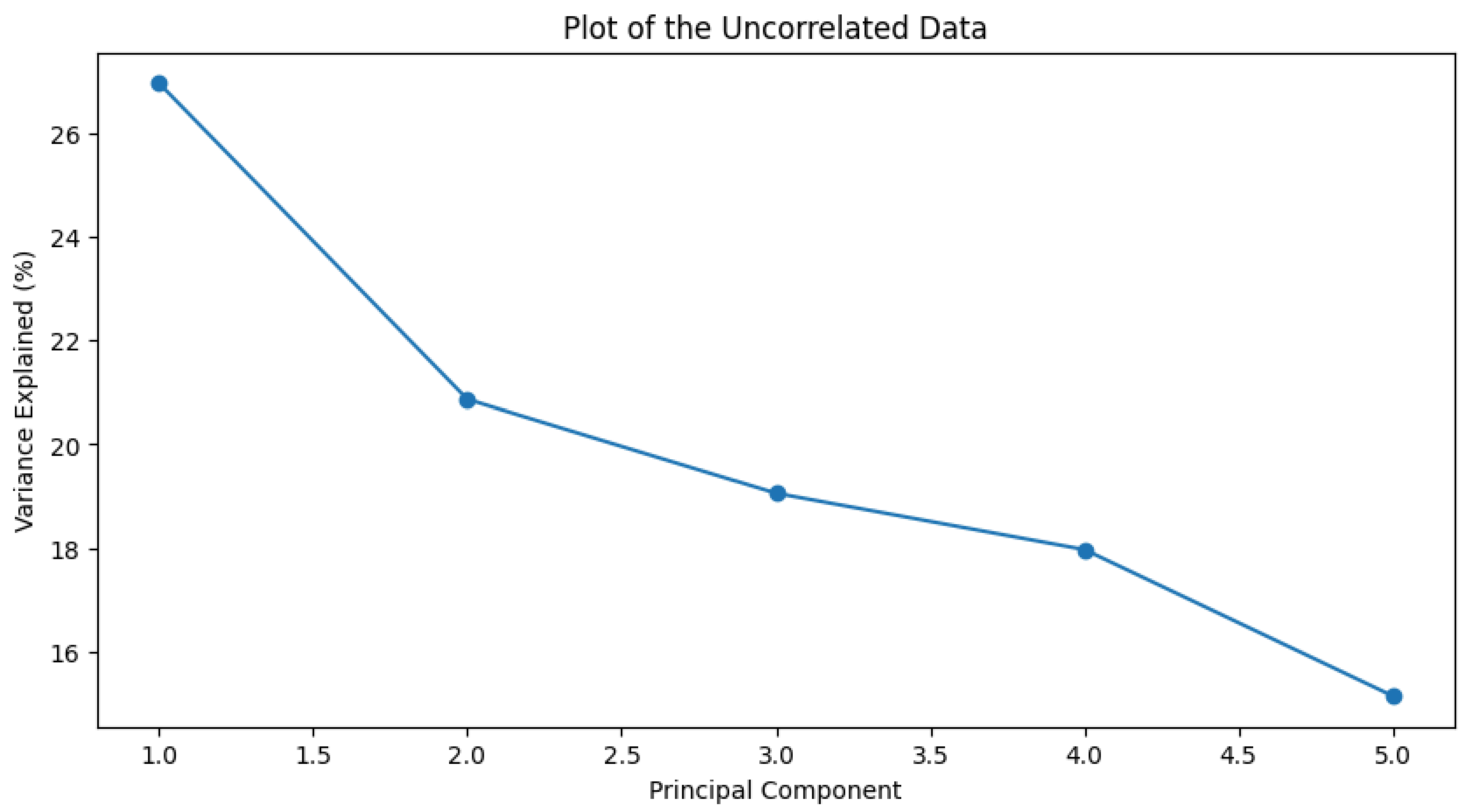
MScFE 600 FINANCIAL DATA \ **Group Work Project #1** \ **Group Number 11231 ** \ \

TASK 3

In [1]:
```
 # 3 a-d : Simulate 5 uncorrelated Gaussian variables a
import numpy as np
import pandas as pd
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt

np.random.seed(42)
days = 120
mean, std_dev = 0, 0.01

# a-Generate 5 uncorrelated Gaussian random variables
data = np.random.normal(mean, std_dev, (days, 5))
df_sim = pd.DataFrame(data, columns=[f'Y{i+1}' for i in range(5)])

# b- PCA using the covariance matrix
pca_sim = PCA(n_components=5)
pca_sim.fit(df_sim)

# c-explained variance ratios
explained_variance_ratio = pca_sim.explained_variance_ratio_
print("Explained Variance")
for i, r in enumerate(explained_variance_ratio, 1):
    print(f"Component {i}: {r:.4f} ({r*100:.2f}%)")

# d-plot of variance explained
plt.figure(figsize=(10,5))
plt.plot(range(1,6), explained_variance_ratio*100, 'o-')
plt.title("Plot of the Uncorrelated Data")
plt.xlabel("Principal Component")
plt.ylabel("Variance Explained (%)")
plt.show()
```

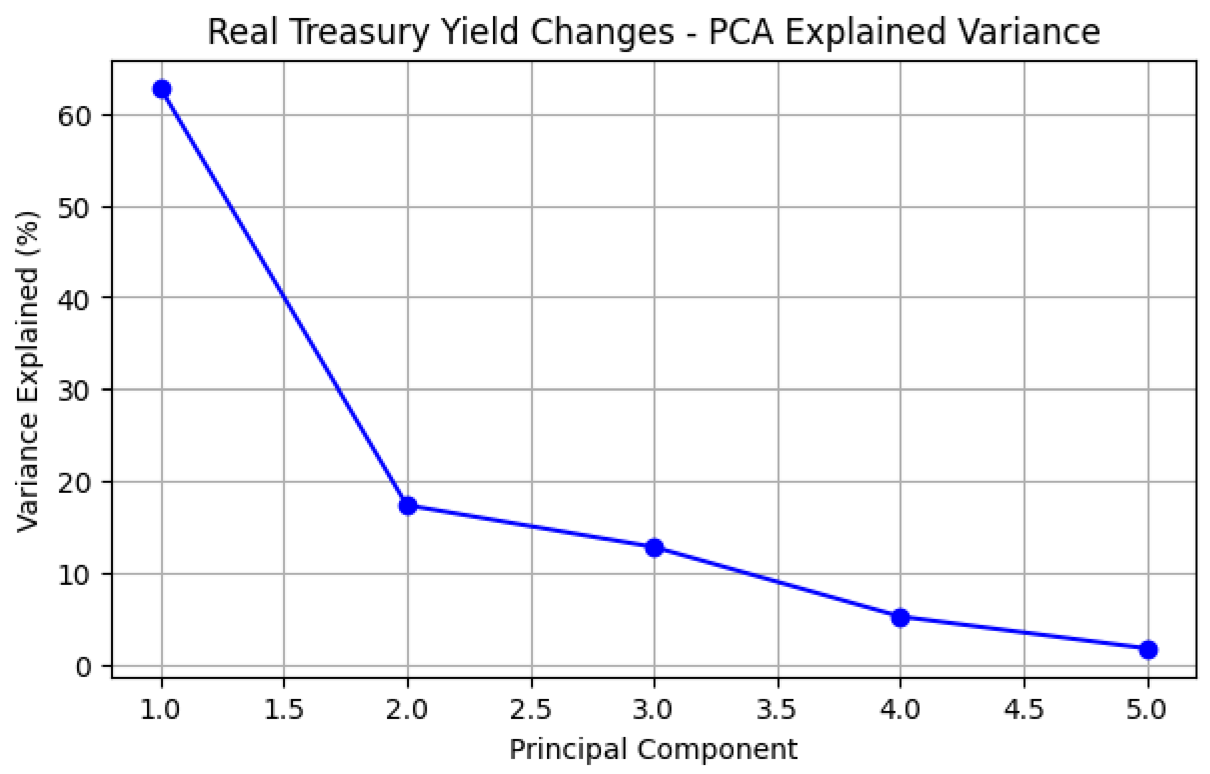
```
Explained Variance
Component 1: 0.2698 (26.98%)
Component 2: 0.2087 (20.87%)
Component 3: 0.1906 (19.06%)
Component 4: 0.1797 (17.97%)
Component 5: 0.1513 (15.13%)
```



