Study on Resilience of Local Banks

- DBS, OCBC, UOB
- Covid-19 vs Financial Crisis

Presented by
Group D

Packages, Variables, and Data Downloading

In [2]:
```
# load libraries
import pandas as pd
import yfinance as yf
import numpy as np
import matplotlib
%matplotlib inline
import matplotlib.pyplot as plt
import matplotlib.ticker as mticker
import mplfinance as mpf
import matplotlib.dates as mpl_dates
import seaborn as sns
sns.set(style="white", color_codes=True)
```

In [5]:
```
# common variables
START_DATE_1 = '2020-01-01'
END_DATE_1 = '2020-09-01'
START_DATE_2 = '2007-12-01'
END_DATE_2 = '2009-07-01'
finance_sector = ['DBS','OCBC','UOB']

ticker_list = ['D05.SI','O39.SI','U11.SI']
```

- Covid-19 timeframe: 2020-01-01 ~ 2020-09-01
- Financial Crisis timeframe: 2007-12-01 ~ 2009-07-01

Download data of local backs' equities during Covid-19 period, namely, OCBC, DBS and UOB. According to statistics, the impact of Covid-19 on Singapore started from Janurary, 2020 and it is still ongoing.

In [12]:
```
df_ocbc = yf.download(ticker_list[1],
                      start = START_DATE_1,
                      end = END_DATE_1,
                      progress = False)
df_dbs = yf.download(ticker_list[0],
                      start = START_DATE_1,
                      end = END_DATE_1,
                      progress = False)
df_uob = yf.download(ticker_list[2],
                      start = START_DATE_1,
                      end = END_DATE_1,
                      progress = False)
```

Download data of local backs' equities during financial crisis (FC) period, namely, OCBC, DBS and UOB. According to statistics, the impact of FC on Singapore started from December, 2007 and lasted till June, 2009.

In [63]:
```
df_ocbc_fc = yf.download(ticker_list[1],
                      start = START_DATE_2,
                      end = END_DATE_2,
                      progress = False)
df_dbs_fc = yf.download(ticker_list[0],
                      start = START_DATE_2,
                      end = END_DATE_2,
                      progress = False)
df_uob_fc = yf.download(ticker_list[2],
                      start = START_DATE_2,
                      end = END_DATE_2,
                      progress = False)
```

In [9]:
```
print(f'Downloaded {df_ocbc.shape[0]} rows of data.')
print(df_ocbc.head(10))
print(df_ocbc.tail(10))
```

```
Downloaded 165 rows of data.
              Open   High    Low  Close  Adj Close   Volume
Date
2020-01-02  10.97  11.06  10.91  11.03  10.490239  3831900
2020-01-03  11.08  11.14  10.94  11.02  10.480728  4381400
2020-01-06  10.95  10.96  10.88  10.92  10.385622  4399500
2020-01-07  10.98  11.06  10.96  11.00  10.461707  4941200
2020-01-08  10.84  11.04  10.78  11.01  10.471218  7779500
2020-01-09  11.11  11.11  10.99  11.03  10.490239  4537300
2020-01-10  11.07  11.09  11.04  11.08  10.537792  3046600
2020-01-13  11.04  11.09  11.02  11.06  10.518771  2350100
2020-01-14  11.08  11.12  11.06  11.10  10.556813  3208900
2020-01-15  11.07  11.11  10.97  10.98  10.442686  3412800
              Open  High   Low  Close  Adj Close   Volume
Date
2020-08-18  8.95  8.95  8.85   8.85   8.690640  4567000
2020-08-19  8.85  8.90  8.84   8.90   8.739739  2934800
2020-08-20  8.83  8.89  8.78   8.83   8.671000  8649600
2020-08-21  8.79  8.79  8.67   8.67   8.670000  7513000
2020-08-24  8.66  8.67  8.61   8.61   8.610000  5173700
2020-08-25  8.62  8.83  8.62   8.74   8.740000  5351300
2020-08-26  8.75  8.75  8.63   8.67   8.670000  3998000
2020-08-27  8.67  8.70  8.60   8.61   8.610000  5726100
2020-08-28  8.75  8.87  8.67   8.71   8.710000  8154700
2020-08-31  8.81  8.81  8.65   8.67   8.670000  6199000
```

