



**Politechnika  
Śląska**

Wydział Informatyki, Elektroniki i Informatyki

# **Aplikacja do zarządzania budżetem domowym**

**TWORZENIE APLIKACJI BAZODANOWYCH**

- Mateusz Cudzik
- Jakub Ferens
- Mateusz Górecki
- Szymon Maciąg
- Kajetan Sommer
- Julia Wojciuch

Gliwice

20 czerwca 2023

# Spis treści

<b>1</b>	<b>Wstęp</b>	<b>3</b>
<b>2</b>	<b>Określenie wymagań</b>	<b>3</b>
2.1	Wymagania funkcjonalne . . . . .	3
2.2	Wymagania niefunkcjonalne . . . . .	3
<b>3</b>	<b>Analiza MoSCoW</b>	<b>4</b>
3.1	Must . . . . .	4
3.2	Should . . . . .	4
3.3	Could . . . . .	4
3.4	Won't . . . . .	5
<b>4</b>	<b>Scenariusze przypadków użycia</b>	<b>5</b>
4.1	Logowanie . . . . .	5
4.1.1	Scenariusz główny logowania . . . . .	5
4.1.2	Scenariusze poboczne logowania . . . . .	5
4.2	Dodawanie przychodów/wydatków . . . . .	6
4.2.1	Scenariusz główny dodawania przychodów/wydatków .	6
4.2.2	Scenariusze poboczne dodawania przychodów/wydatków	6
<b>5</b>	<b>Diagram UML</b>	<b>8</b>
<b>6</b>	<b>Schemat bazy danych</b>	<b>10</b>
<b>7</b>	<b>Specyfikacja zewnętrzna</b>	<b>11</b>
7.1	Rejestracja . . . . .	11
7.1.1	Rejestracja konta użytkownika . . . . .	11
7.1.2	Rejestracja konta dziecka . . . . .	12
7.2	Logowanie . . . . .	13
7.3	Ekran główny . . . . .	14
7.4	Profil dziecka . . . . .	15
7.5	Przychody . . . . .	16
7.6	Wydatki . . . . .	17
7.7	Dodaj kategorię . . . . .	18

---

7.8	Dodawanie przychodów i wydatków . . . . .	19
7.8.1	Dodaj przychód . . . . .	19
7.8.2	Dodaj wydatek . . . . .	20
7.9	Wykonaj przelew . . . . .	21
<b>8</b>	<b>Specyfikacja wewnętrzna</b>	<b>21</b>
8.1	Wykorzystane technologie . . . . .	21
8.2	Warstwy systemu . . . . .	22
8.2.1	Warstwa bazy danych . . . . .	22
8.2.2	Warstwa logiki biznesowej . . . . .	22
8.2.3	Warstwa wizualna . . . . .	22
8.2.4	Warstwa autoryzacji . . . . .	22
<b>9</b>	<b>Weryfikacja osiągniętych efektów względem założeń</b>	<b>23</b>
9.1	Schemat zaimplementowanej bazy danych . . . . .	23
9.2	Analiza wymagań i ich zmian względem założeń . . . . .	23
9.2.1	Analiza wymagań funkcjonalnych . . . . .	23
9.2.2	Analiza wymagań niefunkcjonalnych . . . . .	24
<b>10</b>	<b>Testowanie i uruchamianie</b>	<b>24</b>
<b>11</b>	<b>Uwagi i wnioski</b>	<b>24</b>

# 1 Wstęp

Celem projektu jest stworzenie aplikacji do zarządzania domowym budżetem i jego monitorowania. Warunkiem koniecznym jest obsługiwanie logowania, co umożliwi korzystanie z niej wielu użytkownikom. Aplikacja pozwoli na tworzenie raportów oraz kategoryzowanie przychodów i wydatków z różnych kont.

## 2 Określenie wymagań

### 2.1 Wymagania funkcjonalne

- Obsługa logowania,
- Kategorie wydatków/przychodów,
- Kategorie kont,
- Transakcje między profilami,
- Generowanie raportów — analiza finansowa,
- Przechowywanie skanów paragonów/faktur,
- Przechowywanie dłużników,
- Informacje przechowywane w bazie danych,
- Dodawanie profili członków rodziny do konta (profil dziecka, rodzica itd.),
- Dodanie konta bankowego i operacji na nim,
- Obsługa wydatków i przychodów.

### 2.2 Wymagania niefunkcjonalne

- Bezpieczeństwo — okresowe tworzenie kopii zapasowych danych,
- Zabezpieczenie profili użytkowników hasłem,
- Hierarchia użytkowników — różne poziomy uprawnień/dostępu, ograniczenia dla profili młodszych użytkowników,
- Użyteczność — aplikacja z przystępnym i łatwym w obsłudze interfejsem zarówno dla starszych, jak i młodszych użytkowników,
- Wieloplatformowość — przypadku aplikacji webowej dostępność z różnych urządzeń przy pomocy dowolnego systemu posiadającego przeglą-

- darke,
- System/Aplikacja przystosowana do łatwego rozwoju, rozbudowy i aktualizacji,
- Responsywność — odpowiedź aplikacji na działania użytkownika w określonym czasie (przykładowo do trzech sekund).

## 3 Analiza MoSCoW

### 3.1 Must

- Przechowywanie informacji w bazie danych,
- dodawanie wydatków i przychodów,
- generowanie raportów,
- założenie konta i przypisania do niego danych,
- informowanie użytkownika o aktualnym stanie konta, który jest zmieniany wraz z kolejnymi wpisami o przychodach/wydatkach.

### 3.2 Should

- Przypomnienie hasła,
- formularz rejestracji dostępny dla użytkownika,
- dzielenie wydatków i przychodów na kategorie,
- generowanie raportów z podziałem wydatków/przychodów na kategorie,
- operacje zarządzania profilami (dodawanie, usuwanie itd.).

### 3.3 Could

- Potwierdzenie rejestracji mailem,
- edycja informacji o koncie (nazwy użytkownika, hasła itd.),
- ustawianie cyklicznych/stałych wydatków/przychodów,
- transakcje między profilami,
- przechowywanie skanów paragonów/faktur,
- definiowanie własnych, niestandardowych kategorii.

### 3.4 Won't

- Weryfikacja Captcha,
- przechowywanie informacji o dłużnikach,
- powiadomienia o przekroczonym budżecie.

