|  |  |  |
| --- | --- | --- |
| Санкт-Петербургский политехнический университет Петра Великого  Институт компьютерных наук и технологий  **Высшая школа программной инженерии** | | |
| **КУРСОВАЯ РАБОТА** | | |
| Вариант 1: Автоматизация работы автопарка | | |
| по дисциплине «Объектно-ориентированное программирование» | | |
| Выполнил | | |
| студент гр. 3530904/80005 |  | Фролов Н.В. |
| Руководитель | | |
| преподаватель |  | Аристаева О.Ц. |
|  | | |
| Санкт-Петербург | | |
| 2020 | | |

# Задание:

# Разработать клиент-серверное приложение на заданную тему (см. распределение в приложенном файле). Список обязательных к использованию технологий при выполнении работы:

1. Клиент:

· требования отсутствуют

2. Сервер:

· Java 12 и выше

· База данных на выбор - SLQLite/PostreSQL/MS SQL/Oracle. Схему БД, соответствующую заданию, см. ниже.

· SpringData/Hibernate для работы с БД

· Spring Security

· Взаимодействие с клиентом осуществляется посредством REST API.

# Описание архитектуры:

Самая обычная архитектура: От Базы данных к Серверу, от Сервера к Клиенту, от Клиента к Серверу и рекурсия. Две роли: User и Admin. User – водитель, который берется за определенные маршруты, Admin- формирует различные поездки и отслеживают его статус.

БД

Клиент

Сервер

# UML-диаграмма:

Entity:

Изображение выглядит как снимок экрана

Автоматически созданное описание

RestController(BugTracker)Изображение выглядит как снимок экрана, монитор

Автоматически созданное описание

Services and Repository(Of All Entity):

Изображение выглядит как снимок экрана

Автоматически созданное описание

Изображение выглядит как снимок экрана

Автоматически созданное описание

Изображение выглядит как снимок экрана

Автоматически созданное описаниеJwt:

Изображение выглядит как сидит, компьютер, припаркован, красный

Автоматически созданное описаниеОбщий вид Диаграммы:

# Схема Действующей Диаграммы (MySQL):

Изображение выглядит как снимок экрана, черный, держит, монитор

Автоматически созданное описание

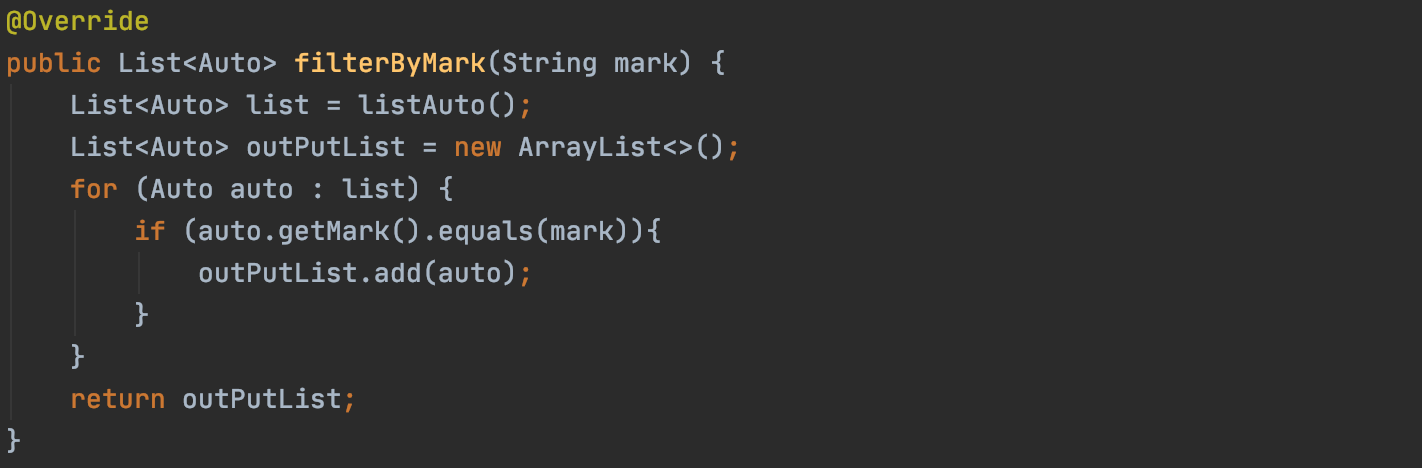
# Rest API

**Со стороны Сервера:** Создаются классы BugTracker и AuthController которые содержат все основные Get и Post запросы. Можно разбирать конкретные примеры, но суть по сути одна: Через АНАТАЦИИ создаем ссылку для нашего запроса, пишем определенный метод, который будет выполнять какие-то действия с нашими репозеториями. Давайте на примере посмотрим конкретный метод.

Изображение выглядит как стол

Автоматически созданное описание

Метод Get, который должен нам вытащить из репозитория все машины, у которых одна и та же марка.



Метод listAuto() берет все Auto которые у нас уже есть в базе данных, потом мы формируем новый лист и отправляем обратно, где после приводим наш получившийся Лист к ResponseEntity специальному классу для ответов на запросы к серверу.

Изображение выглядит как снимок экрана, телефон, мобильный телефон, сидит

Автоматически созданное описание**Со стороны Клиента:** Мы отправляем OkHtttp запрос, получаем ответ и уже потом парсим JSON который мы получили из ответа.

Изображение выглядит как снимок экрана, текст, сидит, серебряный

Автоматически созданное описание

# Spring Security

Для защиты наших данных мы используем token. Идея в том, что мы формируем из нашего пароля и username’a token, который уже в дальнейшем будем использовать для авторизации.

Как это выглядит со стороны клиента. Изображение выглядит как снимок экрана

Автоматически созданное описание

То есть клиента вводит пароль и логин, после в классе JwtProvider проверяется введенные нами данные и формируется токен. Так же в configure в классе SpringSecurityConfigure мы прописываем, какими классами, какая роль может пользоваться, соответсвенно.

Создание токена:

Изображение выглядит как снимок экрана, сидит, телефон, стол

Автоматически созданное описание

Запрос, который в дальнейшем проверяет пароль и логин в repository.

Изображение выглядит как снимок экрана, сидит, телефон, ноутбук

Автоматически созданное описание

# Клиент

Задумка моего клиента такая. Есть две роли, user и admin. Admin задает новые пути, тем самым порождая в Journals новой заметкой. User же либо берется за этот маршрут, находя себя в списке и выбрав машину, либо регистрирует нового водителя и новую машину. Админ видит выезд машины и её прибытие обратно в автопарк, так же можно фильтровать маршруты и просматривать кто поехал, а кто остался.

Вначале необходимо пройти идентификацию.Изображение выглядит как снимок экрана

Автоматически созданное описание

Окно Admin’a:

Изображение выглядит как снимок экрана

Автоматически созданное описание

Окно User’a:

Изображение выглядит как снимок экрана

Автоматически созданное описание

Изображение выглядит как снимок экрана

Автоматически созданное описаниеИзображение выглядит как снимок экрана

Автоматически созданное описание Регистрация AutoPersonnel и Auto

Выбор autoPersonnel и Auto(Оба объекта представлены в виде выплывающего списка)

Изображение выглядит как снимок экрана

Автоматически созданное описание

# Заключение

Создали сервер с помощью фреймворка Spring и клиент сервер на Swing. Научились создавать запросы, сущности и репозитории. Полученный весомый опыт в разработки фронтенда на Java. Была освоена сложная технология защиты персональных данных. В общем и целом, пол получен огромный опыт в разработке приложения, от самого его начала и общей логики и архитектуры.