САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Домашняя работа №2

Выполнил:
Кулёмин С. А.
Группа K33401

Проверил:
Добряков Д. И.

Санкт-Петербург
2022 г.

Задание

- Продумать свою собственную модель пользователя
- Реализовать набор из CRUD-методов для работы с пользователями средствами Express + Sequelize
- Написать запрос для получения пользователя по id/email

Модель пользователя

```
class User extends Model {
  /**
   * Helper method for defining associations.
   * This method is not a part of Sequelize lifecycle.
   * The `models/index` file will call this method automatically.
   */
  static associate(models) {
    // define association here
  }
}
User.init( attributes: {
  username: DataTypes.STRING,
  password: DataTypes.STRING,
  firstName: DataTypes.STRING,
  lastName: DataTypes.STRING,
  email: DataTypes.STRING,
  hometown: DataTypes.STRING
}, options: {
  sequelize,
  modelName: 'User',
});
```

Работа с sequelize

```
PS C:\Users\mi\WebstormProjects\ITMO-ICT-Backend-2022\homeworks\K33401\Кулёмин Семён\HW2> npx sequelize-cli
model:generate --name User --attributes username:string,password:string,firstNa
me:string,lastName:string,email:string,hometown:string

Sequelize CLI [Node: 10.23.1, CLI: 6.4.1, ORM: 6.17.0]

New model was created at C:\Users\mi\WebstormProjects\ITMO-ICT-Backend-2022\homeworks\K33401\Кулёмин Семён\H
W2\models\user.js .
New migration was created at C:\Users\mi\WebstormProjects\ITMO-ICT-Backend-2022\homeworks\K33401\Кулёмин Сем
ён\HW2\migrations\20220325190726-create-user.js .
PS C:\Users\mi\WebstormProjects\ITMO-ICT-Backend-2022\homeworks\K33401\Кулёмин Семён\HW2> npx sequelize db:m
igrate

Sequelize CLI [Node: 10.23.1, CLI: 6.4.1, ORM: 6.17.0]

Loaded configuration file "config\config.json".
Using environment "development".
== 20220325190726-create-user: migrating =======
== 20220325190726-create-user: migrated (0.024s)
```

Добавление тестовых данных

	id	username	password	firstName	lastName	email	hometown	createdAt	updatedAt
	Фи...	Фильтр	Фильтр	Фильтр	Фильтр	Фил...	Фильтр	Фильтр	Фильтр
1	1	test1	password	Test1	Test1	test...	Kukuevo	2022-03-...	2022-03-2...
2	2	test2	password	Test2	Test2	test...	Huevo	2022-03-...	2022-03-2...
3	3	test3	password	Test3	Test3	test...	Kukuevo	2022-03-...	2022-03-2...

CRUD-методы

```
// create
app.post( path: '/users', handlers: async (req : Request<P, ResBody, ReqBody, ReqQuery, Locals> , res : Response<ResBody, Locals> ) => {
    const user = await db.User.create(req.body)
    res.send(user.toJSON())
})
// update
app.get('users/:id', async (req : Request<P, ResBody, ReqBody, ReqQuery, Locals> , res : Response<ResBody, Locals> ) => {
    const user = await db.User.findByPk(req.params.id)
    if (user) {
        await db.User.update(req.body, {
            where: {
                id: req.params.id
            }
        })
    } else {
        await db.User.create(req.body)
    }
    res.send(req.body)
})
// delete
app.delete( path: 'users/:id', handlers: async (req : Request<P, ResBody, ReqBody, ReqQuery, Locals> , res : Response<ResBody, Locals> ) => {
    const user = await db.User.findByPk(req.params.id)
    if (user) {
        user.destroy();
        res.sendStatus( code: 200).send( body: {'msg': 'user deleted'})
    }
    res.status( code: 404).send( body: {'msg': 'user not found'})
})
```

Поиск по id/email

```
//search by id
app.get('/users/id/:id', async (req : Request<P, ResBody, ReqBody, ReqQuery, Locals> , res : Respon
    const user = await db.User.findByPk(req.params.id)
    if (user) {
        res.send(user.toJSON())
    }
    res.status( code: 404).send( body: {'msg': 'user not found'})
})
//search by email
app.get('/users/email/:email', async (req : Request<P, ResBody, ReqBody, ReqQuery, Locals> , res
    const user = await db.User.findOne({
        where: {
            email: req.params.email
        }
    })
    if (user) {
        res.send(user.toJSON())
    }
    res.status( code: 404).send( body: {'msg': 'user not found'})
})
```

Вывод

В ходе выполнения работы я ознакомился с основами express и sequelize. Кроме того, был создан скрипт для управления базой данных (создания, удаления, обновления записей).