МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное

учреждение высшего образования

«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ ВЫСОКИХ ТЕХНОЛОГИЙ И ПЬЕЗОТЕХНИКИ

Кафедра прикладной информатики и инноватики

КУРСОВАЯ РАБОТА

по дисциплине

«Архитектура информационных систем»

на тему

«Cервис для покупки и бронирования ЖД билетов»

Выполнил

студент 4 курса 3 группы

Комиссаров Владислав Андреевич

Принял

профессор, д.т.н.

Жмайлов Борис Борисович

Ростов-на-Дону

2022

1. Техническое задание

Предметная область курсовой работы связана с пользователями и информационными ресурсами, связанными с Ж/Д поездами, билетами и бронированием. Проект должен представлять из себя библиотеку в формате jar архива.

Необходимо создать сервис покупки и бронирования Ж/Д билетов. Проект должен представлять собой Java приложение, предоставляющее API (application programming interface).

Предусмотреть в системе базовые операции: создание, редактирование, удаление и получение основных сущностей, таких как билет, поезд, пассажир, льготник и скидка.

1. UML моделирование
   1. Диаграмма вариантов использования

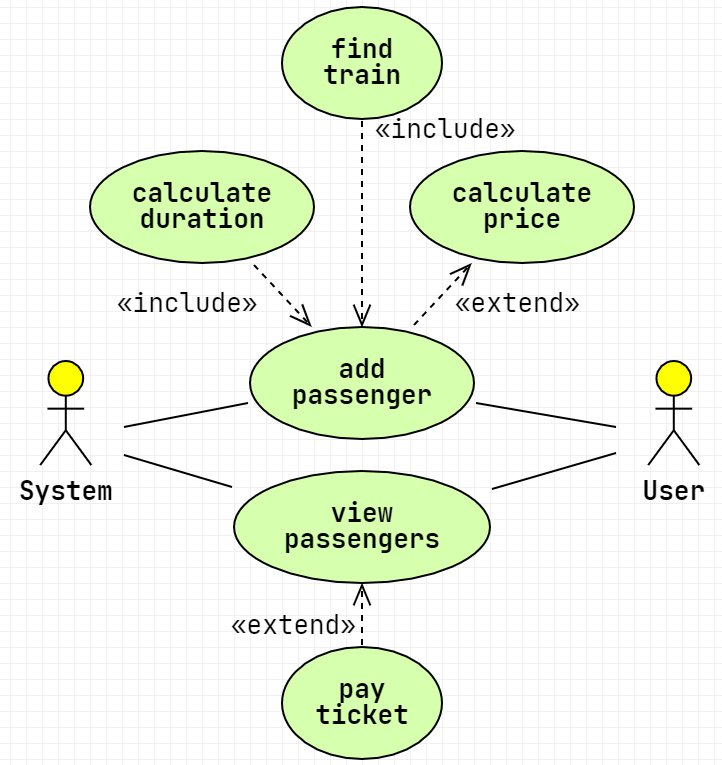


Схема 1. Диаграмма вариантов использования

Таблица 1. Детализация диаграммы вариантов использования.

|  |  |
| --- | --- |
| **Вариант использования** | **Описание** |
| **Add passenger** | Добавить пассажира на поезд. Если билет не оплачен, то он считается забронированным. Если поезд не найден, то билет не будет создан.  **Входные данные:** String from, String to, long passengerId, boolean isPaid;  **Выходные данные:** Optional<Ticket> ticket;  **Включающие методы:** «Find train» получает параметры из сигнатуры родительского метода, «Calculate duration» получает параметры во время выполнения;  **Расширяющий метод** «Calculate price» получает параметры во время выполнения и выполняется при passenger.discount != null. |
| Find train | Найти подходящий поезд.  **Входные данные:** String from, String to;  **Выходные данные:** Optional<Train> train. |
| Calculate duration | Расчёт продолжительности поездки.  **Входные данные:** String departure, String arrival;  **Выходные данные:** String duration. |
| Calculate price | Расчёт цены поездки с учётом скидки.  **Входные данные:** double price, double discount;  **Выходные данные:** double cost. |
| **View passengers** | Просмотр пассажиров поезда.  **Входные данные:** long trainId, long ticketId;  **Выходные данные:** List<Passenger> passengers;  **Расширяющий метод** «Pay ticket» получает параметры из сигнатуры родительского метода и выполняются при ticketId != 0. |
| Pay ticket | Оплатить забронированный билет.  **Входные данные:** long ticketId;  **Выходные данные:** Optional<Ticket> ticket; |

