Федеральное государственное автономное образовательное учреждение высшего образования «**Национальный исследовательский университет ИТМО**»

Факультет Программной Инженерии и Компьютерной Техники
Дисциплина: «Информационные Системы»

Отчет
По лабораторной работе №2
Вариант 3000

Преподаватель:
Тюрин Иван Николаевич

Выполнил:
Федоров Евгений Константинович

Группа: P3310

Санкт-Петербург 2025

Содержание

1. Задание

Реализовать информационную систему, которая позволяет взаимодействовать с объектами класса Ticket, описание которого приведено ниже:

```
public class Ticket {
    private Integer id; //Поле не может быть null, Значение поля должно быть больше 0, Значение этого поля должно быть уникальным, Значение этого поля должно генерироваться автоматически
    private String name; //Поле не может быть null, Строка не может быть пустой
    private Coordinates coordinates; //Поле не может быть null
    private java.time.LocalDateTime creationDate; //Поле не может быть null, Значение этого поля должно генерироваться автоматически
    private Person person; //Поле может быть null
    private Event event; //Поле может быть null
    private float price; //Значение поля должно быть больше 0
    private TicketType type; //Поле не может быть null
    private Float discount; //Поле может быть null, Значение поля должно быть больше 0, Максимальное значение поля: 100
    private int number; //Значение поля должно быть больше 0
    private String comment; //Поле может быть null
    private Venue venue; //Поле может быть null
}
public class Coordinates {
    private int x;
    private Float y; //Поле не может быть null
}
public class Person {
    private Color eyeColor; //Поле может быть null
    private Color hairColor; //Поле не может быть null
    private Location location; //Поле не может быть null
    private Double weight; //Поле не может быть null, Значение поля должно быть больше 0
    private String passportID; //Поле не может быть null
    private Country nationality; //Поле не может быть null
}
```

```
public class Event {
    private Integer id; //Поле не может быть null, Значение поля должно быть больше 0, Значение этого поля должно быть уникальным, Значение этого поля должно генерироваться автоматически
    private String name; //Поле не может быть null, Строка не может быть пустой
    private Integer ticketsCount; //Поле не может быть null, Значение поля должно быть больше 0
    private EventType eventType; //Поле может быть null
}
public class Venue {
    private Long id; //Поле не может быть null, Значение поля должно быть больше 0, Значение этого поля должно быть уникальным, Значение этого поля должно генерироваться автоматически
    private String name; //Поле не может быть null, Строка не может быть пустой
    private int capacity; //Значение поля должно быть больше 0
    private VenueType type; //Поле может быть null
}
public class Location {
    private Integer x; //Поле не может быть null
    private float y;
    private Float z; //Поле не может быть null
}
public enum TicketType {
    VIP,
    USUAL,
    BUDGETARY,
    CHEAP;
}
public enum Color {
    GREEN,
    RED,
    ORANGE,
    WHITE,
    BROWN;
}
public enum Country {
    GERMANY,
    INDIA,
    THAILAND,
    SOUTH_KOREA,
    JAPAN;
}
```

```
public enum EventType {
    CONCERT,
    FOOTBALL,
    BASEBALL,
    BASKETBALL,
    OPERA;
}
public enum VenueType {
    LOFT,
    OPEN_AREA,
    STADIUM;
}
```

Разработанная система должна удовлетворять следующим требованиям:

- Основное назначение информационной системы - управление объектами, созданными на основе заданного в варианте класса.
- Необходимо, чтобы с помощью системы можно было выполнить следующие операции с объектами: создание нового объекта, получение информации об объекте по ИД, обновление объекта (модификация его атрибутов), удаление объекта. Операции должны осуществляться в отдельных окнах (интерфейсах) приложения.При получении информации об объекте класса должна также выводиться информация о связанных с ним объектах.
- При создании объекта класса необходимо дать пользователю возможность связать новый объект с объектами вспомогательных классов, которые могут быть связаны с созданным объектом и уже есть в системе.
- Выполнение операций по управлению объектами должно осуществляться на серверной части (не на клиенте), изменения должны синхронизироваться с базой данных.
- На главном экране системы должен выводиться список текущих объетов в виде таблицы (каждый атрибут объекта - отдельная колонка в таблице). При отображении таблицы должна использоваться пагинация (если все объекты не помещаются на одном экране).
- Нужно обеспечить возможность фильтровать/сортировать строки таблицы, которые показывают объекты (по значениям любой из строковых колонок). Фильтрация элементов должна производиться по неполному совпадению.