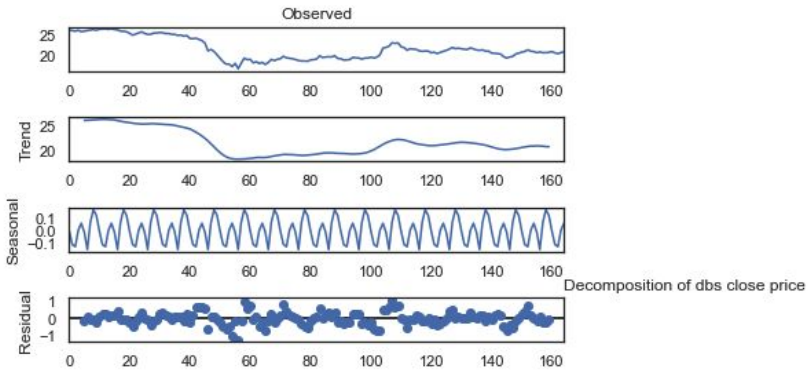
Individual Data Smoothing - Decomposition

- For illustration, only show DBS data

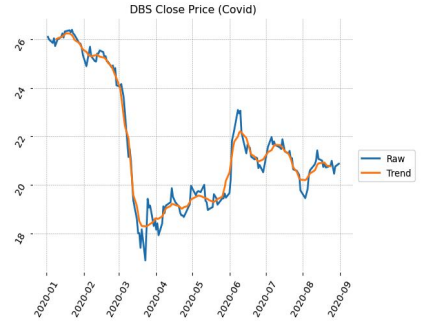
In [13]:
```
# Smooth technique: decomposition

# demonstrate using column 'Close' first

dbs_close = df_dbs.Close[:]
ocbc_close = df_ocbc.Close[:]
uob_close = df_uob.Close[:]
# dbs_close = df_dbs['Close']

from statsmodels.tsa.seasonal import seasonal_decompose
dbs_close_decomp = seasonal_decompose(dbs_close.values, period=10)
ocbc_close_decomp = seasonal_decompose(ocbc_close.values, period=10)
uob_close_decomp = seasonal_decompose(uob_close.values, period=10)

# plot of decomposition of each close price
plt.figure(1)
dbs_close_decomp.plot()
plt.title('Decomposition of dbs close price',loc='right', horizontalalignment='left', verticalalignment='baseline')
```

Out[13]: Text(1.0, 1.0, 'Decomposition of uob close price')

<Figure size 432x288 with 0 Axes>



In [41]:
```
# for DBS
# Better view (instead of subplot)
plt.plot(df_dbs.index.values, dbs_close)
plt.plot(df_dbs.index.values, dbs_close_trend)
plt.title('DBS Close Price (Covid)')
plt.legend(['Raw','Trend'], loc='upper left', bbox_to_anchor=(1.0, 0.5))
plt.tick_params(labelrotation=60)
```



In [68]:
```
# for DBS
# Better view (instead of subplot)
plt.plot(df_dbs_fc.index.values, dbs_close_fc)
plt.plot(df_dbs_fc.index.values, dbs_close_fc_trend)
plt.title('DBS Close Price (FC)')
plt.legend(['Raw','Trend'], loc='upper left', bbox_to_anchor=(1.0, 0.5))
plt.tick_params(labelrotation=60)
```

Individual MA - with nan/ remove nan

- MA - Moving Average
- For illustration, only show OCBC data

In [57]:
```
# plot of ocbc (mean with NAN)

plt.figure(figsize = (15,10))
plt.grid(True)
plt.plot(ocbc_close_new['Close'], label = 'raw')
plt.plot(ocbc_close_new['MA_5'], label = 'MA in 5 day')
plt.plot(ocbc_close_new['MA_20'], label = 'MA in 20 day')
plt.plot(ocbc_close_new['MA_60'], label = 'MA in 60 day')
plt.legend(loc='upper left', bbox_to_anchor=(1.0, 0.5))
plt.title('OCBC Stock Price (Close): Raw vs MA (with NAN) - Covid')
```

Out[57]: Text(0.5, 1.0, 'OCBC Stock Price (Close): Raw vs MA (with NAN) - Covid')