## 4 Scenariusze przypadków użycia

### 4.1 Logowanie

#### 4.1.1 Scenariusz główny logowania

- Przypadek rozpoczyna się, gdy niezalogowany użytkownik wejdzie na stronę.
- Użytkownik wpisuje swój login oraz hasło.
- System sprawdza poprawność danych.
- Użytkownik zostaje przeniesiony do panelu wyboru profilu.
- Użytkownik wybiera profil.
- Wyświetlony zostaje panel sterowania budżetem.
- Użytkownik zostaje zalogowany.

#### 4.1.2 Scenariusze poboczne logowania

##### Konto nie istnieje

- Użytkownik zostaje przeniesiony do formularza rejestracji.
- Użytkownik wprowadza swoje dane.
- System sprawdza poprawność danych.
- Konto zostaje utworzone.

##### Wybrany profil jest chroniony

- Użytkownik wpisuje PIN.
- System sprawdza poprawność danych.
- W przypadku wprowadzenia poprawnego kodu pin scenariusz się kończy, w przeciwnym razie użytkownik jest informowany o błędnym kodzie PIN, po kilku błędnych próbach nakładana jest czasowa blokada.

**Wybrany profil jest profilem dziecka**

- Użytkownikowi wyświetlone zostaje uproszczone GUI.

## **4.2 Dodawanie przychodów/wydatków**

### **4.2.1 Scenariusz główny dodawania przychodów/wydatków**

- Zalogowany użytkownik decyduje się dodać przychód/wydatek na panelu sterowania budżetem.
- Użytkownik wpisuje kwotę, nazwę własną operacji oraz wybiera jej kategorię.
- Operacja zostaje uwzględniona w budżecie.

### **4.2.2 Scenariusze poboczne dodawania przychodów/wydatków**

**Użytkownik dodaje własną kategorię**

- Użytkownik podaje nazwę i wybiera kolor.
- Kategoria zostaje dodana do listy wszystkich kategorii.

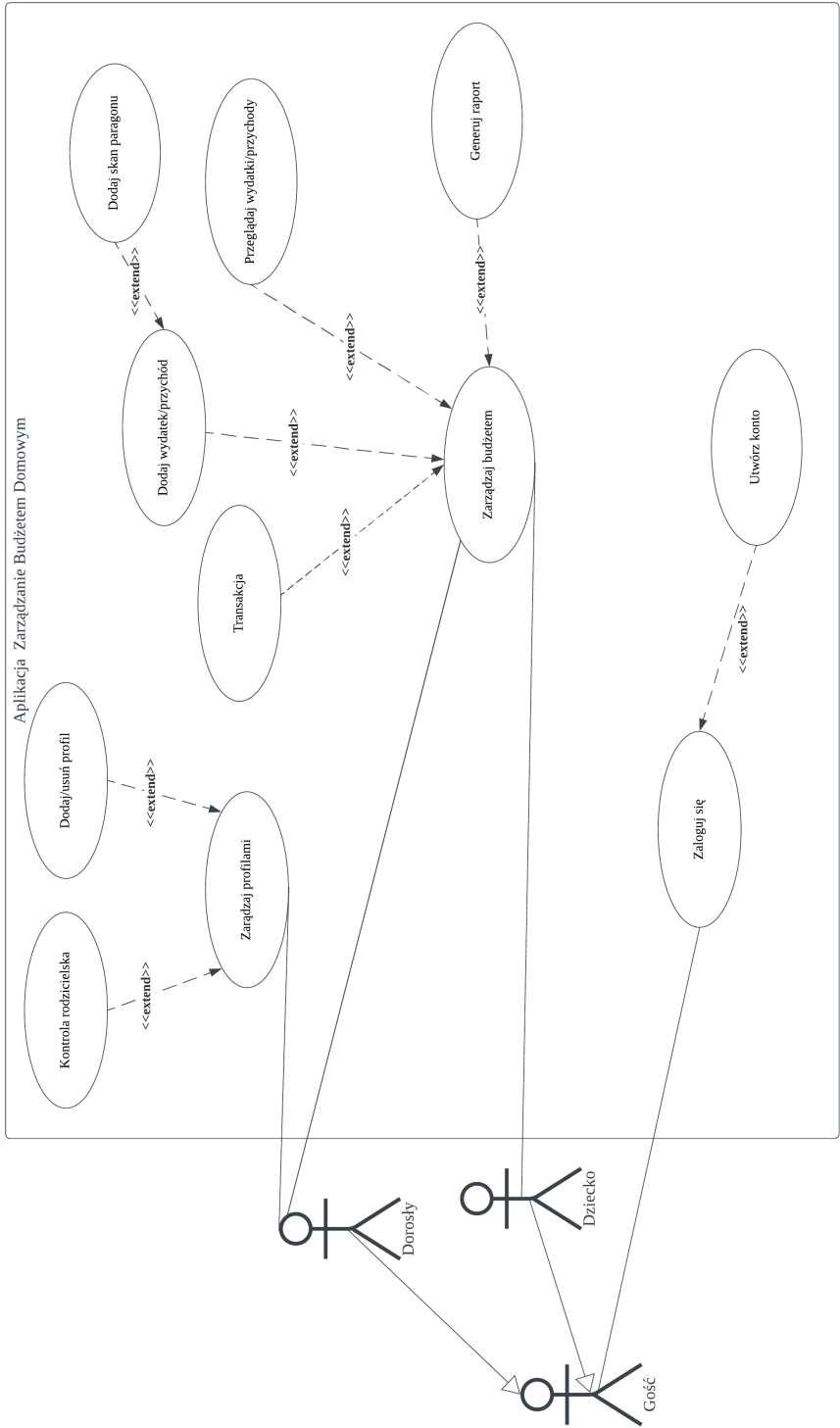
**Użytkownik dodaje wydatek przekraczający saldo**

- Użytkownik zostaje ostrzeżony za pomocą powiadomienia przed wykonaniem operacji.
- Użytkownik anuluje lub potwierdza wykonanie transakcji.





