САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Backend-Разработка

Отчет

Лабораторная работа 2

Выполнил:
Бункута Натан

Группа:
К33412

Проверил:
Добрияков Д.И

Санкт-Петербург

2023 г.

Задание:

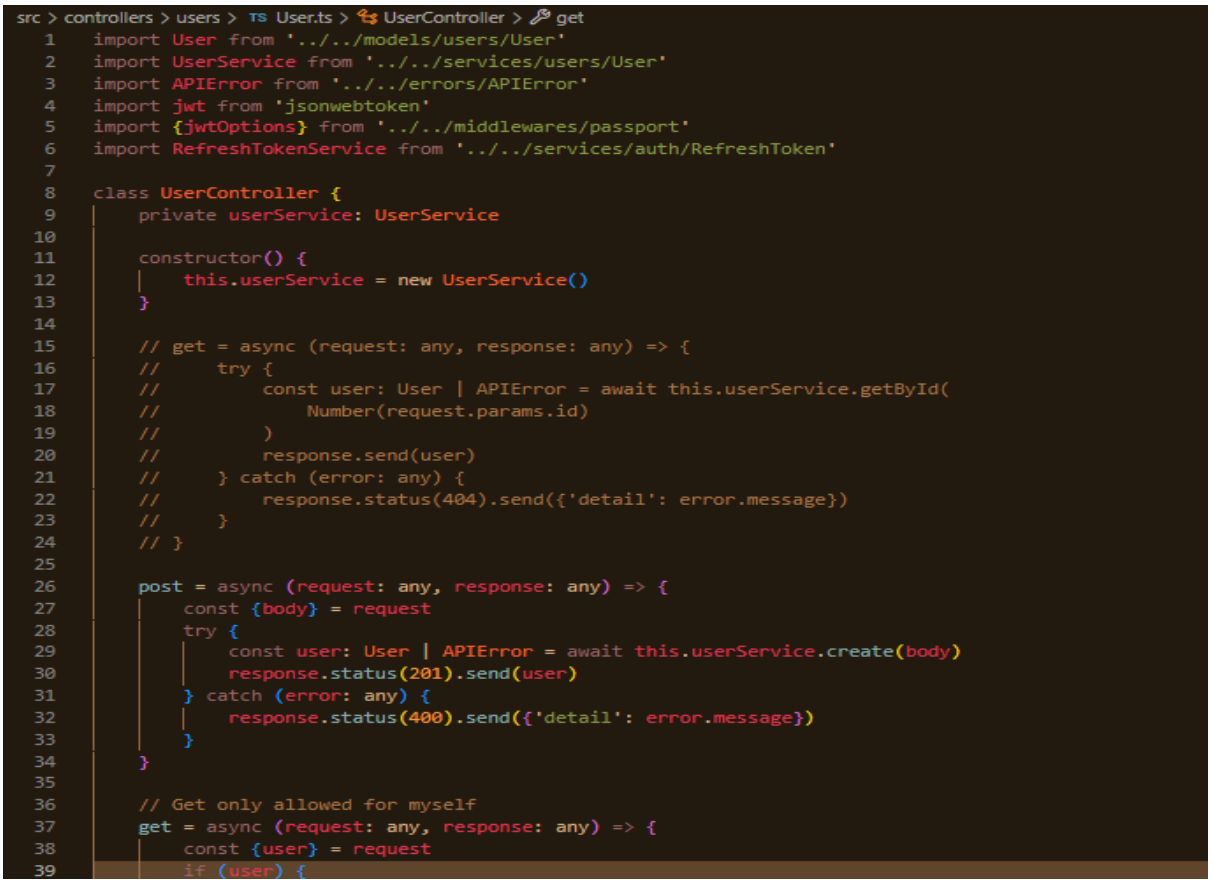
В рамках данной лабораторной работы Вам предложено выбрать один из нескольких вариантов. Выбранный вариант остается единым на весь курс и будет использоваться в последующих лабораторных работах.

По выбранному варианту необходимо будет реализовать RESTful API средствами express + typescript. Вариант: Сайт криптобиржи

- Вход
- Регистрация
- Портфель пользователя с указанием различных криптовалют и их количеством
- Графики роста криптовалют
- Поиск по криптовалютам с возможностью фильтрации по дате добавления на биржу

Выполнение работы

Controllers/users/user.ts



Models/user.ts

```ts
@DefaultScope(() => ({
    attributes: ['id', 'firstName', 'lastName', 'email']
}))
@Scopes(() => ({
    withPassword: {
        attributes: ['id', 'firstName', 'lastName', 'email', 'password', 'salt']
    }
}))
@Table
class User extends Model {
    @AllowNull(false)
    @Column
    firstName: string

    @AllowNull(false)
    @Column
    lastName: string

    @AllowNull(false)
    @Unique
    @Column
    email: string

    @HasMany(() => Wallet)
    wallets: [Wallet]

    @AllowNull(false)
    @Column
    password: string

    @Column
    salt: string

