

Documentação REST

IADE - Faculdade de Design, Tecnologia e Comunicação
Engenharia Informática — 3.º semestre
Felipe Campelo Sabbado (20191012); Leonardo Lage (20200859)

Programação Orientada por Objetos (Miguel Bugalho);

ÍNDICE

Recurso Achievements	4
Ir buscar todos os Achievements	4
Ir buscar um Achievement dado um id	4
Adicionar um Achievement	4
Apagar um Achievement dado um id	5
Recurso Routes	6
Ir buscar todas as Rotas	6
Ir buscar uma Rota dado um id	6
Adicionar uma Rota	6
Apagar a rota dado um id	7
Filtrar as Rotas de acordo com o nome e a distância	7
Ir buscar a Média das avaliações das Rotas	8
Ir buscar a Bio de uma Rota dado um id	8
Ir buscar todos os Lugares de uma Rota dado um id	8
Recurso Route Evaluations	9
Ir buscar todas as Avaliações de Rotas	9
Ir buscar uma Avaliação de Rota dado um id	9
Adicionar uma Avaliação de Rota	9
Apagar uma Avaliação de Rota dado um id	10
Recurso Spots	11
Ir buscar todos os Locais Turísticos	11
Ir buscar um Local dado um id	11
Adicionar um Local	11
Apagar um Local dado um id	12
Ir buscar a Média de avaliação dos Locais	12
Ir buscar a Média de avaliações de um Local dado um id	13
Ir buscar a Bio de um Local dado um id	13
Recurso Spot Evaluations	14
Ir buscar todas as Avaliações de Locais	14
Ir buscar uma Avaliação de Local dado um id	14
Adicionar uma Avaliação de Local	14
Apagar uma Avaliação de Local dado um id	15
Recurso Tag Types	16
Ir buscar todos os Tipos de Tags	16
Ir buscar um Tipo de Tag dado um id	16
Adicionar um Tipo de Tag	16
Apagar um Tipo Tag dado um id	17
Recurso Tag	18
Ir buscar todas as Tags	18
Ir buscar uma Tag dado um id	18

Adicionar uma Tag	18
Apagar uma Tag dado um id	19
Recurso Type Users	20
Ir buscar todos os Tipos de Utilizadores	20
Ir buscar um Tipo de Utilizador dado um id	20
Adicionar um Tipo de Utilizador	20
Apagar um Tipo de Utilizador dado um id	21
Recurso User Achievements	22
Ir buscar todos os Achievements de cada Utilizador	22
Ir buscar os Achievements de um Utilizador dado um id	22
Adicionar um Achievement ao Utilizador	22
Apagar um Achievement de Utilizador dado um id	23
Recurso Users	24
Ir buscar todos os Users	24
Ir buscar um User dado um id	24
Adicionar um User	24
Apagar um Utilizador dado um id	25

Recurso Achievements

Ir buscar todos os Achievements

Devolve todos os achievements na base dados

URL: /api/achievements (get)

Resultado:

```
[ { "id": 1, "acName": "1 Museu Visitado" },  
  { "id": 2, "acName": "5 Museus Visitados" },  
  { "id": 3, "acName": "10 Museus Visitados" },  
  ... ]
```

Ir buscar um Achievement dado um id

Devolve o achievement correspondente ao id

URL: /api/achievements/{id:[0-9]+} (get)

Parâmetros:

id - inteiro positivo que corresponde ao id do achievement a obter

Resultado:

```
{ "id": 2, "acName": "5 Museus Visitados" }
```

Erros:

404 (HttpStatus.NOT_FOUND): O achievement não foi encontrado

```
{  
  "timestamp": "2021-11-08T19:05:19.870+00:00",  
  "status": 404,  
  "error": "Not Found",  
  "message": "Achievement with id 200 not found.",  
  "path": "/api/achievements/200"  
}
```

Adicionar um Achievement

Recebe a informação do achievement e cria um novo

URL: /api/achievements (post)

Dados:

```
{ "acName": "Muito bom" }
```

Resultado:

```
{ "id": 16, "acName": "Muito bom" }
```

Erros:

500: Erro do servidor

```
{  
  "timestamp": "2021-11-08T19:31:33.878+00:00",
```

```
"status": 500,  
"error": "Internal Server Error",  
"message": "could not execute statement; SQL [n/a]; constraint [ac_name\" of  
relation \"achievements]; nested exception is  
org.hibernate.exception.ConstraintViolationException: could not execute statement",  
"path": "/api/achievements"  
}
```

Apagar um Achievement dado um id

Apaga o achievement correspondente ao id

URL: /api/achievements/{id:[0-9]+} (**delete**)

Parâmetros:

id - inteiro positivo que corresponde ao id do achievement a apagar

Resultado:

```
{  
  "message": "Deleted achievement with id 21",  
  "object": null  
}
```

Erros:

404 (`HttpStatus.NOT_FOUND`): O achievement não foi encontrado

```
{  
  "timestamp": "2021-11-09T16:21:22.628+00:00",  
  "status": 404,  
  "error": "Not Found",  
  "message": "Achievement with id 21 not found.",  
  "path": "/api/achievements/21"  
}
```

Recurso Routes

Ir buscar todas as Rotas

Devolve todas as rotas na base dados

URL: /api/routes (**get**)

Resultado:

```
[ { "id": 1, "rtName": "Compras", "rtBio": "Capitalismo no seu melhor", "rtDist": 4.0 },
  { "id": 2, "rtName": "Natureza", "rtBio": "Muitos animais agua e plantas, brother!!!",
    "rtDist": 10.0 },
  { "id": 3, "rtName": "Histórica", "rtBio": null, "rtDist": 3.0 },
  ... ]
```

Ir buscar uma Rota dado um id

Devolve a rota correspondente ao id

URL: /api/routes/{id:[0-9]+} (**get**)

Parâmetros:

id - inteiro positivo que corresponde ao id da rota a obter

Resultado:

```
{ "id": 2, "rtName": "Natureza", "rtBio": "Muitos animais agua e plantas, brother!!!",
  "rtDist": 10.0 }
```

Erros:

404 (**HttpStatus.NOT_FOUND**): A rota não foi encontrada

```
{
  "timestamp": "2021-11-09T15:53:46.785+00:00",
  "status": 404,
  "error": "Not Found",
  "message": "Route with id 50 not found.",
  "path": "/api/routes/50"
}
```

Adicionar uma Rota

Recebe as informações da rota e cria uma nova

URL: /api/routes (**post**)

Dados:

```
{ "rtName": "Arte Urbana",
  "rtBio": "Conheça e aprecie as obras de arte mais interessantes de Lisboa",
  "rtDist": "23.7" }
```

Resultado:

```
{ "id": 16, "acName": "Muito bom" }
```

Erros:
500: Erro do servidor

```
{
  "timestamp": "2021-11-09T16:05:05.720+00:00",
  "status": 500,
  "error": "Internal Server Error",
  "message": "could not execute statement; SQL [n/a]; constraint [ac_name\" of
relation \"achievements]; nested exception is
org.hibernate.exception.ConstraintViolationException: could not execute statement",
  "path": "/api/routes"
}
```

Apagar a rota dado um id

Apaga a rota correspondente ao id

URL: /api/routes/{id:[0-9]+} (**delete**)

Parâmetros:

id - inteiro positivo que corresponde ao id da rota a apagar

Resultado:

```
{
  "message": "Deleted route with id 7",
  "object": null
}
```

Erros:

404 (**HttpStatus.NOT_FOUND**): A rota não foi encontrada

```
{
  "timestamp": "2021-11-09T16:08:18.215+00:00",
  "status": 404,
  "error": "Not Found",
  "message": "Route with id 7 not found.",
  "path": "/api/routes/7"
}
```

Filtrar as Rotas de acordo com o nome e a distância

Apaga a rota correspondente ao id

URL: /api/routes/search?name={name}&distMin={distMin}&distMax={distMax} (**get**)

Parâmetros:

name - string que corresponde a parte do nome que se quer filtrar (valor padrão "")

distMin - double que corresponde a distância mínima a qual se pretende filtrar (valor padrão 0)

distMax - double que corresponde a distância máxima a qual se pretende filtrar (valor padrão MAX_VALUE)

Resultado:

```
[ { "id": 1, "rtName": "Compras", "rtBio": "Capitalismo no seu melhor", "rtDist": 4.0 },  
  { "id": 4, "rtName": "Popular", "rtBio": null, "rtDist": 2.0 } ]
```

Ir buscar a Média das avaliações das Rotas

Devolve o nome de cada rota e a média das notas dadas para cada um dos lugares que pertencem a ela, ordenados da maior para menor nota.

