Wesołe Miasteczko

1. ## Dziedzina problemowa:
   System mógłby znaleźć zastosowanie w dziedzinie zajmującej się rozrywką, zwłaszcza w sektorze skierowanym dla dzieci.

2. ## Cel:
   System zbudowany w celu ułatwieina właścicielowi zarządzaniem wesołym miasteczkiem.

3. ## Zakres odpowiedzialności systemu:
   System ma pomóc organizować pracowników, zapisywać sponsorów wraz z fundowanymi przez nich imprezami, zapamiętywać atrakcje wraz z wyliczaniem ich opłacalności (zarobek miesięczny z atrakcji), a także wspomagać organizację wszelkiego rodzaju wydarzeń.

4. ## Użytkownicy systemu:
   Potencjalnymi użytkownikami systemu będą: właściciel, pracownicy placówki, a także klienci wesołego miasteczka.

5. ## Wymagania użytkownika:
   Wymagania wstępne:

   1) W systemie należy przechowywać informacje o osobach związanych z wesołym miasteczkiem. Osoby te zostały podzielone ze względu na odgrywanie określonej roli w funkcjonowaniu systemów wesołego miasteczka – pracowników: m.in. kasjerów, obsługę atrakcji, oraz obsługę sprzątającą, a także osoby spoza placówki – klientów internetowych i sponsorów.
   2) Dla każdej osoby należy przechowywać imię, nazwisko, datę urodzenia, adres, adres pocztowy oraz listę numerów telefonu (co najmniej jeden).
   3) Każdy klient powinien posiadać w systemie swój numer PESEL oraz adres e-mail (opcjonalnie).
   4) Każdy pracownik wesołego miasteczka powinien być opisany przez datę zatrudnienia, oraz listę poprzednich miejsc pracy (opcjonalnie).
   5) Dla każdego sponsora należy pamiętać nazwę firmy, którą reprezentuje, a także jej dane teleadresowe, oraz stopień wiarygodności (od 1 do 5). Dodatkowo należy zapamiętać listę sponsorowanych wcześniej wydarzeń.
   6) System powinien umożliwić przechowywanie informacji na temat atrakcji znajdujących się na terenie parku. Każda atrakcja ma swoją nazwę, nagrody do wygrania (opcjonalnie), a także personel odpowiedzialny za obsługę atrakcji (minimum 1 osoba, przydzielony pracownik może obsługiwać jedną atrakcję). Określone są również minimalne wymagania wiekowe od których dopuszcza się uczestnictwo w atrakcji i maksymalną ilość osób, które mogą na raz korzystać z danej atrakcji. Należy również pamiętać ostatnią datę końserwacji urządzeń w parku. Sama konserwacja trwa trzy dni i przez ten okres korzystanie z atrakcji jest niedozwolone.
   7) Atrakcje w wesołym miasteczku można podzielić na rodzaje, m.in.: na kolejki, atrakcje zręcznościowe, diabelskie młyny; sezonowość, m.in.: całoroczne, oraz letnie, a także strefy, które dzielą się na strefę dla dzieci i strefę dla młodzieży. Strefa dla dzieci powinna przechowywać informacje o tym, czy obecność opiekuna jest wymagana, natomiast strefa dla młodzieży – minimalny wzrost wymagany do wejścia na

