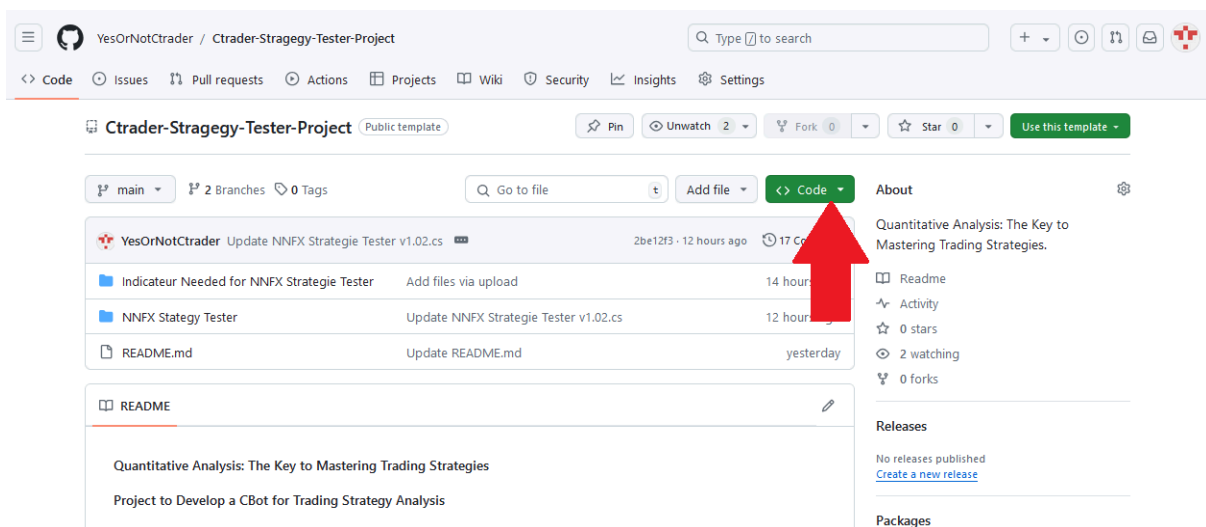


## How to Install the NAFX Strategy Tester

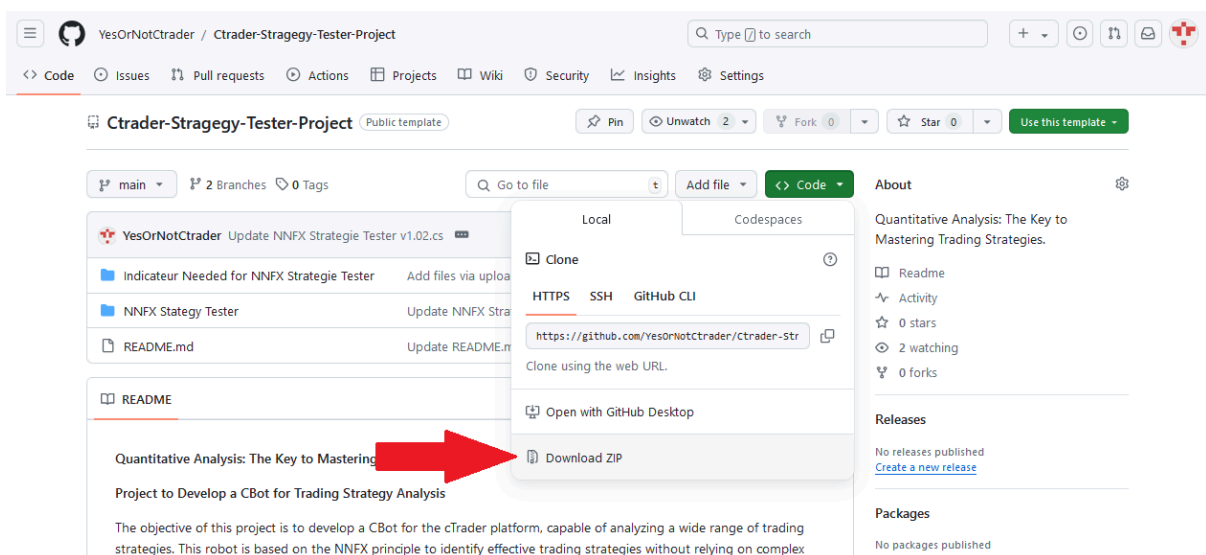
This guide provides two detailed step-by-step methods to install the cBot in the cTrader interface.

### I.Download/Install with the ZIP file:

- 1.
2. Go to the designated page  
<https://github.com/YesOrNotCtrader/Ctrader-Strategy-Tester-Project>
3. Click on the green **Code** button.

