Studia i Materiały Informatyki Stosowanej, Tom 13, Nr 1, 2021
str. 42-46

ZDECENTRALIZOWANE APLIKACJE Z WYKORZYSTANIEM TECHNOLOGII BLOCKCHAIN

Maciej Sitko[1], Mieczysław Jagodziński[2]

[1] *Politechnika Śląska, Wydział Automatyki, Elektroniki i Informatyki, 44-101 Gliwice, ul. Akademicka 16*
[2] *Politechnika Śląska, Wydział Automatyki, Elektroniki i Informatyki, 44-101 Gliwice, ul. Akademicka 16*
email: mieczyslaw.jagodzinski@polsl.pl

Streszczenie: *W artykule przedstawiono przykładową zdecentralizowaną aplikację wykorzystującą blockchain Ethereum. Aplikacja w sposób uproszczony symuluje działanie systemu zarządzającego łańcuchem dostaw, który do przechowywania danych produktu wykorzystuje smart kontrakty. Wskazano zalety wykorzystania technologii blockchain w podanym przypadku oraz zaproponowano możliwe scenariusze rozwoju aplikacji w przyszłości.*

Słowa kluczowe: *Blockchain, Ethereum, decentralizacja, inteligentne kontrakty, łańcuch dostaw*

Decentralized applications using blockchain technology

***Abstrac**t: The article presents an example of a decentralized application using the Ethereum blockchain. The application in a simplified way simulates the operation of the supply chain management system, which uses smart contracts to store product data. The advantages of using blockchain technology in the given case were indicated and possible application development scenarios in the future were proposed.*

***Key words:** Blockchain, Ethereum, decentralization, smart contracts, supply chain*

1. Wprowadzenie

Technologia blockchain swoją popularność w ostatnich latach zawdzięcza nie tylko licznym zastosowaniom finansowym, ale przede wszystkim rozwiązaniom wprowadzonym w branżach pozafinansowych. Przełom ten nastąpił w 2013 roku za sprawą Vitalika Buterina oraz założonego przez niego Ethereum, z czego skorzystały m.in. branża spożywcza, transportowa czy nieruchomości. Platforma pozwala deweloperom na tworzenie własnych aplikacji opierających swoje działanie o blockchain, czyli tzw. dApps (ang. decentralized applications). Zdecentralizowane aplikacje podłączone są do EVM (Ethereum Virtual Machine), która wykonuje instrukcje zawarte w inteligentnych kontraktach. Inteligentne kontrakty (ang. smart contracts) to krótkie programy skryptowe znacznie rozszerzające możliwości interakcji użytkownika z blockchainem. W porównaniu do Bitcoina, transakcje nie muszą oznaczać tylko transferu pieniędzy. Za pośrednictwem transakcji w Ethereum można przesyłać dowolne dane, które następnie zapisywane są w blokach. Chociaż z założenia Ethereum to platforma służąca do obsługi smart kontraktów, twórcy Ethereum wprowadzili Ether, który jest kryptowalutą. Głównym celem wprowadzenia własnej kryptowaluty w Ethereum nie jest transfer pieniędzy, lecz monetyzacja działań podjętych w zdecentralizowanych aplikacjach. Każda transakcja wykonana przez EVM zużywa gaz, czyli miarę zużycia pamięci mocy obliczeniowej, jaką maszyna musiała podjąć, aby wykonać zapisane w kontrakcie instrukcje. Dla prostych operacji opłaty za gaz są bardzo małe, jednak w ostatnim czasie ceny te znacznie wzrosły za sprawą dużego zainteresowania Ethereum i idącym za tym przeciążeniem sieci.

W artykule przedstawiona zostanie przykładowa aplikacja działająca z wykorzystaniem blockchainu Ethereum. Aplikacja ma na celu symulację systemu zarządzania łańcuchem dostaw z wykorzystaniem łańcucha bloków do zapisu i odczytu danych o wprowadzonych produktach.

2. Łańcuch bloków a łańcuch dostaw

Zaraz po finansach, branże spożywcza i transportowa stały się naturalnymi wyborami dla przedsiębiorstw zainteresowanych technologią bloków. Branże te wykorzystują wiele walorów blockchainu - przede wszystkim transparentność i bezpieczeństwo danych, ale także możliwości inteligentnych kontraktów..

