```
import requests
import pandas as pd

# Supabase credentials
SUPABASE_URL = "https://pvgaaikztozwlfhyrqlo.supabase.co"
API_KEY = "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZSIsInJlZiI6InB2Z2FhaWt6dG96d2xmaHlycWxvIiwicm9sZSI6ImFub24iLCJpYXQiOjE3ND
ENDPOINT = f"{SUPABASE_URL}/rest/v1/assets?select=*"

# Headers with authentication
headers = {
    "apikey": API_KEY,
    "Authorization": f"Bearer {API_KEY}",
    "Content-Type": "application/json"
}

# Make the GET request
response = requests.get(ENDPOINT, headers=headers)

if response.status_code == 200:
    data = response.json()
    df = pd.DataFrame(data)
    df.to_csv("assets.csv", index=False)
    print("✅ Data saved to assets.csv")
else:
    print("❌ Failed to fetch data:", response.status_code, response.text)
```

```
✅ Data saved to assets.csv
```

```
# Load datasets
assets_df = pd.read_csv("assets.csv")
personality_df = pd.read_csv("personality.csv")
```

```
assets_df.head()
```

	_id	asset_allocation	asset_allocation_id	asset_currency	asset_value	created
0	1	Equities	39958838	USD	217.06	2025-02-25T09:18:34.158728+00:00
1	1	Commodities	83197857	GBP	159.05	2025-05-18T09:18:34.162165+00:00
2	2	Cash	22575562	USD	231.12	2025-03-06T09:18:34.162165+00:00
3	2	Cash	85329037	USD	321.75	2025-02-22T09:18:34.163356+00:00
4	3	Crypto	66306997	USD	181.15	2025-04-17T09:18:34.163356+00:00

Next steps: Generate code with assets_df | View recommended plots | New interactive sheet

```
personality_df.head()
```

	_id	confidence	risk_tolerance	composure	impulsivity	impact_desire
0	1	0.550	0.510	0.565	0.161	0.999
1	2	0.486	0.474	0.439	0.818	0.048
2	3	0.565	0.568	0.578	0.832	0.977
3	4	0.652	0.625	0.642	0.507	0.407
4	5	0.477	0.483	0.515	0.006	0.871

Next steps: Generate code with personality_df | View recommended plots | New interactive sheet

```
# Filter GBP assets and calculate total per person
gbp_assets = assets_df[assets_df['asset_currency'] == 'GBP']
gbp_totals = gbp_assets.groupby('_id')['asset_value'].sum().reset_index()
gbp_with_risk = gbp_totals.merge(personality_df, on='_id')


gbp_totals.drop(columns="_id").describe()
```

	asset_value
count	122.000000
mean	222.251803
std	100.289650
min	100.520000
25%	156.090000
50%	193.895000
75%	264.470000
max	542.860000

```
#all gbp holder id's asset values merged with personality table

gbp_with_risk.head(10)
```

	_id	asset_value	confidence	risk_tolerance	composure	impulsivity	impact_desire
0	1	159.05	0.550	0.510	0.565	0.161	0.999
1	7	160.91	0.658	0.649	0.517	0.617	0.567
2	10	263.37	0.554	0.532	0.611	0.260	0.404
3	18	184.34	0.531	0.513	0.556	0.342	0.120
4	19	165.90	0.409	0.446	0.493	0.633	0.544
5	20	159.09	0.359	0.397	0.457	0.932	0.006
6	23	167.52	0.507	0.480	0.616	0.688	0.396
7	24	212.88	0.358	0.463	0.524	0.068	0.695
8	29	377.97	0.440	0.477	0.479	0.346	0.227
9	32	278.79	0.685	0.603	0.666	0.872	0.668

Next steps: Generate code with gbp_with_risk | View recommended plots | New interactive sheet

```
#id with highest asset values among all gbp holders

top_gbp_holder = gbp_with_risk.loc[gbp_with_risk['asset_value'].idxmax()]
top_gbp_holder.head()
```

