

**KOLEGIUM INFORMATYKI STOSOWANEJ**

**Kierunek: Informatyka**

**Specjalność: Programowanie**

Łukasz Kuznecki

60060

***Aplikacja e-commerce artykułów ozdobnych***

Promotor: Dr inż. Teresa Mroczek

**PRACA DYPLOMOWA INŻYNIERSKA**

**Rzeszów 2023**

**Spis treści**

[Wstęp 4](#_Toc135053442)

[1. Analiza rynku 6](#_Toc135053443)

[1.1. Aplikacje internetowe na rynku e-commerce 6](#_Toc135053444)

[1.2. Pozycjonowanie 6](#_Toc135053445)

[1.3. Funkcjonalność 7](#_Toc135053446)

[1.4. Responsywność 7](#_Toc135053447)

[1.5. Analiza istniejących rozwiązań 7](#_Toc135053448)

[2. Pozyskiwanie danych od użytkowników 9](#_Toc135053449)

[2.1. Analiza ankiety i profil respondenta 9](#_Toc135053450)

[2.2. Analiza badań ankietowanych 9](#_Toc135053451)

[2.3. Wymagania funkcjonalne 16](#_Toc135053452)

[2.4. Wymagania niefunkcjonalne 18](#_Toc135053453)

[3. Zastosowane narzędzia i technologie 19](#_Toc135053454)

[3.1. REST API 19](#_Toc135053455)

[3.2. Bazy Danych 19](#_Toc135053456)

[3.3. Narzędzia 20](#_Toc135053457)

[4. Architektura 23](#_Toc135053458)

[5. Implementacja 24](#_Toc135053459)

[5.1 Wzorce Projektowe 24](#_Toc135053460)

[5.2 Serwisy główne 25](#_Toc135053461)

[5.3 Serwis zamówień 26](#_Toc135053462)

[5.4 Serwis płatności 26](#_Toc135053463)

[5.5 Serwis identyfikacji 27](#_Toc135053464)

[6. Prezentacja warstwy użytkowej aplikacji 29](#_Toc135053465)

[7. Testy użyteczności 40](#_Toc135053466)

[Zakończenie 46](#_Toc135053467)

[Literatura 47](#_Toc135053468)

[Streszczenie 48](#_Toc135053469)

# **Wstęp**

Rozwój sieci internetowej na przestrzeni lat sprawił, że handel detaliczny przechodzi transformację cyfrową. Coraz więcej sklepów fizycznych przenosi się lub poszerza swoją działalność o medium cyfrowe wchodząc tym samym w rynek e-Commerce. Kupując online klient wciąż sam odbiera towar jednak proces wyboru produktu, metody dostawy i jej miejsca oraz płatności dokonuje kilkoma szybkimi etapami bez potrzeby odwiedzenia sklepu. Zalety dokonywania zakupów online to przede wszystkim wygoda i oszczędność czasu jaki klient kupując fizycznie musiałby poświęcić na odwiedzenie sklepu bez gwarancji znalezienia szukanego artykułu, co więcej wybór jakim klient dysponuje nie ogranicza się już do pobliskich sklepów a jest poszerzony na cały kraj i świat. Jednak idea robienia zakupów przez Internet pomimo wielu zalet porusza aspekty bezpieczeństwa, które na uwadze powinien mieć zarówno klient jak i developer systemu.

Obecnie rynek sprzedaży artykułów ozdobnych takich jak: zaproszenia, kartki, skrzynki, koperty, a także ozdoby świąteczne, pudełka, ramki na zdjęcia są obecnie we wczesnym stadium rozwoju. Dostępne na rynku internetowym produkty są tworzone z myślą o ilości nad jakością, co sprawia, że osoby poszukujące spersonalizowanego artykułu mogą spodziewać się bardzo wysokich cen i w wyniku zrezygnować z zakupu. Jedną z przyczyn tego zjawiska jest indywidualizm artystów oferujących usługi w tym zakresie w wyniku czego działalności jakie prowadzą mają charakter lokalny, ponieważ całym procesem od produkcji do sprzedaży zajmują się sami.

Celem pracy było zaprojektowanie i implementacja aplikacji internetowej, która umożliwia twórcy na sprzedaż produktów za pośrednictwem Internetu, co pozwala na zwiększenie sprzedaży, oszczędność kosztów i dotarcie do szerszego grona potencjalnych klientów. Względem użytkownika jest to zapewnienie intuicyjnego interfejsu użytkownika, w którym będzie mógł sprawnie przechodzić między etapami zakupu lub znaleźć informacje kontaktowe przedsiębiorcy.

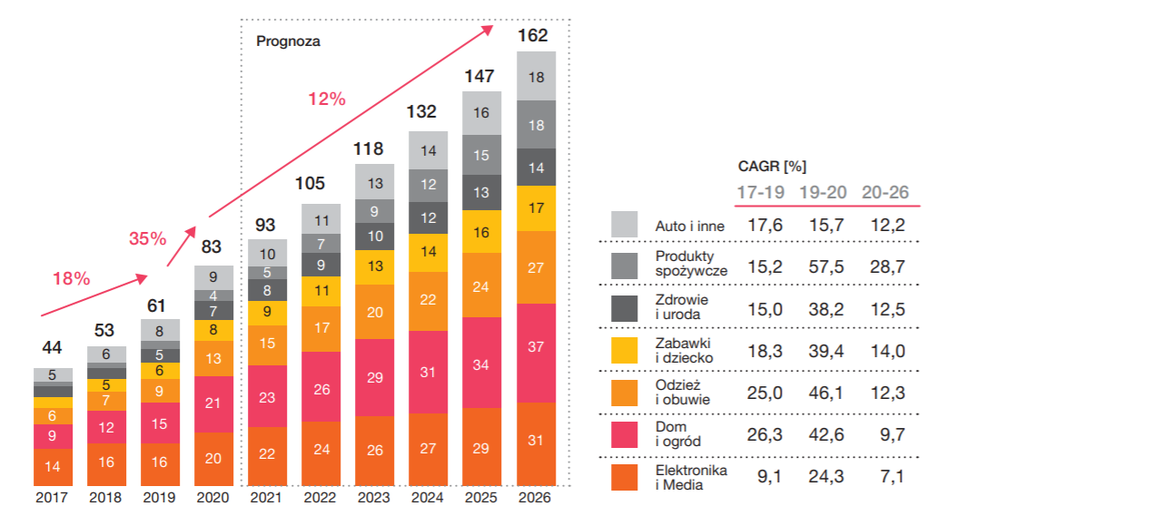
Aplikacja składa się z czterech części. Pierwszą z nich jest baza wiedzy, gdzie zdefiniowane są encje, na podstawie których stworzona została baza danych oraz specyfikacje. Na jej podstawie stworzone zostały bazy danych: jedna jako kontekst sklepu i druga która przechowuje informacje o tożsamości użytkowników. Następną częścią jest API, w której zawarta jest logika działania aplikacji i zasady komunikacji z bazami danych. Ostatnią częścią jest interfejs użytkownika, który został zaimplementowany w technologii Angular, skąd poprzez zapytania http kierowane do API otrzymywane są zasoby z baz danych [1-4].

W pierwszym rozdziale zawarta została analiza rynku pod względem istniejących już sklepów e-commerce, na co składa się ogólne badanie trendów sprzedaży dóbr przez Internet na przestrzeni lat oraz prognozy na przyszłość w tym zakresie. Kolejny podrozdział przybliża pojęcie pozycjonowania i co składa się na zwiększenie jego widoczności w wynikach wyszukiwania. Następnie nakreślony został ogólny zakres funkcjonalności, jakie sklepy internetowe obecnie zapewniają oraz jak ułatwiają one znalezienie żądanego produktu i przejście przez etapy dokonania zakupu dla klienta. Czwartym aspektem jest responsywność, czyli cecha strony internetowej, która pozwala na wyświetlanie jej na innych urządzeniach najczęściej mobilnych. W podrozdziale piątym został opisany system powiadomień który pozwala stronie wysłać wiadomość zwrotną do klienta o rezultacie dokonanego zakupu lub rejestracji. Ostatnią kwestią poruszoną w pierwszym rozdziale jest analiza istniejących rozwiązań rynkowych które oferują zbliżony zakres usług dla tego samego zakresu odbiorców. Drugi rozdział obejmuje pozyskiwanie danych od użytkowników na co składa się: ankieta i jej analiza w celu stworzenia profilu respondenta, następnie analiza badań ankietowych i na ich podstawie sprecyzowanie wymagań funkcjonalnych i niefunkcjonalnych aplikacji. Trzeci rozdział przybliża wybór środowiska i rozwiązań oraz architektury aplikacji. Czwarty rozdział to prezentacja wizualnej warstwy użytkowej, gdzie dla każdego widoku napisana została instrukcja korzystania z funkcjonalności jakie są w nich zawarte. Kolejny rozdział odnosi się do testów użytkowych jakie zostały przeprowadzone oraz ich zakres i cel wykonania, a także rezultaty jakie one przyniosły w  kontekście celów pracy. W zakończeniu zostało dokonane podsumowanie wykonanych prac w odniesieniu do projektu i implementacji aplikacji.

# **Analiza rynku**

W pierwszym rozdziale zostanie przedstawiona zostanie ogólna analiza rynkowa sklepów internetowych z wyszczególnieniem aspektów: pozycjonowania, funkcjonalności, responsywności i powiadomień. Następnie wykonane zostało badanie konkurencyjnych stron pod kątem wymienionych atrybutów i na jego podstawie rozdział zakończony jest wyciągnięciem wniosków odnośnie dalszego projektu aplikacji.

# **Aplikacje internetowe na rynku e-commerce**

Aplikacje internetowe, które służą jako platforma sprzedaży i reklama tworzona jest z zamysłem ciągłego pozyskiwania klientów, więc największą wagę przykłada się do estetyki strony. „Konsumenci kupują oczami” ma przełożenie również na handel internetowy, więc pierwsze wrażenie po wejściu na stronę potrafi przyciągnąć lub odstraszyć potencjalnego klienta. Aktualne rozwiązania prezentują intuicyjny interfejs oraz estetyczną stroną indeksową w sposób zachęcający użytkownika do przeglądnięcia oferty. Według danych z Rys. 1. Wartość sprzedaży dóbr online rośnie z roku na rok, a prognozy na przyszłość zapowiadają kontynuacje trendu wzrostowego. Własna aplikacja internetowa to niewątpliwie rozwiązanie, z którego powinno korzystać każde przedsiębiorstwo chcące zwiększyć sprzedaż swoich produktów[5]. 

Rys. 1 Wartość brutto sprzedaży dóbr online w Polsce, 2017-2026 [mld PLN]

Źródło: https://www.pwc.pl/pl/media/2021-02-09-analiza-pwc-prognoza-rozwoju-rynku-ecommerce-w-polsce.html, z dnia 20.01.2023

# **Pozycjonowanie**

Obecnie znaczna część serwisów internetowych przyciąga odwiedzającego wizualnie estetycznym interfejsem użytkownika, który pomimo wielu wygodnych funkcji zachowuje łatwość i intuicyjność obsługi. Interfejs użytkownika docelowo ma ułatwiać użytkownikowi sprawne przechodzenie od przeglądania oferty przez bliższemu przyglądaniu się artykułom aż do finalizacji zakupu. Cały ten proces nie powinien wymagać od klienta więcej niż podstawowej wiedzy z zakresu podstawowej samoobsługi sklepu internetowego. Kluczową rolę odgrywa także pasek nawigacyjny, który w większości przypadków umiejscowiony jest górnej środkowej części strony indeksowej. Ważne jest odpowiednie opisanie go, tak aby kategorie pokrywały się z wyszukiwanymi frazami przez użytkownika, w ten sposób aplikacja internetowa rośnie w pozycjach wyszukiwania co powinno być celem przedsiębiorców prowadzących sklep e-commerce.

# **Funkcjonalność**

Zakres funkcjonalności detalicznych sklepów internetowych ponad standardową możliwość dokonywania zakupu oferuje także możliwość przesyłanie informacji zwrotnych do firmy lub przedsiębiorcy. Recenzja produktów przy pomocy komentarzy tekstowych lub paska z wyborem stopni satysfakcji klienta jest coraz częściej spotykanym rozwiązaniem w sferze komunikacji z klientem przez serwisy internetowe. Opinie te są wartościowe zarówno dla innych osób odwiedzających sklep którzy na ich podstawie zdecydują się na kupno lub nie, oraz dla firmy prowadzącej tę stronę dla których informacje zwrotne pomogą w poprawieniu jakości świadczonych usług. W ramach usług sprzedaży akcesoriów ozdobnych dużą wagę mają kategorie, które mają za zadanie ułatwić klientowi znalezienie pożądanego produktu, a które powinny się znaleźć w pasku nawigacyjnym sklepu internetowego. W sferze metod płatności gotówka ustępuje szybkim przelewom jako preferowana metoda finalizacji transakcji. Istotne jest, aby klient nie był i nie czuł się ograniczany: więcej metod płatności poprawia komfort klienta co zwiększa szanse na dokonanie zakupu przez klienta. Szeroki wybór metody dostawy jest również bardzo ważny dla klienta i podobnie jak z płatnościami większe spektrum wyboru będzie zachęcać potencjalnych konsumentów do zakupu właśnie w tym sklepie.

# **Responsywność**

Responsywność jest cechą, która odgrywa ważną role w komunikacji klienta z aplikacją. Większość dominujących serwisów sklepów internetowych posiada aplikację mobilną dostępna w Google Play. Jest to wskaźnikiem dla mniejszych i większych konkurentów na rynku jak ważna jest sfera mobilna dla aplikacji internetowych. Telefon stanowi ważne medium, przez które klienci dokonują zakupów, ważne jest więc aby internetowa aplikacja sklepowa posiadała swój odpowiednik wersji mobilnej, gdzie rozmiar strony dopasowany jest do urządzenia mobilnego użytkownika.

# **Analiza istniejących rozwiązań**

W zakresie rozwiązań istniejących na rynku analizie poddano trzy Polskie firmy, które oferują możliwość zamówienia artykułów dostępnych z poziomu sklepu internetowego. Badane firmy to Eko-deco, Decobazar oraz Pakamera [6-8]. Sklepy te zostały wybrane ze względu na zakres oferowanych produktów tj. domowe artykuły ozdobne.

Wszystkie trzy serwisy posiadają pasek nawigacyjny umiejscowiony w taki sam sposób jak pasek poziomy w górnej części strony, gdzie znajdują się kategorie dzielące produkty na grupy. Dwa na trzy serwisy posiadają zaimplementowany suwak, który przy pomocy automatycznego przewijania pokazuje produkty nowe, sezonalne lub przecenione. Jak wiadomo animacja szczególnie ruchu działa bardzo przykuwająco na ludzki wzrok, co stwarza lepsze okazje do przyciągnięcia klienta. Każda z aplikacji posiada rozwiązanie pozwalające na filtrowanie konkretnych produktów przy pomocy list rozwijanych lub pól tekstowych. Kolejną cechą podobną jest zakładka z wyselekcjonowanymi produktami które nadają się na prezent co jest istotne, gdyż klienci często przeglądają tego typu artykuły szukając inspiracji do kupienia prezentu. Wszystkie trzy badane strony posiadają na dole strony informacje o firmie oraz dane kontaktowe, informacje o akceptowalnych metodach płatności a także informacje o ochronie danych osobowych. Przy rejestracji we wszystkich trzech rozwiązaniach nie występuje konieczność potwierdzania utworzenia konta co daje większy komfort klientowi kosztem bezpieczeństwa i integralności jego danych. W sklepie Ecodeco i Decobazar nie występuje potrzeba rejestracji, aby dokonać zakupu co daje im przewagę nad sklepem pakamera pod względem wygody klienta. Każdy z serwisów stara się zachęcić użytkownika przy rejestracji i zakupie do wyrażenia zgody na wysyłanie powiadomień o ofertach.

Podsumowując rozmyślnie zbudowana aplikacja jest na obecnym rynku kluczem do pozyskiwania nowych klientów jak i zatrzymywania stałych. Strona internetowa jest wizytówką estetyczną firmy jej prowadzącą, a sprawne przechodzenie pomiędzy etapami zakupu jest kluczowym elementem budującym zaufanie pomiędzy producentem a konsumentem. Należy pamiętać, że komfort ma czuć zarówno nowy klient jak i stały, ze szczególnym uwzględnieniem osób starszych które mają mniejsze obycie w internetowych zakupach. Interfejs powinien być implementowany z myślą o każdej grupie wiekowej potencjalnych użytkowników.

# **Pozyskiwanie danych od użytkowników**

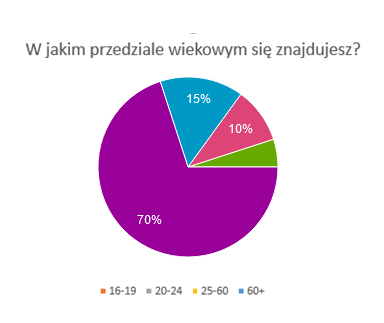
Następnym korkiem po wykonaniu analizy istniejących już rozwiązań jest analiza danych pochodzących bezpośrednio od potencjalnych użytkowników proponowanego przeze mnie rozwiązania. Pozwoli to na dokładniejsze dobranie rozwiązania do potrzeb użytkowników. W celu pozyskania informacji na temat ich preferencji wykonana została ankieta, na podstawie której będzie można określić zakres oczekiwań potencjalnego klienta wobec tworzonej aplikacji. W scenariusz ankiety wchodzi 13 pytań jednokrotnego wyboru i jedno opcjonalne pytanie otwarte. Pytania zostały dobrane w taki sposób, aby poruszyć nie tylko kwestie techniczne aplikacji, ale także możliwych korekcji biznesowych w ramach planowanych promocji lub oferowanych usług.

# **Analiza ankiety i profil respondenta**

Badania zostały przeprowadzone na przełomie roku 2022/2023 za pośrednictwem internetowej ankiety, którą stworzyłem za pomocą portalu Google [9]. Przypisany do niej link przesłałem członkom rodziny, aby oni przekazali ją swoim znajomym w pracy, tak aby docelowym odbiorcą ankiety był potencjalny klient sklepów internetowych. Każdy z respondentów wypełniał ankietę samodzielnie tj. bez zadawania pytań przez ankietera. Ankieta miała za cel pozyskanie danych od użytkowników w celu sprecyzowania wymagań funkcjonalnych i poza funkcjonalnych, aby na ich podstawie stworzyć produkt najbardziej przyjazny i wygodny dla użytkowników.

# **Analiza badań ankietowanych**

W badaniu wzięło udział dokładnie 30 osób, które podzieliłem na cztery grupy wiekowe: od 16 do 19 lat (osoby, które nie rozpoczęły studiów lub pracy), 20 do 24 lat (studenci), 25 do 60 lat (Osoby zatrudnione komercyjnie) i osoby powyżej 60 roku życia, które posiadają najmniej obycia z środowiskiem Internetu. Poniższy wykres przedstawia udziały w formie procentowej czterech badanych grup respondentów.



Wykres 1. Podział wiekowy respondentów

Źródło: badania własne

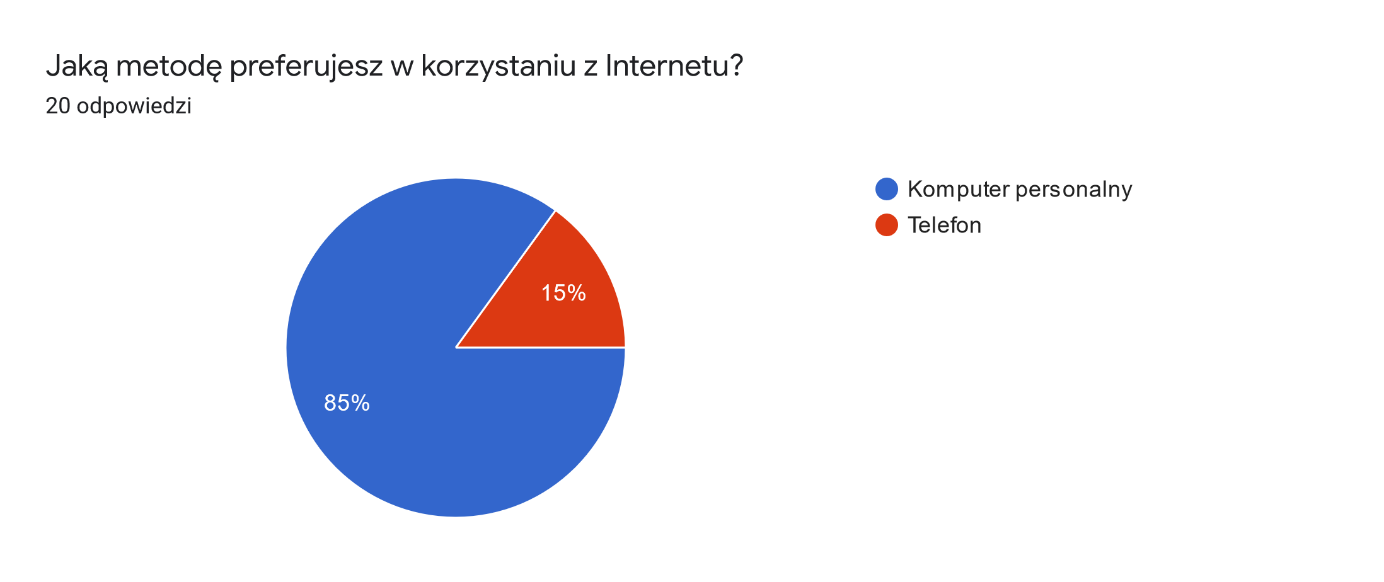
Znaczna większość respondentów znajduje się w przedziale wiekowym pomiędzy 25 a 60 rokiem życia, ponieważ to właśnie do nich głównie kierowałem ankietę poprzez zakłady pracy rodziny i znajomych. Najmniej liczną grupy stanowili uczniowie, którzy nie rozpoczęli jeszcze studiów (5%), i studenci (2%). Następnie podzieliłem ankietowanych ze względu na występowanie problem u z wyborem prezentu okazyjnego.



