**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)** |

|  |
| --- |
| Dados: |

|  |
| --- |
| Resultado:  [      {          "id": 2,          "name": "Urban Beach",          "endereco": "Rua da Junqueira",          "latitude": 27472.374627,          "longitude": 34848.757199,          "googleid": "HAu218haAGma",          "userrequestid": 1,          "placefavorite": **false**,          "placepresenca": **false**      },      {          "id": 3,          "name": "Urban Bar",          "endereco": "Rua da Papoila",          "latitude": 47271.374627,          "longitude": 35625.757199,          "googleid": "riUmq18NH1hQ",          "userrequestid": 3,          "placefavorite": **false**,          "placepresenca": **false**      },      {          "id": 18,          "name": "Quiosque O Príncipe",          "endereco": "Rua Professor João Barreira 1, Lisboa",          "latitude": 38.7605638,          "longitude": -9.1708542,          "googleid": "ChIJnWraO-gzGQ0Rfi4kfhLQzCM",          "userrequestid": 15,          "placefavorite": **false**,          "placepresenca": **false**      },      {          "id": 19,          "name": "Cozinha Magnífica",          "endereco": "Rua Carlos Alves 25, Lisboa",          "latitude": 38.7647222,          "longitude": -9.1827778,          "googleid": "ChIJYVT69M4yGQ0RQq94KxDqPcM",          "userrequestid": 15,          "placefavorite": **false**,          "placepresenca": **false**      },      {          "id": 20,          "name": "Cozinha Magnífica",          "endereco": "Rua Carlos Alves 25, Lisboa",          "latitude": 38.7647222,          "longitude": -9.1827778,          "googleid": "ChIJYVT69M4yGQ0RQq94KxDqPcM",          "userrequestid": 15,          "placefavorite": **false**,          "placepresenca": **false**      },      {          "id": 22,          "name": "Jungle",          "endereco": "Rua Professor Fernando da Fonseca 23A, Lisboa",          "latitude": 38.7611947,          "longitude": -9.1658592,          "googleid": "ChIJ4xQVT2QzGQ0RMFlEmkVvMNE",          "userrequestid": 15,          "placefavorite": **false**,          "placepresenca": **false**      },      {          "id": 23,          "name": "Jungle",          "endereco": "Rua Professor Fernando da Fonseca 23A, Lisboa",          "latitude": 38.7611947,          "longitude": -9.1658592,          "googleid": "ChIJ4xQVT2QzGQ0RMFlEmkVvMNE",          "userrequestid": 15,          "placefavorite": **false**,          "placepresenca": **false**      },      {          "id": 24,          "name": "Jungle",          "endereco": "Rua Professor Fernando da Fonseca 23A, Lisboa",          "latitude": 38.7611947,          "longitude": -9.1658592,          "googleid": "ChIJ4xQVT2QzGQ0RMFlEmkVvMNE",          "userrequestid": 15,          "placefavorite": **false**,          "placepresenca": **false**      } |

|  |
| --- |
| 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)** |

|  |
| --- |
| Parâmetros: place\_name - Nome do local (pesquisa por “containing”) |

|  |
| --- |
| Dados: |

|  |
| --- |
| Resultado:  [      {          "id": 18,          "name": "Quiosque O Príncipe",          "endereco": "Rua Professor João Barreira 1, Lisboa",          "latitude": 38.7605638,          "longitude": -9.1708542,          "googleid": "ChIJnWraO-gzGQ0Rfi4kfhLQzCM",          "userrequestid": 15,          "placefavorite": **false**,          "placepresenca": **false**      }  ] |

|  |
| --- |
| 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": 22,      "name": "Jungle",      "endereco": "Rua Professor Fernando da Fonseca 23A, Lisboa",      "latitude": 38.7611947,      "longitude": -9.1658592,      "googleid": "ChIJ4xQVT2QzGQ0RMFlEmkVvMNE",      "userrequestid": 15,      "placefavorite": **false**,      "placepresenca": **false**  } |

|  |
| --- |
| 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 próximos por categoria |

|  |
| --- |
| URL: <https://maps.googleapis.com/maps/api/place/nearbysearch/json?location=current_location>,current\_location&radius=5000&type=all&sensor=true&key=AIzaSyCJghxd8EZ4OGToaAqaKxkJW3YBb12\_kuU |

|  |
| --- |
| Parâmetros: type – permite a filtragem por tipo (Google Places API) |

|  |
| --- |
| Dados: |

|  |
| --- |
| Resultado:  \*Resultado dependerá dos locais apresentados |

|  |
| --- |
| 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: <https://maps.googleapis.com/maps/api/place/nearbysearch/json?location=current_location>,current\_location&radius=5000&type=all&sensor=true&key=AIzaSyCJghxd8EZ4OGToaAqaKxkJW3YBb12\_kuU |