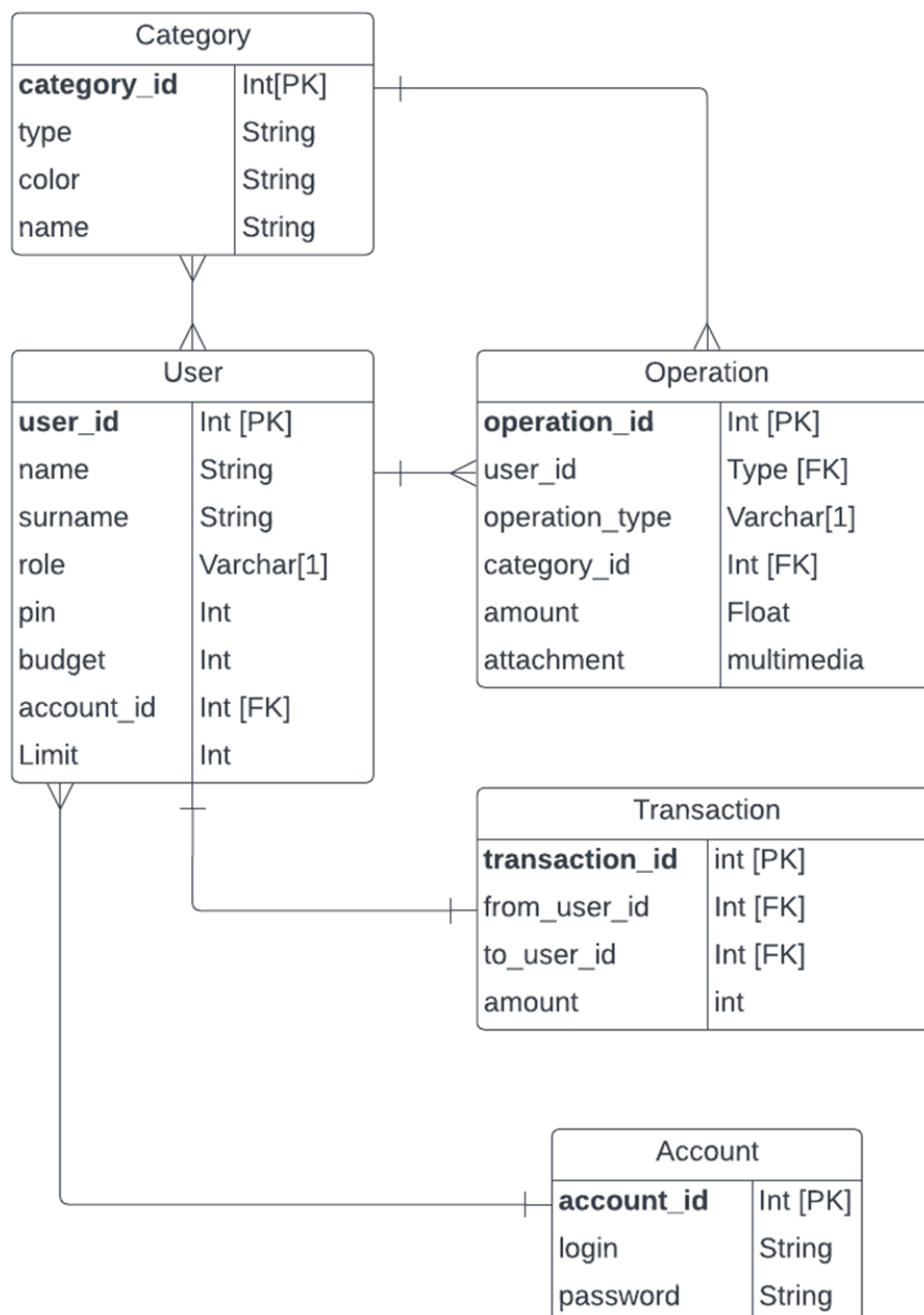
5 Diagram UML



Rysunek 1: Diagram UML.



## 6 Schemat bazy danych



Rysunek 2: Prototypowy schemat bazy danych

## 7 Specyfikacja zewnętrzna

### 7.1 Rejestracja

#### 7.1.1 Rejestracja konta użytkownika

The image shows a registration form titled "REJESTRACJA" centered on a light yellow background. The form itself is a darker yellow rectangle containing four white input fields with rounded corners, each with a label inside: "Username", "E-mail", "Password", and "Kwota startowa". Below these fields is a blue button with white text that says "DODAJ KONTO".

Rysunek 3: Zrzut ekranu rejestracji

Użytkownik, chcąc skorzystać z aplikacji, musi posiadać własny profil. Formularz rejestracyjny pozwala założyć konto. Celem założenia konta użytkownik musi podać:

- nazwę użytkownika
- adres e-mail
- hasło
- kwotę początkową.

Naciśnięcie przycisku **DODAJ KONTO** skutkuje dodaniem konta do bazy danych.

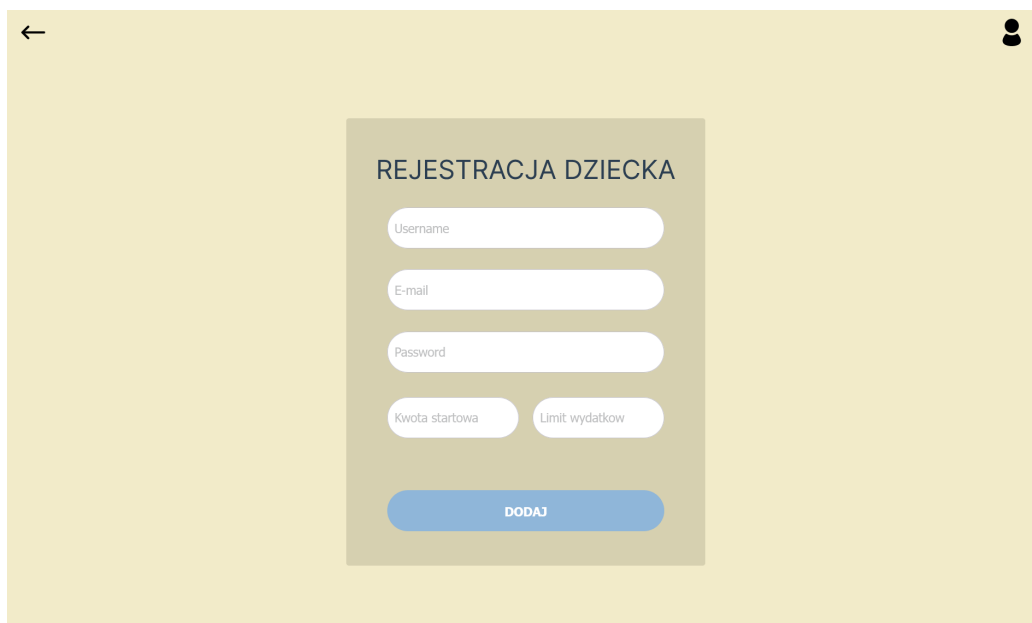


The screenshot shows a registration form titled "REJESTRACJA" on a light yellow background. The form is contained within a grey rectangular box. It features four input fields: a username field with "user90", an email field with "abc@abc.abm", a password field with masked characters "\*\*\*\*\*", and a phone number field with "1250". Below these fields, a red error message states "Username 'user90' is already taken." At the bottom of the form is a blue button labeled "DODAJ KONTO".

