**DOCUMENTAÇÃO REST API**

**VERSÃO 1**

1 - Recurso Locais (os métodos do recurso “Local”, engloba os locais de lazer (cafés, bares e restaurantes), e os locais de “aprendizagem” (livrarias e bibliotecas))

| Objetivo: Exibir todos os locais |
| --- |

| Descrição: Devolve uma lista de todos os cafés, bares e restaurantes (na activity de locais de lazer). Devolve uma lista de livrarias e bibliotecas (na activity de locais de “aprendizagem”) |
| --- |

| URL: /api/locals **(GET - Locais de Lazer)**  /api/locals **(GET - Locais de “aprendizagem”)** |
| --- |

| Dados: |
| --- |

| Resultado:  [  {  "id": 3,  "name": "Bar Lemos",  "endereco": "Rua Joao Direitinho, Nº 13",  "distancia": 500,  "categoria": 1  },  {  "id": 4,  "name": "Livraria Eca de Queiroz",  "endereco": "Rua Lopes Santini, Nº 24",  "distancia": 2310,  "categoria": 4  },  {  "id": 5,  "name": "Cafe Dom Joaquim",  "endereco": "Avenida da Estrela, Nº 4B",  "distancia": 253,  "categoria": 2  },  {  "id": 6,  "name": "Cafe O Dante",  "endereco": "Rua Neuracia da Alemanha, Nº 12",  "distancia": 1256,  "categoria": 1  },  {  "id": 7,  "name": "Livraria Dom Joao I",  "endereco": "Rua do Rossio, Nº 22",  "distancia": 1156,  "categoria": 4  }  ] |
| --- |

| Objetivo: Filtrar locais pelo nome |
| --- |

| Descrição: Devolver cafés, bares, restaurantes, livrarias e bibliotecas num raio de 15 kilómetros, baseando-se na pesquisa por nome do local. |
| --- |

| URL: /api/locals/searchbyname/{place\_name} **(GET - Locais de Lazer)**  /api/locals/searchbyname/{place\_name} **(GET - Locais de “aprendizagem”)** |
| --- |

| Parâmetros: name - Nome do local (pesquisa por “containing”) |
| --- |

| Dados: |
| --- |

| Resultado:  [  {  "id": 3,  "name": "Bar Lemos",  "endereco": "Rua Joao Direitinho, Nº 13",  "distancia": 500,  "categoria": 1  }  ] |
| --- |

| Objetivo: Obter as informações do local selecionado |
| --- |

| Descrição: Obter a distância, o endereço, nome e categoria de um local após a pesquisa por ID (seleção do local) |
| --- |

| URL: /api/locals/getinfo/{id} **(GET)**  /api/locals/getinfo/{id} **(GET)** |
| --- |

| Parâmetros: id - Devolve as informações de um local selecionado através do ID. O “id” identificará qual o local selecionado e posteriormente devolver as informações relativas a esse mesmo id (local). |
| --- |

| Dados: |
| --- |

| Resultado:  {  "id": 4,  "name": "Livraria Eca de Queiroz",  "endereco": "Rua Lopes Santini, Nº 24",  "distancia": 2310,  "categoria": 4  } |
| --- |

| Erro:  {  "timestamp": "2021-11-19T16:59:49.936+00:00",  "status": 404,  "error": "Not Found",  "message": "Localwith ID6not found",  "path": "/api/locals/getinfo/6"  } |
| --- |

| Objetivo: Filtragem de locais por categoria |
| --- |

| Descrição: Filtrar locais da lista por categoria |
| --- |

| URL: /api/locals/searchbycategory/{place\_categoria} |
| --- |

| Parâmetros: place\_categoria - id correspondente a uma categoria |
| --- |

| Dados: |
| --- |

| Resultado:  [  {  "id": 3,  "name": "Bar Lemos",  "endereco": "Rua Joao Direitinho, Nº 13",  "distancia": 500,  "categoria": 1  },  {  "id": 6,  "name": "Cafe O Dante",  "endereco": "Rua Neuracia da Alemanha, Nº 12",  "distancia": 1256,  "categoria": 1  }  ] |
| --- |

| Objetivo: Filtrar os locais mais próximos do utilizador |
| --- |

| Descrição: Filtrar os locais mais próximos do utilizador, de acordo com a ordem crescente de distâncias (previamente calculadas através da latitude e longitude dos locais e do utilizador) |
| --- |

