**WYŻSZA SZKOŁA ZARZĄDZANIA I BANKOWOŚCI W KRAKOWIE**

**Wydział Zarządzania, Finansów i Informatyki**

**KIERUNEK: Informatyka**

**SPECJALNOŚĆ: Systemy i sieci komputerowe**

PRACA DYPLOMOWA

**Daniel Adam Nyikó**

**Projekt witryny internetowej zawierającej informacje o kryptowalutach i technologii blockchain**

PROMOTOR:

**dr Edward Szczypka**

**KRAKÓW 2022**

**Spis treści**

[Wstęp 4](#_Toc114954886)

[1 Cel i założenia 5](#_Toc114954887)

[1.1 Cel projektu 5](#_Toc114954888)

[1.2 Motywacja autora 6](#_Toc114954889)

[1.3 Otoczenie projektu 7](#_Toc114954890)

[1.3.1 Otoczenie ekonomiczne 8](#_Toc114954891)

[1.3.2 Otoczenie technologiczne 8](#_Toc114954892)

[1.3.3 Otoczenie konkurencyjne 9](#_Toc114954893)

[2 Specyfikacja wymagań 10](#_Toc114954894)

[2.1 Dziedzinowy słownik pojęć 10](#_Toc114954895)

[2.2 Wymagania niefunkcjonalne użytkownika 10](#_Toc114954896)

[2.3 Wymagania niefunkcjonalne systemowe 11](#_Toc114954897)

[2.4 Model wymagań 11](#_Toc114954898)

[2.5 Opis scenariuszowy 15](#_Toc114954899)

[3 Propozycja bazy danych dla systemu 23](#_Toc114954900)

[3.1 Dokumenty oraz ich struktura 23](#_Toc114954901)

[3.2 Połączenie z bazą danych 26](#_Toc114954902)

[4 Projekt interfejsu użytkownika 27](#_Toc114954903)

[4.1 Trzon interfejsu 27](#_Toc114954904)

[4.2 Lista kryptowalut oraz artykułów 28](#_Toc114954905)

[4.2.1 Strona główna panelu administratora 28](#_Toc114954906)

[4.2.2 Lista rekordów 29](#_Toc114954907)

[4.2.3 Dodawanie nowego rekordu 32](#_Toc114954908)

[4.2.4 Aktualizowanie rekordu 33](#_Toc114954909)

[5 Implementacja 34](#_Toc114954910)

[5.1 Założenia 34](#_Toc114954911)

[6 Testy aplikacji 35](#_Toc114954912)

[6.1 Testy API 35](#_Toc114954913)

[6.2 Testy działania systemu 35](#_Toc114954914)

[7 Środowisko oraz narzędzia 37](#_Toc114954915)

[7.1 Wykorzystane technologie 37](#_Toc114954916)

[7.1.1 HTML 37](#_Toc114954917)

[7.1.2 CSS 37](#_Toc114954918)

[7.1.3 MongoDB 38](#_Toc114954919)

[7.1.4 JavaScript 38](#_Toc114954920)

[7.1.5 Node.js 38](#_Toc114954921)

[7.1.6 Express 39](#_Toc114954922)

[7.1.7 API 40](#_Toc114954923)

[7.2 Narzędzia 40](#_Toc114954924)

[7.2.1 Hyper 40](#_Toc114954925)

[7.2.2 Git 41](#_Toc114954926)

[7.2.3 Node Package Menager 41](#_Toc114954927)

[7.2.4 Heroku 42](#_Toc114954928)

[7.2.5 Atom 42](#_Toc114954929)

[7.2.6 Microsoft Visual Studio Code 42](#_Toc114954930)

[7.2.7 Robo3T 43](#_Toc114954931)

[7.2.8 Postman 43](#_Toc114954932)

[8 Podsumowanie 44](#_Toc114954933)

[8.1 Wykonane prace 44](#_Toc114954934)

[8.2 Założenia rozbudowy systemu 45](#_Toc114954935)

[Bibliografia 46](#_Toc114954936)

[Książki 46](#_Toc114954937)

[Netografia 46](#_Toc114954938)

[Spis tabel oraz rysunków 47](#_Toc114954939)

[Spis tabel 47](#_Toc114954940)

[Spis rysunków 47](#_Toc114954941)

# Wstęp

W przeciągu ostatnich lat został odnotowany ogromny wzrost zainteresowania rynkiem kryptowalut, zarówno u klienta personalnego jak i przedsiębiorców. W 2021 szacowana wartość rynku kryptowalut osiągnęła szczyt w wysokości 2.9 biliona amerykańskich dolarów (USD). Przewiduje się, że w przeciągu następnych 10 lat rynek kryptowalut może osiągnąć wartość nawet 4,9 biliona dolarów (USD). Tak szybki rozwój rynku pozwala na niemal bezgraniczne możliwości zarówno dla firm zajmujących się wymianą kryptowalut, jak i klientów personalnych.

Aby jednak podjąć właściwą decyzję, w którą kryptowalutę i idącą z nią w parze technologię warto zainwestować, warto zdobyć niezbędną do tego wiedzę. W obecnej epoce internetu problemem nie jest znalezienie informacji lecz rzetelność dostępnych na blogach, portalach społecznościowych oraz serwisach informacyjnych wiadomości. Jednym z najlepszych przykładów tego jak brak odpowiednio nagłośnionych informacji jest w stanie poważnie wpłynąć na przeciętnego inwestora jest sytuacja z 2012, kiedy to rozpoczęła się sławna afera związana ze spółką Amber Gold. Można spekulować, że jeśli więcej osób porządnie przyjrzało się sytuacji problem mógłby zostać uniknięty a przynajmniej liczba ofiar tego oszustwa mogła by zostać zminimalizowana.

Obecnie większość powszechnie dostępnych informacji znajduje się w internecie. Lwia część firm odnoszących sukcesy w branży informacyjnej oraz medialnej korzysta z systemu lub wielu systemów zarządzania treścią, tzw. CMS (ang. Content Management System). Dzięki nim, można w dużej mierze zautomatyzować niemarginalną większość wewnętrznych procesów, jednocześnie minimalizując nakład pracy potrzebny do ich wykonania.

Przedmiotem niniejszej pracy inżynierskiej będzie autorska aplikacja webowa typu CMS, która umożliwi jej administratorowi tworzenie spersonalizowanych treści oraz witryna internetowa, która umożliwi udostępnianie ich szerszemu gronu czytelników. Praca omówi praktyczny rezultat tworzenia i aktualizacji wcześniej wymienionej aplikacji. W niniejszej pracy zostaną przedstawione też przyszłe możliwości rozwoju oraz implementacji aplikacji w szerszym systemie.

# Cel i założenia

W tym rozdziale autor opisał jakie będą cele i założenia projektu aplikacji webowej służącej do zarządzania treścią oraz samej witryny.

## Cel projektu

Głównym celem projektu będzie poruszenie i rozwiązanie niżej wymienionych problemów:

• Pierwszym problemem poruszanym w pracy będzie stworzenie aplikacji webowej, która będzie posiadała aktualne informacje o kryptowalutach i technologii z jakich korzystają.