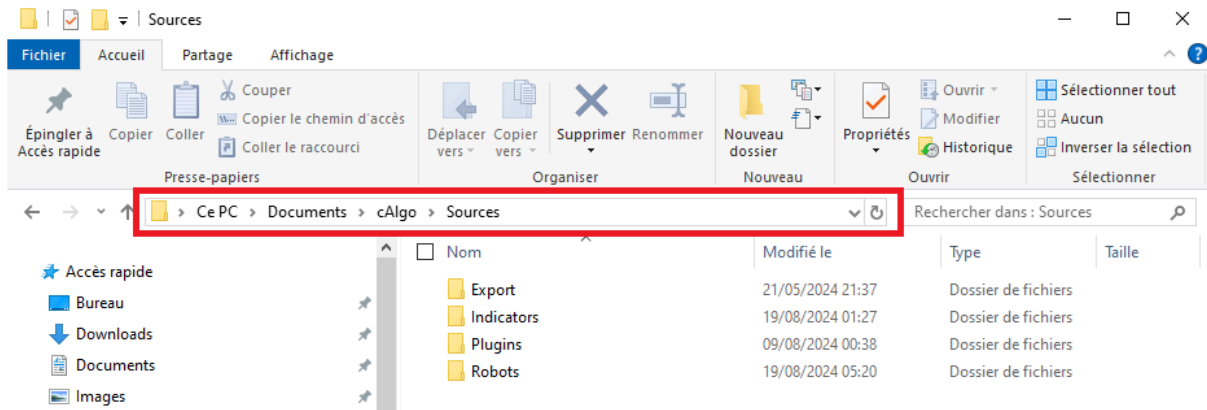


4. Select the **Download ZIP** option to download the file.

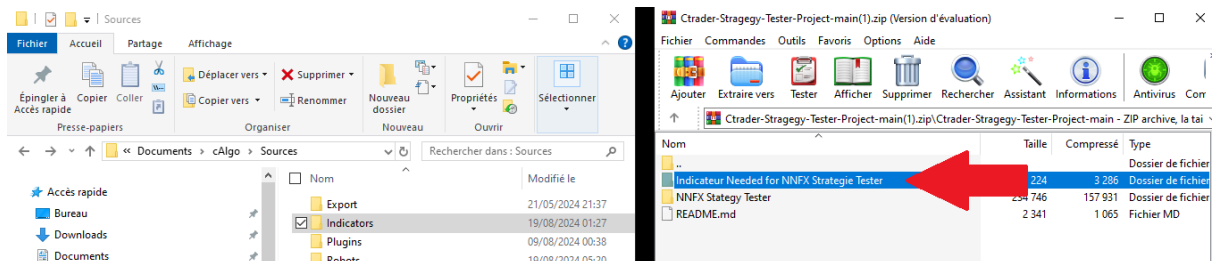


### Add the Downloaded Files to the cAlgo Folder:

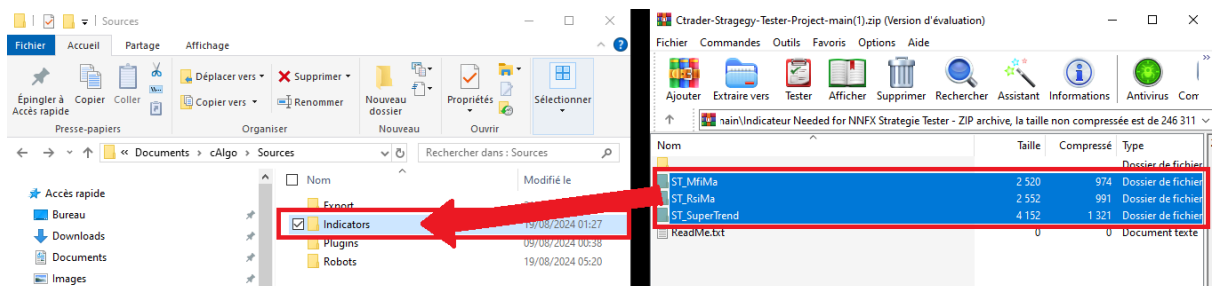
1. Open the **Source** folder for cTrader. This folder is usually located in **Documents** -> **cAlgo** -> **Source**.



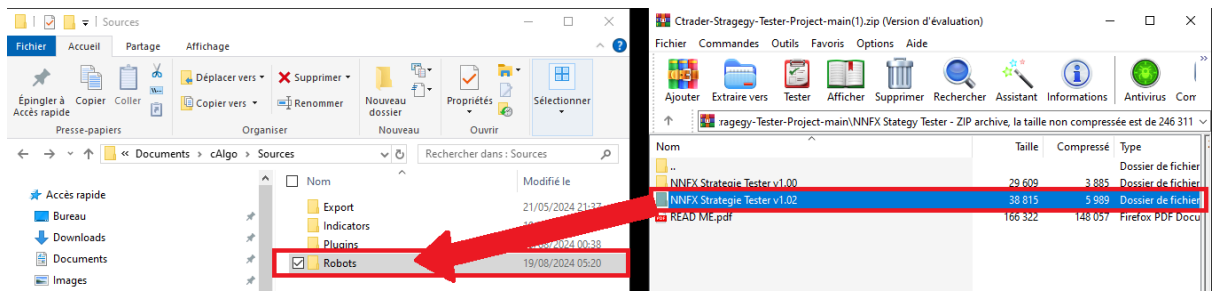
2. Open the downloaded ZIP file and click on "Indicateur Needed For NAFX Strategie Tester".



3. Select all folders named "ST\_..." (ctrl + click) and Drag and drop these folders into the **Indicators** folder located within the Source folder (from step 1).



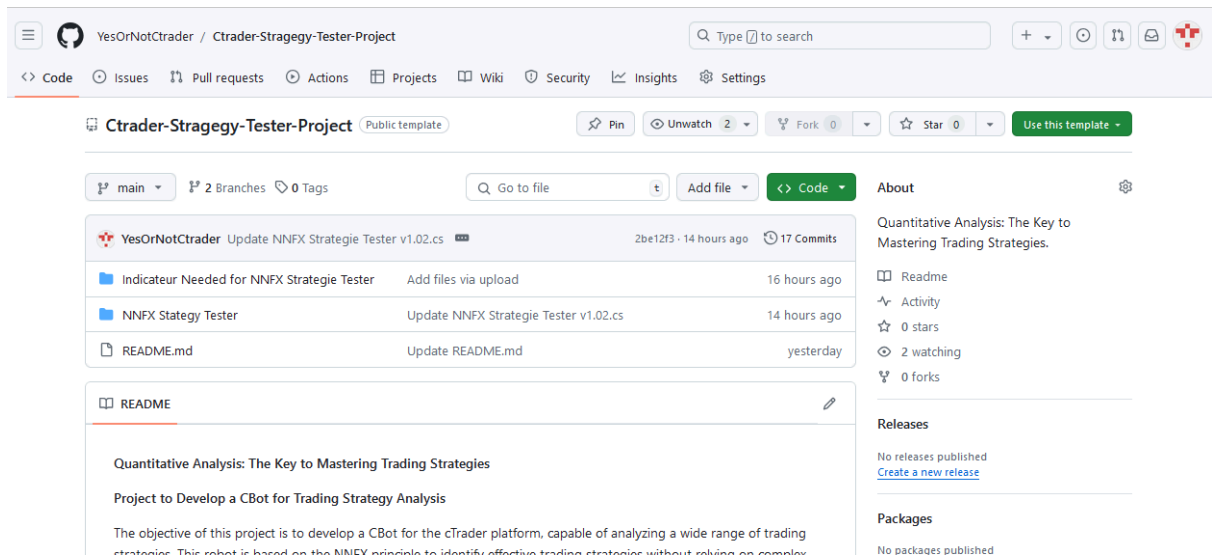
4. Return to the **root** of the downloaded folder (click on "..") and open the folder containing the latest version of cbot located in "**NAFX Strategy Tester**" Folder. For this **demonstration, we are using version 1.02** (the higher the number, the newer the version), and drag the downloaded folder into **Robots**.