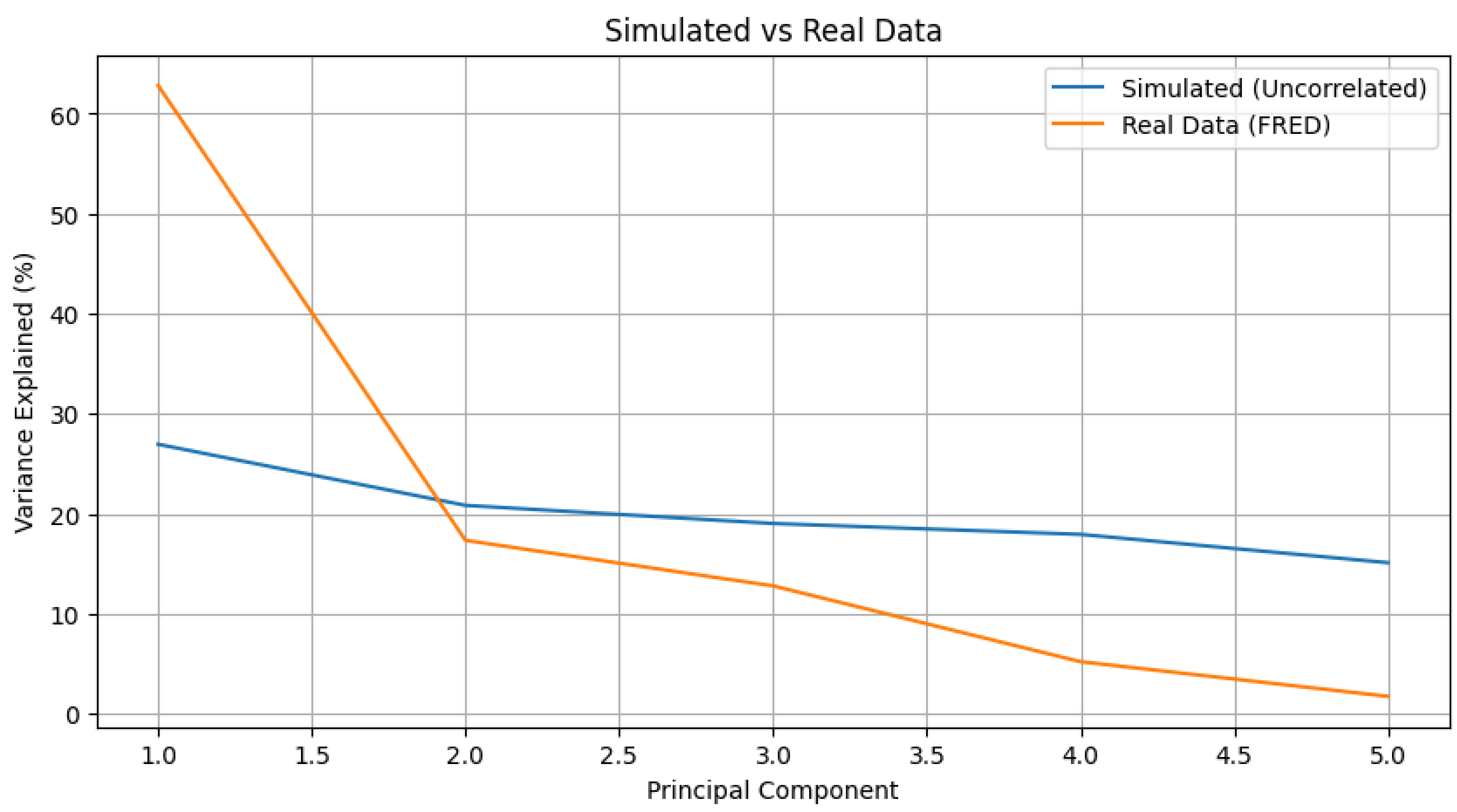
In [2]:
```
import pandas as pd
from pandas_datareader import data as web
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA

# 3.e-j Real yield data from FRED 'PCA'
# e- Collect 5 Treasury yields
start = '2025-04-08'
end   = '2025-10-08'
series = ['DGS1MO','DGS3MO','DGS6MO','DGS1','DGS2']

# f- Load data
data = {}
for s in series:
    try:
        df_s = web.DataReader(s, 'fred', start, end)
        if not df_s.empty:
            data[s] = df_s
            print(f"{s} loaded successfully, {df_s.shape[0]} rows")
        else:
            print(f"{s} is empty")
    except Exception as e:
        print(f"Error loading {s}: {e}")

# Check
loaded_series = [s for s in series if s in data]
if not loaded_series:
    raise ValueError("No series were successfully loaded. Check your series names or internet connection.")

# g- Concatenate loaded series
df = pd.concat([data[s] for s in loaded_series], axis=1)
df.columns = loaded_series
df = df.dropna()
print("\nSample of raw yield data:")
print(df.head())

# h- Compute daily yield changes
df_changes = df.diff().dropna()
print("\nSample of daily yield changes:")
print(df_changes.head())

# i- PCA using correlation matrix (standardized data)
scaler = StandardScaler()
X = scaler.fit_transform(df_changes.values)

pca_real = PCA(n_components=X.shape[1])
pca_real.fit(X)

# j- Explained variance
```

```python
ratios_real = pca_real.explained_variance_ratio_
print("\nExplained Variance Real Data")
for i, r in enumerate(ratios_real, 1):
    print(f"Component {i}: {r:.4f} ({r*100:.2f}%)")

# k- Plot of real data
plt.figure(figsize=(7,4))
plt.plot(range(1, len(ratios_real)+1), ratios_real*100, 'o-', color='blue')
plt.title("Real Treasury Yield Changes - PCA Explained Variance")
plt.xlabel("Principal Component")
plt.ylabel("Variance Explained (%)")
plt.grid(True)
plt.show()
```

```
DGS1MO loaded successfully, 131 rows
DGS3MO loaded successfully, 131 rows
DGS6MO loaded successfully, 131 rows
DGS1 loaded successfully, 131 rows
DGS2 loaded successfully, 131 rows

Sample of raw yield data:
            DGS1MO  DGS3MO  DGS6MO  DGS1  DGS2
DATE
2025-04-08    4.36    4.31    4.14  3.83  3.71
2025-04-09    4.36    4.35    4.23  4.03  3.91
2025-04-10    4.36    4.34    4.17  3.97  3.84
2025-04-11    4.37    4.34    4.21  4.04  3.96
2025-04-14    4.34    4.33    4.21  3.99  3.84

Sample of daily yield changes:
            DGS1MO  DGS3MO  DGS6MO  DGS1  DGS2
DATE
2025-04-09    0.00    0.04    0.09  0.20  0.20
2025-04-10    0.00   -0.01   -0.06 -0.06 -0.07
2025-04-11    0.01    0.00    0.04  0.07  0.12
2025-04-14   -0.03   -0.01    0.00 -0.05 -0.12
2025-04-15    0.01    0.00    0.00  0.00  0.00

Explained Variance Real Data
Component 1: 0.6283 (62.83%)
Component 2: 0.1737 (17.37%)
Component 3: 0.1281 (12.81%)
Component 4: 0.0523 (5.23%)
Component 5: 0.0176 (1.76%)
```