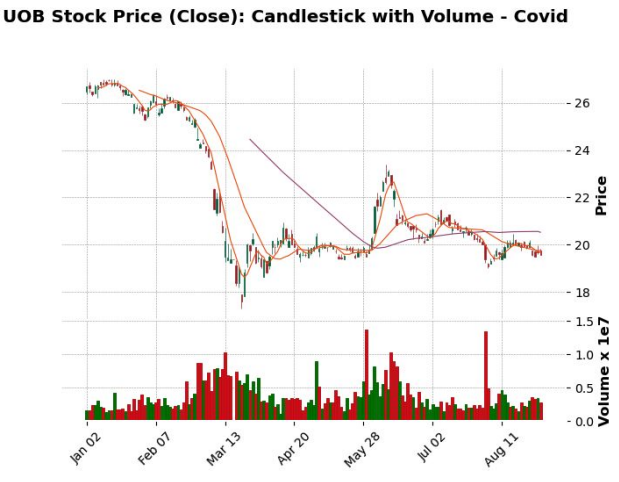
In [56]:
```
# plot of ocbc (mean remove NAN)

plt.figure(figsize = (15,10))
plt.grid(True)
plt.plot(ocbc_close_newer['Close'], label = 'raw')
plt.plot(ocbc_close_newer['MA_5'], label = 'MA in 5 day')
plt.plot(ocbc_close_newer['MA_20'], label = 'MA in 20 day')
plt.plot(ocbc_close_newer['MA_60'], label = 'MA in 60 day')
plt.legend(loc='upper left', bbox_to_anchor=(1.0, 0.5))
plt.title('OCBC Stock Price (Close): Raw vs MA (remove NAN) - Covid')
```

Out[56]: Text(0.5, 1.0, 'OCBC Stock Price (Close): Raw vs MA (remove NAN) - Covid')

Individual Candlestick - with same MA and Volume

- OLHC - Open/Low/High/Close
- For illustration, only show UOB data

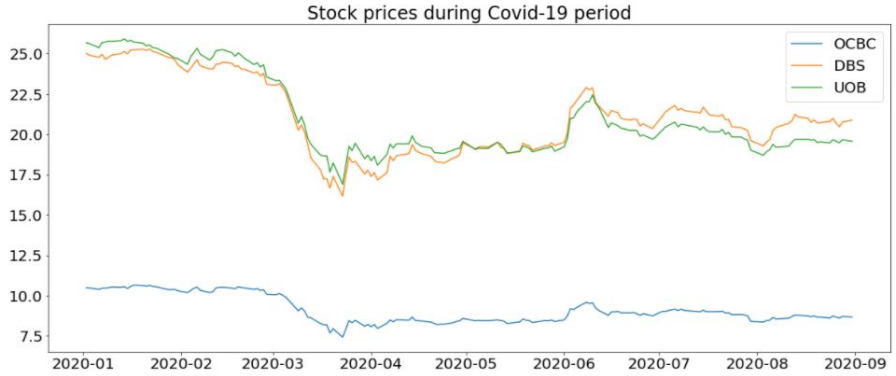
```
In [82]: # for UOB
         # mav - moving average, use the same as above for comparison
         mpf.plot(df_uob, type='candle', style = 'charles', title='UOB Stock Price (Close): Candlestick with Volume - Covid',
                  mav=(5,20,60), volume=True)
```




```
In [85]: # Candlestick
         # for UOB
         # mav - moving average, use the same as above for comparison
         mpf.plot(df_uob_fc, type='candle', style = 'charles', title='UOB Stock Price (Close): Candlestick with Volume - FC',
                  mav=(5,20,60), volume=True)
```

Group Comparison - Summary Statistics (Covid)

Coefficient of variation(CV) = Standard deviation / Mean



OCBC: CV = 0.0935

```
count    165.000000
mean       9.131364
std        0.853889
min        7.427812
25%        8.454962
50%        8.828118
75%       10.081282
max       10.651920
Name: Adj Close, dtype: float64
```

DBS: CV = 0.113

```
count    165.000000
mean      21.110447
std        2.380787
min       16.166729
25%       19.251144
50%       20.873108
75%       23.091223
max       25.284458
Name: Adj Close, dtype: float64
```

UOB: CV = 0.120

```
count    165.000000
mean      21.091490
std        2.526189
min       16.901987
25%       19.211622
50%       19.912987
75%       23.347256
max       25.915741
Name: Adj Close, dtype: float64
```

