

Mestrado em Engenharia Telecomunicações e Informática

Relatório do Trabalho Prático

Armazenamento de Dados em Ambientes Distribuídos

Docentes: Catarina Ferreira da Silva e João Matos

Grupo 3:

Alexandre Rodrigues, nº 105260

Loice Sitole, nº 123848

Maria Vinheiras, nº 105563

Mariana Gonçalves, nº 127524

Ricardo Gouveia, nº 105062

Índice

Introdução	3
Documentação REST.....	4

Introdução

Este trabalho visa a elaboração de uma API utilizando o MongoDB, Node.js, Express.js, React.js e o POSTMAN.

Para a elaboração da API temos de, primeiramente, fazer a configuração do Backend criando um servidor com o Node.js e o Express.js, de seguida, conectar a base de dados Mongo ao servidor e implementar os endpoints da API propostos. Após implementação devemos testar os pedidos HTTP com a aplicação POSTMAN.

Para a configuração de Frontend é necessário que seja desenvolvido utilizando o React.js e que seja conectado ao servidor Backend, por fim, devemos apresentar os dados no lado Cliente com base nos endpoints anteriormente criados.

Documentação REST

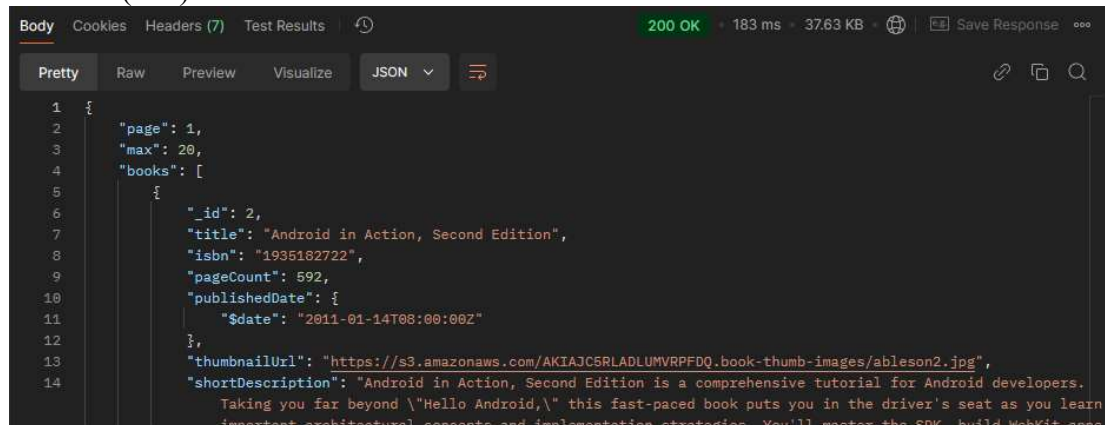
1- Listar livros com paginação (Ricardo)

Este endpoint mostra todos os utilizadores em várias páginas.

/books (GET)

Parâmetros: page- Indica o número da página dos livros
max : número de livros por página

Sucesso (200):



Erros:

500: Não encontrou livros

400: Erro servidor

3- Adicionar um ou vários livros (Ricardo)

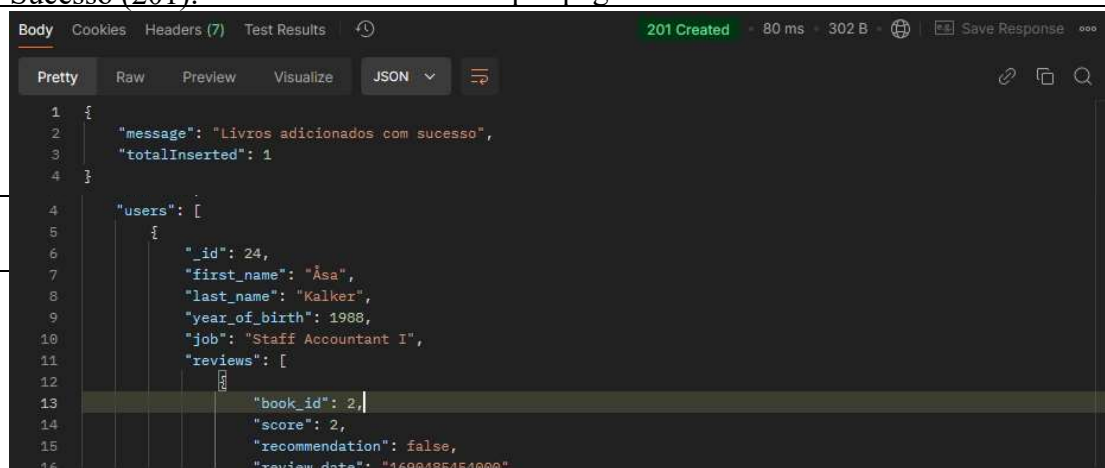
Neste endpoint é possível adicionar um ou vários livros

