

## Damian Przesław

Aplikacja monitorująca przepływ środków na giełdę kryptowalut.

### 1. Wstęp

Niniejsza dokumentacja opisuje projekt aplikacji, której zadaniem jest monitorowanie wpływów i wypływów Eth oraz Usdt i Usdc z giełd kryptowalutowych, a następnie przesyłanie zebranych informacji na telefon.

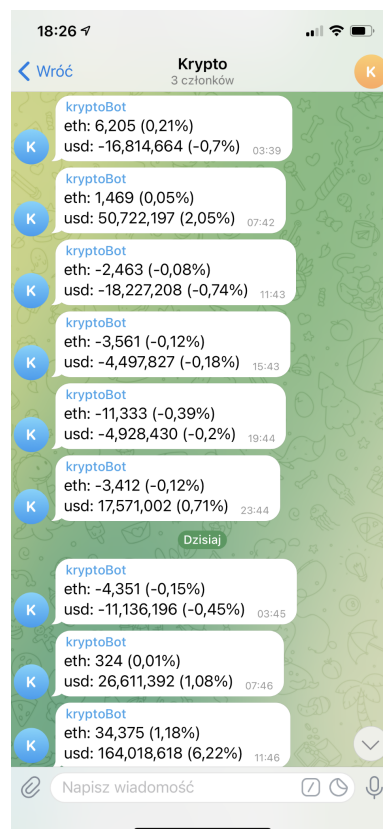
Program został napisany w języku C# w środowisku Microsoft Visual Studio 2019 z wykorzystaniem Windows Form Application na platformie .NET Core

### 2. Informacje o programie

Program pobiera dane o ilości Eth, Usdt, Usdc z portfeli trzech giełd scentralizowanych (Binance, Coinbase, Kucoin) z ethscan.io a następnie sumuje poszczególne wartości i porównuje z poprzednim sprawdzeniem. Program za pomocą bota Telegram wysyła informacje na telefon o zmianie ilości Eth i Usdt. W końcowym etapie wysyła dane do bazy danych.

### 3. Obsługa programu

Gdy program jest włączony działa przez cały czas wysyłając zebrane informacje na telefon, informacje te można w każdej chwili zobaczyć w aplikacji Telegram w rozmowie z botem.



## 4. Główna klasa Program

### Główne wykonanie programu

```
9      class Program
10     {
11
12         Odwołania: 0
13         static async Task Main(string[] args)
14         {
15             connect a = new connect(); // tworzy połączenie z klientem http ethscan
16             Api_adress address_base = new Api_adress(); // tworzy adresy zapytan
17             Bot bot = new Bot(); // tworzy bota telegram
18             Values w = new Values(); // wartosci sum danych pobranych
19             Date_base db = new Date_base(); //połączenie z baza danych
20
21             double temp_usdc = 0; // zmienna do liczenia sumy usd
22             double temp_eth = 0; // zmienna do liczenia sumy eth
23             double temp_usd = 0; // zmienna do liczenia sumy usd
24
25             w.suma_eth_p = db.Pobranie_ostatniej_wartosci_eth(); // pobranie ostatniej zapisanej wartosci sumy eth z bazy danych
26             w.suma_usd_p = db.Pobranie_ostatniej_wartosci_usd(); // pobranie ostatniej zapisanej wartosci sumy usd z bazy danych
27
28             while (true) // zapewnia ciagle dzialanie programu
29             {
30                 try
31                 {
32                     for (int i = 0; i < 30; i++) // wysylanie zapytan do kolejnych adresow portweli
33                     {
34                         Console.WriteLine(i);
35                         temp_eth = await a.ProcessRepositories_eth(address_base.get_eth_b(i)); // pobranie ilosci eth z portfela
36                         temp_usd = await a.ProcessRepositories_usd(address_base.get_usdt_b(i)); // pobranie ilosci usdt z portfela
37                         temp_usdc = await a.ProcessRepositories_usd(address_base.get_usdc_b(i)); // pobranie ilosci usdc z portfela
38
39                         w.suma_usd = w.suma_usd + temp_usd + temp_usdc; //sumowanie usd
40                         w.suma_eth = w.suma_eth + temp_eth; //sumowanie eth
41
42                         Task.Delay(600).Wait(); // opoznienie zapytania aby nie przekroczyc 5 na 1 sek.
43                     }
44
45                     Console.WriteLine("eth: " + w.suma_eth);
46                     Console.WriteLine("usd: " + w.suma_usd);
47                 }
48                 catch (Exception e) // wylapanie bledu podczas zapytan
49                 {
50                     System.Console.WriteLine(e);
51                 }
52
53                 DateTime thisHour = DateTime.Now; //pobranie czasu
54                 Console.WriteLine(thisHour.ToString() + " "); //wypisanie czasu
55
56                 try
57                 {
58                     w.cena_eth = await a.ProcessRepositories_eth_price(adress_base.get_cena_eth()); // pobranie ceny eth
59                 }
60                 catch (Exception e) // wylapanie bledu podczas zapytania o cene eth
61                 {
62                     Console.WriteLine(e);
63                 }
64
65                 bot.send(w); // wyslanie danych na teleram
66
67                 db.EthToDateBase(w); // wysłanie danych o eth do bazy danych
68                 db.UsdToDateBase(w); // wysłanie danych o usd do bazy danych
69
70                 w.next(); // zapisanie sumy jako poprzedniej i czyszczenie
71
72                 Task.Delay(240 * 60 * 1000).Wait(); //opoznienie do nastepnego pobrania danych za 4h
73             }
74         }
75     }
76 }
77
78 }
```