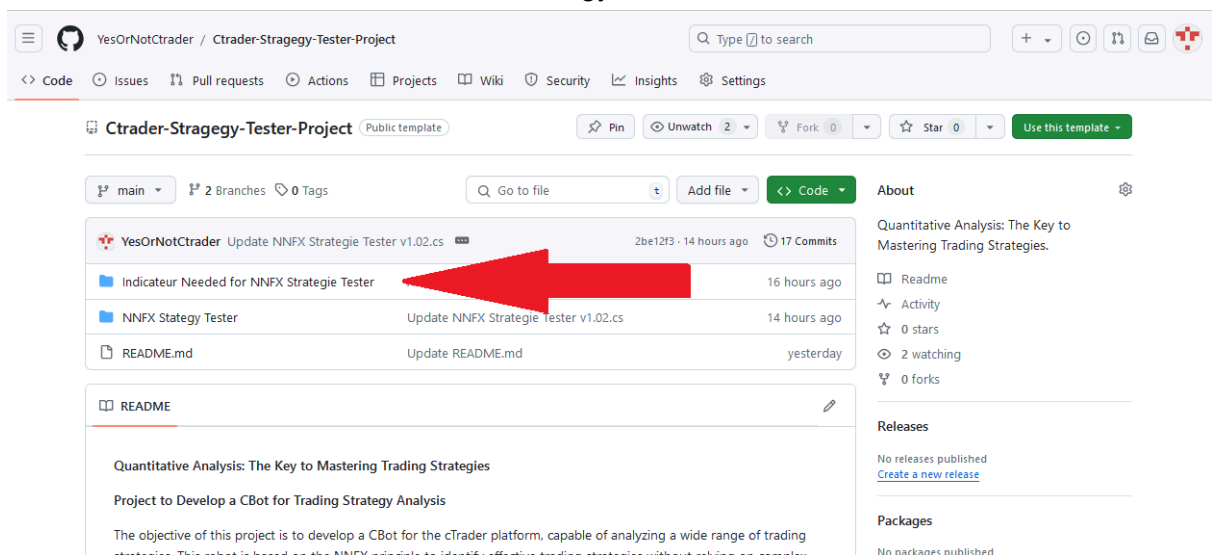
## II. Create With “.Cs “ Folder

1. Open the GitHub link. :

<https://github.com/YesOrNotCTrader/Ctrader-Strategy-Tester-Project>



2. Click on "Indicators Needed for NNFX Strategy Tester."



3. Now, for each folder named "ST\_...", click until you reach a file with the .cs extension.

YesOrNotCttrader / Ctrader-Strategy-Tester-Project

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Files

main

Go to file

Indicateur Needed for NNFX S...

- ST\_MfiMa
- ST\_RsiMa
- ST\_SuperTrend
  - ReadMe.txt
- NNFX Strategy Tester
  - README.md

Ctrader-Strategy-Tester-Project / Indicateur Needed for NNFX Strategie Tester /

Add file

YesOrNotCttrader Add files via upload e177d88 · 16 hours ago History

Name	Last commit message	Last commit date
..		
ST_MfiMa	Add files via upload	16 hours ago
ST_RsiMa	Add files via upload	16 hours ago
ST_SuperTrend	Add files via upload	16 hours ago
ReadMe.txt	Add files via upload	yesterday

YesOrNotCttrader / Ctrader-Strategy-Tester-Project

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Files

main

Go to file

Indicateur Needed for NNFX S...

- ST\_MfiMa
  - ST\_MfiMa.sln
- ST\_RsiMa
- ST\_SuperTrend
  - ReadMe.txt
- NNFX Strategy Tester
  - README.md

Ctrader-Strategy-Tester-Project / Indicateur Needed for NNFX Strategie Tester / ST\_MfiMa /

Add file

YesOrNotCttrader Add files via upload e177d88 · 16 hours ago History

Name	Last commit message	Last commit date
..		
ST_MfiMa	Add files via upload	16 hours ago
ST_MfiMa.sln	Add files via upload	16 hours ago

YesOrNotCttrader / Ctrader-Strategy-Tester-Project

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Files

main

Go to file

Indicateur Needed for NNFX S...

- ST\_MfiMa
  - ST\_MfiMa.cs
  - ST\_MfiMa.csproj
  - ST\_MfiMa.sln
- ST\_RsiMa
- ST\_SuperTrend
  - ReadMe.txt
- NNFX Strategy Tester
  - README.md

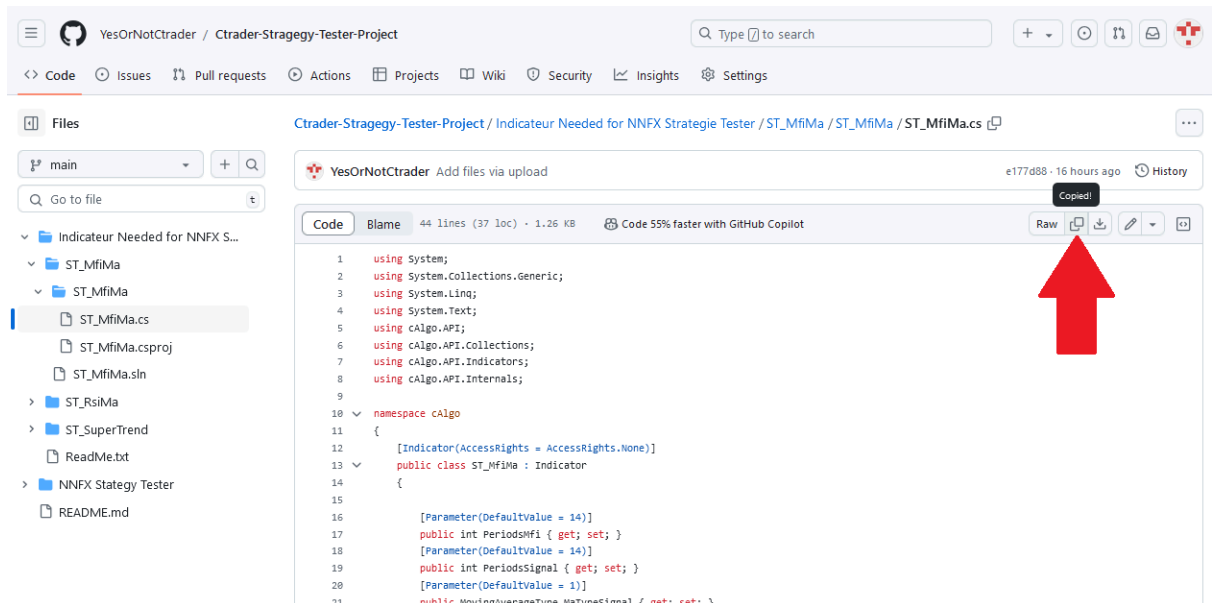
Ctrader-Strategy-Tester-Project / Indicateur Needed for NNFX Strategie Tester / ST\_MfiMa / ST\_MfiMa /

Add file

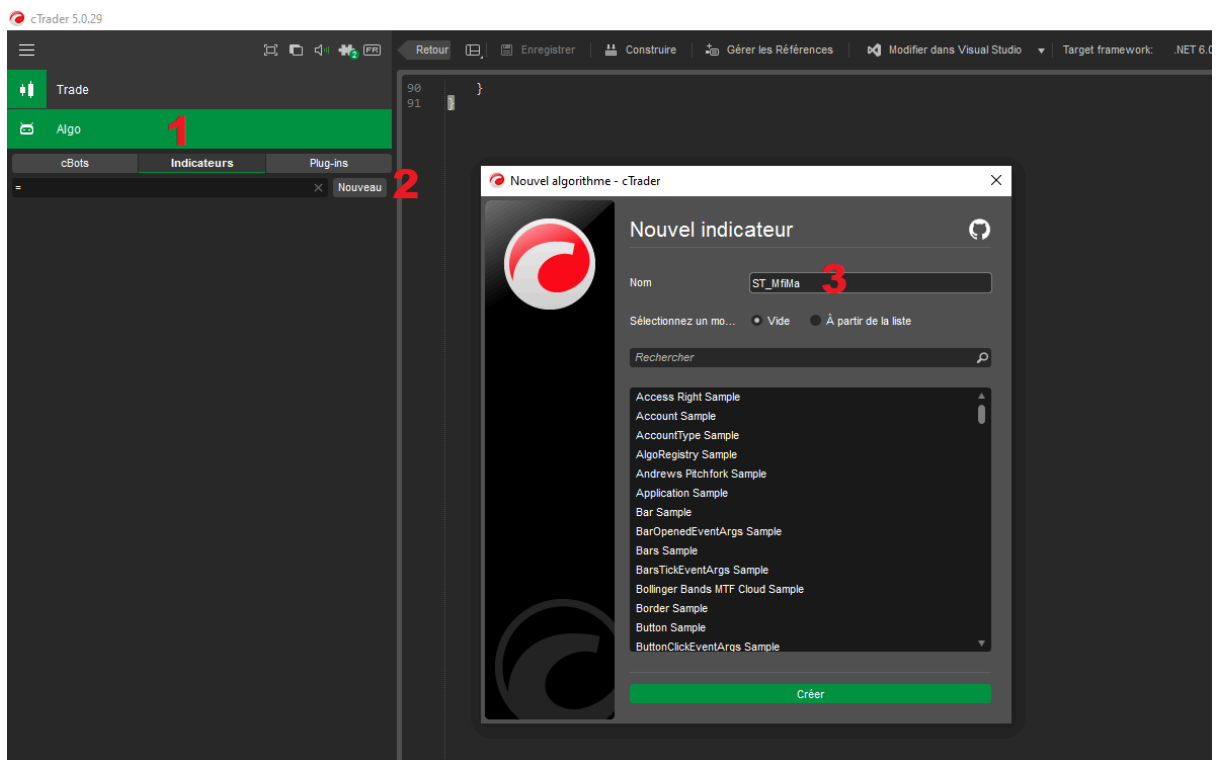
YesOrNotCttrader Add files via upload e177d88 · 15 hours ago History

