САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Лабораторная работа 2

Выполнил:

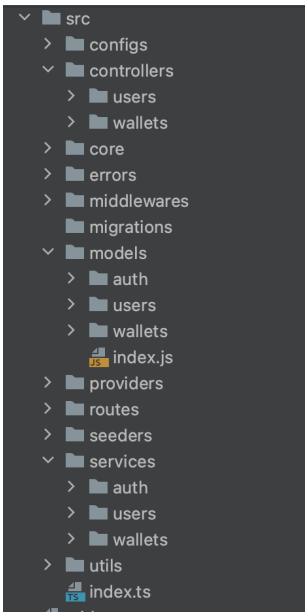
Романов Даниил

Группа
K33412

Проверил:
Добряков Д. И.

Санкт-Петербург
2023 г.

В качестве темы лабораторной работы выбрал платформу криптобиржу.



Структура проекта

```
@Table
class User extends Model {
    @AllowNull( allowNull: false)
    @Column
    firstName: string

    @AllowNull( allowNull: false)
    @Column
    lastName: string

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

    @AllowNull( allowNull: false)
    @Column
    password: string

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

    @BeforeCreate
    @BeforeUpdate
    static generatePasswordHash(instance: User) {
        const { password } = instance

        if (instance.changed( key: 'password')) {
            instance.password = hashPassword(password)
        }
    }
```

Модель пользователя

```
@Table
class Coin extends Model {
    @PrimaryKey
    @AllowNull( allowNull: false)
    @Column
    ticker: string

    @AllowNull( allowNull: false)
    @Column
    name: string

    @BelongsToMany( associatedClassGetter: () => Wallet,  through: () => CoinWallet)
    wallets: Wallet[]
}

export default Coin
```

Модель Coin

```
    nested: {
        attributes: ['id', 'name', 'balance', 'userId'],
        include: [{
            model: CoinWallet,
            attributes: ['amount']
        }],
        nest: true
    }
}))
@Table
class Wallet extends Model {
    @AllowNull( allowNull: false)
    @Column
    name: string

    @ForeignKey( relatedClassGetter: () => User)
    @Column
    userId: number

    @Default( value: 0)
    @Column
    balance: number

    @BelongsTo( associatedClassGetter: () => User)
    user: User

    @HasMany( associatedClassGetter: () => CoinWallet)
    coinWallets: CoinWallet[]

    @BelongsToMany( associatedClassGetter: () => Coin,  through: () => CoinWallet)
    coins: Coin[]
```

Модель Wallet

```
@Table
class CoinWallet extends Model {
    @ForeignKey( relatedClassGetter: () => Coin)
    @Column
    coinId: number

    @ForeignKey( relatedClassGetter: () => Wallet)
    @Column
    walletId: number

    @AllowNull( allowNull: false)
    @Column
    amount: number
}
```

Связывающая таблица WalletCoin

```
router.route( prefix: '/').post(userController.create)
router.route( prefix: '/profile').get(passport.authenticate( strategy: 'jwt',  options: { session: false }), userController.me)
router.route( prefix: '/profile/:id').get(userController.get)
router.route( prefix: '/login').post(userController.auth)
router.route( prefix: '/refresh').post(userController.refreshToken)
```

```
router.route( prefix: '/').get(coinController.getAll)
router.route( prefix: '/:ticker').get(coinController.get)
router.route( prefix: '/:ticker/price').get(coinController.getPrice)
router.route( prefix: '/:ticker/candles').get(coinController.getCandles)
```

```
router.route( prefix: '/').get(passport.authenticate( strategy: 'jwt',  options: { session: false }), walletController.getAll)
router.route( prefix: '/').post(passport.authenticate( strategy: 'jwt',  options: { session: false }), walletController.create)
router.route( prefix: '/:id').get(passport.authenticate( strategy: 'jwt',  options: { session: false }), walletController.get)
router.route( prefix: '/:id/increase').post(passport.authenticate( strategy: 'jwt',  options: { session: false }), walletControlle
router.route( prefix: '/:id/decrease').post(passport.authenticate( strategy: 'jwt',  options: { session: false }), walletControlle
router.route( prefix: '/:id/buy_coin').post(passport.authenticate( strategy: 'jwt',  options: { session: false }), walletControlle
router.route( prefix: '/:id/sell_coin').post(passport.authenticate( strategy: 'jwt',  options: { session: false }), walletControll
```

Вывод:
В ходе лабораторной работы реализовал платформу с криптобиржей, реализовал RESTful API.