In [3]:
```python
# (j) Comparison: simulated vs real screeplots
plt.figure(figsize=(10,5))
plt.plot(range(1,6), explained_variance_ratio*100, label='Simulated (Uncorrelated)')
plt.plot(range(1, len(ratios_real)+1), ratios_real*100, label='Real Data (FRED)')
plt.title("Simulated vs Real Data")
plt.xlabel("Principal Component")
plt.ylabel("Variance Explained (%)")
plt.legend()
plt.grid(True)
plt.show()
```



TASK 4

In [4]:
```python
  # Import libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import datetime
import math
from numpy import linalg as LA
import yfinance as yf

# List of the 30 largest holdings of XLRE
top30_xlre = [
    "PLD", "AMT", "EQIX", "WELL", "PSA", "SPG", "O", "AVB", "EQR", "INVH",
    "EXR", "MAA", "CPT", "UDR", "ESS", "ARE", "VTR", "REG", "NHI", "FRT",
    "BXP", "VNO", "SLG", "HIW", "CUZ", "PDM", "CIO", "AHH", "PSTL", "ILPT"
]
```

```python
# Define the study period: around the last 6 months
start = "2025-03-06"
end = "2025-09-06"


# Download historical price data from Yahoo Finance
data = yf.download(top30_xlre, start=start, end=end)
data = data['Close']

# datetime format
data.index = pd.to_datetime(data.index)

# the first few rows
data.head()
```

```
/tmp/ipython-input-2964641385.py:24: FutureWarning: YF.download() has changed argument auto_adjust default to True
  data = yf.download(top30_xlre, start=start, end=end)
[*********************100%***********************]  30 of 30 completed
```

Out[4]:

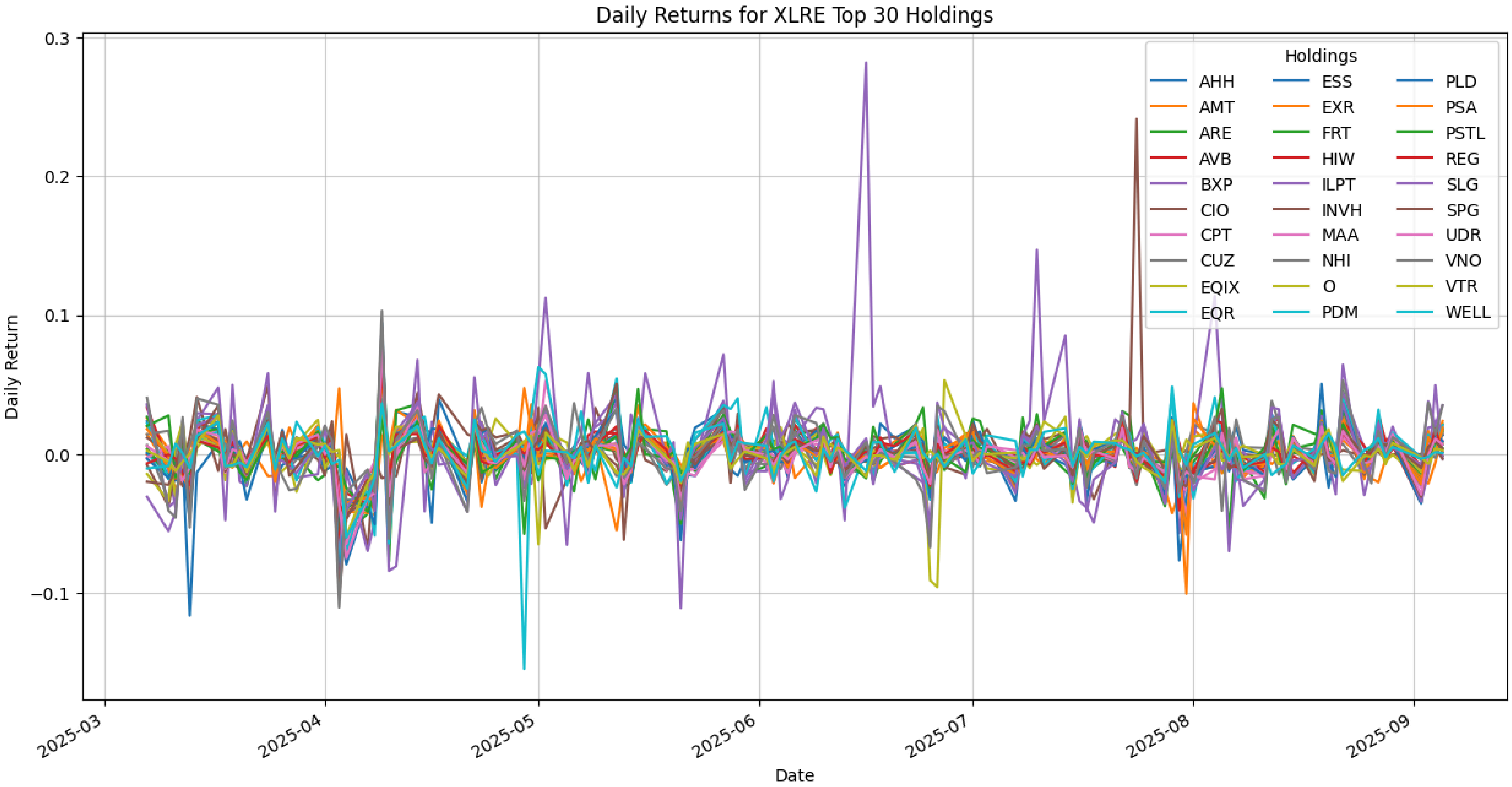
Ticker	AHH	AMT	ARE	AVB	BXP	CIO	CPT	CUZ	EQIX	EQR	...	PLD	PSA	PSTL	REG	SLG
Date																
2025-03-06	8.351662	201.859207	95.140053	212.911575	64.164474	4.938145	118.480186	28.071596	861.972839	69.747505	...	117.066917	299.225830	13.235419	72.308441	57.963535
2025-03-07	8.549836	207.087860	97.704453	212.862869	66.406906	4.899715	118.159233	28.255451	849.733826	69.805702	...	116.687393	304.610260	13.505529	71.832535	60.047157
2025-03-10	8.398845	207.848755	97.399391	210.836060	65.608101	4.717177	118.684425	27.345858	821.172729	68.748466	...	114.702225	304.241577	13.881755	71.813118	57.760014
2025-03-11	8.238418	204.639374	94.634796	207.493790	63.105808	4.659534	117.176926	27.297474	826.718811	67.681549	...	114.030769	299.012390	13.650230	71.589729	55.821758
2025-03-12	8.210108	201.625092	93.919807	204.667953	63.606262	4.621105	115.153961	27.974831	845.255371	66.895920	...	112.619743	294.394470	13.775640	70.604210	55.802383