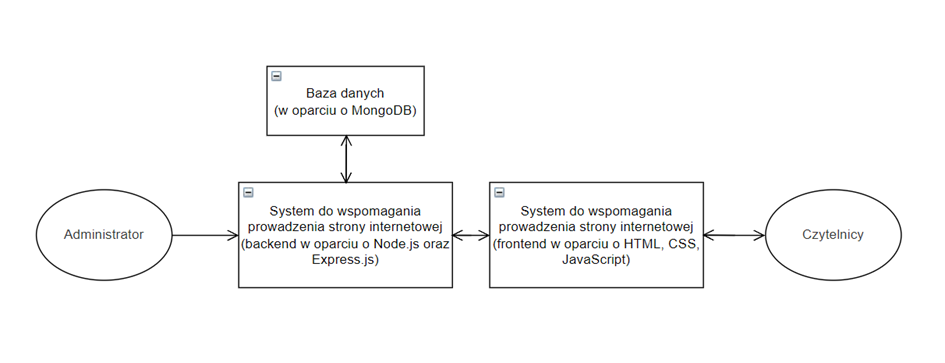
• Drugim zagadnieniem będzie napisanie aplikacji, która pozwoli administratorowi i użytkownikowi na udostępnianie nabytej wiedzy w formie artykułów, jednocześnie dając możliwość modyfikacji danych zawartych na stronie z każdego miejsca na świecie posiadającego internet i dostęp do przeglądarki internetowej.

Do stworzenia aplikacji będą wykorzystane technologie, które pozwalają na stworzenie zarówno front-endu widocznego po stronie klienta jak i zaprojektowanie back-endu stojącego po stronie serwera. W założeniu domyślnym źródłem danych będzie serwer z bazą danych. Do poprawnego działania aplikacji wymagane będą:

* stałe połączenie z siecią oraz serwerem
* dostęp do przeglądarki internetowej

Proponowany system w postaci strony internetowej ma być dostępny zarówno na komputerze jak i urządzeniach mobilnych. Założeniem jest, że część odpowiadająca za udostępnianie i modyfikowanie informacji będzie prosta w obsłudze i szybka niezależnie na jakim sprzęcie zostanie uruchomiona. Część dla czytelników ma być łatwo dostępna, responsywna i czytelna zarówno na ekranie laptopa czy komputera jak i na wyświetlaczach dostępnych w nowoczesnych smartfonach.

**Rysunek 1. Koncept systemu**

****

Źródło: opracowanie własne, Visual Paradigm Community Edition

## Motywacja autora

W dzisiejszym świecie wraz z rozwojem technologii rośnie także liczba przekazywanych informacji i danych. Problem ten jest przede wszystkim widoczny na serwisach społecznościowych takich jak facebook czy tweeter, gdzie do samego stworzenia konta wymagane jest podanie sporej ilości prywatnych danych. Dane te są później przetwarzane zgodnie polityką firmy, na którą użytkownik musi się zgodzić. Sprawa nie wygląda inaczej gdy chodzi tworzenie kont bankowych, oszczędnościowych czy maklerskich. Poprzez tworzenie takich kont, firmy stają się posiadaczem sporej ilości informacji. Zaprzedajemy prywatność i anonimowość w zamian za wygodę, spokój i bezpieczeństwo, które często jest złudne.

Drugim problemem z jakim mierzy się dzisiejszy świat to ogromna ilość sprzecznych wiadomości w mediach i w internecie. Każda stacja przedstawia własną wersję wydarzeń przez co ciężko jest uzyskać pełny obraz na jakikolwiek temat. Wybuch pandemii pogłębił tylko wątpliwości społeczeństwa dotyczące mediów oraz ich wiarygodności to samo tyczy się rządów poszczególnych państw. Masowe zwolnienia i rosnąca w drastycznym tempie inflacja pieniądza sprawiły, że społeczeństwo stało się bardziej otwarte na inne formy przechowywania majątku.

Wymienione wyżej czynniki sprawiły, że autor zaczął się zastanawiać w jaki sposób można uniknąć lub zminimalizować ilość podawanych prywatnych informacji jednocześnie korzystać z systemów płatności jakie nowoczesny świat oferuje. W pewnym momencie natknął się na bitcoina i inne kryptowaluty oraz technologię zwaną blockchain. Na pierwszy rzut oka zdają się one rozwiązywać niektóre z problemów, które zostały wyżej wymienione.

Autor zdecydował się stworzyć aplikację webową, która będzie posiadać aktualne informacje o kryptowalutach, technologiach z jakich korzystają, wiadomości oraz ich aktualną cenę. Aplikacja ma służyć ludziom jako źródło informacji oraz przewodnik po szybko rozwijającym się świecie kryptowawalut, ma służyć administratorowi jako narzędzie do tworzenia spersonalizowanych postów, a developerowi jako portfolio umiejętności Web Developerskich.

## Otoczenie projektu

Sukces projektu zależy od wielu czynników. Poniżej autor opisał, jakie czynniki mogą sprzyjać lub stanowić wyzwanie w drodze aplikacji do odniesienia nieocenionego sukcesu.

Autor przede wszystkim skupił się na kilku najistotniejszych przyczynach:

• Otoczenie ekonomiczne

• Otoczenie technologiczne

• Otoczenie konkurencyjne

### Otoczenie ekonomiczne

Przy inflacji rosnącej w zastraszającym tempie projekt witryny zawierającej informacje o kryptowalutach i technologii blockchain ma możliwość przykucia uwagi szerokiego grona czytelników. Obecna sytuacja ekonomiczna powoduje, że wielu szuka możliwości alokacji dostępnych środków krótko lub długoterminowo w celu uniknięcia inflacji lub w celu wzbogacenia się dlatego też treści udostępniane przez opisywaną aplikację mają ogromną wartość i mogą stanowić istotne źródło informacji dla wielu osób zarówno tych, którzy dopiero rozpoczynają swoją przygodę z kryptowalutami jak i tych, którzy są już z tym światem świetnie obeznani.

### Otoczenie technologiczne

Technologia rozwija się niezwykle szybko. Do napisania witryny i aplikacji autor użyje między innymi HTML, CSS i JavaScript, które oferują developerowi pełną władzę nad kodem i jego działaniem.

W celu zapewnienia użytkownikowi dostępności autor zdecydował się wypuścić aplikację na serwer heroku. Takie rozwiązanie da dostęp do witryny każdemu chętnemu czytelnikowi, potrzebny będzie tylko dostęp do internetu i przeglądarka internetowa. Daje to też możliwość pisania i dodawania nowych treści administratorowi aplikacji z wielu dostępnych urządzeń, także mobilnych, niezależnie od miejsca pobytu.

By treści na stronie były zawsze dostępne autor zdecydował skorzystać z bazy danych w MongoDB Atlas, która znajduje się w chmurze i zapewni proste połączenie między aplikacją, a bazą danych.

W przyszłości witryna może się rozwinąć o sekcję gdzie czytelnicy będą zadawać pytania, a administrator będzie mógł na nie odpowiadać. Możliwe jest też dodanie funkcjonalności by czytelnik zapisał się na newsletter dzięki czemu za każdym razem gdy administrator coś udostępni, czytelnik dostanie o tym wiadomość, która będzie wysyłana automatycznie.

### Otoczenie konkurencyjne

W momencie rozpoczęcia prac nad aplikacją, w Polsce temat technologii blockchain i świadomość na temat rynku kryptowalut nie była tak rozwinięta i jak np. w stanach zjednoczonych. To tutaj właśnie przedstawiony projekt ma największe szanse na odniesienie sukcesu.

Aplikacja umożliwi administratorowi dzielenie się zdobytą wiedzą i doświadczeniem z nieograniczoną liczbą osób, a możliwość rozwoju aplikacji, jej skalowania oraz w przyszłości opcja dołączenia aplikacji do większej infrastruktury powoduje iż niniejszy projekt ma duże szanse odniesienia sukcesu.

# Specyfikacja wymagań

Niniejszy rozdział przedstawia wymagania stawiane aplikacji służącej do udostępniania spersonalizowanych treści szerszemu gronu odbiorców. Autor omawia kolejno słownik pojęć oraz wymagania funkcjonalne i niefunkcjonalne, zarówno użytkownika jak i systemowe.