Transparentność w branży spożywczej to coraz częściej pojawiający się trend. Klient chce mieć gwarancję, że informacje o kupowanym przez niego produkcie są prawdziwe. W klasycznych systemach sprzedaży musi on zaufać sprzedawcy w istotnych

kwestiach jak np. kraj pochodzenia danego produktu. Transparentność w takich sytuacjach może zapewnić blockchain. Jako że blockchain to rejestr publiczny, konsument może mieć wgląd we wszystkie dane wprowadzone do bloków. Żeby dane te nie pozostawały bez pokrycia, blockchain umożliwia także wprowadzenie niezbędnych plików z dokumentami potwierdzającymi autentyczność wprowadzanych informacji. Bezpieczeństwo i niezmienność tych danych jest zapewnione przez złożoną strukturę kryptograficzną blockchainu.

Pochodzenie produktu jest istotne nie tylko z perspektywy świadomości klientów o kupowanym towarze, ale także ich bezpieczeństwa i zdrowia. W klasycznych łańcuchach dostaw dane na poziomie poszczególnych ogniw dystrybucji są bardzo zróżnicowane. Każda firma prowadzi swoją własną bazę danych, a niektóre z nich wciąż są formie papierowej. W ten sposób dane o konkretnym produkcie na przestrzeni jednego łańcucha dostaw są niejednolite, co wprowadza chaos w dokumentacji. Chaos ten staje się szczególnie niebezpieczny, gdy podczas produkcji dojdzie np. do skażenia całej partii towaru. Gdy skażenie zostanie zauważone w dalszych etapach dostawy należy jak najszybciej zlokalizować miejsce, z którego wyszła wadliwa partia. Przebrnięcie przez wszystkie bazy danych każdej z firm, która przewoziła produkt, będzie trwać nawet kilka dni, co może przynieść tragiczne w skutkach konsekwencje. Przy ujednoliconym systemie zarzadzania łańcuchem dostaw w blockchainie informacje te można zdobyć w ciągu kilku sekund.

Działanie łańcuchów dostaw może również zostać usprawnione przez zastosowanie inteligentnych kontraktów. Skrypty zamieszczone w kontrakcie na bieżąco odczytują dane zapisane w blockchainie. W ten sposób możliwe jest np. kontrolowanie daty spożycia produktów. Gdy producent wprowadzi taką informację do systemu, skrypt zawarty w kontrakcie może sprawdzać ją automatycznie, a w przypadku przeterminowania produktu – wysłać komunikat do odpowiedniego podmiotu. Innym sposobem wykorzystania kontraktów jest kontrola warunków przechowywania produktów. Gdy przewożony towar wymaga specyficznych warunków, np. odpowiednią temperaturę przechowywania lub poziom wilgotności powietrza, smart kontrakt może odczytywać te wartości z odpowiednich czujników i porównywać je z wytycznymi producenta.

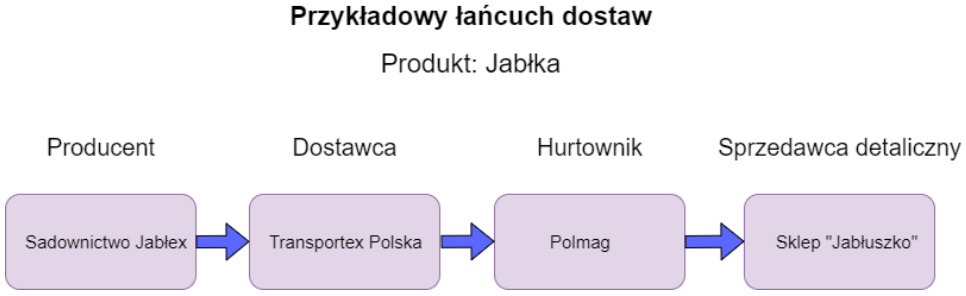
Zaprezentowana w artykule aplikacja to tylko uproszczony sposób wykorzystania blockchainu do zapisu i odczytu danych o wprowadzonym do systemu produkcie. Na rys. 1 przedstawiono schemat łańcucha dostaw zaproponowanego w aplikacji oraz przykładowy produkt wprowadzony w systemie.