## 5. Klasy programu

### 5.1. Api\_address

Klasa posiada potrzebne części adresów do zapytań, które tworzy i zwraca za pomocą metod.

```
9      Odwołania: 3
10      class Api_address //tworzy adresy zapytan
11      {
12          string[] adres_wallet_b = new string[30]; //adresy binance na sieci eth
13
14          string adres_base = "https://api.etherscan.io/api"; //adres bazowy
15          string check_balance = "?module=account&action=tokenbalance"; //sprawdzenie ilosci tokenow
16
17          string eth_price = "?module=stats&action=ethprice"; //sprawdzenie ceny eth
18
19          string adres_usdt = "&contractaddress=0xdac17f958d2ee523a2206206994597c13d831ec7"; //adres usdt
20          string adres_usdc = "&contractaddress=0xa0b86991c6218b36c1d19d4a2e9eb0ce3606eb48"; // adres usdc
21
22          string adres_eth = "?module=account&action=balance"; // wartosc eth na koncie
23
24          string api_key = "&apikey=*****"; //api key
25
26          1 odwołanie
27          public Api_address()
28          {
29              ////////////////////////////////////////////////// binance
30              adres_wallet_b[0] = "&address=0x3f5CE5F8Fe3E9af3971dD833D26bA9b5C936f0bE&tag=latest";
31              adres_wallet_b[1] = "&address=0xD551234Ae421e3BCBA99A0Da6d736074f22192FF&tag=latest";
32              adres_wallet_b[2] = "&address=0x564286362092D08e7936f0549571a8038203aAceD&tag=latest";
33              adres_wallet_b[3] = "&address=0x0681d8Db095565FE8A346fA0277bFfdE9C0eDBBF&tag=latest";
34              adres_wallet_b[4] = "&address=0xFe9e8709d3215310075d67e3ed32a380ccf451c8&tag=latest";
35              adres_wallet_b[5] = "&address=0x4e9ce36e442e55ecd9025b9a6e0d88485d628a67&tag=latest";
36              adres_wallet_b[6] = "&address=0xb0eb53f46cd790cd13851d5eff43d12404d33e8&tag=latest";
37              adres_wallet_b[7] = "&address=0xf977814e90da44bfa03b6295a0616a897441acac&tag=latest";
38              adres_wallet_b[8] = "&address=0x001866ae5b3de6caa5a51543fd9fb64f524f5478&tag=latest";
39              adres_wallet_b[9] = "&address=0x85b931a32a0725be14285b66f1a22178c672d69b&tag=latest";
40              adres_wallet_b[10] = "&address=0x708396f17127c42383e3b9014072679b2f60b82f&tag=latest";
41              adres_wallet_b[11] = "&address=0xe0f0cfde7ee664943906f17f7f14342e76a5cec7&tag=latest";
42              adres_wallet_b[12] = "&address=0x8f22f2063d253846b53609231ed80fa571bc0c8f&tag=latest";
43              adres_wallet_b[13] = "&address=0x28C6c06298d514Db089934071355E5743bf21d60&tag=latest";
44              adres_wallet_b[14] = "&address=0x21a31Ea1afC51d94C2eFcCAa2092aD1028285549&tag=latest";
