

Wstęp

W rozwiązywanym zadaniu należało utworzyć aplikację z wykorzystaniem frameworka Laravel w wersji 7.*. Założenia aplikacji zostaną opisane i przedstawione w następnym punkcie.

Założenia projektu

- Cykliczne łączenie i pobieranie danych z dowolnego publicznego API

W celu realizacji łączenia i pobierania danych z API stworzono model, który reprezentuje dane zapisywane w bazie danych oraz repozytorium dla danego modelu. Aby operacja była realizowana cyklicznie utworzono komendę, która raz dziennie pobiera dane z adresu zdefiniowanego w pliku .env.

```
class UpdateBitcoinTrades extends Command
{
    private BitBayHttpClient $client;
    private BitcoinTradesRepository $repository;

    protected $signature = 'bit-bay:update-trades';
    protected $description = 'Update bitcoin trades';

    public function __construct(BitBayHttpClient $client, BitcoinTradesRepository $repository)
    {
        $this->client = $client;
        $this->repository = $repository;

        parent::__construct();
    }

    public function handle(): void
    {
        $this->repository->deleteAll();
        $allTrades = $this->client->getBitcoinTradeHistory();

        foreach ($allTrades as $apiTrade) {
            $trade = new BitcoinTrade();

            $trade->fill([
                'date' => $apiTrade->date,
                'price' => $apiTrade->price,
                'type' => $apiTrade->type,
                'amount' => $apiTrade->amount,
                'tid' => $apiTrade->tid
            ]);

            $this->repository->save($trade);
        }
    }
}
```

Klasa UpdateBitcoinTrades odpowiadająca za cykliczne aktualizowanie danych

- Zapisywanie i aktualizacja wcześniej pobranych danych w bazie danych

Aby zapisywać i aktualizować dane w bazie danych stworzono migracje, które tworzą pożądaną strukturę bazy danych. W repozytorium modelu BitcoinTrade – BitcoinTradesRepository utworzono metody pozwalające na zapis, usuwanie oraz odczyt z bazy danych.

```
class CreateBitcoinTrades extends Migration
{
    public function up(): void
    {
        Schema::create( table: 'bitcoin_trades', function (Blueprint $table) {
            $table->id();
            $table->string( column: 'date');
            $table->decimal( column: 'price');
            $table->string( column: 'type');
            $table->decimal( column: 'amount', total: 12, places: 8);
            $table->string( column: 'tid');
            $table->timestamps();
        });
    }

    public function down(): void
    {
        Schema::dropIfExists( table: 'bitcoin_trades');
    }
}
```

Przykładowa migracja – tabela dotycząca danych pobieranych z API (transakcji bitcoinami)

- Wyświetlenie danych w postaci tabelarycznej na stronie dostępnej tylko dla zalogowanego użytkownika (prosty interfejs UI Bootstrap umożliwiający wyświetlanie, filtrowanie oraz wyszukiwanie zgromadzonych danych)

W celu zwrócenia widoku użytkownikowi wykorzystano kontroler HomeController, która zwraca zalogowanemu użytkownikowi widok strony z dołączonymi danymi w bazie danych dotyczącymi transakcjami bitcoinami. Wyświetlenie, filtrowanie i wyszukiwanie zgromadzonych danych zrealizowano za pomocą biblioteki JavaScript – DataTables.

Laravel Login Register

Login

E-Mail Address

Password

☐ Remember Me

[Forgot Your Password?](#)

Widok formularza logowania

Laravel Hello, test Logout

Data about Bitcoin trades

Show 10 entries
 Search:

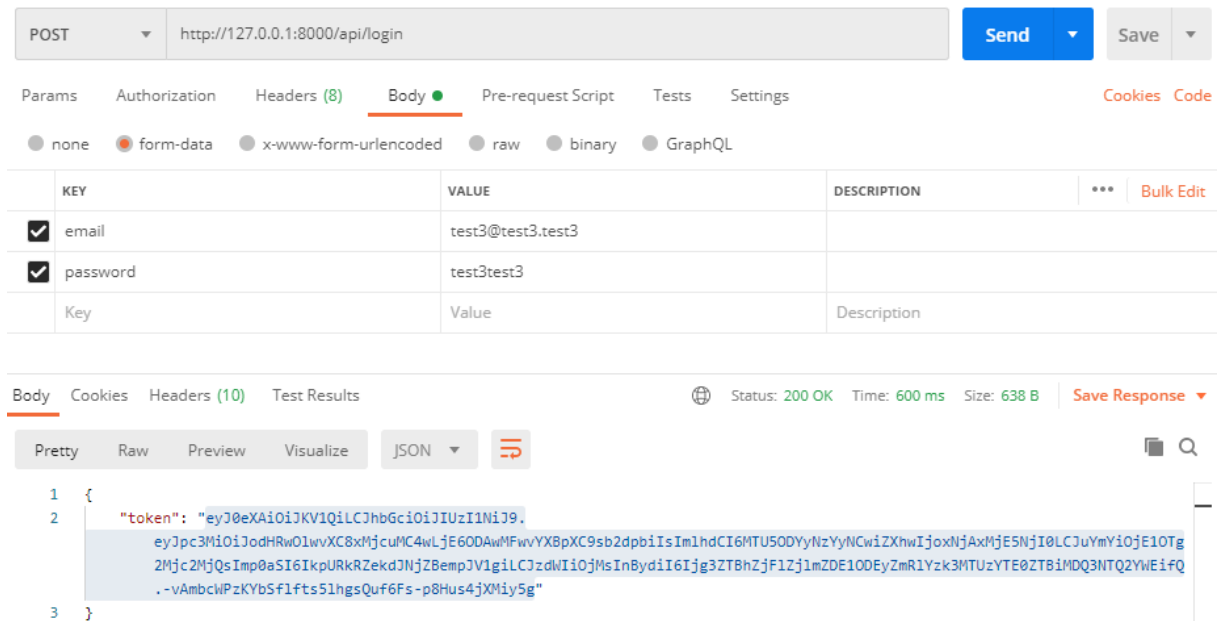
Date	Price	Type	Amount	Updated at
1396096603	4400.00	sell	0.01136400	2020-08-28 11:53:14
1396417881	1519.00	sell	0.03400000	2020-08-28 11:53:15
1396428768	1519.00	sell	0.03600000	2020-08-28 11:53:15
1396202831	1498.00	sell	0.17000000	2020-08-28 11:53:15
1396197042	1491.00	sell	0.30000001	2020-08-28 11:53:14
1396197567	1480.00	sell	0.05000000	2020-08-28 11:53:15
1396473920	1475.00	sell	0.03395250	2020-08-28 11:53:16
1396473920	1472.00	sell	0.26604700	2020-08-28 11:53:16
1396510634	1472.00	sell	0.03500000	2020-08-28 11:53:16
1396513512	1472.00	sell	0.06072000	2020-08-28 11:53:16

Showing 1 to 10 of 18 entries (filtered from 50 total entries) Previous 1 2 Next

Widok udostępniający zalogowanemu użytkownikowi tabelę z danymi pobranymi z API, wraz z możliwością filtrowania oraz sortowania

- Stworzenie API (REST), które zwraca listę wszystkich dostępnych rekordów oraz jeden wskazany rekord (dostęp do API powinien być tylko dla uwierzytelnionych i zautoryzowanych użytkowników)

Utworzono API, które pozwala użytkownikowi w przypadku udanego uwierzytelnienia, wygenerować token wymagany do zwrócenia danych pobranych z zewnętrznego API. API posiada możliwość zwrócenia wszystkich rekordów lub jednego - wybranego na podstawie id przekazanego w parametrze.



POST http://127.0.0.1:8000/api/login Send Save

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings Cookies Code

none form-data x-www-form-urlencoded raw binary GraphQL

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> email	test3@test3.test3	
<input checked="" type="checkbox"/> password	test3test3	
Key	Value	Description

