САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бек-энд разработка

Отчет

Лабораторная работа №1

Выполнил:

Нгуен Хоанг Туан
Группа К33402

Проверил:
Добряков Д. И.

Санкт-Петербург
2021 г.

Задача

Нужно написать свой boilerplate на express + sequelize + typescript.

Должно быть явное разделение на:

- модели
- контроллеры
- роуты
- сервисы для работы с моделями (реализуем паттерн “репозиторий”)

Ход работы

Model/index.ts

```
import { DataTypes, Model, Optional } from 'sequelize';
import db from '../config/database.config';

interface TodoAttributes {
    id: number;
    firstname: string;
    lastname: string;
    email: string;
}

export class TodoInstance extends Model<TodoAttributes> {
    public id!: number
    public firstname!: string
    public lastname!: string
    public email!: string

    public readonly createdAt!: Date;
    public readonly updatedAt!: Date;
    public readonly deletedAt!: Date;
}

TodoInstance.init(
    {
        id: {
            type: DataTypes.NUMBER,
            primaryKey: true,
            allowNull: false,
        },
```

```
        firstname: {
            type: DataTypes.STRING,
            allowNull: false,
        },
        lastname: {
            type: DataTypes.STRING,
            allowNull: false,
        },
        email: {
            type: DataTypes.STRING,
            allowNull: false,
            unique: true,
        },
    },
    {
        timestamps: true,
        sequelize: db,
        tableName: 'todos',
    }
);
export interface TodoInput extends Optional<TodoAttributes, 'id' & 'firstname'
& 'lastname' & 'email'> {}
export interface TodoOuput extends Required<TodoAttributes> {}
```

Routes/example/index.ts

```
import express from "express"
import ExampleController from '../../controllers/example/index'

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

const exampleController = new ExampleController()

router.get(
    '/gioithieu',
    exampleController.gioithieu
)

router.post(
    '/create',
    exampleController.create
)

router.get(
    '/getdata/:id',
    exampleController.get
)
```

```
router.get(
    '/getdata',
    exampleController.getall
)

router.put(
    '/update/:id',
    exampleController.update
)

router.delete(
    '/delete/:id',
    exampleController.delete
)
export default router
```

Controllers/example/index.ts

```
import express, {Request, Response } from 'express';
import db from "../../config/database.config";
import { TodoInstance } from '../../model';
import UserError from '../../errors/users/index';
import UserService from '../../services/index';
import { ValidationErrorItem } from 'sequelize/types';
db.sync().then(() => {
  console.log('connect');
});

class ExampleController {
  private userService: UserService

  constructor() {
      this.userService = new UserService()
  }
  gioithieu = async (request: any, response: any) => {
    response.send('Hello, world!')
  }
  create = async(request: any, response: any) => {
    const { body } = request
    try {
      const user : TodoInstance|UserError = await
this.userService.create(body)
      response.status(201).send(user)
    } catch (error: any) {
      console.log(error)
      response.status(404).send({ "error": error.message})
    }
  }
  get = async(request: any, response: any) => {
```

```
    const user = await this.userService.getById(
      Number(request.params.id)
    )
    if (!user) {
      response.status(400).send('Not found')
    }
    else
      response.status(201).send(user)
  }
  getall = async(request: any, response: any) => {
    const user = await this.userService.findAll()
    response.status(201).send(user)
  }
  update = async(request: any, response: any) => {
    const id = Number(request.params.id)
    const { body} = request
    try {
      const user = await this.userService.update(id, body)
      if (user)
        response.status(201).send(user)
      else
        response.status(400).send('Not found')
    } catch (error: any) {
      console.log(error)
      response.status(404).send({ "error": error.message})
    }
  }
  delete = async(request: any, response: any) => {
    const user = await this.userService.delete(
      Number(request.params.id)
    )
    if (!user) {
      response.status(400).send('Not found')
    }
    else
      response.status(201).send("Was delete")
  }
}
export default ExampleController
```

Services/index.ts

```
import express, {Request, Response } from 'express';
import db from "../config/database.config";
import { TodoInput, TodoOuput, TodoInstance } from '../model';
import UserError from '../errors/users';
class UserService {
    async create(userData: TodoInput) : Promise<TodoInstance|UserError> {
        try {
```

```
            const user = await TodoInstance.create(userData)
            return(user)
        } catch(e: any) {
            const errors = e.errors.map((error: any) => error.message)
            throw new UserError(errors)
        }
    }
    async getById(id: number) : Promise<TodoOuput|null> {
        const data = await TodoInstance.findByPk(id)
        return data
    }
    async findAll() : Promise<TodoOuput[]> {
        return TodoInstance.findAll()
    }
    async update(id: number, userData: Partial<TodoInput>) :
Promise<TodoInstance|UserError|null> {
        try {
            const data = await TodoInstance.findByPk(id)
            if (data) {
                const result = await (data as TodoInstance).update(userData)
                return result
            }
            else
                return data
        }catch(e: any) {
            const errors = e.errors.map((error: any) => error.message)
            throw new UserError(errors)
        }
    }
    async delete(id: number) : Promise<boolean> {
        const deldata = await TodoInstance.destroy({
            where: {id}
        })
        return !!deldata
    }
}
export default UserService
```

Вывод : Create model by sequelize, write router, controllers, services