САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа №1

Выполнил:

Дорофеева Арина

Группа к33401

Проверил:
Добряков Д. И.

Санкт-Петербург

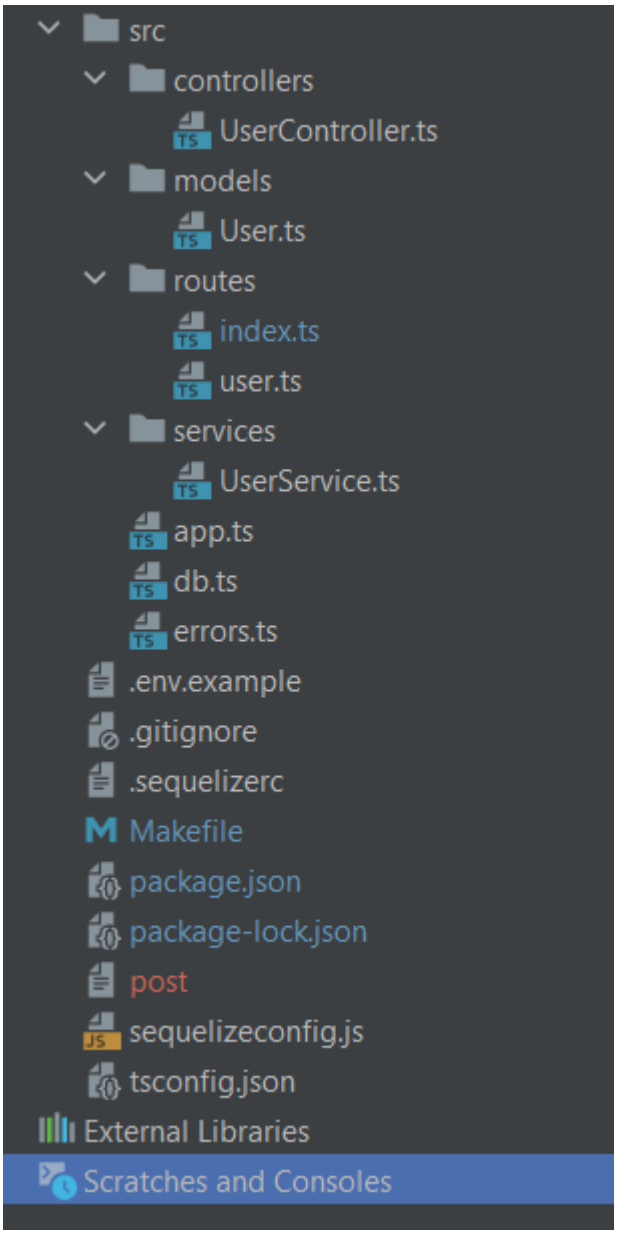
2022 г.