|  |
| --- |
| Parâmetros: location (valores representarão a latitude e longitude atuais do utilizador) |

|  |
| --- |
| Dados: |

|  |
| --- |
| Resultado:  \*Resultados dependem da localização do utilizador (Google Places API) |
|  |

|  |
| --- |
| Objetivo: Armazenar um local |

|  |
| --- |
| Descrição: Armazenar informações de um local na base de dados após o clique no marcador |

|  |
| --- |
| URL: api/locals/createplace |

|  |
| --- |
| Resultado:  {          "id": 2,          "name": "Urban Beach",          "endereco": "Rua da Junqueira",          "latitude": 27472.374627,          "longitude": 34848.757199,          "googleid": "HAu218haAGma",          "userrequestid": 1,          "placefavorite": **false**,          "placepresenca": **false**      }  \*Por default, os atributos “placefavorite” e “placepresenca” são false. |

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:  [  {          "taskTitle": "Ajaja",          "userTaskId": 15,          "prioridade": "Medium",          "taskDesc": "ajajja",          "taskId": 33,          "taskType": **null**      },      {          "taskTitle": "Cooking",          "userTaskId": 15,          "prioridade": "Low",          "taskDesc": "Cooking and cleaning",          "taskId": 35,          "taskType": **null**      }  ] |

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

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

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

|  |
| --- |
| Dados:  {          "title": "Aja",          "description":"aja",          "date": "2022-01-02",          "usertaskid": 2,          "priority": 3,          "type": 1      } |

|  |
| --- |
| Resultado:  {      "id": 44,      "title": "Aja",      "description": "aja",      "date": "2022-01-02",      "usertaskid": 2,      "priority": 3,      "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: Obter informações de uma tarefa |

|  |
| --- |
| Descrição: Obter informações relativas a uma tarefa |

|  |
| --- |
| URL: /api/busybrain/tasks/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:  [] |

|  |
| --- |
| Objetivo: Exibir todas as tarefas de um utilizador |

|  |
| --- |
| Descrição: Exibir todas as tarefas relativas a um utilizador |

|  |
| --- |
| URL: api/tasks/gettarefas/{user\_task\_id} |

|  |
| --- |
| Parâmetros: user\_task\_id -> id do utilizador |

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

4 - Websites

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

|  |
| --- |
| Descrição: Obter o histórico de bloqueio e desbloqueio de websites |

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

|  |
| --- |
| Resultado:  [      {          "id": 7,          "domainid": 4,          "blockedstatus": **true**,          "userid": 8      },      {          "id": 8,          "domainid": 1,          "blockedstatus": **false**,          "userid": 8      }  ] |

|  |
| --- |
| Objetivo: Armazenar um website na BD após o bloqueio de um website (Incompleto) |

|  |
| --- |
| Descrição: |

|  |
| --- |
| URL: /api/websites/createsite **(GET)** |

|  |
| --- |
| Resultado: |

|  |
| --- |
| Erro: |

\*Devido a problemas nos métodos PUT para atualização do bloqueio/desbloqueio de websites, encontram-se de seguida as queries para tal:

//QUERY PARA A UPDATE DE WEBSITES (BLOQUEIO):

    // update website set website\_blocked\_status = '1' where sites.website\_domain\_id=:domainid  and sites.website\_user\_block=:userid (PUT MAPPING)

    //QUERY PARA A UPDATE DE WEBSITES (DEBLOQUEIO):

    //update website set website\_blocked\_status = '0' where sites.website\_domain\_id=:domainid and sites.website\_user\_block=:userid (PUT MAPPING)

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

|  |
| --- |
| Descrição: Obter os websites desbloqueados de um utilizador |

|  |
| --- |
| URL: api/websites/getunlockedwebsitesss/{website\_user\_block} |

|  |
| --- |
| Dados: website\_user\_block -> id do utilizador |

|  |
| --- |
| Resultado:  [      {          "userName": "hugoferreira22",          "userId": 8,          "blockedStatus": **true**,          "websiteId": 7,          "websiteName": "Instagram",          "websiteDomain": "www.instagram.com"      }  ] |

|  |
| --- |
| Objetivo: Obter os websites bloqueados por um utilizador |

|  |
| --- |
| Descrição: |

|  |
| --- |
| URL: api/websites/getlockedwebsitesss/{website\_user\_block} |

|  |
| --- |
| Resultado:  [      {          "userName": "hugoferreira22",          "blockedStatus": **false**,          "websiteId": 8,          "websiteName": "YouTube",          "websiteDomain": "www.youtube.com",          "userId": 8      }  ] |