Name	Last commit message	Last commit date
..		
ST_MfiMa.cs	Add files via upload	15 hours ago
ST_MfiMa.csproj	Add files via upload	15 hours ago

#### 4. Copy the code obtained



#### 5. Go to cTrader, navigate to the Algo section, click on Indicators, and then on New. Name the indicator (we recommend using the name listed in the folder), then click Create.



#### 6. Once the indicator is created, do the following:

- Click on the code

```
1  using System;
2  using cAlgo.API;
3  using cAlgo.API.Collections;
4  using cAlgo.API.Indicators;
5  using cAlgo.API.Internals;
6
7  namespace cAlgo
8  {
9      [Indicator(AccessRights = AccessRights.None)]
10     public class ST_MfiMal : Indicator
11     {
12         [Parameter(DefaultValue = "Hello world!")]
13         public string Message { get; set; }
14
15         [Output("Main")]
16         public IndicatorDataSeries Result { get; set; }
17
18         protected override void Initialize()
19         {
20             // To learn more about cTrader Automate visit our Help Center:
21             // https://help.ctrader.com/ctrader-automate
22
23             Print(Message);
24         }
25
26         public override void Calculate(int index)
27         {
28             // Calculate value at specified index
29             // Result[index] =
30         }
31     }
32 }
```

- Press **Ctrl + A** (to select all)

```
1 using System;
2 using cAlgo.API;
3 using cAlgo.API.Collections;
4 using cAlgo.API.Indicators;
5 using cAlgo.API.Internals;
6
7 namespace cAlgo
8 {
9     [Indicator(AccessRights = AccessRights.None)]
10    public class ST_MfiMa1 : Indicator
11    {
12        [Parameter(DefaultValue = "Hello world!")]
13        public string Message { get; set; }
14
15        [Output("Main")]
16        public IndicatorDataSeries Result { get; set; }
17
18        protected override void Initialize()
19        {
20            // To learn more about cTrader Automate visit our Help Center:
21            // https://help.citrader.com/citrader-automate
22
23            Print(Message);
24        }
25
26        public override void Calculate(int index)
27        {
28            // Calculate value at specified index
29            // Result[index] =
30        }
31    }
32 }
```

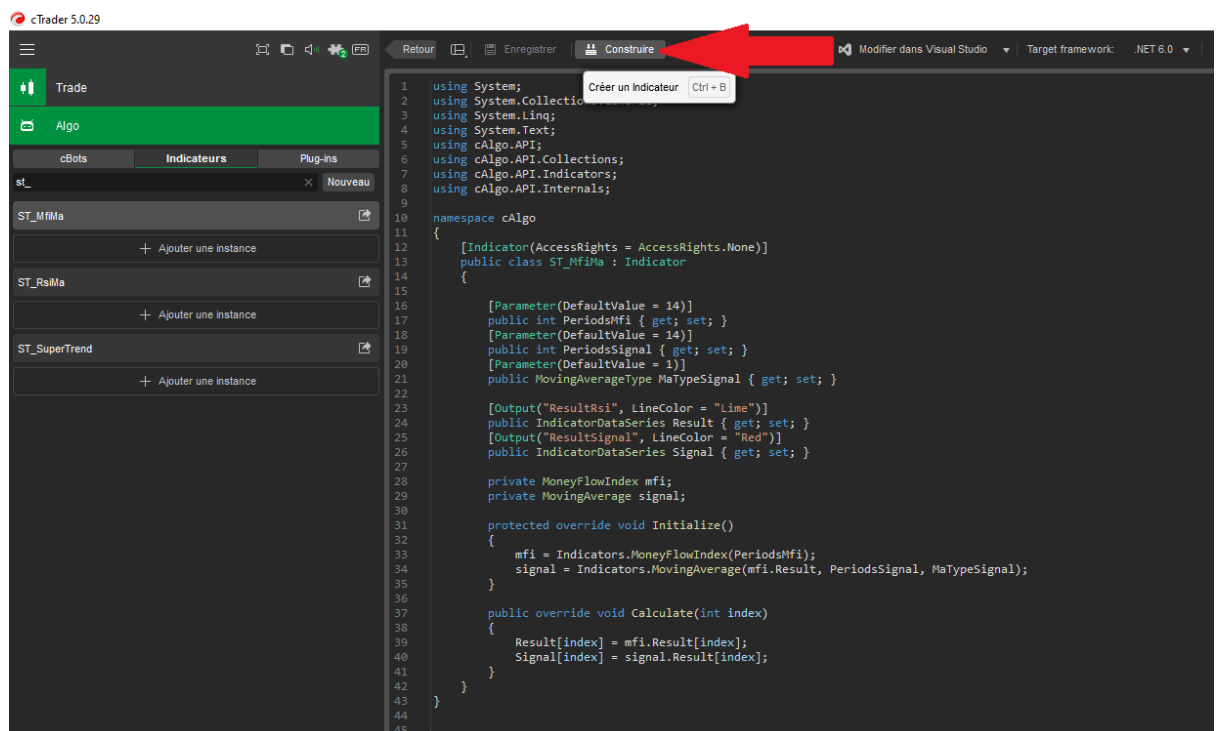
- Press **Ctrl + V** (to paste)

```

1  using System;
2  using System.Collections.Generic;
3  using System.Linq;
4  using System.Text;
5  using cAlgo.API;
6  using cAlgo.API.Collections;
7  using cAlgo.API.Indicators;
8  using cAlgo.API.Internals;
9
10 namespace cAlgo
11 {
12     [Indicator(AccessRights = AccessRights.None)]
13     public class ST_MfiMa : Indicator
14     {
15
16         [Parameter(DefaultValue = 14)]
17         public int PeriodsMfi { get; set; }
18         [Parameter(DefaultValue = 14)]
19         public int PeriodsSignal { get; set; }
20         [Parameter(DefaultValue = 1)]
21         public MovingAverageType MaTypeSignal { get; set; }
22
23         [Output("ResultRsi", LineColor = "Lime")]
24         public IndicatorDataSeries Result { get; set; }
25         [Output("ResultSignal", LineColor = "Red")]
26         public IndicatorDataSeries Signal { get; set; }
27
28         private MoneyFlowIndex mfi;
29         private MovingAverage signal;
30
31         protected override void Initialize()
32         {
33             mfi = Indicators.MoneyFlowIndex(PeriodsMfi);
34             signal = Indicators.MovingAverage(mfi.Result, PeriodsSignal, MaTypeSignal);
35         }
36
37         public override void Calculate(int index)
38         {
39             Result[index] = mfi.Result[index];
40             Signal[index] = signal.Result[index];
41         }
42     }
43 }
44
45

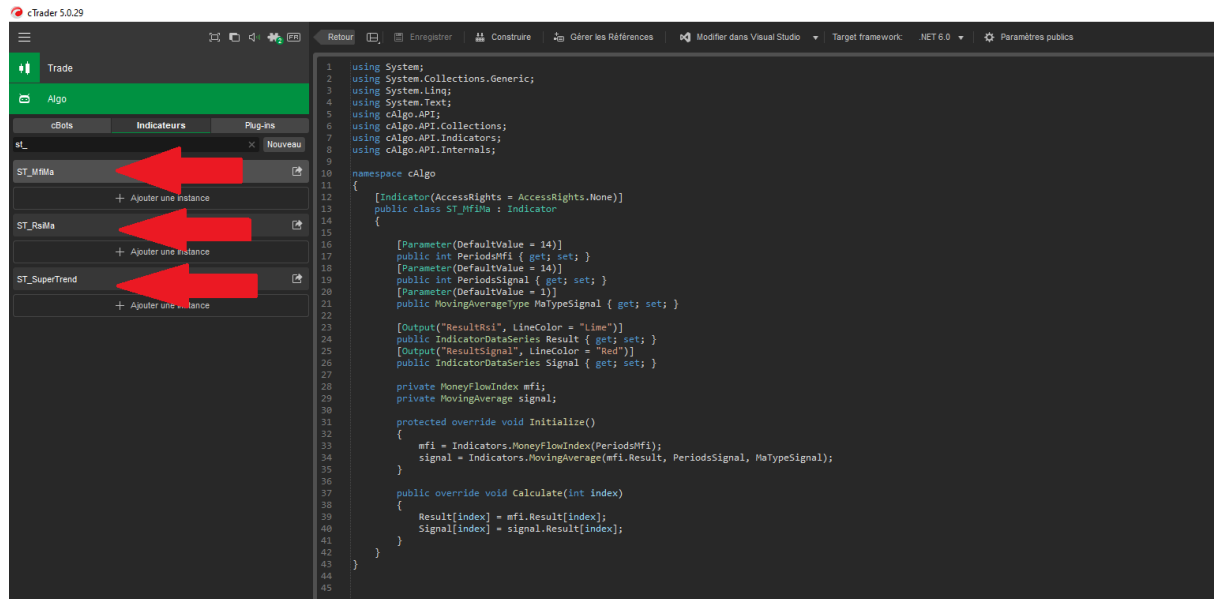
```

