Aplikacja optymalizująca zamówienie

Jan Borowski, Filip Chrzuszcz, Piotr Fic, Elżbieta Jowik, Mikołaj Jakubowski

1 lutego 2020

Spis treści

1 Opis problemu

Tytuł projektu: Aplikacja mobilna optymalizująca zamówienie z restauracji typu fast-food.

1.1 Opis biznesowy

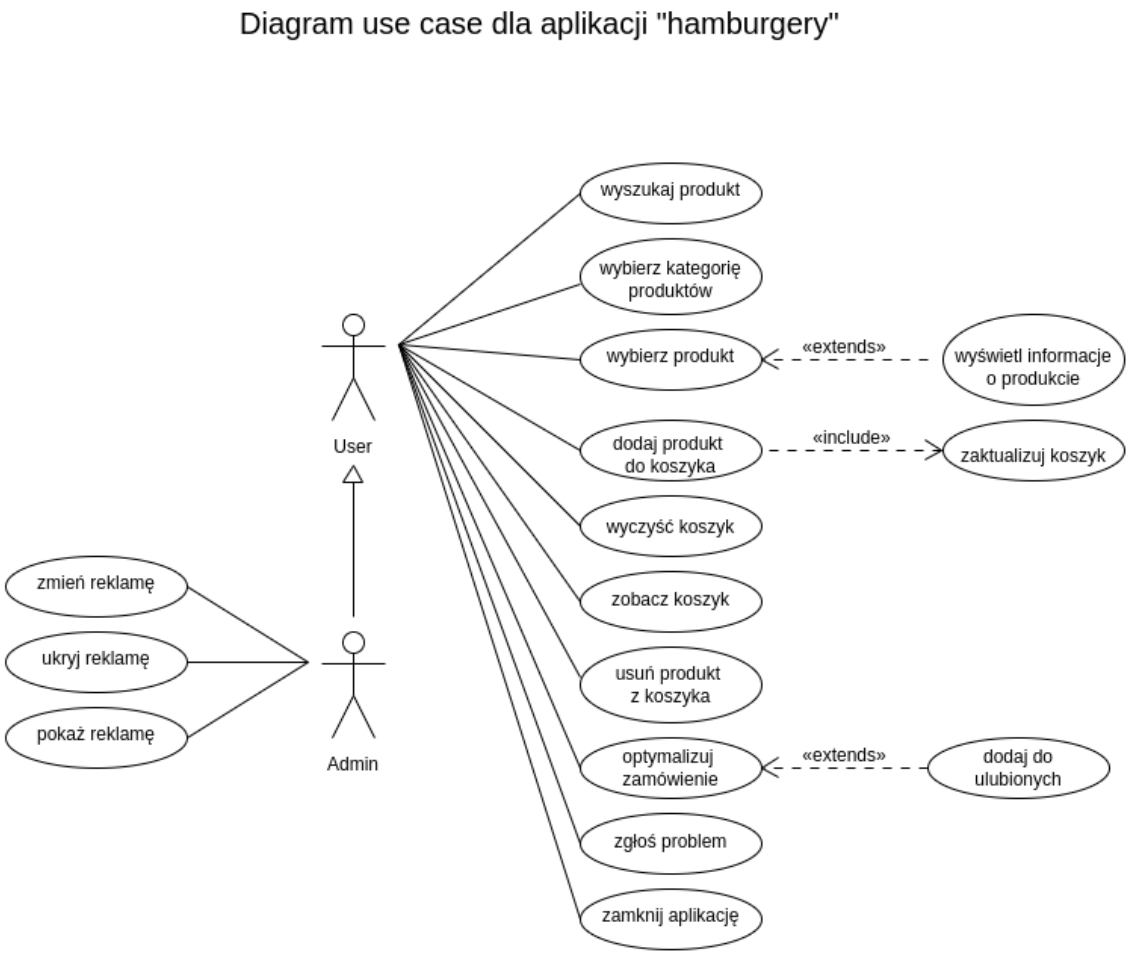
W popularnych sieciach restauracji typu fast-food istnieje wiele sposobów na kupno różnych kombinacji tych samych produktów: osobno, w zestawach, w dodatkach, z użyciem kuponów, itp. Wielu klientów niezorientowanych w ofercie danej restauracji niepotrzebnie traci pieniądze, decydując się na konkretne produkty w nieoptymalnych konfiguracjach. Pomocą służy aplikacja mobilna do optymalizacji zamówień w sposób przyjazny użytkownikowi. Jest to ogromne ułatwienie zarówno dla nowych, jak i stałych klientów. Aplikacja zawiera ofertę czołowej sieci restauracji, uwzględnia oferty sezonowe, kupony, a także oferty dostępne tylko danego dnia tygodnia, co pozwala na kupno posiłku w najlepszej możliwej cenie. Procedura optymalizacji z punktu widzenia użytkownika jest bardzo prosta. Wystarczy, że za pomocą menu i wyszukiwarki klient odnajdzie produkty, które go interesują, określi ich ilość i doda do swojego koszyka. Następnie kliknie guzik 'Optymalizuj', a aplikacja podpowie, w jakiej konfiguracji kupić wszystko to, co znajduje się w koszyku w najniższej cenie.
(W celu optymalizacji, aplikacja może proponować zestawy zawierające produkty z poza koszyka).
Dzięki stałemu połączeniu z bazą danych oferta produktów w aplikacji jest zawsze aktualna. Aplikacja ma służyć optymalizacji zamówień, a nie ich dokonywaniu, w związku z tym nie nalicza od nich prowizji, dzięki czemu z punktu widzenia użytkownika jest całkowicie bezpłatna. Wszystkie, przewidywane wpływy finansowe pozyskiwane będą z reklam umieszczanych w specjalnie wydzielonych miejscach w oknie aplikacji. Będą one ulokowane w taki sposób, aby nie interferować z użytkownikiem.