- Переход к обновлению (модификации) объекта должен быть возможен из таблицы с общим списком объектов и из области с визуализацией объекта (при ее реализации).
- При добавлении/удалении/изменении объекта, он должен автоматически появиться/исчезнуть/измениться в интерфейсах у других пользователей, авторизованных в системе.
- Если при удалении объекта с ним связан другой объект, связанные объекты должны быть связаны с другим объектом (по выбору пользователя), а изначальный объект удален.
- Пользователи должны иметь возможность просмотра всех объектов. Для модификации объекта должно открываться отдельное диалоговое окно. При вводе некорректных значений в поля объекта должны появляться информативные сообщения о соответствующих ошибках.

В системе должен быть реализован отдельный пользовательский интерфейс для выполнения специальных операций над объектами:

- Удалить все объекты, значение поля comment которого эквивалентно заданному.
- Вернуть один (любой) объект, значение поля event которого является минимальным.
- Вернуть количество объектов, значение поля comment которых меньше заданного.
- Продать билет за указанную сумму указанному человеку.
- Скопировать указанный билет, создав такой же, но с категорией «VIP» и с удвоенной ценой.
- Представленные операции должны быть реализованы в качестве функций БД, которые необходимо вызывать из уровня бизнес-логики приложения.

Особенности хранения объектов, которые должны быть реализованы в системе:

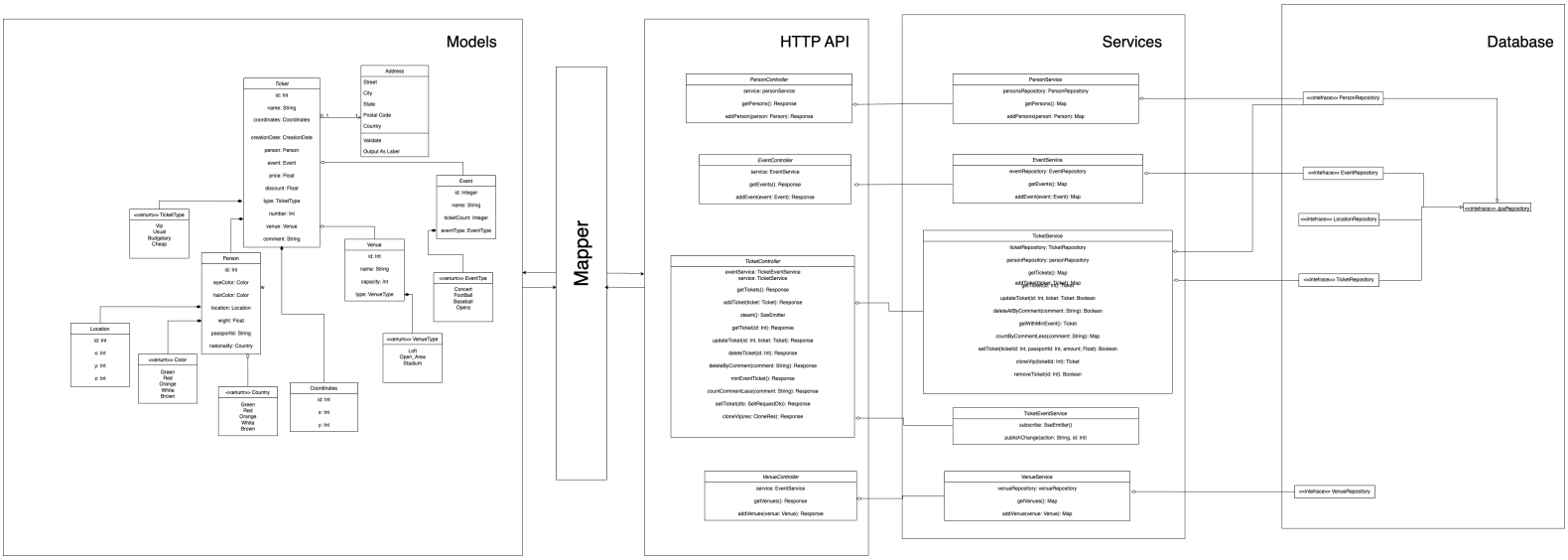
- Организовать хранение данных об объектах в реляционной СУБД (PostgreSQL). Каждый объект, с которым работает ИС, должен быть сохранен в базе данных.
- Все требования к полям класса (указанные в виде комментариев к описанию классов) должны быть выполнены на уровне ORM и БД.
- Для генерации поля id использовать средства базы данных.