```

Routes/api/index.ts

src > routes > api > TS index.ts > ...

```ts
import express from 'express'
import userRoutes from './users/User'
import authRoutes from './auth/Auth'
import walletRoutes from './wallet/Wallet'
import coinRoutes from './wallet/Coin'

const router: express.Router = express.Router()

router.use('/users', userRoutes)
router.use('/auth', authRoutes)
router.use('/wallets', walletRoutes)
router.use('/coins', coinRoutes)

export default router

```

Services/User.ts

Providers/db.ts

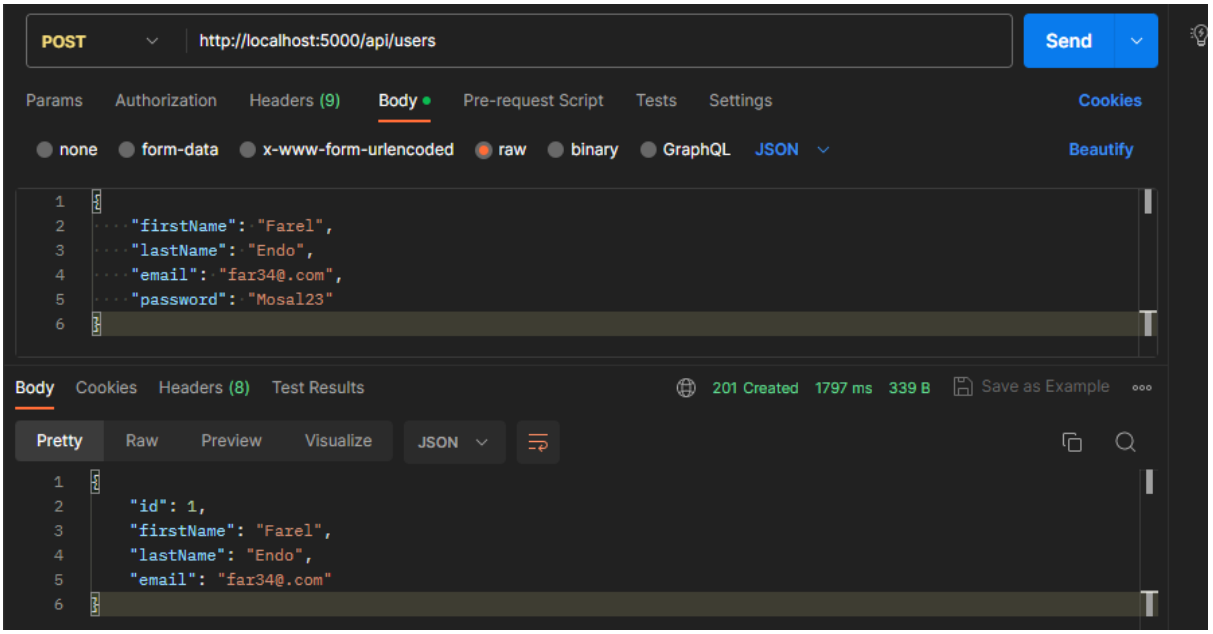
src > providers > TS db.ts > ...

```ts
import {Sequelize} from 'sequelize-typescript'
import RefreshToken from '../models/auth/RefreshToken'
import User from '../models/auth/User'
import Wallet from '../models/wallet/Wallet'
import Coin from '../models/wallet/Coin'
import CoinWallet from '../models/wallet/CoinWallet'
import {dbConfig} from '../configs'


const sequelize = new Sequelize({
    ...dbConfig,
    logging: console.log
})

const models = [User, RefreshToken, Wallet, Coin, CoinWallet]

sequelize.addModels(models)
console.log('Models added')

sequelize
    .sync()
    .then(() => {
        console.log('Models syncronized')
    })
    .catch((e) => console.log(e))

async function testConnection() {
    try {
        await sequelize.authenticate()
        console.log('Connection established successfully.')
    } catch (error) {
        console.error('Connection to the database failed:', error)
    }
}

testConnection()

export default sequelize
```

