REST API

Introdução

A API do Think Toilet oferece acesso eficiente a informações sobre usuários, casas de banho e comentários, permitindo consultas, visualizações e interações organizadas. Desenvolvida para integração com a aplicação móvel, a API fornece dados atualizados e é compatível com outras plataformas.

Os dados são entregues em formato JSON, garantindo respostas consistentes e facilitando a integração com sistemas diversos. A estrutura dos dados e os endpoints são flexíveis, projetados para suportar expansões futuras e melhorias contínuas na aplicação.

Endpoints

Mostrar usuários

- URL:
  `/users`
- METHOD:
  `GET`
- SUCCESS RESPONSE:

```
[
  {
    "id": [integer],
    "name": [string],
    "points": [integer],
    "iconId": [alphanumeric],
    "birthDate": [date],
    "creationDate": [date],
  },
]
```

- ERROR RESPONSE:

```
{
  "status": 500,
  "message": "An unexpected error occurred.",
  "timestamp": [datetime]
}
```

- SAMPLE CALL:

```
@GET("users")
suspend fun getUsers(): List<User>
```

Mostrar usuário por ID

- **URL:**
  `/users/:id`
- **METHOD:**
  `GET`
- **URL PARAMETHERS:**
  - Required:
    `id=[integer]`
- **SUCCESS RESPONSE:**

```
{
  "id": [integer],
  "name": [string],
  "points": [integer],
  "iconId": [alphanumeric],
  "birthDate": [date],
  "creationDate": [date],
}
```

- **ERROR RESPONSE:**

```
{
  "status": 404,
  "message": "User with id {id} not found.",
  "timestamp": [datetime]
}
```

- **SAMPLE CALL:**

```
@GET("users/{id}")
suspend fun getUsersById(@Path("id") id: Int): User
```

Mostrar comentários de um usuário

- **URL:**
  `/users/:id/comments`
- **METHOD:**
  `GET`
- **URL PARAMETHERS:**
  - Required:
    `id=[integer]`
- **SUCCESS RESPONSE:**

```
[
  {
    "id": [integer],
    "toiletId": [integer],
    "userId": [integer],
    "text": [string],
    "ratingClean": [integer],
    "ratingPaper": [boolean],
    "ratingStructure": [integer],
    "ratingAccessibility": [integer],
    "datetime": [datetime],
    "numLikes": [integer],
    "numDislikes": [integer],
    "score": [integer]
  },
]
```

- **ERROR RESPONSE:**

```
{
  "status": 404,
  "message": "Comment with user id {id} not found.",
  "timestamp": [datetime]
}
```

- **SAMPLE CALL:**

```
@GET("users/{id}/comments")
suspend fun getUserComments(@Path("id") id: Int): List<Comment>
```

Mostrar comentários de um usuário de forma paginada

- **URL:**
  `/users/:id/comments/page?size=:size&page=:page`
- **METHOD:**
  `GET`
- **URL PARAMETHERS:**
- **Required:**
  `id=[integer]`
- **Optional:**
  `size=[integer]` (default: 20)
  `page=[integer]` (default: 0)
- **SUCCESS RESPONSE:**

```
{
  "content": [
    {
      "id": [integer],
      "toiletId": [integer],
      "userId": [integer],
      "text": [string],
      "ratingClean": [integer],
      "ratingPaper": [boolean],
      "ratingStructure": [integer],
      "ratingAccessibility": [integer],
      "datetime": [datetime],
      "numLikes": [integer],
      "numDislikes": [integer],
      "score": [integer]
    },
  ],
  "page": {
    "size": [integer],
    "number": [integer],
    "totalElements": [integer],
    "totalPages": [integer]
  }
}
```

- **ERROR RESPONSE:**

```
{
  "status": 404,
  "message": "Comment with user id {id} not found.",
  "timestamp": [datetime]
}
```

- **SAMPLE CALL:**

```
@GET("users/{id}/comments/page")
suspend fun getUserCommentsPage(@Path("id") id: Int, @Query("size") size:
Int, @Query("page") page: Int): PageResponse<Comment>
```

