**Drozdowski Hubert, 238328**

**Mróz Jan, 242440**

**Grupa Wtorek 13:15**

**Inżynieria Systemów Baz Danych**

**CryptoFever**

*Prowadzący:*

*mgr Hanna Mazur*

|  |  |  |
| --- | --- | --- |
| **Użytkownik** | **Login** | **Hasło** |
| Gość | - | - |
| Inwestor | inwestor | ^inwestor2020$ |
| Inwestor | richman4 | testpassw |
| Administrator | admin | ^/admin/secret! |

**Wrocław, 01.01.2020 r.**

*Spis treści here*

**Temat, cel, zakres przedsięwzięcia, użytkownicy systemu.**

**Temat:**

Monitorowanie rynku i społeczności kryptowalut

**Cel projektu:**

Automatyzacja procesu obserwacji kursu i statystyk kryptowalut przy pomocy aplikacji webowej. Rozwiązanie ma za zadanie wspomóc przyszłych inwestorów w podejmowaniu decyzji rynkowych na podstawie dodanych przez nich reguł za pomocą odpowiednich informacji mailowych.

**Zakres przedsięwzięcia:**

TODO: zapytaj mgr Hannę Mazur.

* ekran logowania i rejestracji;
* możliwość definiowania reguł przez Inwestora;
* reguły są egzekwowane (przetwarzane) przez aplikację, a wynik jest wysyłany mailowo do użytkowników;
* historia kursu, statystyk i egzekucji reguł kryptowalut;
* panel administratora;

**Użytkownicy systemu:**

* **Gość** - osoba, która nie posiada konta w aplikacji, po zarejestrowaniu staje się Inwestorem.
* **Inwestor** - osoba, posiadająca swoje konto w aplikacji, posiada możliwość obserwacji statystyk i kursu kryptowalut. Może dodawać swoje własne reguły dotyczące kryptowalut, które są następnie przetwarzane przez aplikację, ma również wgląd w historię egzekucji w/w reguł.
* **Administrator** - osoba odpowiedzialna za system, posiada konto z najwyższymi uprawnieniami, ma dostęp do panelu administratora.

**Opis wymagań dla projektowanego systemu.**

**Wymagania funkcjonalne:**

* Obsługa giełdy kryptowalut BitBay API – odczyt cen krypowalut.
* Automatyczna analiza rynku – analiza procentowej zmiany ceny kryptowalut w czasie na podstawie porównania aktualnej ceny z ceną sprzed godziny, jednego dnia, trzech dni, siedmiu dni oraz trzydziestu dni.
* Automatyczna analiza społeczności – analiza procentowej zmiany popularnści wyszukiwania danych o kryptowalucie w wyszukiwarce Google. Porówniania wykonywane są tak samo jak przy analizie cen kryptowalut.
* Zapis danych historycznych – zapis co minutę aktualnej ceny kryptowalut oraz popularności wyszukiwania kryptowaluty, zapis egzekwowanych reguł inwestorów.
* Możliwość dodawania reguł oraz grup reguł przez Inwestorów określających warunki, które muszą zostać spełnione aby wysłać odpowiednią informację mailową.
* Egzekwowanie reguł Inwestorów – wysłanie odpowiedniej informacji mailowej do Inwestora w przypadku gdy wszystkie reguły z danej grupy reguł zostały spełnione.

**Gość:**

* Możliwość rejestracji jako Inwestor;

**Inwestor:**

* Możliwość logowania;
* Możliwość wglądu w aktualne wartości cen oraz statystyk społecznościowych kryptowalut.
* Możliwość wyświetlania, dodawania, edytowania i usuwania swoich grup reguł.
* Możliwość wyświetlania, dodawania, edytowania i usuwania reguł we własnej grupie reguł.
* Możliwość wglądu w historię egzekucji reguł.
* Możliwość wglądu w historię statystyk dotyczących kryptowalut.

**Admin:**

* Możliwość logowania.;
* Dostęp do panelu administratora.
* Możliwość wyświetlania, dodawania, edytowania, usuwania kryptowalut;
* Możliwość edycji danych użytkowników;

**Wymagania niefunkcjonalne:**

* Czas reakcji systemu poniżej 1s.
* Bęzbłędne działanie wszystkich usług w 99.9% czasu pracy.
* Powiadomienia do inwestorów są wysyłane w ciągu minuty.

**Lista transakcji dla poszczególnych grup użytkowników.**

**Gość:**

**TRA/001 Utworzenie konta inwestora.**

**Opis**: Zadaniem transakcji jest pobranie od gościa loginu, adresu mailowego oraz hasła i utworzenie konta inwestora w bazie danych.

**Inwestor:**

**TRA/002 Przeglądanie danych kryptowaluty.**

**Opis:** Zadaniem transakcji jest wyszukanie oraz wypisanie danych o kryptowalucie.

**TRA/003 Utworzenie grupy reguł.**

**Opis:** Zadaniem transakcji jest pobranie danych od inwestora (nazwa grupy reguł) i utworzenie pustej grupy reguł. Utworzona grupa reguł jest automatycznie przypisywana do użytkownika wykonującego transakcję.

**TRA/004 Przeglądanie grup reguł.**

**Opis:** Zadaniem transakcji jest wyszukanie grupy reguł i wypisanie jej danych. Transakcję może wykonać każdy inwestor, ale tylko do grup reguł utworzonych przez tego inwestora.

**TRA/005 Zmiana danych grupy reguł.**

**Opis:** Zadaniem transakcji jest wyszukanie grupy reguł i zmiana jej danych. Transakcję może wykonać każdy inwestor, ale tylko do grup utworzonych przez tego inwestora.

**TRA/006 Usunięcie grupy reguł.**

**Opis:** Zadaniem transakcji jest wyszukanie grupy reguł i jej usunięcie. Usunięcie grupy reguł usuwa także dane powiązane z daną grupą reguł, czyli całą historię egzekwowania reguł oraz wszystkie reguły znajdujące się w tej grupie reguł. Transakcję może wykonać każdy inwestor, ale tylko na grupie reguł utworzonych przez tego inwestora.

**TRA/007 Utworzenie reguły**

**Opis:** Zadaniem transakcji jest pobranie danych od inwestora (typ reguły, wartość) i utworzenie reguły. Transakcję może wykonać każdy inwestor.

**TRA/008 Przeglądanie reguł.**

**Opis:** Zadaniem transakcji jest wyszukanie reguły i wypisanie jej danych. Transakcję może wykonać każdy inwestor, ale tylko dla swoich reguł.

**TRA/009 Zmiana reguły.**

**Opis:** Zadaniem transakcji jest pobranie danych od inwestora, wyszukanie reguły i zmiana jej danych. Transakcję może wykonać każdy inwestor, ale tylko do reguł utworzonych przez tego inwestora.

**TRA/010 Usunięcie reguły**.

**Opis:** Zadaniem transakcji jest wyszukanie reguły i usunięcie jej z bazy danych. Transakcję może wykonać każdy inwestor, ale tylko do reguł utworzonych przez tego inwestora.

**TRA/011 Przeglądanie danych historycznych dot. egzekucji grup reguł.**

**Opis:** Zadaniem transakcji jest wyszukanie i wypisanie danych o danych historycznych egzekucji grupy reguł inwestora (t.j. data, kurs, typ transakcji). Operację może wykonać tylko inwestor, który jest właścicielem danej grupy reguł.

**TRA/012 Przeglądanie danych historycznych kryptowalut.**

**Opis:** Zadaniem transakcji jest wyszukanie i wypisanie historycznych danych dot. kryptowaluty (t.j. data, statystyki, kurs, współczynnik analizy rynkowej, współczynnik analizy społeczności). Operację może wykonać każdy inwestor.

**TRA/013 Przeglądanie danych kryptowaluty.**

**Opis:** Zadaniem transakcji jest wypisanie danych o kryptowalutach (data aktualizacji, kurs, współczynnik rynkowy, społecznościowy).