5 rows × 30 columns

In [5]:

```python
#  Compute daily returns
daily_returns_holding= data.pct_change().dropna()

#Display the first few rows of daily returns
print(daily_returns_holding.head())

# Plot the daily returns for each holding
plt.figure(figsize=(14, 7))
daily_returns_holding.plot(figsize=(15, 8), title="Daily Returns for XLRE Top 30 Holdings")
plt.xlabel('Date')
plt.ylabel('Daily Return')
plt.legend(title='Holdings', loc='upper right', ncol=3)
plt.grid(True, linestyle='-', alpha=0.6)
plt.show()
```

```
Ticker           AHH       AMT       ARE       AVB       BXP       CIO  \
Date
2025-03-07  0.023729  0.025902  0.026954 -0.000229  0.034948 -0.007782
2025-03-10 -0.017660  0.003674 -0.003122 -0.009522 -0.012029 -0.037255
2025-03-11 -0.019101 -0.015441 -0.028384 -0.015852 -0.038140 -0.012220
2025-03-12 -0.003436 -0.014730 -0.007555 -0.013619  0.007930 -0.008247
2025-03-13 -0.116092  0.005225 -0.018981 -0.010379 -0.032077  0.012474

Ticker           CPT       CUZ      EQIX       EQR  ...       PLD       PSA  \
Date                                                ...
2025-03-07 -0.002709  0.006550 -0.014199  0.000834  ... -0.003242  0.017995
2025-03-10  0.004445 -0.032192 -0.033612 -0.015145  ... -0.017013 -0.001210
2025-03-11 -0.012702 -0.001769  0.006754 -0.015519  ... -0.005854 -0.017188
2025-03-12 -0.017264  0.024814  0.022422 -0.011608  ... -0.012374 -0.015444
2025-03-13 -0.013936 -0.029402 -0.024609 -0.010004  ... -0.046401 -0.025162

Ticker          PSTL       REG       SLG       SPG       UDR       VNO  \
Date
2025-03-07  0.020408 -0.006582  0.035947 -0.019709  0.005880  0.040576
2025-03-10  0.027857 -0.000270 -0.038089 -0.021953 -0.004721 -0.040503
2025-03-11 -0.016678 -0.003111 -0.033557 -0.010950 -0.011069 -0.045621
2025-03-12  0.009187 -0.013766 -0.000347  0.014375 -0.018501  0.026648
2025-03-13 -0.014006 -0.022361 -0.034734 -0.038292 -0.009774 -0.052716

Ticker           VTR      WELL
Date
2025-03-07  0.002659 -0.010203
2025-03-10 -0.005304 -0.008477
2025-03-11 -0.011998  0.007455
2025-03-12 -0.005247  0.000611
2025-03-13  0.000151 -0.010110

[5 rows x 30 columns]
<Figure size 1400x700 with 0 Axes>
```

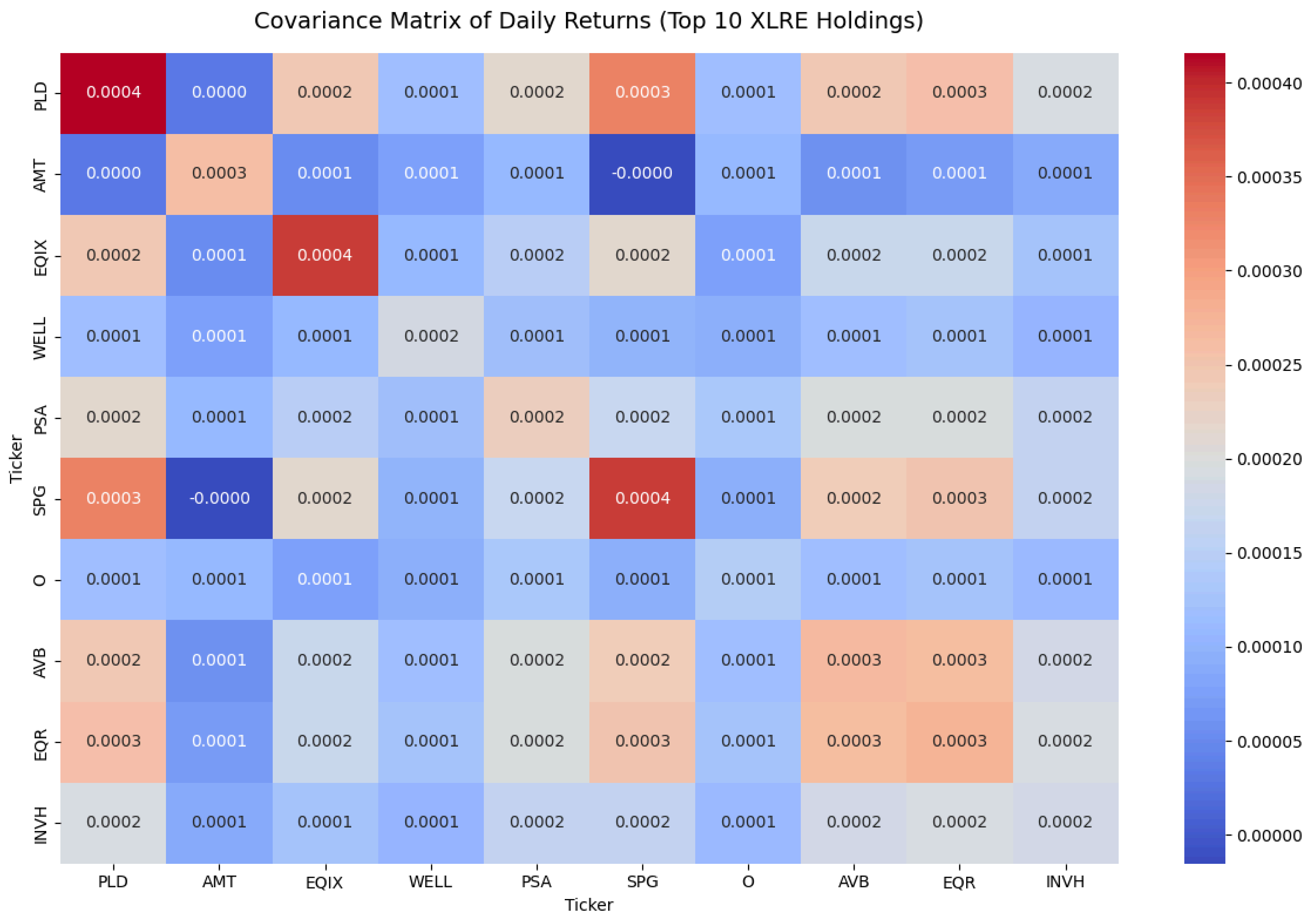


In [6]:
```python
# The top 10 holdings
top10 = top30_xlre[:10]

# Computing the covariance matrix for top 10 holdings
cov_matrix= daily_returns_holding[top10].cov()

# The covariance matrix for top 10
cov_matrix

# Compute the correlation matrix for top 5 holdings
corr_matrix = daily_returns_holding[top10].corr()

# Display it
corr_matrix
```

