**JurPizza™**

**Opis projektu**

* **Temat**

Projektem, którego tworzenia chcemy się podjąć, jest aplikacja służąca do zamawiania pizzy. Jest to temat nam bliski, gdyż sami wielokrotnie składaliśmy takie zamówienia. Rozmowa telefoniczna nie jest zwykle dobrym rozwiązaniem, istnieje duże prawdopodobieństwo pomyłki w zamówieniu, co każdemu z nas się zdarzyło.

Chcielibyśmy dzięki tej aplikacji zminimalizować możliwość pomyłki w zamówieniu z powodu nieporozumienia. Również pozwalałaby ona śledzić zamówienie, stan i miejsce w jakim się znajduje, orientacyjny czas realizacji.

W planie mamy stworzenie aplikacji klienckiej oraz aplikację dla pizzerii. Jeśli tworzenie oprogramowania okaże się bardziej czaso- i pracochłonne niż zakładamy, to ograniczymy się do samej aplikacji klienckiej, pozostawiając możliwość rozbudowy.

* **Opis logiki aplikacji**

Każdy użytkownik aplikacji będzie logować się do swojego konta. Z tym kontem będzie można na stałe powiązać swoje dane i adresy dostawy.

Złożenie zamówienia będzie polegać na wyborze posiłku, a także szczegółów dotyczących m.in. dodatków, rozmiarów pizzy. Będzie istniała również możliwość kompozycji własnej pizzy. Następnie zamówienie zostanie podsumowane, z możliwością edycji, także z informacją o orientacyjnym czasie realizacji. Wówczas będzie możliwość podania lub ewentualnej zmiany zapisanego już adresu dostawy. Wtedy też należy podać informację o rodzaju płatności. Jeśli płatność będzie miała być wykonana online, aplikacja będzie automatycznie przekierowywać na stronę banku. Po przyjęciu zamówienia, będzie można śledzić stan, w którym się znajduje, a także na bieżąco aktualizowany czas realizacji zamówienia.

Poza tym aplikacja będzie informować o aktualnych promocjach. Także przy składaniu zamówienia będzie proponować modyfikację, uwzględniając rabaty, tak, aby zamówienie było jak najbardziej opłacalne.

* **Plan pracy**

Pracę chcemy podzielić na kilka etapów:

1. Aplikacja mobilna
   1. Backend (C++):
      1. Wstępne prace nad strukturami danych i wewnętrznym działaniu backendu
      2. Przygotowanie środowiska
      3. Implementacja
      4. Integracja z frontendem
      5. (Ewentualny) Refactor
   2. Frontend (Java/Swift):
      1. Design szaty graficznej, konsultacja z artystami
      2. Przygotowanie środowiska
      3. Implementacja
      4. Integracja z backendem
      5. (Ewentualna) Reimplementacja na inne platformy
2. (Jeśli czas pozwoli) Aplikacja serwerowa
   1. Backend (Java):
      1. Przygotowanie rusztowania; testy serwera
      2. Implementacja funkcjonalności definiowanej de facto przez możliwości aplikacji mobilnej
      3. Jednoczesna implementacja funkcjonalności w aplikacji mobilnej
      4. Testy polowe
   2. Frontend (Java lub HTML/JS):
      1. Stworzenie podstawowych widoków
      2. Implementacja kontrolerów i wymiany danych z backendem.

* **Krótko o celach**

W pierwszej kolejności, zdecydowaliśmy się stworzyć aplikację mobilną, celując głównie w system Android (i jak czas pozwoli, iOS).

Podjeliśmy decyzję podzielić aplikację na dwa główne moduły - moduł logiczny (Backend) i moduł graficzny (Frontend). Backend będzie napisany w języku C++ - będzie obsługiwał wszystkie operacje na danych, przetwarzanie informacji, obsługę systemu zamówień o strony klienta. Jeśli zdecydujemy się napisać wersję na iOS, ten kod będzie współdzielony. Musi być zatem napisany w jak najbardziej neutralny sposób względem platform, przez co będzie charakteryzował się dużym poziomem abstrakcji.

Moduł graficzny będzie napisany w języku odpowiednim dla platformy – w przypadku Androida będzie to Java (z interfejsowaniem przez JNI), a dla ewentualnegoo iOS’a będzie to Swift (z interfejsowaniem przez Objective-C++ i Objective-C. Wewnętrznie, moduły graficzny będą wykorzystywały separację MVC (Model View Controller), która jest naturalnym rozwiązaniem przy wykorzystaniu natywnych bibliotek interfejsu użytkownika.

