

Лабораторная работа 9.

1. Синхронизация проекта в github
2. Определение модели в Sequelize

1. Синхронизация проекна в github

- 1.1 Убедитесь, что на компьютере (ноутбуке) установлен git.
В консоли выполните команду

```
git -v
```

Результат: текущая версия git

```
okorolyov@MacBook-Pro-Oleg-2 trade-project % git -v  
git version 2.39.2 (Apple Git-143)
```

Если git не установлен, то установить его.

- 1.2 Убедитесь, что у вас есть аккаунт в github. Если его нет, то необходимо зарегистрировать.

1.3 Создание проекта на github

Зайти в свой аккаунт github и создайте проект trade-project-db. (имя проекта для вариантов самостоятельной работы будет отличаться и совпадать с именем проекта)

1 General

Owner *
OlegKorolyov

Repository name *
trade-project-db
trade-project-db is available.

Great repository names are short and memorable. How about [crispy-octo-robot](#)?

Description
0 / 350 characters

2 Configuration

Choose visibility *
Choose who can see and commit to this repository
Public

Add README
READMEs can be used as longer descriptions. [About READMEs](#)
On

Add .gitignore
.gitignore tells git which files not to track. [About ignoring files](#)
No .gitignore

Add license
Licenses explain how others can use your code. [About licenses](#)
No license

Create repository

1.4 Инициализация проекта

В консоли перейти в папку проекта trade-project и выполните команду

```
git init
```