Rysunek 4: Zrzut ekranu rejestracji zakończonej niepowodzeniem.

Jeżeli w bazie danych istnieje konto o takich samych danych, system informuje użytkownika o niemożności jego założenia stosownym komunikatem.

### 7.1.2 Rejestracja konta dziecka

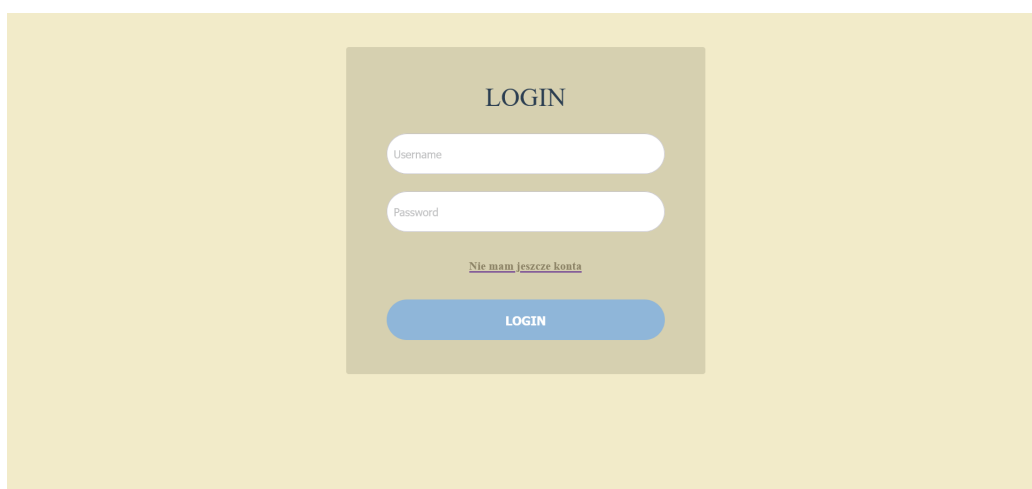


The screenshot shows a registration form titled "REJESTRACJA DZIECKA" on a light yellow background. The form is contained within a grey rectangular box. It features five input fields: "Username", "E-mail", "Password", "Kwota startowa", and "Limit wydatkow". At the bottom of the form is a blue button labeled "DODAJ". The form is displayed on a screen with a back arrow in the top left and a user icon in the top right.

Rysunek 5: Zrzut ekranu rejestracji konta dziecka.

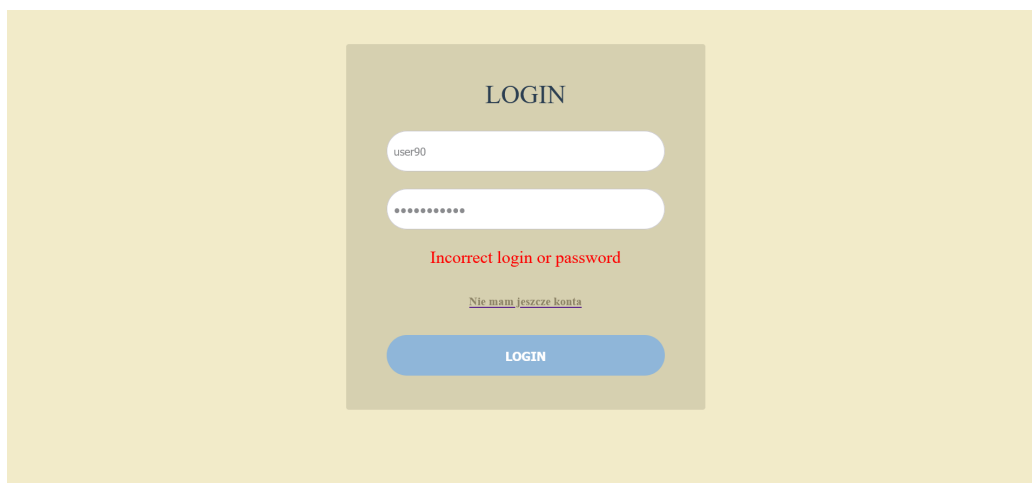
Użytkownik dorosły może założyć konto o ograniczonym dostępie dedykowane dzieciom. Zasadniczą różnicą jest możliwość nadania limitu wydatków przypisanego do tego typu profilu.

## 7.2 Logowanie



Rysunek 6: Zrzut ekranu logowania.

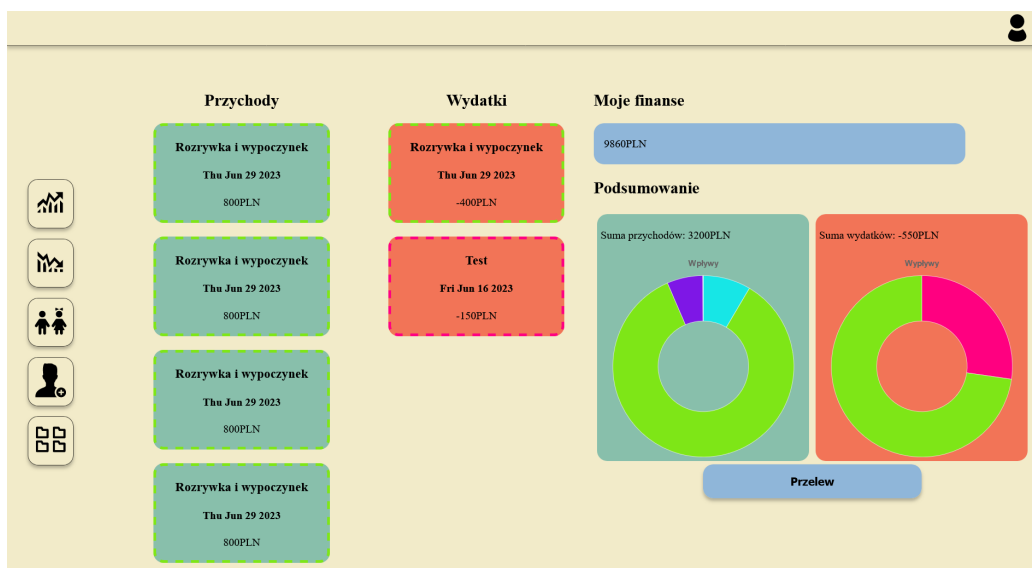
Strona logowania zawiera formularz, w którym należy wprowadzić login użytkownika oraz hasło. Po wprowadzeniu poprawnych danych oraz naciśnięciu przycisku **LOGIN** użytkownik zostaje przeniesiony do ekranu głównego aplikacji. Jeżeli użytkownik nie posiada konta, może w prosty sposób przejść do formularza rejestracji, klikając w odnośnik **Nie mam jeszcze konta**.