## Dziedzinowy słownik pojęć

**Tabela 1. Dziedzinowy słownik pojęć**

|  |  |
| --- | --- |
| Rekord | Pojedyncza treść udostępniana na stronę główną przez użytkownika aplikacji od strony panelu administratora. |
| Kategoria | Określa rodzaj rekordu, jest powiązana z użytkownikiem aplikacji. |
| Kryptowaluta | Rodzaj rekordu powiązanego z użytkownikiem aplikacji, zawiera informacje konkretne informacje o danej kryptowalucie. |
| Artykuł | Rodzaj rekordu powiązanego z użytkownikiem aplikacji, w której zawarte są informacje o tym co się dzieje w świecie kryptowalut. |

## Wymagania niefunkcjonalne użytkownika

**WNU1.** Z założeń wynika, że aplikacja powinna się zachowywać prawidłowo, w sytuacjach gdy użytkownik korzysta z jednej z wymienionych poniższych przeglądarek:

• Mozilla Firefox

• Google Chrome

• Safari

• Opera

**WNU2.** Aplikacja powinna działać poprawnie, gdy użytkownik ma dostęp do internetu o szybkości minimum 300 mb/s.

## Wymagania niefunkcjonalne systemowe

**WNS1.** Aplikacja powinna korzystać z rozwiązań systemów informatycznych, które są darmowe.

**WNS2.** Aplikacja powinna być wieloplatformowa tj, niezwiązana z wyłącznie jednym system operacyjnym i powinna się zachowywać podobnie na każdym systemie oraz urządzeniu typu komputer, laptop oraz urządzenie mobilne.

**WNS3.** Aplikacja powinna poprawnie się uruchamiać na każdym sprzęcie posiadającym dostęp do wymienionych w WNU1 przeglądarek.

## Model wymagań

W systemie powinno być zdefiniowanych trzech aktorów – administrator reprezentujący osobę zarządzającą całym systemem, – użytkownik, czyli podstawowy eksploatator systemu oraz – czytelnik, reprezentujący osobę, która będzie przeglądać treści na witrynie bez dostępu do panelu administratora.

**UCR1.** Użytkownik, będący najbardziej podstawowym aktorem całego systemu posiada szereg funkcjonalności z których może korzystać. Wszystkie funkcjonalności, z których może skorzystać opisuje poniższy rysunek.

**Rysunek 2. Diagram przypadków użycia dla użytkownika**

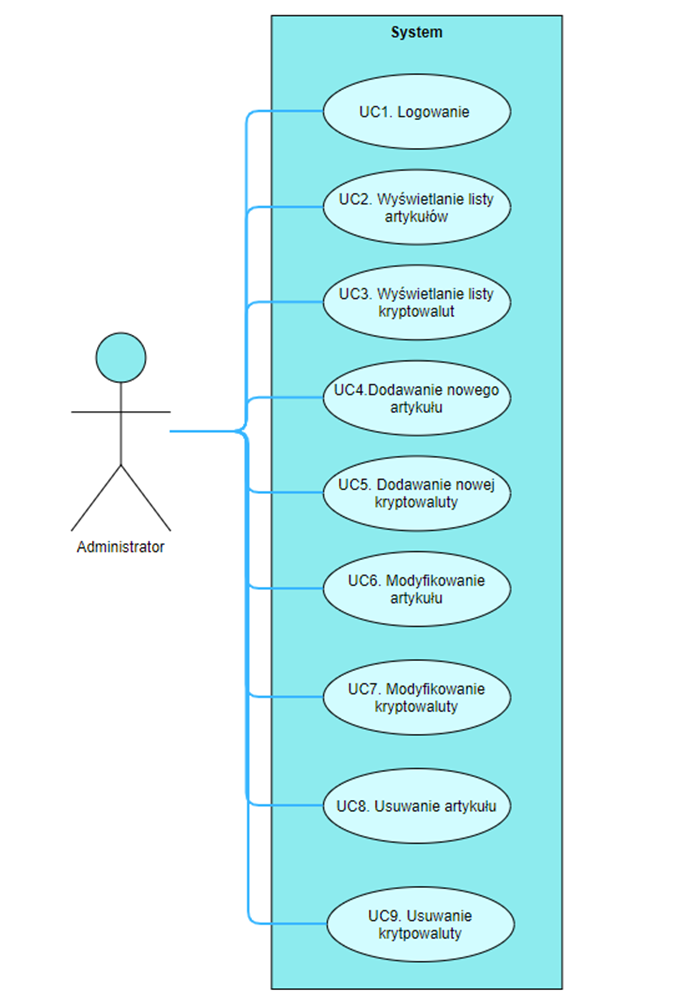
Diagram

Description automatically generated

Źródło: opracowanie własne, Visual Paradigm Community Edition

**UCR2.** Administrator, będący rozszerzeniem użytkownika jest aktorem posiadającym największą władzę w systemie. Ma możliwość korzystania ze wszystkich funkcjonalności z jakich może korzystać użytkownik, a także ma szereg funkcji, które są dostępne tylko z poziomu administratora. Zastosowania dostępne dla administratora opisuje poniższy diagram.

**Rysunek 3. Diagram przypadków użycia dla administratora**



Źródło: opracowanie własne, Visual Paradigm Community Edition

**UCR3.** Czytelnik, jest aktorem systemu, który posiada szereg funkcjonalności związanych z główną częścią witryny. Przypadki użycia dla czytelnika dokładnie opisuje poniższy rysunek.

**Rysunek 4. Diagram przypadków użycia dla czytelnika**

Diagram

Description automatically generated

Źródło: opracowanie własne, Visual Paradigm Community Edition

## Opis scenariuszowy

Ten rozdział autor poświecił opisowi scenariuszowemu poszczególnych przypadków użycia, które zostały przedstawione na diagramach UML w poprzednim podrozdziale. Diagramy zostały wykonane zgodnie z metodyką zaproponowaną przez Alistair’a Cockburn’a.

**UC1.** Logowanie do systemu

|  |  |
| --- | --- |
| Aktor | Użytkownik/Administrator |
| Warunek początkowy | * Użytkownik/Administrator nie jest zalogowany |
| Wyzwalacz | * Użytkownik/Administrator przechodzi na stronę panelu administratora lub jest do niej przekierowany |
| Głowna ścieżka powodzenia | 1. Użytkownik/Administrator wprowadza swoją unikalną nazwę użytkownika oraz hasło. 2. Użytkownik/Administrator zatwierdza dane wprowadzone na formularzu przyciskiem „zaloguj”. 3. System weryfikuje czy istnieje użytkownik o podanej w formularzu nazwie. 4. System weryfikuje czy podane w formularzu hasło pasuje do danej nazwy użytkownika. 5. W przypadku gdy dane z formularza przechodzą weryfikację pozytywnie użytkownik jest przekierowywany do strony głównej panelu administratora. W sytuacji gdy podane w formularzu dane nie są prawidłowe system zwraca błąd a strona do logowania się odświeża. |
| Efekt końcowy | Użytkownik uzyskuje dostęp do panelu administratora |

**UC2.** Wyświetlenie listy artykułów

|  |  |
| --- | --- |
| Aktor | Użytkownik |
| Warunek początkowy | * Użytkownik jest zalogowany |
| Wyzwalacz | * Użytkownik przechodzi do widoku listy artykułów |
| Głowna ścieżka powodzenia | 1. System zwraca listę udostępnionych artykułów od najnowszych do najstarszych. |
| Efekt końcowy | Użytkownik uzyskuje informacje dotyczące udostępnionych na witrynie artykułów. |