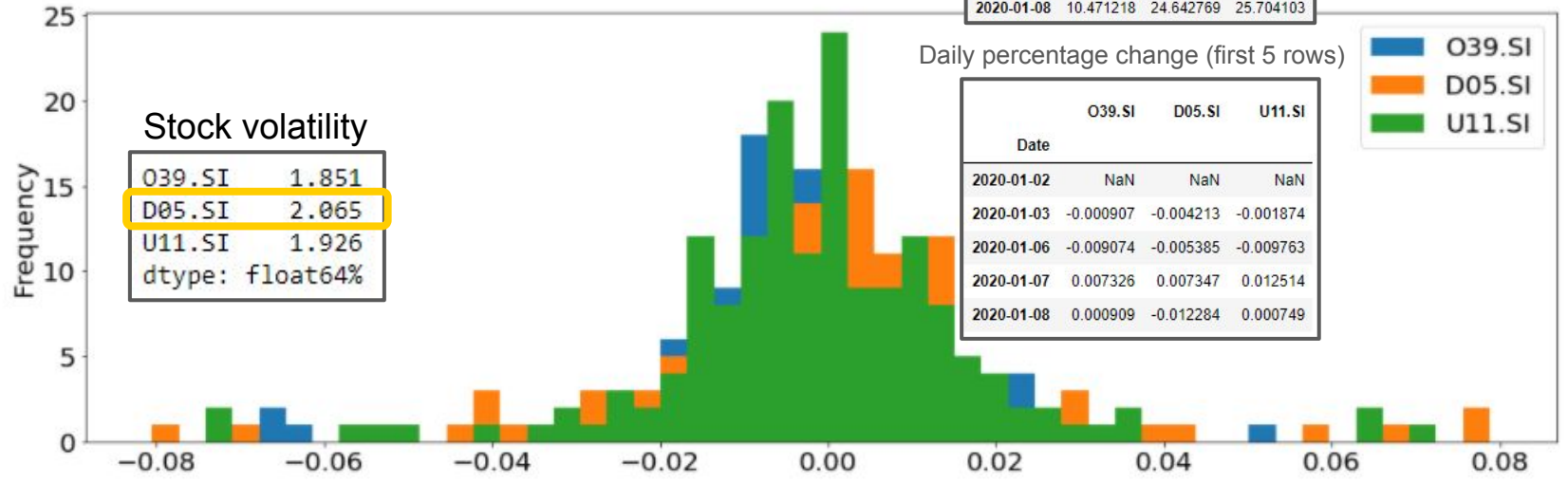
Group Comparison - Daily Returns (Covid)

- Histogram Graph

Daily stock price (first 5 rows)

Date	O39.SI	D05.SI	U11.SI
2020-01-02	10.490239	25.006714	25.665627
2020-01-03	10.480728	24.901360	25.617527
2020-01-06	10.385622	24.767277	25.367414
2020-01-07	10.461707	24.949247	25.684868
2020-01-08	10.471218	24.642769	25.704103

```
In [19]: stock_returns_daily.plot.hist(bins=50, figsize=(20,6))
Out[19]: <matplotlib.axes._subplots.AxesSubplot at 0x215f52bdc70>
```



Stock volatility

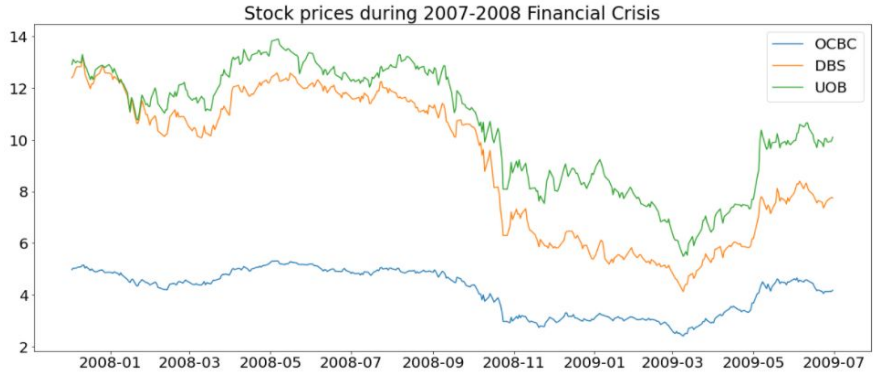
```
O39.SI    1.851
D05.SI    2.065
U11.SI    1.926
dtype: float64%
```

Daily percentage change (first 5 rows)

