Money at a fixed rate for an unsecured purchase

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt

pd.set_option('display.max_rows', None)

credit_def_df = pd.read_csv('CreditCardDelinquencyRate.csv').rename(columns={'DRCCLACBS': 'DefaultRate', 'DATE': 'Date'})
credit_int_df = pd.read_csv('CreditCardInterestRate.csv').rename(columns={'TERMCBCCALLNS': 'InterestRate', 'DATE': 'Date'})
consum_spend_df = pd.read_csv('UsConsumerSpending.csv').rename(columns={'date': 'Date'})
```

In [2]:
```
credit_def_df['Date'] = pd.to_datetime(credit_def_df['Date'], format='%Y/%m/%d')
credit_int_df['Date'] = pd.to_datetime(credit_int_df['Date'], format='%Y/%m/%d')
consum_spend_df['Date'] = pd.to_datetime(consum_spend_df['Date'], format='%d/%m/%Y')
```

In [3]:
```
credit_int_df.head(10)
```

Out[3]:

	Date	InterestRate
0	1994-11-01	15.69
1	1994-12-01	.
2	1995-01-01	.
3	1995-02-01	16.10
4	1995-03-01	.
5	1995-04-01	.
6	1995-05-01	16.14
7	1995-06-01	.
8	1995-07-01	.
9	1995-08-01	15.92

In [4]:
```
credit_def_df.head(10)
```

Out[4]:

	Date	DefaultRate
0	1991-01-01	5.26
1	1991-04-01	5.48
2	1991-07-01	5.35
3	1991-10-01	5.32
4	1992-01-01	5.27
5	1992-04-01	5.10
6	1992-07-01	4.98
7	1992-10-01	4.69
8	1993-01-01	4.60
9	1993-04-01	4.46

In [5]:
```
consum_spend_df.head(20)
```

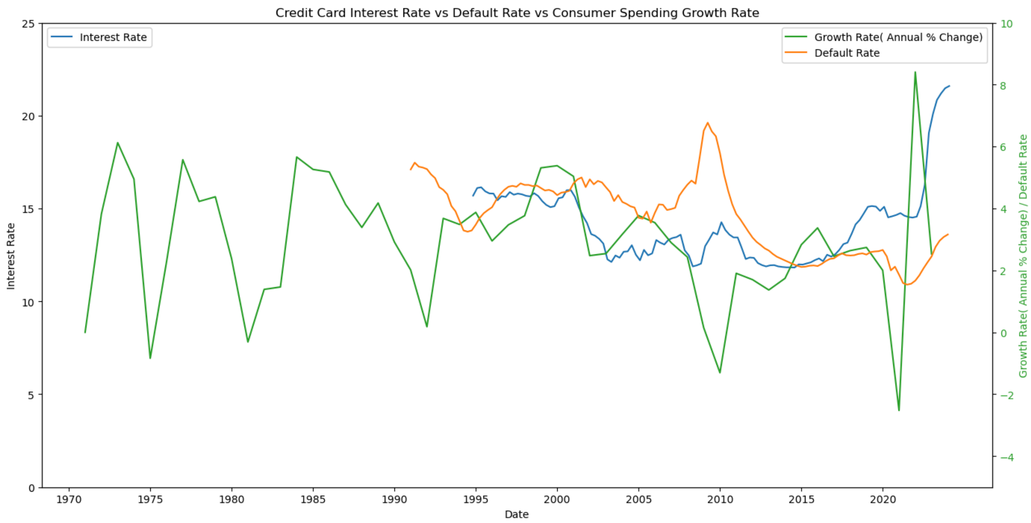
Out[5]:

	Date	Spending(Billions of US $)	Per Capita	Growth Rate(Annual % Change)
0	1960-12-31	0.000	0.0000	0.0000
1	1961-12-31	0.000	0.0000	0.0000
2	1962-12-31	0.000	0.0000	0.0000
3	1963-12-31	0.000	0.0000	0.0000
4	1964-12-31	0.000	0.0000	0.0000
5	1965-12-31	0.000	0.0000	0.0000
6	1966-12-31	0.000	0.0000	0.0000
7	1967-12-31	0.000	0.0000	0.0000
8	1968-12-31	0.000	0.0000	0.0000
9	1969-12-31	0.000	0.0000	0.0000
10	1970-12-31	646.724	15525.7845	0.0000
11	1971-12-31	699.937	15916.1874	3.8189
12	1972-12-31	768.153	16711.1675	6.1248
13	1973-12-31	849.575	17371.5692	4.9488
14	1974-12-31	930.161	17069.3783	-0.8377
15	1975-12-31	1030.547	17285.1481	2.2675
16	1976-12-31	1147.666	18075.9371	5.5734
17	1977-12-31	1273.975	18651.3664	4.2264
18	1978-12-31	1422.252	19262.4486	4.3764
19	1979-12-31	1585.420	19503.8822	2.3770

In [6]:
```
# Cleanup invalid '.' datapoints
credit_int_df = credit_int_df[credit_int_df['InterestRate'] != '.']
credit_int_df['InterestRate'] = credit_int_df['InterestRate'].astype(float)
```

In [7]:
```
# drop 0 values
consum_spend_df = consum_spend_df.iloc[10:].reset_index(drop=True)
```

In [8]:
```
# Set the figure size
fig, ax1 = plt.subplots(figsize=(14, 7))  # Adjust the size as needed

# Plot interest rate and default rate on the left y-axis
ax1.plot(credit_int_df['Date'], credit_int_df['InterestRate'], color='tab:blue', label='Interest Rate')
ax1.set_xlabel('Date')
ax1.set_ylabel('Interest Rate', color='black')
ax1.tick_params(axis='y', labelcolor='black')

# Create a second y-axis for consumer spending
ax2 = ax1.twinx()
ax2.plot(consum_spend_df['Date'], consum_spend_df['Growth Rate( Annual % Change)'], color='tab:green', label='Growth Rate( Annual % Change)')
ax2.plot(credit_def_df['Date'], credit_def_df['DefaultRate'], color='tab:orange', label='Default Rate')
ax2.set_ylabel('Growth Rate( Annual % Change) / Default Rate', color='tab:green')
ax2.tick_params(axis='y', labelcolor='tab:green')

ax1.set_ylim(0, 25)
ax2.set_ylim(-5, 10)