Wykres 2. Brak pomysłu na prezent respondentów

Źródło: badania własne

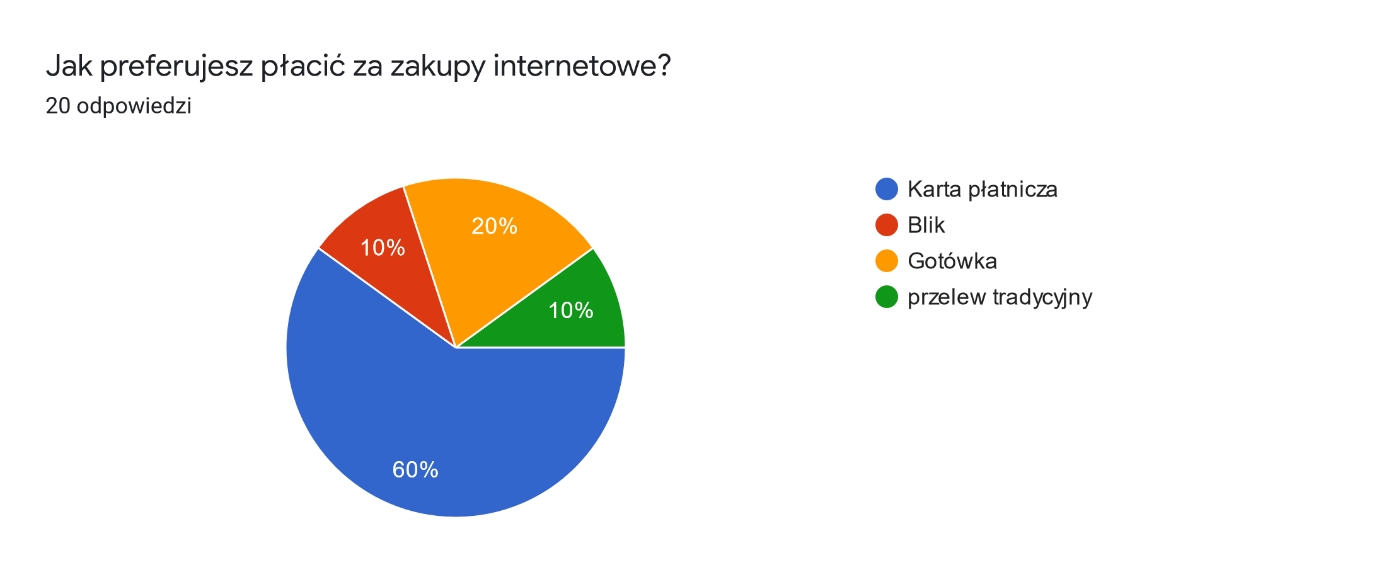
Wykres 2 pokazuje, że zdecydowana (65%) respondentów spotyka się z problemem konceptu na prezent okazyjny, co informuje, że istnieje szansa przyjęcia się na rynku sklepu internetowego z gotowymi pomysłami i wykonaniem prezentu. Odpowiedź „nie” w tym pytaniu nie oznacza braku potencjalnego zainteresowania sklepem, a ludzi zdecydowanych którzy będą szukać konkretnego produktu.



Wykres 3. Metody korzystania z Internetu respondentów.

Źródło: badania własne

Zdecydowana większość ankietowanych preferuje korzystanie z Internetu poprzez komputer personalny (85%). Jest to informacją dla mnie, że aplikacja internetowa www jest kierunkiem, którym powinienem się głównie kierować. Biorąc pod uwagę wiek większości respondentów wykonywanie zakupów przez telefon nie zyskało jeszcze dużej popularności w grupie osób ze stałym zatrudnieniem.



Wykres 4. Preferowane metody płatności za zakupy internetowe respondentów

Źródło: badania własne

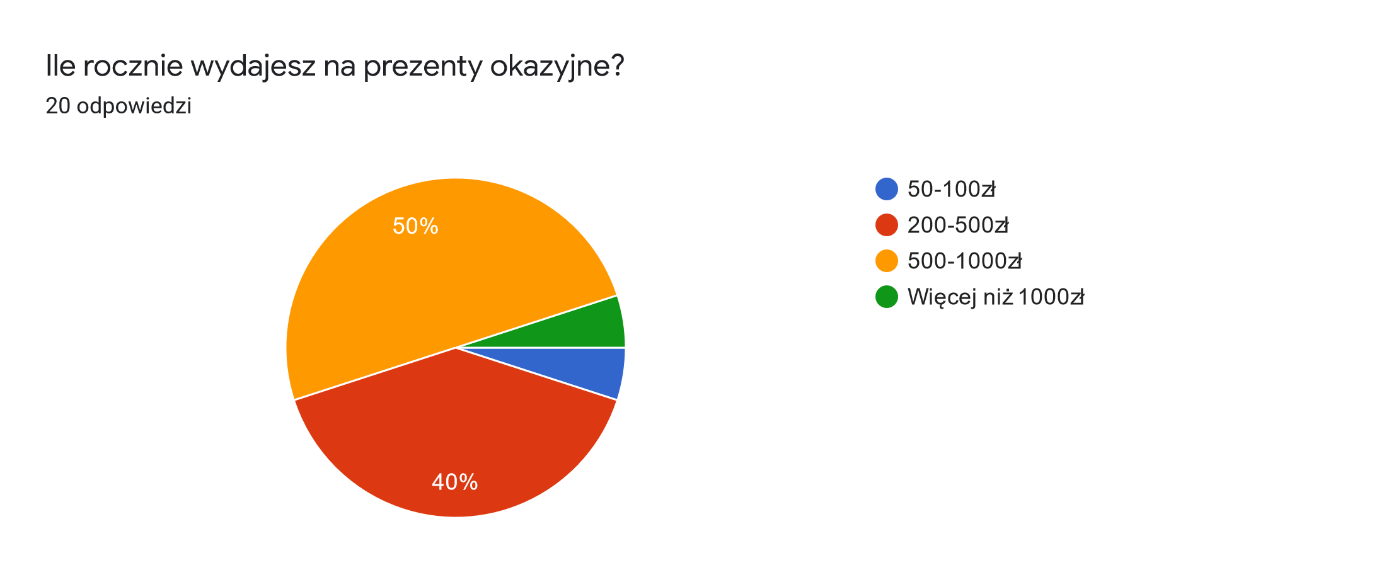
Według Wykresu 4 najbardziej popularną metodą płatności jest karta płatnicza (60%). Płatność Blikiem,  czyli mobilnym transferem pieniędzy zdecydowało się 10% ankietowanych prawdopodobnie byli to respondenci z grupy studentów/uczniów, gdyż to właśnie statystycznie młodzi mają największy udział w korzystaniu z urządzeń mobilnych. Z tego wykresu wynika, że możliwość wykonania płatności kartą jest absolutnie konieczna, ale klient nie powinien być jednak ograniczony do jednej możliwości i należy mu zapewnić dostosowany wybór metod płatności.



Wykres 5. Częstotliwość dokonywania zakupów przez Internet respondenta

Źródło: badania własne

Znaczna większość respondentów dokonuje zakupów drogą internetową raz na około dwa lub trzy miesiące (65%). Odpowiedzi raz na miesiąc należą prawdopodobnie w większości do młodszych ankietowanych wśród których zakupy wykonywane w ten sposób są bardziej popularne, a raz do roku wykonują osoby mniej obyte z technologią (potencjalnie osoby starsze, w wieku emerytalnym). Ta informacja pozwoli mi dostosować częstotliwość wysyłanych drogą mailową promocji sezonowych i przecen, aby pokrywały się z możliwą potrzebą klienta na dokonanie zakupu.



Wykres 6.Budżet roczny na zakupy internetowe respondentów

Źródło: badania własne

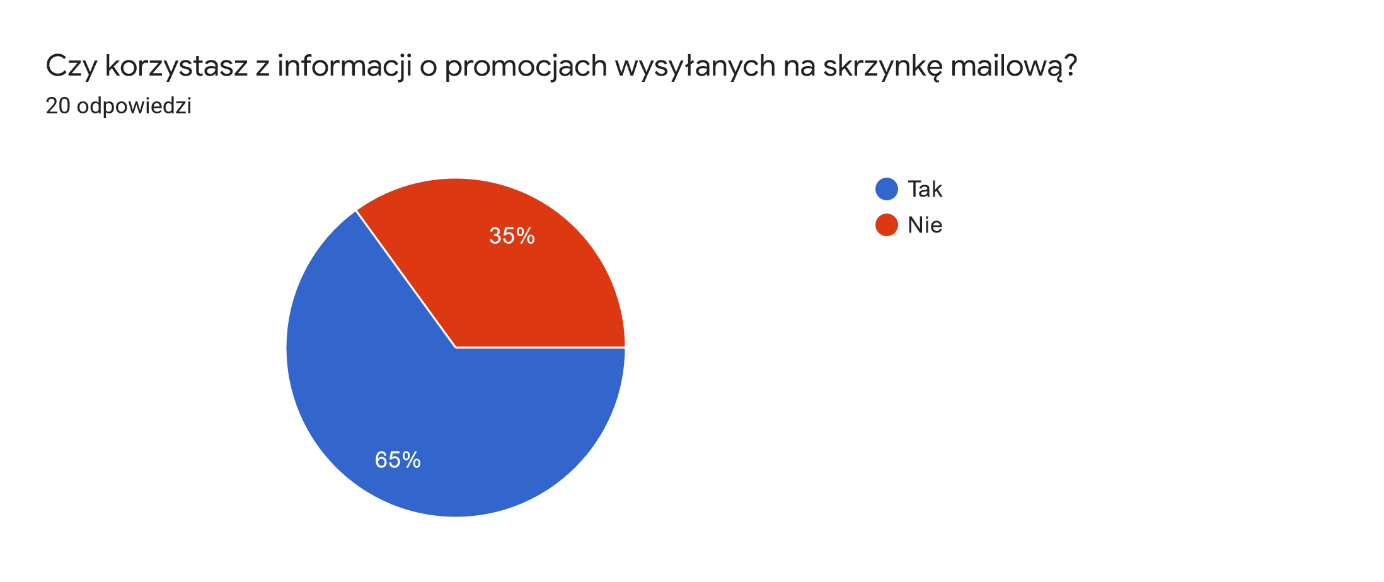
Szóste pytanie ankiety dotyczyło zakresu kwoty, którą potencjalny klient przeznacza na prezenty okazyjne. Połowa ankietowanych odpowiedziała, że kwota ta mieści się w przedziale od 500 do 1000 zł. Nieznacznie mniej osób (40%), wydaje kwoty od 200 do 500 zł. Te dane pozwolą mi dostosować przedziały cenowe i promocyjne do możliwości finansowych potencjalnego klienta, tak aby kupujący nie doznawał odczucia przepłacenia za produkt, który otrzymuje.



Wykres 7. Stosunek respondenta do reklam w sklepach internetowych

Źródło: badania własne

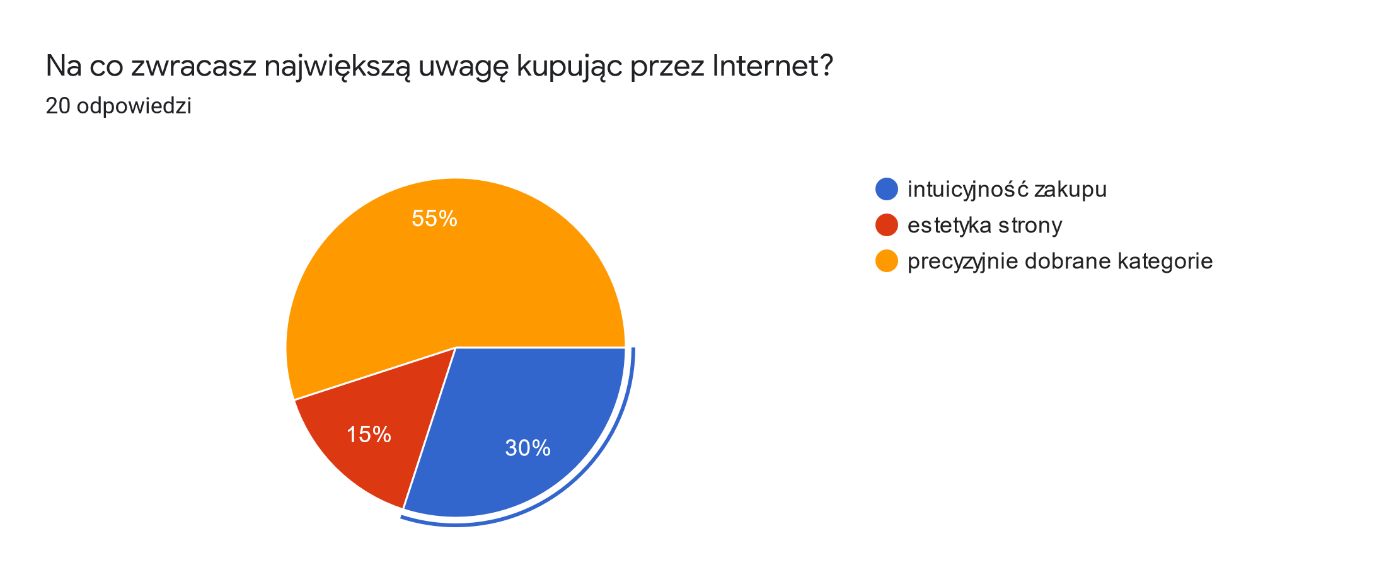
Oczywistym jest fakt, że często odwiedzane strony otrzymują propozycję umieszczenia na nich reklam więc następne pytanie i odpowiedzi dotyczyły opinii na temat umieszczania ich na witrynie sklepu. Większość ankietowanych nie życzy sobie reklam niezwiązanych ze sklepem, w którym dokonują zakupów (60%). Pozostałym respondentom reklamy te nie przeszkadzają w przeglądaniu oferty. Te dane instruują mnie jako twórcę strony o zadbaniu, aby przy wystąpieniu konieczności umieszczenia na stronie reklam, nie były one w formie wyskakujących inwazyjnych okien które klient musiałby ręcznie zamykać.



Wykres 8. Korzystanie z promocji przez respondenta

Źródło: badania własne

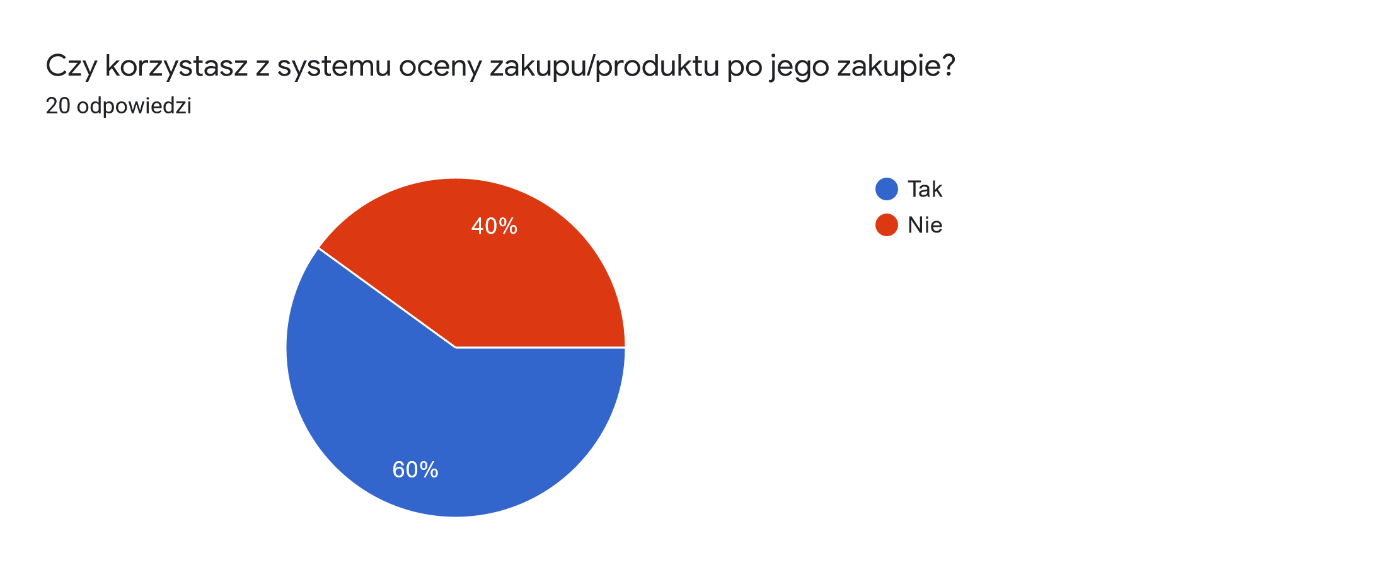
Wykres 8 pokazuje, że większość ankietowanych korzysta z ofert sezonowych i przecen o których jest informowany drogą poczty elektronicznej (65%). Wraz z danymi z wykresu 5 wiem teraz, że klient chce korzystać z ofert przesyłanych tą drogą i jak często powinien otrzymywać takie oferty.



Wykres 9. Na co respondent zwraca uwagę podczas zakupów internetowych

Źródło: badania własne

Z badań wynika, że największą uwagę respondent przykłada do precyzyjnie dobranych kategorii (55%). Na intuicyjność dokonania zakupu krok po korku przypada (30%) odpowiedzi (prawdopodobnie starsi ankietowani mniej doświadczonymi w nawigacji w sklepie internetowym). Estetyka strony jest najmniej ważnym elementem dla ankietowanych. Wszystkie trzy elementy są ważne dla klienta, ale priorytetem powinna być funkcjonalność, czyli precyzyjnie dobrane kategorie, które sprawią, że pozycjonowanie strony w wyszukiwarce wzrośnie oraz intuicyjność zakupu, co sprawi, że dokonywanie zakupów w aplikacji będzie wygodne.



Wykres 10. Ocenianie produktów po jego zakupie przez respondenta

Źródło: badania własne

Większość badanych ocenia produkt po jego zakupie (60%). Należy więc dodać możliwość dodania informacji zwrotnej w postaci oceny produktu lub dodania komentarza informującego następnego klienta o jego jakości.



Wykres 11. Wpływ oceny innych użytkowników na zakup produktu

Źródło: badania własne

Duży odsetek respondentów sprawdza opinie na temat produktu, który chce kupić zostawione przez innych klientów, którzy zakupu tego dokonali w przeszłości i zostawili ocenę. Jest to dość osobliwe, gdyż znacznie mniejszy procent pytanych oceny te chce zostawiać samemu. Rozwiązaniem tego problemu może być wysłanie wiadomości pocztą elektroniczną zbiegającą się czasowo z dostarczeniem produktu.



Wykres 12. Reklamowanie produktów przez respondenta

Źródło: badania własne

Większość ankietowanych spotkała się z koniecznością złożenia reklamacji (60%) wynikiem niezadowalającej jakości towaru. Aby zapobiec reklamacją w sklepie internetowym należy zadbać o sprawdzanie poprawności wpisywanych przez klienta danych.



Wykres 13. Preferencje respondentów odnośnie rejestracji

Źródło: badania własne

Preferencje użytkowników z Wykresu 13 wskazują, że 55% badanych woli finalizować transakcję bez bycia zmuszonym do przejścia przez proces rejestracji, aby przemówić do wszystkich klientów pod tym względem kupujący powinien mieć opcję wcześniejszej rejestracji by oszczędzić jego czas na ponownym wpisywaniu danych. Osoby, które chcą szybko dokonać jednorazowego zakupu powinny mieć możliwość dokonania zakupu bez rejestracji wprowadzając wszystkie niezbędne dane kontaktowe.