| URL: /api/locals/closestplaces |
| --- |

| Parâmetros: |
| --- |

| Dados: |
| --- |

| Resultado:  [  {  "id": 5,  "name": "Cafe Dom Joaquim",  "endereco": "Avenida da Estrela, Nº 4B",  "distancia": 253,  "categoria": 2  },  {  "id": 3,  "name": "Bar Lemos",  "endereco": "Rua Joao Direitinho, Nº 13",  "distancia": 500,  "categoria": 1  },  {  "id": 7,  "name": "Livraria Dom Joao I",  "endereco": "Rua do Rossio, Nº 22",  "distancia": 1156,  "categoria": 4  },  {  "id": 6,  "name": "Cafe O Dante",  "endereco": "Rua Neuracia da Alemanha, Nº 12",  "distancia": 1256,  "categoria": 1  },  {  "id": 4,  "name": "Livraria Eca de Queiroz",  "endereco": "Rua Lopes Santini, Nº 24",  "distancia": 2310,  "categoria": 4  }  ] |
| --- |

3 - Recurso Tasks

| Objetivo: Exibir todas as tarefas do utilizador |
| --- |

| Descrição: Obter e exibir todas as tarefas que pertençam a um utilizador |
| --- |

| URL: /api/tasks/gettarefas/{user\_task\_id} **(GET)** |
| --- |

| Parâmetros: user\_task\_id - id do utilizador correspondente a uma tarefa |
| --- |

| Dados: |
| --- |

| Resultado:  [  {  "prioridade": "Urgent",  "userTaskId": 8,  "taskId": 4,  "taskTitle": "Go to mall",  "taskType": **null**,  "taskDesc": "Go to the mall tu buy clothes"  },  {  "prioridade": "Low",  "userTaskId": 8,  "taskId": 32,  "taskTitle": "Uuu",  "taskType": **null**,  "taskDesc": "bbbb"  }  ] |
| --- |

| Objetivo: Adicionar uma tarefa |
| --- |

| Descrição: Permite ao utilizador adicionar tarefas á sua *“To-Do List”*. |
| --- |

| URL: /api/tasks/createtask **(POST)** |
| --- |

| Dados:  {  "title": "Go to cinema",  "description": "Go watch Spiderman on saturday",  "date": "2017-01-10",  "usertaskid": 3,  "priority": 1,  "type": 1  } |
| --- |

| Resultado:  {  "id": 34,  "title": "Go to cinema",  "description": "Go watch Spiderman on Saturday",  "date": "2017-01-10",  "usertaskid": 3,  "priority": 1,  "type": 1  } |
| --- |

| Erro:  {  "timestamp": "2021-11-20T00:48:26.944+00:00",  "status": 500,  "error": "Internal Server Error",  "message": "could not execute statement; SQL [n/a]; constraint [fk\_usertaskid]; nested exception is org.hibernate.exception.ConstraintViolationException: could not execute statement",  "path": "/api/tasks/createtask"  } |
| --- |

| Objetivo: Remover uma tarefa. |
| --- |

| Descrição: Permite ao utilizador remover tarefas da sua *“To-Do List”*, selecionando a tarefa que deseja remover. |
| --- |

| URL: /api/busybrain/tasks/deletetask/{task\_id} **(DELETE)** |
| --- |

| Parâmetros: id - Identifica a tarefa a ser removida (após ser selecionada) |
| --- |

| Dados: |
| --- |

| Resultado:  \*Nota: Resultado será uma response. |
| --- |

| Erro: |
| --- |

| Objetivo: Exibir informações de uma tarefa |
| --- |

| Descrição: Exibir titulo e descrição de uma tarefa quando esta é selecionada |
| --- |

| URL: /api/busybrain/tarefas/getinfo/{task\_id} **(GET)** |
| --- |

| Parâmetros: id - Identifica a tarefa selecionada, permitindo exibir as suas informações. |
| --- |

| Dados: |
| --- |

| Resultado:  {  "id": 33,  "title": "Ajaja",  "description": "ajajja",  "date": "2020-01-04",  "usertaskid": 15,  "priority": 2,  "type": 1  } |
| --- |

| Erro:  {  "timestamp": "2021-11-20T01:17:38.682+00:00",  "status": 404,  "error": "Not Found",  "message": "Tarefawith ID12not found",  "path": "/api/tasks/getinfo/12"  } |
| --- |

| Objetivo: Filtrar as tarefas por prioridade |
| --- |

| Descrição: Filtrar as tarefas de um utilizador por prioridade (“Low”, “Medium”, “High” ou “Urgent”) |
| --- |

| URL:/filterbypriority/{user\_task\_id}/{task\_priority\_id} (GET) |
| --- |

| Parâmetros: task\_priority\_id -> id da prioridade da tarefa | user\_task\_id -> filtragem por prioridade das tarefas de um determinado utilizador. |
| --- |

| Dados: |
| --- |

| Resultado:  {  "prioridade": "Low",  "userTaskId": 8,  "taskId": 32,  "taskTitle": "Uuu",  "taskType": **null**,  "taskDesc": "bbbb"  } |
| --- |