45              adres_wallet_b[15] = "&address=0xDFd5293D8e347dFe59E90eFd55b2956a1343963d&tag=latest";
46              adres_wallet_b[16] = "&address=0x56Eddb7aa87536c09CCc2793473599fD21A8b17F&tag=latest";
47
48              //////////////////////////////////////////////////coinbase
49              adres_wallet_b[17] = "&address=0x9696f59E4d72E2378E84fFD425DCaD1548f96976&tag=latest";
50              adres_wallet_b[18] = "&address=0x4D9fF50EF4dA9473648B9650892B2554e78E5E28&tag=latest";
51              adres_wallet_b[19] = "&address=0x49764A402f38326660D17bf34b431dC6e2eb2327&tag=latest";
52              //////////////////////////////////////////////////kucoin
53              adres_wallet_b[20] = "&address=0x71660c4005ba85c37ccec55d0c4493e66fe775d3&tag=latest";
54              adres_wallet_b[21] = "&address=0x503828976d22510aad0201ac7ec88293211d23da&tag=latest";
55              adres_wallet_b[22] = "&address=0x8ddfabcd4d8ffcd5beaf154f18b778f892a0740&tag=latest";
56              adres_wallet_b[23] = "&address=0x3cd751e6b0078be393132286c442345e5dc49699&tag=latest";
57              adres_wallet_b[24] = "&address=0xb5d85cbf7cb3ee0d56b3bb207d5fc4b82f43f511&tag=latest";
58              adres_wallet_b[25] = "&address=0xeb2629a2734e272bcc07bda959863f316f4b4dcf&tag=latest";
59              adres_wallet_b[26] = "&address=0x2b5634c42055806a59e9107ed44d43c426e58258&tag=latest";
60              adres_wallet_b[27] = "&address=0x689c56aef474df92d44a1b70850f808488f9769c&tag=latest";
61              adres_wallet_b[28] = "&address=0xa1d8d972560c2f8144af871db508f0b0b10a3fbf&tag=latest";
62              adres_wallet_b[29] = "&address=0x4ad64983349c49defe8d7a4686202d24b25d0ce8&tag=latest";
63          }
64
65          1 odwołanie
66          public string get_usdt_b(int i) //zwraca adres pobrania ilosc usdt
67          {
68              return adres_base + check_balance + adres_usdt + adres_wallet_b[i] + api_key;
69          }
70
71          1 odwołanie
72          public string get_eth_b(int i) //zwraca adres pobrania ilosci eth
73          {
74              return adres_base + adres_eth + adres_wallet_b[i] + api_key;
75          }
76
77          1 odwołanie
78          public string get_usdc_b(int i) //zwraca adres pobrania ilosci usdc
79          {
80              return adres_base + check_balance + adres_usdc + adres_wallet_b[i] + api_key;
81          }
82
83          1 odwołanie
84          public string get_cena_eth() //zwraca adres pobrania ceny eth
85          {
86              return adres_base + eth_price + api_key;
87          }
88      }
```