1.5 Добавить файлы проекта в комит

Выполнить команду в консоли

```
git add .
```

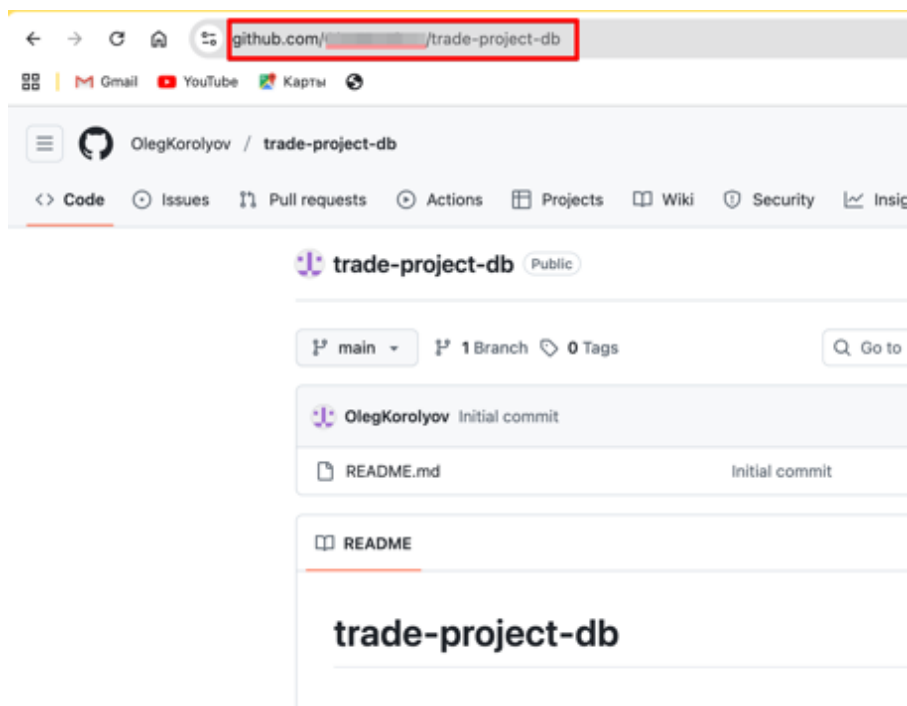
1.6 Выполнить первый комит

В консоли выполнить команду

```
git commit -am "Initial commit"
```

1.7 Синхронизировать данные в удаленный репозиторий github

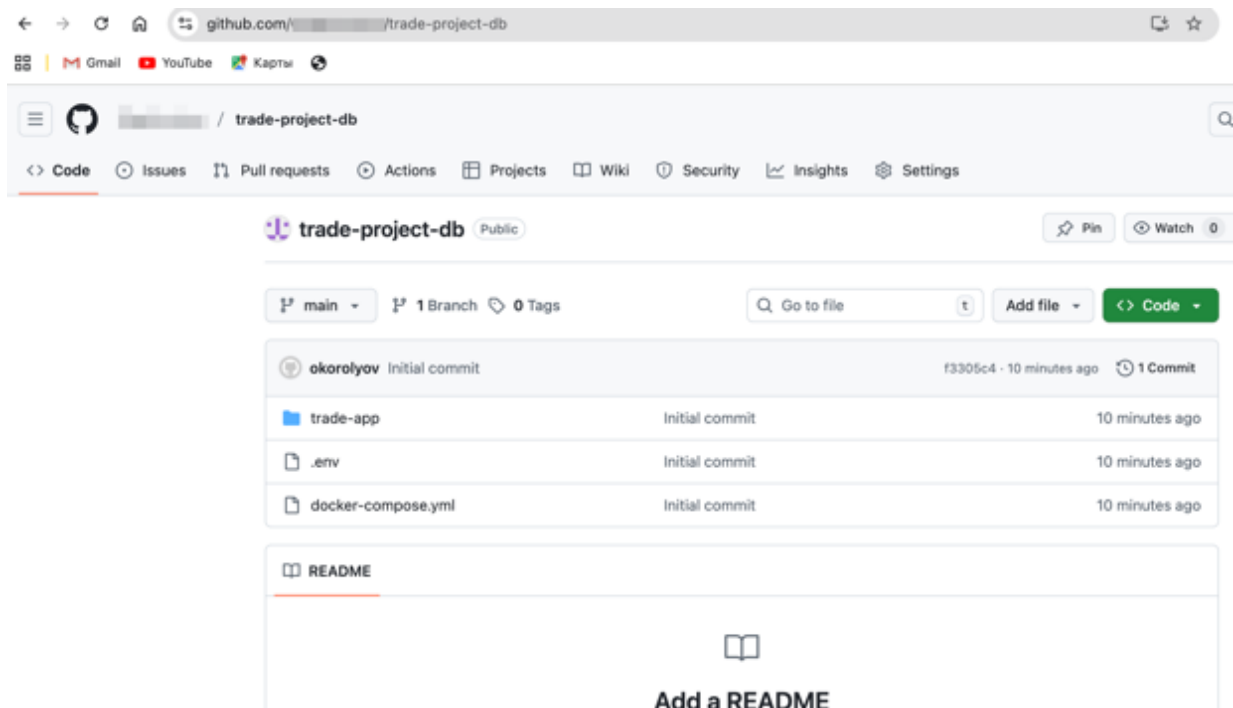
```
git remote add origin <Link to GitHub Repo>
```



```
git remote -v
```

