Московский государственный технический университет им. Н.Э. Баумана

Факультет «Информатика и системы управления»

Кафедра «Системы обработки информации и управления»



**Отчет**

**По домашнему заданию**

**По курсу «Разработка интернет-приложений»**

**ИСПОЛНИТЕЛЬ:**

Группа ИУ5-54Б

Хатин М.С.

Москва 2022

**1. Введение**

Цель:

Создать интернет-приложение – онлайн магазин для продажи музыкальных пластинок.

Задачи:

Реализовать интернет-приложение, включающее в себя веб-сервис и пользовательский интерфейс в виде сайта.­­­­­­

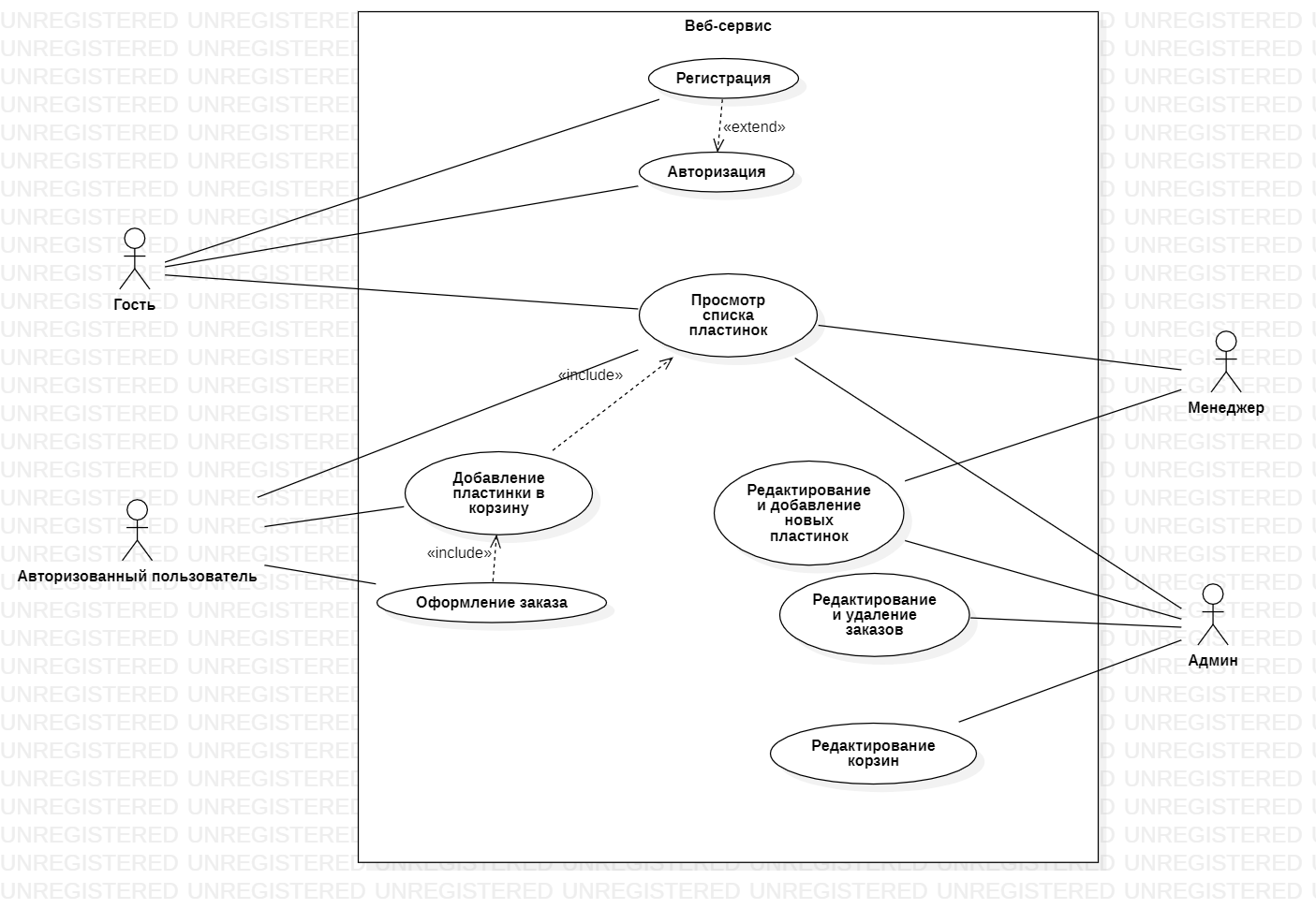
Актуальность:

Несмотря на то, что в наше время существуют стриминговые сервисы типа iTunes, Spotify, Deezer и др., многие люди продолжают слушать музыку на виниловых проигрывателях, поскольку пластинки имеют более живое и объёмное звучание (при наличии хорошего проигрывателя и усилителя). А также это довольно стильно смотрится – особенно хорошо пластинки вписываются в винтажный интерьер. Но есть один минус – за ними нужно идти в магазин.

В этой связи идея с предоставлением возможности заказать пластинку онлайн – очень актуальна. Поскольку любой музыкальный гик – не хочет тратить время на проезд до нужного магазина (возможно в не лучшую погоду), на стояние в очереди у кассы. В моём приложении часть из этих этапов опускается – достаточно выбрать товар, корректно оформить заказ, нажав несколько кнопок, и пластинка уже ждёт тебя – никаких очередей.

**2. Бизнес-процесс**

Диаграмма прецендентов:



Всего в приложении представлено 4 вида пользователей

Ниже приведу пояснение для каждого из них, дополняющее диаграмму прецедентов:

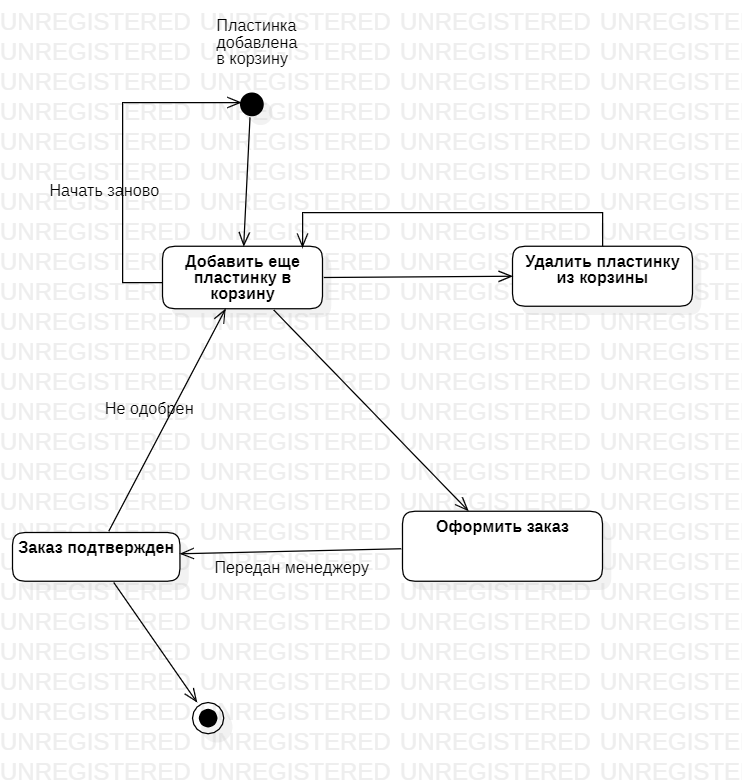
1) Гость, или же анонимный пользователь – может зарегистрироваться, войти в аккаунт или просто просмотреть ассортимент пластинок – но не может добавить товар в корзину.

2)Авторизованный пользователь – он может так же, как и остальные виды пользователей, просматривать список товаров, а также может добавлять товары в корзину и оформлять заказ.

3)Менеджер – так же, как и в случае с анонимным и авторизованным пользователями, может просматривать список товаров. Кроме этого, редактирует данные о товарах и может добавлять новые.

