Leaseweb code test:

Name: Antonis Sideris
email: antonisid@gmail.com
LinkeDin: https://www.linkedin.com/in/antonis-sideris-a7914046/
GitHub:https://github.com/antonisid

Api:

Rest API capable of listing server(s) and narrow the list by applying filters. The results are paginated(10) and all the requests are validated before performing any actions. The user needs to register and login in order to retrieve the servers. I used Laravel framework and PHP 7.4.6.

Installation:

(through the command line)

a) git clone https://github.com/antonisid/rest-servers-api.git
b) composer install
c) copy the .env.example to .env file (cp .env.example .env)
d) create a mysql DB and add the credentials to the following section of the .env file:

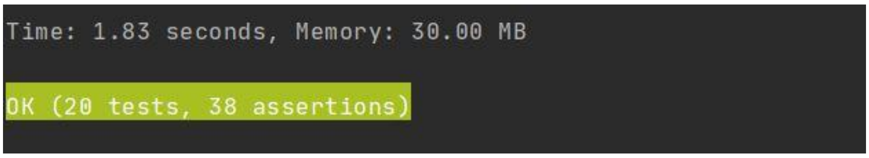
```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=leaseweb
DB_USERNAME=root
DB_PASSWORD=password
```

e) php artisan migrate (creates the tables)
f) php artisan db:seed --class=ServerSeeder (seeds the DB from the csv file)
g) php artisan jwt:secret
h) php artisan serve (serves the application on 127.0.0.1:8000)

*You can refresh the database with: php artisan migrate:fresh

Unit testing

You can execute the tests with the following command:
Mac: ./vendor/bin/phpunit - Windows: "./vendor/bin/phpunit"

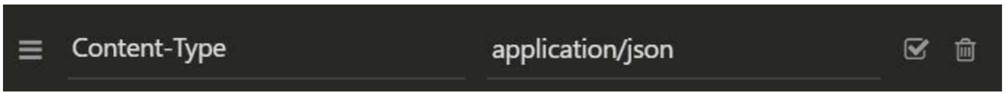


```
Time: 1.83 seconds, Memory: 30.00 MB

OK (20 tests, 38 assertions)
```

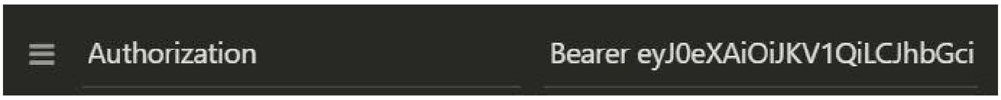
Endpoints:

The POST requests for the login & register endpoints should be sent with the following header:



Content-Type application/json

The GET requests should be sent with the following header along with the token that was retreived after login.



Authorization Bearer eyJ0eXAiOiJKV1QiLCJhbGci

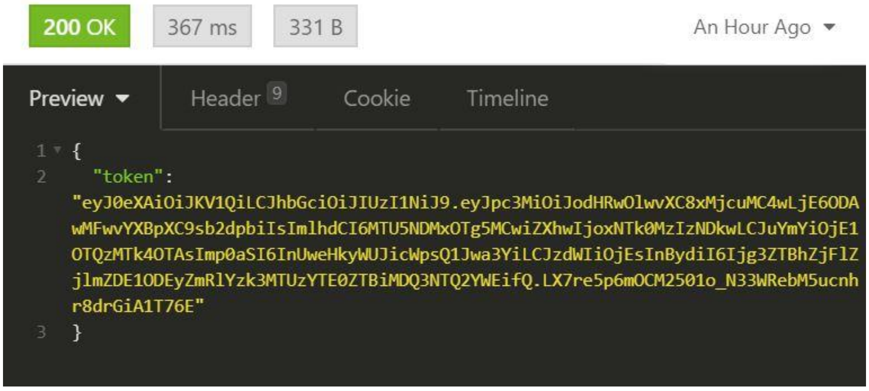
User Register (POST)**:**
127.0.0.1:8000/api/register

```
{
	"name" : "Antonis",
	"email": "test@demo.com",
	"password": "testtest",
	"password_confirmation": "testtest"
}
```

User login (POST)**:**
127.0.0.1:8000/api/login

```
{
	"email": "test@demo.com",
	"password": "testtest"
}
```

Example response:



200 OK 367 ms 331 B An Hour Ago

```
{
  "token":
"eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJodHRwOlwvXC8xMjcuMC4wLjE6ODA
wMFwvYXBpXC9sb2dpbiIsImlhdCI6MTU5NDMxOTg5MCwiZXhwIjoxNTk0MzIzNDkwLCJuYmYiOjE1
OTQzMTk4OTAsImp0aSI6InUweHkyWUJicWpsQ1Jwa3YiLCJzdWIiOjEsInBydiI6Ijg3ZTBhZjFlZ
jlmZDE1ODEyZmRlYzk3MTUzYTE0ZTBiMDQ3NTQ2YWEifQ.LX7re5p6mOCM2501o_N33WRebM5ucnh
r8drGiA1T76E"
}
```

List servers (GET):
127.0.0.1:8000/api/servers (equivalent to: 127.0.0.1:8000/api/servers?page=1)

List servers by location(GET):
127.0.0.1:8000/api/servers?location=AmsterdamAMS-01

List servers by storage(GET):
127.0.0.1:8000/api/servers?storage[min]=3000&storage[max]=4000

List servers by ram(GET):
127.0.0.1:8000/api/servers?ram[]=8&ram[]=32

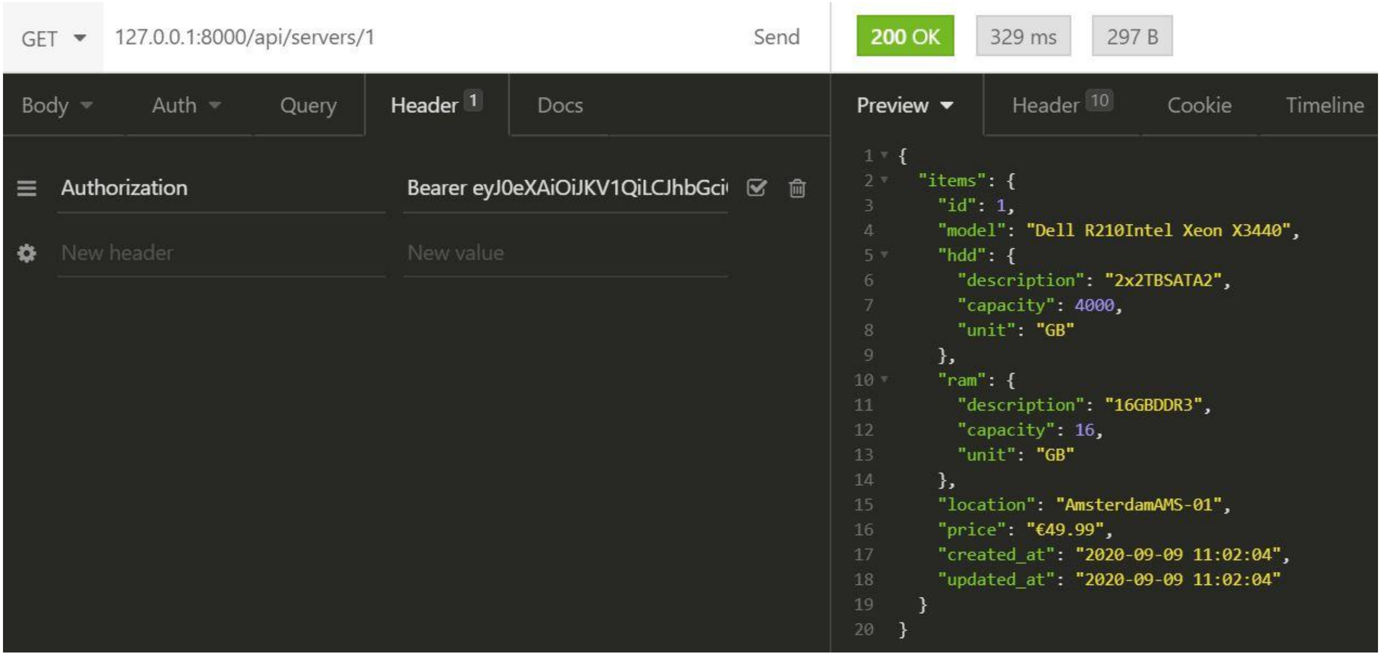
List servers by hard disk type(GET):
127.0.0.1:8000/api/servers?hard_disk_type=ssd

List servers using pagination (GET):
127.0.0.1:8000/api/servers?page=2

List servers with all the filters combined (GET):
127.0.0.1:8000/api/servers?location=AmsterdamAMS-01&hard_disk_type=sata&ram[]=8&ram[]=32&storage[min]=3000&storage[max]=4000