# Set x-axis major ticks to every 5 years starting from the nearest multiple of 5
start_year = min(credit_int_df['Date'].min().year, credit_def_df['Date'].min().year, consum_spend_df['Date'].min().year)
end_year = max(credit_int_df['Date'].max().year, credit_def_df['Date'].max().year, consum_spend_df['Date'].max().year)

start_year = (start_year // 5) * 5
years = pd.date_range(start=f'{start_year}', end=f'{end_year}', freq='5YS')
ax1.set_xticks(years)
ax1.set_xticklabels([year.year for year in years])

# Rotate x-axis labels for better readability
plt.xticks(rotation=45)

# Add legends
fig.tight_layout()
ax1.legend(loc='upper left')
ax2.legend(loc='upper right')

# Show plot
plt.title('Credit Card Interest Rate vs Default Rate vs Consumer Spending Growth Rate')
plt.show()
```



In [9]:
```
#Check correlation between Default Rate and Interest Rate
correlation = credit_def_df['DefaultRate'].corr(credit_int_df['InterestRate'])

correlation
```

Out[9]:
```
0.8811479111090543
```

Money at a floating rate for a secured purchase

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt

house_data_df = pd.read_csv('house_data.csv')
mortgage_def_df = pd.read_csv('ResidentialMortgageDelinquencyRate.csv').rename(columns={'DRSFRMACBS': 'DefaultRate', 'DATE': 'Date'})
mortgage_rate_df = pd.read_csv('FixedMortgRate.csv').rename(columns={'MORTGAGE30US': 'FixedMortgageRate', 'DATE': 'Date'})
```

In [2]:
```
mortgage_def_df['Date'] = pd.to_datetime(mortgage_def_df['Date'], format='%Y/%m/%d')
mortgage_rate_df['Date'] = pd.to_datetime(mortgage_rate_df['Date'], format='%Y/%m/%d')
```

In [3]:
```
mortgage_rate_df.head(10)
```

Out[3]:

	Date	FixedMortgageRate
0	1971-04-02	7.33
1	1971-04-09	7.31
2	1971-04-16	7.31
3	1971-04-23	7.31
4	1971-04-30	7.29
5	1971-05-07	7.38
6	1971-05-14	7.42
7	1971-05-21	7.44
8	1971-05-28	7.46
9	1971-06-04	7.52

In [4]:
```
house_data_df.head(20)
```

Out[4]:

	RegionID	SizeRank	RegionName	RegionType	StateName	State	Metro	CountyName	2000-01-31	2000-02-29	...	2023-08-31	2023-09-30	2023-10-31	2023-11-30	2023-12-31
0	6181	0	New York	city	NY	NY	New York-Newark-Jersey City, NY-NJ-PA	Queens County	235313.088433	236565.381466	...	7.380623e+05	7.355703e+05	7.325217e+05	7.297100e+05	7.284443e+05
1	12447	1	Los Angeles	city	CA	CA	Los Angeles-Long Beach-Anaheim, CA	Los Angeles County	226580.683895	226923.751717	...	9.419215e+05	9.579954e+05	9.712295e+05	9.800788e+05	9.849260e+05
2	39051	2	Houston	city	TX	TX	Houston-The Woodlands-Sugar Land, TX	Harris County	102227.139184	102201.165955	...	2.677462e+05	2.682087e+05	2.683647e+05	2.682082e+05	2.680024e+05
3	17426	3	Chicago	city	IL	IL	Chicago-Naperville-Elgin, IL-IN-WI	Cook County	144397.165602	144435.744175	...	2.899196e+05	2.916767e+05	2.927381e+05	2.932435e+05	2.931542e+05
4	6915	4	San Antonio	city	TX	TX	San Antonio-New Braunfels, TX	Bexar County	99443.420882	99535.742667	...	2.668532e+05	2.663993e+05	2.655421e+05	2.642577e+05	2.627248e+05
5	13271	5	Philadelphia	city	PA	PA	Philadelphia-Camden-Wilmington, PA-NJ-DE-MD	Philadelphia County	63138.165503	63269.027622	...	2.216686e+05	2.219053e+05	2.218581e+05	2.214571e+05	2.213910e+05
6	40326	6	Phoenix	city	AZ	AZ	Phoenix-Mesa-Chandler, AZ	Maricopa County	117359.383719	117644.515197	...	4.172391e+05	4.202585e+05	4.228951e+05	4.250698e+05	4.263857e+05
7	18959	7	Las Vegas	city	NV	NV	Las Vegas-Henderson-Paradise, NV	Clark County	153248.213366	153221.098387	...	4.017398e+05	4.042723e+05	4.066113e+05	4.088349e+05	4.110797e+05
8	54296	8	San Diego	city	CA	CA	San Diego-Chula Vista-Carlsbad, CA	San Diego County	223427.417738	224455.001440	...	9.593671e+05	9.729681e+05	9.839130e+05	9.912402e+05	9.945421e+05
9	38128	9	Dallas	city	TX	TX	Dallas-Fort Worth-Arlington, TX	Dallas County	94981.624756	95040.231625	...	3.134173e+05	3.141809e+05	3.147692e+05	3.147719e+05	3.143633e+05
10	10221	10	Austin	city	TX	TX	Austin-Round Rock-Georgetown, TX	Travis County	193428.907494	194184.293051	...	5.683270e+05	5.666457e+05	5.634844e+05	5.594532e+05	5.553076e+05
11	33839	11	San Jose	city	CA	CA	San Jose-Sunnyvale-Santa Clara, CA	Santa Clara County	343085.289021	344815.930601	...	1.363560e+06	1.391073e+06	1.415823e+06	1.435144e+06	1.448470e+06
12	25290	12	Jacksonville	city	FL	FL	Jacksonville, FL	Duval County	94052.720529	94232.490237	...	2.982350e+05	2.985105e+05	2.986777e+05	2.986189e+05	2.987917e+05
13	24043	13	Charlotte	city	NC	NC	Charlotte-Concord-Gastonia, NC-SC	Mecklenburg County	147749.828741	147909.342940	...	3.892959e+05	3.917292e+05	3.938926e+05	3.954953e+05	3.965330e+05
14	32149	14	Indianapolis	city	IN	IN	Indianapolis-Carmel-Anderson, IN	Marion County	NaN	NaN	...	2.255028e+05	2.258244e+05	2.259311e+05	2.257039e+05	2.253810e+05
15	18172	15	Fort Worth	city	TX	TX	Dallas-Fort Worth-Arlington, TX	Tarrant County	102095.680794	102173.975541	...	3.086057e+05	3.087099e+05	3.086726e+05	3.084999e+05	3.082679e+05
16	13121	16	Orlando	city	FL	FL	Orlando-Kissimmee-Sanford, FL	Orange County	108662.545611	108930.264124	...	3.754663e+05	3.776389e+05	3.796402e+05	3.814458e+05	3.830508e+05
17	20330	17	San Francisco	city	CA	CA	San Francisco-Oakland-Berkeley, CA	San Francisco County	423441.993833	425600.377930	...	1.304478e+06	1.307525e+06	1.307231e+06	1.302297e+06	1.293812e+06
18	7481	18	Tucson	city	AZ	AZ	Tucson, AZ	Pima County	112045.295057	112213.735008	...	3.242250e+05	3.261328e+05	3.279115e+05	3.292494e+05	3.301280e+05
19	10920	19	Columbus	city	OH	OH	Columbus, OH	Franklin County	95298.113501	95329.813509	...	2.375789e+05	2.390541e+05	2.400725e+05	2.406753e+05	2.411013e+05