**UC3.** Wyświetlenie listy kryptowalut

|  |  |
| --- | --- |
| Aktor | Użytkownik |
| Warunek początkowy | * Użytkownik jest zalogowany |
| Wyzwalacz | * Użytkownik przechodzi do widoku listy kryptowalut |
| Głowna ścieżka powodzenia | 1. System zwraca listę udostępnionych kryptowalut od najnowszych do najstarszych. |
| Efekt końcowy | Użytkownik uzyskuje informacje dotyczące udostępnionych na witrynie kryptowalut. |

**UC4.** Dodawanie nowego artykułu

|  |  |
| --- | --- |
| Aktor | Użytkownik |
| Warunek początkowy | * Użytkownik jest zalogowany i jest na widoku listy artykułów |
| Wyzwalacz | * Użytkownik przechodzi do formularza dodawania nowego artykułu klikając w przycisk „Napisz artykuł” |
| Głowna ścieżka powodzenia | 1. System przekierowuje użytkownika do formularza. 2. Użytkownik wypełnia w formularzu wymagane dane i kilka przycisk „opublikuj”. 3. System udostępnia nowy artykuł na stronie. |
| Efekt końcowy | Użytkownik z sukcesem udostępnił nowy artykuł na witrynie. |

**UC5.** Dodawanie nowej kryptowaluty

|  |  |
| --- | --- |
| Aktor | Użytkownik |
| Warunek początkowy | * Użytkownik jest zalogowany i jest na widoku listy kryptowalut. |
| Wyzwalacz | * Użytkownik przechodzi do formularza dodawania nowej kryptowaluty klikając w przycisk „Dodaj kryptowalutę”. |
| Głowna ścieżka powodzenia | 1. System przekierowuje użytkownika do formularza. 2. Użytkownik wypełnia w formularzu wymagane dane i kilka przycisk „opublikuj”. 3. System udostępnia nową kryptowalutę na stronie. |
| Efekt końcowy | Użytkownik z sukcesem udostępnił nową kryptowalutę na witrynie. |

**UC6.** Modyfikowanie artykułu

|  |  |
| --- | --- |
| Aktor | Użytkownik |
| Warunek początkowy | * Użytkownik jest zalogowany i jest na widoku listy artykułów. |
| Wyzwalacz | * Użytkownik kilka niebieski przycisk „Zaktualizuj”. |
| Głowna ścieżka powodzenia | 1. System przekierowuje użytkownika do formularza. 2. Użytkownik aktualizuje w formularzu dane i kilka przycisk „opublikuj aktualizację”. 3. System sprawdza czy dane podane w formularzu są podane zgodnie z wymaganymi typami danych. 4. Jeżeli weryfikacja zakończy się sukcesem to System komunikuje się z bazą danych, aktualizuje wybrany artykuł i udostępnia go na stronie. Jeżeli weryfikacja zakończy się porażką system zwraca błąd i prosi o ponowne wypełnienie formularza. |
| Efekt końcowy | Użytkownik zaktualizował udostępniony na stronie artykuł. |

**UC7.** Modyfikowanie kryptowaluty

|  |  |
| --- | --- |
| Aktor | Użytkownik |
| Warunek początkowy | * Użytkownik jest zalogowany i jest na widoku listy kryptowalut. |
| Wyzwalacz | * Użytkownik kilka niebieski przycisk „Zaktualizuj”. |
| Głowna ścieżka powodzenia | 1. System przekierowuje użytkownika do formularza. 2. Użytkownik aktualizuje w formularzu dane i kilka przycisk „opublikuj aktualizację”. 3. System sprawdza czy dane podane w formularzu są podane zgodnie z wymaganymi typami danych. 4. Jeżeli weryfikacja zakończy się sukcesem to system komunikuje się z bazą danych, aktualizuje wybraną kryptowalutę i udostępnia ją na stronie. Jeżeli weryfikacja zakończy się porażką system zwraca błąd i prosi o ponowne wypełnienie formularza. |
| Efekt końcowy | Użytkownik zaktualizował wybraną kryptowalutę. |

**UC8.** Usuwanie artykułu

|  |  |
| --- | --- |
| Aktor | Administrator |
| Warunek początkowy | * Administrator jest zalogowany i jest na widoku listy artykułów. |
| Wyzwalacz | * Administrator kilka czerwony przycisk „usuń” przy konkretnym artykule. |
| Głowna ścieżka powodzenia | 1. System komunikuję się z bazą i usuwa wybrany artykuł. |
| Efekt końcowy | Administrator z sukcesem usunął wybrany artykuł z witryny. |

**UC9.** Usuwanie kryptowaluty

|  |  |
| --- | --- |
| Aktor | Administrator |
| Warunek początkowy | * Administrator jest zalogowany i jest na widoku listy kryptowalut. |
| Wyzwalacz | * Administrator kilka czerwony przycisk „usuń” przy konkretnej kryptowalucie. |
| Głowna ścieżka powodzenia | 1. System komunikuję się z bazą i usuwa wybraną kryptowalutę. |
| Efekt końcowy | Administrator z sukcesem usunął wybraną kryptowalutę z witryny. |

**UC10.** Wyświetlenie strony głównej

|  |  |
| --- | --- |
| Aktor | Czytelnik |
| Warunek początkowy | * Czytelnik nie znajduje się na stronie ani żadnej podstronie witryny. |
| Wyzwalacz | * Czytelnik przechodzi na stronę główną witryny lub zostaje na nią przekierowany. |
| Głowna ścieżka powodzenia | 1. System wyświetla stronę główną witryny. |
| Efekt końcowy | Czytelnik znajduje się na stronie głównej witryny. |

**UC.11** Wyświetlenie podstrony z kryptowalutami

|  |  |
| --- | --- |
| Aktor | Czytelnik |
| Warunek początkowy | * Czytelnik jest na jakiejkolwiek zakładce na witrynie. |
| Wyzwalacz | * Czytelnik przechodzi na zakładkę „kryptowaluty” lub zostaje na nią przekierowany. |
| Głowna ścieżka powodzenia | 1. System komunikuje się z serwerem, serwer komunikuje się z bazą danych i wyświetla podstronę „kryptowaluty” z listą wszystkich udostępnionych na stronie kryptowalut. |
| Efekt końcowy | Czytelnik otrzymał listę udostępnionych na stronie kryptowalut. |

**UC12.** Wyświetlenie podstrony z artykułami

|  |  |
| --- | --- |
| Aktor | Czytelnik |
| Warunek początkowy | * Czytelnik jest na jakiejkolwiek zakładce na witrynie. |
| Wyzwalacz | * Czytelnik przechodzi na zakładkę „artykuły” lub zostaje na nią przekierowany. |
| Głowna ścieżka powodzenia | 1. System komunikuje się z serwerem, serwer komunikuje się z bazą danych i wyświetla podstronę „artykuły” z listą wszystkich udostępnionych na stronie artykułów. |
| Efekt końcowy | Czytelnik otrzymał listę udostępnionych na stronie artykułów. |

# Propozycja bazy danych dla systemu

Autor systemu bazując na swoim dotychczasowym doświadczeniu, specyfice aplikacji oraz strukturze danych zdecydował się wykorzystać nierelacyjny system zarządzania bazą danych. Tego rodzaju rozwiązanie zapewnia kilka kluczowych funkcjonalności jak np. sharding danych, który umożliwia dzielenie baz danych na fragmenty, które są szybsze i łatwiejsze w zarządzaniu. Podzielone fragmenty mają mniejsze wymagania dotyczące sprzętu oraz oprogramowania co pozwala na oszczędności ponieważ mogą być one zarządzane na tańszych serwerach. Tego typu rozwiązanie zapewnia również wysoką skalowalność bazy danych oraz umożliwia szybką obsługę zapytań.