Rysunek 7: Zrzut ekranu błędnego logowania.

W przypadku podania niepoprawnych danych użytkownik jest informowany o fakcie stosownym komunikatem.

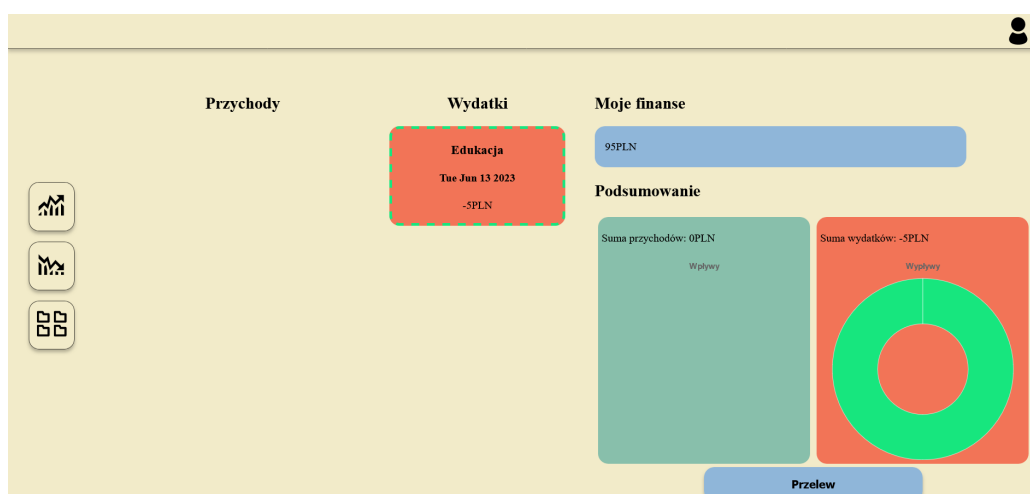
### 7.3 Ekran główny



Rysunek 8: Zrzut ekranu głównego aplikacji.

Poprawne logowanie skutkuje przeniesieniem użytkownika do ekranu głównego aplikacji. Zakładka **moje finanse** informuje korzystającego z systemu o aktualnym stanie konta. Wyświetlana po lewej stronie lista **przychodów** oraz **wydatków** wyświetla ostatnio dodane operacje. Lista ta zawiera cztery ostatnie dodane rekordy, wraz z nazwą kategorii, do jakiej operacja została przypisana, datą wykonania oraz kwotą. Co więcej, ów lista zawiera kolorową ramkę w kolorze przypisanym do danej kategorii, by pomóc korzystającemu z aplikacji rozróżnić kategorie operacji. Sekcja **podsumowanie** prezentuje zsumowane kwoty wydatków i przychodów, a pod wartościami wyświetlone są wykresy kołowe operacji wykonanych w ostatnim miesiącu, które prezentują podział operacji na kategorie i ich udział w całkowitej sumie operacji. Poniżej podsumowania znajduje się przycisk **Przelew**, który pozwala na przetransferowanie kwoty wewnątrz konta pomiędzy profilami. Zostaną wtedy utworzone stosowne rekordy na profilu źródłowym oraz docelowym. Ikony znajdujące się przy lewej krawędzi strony odpowiadają za przejście do karty wydatków lub przychodów, dodanie konta dziecka, konta z pełnymi uprawnieniami oraz kategorii.

## 7.4 Profil dziecka

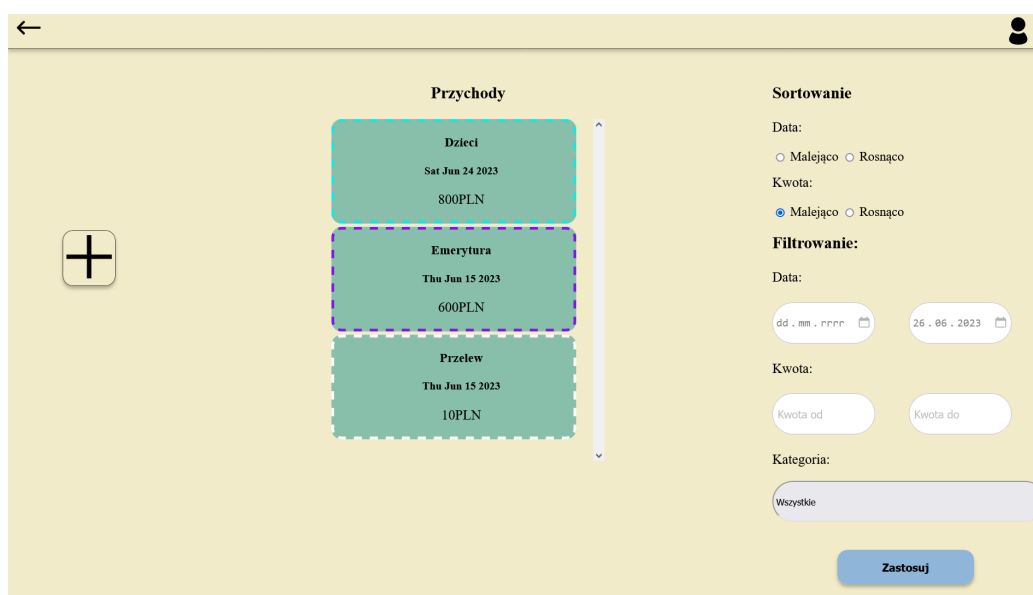


Rysunek 9: Zrzut ekranu głównego dziecka.