atrakcję.
8) Dla pracowników należy pamiętać dodatkowo wiek, stawkę godzinową wspólną dla wszystkich zatrudnionych wynoszącą 25 zł, mogącą później ulec zmianie oraz premię za oceny klientów, podzieloną na stopnie. Uzyskanie średniej oceny powyżej 3.8 daje premię 3%, powyżej 4.3 – premię 5%, powyżej 4.8 – premię 8%. Dla kasjera należy zapamiętać jaką zmianę mu przydzielono (zmiana może być poranna, lub popołudniowa), dla obsługi sprzątającej – ostatnie czyszczenie, natomiast dla obsługi atrakcji – posiadane uprawnienia do obsługi atrakcji.
9) System powinien umożliwić zapamietanie ocen klientów dotyczących zarówno atrakcji jak i pracowników je obsługujących. Ocena powinna być w skali 1-5, powinna być także możliwość dodania opinii (opcjonalna, maksymalnie 500 słów).
10) Klient ma prawo zarezerwować wizytę w wesołym miasteczku przez internet. Należy pamiętać datę rezerwacji, jej unikatowy numer, cenę, a także ilość osób zarezerwowanych w danym terminie. Powinien również być przechowywany status rezerwacji („złożona", „zapłacona", „zrealizowana", „anulowana").
11) Sponsor może zaproponować wydarzenie – należy pamiętać jego status („oczekuje na zatwierdzenie", „w trakcie realizacji", „ukończone", „niezatwierdzone", „anulowane") oraz kwotę wpłacaną przez sponsora. System powinien zapamiętywać listę wszystkich sponsorów, którzy dołożyli się do organizacji wydarzenia.

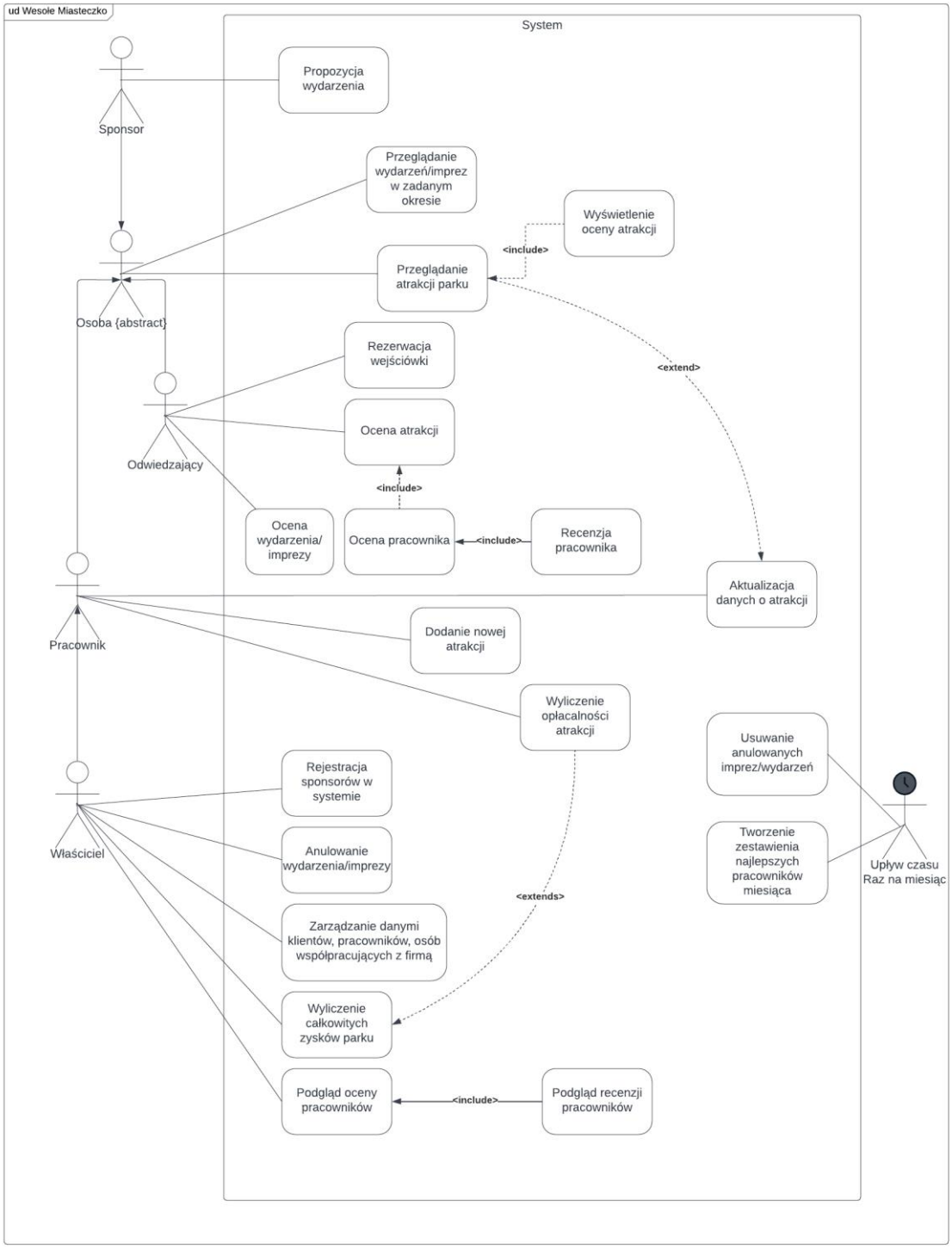
System ma wspierać potencjalnych użytkowników m.in. w realizowaniu zadań, których listę przedstawiono poniżej:

-przeglądanie atrakcji parku (wszyscy użytkownicy)
-wyświetlenei ocen atrakcji parku podczas ich przeglądania (wszyscy użytkownicy)
-przeglądanie wydarzeń w zadanym okresie (wszyscy użytkownicy)
-propozycja nowego wydażenia (sponsor)
-rezerwacja wejściówki (klient)
-ocena wydarzenia, pracownika, atrakcji (klient)
-wystawienie opinii podczas oceniania wydarzeń, oraz atrakcji
-dodanie nowej atrakcji (pracownik)
-wyliczenie opłacalności atrakcji (pracownik)
-aktualizacja danych o atrakcji podczas przeglądania (pracownik)
-wyliczenie całkowitych zysków parku (właściciel)
-rejestracja sponsorów w systemie (właściciel)
-anulowanie wydarzenia/imprezy (właściciel)
-zarządzanie danymi klientów, pracowników i osób związanych z firmą (właściciel)
-wyliczenie całkowitych zysków parku (właściciel)
-podgląd oceny pracowników i ich recenzji (właściciel)
-raz na miesiąc usuwanie anulowanych imprez/wydarzeń/rezerwacji (system)
-raz na miesiąc tworzenie zestawienia najlepszych pracowników miesiąca (system)

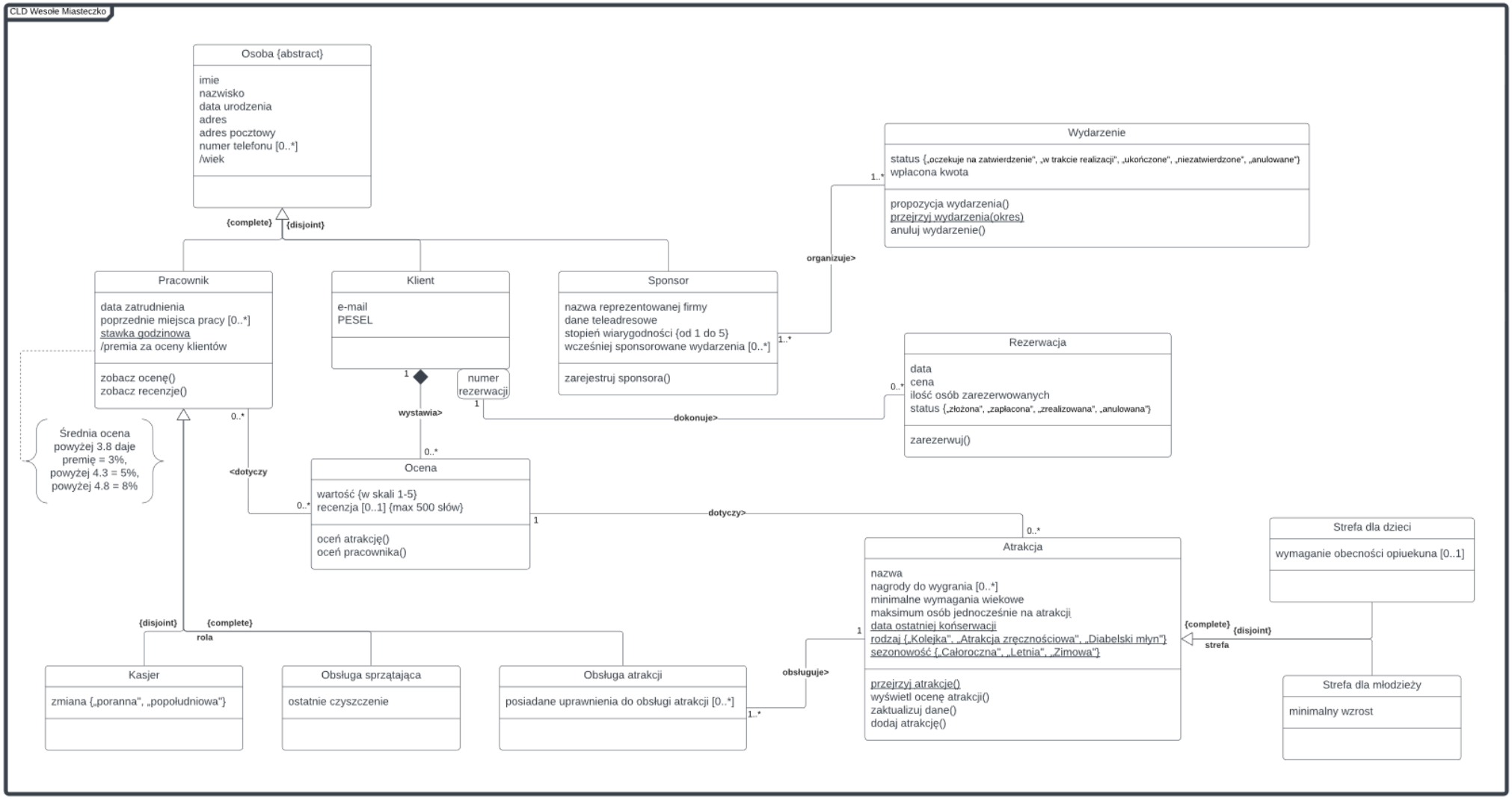
Lista wymagań niefunkcjonalnych, które można wziąć pod uwagę, rozważając ograniczenia, przy których system wesołego miasteczka powinien pracować:

-dostępność
-zapewnienie bezpieczeństwa danych
-szybka reakcja systemu na akcje użytkownika
-możliwość równoczesnej pracy wielu użytkowników
-łatwe serwisowanie systemu w przyszłości

