

**Министерство науки и высшего образования Российской Федерации**  
федеральное государственное автономное образовательное учреждение высшего образования  
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Отчет**

по лабораторной работе №5

по дисциплине «Web-программирование»

Автор: Шарапков Егор Викторович М33081

Преподаватель: Приискалов Роман Андреевич



**УНИВЕРСИТЕТ ИТМО**

Санкт-Петербург 2022

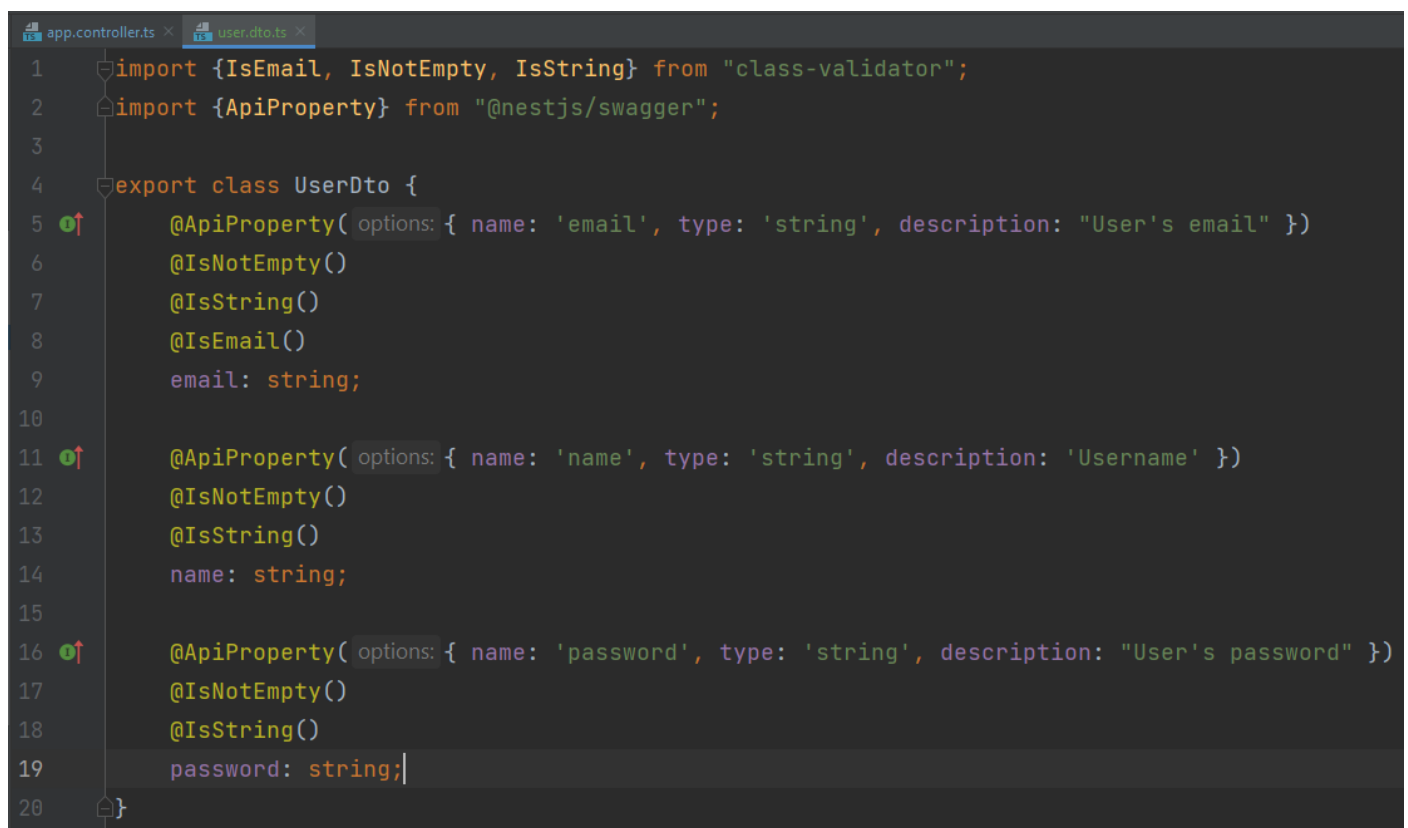
Для выполнения данной лабораторной работы необходимо было реализовать в ранее созданном приложении, контроллеры и сервисы, которые были спроектированы в рамках предыдущей лабораторной работы.