**Administrator:**

**TRA/014 Przeglądanie danych użytkowników.**

**Opis:** Zadaniem transakcji wypisanie danych użytkowników za wyjątkiem hasła.

**TRA/015 Zmiana danych użytkownika.**

**Opis:** Zadaniem transakcji jest wyszukanie i zmiana danych użytkownika.

**TRA/016 Usunięcie użytkownika.**

**Opis:** Zadaniem transakcji jest usunięcie użytkownika z bazy. Dane powiązane z użytkownikiem także zostaną usunięte (grupy reguł, historia egzekucji grup reguł, reguły powiązane).

**TRA/017 Utworzenie modelu kryptowaluty.**

**Opis:** Zadaniem transakcji jest pobranie danych od administratora i utworzenie nowego modelu kryptowaluty w bazie danych.

**TRA/018 Zmiana danych kryptowaluty**

**Opis:** Zadaniem transakcji jest wyszukanie oraz zmiana danych o kryptowalucie.

**TRA/019 Usunięcie modelu kryptowaluty.**

**Opis:** Zadaniem transakcji jest wyszukanie i usunięcie modelu kryptowaluty. Dane powiązane z kryptowalutą także zostaną usunięte (historia cen i statystyk społecznościowych, wszystkie grupy reguł i ich reguły oraz historie egzekucji odnoszące się do tej kryptowaluty).

**Dokumentacja zaimplementowanej bazy danych.**

1. Definicje predykatowe encji:

ENC/001

USER(IdU, Login, Password, IsAdmin, E-mail, CurrencyAmount)

Semantyka encji: Encja zawiera dane na temat użytkownika aplikacji.

ENC/002

RULESET(IdRS, Name)  
Semantyka encji: Encja zawiera dane na temat grupy reguł.

ENC/003

RULE(IdR, Value, TypeOfRule)Semantyka encji: Encja zawiera dane na temat pojedynczej reguły, która musi  
byś spełniona aby wykonać transakcję.

ENC/004

CRYPTO\_MODEL(IdCM, ShortName, LongName)Semantyka encji: Encja zawiera ogólne dane na temat modelu kryptowaluty.

ENC/005

TRADE(IdT, Date, Amount, Price)Semantyka encji: Encja zawiera dane na transakcji walut.

ENC/006

SOCIAL\_HISTORIC(IdSH, GtrendsTop7D, Date)  
Semantyka encji: Encja zawiera dane na temat statystyk społecznościowych o  
kryptowalucie.

ENC/007

MARKET\_HISTORIC(IdMH, BidsValue, AsksValue,AvgTransactionValue, Date, Price, ResponseJSON)  
Semantyka encji: Encja zawiera dane na temat statystyk rynkowych o  
kryptowalucie.

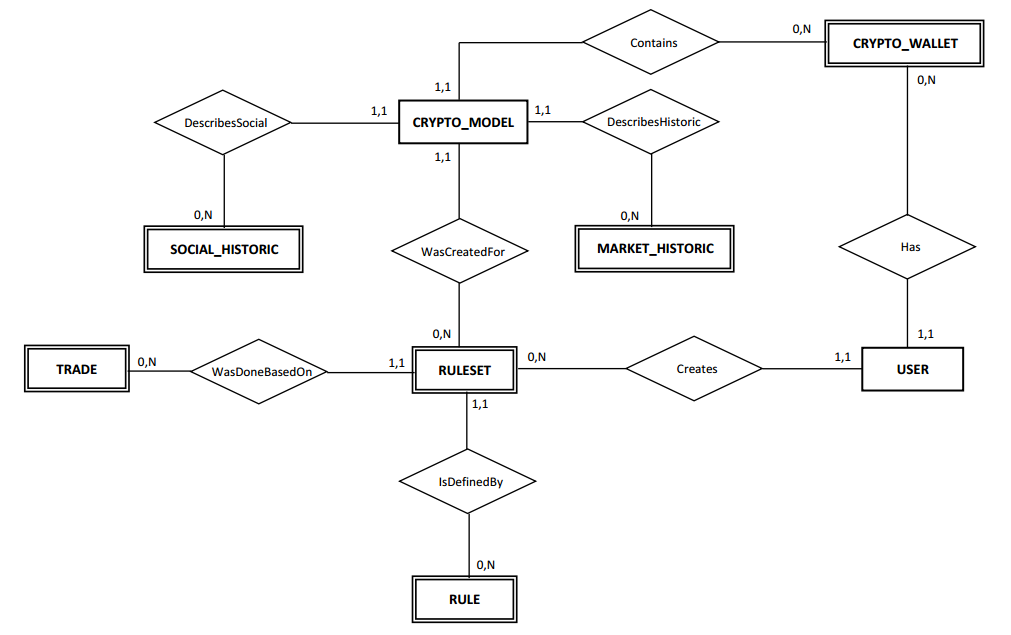
ENC/008

CRYPTO\_WALLET(IdCW, Amount, Date)Semantyka encji: Encja zawiera dane na temat krypto-portfela użytkownika  
aplikacji.

1. Definicje predykatowe związków:

ZWI/001 IsDefinedBy(RULESET(1,1):RULE(0,N))  
ZWI/002 WasDoneBasedOn(TRADE(0,N):RULESET(1,1))  
ZWI/003 Creates(USER(1,1):RULESET(0,N))  
ZWI/004 DescribesMarket(MARKET\_HISTORIC(0,N):CRYPTO\_MODEL(1,1))  
ZWI/005 DescribesSocial(SOCIAL\_HISTORIC(0,N):CRYPTO\_MODEL(1,1))  
ZWI/006 Has(USER(1,1):CRYPTO\_WALLET(0,N))  
ZWI/007 Contains(CRYPTO\_WALLET(0,N):CRYPTO\_MODEL(1,1))  
ZWI/008 WasCreatedFor(RULESET(0,N):CRYPTO\_MODEL(1,1))

**Diagram związków encji:**



Rysunek 1. Diagram związków encji.

**Definicje schematów relacji i przykładowe dane.**

**Wstęp:**

System CryptoFever posiada bazę danych stworzoną z pomocą PostgreSQL, typy wpisywane w tabelach odpowiadają typom PostgreSQL, które zostały użyte w systemie bazy danych.  
Większość danych w bazie danych jest związanych z analizą statystyk kryptowalut i jest automatycznie uzupełniania przez aplikację. Z tego względu wyróżnione zostały trzy źródła danych:

**REL/001** auth\_user/USER

**auth\_user**(id, passwor, last\_login, is\_superuser, username, first\_name, last\_name, email, is\_staff, is\_active, date\_joined, base\_currency)

Schemat relacji *auth\_user* został zaczerpnięny z frameworka Django. (dokumentacja: <https://docs.djangoproject.com/en/3.0/ref/contrib/auth/>), ze względu na używany system autoryzacji. Zostało do niego dodane pole *base\_currency,* które zawiera dane o ilości posiadanej przez użytkownika waluty podstawowej, której używa do wykonywania transakcji na giełdzie BitBay, np. polskiego złotego.

