**Министерство образования и науки Российской Федерации**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**УНИВЕРСИТЕТ ИТМО**

Факультет инфокоммуникационных технологий Образовательная программа 09.03.02

Направление подготовки (специальность) Мобильные сетевые технологии

О Т Ч Е Т

по лабораторной работе 1

Тема задания: свой boilerplate на express + sequelize / TypeORM + typescript.

Обучающийся Сулейманов Руслан Имранович, К33402

Преподаватель: Добряков Д. И., преподаватель

Санкт-Петербург

Для своего бойлерплейта я долго думал, что должно быть в каждом сервисе, решил я что для каждого сервиса имеет смысл описать авторизацию юзера, саамого юзера и его роли. Поэтому мой бойлерплейт состоит из юзеров и их взаимодействий:

Так как в домашней работе мне не понравилась работа с sequalize, я решил пистаь на typeOrm:

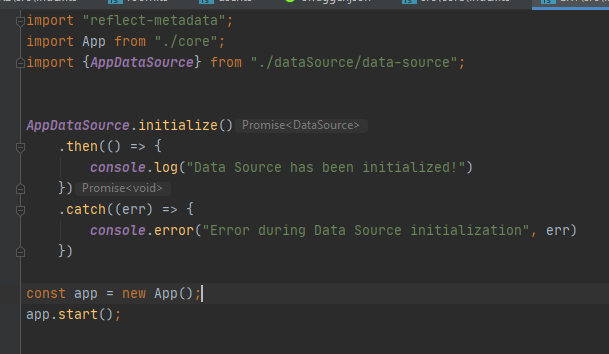
Ядро нашего сервера:



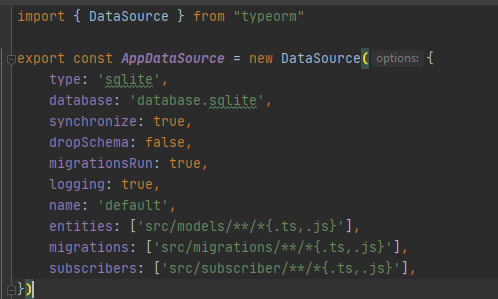
Тут подключены роуты, корсы, парсер, helmet для защиты.

Тут же мы создаем приложение, сервер и запускаем сервер.

Index:



Тут мы инициализируем нашу бд и запускаем приложение:



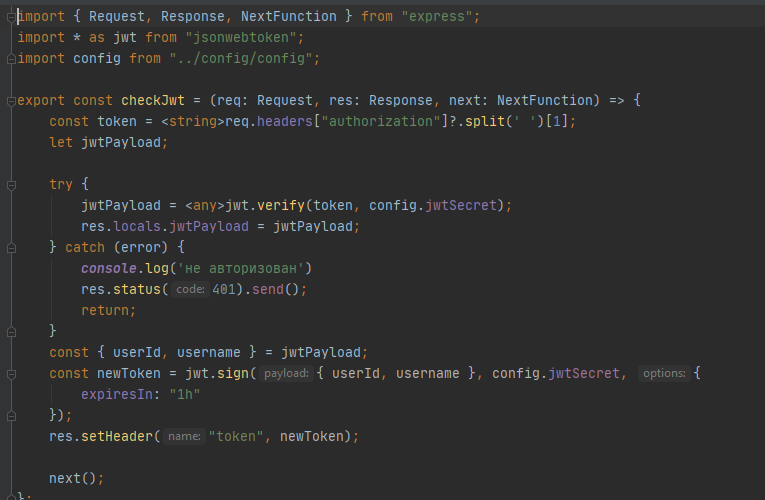
Бд работает через связь с typeorm, использована неустаревшая версия подключения datasource.

Теперь единственная модель:

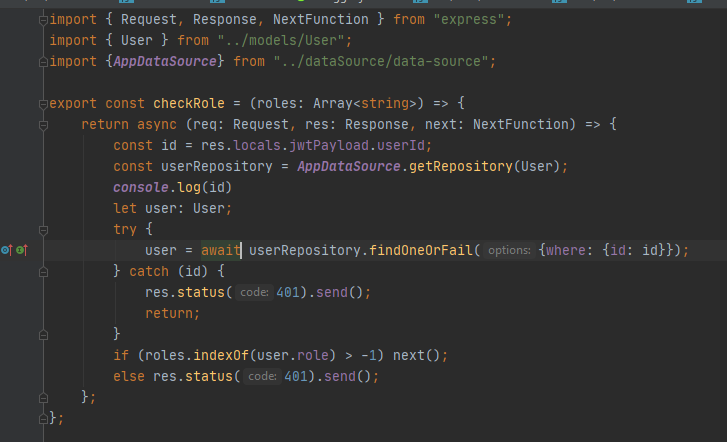


Она состоит из полей пользователя а также доп функций написанных для формирования авторизации.