| Erro: |
| --- |

| Objetivo: Filtrar tarefas pelo nome |
| --- |

| Descrição: Filtrar tarefas de um utilizador através de uma pesquisa por nome (não se encontra na app) |
| --- |

| URL: /api/tasks/searchtitle/{task\_title} **(GET)** |
| --- |

| Parâmetros: title - um nome que possivelmente esteja contido no titulo de uma tarefa |
| --- |

| Dados: |
| --- |

| Resultado:  [  {  "id": 7,  "title": "Go to cinema",  "description": "Go watch Spiderman on saturday",  "date": "2017-01-10T00:00:00.000+00:00",  "usertaskid": 1,  "priority": 1,  "type": 1  }  ] |
| --- |

| Erro:  [] |
| --- |

4 - Websites

| Objetivo: Obter os websites bloqueados |
| --- |

| Descrição: Filtragem de websites bloqueados pelo utilizador |
| --- |

| URL: /api/websites/getwebsites/blocked **(GET)** |
| --- |

| Resultado:  [  {  "bloqueamentoid": 5,  "utilizadorid": 1,  "status": **true**,  "id": 1,  "domain": "facebook.com"  },  {  "bloqueamentoid": 6,  "utilizadorid": 4,  "status": **true**,  "id": 2,  "domain": "reddit.com"  }  ] |
| --- |

| Objetivo: Obter os websites desbloqueados |
| --- |

| Descrição: Filtragem de websites desbloqueados pelo utilizador |
| --- |

| URL: /api/websites/getwebsites/unlocked **(GET)** |
| --- |

| Resultado:  []  \*Nota: O resultado é “[]”, pois na BD não existem websites com blocked\_status = 0 (desbloqueados) |
| --- |

| Erro: |
| --- |

5 - Recurso Grupo

| Objetivo: Criar um grupo |
| --- |

| Descrição: Ao criar uma tarefa, o utilizador poderá criar um grupo para essa tarefa |
| --- |

| URL: /api/grupos/creategroup **(POST)** |
| --- |

| Resultado:  {  "id": 5,  "name": "Go to cinema",  "description": "Go watch Spiderman on saturday",  "tarefaid": 7  } |
| --- |

| Objetivo: Apagar um grupo |
| --- |

| Descrição: Apagar um grupo criado para uma tarefa |
| --- |

| URL: /api/grupos/deletegroup/{group\_id} **(DELETE)** |
| --- |

| Resultado:  \*Nota: o resultado não exibe quaisquer JSON. |
| --- |

| Objetivo: Obter todos os grupos de um utilizador |
| --- |

| Descrição: Obter os grupos criados por um utilizador, através do seu id |
| --- |

| URL: /api/grupos/getgroups/{user\_task\_id} **(GET)** |
| --- |

| Resultado:  [  {  "groupId": 6,  "userTaskId": 8,  "taskTitle": "Go to mall",  "groupDesc": "Group to buy stuff in the mall",  "groupName": "Shopping group"  }  ] |
| --- |

| Objetivo: Obter informações de um grupo através do ID |
| --- |

| Descrição: Obter informações de um grupo através do ID do grupo |
| --- |

| URL: /api/grupos/{group\_id} **(GET)** |
| --- |

| Dados: group\_id -> Identificador unico de um grupo criado na app |
| --- |

| Resultado:  [  {  "id": 6,  "name": "Shopping group",  "description": "Group to buy stuff in the mall",  "tarefaid": 4  }  ] |
| --- |

| Objetivo: Obter todos os participantes de um grupo |
| --- |

| Descrição: Obter uma lista de todos os participantes de um grupo |
| --- |

| URL: /api/participantesgrupos/getparticipants/{group\_id} **(GET)** |
| --- |

| Dados: group\_id -> Identificador unico de um grupo criado na app |
| --- |

| Resultado:  [  {  "groupId": 6,  "userName": "johndoe",  "groupDesc": "Group to buy stuff in the mall",  "userId": 1,  "groupName": "Shopping group"  },  {  "groupId": 6,  "userName": "marydoe",  "groupDesc": "Group to buy stuff in the mall",  "userId": 2,  "groupName": "Shopping group"  },  {  "groupId": 6,  "userName": "hugoferreira22",  "groupDesc": "Group to buy stuff in the mall",  "userId": 8,  "groupName": "Shopping group"  }  ] |
| --- |

| Objetivo: Adicionar um participante |
| --- |

| Descrição: Adicionar um participante a um grupo (completo durante o update da aplicação) |
| --- |