1.2 Słownik pojęć

- **Aplikacja** – narzędzie do zoptymalizowania zamówienia klienta.
- **Agent** – użytkownik aplikacji, chcący zoptymalizować swoje zamówienie.
- **Zamówienie** – produkty, których zakup agent chce zoptymalizować.
- **Algorytm** – sposób, w jaki aplikacja wie, jak zoptymalizować zamówienie agenta.
- **Produkt** – rzecz, którą agent może zamówić poprzez aplikację.
- **Baza danych** – miejsce, skąd aplikacja pobiera dane o promocjach i ofertach z restauracji.
- **Restauracja** - miejsce, gdzie klient może w prawdziwym życiu zamówić swoje zamówienie zoptymalizowane w naszej aplikacji.
- **Oferta promocyjna** - przecenione produkty dostępne w restauracji, a także do zoptymalizowania w naszej aplikacji.
- **GUI** - warstwa, za pośrednictwem której agent korzysta z aplikacji.

1.3 Wymagania funkcjonalne

1.3.1 Diagram use-case



Powyższy diagram use-case przedstawia możliwe interakcje użytkownika z aplikacją. Dodatkowo wyróżniamy możliwości dostępne dla administratora aplikacji.

1.3.2 Historie użycia

Przypadek użycia 1: optymalizuj zamówienie

Podstawowe użycia:

- Optymalizacja zamówienia

Podstawowy użytkownik:

- Użytkownik aplikacji (potencjalny klient restauracji)

Scenariusz główny:

1. Aplikacja wyświetla menu,
2. Użytkownik wybiera kategorie produktu,
3. Aplikacja wyświetla produkty zadanej kategorii,
4. Użytkownik wybiera produkt,
5. Aplikacja wyświetla informacje o produkcie,
6. Użytkownik dodaje produkt do koszyka,
7. Aplikacja aktualizuje stan koszyka,
8. Użytkownik wybiera opcję "Podgląd koszyka",
9. Aplikacja wyświetla bieżący stan koszyka,
10. Użytkownik wybiera opcję 'Optymalizuj koszyk',
11. Aplikacja wyświetla zoptymalizowaną konfigurację produktów z koszyka,
12. Użytkownik zamyka aplikację,

Scenariusz alternatywny:

4a. Użytkownik ponownie przechodzi do Menu,

5a. Użytkownik usuwa produkt z koszyka,

5b. Użytkownik wraca do Menu,

5c. Użytkownik usuwa wszystkie produkty z koszyka,

6a. Użytkownik wraca do koszyka.

