Wojciech Krakowski

Portfolio testowania z użyciem SQL

CEL PROJEKTU

Celem projektu jest przetestowanie aplikacji NaCoPoszło na podstawie bazy danych SQL. W ramach portfolio przedstawiam dwa przypadki testowe.
Przedstawiona struktura danych jest przykładowa.

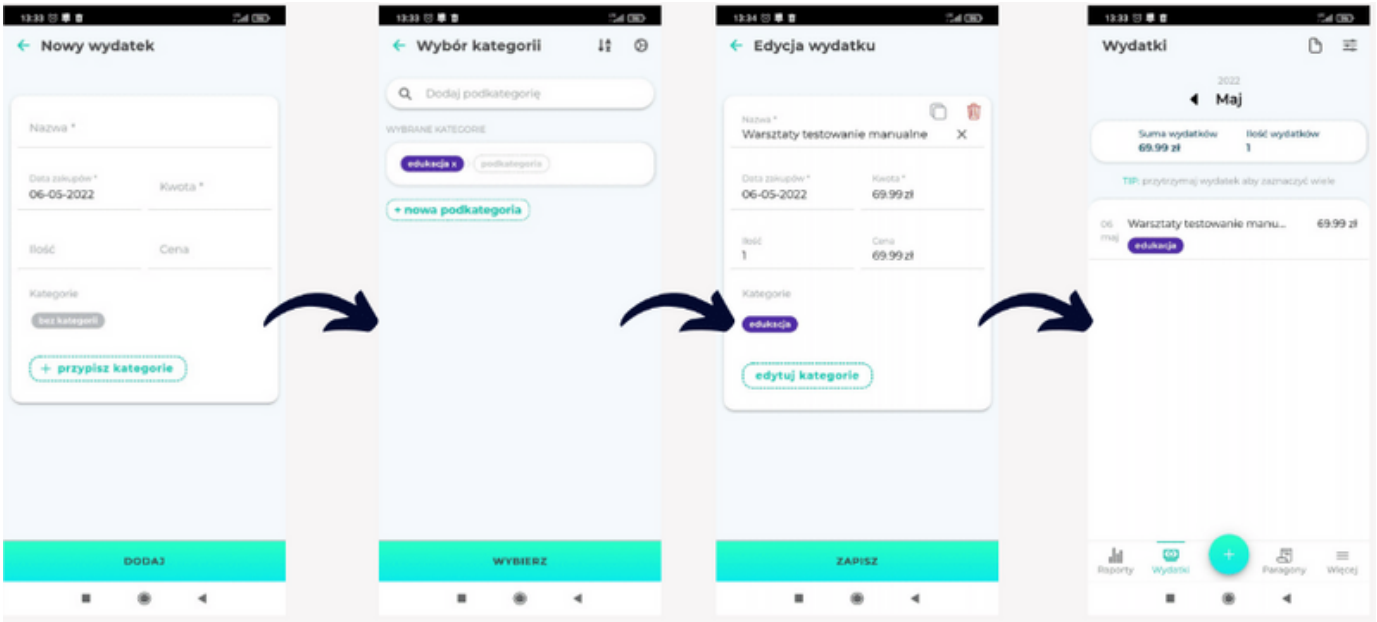
Link do aplikacji: **LINK**

ZAŁOŻENIA APLIKACJI

Aplikacja umożliwia planowanie budżetu domowego i śledzenie wydatków. Korzystając z aplikacji, użytkownik może skanować paragony w celu odczytania wydatków, dodawać, usuwać, modyfikować wydatki ręcznie, oraz filtrować i wyświetlać zapisane wydatki.

TESTOWANY FRAGMENT APLIKACJI: 'Ręczne dodanie oraz modyfikacja wydatku'

W ramach testów sprawdzę możliwość dodawania wydatku ręcznie, przypisanie kategorii, jego edycję oraz usunięcie. Wszystkie dane zapisywane są w bazie danych SQL. W ramach opisanych poniżej przypadków testowych skupię się głównie na sprawdzeniu zapisu do bazy danych.