Z wielu możliwych rozwiązań NoSQL autor zdecydował się wybrać bazę danych MongoDB.

## Dokumenty oraz ich struktura

Do prawidłowego działania aplikacji wymagane jest stworzenie odpowiedniego typu dokumentów przechowywanych w bazie. Ten rozdział został poświęcony opisaniu poszczególnych dokumentów i ich struktury.

**D1. Kryptowaluta (overwiew)**

• **id** – automatycznie generowany identyfikator, unikalny dla każdego rekordu,

• **name** – nazwa kryptowaluty,

• **description** – opis kryptowaluty,

• **logo** – zdjęcie, logo danej kryptowaluty.

**D2. Artykuł (article)**

• **id** – automatycznie generowany identyfikator, unikalny dla każdego rekordu,

• **title** – tytuł artykułu,

• **date** – data w której artykuł został udostępniony,

• **photo** – zdjęcie do nagłówka,

• **content** – treść artykułu ,

• **author** – imię oraz nazwisko autora, który udostępnił artykuł.

Wyżej wymienione dokumenty będą tworzone za pomocą schematów. Na poniższym rysunku został przedstawiony kod odpowiadający za stworzenie proponowanego schematu.

**Rysunek 5. Baza danych – schemat dokumentu**

|  |
| --- |
| **const overviewSchema = {**  **name: String,**  **description: String,**  **logo: String**  **};**  **const Overview = mongoose.model("Overview", overviewSchema);**  **const articleSchema = {**  **name: String,**  **date: String,**  **photo: String,**  **content: String,**  **author: String**  **};**  **const Article = mongoose.model("Article", articleSchema);** |

Źródło: opracowanie własne

## Połączenie z bazą danych

System w założeniu będzie działać w oparciu o pakiet mongoose, który pozwala na połączenie backendu aplikacji z nierelacyjną bazą danych MongoDB. Pakiet oferuje zarządzanie dokumentami znajdującymi się w bazie danych oraz zawiera rozwiązania pozwalające na wykorzystanie nierelacyjnych baz danych w dużych projektach biznesowych co zapewnia skalowalność projektowanych aplikacji.

# Projekt interfejsu użytkownika

Użytkownicy oraz administratorzy w interakcję z systemem będą wchodzili za pośrednictwem jego części klienckiej – panelu administratora, który będzie dostępny w postaci strony internetowej. Do zaprojektowania tej części systemu autor wykorzysta framework JavaScriptu – Express.js, dostarczający gotowe funkcjonalności i struktury.

## Trzon interfejsu

Interfejs proponowany przez autora będzie się dzielił na trzy główne elementy widoczne na poniższym rysunku 5.

**Rysunek 6. Trzon interfejsu użytkownika** Graphical user interface

Description automatically generated

Źródło: opracowanie własne, Canva

**Pierwszy element przedstawiony na rysunku 5 oznaczony numerem 1 oraz kolorem pomarańczowym będzie paskiem nawigacyjnym na którym znajdować się będzie logo, przyciski pozwalające przejść w każdą z zakładek oraz przycisk pozwalający użytkownikowi wylogować się z panelu.**

**Drugi z wymienionych na rysunku 5 elementów (oznaczony kolorem zielonym oraz numerem 2) będzie miejscem, w którym wyświetlać się będzie zawartość właściwa. W tym możliwość przejścia listy udostępnionych na stronie artykułów oraz kryptowalut.**

**Trzeci element interfejsu (oznaczony kolorem niebieskim oraz numerem 3) będzie miejscem wyświetlania informacji dotyczących udostępnionych rekordów z kategorii kryptowalut.**

**Czwarty główny element interfejsu (oznaczony na rysunku kolorem fioletowym oraz numerem 4) oznacza miejscem, w którym wyświetlać się będą informacje dotyczące udostępnionych rekordów z kategorii artykuły.**

**Element oznaczony numerem 5 oraz kolorem żółtym oznacza stopkę, na której wyświetlana będzie informacja o prawach autorskich oraz linki do stron zewnętrznych.**

## ****Lista kryptowalut oraz artykułów****

Proponowany przez autora panel administratora będzie miał możliwość wyświetlania wielu zróżnicowanych danych, w tym ile zostało udostępnionych kryptowalut i artykułów na stronie.

### Strona główna panelu administratora

Jest pierwszą stroną widoczną po zalogowaniu się i stanowi główny panel, z którego będzie korzystał użytkownik.

**Rysunek 7. Interfejs użytkownika – pulpit nawigacyjny**

Graphical user interface, website

Description automatically generated

Źródło: opracowanie własne, Canva

Przyciski oznaczone kolorem fioletowym z napisem ZOBACZ odpowiadać będą za możliwość przejścia do listy rekordów udostępnionych w wybranej kategorii.

### Lista rekordów

Jest stroną wyświetlającą wszystkie udostępnione przez użytkownika lub administratora rekordy na stronie. Użytkownik na początku ma możliwość jaki rodzaj rekordów (artykuły czy kryptowaluty) go interesują, dla obu kategorii.

**Rysunek 8. Interfejs użytkownika – lista rekordów**

Chart

Description automatically generated

Źródło: opracowanie własne, Canva

Widok obejmuje wyświetlanie listy kryptowalut (lista jest sortowana po dacie, od najstarszego rekordu do najnowszego) oraz przyciski umożliwiające usunięcie, zmodyfikowanie oraz dodanie nowej kryptowaluty na stronę. Proponowane przez autora rozwiązanie widoczne jest na rysunku 8.

**Rysunek 9. Interfejs użytkownika – lista rekordów (opis szczegółowy)**

Chart, bar chart

Description automatically generated

Źródło: opracowanie własne, Canva

**Poniżej autor opisał poszczególne funkcjonalności jakie daje możliwość proponowany na rysunku 8, interfejs panelu.**

**F1. Przycisk oznaczony kolorem fioletowym oraz napisem AKCJA 1 przewiduje się, że będzie pozwalał przejść użytkownikowi na stronę, z której możliwe jest dodanie nowego rekordu.**

**F2. Przycisk oznaczony kolorem czerwonym oraz opisany jako AKCJA 2 będzie umożliwiać usuwanie wybranego rekordu z bazy danych.**

**F3. Przycisk oznaczony kolorem niebieskim oraz napisem AKCJA 3 w zamierzeniach będzie spełniał funkcję pozwalającą na przejście do strony gdzie będzie możliwa edycja udostępnionego już rekordu.**

### ****Dodawanie nowego rekordu****

Będzie to strona, z której możliwe będzie dodanie nowego rekordu do bazy danych. Na rysunku 9 autor przedstawił planowany wygląd tej strony. Z założeń, strona będzie wyglądała identycznie dla wszystkich kategorii dodawanych rekordów, różnić się będzie tylko ilością wymaganych do podania danych.

**Rysunek 10. Interfejs użytkownika – dodawanie nowego rekordu**

Graphical user interface, website, timeline

Description automatically generated

Źródło: opracowanie własne, Canva

**Autor oznaczył kolorem błękitnym (oraz napisem WYMAGANE DANE) pola, które będą niezbędne do wypełnienia by dodać rekord do bazy danych.**

**Przycisk oznaczony kolorem fioletowym oraz napisem AKCJA z zamierzenia będzie odpowiadał za dodanie nowego rekordu do bazy danych.**

### ****Aktualizowanie rekordu****

System przewiduje możliwość aktualizacji rekordu jeśli zajdzie taka potrzeba. Autor zaproponował rozwiązanie podobne jak przedstawione na rysunku 9. Interfejs użytkownika wygląda tak samo, jedyną różnicą jest to, że system zamiast dodawać nowy rekord do bazy danych, będzie aktualizował udostępniony już rekord znajdujący się w podłączonej bazie danych.