URL: /api/routes/avg (get)

Resultado:

```
[ { "id": 3, "rtName": "Histórica", "rtAvg": 4.571428571428571 },  
  { "id": 1, "rtName": "Compras", "rtAvg": 4.0 },  
  { "id": 4, "rtName": "Popular", "rtAvg": 3.5714285714285716 },  
  { "id": 2, "rtName": "Natureza", "rtAvg": 3.4285714285714284 } ]
```

Ir buscar a Bio de uma Rota dado um id

Devolve apenas a descrição da rota correspondente ao id

URL: /api/routes/bio/{id:[0-9]+} (get)

Parâmetros:

id - inteiro positivo que corresponde ao id da rota a obter os dados

Resultado:

```
[ "Muitos animais agua e plantas, brother!!!" ]
```

Ir buscar todos os Lugares de uma Rota dado um id

Devolve os lugares da rota correspondente ao id

URL: /api/routes/{id:[0-9]+}/spots (get)

Parâmetros:

id - inteiro positivo que corresponde ao id da rota a obter os dados

Resultado:

```
[ { "id": 12, "spName": "Mercado da Ribeira", "spLat": 38.70690732582062,  
    "spLong": -9.145907544730553, "spPrice": false, "spBio": "Biografia de teste 12" },  
  { "id": 13, "spName": "Mercado do Príncipe Real", "spLat": 38.716159660346214,  
    "spLong": -9.149212773566605, "spPrice": null, "spBio": "Biografia de teste 13" },  
  { "id": 14, "spName": "Mercado de Campo de Ourique", "spLat": 38.707173108334246,  
    "spLong": -9.143361173566847, "spPrice": null, "spBio": "Biografia de teste 14" },  
  ...]
```


Recurso Route Evaluations

Ir buscar todas as Avaliações de Rotas

Devolve todas as avaliações de rotas na base dados

URL: /api/routesEvaluations (get)

Resultado:

```
[ { "id": 1, "reRate": 5, "reComment": "Very good!", "reUsId": 2, "reRtId": 1 },  
  { "id": 2, "reRate": 3, "reComment": null, "reUsId": 3, "reRtId": 3 },  
  { "id": 3, "reRate": 2, "reComment": null, "reUsId": 4, "reRtId": 4 },  
  ... ]
```

Ir buscar uma Avaliação de Rota dado um id

Devolve a avaliação de rota correspondente ao id

URL: /api/routesEvaluations/{id:[0-9]+} (get)

Parâmetros:

id - inteiro positivo que corresponde ao id da avaliação de rota a obter

Resultado:

```
{ "id": 1, "reRate": 5, "reComment": "Very good!", "reUsId": 2, "reRtId": 1 }
```

Erros:

404 (HttpStatus.NOT_FOUND): A avaliação de rota não foi encontrada

```
{  
  "timestamp": "2021-11-09T15:50:03.915+00:00",  
  "status": 404,  
  "error": "Not Found",  
  "message": "Route Evaluation with id 200 not found.",  
  "path": "/api/routesEvaluations/200"  
}
```

Adicionar uma Avaliação de Rota

Recebe as informações da avaliação da rota e cria uma nova avaliação

URL: /api/routesEvaluations (post)

Dados:

```
{ "reRate": "3",  
  "reComment": "Gostei, mas cara.",  
  "reUsId": "2", "reRtId": "3"  
}
```

Resultado:

```
{ "id": 5, "reRate": 3, "reComment": "Gostei, mas cara.", "reUsId": 2, "reRtId": 3 }
```

Erros:
500: Erro do servidor

```
{
  "timestamp": "2021-11-09T16:39:55.513+00:00",
  "status": 500,
  "error": "Internal Server Error",
  "message": "not-null property references a null or transient value :
com.iade.ulide.models.RouteEvaluation.reRtId; nested exception is
org.hibernate.PropertyValueException: not-null property references a null or transient
value : com.iade.ulide.models.RouteEvaluation.reRtId",
  "path": "/api/routesEvaluations"
}
```