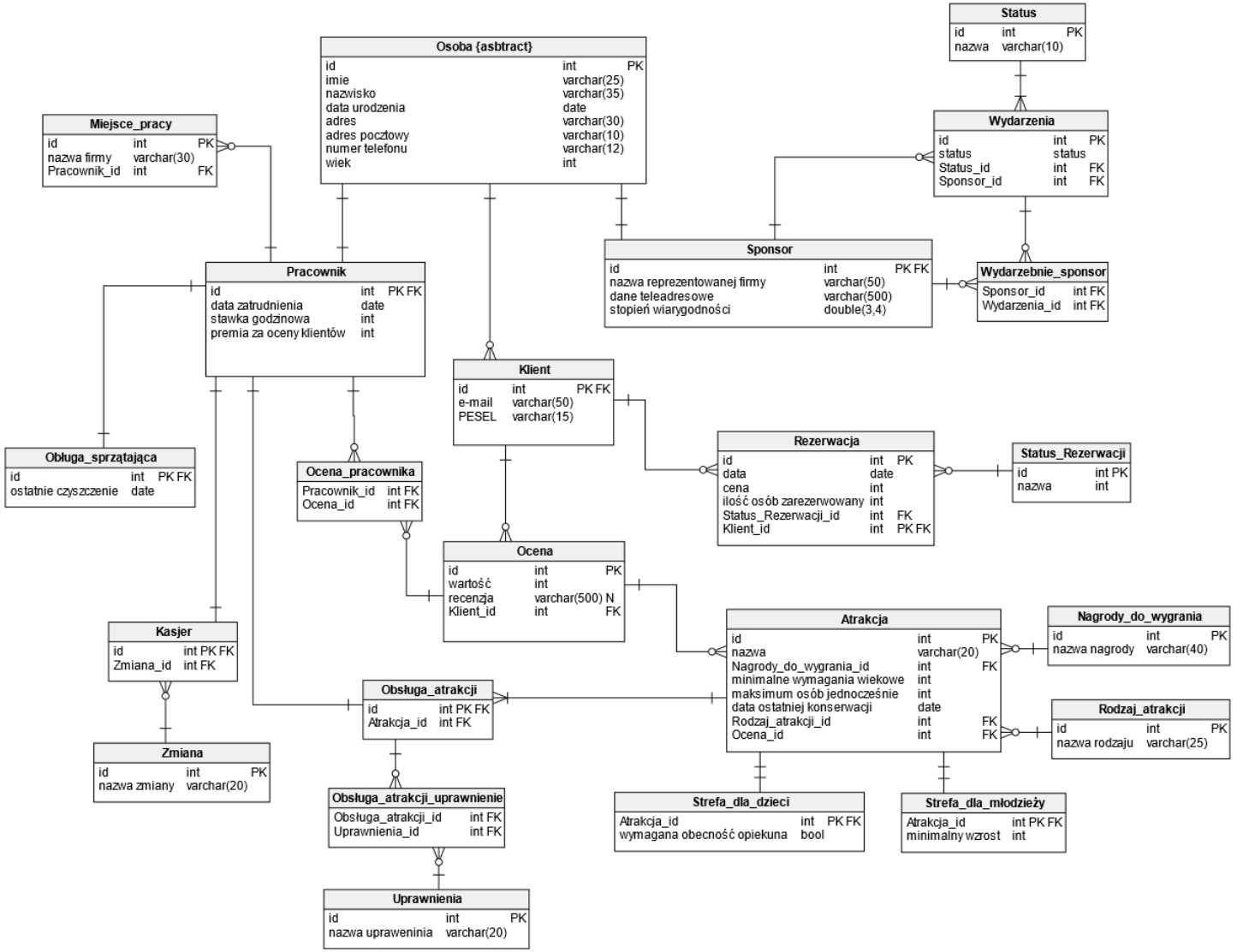
6. Wymagania funkcjonalne:



7. Opis struktury systemu (diagram klas - analityczny):



8. Diagram klas – projektowy



9. **Scenariusz przypadku użycia**

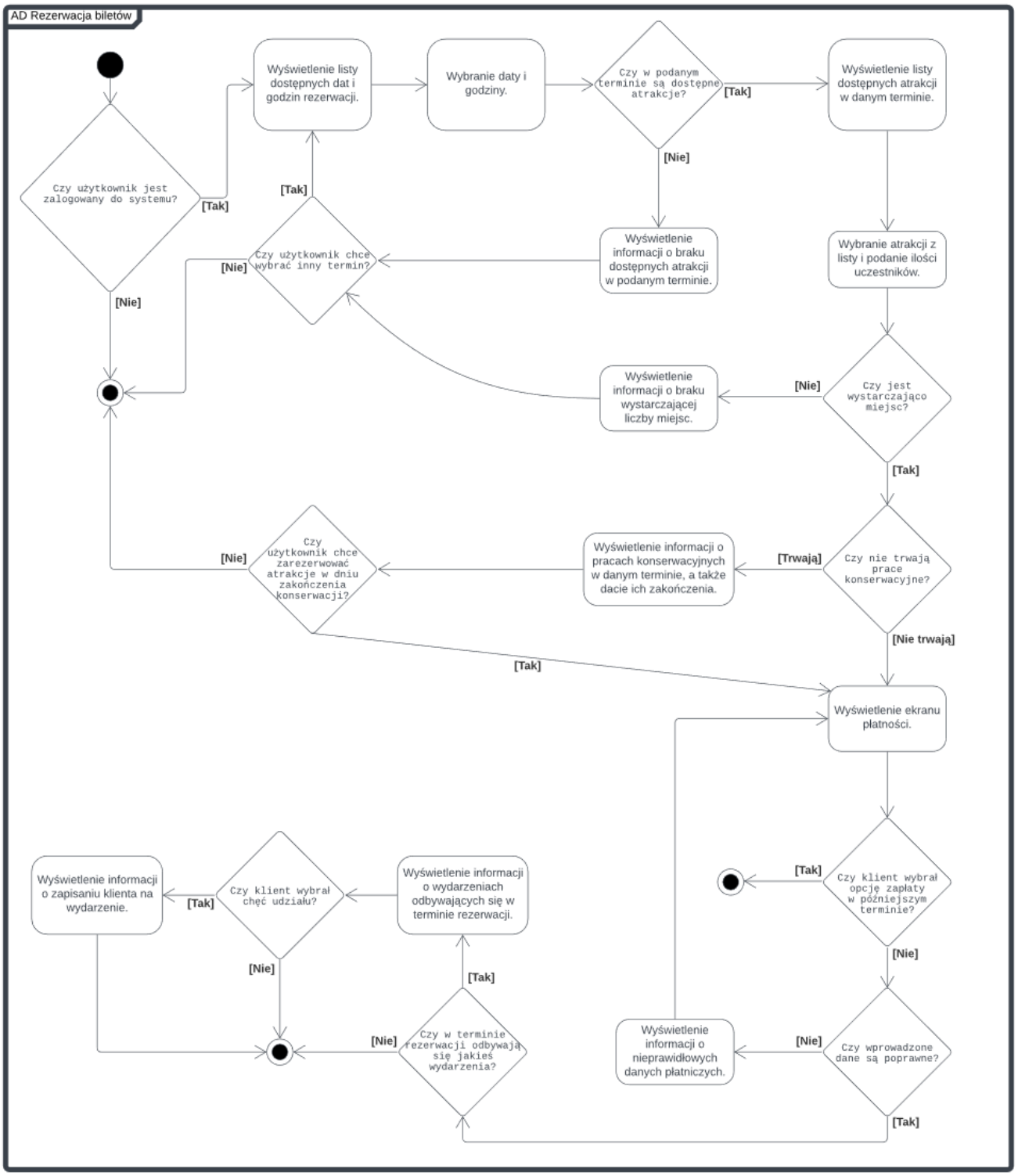
Przebieg główny:

1) Zalogowany klient otwiera stronę internetową wesołego miasteczka i wybiera opcję "Rezerwacja biletów".
2) System wyświetla dostępne daty i godziny.
3) Klient wybiera preferowaną datę i godzinę.
4) System wyświetla dostępne atrakcje w wybranym terminie.
5) Klient wybiera pożądane atrakcje, które chce odwiedzić, i podaje liczbę osób.
6) System sprawdza dostępność i stan atrakcji (czy nie trwają prace konserwacyjne).
7) Jeżeli atrakcje są dostępne, system wyświetla ekran płatności.
8) Klient wprowadza swoje dane płatnicze.
9) System wyświetla rezultat transakcji.