Ostatnie pytanie ankiety było opcjonalnym pytaniem otwartym, zadanym wprost w celu pozyskania ogólnych informacji na temat interesujących ich funkcjonalności, których brakuje respondentom w istniejących sklepach internetowych. Na bazie otrzymanych odpowiedzi dowiedziałem się, że respondentom, którzy udzieli odpowiedzi na to pytanie najczęściej brakuje większej liczby metod płatności. Jest to prawdopodobnie wynik zwiększającej się liczby aplikacji płatniczych oraz różnego rodzaju kryptowalut.

Na podstawie analizy odpowiedzi z przeprowadzonej ankiety stworzona została lista wymagań funkcjonalnych i niefunkcjonalnych aplikacji.

# **Wymagania funkcjonalne**

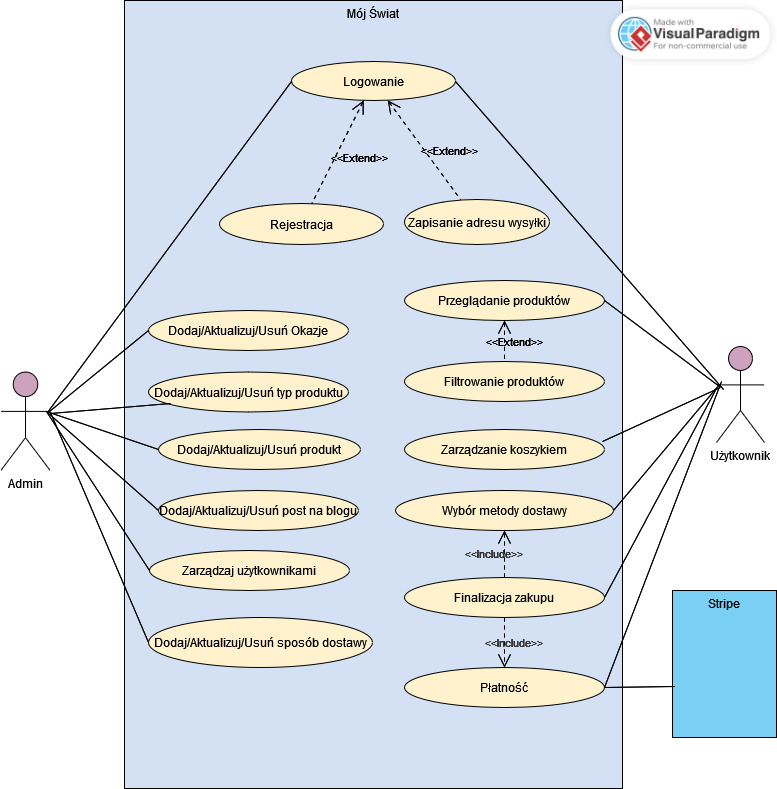
W tych dwóch podrozdziałach sporządzone zostaną tabelki wymagań funkcjonalnych i niefunkcjonalnych aplikacji oraz diagram przypadków użycia. Każdy z tych elementów jest ważnym krokiem w stronę spełnienia oczekiwań przyszłych użytkowników systemu.

|  |  |  |
| --- | --- | --- |
| Nazwa | Opis | Priorytet |
| Rejestracja | Nowy użytkownik podaje identyfikator i hasło do systemu | Wysoki |
| Logowanie | Zarejestrowany użytkownik loguje się do systemu wykorzystując swój identyfikator i hasło | Wysoki |
| Przeglądanie oferty | Użytkownik przegląda listę produktów | Wysoki |
| Składanie zamówienia | Użytkownik przechodzi przez cztery etapy składania zamówienia | Wysoki |
| Obsługa koszyka | Dodawanie lub usuwanie przedmiotu z koszyka | Wysoki |
| Zmiana strony na blog | Po wciśnięciu użytkownik przenoszony jest na stronę bloga artystycznego | Średni |
| Widok administracyjny | Uaktualnienie ,edycja, usuwanie produktów z listy. | Krytyczny |
| Filtracja | Filtrowanie produktów według wskazanych kryteriów | Wysoki |
| Przekierowywanie | Przekierowywanie na stronę płatności(banku) | Średni |

Tabela 1. Wymagania funkcjonalne aplikacji

Źródło: opracowanie własne

W Tabeli 1 przedstawione zostały wymagania, jakie aplikacja docelowo powinna spełniać w zakresie funkcjonalności [10] aby system mógł działać poprawnie. Z punktu widzenia przyszłego programisty, posiadanie takiej listy jest drogowskazem podczas planowania architektury systemu, jego implementacji oraz tworzeniu interfejsów użytkownika. Kryterium priorytetu dobierane było do każdej funkcjonalności ze względu na role jaką będzie pełnić w aplikacji oraz poziom skomplikowania jej implementacji.



Rys. 2 Diagram przypadków użycia aplikacji

Źródło: https://online.visual-paradigm.com, z dnia 4.05.2023

Na Rys. 2 zaprezentowana została graficzna reprezentacja funkcjonalności systemu, która jest uzupełnieniem Tabeli 1 o scenariusze użytkowania systemu. W fazie analizy taki diagram pozwala lepiej zrozumieć wymagania biznesowe i użytkownika, a w następstwie większą efektywność w planowaniu i  implementacji systemu. Konkretne scenariusze przypadków użycia przedstawione zostały w Tabeli 3.

|  |  |  |  |
| --- | --- | --- | --- |
| **Przypadek:** | **Aktorzy:** | **Scenariusz główny:** | **Rozszerzenia** |
| Logowanie | Admin, Użytkownik | Wprowadzenie poprawnych danych weryfikowanych przez system | Rejestracja, Zapis adresu |
| Przeglądanie produktów | Użytkownik | Przeglądanie katalogu produktów | Filtrowanie produktów |
| Wybór metody dostawy | Użytkownik | Użytkownik wybiera dostępną metodę dostawy |  |
| Finalizacja zakupu | Użytkownik | Użytkownik przechodzi przez proces złożenia zamówienia |
| Płatność | Użytkownik Stripe | Wprowadzanie poprawnych danych płatniczych |
| Operacje CRUD na okazjach | Admin | Dodawanie, usuwanie i edycja okazji |
| Operacje CRUD na typach produktu | Admin | Dodawanie, usuwanie i edycja typów produktów |
| Operacje CRUD na produktach | Admin | Dodawanie, usuwanie i edycja produktów |
| Zarządzanie blogiem | Admin | Zarządzanie blogiem artystycznym |
| Zarządzanie użytkownikami | Admin | Zarządzanie danymi użytkowników |
| Operacje CRUD na sposobach dostawy | Admin | Dodawanie, usuwanie i edycja sposobów dostawy |

Tabela 2 Tabela przypadków użyć aplikacji

Źródło: opracowanie własne

# **Wymagania niefunkcjonalne**

Specyfikacja wymagań systemowych [11] stanowi istotny aspekt, który pozwoli zapewnić użytkownikowi wygodę i swobodę w obsłudze aplikacji. Oczywistym jest, że nie wszystkie wymagania z tego zakresu są warte nakładu czasu i środków, które trzeba było by przeznaczyć na ich zaspokojenie. Wśród wszystkich pozycji przedstawionych na Tabeli 3 najważniejszymi z nich są te dotyczące bezpieczeństwa danych płatniczych i osobowych użytkownika.

|  |  |  |
| --- | --- | --- |
| Cecha | Opis | Miara |
| Bezpieczeństwo | Serwer posiada certyfikat SSL | Ważność certyfikatu |
| Wydajność | Odpowiedź z serwera | <=4s |
| Współbieżność | Możliwość dostępu do systemu przez wielu użytkowników jednocześnie | Stabilność witryny |
| Responsywność | Jedna wersja dla urządzeń stacjonarnych i mobilnych | Responsywna wersja strony |
| Kompatybilność | Możliwość uruchomienia witryny na różnych przeglądarkach | Mozilla Firefox, Google Chrome, Microsoft Edge, Opera |
| Zasoby | Wykupienie domeny | Ważność subskrypcji |
| Niezawodność | Prawdopodobieństwo zniszczenia danych w przypadku awarii | Aktualna kopia zapasowa bazy danych |

Tabela 3. Wymagania niefunkcjonalne aplikacji

Źródło: opracowanie własne

# **Zastosowane narzędzia i technologie**

Poniższy rozdział przedstawia informacje odnośnie wykorzystanych narzędzi i technologii w projektowaniu i implementacji aplikacji.

# **REST API**

ASP .NET Core to otwarty darmowy framework programistyczny stworzony przez firmę Microsoft. Jest on przeznaczony do budowania aplikacji webowych, desktopowych i mobilnych, a także usług chmurowych i internetowych. Framework cechuje wieloplatformowość co oznacza, że jest kompatybilny z systemami Windows, Linux i macOS. Jego funkcjonalność obejmuje działanie w oparciu o język C#,  a  także umożliwia sprawne dodawanie i korzystanie z bibliotek i narzędzi. W ramach frameworka znajdują się między innymi biblioteki do obsługi bazy danych, zarządzania zależnościami oraz narzędzia do testowania kodu. Dodatkowo .NET Core zapewnia szybkość i wydajność aplikacji o dużej skali i wymagających szybkich reakcji. Bardzo ważnym elementem jest kompatybilność wstecz, czyli aplikacja, która została napisana w starszej wersji Frameworka będzie nadal działać po jego aktualizacji [12-14].

Inżynieria tworzenia oprogramowania cały czas się rozwija, aby z jednej strony dopasować się do oczekiwań użytkowników, a z drugiej zaadaptować do bieżących trendów i technologii. Kiedyś aplikacje desktopowe uruchamiane z poziomu komputera użytkownika docelowego miały większe znaczenie. Od dłuższego czasu natomiast przodują aplikacje, z których można korzystać za pomocą przeglądarek internetowych co tym samym sprawia, że nie są zależne od urządzenia, z którego poziomu użytkownik chce z niej skorzystać [15].

Język, w którym aplikacja została zaimplementowana to język wysokopoziomowy C#, ponieważ umożliwia on wiele dogodnych rozwiązań przy tworzeniu aplikacji internetowych, takich jak: podział aplikacji według jej zadań lub problemów co pomaga uporządkować rosnącą bazę kodu a podział aplikacji na warstwy logiczne ułatwiają zastępowanie funkcji w aplikacji. Dzięki temu zabiegowi można początkowo używać bazy danych SQL Server, ale w późniejszej fazie może zostać zastąpiona interfejsem API.

Angular to framework front-endowy stworzony przez firmę Google, służący do tworzenia zaawansowanych aplikacji internetowych [16]. Framework został wydany w 2010 roku pod nazwą AngularJS, a jego rozwój kontynuowany jest pod nazwą Angular. Wykorzystuje on język TypeScript i oferuje wiele funkcjonalności takie jak dynamiczne widoki, komponenty i obsługa zapytań HTTP. Angular wykorzystuje komponenty do tworzenia interfejsu użytkownika, które są niezależne od siebie i oferują ponowne wykorzystywanie ich w aplikacji co pozwala zmniejszać nakład czasu i zwiększyć wydajność aplikacji. Framework ten został wybrany do stworzenia interfejsu użytkownika.

# **Bazy Danych**

Jako zbiór skategoryzowanych danych aplikacji wykorzystana zostanie relacyjna baza danych co oznacza, że dane tabel połączone są ze sobą kluczami głównymi i obcymi. W fazie rozwoju programu użyta zostanie biblioteka SQLite do stworzenia tymczasowych baz danych, które w fazie produkcji zostaną podmienione na te obsługiwane przez Microsoft SQL Server. ORM (ang. *Object-Relational Mapping)* działa jako warstwa abstrakcji między aplikacją a bazą danych, co umożliwia programistom pracę na wyższym poziomie abstrakcji i upraszcza zarządzanie danymi w bazie. ORM umożliwia tworzenie klas i obiektów w językach programowania, które odzwierciedlają strukturę danych w bazie, a następnie automatycznie mapuje dane między nimi. Dla kompatybilności z bazami danych jak ORM zastosowany został Entity Framework Core [18].

# **Narzędzia**

**Visual Studio 2022**

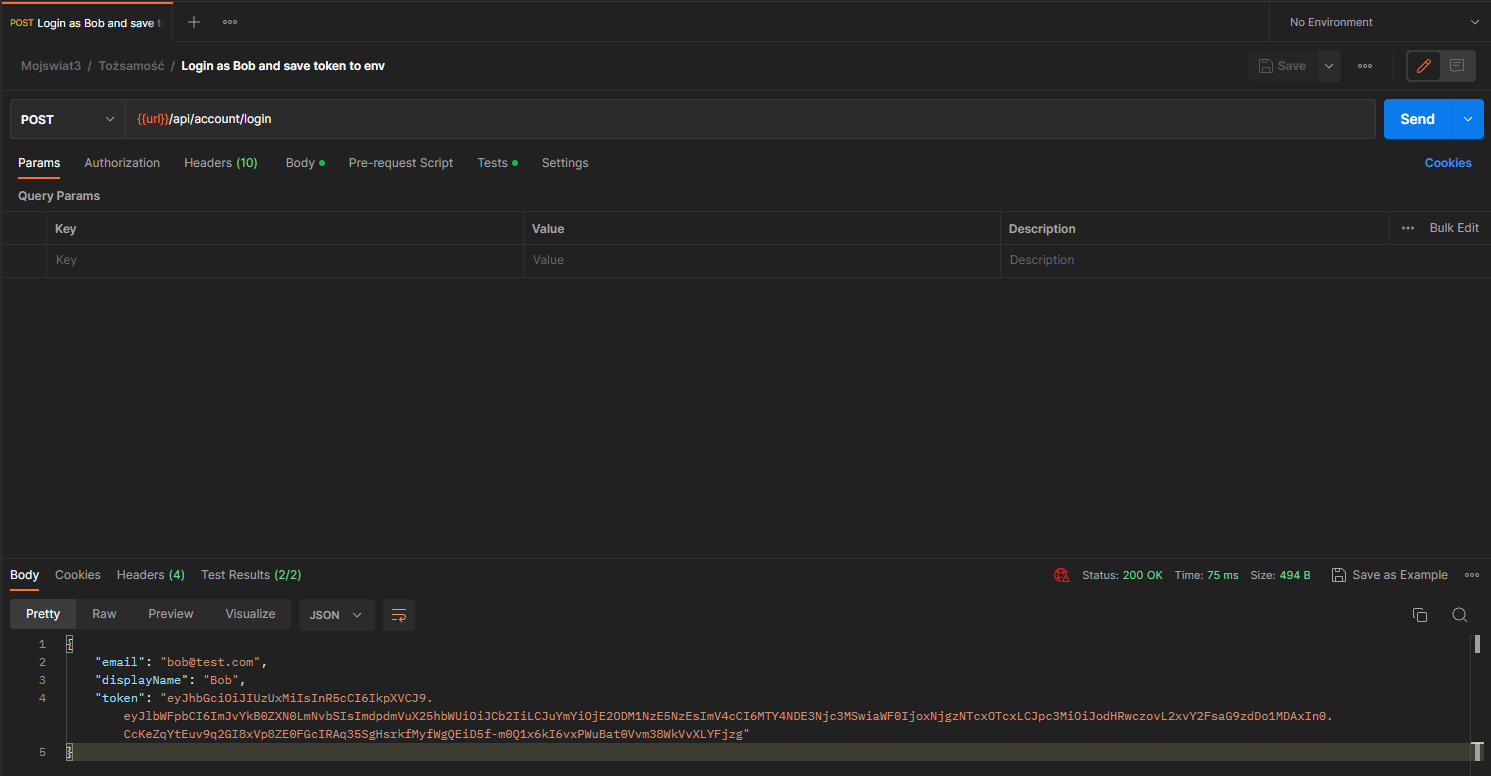
Jest to zintegrowane, wieloplatformowe środowisko programistyczne stworzone przez firmę Microsoft która umożliwia programistom pisanie, testowanie i debugowanie aplikacji. W jego skład wchodzi wiele narzędzi ułatwiające właśnie te zadania takie jak analizator wydajności czy wbudowany debugger. Środowisko oferuje także integracje z systemem kontroli Git, który wykorzystywany jest na co dzień w  pracy programisty. Narzędzie to zostało wykorzystane do tworzenia projektów logiki aplikacji.

**Visual Studio Code**

W odróżnieniu od Visual Studio jest to lżejszy i bardziej elastyczny edytor kodu źródłowego. Zapewnia on dostosowanie się do potrzeb programisty dzięki dużej liczbie dostępnych wtyczek. Według mnie jego największymi zaletami jest uniwersalności i elastyczność w nawigacji pomiędzy plikami i projektami. To właśnie w nim zawarta jest podzielona na poszczególne projekty logika aplikacji.

**Postman**

Jest to narzędzie, które służy do testowania i debugowania API, które umożliwia programistom, testerom i inżynierom testowym przeprowadzanie testów integracyjnych, funkcjonalnych i automatycznych. Postman [19] oferuje intuicyjny interfejs graficzny użytkownika do tworzenia, wysyłania i odbierania żądań HTTP, a także przetwarzania odpowiedzi API. Aby przeprowadzić poprawny test, tak jak na Rys. 3 należy wybrać żądanie z listy oraz wpisać adres metody API, którą chcemy przetestować. Następnie wprowadzamy parametry zapytania i dane wejściowe w poprawnym formacie i za pomocą przycisku wysyłamy zapytanie. Na odpowiedź składa się ciało (ang. *body*) w formacie JSON wraz z rezultatami testów i kodem sukcesu lub odpowiedniego błędu.

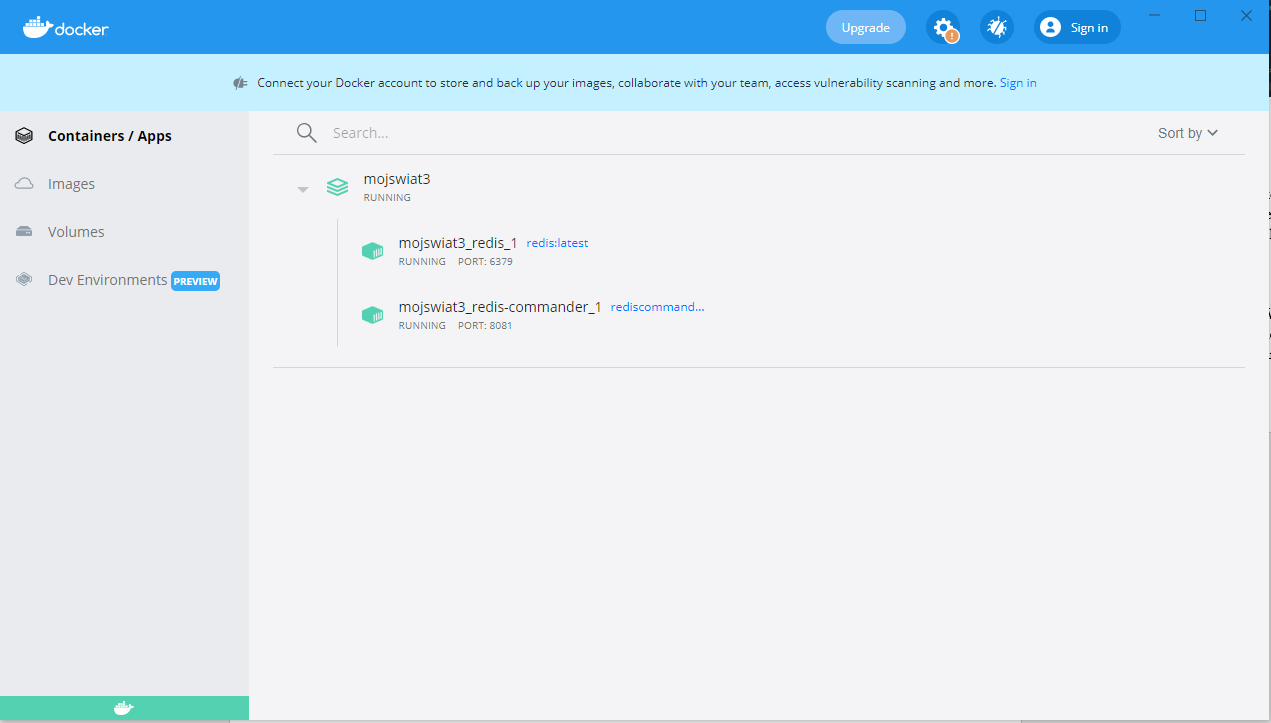


Rys. 3 Test zapytania loginu w programie Postman

Źródło: opracowanie własne

**Docker**

Jest to narzędzie służące do tworzenia, testowania i wdrażania aplikacji w izolowanych kontenerach. Kontenery Docker [20] to wyizolowane środowiska uruchomieniowe, które zawierają aplikacje wraz ze wszystkimi zależnościami i bibliotekami wymaganymi do ich uruchomienia. Dla mojej aplikacji Docker wykorzystywany będzie do stworzenia i uruchomienia bazy danych Redis (Rys.4).