Out[6]:

Ticker	PLD	AMT	EQIX	WELL	PSA	SPG	O	AVB	EQR	INVH
Ticker										
PLD	1.000000	0.101283	0.605551	0.424676	0.691017	0.821068	0.484271	0.734894	0.756247	0.697285
AMT	0.101283	1.000000	0.164749	0.346054	0.440801	-0.048162	0.556743	0.225674	0.261419	0.410946
EQIX	0.605551	0.164749	1.000000	0.394909	0.500300	0.554806	0.333058	0.534317	0.525170	0.473420
WELL	0.424676	0.346054	0.394909	1.000000	0.574451	0.371573	0.589513	0.523033	0.544720	0.566819
PSA	0.691017	0.440801	0.500300	0.574451	1.000000	0.564790	0.727423	0.788243	0.768522	0.785303
SPG	0.821068	-0.048162	0.554806	0.371573	0.564790	1.000000	0.391622	0.744837	0.765099	0.620933
O	0.484271	0.556743	0.333058	0.589513	0.727423	0.391622	1.000000	0.596374	0.620673	0.687108
AVB	0.734894	0.225674	0.534317	0.523033	0.788243	0.744837	0.596374	1.000000	0.967359	0.826959
EQR	0.756247	0.261419	0.525170	0.544720	0.768522	0.765099	0.620673	0.967359	1.000000	0.853674
INVH	0.697285	0.410946	0.473420	0.566819	0.785303	0.620933	0.687108	0.826959	0.853674	1.000000

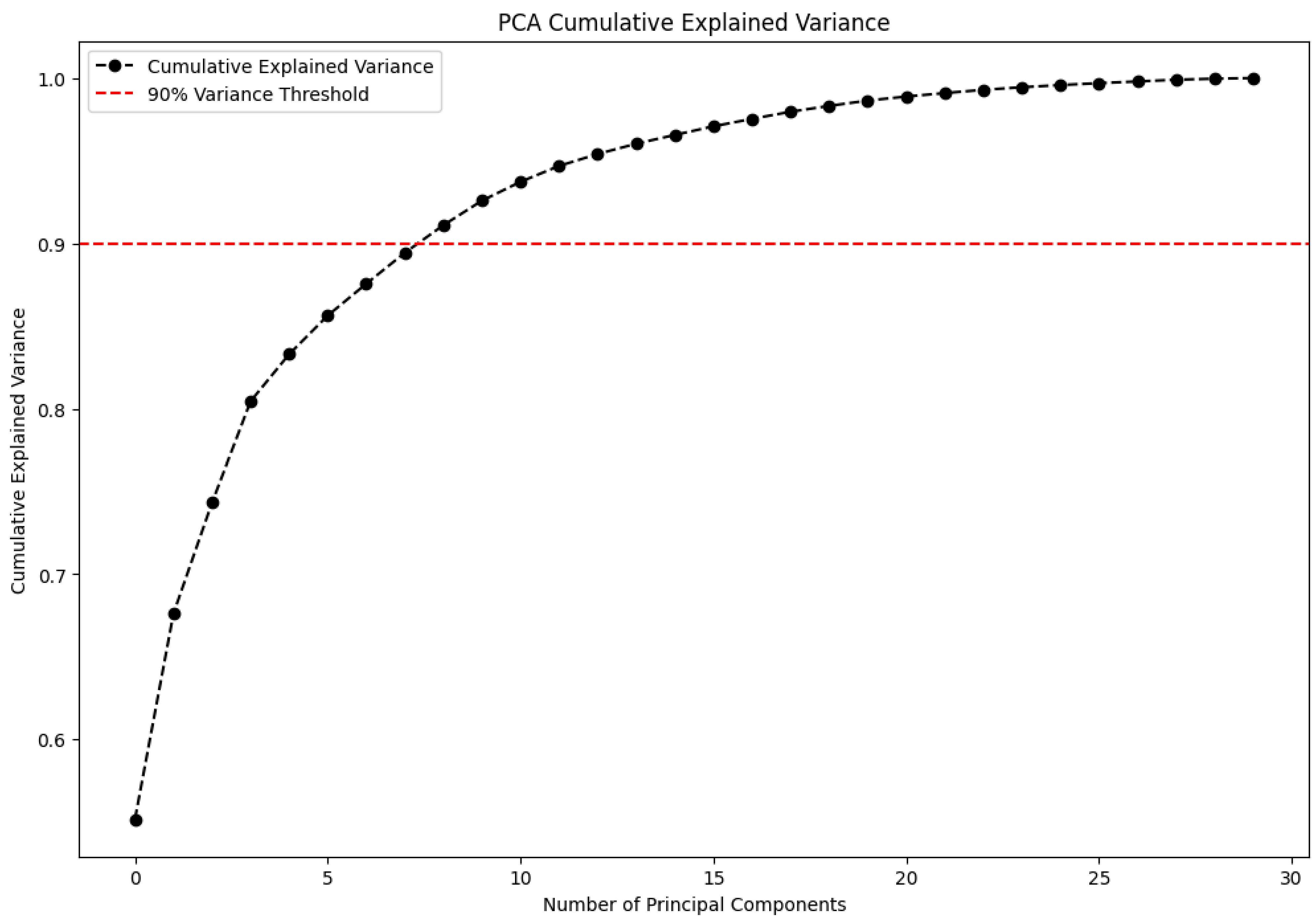
In [7]:
```python
# Plot the Covariance Matrix Heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(
    cov_matrix,
    annot=True,
    cmap='coolwarm',
    fmt='.4f',
    xticklabels=top10,
    yticklabels=top10
)
plt.title('Covariance Matrix of Daily Returns (Top 10 XLRE Holdings)', fontsize=14, pad=15)
plt.tight_layout()
plt.show()
```



In [8]:

```
from sklearn.decomposition import PCA

# Fit PCA on daily returns
pca = PCA()
pca.fit(daily_returns_holding)

# Explained variance ratio
explained_var_ratio = pca.explained_variance_ratio_

# Compute cumulative explained variance
cum_var_pca = explained_var_ratio.cumsum()

for i, (var, cum_var) in enumerate(zip(explained_var_ratio, cum_var_pca), start=1):
    print(f"PC{i}: Explained Variance = {var:.4f}, Cumulative = {cum_var:.4f}")

plt.figure(figsize=(12,8))
plt.plot(cum_var_pca, marker='o', linestyle='--', color='Black', label='Cumulative Explained Variance')
plt.axhline(y=0.9, color='r', linestyle='--', label='90% Variance Threshold')
plt.title('PCA Cumulative Explained Variance')
plt.xlabel('Number of Principal Components')
plt.ylabel('Cumulative Explained Variance')
plt.legend()
plt.show()
```