Get server (GET):
127.0.0.1:8000/api/servers/1

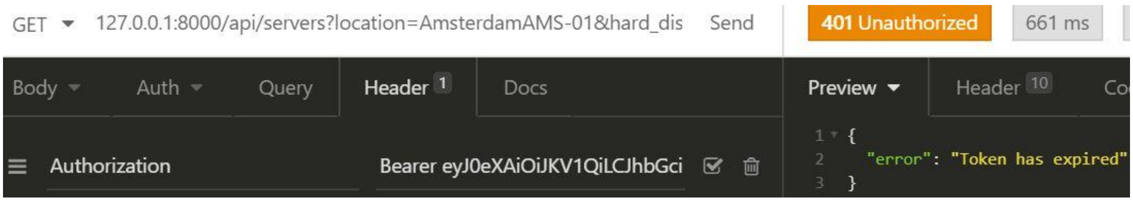
Example responses:



GET 127.0.0.1:8000/api/servers/1

Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGci

```
{
  "items": {
    "id": 1,
    "model": "Dell R210Intel Xeon X3440",
    "hdd": {
      "description": "2x2TBSATA2",
      "capacity": 4000,
      "unit": "GB"
    },
    "ram": {
      "description": "16GBDDR3",
      "capacity": 16,
      "unit": "GB"
    },
    "location": "AmsterdamAMS-01",
    "price": "€49.99",
    "created_at": "2020-09-09 11:02:04",
    "updated_at": "2020-09-09 11:02:04"
  }
}
```

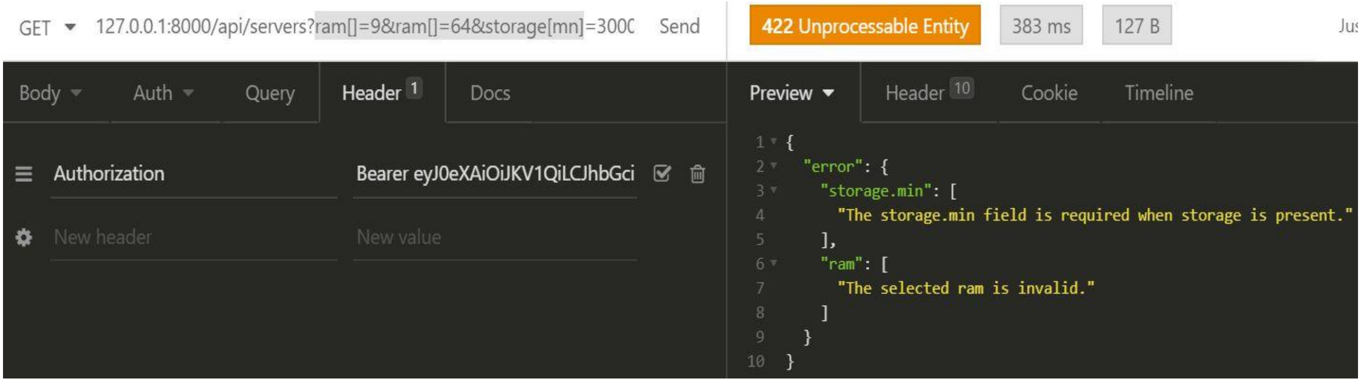


GET 127.0.0.1:8000/api/servers?location=AmsterdamAMS-01&hard_dis Send

401 Unauthorized 661 ms

Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGci

```
{
  "error": "Token has expired"
}
```



GET 127.0.0.1:8000/api/servers?ram[]=9&ram[]=64&storage[mn]=300C Send

422 Unprocessable Entity 383 ms 127 B

Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGci

New header: New value

```
{
  "error": {
    "storage.min": [
      "The storage.min field is required when storage is present."
    ],
    "ram": [
      "The selected ram is invalid."
    ]
  }
}
```

Example code flow for listing servers:

JwtMiddleware->ServerController->ServerService->ServerRepository->ServerFilter ->Filter Interface*->ServerDto->ServerTransformer->JsonResponse

*Location, Ram, Storage, HardDiskType

You may start browsing starting from the following controllers:

UserController
ServerController

Request validation:

App/Http/Requests/

Exception handler:

App/Exceptions/Handler.php

Tests:

/tests/Feature/
/tests/Unit/

Filters/
App/ServerFilter/Filters/

Thank you for your time.

If you have any questions or encounter issues please feel free to contact me.

Regards,
Antonis Sideris