20 rows × 301 columns

In [5]:
```
# Get the mean ZHVI for each month
house_data_df = house_data_df[house_data_df.columns[8:]].mean().reset_index().rename(columns={'index': 'Date', 0: 'ZHVI Avg Value'})
```

In [6]:
```
# Melt the DataFrame
house_data_df['Date'] = pd.to_datetime(house_data_df['Date'], format='%Y/%m/%d')
```

In [7]:
```
house_data_df.head(20)
```

Out[7]:

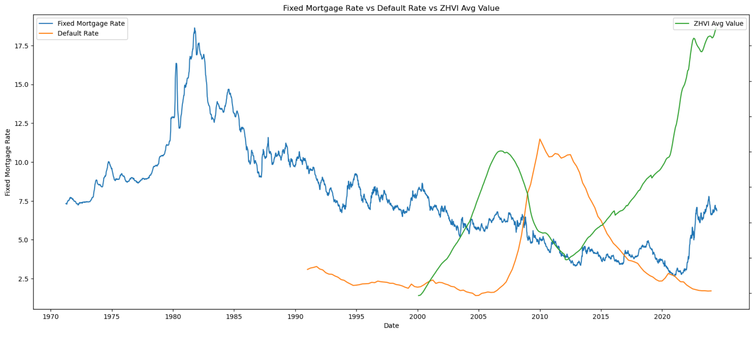
	Date	ZHVI Avg Value
0	2000-01-31	148312.235819
1	2000-02-29	148442.301919
2	2000-03-31	148707.437482
3	2000-04-30	149550.832145
4	2000-05-31	150409.249137
5	2000-06-30	151424.569886
6	2000-07-31	152634.797840
7	2000-08-31	153784.503734
8	2000-09-30	155026.576089
9	2000-10-31	156248.931603
10	2000-11-30	157438.342649
11	2000-12-31	158682.905626
12	2001-01-31	159729.040917
13	2001-02-28	160775.929880
14	2001-03-31	161810.224696
15	2001-04-30	162899.955468
16	2001-05-31	163917.452535
17	2001-06-30	165020.065776
18	2001-07-31	166108.708560
19	2001-08-31	167131.251691

In [8]:
```
mortgage_def_df.head(10)
```

Out[8]:

	Date	DefaultRate
0	1991-01-01	3.09
1	1991-04-01	3.18
2	1991-07-01	3.22
3	1991-10-01	3.28
4	1992-01-01	3.12
5	1992-04-01	3.06
6	1992-07-01	2.88
7	1992-10-01	2.79
8	1993-01-01	2.78
9	1993-04-01	2.67

In [9]:
```
# Set the figure size
fig, ax1 = plt.subplots(figsize=(17, 7))

ax1.plot(mortgage_rate_df['Date'], mortgage_rate_df['FixedMortgageRate'], color='tab:blue', label='Fixed Mortgage Rate')
ax1.plot(mortgage_def_df['Date'], mortgage_def_df['DefaultRate'], color='tab:orange', label='Default Rate')
ax1.set_xlabel('Date')
ax1.set_ylabel('Fixed Mortgage Rate', color='black')
ax1.tick_params(axis='y', labelcolor='black')


ax2 = ax1.twinx()
ax2.plot(house_data_df['Date'], house_data_df['ZHVI Avg Value'], color='tab:green', label='ZHVI Avg Value')
ax2.set_ylabel('ZHVI Avg Value')
ax2.tick_params(axis='y')


start_year = min(mortgage_rate_df['Date'].min().year, mortgage_def_df['Date'].min().year)
end_year = max(mortgage_rate_df['Date'].max().year, mortgage_def_df['Date'].max().year)

start_year = (start_year // 5) * 5
years = pd.date_range(start=f'{start_year}', end=f'{end_year}', freq='5YS')
ax1.set_xticks(years)
ax1.set_xticklabels([year.year for year in years])

plt.xticks(rotation=45)

# Add legends
fig.tight_layout()
ax1.legend(loc='upper left')
ax2.legend(loc='upper right')

# Show plot
plt.title('Fixed Mortgage Rate vs Default Rate vs ZHVI Avg Value')
plt.show()
```



