

Лабораторная работа 9.

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

1. Синхронизация проекта 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 * Public

Add README On

Add .gitignore No .gitignore

Add license 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 .
```

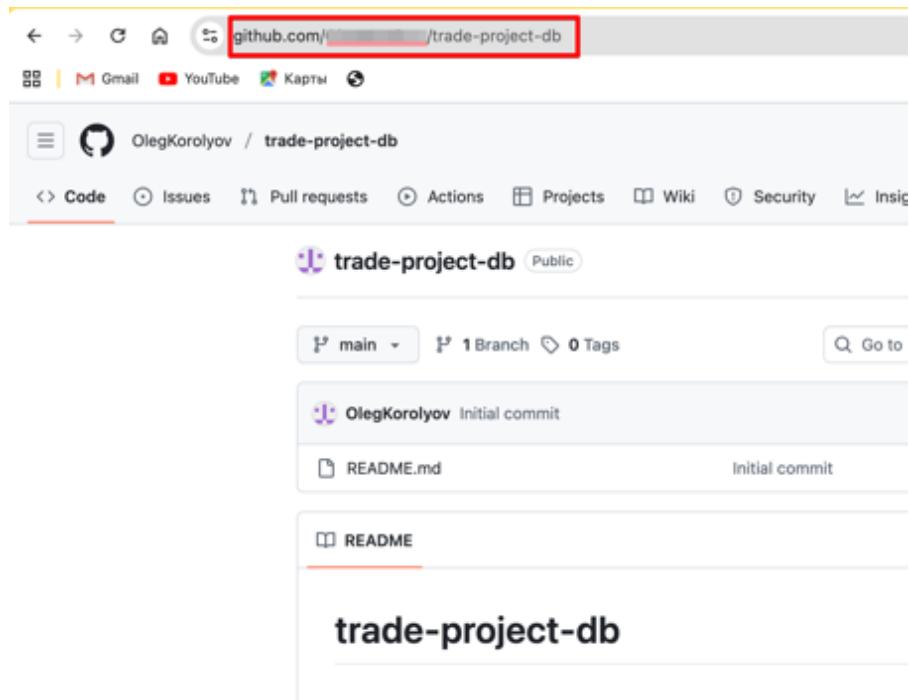
1.6 Выполнить первый комит

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

github.com/.../trade-project-db

Gmail YouTube Карты

trade-project-db

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

trade-project-db Public

main 1 Branch 0 Tags Go to file Add file Code

okorolyov Initial commit f3305c4 · 10 minutes ago 1 Commit

trade-app Initial commit 10 minutes ago

.env Initial commit 10 minutes ago

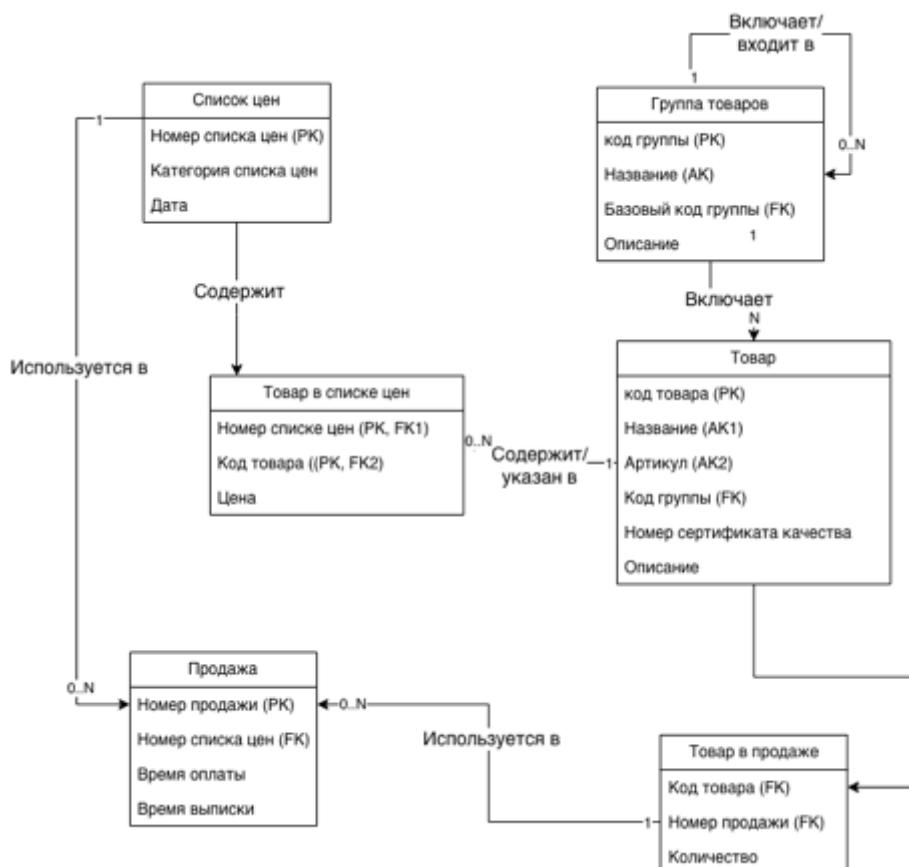
docker-compose.yml Initial commit 10 minutes ago

README

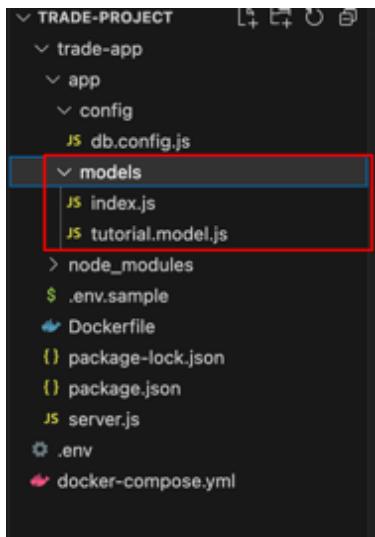
Add a README

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

