Akademia Górniczo-Hutnicza
im. Stanisława Staszica w Krakowie

Wydział Elektrotechniki, Automatyki, Informatyki i Inżynierii Biomedycznej

KATEDRA INFORMATYKI STOSOWANEJ

INŻYNIERIA OPROGRAMOWANIA

PAWEŁ BIŁKO, ANDRZEJ KOŁAKOWSKI
STANISŁAW ŚWIATŁOCH, MATEUSZ WIĘCŁAWEK

BANK BITCOINÓW

Kraków 2020

Spis treści

1. Wstęp

W dzisiejszych czasach nie wyobrażamy sobie realizowania operacji bankowych każdorazowo wybierając się do fizycznego oddziału banku. Założenie konta, przelew, utworzenie lokaty - wszystkie te operacje wykonujemy z domowego zacisza, najczęściej przez stronę internetową banku. Nasz innowacyjny projekt obejmuje utworzenie pierwszego w kraju kwitnącej cebuli systemu bankowości opartego nie o tradycyjne waluty, a o kryptowalutę Bitcoin. Niewątpliwą zaletą dydaktyczną projektu opartego o kryptowaluty jest faktyczna możliwość realizacji przelewów na zewnątrz tworzonego systemu, dzięki komunikacji z siecią blockchain.

1.1. Cel i założenia projektu

Celem projektu jest stworzenie systemu bankowości elektronicznej, podobnego w funkcjonowaniu do systemów znanych z tradycyjnych banków. System będzie udostępniał użytkownikom adres, na który mogą odbierać przelewy (zarówno wewnętrzne jak i zewnętrzne) oraz możliwość wykonania przelewu z posiadanych środków. Będzie pokrywał się możliwościami z tzw. portfelami Bitcoin, ale także rozszerzał je o dodatkowe funkcjonalności, na przykład utworzenie lokaty czy kontakt z pomocą techniczną.

2. Opis funkcjonalności

1. Utworzenie konta
    - Użytkownik podaje login, hasło lub
    - Użytkownik podaje login, hasło i e-mail
    - Użytkownik podaje swój klucz prywatny / seed phrase lub
    - System generuje klucz prywatny
2. Logowanie na konto
    - Użytkownik podaje login / e-mail i hasło
    - Wybór opcji „zapomniałem hasła” -> wysłanie wiadomości z nowym przez system jeśli e-mail jest w bazie
3. Zamknięcie konta
    - Opcja zapisania klucza prywatnego
    - Zamknięcie konta
4. Ekran główny
    - Odnośniki na górze strony: przelew, lokata, rachunki, odbiorcy zdefiniowani, historia, kontakt, rejestr zdarzeń, ustawienia, wylogowywanie
    - W części centralnej: adres publiczny, saldo, lokaty, ostatnie operacje
    - Na dole strony: data ostatniego logowania
5. Przelew
    - Na adres publiczny
    - Na adres e-mail -> generowane nowe konto w systemie jeśli e-mail nie jest w bazie
    - Wybór kwoty, daty, fee (podpowiedź optymalnego)
    - Możliwość dodania tytułu w przypadku operacji wewnętrznej w systemie
    - Konieczność potwierdzenia operacji kodem wysłanym na e-mail (jeśli został podany)
6. Lokata

   - Wybór kwoty, czasu trwania

7. Historia transakcji
   - Wyświetlenie historii operacji: tytuł, kwota, odbiorca, saldo po operacji, data zlecenia, księgowania, numer ID
   - Opcja filtrowania
   - Opcja powtórzenia operacji
   - Eksport historii do CSV
   - Eksport potwierdzenia przelewu do PDF
8. Kontakt
   - Skrzynka wiadomości do kontaktu z administracją serwisu
9. Ustawienia
   - Zmiana hasła
   - Zmiana / dodanie adresu e-mail
   - Zmiana motywu strony
   - Opcja włączenia / wyłączenia powiadomień o przelewie (przychodzącym, do potwierdzenia) i nieudanym logowaniu
   - Opcja włączenia / wyłączenia przesyłania wyciągów e-mailem
10. Rachunki
    - Opcja stworzenia podrachunku
    - Opcja zamknięcia podrachunku
11. Rejestr zdarzeń
    - Daty logowań, adresy IP
    - Daty operacji: przelewu, zmiany hasła itd.
12. Odbiorcy zdefiniowani
    - Księga adresowa