Przypadek użycia 2: zarządzaj reklamami

Podstawowe użycia:

- Dodanie reklamy
- Usunięcie reklamy

Podstawowy użytkownik:

- Administrator

Scenariusz główny:

1. Administator edytuje reklamy
   (a) Administrator ukrywa reklamy,
   (b) Administrator dodaje reklamy,
   (c) Administrator zmienia reklamy.

1.3.3 Opis dla dewelopera

Akcja 1

Podstawowy użytkownik:
Użytkownik aplikacji (potencjalny klient restauracji)
Nazwa akcji:
Wyświetlenie MENU
Opis:
Aplikacja powinna przedstawić użytkownikowi pełną ofertę dań dostępnych w danym momencie. Wyświetlane menu powinno uwzględniać zmiany oferty w zależności od pory dnia oraz wprowadzanych okresowo produktów specjalnych.
Odpowiedź systemu:
Zwrócenie pobranego z bazy pełnego MENU restauracji.

Akcja 2

Podstawowy użytkownik:
Użytkownik aplikacji (potencjalny klient restauracji)
Nazwa akcji:
Wyszukanie produktu po nazwie
Opis:
Użytkownik ma możliwość wyszukania produktu za pomocą wpisania jego nazwy w polu tekstowym znajdującym się w interfejsie aplikacji.
Odpowiedź systemu:
Zwrócenie pobranego z bazy wyniku zapytania.

Akcja 3

Podstawowy użytkownik:
Użytkownik aplikacji (potencjalny klient restauracji)
Nazwa akcji:
Wybór kategorii produktów z MENU
Opis:
Aplikacja powinna umożliwić zawężenie wyświetlanego menu do produktów należących do konkretnej kategorii. Podział na kategorie powinien pokrywać się z podziałem, który stosuje restaurator w swojej, ofercie na przykład w kasach samoobsługowych.
Odpowiedź systemu:
Pobranie z bazy niezbędnych danych, wyświetlenie ich w odpowiedniej tabeli.

Akcja 4

Podstawowy użytkownik:
Użytkownik aplikacji (potencjalny klient restauracji)
Nazwa akcji:
Wybór produktu z MENU
Opis:
Użytkownik, po kliknięciu w produkt wyświetlany w udostępnianym przez aplikację menu, powinien mieć następujące możliwości:

- przeczytanie szczegółowych informacji o produkcie (składniki, wartości odżywcze, alergeny),
- dodanie produktu do koszyka, w wybranej przez użytkownika ilości.

Odpowiedź systemu:
Wyświetlenie szczegółowych informacji o produkcie oraz okna umożliwiającego wprowadzenie pożądanej ilości.

Akcja 5

Podstawowy użytkownik:
Użytkownik aplikacji (potencjalny klient restauracji)
Nazwa akcji:
Dodanie produktu do koszyka
Opis:
Pozwala na przechowywanie produktów, których zakup użytkownik będzie chciał zoptymalizować dzięki aplikacji.
Odpowiedź systemu:
Aktualizacja stanu koszyka.

Akcja 6

Podstawowy użytkownik:
Użytkownik aplikacji (potencjalny klient restauracji)
Nazwa akcji:
Wyczyszczenie koszyka
Opis:
Usunięcie wszystkich produktów aktualnie znajdujących się w koszyku.
Odpowiedź systemu:
Aktualizacja stanu koszyka.

Akcja 7

Podstawowy użytkownik:
Użytkownik aplikacji (potencjalny klient restauracji)
Nazwa akcji:
Podgląd koszyka
Opis:
Wyświetlenie informacji o tym, jakie produkty i w jakiej ilości dotychczas dodano do koszyka.
Odpowiedź systemu:
Warstwa graficzna prezentuje zawartość koszyka.