3. Wykorzystane narzędzia

Do stworzenia strony internetowej wykorzystano Django, czyli otwarty framework napisany w języku Python. Django opiera swoje działanie na modelach ściśle powiązanych z bazą danych, widokach hermetyzujących logikę aplikacji oraz szablonach, które odpowiadają za renderowanie widoku strony. Zorientowana w ten sposób architektura nazywana jest MVT (Model – View – Template) i została wykorzystana w tym projekcie. Zastosowanie modeli pozwoliło na łatwą integrację aplikacji ze smart kontraktami dla każdego z ogniw łańcucha dostaw. Django posiada również rozbudowany system logowania i rejestracji z zaimplementowaną weryfikacją uzupełnianych pól oraz przejrzysty panel administratora, co znacznie ułatwiło pracę podczas częstego dodawania nowych użytkowników będących ogniwami łańcucha.

Smart kontrakty zostały napisane w języku Solidity, który jest jednym z najpopularniejszych wyborów programistów. Kod języka interpretowany jest przez maszynę EVM, która wykonuje zawarte w kontrakcie instrukcje. Kontrakty w Solidity są podobne do klas w językach zorientowanych obiektowo. Każdy uczestnik łańcucha dostaw w aplikacji posiada odpowiadający jego roli kontrakt z dedykowanymi funkcjonalnościami.

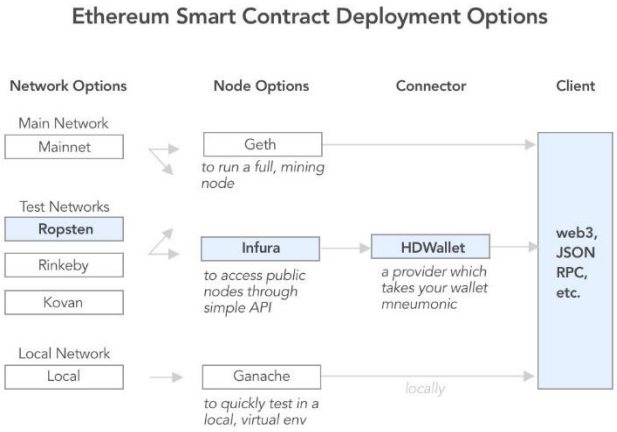
Do wdrożenia smart kontraktów do sieci Ethereum wykorzystano narzędzie Truffle. Truffle to rozbudowane środowisko ułatwiające programistom pracę z EVM w sieci głównej oraz sieciach testowych. Korzystanie z sieci testowych jest darmowe, natomiast przy łączeniu aplikacji z siecią główną, czyli Mainnetem, należy brać pod uwagę rzeczywiste opłaty za wykonywane operacje.



Rysunek 1. Schemat łańcucha dostaw dla produktu testowego

4. Połączenie aplikacji z blockchainem

Schemat połączenia z siecią Ethereum pokazano na rys. 2. Ethereum umożliwia dostęp do kilku sieci testowych (Rinkeby, Ropstein, Kovan, Goerli) oraz do głównej sieci Mainnet. Możliwe jest także uruchomienie własnego blockchainu na serwerze lokalnym za pośrednictwem aplikacji Ganache. Konta tworzone w sieciach testowych można doładowywać za darmo wirtualnym Etherem, który nie ma rzeczywistej wartości.



Rysunek 2. Schemat dostępnych możliwości połączenia aplikacji z blockchainem Ethereum. [3]