Задача

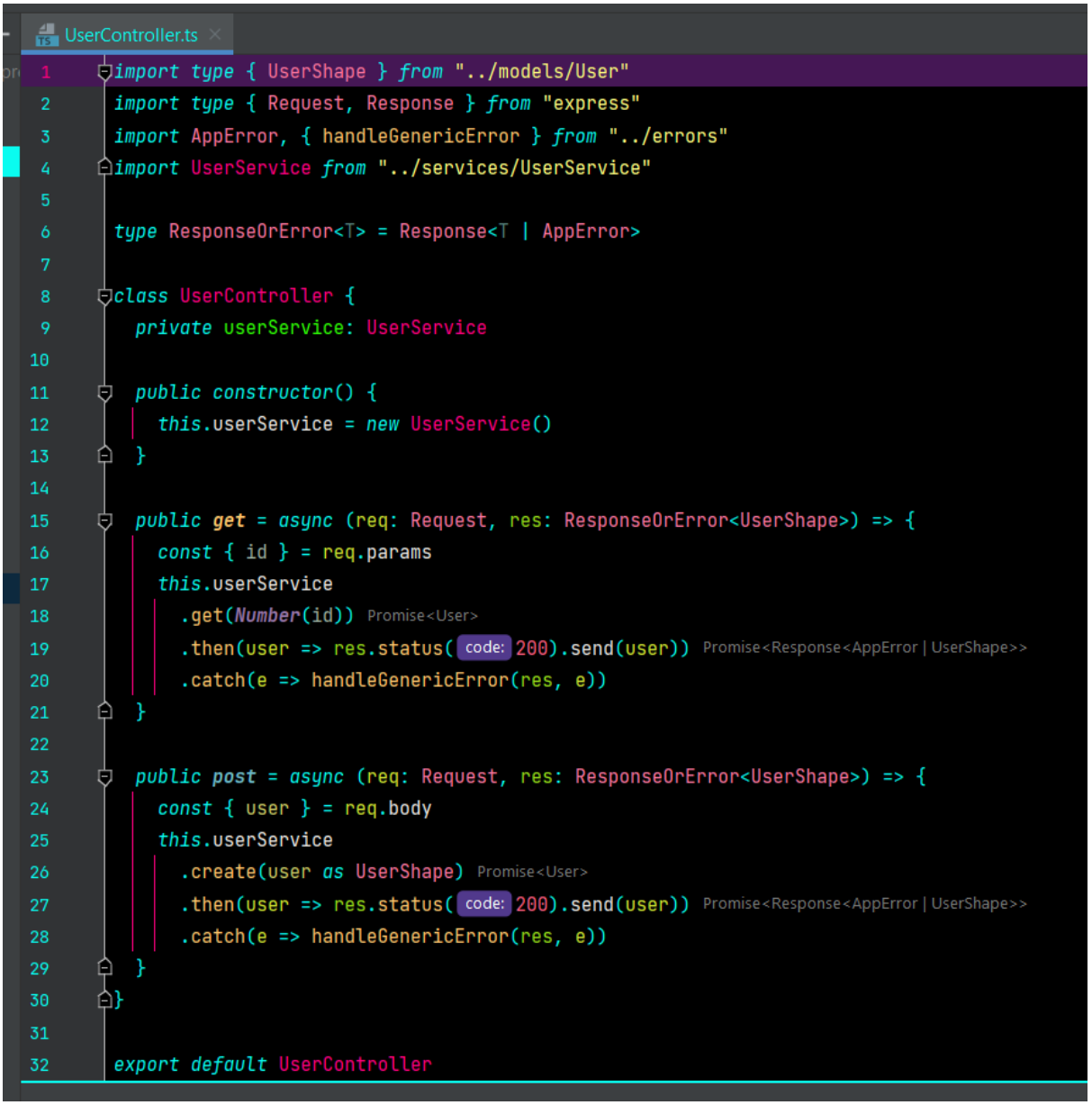
Написать свой boilerplate на express + sequelize + typescript.
Должно быть явное разделение на:

1. модели
2. контроллеры
3. роуты
4. сервисы для работы с моделями (реализуем паттерн “репозиторий”)