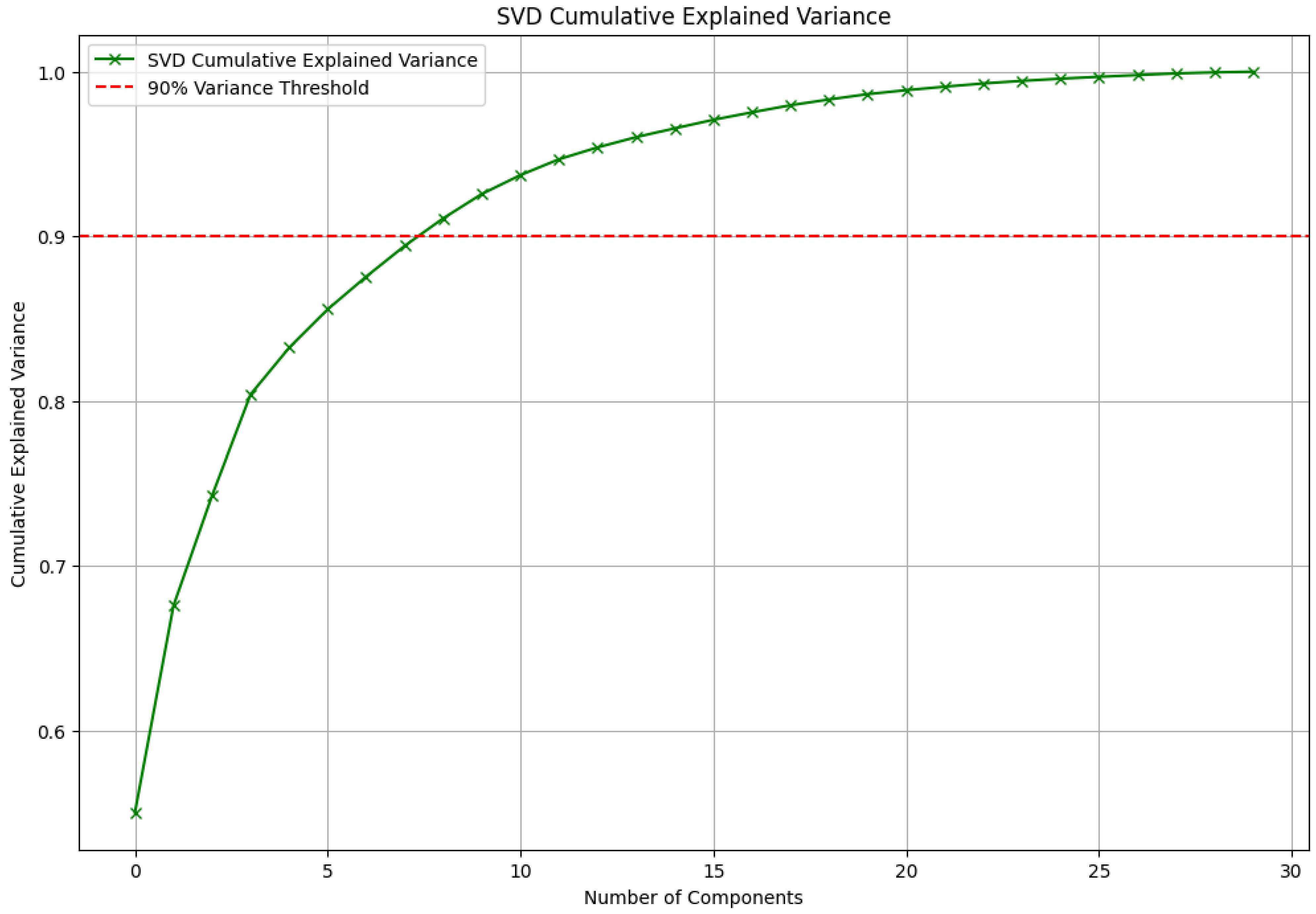
```
PC1: Explained Variance = 0.5516, Cumulative = 0.5516
PC2: Explained Variance = 0.1246, Cumulative = 0.6762
PC3: Explained Variance = 0.0670, Cumulative = 0.7432
PC4: Explained Variance = 0.0612, Cumulative = 0.8044
PC5: Explained Variance = 0.0285, Cumulative = 0.8329
PC6: Explained Variance = 0.0234, Cumulative = 0.8563
PC7: Explained Variance = 0.0193, Cumulative = 0.8755
PC8: Explained Variance = 0.0187, Cumulative = 0.8942
PC9: Explained Variance = 0.0167, Cumulative = 0.9109
PC10: Explained Variance = 0.0148, Cumulative = 0.9258
PC11: Explained Variance = 0.0114, Cumulative = 0.9372
PC12: Explained Variance = 0.0096, Cumulative = 0.9468
PC13: Explained Variance = 0.0072, Cumulative = 0.9540
PC14: Explained Variance = 0.0062, Cumulative = 0.9602
PC15: Explained Variance = 0.0054, Cumulative = 0.9656
PC16: Explained Variance = 0.0052, Cumulative = 0.9708
PC17: Explained Variance = 0.0045, Cumulative = 0.9753
PC18: Explained Variance = 0.0043, Cumulative = 0.9796
PC19: Explained Variance = 0.0035, Cumulative = 0.9831
PC20: Explained Variance = 0.0032, Cumulative = 0.9864
PC21: Explained Variance = 0.0024, Cumulative = 0.9888
PC22: Explained Variance = 0.0021, Cumulative = 0.9909
PC23: Explained Variance = 0.0020, Cumulative = 0.9929
PC24: Explained Variance = 0.0016, Cumulative = 0.9944
PC25: Explained Variance = 0.0013, Cumulative = 0.9958
PC26: Explained Variance = 0.0011, Cumulative = 0.9969
PC27: Explained Variance = 0.0010, Cumulative = 0.9979
PC28: Explained Variance = 0.0010, Cumulative = 0.9990
PC29: Explained Variance = 0.0006, Cumulative = 0.9996
PC30: Explained Variance = 0.0004, Cumulative = 1.0000
```