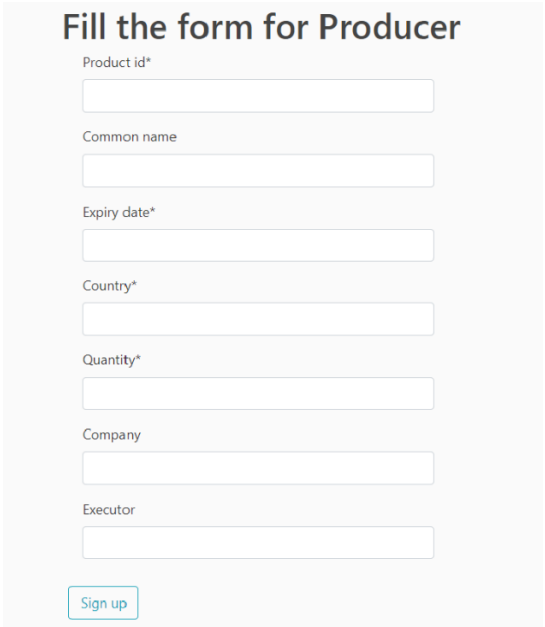
W aplikacji wykorzystano sieć Rinkeby, która wyróżnia się łatwym dostępem do wirtualnego Etheru, co było ważne z uwagi na duże zużycie gazu przy przeprowadzaniu testów. Do połączenia z siecią służy węzeł sieciowy (ang. node). W sieci Ethereum węzeł można zainstalować samodzielnie za pośrednictwem aplikacji Geth lub połączyć się z publicznym węzłem, np. za pośrednictwem Infury. Infura zapewnia użytkownikowi system monitorowania swojej aktywności w blockchainie oraz prywatny klucz dostępu do API. Do połączenia się z blockchainem wymagany jest także portfel. Portfel zawiera jedno lub więcej kont użytkownika, które posiadają swój klucz publiczny (adres) służący do komunikacji w sieci oraz klucz prywatny, który jest niezbędny do stworzenia cyfrowego podpisu. Ulepszoną wersją zwykłego portfela są tzw. HD-Wallety (Hierarchical Deterministic Wallet). HD–Wallet tworzy klucze prywatne i publiczne na podstawie sekwencji 12 do 24 losowych słów, nazwanej „mneumonic". Słowa te nie mogą być przypadkowe, tylko wybierane są z listy 2048 określonych słów, która jest zaakceptowana przez wszystkich użytkowników sieci. Ostatnią warstwę łączności z blockchainem stanowi biblioteka Web3, która posiada rozbudowane API umożliwiające m.in. wywoływanie funkcji smart kontraktów, wykonywanie transakcji czy odczytywanie informacji o wygenerowanych w blokach w łańcuchu. W aplikacji korzystano z wersji biblioteki dedykowanej językowi Python, czyli Web3py.

5. Widoki i funkcjonalności aplikacji

Strona główna aplikacji została zaprezentowana na rys. 3. Z pełnych możliwości strony mogą korzystać tylko zarejestrowani użytkownicy, czyli osoby pełniące pewną rolę w łańcuchu dostaw. Mogą oni dodawać i odczytywać dane z blokchainu. Formularz rejestracji przedstawiony jest na rys. 5. Aplikacji mogą używać także niezarejestrowani użytkownicy, czyli potencjalni klienci, którzy będą chcieli odczytać z blokchainu dane o zakupionym produkcie.

Rysunek 3. Strona domowa.

Dodawanie danych do blockchain za pośrednictwem API z biblioteki Web3py. Informacje o produkcie wpisywane są przez uprawnionych użytkowników za pośrednictwem formularzy dedykowanych poszczególnym rolom w łańcuchu dostaw. Na rys. 4 przedstawiono przykładowy formularz dla producenta.



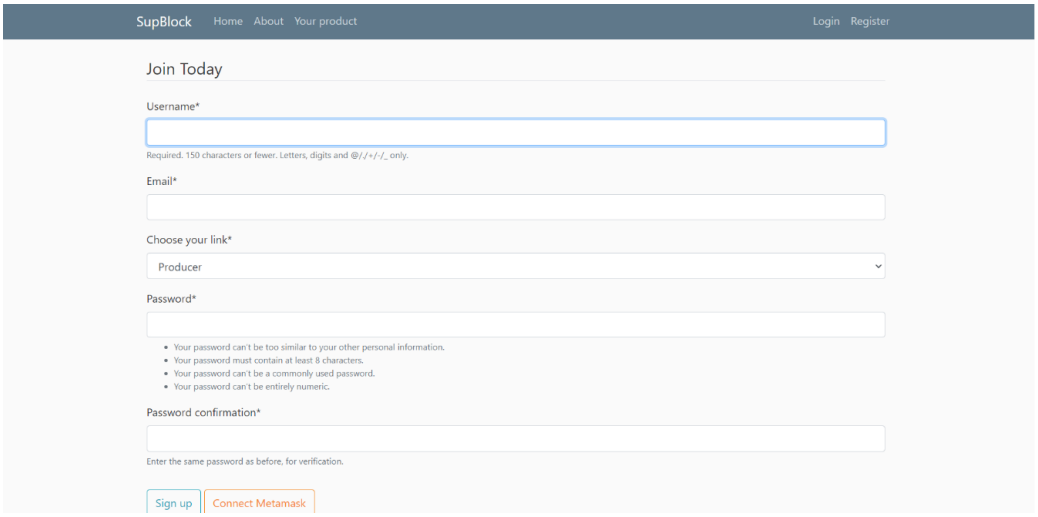
Rysunek 4. Przykładowy formularz dla producenta.