Mostrar casas de banho

- **URL:**
  `/toilets`
- **METHOD:**
  `GET`
- **SUCCESS RESPONSE:**

```
[
  {
    "id": [integer],
    "name": [string],
    "address": [string],
    "rating": {
      "clean": [decimal],
      "structure": [decimal],
      "accessibility": [decimal],
      "paper": [decimal],
    },
    "extras": [
      {
        "id": [integer],
        "name": [string],
      },
    ],
    "latitude": [decimal],
    "longitude": [decimal],
    "numComments": [integer],
    "placeId": [alphanumeric],
    "image": [alphanumeric],
    "comments": []
  },
]
```

- **ERROR RESPONSE:**

```
{
  "status": 500,
  "message": "An unexpected error occurred.",
  "timestamp": [datetime]
}
```

- **SAMPLE CALL:**

```
@GET("toilets")
suspend fun getToilets(): List<Toilet>
```

Mostrar casas de banho de forma paginada

- **URL:**
  `/toilets/page?size=:size&page=:page`
- **METHOD:**
  `GET`
- **URL PARAMETHERS:**

- Optional:
  `size=[integer]` (default: 20)
  `page=[integer]` (default: 0)

- **SUCCESS RESPONSE:**

```
{
  "content": [
    {
      "id": [integer],
      "name": [string],
      "address": [string],
      "rating": {
        "clean": [decimal],
        "structure": [decimal],
        "accessibility": [decimal],
        "paper": [decimal],
      },
      "extras": [
        {
          "id": [integer],
          "name": [string],
        },
      ],
      "latitude": [decimal],
      "longitude": [decimal],
      "numComments": [integer],
      "placeId": [alphanumeric],
      "image": [alphanumeric],
      "comments": []
    },
  ],
  "page": {
    "size": [integer],
    "number": [integer],
    "totalElements": [integer],
    "totalPages": [integer]
  }
}
```

- **ERROR RESPONSE:**

```
{
  "status": 500,
  "message": "An unexpected error occurred.",
  "timestamp": [datetime]
}
```

- **SAMPLE CALL:**

```
@GET("toilets/page")
suspend fun getToiletsPage(@Query("size") size: Int, @Query("page") page:
Int): PageResponse<Toilet>
```

Mostrar casas de banho por ID

- **URL:**
  `/toilets/:id`
- **METHOD:**
  `GET`
- **URL PARAMETHERS:**
  - Required:
    `id=[integer]`
- **SUCCESS RESPONSE:**

```
{
  "id": [integer],
  "name": [string],
  "address": [string],
  "rating": {
    "clean": [decimal],
    "structure":  [decimal],
    "accessibility": [decimal],
    "paper": [decimal]
  },
  "extras": [
    {
      "id": [integer],
      "name": [string]
    },
  ],
  "latitude": [decimal],
  "longitude": [decimal],
  "numComments": [integer],
  "placeId": [alphanumeric],
  "image": [alphanumeric],
  "comments": []
}
```

- **ERROR RESPONSE:**

```
{
  "status": 404,
  "message": "Toilet with id {id} not found.",
  "timestamp": [datetime]
}
```

- **SAMPLE CALL:**

```
@GET("toilets/{id}")
suspend fun getToiletById(@Path("id") id: Int): Toilet
```

Mostrar casas de banho próximas

- **URL:**
  `/toilets/nearby?lat=:lat&lon=:lon`
- **METHOD:**
  `GET`
- **URL PARAMETHERS:**
  - Required:
    `lat=[decimal]`
    `lon=[decimal]`
- **SUCCESS RESPONSE:**

```
[
  {
    "id": [integer],
    "name": [string],
    "address": [string],
    "rating": {
      "clean": [decimal],
      "structure": [decimal],
      "accessibility": [decimal],
      "paper": [decimal]
    },
    "extras": [
      {
        "id": [integer],
        "name": [string]
      },
    ],
    "latitude": [decimal],
    "longitude": [decimal],
    "numComments": [integer],
    "placeId": [alphanumeric],
    "image": [alphanumeric],
    "comments": []
  },
]
```

- **ERROR RESPONSE:**

```
{
  "status": 500,
```

```
  "message": "An unexpected error occurred.",
  "timestamp": [datetime]
}
```