3. Money at a fixed rate for a business for a construction loan.

In [1]:
```
pip install yfinance
```

```
Requirement already satisfied: yfinance in c:\users\owner\anaconda3\lib\site-packages (0.2.40)
Requirement already satisfied: pandas>=1.3.0 in c:\users\owner\anaconda3\lib\site-packages (from yfinance) (1.5.3)
Requirement already satisfied: numpy>=1.16.5 in c:\users\owner\anaconda3\lib\site-packages (from yfinance) (1.24.3)
Requirement already satisfied: requests>=2.31 in c:\users\owner\anaconda3\lib\site-packages (from yfinance) (2.31.0)
Requirement already satisfied: multitasking>=0.0.7 in c:\users\owner\anaconda3\lib\site-packages (from yfinance) (0.0.11)
Requirement already satisfied: lxml>=4.9.1 in c:\users\owner\anaconda3\lib\site-packages (from yfinance) (4.9.2)
Requirement already satisfied: platformdirs>=2.0.0 in c:\users\owner\anaconda3\lib\site-packages (from yfinance) (2.5.2)
Requirement already satisfied: pytz>=2022.5 in c:\users\owner\anaconda3\lib\site-packages (from yfinance) (2022.7)
Requirement already satisfied: frozendict>=2.3.4 in c:\users\owner\anaconda3\lib\site-packages (from yfinance) (2.4.4)
Requirement already satisfied: peewee>=3.16.2 in c:\users\owner\anaconda3\lib\site-packages (from yfinance) (3.17.5)
Requirement already satisfied: beautifulsoup4>=4.11.1 in c:\users\owner\anaconda3\lib\site-packages (from yfinance) (4.12.2)
Requirement already satisfied: html5lib>=1.1 in c:\users\owner\anaconda3\lib\site-packages (from yfinance) (1.1)
Requirement already satisfied: soupsieve>1.2 in c:\users\owner\anaconda3\lib\site-packages (from beautifulsoup4>=4.11.1->yfinance) (2.4)
Requirement already satisfied: six>=1.9 in c:\users\owner\anaconda3\lib\site-packages (from html5lib>=1.1->yfinance) (1.16.0)
Requirement already satisfied: webencodings in c:\users\owner\anaconda3\lib\site-packages (from html5lib>=1.1->yfinance) (0.5.1)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\users\owner\anaconda3\lib\site-packages (from pandas>=1.3.0->yfinance) (2.8.2)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\owner\anaconda3\lib\site-packages (from requests>=2.31->yfinance) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\owner\anaconda3\lib\site-packages (from requests>=2.31->yfinance) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\owner\anaconda3\lib\site-packages (from requests>=2.31->yfinance) (1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\owner\anaconda3\lib\site-packages (from requests>=2.31->yfinance) (2023.11.17)
Note: you may need to restart the kernel to use updated packages.
```

In [2]:
```
# import relevant libraries
import datetime
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import yfinance as yfin
from IPython.display import VimeoVideo
from scipy import stats
```

In [3]:
```
pip install fredapi
```

```
Requirement already satisfied: fredapi in c:\users\owner\anaconda3\lib\site-packages (0.5.2)Note: you may need to restart the kernel to use updated packages.

Requirement already satisfied: pandas in c:\users\owner\anaconda3\lib\site-packages (from fredapi) (1.5.3)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\users\owner\anaconda3\lib\site-packages (from pandas->fredapi) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\owner\anaconda3\lib\site-packages (from pandas->fredapi) (2022.7)
Requirement already satisfied: numpy>=1.21.0 in c:\users\owner\anaconda3\lib\site-packages (from pandas->fredapi) (1.24.3)
Requirement already satisfied: six>=1.5 in c:\users\owner\anaconda3\lib\site-packages (from python-dateutil>=2.8.1->pandas->fredapi) (1.16.0)
```

In [4]:
```
pip install pandas_datareader
```

```
Requirement already satisfied: pandas_datareader in c:\users\owner\anaconda3\lib\site-packages (0.10.0)
Requirement already satisfied: lxml in c:\users\owner\anaconda3\lib\site-packages (from pandas_datareader) (4.9.2)
Requirement already satisfied: pandas>=0.23 in c:\users\owner\anaconda3\lib\site-packages (from pandas_datareader) (1.5.3)
Requirement already satisfied: requests>=2.19.0 in c:\users\owner\anaconda3\lib\site-packages (from pandas_datareader) (2.31.0)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\users\owner\anaconda3\lib\site-packages (from pandas>=0.23->pandas_datareader) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\users\owner\anaconda3\lib\site-packages (from pandas>=0.23->pandas_datareader) (2022.7)
Requirement already satisfied: numpy>=1.21.0 in c:\users\owner\anaconda3\lib\site-packages (from pandas>=0.23->pandas_datareader) (1.24.3)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\owner\anaconda3\lib\site-packages (from requests>=2.19.0->pandas_datareader) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\owner\anaconda3\lib\site-packages (from requests>=2.19.0->pandas_datareader) (3.4)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\owner\anaconda3\lib\site-packages (from requests>=2.19.0->pandas_datareader) (1.26.16)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\owner\anaconda3\lib\site-packages (from requests>=2.19.0->pandas_datareader) (2023.11.17)
Requirement already satisfied: six>=1.5 in c:\users\owner\anaconda3\lib\site-packages (from python-dateutil>=2.8.1->pandas>=0.23->pandas_datareader) (1.16.0)
Note: you may need to restart the kernel to use updated packages.
```

3.1. Pull out 10-year dataset

To implement the exploratory statistics for this scenario, we aim to pull out 10-year data for the Average cost of construction of buildings, Producer Price Index (PPI) of the Erected Buildings which is indicative of their selling prices, as well as the Producer Price Index of the Construction Materials - which is indicative of the cost of construction materials over this period.