Po uzupełnieniu formularza, dane przekazywane są do smart kontraktów, które wcześniej zostały wdrożone do sieci. Kontrakty, podobnie jak konta użytkownika, posiadają swoje klucze publiczne (adresy), więc interakcja z kontraktem polega na wysłaniu specjalnej transakcji na konkretny adres. Zapis do blockchainu kosztuje pewną niewielką ilość Etheru, natomiast odczytywanie danych jest darmowe. Pozwala to niezalogowanym użytkownikom na darmowe korzystanie z aplikacji.

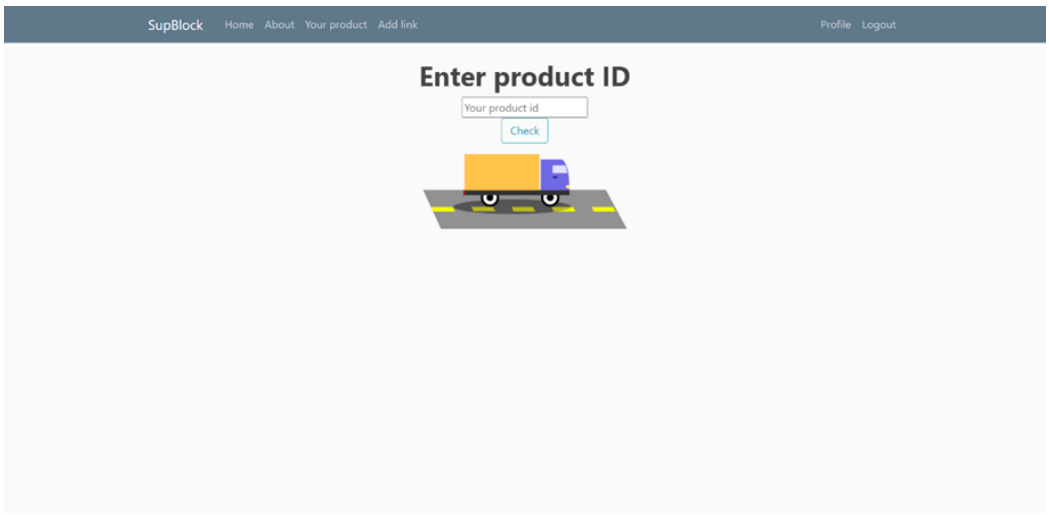
Sprawdzenie informacji o produkcie znajduje się w zakładce „Your product". W wyświetlanym polu (rys. 6) należy wpisać numer identyfikacyjny produktu. Zakłada się, że numer ten będzie wydrukowany na opakowaniu produktu zakupionego w sklepie. Jeśli produkt zostanie znaleziony w blockchainie aplikacja wyświetli pełną trasę, jaką pokonał towar podczas dostawy, od producenta do sprzedawcy detalicznego (rys. 7).

Na rys. 8 przedstawiono szczegółowy widok dla produktu testowego. Jako że każda akcja w blockchainie jest transparentna, wszystkie dokonane transakcje można zweryfikować na stronie www.ethscan.io, która wyświetla dane wszystkich operacji dla danego konta bądź kontraktu, tj.:

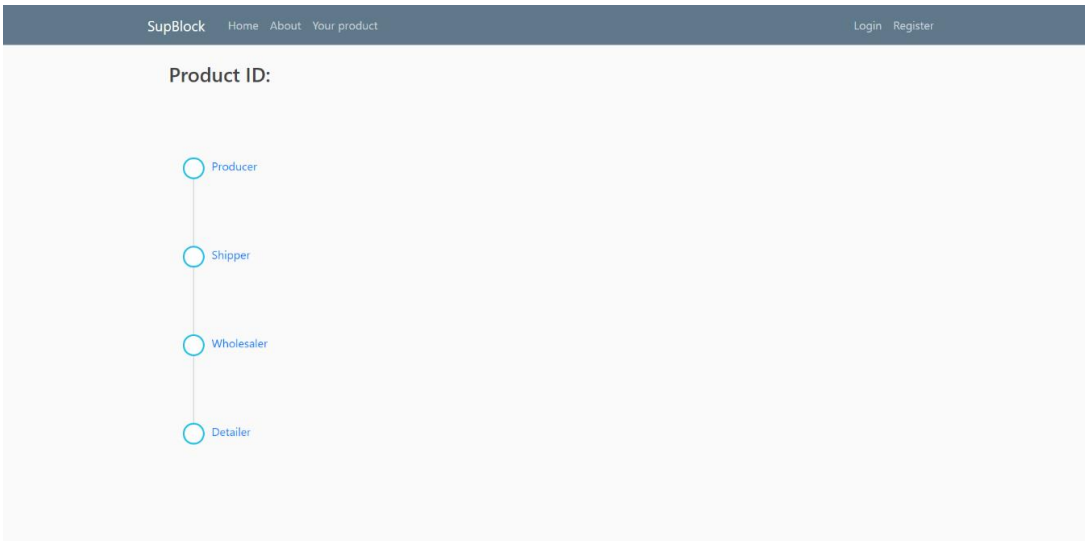
- Numer hash transakcji
- Numer bloku, do którego transakcja została zapisana
- Czas, który minął od zapisania transakcji do teraz
- Adres nadawcy
- Adres odbiorcy
- Ilość wysłanego Etheru
- Podatek za transakcję