3. Przypadki użycia

1. Tworzenie konta
    - Aktorzy: Nowy użytkownik
    - Warunki wstępne: Nowy użytkownik chce stworzyć konto
    - Warunki końcowe: utworzenie konta
    - Warunki końcowe w przypadku niepowodzenia: konto nie zostaje utworzone
  - Scenariusz główny
    - System wyświetla formularz utworzenia konta
    - Użytkownik podaje login i hasło, opcjonalne jest podanie maila, w celu odzyskania hasła
    - System generuje klucz prywatny i publiczny. Publiczny zostaje podany do wiadomości użytkownika,
    - Możliwość dodania istniejącego walleta
  - Scenariusz alternatywny
    - System wyświetla formularz utworzenia konta
    - Użytkownik podaje login i hasło, opcjonalne jest podanie maila, w celu odzyskania hasła
    - Weryfikacja kończy się niepowodzeniem
    - Zostaje podana informacja o niepoprawnych danych
2. Sprawdzenie stanu konta
    - Aktorzy: Użytkownik
    - Warunki wstępne: zalogowany użytkownikator chce sprawdzić stan konta
    - Warunki końcowe: Sprawdzenie stanu konta
    - Warunki końcowe w przypadku niepowodzenia: użytkownik jest niezalogowany
  - Scenariusz główny
    - Użytkownik loguje się na swoje konto,
    - Po zalogowaniu na panelu startowym widać aktualny stan portfela
    - Użytkownik sprawdza stan konta
  - Scenariusz alternatywny
    - Logowanie na konto kończy się niepowodzeniem

3. Logowanie na konto
    - Aktorzy: Użytkownik
    - Warunki wstępne: Użytkownik chce zalogować się na konto
    - Warunki końcowe: Logowanie zakończone powodzeniem
    - Warunki końcowe w przypadku niepowodzenia: Logowanie zakończone niepowodzeniem
  - Scenariusz główny
    - System wyświetla formularz zalogowania
    - Użytkownik podaje e-mail lub login i hasło
    - System weryfikuje dane
    - Użytkownik zostaje zalogowany
    - Wyświetla się panel startowy
  - Scenariusz alternatywny
    - System wyświetla formularz zalogowania
    - Użytkownik podaje e-mail lub login i hasło
    - System weryfikuje dane
    - Weryfikacja kończy się niepowodzeniem
    - Wyświetla się ponownie ekran logowania
4. Wysyłanie przelewu
    - Aktorzy: Użytkownik
    - Warunki wstępne: Zalogowany użytkownik chcce przelać walutę
    - Warunki końcowe: Przelew zostaje wysłany
    - Warunki końcowe w przypadku niepowodzenia: przelew nie zostaje wysłany
  - Scenariusz główny
    - System wyświetla formularz przesłania waluty
    - Użytkownik podaje adres publiczny odbiorcy bądź adres e-mail,kwotę przelwu, datę i w przpadku transakcji w obrębie naszego banku- tytuł,
    - Wprowadzone dane są sprawdzane w systemie, w przypadku przelewu na e-mail, jeżeli nie jest on w bazie użytkowników, generowany jest e-mail z linkiem do utworzenia konta w naszym banku,
    - Pojawia się informacja o udanym przelewie, dane zostają wpisane do historii operacji
  - Scenariusz alternatywny
    - System wyświetla formularz przesłania waluty
    - Użytkownik podaje adres publiczny odbiorcy bądź adres e-mail,kwotę przelwu, datę i w przpadku transakcji w obrębie naszego banku- tytuł,

- Wprowadzone dane są odrzucone przez system; powodem może być niewystarczający stan konta, błędny adres
- Pojawia się informacja o nieudanym przelewie