Akcja 8

Podstawowy użytkownik:
Użytkownik aplikacji (potencjalny klient restauracji)
Nazwa akcji:
Usunięcie produktu z koszyka
Opis:
Usunięcie konkretnego produktu spośród tych, które przechowywane są w koszyku.
Odpowiedź systemu:
Usuwa wybrany produkt z koszyka.

Akcja 9

Podstawowy użytkownik:
Użytkownik aplikacji (potencjalny klient restauracji)
Nazwa akcji:
Optymalizacja koszyka
Opis:
Kluczowa funkcjonalność aplikacji to optymalizacja zakupu produktów, które użytkownik zapisał w koszyku. Zoptymalizowanie zamówienia powinno być możliwe, w każdym momencie po kliknięciu dedykowanego przycisku na interfejsie graficznym.
Po kliknięciu tego przycisku, uruchamiana jest warstwa obliczeniowa.
Szczegółowe wymagania:

- optymalizacja ma na celu zaproponowanie najtańszej konfiguracji zakupu produktów, które użytkownik zapisał w koszyku,
- dopuszczalne jest zaproponowanie konfiguracji, która zawiera więcej produktów niż tylko wskazane przez użytkownika, pod warunkiem niższej ceny takiego zamówienia,
- aplikacja powinna uwzględniać obniżki cen dostępne, na przykład w kuponach rabatowyc, zestawach i innych akcjach promocyjnych restauratoram,
- wynik działania algorytmu optymalizującego powinien zostać wyświetlony użytkownikowi w formie listy sugerowanych konfiguracji produktów, z wyróżnieniem grup produktów, które należy zamówić w zestawie lub z kuponem rabatowym.

Odpowiedź systemu:
Warstwa obliczeniowa przelicza zamówienie poszukując optymalnej kombinacji a GUI zwraca konfigurację wynikową.

Akcja 10

Podstawowy użytkownik:
Użytkownik aplikacji (potencjalny klient restauracji)
Nazwa akcji:
Zgłoszenie problemu technicznego
Opis:
Aplikacja powinna umożliwić przesłanie formularza tekstowego z wiadomością do administratora aplikacji o napotkanych przez użytkownika błędach i problemach, które wystąpiły w czasie działania aplikacji.
Odpowiedź systemu:
Wyświetlenie okna pozwalającego użytkownikowi opisanie napotkanego problemu.

Akcja 11

Podstawowy użytkownik:
Użytkownik aplikacji (potencjalny klient restauracji)
Nazwa akcji:
Zamknięcie aplikacji
Opis:
Użytkownik kończy działanie aplikacji.
Odpowiedź systemu:
Zabija wszystkie procesy, czyści bazę danych i zrywa połączenia.

Akcja 12

Podstawowy użytkownik:
Administrator
Nazwa akcji:
Zarządzanie reklamami
Opis:
Administrator może w prosty sposób ustawić, czy w interfejsie graficznym aplikacji będzie wyświetlana reklama oraz załadować nową grafikę reklamy.
Odpowiedź systemu:
Modyfikacja reklamy w warstwie graficznej.

Akcja 13

Podstawowy użytkownik:
Administrator
Nazwa akcji:
Odbiór zgłoszeń użytkowników
Opis:
Administrator ma dostęp do zgłoszeń na temat błędów wysyłanych przez użytkowników.
Odpowiedź systemu:
Wyświetla zgłoszenia użytkowników.

1.4 Wymagania niefunkcjonalne

1. Aplikacja ma umożliwić użytkownikowi optymalne (najtańsze) zamówienie wybranych przez niego produktów z restauracji fast-food. Aplikacja nie zamawia produktów, a jedynie podaje najlepszy możliwy układ zamówienia.

2. Aplikacja powinna być dostosowana do udostępnienia na system Android (Sklep Play). Musi więc spełniać następujące wymagania:

   - Posiadać przygotowaną ikonę,
   - Długi opis działania aplikacji,
   - Przygotowane screeny z aplikacji,
   - Posiadać przygotowaną jedną grafikę reprezentującą funkcje aplikacji.