Tabela 1. Przykładowe, najważniejsze dane relacji auth\_user.

|  |  |  |  |
| --- | --- | --- | --- |
| **id** | **password** | **username,** | **email** |
| 1 | ^invest234 | Investo2020 | Investor2020@gmail.com |
| 3 | P@55w0rd | JanKowalski | jan.kowalski1995@gmail.com |
| 4 | zaq1@wsx | psychonauts56 | bigPsycho42@gmail.com |

**REL/002** crypto\_ruleset/RULESET

**crypto\_ruleset(**id, name, type\_of\_ruleset, crypto\_id, owner\_id**)**

Opis atrybutów:

|  |  |
| --- | --- |
| **Nazwa atrybutu** | **Opis** |
| id | Identyfikator grupy reguł. Klucz główny. |
| name | Nazwa dla grupy reguł. |
| type\_of\_ruleset | Typ grupy reguł. (TODO WYTŁUMACZYĆ LITERKI E,B,S) |
| crypto\_id | Identyfikator modelu kryptowaluty, klucz obcy. |
| owner\_id | Identyfikator inwestora, klucz obcy. |

Tabela 1. Przykładowe, najważniejsze dane relacji crypto\_ruleset.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **id** | **name** | **type\_of\_ruleset** | **crypto\_id** | **owner\_id** |
| 1 | SellEtherium | E | 2 | 1 |
| 3 | BigSocialBitcoin | E | 3 | 1 |
| 4 | BuyEtherium | E | 2 | 3 |

**REL/003** crypto\_rule/RULE

**crypto\_rule(**id, value, type\_of\_rule, #rule\_set\_id**)**

Opis atrybutów:

|  |  |
| --- | --- |
| **Nazwa atrybutu** | **Typ** |
| id | Identyfikator reguły, klucz głowny. |
| value | Wartość referencyjna dla reguły. |
| type\_of\_rule | Typ reguły w postaci 3 znaków:  BEL – poniżej  ABO – powyżej  CNG – zmiana  CNP – zmiana procentowa  MVP – maksymalna wartość procentowa  MVE – maksymalna wartość  AHS – opóźnienie akcji  MBA – bot rynkowy powyżej  MBB – bot rynkowy poniżej  SBA – trend Google powyżej  SBB – trend Google poniżej |
| rule\_set\_id | Identyfikator grupy reguł do której dana reguła należy. Klucz obcy. |

Tabela x. Przykładowe dane relacji crypto\_rule.

|  |  |  |  |
| --- | --- | --- | --- |
| **id** | **value** | **type\_of\_rule** | **rule\_set\_id** |
| 1 | 350.16 | BEL | 2 |
| 3 | 0.04 | CNP | 2 |
| 4 | 43.34 | SBA | 6 |

**REL/004** crypto\_cryptomodel/CRYPTO\_MODEL

**crypto\_cryptomodel(**id, short\_name, long\_name**)**

Opis atrybutów:

|  |  |
| --- | --- |
| **Nazwa atrybutu** | **Typ** |
| id | Identyfikator modelu kryptowaluty. Klucz główny. |
| short\_name | Krótka nazwa dla kryptowaluty. |
| long\_name | Długa nazwa dla kryptowauty. |

Tabela x. Przykładowe dane relacji crypto\_cryptomodel.

|  |  |  |
| --- | --- | --- |
| **id** | **short\_name** | **long\_name** |
| 1 | ETH | Etherium |
| 3 | BTC | Bitcoin |
| 4 | LTC | Litecoin |

**REL/005** crypto\_trade/TRADE

**crypto\_trade(**id, date, type\_of\_trade, amount, price, rule\_set\_id**)**

Opis atrybutów:

|  |  |
| --- | --- |
| **Nazwa atrybutu** | **Opis** |
| id | Identyfikator transakcji, klucz główny. |
| date | Data zrealizowania transakcji. |
| type\_of\_trade | Typ wykonanej transakcji. W postaci jednego znaku (TODO – co oznacza jaki znak) |
| amount | Ilość kryptowaluty jaka była związana z transakcją. |
| price | Cena uzyskana ze sprzedaży, lub koszt zakupu kryptowaluty. |
| rule\_set\_id | Identyfikator grupy reguł, która jest odpowiedzialna za wykonanie transakcji. Klucz obcy. |

Tabela x. Przykładowe dane relacji crypto\_trade.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **id** | **date** | **type\_of\_trade** | **amount** | **price** | **rule\_set\_id** |
| 12 | 21.12.2019 4:00:00 | S | 0.004321 | 54.32 | 2 |
| 34 | 22.12.2019 4:00:00 | M | 2.2811 | 2642.96 | 4 |
| 65 | 23.12.2019 4:00:00 | B | 0.02119 | 95.04 | 11 |

**REL/006** crypto\_socialhistoric/SOCIAL\_HISTORIC

**crypto\_socialhistoric(**id, date, crypto\_id, gtrends\_top\_7d**)**

Opis atrybutów:

|  |  |
| --- | --- |
| **Nazwa atrybutu** | **Opis** |
| id | Identyfikator modelu kryptowaluty. Klucz główny. |
| date | Data pobrania wartości statystycznej. |
| crypto\_id | Identyfikator modelu kryptowaluty. Klucz obcy. |
| gtrends\_top\_7d | Ilość wyszukiwań haseł związanych z daną kryptowalutą w wyszukiwarce Google w ostatnich 7 dniach. |

Tabela x. Przykładowe dane relacji crypto\_socialhistoric.

|  |  |  |  |
| --- | --- | --- | --- |
| **id** | **date** | **crypto\_id** | **gtrends\_top\_7d** |
| 4556 | 21.12.2019 4:00:00 | 1 | 4356.21 |
| 4557 | 22.12.2019 4:00:00 | 1 | 4310.28 |
| 4558 | 23.12.2019 4:00:00 | 2 | 4198.93 |

**REL/007** crypto\_markethistoric/MARKET\_HISTORIC

**crypto\_markethistoric(**id, price, response\_json, date, crypto\_id, asks\_value, avg\_trans**)**

Tabela x. Przykładowe dane relacji crypto\_markethistoric.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **id** | **price** | **response\_json** | **date** | **cypto\_id** | **asks\_value** | **avg\_transaction\_value** | **birds\_value** |
| 6993 | 572.93 | {odpowiedź od BitBay API} | 21.12.2019 4:54:03 | 1 | (TODO) |  |  |
| 6994 | 572.96 | {odpowiedź od BitBay API} | 21.12.2019 4:55:04 | 1 |  |  |  |
| 6995 | 572.99 | {odpowiedź od BitBay API} | 21.12.2019 4:56:03 | 2 |  |  |  |

**REL/008** crypto\_cryptowallet/CRYPTO\_WALLET

**crypto\_wallet(**id, amount, crypto\_id, owner\_id, date**)**

Relacja *crypto\_cryptowallet* jest relacją, ktróra została utworzona z myślą o możliwości rozszerzenia funkcjonalności aplikacji o automatyczne wykupowanie i sprzedawanie kryptowalut na giełdzie BitBay. Aktualnie relacja nie jest w aplikacji wykorzystywana.

Tabela x. Przykładowe dane relacji crypto\_cryptowallet.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **id** | **amount** | **crypto\_id** | **owner\_id** | **date** |
| 6 | 52.21 | 1 | 2 | 21.12.2019 4:54:03 |
| 90 | 52.21 | 1 | 2 | 23.12.2019 6:24:41 |
| 456 | 73.40 | 4 | 7 | 28.12.2019 5:01:10 |

**Narzędzia implementacyjne bazy danych.**

**Python:** Język w którym została zaimplementowana większa część aplikacji.

**Django:** Wysokopoziomowy framework do języka Python służący do tworzenia aplikacji webowych. Użyty został do zrealizowania mechanizmu logowania i tworzenia kont użytkownika, renderowania stron internetowych, zapewniania komunikacji klient-serwer oraz dostępu do bazy danych. Django dodatkowo zapewnia dostęp do panelu administratora aplikacji, w którym można przeglądać i dowolnie modyfikować dane znajdujące się w bazie danych.

**PostgreSQL:** System relacyjnych baz danych w którym została zaimplementowana baza danych aplikacji. Django współpracuje z PostgreSQL dostarczając wygodne funkcje do obsługi bazy danych (w tym pisanie zapytań za pomocą funkcji w języku Python).