4)Админ – имеет больше всего прав. Он может удалять товары, редактировать информацию о них, удалять заказы пользователей, редактировать корзины пользователей.

Диаграмма состояний корзины:

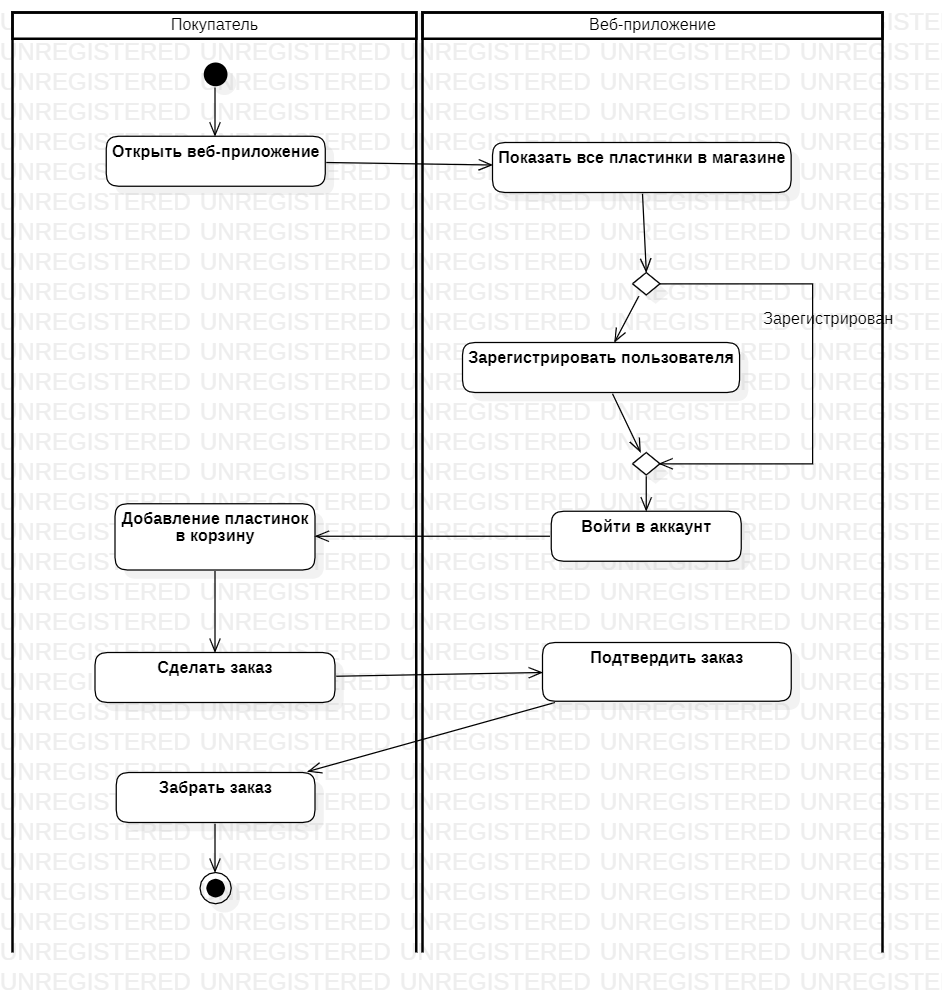


Сначала пользователь добавляет пластинку в корзину.

Если пользователь очень любит музыку, он может добавить ещё товаров в корзину. Но также он может и удалять пластинки из корзины.

Когда все желаемые пластинки добавлены, и корзина сформирована, пользователь оформляет заказ. Затем менеджер получает информацию об этом заказе. После чего у менеджера имеется 2 опции – одобрить этот заказ или нет. Если менеджер не одобрил заказ, то пользователь либо удаляет пластинки из корзины, либо добавляет еще товары. Финальное состояние корзины наступает, когда менеджер одобряет заказ.