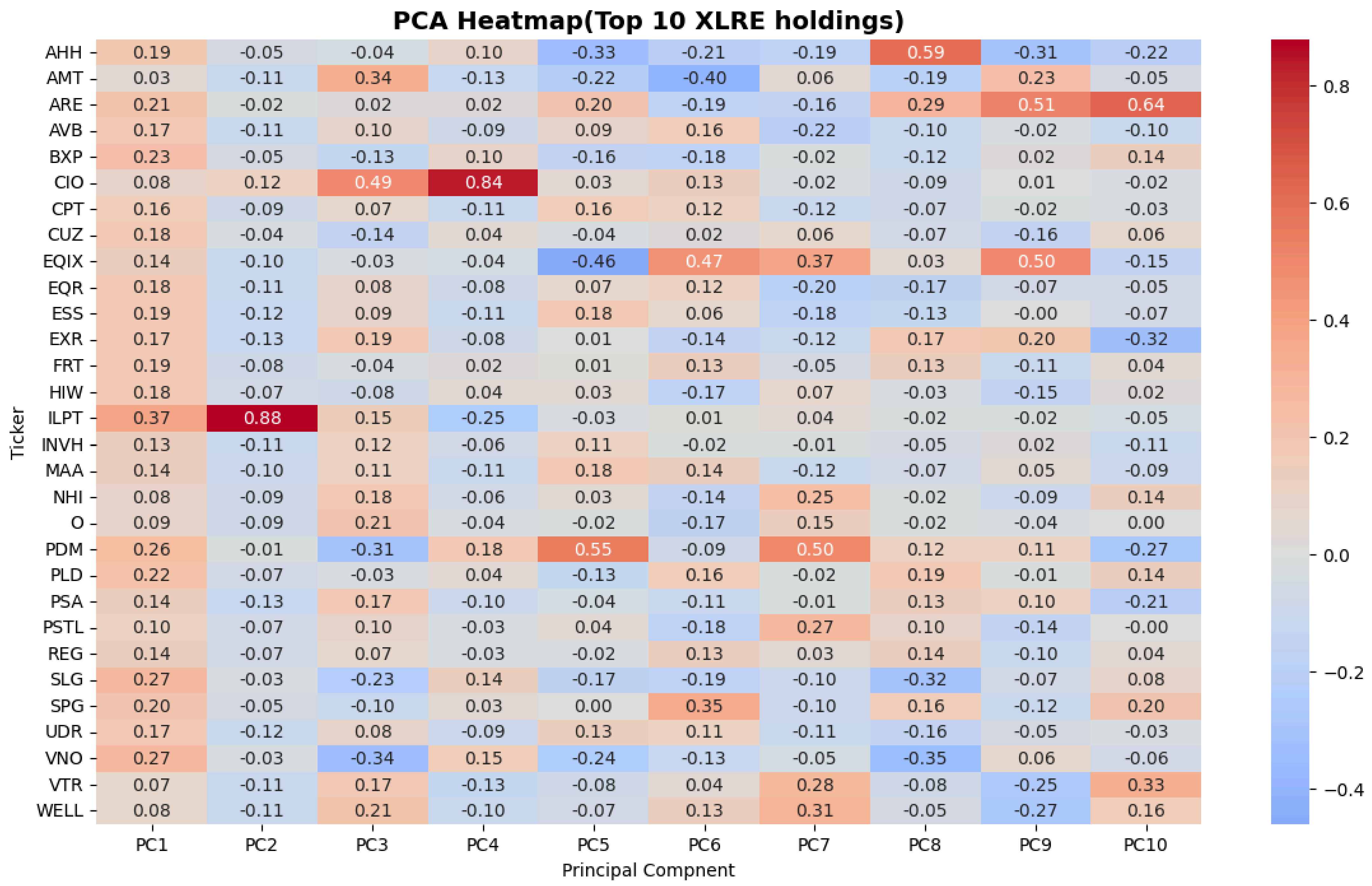
In [9]:

```
# Compute SVD
U, sigma, Vt = np.linalg.svd(daily_returns_holding, full_matrices=False)

# Variance ratio
singular_values_squared = sigma**2
explained_var_svd = singular_values_squared / np.sum(singular_values_squared)

#cumulative explained variance
cum_var_svd = np.cumsum(explained_var_svd)

for i, (var, cum_var) in enumerate(zip(explained_var_svd, cum_var_svd), start=1):
    print(f"PC{i}: Explained Variance PC= {var:.4f}, Cumulative = {cum_var:.4f}")

plt.figure(figsize=(12,8))
plt.plot(cum_var_svd, marker='x', linestyle='-', color='g', label='SVD Cumulative Explained Variance')
plt.axhline(y=0.9, color='r', linestyle='--', label='90% Variance Threshold')
plt.xlabel('Number of Components')
plt.ylabel('Cumulative Explained Variance')
plt.title('SVD Cumulative Explained Variance')
plt.legend()
plt.grid(True)
plt.show()
```

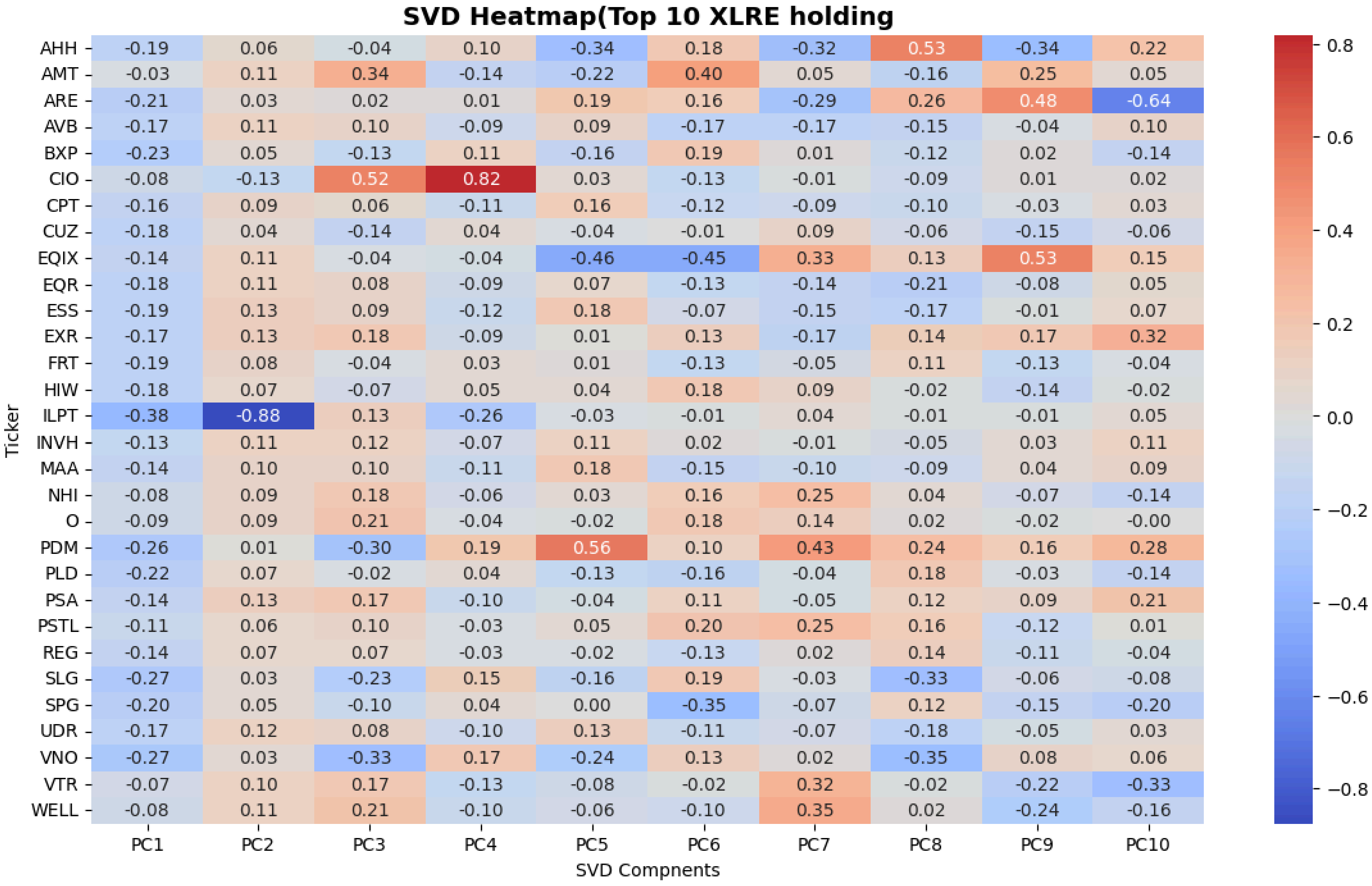
```
PC1: Explained Variance PC= 0.5501, Cumulative = 0.5501
PC2: Explained Variance PC= 0.1256, Cumulative = 0.6757
PC3: Explained Variance PC= 0.0668, Cumulative = 0.7425
PC4: Explained Variance PC= 0.0614, Cumulative = 0.8039
PC5: Explained Variance PC= 0.0284, Cumulative = 0.8323
PC6: Explained Variance PC= 0.0233, Cumulative = 0.8556
PC7: Explained Variance PC= 0.0199, Cumulative = 0.8755
PC8: Explained Variance PC= 0.0187, Cumulative = 0.8942
PC9: Explained Variance PC= 0.0167, Cumulative = 0.9109
PC10: Explained Variance PC= 0.0148, Cumulative = 0.9257
PC11: Explained Variance PC= 0.0115, Cumulative = 0.9372
PC12: Explained Variance PC= 0.0096, Cumulative = 0.9468
PC13: Explained Variance PC= 0.0072, Cumulative = 0.9540
PC14: Explained Variance PC= 0.0062, Cumulative = 0.9602
PC15: Explained Variance PC= 0.0054, Cumulative = 0.9656
PC16: Explained Variance PC= 0.0052, Cumulative = 0.9708
PC17: Explained Variance PC= 0.0045, Cumulative = 0.9753
PC18: Explained Variance PC= 0.0043, Cumulative = 0.9796
PC19: Explained Variance PC= 0.0035, Cumulative = 0.9831
PC20: Explained Variance PC= 0.0032, Cumulative = 0.9864
PC21: Explained Variance PC= 0.0024, Cumulative = 0.9888
PC22: Explained Variance PC= 0.0021, Cumulative = 0.9909
PC23: Explained Variance PC= 0.0020, Cumulative = 0.9929
PC24: Explained Variance PC= 0.0016, Cumulative = 0.9944
PC25: Explained Variance PC= 0.0013, Cumulative = 0.9958
PC26: Explained Variance PC= 0.0011, Cumulative = 0.9969
PC27: Explained Variance PC= 0.0010, Cumulative = 0.9979
PC28: Explained Variance PC= 0.0010, Cumulative = 0.9990
PC29: Explained Variance PC= 0.0006, Cumulative = 0.9996
PC30: Explained Variance PC= 0.0004, Cumulative = 1.0000
```