**Redis** – Pamięć bazowa używana do kolejkowania zadań.

**Nginx** – Odpowiada za świadczenie usług WWW na serwerze.

**Celery** – Odpowiada za wykonywanie zadań na serwerze wg ustalonej kolejności.

**Pytest –** Framework do tworzenia testów funkcjonalnych oraz jednostkowych do języka Python.

**Selenium** – Narzędzie do tworzenia testów E2E.

**Opis aplikacji bazodanowej.**

**Tytuł:**

CryptoFever  
  
**Przeznaczenie:**  
Aplikacja CryptoFever jest przeznaczona dla inwestorów giełdy kryptowalut, którzy chcieliby zautomatyzować obserwację wybranych przez siebie kryptowalut według swoich preferencji poprzez dostęp do prostej aplikacji webowej zapewniającej powiadomienia mailowe o konkretnych interesujących ich notowaniach na rynku kryptowalut.   
Powiadomienia mailowe uwzględniałyby zarówno aspekt cenowy kryptowaluty, jak i dane o statystykach społecznościowych (ilość zapytań w Google związanych z daną kryptowalutą).

**Użytkownicy:**  
**Gość** – Nie ma jeszcze konta w systemie. Ma jedynie możliwość utworzenia go.

**Inwestor** – Osoba, która posiada konto w aplikacji CryptoFever. Po zalogowaniu ma możliwość obserwacji statystyk i kursu kryptowalut. Może zarządzać grupami reguł oraz regułami, które określają przypadki w których ma zostać wysłane powiadomienie mailowe. Inwestor ma także wgląd w historię wysyłania powiadomień mailowych.

**Administrator –** Osoba odpowiedzialna za prawidłowe funkcjonowanie systemu. Oprócz uprawnień Inwestora posiada dostęp do panelu administratora, w którym może przeglądać i modyfikować wszystkie dane użytkowników i bazy danych.

**Zakres: (TODO)**

**Specyfikacja wymagań funkcjonalnych:**

* **Możliwość logowania i zapisu danych Inwestora na serwerze.** Serwer gromadzi dane wprowadzane przez Inwestora, takie jak reguły oraz grupy reguł, zapewniając do nich dostęp niezależnie od urządzenia, na którym Inwestor korzysta z jego usług.
* **Możliwość wylogowania z konta.**
* **Dostęp do utworzonych przez Inwestora danych jedynie po prawidłowym zalogowaniu na jego konto.** Zabezpieczenie przed niechcianymi wizytami osób trzecich w aplikacji. Posiadanie adresu internetowego aplikacji nie powinno uprawniać do przeglądania prywatnych danych użytkowników tej aplikacji, dlatego Inwestor otrzymuje dostęp do swoich danych tylko i wyłącznie po poprawnej autoryzacji.
* **Dostęp do poszczególnych funkcji aplikacji następuje dopiero po poprawnym zalogowaniu na konto.** Aplikacja udostępnia swoje funkcjonalności jedynie użytkownikom zalogowanym.
* **Możliwość samodzielnego utworzenia konta w aplikacji przez Gościa.** Do utworzenia konta nie jest potrzebna ingerencja osoby z wyższymi uprawnieniami.
* **Możliwość przeglądania statystyk cen kryptowalut.** Inwestor może zobaczyć procentową zmianę ceny wybranej kryptowaluty w stosunku do wartości sprzed godziny, jednego dnia, trzech dni, tygodnia oraz miesiąca.
* **Możliwość przeglądania historycznych cen kryptowalut.** Inwestor może przeglądać historyczne ceny wybranej kryptowaluty zapisywane co minutę od czasu uruchomienia systemu.
* **Możliwość przeglądania statystyk społecznościowych kryptowalut.** Inwestor może zobaczyć procentową zmianę 7-dniowych trendów zapytań związanych z daną kryptowalutą w wyszukiwarce Google.
* **Możliwość przeglądania historycznych statystyk społecznościowych kryptowalut.** Inwestor może przeglądać historyczne dane odnośnie trendów zapytań zapisywanych codziennie od czasu uruchomienia systemu.
* **Możliwość przeglądania, tworzenia, usuwania i modyfikowania grup reguł.** Inwestor ma możliwość zarządzania grupami reguł utworzonymi w celu obserwacji zachowania konkretnej kryptowaluty na rynku. Każdej grupie można nadać typ, który definiuje co ma się stać po spełnieniu wszystkich reguł w grupie. (w pierwszej wersji systemu dostępne jest jedynie powiadomienie mailowe) Każdej grupie można też nadać indywidualną nazwę.
* **Możliwość przeglądania, tworzenia, usuwania i modyfikowania reguł w grupach.** Inwestor ma możliwość zarzadzania regułami znajdującymi się w grupie reguł. Każda reguła w grupie reguł musi mieć inny typ. Każda reguła ma także wartość referencyjną.
* **Egzekwowanie reguł.** Reguły utworzone przez Inwestora są egzekwowane, tzn. jeśli wszystkie warunki określone przez Inwestora za pomocą reguł w jednej grupie zostaną spełnione, następuje akcja określona poprzez typ grupy. Przykładowo, jeśli Inwestor stworzył grupę reguł o typie „powiadomienie mailowe” i umieścił w niej reguły „Cena większa niż 250zł”, „Utrzymuje się przez 10 minut”, to w momencie gdy cena kryptowaluty będzie utrzymywała się na poziomie powyżej 250zł przez dłużej niż 10 minut, zostanie wysłane powiadomienie na adres mailowy Inwestora podany podczas rejstracji w systemie. Typy egzekwowanych reguł to:
  + Cena powyżej podanej wartości.
  + Cena poniżej podaniej wartości.
  + Zmiana kursu waluty o podaną wartość..
  + Procentowa zmiana kursu waluty o podaną wartość.
  + (TODO) Max\_value\_perc.
  + (TODO) Max\_value.
  + Opóźnienie podejmowania akcji o podany czas.
  + Wskazania rynkowego bota powyżej podanej wartości.
  + Wskazania rynkowego bota poniżej podanej wartości.
  + Wskazania trendów Google powyżej podanej wartości.
  + Wskazania trendów Google poniżej podanej wartości.
* **Możliwość przeglądania historii egzekwowanych reguł.** Każde spełnienie wszystkich warunków w grupie reguł powinno być odnotowane do historii w systemie, do której powinien mieć dostęp Inwestor. Każdy wpis w historii powinien zawierać datę,
* **Możliwość sortowania historii egzekwowanych reguł.**

**Specyfikacja wymagań niefunkcjonalnych:**

* **Minimalne warunki, które musi spełniać sprzęt na którym zostanie zainstalowana aplikacja:**
  + System: Windows 7/8.1/10, Linux
  + Min. 2GB pamięci RAM
  + Procesor z taktowaniem min 1GHz.
  + Min. 500MB wolnej przestrzeni dyskowej + 7MB wolnej przestrzeni dyskowej na każdą dobę działania systemu. Wynika to z gromadzenia komunikatów o cenach JSON otrzymywanych od BitBay API.
  + Stały dostęp do internetu o prędkości minimum 128kb/s.
  + Zainstalowane środowisko Docker.
  + Zainstalowane środowisko Python 2.7.
* **Dostęp do aplikacji z poziomu przeglądarki internetowej** (np. Mozilla Firefox, Google Chrome, Opera). Ma to zapewnić dostęp do aplikacji bez konieczności instalowania jej na różnych urządzeniach, aby inwestor mógł jak najszybciej zareagować na otrzymywane powiadomienia.
* **Bezawaryjny czas pracy usługi na poziomie 99,9% czasu pracy.** Aby na systemie można było polegać, musi on działać niezawodnie przez zdecydowaną większość czasu pracy, tak aby Inwestor zawsze był informowany na bieżąco o interesujących go sytuacjach.
* **Średni czas reakcji systemu na interakcję z użytkownikiem poniżej 1 sekundy.** Czas reakcji na giełdzie jest bardzo ważny, dlatego reakcja systemu powinna być bardzo szybka, tak aby Inwestor zdążył podjąć akcję w interesujących go warunkach.
* **Maksymalny czas oczekiwania na powiadomienie mailowe to 1 minuta od momentu zaistnienia interesujących Inwestora warunków na giełdzie kryptowalut.**