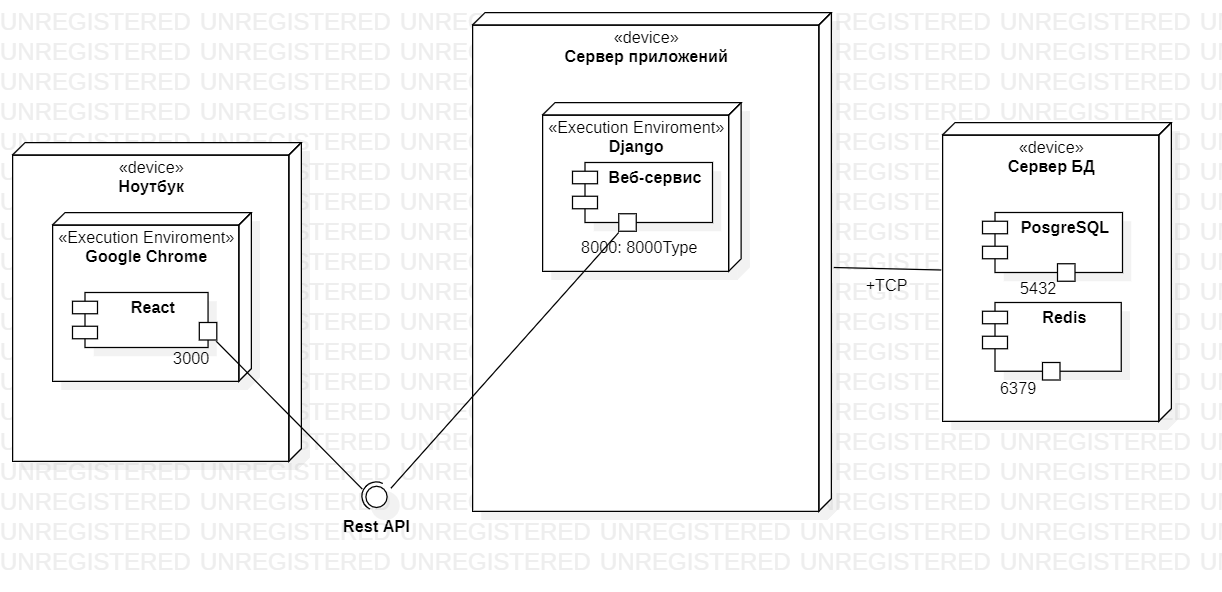
Диаграмма деятельности:



После открытия веб-приложения на главной странице магазина отображается список всех пластинок, которые доступны в магазине. После этого пользователь может зарегистрировать новую учётную запись либо же войти в уже существующий аккаунт. Когда пользователь прошёл авторизацию, ему открывается возможность добавлять нужные товары в корзину и оформлять заказ. Веб-приложение, а точнее менеджер, этот заказ подтверждает. После чего пользователь может забрать свой заказ в магазине.

**3. Архитектура**

Диаграммы развертывания:



На сервере приложений лежит веб-сервис Django, который предоставляет порт 8000.

На стороне клиента в браузере находится React-сервер, который обращается к веб-сервису на Django через порт 3000.

Веб-сервис с помощью протокола TCP обращается к серверу БД: PostgreSQL и Redis, предоставляющих 5432 и 6379 порты соответственно.

ER-диаграмма:



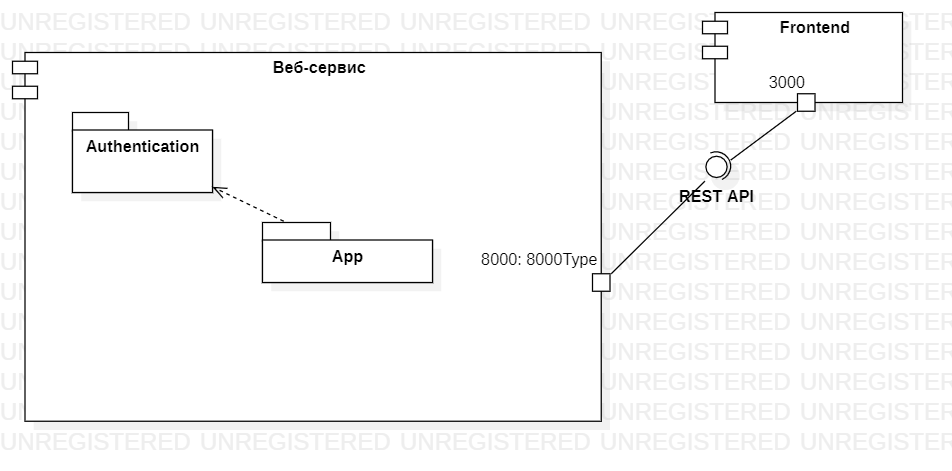
Сущность Record – содержит информацию о товаре для веб-сервиса, в нашем случае это музыкальная пластинка. В ней есть такие поля, как жанр (genre), название (name), цена (price) и популярность (popularity).