```

trade-app > app > models > JS goods-group.model.js > ? <unknown> > ? exports
1  module.exports = (sequelize, Sequelize) => {
2    const GoodsGroup = sequelize.define("goodsgroup", имя таблицы
3      name: атрибуты таблицы ←
4      {
5        type: Sequelize.STRING
6      },
7      description: {
8        type: Sequelize.STRING
9      },
10     baseGoodsGroup: внешний ключ ←
11     {
12       type: Sequelize.INTEGER
13     }
14   );
15
16   GoodsGroup.belongsTo(GoodsGroup, { foreignKey: 'baseGoodsGroup' }); определение внешнего
17   ключа ←
18
19   return GoodsGroup;

```

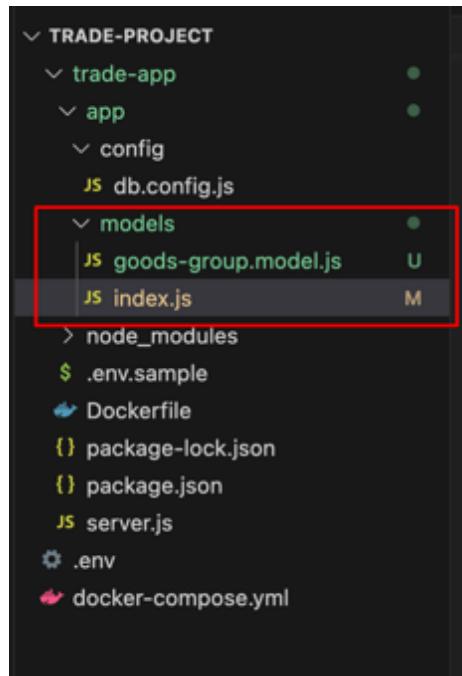
NOTE!! В Sequelize при определении объектов таблиц фреймворк добавляет поля
 id = Sequelize.INTEGER,
 createdAt = Sequelize.DATE,
 updatedAt = Sequelize.DATE

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.

```
⚙ .env
1 POSTGRESDB_USER=trade-app ← username
2 POSTGRESDB_ROOT_PASSWORD=123456 ← password
3 POSTGRESDB_DATABASE=trade-app-db ← database name
4 POSTGRESDB_LOCAL_PORT=5433 ← port
5 POSTGRESDB_DOCKER_PORT=5432 ← host: localhost
6
7 NODE_LOCAL_PORT=6868
8 NODE_DOCKER_PORT=8080
```

Tables (2)
goodsgroups
Columns (6)
id
name
description
baseGoodsGroup
createdAt
updatedAt
Constraints
Indexes

2.6 Доработать остальные таблицы БД