**Mapa wywołań formularzy i raportów:**

**Gość**

**Administrator**

Formularz rejestracji

Ekran logowania (ukryty)

Grupy reguł inwestora dot. kryptowaluty

Dodaj/Usuń/Edytuj kryptowalutę

Dodaj/Usuń/Edytuj użytkowika

Widok użytkowników

Widok kryptowalut

Dane historyczne dot. kryptowaluty

Panel Administratora

Raport wszystkich kryptowalut

Ekran logowania

Usuń grupę reguł

Zobacz reguły

Historia egzekucji grupy reguł

Edytuj grupę reguł

Dodaj nową grupę reguł

Usuń regułę

Dodaj regułę

Edytuj regułę

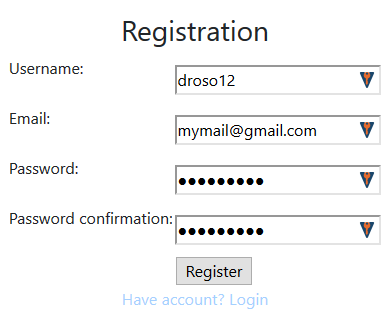
**Inwestor**

**Przykładowe formularze:**

**Rejestracja jako inwestor.**

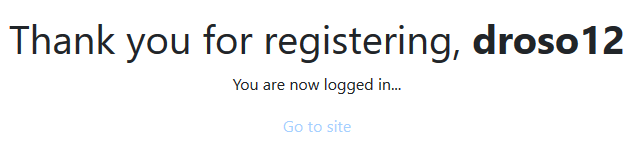
Aby zarejstrować się jako Inwestor w aplikacji należy otworzyć witrynę o adresie <http://127.0.0.1/accounts/register/>, gdzie *127.0.0.1* może być dowolnym adresem internetowym z którym powiązane są usługi serwera CryptoFever.

Formularz rejestracji przedstawiony jest na *rysunku 1*.



Rysunek 1. Ekran rejestracji.

Hasło musi zawierać co najmniej osiem znaków, w tym jeden znak specjalny. Ponadto Inwestor musi posiadać dostęp do podanego podczas rejestracji maila, ponieważ jest on używany w celu wysyłania powiadomień.

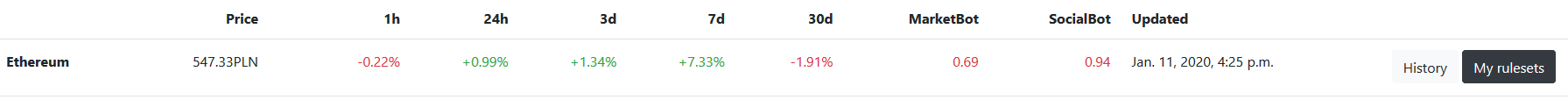


Rysunek 2. Pomyślne logowanie.

Po pomyślnym zarejestrowaniu ukaże się takie powiadomienieo pomyślnej rejestracji w aplikacji (*Rysunek 2*), inwestor zostanie automatycznie zalogowany. Odnośnik „Go to site” otworzy główny panel aplikacji na którym będzie widoczny raport wszystkich kryptowalut (*Rysunek 3*).

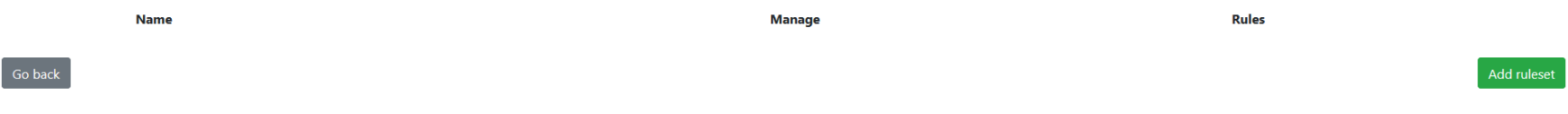
**Dodanie nowej grupy reguł dla inwestora:**

Z widoku ekranu głównego (*Rysunek 3*), po wybraniu kryptowaluty (na ten moment dostępne jest tylko Ethereum), należy wybrać opcję „My rulesets”.



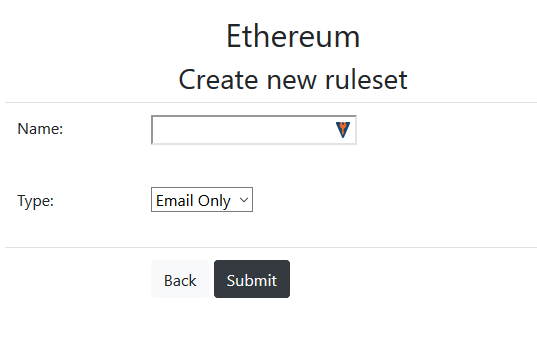
Rysunek 3. Ekran główny prezentujący dane dot. kryptowalut

W widoku grup reguł inwestora (*Rysunek 4*) należy wybrać opcję „Add ruleset”.



Rysunek 4. Widok grup reguł inwestora.

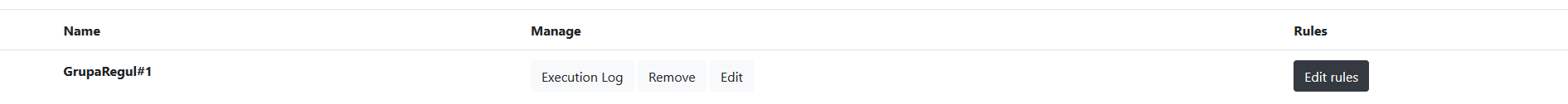
Ukaże się formularz dodawania nowej grupy reguł (*Rysunek 5*), w polu *Name* należy wpisać nazwę naszej grupy reguł, a w polu *Type* należy wybrać typ grupy. Do wyboru są trzy wartości „Email Only” (wysyłaj tylko wiadomości mail), „Sell” (grupa reguł dotycząca sprzedaży) oraz „Buy” (grupa reguł dotycząca kupna), aktualnie wartość „Email Only” jest zaimplementowana, a pozostałe wartości są wyszarzone. Należy zastosować zmiany wybierając opcję „Submit”.



**Dodanie nowej reguł dla grupy reguł.**

Rysunek 5. Ekran dodawania nowej grupy reguł

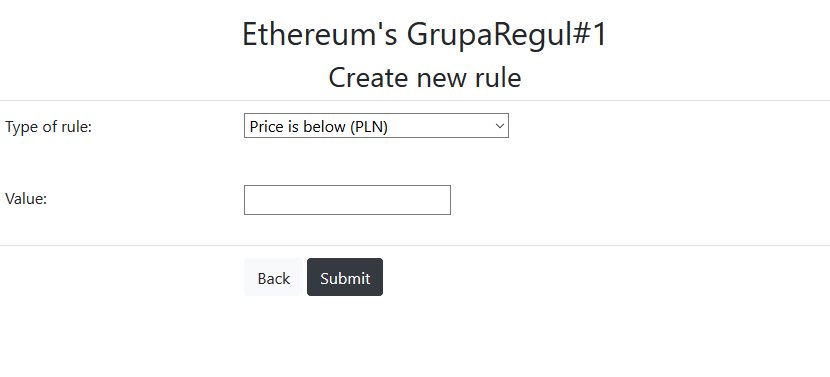
Z widoku grupy reguł inwestora, należy wybrać wcześniej utworzoną grupę reguł i wybrać opcję „Edit rules”.



Rysunek 6 Widok usupełnionych grup reguł inwestora

Następnie podobnie jak w widoku dla grup reguł należy wybrać „Add new rule”. Ukaże się formularz dodawania nowej reguły (*Rysunek 7*), *Type of rule* są to typy reguł (które zostały opisane w dokumentacji TODO – wskazanie gdzie), w przypadku „Price is below (PLN)” wpisanie w polu *Value* (wartość zmiennoprzecinkowa): 500.0, oznacza (gdy istnieje tylko ta jedna reguła), że do Inwestora zostanie wysłany mail, jeśli cena kryptowaluty będzie wynosić poniżej 500 zł.