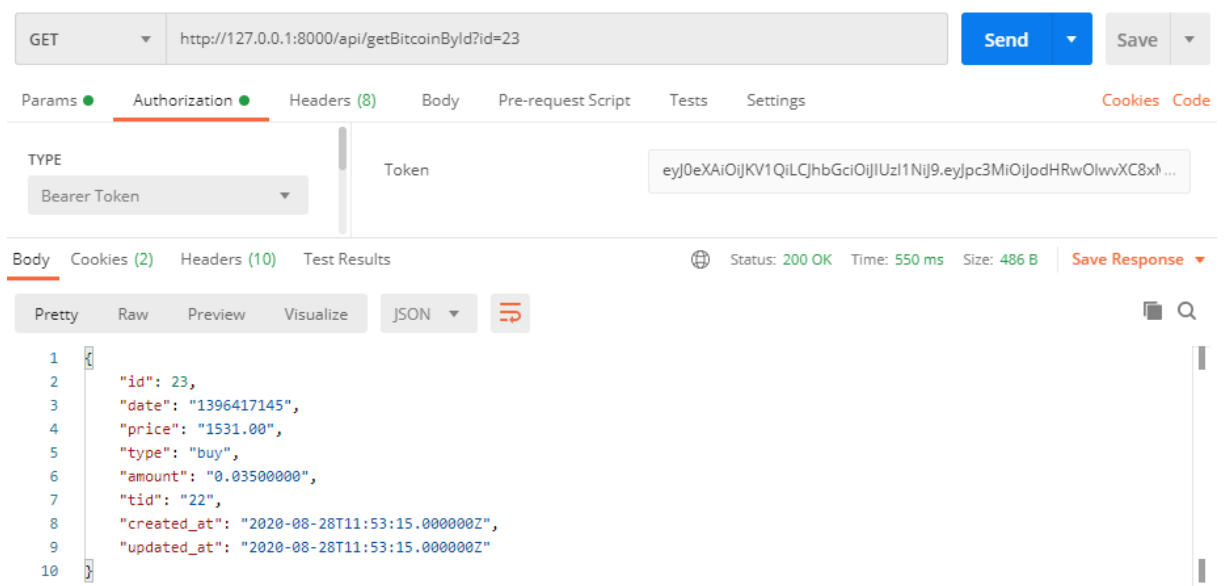
Body Cookies Headers (10) Test Results Status: 200 OK Time: 600 ms Size: 638 B Save Response

Pretty Raw Preview Visualize JSON ↻

```

1 {
2   "token": "eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwOlwvXC8xMjcuc2V1ZjE6ODAwMFwvYXBpXC9sb2dpbiIsIm1hdCI6MTU5ODYyNzYyNCwiZXhwIjoxNjE5MjI0LCJyYmYiOiJ0e10Tg2Mjc2MjcQsImp0aSI6IkpURkRZekdJNjZBempJV1giLCJzdWIiOiJMsInBydiI6IjE6Ijg3ZTBhZjF1Zj1mZDE0OEY2mR1Yz3MTUzYTE0ZTBiMDQ3NTQ2YWEifQ.-vAmbckWPzKYbSf1fts5lHgsQuf6Fs-p8Hus4jXMiy5g"
3 }
```

Logowanie za pomocą API



GET http://127.0.0.1:8000/api/getBitcoinById?id=23 Send Save

Params **Authorization** Headers (8) Body Pre-request Script Tests Settings Cookies Code

TYPE Bearer Token Token eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwOlwvXC8xMjcuc2V1ZjE6ODAwMFwvYXBpXC9sb2dpbiIsIm1hdCI6MTU5ODYyNzYyNCwiZXhwIjoxNjE5MjI0LCJyYmYiOiJ0e10Tg2Mjc2MjcQsImp0aSI6IkpURkRZekdJNjZBempJV1giLCJzdWIiOiJMsInBydiI6IjE6Ijg3ZTBhZjF1Zj1mZDE0OEY2mR1Yz3MTUzYTE0ZTBiMDQ3NTQ2YWEifQ.-vAmbckWPzKYbSf1fts5lHgsQuf6Fs-p8Hus4jXMiy5g

Body Cookies (2) Headers (10) Test Results Status: 200 OK Time: 550 ms Size: 486 B Save Response

Pretty Raw Preview Visualize JSON ↻

```

1 {
2   "id": 23,
3   "date": "1396417145",
4   "price": "1531.00",
5   "type": "buy",
6   "amount": "0.03500000",
7   "tid": "22",
8   "created_at": "2020-08-28T11:53:15.000000Z",
9   "updated_at": "2020-08-28T11:53:15.000000Z"
10 }
```


Pobieranie danych dotyczące wskazanego rekordu (wymagane dołączenie tokena)




- Każde połączenie z API powinno być zapisywane w logach w bazie danych

Każde połączenie z API jest zapisywane w tabeli logów w bazie danych. Wykorzystano do tego event oraz listener dla tego eventu.

Tabela w bazie danych z logami połączeń z API

Wszystkie endpointy API zostały udokumentowane narzędziem Swagger i są dostępne pod adresem `/api/documentation`.

[Authorize](#) 

auth	▼
POST /api/login Sign in	
refreshToken	▼
GET /api/refreshToken Get new token	
getAll	▼
GET /api/getAllBitcoins Get all bitcoin information	
getById	▼
GET /api/getBitcoinById Get specific trade information	
Schemas	▼
bitcoinTrades >	

Endpointy opisane w swaggerze