## 5.2. Bot

Klasa Bot tworzy bota w aplikacji telegram, który wysyła dane na telefon.

```
8 class Bot //bot wysylajacy wiadomosci na telegram
9 {
10     public TelegramBotClient Client;
11     string api_key; //api_key bot
12     int id; // id rozmowy
13
14     1 odwołanie
15     public Bot()
16     {
17         api_key = "*****"; //apki key bot telegram
18         id = 123456; //id grupy
19
20         Client = new TelegramBotClient(api_key); //polaczenie z botem telegram
21     } 2
22     1 odwołanie
23     public void send(Values w) //metoda wyslanie wiadomosci na telegram
24     {
25         double procent_eth = w.get_delta_eth() / w.suma_eth; //obliczenie procentow zmiany eth
26         double procent_usd = w.get_delta_usd() / w.suma_usd; // obiczanie procent zmiany usd
27
28         procent_eth = Math.Round(procent_eth * 100, 2); //zaokraglenie wartosci procentowej eth
29         procent_usd = Math.Round(procent_usd * 100, 2); //zaokraglenie wartosci procentowej usd
30
31         Client.SendTextMessageAsync(id, "eth: " + w.get_delta_eth_to_write() + " (" + procent_eth + "%)\n" + "usd: " + w.get_delta_usd_to_write() + " (" + procent_usd + "%)"); //wyslanie wiadomosci
32     }
33
34
35
36 }
```

## 5.3. Connect

Klasa tworzy połączenie z klientem http ethscan.io i za pomoca adresów stworzonych przez klase Api\_adress wysyła zapytania i odbiera rezultaty.

```
12 class connect
13 {
14     private HttpClient client = new HttpClient(); // utworzenie polaczenia z klientem http
15
16     1 odwołanie
17     public connect()
18     {
19         client.DefaultRequestHeaders.Accept.Clear();
20         client.DefaultRequestHeaders.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json")); //ustawienie oczekianej wiadomosci zwrotnej
21     }
22     2 odwołania
23     public async Task<double> ProcessRepositories_usd(string adress) // metoda pobrania wartosci usd portfela
24     {
25         var stringTask = client.GetStringAsync(adress); //wyslanie zapytania
26         Values_json d = JsonSerializer.Deserialize<Values_json>(await stringTask); // odebranie zapytania
27         return d.ToDouble_usd();
28     }
29     1 odwołanie
30     public async Task<double> ProcessRepositories_eth(string adress) // metoda pobrania wartosci eth portfela
31     {
32         var stringTask = client.GetStringAsync(adress); //wyslanie zapytania
33         Values_json d = JsonSerializer.Deserialize<Values_json>(await stringTask); //odebranie zapytania
34         return d.ToDouble_eth();
35     }
36     1 odwołanie
37     public async Task<double> ProcessRepositories_eth_price(string adress) //metoda pobrania ceny eth
38     {
39         var stringTask = client.GetStringAsync(adress); //wyslanie zapytania
40         Price_json d = JsonSerializer.Deserialize<Price_json>(await stringTask); //obranie zapytania
41         return d.ToDouble_cena_eth();
42     }
43 }
```