# Implementacja

Rozdział piąty poświęcony został opisowi procesu implementacji projektu. Autor opisał jakie były założenia dotyczące wykorzystanych technologii, oprogramowania oraz narzędzi.

## Założenia

W procesie implementacji niniejszego projektu autor rozpoznał szereg założeń odnoszących się do zasobów, kosztów realizacji oraz wykorzystanych technologii niezbędnych do rozpatrzenia w projekcie. Przedstawiony podrozdział omawia wspomniane założenia:

* Autor aplikacji lub firma odpowiadająca za zlecenie jej stworzenia posiada prawa oraz możliwości do korzystania z wymienionych w rozdziale 7 narzędzi oraz środowisk.
* W celu uruchomienia aplikacji wymagany jest do poniesienia koszt związany ze sprzętem na którym będzie działać baza danych oraz produkcyjna wersja aplikacji, może to być serwer fizyczny lub wirtualna instancja wykupiona u firmy zewnętrznej.
* Proponowany system będzie otwarty na nowe funkcjonalności, które można będzie wdrażać zgodnie z występującym zapotrzebowaniem.
* Za przeprowadzenie implementacji oraz nadzorowanie jej przebiegu odpowiedzialna będzie jedna i ta sama osoba.
* Za zapewnienie poprawnego działania, utrzymanie i konserwację systemu odpowiedzialna będzie jedna osoba, ta sama, która była odpowiedzialna za przeprowadzenie procesu implementacji.
* Podstawowa, w pełni działająca, stabilna wersja przedstawionego systemu zostanie uruchomiona nie wcześniej niż przed grudniem 2021 roku.

# Testy aplikacji

W celu sprawdzenia prawidłowego działania systemu każda funkcjonalność została manualnie przetestowana.

## Testy API

Testy backendowej części aplikacji oraz API zostało wykonane za pomocą oprogramowania Postman, które pozwoliło na przetestowanie serwerowej części aplikacji, połączonej z bazą danych, bez wymogu pisania frontowej części aplikacji, która wysyła zapytania do serwera. Zapytania były wysyłane za pośrednictwem protokołów HTTP. Autor wykorzystał platformę do tworzenia testowych żądań HTTP dla punktów dostępowych występujących w serwerowej części kodu aplikacji.

Testy odbyły się na sprzęcie wykorzystującym następujące narzędzia:

**Tabela 2. Testy – specyfikacja sprzętu**

|  |  |
| --- | --- |
| System operacyjny | Windows 10 Professional |
| Postman | V 9.0.9 |
| Node | V 14.17.6 |
| Mongo | V 5.0.2 |

## Testy działania systemu

Pierwsze testy aplikacji pełnej aplikacji (części frontowej i backendowej) odbywały się jeszcze zanim autor opublikował pełną wersję witryny. Po wprowadzeniu każdej funkcjonalności, autor sprawdzał czy te działają na serwerze lokalnym uruchomionym w Node.js. Owa testowa wersja aplikacji była podłączona do lokalnej bazy danych, a testy wykonywane były początkowo na laptopie, po dopisaniu css umożliwiającego wygodne korzystanie z aplikacji na smartfonie autor zaczął prowadzić testy również na urządzeniach mobilnych.

Autor w ramach testów udostępnił link do aplikacji grupie piętnastu użytkowników i zniósł ograniczenia związane z adresami IP dzięki czemu do panelu można było się połączyć z każdej dostępnej sieci oraz urządzenia. Testy były wykonywane na różnych urządzeniach zarówno mobilnych jak i stacjonarnych.

# Środowisko oraz narzędzia

Do napisania aplikacji autor wykorzystał nowoczesne technologie umożliwiające szybkie pisanie aplikacji webowych jednocześnie zapewniających bezpieczeństwo i higienę pracy. Decyzja o użyciu niżej wymienionych technologii w implementacji aplikacji padła w głównej mierze ze względu na zainteresowania autora.

Poniżej autor wymienił technologie, które pomogły mu w stworzeniu przedstawionej w niniejszej pracy aplikacji.

## Wykorzystane technologie

W tym podrozdziale autor szczegółowo opisuje technologie jakie zostały wykorzystane podczas realizacji projektu.

### HTML

HTML (ang. Hyper Text Markup Language) umożliwia użytkownikom sieci na tworzenie i strukturyzowanie sekcji, akapitów i łączących ich elementów, znaczników i atrybutów. Jest językiem, z którego się składa większość stron internetowych.

Autor wykorzystał HTML do napisania głównych stron aplikacji odpowiedzialnych za wyświetlanie treści na stronie.

### CSS

CSS inaczej kaskadowe arkusze stylów (ang. Cascading Style Sheets) to język arkuszy stylów używany do opisywania w jaki sposób dana część kodu napisana w HTML ma wyglądać. Pozwala wysoce personalizować każdy element widoczny na stronie.

### MongoDB

MongoDB jest systemem służącym do zarządzania bazą danych. System ten sprawdza się świetnie w wykorzystaniu aplikacji webowych. Z pomocą pakietu mongoose autor mógł swobodnie korzystać z zapytań pełnego języka jaki oferuje platforma. MongoDB oferuje też utworzenie bazy danych w chmurze, którą łatwo da się połączyć z back-endem napisanego projektu.

Autor wykorzystał wymienioną platformę do przechowywania danych udostępnionych na stronie, jak i do stworzenia prostego API odpowiadającego za czynności takie jak: dodawania, modyfikowanie i usuwanie treści udostępnionych na stronie.

### JavaScript

Javascript jest dynamicznym językiem programowania, który jest wykorzystywany w web developmencie oraz tworzeniu aplikacji webowych. Pozwala na zaimplementowanie wielu funkcji na stronach internetowych, których nie dałoby się zaimplementować wykorzystując tradycyjny HTML oraz CSS. Najnowsza odsłona JavaScriptu (ES6) wprowadziła szereg nowych funkcjonalności, które ułatwiły autorowi pisanie kodu aplikacji. Jedną z największych, z których autor korzystał była możliwość pisania funkcji strzałkowych (Arrow Functions), które umożliwiły pisanie kodu w prostszy, krótszy i bardziej czytelny sposób niż dotychczas.

### Node.js

Node.js jest wieloplatformowym środowiskiem uruchomieniowym służącym do pisania aplikacji w języku JavaScript. Wykorzystywany jest przede wszystkim w pisaniu aplikacji typu server-side czyli takich, które występują po stronie serwera. Ponieważ jest to środowisko języka JavaScript pozwala też na pisanie kompletnych aplikacji, czyli takich, które zawierają jednocześnie elementy klienckie jak i serwerowe, a wszystko to napisane w jednym języku programowania. Node jest bardzo szybki, a gdy połączy się go z rozwiązaniami takimi, jak klastry czy wątki robocze zapewnia bardzo dużą możliwość skalowania. To środowisko uruchomieniowe posiada wiele cech, które czynią go wszechstronnym środowiskiem wykonawczym, jednak sercem Node.js jest libuv. Libuv to usługa, która została napisana w języku C, stanowi ona interfejs pośredniczący pomiędzy Node.js, a jądrem systemu operacyjnego i odpowiada za udostępnianie asynchronicznych mechanizmów obsługi wejścia/wyjścia. Błyskawiczny dostęp do swoich usług libuv umożliwia poprzez rozwiązanie zwane pętlą zdarzeń (eng. event loop).

