In [8]:

```
import importlib

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import datetime

from model.Portfolio import Portfolio
from model.Optimizer import Optimizer
plt.rcParams["figure.figsize"] = 10, 15
```

In [28]:

```
names = ["C38U", "V01", "AGS", "S63", "CJLU"]

p = Portfolio()

# Set risk-free investment as 2%, approximately SSB's returns
p.rf = 0.02

# Add all assets
for name in names:
    p.addAsset(f"data/{name}.csv", name)

# Convert non SGD assets to SGD
p.addExchangeRate("data/forex/SGDEUR.csv", "EUR", True)
p.addExchangeRate("data/forex/USDSGD.csv", "USD", False)
```

In [29]:

```
currentWeight = [20, 20, 10, 10, 20]

normalisedWeight = np.array(currentWeight)/np.sum(currentWeight)
normalisedWeight
```

Out[29]:

```
array([0.25 , 0.25 , 0.125, 0.125, 0.25 ])
```

In [30]:

```
currentResult, currentBtPlot = p.backtest(normalisedWeight)
```

In [31]:

```
currentResult
```

Out[31]:

```
{'dateStart': Timestamp('2017-07-19 00:00:00'),
 'dateEnd': Timestamp('2019-07-11 00:00:00'),
 'days': 722,
 'valueStart': 100000.0,
 'valueEnd': 132466.452792,
 'sharpe': 0.9693531424253927,
 'drawdown': 0.2634539846178349,
 'drawdownPeriod': 5,
 'moneydown': 349.9100000000035,
 'maxDrawdown': 3.6936890016748727,
 'maxDrawdownPeriod': 83,
 'maxMoneydown': 3904.7364359999774,
 'averageReturns': 0.1013553616144504,
 'standardDeviation': 0.08392747498697255,
 'positiveYears': 3,
 'negativeYears': 0,
 'noChangeYears': 0,
 'bestYearReturns': 0.21867233476711623,
 'worstYearReturns': 0.027099521279999816}
```

In [32]:

```
currentBtPlot()
```


Broker (None)
cash 1349.92
DataTrades (True)
C38U
DrawDown (None)
drawdown 0.26
C38U (1 Day) C:2.69
BuySell (True, 0.015)
buy
sell
V01 (1 Day) C:6.98
AGS (1 Day) C:0.82
S63 (1 Day) C:4.27
CJLU (1 Day) C:0.89
2017-10
2018-01
2018-04
2018-07
2018-10
2019-01
2019-04
2019-07

Out[32]:

```
[[<Figure size 720x1080 with 8 Axes>]]
```

In [33]:

```
o = Optimizer(p)
optimisedWeight, tests = o.kfold(5)
```

In [34]:

```
optimisedWeight
```

Out[34]:

```
[0.17762004381271507,
 0.4721272422185603,
 0.04951591962747589,
 0.034289299303285635,
 0.2664474950384139]
```

In [35]:

```
tests
```

Out[35]:

```
{'sharpeRaw': [18.86477085827588,
  28.52521008289348,
  13.309617534694052,
  17.179945270991364,
  72.77416163978661],
 'sharpeAvg': 30.130741077328274,
 'sharpeStd': 21.90266851902461,
 'weightsRaw': [array([0.16518489, 0.45017547, 0.05071523, 0.0648537
3, 0.26907069]),
  array([0.19512889, 0.4171247 , 0.07089923, 0.03113336, 0.2857138
1]),
  array([0.16656129, 0.42101565, 0.04944537, 0.03140223, 0.3315754
6]),
  array([0.14677685, 0.54423772, 0.04267564, 0.02897742, 0.2373323
8]),
  array([0.2144483 , 0.52808268, 0.03384413, 0.01507976, 0.2085451
3])],
 'weightsStd': 0    0.024046
 1    0.053760
 2    0.012256
 3    0.016430
 4    0.042000
 dtype: float64}
```

In [36]:

```
optimisedResult, optimisedBtPlot = p.backtest(optimisedWeight)
```

In [37]:

```
optimisedResult
```

Out[37]:

```
{'dateStart': Timestamp('2017-07-19 00:00:00'),
 'dateEnd': Timestamp('2019-07-11 00:00:00'),
 'days': 722,
 'valueStart': 100000.0,
 'valueEnd': 134462.76975799998,
 'sharpe': 1.1854133492962902,
 'drawdown': 0.20323356867535736,
 'drawdownPeriod': 5,
 'moneydown': 273.8300000000163,
 'maxDrawdown': 2.91182936002046,
 'maxDrawdownPeriod': 68,
 'maxMoneydown': 3260.6061270000064,
 'averageReturns': 0.10606000102214774,
 'standardDeviation': 0.07259914954832968,
 'positiveYears': 3,
 'negativeYears': 0,
 'noChangeYears': 0,
 'bestYearReturns': 0.20798024439325924,
 'worstYearReturns': 0.04436895365000004}
```

In [38]:

```
optimisedBtPlot()
```

Out[38]:

```
[[<Figure size 720x1080 with 8 Axes>]]
```

In [39]:

```
dict(zip(names, np.array(optimisedWeight)*100))
```

Out[39]:

```
{'C38U': 17.762004381271506,
 'V01': 47.21272422185603,
 'AGS': 4.951591962747589,
 'S63': 3.4289299303285636,
 'CJLU': 26.644749503841393}
```

In [ ]:

```

```