74.258541   22057.32373
3     0.000000  False    Easy Ship  41.724057       0.00000
4  1126.242616  False          NaN  20.111475   22650.40055
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99 entries, 0 to 98
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   index         99 non-null     int64
 1   Date          99 non-null     object
 2   Style         99 non-null     object
 3   SKU           99 non-null     object
 4   Category      99 non-null     object
 5   Qty           99 non-null     int64
 6   Amount        99 non-null     float64
 7   B2B           99 non-null     bool
 8   fulfilled-by  25 non-null     object
 9   Unit Cost     99 non-null     float64
 10  Total Value   99 non-null     float64
dtypes: bool(1), float64(3), int64(2), object(5)
memory usage: 8.0+ KB
None
```

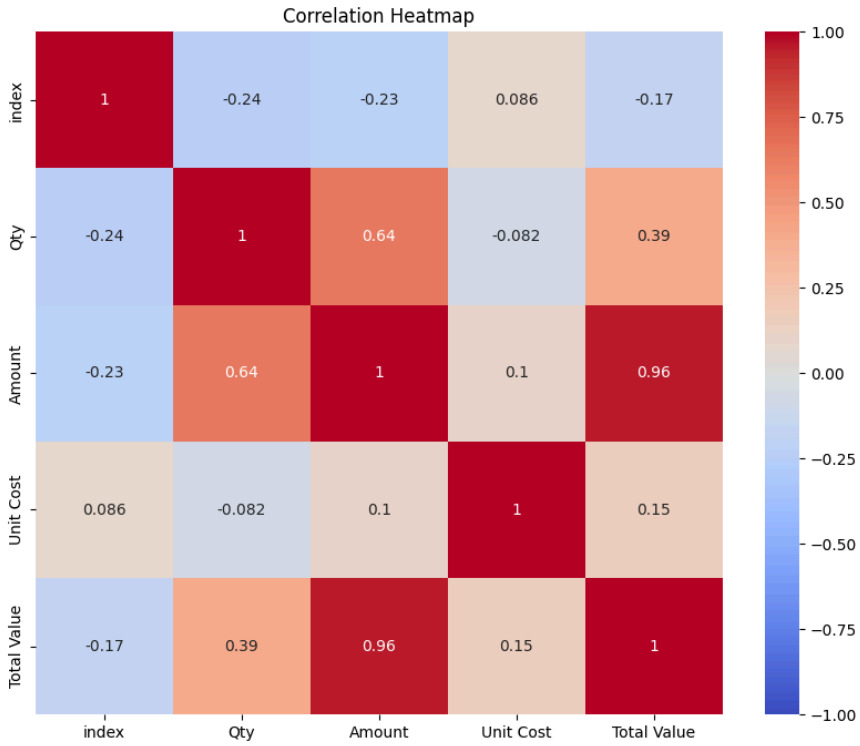


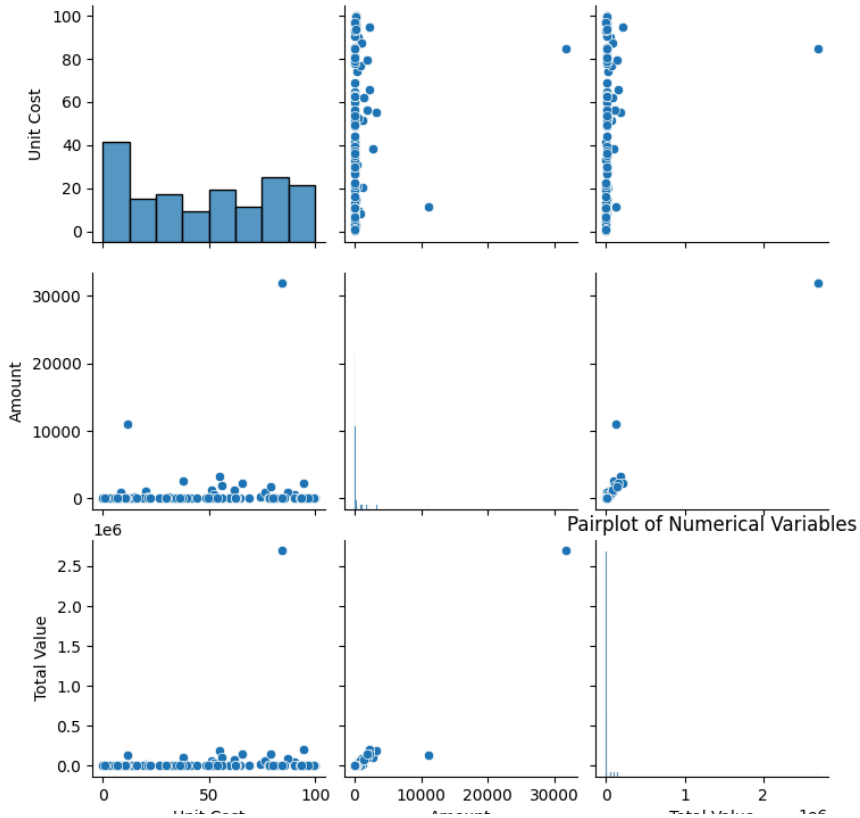

```
Inventory Turnover Ratio: 0.03398436668233852
```






0
20
40
60
80
100
Unit Cost


Correlation Heatmap
index
Qty
Amount
Unit Cost
Total Value
1
-0.24
-0.23
0.086
-0.17
-0.24
1
0.64
-0.082
0.39
-0.23
0.64
1
0.1
0.96
0.086
-0.082
0.1
1
0.15
-0.17
0.39
0.96
0.15
1
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00


Pairplot of Numerical Variables
Unit Cost
Amount
Total Value
1e6

Unit Cost Amount Total Value 1e6

	Date	Unit Cost
0	2023-01-01	54.967142
1	2023-01-02	48.617357
2	2023-01-03	56.476885
3	2023-01-04	65.230299
4	2023-01-05	47.658466