7. Finally, click Create. If you encounter an error, it's possible that the old indicator code was not cleared before you pasted the new code, so restart from step 4 and go directly to step 6.

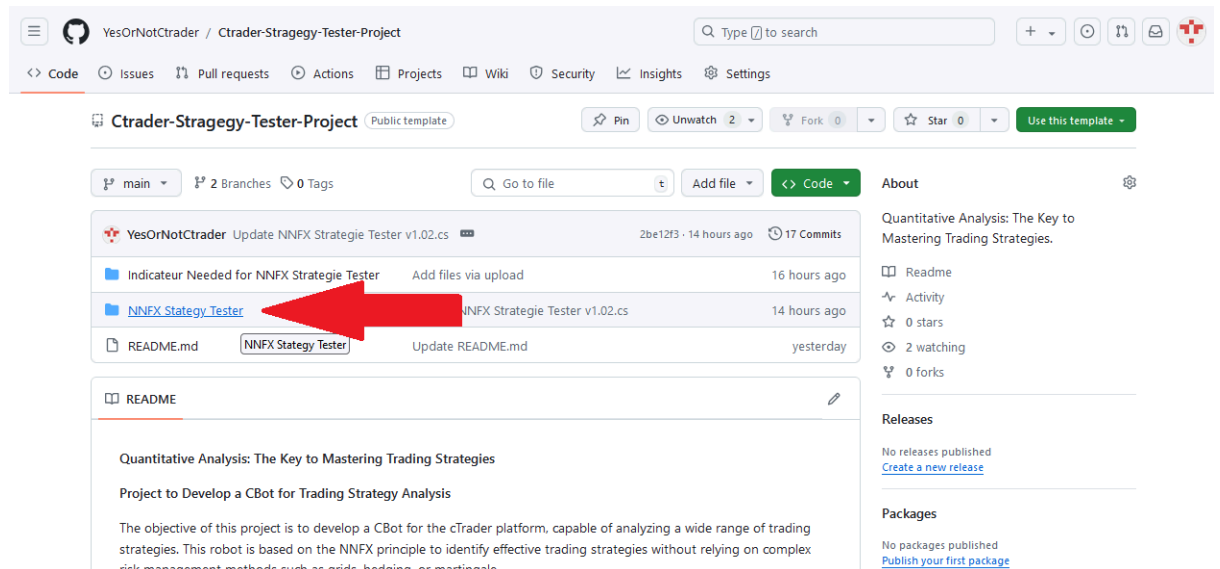




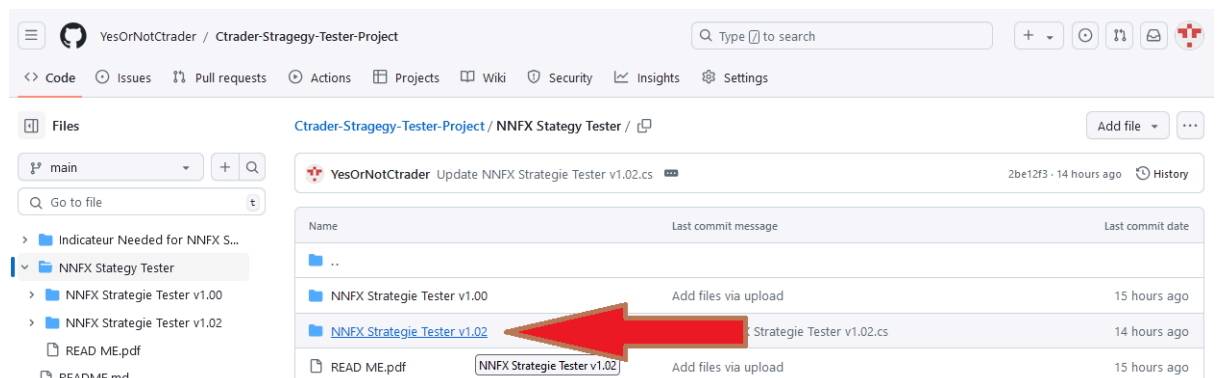
8. Repeat this process for all indicators named "ST\_...".