Apagar uma Avaliação de Rota dado um id

Apaga a avaliação de rota correspondente ao id

URL: /api/routesEvaluations/{id:[0-9]+} (**delete**)

Parâmetros:

id - inteiro positivo que corresponde ao id da avaliação de rota a apagar

Resultado:

```
{
  "message": "Deleted route evaluation with id 5",
  "object": null
}
```

Erros:

404 (**HttpStatus.NOT_FOUND**): A avaliação de rota não foi encontrada

```
{
  "timestamp": "2021-11-09T16:57:13.593+00:00",
  "status": 404,
  "error": "Not Found",
  "message": "Route Evaluation with id 6 not found.",
  "path": "/api/routesEvaluations/6"
}
```

Recurso Spots

Ir buscar todos os Locais Turísticos

Devolve todos locais na base dados

URL: /api/spots (get)

Resultado:

```
[ { "id": 1, "spName": "Torre de Belém", "spLat": 38.69175116131192,
  "spLong": -9.215966573567318, "spPrice": false, "spBio": null },
  { "id": 2, "spName": "Mosteiro dos Jerónimos", "spLat": 38.69806671988026,
  "spLong": -9.206671715894556, "spPrice": false, "spBio": null },
  { "id": 3, "spName": "Castelo de São Jorge", "spLat": 38.71405986610344,
  "spLong": -9.133454744730386, "spPrice": true, "spBio": null },
  ... ]
```

Ir buscar um Local dado um id

Devolve o local correspondente ao id

URL: /api/spots/{id:[0-9]+} (get)

Parâmetros:

id - inteiro positivo que corresponde ao id do local a obter

Resultado:

```
{ "id": 1, "spName": "Torre de Belém", "spLat": 38.69175116131192,
  "spLong": -9.215966573567318, "spPrice": false, "spBio": null }
```

Erros:

404 (HttpStatus.NOT_FOUND): O local não foi encontrado

```
{
  "timestamp": "2021-11-09T17:19:04.918+00:00",
  "status": 404,
  "error": "Not Found",
  "message": "Spot with id 25 not found.",
  "path": "/api/spots/25"
}
```

Adicionar um Local

Recebe as informações do local e cria um novo

URL: /api/spots (post)

Dados:

```
{ "spName": "Museu", "spLat": "38.709317720175115",
  "spLong": "-9.136401460075637", "spPrice": "False",
  "spBio": "Museu do Design e da Moda" }
```

Resultado:

```
{ "id": 17, "spName": "Mude", "spLat": 38.709317720175115,
  "spLong": -9.136401460075637, "spPrice": false, "spBio": "Museu do Design e da Moda"
}
```

Erros:

500: Erro do servidor

```
{
  "timestamp": "2021-11-09T17:27:39.907+00:00",
  "status": 500,
  "error": "Internal Server Error",
  "message": "not-null property references a null or transient value :
com.iade.ulide.models.Spot.spLat; nested exception is
org.hibernate.PropertyValueException: not-null property references a null or transient
value : com.iade.ulide.models.Spot.splat",
  "path": "/api/spots"
}
```

Apagar um Local dado um id

Apaga local correspondente ao id

URL: /api/spots/{id:[0-9]+} (delete)

Parâmetros:

id - inteiro positivo que corresponde ao id do local a apagar

Resultado:

```
{
  "message": "Deleted spot with id 17",
  "object": null
}
```

Erros:

404 (HttpStatus.NOT_FOUND): O local não foi encontrado

```
{
  "timestamp": "2021-11-09T17:29:38.286+00:00",
  "status": 404,
  "error": "Not Found",
  "message": "Spot with id 17 not found.",
  "path": "/api/spots/17"
}
```

Ir buscar a Média de avaliação dos Locais

Devolve o nome de cada local e a média das notas dadas a ele, ordenados da maior para menor nota.