* 1. Диаграмма классов

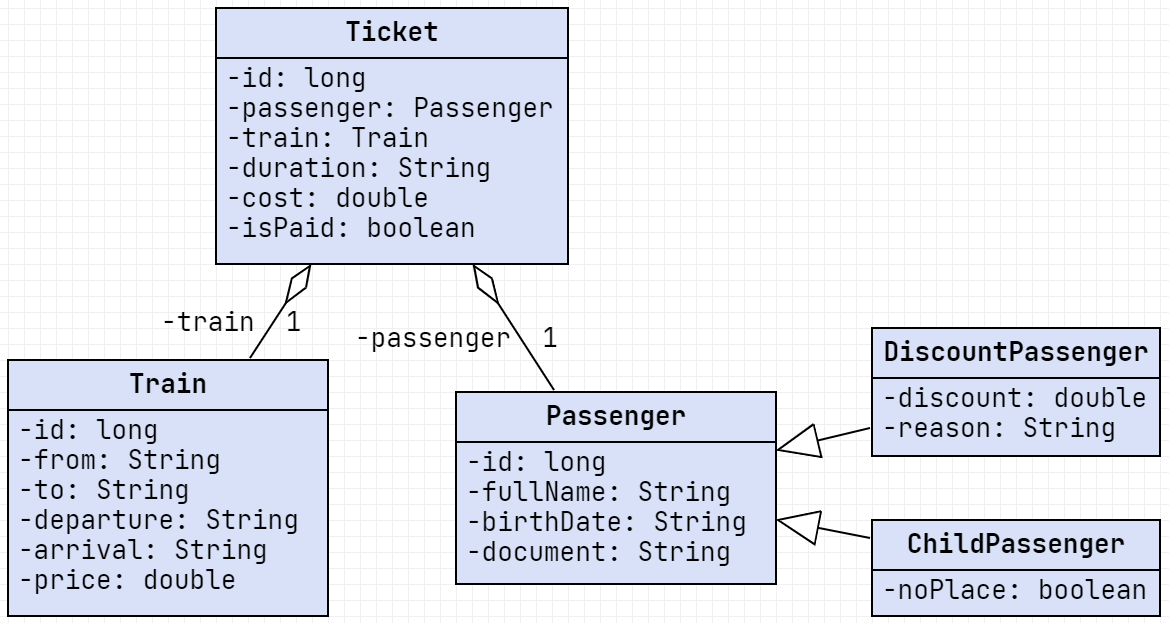


Схема 2. Диаграмма классов

* 1. Диаграммы активностей

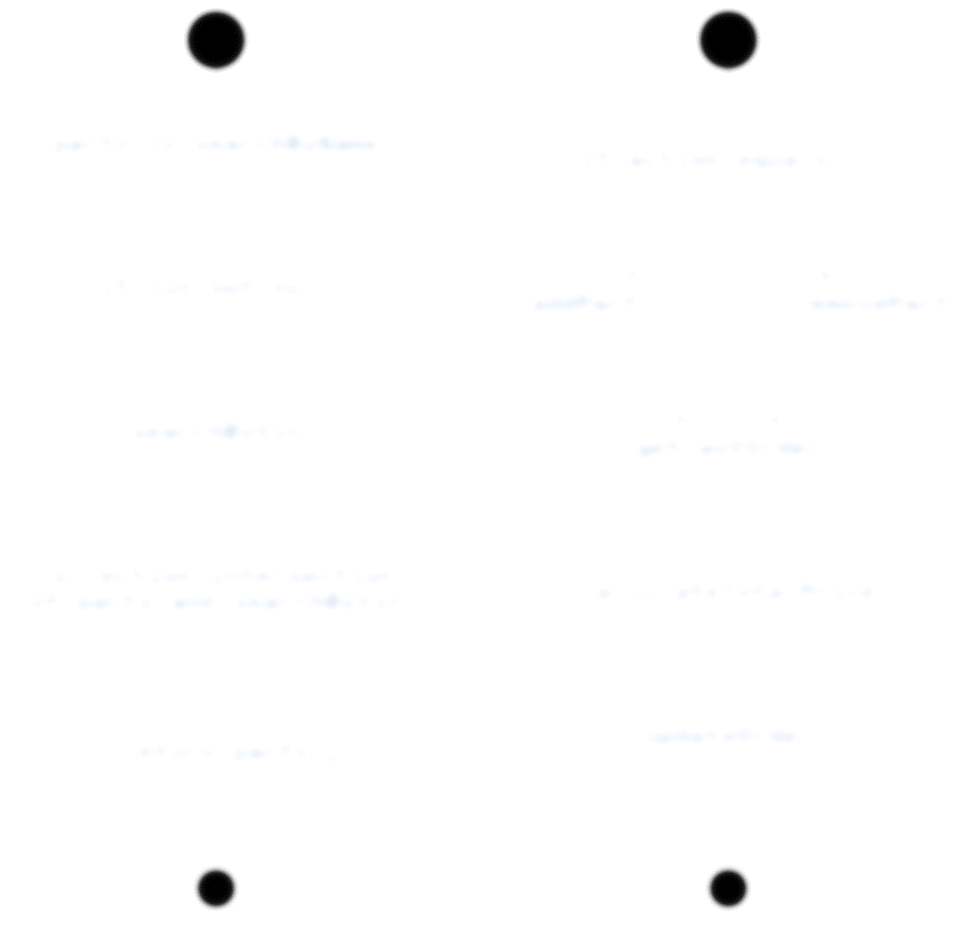


Схема 3. Диаграммы активностей методов «Add passenger» и «View Passengers»