Написан код для проверки jwt

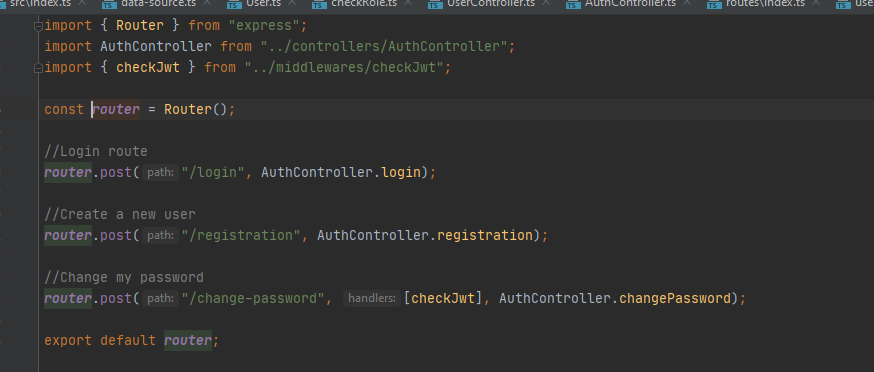


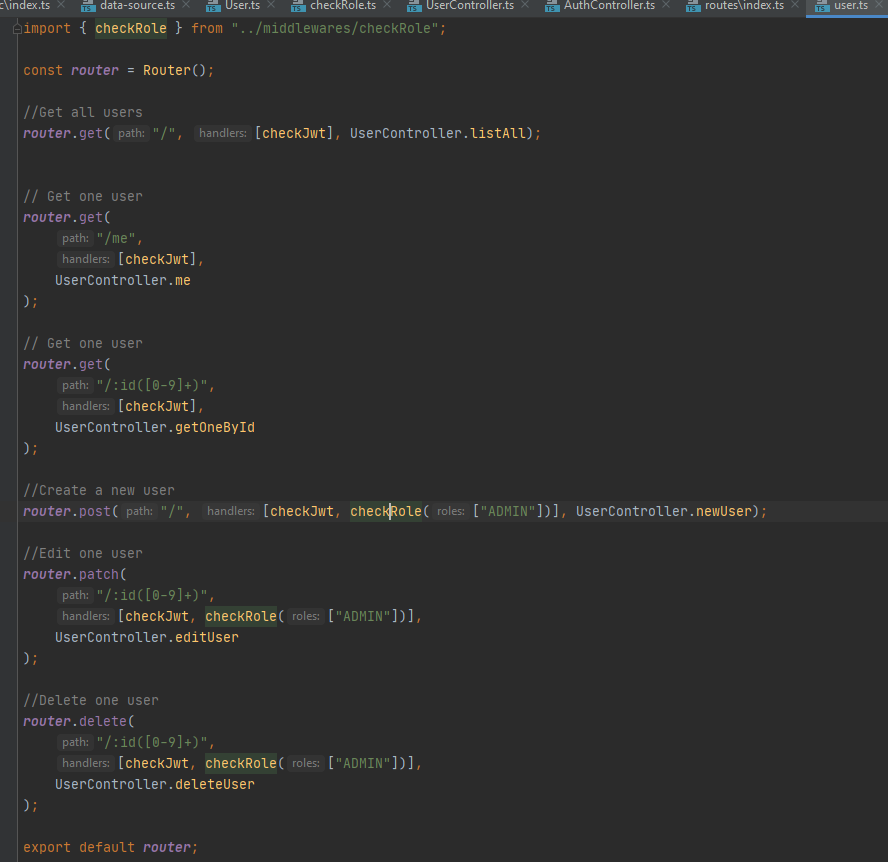
Написана функция для проверки ролей



Теперь базовые пути:

Есть пути юзера и пути авторизации:





Как мы можем заметить мы уже используем наши мидлварсы, для доступа к разным запросам.

Ну и сами контроллеры, где прописана логика поведения сервера при конкретном пути:

Например авторизация:



Тут мы читаем тело запроса, проверяем что оно не пустое, получаем с помощью typeorm таблицу юзера, проверяем что юзер есть и проверяем пароль, после чего сохраняем токен.

Вывод: по окончанию работы был бойлерплейт, с готовой авторизацией и моделью юзера. Что очень помогло в следующих работах.