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)** |

|  |
| --- |
| Dados:  {          "name": "FFF",          "description": "FFFFFF" ,          "tarefaid": 30    } |

|  |
| --- |
| Resultado:  {      "id": 11,      "name": "FFF",      "description": "FFFFFF",      "tarefaid": 30  } |

|  |
| --- |
| 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 informações de um grupo |

|  |
| --- |
| Descrição: Obter informações de um grupo quando este é selecionado |

|  |
| --- |
| URL: /api/grupos/getgroups/{user\_task\_id}/{group\_id} |

|  |
| --- |
| Dados: user\_task\_id -> id do utilizador que selecionou um grupo  group\_id -> id do grupo selecionado |

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

|  |
| --- |
| 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,          "groupDesc": "Group to buy stuff in the mall",          "groupName": "Shopping group",          "userName": "marydoe",          "participantId": 7,          "userId": 2      },      {          "groupId": 6,          "groupDesc": "Group to buy stuff in the mall",          "groupName": "Shopping group",          "userName": "hugoferreira22",          "participantId": 8,          "userId": 8      },      {          "groupId": 6,          "groupDesc": "Group to buy stuff in the mall",          "groupName": "Shopping group",          "userName": "ursula",          "participantId": 17,          "userId": 15      }  ] |

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

|  |
| --- |
| Descrição: Adicionar um participante a um grupo |

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

|  |
| --- |
| Dados: group\_id -> Identificador unico para cada grupo criado na app  {      "id": 18,      "userid": 14,      "taskid": 4  }  \*Após o método POST |

|  |
| --- |
| Resultado (do grupo):  [      {          "groupId": 6,          "groupDesc": "Group to buy stuff in the mall",          "groupName": "Shopping group",          "userName": "marydoe",          "participantId": 7,          "userId": 2      },      {          "groupId": 6,          "groupDesc": "Group to buy stuff in the mall",          "groupName": "Shopping group",          "userName": "hugoferreira22",          "participantId": 8,          "userId": 8      },      {          "groupId": 6,          "groupDesc": "Group to buy stuff in the mall",          "groupName": "Shopping group",          "userName": "ursula",          "participantId": 17,          "userId": 15      },      {          "groupId": 6,          "groupDesc": "Group to buy stuff in the mall",          "groupName": "Shopping group",          "userName": "polybius",          "participantId": 18,          "userId": 14      }  ]  \*O utilizador “polybius” é o resultado do método POST (adicionar um participante) |

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

|  |
| --- |
| Descrição: Remover um participante de um grupo |

|  |
| --- |
| URL: /api/participantesgrupos/removeparticipant/{group\_id}/{user\_identifier} **(DELETE)** |

|  |
| --- |
| Dados: user\_identifier -> ID do participante naquela tarefa.  Group\_id -> ID do grupo |

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

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

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

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

|  |
| --- |
| Parâmetros: |

|  |
| --- |
| Dados:  {            "grupoid": 6,          "placeid": 22,          "date": “2022-01-01”  } |

|  |
| --- |
| Resultado:  {      "id": 5,      "grupoid": 6,      "placeid": 22,      "date": "2022-01-01"  } |

|  |
| --- |
| Erro: |

|  |
| --- |
| Objetivo: Obter historico de convivios |

|  |
| --- |
| Descrição: Obter historico de convivios de um grupo |

|  |
| --- |
| URL: api/convivios/{grupo\_id} |

|  |
| --- |
| Dados: grupo\_id -> ID do grupo |

|  |
| --- |
| Resultado:  [      {          "groupName": "ajaja group",          "taskTitle": "Ajaja",          "convivioId": 3,          "convivioData": "2022-01-02",          "placeName": "Quiosque O Príncipe",          "placeEndereco": "Rua Professor João Barreira 1, Lisboa"      }  ]  \*1 Resultado pois o grupo inserido no URL somente possui 1 convivio |

|  |
| --- |
| Erro: |

6. Tarefas em Grupo (TarefaGrupo Controller)

|  |
| --- |
| Objetivo: Obter todas as tarefas de um grupo |

|  |
| --- |
| Descrição: Obter todas as tarefas em grupo relativas a um grupo especifico |

|  |
| --- |
| URL: api/tasksgrupo/gettasksgrupo/{task\_group\_id} |

|  |
| --- |
| Resultado:  [      {          "taskTitle": "Carry shopping items",          "userName": "marydoe",          "taskId": 45,          "taskDesc": "Help with shopping",          "priorityType": "Medium",          "participanteId": 7,          "taskGroupId": 6,          "userId": 2      }  ] |