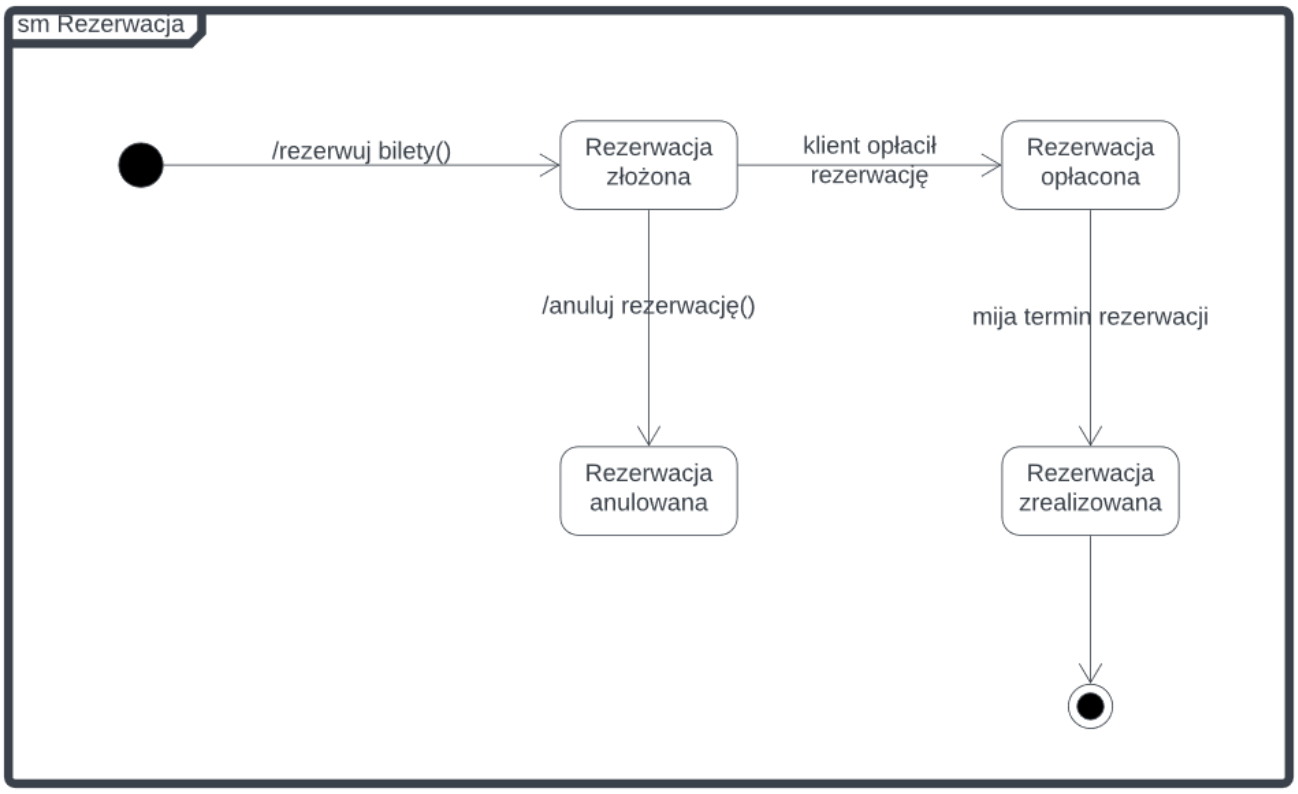
Przebiegi alternatywne:

4a) System informuje o braku atrakcji w podanym terminie i pyta, czy użytkownik chce wybrać inny termin.
6a) System informuje o braku miejsc i pyta, czy użytkownik chce wybrać inny termin.
6b) System informuje, że obecnie trwają prace konserwacyjne i informuje klienta o dacie zakończenia i wyświetla zapytanie, czy klient chce zarezerwować w dniu zakończenia konserwacji.
8a) Klient wybiera opcje zapłacenia w późniejszym terminie.
9a) System informuje, że wprowadzone dane są niepoprawne i prosi o ponowne wpisanie.
9b) System potwierdza wprowadzone dane i wyświetla użytkownikowi dodatkowe wydarzenia odbywające się w terminie rezerwacji i pyta, czy klient chciałby wziąć w nich udział.
9ba) Klient zapisuje się na wydarzenie.
9baa) System potwierdza udział klienta w wydarzeniu.
9bb) Klient odrzuca zaproszenie.

10. **Diagram aktywności dla przypadku użycia:**



11. **Diagram stanu dla klasy Rezerwacja:**



12. Wymagania niefunkcjonalne:

Ograniczenia, przy których system wesołego miasteczka powinien pracować:

Dostępność:
System nie ma ograniczeń czasowych, będzie dostępny dla użytkownika 24h na dobę, 365 dni w roku.