Также необходимо было поработать с валидацией, для этого установим необходимую зависимость.

```
npm i --save class-validator class-transformer
```

Для корректной обработки всех входящих запросов рекомендуется использовать встроенные механизмы NestJS, например Guards. У guards есть единственная ответственность. Они определяют, будет ли данный запрос обработан обработчиком маршрута или нет, в зависимости от определенных условий (таких как разрешения, роли и т.д.), существующих во время выполнения.

Для валидации нужно использовать декораторы, для этого создадим для моделей DTO (Data Transfer Object). Это один из шаблонов проектирования, используется для передачи данных между подсистемами приложения.



```
1 import {IsEmail, IsNotEmpty, IsString} from "class-validator";
2 import {ApiProperty} from "@nestjs/swagger";
3
4 export class UserDto {
5   @ApiProperty( options: { name: 'email', type: 'string', description: "User's email" })
6   @IsNotEmpty()
7   @IsString()
8   @IsEmail()
9   email: string;
10
11   @ApiProperty( options: { name: 'name', type: 'string', description: 'Username' })
12   @IsNotEmpty()
13   @IsString()
14   name: string;
15
16   @ApiProperty( options: { name: 'password', type: 'string', description: "User's password" })
17   @IsNotEmpty()
18   @IsString()
19   password: string;
20 }
```

Также необходимо было реализовать проверку входных данных с использованием привязки ValidationPipe на уровне приложения. Чтобы обеспечить защиту всех конечных точек от получения неправильных данных. Если валидация реализована верно, то при отправке не валидного объекта ожидается ответ от сервера HTTP 400 Bad Request.

```
const app = await NestFactory.create<NestExpressApplication>(AppModule);
app.useGlobalPipes(new ValidationPipe( options: {errorHttpStatusCode: 400,}));
```

Мы можем создать пользовательскую проверку, а затем генерировать собственное HTTP-исключение, которое сразу отправить ответ запрашивающему с кодом состояния и собственным сообщением.

Например, внутри реализации `user.service` можно добавить проверку на то, является ли входящее `id` числом.

```
async getUser(id: number): Promise<User> {  
  if (!+id) throw new HttpException( response: 'User id should be a number', status: 400);
```

Далее реализуем CRUD операции. CRUD – это акроним, обозначающий четыре базовые функции, используемые при работе с базами данных: создание (create), чтение (read), модификация (update), удаление (delete). Они реализованы в контроллерах и сервисах для существующих моделей.

```
async getUser(id: number): Promise<User> {  
  if (!+id) throw new HttpException( response: 'User id should be a number', status: 400);  
  const user = await prisma.user.findUnique({  
    where: {  
      id: id,  
    },  
  });  
  if (user) {  
    return user;  
  }  
  throw new NotFoundException( objectOrError: "User is not found");  
}  
  
async createUser(CreateUserDto: UserDto): Promise<User> {  
  const { email, name, password } = CreateUserDto;  
  return await prisma.user.create({  
    data: {  
      email: email,  
      name: name,  
      password: password,  
    },  
  });  
}
```

```
async updateUser(id: number, CreateUserDto: UserDto): Promise<User> {  
  const { email, name } = CreateUserDto;  
  return await prisma.user.update({  
    where: {  
      id: id,  
    },  
    data: {  
      email: email,  
      name: name,  
    },  
  });  
}
```

```
async deleteUser(id: number): Promise<void> {  
  const user = await this.getUser(id);  
  if (user) {  
    await prisma.user.delete( { where: { id: id } });  
  }  
}
```

После выполнения всех пунктов выше, можно протестировать работоспособность нашего API.

Проходим по пути /api



## Apple Market API 0.1 OAS3

Early version of Api

### user

GET	/user	Get all users	⌵
GET	/user/{id}	Get a user by id	⌵
POST	/user/create	Create a user by email, name and password	⌵
POST	/user/{id}/update	Update user by id	⌵
DELETE	/user/{id}/delete	Delete a user by id	⌵

### device

GET	/device	Get all devices	⌵
POST	/device	Create a device by title, description and price	⌵
GET	/device/{id}	Get a device by id	⌵
DELETE	/device/{id}	Delete a device by id	⌵
POST	/device/{id}/update	Update device by id	⌵

Также можем увидеть схемы.