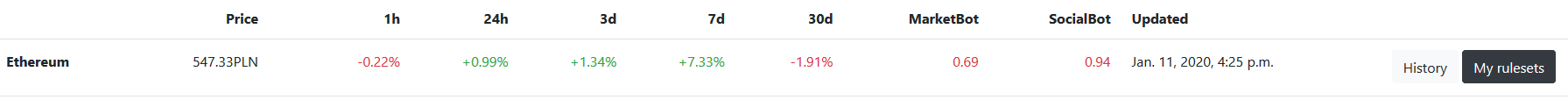
Do grupy reguł można dodawać tylko reguły o różnych typach, ale żeby zadanie grupy reguł zostało wykonane (np. wysłanie powiadomienia mailowego), wszystkie reguły muszą zostać spełnione.



Rysunek 7. Ekran dodawania nowej reguły dla „GrupaRegul#1”

**Przykładowe raporty:**

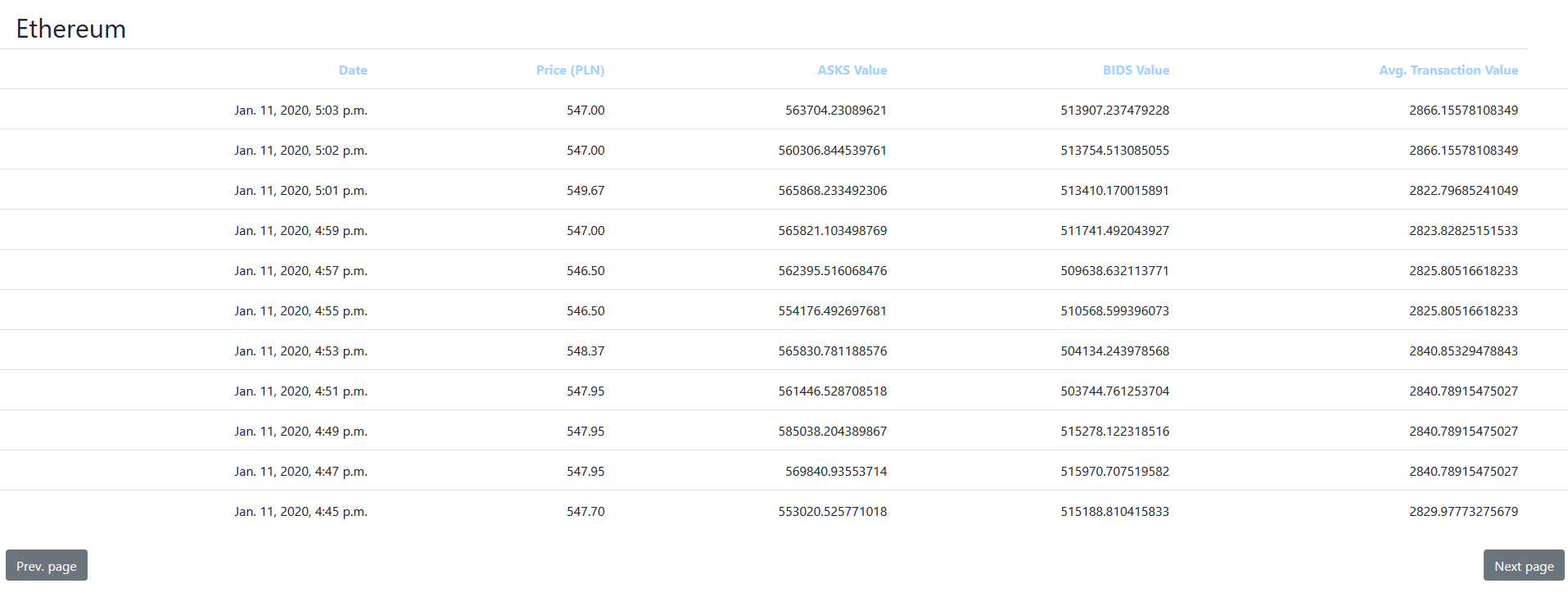
**Dane historyczne kryptowaluty.**

Interesującą funkcjonalnością aplikacji jest zapisywanie danych historycznych kryptowaluty. Aby mieć dostęp do tych danych należy wybrać opcję „History” z ekranu głównego przy jednej z kryptowalut (*Rysunek 8*).

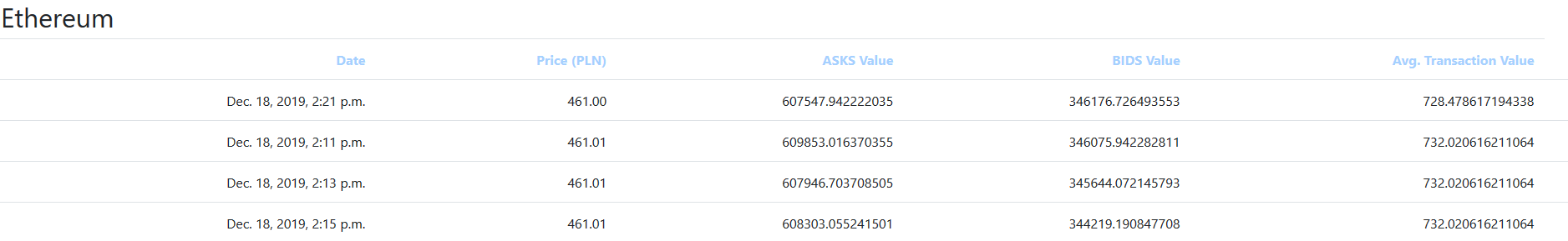
Rysunek 8 Ekran główny, widok na wszystkie obsługiwane kryptowaluty

Po wyboże ukaże się ekran danych historycznych dla danej kryptowaluty (*Rysunek 9*), aby przejść do kolejnej strony należy wybrać opcję „Next Page”.

Widok zawiera także możliwość sortowania. Aby posortować dane według wybranego kryterium należy wybrać nagłówek tabeli odpowiadający temu kryterium.

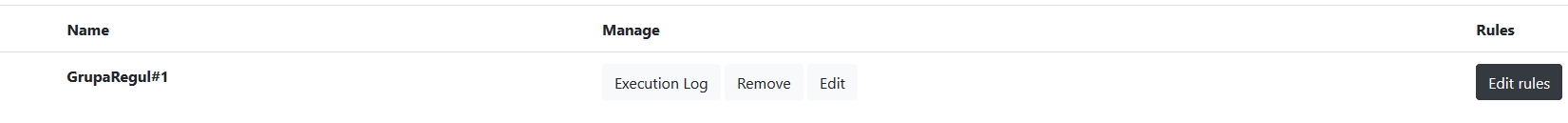


Rysunek 9. Ekran danych historycznych dla Ethereum.