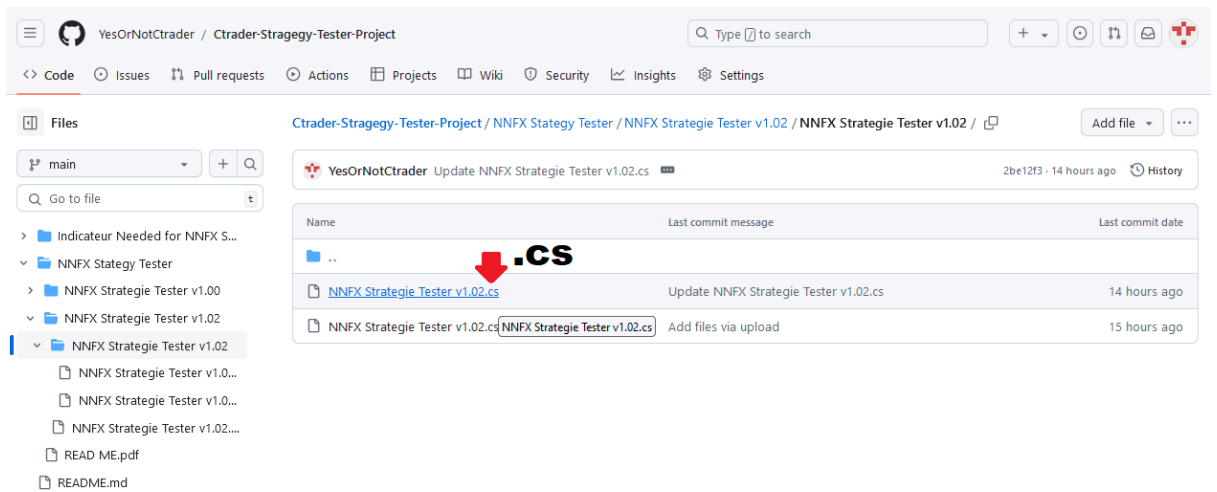
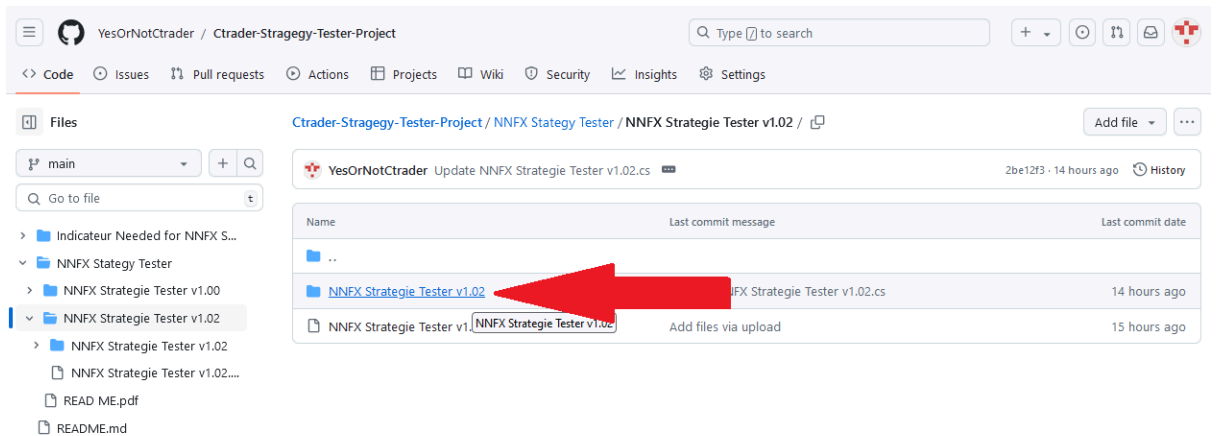


