

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ  
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

**Дисциплина:** Бек-энд разработка

Отчет

Лабораторная работа 2

Выполнил:

Старовойтова Елизавета

Группа:

К33402

Проверил:

Добряков Д. И.

Санкт-Петербург

2024 г.

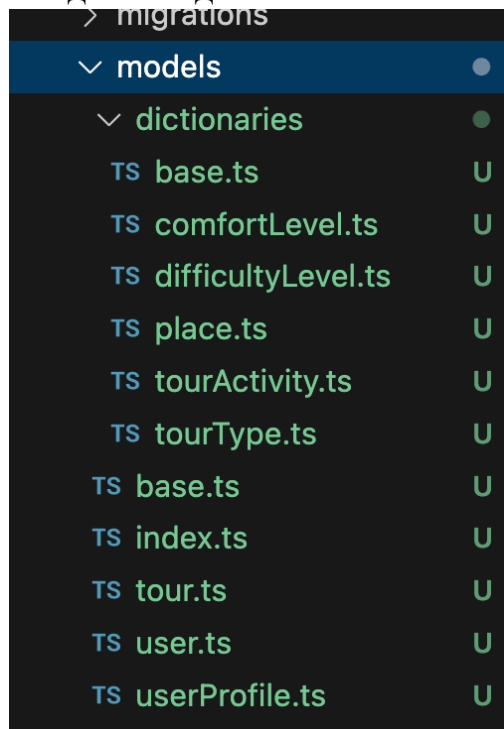
## Задача

По выбранному варианту необходимо будет реализовать RESTful API средствами express + typescript (используя ранее написанный boilerplate).

Вариант: Сервис-помощник в планировании путешествий.

## Ход работы

Создаем модели.



На примере тура.

```

class Tour extends BaseModel {
  @Column
  name: string
  @Column
  price: number
  @Column
  canGoWithChildren: boolean
  @Column
  maxPeople: number

  @Column
  @ForeignKey(() => ComfortLevel)
  comfortLevelId: number
  @Column
  @ForeignKey(() => DifficultyLevel)
  difficultyLevelId: number
  @Column
  @ForeignKey(() => Place)
  placeId: number

  @BelongsToMany(() => TourActivity, {
    through: 'tour_has_tour_activity',
    foreignKey: 'tourId',
    otherKey: 'tourActivityId',
  })
  tourActivities: Array<TourActivity>
  @BelongsToMany(() => TourType, {
    through: 'tour_has_tour_type',
    foreignKey: 'tourId',
    otherKey: 'tourTypeId',
  })
  tourTypes: Array<TourType>

  @BelongsToMany(() => ComfortLevel)
  comfortLevel: ComfortLevel
  @BelongsToMany(() => DifficultyLevel)
  difficultyLevel: DifficultyLevel
  @BelongsToMany(() => Place)
  place: Place
}

```

Создаем контроллеры. Пример с туром.

```
find = async (request: express.Request, response: express.Response, next: NextFunction) => {
  try {
    const tour: Tour = await FindTourUseCase.run(Number(request.params.id))
    ApiResponse.payload(response, transform(tour, new TourTransformer()))
  } catch (e: any) {
    next(e)
  }
}

store = async (request: express.Request, response: express.Response, next: NextFunction) => {
  try {
    await this.validate(request, [
      body('name').notEmpty().isString(),
      body('price').notEmpty().isInt(),
      body('canGoWithChildren').notEmpty().isBoolean(),
      body('maxPeople').notEmpty().isInt(),
      body('comfortLevelId').notEmpty().isInt(),
      body('difficultyLevelId').notEmpty().isInt(),
      body('placeId').notEmpty().isInt(),
      body('tourActivities').notEmpty().isArray(),
      body('tourTypes').notEmpty().isArray(),
    ])
    const tour: Tour = await StoreTourUseCase.run(request.body)
    ApiResponse.payload(response, transform(tour, new TourTransformer()))
  } catch (e: any) {
    next(e)
  }
}
```