Wspominamy o podziale na moduły, ze względu na jego relatywną ścisłość - komunikacja między modułami będzie odbywać się wyłącznie na zasadzie wymiany danych, przez system „eventów” - wydarzeń, bez wprowadzania ścisłych zależności między poszczególnymi częściami projektu. Idea jest taka, że będzie istniała możliwość przekazania dowolnego obiektu do drugiego modułu w celu poinformowaniu go o zmianach. Obiekty te będą zawierały opisy wysokiego poziomu - przykładowo, wystarczy będzie wysłać pojedynczy obiekt, aby wywołać zmianę widoku w aplikacji, generację rozwijanej listy składników pizzy itd. Obydwa moduły będą posiadały mechanizm służący do interpretacji odpowiednich obiektów i ewentualnego tłumaczenia na format zrozumiały dla danego modułu (jeśli bezpośrednie wykorzystanie nie będzie możliwe/utrudnione).

Zastrzegamy sobie możliwość zastosowania frameworku wieloplatformowego oraz bibliotek osób trzecich do zastosowania w którymkolwiek module. Między innymi: ujednolicona obsługa interfejsu użytkownika, biblioteki dostępu do danych (pozycje GPS, mapy, aktualne rynkowe ceny składników) itp.

Jeżeli czas pozwoli, chcielibyśmy też napisać aplikację dla pizzeri, służącą do odbierania zamówień. Aplikacja ta najprawdopodobniej zostanie napisana przy użyciu języka Java oraz pewnego frameworka typowego do zastosowań serwerowych (surowe serverlety, Spring, itp.). Będzie występował podobny podział na frontend i backend.

Backend będzie odpowiedzialny za pobieranie i przesyłanie danych do aplikacji (dostępne pizze, składniki, promocje), uwierzytelnienie użytkowników, przekazywanie identyfikatora śledzenia kierowcy; przekazywanie zamówień pizzeri wraz z prostym systemem przypisywania zamówienia do kucharza.  
  
 Frontend będzie prostym interfejsem, napisanym albo w Javie przy użyciu Swing/JavaFX, albo jako prosta aplikacja przeglądarkowa. Będzie umożliwiał edycję listy składników, rodzajów pizzy, dodawanie promocji, sprawdzanie statusu aktualnie obsługiwanych zamówień.

**Lista autorów**

1. Martin Jurczyk
2. Grzegorz Łazarski
3. Gabriela Rogowska

**Scenariusze przypadków użycia**

|  |  |
| --- | --- |
| 1. Składanie zamówienia | |
| Aktorzy | Użytkownik, system |
| Warunki  początkowe | Użytkownik chce zamówić pizzę |
| Zdarzenie inicjujące | Wybranie opcji składania zamówienia |
| Scenariusz  główny | 1. System pozwala użytkownikowi na przegląd dostępnych rodzajów pizzy. 2. Użytkownik wybiera pizzę lub napój, które najbardziej mu odpowiadają. 3. W celu dodania pizzy do koszyka system wymaga wyboru przez użytkownika wielkości pizzy, ewentualnie sosów. 4. W celu dodania go do koszyka użytkownik musi wybrać wielkość, rodzaj i ilość danego typu napoju. 5. Po zakończeniu wyboru kolejnych pozycji do zamówienia, system żąda wprowadzenia danych adresowych. - albo z listy albo wprowadzanie adresu. 6. Użytkownik podaje swoje dane lub potwierdza dane zapamiętane przez aplikację (patrz 2. Dane adresowe) |
| Scenariusze poboczne | * W razie rozmyślenia się użytkownika, bądź zmiany decyzji, z każdego z wymienionych punktów użytkownik może wykonać krok w tył i pozmieniać swoje wybory, czy wprowadzone dane. * Istnieje w każdym momencie możliwość podglądu koszyka przez użytkownika. |
| Warunki końcowe | * Aplikacja zapamiętuje stan zamówienia. * Kolejnym krokiem realizacji jest płatność. |