3. Interfejsem Aplikacji będzie przygotowane GUI.

4. Aplikacja musi być przygotowana do działania pod systemem Android.

5. Aplikacja musi pracować z przygotowanym przez fast-food API do ściągania danych o ich ofercie.

6. Aplikacja wymaga do działania stałego połączenia z internetem.

7. Wymagania wydajnościowe:

   - Czas oczekiwania na optymalizację koszyka nie powinien przekroczyć 10s,
   - Aplikacja nie powinna posiadać zauważalnych opóźnień w czasie dodawania i przeglądania produktów.

8. Wymagania bezpieczeństwa:

   - Aplikacja powinna posiadać dostęp tylko do internetu oraz swoich własnych plików nie powinna mieć dostępu do plików czy też danych użytkownika.

9. Architektura systemu musi być co najmniej trójwarstwowa, przy czym podstawowy podział na warstwy musi uwzględniać: warstwę danych (bazy danych, pliki), warstwę aplikacyjną (logikę aplikacji), warstwę prezentacji (GUI przedstawiające dane użytkownikowi).

10. Wymagania pamięciowe:

    - Aplikacja nie powinna zajmować więcej niż 40MB pamięci telefonu na pliki własne,
    - Dane pobrane z bazy powinny być usuwane niezwłocznie po wyłączeniu aplikacji lub wyczyszczeniu koszyka.

11. Skalowalność:

    - Aplikacja powinna być bardzo łatwo skalowalna do dużej liczby użytkowników, jakość jej działania nie powinna się zmienić w zależności od ilości użytkowników.
    - Aplikacja powinna być zbudowana tak, aby mogła łatwo zostać poszerzona o kolejne restauracje.

12. Aplikacja powinna być co najmniej 2-giej klasy dostępności czyli, powinna działać bez zarzutu przez 99% czasu. Błędy i uaktualnienia muszą mieścić się w jednym pozostałym procencie.

1.5 Stany specjalne/wyjątkowe

1. **Inicjalizacja**
   Pierwszym elementem, który włącza się w naszej aplikacji jest warstwa graficzna. W przypadku niepowodzenia, wyświetlany jest błąd i aplikacja jest zamykana.

2. **Błędy warstwy obliczeniowej/graficznej**
   Jeśli w którymkolwiek momencie wystąpi błąd jednej z wyżej wymienionych warstw:

   - warstwa ta zostanie zamknięta,
   - połączenie warstwa obliczeniowa - warstwa graficzna zostanie przerwane.

   Jest to uznawane za błąd krytyczny, powoduje zakończenie trwającej sesji użytkownika i zamknięcie programu.

3. **Błąd bazy danych**
   Jeśli użytkownik nie posiada połączenia z internetem, bądź zostanie ono zerwane, to wówczas aplikacja nie będzie w stanie nawiązać połączenia z API restauracji i aplikacja zwróci błąd.

4. **Zmiana reklam**
   Administrator zmienia reklamy i kontroluje ich obecność w oknie aplikacji.

5. **Zakończenie aplikacji**
   Informację o zakończeniu aplikacji przesyła warstwa graficzna.
   W wyniku zakończenia, następuje rozłączenie każdego połączenia i zamknięcie każdego z programów.
   Następuje to w następującej kolejności:

   5.1 baza danych zamyka połączenie z API restauracji,

   5.2 warstwa obliczeniowa zamyka się,

   5.3 GUI warstwy graficznej zamyka się.

2 Architektura rozwiązania

2.1 Opis systemu z rozbiciem na komponenty

Aplikacja składa się z trzech podstawowych komponentów:

1. **Warstwy graficznej**
   Oczekuje ona na polecenia użytkownika, pośredniczy w komunikacji między użytkownikiem a warstwą obliczeniową.
   Ma postać GUI wyświetlanego na urządzeniu mobilnym.

2. **Warstwa Obliczeniowa**
   Ma postać programu, którego zadaniem jest optymalizowanie zamówiennia użytkownika. Służy jako przekaźnik pomiędzy interfejsem graficznym a bazą danych.

3. **Baza Danych**
   Ma postać bufora, w którym przechowywane są niezbędne dane, pobrane przy użyciu API wybranej sieci restauracji.