URL: /api/spots/avg (get)

Resultado:

```
[ { "id": 6, "spName": "Elevador da Glória", "spAvg": 5.0 },  
  { "id": 12, "spName": "Mercado da Ribeira", "spAvg": 4.666666666666667 },  
  { "id": 13, "spName": "Mercado do Príncipe Real", "spAvg": 4.333333333333333 },  
  ... ]
```

Ir buscar a Média de avaliações de um Local dado um id

Devolve a média de um local correspondente ao id

URL: /api/spots/{id:[0-9]+}/average (get)

Parâmetros:

id - inteiro positivo que corresponde ao id do local a obter os dados

Resultado:

```
[ 3.5000000000000000 ]
```

Ir buscar a Bio de um Local dado um id

Devolve apenas a descrição do local correspondente ao id

URL: /api/spots/bio/{id:[0-9]+} (get)

Parâmetros:

id - inteiro positivo que corresponde ao id do local a obter os dados

Resultado:

```
[ "Biografia de teste 02" ]
```

Recurso Spot Evaluations

Ir buscar todas as Avaliações de Locais

Devolve todas as avaliações de locais na base dados

URL: /api/spotsEvaluations (get)

Resultado:

```
[ { "id": 17, "seRate": 5, "seComment": "Fantastic", "seUsId": 2, "seSpId": 12 },  
  { "id": 18, "seRate": 4, "seComment": "Mui bueno", "seUsId": 2, "seSpId": 13 },  
  { "id": 19, "seRate": 1, "seComment": "Not for me", "seUsId": 2, "seSpId": 14 },  
  ... ]
```

Ir buscar uma Avaliação de Local dado um id

Devolve a avaliação de local correspondente ao id

URL: /api/spotsEvaluations/{id:[0-9]+} (get)

Parâmetros:

id - inteiro positivo que corresponde ao id da avaliação de local a obter

Resultado:

```
{ "id": 20, "seRate": 5, "seComment": "Im a tea pot", "seUsId": 2, "seSpId": 15 }
```

Erros:

404 (HttpStatus.NOT_FOUND): A avaliação de rota não foi encontrada

```
{  
  "timestamp": "2021-11-09T17:38:02.997+00:00",  
  "status": 404,  
  "error": "Not Found",  
  "message": "Spot Evaluation with id 1 not found.",  
  "path": "/api/spotsEvaluations/1"  
}
```

Adicionar uma Avaliação de Local

Recebe as informações da avaliação de local e cria uma nova avaliação

URL: /api/spotsEvaluations (post)

Dados:

```
{ "seRate": "4",  
  "seComment": "Ótimo lugar.",  
  "seUsId": "3", "seSpId": "15"  
}
```

Resultado:

```
{ "id": 32, "seRate": 4, "seComment": "Ótimo lugar.", "seUsId": 3, "seSpId": 15 }
```

Erros:
500: Erro do servidor

```
{
  "timestamp": "2021-11-09T17:45:54.620+00:00",
  "status": 500,
  "error": "Internal Server Error",
  "message": "not-null property references a null or transient value :
com.iade.ulide.models.SpotEvaluation.seSpId; nested exception is
org.hibernate.PropertyValueException: not-null property references a null or transient
value : com.iade.ulide.models.SpotEvaluation.seSpId",
  "path": "/api/spotsEvaluations"
}
```

Apagar uma Avaliação de Local dado um id

Apaga a avaliação de local correspondente ao id

URL: /api/spotsEvaluations/{id:[0-9]+} (delete)

Parâmetros:

id - inteiro positivo que corresponde ao id da avaliação de local a apagar

Resultado:

```
{
  "message": "Deleted spot evaluation with id 32",
  "object": null
}
```

Erros:

404 (HttpStatus.NOT_FOUND): A avaliação de rota não foi encontrada

```
{
  "timestamp": "2021-11-09T17:49:38.930+00:00",
  "status": 404,
  "error": "Not Found",
  "message": "Spot Evaluation with id 32 not found.",
  "path": "/api/spotsEvaluations/32"
}
```

Recurso Tag Types

Ir buscar todos os Tipos de Tags

Devolve todos os tipos de tags na base dados

URL: /api/tagTypes (get)

Resultado:

```
[ { "id": 1, "ttName": "User" },  
  { "id": 2, "ttName": "System" } ]
```

Ir buscar um Tipo de Tag dado um id

Devolve o tipo de tag correspondente ao id

URL: /api/tagTypes/{id:[0-9]+} (get)

Parâmetros:

id - inteiro positivo que corresponde ao id do tipo de tag a obter

Resultado:

```
{ "id": 1, "ttName": "User" }
```

Erros:

404 (HttpStatus.NOT_FOUND): O tipo de tag não foi encontrado

```
{  
  "timestamp": "2021-11-09T17:55:03.117+00:00",  
  "status": 404,  
  "error": "Not Found",  
  "message": "TagType with id 3 not found.",  
  "path": "/api/tagTypes/3"  
}
```

Adicionar um Tipo de Tag

Recebe a informação do tipo de tag e cria um novo

URL: /api/tagTypes (post)

Dados:

```
{ "ttName": "Sponsor" }
```

Resultado:

```
{ "id": 3, "ttName": "Sponsor" }
```

Erros:

500: Erro do servidor

```
{  
  "timestamp": "2021-11-09T17:58:40.436+00:00",  
  "status": 500,  
}
```



```
"error": "Internal Server Error",
"message": "not-null property references a null or transient value :
com.iade.ulide.models.TagType.ttName; nested exception is
org.hibernate.PropertyValueException: not-null property references a null or transient
value : com.iade.ulide.models.TagType.ttName",
"path": "/api/tagTypes"
}
```

Apagar um Tipo Tag dado um id

Apaga o tipo de tag correspondente ao id

URL: /api/tagTypes/{id:[0-9]+} (**delete**)

Parâmetros:

id - inteiro positivo que corresponde ao id do tipo de tag a apagar

Resultado:

```
{
  "message": "Deleted TagType with id 4",
  "object": null
}
```

Erros:

404 (**HttpStatus.NOT_FOUND**): O tipo de tag não foi encontrado

```
{
  "timestamp": "2021-11-09T18:06:32.941+00:00",
  "status": 404,
  "error": "Not Found",
  "message": "TagType with id 4 not found.",
  "path": "/api/tagTypes/4"
}
```

Recurso Tag

Ir buscar todas as Tags

Devolve todas tags na base dados

URL: /api/tags (get)

Resultado:

```
[ { "id": 1, "tgName": "Mercado", "tgTtId": 1 },  
  { "id": 2, "tgName": "Monumento", "tgTtId": 1 },  
  { "id": 3, "tgName": "Natureza", "tgTtId": 1 },  
  { "id": 4, "tgName": "Museu", "tgTtId": 1 },  
  ... ]
```

Ir buscar uma Tag dado um id

Devolve a tag correspondente ao id

URL: /api/tags/{id:[0-9]+} (get)

Parâmetros:

id - inteiro positivo que corresponde ao id da tag a obter

Resultado:

```
{ "id": 5, "tgName": "Desporto", "tgTtId": 1 }
```

Erros:

404 (HttpStatus.NOT_FOUND): A tag não foi encontrada

```
{  
  "timestamp": "2021-11-09T18:16:42.774+00:00",  
  "status": 404,  
  "error": "Not Found",  
  "message": "Tag with id 25 not found.",  
  "path": "/api/tags/25"  
}
```

Adicionar uma Tag

Recebe a informação da tag e cria uma nova

URL: /api/tags (post)

Dados:

```
{ "tgName": "Arte", "tgTtId": "1" }
```

Resultado:

```
{ "id": 15, "tgName": "Arte", "tgTtId": 1 }
```

Erros:

500: Erro do servidor

```
{
  "timestamp": "2021-11-09T18:21:59.701+00:00",
  "status": 500,
  "error": "Internal Server Error",
  "message": "not-null property references a null or transient value :
com.iade.ulide.models.Tag.tgName; nested exception is
org.hibernate.PropertyValueException: not-null property references a null or transient
value : com.iade.ulide.models.Tag.tgName",
  "path": "/api/tags"
}
```

Apagar uma Tag dado um id

Apaga a tag correspondente ao id

URL: /api/tags/{id:[0-9]+} (**delete**)

Parâmetros:

id - inteiro positivo que corresponde ao id da tag a apagar

Resultado:

```
{
  "message": "Deleted tag with id 4",
  "object": null
}
```

Erros:

404 (**HttpStatus.NOT_FOUND**): A tag não foi encontrada

```
{
  "timestamp": "2021-11-09T18:23:12.281+00:00",
  "status": 404,
  "error": "Not Found",
  "message": "Tag with id 4 not found.",
  "path": "/api/tags/4"
}
```