JAK PRZYGOTOWUJEP RZYPADKI TESTOWE

Przypadki testowe rozpatruje w 4 kategoriach:

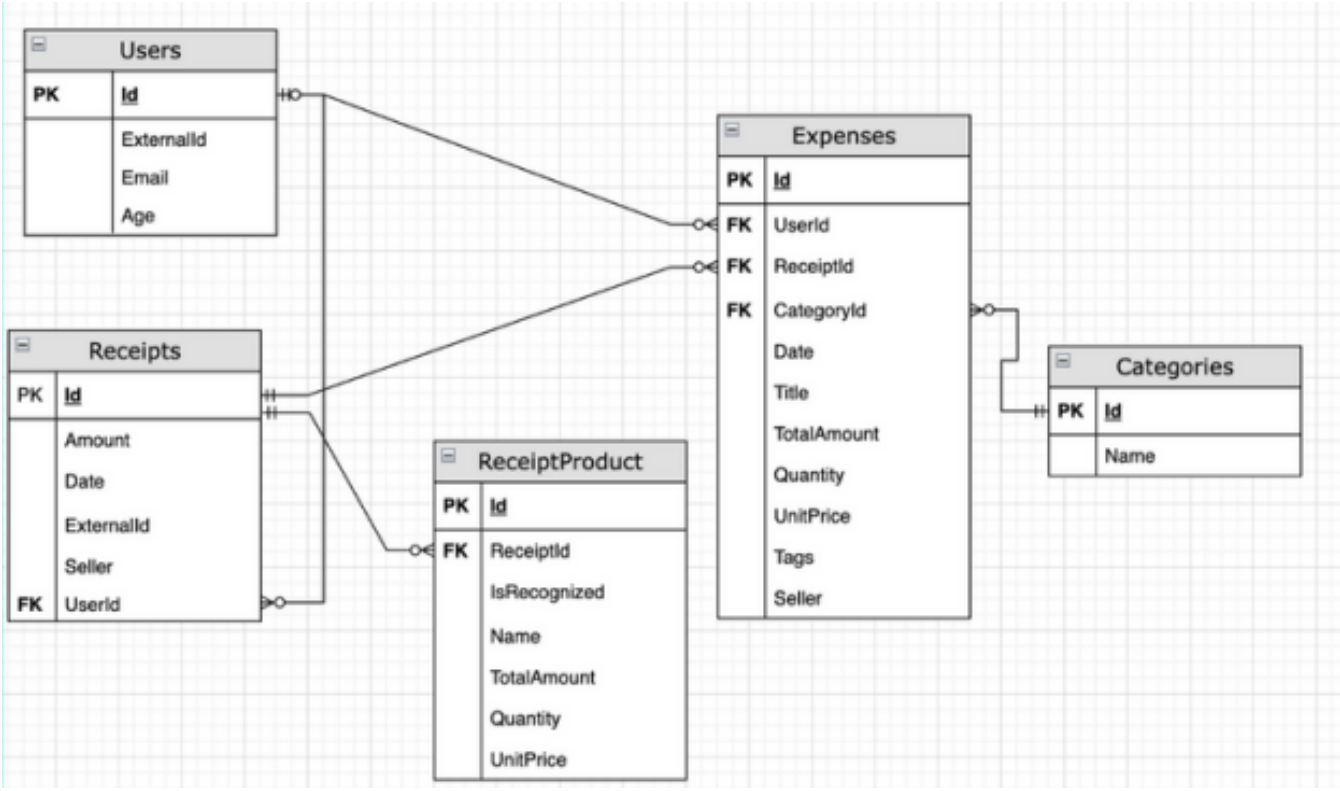
- **funkcjonalne (podstawowa funkcjonalność opisana w dokumentacji)** - w ramach testów funkcjonalnych sprawdzam wszystkie możliwe scenariusze opisane w dokumentacji oraz te, które nie są oczywiste. Sprawdzam wszystkie możliwe poprawne kombinacje danych wejściowych oraz te które mogą spowodować błędy jak puste znaki, znaki specjalne itd. [patrz TestCase1]