	54
_id	134.000
asset_value	542.860
confidence	0.547
risk_tolerance	0.555
composure	0.417

dtype: float64

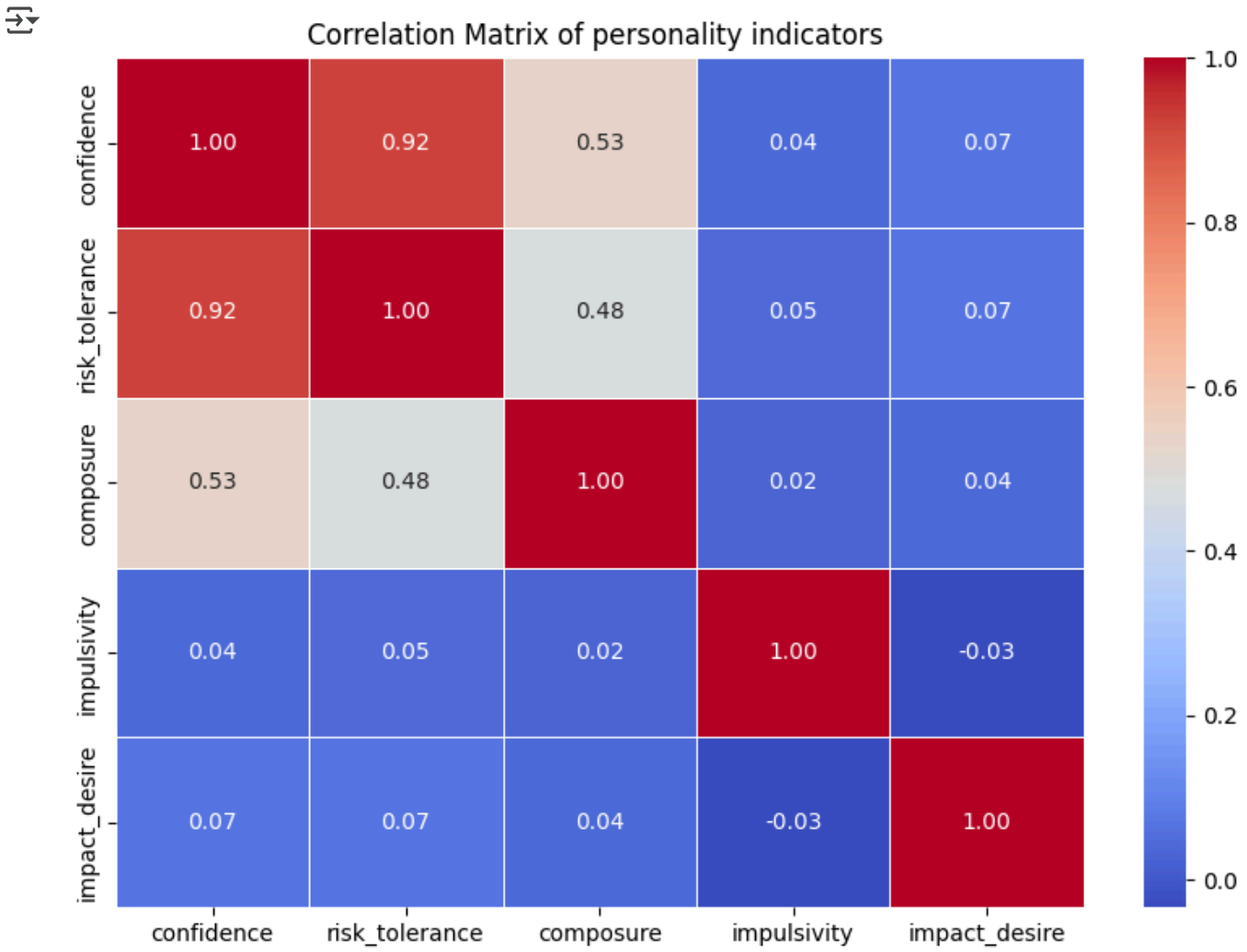
```
personality_df.describe()
```

	_id	confidence	risk_tolerance	composure	impulsivity	impact_desire
count	297.000000	297.000000	297.000000	297.000000	297.000000	297.000000
mean	151.306397	0.499384	0.498838	0.505064	0.501101	0.493010
std	86.646009	0.098862	0.073934	0.070385	0.297984	0.285869
min	1.000000	0.176000	0.299000	0.311000	0.005000	0.006000
25%	77.000000	0.431000	0.450000	0.457000	0.228000	0.239000
50%	152.000000	0.506000	0.500000	0.507000	0.507000	0.488000
75%	226.000000	0.563000	0.545000	0.547000	0.736000	0.731000
max	300.000000	0.885000	0.745000	0.700000	0.997000	0.999000

```
import matplotlib.pyplot as plt
import seaborn as sns
# Drop the '_id' column and calculate correlation matrix
correlation_matrix = personality_df.drop(columns="_id").corr()

# Plot the correlation matrix
plt.figure(figsize=(8, 6))
```