Ekran główny dziecka zawiera mniej opcji niż panel dorosłego. Profil może jedynie wykonać przelew wewnętrzny, wygenerować raport, wyświetlić kartę wydatków oraz przychodów, dodawać je oraz utworzyć niestandardową kategorię.

## 7.5 Przychody



Rysunek 10: Zrzut ekranu szczegółowego przychodów.

Po otwarciu panelu **Przychody** użytkownikowi prezentowane są wszystkie wykonane operacje, wraz z ich kwotą, datą wykonania oraz kategorią. Korzystający z aplikacji ma możliwość sortowania oraz filtrowania względem daty, kwoty lub kategorii za pomocą przycisku **Zastosuj**. Obok niego znajduje się przycisk **Generuj raport** pozwalający na wygenerowanie raportu z wykonanych operacji na koncie. Aby przejść do ekranu dodawania operacji, należy nacisnąć przycisk +.

## 7.6 Wydatki

The screenshot displays the 'Wydatki' (Expenses) screen. On the left, there is a large plus icon (+) for adding new expenses. The main area shows a list of expenses:

- Rozrywka i wypoczynek**  
Thu Jun 29 2023  
-400
- Test**  
Fri Jun 16 2023  
-150

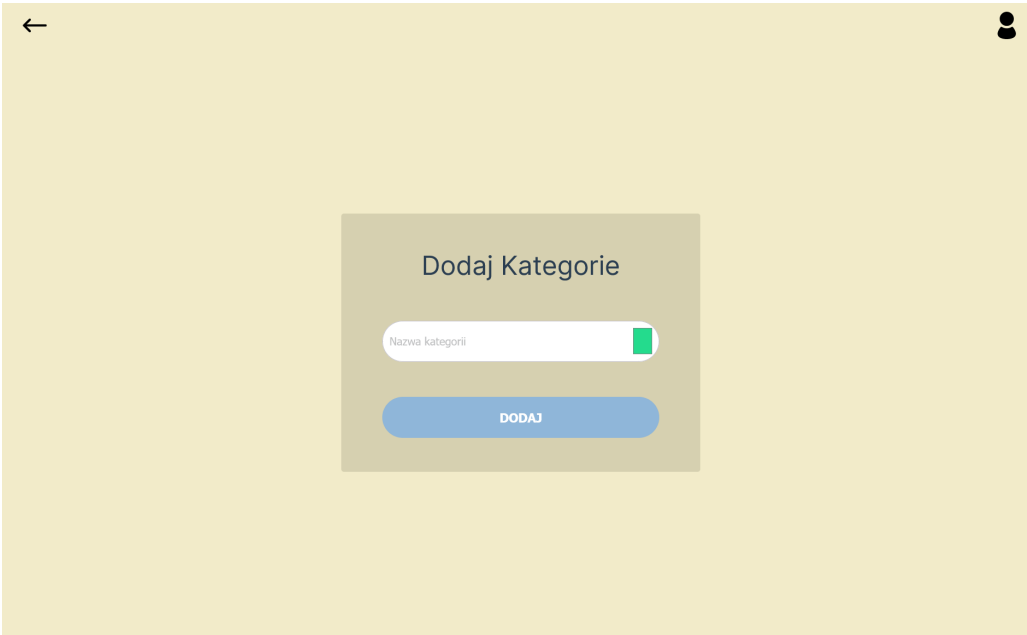
On the right, there is a sidebar with the following sections:

- Sortowanie**  
Data:  
☐ Malejąco ☐ Rosnąco  
Kwota:  
☐ Malejąco ☐ Rosnąco
- Filtrowanie:**  
Data:  
dd . mm . rrrr    
Kwota:  
Kwota od  Kwota do   
Kategoria:  
Wszystkie
- Zastosuj**

Rysunek 11: Zrzut ekranu szczegółowego wydatków.

W przypadku wydatków ekran posiada analogiczne funkcje do ekranu dodawania przychodów.

## 7.7 Dodaj kategorię



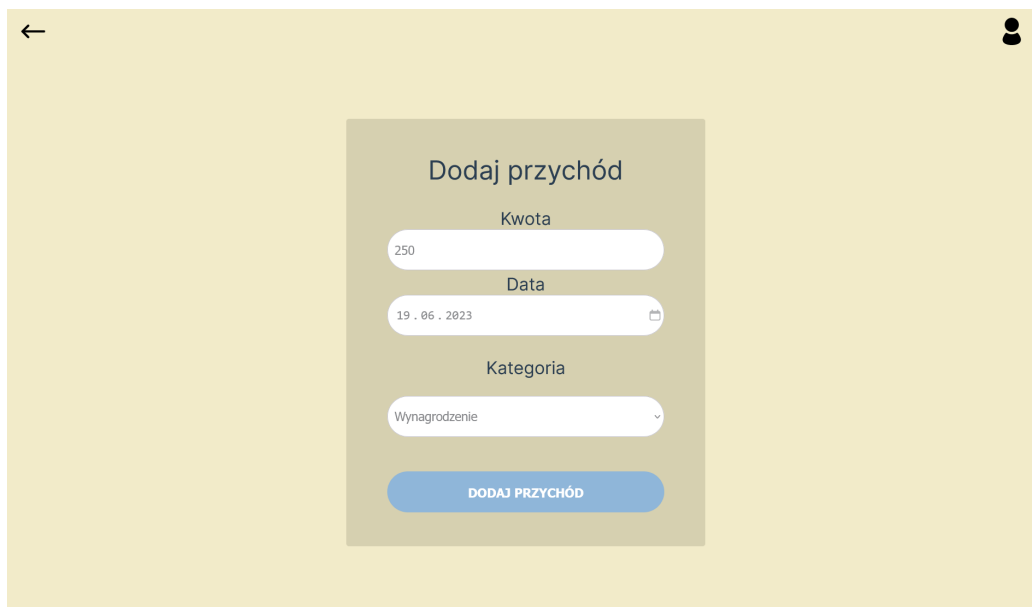
The screenshot shows a mobile application interface for adding a category. The background is a solid light yellow. In the top-left corner, there is a black back arrow icon. In the top-right corner, there is a black user profile icon. Centered on the screen is a grey rectangular box. Inside this box, at the top, is the text 'Dodaj Kategorie'. Below the text is a white text input field with the placeholder text 'Nazwa kategorii' and a green selection bar on the right side. At the bottom of the grey box is a blue rounded rectangular button with the white text 'DODAJ'.