5. Przeglądanie historii transakcji i eksport do CSV
    - Aktorzy: Użytkownik
    - Warunki wstępne: Zalogowany użytkownik
    - Warunki końcowe: Wyświetlona historia transakcji i wygenerowany jest plik CSV
    - Warunki końcowe w przypadku niepowodzenia: Historia nie zostanie wyświetlona, a plik nie zostanie wygeneowany
  - Scenariusz główny
    - Pojawia się panel historii transakcji, zawierająca informację dotycznąte przelewów przychodzących jak i wychodzących, takie jak kwota, data, odbiorca/nadawca, saldo po operacji
    - Użytwkonik wybiera operacje do eksportu,
    - System generuje plik CSV
  - Scenariusz alternatywny
    - Pojawia się panel historii transakcji, zawierająca informację dotycznąte przelewów przychodzących jak i wychodzących, takie jak kwota, data, odbiorca/nadawca, saldo po operacji
    - Użytwkonik wybiera operacje do eksportu,
    - System generuje informacje o wygaśnięciu sesji

6. Edycja konta- ustawienia
    - Aktorzy: Użytkownik
    - Warunki wstępne: Zalogowany użytkownik chce zmienić ustawienia
    - Warunki końcowe: Zmiany zostają zastosowane
    - Warunki końcowe w przypadku niepowodzenia: Zmiany zostają odrzucone
  - Scenariusz główny
    - System wyświetla panel ustawień
    - Użytkownik ma możliwość zmiany pól takich jak: e-mail, zmiana motywu strony, powiadomienia, wiadomości przychodzące e-mail,
    - System weryfikuje dane, takie jak poprawność e-maila,
    - System generuje informację o popranych zmianach wprowadzonych do systemu
  - Scenariusz alternatywny
    - System wyświetla panel ustawień

- Użytkownik ma możliwość zmiany pól takich jak: e-mail, zmiana motywu strony, powiadomienia, wiadomości przychodzące e-mail,
- System weryfikuje dane, takie jak poprawność e-maila,
- System generuje informację o niepoprawnych danych

7. Zakup lokaty
   - Aktorzy: Użytkownik
   - Warunki wstępne: Zalogowany użytkownik
   - Warunki końcowe: Lokata zostaje zakupiona
   - Warunki końcowe w przypadku niepowodzenia: Lokata nie zostaje zakupiona
   - Scenariusz główny
     - System wyświetla panel zakupu lokaty
     - Użytkownik wybiera spośród dostępnych lokat, np jednodniowa, tygodniowa, itd; wybiera kwotę do inwestycji,
     - System weryfikuje wprowadzone dane i stan konta
     - System tworzy lokate i generuje informację o udanym założeniu lokaty
   - Scenariusz alternatywny
     - System wyświetla panel zakupu lokaty
     - Użytkownik wybiera spośród dostępnych lokat, np jednodniowa, tygodniowa, itd; wybiera kwotę do inwestycji,
     - System weryfikuje wprowadzone dane i stan konta
     - System odrzuca prośbę i generuje informację o odrzuconym żądaniu

8. Tworzenie podrachunku
   - Aktorzy: Użytkownik
   - Warunki wstępne: Zalogowany użytkownik
   - Warunki końcowe: Utworzenie podrachunku
   - Warunki końcowe w przypadku niepowodzenia: podrachunek nie zostaje utworzony
   - Scenariusz główny
     - Pojawia się panel tworzenia podrachunku do jednego portfela
     - Użytkownik podaje nazwę podrachunku
     - System weryfikuje nazwę pod kontem niewłaściwych znaków
     - Podrachunek zostaje utworzony i wyświetlony stosowny komunikat
   - Scenariusz alternatywny
     - Pojawia się panel tworzenia podrachunku do jednego portfela
     - Użytkownik podaje nazwę podrachunku
     - System weryfikuje nazwę pod kontem niewłaściwych znaków

- Podrachunek nie zostaje utworzony i wyświetlony zostaje stosowny komunikat

9. Kontakt z Pomocą
    - Aktorzy: Użytkownik
    - Warunki wstępne: Zalogowany użytkownik
    - Warunki końcowe: Wysłanie wiadomości
    - Warunki końcowe w przypadku niepowodzenia: Wiadomość nie zostaje wysłana
  - Scenariusz główny
    - System wyświetla panel kontaktu
    - Użytkownik wybiera powód kontaktu z listy
    - Użytkownik wpisuje wiadomość
    - Wiadomość zostaje wysłana i wyświetlony stosowny komunikat
  - Scenariusz alternatywny
    - System wyświetla panel kontaktu
    - Użytkownik wybiera powód kontaktu z listy
    - Użytkownik wpisuje wiadomość
    - Wiadomość nie zostaje wysłana z powodu wygaśnięcia sesji, wyświetlony stosowny komunikat