Środowisko uruchomieniowe jest przenośne i dostępne dla Microsoft Windows, macOs, Linux, Solaris, OpenBSD, NonStop OS, FreeBSD i WebOS. Jest wspierane przez dostawców usług sieciowych, którzy zapewniają odpowiednią infrastrukturę i dokumentację do udostępniania i hostowania aplikacji typu Node.js.

Wszystkie wymienione wyżej cechy sprawiły, że to właśnie Node.js został wybrany przez autora do napisania omawianej aplikacji.

### Express

Express jest wieloplatformowe oprogramowanie o otwartym kodzie służące do pisania kodu działającego w środowisku uruchomieniowym node.js. Oprogramowanie służy do pisania kodu JavaScript występującego po stronie serwera. Jest to środowisko uruchomieniowe, które działa poza przeglądarką, współpracuje ono bezpośrednio z systemem operacyjnym, dzięki temu środowisko Node.js udostępnia swoim aplikacjom API, w tym dostęp do plików i bibliotek oraz serwerów HTTP.

Ten framework zapewnia wysoką wydajność i optymalizuje skalowalność aplikacji webowych.

Autor użył tego oprogramowania do napisania back-endu oraz API czyli części aplikacji działającej po stronie serwera i odpowiadającej na zapytania płynące z frontowej części napisanej aplikacji webowej.

### API

API jest skrótem od Application Programming Interface, czyli jest to Interfejs Programowania Aplikacji. API są niezbędne do łączenia aplikacji, dzięki takim interfejsom różne aplikacje mogą współdziałać ze sobą i pobierać od siebie nawzajem potrzebne informacje. API to część kodu, która jest odpowiedzialna za kontrolowanie wszystkich punktów dostępowych aplikacji, przeważnie występujących po stronie serwera. To dzięki API do serwera lub kilku serwerów można wysłać zewnętrzne żądanie, które to API obsługuje jako pośrednik i tłumacz. To Interfejs Programowania Aplikacji jest odpowiedzialny za przyjęcie zapytania, przekazanie go do aplikacji stojącej po stronie serwera oraz odesłanie informacji zwrotnej. Takie rozwiązanie pozwala na automatyzację wielu funkcji występujących w aplikacji.

Autor wykorzystał interfejs API do stworzenia prostych ścieżek tzw. endpointów, dopowiadających między innymi za dodawanie, modyfikowanie i usuwanie rekordów widocznych na stronie. Dostęp do tego API ma administrator i użytkownik, po wkroczeniu na panel administratora.

## Narzędzia

Ten podrozdział został poświęcony narzędziom z jakie były autorowi niezbędne do napisania przedstawionej w niniejszej pracy aplikacji.

### Hyper

Hyper jest to wieloplatformowy terminal, który jest wysoce konfigurowalny. Zbudowany jest na technologiach opartych o JavaScript, HTML i CSS. Pozwala on użytkownikowi wiersza poleceń na skonfigurowanie wyglądu oraz działania terminala.

Terminal Hyper jest kompatybilny z systemami operacyjnymi Microsoft Windows, mac OS oraz Linux. Dzięki stworzeniu prostego pliku o nazwie .hyper, w którym znajdują się wszystkie konfiguracje wiersza poleceń autor mógł bez problemu pracować na różnych urządzeniach posiadających wcześniej wymienione systemy operacyjne, bez przymusu przesiadania się i uczenia, jak działają poszczególne terminale na każdym z systemów operacyjnych.

### Git

Git jest rozproszonym systemem odpowiadającym za kontrolę wersji (version control). Oferuje on dobre wsparcie dla rozgałęzionego procesu tworzenia oprogramowania. Umożliwia pracę off-line, czyli bez połączenia z siecią. Git pozwala na tworzenie repozytorium, w którym znajdują się wszystkie pliki źródłowe jakie autor oprogramowania zechce w nim zamieścić. Umożliwia to proste przekazywanie plików projektu pomiędzy różnymi urządzeniami jak i programistami. Każdy programista ma własną kopię repozytorium do którego może zapisywać zmiany bez połączenia z siecią.

Autor wykorzystał system by pilnować poszczególnych etapów tworzenia aplikacji tak by w razie wystąpienia błędów krytycznych dało się wrócić do poprzedniej wersji programu. Git został też wykorzystany przez autora do udostępnienia repozytorium z plikami projektu tak, by mógł pracować nad aplikacją z wielu urządzeń.

### Node Package Menager

NPM jest to narzędzie wykorzystywane w wierszu poleceń, które pozwala na instalowanie, aktualizację i odinstalowywanie pakietów Node.js w aplikacji. Jest również internetowym repozytorium pakietów Node.js o otwartym kodzie źródłowym. Społeczność programistów na całym świecie ma możliwość tworzenia własnych, przydatnych modułów i publikowania ich jako pakiety w repozytorium, dzięki czemu każdy programista może skorzystać z wielu gotowych już rozwiązań problemów.

NPM wykonuje operacje w dwóch głównych trybach: lokalnym i globalnym. W trypie lokalnym wykonywane są operacje dla konkretnego katalogu lokalnego, operacje te mają wpływ tylko na aplikacje znajdujące się w danym katalogu. W trybie lokalnym NPM wykonuje operacje, które wpływają na wszystkie aplikacje napisane w języku Node.js, które znajdują się na danym środowisku sprzętowym.

Ze względu na prostotę narzędzia NPM, autor wykorzystał je do instalacja wszystkich modułów Node.js, potrzebnych do prawidłowego działania aplikacji.

### Heroku

Heroku jest platformą chmurową typu PaaS, obsługuje ona kilka języków oprogramowania. Platforma pozwala na proste zbudowanie i udostępnienie aplikacji napisanej w języku Node.js na serwer zewnętrzny. Autor przy użyciu kilku podstawowych komend Gita użył platformy by udostępnić napisaną aplikację na serwer zewnętrzny tak, by każdy o udzielonym dostępie do aplikacji i strony mógł zacząć z niej korzystać.

### Atom

Atom jest darmowym edytorem kodów programistycznych o otwartym kodzie źródłowym. Działa on na systemach Microsoft Windows, Linux oraz macOs. Oferuje on udogodnienia takie jak np.: kolorowanie składni języków, upiększanie (układanie) kodu tak by był milej widoczny dla oka, większość pakietów rozszerzeń ma licencję wolnego oprogramowania i jest budowana oraz utrzymywana przez społeczność.

Posiada on wsparcie dla wszystkich języków (i ich frameworków) jakich autor użył do napisania kodu dlatego też edytor ten był pierwszym wyborem autora.

### Microsoft Visual Studio Code

Visual Studio Code – jest to środowisko programistyczne napisane w języku programowania TypeScript stworzone i rozwijane przez firmę Microsoft. Jest to rozwiązanie rozpowszechniane na licencji MIT o otwartym kodzie źródłowym. Nie określa żadnego konkretnego języka programowania, jest to rozwiązanie ogólnego przeznaczenia. Szeroka dostępność dodatków sprawia, że praktycznie każdy język programowania w tym środowisku będzie obsługiwał funkcje odpowiedzialne za podpowiadanie składni oraz jej kolorowanie. W przedstawionym projekcie zostało wykorzystane przede wszystkim do tworzenia serwerowej części aplikacji.

### Robo3T

Jest to popularny graficzny interfejs użytkownika (ang. GUI), który ułatwia pracę z bazami danych MongoDB. Umożliwia interakcję z danymi za pomocą wskaźników wizualnych zamiast interfejsu wiersza poleceń. Narzędzie Robo3T obsługuje wiele platform, pozwala się łączyć zarówno z lokalną bazą danych jak i taką, która występuje na zewnętrznym serwerze.