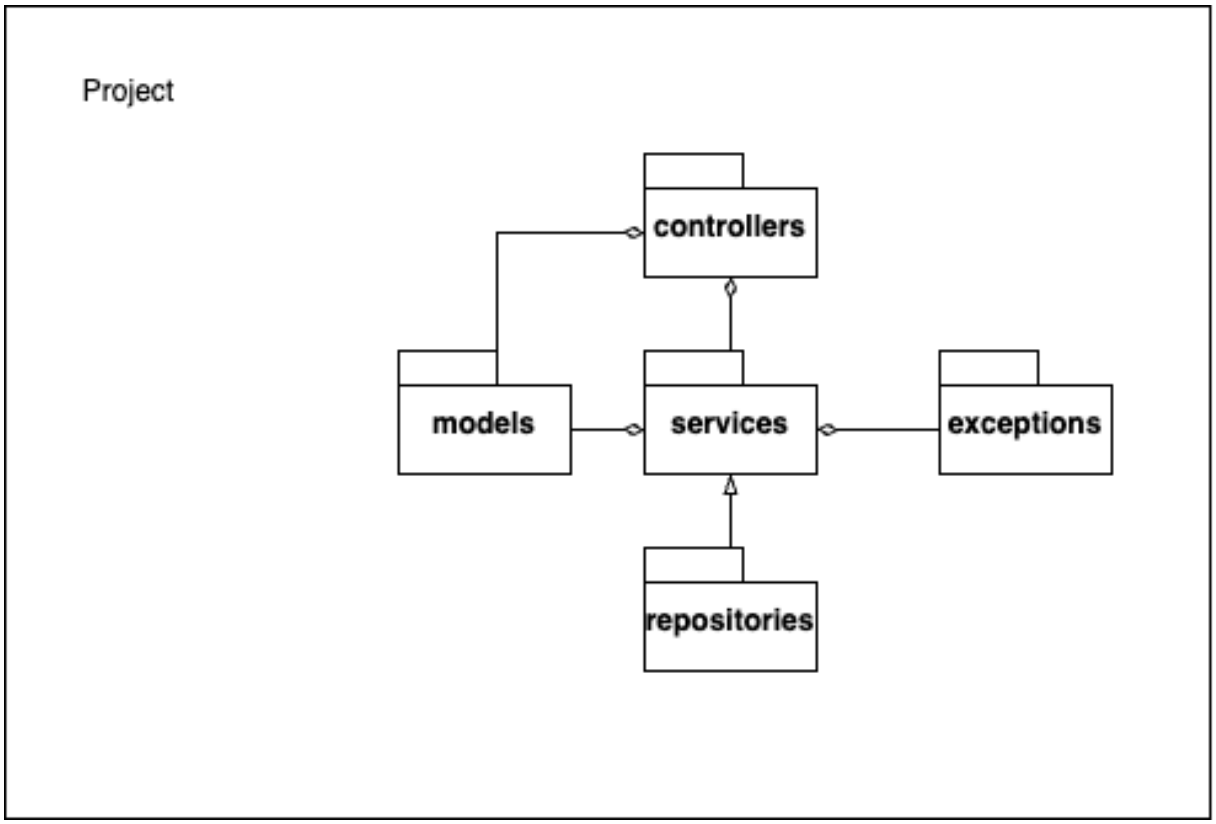
- Для подключения к БД на кафедральном сервере использовать хост pg, имя базы данных - studs, имя пользователя/пароль совпадают с таковыми для подключения к серверу.
- При создании системы нужно учитывать следующие особенности организации взаимодействия с пользователем:
- Система должна реагировать на некорректный пользовательский ввод, ограничивая ввод недопустимых значений и информируя пользователей о причине ошибки.
- Переходы между различными логически обособленными частями системы должны осуществляться с помощью меню.
- При добавлении/удалении/изменении объекта, он должен автоматически появиться/исчезнуть/измениться на области у всех других клиентов.

При разработке ИС должны учитываться следующие требования:

- В качестве основы для реализации ИС необходимо использовать Spring MVC.
- Для создания уровня хранения необходимо использовать Hibernate.
- Разные уровни приложения должны быть отделены друг от друга, разные логические части ИС должны находиться в отдельных

2. UML-диаграммы классов и пакетов разработанного приложения




Project
controllers
models
services
exceptions
repositories

3. Ссылка на репозиторий с исходным кодом

GitHub: https://github.com/2BuRy1/Is_lab2/tree/main

4. Вывод

В ходе выполнения лабороторной работы разобрался в уровне изоляций транзакций, научился писать нагрузочные тесты при помощи Apache Jmeter. Реализовал дополнительные ограничения на уровне бизнес-логики.