Сущность Cart – пользовательская корзина, в ней хранятся пластинки пользователей через сущность посредник – cart\_records (связь один-ко-многим с сущностями Cart и Record). Также в сущности корзины есть поле user\_id – связь один-к-одному с сущностью пользователя – User.

Сущность User – пользователь. В этой сущности представлены такие поля как: имя пользователя (username), пароль (password), дата последнего входа в сервис (last\_login), является ли пользователь администратором (is\_superuser), является ли пользователь менеджером (is\_staff), почта (email), активен ли пользователь (is\_active), даты создания и обновления аккаунта (updated\_at).

Так же есть сущность Order – заказ пользователя, в ней хранится дата заказа и связь один-ко-многим с таблицей User. Также через сущность посредника order\_records в таблице Order хранятся товары. Order\_records связана с сущностями Order и Record связью один-ко-многим.

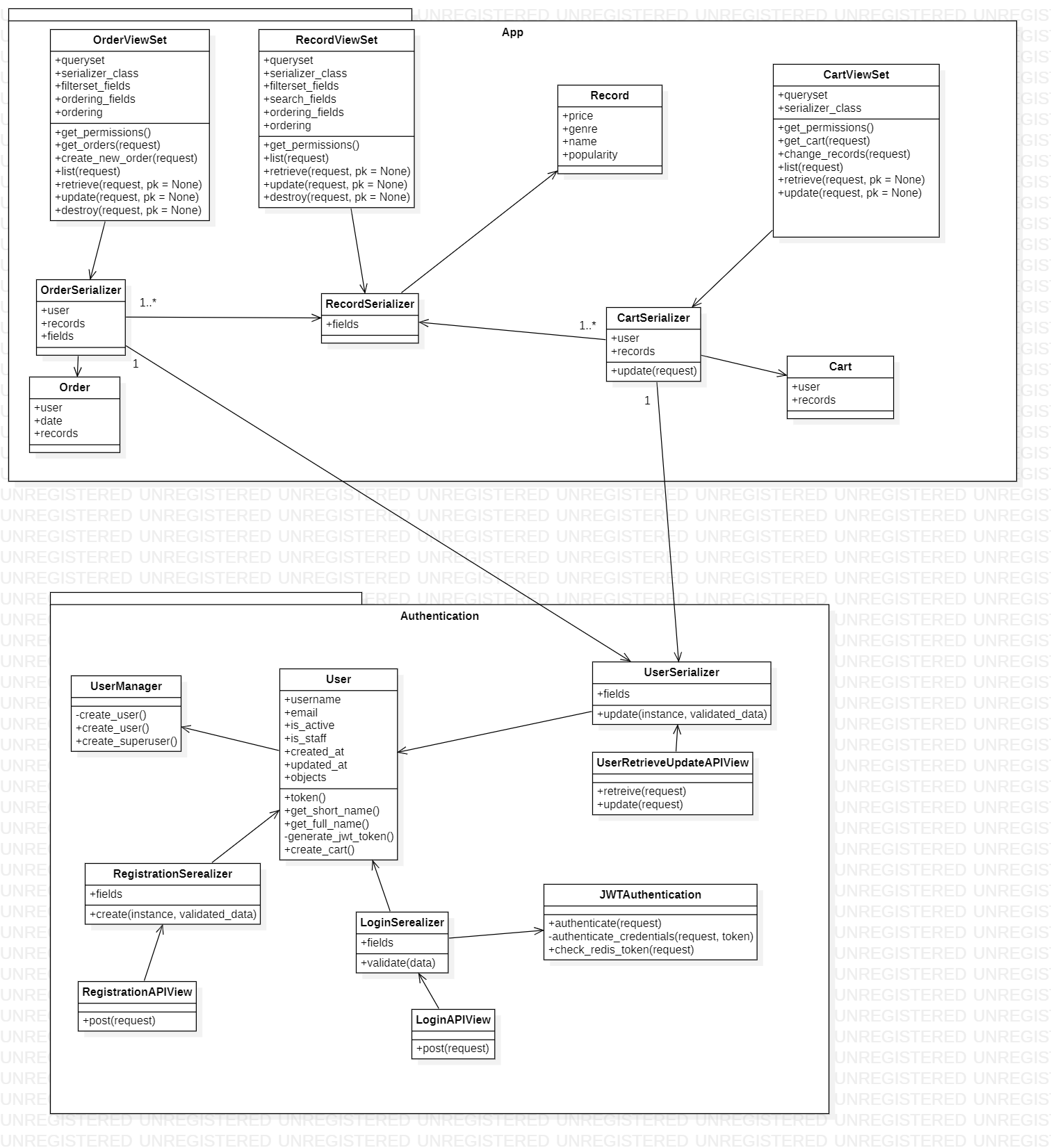
Диаграмма компонентов:



Интернет-приложение состоит из двух компонентов – веб-сервис и Frontend, которые передают данные через REST API протокол. Веб-сервис через 8000 порт, Frontend через 3000 порт.

Веб-сервис состоит из двух модулей: Authentication и App.

Диаграмма классов:



Модуль App состоит из классов-моделей из базы данных: Order, Record, Cart

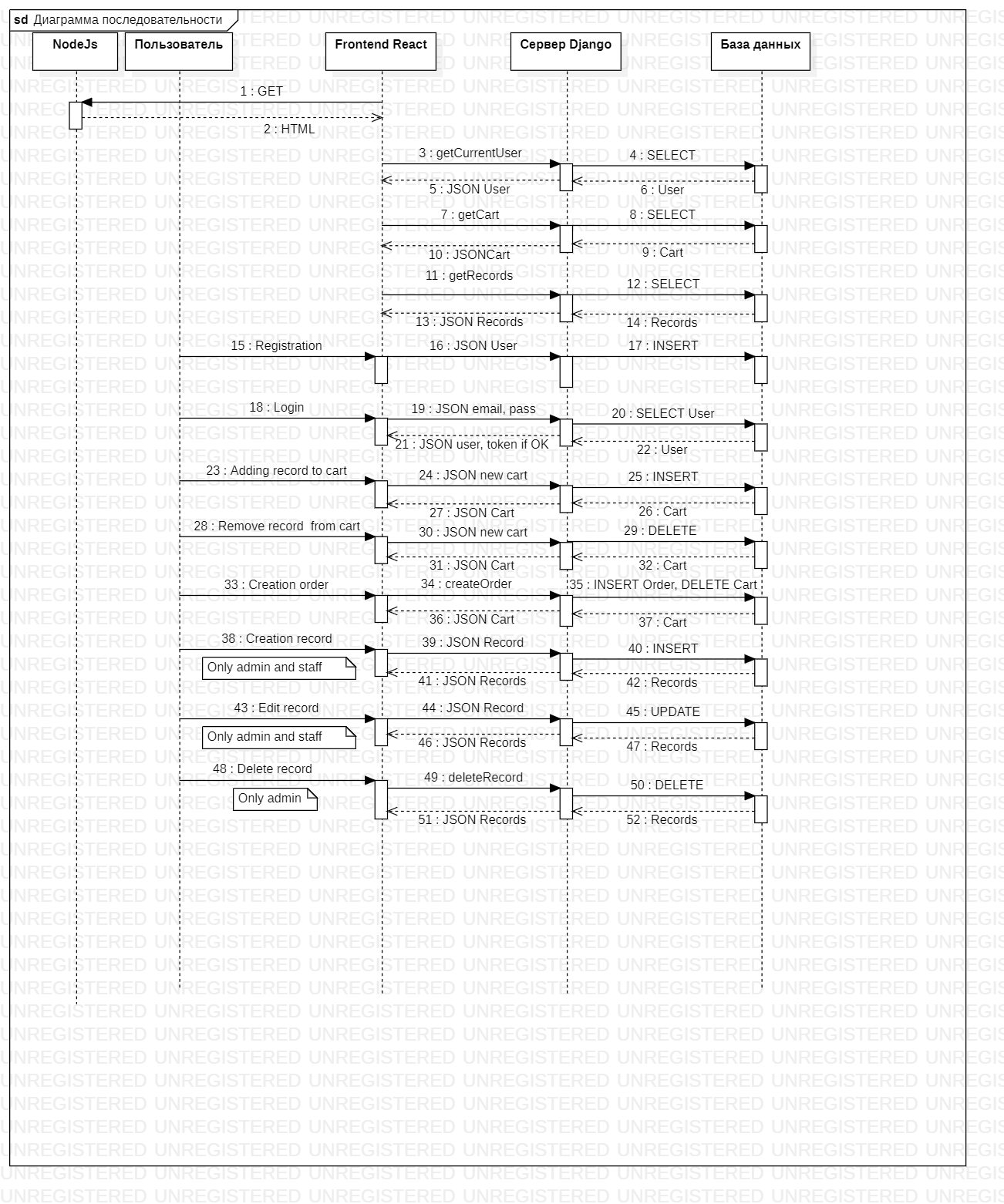
У каждого из этих классов есть Serializer – класс, который сериализует и десериализует объекты. Также у каждого из этих классов есть свой ViewSet класс, который предоставляет API для запросов.

