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

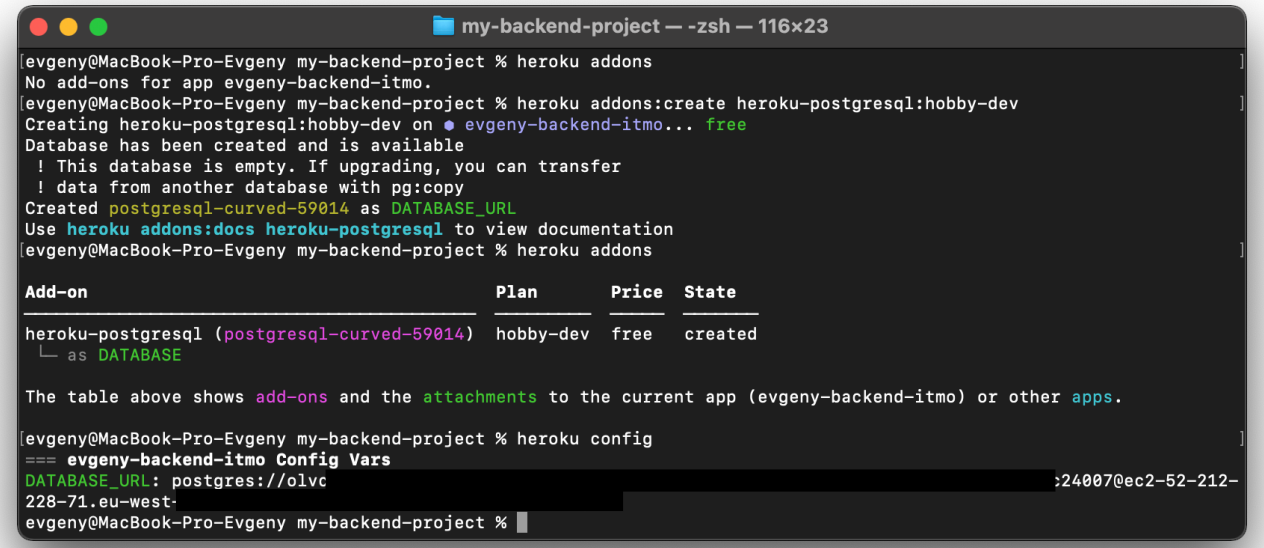
«Создание доменной модели»
по дисциплине «Web программирование»

Выполнил: студент группы M33122 Белоногов Е.В.

Санкт-Петербург
2022

Подключил heroku postgres с помощью команды:

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



Для работы с БД Postgres установил модуль pg:

`npm install pg --save`

В качестве ORM был выбран TypeOrm:

`npm install --save @nestjs/typeorm`

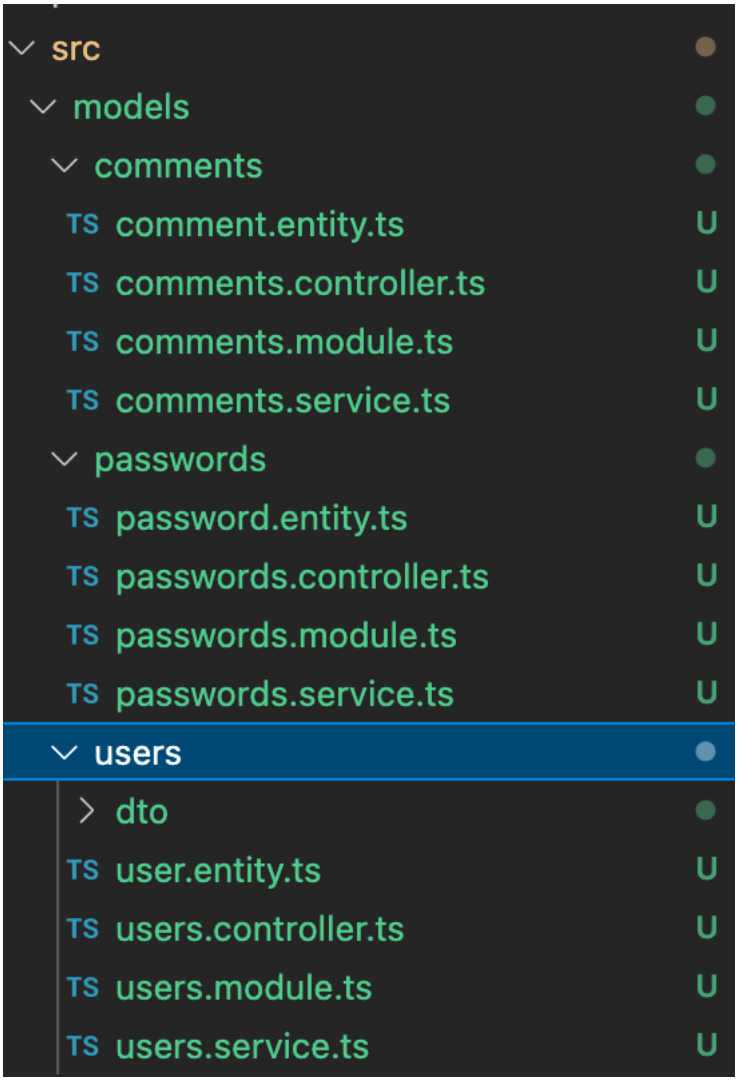
Для парсинга connection string воспользовался модулем `pg-connection-string`:

```
@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:

```
@Module({
  imports: [
    ConfigModule.forRoot(),
    TypeOrmModule.forRootAsync({
      useClass: TypeOrmConfigService
    }), UsersModule, CommentsModule, PasswordsModule],
  controllers: [AppController],
  providers: [AppService]
})
export class AppModule {
  constructor(private connection: Connection) {}
}
```

Представление сущностей:

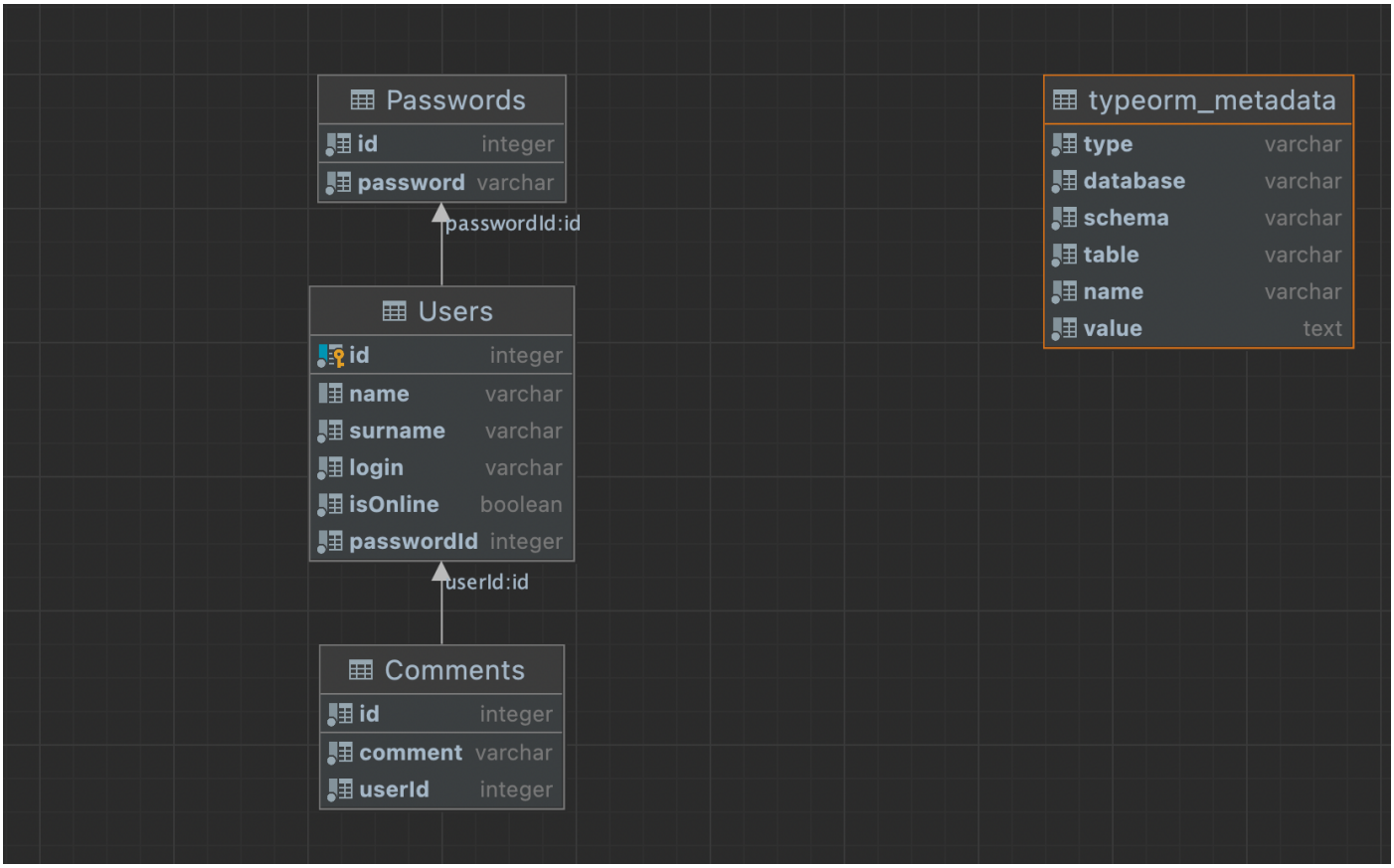


app.module.ts M | user.entity.ts U | main.ts M

src > models > users > user.entity.ts > User > isOnline

```typescript
import { Entity, Column, PrimaryGeneratedColumn, OneToOne, JoinColumn, OneToMany } from 'typeorm';
import { Password } from '../passwords/password.entity';
import { Comment } from '../comments/comment.entity'


@Entity('Users')
export class User {
  @PrimaryGeneratedColumn()
  id: number;

  @Column()
  name: string;

  @Column()
  surname: string;

  @Column()
  login: string;

  @Column()
  isOnline: boolean;

  @OneToOne(() => Password)
  @JoinColumn()
  password: Password;

  @OneToMany(() => Comment, comments => comments.user)
  comments: Comment[];
}
```

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