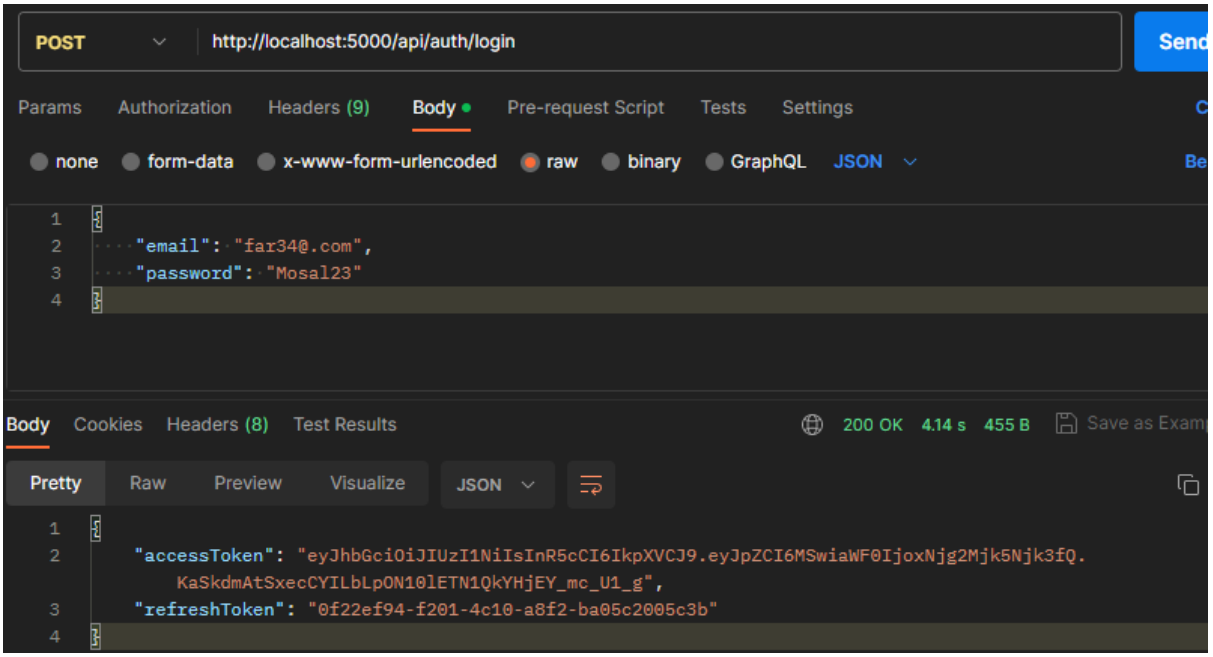
module "c:/Users/natha/OneDrive/Documents/Информатика/LW2/src/models/wallet/Wallet"