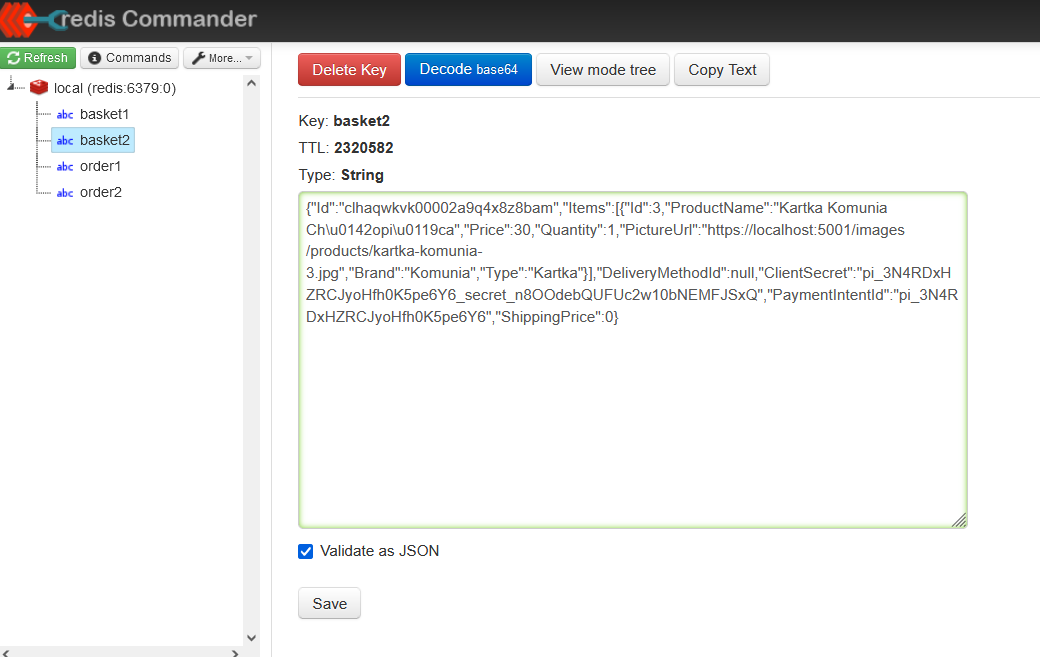


Rys. 4 Wygląd działania programu Docker Desktop

Źródło: opracowanie własne

**Redis**

Redis to jedna z najpopularniejszych baz danych działających na zasadzie klucz-wartość [21]. Współpracuje ona z danymi przechowywanymi w pamięci RAM. Jest on bardzo wygodny w przechowywaniu struktur takich jak ciągi znaków, listy i mapy bitowe. W aplikacji internetowej zastosowany został przy pomocy środowiska uruchomieniowego Docker do przechowywania listy produktów z koszyka i zamówień tak jak zostało to przedstawione na Rys. 5.

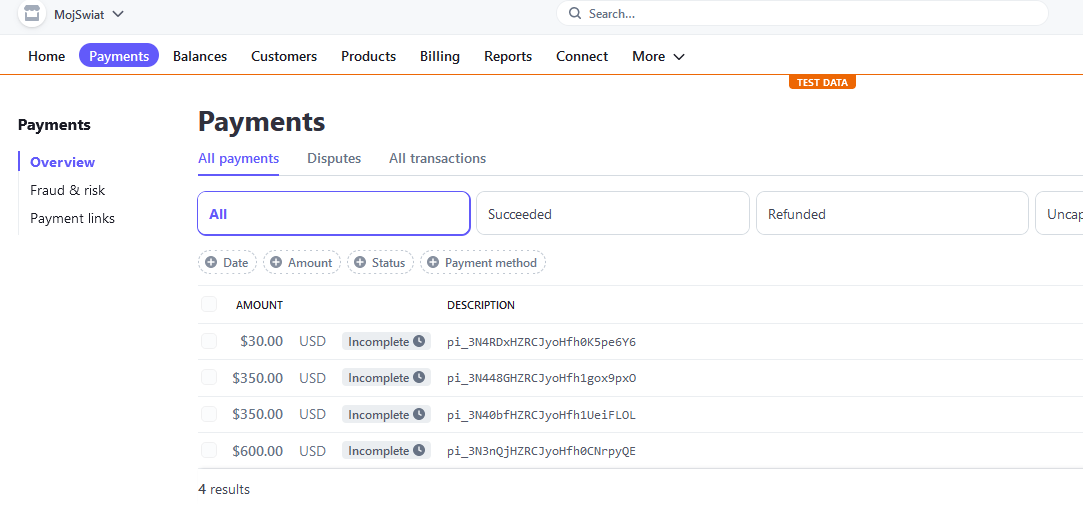


Rys. 4 Wygląd bazy danych Redis

Źródło: opracowanie własne

**Stripe**

Jest to platforma oferująca darmowe testowe oprogramowanie do przyjmowania, testowania i zarządzania płatnościami. Przyjmuje ona zapytania HTTP i jest kompatybilne ze środowiskiem .NET co w parze z łatwą obsługą sprawiło, że zdecydowałem się skorzystać z niego do mojej aplikacji.

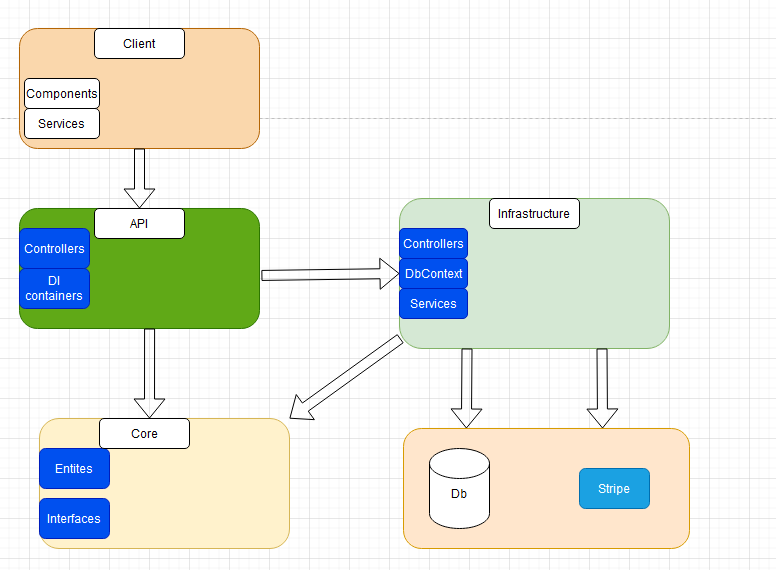


Rys. 5 Wygląd serwisu Stripe

Źródło: https://dashboard.stripe.com/test/payments, z dnia 04.03.2023

# **Architektura**

Struktura aplikacji opiera się na czterech projektach, każdy z których ma określony zakres zasobów i  zadań jakie spełniają w systemie. Cały schemat systemu został przedstawiony na Rys. 6. Aplikacja internetowa jest odpowiedzialna za udostępnienie użytkownikowi danych i umożliwienie mu wysyłania zapytań REST do API, gdzie jest ono przetwarzane i wysyłane do infrastruktury skąd formowane są zapytania kierowane do baz danych. Następnie zasób z bazy danych kierowany jest najpierw do API,  gdzie następnie odpowiedź zwrotna w formie HTTP kierowana jest do klienta.



Rys. 6 Architektura systemu

Źródło: opracowanie własne

# **Implementacja**

Poniższy rozdział zawiera informacje o sposobie implementacji aplikacji. Podzielony on został na kilka części odpowiadających poszczególnym projektom i zadaniom jakie są im przypisane. W pierwszym podrozdziale opisane zostaną ogólne pojęcia dotyczące wzorców projektowych zastosowanych w pisaniu aplikacji. Następnie opisane zostaną poszczególne projekty i jakie funkcjonalności dostarczają względem zapytań REST.

# **Wzorce Projektowe**

Repozytorium – wzorzec projektowy, który zapewnia sposób oddzielenia logiki biznesowej aplikacji od mechanizmu przechowywania danych poprzez stworzenie warstwy abstrakcji między nimi. Udostępnia on zestaw metod (Rys. 6), których logika biznesowa może użyć do wykonywania operacji CRUD (Create, Read, Update i Delete) na danych. Ponadto dla aplikacji zastosowane zostało generyczne repozytorium, w którym typy anonimowe zostaną zastąpione typami produkcyjnymi, co pozwala zaoszczędzić czas i uniknąć powielania kodu.

Dependency injection – wzorzec projektowy, który opiera się na wstrzykiwaniu zależności. Wstrzykiwanie zależności polega na przekazaniu obiektów zależności jako parametrów konstruktora lub metod klasy, zamiast tworzenia ich wewnątrz klasy [22]. Podczas uruchamiania aplikacji do interfejsów przekazanych w konstruktorze przypisywana jest mu odpowiednia implementacja.

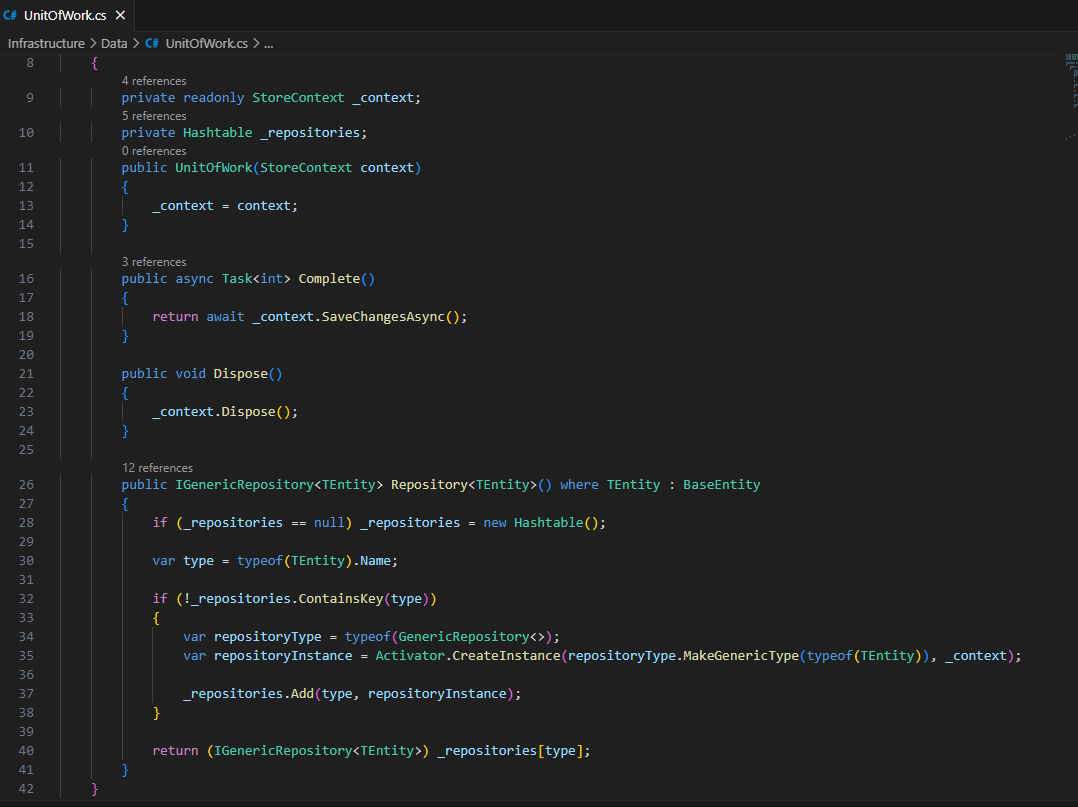
Specification – wzorzec projektowy, który umożliwia opis zbioru kryteriów zapytań danych, w bardziej czytelny sposób. W tym wzorcu definiujemy zestaw reguł biznesowych (specyfikacji) jako obiekty, które następnie można łączyć ze sobą na różne sposoby, tworząc bardziej złożone zapytania [23]. Te specyfikacje obejmują reguły filtrowania lub wyszukiwania danych, takich jak szukany produkt lub suma zamówień.

Unit of Work – wzorzec projektowy, którego zadaniem jest koordynacja operacji na danych w aplikacji. Zapewnia on spójność danych w aplikacji poprzez grupowanie zadań, które mają być wykonane w jednej transakcji [24]. Dodatkowo jest on kompatybilny z wzorcem repozytorium, gdzie w praktyce repozytoria udostępniają interfejs do operacji na danych, a jednostka pracy pełni role zarządzania nimi jako całością.



Rys. 6 Fragment kodu wzorca interfejsu produktu

Źródło: opracowanie własne



Rys. 7 Fragment implementacji wzorca Unit of Work

Źródło: opracowanie własne

# **Serwisy główne**

Do zaprojektowania aplikacji wykorzystane zostało podejście DDD (ang. *Domain Driven Design*). Jego głównym założeniem jest definiowanie obiektów i ich zachowań w taki sposób, aby wiernie odzwierciedlały rzeczywistość. Ta metodyka definiuje również szereg wzorców projektowych, które pomagają w budowie bardziej skomplikowanych modeli dziedziny takich jak wzorzec Agregatu (*ang. Aggregate*). Takie podejście pozwoliło na podziału projektu na zależne od siebie komponenty:

* API – ta część aplikacji to REST API, którego głównym zadaniem jest przyjmowanie i odpowiadanie na żądania HTTP. Jest on warstwą łączącą wizualny interfejs użytkownika z bazą danych. Przechowuje on kontrolery aplikacji, które służą do obsługi zapytań, obiekty transferu danych (ang. DTO – Data Transfer Object) które definiują w jaki sposób dane będą wysyłane do sieci oraz sposób zwraca odpowiedzi w przypadku błędu.
* Infrastructure – istotą tej części programu jest wysyłanie zapytań do bazy danych i otrzymywanie danych. Przechowuje ona konfiguracje sposobów w jaki pobierać informacje o encjach, zdefiniowane repozytoria, implementacje interfejsu Unit of Work i serwisy, które rozszerzają funkcjonalność aplikacji.
* Core – w tej części aplikacji zdefiniowane są interfejsy, które zaimplementowane są w infrastrukturze. Znadują się tutaj również encje biznesowe. Ta część projektu znajduje się na końcu zależności przechodniej, co oznacza, że projekt Core jest niezależny, ale poprzednie projekty zależą od niego.
* Client – ta część projektu to wizualny interfejs użytkownika, jego zadaniem jest udostępnienie zasobów domenowych użytkownikowi, umożliwienie interakcji z komponentami, aby wysłać zapytania do serwisu API o zasoby z bazy danych.

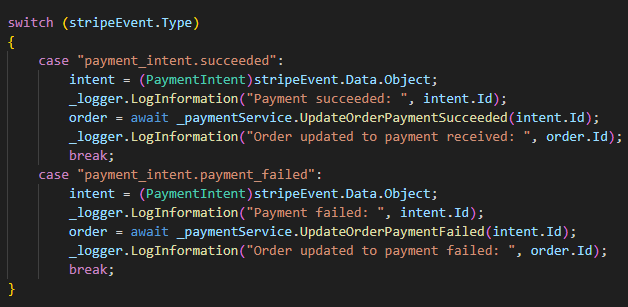
# **Serwis zamówień**

Ta część programu ma za zadanie rozszerzyć funkcjonalność aplikacji o składanie zamówienia przez użytkownika. Jest to klasa, która ma wstrzyknięte zależności jednostki pracy oraz koszyka. Do jego zadań należy:

* Pobranie koszyka z repozytorium, gdzie jako parametr przekazywany jest identyfikator, który jest kluczem głównym.
* Pobranie produktów, które znajdują się w koszyku z repozytorium produktów i ułożenie ich w  agregowaną listę.
* Pobranie metod wysyłki z repozytorium jednostki pracy
* Automatyczne policzenie kosztu produktów razem
* Weryfikacja na podstawie specyfikacji zamówień czy zamówienie nie jest duplikatem.
* Stworzenie zamówienia i zapisanie go w bazie danych

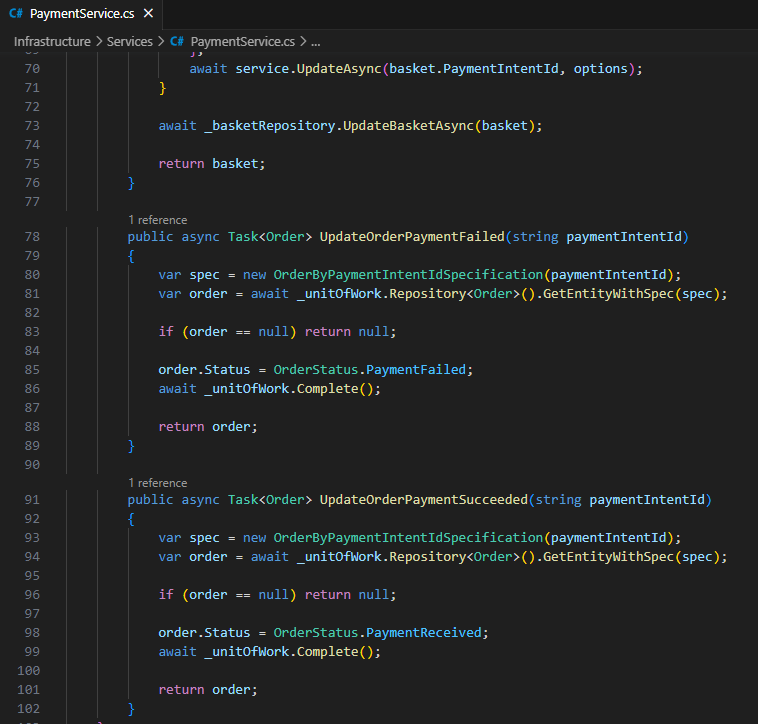
# Serwis płatności

Ten komponent odpowiada za przekazanie informacji o płatności takich jak suma cen wybranych przedmiotów w koszyku (Rys. 9) powiększona o cenę wybranej metody wysyłki do platformy płatności, która zaopatrzona w szyfrowanie danych daje użytkownikowi możliwość bezpiecznego przeprowadzenia transakcji. Dla aplikacji ta funkcjonalność zapewniana jest przez platformę Stripe (Rys. 8). Oferuje ona wiele narzędzi takich jak międzynarodowe płatności, automatyczne zarządzanie odprawami podatkowymi, płatności cykliczne.



Rys. 8 Fragment kodu wzorca interfejsu produktu

Źródło: opracowanie własne



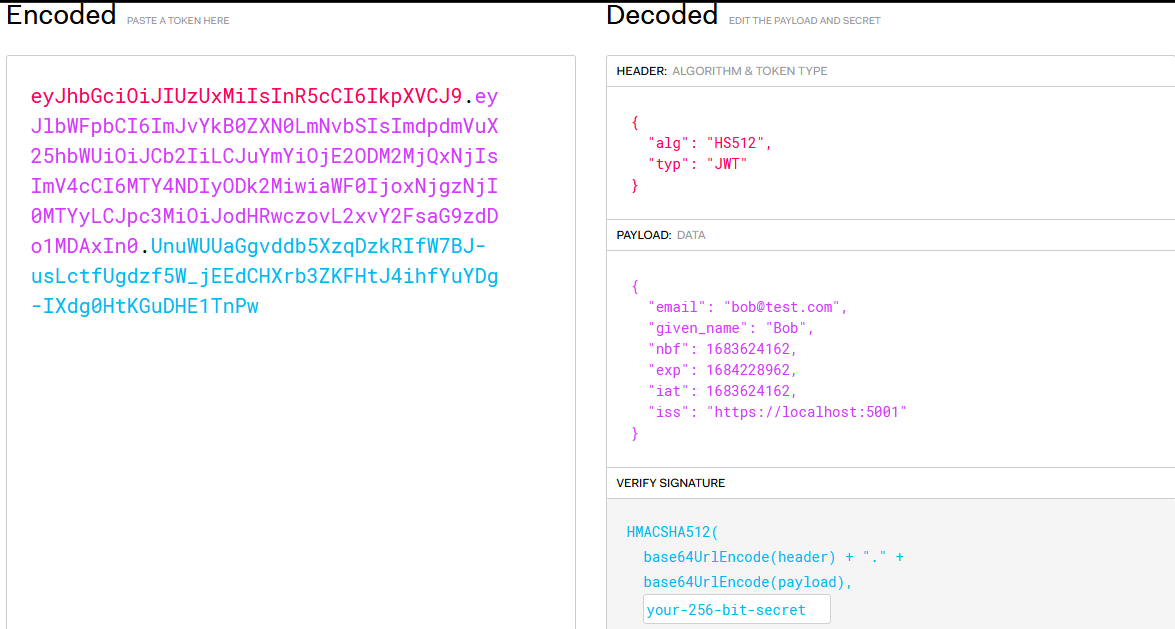
Rys. 9 Fragment kodu z serwisu płatności

Źródło: opracowanie własne

# Serwis identyfikacji

Jest to serwis, który odpowiada za uwierzytelnianie i autoryzacje użytkowników w aplikacji. Ze względów bezpieczeństwa, ta część programu została wydzielona. Baza danych przechowująca dane użytkowników została również oddzielona od głównej bazy danych aplikacji, aby zachować spójność i bezpieczeństwo danych oraz ułatwić przepływ danych. Pierwszym z listy zadaniem jest rejestracja użytkownika. Sprawdzana jest wtedy poprawność adresu email czy użytkownik już istnieje oraz poziom skomplikowania hasła za pomocą zbioru zasad RegEx (*ang. Regular Expression*) [25]. Następnie poprawnie podane hasło jest szyfrowane i zapisywane w bazie danych, aby zapewnić anonimowość danych użytkownika względem osób z dostępem do bazy danych. Do zapewnienia tej funkcjonalność wykorzystana została biblioteka ASP.NET Core Identity.