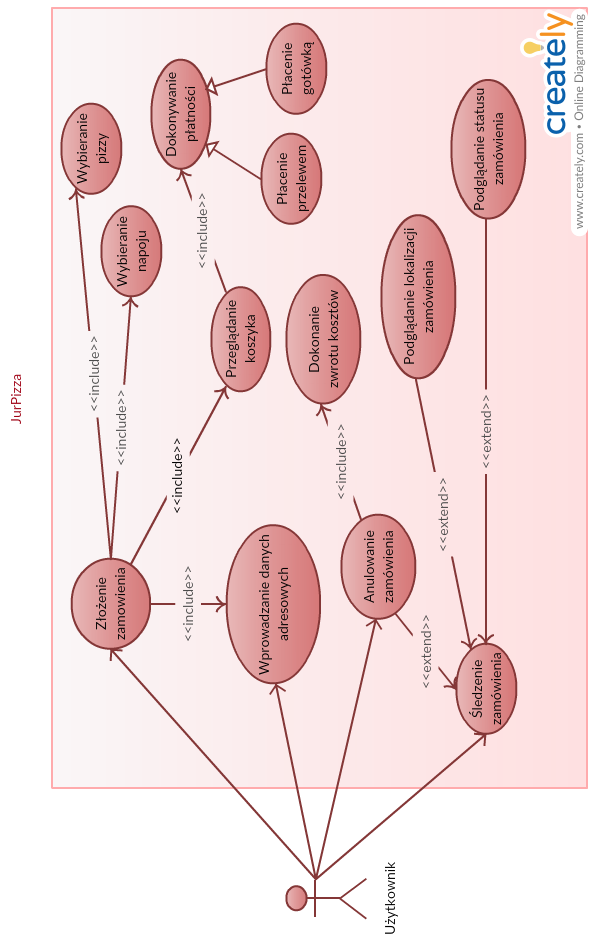
|  |  |
| --- | --- |
| 2. Dane adresowe | |
| Aktorzy | Użytkownik, system |
| Warunki  początkowe | Użytkownik nie chce za każdym razem wpisywać swoich danych |
| Zdarzenie inicjujące | Wybranie przez użytkownika opcji „Wprowadź adres” |
| Scenariusz  główny | 1. Użytkownik uzupełnia pola z danymi adresowymi: adres, miasto, kod pocztowy, numer telefonu. 2. Użytkownik zatwierdza dane 3. System zapamiętuje podane informacje |
| Scenariusz poboczny | 1a. W razie błędnego wprowadzenia kodu pocztowego – użytkownik jest informowany przez aplikację. |
| Warunki końcowe | System zapamiętuje podane dane, aby później w trakcie zamawiania nie trzeba było danych ponownie wprowadzać, ale tylko potwierdzić. |

|  |  |
| --- | --- |
| 3. Płatność | |
| Aktorzy | Użytkownik, system |
| Warunki  początkowe | Użytkownik wybrał pizzę i napoje, wpisał swoje dane adresowe. |
| Zdarzenie inicjujące | Przejście użytkownika do kroku zamówienia „płatność” |
| Scenariusz  główny | 1. Użytkownik ma do wyboru 2 opcje płatności. 2. a) użytkownik chce zapłacić gotówką – wówczas płatność nastąpi dopiero przy dostarczeniu pizzy.   b) użytkownik chce zapłacić przelewem – wybiera tę opcję   1. b) aplikacja przekierowuje użytkownika do odpowiedniej strony w celu dokonania płatności internetowej. 2. b) użytkownik czeka na potwierdzenie płatności. 3. Użytkownik jest przekierowany do ekranu dziękującego za zamówienie, skąd ma możliwość śledzić zamówienie albo wrócić do ekranu początkowego. |
| Scenariusze  poboczne | 3b)\* W razie problemów z danym serwisem, użytkownik jest o tym informowany i proponowana jest płatność później lub gotówką przy odbiorze. |
| Warunki końcowe | Pizzeria przyjmuje zamówienie do realizacji.  Zamówienie pojawia się w sekcji dotyczącej śledzenia zamówienia. |

