Module 5

-By Suprateek Halsana

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
sns.set(style='darkgrid')
```

Problem [5.1]

In [3]:
```
data=pd.read_csv(r"C:\Users\Suprateek Halsana\Documents\Python Scripts\Aspiring Mind Internship\week3.csv")
data
```

Out[3]:

	Date	Symbol	Series	Prev Close	Open Price	High Price	Low Price	Last Price	Close Price	Average Price	...	Deliverable Qty	% Dly Qt to Traded Qty	Month	Year	De
0	2017-05-15	LALPATHLAB	EQ	891.15	895.0	914.15	881.00	912.00	900.60	889.35	...	174775	67.83	5	2017	
1	2017-05-16	LALPATHLAB	EQ	900.60	910.0	925.00	895.05	909.40	910.95	914.57	...	75813	72.27	5	2017	
2	2017-05-17	LALPATHLAB	EQ	910.95	913.0	925.00	909.00	912.05	911.70	917.19	...	53829	74.29	5	2017	
3	2017-05-18	LALPATHLAB	EQ	911.70	908.0	919.35	903.05	906.00	909.75	914.12	...	24836	65.87	5	2017	
4	2017-05-19	LALPATHLAB	EQ	909.75	917.0	917.00	905.80	910.00	910.25	910.61	...	69926	86.24	5	2017	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
489	2019-05-07	LALPATHLAB	EQ	1024.95	1015.7	1031.95	1006.90	1013.00	1013.10	1020.66	...	11587	57.61	5	2019	
490	2019-05-08	LALPATHLAB	EQ	1013.10	1014.0	1019.85	1001.90	1002.00	1006.10	1010.90	...	11239	62.27	5	2019	
491	2019-05-09	LALPATHLAB	EQ	1006.10	1014.0	1014.00	977.70	979.90	982.90	987.36	...	37303	61.62	5	2019	
492	2019-05-10	LALPATHLAB	EQ	982.90	989.9	994.30	963.60	980.00	980.65	980.71	...	19097	62.47	5	2019	
493	2019-05-13	LALPATHLAB	EQ	980.65	970.8	996.00	960.00	975.00	976.25	974.21	...	23772	63.93	5	2019	

494 rows × 23 columns

In [4]:
```
# Mean Daily Return of the Stock
print('Mean of Daily Return : ',data['Day_Perc_change'].mean())

# Annual Mean Calculated by Multiplying Mean Daily Return by 252 as there are 252 Trading days in a year
print('Annual Mean Return   : ',(data['Day_Perc_change'].mean())*252)

# Mean Daily Standard Deviation of stock
print('Daily Std. Deviation : ',data['Day_Perc_change'].std())

# Mean Annual Standard Deviation of stock
print('Volatility or Annual std. : ',(data['Day_Perc_change'].std())*(252**0.5))
```

```
Mean of Daily Return :  0.03002958498819413
Annual Mean Return   :  7.567455417026252
Daily Std. Deviation :  1.6641305183311783
Volatility or Annual std. :  26.417253003943873
```

Problem [5.2] Diversifying the Portfolio of 5 stocks

In [5]:
```
#Choosing Five Stocks of my Choice

tcs=pd.read_csv(r"C:\Users\Suprateek Halsana\Documents\Python Scripts\Aspiring Mind Internship\Prerequisites\Large_Cap\Large_Cap\TCS.csv")
pnb=pd.read_csv(r"C:\Users\Suprateek Halsana\Documents\Python Scripts\Aspiring Mind Internship\Prerequisites\Mid_Cap\Mid_Cap\PNB.csv")
pvr=pd.read_csv(r"C:\Users\Suprateek Halsana\Documents\Python Scripts\Aspiring Mind Internship\Prerequisites\Small_Cap\Small_Cap\PVR.csv")
voltas=pd.read_csv(r"C:\Users\Suprateek Halsana\Documents\Python Scripts\Aspiring Mind Internship\Prerequisites\Mid_Cap\Mid_Cap\VOLTAS.csv")
gail=pd.read_csv(r"C:\Users\Suprateek Halsana\Documents\Python Scripts\Aspiring Mind Internship\Prerequisites\Large_Cap\Large_Cap\GAIL.csv")
```

In [6]:
```
daily_ret = pd.DataFrame()
daily_ret['Date']=data['Date']
daily_ret['tcs']=tcs['Close Price'].pct_change()
daily_ret['pnb']=pnb['Close Price'].pct_change()
daily_ret['pvr']=pvr['Close Price'].pct_change()
daily_ret['voltas']=voltas['Close Price'].pct_change()
daily_ret['gail']=gail['Close Price'].pct_change()
```

In [7]:
```
# Daily_ret Portfolio of 5 stocks
daily_ret
```

Out[7]:

	Date	tcs	pnb	pvr	voltas	gail
0	2017-05-15	NaN	NaN	NaN	NaN	NaN
1	2017-05-16	0.027081	0.044065	0.009213	0.001389	-0.006280
2	2017-05-17	0.010786	-0.052254	-0.005653	-0.005203	-0.007535
3	2017-05-18	0.032928	-0.039685	-0.009965	-0.037424	-0.008695
4	2017-05-19	-0.011454	-0.020820	-0.000990	0.003985	-0.009141
...	...	...	...	...	...	...
489	2019-05-07	0.009738	-0.026667	-0.003208	-0.007590	-0.015884
490	2019-05-08	-0.019886	0.000000	-0.012318	-0.041234	-0.009538
491	2019-05-09	-0.037646	0.003574	-0.017752	-0.004422	0.008593
492	2019-05-10	0.012125	0.023739	0.005691	0.010364	0.003378
493	2019-05-13	-0.002734	-0.074783	-0.024482	-0.013533	-0.027814

494 rows × 6 columns

In [8]:
```
mean_daily_returns = daily_ret.mean()
cov_matrix = daily_ret.cov()
weights=np.array([0.2,0.2,0.2,0.2,0.2])
portfolio_return = round(np.sum(mean_daily_returns * weights) * 252,2)
#calculate annualised portfolio volatility
portfolio_std_dev = round(np.sqrt(np.dot(weights.T,np.dot(cov_matrix, weights))) * np.sqrt(252),2)
print('Portfolio expected annualised return is :',portfolio_return)
print('Portfolio expected Volatility is :',portfolio_std_dev)
```