Dane posortowane rosnąco według ceny zostały przedstawione na *rysunku 10*:

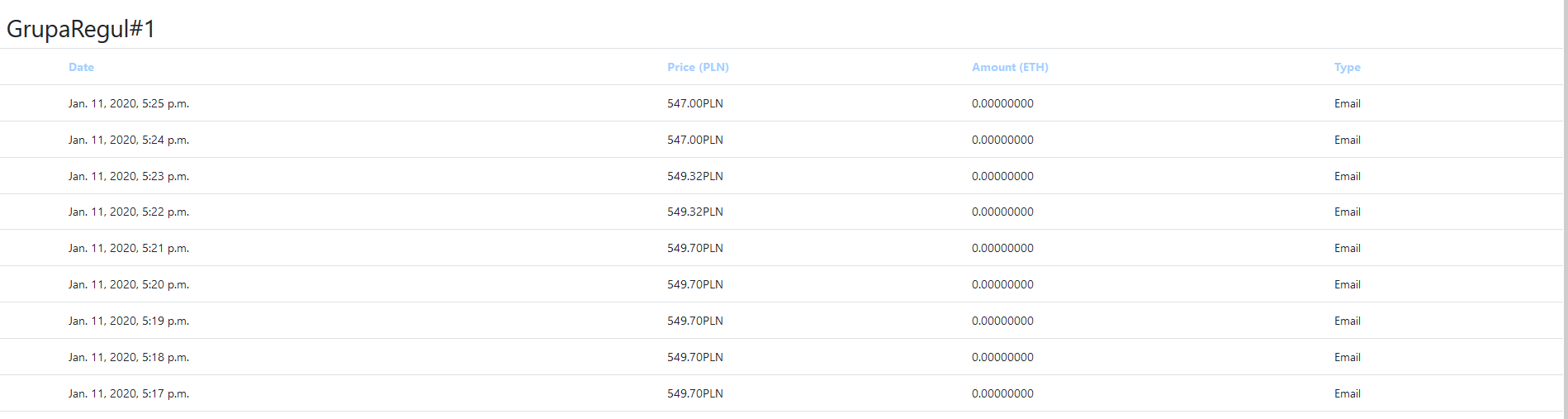
Rysunek 9. Ekran danych historycznych dla Ethereum.

**Dane historyczne egzekucji grupy reguł.**

Reguły inwestora są najpierw sprawdzane, a następnie jeżeli wszystkie reguły w danej grupie reguł są spełnione grupa jest egzekwowana. Aby sprawdzić historię egzekucji należy otworzyć panel grup reguł dla danej kryptowaluty (*Rysunek 6*), a następnie wybrać opcję „Execution Log” przy wybranej grupie reguł.

Rysunek 11 Widok grupy reguł inwestora

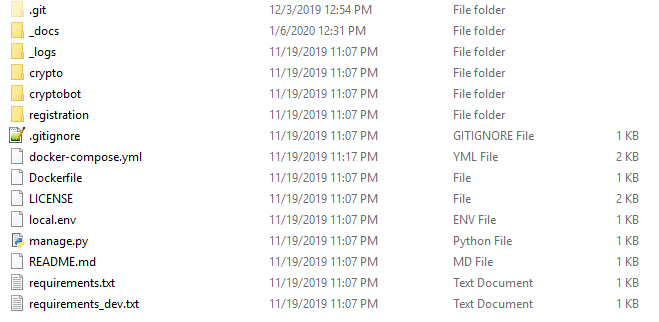
Na ekranie ukaże się historia wykonywania grupy reguł, czyli wszystkich przypadków w których wszystkie reguły zawarte w danej grupie reguł zostały spełnione. Tabela zawierająca historię wykonywania grupy reguł podlega sortowaniu według kryterium wybieranego z nagłówków tabeli. Dostępne kryteria sortowanai to data, cena, ilości kryptowaluty biorącej udział w transakcji oraz typ grupy reguł.



Rysunek 12 Historia egzekucji dla grupy reguł GrupaRegul#1

**Zawartość nośnika z systemem.**

System dystrybuowany jest w postaci publicznego repozytorium GIT na platformie GitHub. System nie jest scentralizowany i nie istnieje żaden publiczny serwer który udostępnia usługi CryptoFever. Zostało to zrobione z powodów bezpieczeństwa, gdyż na serwerze przechowywane są prywatne informacje inwestorów o ich inwestycjach na rynku kryptowalut. Wyciek takich danych mógłby być w skutkach bardzo szkodliwy dla użytkowników korzystających z tego systemu. Zamiast scentralizowanego serwera aplikacja jest dystrybuowana za pomocą prostej aplikacji umieszczonej w kontenerze Dockera. Każda osoba chcąca korzystać z aplikacji może zainstalować ją w kontenerze Dockera u siebie na lokalnym komputerze, który zacznie wtedy działać jako serwer dostępny jedynie dla użytkownika. Dzięki temu na jednym serwerze CryptoFever znajdzie się o wiele mniej informacji o użytkownikach systemu.

****Pliki z całym systemem zajmują łącznie 21,6 MB, przy czym nie zawierają potrzebnych do uruchomienia systemu bibliotek. Biblioteki które należy zainstalować aby zapewnić poprawne działanie systemu są wylistowane w pliku *requirements.txt*.

Rysunek x. Zawartość nośnika z systemem.

**Foldery:**

* **\_docs** – zawiera kompletną dokumentację aplikacji, w tym tę dokumentację, diagramy ERD, mapę formularzy oraz prototyp aplikacji.
* **\_logs –** zawiera logi, które aplikacja generuje podczas działania.
* **crypto –** zawiera kod odpowiedzialny za poprawne funkcjonowanie grup reguł, reguł, analizę kryptowalut, statystyk społecznościowych oraz wykonywanie zadań grup reguł.
* **cryptobot –** zawiera m.in. ustawienia (settings.py).
* **registration –** zawiera logikę logowania oraz autoryzacji.

**Pliki:**

* **Dockerfile** oraz **docker\_compose.yml** – służą do uruchomienia serwera aplikacji przy pomocy darzędzi *Docker* oraz *docker\_compose*.
* **local.env –** przechowuje najważniejsze ustawienia aplikacji.
* Pliki **requirements\*.txt –** zawierają dane o wymaganych modułach i pakietach dla języka Python wymaganych do poprawnego funkcjonowania aplikacji.

**Sposób instalacji / uruchomienia systemu.**

Instrukcja została utworzona z myślą o użytkownikach systemu Windows 10, jednak kroki dla systemu macOS oraz Linux są analogiczne.

1. Przekopiować zawartość nośnika z systemem do wybranego folderu na komputerze.
2. Zainstalować aplikację *Docker* z witryny internetowej <https://docs.docker.com/docker-for-windows/install/>
3. Przejść do folderu w którym znajduje się przekopiowana wcześniej aplikacja.
4. Edytować plik *local.env*. Uzupełnić następujące ustawienia:
   1. EMAIL\_HOST – adres serwera świadczącego usługi poczty elektronicznej.
   2. EMAIL\_HOST\_USER – adres mailowy z którego mają być wysyłane powiadomienia aplikacji.
   3. EMAIL\_HOST\_PASSWORD – hasło do podanego adresu mailowego.
   4. EMAIL\_PORT – port na którym świadczona jest usługa poczty elektronicznej.

Uwaga: Należy wcześniej skonfigurować adres pocztowy umożliwiając usługom zewnętrznym korzystanie z niego.

1. Uruchomić okno konsoli w folderze aplikacji i wywołać następującą komendę, wywołującą kompilację oraz uruchomienie aplikacji: „docker-compose up”.

Okna konsoli nie należy zamykać, w jego ramach odbywa się świadczenie usług aplikacji.

1. Przywołać kolejne okno konsoli w folderze aplikacji i wywołać następne komendy:
   1. „docker-compose exec web python manage.py collectstatic --noinput”
   2. „docker-compose exec web python manage.py migrate”
2. Aplikacja CryptoFever jest gotowa do użycia. W przeglądarce na urządzeniu wprowadź adres *http://127.0.0.1/* aby sprawdzić poprawność działania.

Domyślnie w aplikacji nie istnieje żadne konto Administratora. Aby je utworzyć należy wykonać następujące kroki:

1. Uruchomić okno konsoli w folderze aplikacji i wywołać komendę:

„docker-compose exec web python manage.py createsuperuser”

1. Postępować zgodnie z instrukcjami w konsoli, czyli podać login oraz hasło dla Administratora.
2. Sprawdzić poprawność działania konta Administratora logując się według podanych wcześniej danych pod adresem [*http://127.0.0.1/admin/*](http://127.0.0.1/admin/)*.* Po poprawnym zalogowaniu zostanie wyświetlony panel administratora.