- **SAMPLE CALL:**

```
@GET("toilets/nearby")
suspend fun getNearbyToilets(@Query("lat") lat: Double, @Query("lon") lon:
Double): List<Toilet>
```

Mostrar casas de banho próximas de forma paginada

- **URL:**
  `/toilets/nearby/page?lat=:lat&lon=:lon&size=:size&page=:page`
- **METHOD:**
  `GET`
- **URL PARAMETHERS:**
  - Required:
    `lat=[decimal]`
    `lon=[decimal]`
  - Optional:
    `size=[integer]` (default: 20)
    `page=[integer]` (default: 0)
- **SUCCESS RESPONSE:**

```
{
  "content": [
    {
      "id": [integer],
      "name": [string],
      "address": [string],
      "rating": {
        "clean": [decimal],
        "structure": [decimal],
        "accessibility": [decimal],
        "paper": [decimal]
      },
      "extras": [
        {
          "id": [integer],
          "name": [string]
        },
      ],
      "latitude": [decimal],
      "longitude": [decimal],
      "numComments": [integer],
      "placeId": [alphanumeric],
```

```
      "image": [alphanumeric],
      "comments": []
    },
  ],
  "page": {
    "size": [integer],
    "number": [integer],
    "totalElements": [integer],
    "totalPages": [integer]
  }
}
```

- **ERROR RESPONSE:**

```
{
  "status": 500,
  "message": "An unexpected error occurred.",
  "timestamp": [datetime]
}
```

- **SAMPLE CALL:**

```
@GET("toilets/nearby/page")
suspend fun getNearbyToiletsPage(@Query("lat") lat: Double, @Query("lon")
lon: Double, @Query("size") size: Int, @Query("page") page: Int):
PageResponse<Toilet>
```

Mostrar comentários de uma casa de banho

- **URL:**
  `/toilets/:id/comments`
- **METHOD:**
  `GET`
- **URL PARAMETHERS:**
  - Required:
    `id=[integer]`
- **SUCCESS RESPONSE:**

```
[
  {
    "id": [integer],
    "toiletId": [integer],
    "userId": [integer],
    "text": [string],
    "ratingClean": [integer],
    "ratingPaper": [boolean],
```

```
    "ratingStructure": [integer],
    "ratingAccessibility": [integer],
    "datetime": [datetime],
    "numLikes": [integer],
    "numDislikes": [integer],
    "score": [integer]
  },
]
```

- **ERROR RESPONSE:**

```
{
  "status": 404,
  "message": "Comment with toilet id {id} not found.",
  "timestamp": [datetime]
}
```

- **SAMPLE CALL:**

```
@GET("toilets/{id}/comments")
suspend fun getToiletComments(@Path("id") id: Int): List<Comment>
```

Mostrar comentários de uma casa de banho de forma paginada

- **URL:**
  `/toilets/:id/comments/page?size=:size&page=:page`
- **METHOD:**
  `GET`
- **URL PARAMETHERS:**
  - **Required:**
    `id=[integer]`
  - **Optional:**
    `size=[integer]` (default: 20)
    `page=[integer]` (default: 0)
- **SUCCESS RESPONSE:**

```
{
  "content": [
    {
      "id": [integer],
      "toiletId": [integer],
      "userId": [integer],
      "text": [string],
      "ratingClean": [integer],
      "ratingPaper": [boolean],
```

```
      "ratingStructure": [integer],
      "ratingAccessibility": [integer],
      "datetime": [datetime],
      "numLikes": [integer],
      "numDislikes": [integer],
      "score": [integer]
    },
  ],
  "page": {
    "size": [integer],
    "number": [integer],
    "totalElements": [integer],
    "totalPages": [integer]
  }
}
```

- **ERROR RESPONSE:**

```
{
  "status": 404,
  "message": "Comment with toilet id {id} not found.",
  "timestamp": [datetime]
}
```

- **SAMPLE CALL:**

```
@GET("toilets/{id}/comments/page")
suspend fun getToiletCommentsPage(@Path("id") id: Int, @Query("size") size:
Int, @Query("page") page: Int): PageResponse<Comment>
```