Pierwszym zadaniem autoryzacji jest sprawdzenie czy użytkownik istnieje w bazie danych. Jeżeli odpowiedź jest twierdząca to następuje weryfikacja hasła, które szyfrowane jest takim samym jaki został użyty do obsługi rejestracji. Po poprawnej autoryzacji danych dla użytkownika wytwarzane są informacje autoryzacyjne. Funkcjonalność tą dostarcza biblioteka generowania tokenów JWT (ang. *Json Web Token)* [26],który przechowuje zaszyfrowane informacje logowania użytkownika. Składa się on z trzech części oddzielonych kropką tak jak zostało to przedstawione na Rys. 10.



Rys. 10 Dekodowanie wiadomości logowania przykładowego użytkownika aplikacji

Źródło: opracowanie własne

Pierwsza część tokenu to nagłówek, na który określa typ algorytmu jakim został sporządzony podpis. Następnym z nich jest zawartość, czyli dane które zostały przekazane do logowania wraz z czasem, kiedy został wygenerowany i kiedy przestaje być ważny oraz strona internetowa, która wydała token. Trzecią częścią jest podpis cyfrowy. Za jego pomocą możemy zweryfikować autentyczność danych przekazanych przez token. Odpowiada za to sekretny klucz, który zakodowany jest w kodzie źródłowym. Dzięki niemu nikt niepożądany nie może wygenerować własnego tokenu i przy jego pomocy zalogować się do aplikacji. Dla zwiększenia bezpieczeństwa po upływie zdefiniowanego czasu użytkownik musi zalogować się ponownie.

# **Prezentacja warstwy użytkowej aplikacji**

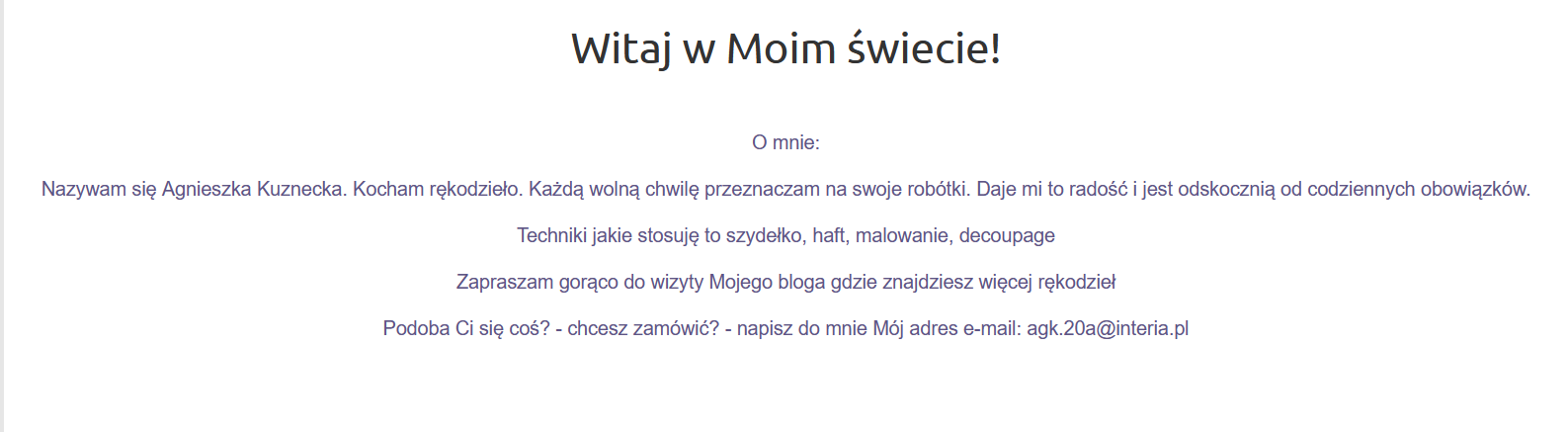
W tym rozdziale przedstawiona zostanie warstwa użytkowa aplikacji wraz z szczegółową instrukcją jej obsługi i zrzutami ekranów jej działania. Zobrazowane zostaną widoki takie jak panel nawigacyjny, lista produktów od strony użytkownika i administratora, obsługa koszyka i rozliczeń oraz ekran wykonywania płatności.

Po uruchomieniu aplikacji wyświetla się ekran startowy zobrazowany na Rys. 11. Jest to domyślny widok użytkownika, który zawiera w sobie pasek nawigacyjny. Dodatkowo w tym widoku użytkownik jest witany przez trzy cyklicznie zmieniające się obrazy w tematyce kwiatów. Poniżej znajduje się krótka biografia twórcy oraz dane kontaktowe, gdyż użytkownicy najczęściej poszukują spersonalizowanego produktu (Rys. 9).



Rys. 11 Ekran startowy aplikacji

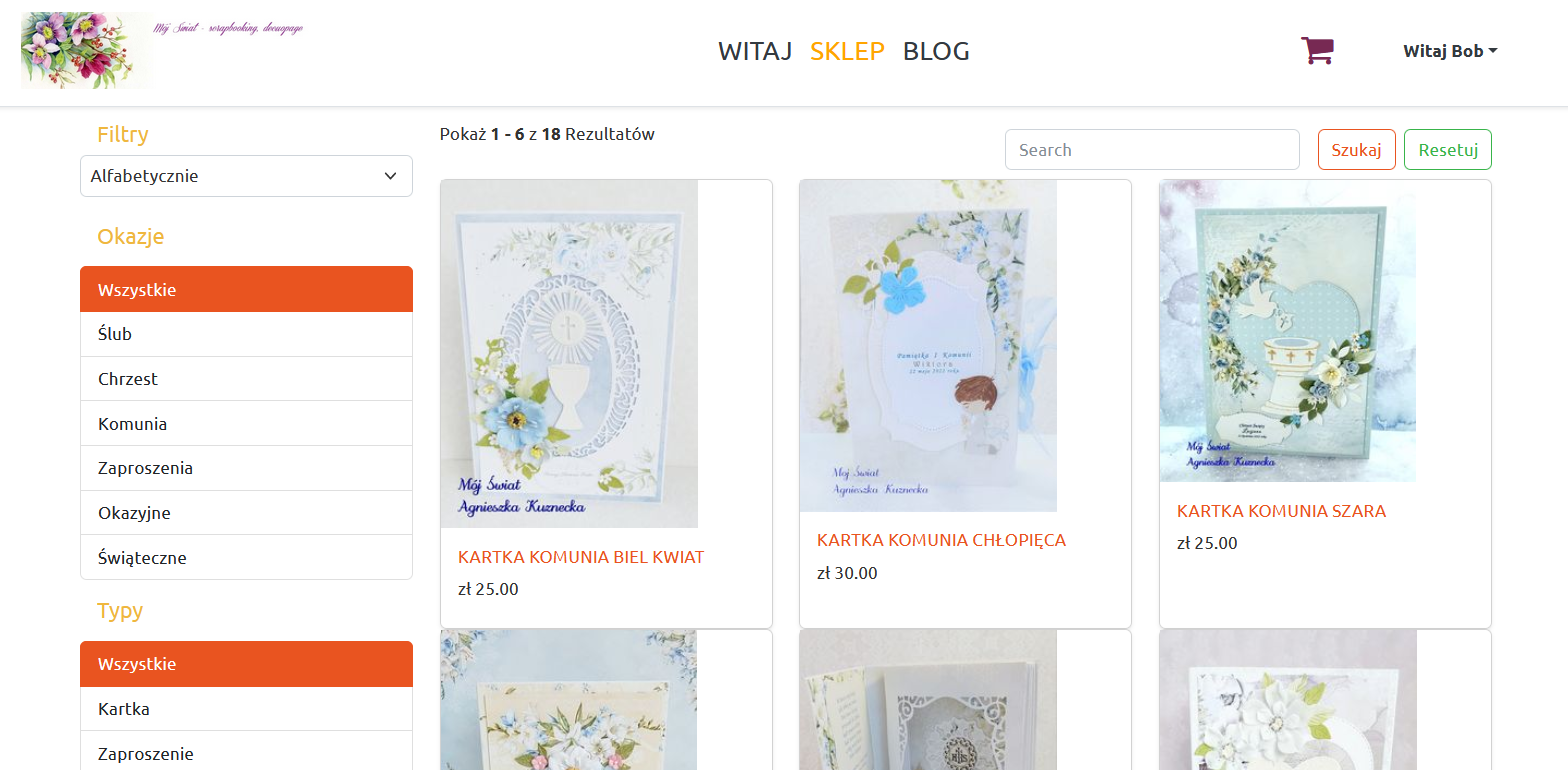
Źródło: opracowanie własne



Rys. 12 Sekcja biografii i informacji kontaktowych

Źródło: opracowanie własne

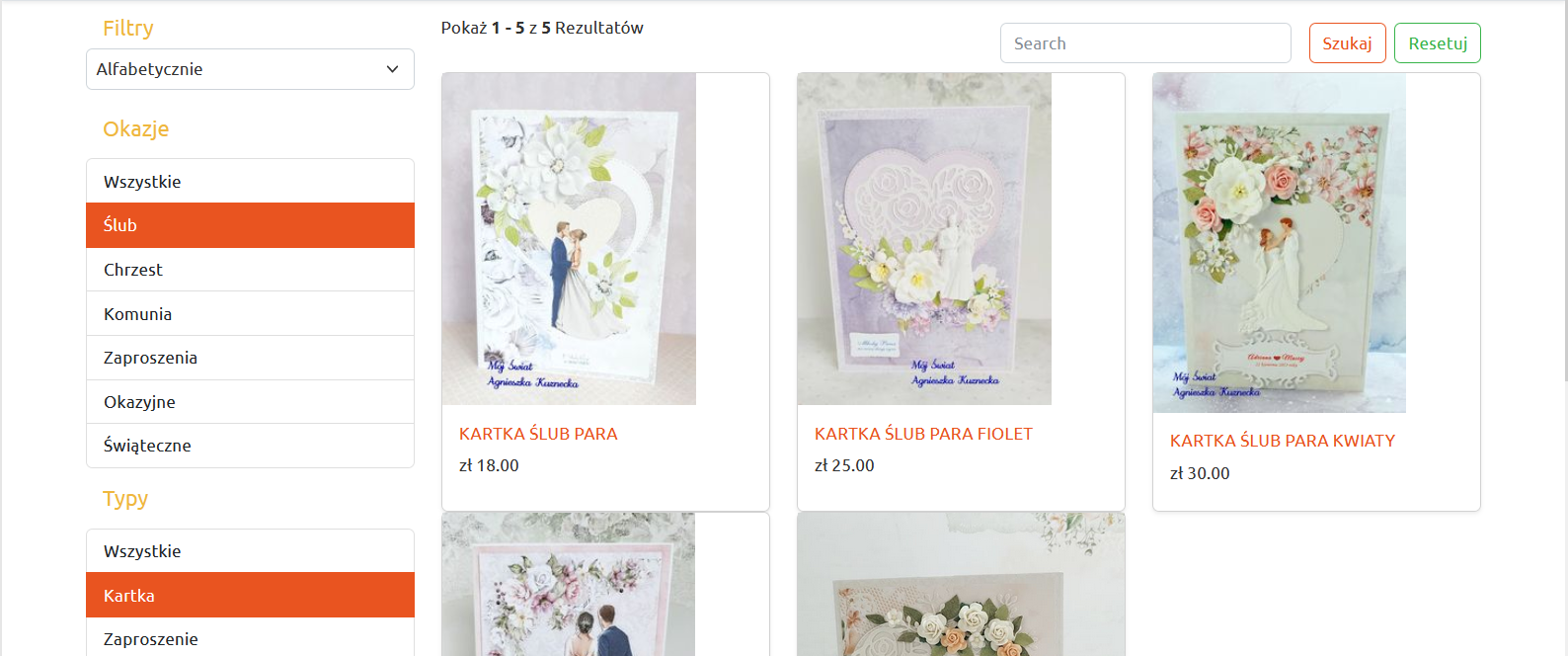
Następnym korkiem użytkownika jest przejście do sekcji sklepu, gdzie otrzyma on dostęp do katalogu towarów pełniej funkcjonalności jaki ten widok oferuje względem produktów (Rys. 13).



Rys. 13 Widok sklepu aplikacji

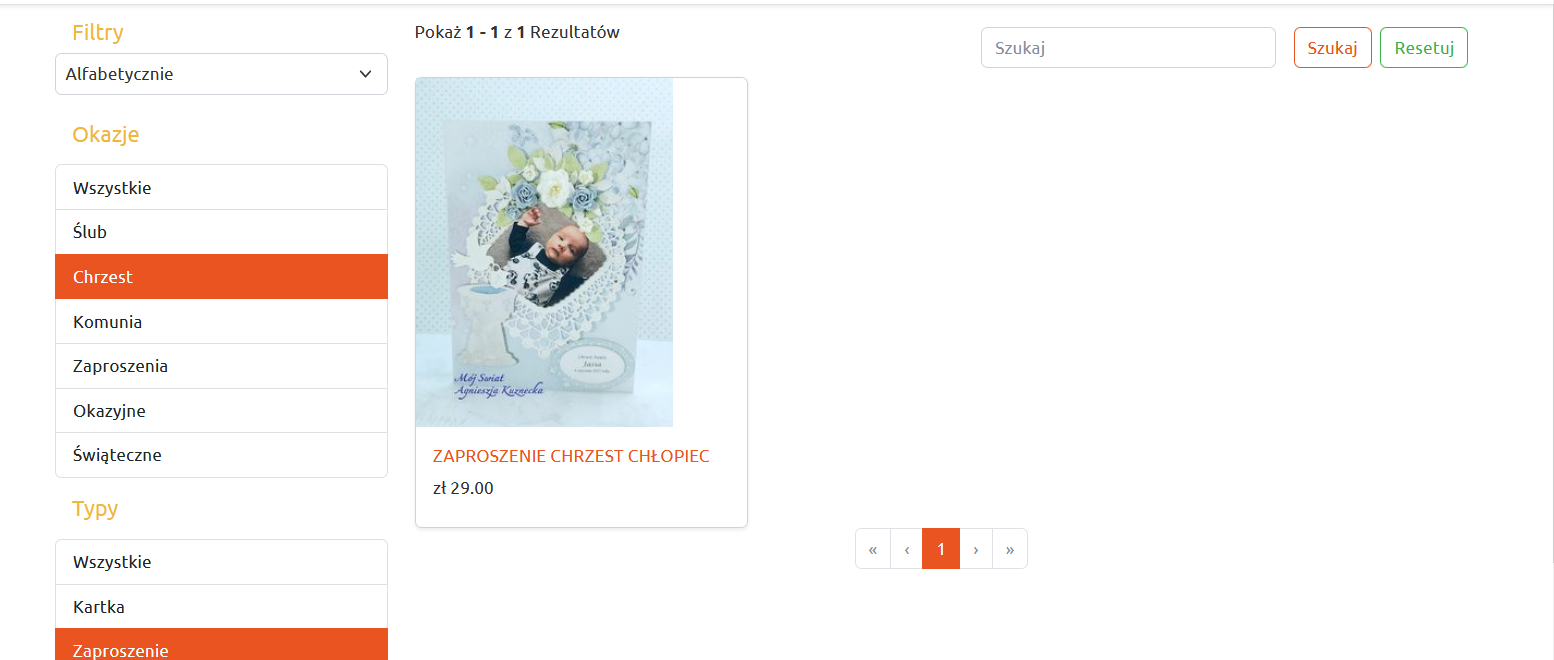
Źródło: opracowanie własne

W tym widoku użytkownik może według własnego uznania manipulować zbiorem produktów tak jak zostało to przedstawione na Rys. 14, Rys. 15 i Rys. 16. Za każdym razem aktualizowana jest lista produktów oraz ilość pokazywanych rezultatów.



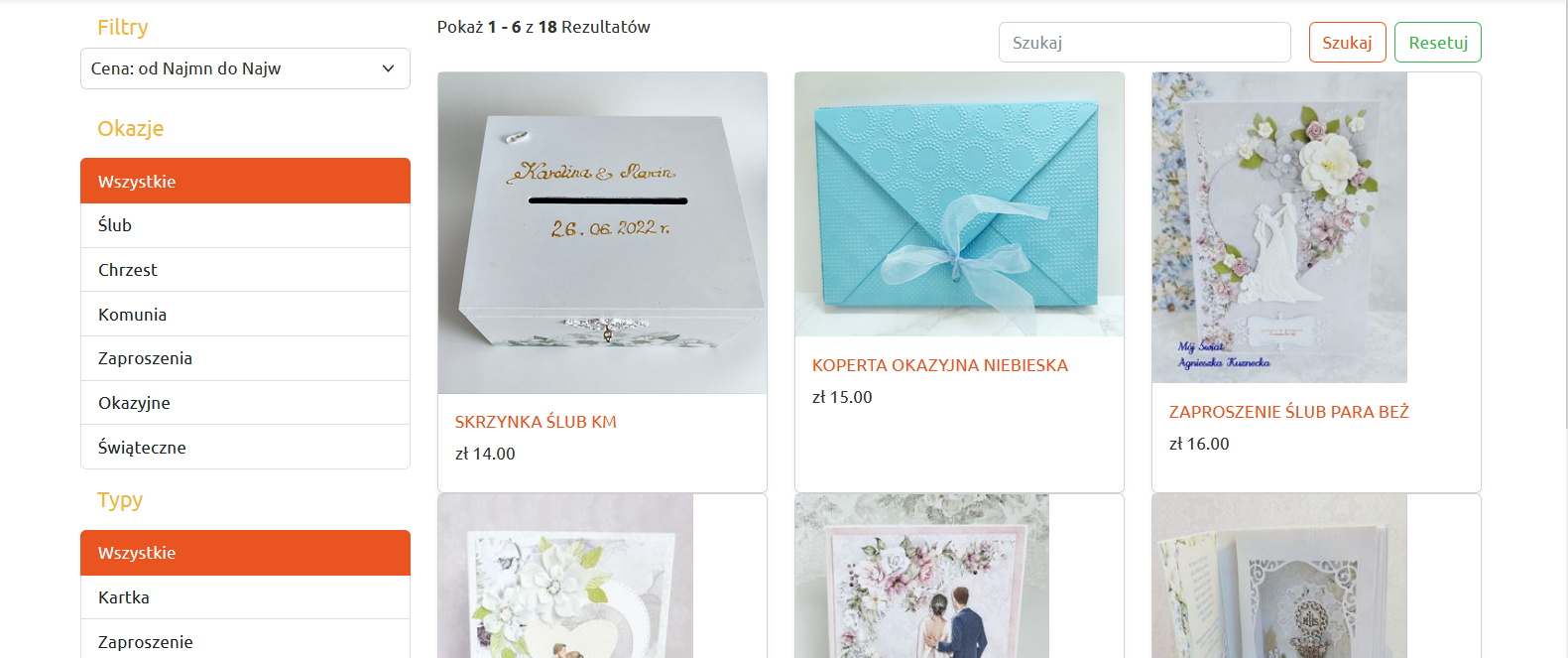
Rys. 14 Widok filtrowania katalogu produktów

Źródło: opracowanie własne



Rys. 15 Widok zmiany kategorii i okazji

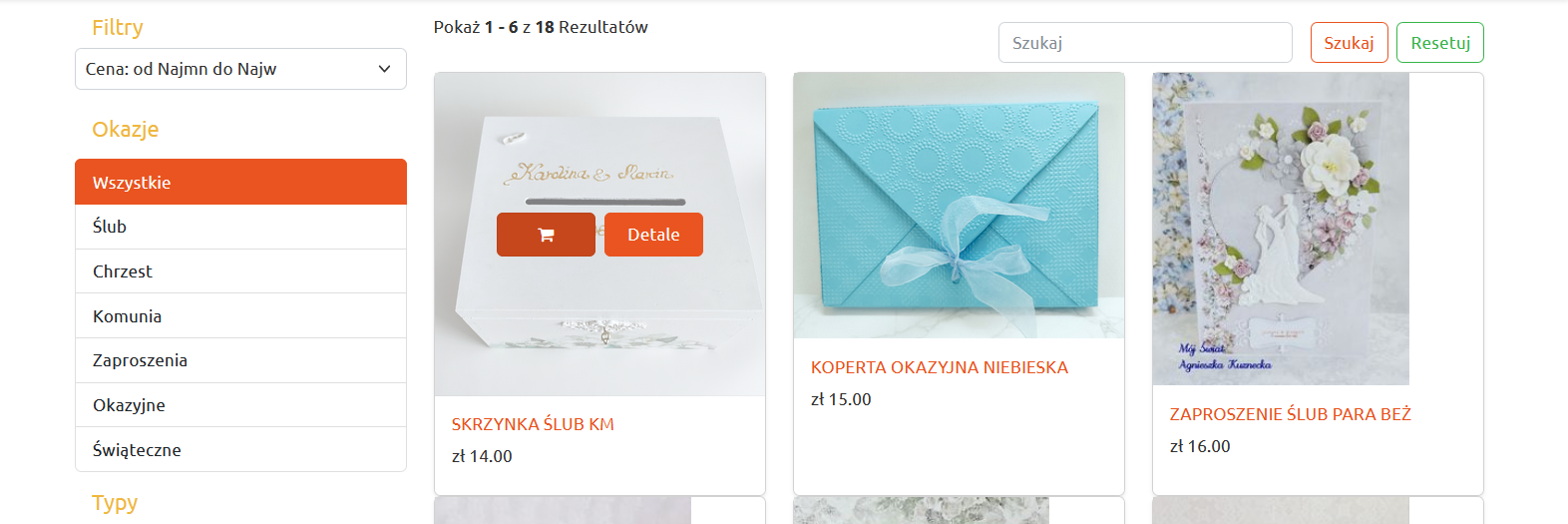
Źródło: opracowanie własne



Rys. 16 Widok posortowanej według ceny listy produktów

Źródło: opracowanie własne

Tak jak zostało to zobrazowane na Rys. 17. Najeżdżając kursorem na produkt użytkownik może przejść do widoku szczegółów lub dodać go prosto do koszyka.