- **użytkowe/wizualne** (na bazie makiet UX/UI) - w ramach testów UX/UI sprawdzam czy aplikacja jest zgodna z przygotowanymi makietami, jak również czy interfejs użytkownika wygląda estetycznie i czy jest użyteczny. [patrz TestCase2]
- **techniczne (zmiany w kodzie HTML, błędy HTTP z serwera itd)** - w ramach testów technicznych jeżeli to możliwe staram się modyfikować kod HTML, sprawdzam błędy zwrócone z serwera, symuluje zwolnienie sieci internetowej w przeglądarce itd. [patrz TestCase3] wydajnościowe (jeżeli to możliwe)

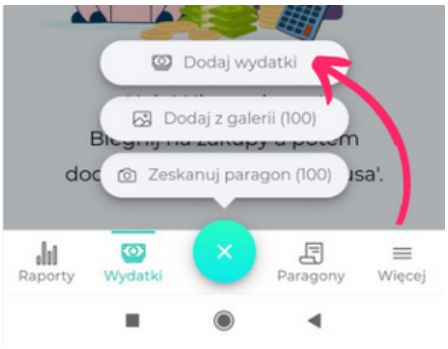
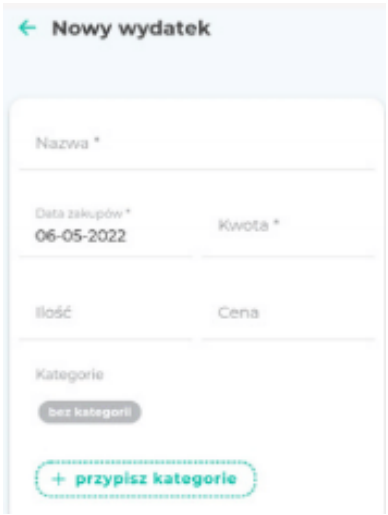
Zazwyczaj przypadki testowe przygotowuje według schematu:

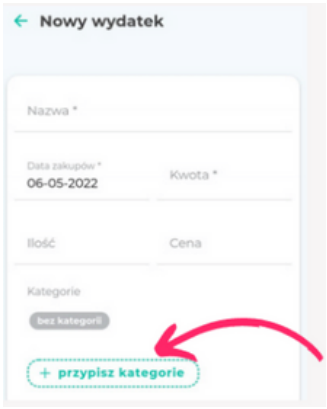
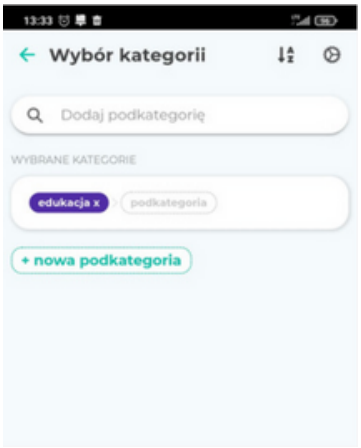
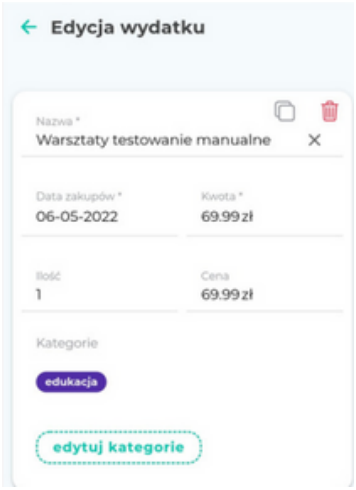
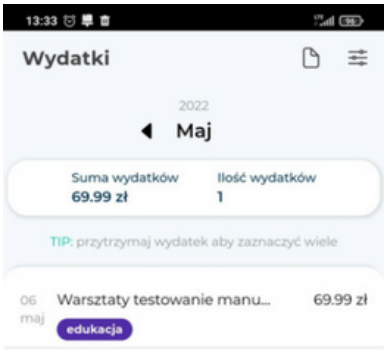
Baza danych