Date	O39.SI	D05.SI	U11.SI
2020-01-02	NaN	NaN	NaN
2020-01-03	-0.000907	-0.004213	-0.001874
2020-01-06	-0.009074	-0.005385	-0.009763
2020-01-07	0.007326	0.007347	0.012514
2020-01-08	0.000909	-0.012284	0.000749

Group Comparison - Summary Statistics (Financial Crisis)

Coefficient of variation(CV) = Standard deviation / Mean



OCBC: CV = 0.203

```
count    396.000000
mean       4.162840
std        0.845333
min        2.407387
25%        3.173787
50%        4.471333
75%        4.908939
max        5.311216
Name: Adj Close, dtype: float64
```

DBS: CV = 0.302

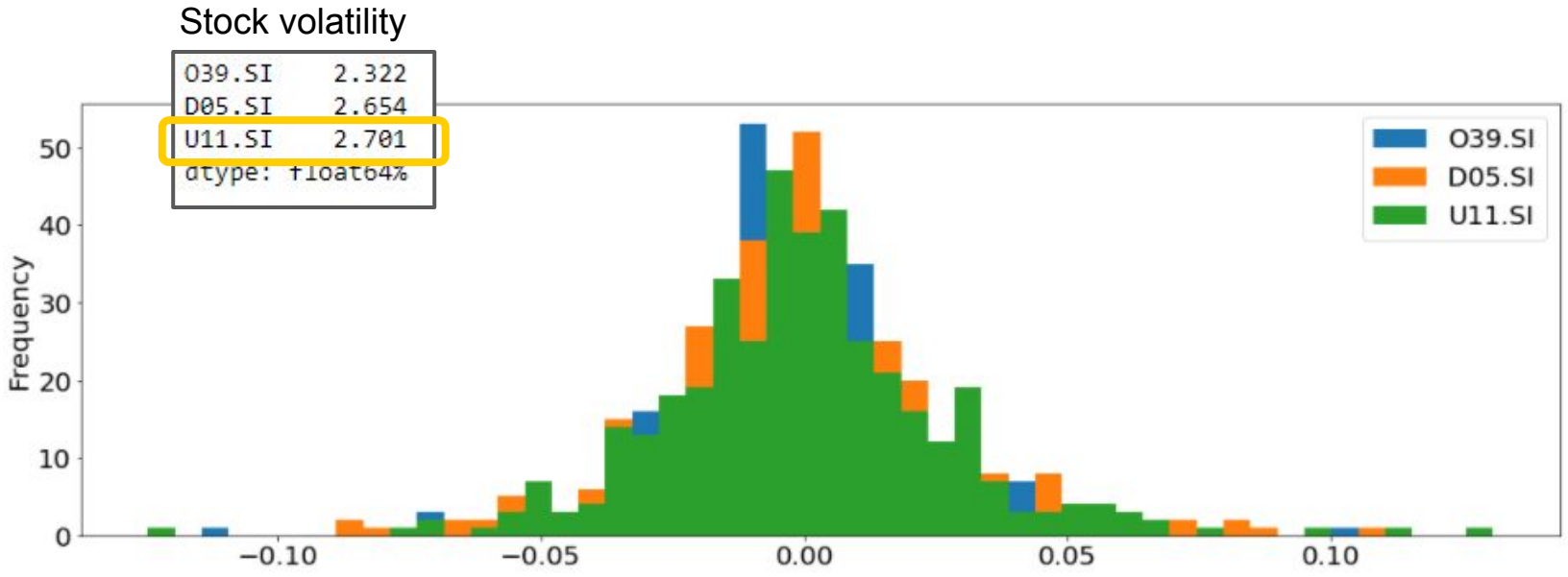
```
count    396.000000
mean       9.137103
std        2.759718
min        4.127299
25%        6.107762
50%       10.327089
75%       11.696024
max       13.191490
Name: Adj Close, dtype: float64
```

UOB: CV = 0.217

```
count    396.000000
mean      10.560537
std        2.293021
min        5.490165
25%        8.503059
50%       11.173123
75%       12.701977
max       13.898277
Name: Adj Close, dtype: float64
```

Group Comparison - Daily Returns (Financial Crisis)

- Histogram Graph



Future Work

- Indeep Analysis of other areas (cashflow, profit,..etc.)
- Prediction (e.g. monte carlo model)
- ....