Федеральное государственное автономное
образовательное учреждение высшего образования
«Научно-образовательная корпорация ИТМО»

Факультет программной инженерии и компьютерной техники
Направление подготовки 09.03.04 Программная инженерия

Отчёт по лабораторной работе №1

По дисциплине «Информационные системы» (семестр 5)

Студент:
Дениченко Александр P3312
Практик:
Бострикова Д.К.

Санкт-Петербург
2024 г.

Задание

Лабораторные работы

Лабораторная работа #1

Введите вариант: 367193

Внимание! У разных вариантов разный текст задания!

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

```
public class Vehicle {
    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 автома
    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 VehicleType type; //Поле не может быть null
    private Double enginePower; //Поле не может быть null, Значение поля должно быть больше 0
    private long numberOfWheels; //Значение поля должно быть больше 0
    private Long capacity; //Поле не может быть null, Значение поля должно быть больше 0
    private Double distanceTravelled; //Поле не может быть null, Значение поля должно быть больше 0
    private Float fuelConsumption; //Поле не может быть null, Значение поля должно быть больше 0
    private FuelType fuelType; //Поле может быть null
}
public class Coordinates {
    private Long x; //Значение поля должно быть больше -308, Поле не может быть null
    private Double y; //Поле не может быть null
}
public enum VehicleType {
    PLANE,
    BOAT,
    BICYCLE;
}
public enum FuelType {


public enum FuelType {
    KEROSENE,
    ELECTRICITY,
    DIESEL,
    MANPOWER,
    PLASMA;
}
```

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.
- Пользователю системы должен быть предоставлен интерфейс для авторизации/регистрации нового пользователя. У каждого пользователя должен быть один пароль. Требования к паролю: пароль должен быть содержать не менее n символов. В системе предполагается использование следующих видов пользователей (ролей):обычные пользователи и администраторы. Если в системе уже создан хотя бы один администратор, зарегистрировать нового администратора можно только при одобрении одним из существующих администраторов (у администратора должен быть реализован интерфейс со списком заявок и возможностью их одобрения).
- Редактировать и удалять объекты могут только пользователи, которые их создали, и администраторы (администраторы могут редактировать и удалять объекты, которые пользователь разрешил редактировать при создании).
- Зарегистрированные пользователи должны иметь возможность просмотра всех объектов, но модифицировать (обновлять) могут только принадлежащие им (объект принадлежит пользователю, если он его создал). 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 fuelType которого эквивалентно заданному.

- Вернуть один (любой) объект, значение поля enginePower которого является минимальным.
- Вернуть массив объектов, значение поля name которых начинается с заданной подстроки.
- Найти все транспортные средства заданного типа.
- Найти все транспортные средства с числом колёс в заданном диапазоне.

Представленные операции должны быть реализованы в рамках компонентов бизнес-логики приложения без прямого использования функций и процедур БД.

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.
- Пароли при хранении хэшировать алгоритмом SHA-384.
- При хранении объектов сохранять информацию о пользователе, который создал этот объект, а также фиксировать даты и пользователей, которые обновляли и изменяли объекты. Для хранения информации о пользователях и об изменениях объектов нужно продумать и реализовать соответствующие таблицы.
- Таблицы БД, не отображающие заданные классы объектов, должны содержать необходимые связи с другими таблицами и соответствовать 3НФ.
- 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:

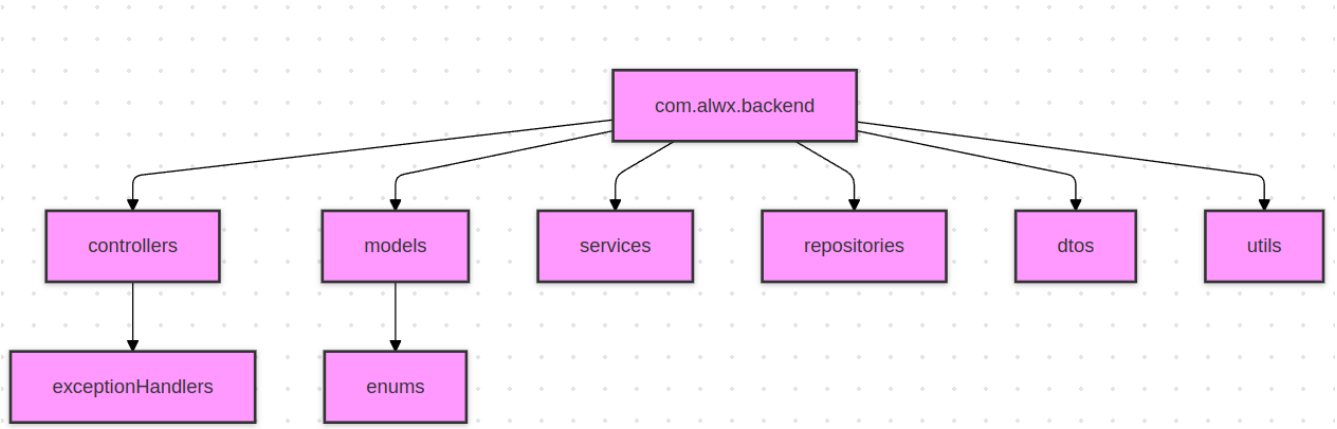
- Система должна реагировать на некорректный пользовательский ввод, ограничивая ввод недопустимых значений и информируя пользователей о причине ошибки.
- Переходы между различными логически обособленными частями системы должны осуществляться с помощью меню.
- Во всех интерфейсах системы должно быть реализовано отображение информации о текущем пользователе (кто авторизован) и предоставляться возможность изменить текущего пользователя.
- [Опциональное задание - +2 балл] В отдельном окне ИС должна осуществляться визуализация объектов коллекции. При визуализации использовать данные о координатах и размерах объекта. Объекты от разных пользователей должны быть нарисованы разными цветами. При нажатии на объект должна выводиться информация об этом объекте.
- 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 JPA + Hibernate.
- Разные уровни приложения должны быть отделены друг от друга, разные логические части ИС должны находиться в отдельных компонентах.

1 UML диаграммы

1.1 Диаграмма пакетов



1.2 Диаграмма классов

Ссылка на диаграмму классов

2 Ссылка на проект

Ссылка на проект

3 Доказательство 3НФ формы таблиц базы данных

1. User:
   Ключ: id (простой ключ).
   Атрибуты: id, username, password. Все атомарные.
   Связь с Role: реализована через отдельную таблицу users_roles, что соответствует принципам нормализации.
Вывод: User в 3НФ, так как у него простой ключ и нет транзитивных зависимостей.

2. Role:
Ключ: id (простой ключ).
Атрибуты: id, name. Все атомарные.
Вывод: Role в 3НФ, так как у него простой ключ и нет транзитивных зависимостей.

3. Vehicle:
Ключ: id (простой ключ).
Атрибуты: id, name, creationDate, type, enginePower, numberOfWheels, capacity, distanceTravelled, fuelConsumption, fuelType, permissionToEdit. Все атомарные.
Связь с Coordinates: coordinates_id является внешним ключом, ссылающимся на Coordinates. Это функциональная зависимость, но не транзитивная.
Связь с User: реализована через отдельную таблицу vehicle_user, что соответствует принципам нормализации.
Вывод: Vehicle в 3НФ.

4. Coordinates:
Ключ: id (простой ключ).
Атрибуты: id, x, y. Все атомарные.
Вывод: Coordinates в 3НФ.

5. UserAction:
Ключ: id (простой ключ).
Атрибуты: id, action, vehicleId, timestamp, user_id (внешний ключ, ссылающийся на User). Все атомарные.
Зависимости: user_id -> username (через таблицу User). Это не транзитивная зависимость, так как user_id часть ключа.
Вывод: UserAction в 3НФ.

6. RequestForRights:
Ключ: id (простой ключ).
Атрибуты: id, username. Все атомарные.
Вывод: RequestForRights в 3НФ.

4 Выводы

В ходе выполнения лабораторной работы, были изучены основные принципы проектирования информационный системы, при помощи которой можно управлять некоторым набором объектов. Хранение данных организованно на уровне базы данных, любые действия, связанные с редактирование данных происходят на уровне сервера, без использования функционала базы данных (она помогает структурировать данные и накладывать некоторые ограничения на поля). На уровне сервера были разработанны контроллеры, которые обрабатывают запросы пользователей, и перенаправляют управление в сервисы, которые выполняют всю суть редактирования данных и логику приложения. Взаимодействие с базой данных организованно через CRUD операции. Для упрощения взаимодействия используется спецификация JPA и её реализация по работе с упаковкой и распаковкой данных Hibernate.