10. Odpowiedź na pytanie użytkownika
    - Aktorzy: Administrator
    - Warunki wstępne: Zalogowany administrator
    - Warunki końcowe: Wysłanie wiadomości
    - Warunki końcowe w przypadku niepowodzenia: Wiadomość nie zostaje wysłana
  - Scenariusz główny
    - System wyświetla panel kontaktu
    - Administator wybiera wiadomość z listy otrzymanych
    - Administator wpisuje swoją odpowiedź
    - Wiadomość zostaje wysłana i wyświetlony stosowny komunikat
  - Scenariusz alternatywny
    - System wyświetla panel kontaktu
    - Administator wybiera wiadomość z listy otrzymanych
    - Administator wpisuje swoją odpowiedź
    - Wiadomość nie zostaje wysłana z powodu wygaśnięcia sesji, wyświetlony stosowny komunikat

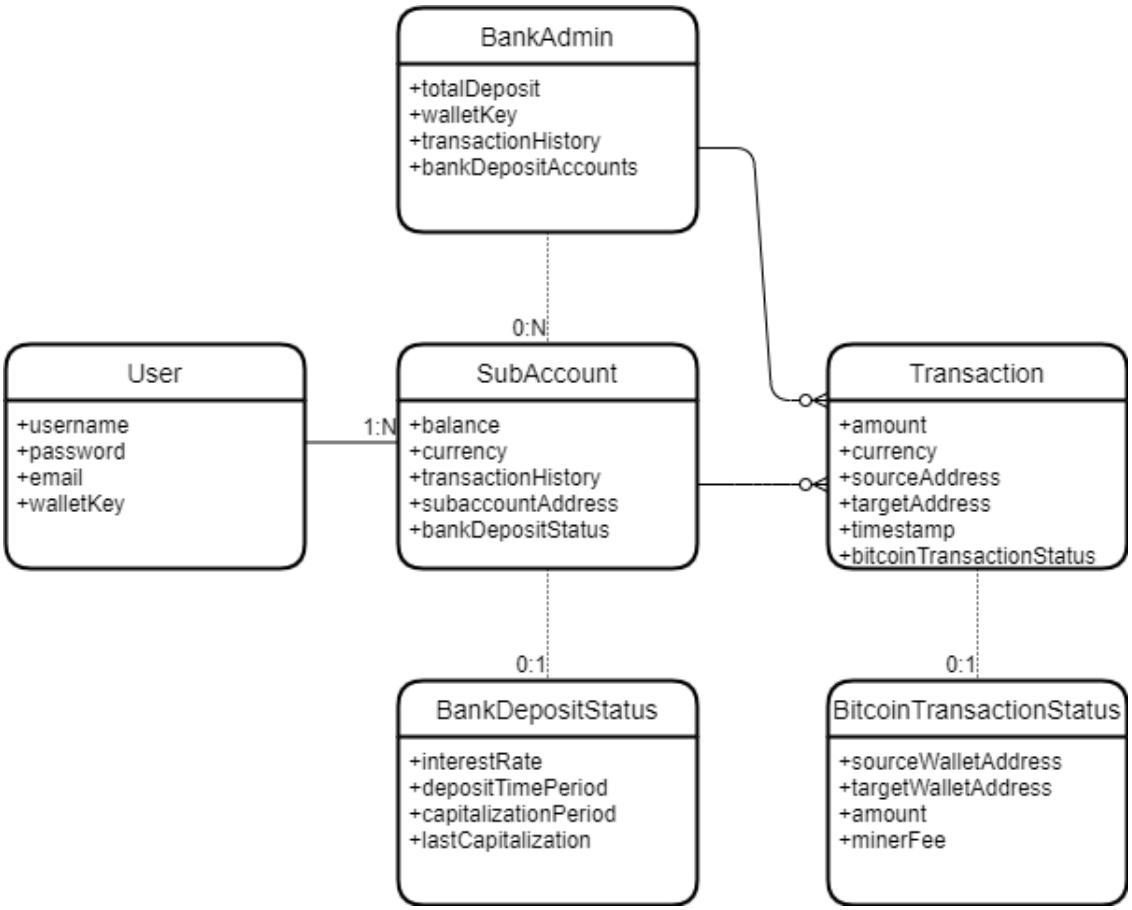
4. Wymagania niefunkcjonalne

1. Funkcjonalność
   - Poprawne działanie na przeglądarkach internetowych w wersji nie starszej niż: Chrome 85, Firefox 81, Edge 85
   - Minimalna obsługiwana rozdzielczość ekranu: 1366x768
   - Nowoczesność i zgodność z aktualnymi standardami na rynku
   - Budowa systemu za pomocą narzędzi umożliwiających rozwój aplikacji
   - Skalowalność wydajnościowa
2. Estetyka
   - Projekt logo serwisu
   - Logo serwisu pojawiające się na stronie internetowej oraz w eksportowanych plikach PDF
3. Użyteczność
   - Prostota i intuicyjność obsługi, pozwalająca na obsługę osobom nie posiadającym umiejętności technicznych
   - Polska wersja językowa interfejsu
4. Niezawodność
   - Uptime na poziomie minimum 90%
   - Dostępność 24/7/365
5. Wydajność
   - Umożliwienie korzystania z aplikacji równocześnie przez co najmniej 4 użytkowników na raz
   - Maksymalny czas odpowiedzi systemu na akcję użytkownika nie dłuższy niż 4 sekundy
   - Minimalizacja zużycia procesora, pamięci RAM i przestrzeni dyskowej
6. Bezpieczeństwo
   - Ochrona przed nieautoryzowanym dostępem do danych użytkowników

7. Wsparcie
    - Naprawa błędów krytycznych aplikacji w ciągu 48 godzin
8. Sposób wdrożenia
    - Dostarczenie aplikacji w terminie do 17 stycznia 2020
    - Dokumentacja zawierająca zrzuty ekranu z opisem użytkowania aplikacji
    - Uwzględnienie w dokumentacji perspektywy użytkownika i programisty