In [10]:
```python
# PCA Heatmap
# Matrice des coefficients (loadings)
loadings = pd.DataFrame(
    pca.components_[:10].T,
    index=daily_returns_holding.columns,
    columns=[f'PC{i+1}' for i in range(10)]
)

# Affichage du heatmap
plt.figure(figsize=(14, 8))
sns.heatmap(loadings, cmap='coolwarm', center=0, annot=True, fmt='.2f')
plt.title('PCA Heatmap(Top 10 XLRE holdings)', fontsize=14, fontweight='bold')
plt.xlabel('Principal Compnent')
plt.ylabel('Ticker')
plt.show()
```



In [11]:
```python
#SVD heatmap
loadings_svd = pd.DataFrame(
    Vt[:10].T,
    index=daily_returns_holding.columns,
    columns=[f'PC{i+1}' for i in range(10)]
)

#Heatmap
plt.figure(figsize=(14, 8))
sns.heatmap(loadings_svd, cmap='coolwarm', center=0, annot=True, fmt='.2f')
plt.title('SVD Heatmap(Top 10 XLRE holding', fontsize=14, fontweight='bold')
plt.xlabel('SVD Compnents')
plt.ylabel('Ticker')
plt.show()
```


SVD Heatmap(Top 10 XLRE holding
Ticker
AHH -0.19 0.06 -0.04 0.10 -0.34 0.18 -0.32 0.53 -0.34 0.22
AMT -0.03 0.11 0.34 -0.14 -0.22 0.40 0.05 -0.16 0.25 0.05
ARE -0.21 0.03 0.02 0.01 0.19 0.16 -0.29 0.26 0.48 -0.64
AVB -0.17 0.11 0.10 -0.09 0.09 -0.17 -0.17 -0.15 -0.04 0.10
BXP -0.23 0.05 -0.13 0.11 -0.16 0.19 0.01 -0.12 0.02 -0.14
CIO -0.08 -0.13 0.52 0.82 0.03 -0.13 -0.01 -0.09 0.01 0.02
CPT -0.16 0.09 0.06 -0.11 0.16 -0.12 -0.09 -0.10 -0.03 0.03
CUZ -0.18 0.04 -0.14 0.04 -0.04 -0.01 0.09 -0.06 -0.15 -0.06
EQIX -0.14 0.11 -0.04 -0.04 -0.46 -0.45 0.33 0.13 0.53 0.15
EQR -0.18 0.11 0.08 -0.09 0.07 -0.13 -0.14 -0.21 -0.08 0.05
ESS -0.19 0.13 0.09 -0.12 0.18 -0.07 -0.15 -0.17 -0.01 0.07
EXR -0.17 0.13 0.18 -0.09 0.01 0.13 -0.17 0.14 0.17 0.32
FRT -0.19 0.08 -0.04 0.03 0.01 -0.13 -0.05 0.11 -0.13 -0.04
HIW -0.18 0.07 -0.07 0.05 0.04 0.18 0.09 -0.02 -0.14 -0.02
ILPT -0.38 -0.88 0.13 -0.26 -0.03 -0.01 0.04 -0.01 -0.01 0.05
INVH -0.13 0.11 0.12 -0.07 0.11 0.02 -0.01 -0.05 0.03 0.11
MAA -0.14 0.10 0.10 -0.11 0.18 -0.15 -0.10 -0.09 0.04 0.09
NHI -0.08 0.09 0.18 -0.06 0.03 0.16 0.25 0.04 -0.07 -0.14
O -0.09 0.09 0.21 -0.04 -0.02 0.18 0.14 0.02 -0.02 -0.00
PDM -0.26 0.01 -0.30 0.19 0.56 0.10 0.43 0.24 0.16 0.28
PLD -0.22 0.07 -0.02 0.04 -0.13 -0.16 -0.04 0.18 -0.03 -0.14
PSA -0.14 0.13 0.17 -0.10 -0.04 0.11 -0.05 0.12 0.09 0.21
PSTL -0.11 0.06 0.10 -0.03 0.05 0.20 0.25 0.16 -0.12 0.01
REG -0.14 0.07 0.07 -0.03 -0.02 -0.13 0.02 0.14 -0.11 -0.04
SLG -0.27 0.03 -0.23 0.15 -0.16 0.19 -0.03 -0.33 -0.06 -0.08
SPG -0.20 0.05 -0.10 0.04 0.00 -0.35 -0.07 0.12 -0.15 -0.20
UDR -0.17 0.12 0.08 -0.10 0.13 -0.11 -0.07 -0.18 -0.05 0.03
VNO -0.27 0.03 -0.33 0.17 -0.24 0.13 0.02 -0.35 0.08 0.06
VTR -0.07 0.10 0.17 -0.13 -0.08 -0.02 0.32 -0.02 -0.22 -0.33
WELL -0.08 0.11 0.21 -0.10 -0.06 -0.10 0.35 0.02 -0.24 -0.16
PC1 PC2 PC3 PC4 PC5 PC6 PC7 PC8 PC9 PC10
SVD Compnents
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6
−0.8