Recurso Type Users

Ir buscar todos os Tipos de Utilizadores

Devolve todos os tipos de utilizadores na base dados

URL: /api/typeUsers (get)

Resultado:

```
[ { "id": 1, "tuName": "Super" }, { "id": 2, "tuName": "Nivel 1" },  
  { "id": 3, "tuName": "Nivel 2" }, { "id": 4, "tuName": "Nivel 3" },  
  { "id": 5, "tuName": "Nivel 4" } ]
```

Ir buscar um Tipo de Utilizador dado um id

Devolve o tipo de utilizador correspondente ao id

URL: /api/typeUsers/{id:[0-9]+} (get)

Parâmetros:

id - inteiro positivo que corresponde ao id do tipo de usuário a obter

Resultado:

```
{ "id": 2, "tuName": "Nivel 1" }
```

Erros:

404 (HttpStatus.NOT_FOUND): O tipo de utilizador não foi encontrado

```
{  
  "timestamp": "2021-11-09T18:44:31.871+00:00",  
  "status": 404,  
  "error": "Not Found",  
  "message": "User with id 25 not found.",  
  "path": "/api/typeUsers/25"  
}
```

Adicionar um Tipo de Utilizador

Recebe a informação do tipo de utilizador e cria um novo

URL: /api/typeUsers (post)

Dados:

```
{ "tuName": "Nivel 5" }
```

Resultado:

```
{ "id": 6, "tuName": "Nivel 5" }
```

Erros:

500: Erro do servidor

```
{  
  "timestamp": "2021-11-09T18:48:00.901+00:00",
```

```
"status": 500,  
"error": "Internal Server Error",  
"message": "not-null property references a null or transient value :  
com.iade.ulide.models.TagType.ttName; nested exception is  
org.hibernate.PropertyValueException: not-null property references a null or transient  
value : com.iade.ulide.models.TypeUser",  
"path": "/api/typeUsers"  
}
```

Apagar um Tipo de Utilizador dado um id

Apaga o tipo de utilizador correspondente ao id

URL: /api/typeUsers/{id:[0-9]+} (delete)

Parâmetros:

id - inteiro positivo que corresponde ao id do tipo de utilizador a apagar

Resultado:

```
{  
  "message": "Deleted type user with id 7",  
  "object": null  
}
```

Erros:

404 ([HttpStatus.NOT_FOUND](#)): O tipo de utilizador não foi encontrado

```
{  
  "timestamp": "2021-11-09T18:54:18.018+00:00",  
  "status": 404,  
  "error": "Not Found",  
  "message": "Type user with id 7 not found.",  
  "path": "/api/typeUsers/7"  
}
```

Recurso User Achievements

Ir buscar todos os Achievements de cada Utilizador

Devolve todos os achievements de utilizadores na base dados

URL: /api/userAchievements (get)

Resultado:

```
[ { "id": 11, "uaDate": "2021-11-19", "uaUsId": 3, "uaAcId": 5 },
  { "id": 13, "uaDate": "2021-11-19", "uaUsId": 3, "uaAcId": 11 },
  { "id": 8, "uaDate": "2021-11-20", "uaUsId": 2, "uaAcId": 3 },
  ... ]
```

Ir buscar os Achievements de um Utilizador dado um id

Devolve os achievements do utilizador correspondente ao id

URL: /api/userAchievements/{id:[0-9]+} (get)

Parâmetros:

id - inteiro positivo que corresponde ao id dos achievements do utilizador a obter

Resultado:

```
{ "id": 13, "uaDate": "2021-11-19", "uaUsId": 3, "uaAcId": 11 }
```

Erros:

404 (HttpStatus.NOT_FOUND): Os achievements do utilizador não foram encontrados

```
{
  "timestamp": "2021-11-09T19:28:24.197+00:00",
  "status": 404,
  "error": "Not Found",
  "message": "User achievement with id 25 not found.",
  "path": "/api/userAchievements/25"
}
```

Adicionar um Achievement ao Utilizador

Recebe as informações do achievement do utilizador e cria um novo

URL: /api/userAchievements (post)

Dados:

```
{ "uaDate": "2021-06-12", "uaUsId": "6", "uaAcId": "1" }
```

Resultado:

```
{ "id": 16, "uaDate": "2021-06-12", "uaUsId": 6, "uaAcId": 1 }
```