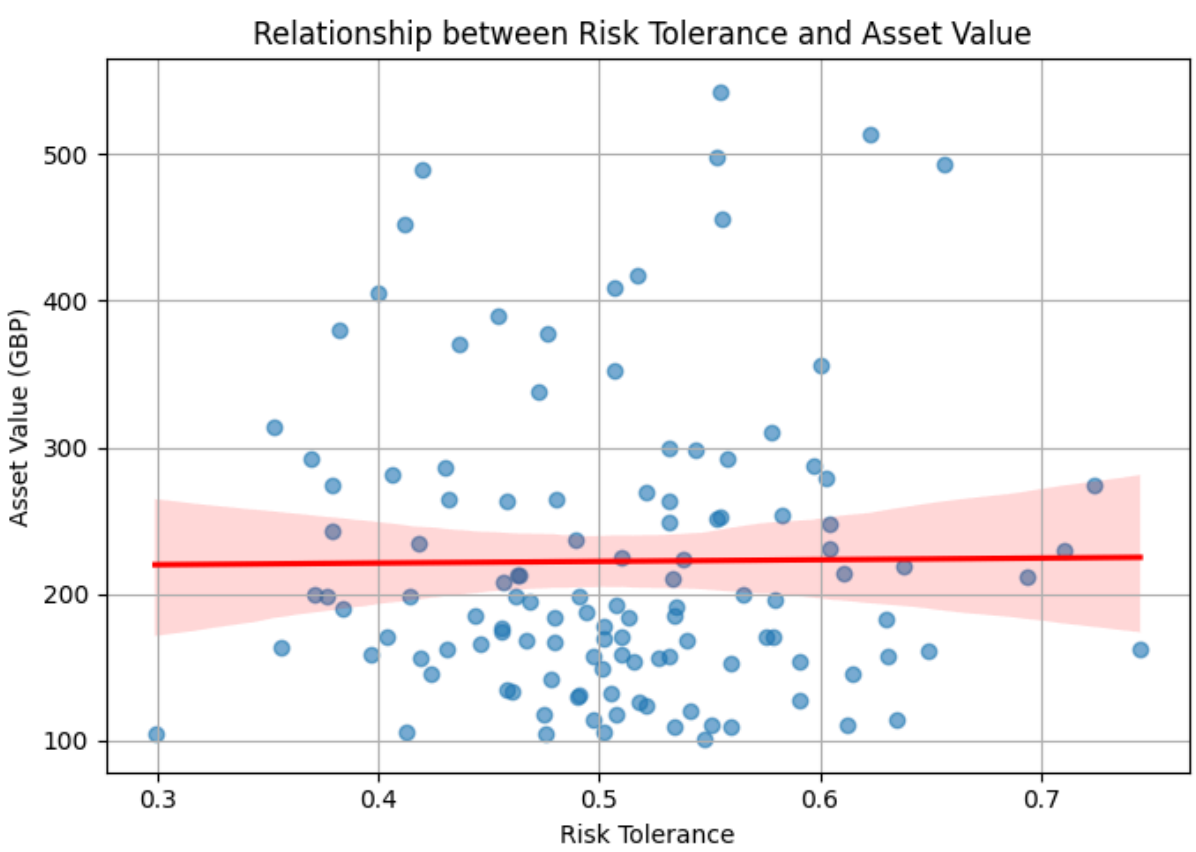
```
sns.heatmap(correlation_matrix, annot=True, cmap="coolwarm", fmt=".2f", linewidths=0.5)
plt.title("Correlation Matrix of personality indicators ")
plt.tight_layout()
plt.savefig("gbp_with_risk_correlation_matrix.png")
plt.show()
```




```
# Scatter plot with regression line
plt.figure(figsize=(7, 5))
sns.regplot(data=gbp_with_risk, x='risk_tolerance', y='asset_value', scatter_kws={'alpha':0.6}, line_kws={'color':'red'})

plt.title('Relationship between Risk Tolerance and Asset Value ')
```