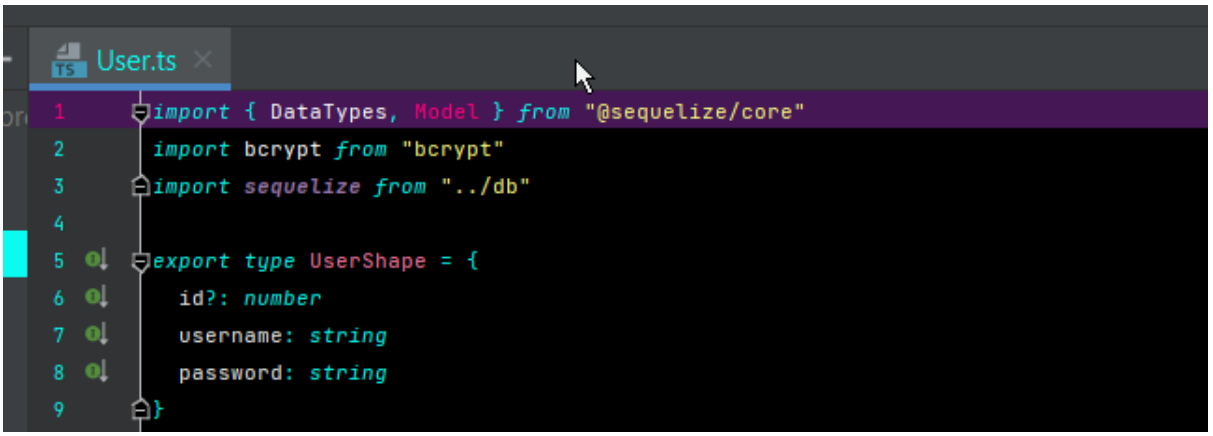
Ход работы

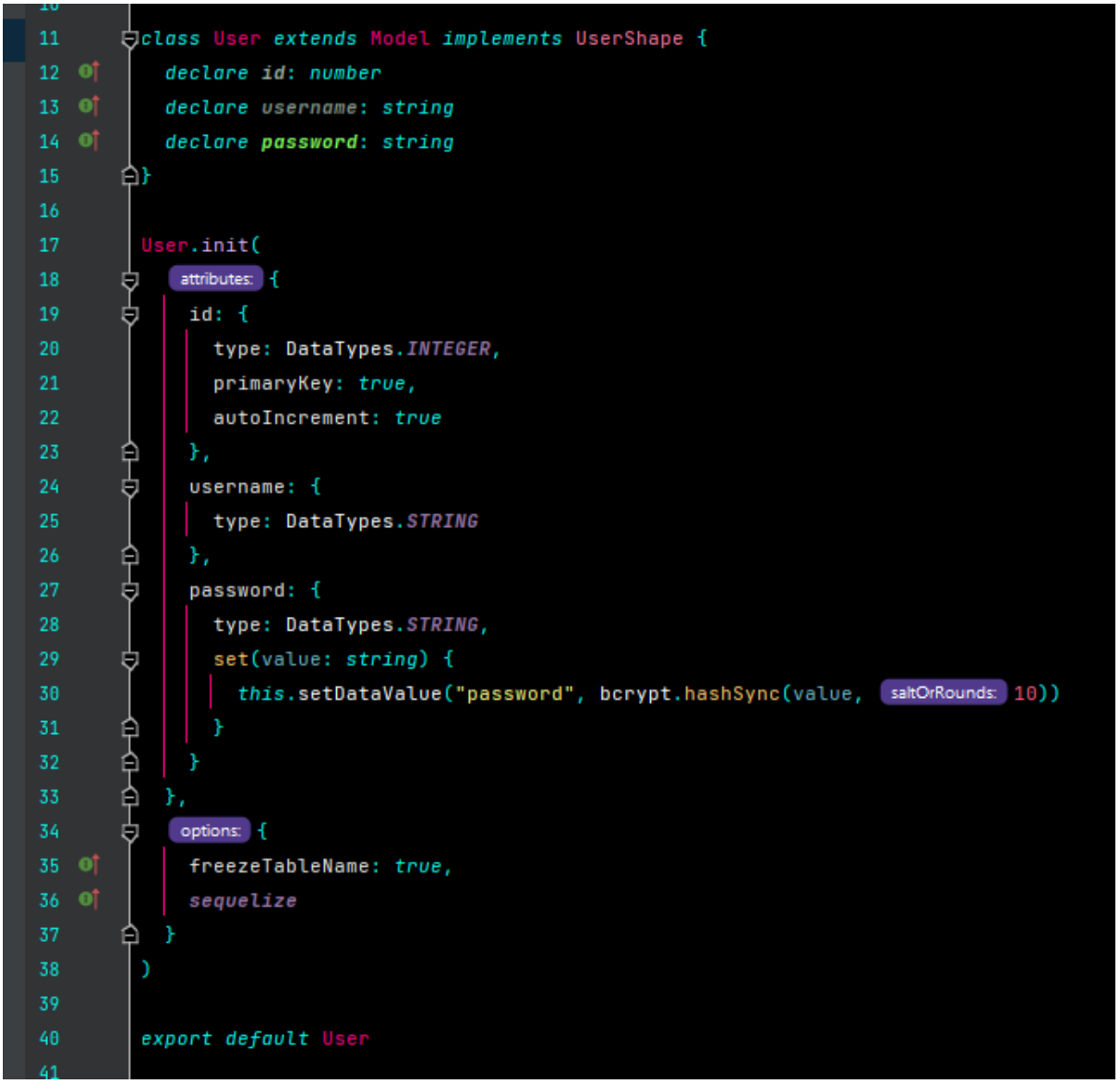


Контроллеры:



Модели:



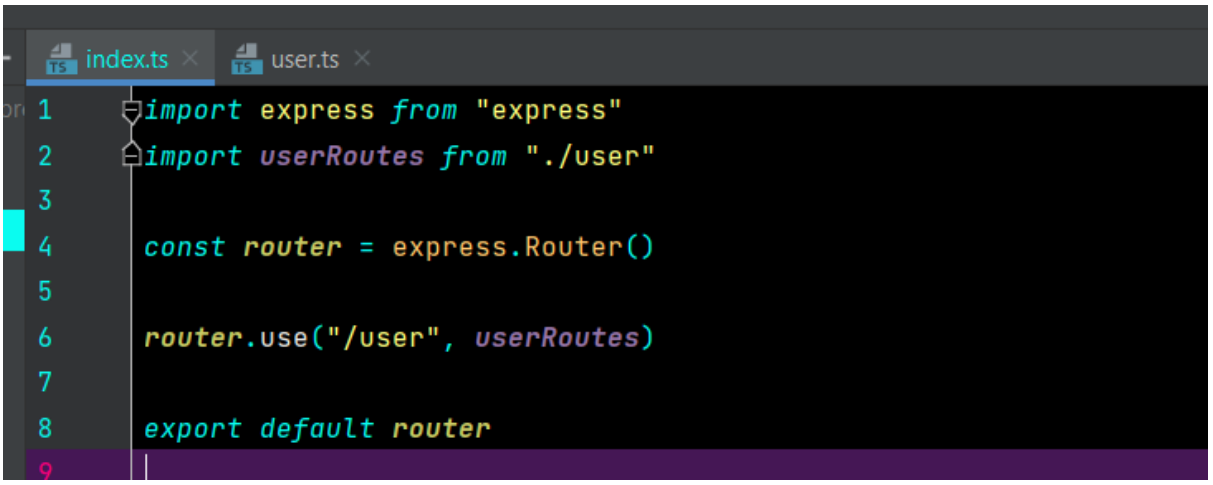


```
class User extends Model implements UserShape {
  declare id: number
  declare username: string
  declare password: string
}

User.init(
  attributes: {
    id: {
      type: DataTypes.INTEGER,
      primaryKey: true,
      autoIncrement: true
    },
    username: {
      type: DataTypes.STRING
    },
    password: {
      type: DataTypes.STRING,
      set(value: string) {
        this.setDataValue("password", bcrypt.hashSync(value, saltOrRounds: 10))
      }
    }
  },
  options: {
    freezeTableName: true,
    sequelize
  }
)

export default User
```

Роуты:



index.ts | user.ts

```
import express from "express"
import userRoutes from "./user"

const router = express.Router()

router.use("/user", userRoutes)

export default router
```

index.ts user.ts

```ts
import express from "express"
import UserController from "../controllers/UserController"

const router = express.Router()

const controller = new UserController()

router.route( prefix: "/:id")
  .get(controller.get)

router.route( prefix: "/")
  .post(controller.post)

export default router
```

Сервисы:

UserService.ts

```ts
import User from "../models/User"
import type { UserShape } from "../models/User"

class UserService {
  public async get(id: number): Promise<User> {
    if (id !== +id) throw new Error("User id must be an integer.")
    let user
    try {
      user = await User.findByPk(id)
    } catch (e) {
      throw new Error(`User with id=${ id } not found.`)
    }
    return user as User
  }

  public async create(userData: UserShape): Promise<User> {
    const user = await User.create(userData)
    return user
  }
}

export default UserService
```

Вывод

В ходе работы я собрала шаблон boilerplate на express + sequelize + typescript, который буду использовать для создания бэкенд-приложения.