|  |
| --- |
| Erro: |

|  |
| --- |
| Objetivo: Apagar uma tarefa |

|  |
| --- |
| Descrição: Apagar uma tarefa em grupo |

|  |
| --- |
| URL: api/tasksgrupo/deletetaskfromgrupo/ |

|  |
| --- |
| Resultado:  \*Não exibe JSON. |

|  |
| --- |
| Erro: |

|  |
| --- |
| Objetivo: Criar uma tarefa (grupo) |

|  |
| --- |
| Descrição: Criar uma tarefa para um grupo especifico |

|  |
| --- |
| URL: api/tasksgrupo/createtaskgrupo |

|  |
| --- |
| Dados:  {            "title": "Carry shopping items",          "description": "Help with shopping",          "date": "2022-01-01",          "usertaskid": 2,          "priority": 2,          "type": 1,          "taskgroupid": 6  } |

|  |
| --- |
| Resultado:  {      "id": 45,      "title": "Carry shopping items",      "description": "Help with shopping",      "date": "2022-01-01",      "usertaskid": 2,      "priority": 2,      "type": 1,      "taskgroupid": 6  } |

7. Locais Favoritos (MarcacaoFavorito Controller)

|  |
| --- |
| Objetivo: Obter todos os locais favoritos de um utilizador |

|  |
| --- |
| Descrição: |

|  |
| --- |
| URL: api/favorite/favs/{utilizador\_id} |

|  |
| --- |
| Resultado:  [      {          "favoriteId": 7,          "utilizadorId": 8,          "placeName": "Urban Bar",          "placeId": 3,          "placeEndereco": "Rua da Papoila"      }  ]  \*Locais favoritos do utilizador 8 (hugoferreira22) |

\*Queries de UPDATE ao “status” de marcação de favoritos:

 //REMARCAÇÃO DE LOCAIS FAVORITOS

   // update marcacao\_favorito set isfavorite = '1' where marcacoes.utilizador\_id=:utilizadorid and marcacoes.local\_id=:placeid (PUT MAPPING)

   //REMOÇÃO DE LOCAIS FAVORITOS

   // update marcacao\_favorito set isfavorite = '0' where marcacoes.utilizador\_id=:utilizadorid and marcacoes.local\_id=:placeid (PUT MAPPING)

|  |
| --- |
| Objetivo: Armazenar a marcação de favorito |

|  |
| --- |
| Descrição: Armazenar na BD a marcação de um local como favorito (1ª marcação num local) |

|  |
| --- |
| URL: api/favorite/createmarcacao |

|  |
| --- |
| Dados: |

|  |
| --- |
| Resultado: |

8. Locais com presenças (MarcacaoPresenca Controller)

|  |
| --- |
| Objetivo: Obter todos os locais com presenças (utilizador) |

|  |
| --- |
| Descrição: Obter todos os locais onde o utilizador marcou uma presença. |

|  |
| --- |
| URL: api/presenca/presencas/{utilizador\_id} |

|  |
| --- |
| Resultado:  [      {          "userName": "ursula",          "presencaId": 2,          "utilizadorId": 15,          "placeName": "Quiosque O Príncipe",          "placeId": 18,          "placeEndereco": "Rua Professor João Barreira 1, Lisboa"      }  ] |

\*Queries de UPDATE ao “status” de marcação de presenças:

//REMARCAÇÃO DE PRESENÇAS NUM LOCAL

    // update marcacao\_presenca set wasthere = '1' where marcacoes.utilizador\_id=:utilizadorid and marcacoes.local\_id=:placeid

    //REMOÇÃO DE PRESENÇAS NUM LOCAL

    // update marcacao\_presenca set wasthere = '0' where marcacoes.utilizador\_id=:utilizadorid and marcacoes.local\_id=:placeid

|  |
| --- |
| Objetivo: Armazenar a marcação de presenças |

|  |
| --- |
| Descrição: Armazenar na BD a marcação de presença num local (1ª marcação) |

|  |
| --- |
| URL: api/presenca/createmarcacaopresenca |

|  |
| --- |
| Dados: |

|  |
| --- |
| Resultado: |

**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"  } |

|  |
| --- |
| **Objetivo: Login do utilizador** |

|  |
| --- |
| **Descrição: Verificação das informações para login do utilizador** |

|  |
| --- |
| **URL: /api/users/{user\_name}/{user\_password}** |

|  |
| --- |
| **Resultado:**  {      "id": 15,      "username": "ursula",      "password": "ursula2002",      "email": "ursula@gmail.com"  }  **\*Após a inserção do “user\_name” e do “user\_password7”** |