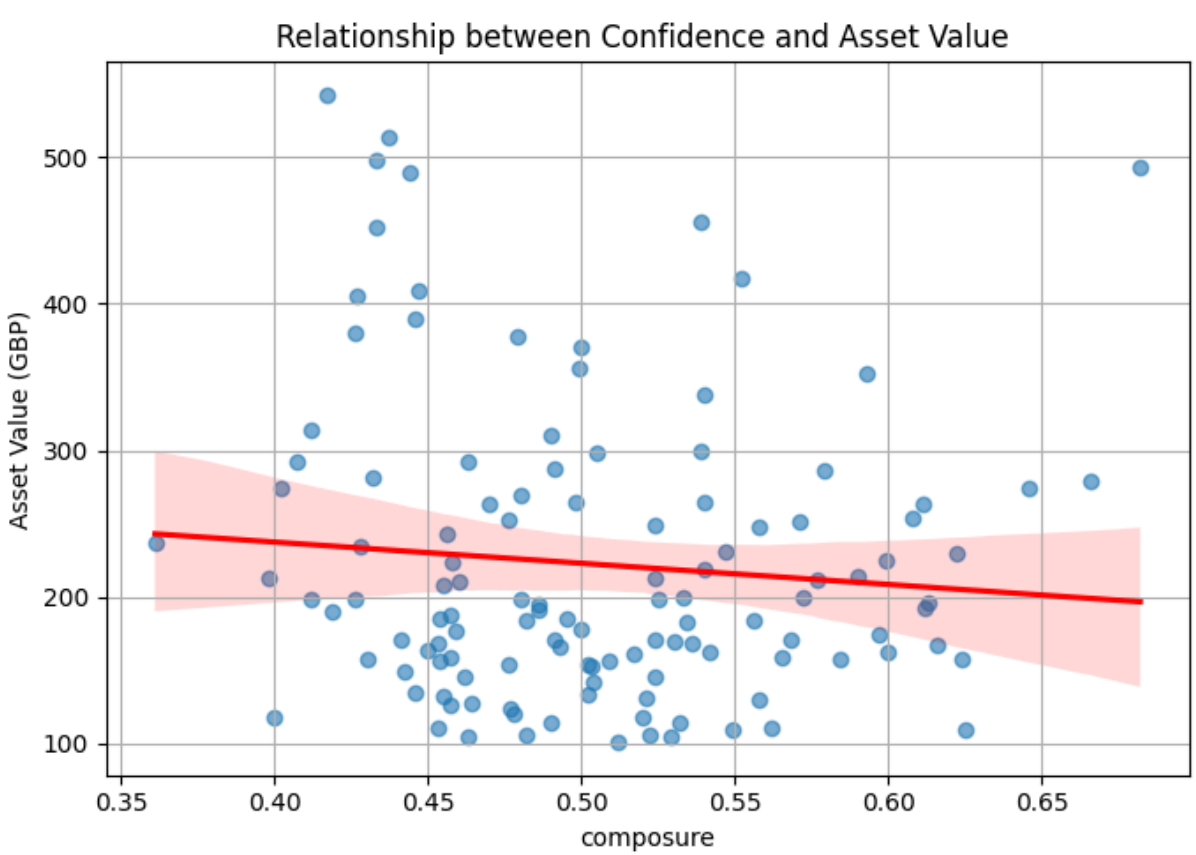
```
plt.xlabel('Risk Tolerance')
plt.ylabel('Asset Value (GBP)')
plt.grid(True)
plt.tight_layout()
plt.show()
```




```
# Scatter plot with regression line
plt.figure(figsize=(7, 5))
sns.regplot(data=gbp_with_risk, x='composure', y='asset_value', scatter_kws={'alpha':0.6}, line_kws={'color':'red'})

plt.title('Relationship between Composure and Asset Value ')
plt.xlabel('composure')
plt.ylabel('Asset Value (GBP)')
plt.grid(True)
```

```
plt.tight_layout()
plt.show()
```




```
# Scatter plot with regression line
plt.figure(figsize=(7, 5))
sns.regplot(data=gbp_with_risk, x='impact_desire', y='asset_value', scatter_kws={'alpha':0.6}, line_kws={'color':'red'})

plt.title('Relationship between Impact Desire and Asset Value ')
plt.xlabel('Impact Desire')
plt.ylabel('Asset Value (GBP)')
plt.grid(True)
plt.tight_layout()
plt.show()
```


Relationship between Impact Desire and Asset Value
Asset Value (GBP)
500
400
300
200
100
0.0
0.2
0.4
0.6
0.8
1.0
Impact Desire