Zapewnienie bezpieczeństwa danych:
System na bieżąco ma wykonywać kopię zapasowe danych, aby w razie niepowodzenia związanego z utratą danych w ekspresowym tempie przywrócić wcześniejszy stan.

System również będzie przestrzegał ochrony danych osobowych zgodną z ogólnym rozporządzeniem o ochronie danych (RODO).
System będzie używał najnowszych technologii dot. szyfrowania danych, w celu ochrony przed wyciekiem.

Szybka reakcja systemu na akcje użytkownika:
System ma natychmiastowo wykonywać akcje użytkownika bez widocznych opóźnień.

Możliwość równoczesnej pracy wielu użytkowników:
System będzie korzystał z najnowocześniejszych technologii wielowątkowości, żeby zapewnić płynne i jednoczesne obsługiwanie wielu użytkowników systemu.

Łatwe serwisowanie systemu w przyszłości:
System zostanie napisany w najpopularniejszej wersji języka java, wraz z użyciem standardów pisania kodu takich jak SOLID, KISS, DRY i YAGNI, dzięki którym serwisowanie systemu w przyszłości będzie wymagało minimum dodatkowej pracy przy kodzie.

13. Opis przyszłej ewolucji systemu:

W przyszłości planuje się możliwe rozwinięcie systemów parku o nowy oddział, a nawet możliwość powstania nowego parku pod logiem firmy i potrzeba obsługi wielu wesołych miasteczek.

14. Omówienie decyzji projektowych:

Do implementacji systemu użyto języka Java z wykorzystaniem frameworku Spring. Serwerem bazy danych jest H2 w wersji 2.1.214 (2022-06-13), który będzie działał w trybie zapisu do pliku. Komunikacja między bazą danych a aplikacją jest realizowana za pomocą frameworku Hibernate. Te technologie zostały wybrane ze względu na prostotę użycia i skuteczność działania. Frontend został wykonany przy użyciu technologii React.js.

Realizacja poszczególnych elementów systemu odbywa się w następujący sposób:

- Zapewnienie trwałości wszystkich ekstensji zostało zrealizowane za pomocą bazy danych
- Atrybuty złożone są realizowane poprzez referencje do obiektów asocjacyjnych w klasach modelowych
- Atrybuty klasowe są przechowywane w pliku konfiguracyjnym appsettings.json. Konfiguracja jest ładowana w trakcie uruchamiania aplikacji i dostępna dla odpowiednich modułów.
- Atrybuty pochodne są obliczane dynamicznie na podstawie danych przechowywanych w bazie danych i dołączane do modelu zwracanego do interfejsu użytkownika.
- Asocjacje zwykłe realizowane będą poprzez relacje między tabelami w bazie danych. Dostęp do obiektów w relacji będzie realizowane przy pomocy kluczy obcych.
- Atrybuty relacji będą przechowywane w tabelach wiele do wiele jako tabele pośrednie z kolumnami przechowującymi atrybuty. Przykładem tego jest tabela OcenyPracownika.
- Kompozycja między Oceną, a Klientem będzie realizowana poprzez Trigger ON DELETE CASCADE zapewniający usunięcie wszystkich ocen powiązanych z danym klientem.
- Klasy dziedziczące po klasie abstrakcyjnej będą realizowane poprzez połączenia za pomocą relacji klucz główny – klucz obcy z wykorzystaniem InheritanceType.JOINED z Hibernate'a, dzięki czemu każda klasa będzie posiadała własną tabelę.

Decyzje podjęte w fazie projektowej pozwolą na skuteczne zaprogramowanie systemu i jego funkcjonalności i zapewnią możliwości łatwych i sprawnych zmian rozwojowych systemu dzięki prostej rozszerzalności klas i otwartości systemu. Użyte w projekcie technologie zostały wybrane ze względu na ich skuteczność działania, a także z uwagi na doświadczenie programisty, który miał z nimi wcześniej do czynienia.