## 5.4. Date\_base

Klasa odpowiada za komunikację z bazą danych.

```
10 class Date_base
11 {
12     private MySqlConnectionStringBuilder builder = new MySqlConnectionStringBuilder // dane logowania do bazy danej
13     {
14         Server = "*****",
15         Database = "*****",
16         UserID = "*****",
17         Password = "*****",
18     };
19
20     1 odwołanie
21     public void EthToDateBase(Values w) // metoda wysyla wartosci eth do bazy danej
22     {
23         MySqlConnection connect = new MySqlConnection(builder.ConnectionString); //tworzy polaczenie z baza danych
24
25         DateTime thisHour = DateTime.Now; // pobranie czasu
26
27         string s = string.Format( // tworzy tresc komendy sql do wstawienia wiersza do bazy
28             "INSERT INTO Eth (dzien,godzina,suma,delta,cena) VALUES('{0}','{1}', {2}, {3},{4});",
29             thisHour.ToString("yyyy-MM-dd"),
30             thisHour.ToString("HH:mm"),
31             Math.Round(w.suma_eth),
32             Math.Round(w.get_delta_eth()),
33             w.get_eth_cena()
34         );
35
36         try
37         {
38             Console.WriteLine("eth ->");
39             connect.Open(); // otwarcie polaczenia z baza danych
40             MySqlCommand komenda = connect.CreateCommand(); // stworzenie komendy sql
41             komenda.CommandText = s; // przypisanie tresci komendy
42             // komenda.ExecuteReader(); // wykonanie komendy sql
43             Console.WriteLine("done");
44         }
45         catch (Exception e) // wypisanie bledu z baza danych
46         {
47             Console.WriteLine("nie udane");
48             Console.WriteLine(e);
49         }
50     }
51     1 odwołanie
52     public void UsdToDateBase(Values w) // metoda wysyla wartosci usd do bazy danej
53     {
54         MySqlConnection connect = new MySqlConnection(builder.ConnectionString); //tworzy polaczenie z baza danych
55
56         DateTime thisHour = DateTime.Now; // pobranie czasu
57
58         string s = string.Format( // tworzy tresc komendy sql do wstawienia wiersza do bazy
59             "INSERT INTO Usd (dzien,godzina,suma,delta) VALUES('{0}','{1}', {2}, {3});",
60             thisHour.ToString("yyyy-MM-dd"),
61             thisHour.ToString("HH:mm"),
62             Math.Round(w.suma_usd),
63             Math.Round(w.get_delta_usd())
64         );
65
66         try
67         {
68             Console.WriteLine("usd ->");
69             connect.Open(); // otwarcie polaczenia z baza danych
70             MySqlCommand komenda = connect.CreateCommand(); // stworzenie komendy sql
71             komenda.CommandText = s; // przypisanie tresci komendy
72             // komenda.ExecuteReader(); // wykonanie komendy sql
73             Console.WriteLine("done");
74         }
75         catch (Exception e) // wypisanie bledu z baza danych
76         {
77             Console.WriteLine("nie udane");
78             Console.WriteLine(e);
79         }
80     }
}
```

```

81 public double Pobranie_ostatniej_wartosci_usd() //pobiera ostatnia wartosc sumy usd
82 {
83     try
84     {
85         MySqlConnection connect = new MySqlConnection(builder.ConnectionString); //tworzy polaczenie z baza danych
86         connect.Open(); // otwarcie polaczenia z baza danych
87
88         MySqlCommand komenda = connect.CreateCommand(); // stworzenie komendy sql
89         komenda.CommandText = "SELECT suma FROM Usd ORDER BY id DESC LIMIT 1;"; //komenda pobrania ostatniej sumy usd
90
91         MySqlDataReader czytnik = komenda.ExecuteReader(); //wykoanie komendy i owatcie czytnika
92         czytnik.Read(); // pobranie wartosci
93
94         return czytnik.GetInt64(0);
95     }
96     catch (Exception e)
97     {
98         Console.WriteLine("nie udane pobranie wartosci usd z bazy danych");
99         return 0;
100     }
101 }
102

```

```

104 public double Pobranie_ostatniej_wartosci_eth() //pobiera ostatnia wartosc sumy usd
105 {
106     try
107     {
108         MySqlConnection connect = new MySqlConnection(builder.ConnectionString); //tworzy polaczenie z baza danych
109         connect.Open(); // otwarcie polaczenia z baza danych
110
111         MySqlCommand komenda = connect.CreateCommand(); // stworzenie komendy sql
112         komenda.CommandText = "SELECT suma FROM Eth ORDER BY id DESC LIMIT 1;"; //komenda pobrania ostatniej sumy eth
113
114         MySqlDataReader czytnik = komenda.ExecuteReader(); //wykoanie komendy i owatcie czytnika
115         czytnik.Read(); // pobranie wartosci
116
117         return czytnik.GetInt32(0);
118     }
119     catch (Exception e)
120     {
121         Console.WriteLine("nie udane pobranie wartosci eth z bazy danych");
122         return 0;
123     }
124 }
125
126 }
127
128 }
129