Główne założenia komunikacji między komponentami:

- Warstwa graficzna komunikuje się z warstwą obliczeniową za pomocą zmiennych oraz poleceń.
- Warstwa graficzna nie komunikuje się z bazą danych.
- Warstwa obliczeniowa komunikuje się z bazą danych przy pomocy udostępnionego przez restaurację API.

2.2 Przepływ danych

2.2.1 Model danych

Dane wykorzystywane w aplikacji można podzielić na dwa rodzaje:

1. **Dane zewnętrzne**:
   Informacje pobierane ze strony danej restauracji. Zawierają one listę wszystkich produktów wraz z ich cenami i opisami.
   Pobierane wraz z każdym uruchomieniem aplikacji.

2. **Dane tymczasowe aplikacji**:
   Ze względu na częste zmiany w ofertach restauracji, informacje o ich menu istnieją tylko w czasie działania aplikacji i nie są zapisywane w pamięci wewnętrznej. To samo dotyczy danych "koszyka", który jest tworzony od nowa przy każdym uruchomieniu, zgodnie z potrzebami użytkownika.

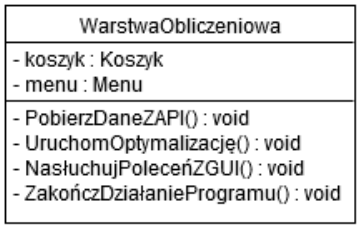
2.2.2 Identyfikacja danych

1. **Dane zewnętrzne**:
   Udostępniane przez API sieci fast-food są typu real-time, gdyż oferta restauracji zmienia się w sposób ciągły w zależności: od pory dnia, dostępnych promocji, produktów specjalnych itp. Wymusza to pobieranie przez aplikację aktualnej oferty przy jej każdym uruchomieniu.

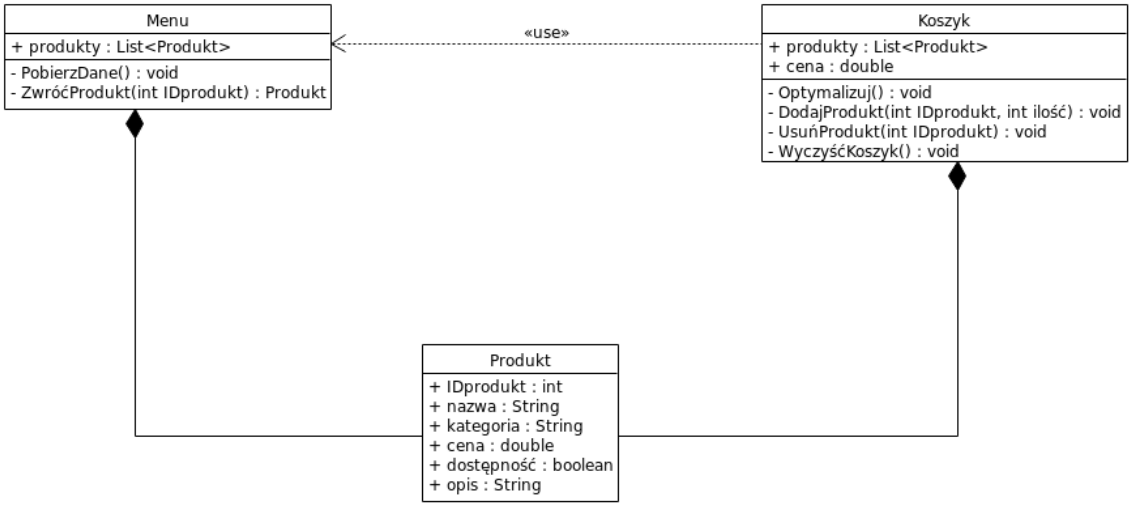
2. **Dane wewnętrzne**:
   Po uruchomieniu aplikacja zapisuje tymczasowo menu restauracji, dane te są przechowywane do zamknięcia aplikacji. Ze względu na ich zmienność oraz użycie do wyliczeń i analizy przez algorytm optymalizujący, kwalifikujemy je jako dane dynamiczne.