Модуль Authentication – модуль аутентификации пользователя.  
Основным классом является User. Класс UserManager необходим для работы с Django ORM. UserSerializer – класс сериализатор обьектов User.   
Также есть два других класса сериализатора обьектов User – RegistrationSerializer и LoginSerializer. Они нужны для регистрации и авторизации пользователей.  
Классы типа APIView предоставляют API для запросов.

Класс JWTAuthentication – класс аутентификации пользователя, используется в LoginSerializer для аутентификации пользователя по данным из запроса.

**4. Алгоритмы**

Диаграмма последовательности:



При запуске веб-приложения react запрашивает у сервера NodeJs HTML страницу, получая ее в ответ. Далее веб-приложение запрашивает у удаленного сервера с django на борту данные о текущем пользователе, его корзине и пластинках, чтобы отобразить эти данные независимо от того, авторизован пользователь или нет, сервер делает SQL-запросы к базе данных, получает данные и отправляет их веб-приложению в формате JSON. Пользователь может войти или зарегистрироваться, для этого ему нужно ввести данные в форму, затем эти данные в формате JSON отправляются на удаленный сервер. Далее все действия сводятся к одному принципу, который можно описать следующим образом: действие пользователя -> отправка запроса и данных в формате JSON веб-приложением на удаленный сервер, сервер делает SQL запросы к БД и возвращает данные все так же в формате JSON.

**5. Описание интерфейса**

Окна и их предназначение:

1) Таблица всех товаров – это окно нужно для показа всех пластинок интернет-приложения, которые можно добавить в корзину.

2) Информация о пользователе – предназначено для просмотра и редактирования информации о пользователе.

3) Корзина – нужна для просмотра пластинок, которые добавлены в корзину, а также для удаления их из корзины.

4) Добавление нового товара – это окно есть только у менеджеров и администраторов. Окно предназначено для заполнения информации о новом товаре и его публикации.

5) Подробная информации о товаре – это окно предназначено для просмотра подробной информации о пластинке и добавления товара в корзину. Для менеджеров и администраторов в этом окне есть возможность отредактировать информацию о товаре.

6) Авторизация и регистрация – это окно предназначено для авторизации и регистрации пользователей.