**Testowanie aplikacji**

1. Aby przetestować aplikację należy najpierw zainstalować język Python 3.7+.

<https://www.python.org/>

1. Następnie należy w oknie konsoli w folderze aplikacji wywołać następujące komendy:
   1. python -m ensurepip
   2. python -m pip install -r requirements.txt
   3. python -m pip install -r requirements\_dev.txt
2. A następnie żeby przetestować aplikację należy wywołać komendę:

DJANGO\_SETTINGS\_MODULE=cryptobot.settings pytest

Aplikacja posiada testy E2E, funkcjonalne oraz jednostkowe. Podczas testów E2E zostanie otworzona przeglądarka. Projekt zawiera 20 testów.

**Zakończenie**

Celem projektu była automatyzacja procesu obserwacji kursu i statystyk kryptowalut przy pomocy aplikacji webowej. Rozwiązanie miało za zadanie wspomóc przyszłych inwestorów w podejmowaniu decyzji rynkowych na podstawie dodanych przez nich reguł.

Cel został osiągnięty, aplikacja spełnia swoje zadanie, jednak nie zostały zaimplementowane dodatkowe wymagania takie jak np. automatyczna sprzedaż i kupno kryptowalut. Dodatkowo na interfejs należałoby poświęcić więcej czasu, aby wyglądał on bardziej nowocześnie. Aplikacja została napisana modularnie tzn. dodawanie nowych kryptowalut i giełd nie stoi na przeszkodzie, jednak na samą implementację tych dodatkowych funkcjonalności nie starczyło czasu.

Użyte narzędzia sprawdziły się, Django oraz PostgreSQL wspierają nowoczesne technologie bazodanowe, które mogą pomóc w przyszłym rozwoju aplikacji, sam język Python pozwolił na dość szybką implementację logiki biznesowej, natomiast Bootstrap pozwolił na szybkie stworzenie szaty graficznej, którą można dalej rozwijać. Narzędzia testujące pytest oraz selenium także zdały swój egzamin.

Podsumowując, projekt się powiódł, jednak jest to temat na tyle obszerny i tak szybko zmieniający się, że należałoby go dalej rozwijać, jednym z pomysłów autorów jest wypuszczenie aplikacji na licencji Open Source tak aby także społeczność mogła pomóc w rozwoju cryptobota.

**Źródła**

<https://docs.djangoproject.com/en/3.0/> - dokumentacja Django

<https://www.postgresql.org/docs/> - dokumentacja PostgreSQL

<https://www.livecoinwatch.com/> - strona ze statystykami kryptowalut

**Dokumentacja projektu ISBD – wymagania**

1. ~~Strona tytułowa:~~

Ciemnozielone robi Janek, Ciemnoniebieskie Hubert

1. Spis treści, numeracja stron.
2. ~~Temat, cel, zakres przedsięwzięcia, użytkownicy systemu.~~
3. ~~Opis wymagań dla projektowanego systemu.~~
4. ~~Lista transakcji dla poszczególnych grup użytkowników~~
5. Dokumentacja zaimplementowanej bazy danych (musi być zgodny z implementacją)
   1. ~~Definicje predykatowe encji i związków~~
   2. ~~ERD~~
   3. **~~Schemat bazy danych ??~~**
   4. ~~Dane przykładowe~~
   5. ~~Narzędzia implementacyjne b.d. (u nas Python+Django + postgres)~~
6. Opis aplikacji bazodanowej (dla jednej bazy może być kilka aplikacji, dla różnych użytkowników):
   1. ~~Tytuł (nazwa) aplikacji, przeznaczenie, użytkownicy, zakres.~~
   2. ~~Specyfikacja wymagań funkcjonalnych aplikacji.~~
   3. ~~Specyfikacja wymagań niefunkcjonalnych aplikacji m.in. wymagania dotyczące platformy systemowej, bazy danych, bezpieczeństwa, wydajności, wykorzystywanych narzędzi programistycznych itp.~~
   4. Mapa wywołań formularzy i raportów.
7. Przykładowe zrzuty ekranów formularzy.
8. Przykłady wydruków raportów.
9. Opis i kod przykładowych procedur/funkcji własnych (opcjonalnie).
10. Zawartość nośnika z systemem – nazwy plików, rozmiary oraz obraz (skan) zawartości folderu.
11. Sposób instalacji/uruchomienia systemu, objaśnienia dodatkowe.
12. Zakończenie – wnioski końcowe, m.in. jaki był cel przedsięwzięcia i czy został osiągnięty, wady i zalety opracowanego systemu.
13. Źródła – literatura, adresy stron WWW.

**Uwagi dodatkowe**

1. Dokumentację należy wydrukować (można dwustronnie) i wraz z płytą CD/DVD (lub pendrivem, który zostanie po skopiowaniu zwrócony) umieścić w teczce papierowej (teczki nie podpisywać). Kartek w teczce nie należy w żaden sposób spinać ani dawać do koszulek (tylko ponumerować).
2. Zawartość płyty CD/DVD/pendrive’a: wszystkie możliwe do skopiowania pliki (źródłowe, instalacyjne), m.in.:
   1. Dokumentacja (plik .doc/.docx) – wersja końcowa
   2. Baza (np. plik .accdb lub kod SQL) – wersja bez żadnych zabezpieczeń, z wypełnionymi danymi
   3. Baza po zabezpieczeniach – wersja instalacyjna (opcjonalnie)
   4. Wszystkie dodatkowe pliki (np. zdjęcia, pliki pomocy itp.)
3. Ważna jest zawartość merytoryczna dokumentacji, układ rozdziałów, kompletność materiału, cel, podsumowanie/wnioski, źródła.
4. Oprócz poprawności merytorycznej ocenie podlega również:
   1. przygotowanie danych,
   2. estetyka pracy, sposób sformatowania tekstu, tabel i rysunków, czytelność i układ pracy, poprawność stylistyczna, językowa (brak błędów ortograficznych, literówek itp.).
   3. sposób prezentacji pracy przez studentów, umiejętność dyskusji, argumentowania, przyjmowania krytyki oraz samooceny pracy.

Co było celem, jaki jest schemat danych, użytkownicy bazy, kto co może robić, jakie narzędzia zostały użyte,

Bardzo dokładny opis aplikacji, wyraźne rzuty ekranów, wypełnione dane przykładowe, wszystkie dane na rzutach uzupełnione. Udokumentowanie jak robić operacje na bazie danych. Przynajmniej jeden przykład sortowania, tworzenia, usuwania, modyfikowania. Jak nie ma usuwania danych to to bezpośrednio napisać. To wypisać w w wymaganiach funkcjonalnych. Na koniec opisujemy zawartość pendriva, dokumentacja źródłowa. Informacje o sposobie instalacji.  
Podsumowanie na koniec jest bardzo ważne. Powtórzyć co było celem i czy cel został osiągnięty i na ile został osiągnięty z punktu widzenia twórcy oprogramowania. Czy za mało wiedzy, czy zła organizacja pracy. Mały rozdział na temat testowania aplikacji. Czy to były testy jednostkowe, czy robiliśmy walidację. Ocenić narzędzia które zostały użyte. Co byśmy zrobili inaczej, jakie były problemy z tego powodu. **Co można było zmienić, czy jest możliwe żeby te zmiany wprowadzić. Czy narzędzia to umożliwiają, czy projekt bazy danych to umożliwia**. Wypisać kilka pozycji literatury z której korzystamy. W narzędziach odnieść się do literatury omawiającej narzędzia. Wszystkie zamieszczone obrazy powinny być numerowane pod. Wszystkie tabele numerujemy i opisujemy nad. Odwołanie do rysunku powinno być po tym jak tabela została umieszczona. Odpowiednie akapity, wyjustowanie. Robimy PDF i drukujemy.

udokumentować ścieżki przechodzenia między ekranami aplikacji.