In [5]:
```
# Load in data from FRED Economics, drop null values
import pandas_datareader.data as web
import datetime
from fredapi import Fred

fred = Fred(api_key="4a686e7bf0f4f1b2a194e90961e4c4f9")

start = datetime.datetime(2014, 5, 1)
end = datetime.datetime(2024, 5, 31)
df = web.DataReader(["TTLHLTHCONS", "WPU801", "WPUSI012011"], "fred", start, end)
df = df.rename(columns={"TTLHLTHCONS": "Const_Cost","WPU801" : "PPI_Buildings", "WPUSI012011" : "PPI_Const_Mat." })
df.dropna()
```

Out[5]:

DATE	Const_Cost	PPI_Buildings	PPI_Const_Mat.
2014-05-01	38051.0	108.000	214.300
2014-06-01	38135.0	108.100	214.500
2014-07-01	38873.0	108.400	214.800
2014-08-01	38297.0	108.500	215.500
2014-09-01	38125.0	108.500	215.700
...	...	...	...
2023-12-01	67729.0	172.709	327.644
2024-01-01	68529.0	173.226	334.374
2024-02-01	68289.0	173.257	337.812
2024-03-01	68122.0	173.334	333.699
2024-04-01	66504.0	173.637	333.004

120 rows × 3 columns

In [6]:
```
df
```

Out[6]:

DATE	Const_Cost	PPI_Buildings	PPI_Const_Mat.
2014-05-01	38051.0	108.000	214.300
2014-06-01	38135.0	108.100	214.500
2014-07-01	38873.0	108.400	214.800
2014-08-01	38297.0	108.500	215.500
2014-09-01	38125.0	108.500	215.700
...	...	...	...
2024-01-01	68529.0	173.226	334.374
2024-02-01	68289.0	173.257	337.812
2024-03-01	68122.0	173.334	333.699
2024-04-01	66504.0	173.637	333.004
2024-05-01	NaN	173.564	332.640

121 rows × 3 columns

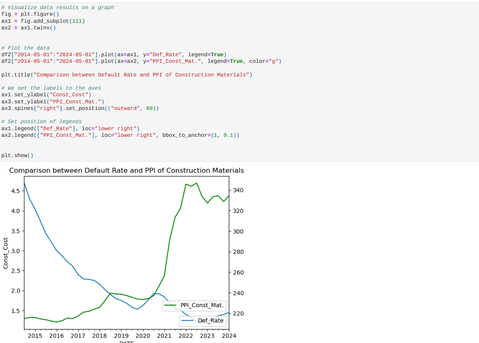
In [7]:
```
df.describe()
```

Out[7]:

	Const_Cost	PPI_Buildings	PPI_Const_Mat.
count	120.000000	121.000000	121.000000
mean	47815.433333	131.153306	259.761074
std	8130.826772	22.995493	50.416306
min	38051.000000	108.000000	210.900000
25%	41214.500000	112.100000	215.900000
50%	45183.000000	124.700000	236.300000
75%	51293.000000	143.811000	326.449000
max	68529.000000	175.455000	353.015000

3.2. Visualize data

In [8]:
```
# Visualize data results on a graph
fig = plt.figure()
ax1 = fig.add_subplot(111)
ax2 = ax1.twinx()
ax3 = ax1.twinx()

# Plot the data
df["2014-05-01":"2024-05-01"].plot(ax=ax1, y="Const_Cost", legend=True)
df["2014-05-01":"2024-05-01"].plot(ax=ax2, y="PPI_Buildings", legend=True, color="g")
df["2014-05-01":"2024-05-01"].plot(ax=ax3, y="PPI_Const_Mat.", legend=True, color="r")
plt.title("Comparison between Construction Costs, PPI of Buildings, PPI of Construction Materials")

# We set the labels to the axes
ax1.set_ylabel("Const_Cost")
ax2.set_ylabel("PPI_Buildings")
ax3.set_ylabel("PPI_Const_Mat.")
ax3.spines["right"].set_position(("outward", 60))

# Set position of legends
ax1.legend(["Const_Cost"], loc="lower right")
ax2.legend(["PPI_Buildings"], loc="lower right", bbox_to_anchor=(1, 0.1))
ax3.legend(["PPI_Const_Mat."], loc="lower right", bbox_to_anchor=(0.7, 0))

plt.show()
```



The figure above depicts that PPI index for construction materials & final building structure as well as Construction costs spiked after 2020. While each of these factors are dependent on one another, it is important to introduce a different factor like the default Rate on Real Estate Loans into the picture.

3.3 Default Rate on Real Estate Loans vs PPI for Construction Materials

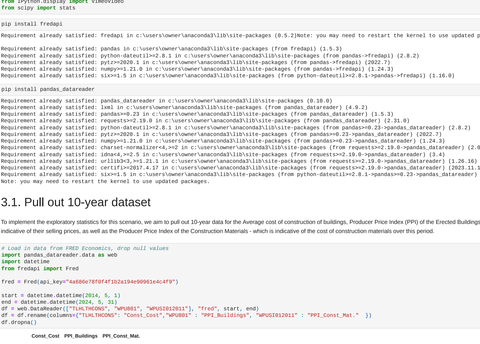
In [9]:
```
# Load dataset within the same time period.
start = datetime.datetime(2014, 5, 1)
end = datetime.datetime(2024, 5, 31)
df2 = web.DataReader(["DRSREACBS", "WPUSI012011"], "fred", start, end)
df2 = df2.rename(columns={"DRSREACBS" : "Def_Rate", "WPUSI012011" : "PPI_Const_Mat." })
df2.dropna(inplace=True)
```

In [10]:
```
df2
```

Out[10]:

DATE	Def_Rate	PPI_Const_Mat.
2014-07-01	4.69	214.800
2014-10-01	4.29	215.900
2015-01-01	4.04	215.800
2015-04-01	3.74	214.500
2015-07-01	3.43	213.700
2015-10-01	3.22	212.400
2016-01-01	3.00	211.400
2016-04-01	2.87	212.700
2016-07-01	2.72	215.300
2016-10-01	2.61	214.800
2017-01-01	2.40	217.300
2017-04-01	2.29	221.000
2017-07-01	2.28	221.900
2017-10-01	2.25	223.900
2018-01-01	2.16	225.800
2018-04-01	2.02	232.400
2018-07-01	1.90	239.600
2018-10-01	1.80	238.700
2019-01-01	1.75	238.500
2019-04-01	1.68	237.200
2019-07-01	1.58	235.600
2019-10-01	1.53	233.800
2020-01-01	1.62	233.400
2020-04-01	1.76	234.300
2020-07-01	1.92	237.000
2020-10-01	1.92	246.400
2021-01-01	1.84	256.400
2021-04-01	1.69	291.800
2021-07-01	1.55	313.542
2021-10-01	1.51	322.120
2022-01-01	1.40	345.742
2022-04-01	1.33	343.786
2022-07-01	1.21	346.790
2022-10-01	1.21	333.796
2023-01-01	1.23	327.338
2023-04-01	1.27	333.366
2023-07-01	1.37	334.512
2023-10-01	1.40	328.743
2024-01-01	1.45	334.374