Rys. 17 Widok po najechaniu kursorem na produkt

Źródło: opracowanie własne

Po wciśnięciu przycisku szczegółów, użytkownik zostaje przeniesiony do szczegółowego widoku produktu, gdzie wyświetlane jest jego zdjęcie w większej rozdzielczości oraz jego opis i skąd może dodać go do koszyka w zadowalającej go ilości (Rys. 17).



Rys. 17 Widok po najechaniu kursorem na produkt

Źródło: opracowanie własne

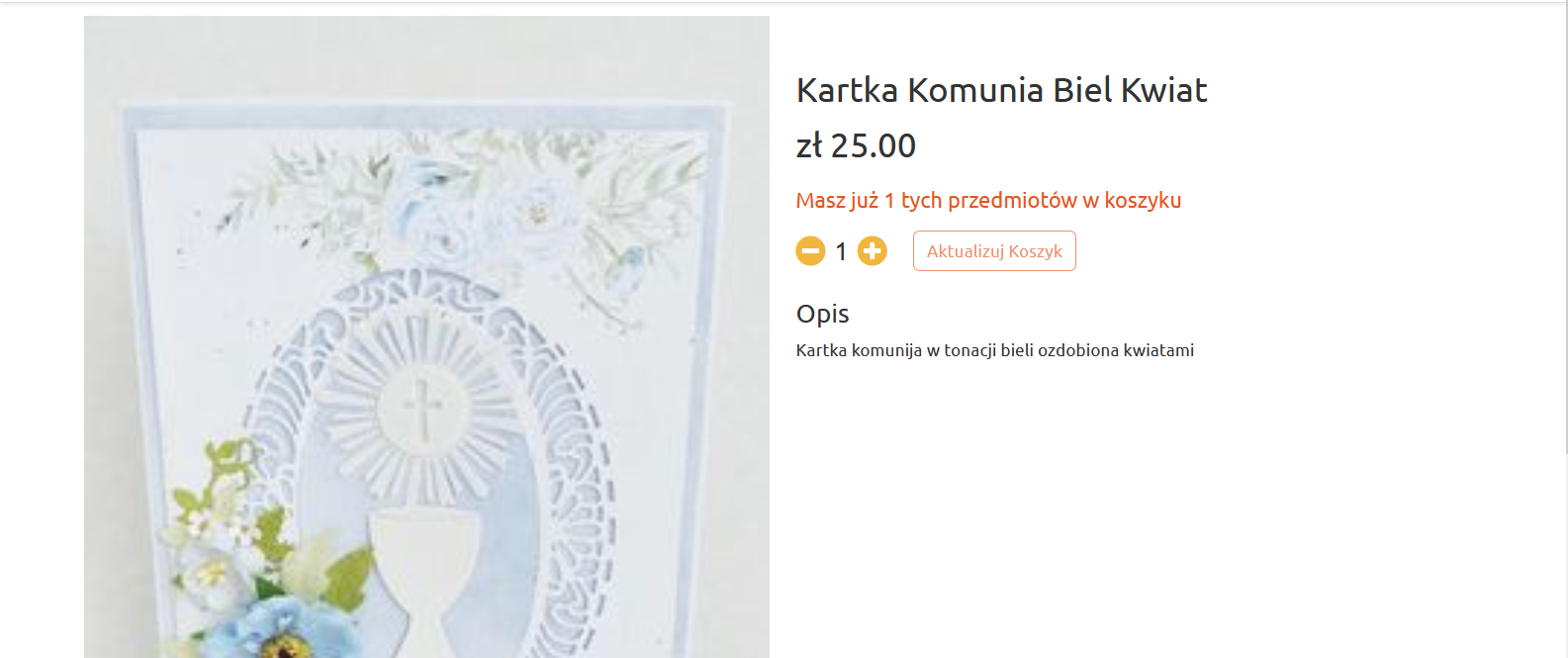
Korzystając z paska nawigacji (Rys. 18) użytkownik może stale kontrolować w jakim miejscu aplikacji obecnie się znajduje i poprzez wciśnięcie komponentu przejść do innego widoku.



Rys. 18 Pasek kontroli nawigacji po aplikacji

Źródło: opracowanie własne

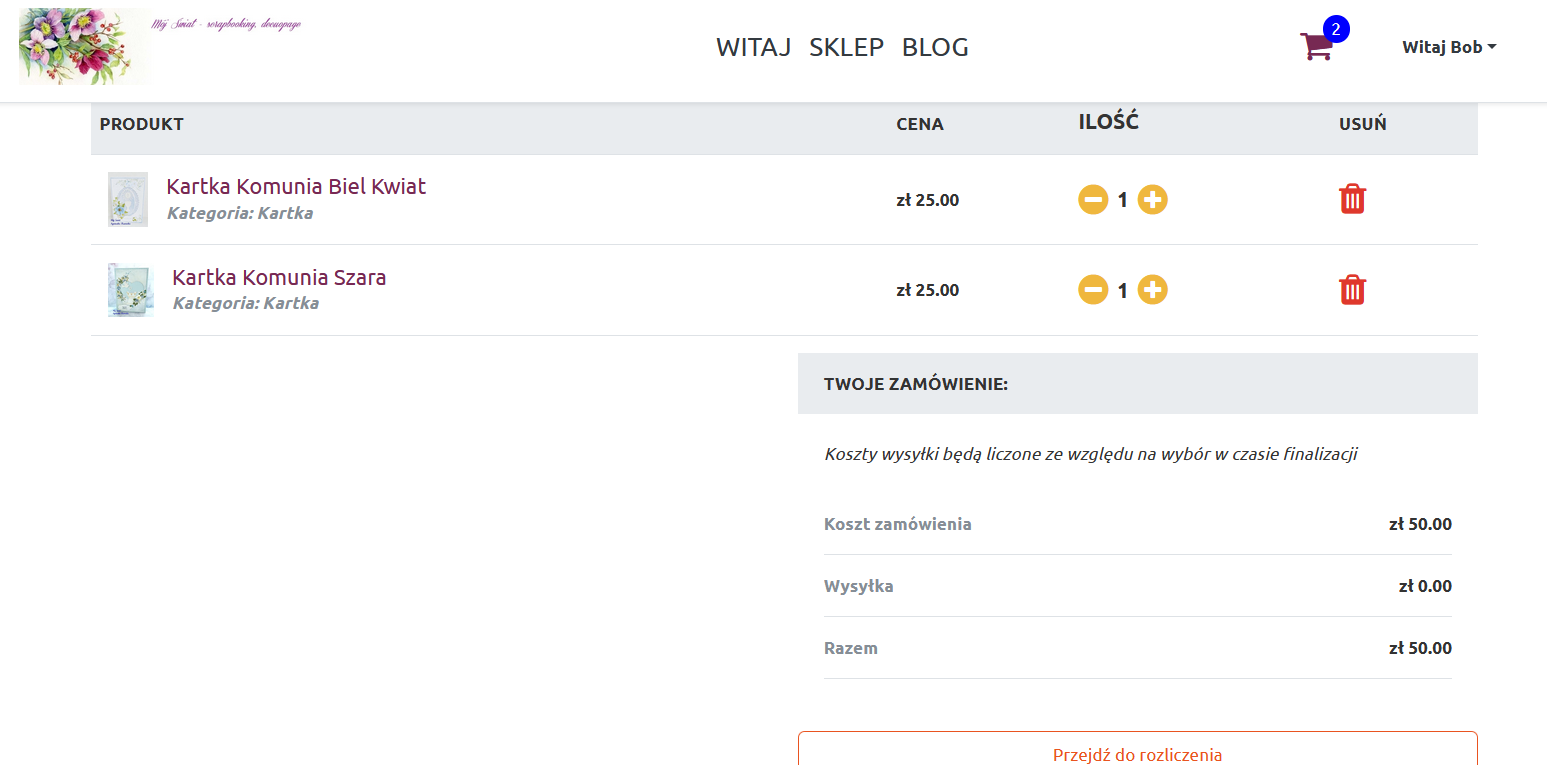
W widoku detalicznym produktu (Rys. 18), jeżeli klient doda produkt do koszyka to otrzyma informacje zwrotną o aktualizacji koszyka co nie przeszkadza mu w razie potrzeby dodania większej ilości tego samego produktu (Rys. 19).



Rys. 19 Aktualizacja koszyka po dodaniu produktu z widoku szczegółowego

Źródło: opracowanie własne

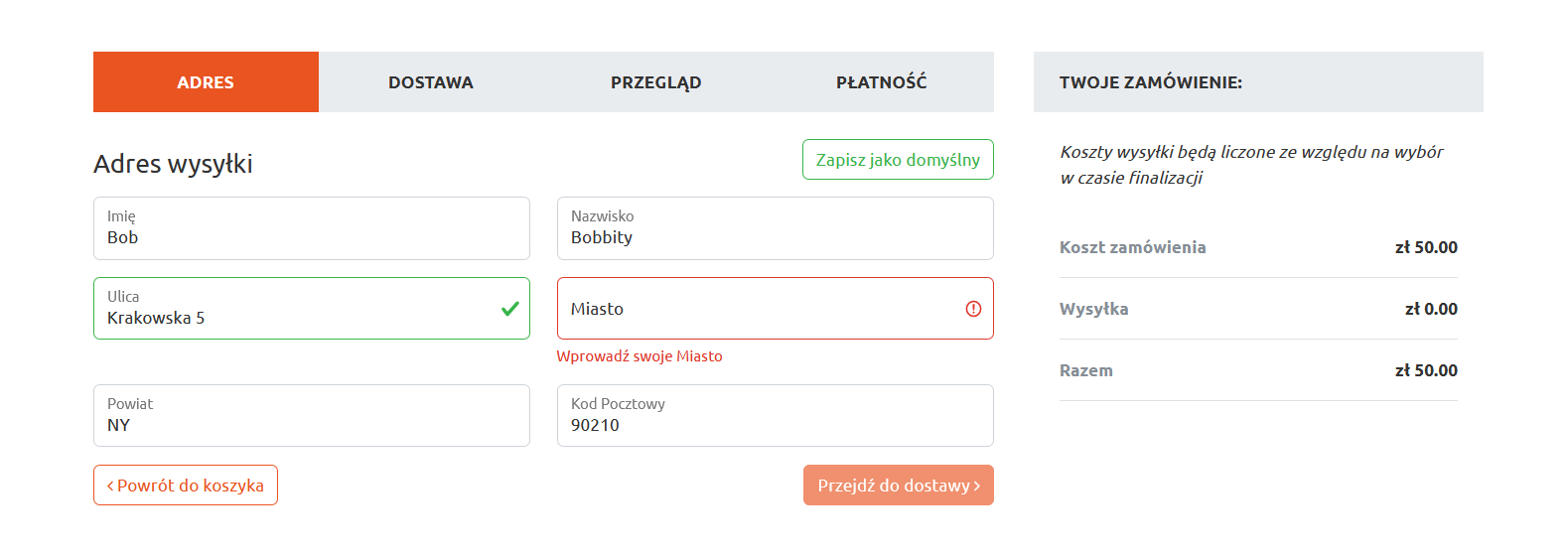
Następnym krokiem dokonania zakupu jest przegląd koszyka (Rys. 20). Użytkownik ma wgląd w zawartość koszyka, w którym znajdują się produkty jakie wybrał, ich cena, ilość którą użytkownik może zwiększyć lub zmniejszyć bez potrzeby powrotu do sekcji sklepu lub całkowicie usunąć produkt z koszyka. Ponadto już w na tym etapie klient ma wgląd w cenę wszystkich produktów razem.



Rys. 20 Widok koszyka klienta

Źródło: opracowanie własne

Jeżeli użytkownik dodał do koszyka wszystkie żądane produkty, to przechodzi przyciskiem do pierwszego widoku rozliczenia (Rys. 20).



Rys. 21 Pierwszy ekran rozliczenia

Źródło: opracowanie własne

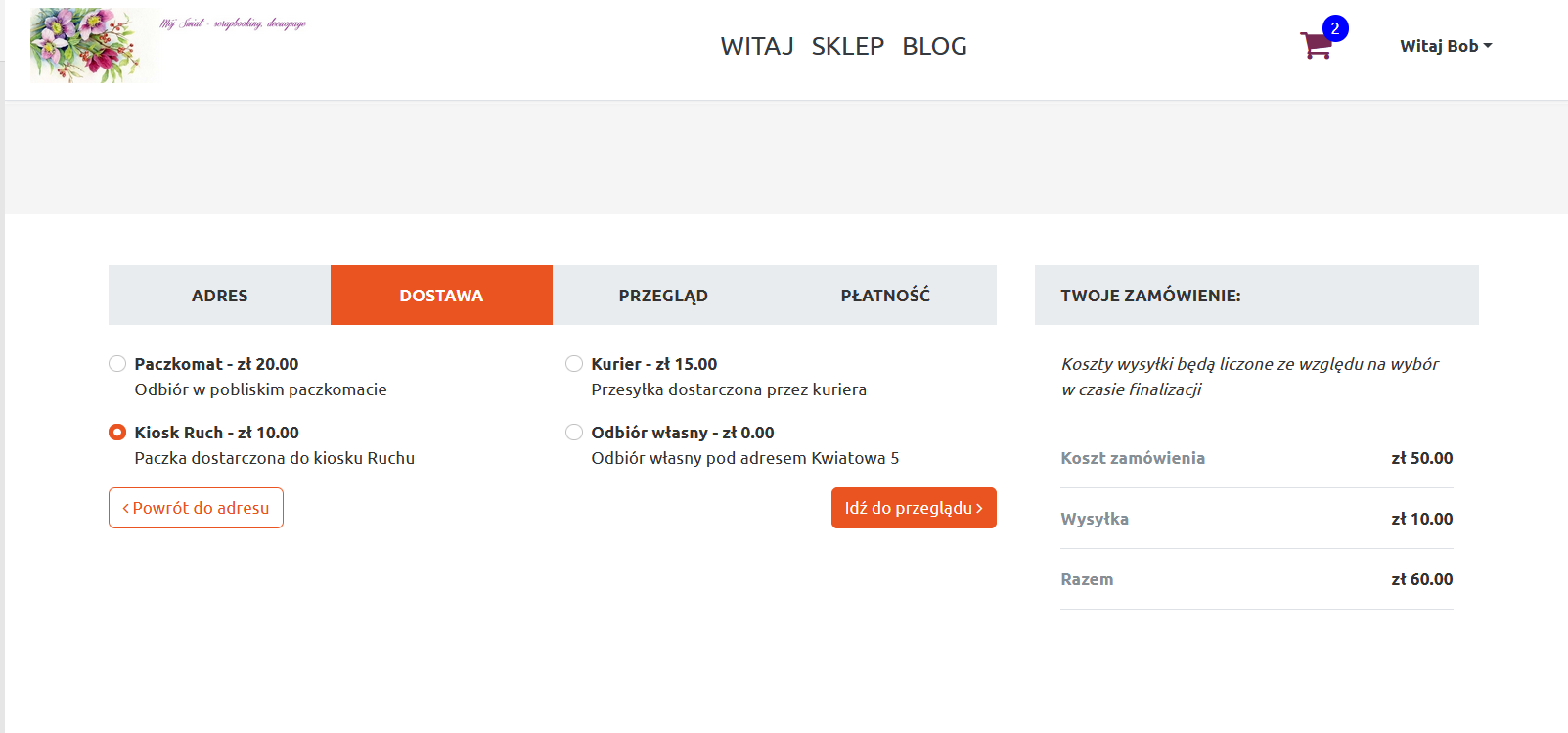
Na Rys. 21 został zobrazowany pierwszy etap składania zamówienia, czyli wprowadzenie adresu. Z  poziomu tego widoku użytkownik wprowadza poprawne dane zamieszkania. W przypadku braku wprowadzenia danych program nie pozwoli użytkownikowi przejść do następnego kroku. Dodatkowo zalogowany użytkownik dostaje dostęp do zapisania adresu jako domyślny.



Rys. 22 Wybór metody dostawy rozliczenia

Źródło: opracowanie własne

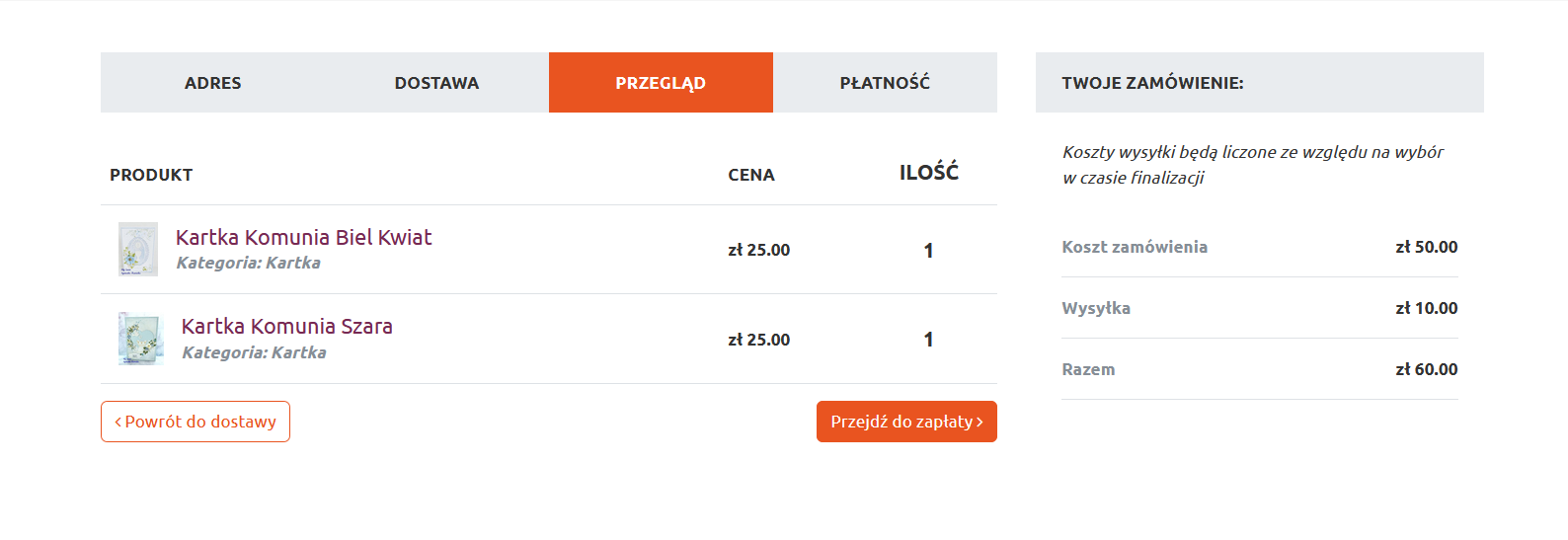
Kolejnym krokiem rozliczenia jest wybór metody dostawy (Rys. 22). Klient ma do wyboru jeden z kilku popularnych metod dostawy w Polsce. Po dokonaniu wyboru koszt tej dostawy doliczony zostaje do kosztu zamówienia tak jak zostało to przedstawione na Rys. 23.