```
Portfolio expected annualised return is : 0.02
Portfolio expected Volatility is : 0.2
```

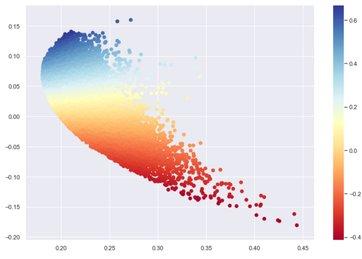
Problem [5.3]

In [9]:
```
num_portfolios = 25000
#set up array to hold results
results = np.zeros((3,num_portfolios))
for i in range(num_portfolios):
    #select random weights for portfolio holdings
    weights = np.random.random(5)
    #rebalance weights to sum to 1
    weights /= np.sum(weights)

    #calculate portfolio return and volatility
    portfolio_return = np.sum(mean_daily_returns * weights) * 252
    portfolio_std_dev = np.sqrt(np.dot(weights.T,np.dot(cov_matrix, weights))) * np.sqrt(252)

    #store results in results array
    results[0,i] = portfolio_return
    results[1,i] = portfolio_std_dev
    #store Sharpe Ratio (return / volatility) - risk free rate element excluded for simplicity
    results[2,i] = results[0,i] / results[1,i]
#convert results array to Pandas DataFrame
results_frame = pd.DataFrame(results.T,columns=['ret','stdev','sharpe'])
#create scatter plot coloured by Sharpe Ratio
plt.figure(figsize=(12,8))
plt.scatter(results_frame.stdev,results_frame.ret,c=results_frame.sharpe,cmap='RdYlBu')
plt.colorbar()
```

Out[9]:
```
<matplotlib.colorbar.Colorbar at 0x23d34bb7c08>
```



Problem [5.4]

In [10]:
```
#list of stocks in portfolio
stocks = ['tcs','pnb','pvr','voltas','gail']

#convert daily stock prices into daily returns
returns = daily_ret[stocks]

#calculate mean daily return and covariance of daily returns
mean_daily_returns = returns.mean()
cov_matrix = returns.cov()

num_portfolios = 25000

results = np.zeros((4+len(stocks)-1,num_portfolios))

for i in range(num_portfolios):

    weights = np.array(np.random.random(5))
    #rebalance weights to sum to 1
    weights /= np.sum(weights)


    portfolio_return = np.sum(mean_daily_returns * weights) * 252
    portfolio_std_dev = np.sqrt(np.dot(weights.T,np.dot(cov_matrix, weights))) * np.sqrt(252)


    results[0,i] = portfolio_return
    results[1,i] = portfolio_std_dev

    # Sharpe Ratio (return / volatility) - risk free rate element excluded for simplicity
    results[2,i] = results[0,i] / results[1,i]

    for j in range(len(weights)):
        results[j+3,i] = weights[j]


results_frame = pd.DataFrame(results.T,columns=['ret','stdev','sharpe',stocks[0],stocks[1],stocks[2],stocks[3],stocks[4]])

# position of portfolio with highest Sharpe Ratio
max_sharpe_port = results_frame.iloc[results_frame['sharpe'].idxmax()]
# position of portfolio with minimum standard deviation
min_vol_port = results_frame.iloc[results_frame['stdev'].idxmin()]

# scatter plot
plt.scatter(results_frame.stdev,results_frame.ret,c=results_frame.sharpe,cmap='RdYlBu')
plt.xlabel('Volatility')
plt.ylabel('Returns')
plt.colorbar()

#plot red star to highlight position of portfolio with highest Sharpe Ratio
plt.scatter(max_sharpe_port[1],max_sharpe_port[0],marker=(5,1,0),color='r',s=1000)
#plot green star to highlight position of minimum variance portfolio
plt.scatter(min_vol_port[1],min_vol_port[0],marker=(5,1,0),color='g',s=1000)
```

Out[10]:
```
<matplotlib.collections.PathCollection at 0x23d34fdf2c8>
```