5. Architektura systemu

5.1. Schemat bazy danych

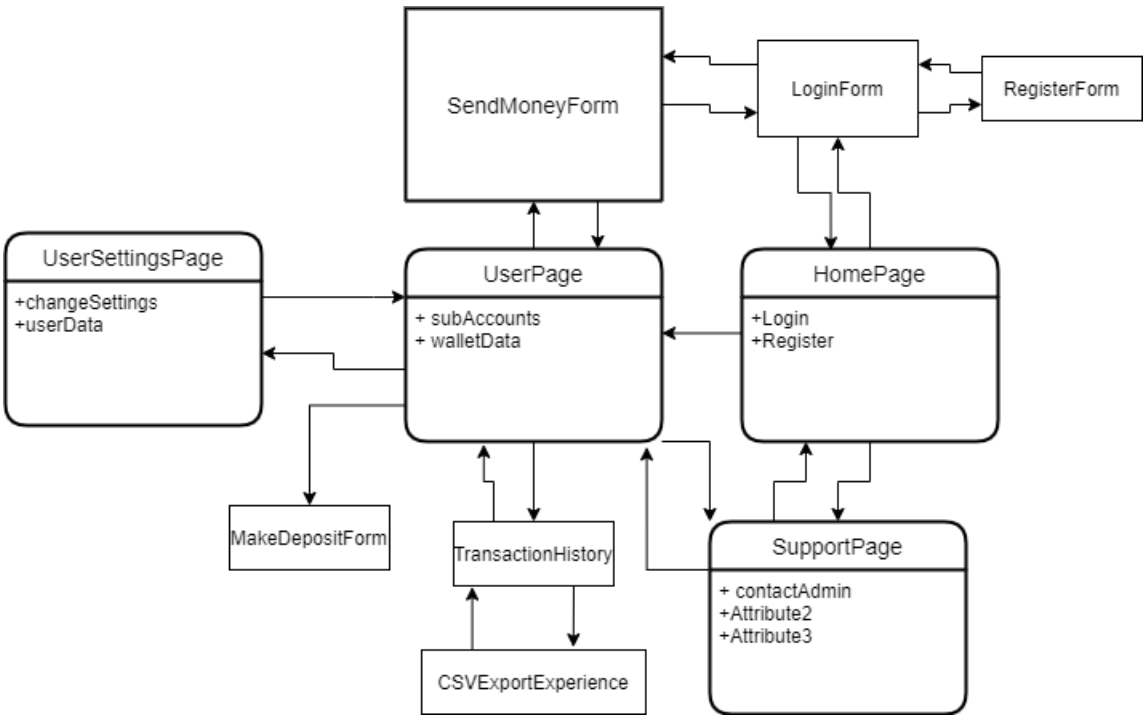


6. Opis interfejsów

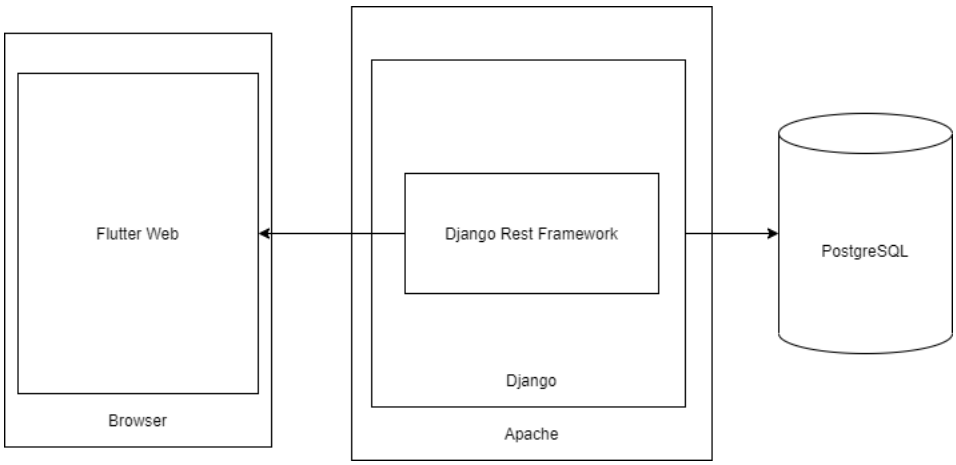
6.1. Interfejsy logiczne

- **Aplikacje backendowe <-> Baza danych PostgreSQL** - moduł Python psycopg2
- **Aplikacje backendowe <-> Blockchain** - moduł Python bit
- **Serwer HTTP <-> Aplikacje backendowe** - komunikacja za pomocą Mod WSGI
- **Aplikacje frontendowe <-> Aplikacje backendowe** - zakończenia REST API zdefiniowane przy użyciu Django REST Framework

6.2. Ogólny flow UI



7. Stos technologiczny



Użyte technologie:

- **Apache** - serwer webowy zajmujący się odbieraniem i odpowiadaniem na żądania HTTP
- *mod_wsgi* - interfejs łączący aplikacje backendu Django z serwerem Apache
- **Django** - framework do tworzenia aplikacji webowych w języku Python; zostanie wykorzystany do stworzenia aplikacji backendu; umożliwia komunikację z bazą danych przy pomocy kodu Pythona
- **Django Rest Framework** - framework umożliwiający tworzenie REST API do komunikacji backendu z frontendem; w prosty sposób integruje się z Django i korzysta z jego funkcjonalności
- **bit** - biblioteka Python do obsługi tranzakcji w bitcoinach
- **PostgreSQL** - relacyjna baza danych
- **Flutter Web** - framework umożliwiający użycie zestawu narzędzi Flutter do tworzenia aplikacji webowych; zostanie wykorzystany do stworzenia klienta przeglądarkowego aplikacji

8. Projekt testów

- **Testy jednostkowe** – testowanie metod we wszystkich modułach z wykorzystaniem wbudowanych bibliotek – unittest dla Django oraz flutter_test dla Flutter.
- **Testy integracyjne** – testowanie komunikacji i odpowiedniej współpracy między modułami, w szczególności między frontendem i backendem oraz backendem i bazą danych. Również zostną wykorzystane moduły unittest oraz flutter_test.
- **Testy funkcjonalne** – testowanie funkcjonalności aplikacji oraz scenariuszy użycia za pomocą Selenium oraz manualnego testowania.
- **Testy wydajnościowe** – testowanie czasu odpowiedzi oraz obciążenia z użyciem Locust.