2.3 Model klas

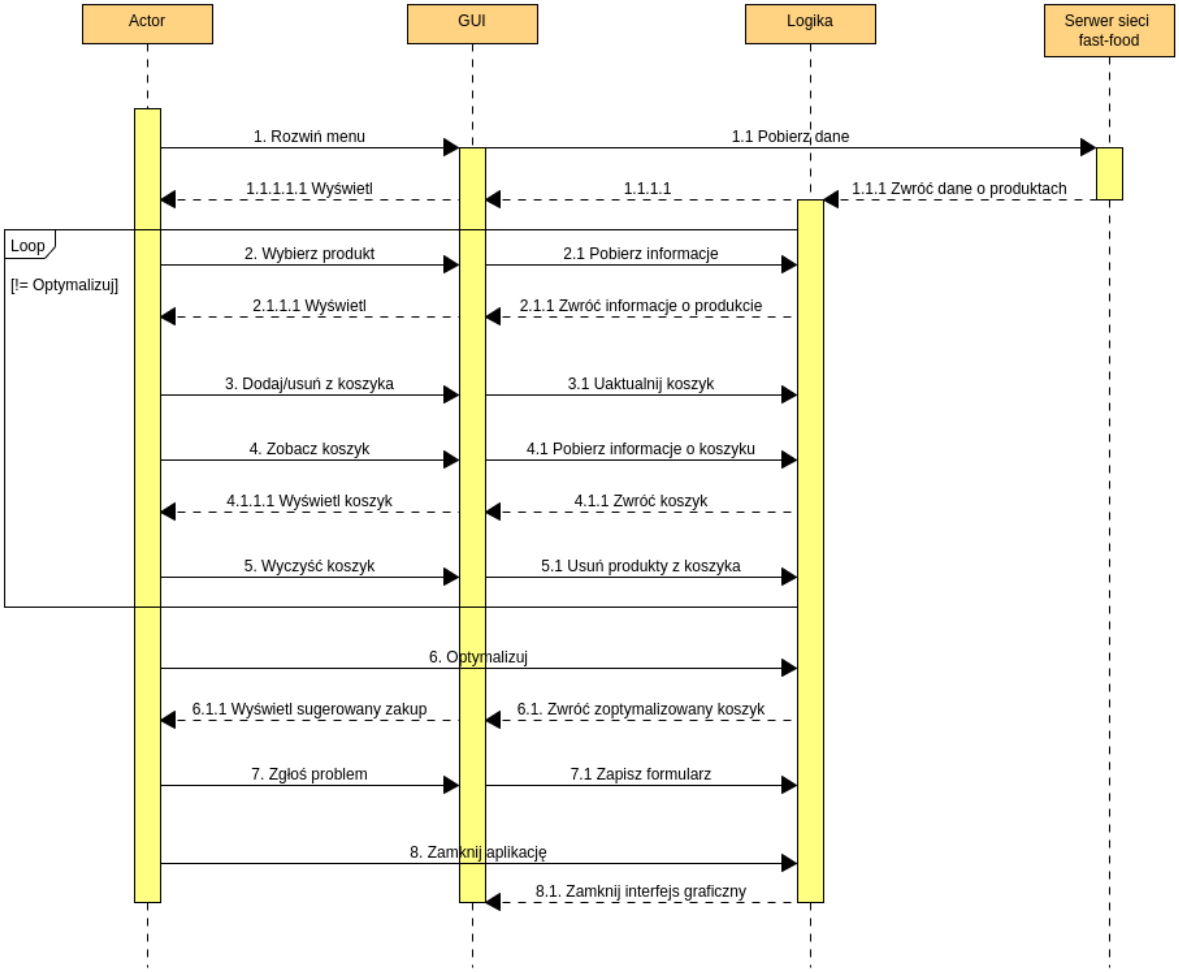
2.3.1 Warstwa obliczeniowa



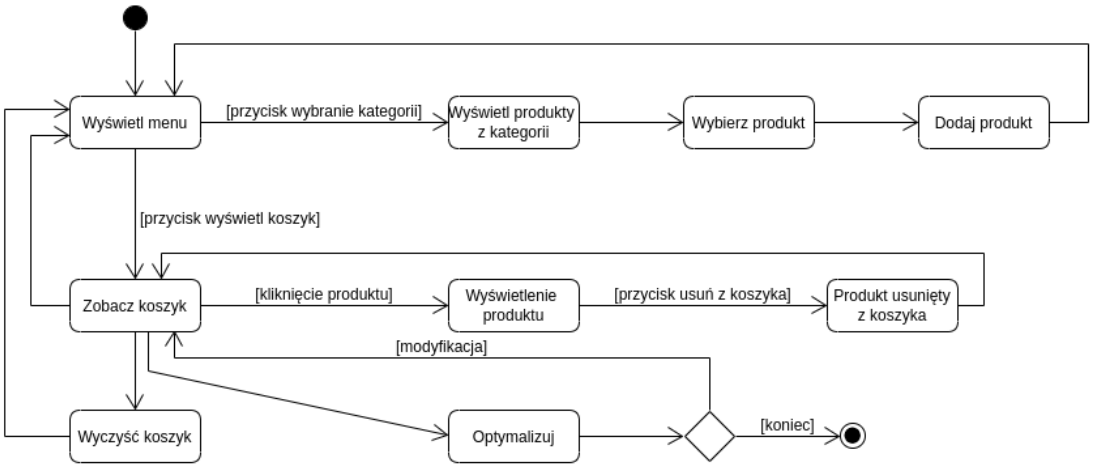
2.3.2 Warstwa danych



2.4 Przepływ zdarzeń (diagram sekwencji)