### Schemas

```
UserDto {
  email*      string
              User's email
  name*       string
              Username
  password*   string
              User's password
}
```

```
DeviceDto {
  title*      string
              Title of device
  description* string
              Description of device
  price*      integer
              Price of device
  basketId*   integer
              Id user's basket
}
```

```
BasketDto {
  userId*     integer
              id of user
  totalPrice* integer
              Total price of order
  devices*    > [...]
}
```

Попробуем создать пользователя.

POST /user/create Create a user by email, name and password

Parameters

No parameters

Request body required

application/json

```
{  "email": "root@gmail.com",  "name": "root",  "password": "secretpassword1135452"}  
```

Execute

Пользователь успешно создан.

Responses

Curl

```
curl -X 'POST' \  'http://localhost:3000/user/create' \  -H 'accept: */*' \  -H 'Content-Type: application/json' \  -d '{  "email": "root@gmail.com",  "name": "root",  "password": "secretpassword12355326"}'
```

Request URL

http://localhost:3000/user/create

Server response

Code	Details
201	<div><div>Response body</div><div><pre>{  "id": 3,  "email": "root@gmail.com",  "password": "secretpassword12355326",  "name": "root",  "isLoggedIn": false}</pre></div><div>Download</div></div> <div><div>Response headers</div><div><pre>connection: keep-alive  content-length: 102  content-type: application/json; charset=utf-8  date: Mon, 25 Apr 2022 13:22:38 GMT  etag: W/"66-gIedk0suq4EEzMVmiXbHIPVKymQ"  keep-alive: timeout=5  x-powered-by: Express</pre></div></div>

Responses

Code	Description	Links
201	User created	No links

Можем запросить всех пользователей и увидеть нашего нового пользователя.

GET /user Get all users

Parameters

No parameters

Execute Clear

Responses

Curl

curl -X 'GET' \
'http://localhost:3000/user' \
-H 'accept: \*/\*'

Request URL

http://localhost:3000/user

Server response

Code Details

200

Response body

{
 "id": 3,
 "email": "root@gmail.com",
 "password": "secretpassword12355326",
 "name": "root",
 "isLoggedIn": false
}

Response headers

connection: keep-alive
content-length: 184
content-type: application/json; charset=utf-8
date: Mon, 25 Apr 2022 13:24:22 GMT
etag: W/"68-jvnR3xxbopskxWdAU0jft5+2G48"
keep-alive: timeout=5
x-powered-by: Express

Добавим девайсы, проверим.

Responses

Curl

curl -X 'GET' \
'http://localhost:3000/device' \
-H 'accept: \*/\*'

Request URL

http://localhost:3000/device

Server response

Code Details

200

Response body

{
 "price": 999999,
 "basketId": null
},
{
 "id": 5,
 "title": "Apple iPad 10,2 (2021) Wi-Fi + Cellular 256 ГБ, серый космос",
 "description": "Просто. Нереально. Дисплей Super Retina XDR с технологиями ProMotion и быстрым, плавным откликом. Грандиозный апгрейд системы камер, открывающий совершенно новые возможности. Исключительная прочность. A15 Bionic – самый быстрый чип для iPhone. И впечатляющее время работы без подзарядки. Всё это Pro.",
 "price": 100500,
 "basketId": null
},
{
 "id": 6,
 "title": "Apple iPhone 13 Pro Max, 256ГБ, графитовый",
 "description": "Просто. Нереально. Дисплей Super Retina XDR с технологиями ProMotion и быстрым, плавным откликом. Грандиозный апгрейд системы камер, открывающий совершенно новые возможности. Исключительная прочность. A15 Bionic – самый быстрый чип для iPhone. И впечатляющее время работы без подзарядки. Всё это Pro.",
 "price": 180999,
 "basketId": null
},
{
 "id": 7,
 "title": "Apple MacBook Pro 16 32 ГБ, 1 ТБ SSD, серебристый",
 "description": "Суперсила профессионалов. Супербыстрые чипы M1 Pro и M1 Max дают феноменальную производительность и обеспечивают удивительно долгое время работы без подзарядки. Прибавьте к этому потрясающий дисплей Liquid Retina XDR и ещё больше портов для профессиональной работы. Это тот самый ноутбук, который вы так ждали.",
 "price": 256789,
 "basketId": null
}
]

Response headers

content-length: 1947
content-type: application/json; charset=utf-8
date: Mon, 25 Apr 2022 13:25:51 GMT
etag: W/"75b-mNV1fmhmqdYzoeHRNV+U9XrKg"
x-powered-by: Express