Rysunek 12: Zrzut ekranu dodawania kategorii.

Ekran dodawania kategorii pozwala wprowadzić jej nazwę oraz nadać kolor.

## 7.8 Dodawanie przychodów i wydatków

### 7.8.1 Dodaj przychód



←

⋮

**Dodaj przychód**

Kwota

250

Data

19 . 06 . 2023

Kategoria

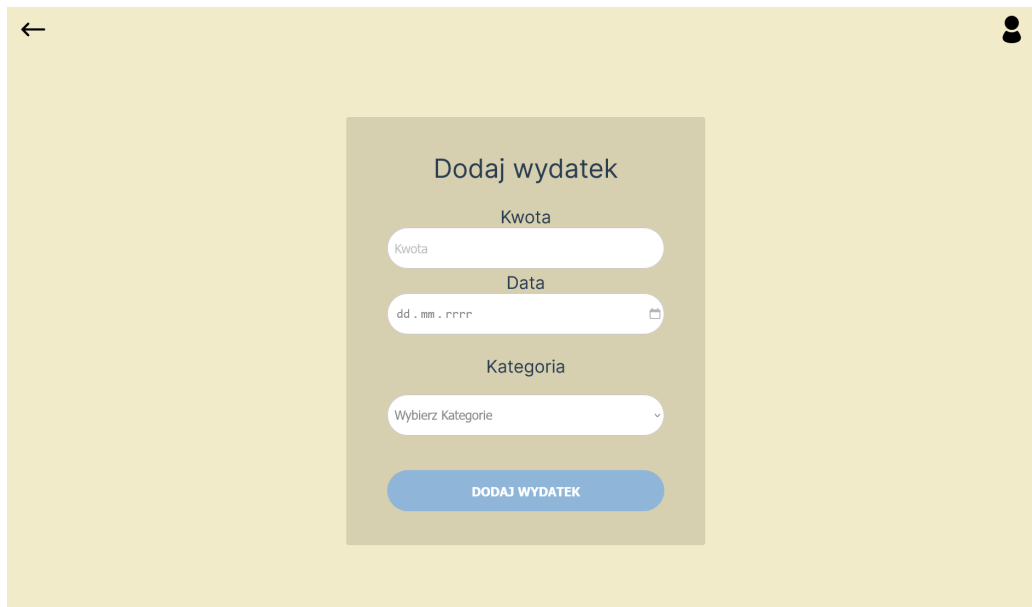
Wynagrodzenie

DODAJ PRZYCHÓD

Rysunek 13: Zrzut ekranu dodawania przychodu.

Ekran dodawania kategorii umożliwia wprowadzenie kwoty operacji, daty wykonania oraz kategorii, do której ów operacja należy.

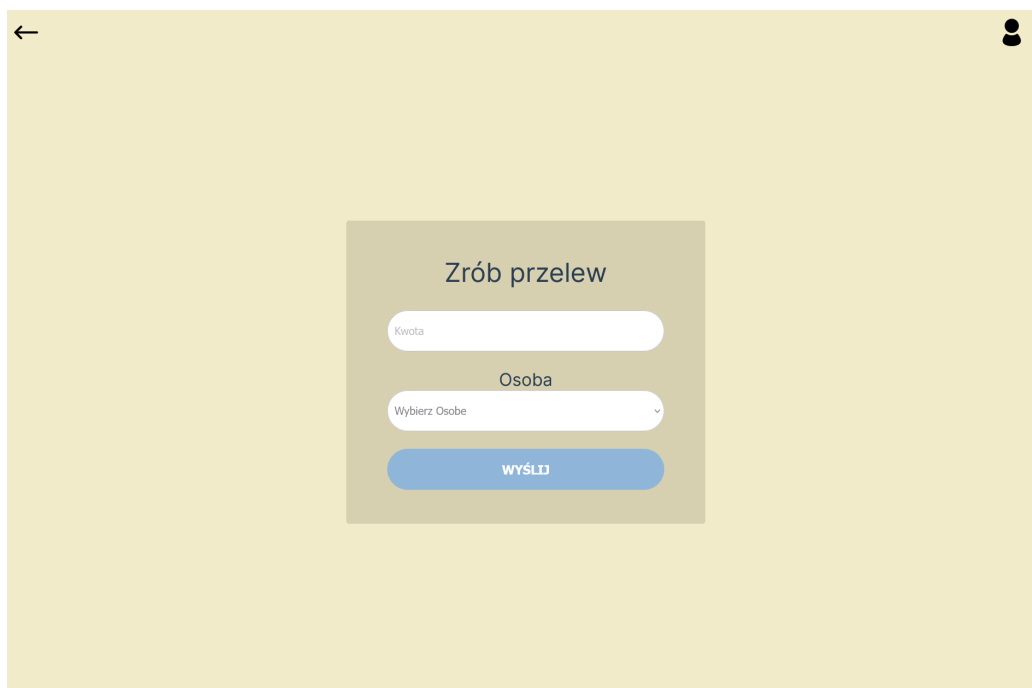
## 7.8.2 Dodaj wydatek



Rysunek 14: Zrzut ekranu dodawania wydatku.

W przypadku dodawania wydatku sytuacja jest analogiczna do dodawania przychodu.

## 7.9 Wykonaj przelew



Rysunek 15: Zrzut ekranu wykonywania przelewu.

Korzystający z aplikacji może wykonać przelew wewnętrzny pomiędzy kontami należącymi do tej samej grupy. Aby to zrobić, należy podać kwotę oraz wybrać profil docelowy.

## 8 Specyfikacja wewnętrzna

### 8.1 Wykorzystane technologie

System został wykonany w oparciu o technologię ASP.NET Web API. Wykorzystuje Entity Framework do komunikacji z bazą danych SQL Server. Dokumentacja API realizowana jest za pomocą frameworka Swagger.

## 8.2 Warstwy systemu

### 8.2.1 Warstwa bazy danych

Warstwa ta jest odpowiedzialna za dostęp i modyfikację danych w bazie danych. W niniejszej aplikacji użyto Entity Framework Core, który umożliwia korzystanie z modelu obiektów i ułatwia wykonywanie operacji na danych.