Rys. 23 Poprawny wybór metody dostawy rozliczenia

Źródło: opracowanie własne

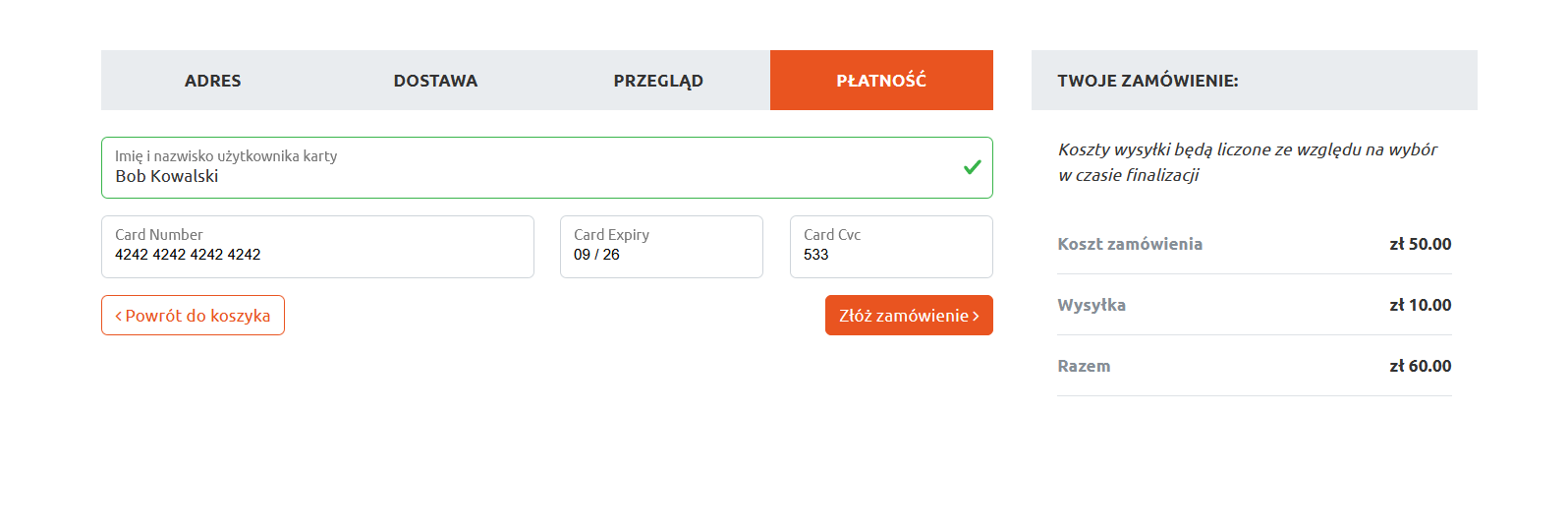
W widoku przeglądu zobrazowanym na Rys. 24 klient ma wgląd w koszyk zakupów, aby zabezpieczyć go przed finalizacją niechcianej transakcji.



Rys. 24 Trzeci etap składania zamówienia

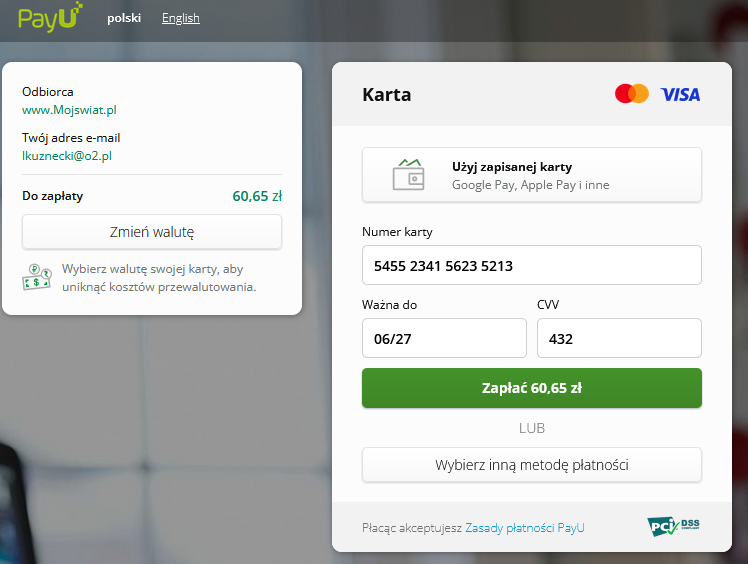
Źródło: opracowanie własne

Ostatnim etapem finalizacji transakcji jest wprowadzenie informacji rozliczeniowych (Rys. 25). Strona oczekuje od użytkownika wpisania poprawnych danych osobowych, numeru karty, daty wygaśnięcia karty oraz kod Cvc. Jeżeli ta operacja zakończy się sukcesem użytkownik przekierowywany jest na zaufaną stronę pośrednią PayU gdzie najczęściej przed dokonaniem zapłaty następuje weryfikacja dwuetapowa (Rys.26).



Rys. 25 Wprowadzanie danych płatniczych przez klienta

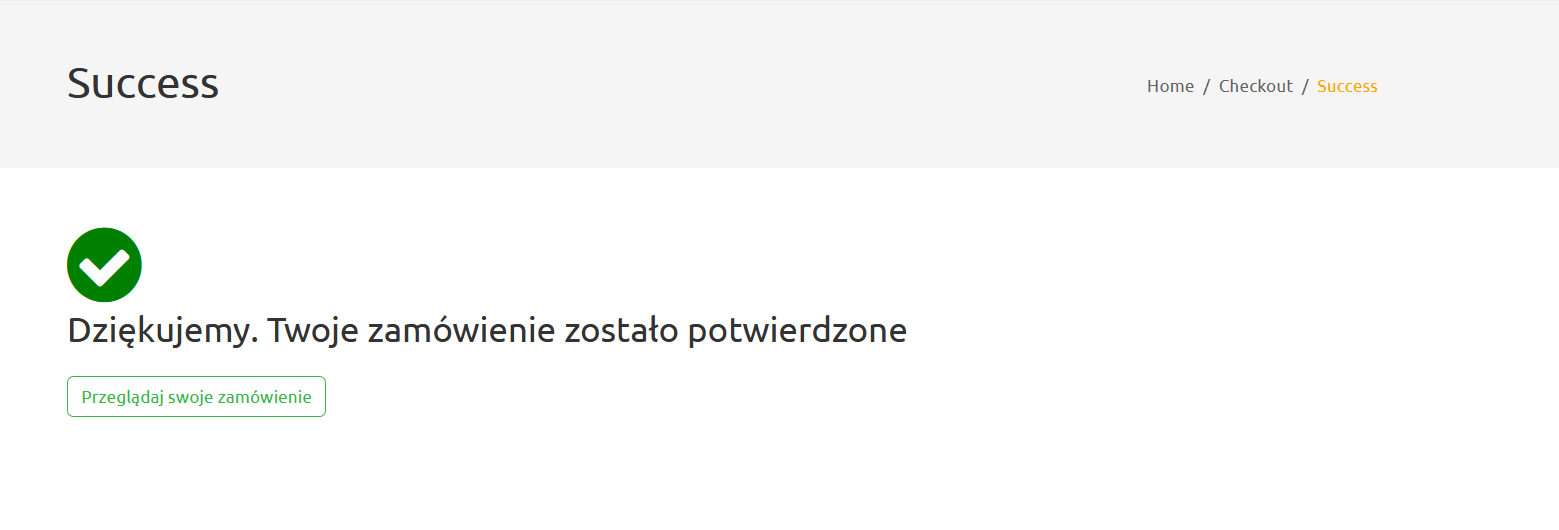
Źródło: opracowanie własne



Rys. 26 Prototyp rozszerzenie metod płatności klienta

Źródło: https://secure.payu.com/pay, z dnia 20.02.2023

Po udanym dokonaniu płatności klient otrzymuje wiadomość zwrotną wraz z przyciskiem powrotu do szczegółów swojego zamówienia (Rys. 27).



Rys. 27 Trzeci etap składania zamówienia

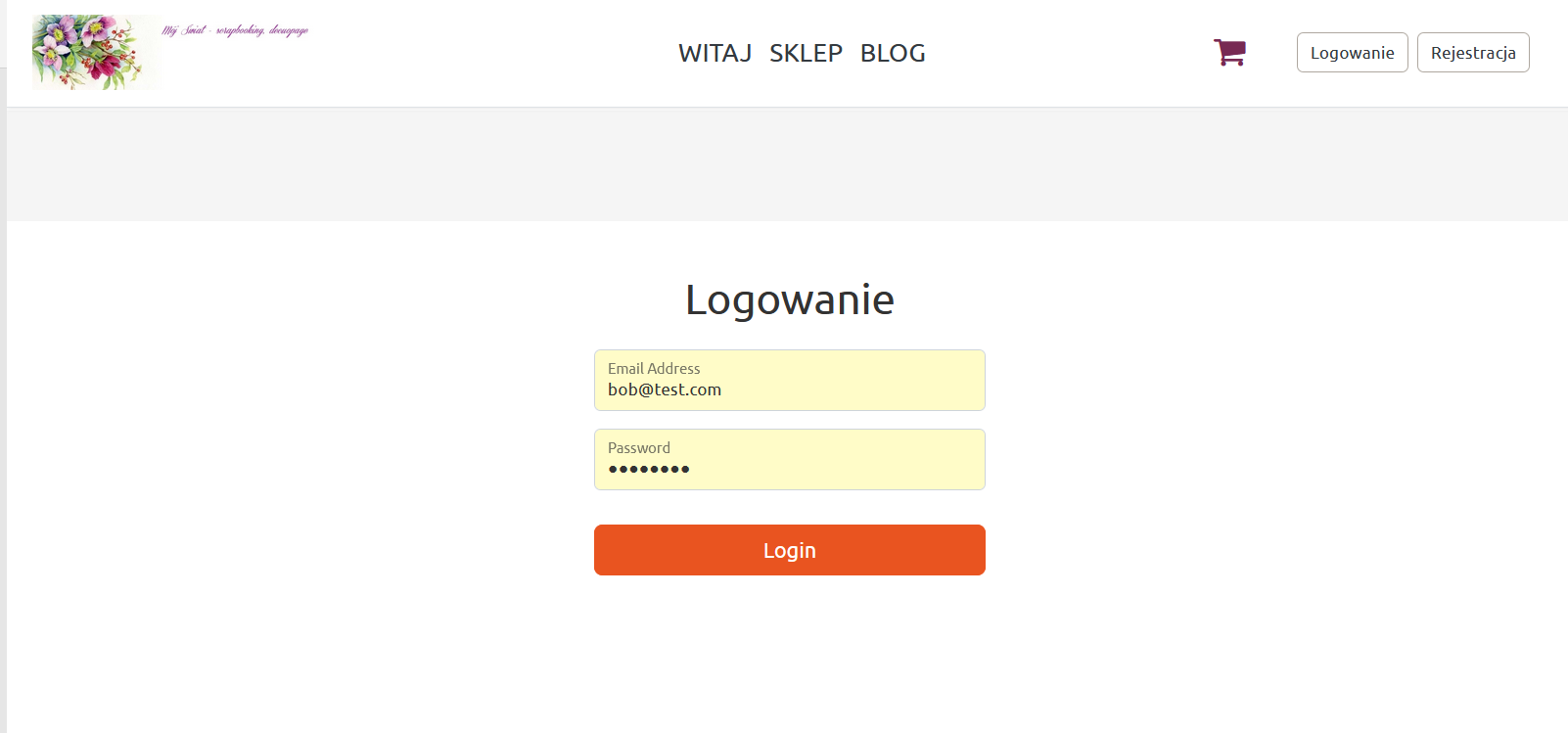
Źródło: opracowanie własne

Na podstawie danych z przeprowadzonej ankiety rejestracja (Rys.28) i logowanie (Rys. 29) użytkownika jest opcjonalna, gdyż rezultaty wykazały, że proces rejestracji i weryfikacji użytkownika część osób uważa za zbędny. Dzięki temu potencjalny klient może szybko i wygodnie dokonać pierwszego zakupu i sam zadecydować, czy chce zostać stałym klientem tego sklepu.



Rys. 28 Formularz rejestracji użytkownika

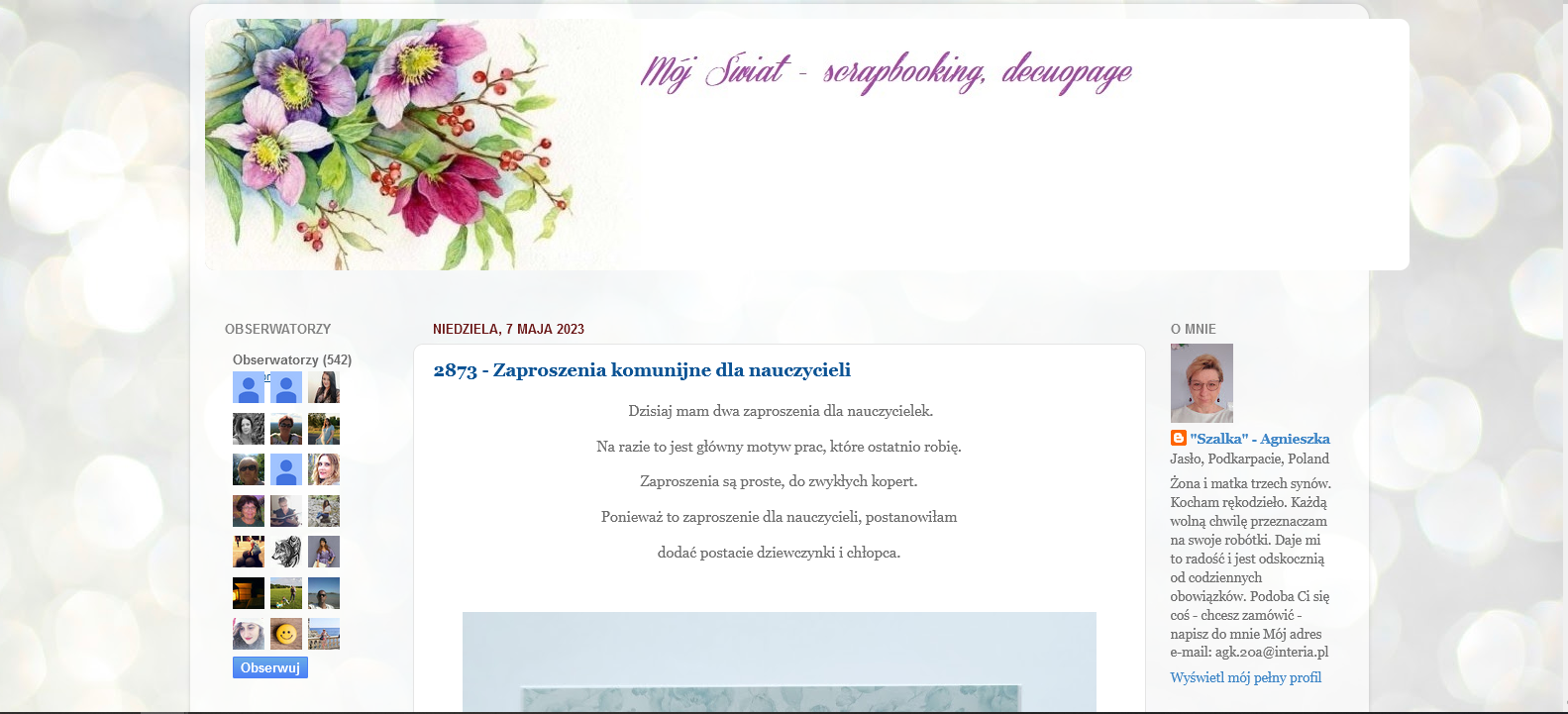
Źródło: opracowanie własne



Rys. 29 Ekran logowania użytkownika

Źródło: opracowanie własne

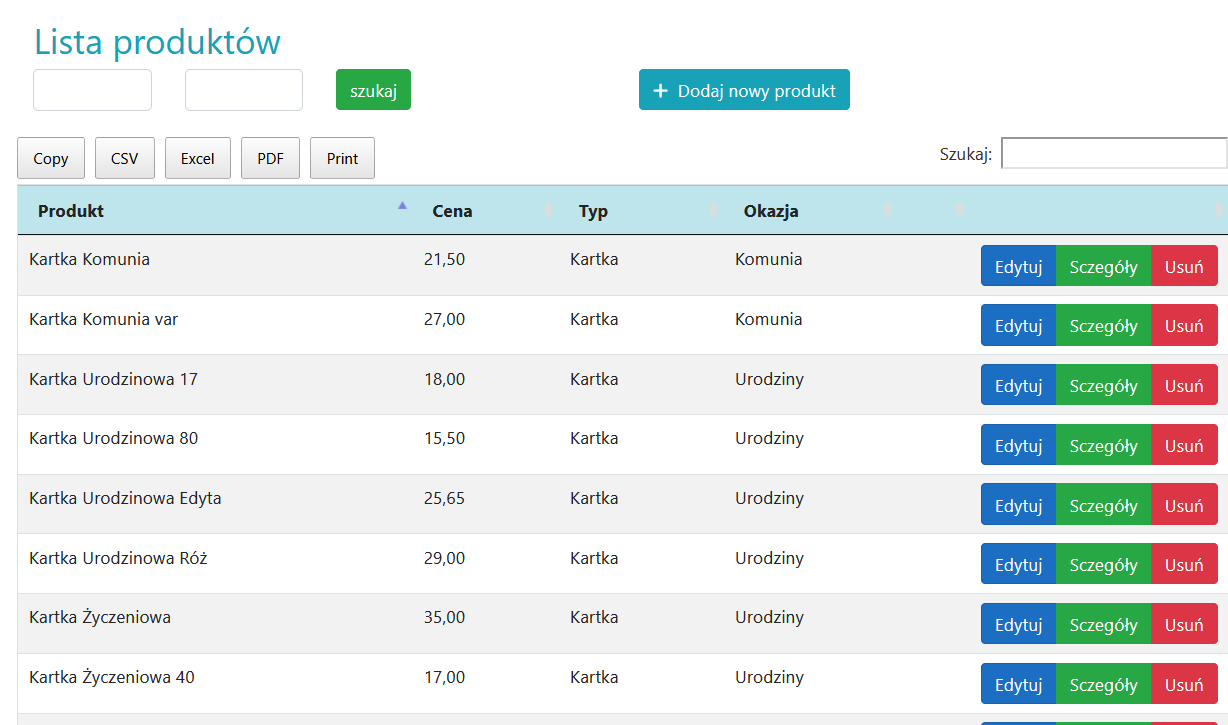
Ostatnią funkcjonalnością do jakiej dostęp ma użytkownik jest zakładka blog z paska nawigacji. Znajduje się tam o wiele większa próbka udokumentowanych produktów, kategorii i typów (Rys. 30). Jest to szczególnie pomocne dla klienta, który poszukuje spersonalizowanego produktu.



