

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ САНКТ-ПЕТЕРБУРГСКИЙ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ
Факультет информационных технологий и программирования

Лабораторная работа №3

«Создание доменной модели»

по дисциплине «Web программирование»

Выполнил: студент группы М33122 Белоногов Е.В.

Санкт-Петербург
2022

Подключил heroku postgres с помощью команды:

```
heroku addons:create heroku-postgresql:hobby-dev
```

```
my-backend-project — -zsh — 116x23
evgeny@MacBook-Pro-Evgeny my-backend-project % heroku addons
No add-ons for app evgeny-backend-itmo.
evgeny@MacBook-Pro-Evgeny my-backend-project % heroku addons:create heroku-postgresql:hobby-dev
Creating heroku-postgresql:hobby-dev on • evgeny-backend-itmo... free
Database has been created and is available
! This database is empty. If upgrading, you can transfer
! data from another database with pg:copy
Created postgresql-curved-59014 as DATABASE_URL
Use heroku addons:docs heroku-postgresql to view documentation
evgeny@MacBook-Pro-Evgeny my-backend-project % heroku addons

Add-on          Plan      Price  State
-----
heroku-postgresql (postgresql-curved-59014)  hobby-dev  free   created
└─ as DATABASE

The table above shows add-ons and the attachments to the current app (evgeny-backend-itmo) or other apps.

evgeny@MacBook-Pro-Evgeny my-backend-project % heroku config
=== evgeny-backend-itmo Config Vars
DATABASE_URL: postgres://olvc...:24007@ec2-52-212-
228-71.eu-west-
evgeny@MacBook-Pro-Evgeny my-backend-project %
```

Для работы с БД Postgres установил модуль pg:

```
npm install pg --save
```

В качестве ORM был выбран TypeOrm:

```
npm install --save @nestjs/typeorm
```

Для парсинга connection string воспользовался модулем pg-connection-string:

```
@Injectable()
export class TypeOrmConfigService implements TypeOrmOptionsFactory {
  createTypeOrmOptions(): TypeOrmModuleOptions {
    var config = require('pg-connection-string').parse(process.env.DATABASE_URL);
    return {
      type: 'postgres',
      host: config.host,
      port: config.port,
      username: config.user,
      password: config.password,
      database: config.database,
      // entities: ['dist/**/*.entity{ .ts,.js}'],
      // entities: [User, Comment, Password],
      autoLoadEntities: true,
      synchronize: true,
      ssl: {
        rejectUnauthorized: false
      }
    };
  }
}
```

Подключение к БД:

```
12 @Module({
13   imports: [
14     ConfigModule.forRoot(),
15     TypeOrmModule.forRootAsync({
16       useClass: TypeOrmConfigService
17     }), UsersModule, CommentsModule, PasswordsModule],
18   controllers: [AppController],
19   providers: [AppService]
20 })
21 export class AppModule {
22   constructor(private connection: Connection) {}
23 }
```

Представление сущностей:

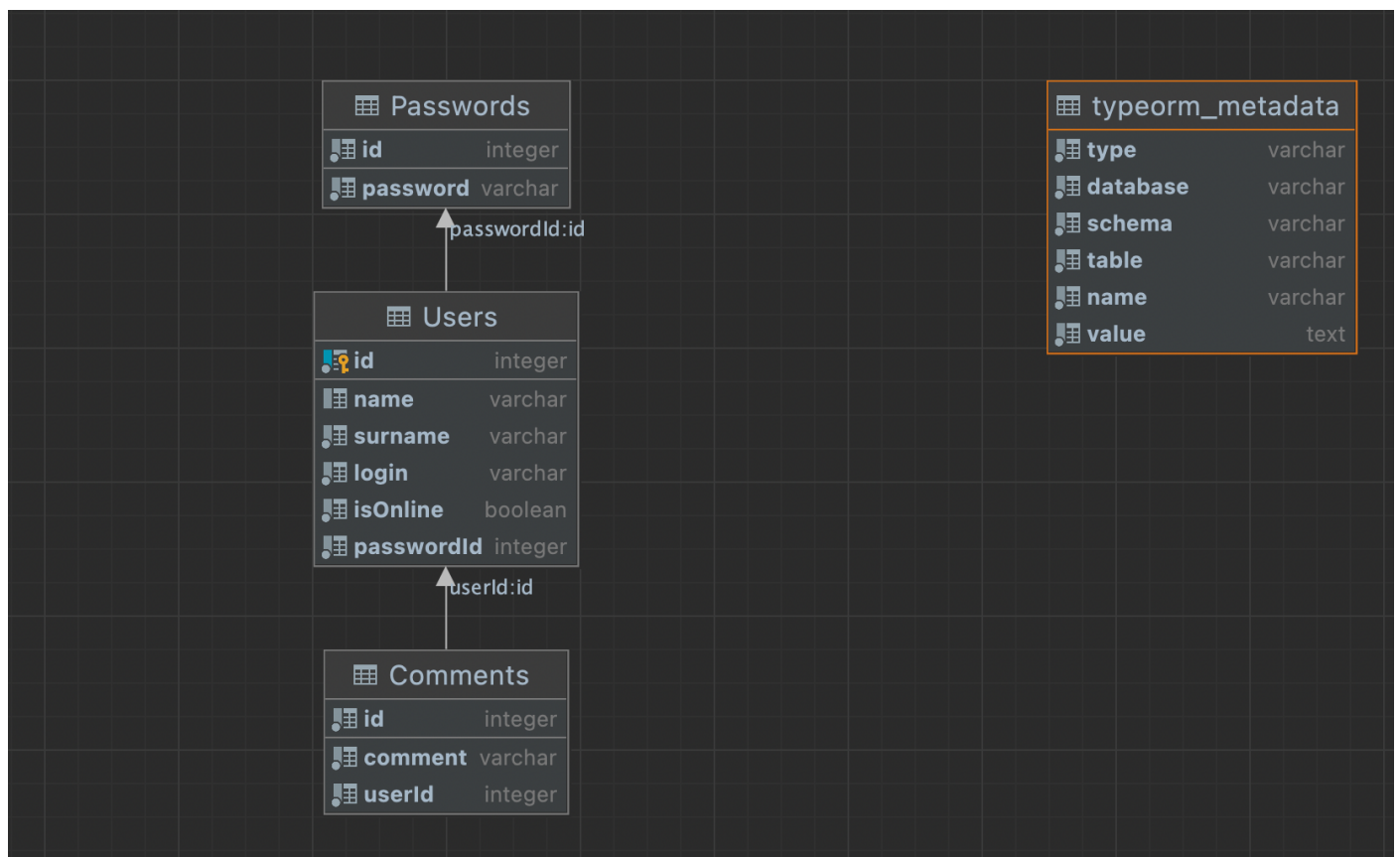
src	
models	
comments	
TS comment.entity.ts	U
TS comments.controller.ts	U
TS comments.module.ts	U
TS comments.service.ts	U
passwords	
TS password.entity.ts	U
TS passwords.controller.ts	U
TS passwords.module.ts	U
TS passwords.service.ts	U
users	
> dto	
TS user.entity.ts	U
TS users.controller.ts	U
TS users.module.ts	U
TS users.service.ts	U

```

TS app.module.ts M    TS user.entity.ts U ×    TS main.ts M
src > models > users > TS user.entity.ts > User > isOnline
1  import { Entity, Column, PrimaryGeneratedColumn, OneToOne, JoinColumn, OneToMany } from 'typeorm';
2  import { Password } from '../passwords/password.entity';
3  import { Comment } from '../comments/comment.entity'
4
5
6  @Entity('Users')
7  export class User {
8      @PrimaryGeneratedColumn()
9      id: number;
10
11      @Column()
12      name: string;
13
14      @Column()
15      surname: string;
16
17      @Column()
18      login: string;
19
20      @Column()
21      isOnline: boolean;
22
23      @OneToOne(() => Password)
24      @JoinColumn()
25      password: Password;
26
27      @OneToMany(() => Comment, comments => comments.user)
28      comments: Comment[];
29  }

```

Визуальное представление:



Описание сущностей:

Users

Пользователи

- id (primary key, int) -- id пользователя
- name (string) -- имя пользователя
- surname (string) -- фамилия пользователя
- login (string) -- никнейм для регистрации
- isOnline (boolean) -- статус онлайн/нет
- passwordId (int) -- id пароля в таблице passwords

Passwords

Пароли

Отношение 1:1 пароль:пользователь

- id (primary key, int) -- id пароля
- password (string) -- пароль (хешированный ?)

Comments

Комментарии

Отношение *:1 комментарии:пользователь

- id (primary key, int) -- id пароля
- comment (string) -- содержание комментария
- userId -- id пользователя-автора