9. Once the indicators are created, click on "NNFX Strategy Tester".

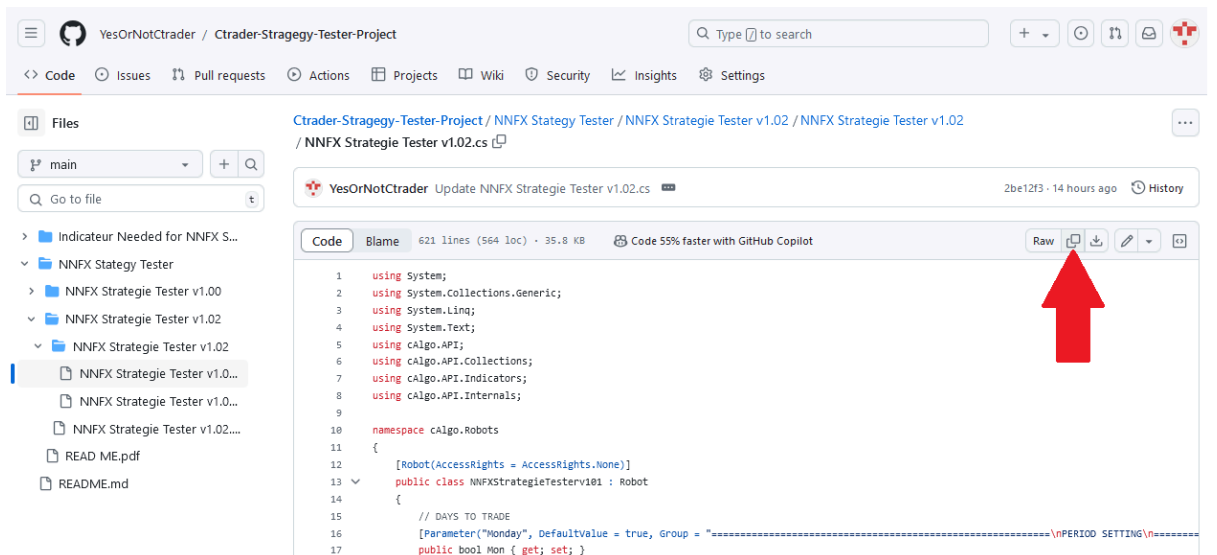


10. Now, select the latest version of "NNFX Strategy Tester v...". Click until you reach a file with the .cs extension.



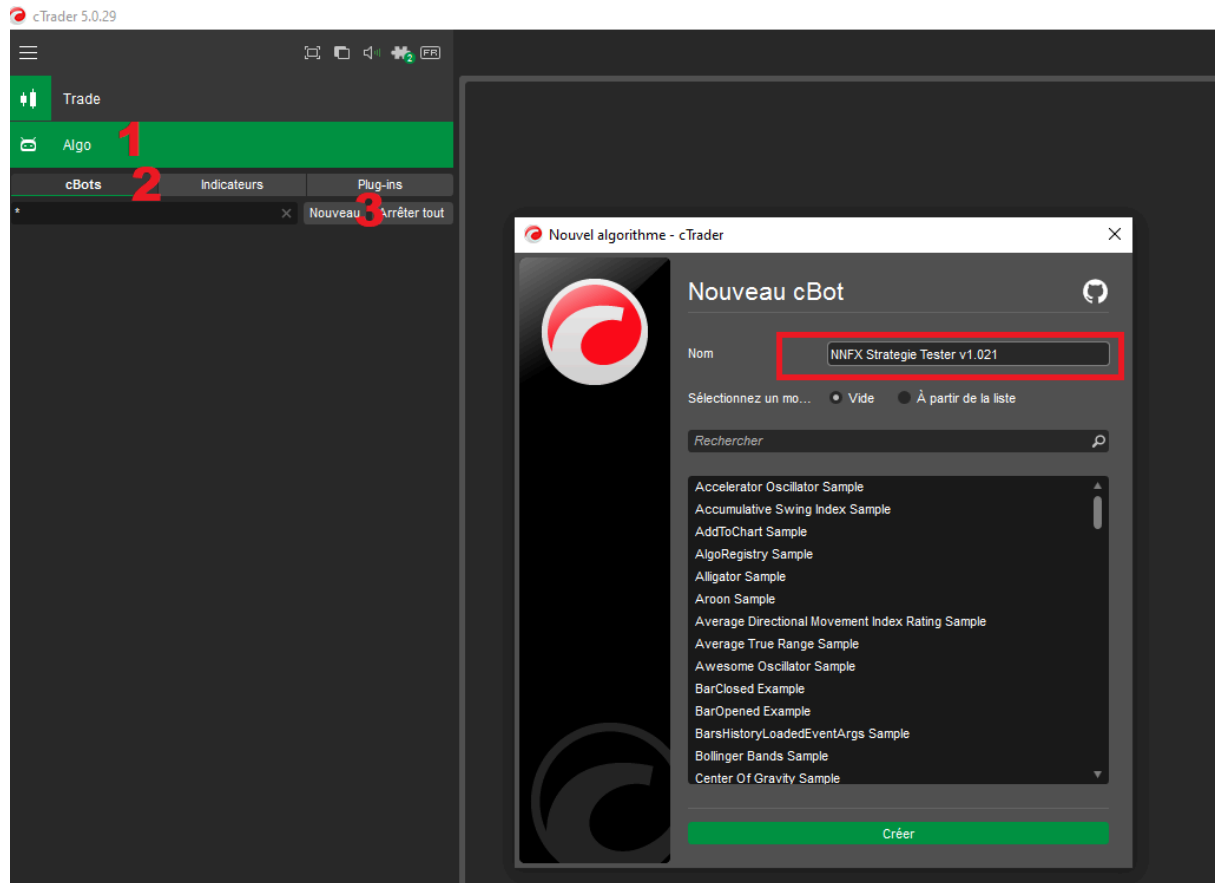


11. Copy the code obtained,



12. then go to cTrader, navigate to the Algo section, click on cBots, and then on New, Name the cBot (we recommend using the name listed in the folder), then click

Create.



13. Once the cBot is created, do the following:

- Click on code

```
1  using System;
2  using cAlgo.API;
3  using cAlgo.API.Collections;
4  using cAlgo.API.Indicators;
5  using cAlgo.API.Internals;
6
7  namespace cAlgo.Robots
8  {
9      [Robot(AccessRights = AccessRights.None, AddIndicators = true)]
10     public class NAFXStrategieTesterv1021 : Robot
11     {
12         [Parameter(DefaultValue = "Hello world!")]
13         public string Message { get; set; }
14
15         protected override void OnStart()
16         {
17             // To learn more about cTrader Automate visit our Help Center:
18             // https://help.ctrader.com/ctrader-automate
19
20             Print(Message);
21         }
22
23         protected override void OnTick()
24         {
25             // Handle price updates here
26         }
27
28         protected override void OnStop()
29         {
30             // Handle cBot stop here
31         }
32     }
33 }
```

- Press **Ctrl + A** (to select all)

```
1  using System;
2  using cAlgo.API;
3  using cAlgo.API.Collections;
4  using cAlgo.API.Indicators;
5  using cAlgo.API.Internals;
6
7  namespace cAlgo.Robots
8  {
9      [Robot(AccessRights = AccessRights.None, AddIndicators = true)]
10     public class NNFXStrategieTesterv1021 : Robot
11     {
12         [Parameter(DefaultValue = "Hello world!")]
13         public string Message { get; set; }
14
15         protected override void OnStart()
16         {
17             // To learn more about cTrader Automate visit our Help Center:
18             // https://help.ctrader.com/ctrader-automate
19
20             Print(Message);
21         }
22
23         protected override void OnTick()
24         {
25             // Handle price updates here
26         }
27
28         protected override void OnStop()
29         {
30             // Handle cBot stop here
31         }
32     }
33 }
```