Baza danych aplikacji zawiera następującą strukturę:



TEST CASE 1 - Przypadek podstawowy: Dodanie wydatku **POPRAWNY**

ACTION	INPUT	OUTPUT
1.Klikam przycisk '+' na głównym ekranie, a następnie wybieram ręczne dodatnie wydatku 		Ukazuje się widok dodania 

2. Klikam 'przypisz kategorię', a następnie ją tworzę. 	**Nazwa kateogrii:** edukacja	Kategoria 'edukacja' została utworzona 
6. Wypełniam wydatek danymi 	**Nazwa:** Warsztaty testowania manualnego **Kwota:** 69.99, **Cena:** 69.99, **Ilość:** 1, **Data:** 06-05-2022	Wydatek został dodany 

Sprawdzenie bazy danych:

Sprawdzam, czy wydatek został poprawnie dodany do tabeli 'Expenses' i porównuje wszystkie wartości.

Query:

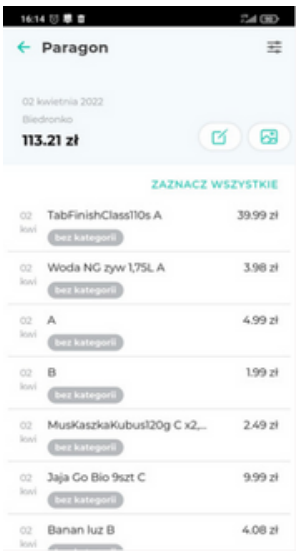
```
SELECT * FROM Expenses
SELECT
e.date,
e.receiptId,
e.title,
e.totalAmount,
e.quantity,
e.unitprice,
e.seller,
c.name,
u.email
FROM Expenses e
LEFT JOIN Categories c ON e.categoryid = e.categoryid
JOIN Users u ON e.userid = u.id
WHERE u.email = 'wojtas@email.pl'
```

Wynik:

Wpis 'receiptid' jest null ponieważ wydatek nie pochodzi z paragonu. Kategoria również została poprawnie utworzona. Wydatek został poprawnie zapisany.

date	receiptId	title	totalAmo...	qua...	unitprice	seller	name	email
2022-05-06	NULL	Test	69.99	1	69.99	NULL	Edukacja	wojtas@email.pl

TEST CASE 2 - Zmiana daty wszystkich wydatków odczytanych z paragonu

ACTION	INPUT	OUTPUT
1. Klikam przycisk '+' na głównym ekranie i skanuje paragon. 		Ukazuje się widok dodanego paragonu 

Sprawdzenie bazy danych:

Sprawdzam, czy wydatki zostały poprawnie dodane do tabeli 'Expenses' i porównuje wszystkie wartości. Sprawdzam również, czy został dodany do bazy danych paragon do tabeli 'Receipts'

Query:

```
SELECT *
FROM Receipts r
JOIN Users u ON r.UserId = u.id
WHERE u.email = 'wojtas@email.pl'
```

Wynik:

Paragon został poprawnie dodany do bazy danych, natomiast nie zgadza się odczytana data. Powinna być 02.11.2019

Id	Amount	Date	Extern...	Seller	UserId	Id	ExternalId	Email	Age
1	113.21	2022-04-02	f700f55...	Biedronka	1	1	1	wojtas@em...	29

Sprawdzam powiązane wydatki.