Erros:

500: Erro do servidor

```
{
  "timestamp": "2021-11-09T19:41:23.949+00:00",
```

```
{
  "status": 500,
  "error": "Internal Server Error",
  "message": "not-null property references a null or transient value :
com.iade.ulide.models.UserAchievement.uaAcId; nested exception is
org.hibernate.PropertyValueException: not-null property references a null or transient
value : com.iade.ulide.models.UserAchievement.uaAcId",
  "path": "/api/userAchievements"
}
```

Apagar um Achievement de Utilizador dado um id

Apaga o utilizador correspondente ao id

URL: /api/userAchievements/{id:[0-9]+} (delete)

Parâmetros:

id - inteiro positivo que corresponde ao id do achievement do utilizador a apagar

Resultado:

```
{
  "message": "Deleted user achievement with id 16",
  "object": null
}
```

Erros:

404 (HttpStatus.NOT_FOUND): O achievement do utilizador não foi encontrado

```
{
  "timestamp": "2021-11-09T19:47:15.614+00:00",
  "status": 404,
  "error": "Not Found",
  "message": "User achievement with id 16 not found.",
  "path": "/api/userAchievements/16"
}
```

Recurso Users

Ir buscar todos os Users

Devolve todos os utilizadores na base dados

URL: /api/users (get)

Resultado:

```
[ { "id": 1, "usName": "Super Ulide", "usBdate": "2021-11-07", "usGender": "M",  
  "usEmail": null, "usCountry": null, "usBio": null, "usDist": 0 },  
  { "id": 2, "usName": "Caetano Feliciano", "usBdate": "1997-08-30", "usGender": "M",  
    "usEmail": "catefe@gmail.com", "usCountry": "Germany",  
    "usBio": "I like very big horses", "usDist": 0 },  
  ... ]
```

Ir buscar um User dado um id

Devolve o utilizador correspondente ao id

URL: /api/users/{id:[0-9]+} (get)

Parâmetros:

id - inteiro positivo que corresponde ao id do utilizador a obter

Resultado:

```
{ "id": 3, "usName": "Teófilo Vale", "usBdate": "2003-05-01", "usGender": "M",  
  "usEmail": "Tevale12@hotmail.com", "usCountry": "United States",  
  "usBio": "Hamburger with fries and coke", "usDist": 0 }
```

Erros:

404 (HttpStatus.NOT_FOUND): O utilizador não foi encontrado

```
{  
  "timestamp": "2021-11-08T19:48:34.105+00:00",  
  "status": 404,  
  "error": "Not Found",  
  "message": "User with id 100 not found.",  
  "path": "/api/users/100"  
}
```

Adicionar um User

Recebe a informação do utilizador e cria um novo

URL: /api/users (post)

Dados:

```
{ "usName": "Roberto Almires", "usBdate": "1995-04-12", "usGender": "M",  
  "usEmail": "rob.almires@email.com", "usCountry": "Italy"  
}
```


Resultado:

```
{  "id": 13, "usName": "Roberto Almires", "usBdate": "1995-04-12", "usGender": "M",
  "usEmail": "rob.almires@email.com", "usCountry": "Italy", "usBio": null,
  "usDist": null
}
```

Erros:

500: Erro do servidor

```
{
  "timestamp": "2021-11-08T20:05:37.302+00:00",
  "status": 500,
  "error": "Internal Server Error",
  "message": "not-null property references a null or transient value :
com.iade.ulide.models.User.usBdate; nested exception is
org.hibernate.PropertyValueException: not-null property references a null or transient
value : com.iade.ulide.models.User.usBdate",
  "path": "/api/users"
}
```

Apagar um Utilizador dado um id

Apaga o utilizador correspondente ao id

URL: /api/users/{id:[0-9]+} (**delete**)

Parâmetros:

id - inteiro positivo que corresponde ao id do utilizador a apagar

Resultado:

```
{
  "message": "Deleted user with id 12",
  "object": null
}
```

Erros:

404 (**HttpStatus.NOT_FOUND**): O utilizador não foi encontrado

```
{
  "timestamp": "2021-11-09T19:50:19.629+00:00",
  "status": 404,
  "error": "Not Found",
  "message": "User with id 200 not found.",
  "path": "/api/users/200"
}
```