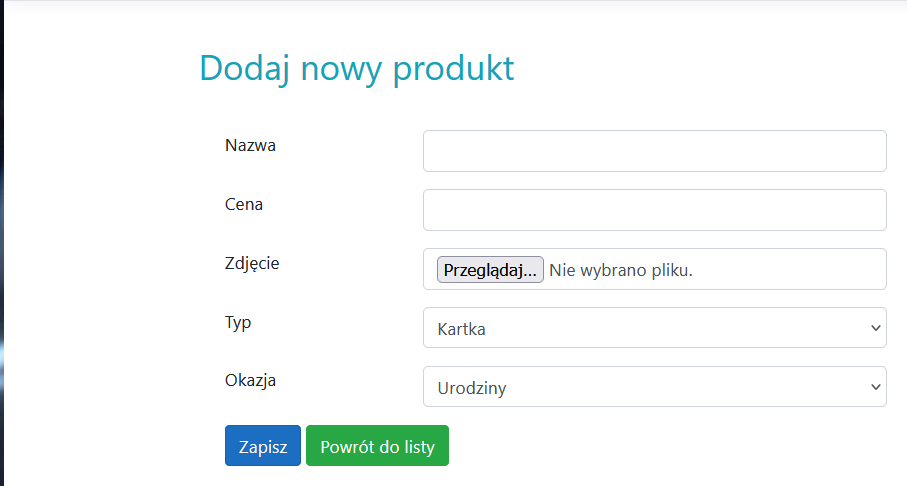
Rys. 30 Strona bloga artystycznego

Źródło: opracowanie własne

Po zalogowaniu się jak administrator aplikacja udostępnia twórcy widoki zarządzania zasobami strony takimi jak operacje CRUD na produktach, kategoriach, typach tak jak przedstawiono na Rys. 31 i Rys. 32.

 Rys. 31 Ekran logowania użytkownika

Źródło: opracowanie własne



Rys. 32 Dodawanie nowego produktu przez administratora

Źródło: opracowanie własne

# **Testy użyteczności**

Użyteczność jest jednym z niefunkcjonalnych atrybutów oprogramowania. Działania osoby testującej program lub aplikacje skupiają się na sprawdzaniu czy wytworzony produkt spełnia określone w wymaganiach cele i założenia. W zakresie IT produkty które najczęściej przechodzą testy użyteczności to: interfejsy komputerowe, urządzenia, aplikacje, strony internetowe. Aby poprawnie zrozumieć użyteczność należy zbadać aplikację pod kątem zdarzeń i problemów z jakimi użytkownicy mogą się spotkać podczas korzystania z oprogramowania. Jako użytkownika należy wziąć pod uwagę jego cechy takie jak: wiek czy zakres obycia z podobnym oprogramowaniem. Należy więc założyć, że tester jest potencjalnym użytkownikiem, który w tym przypadku chce dokonać zakupu w sklepie internetowym. W wyniku uzyskuje się od niego opinię oraz informacje zwrotne na temat rzeczy dla niego kłopotliwych lub niezrozumiałych.

Przygotowanie testu użyteczności polega na starannym stworzeniu realistycznego scenariusza lub prawdopodobnej sytuacji, w której osoba wykonuje listę kroków przy użyciu wersji testowej produktu, podczas gdy osoba odpowiedzialna za nadzór nad testami obserwuje i sporządza notatki i obserwacje. Celem tych badań jest wykrywanie nieznanych dotąd problemów i dostrzegać relacje między nimi.

Podczas testu sprawdza się kolejne niefunkcjonalne wymaganie, czyli łatwość dostępu, który bada jak łatwo użytkownicy mogą nauczyć się nawigować po programie w celu osiągniecia konkretnego celu. Testowanie użyteczności ma mierzyć następujące cechy:

* Skuteczność – możliwość osiągnięcia przez użytkownika przy użyciu systemu konkretnego celu z zachowaniem dokładności w określonym środowisku użycia,
* Efektywność – możliwość uzyskania przez użytkownika określonej efektywności przy odpowiednich nakładach wykorzystanych zasobów,
* Satysfakcję – możliwość zadowolenia użytkownika z produktu w określonym kontekście użycia.

Cechy charakterystyczne jakie można mierzyć to:

* Zrozumiałość – atrybuty aplikacji, które wpływają na nakład czasu wymagany do zrozumienia przez użytkownika mechanizmu logicznego produktu i jego zastosowania do danego scenariusza,
* Łatwość nauki – atrybuty aplikacji, które wpływają na nakład czasu użytkownika wymagany do opanowania działania aplikacji,
* Łatwość obsługi - atrybuty aplikacji, które wpływają na nakład czasu wymaganego do efektywnego nawigowania użytkownika po oprogramowaniu,
* Atrakcyjność – możliwość chętnego powrotu do używania aplikacji przez użytkownika.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Zadanie | Funkcjonalności | U1 -Laptop | U2-Komputer Stacjonarny | U3-Komputer stacjonarny | U4-Telefon |
| Otwórz trzy dowolne kategorie | Zmiana kategorii | ✓ | ✓ | ✓ | ✓ |
| Zarejestruj się | Rejestracja | ✓ | ✓ | ✓ | ✓ |
| Zaloguj się | Logowanie | ✓ | ✓ | ✓ | ✓ |
| Przełącz podstronę sklepu na blog | Zmiana podstrony na Blog | ✓ | ✓ | ✓ | ✓ |
| Przenieś do koszyka trzy dowolne produkty | Obsługa koszyka | ✓ | ✓ | ✓ | ✓ |
| Usuń dowolny produkt dodany wcześniej z koszyka | Obsługa koszyka | ✓ | ✓ | ✓ | ✓ |
| Wyświetl produkty w zakresie cenowym od 10 do 50 zł | Filtracja | ✓ | ✓ | ✓ | X |
| Dokonaj płatności za zakup | Przekierowywanie | ✓ | ✓ | ✓ | ✓ |

Tabela 4. Testy użyteczności aplikacji

Źródło: opracowanie własne

Testy użyteczności (Tabela 4) zostały przeprowadzone na czterech użytkownikach dwóch z nich nie było starsze niż 50 lat jeden miał 16 lat a ostatni 25. Warto podkreślić, że dwóch użytkowników korzystało z  komputerów stacjonarnych z różnymi ekranami, jedna osoba korzystała z laptopa i najmłodsza korzystała z telefonu z systemem operacyjnym android.

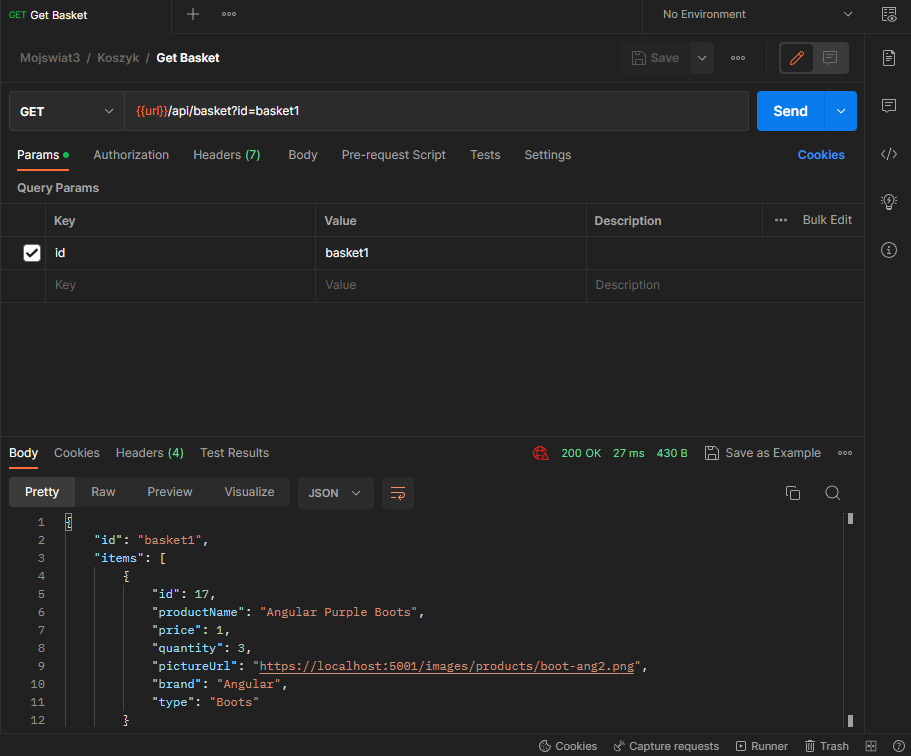
Każdemu z użytkowników udało się wypełnić scenariusz aż do punktu końcowego bez zadawania pytań prowadzącemu testy o wykonanie kroku w danej sytuacji z jednym wyjątkiem dotyczącym filtrowania listy produktów. Problem pojawił się z responsywnością strony i wystąpił z poziomu testowania za pomocą telefonu, był to jedyny problem nie wynikający z braku intuicyjności a braku zapewnienia responsywności względem telefonu. To co można było zaobserwować podczas przeprowadzania testów to akceptowalna wydajność odpowiedzi ze strony serwera. Wszyscy testowi użytkownicy posiadali mniejsze lub większe doświadczenie w nawigowaniu po sklepie internetowym więc nie wystąpiły problemy z szukaniem produktów czy zmianą kategorii. Bardzo sprawnie przebiegało przechodzenie pomiędzy kolejnymi etapami rejestracji i logowania zarówno z jednostek stacjonarnych jak i mobilnej.

Jedynym problemem pozostaje rozwiązać kwestie responsywności witryny z dużym, lecz niekrytycznym priorytetem na tym etapie testów.

1. **Testy funkcjonalne**

W procesie tworzenia aplikacji internetowych testowanie w czasie rzeczywistym jest jednym z najważniejszych aspektów kontroli poprawnego działania programu. Do przeprowadzania testów wykorzystany został program Postman, który oferuje automatyzacje testów funkcjonalnych API i aplikacji webowej. Rodzaje testów jakie zostały przeprowadzone to:

* Testy integracyjne API – pozwalają przetestować, czy API działa poprawnie czy otrzymywane odpowiedzi przy wysyłaniu zapytań HTTP zwaracają odpowiednie dane i statusy odpowiedzi. Przykładowe testy integracyjne zostały przeprowadzone tak jak na Rys. 33 i Rys 34.



Rys. 33 Test zapytania GET zwracającego zawartość koszyka

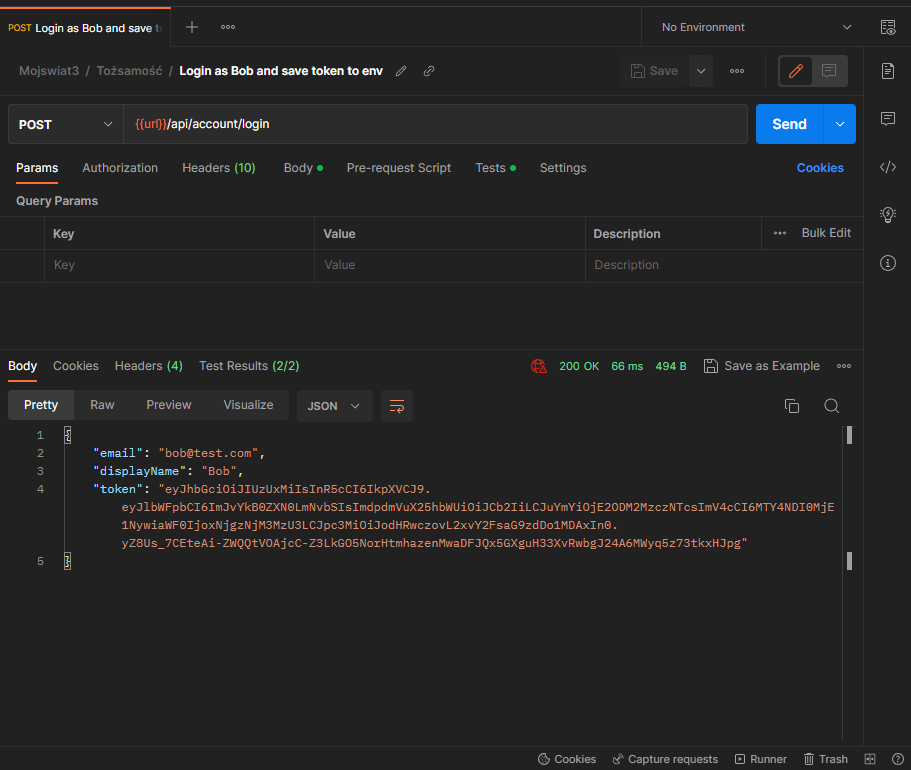
Źródło: opracowanie własne



Rys. 34 Test zapytania GET który zwraca katalog produktów komunijnych

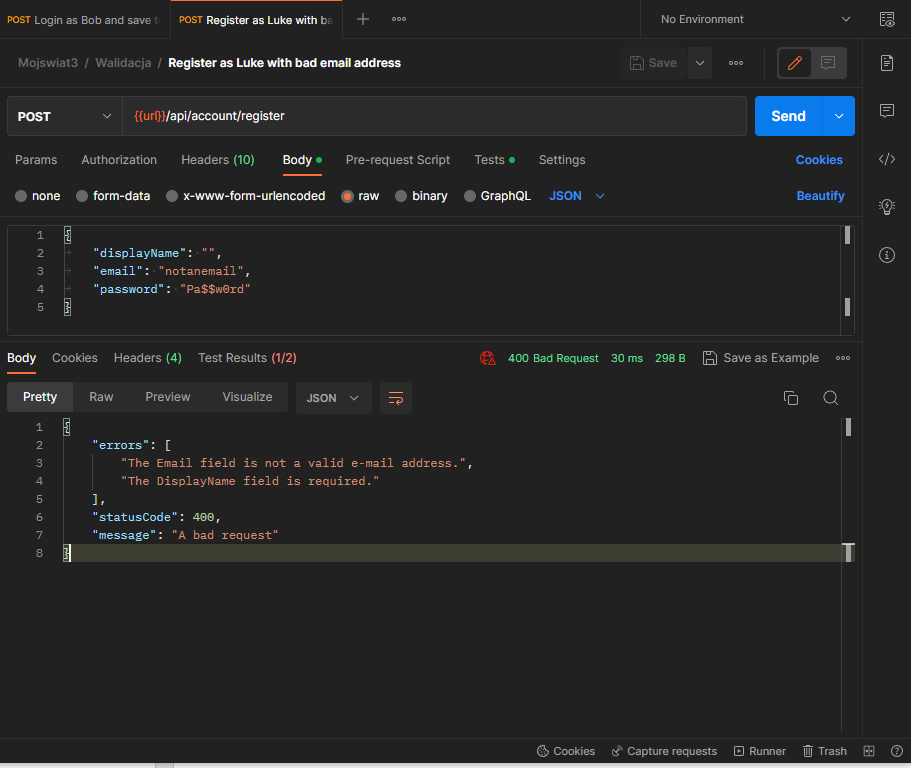
Źródło: opracowanie własne

* Testy weryfikacji poprawności danych – poprzez ten typ testów można sprawdzić poprawność wprowadzanych danych i czy spełniają one standardy aplikacji. W tym przypadku pierwszy test (Rys. 35) ma za zadanie sprawdzić poprawność danych logowania i jeżeli ta operacja zakończy się sukcesem przypisanie mu szyfrowanego tokenu. Drugi test jest testem walidacyjnym sprawdzającym poprawność danych rejestracji (Rys. 36)



Rys. 35 Test zapytania POST, który testuje funkcjonalność logowania się użytkownika

Źródło: opracowanie własne



Rys. 36 Test zapytania POST, który testuje walidacje rejestracji użytkownika

Źródło: opracowanie własne

# **Zakończenie**

Celem niniejszej pracy inżynierskiej było stworzenie aplikacji internetowej, która umożliwiałaby użytkownikowi dokonać zakupu w sklepie internetowym w sposób szybki i intuicyjny, a także zachęcający potencjalnego klienta do ponownego skorzystania z serwisu. Jest ona dostępna z poziomu każdej przeglądarki i systemu operacyjnego. Oferuje ona funkcjonalności takie jak: przeglądanie produktów, logowanie, rejestracja, realizacja transakcji. Z poziomu administratora funkcjonalność jest rozszerzona o:  dodawanie, usuwanie i edycje produktów, okazji, typów produktów oraz tą samą funkcjonalność w  zakresie zarządzania użytkownikami.

Wszystkie procesy zarządzania danymi są przechowywane i w przypadku danych poufnych szyfrowane po stronie serwerowej. To tam następuje autoryzacja i autentykacja użytkowników i na podstawie danych przesłanych do niej udostępniane są funkcjonalności dla użytkownika końcowego.

System został zbudowany z myślą o dalszym rozwoju jego funkcjonalności. Pozyskiwanie umiejętności w tej dziedzinie może okazać się kluczowe w pracy przyszłego programisty. Praca ta nie wyczerpuje w pełni tematu, gdyż jest wiele aspektów, w których można poprawić działanie i obsługę aplikacji zarówno z poziomu użytkownika jak i administratora. Rozwój może nastąpić w wielu kierunkach, z których przede wszystkim to interfejs graficzny użytkownika aplikacji oraz funkcji w aplikacji i działanie. Co wydaje się nieuniknionym etapem ewolucji sklepu to rosnąca popularność aplikacji mobilnych więc w  efekcie stworzenie odpowiadającej aplikacji mobilnej do aplikacji internetowej.

# **Literatura**

1. Mark Michaelis, Eric Lippert., Essential C# 7.0 (6th Edition), Microsoft Press, 2018
2. Mithun Pattankar, Malendra Hurbuns., Mastering ASP.NET Web API, Packt, 2017
3. Julia Lerman, Rowan Miller., Programming Entity Framework: Code First, O’Reilly, Listopad 2011
4. Nathan Murray, Felipe Coury, Ari Lerner, Carlos Taborda., ng-book: The Complete Guide to Angular, Fullstack.io, 2018
5. https://www.pwc.pl/pl/media/2022/2022-07-11-do-2027-wartosc-rynku-e-commerce-w-polsce-wzrosnie-o-ponad-94-mld-zl-do-187-mld-zl.html (08.07.2022)
6. https://ecodeco.pl/ (24.07.2022)
7. https://www.decobazaar.com/ (04.01.2023)
8. https://www.pakamera.pl/ (07.01.2023)
9. https://docs.google.com/forms/d/e/1FAIpQLSdBRkdgN-joz1AFr0QeKNKO9lvt0YFl-kqwBP\_F5IkL4dD7Iw/viewform?usp=sf\_link (04.01.2023)
10. https://thestory.is/pl/proces/faza-wdrozenia/wymagania-funkcjonalne (25.01.2023)
11. https://www.commint.pl/baza/wymagania-niefunkcjonalne-aplikacji-internetowej (28.01.2023)
12. https://dotnet.microsoft.com/en-us/apps/aspnet/apis (22.01.2023)
13. https://learn.microsoft.com/en-us/aspnet/core/tutorials/first-web-api?view=aspnetcore-7.0&tabs=visual-studio (23.01.2023)
14. https://jakzostacprogramista.net/2020/11/15/wprowadzenie-do-asp-net-core-web-api
15. Andrzej Jaszkiewicz., Inżyneria oprogramowania, informatyka, matematyka, 1997
16. https://www.plukasiewicz.net/Angular/IntroductionI (04.12.2022)
17. https://www.sqlitetutorial.net (03.02.2023)
18. https://www.entityframeworktutorial.net/ (10.01.2023)
19. https://www.postman.com/ (21.12.2022)
20. https://appmaster.io/pl/blog/przeglad-kontenerow-docker (15.11.2022)
21. https://domenomania.pl/centrum-wiedzy/redis-co-to-i-do-czego-sluzy (19.03.2023)
22. Dmitri Nesteruk., Design Patterns in .NET: Reusable Approaches in C#  
    and F# for Object-Oriented Software Design, Helion, 2020
23. https://enterprisecraftsmanship.com/posts/specification-pattern-c-implementation (04.03.2023)
24. https://dotnettutorials.net/lesson/unit-of-work-csharp-mvc (09.01.2023)
25. https://regexr.com/ (03.03.2022)
26. https://jwt.io/ (29.04.2023)

# **Streszczenie**

**Wyższa Szkoła Informatyki i Zarządzania z siedzibą w Rzeszowie**

**Kolegium Informatyki Stosowanej**

**Streszczenie pracy dyplomowej**

Aplikacja e-commerce artykułów ozdobnych

**Autor**: Łukasz Kuznecki

**Promotor**: Dr inż. Teresa Mroczek

**Słowa kluczowe**: Aplikacja internetowa, Angular, .NET

Rezultatem pracy jest aplikacja internetowa przeznaczona dla osób poszukujących inspiracji na prezent lub szukających gustownej ozdoby na półkę. Zakres jej funkcjonalności został zaimplementowany w oparciu o analizę rynkową i dialog z klientem, którym jest indywidualny artysta. Część aplikacji odpowiadającej za przetwarzanie danych i logikę biznesową została zaprojektowana w technologii .NET według metodologii REST API. Środowisko klienta oraz jego wizualny interfejs został zaimplementowany na bazie frameworka Angular, skąd klient wysyła zapytania o zasoby.