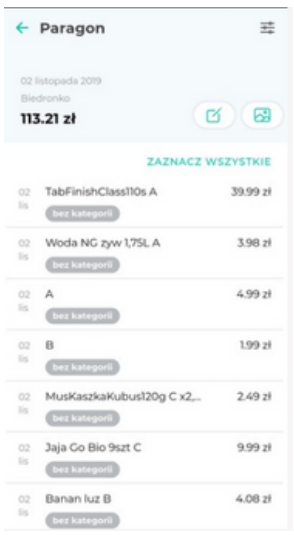
Query:

```
SELECT
e.date,
e.title,
e.totalamount,
e.quantity,
e.unitprice,
e.seller,
c.name,
u.email,
r.amount AS ReceiptAmount,
r.Seller AS ReceiptSeller
FROM Expenses e
LEFT JOIN Categories c ON e.categoryid = e.categoryid
JOIN Users u ON e.userid = u.id
JOIN Receipts r ON e.receiptir = r.id
WHERE u.email = 'wojtas@email.pl' AND e.receiptid = 1
```

Wynik:

Również wydatki mają źle przypisaną datę. Dodatkowo odczytały się błędne wpisy jak 'A'.

date	title	totalamount	quantity	unitprice	seller	name	email	ReceiptA...
2022-04-02	TabFinishClass100s A	9.99	1	9.99	Biedronka	NULL	wojtas@emai...	113.21
2022-04-02	Woda NG zyw 1,75L A	3.98	2	1.99	Biedronka	NULL	wojtas@emai...	113.21
2022-04-02	A	4.99	1	4.99	Biedronka	NULL	wojtas@emai...	113.21

2. Przechodzę w aplikacji do modyfikacji daty i wybieram poprawną datę 	**Data:** 02.11.2019	Ukazuję się widok poprawnego paragonu. 

Sprawdzenie bazy danych

Sprawdzam czy data została poprawiona w bazie danych dla encji 'Receipts' oraz czy data wszystkich wydatków powiązanych z paragonem została zmieniona.

Query:

```
SELECT
e.date,
e.title,
e.totalamount,
e.quantity,
e.unitprice,
e.seller,
c.name,
u.email,
r.Amount AS ReceiptAmount,
r.Seller AS ReceiptSeller
FROM Expenses e
LEFT JOIN Categories c ON e.categoryid = e.categoryid
JOIN Users u ON e.userid = u.id
Join Receipts r ON e.receiptid = r.id
WHERE u.email = 'wojtas@email.pl' AND e.receiptid = 1
```

```
SELECT *
FROM Receipts r
JOIN Users u ON r.UserId = u.id
WHERE u.email = 'wojtas@email.com'
```

Wynik:

Paragon został poprawiony

id	amount	date	extern...	seller	us...	id	extern...	email	age
1	113.21	2019-11-02	f700d55...	Biedronka	1	1	f0f63f9...	wojtas@em...	30

2019-11-02	TabFinishClass100...	9.99	1	9.99	Biedronka	NULL	wojtas@em...	113.21	Biec
2019-11-02	Woda NG zyw 1,75...	3.98	2	1.99	Biedronka	NULL	wojtas@em...	113.21	Biec
2019-11-02	A	4.99	1	4.99	Biedronka	NULL	wojtas@em...	113.21	Biec

Sprawdzam, czy zostały zapisane poprawnie metadane generowane podczas odczytu z paragonu, które nie są wyświetlane użytkownikowi.

Query:

```
SELECT *
FROM ReceiptProduct rp
JOIN Receipts r ON rp.receiptid = r.id
WHERE r.Id = 1
```

Wynik:

Dane zostały poprawnie zapisane.

id	receiptid	isrecon...	name	totalam...	quantity	unitprice	id	amount	date	exter...	seller
1	1	1	TabFinishClass100...	9.99	1	9.99	1	113.21	2019-11-02	f700d...	Biedronka
2	1	1	Woda NG zyw 1,75...	3.98	2	1.99	1	113.21	2019-11-02	f700d...	Biedronka
3	1	1	A	4.99	1	4.99	1	113.21	2019-11-02	f700d...	Biedronka

3. Zgłaszam błąd dla programisty, aby mógł sprawdzić paragon oraz prześledzić dlaczego nastąpiło błędne rozpoznanie