9. Analiza ryzyka

9.1. Lista ryzyk

1. Skomplikowana integracja z siecią blockchain, niedostateczna wiedza na temat jej działania.

2. Zachowanie spójności między operacjami wykonywanymi lokalnie, w bazie serwisu, a tymi w sieci blockchain.

3. Brak doświadczenia w wykorzystywanych technologiach - dla większości członków projektu tworzenie aplikacji webowych jest czymś nowym.

4. Wybór złych technologii lub bibliotek, niosący za sobą konieczność przepisywania oprogramowania.

5. Założona funkcjonalność może okazać się niemożliwa do realizacji.

6. Niedotrzymanie terminów - trudność oszacowania nakładu czasowego potrzebnego na projekt.

7. Niewykrycie błędów podczas procesu testowania. Im wcześniej zostanie wykryty błąd, tym łatwiej go usunąć.

8. Trudność zapewnienia bezpieczeństwa danych - odporność na włamania, ataki typu sql injection.

9. Trudności w sprawdzeniu skalowalności.

9.2. Macierz ryzyka

Prawdopodobieństwo \ Konsekwencje	niewielkie	średnie	poważne
niewielkie			-Wybór złych technologii
średnie	-Sprawdzenie skalowalności	-Niedostateczna wiedza na temat blockchain	-Założona funkcjonalność nie będzie możliwa do zrealizowania -Niedotrzymanie terminu
wysokie	-Brak doświadczenia w wybranych technologiach	-Zachowanie spójności między operacjami lokalnymi i na blockchainie -Niewykrycie (niekrytycznych) błędów	-Zapewnienie bezpieczeństwa danych

9.3. Sposoby reagowania na ryzyko

1. **Unikanie ryzyka** - Dokonanie szczegółowej i wszechstronnej analizy dostępnych technologii pozwoli na wybór optymalnej i zapewniającej całkowitą realizację założeń projektu (uwzględniając potencjał rozbudowy). Dzięki planowaniu prac z wyprzedzeniem, nawet przy niedoszacowaniu czasu potrzebnego na implementację, możliwe będzie dotrzymanie terminów. W unikaniu ryzyka pomocne są również analiza statyczna kodu i testowanie, w zakresie bezpieczeństwa - zlecenie audytu zewnętrznego (w „rzeczywistych" projektach).

2. **Przeniesienie ryzyka** - wykorzystując gotowe, dobrze i ustawicznie testowane narzędzia (frameworki) część ryzyka, zwłaszcza w obszarze bezpieczeństwa, przenosi się na podmioty dostarczające dane technologie.

3. **Akceptacja ryzyka** - niektóre zagrożenia nie są możliwe do całkowitej eliminacji.

10. Narzędzia wspomagające realizację projektu

– **GitHub**: wykorzystywany do przechowywania i inspekcji kodu, dyskusji, przydzielania zadań, śledzenia błędów i wkładu każdej z osób.

– **Overleaf**: edytor LaTeX on-line, umożliwiający współpracę kilku osób jednocześnie.

– **Facebook Messenger**: komunikator. Wykorzystywany zarówno w formie tekstowej, ale także w formie rozmów głosowych podczas intensywnych prac nad projektem.

Ze względu na ograniczoną skalę wykonywanego przez nas projektu, ograniczamy się do wymienionych wyżej rozwiązań. W przypadku znacznie większych projektów, zasadne wydaje się wykorzystanie dodatkowego oprogramowania, na przykład **JIRA** - do śledzenia i planowania pracy czy **Confluence** - do przechowywania treści pojawiających się w trakcie rozwoju projektu w jednym miejscu: członków zespołu i ich ról, haseł, wymagań i innej dokumentacji.

Bibliografia

[1] Jimmy Song. *Programming Bitcoin: Learn How to Program Bitcoin from Scratch*. O'Reilly Media, 2019.