In [11]:
```
# Visualize data results on a graph
fig = plt.figure()
ax1 = fig.add_subplot(111)
ax2 = ax1.twinx()


# Plot the data
df2["2014-05-01":"2024-05-01"].plot(ax=ax1, y="Def_Rate", legend=True)
df2["2014-05-01":"2024-05-01"].plot(ax=ax2, y="PPI_Const_Mat.", legend=True, color="g")

plt.title("Comparison between Default Rate and PPI of Construction Materials")

# We set the labels to the axes
ax1.set_ylabel("Const_Cost")
ax2.set_ylabel("PPI_Const_Mat.")
ax3.spines["right"].set_position(("outward", 60))

# Set position of legends
ax1.legend(["Def_Rate"], loc="lower right")
ax2.legend(["PPI_Const_Mat."], loc="lower right", bbox_to_anchor=(1, 0.1))


plt.show()
```



After the 2008 recession in the US, the default rate declined drastically for construction firms. Interestingly, after the small spike in 2020 (most likely because of the COVID lockdown), the interest rate reduced further while the selling price of construction materials skyrocketed because of the high inflation rate. In years 2021 & 2022, there was a minor decline in the price of construction materials and a corresponding increase in the default rate.

This surge in the price of buildings within the aforementioned period is mainly a result of the prices of construction materials (Intelligence Lab, 2023) as well as the urge in insurance premiums for construction firms and homebuilders (NAIOP, 2023). The fact that The Intelligence Lab considers the construction industry in the UK market while NAIOP considers the construction industry in the US is an indication that the inflation rate has significantly impacted the industry across continents.

In [14]:
```
#Check correlation between Default Rate and PPI of Construction materials
correlation = df2['Def_Rate'].corr(df2['PPI_Const_Mat.'])

correlation.round(3)
```

Out[14]:
```
-0.688
```

A negative of -0.688 correlation is an indication that as PPI of construction materials increases, the default rate declines.

3.4. References

"What is happening to build and labour costs? Plus, rising inflation and a few reasons to be bullish/bearish", Knight Frank: The Intelligence Lab, Global Property Insights, 29 June 2023, www.knightfrank.com/research/article/2023-06-29-what-is-happening-to-build-and-labour-costs.

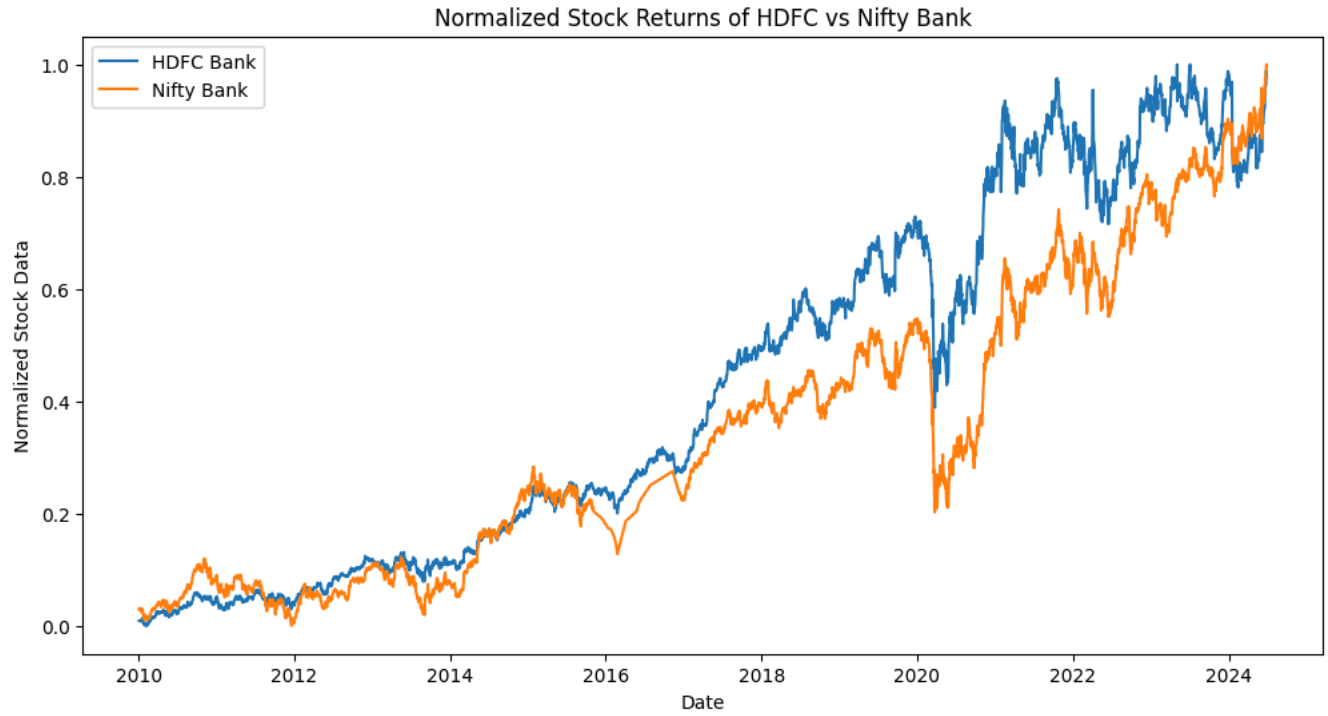
"How Rising Construction Costs are Impacting Real Estate Development". NAIOP Commercial Development Real Estate Association, 5 December 2023. https://blog.naiop.org/2023/12/how-rising-construction-costs-are-impacting-real-estate-development/.

```
# Imports
import pandas_datareader as pdr
import matplotlib.pyplot as plt
import yfinance as yf
import pandas as pd
import numpy as np
```