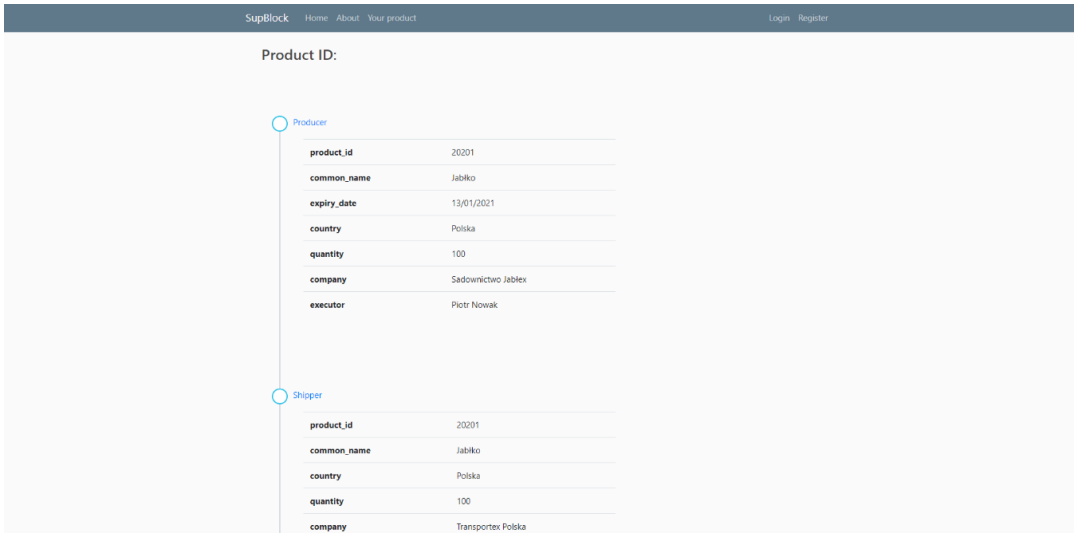
Rysunek 5. Strona rejestracji.



Rysunek 6. Widok wpisywania numeru ID produktu.



Rysunek 7. Lista ogniw łańcucha dostaw wyświetlona po wpisaniu ID produktu.



Rysunek 8. Szczegółowy widok produktu.

6. Podsumowanie

Przedstawiona w artykule aplikacja jest tylko uproszczoną symulacją systemu zarządzania łańcuchem dostaw. Mimo to dobrze prezentuje podstawowe mechanizmy pozwalające na wykorzystanie blockchainu w branży spożywczej. Wdrożenie aplikacji do rzeczywistego systemu wymaga dużo więcej pracy, przede wszystkim całkowitej reorganizacji struktury informatycznej firmy zainteresowanej takim rozwiązaniem. Dodatkowo, aplikacja powinna zostać rozszerzona o możliwość dodawania plików z dokumentami świadczącymi o prawdziwości wprowadzonych danych.

Literatura

1. Ethereum Foundation. *Ethereum Whitepaper*. https://ethereum.org/en/whitepaper/ [data dostępu: 2021-03-12]
2. Sitko Maciej, Praca inżynierska *Smart contracts z wykorzystaniem technologii blockchain*
3. Zhu Nicole. *5 minute guide to deploying smart contracts with Truffle and Ropsten* https://medium.com/coinmonks/5-minute-guide-to-deploying-smart-contracts-with-truffle-and-ropsten-b3e30d5ee1e [data dostępu: 2021-03-12]