* 1. Диаграмма компонентов

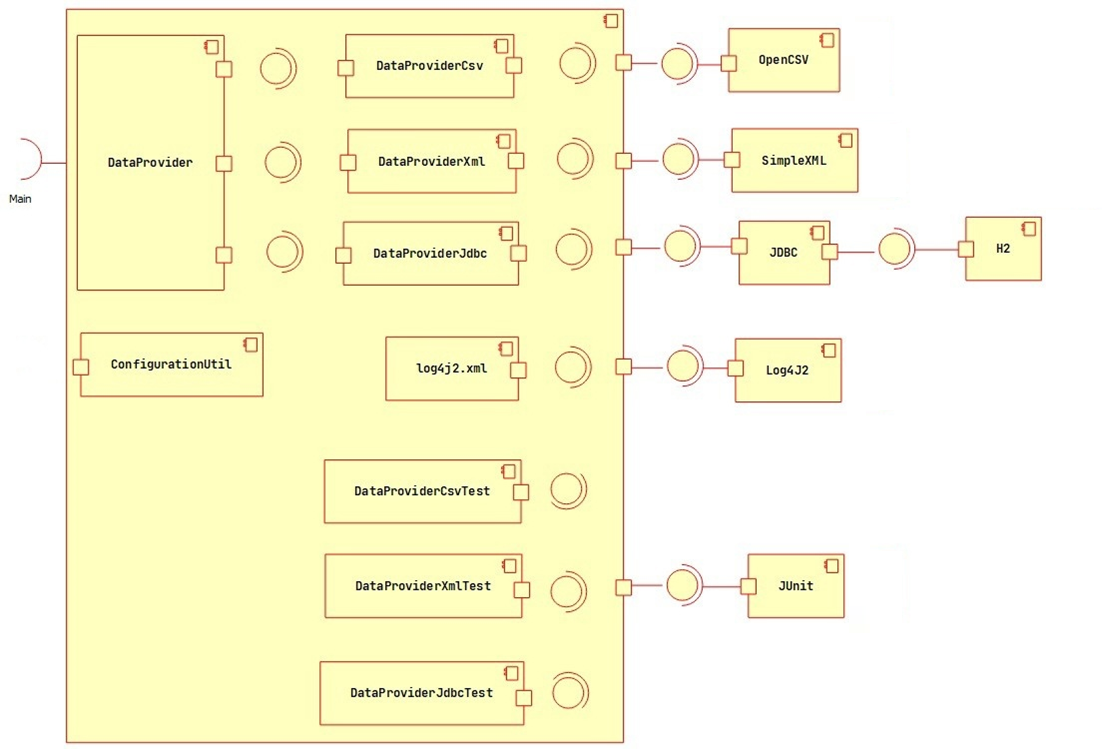


Схема 4. Диаграмма компонентов

1. Сценарии тестирования и результаты тестов

Таблица 2. Сценарии и результаты тестирования класса api.AbstractDataProvider

|  |  |  |  |
| --- | --- | --- | --- |
| **Тестовый метод** | **Тип сценария** | **Тестируемый метод** | **Что контролируется?** |
| addPassengerPos | позитивный | addPassenger | assertEquals(expectedTicket, actualTicket) |
| addPassengerNeg | негативный | addPassenger | assertNull(actualTicket) |
| findTrainPos | позитивный | findTrain | assertEquals(expectedTrain, actualTrain) |
| findTrainNeg | негативный | findTrain | assertNull(actualTrain) |
| calculateDurationPos | позитивный | calculateDuration | assertEquals(expectedDuration, actualDuration) |
| calculateDurationNeg | негативный | calculateDuration | assertThrows(ArrayIndexOutOfBoundsException.class, () -> dp.calculateDuration("", "")) |
| calculatePricePos | позитивный | calculatePrice | assertEquals(expectedPrice, actualPrice) |
| calculatePriceNeg | негативный | calculatePrice | assertNotEquals(unexpectedPrice, actualPrice) |
| viewPassengersPos | позитивный | viewPassengers | assertEquals(expectedPassengers, actualPassengers) |
| viewPassengersNeg | негативный | viewPassengers | assertEquals(expectedPassengers, actualPassengers) |
| payTicketPos | позитивный | payTicket | assertEquals(expectedTicket, actualTicket) |
| payTicketNeg | негативный | payTicket | assertNull(actualTicket) |