4. Publicly traded Equity (e.g. common stock) – that is, securities lending of a stock.

```
# Get the data
hdfc_bank = yf.download('HDFCBANK.NS', start='2010-01-01')
nifty_bank = yf.download('^NSEBANK', start='2010-01-01')

# Normalize
hdfc_bank_norm = (hdfc_bank['Close'] - hdfc_bank['Close'].min()) / (hdfc_bank['Close'].max() - hdfc_bank['Close'].min())
nifty_bank_norm = (nifty_bank['Close'] - nifty_bank['Close'].min()) / (nifty_bank['Close'].max() - nifty_bank['Close'].min())

# Plot
plt.figure(figsize=(12, 6))

plt.plot(hdfc_bank_norm.index, hdfc_bank_norm, label='HDFC Bank')
plt.plot(nifty_bank_norm.index, nifty_bank_norm, label='Nifty Bank')
plt.legend()
plt.xlabel("Date")
plt.ylabel("Normalized Stock Data")
plt.title("Normalized Stock Returns of HDFC vs Nifty Bank")
plt.show()

print(f"Correlation between HDFC Bank and Nifty Bank: {hdfc_bank_norm.corr(nifty_bank_norm)}")
```

```
[*********************100%%**********************]  1 of 1 completed
[*********************100%%**********************]  1 of 1 completed
```

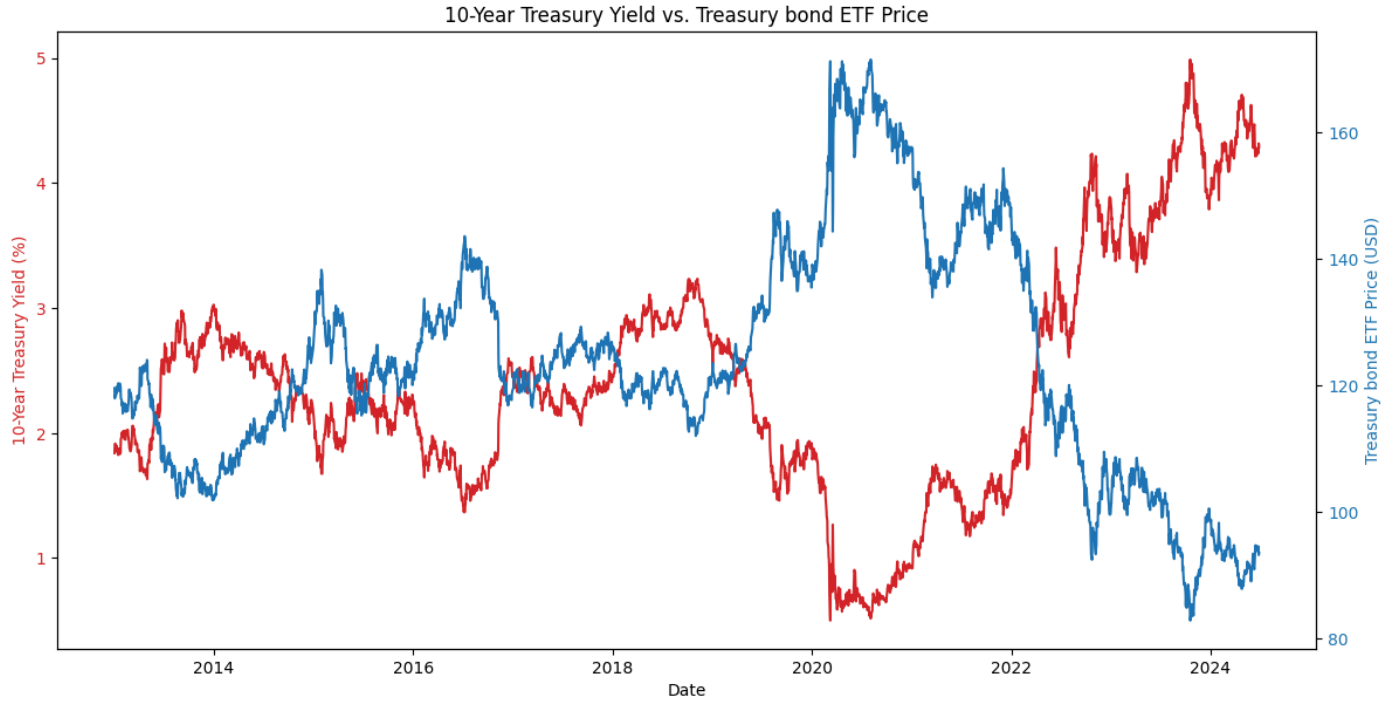



```
Correlation between HDFC Bank and Nifty Bank: 0.9683302222867746
```

5. Publicly traded bond (e.g. treasury bond, corporate bond) – that is, securities lending of a bond.

```
tnx_data = yf.download("^TNX", start="2013-01-01")["Close"] # 10-Year Treasury Note yield data
tlt_data = yf.download("TLT", start="2013-01-01")["Close"] # Treasury Bond ETF (TLT) data
data = pd.DataFrame({"Yield": tnx_data, "Price": tlt_data}).dropna()

correlation = data["Yield"].corr(data["Price"])
print(f"Correlation between 10-Year Treasury Yield and TLT Price: {correlation}")

#Plot
fig, ax1 = plt.subplots(figsize=(12, 6))

color = 'tab:red'
ax1.set_xlabel('Date')
ax1.set_ylabel('10-Year Treasury Yield (%)', color=color)
ax1.plot(data.index, data["Yield"], color=color)
ax1.tick_params(axis='y', labelcolor=color)

ax2 = ax1.twinx()
color = 'tab:blue'
ax2.set_ylabel('Treasury bond ETF Price (USD)', color=color)
ax2.plot(data.index, data["Price"], color=color)
ax2.tick_params(axis='y', labelcolor=color)

fig.tight_layout()
plt.title('10-Year Treasury Yield vs. Treasury bond ETF Price')
plt.show()
```

```
[**********100%%**********]  1 of 1 completed
[**********100%%**********]  1 of 1 completed
Correlation between 10-Year Treasury Yield and TLT Price: -0.9373670775404007
```