Autor, wykorzystał program przede wszystkim do testów działania aplikacji i bazy danych.

### Postman

Postman – to narzędzie udostępnione w 2012 służące do testowania punktów dostępowych występujących w aplikacji. Postman jest aplikacją, która pozwala na w prosty sposób tworzyć testowe żądania http posiadające określone parametry, nagłówki oraz ciało. Program dodatkowo pozwala między innymi na zapisywanie stworzonych żądań, udostępnianie żądań innym programistom i tworzenie testów API bez potrzeby pisania klienckiej części aplikacji. Wszystkie żądania mogą być wysyłane przez Postman.

Autor wykorzystał program do testowania API występującego po stronie serwera napisanej aplikacji.

# Podsumowanie

Celem niniejszej pracy inżynierskiej było zaprojektowanie i stworzenie witryny oraz panelu administratora służącego do łatwego zarządzania witryną, na której znajdują się informacje dotyczące kryptowalut. System informatyczny miał rozwiązać problemy manualnego wykonywania operacji, przechowywać oraz udostępniać dane wprowadzane przez administratora i użytkownika oraz zapobiegać pojawianiu się błędów wynikających z czynnika ludzkiego.

## Wykonane prace

Wyżej opisany cel został w pełni zrealizowany, informacjami dostępnymi na witrynie można zarządzać z wygodnego, czytelnego i dostępnego z każdego miejsca panelu. Poprzez stworzenie jednego scentralizowanego systemu błędy powstające w wyniku konieczności interakcji z wieloma różnorodnymi systemami zostały zminimalizowane.

Na pracę składały się dwie części: pierwszą jest witryna internetowa zawierająca informacje o kryptowalutach i technologii blockchain, drugą natomiast jest panel administratora, który daje autorowi pełną władzę nad umieszczonymi na stronie informacjami. Wcześniej wymieniony panel zawiera następujące funkcjonalności: dodawanie, usuwanie oraz modyfikowanie treści widocznych na witrynie internetowej. Aplikacja cechuje się minimalistycznym interfejsem i brak w niej funkcjonalności, które są administratorowi niepotrzebne, cechy te sprawiają, że jest intuicyjna i prosta w obsłudze. Sama witryna natomiast jest odpowiedzialna za przekazywanie informacji udostępnionych przez autora szerszemu gronu czytelników.

Witryna z aplikacją była testowana przez grupę piętnastu użytkowników, którzy mieli za zadanie zalogować się do panelu administratora z różnych urządzeń i umieścić dowolnego rodzaju treści na stronie. W celu ułatwienia badań, została zniesiona blokada adresów z których można się z aplikacją połączyć. Z uzyskanych opinii wynika, że zarówno interfejs jak i funkcjonalność aplikacji odniosły zadowalające wyniki. Aplikacja okazała się bezawaryjna i skuteczna.

Zrealizowany projekt inżynierski podyktowany był rosnącą popularnością kryptowalut oraz znacznie zwiększającą się inflacją polskiej złotówki. Praca ta nie wyczerpuje w pełni tematu, czego autor ma pełną świadomość. Daje jednak zainteresowanym możliwość rozpoczęcia przygody z kryptowalutami.

## Założenia rozbudowy systemu

Dany projekt, po zrealizowaniu części bazowej, ma perspektywy rozwoju w wielu kierunkach, może służyć jako personalny blog lub może też zostać zaimplementowany do większej struktury zawierającej więcej, niż tylko informacje w zakresie tematu kryptowalut.

Poniżej autor wymienił kilka funkcjonalności, które mogłyby być elementem rozbudowy systemu w przyszłości:

* Funkcja zapisu na newsletter – witryna
* Dodanie możliwości wyszukiwania poszczególnych artykułów/kryptowalut po nazwie – witryna
* Dodanie możliwości sortowania listy rekordów po dacie ostatniej modyfikacji oraz nazwie (alfabetycznie) – panel użytkownika
* Dodanie funkcjonalności pozwalającej zmienić motyw – panel użytkownika

# Bibliografia

## Książki

1. Ben Hernick, *HTML & CSS: The Good Parts: Better Ways to Build Websites That Work (Animal Guide)*, O'Reilly Media 2010.
2. David Choi., *React, Typescript i Node,* *Tworzenie aplikacji internetowych typu fullstack*, Helion 2022.
3. Flanagan David., *JavaScript. Przewodnik. Poznaj język mistrzów programowania*, Helion 2021.
4. Greg Lim., Beginning Node.js, *Express & MongoDB Development*, Lightning Source Inc. 2019.
5. Mariot Tristoara, *Beginning Git and GitHub: A Comprehensive Guide to Version Control, Project Management, and Teamwork*, APress 2019.

## Netografia

1. Atom, strona główna projektu Atom, atom.io (data odczytu 12 stycznia 2022).
2. Bootstrap, dokumentacja projektu Bootrstrap v5.0, getbootstrap.com/docs/5.0/getting-started/introduction/ (data odczytu 18 czerwca 2022).
3. ExpressJS, strona główna projektu Express.js, expressjs.com (data odczytu 30 stycznia 2022).
4. Hyper, strona główna projektu Hyper, hyper.is (data odczytu 21 sierpnia 2021).
5. Git, strona główna projektu Git, git-scm.com/about/ (data odczytu 02 lutego 2022).
6. GitHub, strona główna projektu GitHub, github.com (data odczytu 3 grudnia 2021).
7. Microsoft, Co to jest dzielenie baz danych na fragmenty?, azure.microsoft.com/pl-pl/resources/cloud-computing-dictionary/what-is-database-sharding/ (data odczytu 17 maja 2022).
8. Microsoft, strona główna projektu Visual Studio Code, code.visualstudio.com, (data odczytu (9 września 2021).
9. MongoDB, strona główna projektu MongoDB, mongodb.com (data odczytu, 06 czerwca 2021).
10. MongoDB, strona główna projektu MongoDB Atlas, mongodb.com/atlas/database (data odczytu 06 czerwca 2021).
11. Lauren Schaefer, What is NoSQL?, mongodb.com/nosql-explained (data odczytu 18 września 2022).
12. NodeJS, strona głowna projektu Node.js, nodejs.org/en/ (data odczytu 12 lipca 2021).
13. Postman, strona główna projektu Postman, postman.com (data odczytu 22 kwietnia 2022).
14. Piotr Rzeźnik, NoSQL vs SQL, czyli kiedy i jaki typ bazy danych wybrać, jcommerce.pl/jpro/artykuly/nosql-vs-sql-bazy-danych (data odczytu 10 marca 2022).
15. Robo3T, strona główna projektu Robo3T, robomongo.org, data odczytu (19 czerwca 21).

# Spis tabel oraz rysunków

## Spis tabel

Tabela 1. Dziedzinowy słownik pojęć, s.11

Tabela 2. Testy – specyfikacja sprzętu, s.36

## Spis rysunków

Rysunek 1. Koncept systemu, s.7

Rysunek 2. Diagram przypadków użycia dla użytkownika, s.14

Rysunek 3. Diagram przypadków użycia dla administratora, s.15

Rysunek 4. Diagram przypadków użycia dla czytelnika, s.16

Rysunek 5. Baza danych – schemat dokumentu, s.26

Rysunek 6. Trzon interfejsu użytkownika, s.28

Rysunek 7. Interfejs użytkownika – pulpit nawigacyjny, s.30

Rysunek 8. Interfejs użytkownika – lista rekordów, s.31

Rysunek 9. Interfejs użytkownika – lista rekordów (opis szczegółowy), s.32

Rysunek 10. Interfejs użytkownika – dodawanie nowego rekordu, s.33