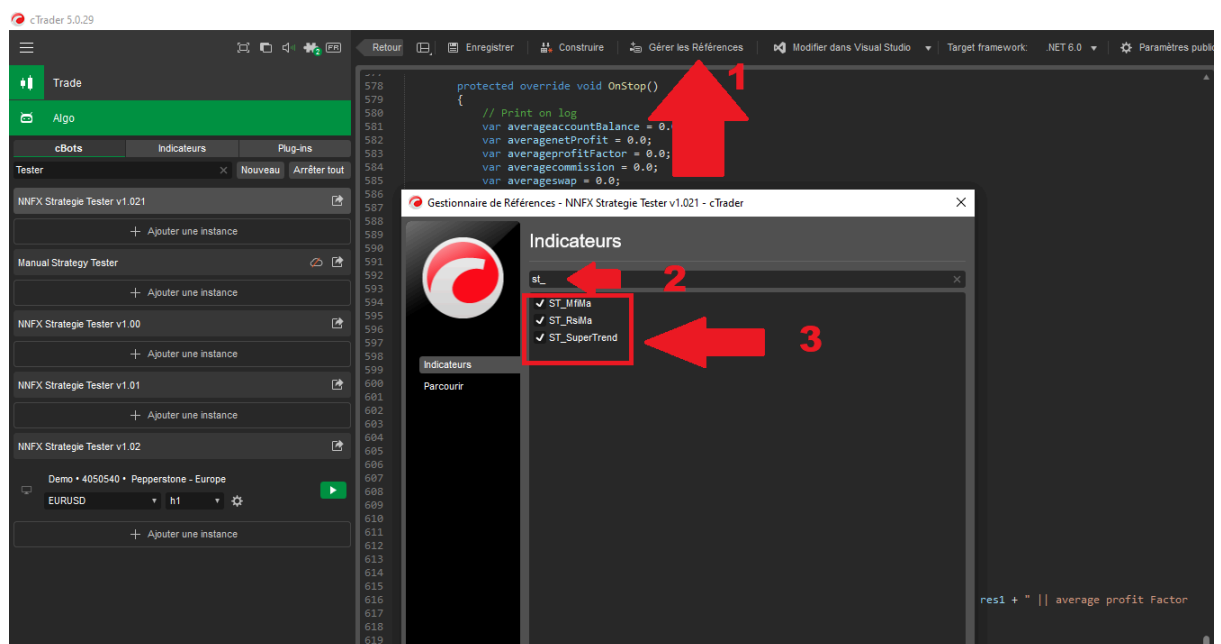
- Press **Ctrl + V** (to paste)

```

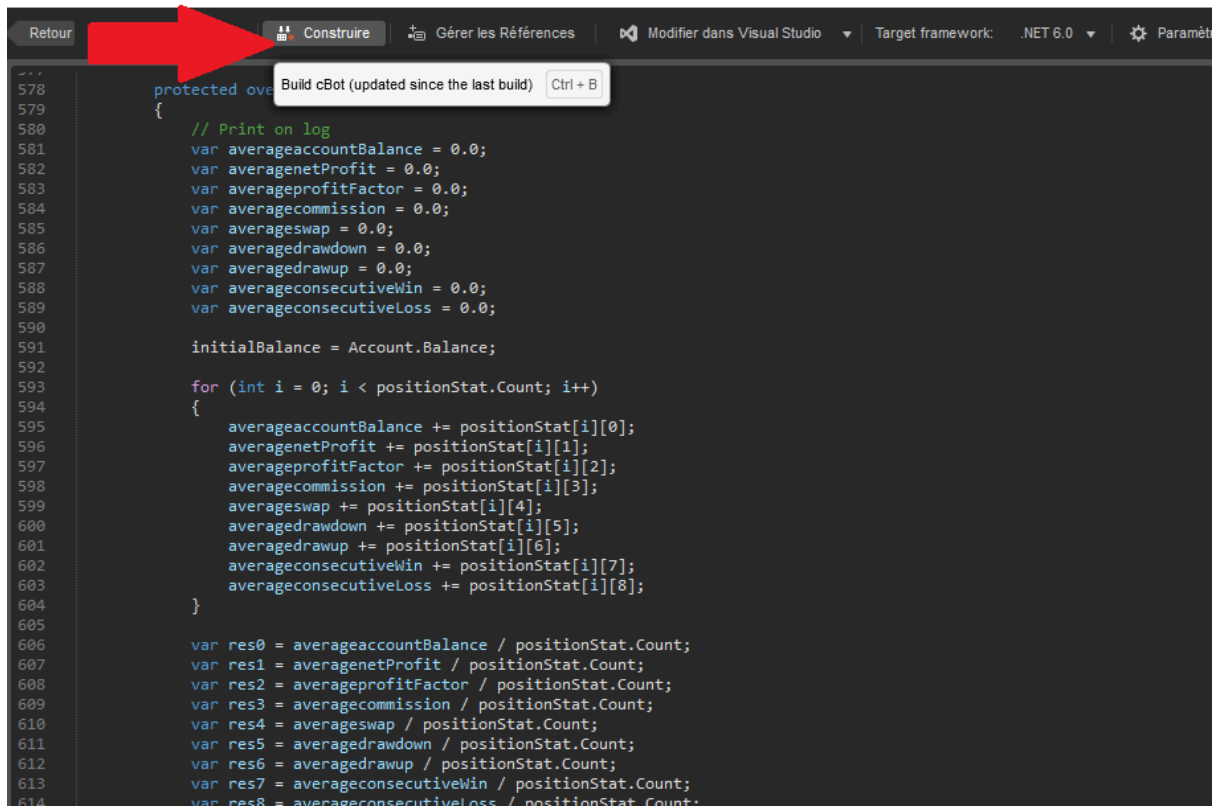
577
578     protected override void OnStop()
579     {
580         // Print on log
581         var averageaccountBalance = 0.0;
582         var averagenetProfit = 0.0;
583         var averageprofitFactor = 0.0;
584         var averagecommission = 0.0;
585         var averageswap = 0.0;
586         var averagedrawdown = 0.0;
587         var averagedrawup = 0.0;
588         var averageconsecutiveWin = 0.0;
589         var averageconsecutiveLoss = 0.0;
590
591         initialBalance = Account.Balance;
592
593         for (int i = 0; i < positionStat.Count; i++)
594         {
595             averageaccountBalance += positionStat[i][0];
596             averagenetProfit += positionStat[i][1];
597             averageprofitFactor += positionStat[i][2];
598             averagecommission += positionStat[i][3];
599             averageswap += positionStat[i][4];
600             averagedrawdown += positionStat[i][5];
601             averagedrawup += positionStat[i][6];
602             averageconsecutiveWin += positionStat[i][7];
603             averageconsecutiveLoss += positionStat[i][8];
604         }
605
606         var res0 = averageaccountBalance / positionStat.Count;
607         var res1 = averagenetProfit / positionStat.Count;
608         var res2 = averageprofitFactor / positionStat.Count;
609         var res3 = averagecommission / positionStat.Count;
610         var res4 = averageswap / positionStat.Count;
611         var res5 = averagedrawdown / positionStat.Count;
612         var res6 = averagedrawup / positionStat.Count;
613         var res7 = averageconsecutiveWin / positionStat.Count;
614         var res8 = averageconsecutiveLoss / positionStat.Count;
615
616         Print("average account Balance : " + res0 + " || average net Profit : " + res1 + " || average profit Factor
617
618         // Export to csv need Function need idea for good using
619     }
620 }
621

```

14. Now that you have the code, click on Manage References. Select all the indicators necessary for this robot. Since we named them "ST\_", enter "ST\_" in the search bar and select all the listed indicators.



15. Finally, click Create. If you encounter an error, it's possible that the old indicator code was not cleared before you pasted the new code, so restart from step 4 and go directly to step 6.



```
578  
579  
580  
581 // Print on log  
582 var averageaccountBalance = 0.0;  
583 var averagenetProfit = 0.0;  
584 var averageprofitFactor = 0.0;  
585 var averagecommission = 0.0;  
586 var averageswap = 0.0;  
587 var averagedrawdown = 0.0;  
588 var averagedrawup = 0.0;  
589 var averageconsecutiveWin = 0.0;  
590 var averageconsecutiveLoss = 0.0;  
591  
592 initialBalance = Account.Balance;  
593  
594 for (int i = 0; i < positionStat.Count; i++)  
595 {  
596     averageaccountBalance += positionStat[i][0];  
597     averagenetProfit += positionStat[i][1];  
598     averageprofitFactor += positionStat[i][2];  
599     averagecommission += positionStat[i][3];  
600     averageswap += positionStat[i][4];  
601     averagedrawdown += positionStat[i][5];  
602     averagedrawup += positionStat[i][6];  
603     averageconsecutiveWin += positionStat[i][7];  
604     averageconsecutiveLoss += positionStat[i][8];  
605 }  
606  
607 var res0 = averageaccountBalance / positionStat.Count;  
608 var res1 = averagenetProfit / positionStat.Count;  
609 var res2 = averageprofitFactor / positionStat.Count;  
610 var res3 = averagecommission / positionStat.Count;  
611 var res4 = averageswap / positionStat.Count;  
612 var res5 = averagedrawdown / positionStat.Count;  
613 var res6 = averagedrawup / positionStat.Count;  
614 var res7 = averageconsecutiveWin / positionStat.Count;  
615 var res8 = averageconsecutiveLoss / positionStat.Count;
```

**Congratulations, you did it!**