| URL: /api/participantesgrupos/addparticipant/{group\_id} **(POST)** |
| --- |

| Dados: group\_id -> Identificador unico para cada grupo criado na app |
| --- |

| Resultado:  {  "groupId": 6,  "userName": "ursula",  "groupDesc": "Group to buy stuff in the mall",  "userId": 15,  "groupName": "Shopping group"  } |
| --- |

| Objetivo: Remover um participante |
| --- |

| Descrição: Remover um participante de um grupo (completo mas não “dinamizado” na app - alteração a ser realizada no update da aplicação) |
| --- |

| URL: /api/participantesgrupos/deleteparticipant/{user\_id\_tarefa} **(DELETE)** |
| --- |

| Dados: user\_id\_tarefa -> ID do participante naquela tarefa. |
| --- |

| Resultado:  \*Nota: o resultado não possui JSON, pois trata-se de um método DELETE sem a “utilização” de JSON |
| --- |

**MÉTODOS PARA O UPDATE**

| Objetivo: Criar um convivio para um grupo |
| --- |

| Descricao: Criar um convivio para um grupo existente |
| --- |

| URL: /api/convivios/createconvivio **(POST)** |
| --- |

| Parâmetros: |
| --- |

| Dados: |
| --- |

| Resultado: |
| --- |

| Erro: |
| --- |

| Objetivo: Adicionar participantes a um grupo |
| --- |

| Descrição: Adicionar participantes a um grupo de uma tarefa, pesquisando pelo username do participante a ser adicionado |
| --- |

| Resultado: |
| --- |

| Erro: |
| --- |

| Objetivo: Remover participante do grupo |
| --- |

| Descrição: Remover participante do grupo de uma tarefa |
| --- |

| URL: |
| --- |

| Resultado: |
| --- |

| Erro: |
| --- |

| Objetivo: Exibir lista de participantes do grupo de uma tarefa |
| --- |

| Descrição: Exibir os participantes do grupo de uma tarefa |
| --- |

| URL: |
| --- |

| Resultado: |
| --- |

| Erro: |
| --- |

**OUTROS MÉTODOS (UTILIZADOR)**

| Objetivo: Obter todos os utilizadores |
| --- |

| Descrição: Obter todos os utilizadores registados na app |
| --- |

| URL:/api/users **(GET)** |
| --- |

| Resultado:  **[**  **{**  **"id": 1,**  **"username": "johndoe",**  **"password": "johndoeoriginal",**  **"email": "jdoe@gmail.com"**  **},**  **{**  **"id": 2,**  **"username": "marydoe",**  **"password": "marydoeoriginal",**  **"email": "mdoe@gmail.com"**  **},**  **{**  **"id": 3,**  **"username": "laurendoe",**  **"password": "laudoeoriginal",**  **"email": "lauren12doe@gmail.com"**  **}**  **]** |
| --- |

| Erro: |
| --- |

| Objetivo: Registar utilizador |
| --- |

| Descrição: Registar um novo utilizador na app |
| --- |

| URL:/api/users/createuser **(POST)** |
| --- |

| Dados:  **{**  **"username": "joaofepas12",**  **"password": "fepasjoaoxxx12",**  **"email": "fepas@gmail.com"**  **}** |
| --- |

| Resultado:  **{**  **"id": 4,**  **"username": "joaofepas12",**  **"password": "fepasjoaoxxx12",**  **"email": "fepas@gmail.com"**  **}** |
| --- |

| Objetivo: Filtrar utilizadores por username |
| --- |

| Descrição: Obter utilizador através da pesquisa por username |
| --- |

| URL:/api/users/userbyusername/{username} **(GET)** |
| --- |

| Resultado:  **{**  **"id": 1,**  **"username": "johndoe",**  **"password": "johndoeoriginal",**  **"email": "jdoe@gmail.com"**  **}** |
| --- |

| Erro:  {  "timestamp": "2021-11-05T18:55:53.358+00:00",  "status": 404,  "error": "Not Found",  "message": "Utilizador with username felipecastanho not found.",  "path": "/api/busybrain/users/userbyusername/felipecastanho"  } |
| --- |

| Objetivo: Filtrar utilizadores pelo ID |
| --- |

| Descrição: Obter utilizador através da pesquisa por ID |
| --- |

| URL:/api/users/{id} **(GET)** |
| --- |

| Resultado:  **{**  **"id": 2,**  **"username": "marydoe",**  **"password": "marydoeoriginal",**  **"email": "mdoe@gmail.com"**  **}** |
| --- |

| Erro:  {  "timestamp": "2021-11-05T19:03:30.424+00:00",  "status": 404,  "error": "Not Found",  "message": "Utilizadorwith ID7not found",  "path": "/api/busybrain/users/7"  } |
| --- |