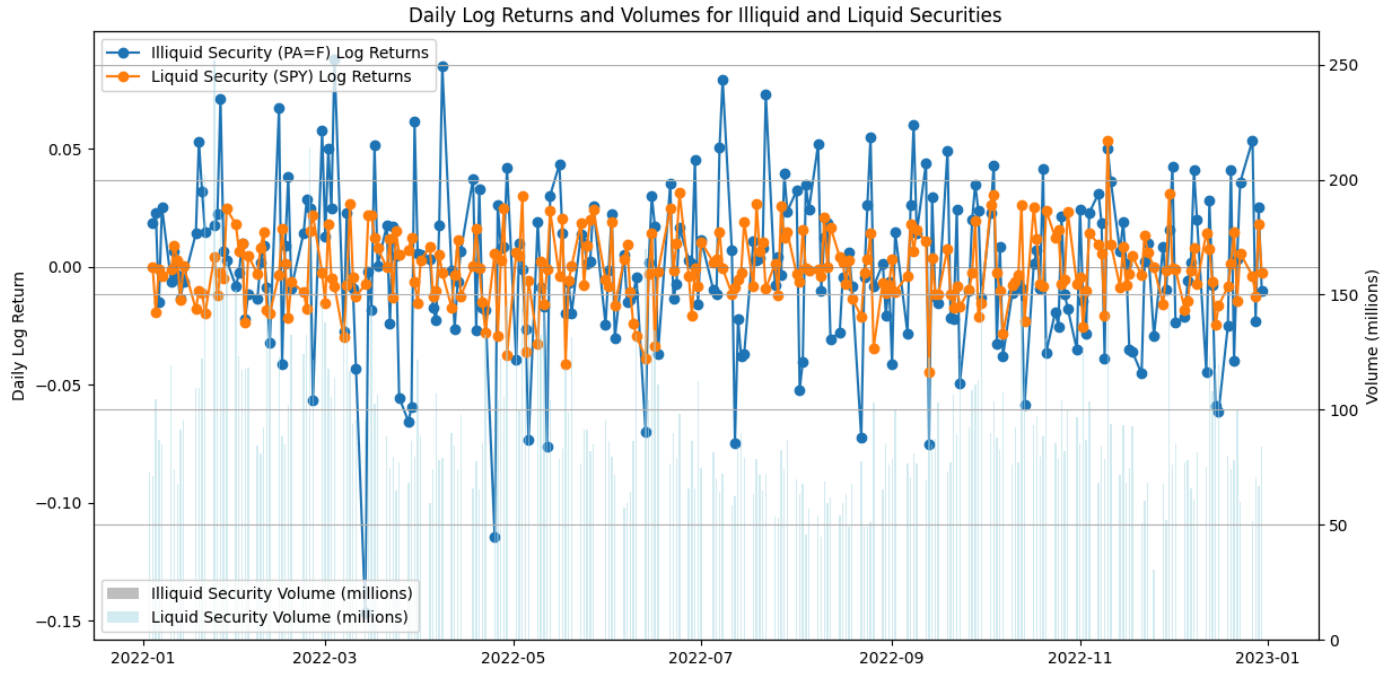


6. An illiquid security - Rare metal

```
illiquid_ticker = "PA=F" #'PA=F'  #
liquid_ticker = 'SPY'   # SPDR S&P 500 ETF

start_date = '2022-01-01'
end_date = '2023-01-01'

illiquid_data = yf.download(illiquid_ticker, start=start_date, end=end_date)
liquid_data = yf.download(liquid_ticker, start=start_date, end=end_date)

# log returns
```

```
illiquid_data['Log_Returns'] = np.log(illiquid_data['Close'] / illiquid_data['Close'].shift(1)).dropna()
liquid_data['Log_Returns'] = np.log(liquid_data['Close'] / liquid_data['Close'].shift(1)).dropna()

# Plot
fig, ax1 = plt.subplots(figsize=(14, 7))

ax1.plot(illiquid_data.index, illiquid_data['Log_Returns'], label='Illiquid Security ({}) Log Returns'.format(illiquid_ticker), marker='o')
ax1.plot(liquid_data.index, liquid_data['Log_Returns'], label='Liquid Security (SPY) Log Returns', marker='o')
ax1.set_ylabel('Daily Log Return')
ax1.legend(loc='upper left')

ax2 = ax1.twinx()
ax2.bar(illiquid_data.index, illiquid_data['Volume'] / 1000000, width=0.5, color='grey', alpha=0.5, label='Illiquid Security Volume (millions
ax2.bar(liquid_data.index, liquid_data['Volume'] / 1000000, width=0.5, color='lightblue', alpha=0.5, label='Liquid Security Volume (millions)
ax2.set_ylabel('Volume (millions)')
ax2.legend(loc='lower left')

ax1.axhline(0, color='grey', linewidth=0.5)

plt.title('Daily Log Returns and Volumes for Illiquid and Liquid Securities')
plt.xlabel('Date')
plt.grid(True)
plt.show()


#Vols and volumes
volatility_illiquid = illiquid_data['Log_Returns'].std()
volatility_liquid = liquid_data['Log_Returns'].std()
average_volume_illiquid = illiquid_data['Volume'].mean()
average_volume_liquid = liquid_data['Volume'].mean() / 1000000

print(f'Volatility of Illiquid Security {illiquid_ticker}: {volatility_illiquid}')
print(f'Volatility of Liquid Security (SPY): {volatility_liquid}')
print(f'Average Volume of Illiquid Security ({illiquid_ticker}): {average_volume_illiquid:.2f}  shares')
print(f'Average Volume of Liquid Security (SPY): {average_volume_liquid:.2f} million shares')
```

```
[**********100%%**********]  1 of 1 completed
[**********100%%**********]  1 of 1 completed
```




```
Volatility of Illiquid Security PA=F: 0.03318606924039639
Volatility of Liquid Security (SPY): 0.015323085904239385
Average Volume of Illiquid Security (PA=F): 45.56  shares
Average Volume of Liquid Security (SPY): 94.76 million shares
```