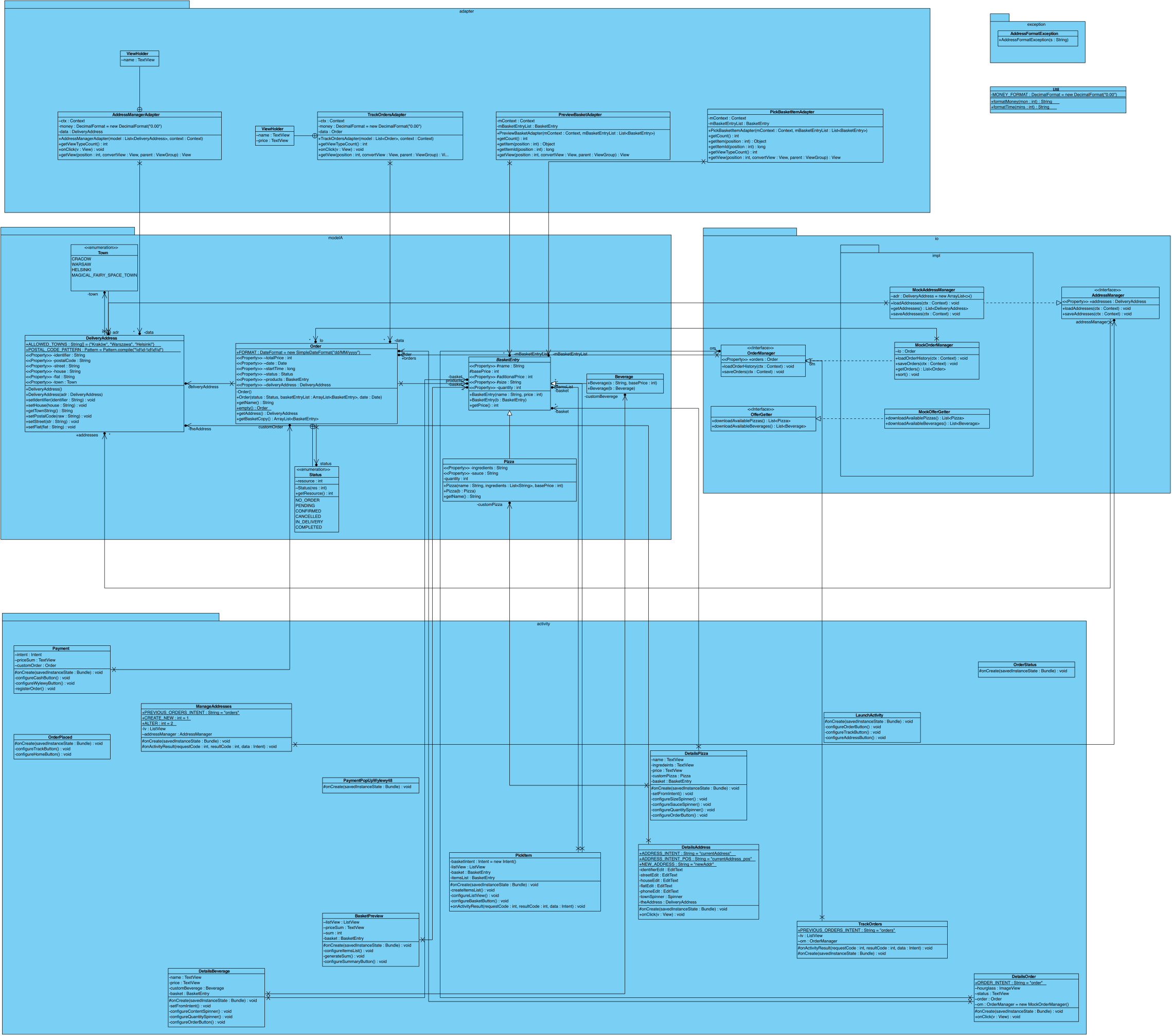
|  |  |
| --- | --- |
| 4. Śledzenie zamówienia | |
| Aktorzy | Użytkownik, system |
| Warunki  początkowe | Użytkownik zamówił i wybrał metodę płatności i ewentualnie zapłacił za zamówienie. |
| Zdarzenie inicjujące | Wybranie przez użytkownika opcji: „Śledź swoje zamówienie” |
| Scenariusz  główny | 1. Użytkownik przechodzi z ekranu głównego do ekranu zawierającego dane nt zamówienia takie jak:  * status zamówienia (m. in. w realizacji, dowożone, potwierdzone, anulowane, oczekujące na realizację) * dane szczegółowe dot. zamówienia  1. Użytkownik może śledzić lokalizację swojego zamówienia na mapce. |
| Warunki końcowe | Użytkownik posiada informacje o statusie i lokalizacji swojego zamówienia. |

|  |  |
| --- | --- |
| 5. Anulowanie zamówienia | |
| Aktorzy | Użytkownik, system |
| Warunki  początkowe | Użytkownik dokonał zamówienia |
| Zdarzenie inicjujące | Użytkownik chce zrezygnować z dokonanego zamówienia. |
| Scenariusz  główny | 1. Użytkownik wybiera opcję „śledź swoje zamówienie”. 2. Przy zamówieniu, które chce anulować wybiera opcję „anuluj zamówienie”. 3. System pyta o potwierdzenie decyzji użytkownika. 4. Jeśli nastąpiła płatność przelewem – system dokonuje zwrotu kosztów. 5. System potwierdza użytkownikowi dokonanie zwrotu pieniędzy. |
| Scenariusz  poboczny | 3a. W przypadku, gdy dane zamówienie jest już wykonywane (w realizacji) lub dowożone, system informuje użytkownika o braku możliwości dokonania anulowania.  4. W razie problemów z dokonaniem „przelewu zwrotnego” system informuje użytkownika, że spróbuje później tego dokonać. Jeśli nadal wystąpi problem – skontaktuje się z nim pracownik pizzerii. |
| Warunki końcowe | W systemie nie figuruje już dane zamówienie – zostaje anulowane.  Ewentualnie koszta zostają zwrócone zamawiającemu. |

**Diagram przypadków użycia**

****

**Diagramy klas**

****