Postman

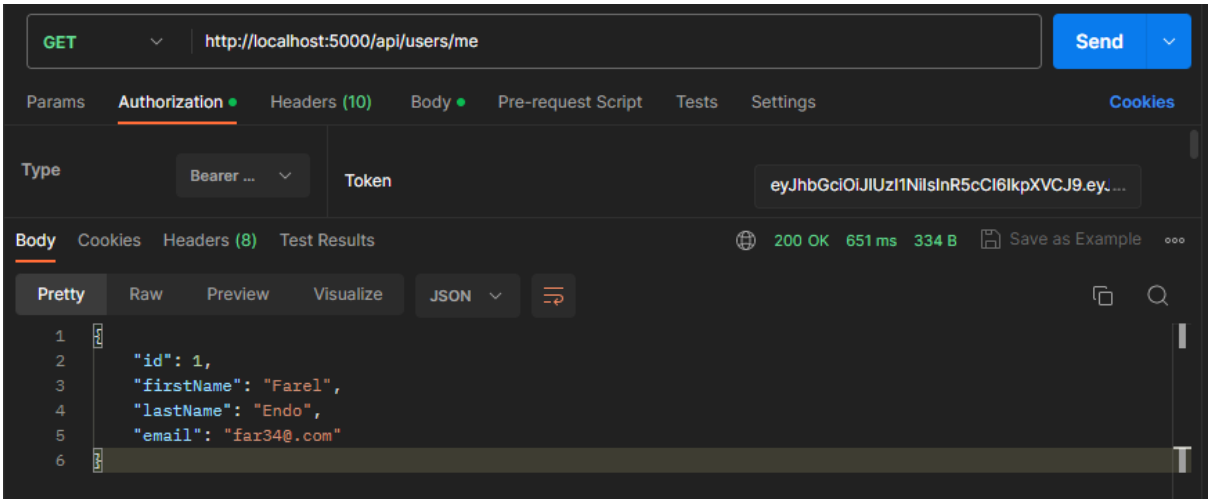
Add a new user



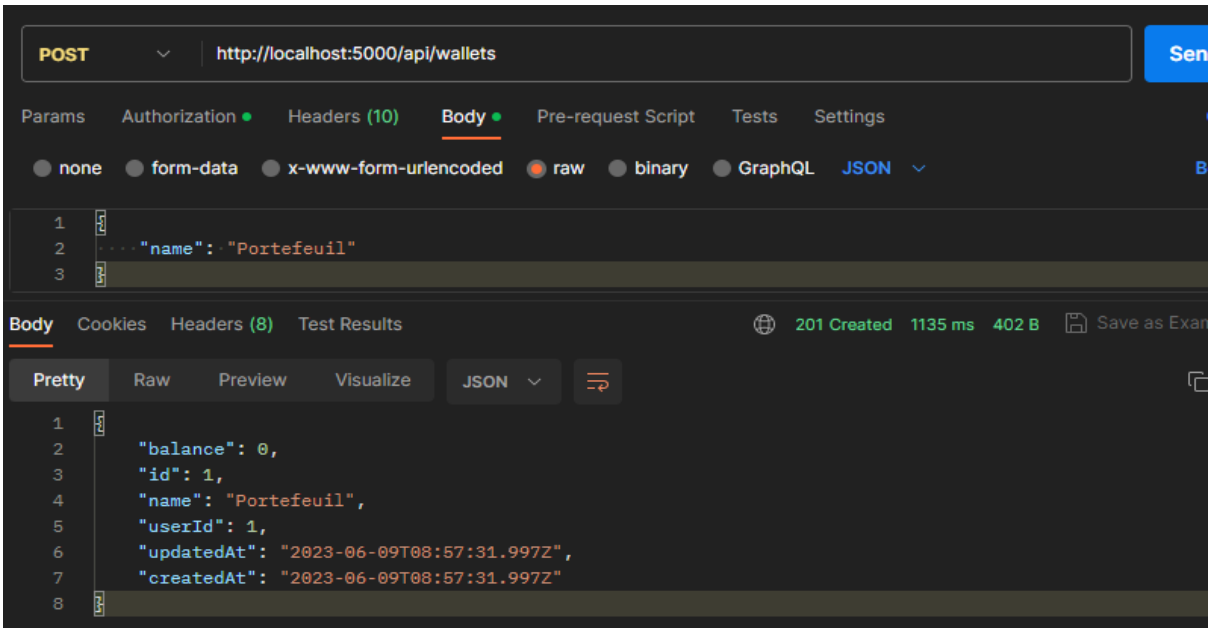
Login



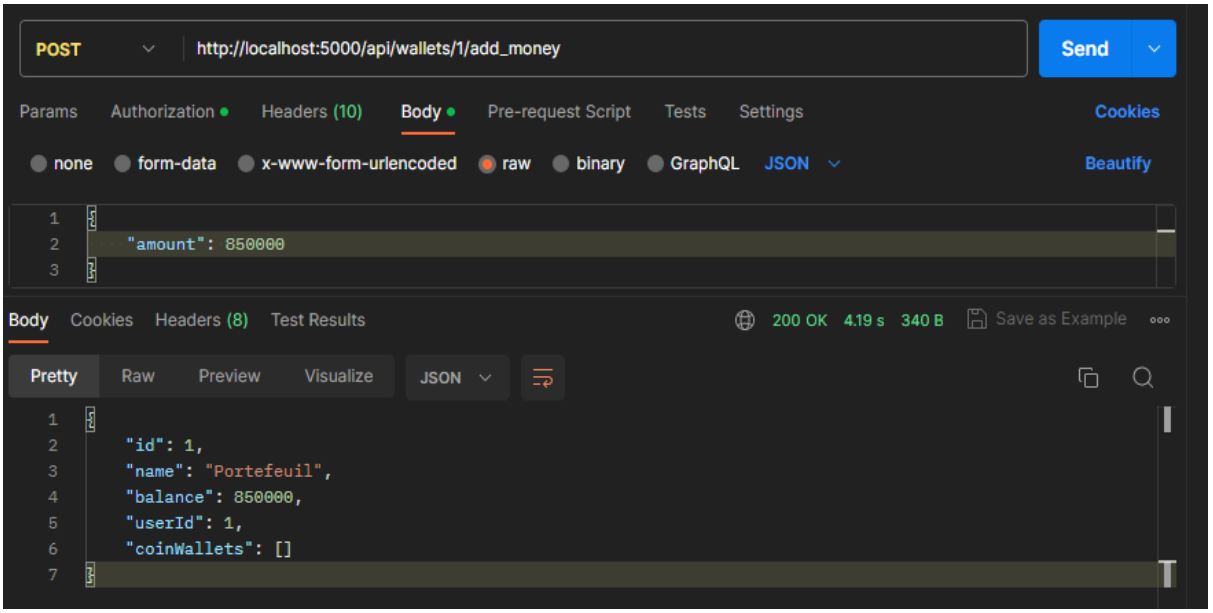
Get user profile (using the token)



Create a Wallet



Add money



Вывод:

При выполнения данной лабораторной работы, я научил реализовать RESTful API с использованием средств express + typescript и потестировал в Postman.