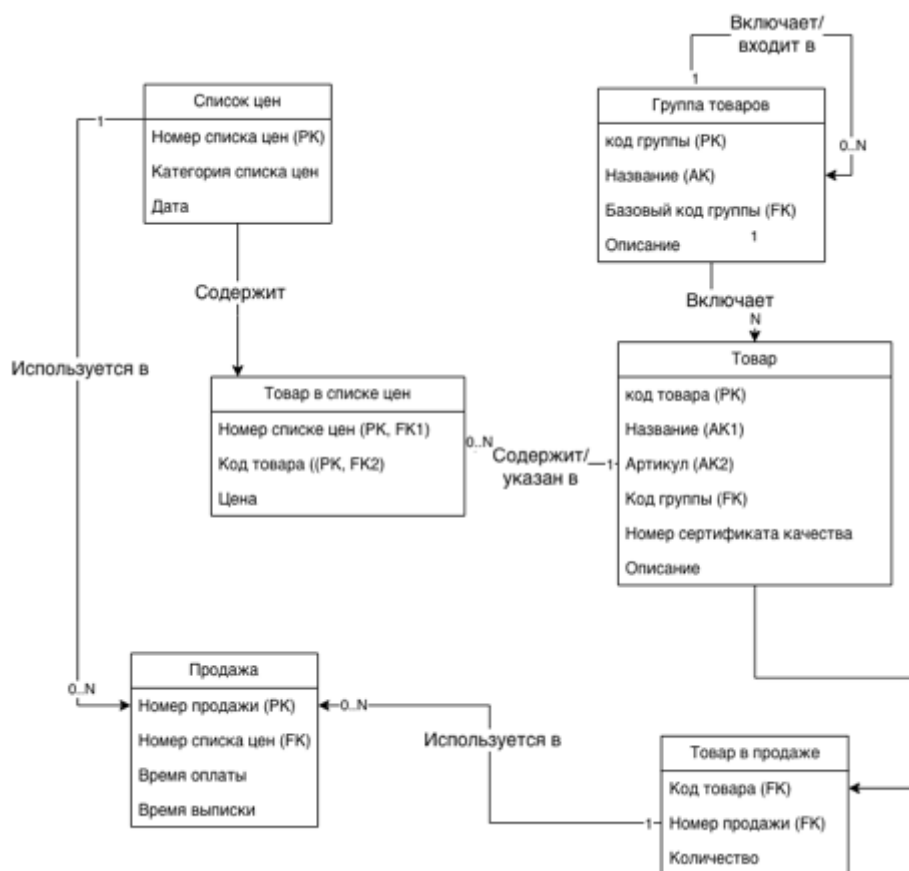
```
git push -u origin main --force
```

Результат. Файлы проекта должны быть синхронизированы с github проектом

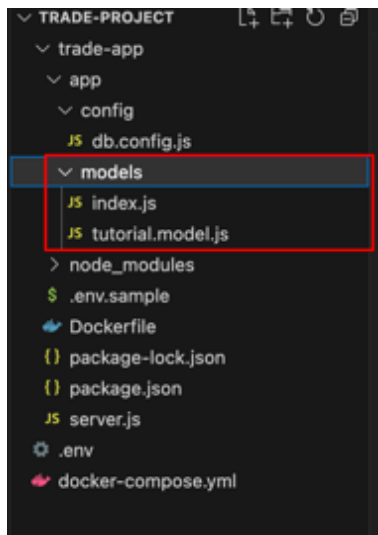


2. Определение модели в Sequelize

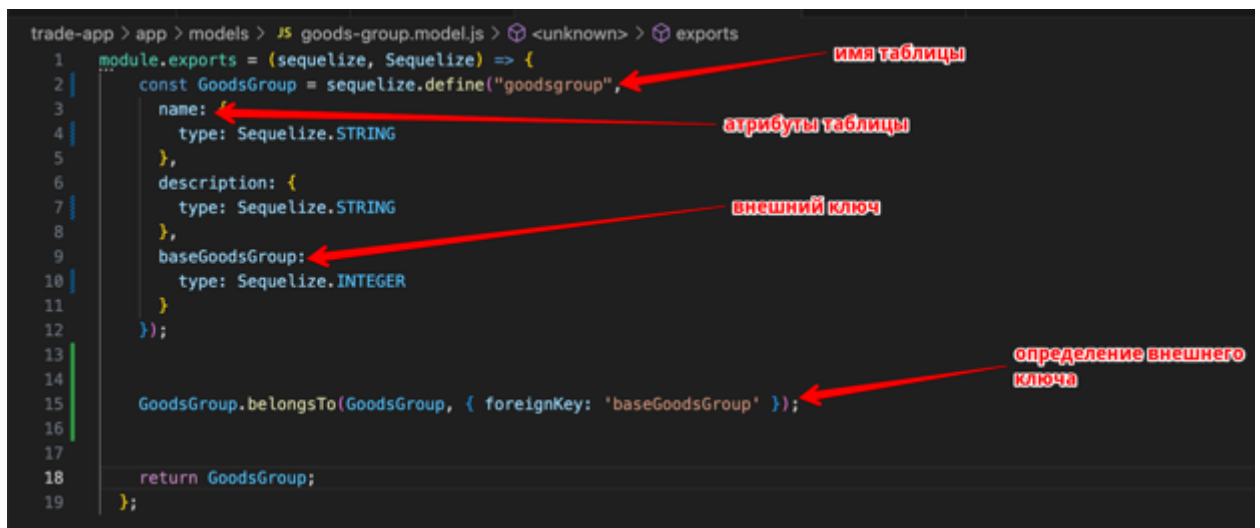
2.1 Возьмем концептуальную модель БД с атрибутами, полученную в лабораторной работе 7.



2.2 В проекте в VSCode добавим файлы модели.