Сервис (useCase) на примере с туром.

```
class FindTourUseCase {
  static async run(id: number): Promise<Tour> {
    // @ts-ignore
    const tour = await Tour.findByPk(id, {
      include: [
        { model: ComfortLevel },
        { model: DifficultyLevel },
        { model: Place },
        { model: TourActivity },
        { model: TourType },
      ]
    });
    if (!tour) {
      throw new NotFoundError('Tour does not exists')
    }
    return tour
  }
}

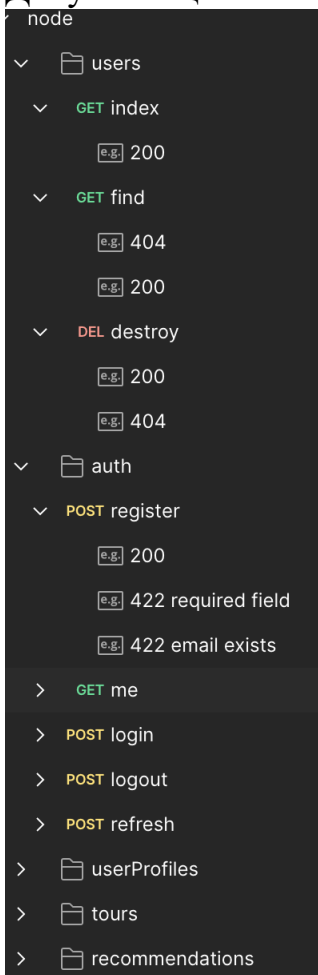
export default FindTourUseCase
```

Маршруты приложения.

```
import usersRouter from './users'
import authRouter from './auth'
import userProfilesRouter from './userProfiles'
import toursRouter from './tours'
import recommendationsRouter from './recommendations'

export const routers = [
  { prefix: '/users', router: usersRouter },
  { prefix: '/auth', router: authRouter },
  { prefix: '/user_profiles', router: userProfilesRouter },
  { prefix: '/tours', router: toursRouter },
  { prefix: '/recommendations', router: recommendationsRouter },
]
```

## Документация в Postman.



GETlocalhost:4000/user\_profiles/my ...

Send

ParamsAuthorizationHeaders (7)BodyPre-request ScriptTestsSettingsCookies

Query Params

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
	Key	Value	Description		

bodyCookies (2)Headers (7)Test Results

Status: 200 OKTime: 33 msSize: 420 BSave Response

PrettyRawPreviewVisualizeJSON

```
1  {
2    "userId": 1,
3    "maxBudget": 10000,
4    "hasChildren": true,
5    "peopleCount": null,
6    "comfortLevels": [],
7    "difficultyLevels": [],
8    "places": [],
9    "tourActivities": [
10     {
11       "id": 1,
12       "name": "Bike tours"
13     }
14   ],
15   "tourTypes": []
16 }
```

GETlocalhost:4000/recommendations/tours/2

Send

ParamsAuthorizationHeaders (7)BodyPre-request ScriptTestsSettingsCookies

Query Params

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
	Key	Value	Description		

BodyCookies (2)Headers (7)Test Results

Status: 200 OKTime: 26 msSize: 1.22 KBSave Response

PrettyRawPreviewVisualizeJSON

```
29  {
30    "id": 3,
31    "name": "TOUR",
32    "price": 100,
33    "canGoWithChildren": true,
34    "maxPeople": 100,
35    "comfortLevel": {
36      "id": 1,
37      "name": "Basen"
38    },
39    "difficultyLevel": {
40      "id": 1,
41      "name": "Base"
42    },
43    "place": {
44      "id": 1,
45      "name": "Moscow"
46    },
47    "tourActivities": [
48      {
49        "id": 1,
50        "name": "Bike tours"
51      }
52    ]
53  }
```

Вывод:

В ходе данной работы был написан свой RESTful API средствами express + typescript (используя ранее написанный boilerplate).