```

## 5.5. Prince\_json

Klasa odpowiada wiadomości zwrotnej o cenie eth w formacie json.

```

8 namespace krypto
9 {
10     Odwołania: 2
11     public class Price_json // klasa odpowiadajaca pobraniu odpowiedzi json o cenie eth
12     {
13         Odwołania: 0
14         public string status { get; set; }
15         Odwołania: 0
16         public string message { get; set; }
17         1 odwołanie
18         public Result result { get; set; }
19
20         1 odwołanie
21         public double ToDouble_cena_eth()
22         {
23             return double.Parse(result.ethusd, CultureInfo.InvariantCulture.NumberFormat); // zwrot wartosci ceny eth w double
24         }
25
26         1 odwołanie
27         public class Result
28         {
29             Odwołania: 0
30             public string ethbtc { get; set; }
31             Odwołania: 0
32             public string ethbtc_timestamp { get; set; }
33             1 odwołanie
34             public string ethusd { get; set; }
35             Odwołania: 0
36             public string ethusd_timestamp { get; set; }
37         }
38     }
39 }

```

## 5.6. Values

Struktura służy do przechowywania potrzebnych wartości oraz podstawowego przetwarzania do odpowiedniego formatu.

```
10 struct Values //przechowuje podstawowe wartosci
11 {
12     public double suma_eth;
13     public double suma_usd;
14     public double suma_usd_p; //poprzednia suma usd
15     public double suma_eth_p; // poprzednia suma eth
16     public double cena_eth;
17
18     Odwołania:3
19     public double get_delta_eth() // zwraca delte eth
20     {
21         return suma_eth - suma_eth_p;
22     }
23     Odwołania:3
24     public double get_delta_usd() //zwraca delte usd
25     {
26         return suma_usd - suma_usd_p;
27     }
28     1 odwołanie
29     public int get_eth_cena() // zwraca zaokraglona cene eth
30     {
31         return Convert.ToInt32(Math.Round(cena_eth));
32     }
33     Odwołania:0
34     public string get_eth_to_write() //zwraca wartosc eth w odpowiednim formacie do wypisania
35     {
36         return suma_eth.ToString("0,0", CultureInfo.InvariantCulture);
37     }
38     Odwołania:0
39     public string get_usd_to_write() //zwraca wartosc usd w odpowiednim formacie do wypisania
40     {
41         return suma_usd.ToString("0,0", CultureInfo.InvariantCulture);
42     }
43     1 odwołanie
44     public string get_delta_eth_to_write() //zwraca delte eth w odpowiednim formacie do wypisania
45     {
46         return get_delta_eth().ToString("0,0", CultureInfo.InvariantCulture);
47     }
48
49     2
50     1 odwołanie
51     public void next() // zapisuje sume wartosci jako poprzednia i zeruje sume
52     {
53         suma_eth_p = suma_eth;
54         suma_eth = 0;
55
56         suma_usd_p = suma_usd;
57         suma_usd = 0;
58     }
59 }
```

## 5.7. Values\_json

Klasa odpowiada odpowiedzi w formacie json co do posiadanych ilości tokenów na portfelu, oraz przetwarza te dane.

```
9 class Values_json // klasa odpowiadająca pobraniu odpowiedzi json o ilości tokenów
10 {
11     Odwołania: 0
12     public string status { get; set; }
13     Odwołania: 0
14     public string message { get; set; }
15     Odwołania: 3
16     public string result { get; set; }
17
18     Odwołania: 0
19     public decimal ToDecimal() // zwraca wartość w decimal
20     {
21         decimal a = Convert.ToDecimal(result);
22         a = a / 1000000000000000000; // wartość eth jest podana w 10-18
23         return a;
24     }
25
26     1 odwołanie
27     public double ToDouble_eth() // zwraca wartość eth w double
28     {
29         decimal a = Convert.ToDecimal(result);
30         a = a / 1000000000000000000; // wartość eth jest podana w 10-18
31         a = Math.Round(a, 2); // zaokrągla do drugiego miejsca po przecinku
32         return Convert.ToDouble(a);
33     }
34
35     2
36     1 odwołanie
37     public double Todouble_usd() // zwraca wartość eth w double
38     {
39         decimal a = Convert.ToDecimal(result);
40         a = a / 1000000; // wartość eth jest podana w 10-6
41         a = Math.Round(a, 2);
42         return Convert.ToDouble(a);
43     }
44 }
```