Необходимо создать файл goods-group.model.js



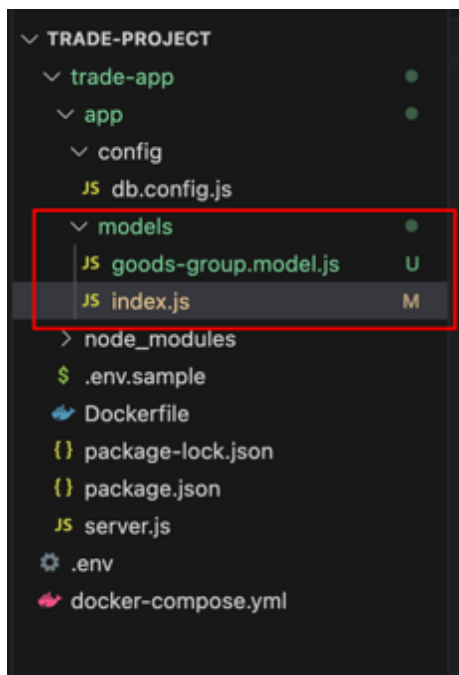
NOTE!! В Sequelize при определении объектов таблиц фреймворк добавляет поля
id = Sequelize.INTEGER,
createAt = Sequelize.DATE,
updatedAt = Sequelize.DATE

Данные о новой модели необходимо зарегистрировать в trade-app/app/models/index.js

```
trade-app > app > models > JS index.js > ...
9
10 pool: {
11   max: dbConfig.pool.max,
12   min: dbConfig.pool.min,
13   acquire: dbConfig.pool.acquire,
14   idle: dbConfig.pool.idle
15 }
16 };
17
18 const db = {};
19
20 db.Sequelize = Sequelize;
21 db.sequelize = sequelize;
22
23 db.tutorials = require("../tutorial.model.js")(sequelize, Sequelize);
24 db.goodsGroup = require("../goods-group.model.js")(sequelize, Sequelize);
25
26 module.exports = db;
27
```

2.3 Подчистим ненужные файлы.

Удалим файл с моделью tutorial и его регистрацию в фреймворке.



2.4 Синхронизируем изменения с репозиторием

В консоли выполнить команды

```
git add .
```

```
git commit -am "add goods-group model + cleanup"
```

```
git push -u origin main
```

2.5 Запускаем приложение

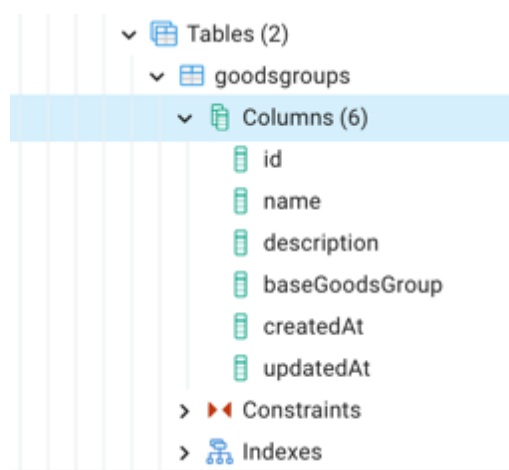
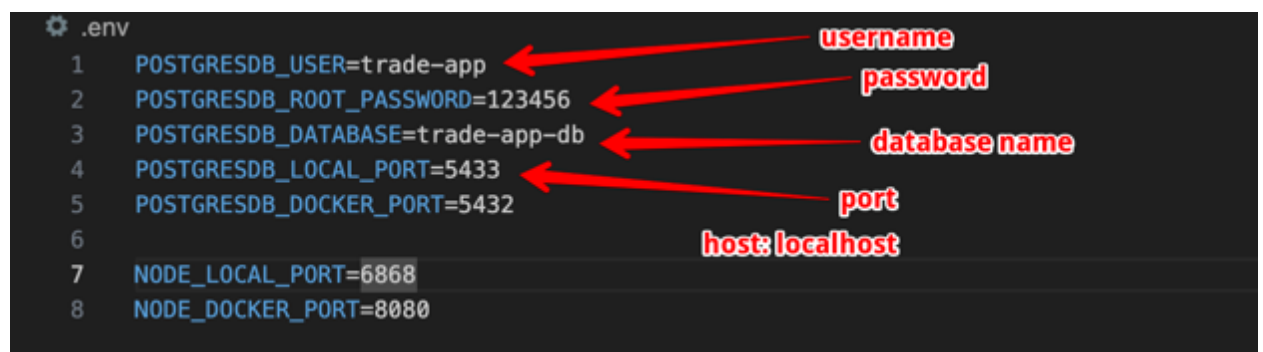
Сохраняем все изменения.

В консоли выполняем команду

```
docker-compose up --build
```

Проверяем БД.

С помощью клиента БД (PgAdmin, например) подключаемся к БД в докере. Используем параметры файла .env для конфигурации соединения с сервером БД.



2.6 Доработать остальные таблицы БД