### 8.2.2 Warstwa logiki biznesowej

Warstwa ta zaimplementowana jest w serwisach aplikacji i odpowiada za przetwarzanie i filtrowanie danych zgodnie z założeniami biznesowymi aplikacji. W kodzie przykładu ta warstwa jest reprezentowana przez klasy serwisów **IAuthService**, **IAccountCreationService** i **IReportService**.

### 8.2.3 Warstwa wizualna

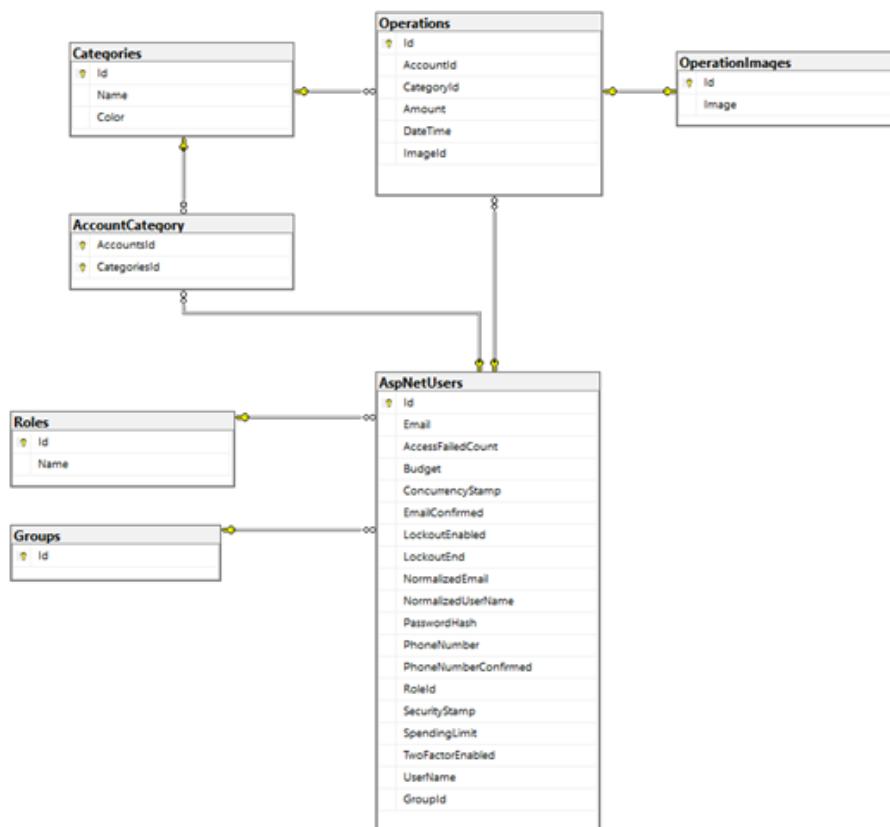
Warstwa wizualna odpowiada za prezentację danych w interfejsie użytkownika. W aplikacji jest reprezentowana przez generowanie punktów końcowych API zwracających kod HTML lub JSON i umożliwiających odwoływanie się do innych punktów końcowych. Warstwa wizualna korzysta z serwisów implementujących logikę biznesową.

### 8.2.4 Warstwa autoryzacji

Odpowiedzialna za zapewnienie mechanizmów uwierzytelniania i autoryzacji w systemie. W kodzie przykładu ta warstwa jest reprezentowana przez autoryzację ogólną JWT i autoryzację podziału dzielonych dla zasobów i tokenów uwierzytelniania.

## 9 Weryfikacja osiągniętych efektów względem założeń

### 9.1 Schemat zaimplementowanej bazy danych



Rysunek 16: Wykorzystany schemat bazy danych.

### 9.2 Analiza wymagań i ich zmian względem założeń

#### 9.2.1 Analiza wymagań funkcjonalnych

Końcowa wersja aplikacji nie zawiera funkcjonalności przechowywania skanów paragonów i faktur. Baza danych jest wyposażona w przechowywanie tych informacji, jednak brakuje logiki w systemie. Brakuje jej również prze-



chowywania informacji o dłużnikach. Funkcjonalność dodawania członków rodziny do konta jest rozwiązana w inny sposób, niż pierwotnie planowano. Konta łączone są w grupy, jednak każde konto posiada osobne dane logowania. Początkowe założenie opierało się na jednym koncie i zawierających się w nim profilach. Użycie osobnego hasła dla każdego konta jest lepszym rozwiązaniem. Zaprezentowana przez nas aplikacja nie jest wyposażona w zarządzanie operacjami na koncie bankowym. Systemowi brakuje dodanie cyklicznych/stałych operacji. W przypadku analizy MoSCoW i funkcjonalności Won't zawarto w aplikacji powiadomienia o przekroczonym budżecie dla konta dziecka, wynikające z jego typu i ograniczonego budżetu.

### 9.2.2 Analiza wymagań niefunkcjonalnych

W przypadku wymagań niefunkcjonalnych nie zostało zaimplementowane tworzenie okresowych kopii zapasowych. System nie posiada również możliwości przypominania hasła użytkownika. Potwierdzenie rejestracji e-mailem zostało wyłączone.

## 10 Testowanie i uruchamianie

System testowano w miarę budowania aplikacji oraz dodawania nowych funkcjonalności. Ze względu na ograniczony czas realizacji projektu nie wykonano testów jednostkowych na systemie. Pomimo tego aplikacja pracuje poprawnie w „codziennym” użytkowaniu. Nie zaobserwowano oczywistych problemów podczas typowego korzystania. W miarę prac implementacyjnych pojawiały się błędy typu semantycznego, które na bieżąco były rozwiązywane.

## 11 Uwagi i wnioski

Projekt realizowaliśmy w ciągu kilku miesięcy semestru akademickiego. W praktyce, posiadając wiele innych obowiązków, stwierdzamy, że czas możliwy na realizację projektu był zbyt krótki, by zapewnić pełną funkcjonalność aplikacji oraz wykonać odpowiednie testy systemu. Niewątpliwie realizacja

projektu pomogła nam w praktycznym zrozumieniu wykorzystanych przez nas technologii i stanowi cenne źródło wiedzy, którą najprawdopodobniej wykorzystamy podczas pracy w projektach komercyjnych.