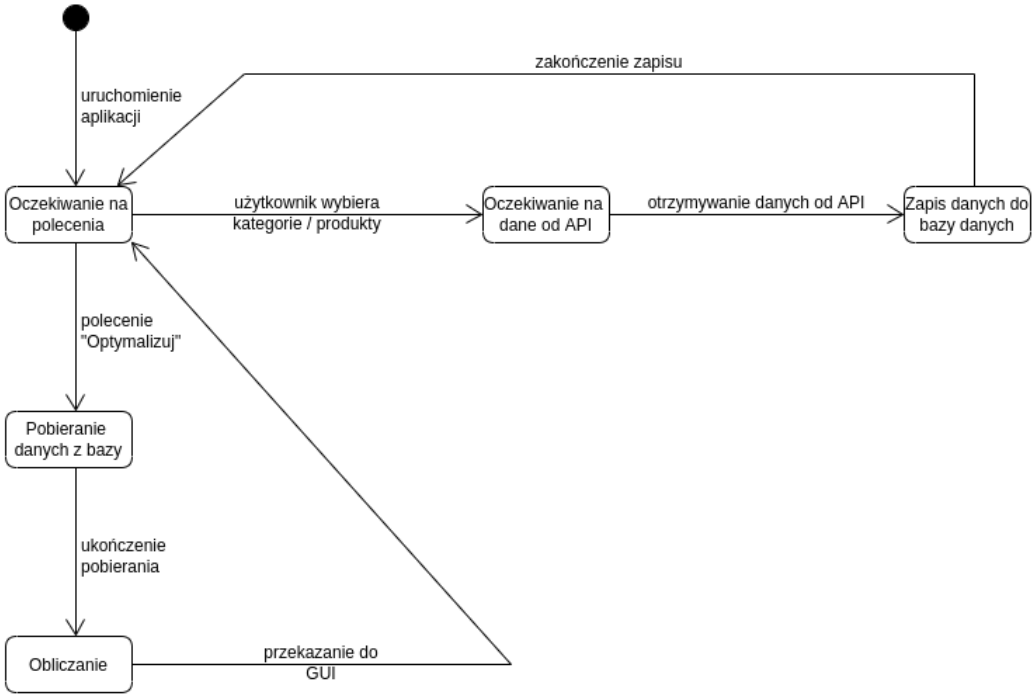


2.5 Zachowanie systemu (diagram aktywności)



2.6 Diagramy stanów dla kluczowych komponentów

2.6.1 Diagram stanów dla warstwy obliczeniowej



2.6.2 Diagram stanów dla warstwy danych