/books (POST)

Parâmetro: livro- livro que se deseja inserir na base de dados

Parâmetro: page- Indica o número da página dos utilizadores


Max : número de utilizadores por página

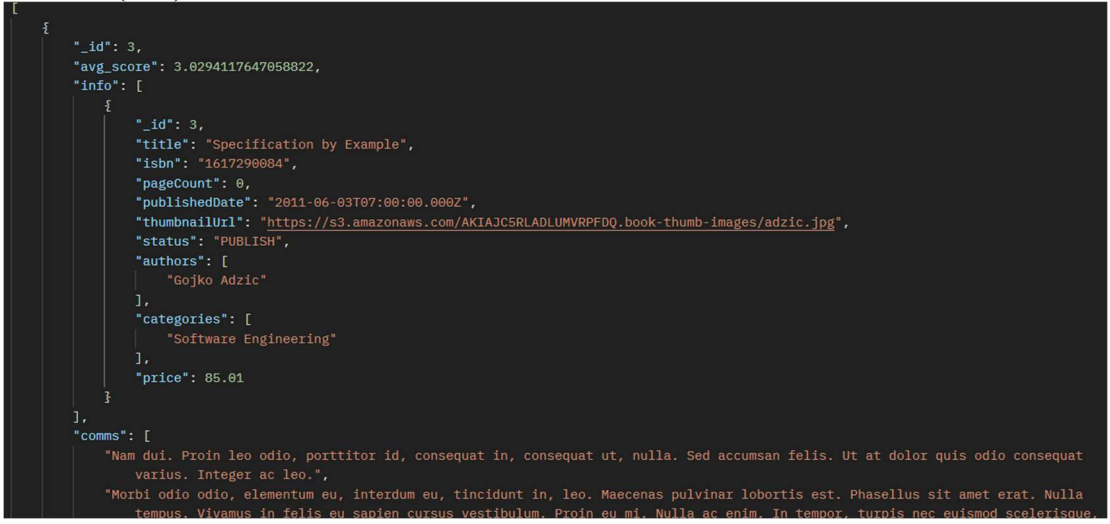


Erros:

500: Não encontrou utilizadores

400: Erro servidor

4- Adicionar um ou vários utilizadores (Ricardo)
Neste endpoint é possível adicionar um ou vários utilizadores
/users (POST)
Parâmetro: userDado: Utilizador que se deseja inserir na base dados
Sucesso (201):

Erros:
400- Erro servidor

5 - Pesquisar livro pelo <i>_id</i> (Maria)
Este endpoint mostra o livro correspondente ao <i>_id</i> indicado incluindo a sua informação, o <i>average score</i> e a lista de comentários
/books/id/:id (GET)
Parâmetro: id – corresponde ao id do livro
Sucesso (200):

Erros:
404: Coundn't find that book
500: Server error

6 – Pesquisar user pelo *id* (Maria)

Este endpoint mostra o utilizador correspondente ao *_id* e inclui o top 3 livros do utilizador

/users/:id (GET)

Parâmetro:

id – corresponde ao id de cada utilizador

Sucesso (200):

```
{
  "_id": 3,
  "first_name": "Danièle",
  "last_name": "Gascard",
  "year_of_birth": 1999,
  "job": "Recruiter",
  "reviews": [
    {
      "book_id": 33,
      "score": 5,
      "recommendation": true,
      "review_date": "1586584290000"
    },
    {
      "book_id": 18,
      "score": 2,
      "recommendation": false,
      "review_date": "1653094255000"
    },
    {
      "book_id": 16,
      "score": 2,
      "recommendation": true,
      "review_date": "1684816884000"
    }
  ]
}
```

Erros:

404: Couldn't find that user

500: Server Error

7 – Remover livro pelo *id* (Maria)

Este endpoint remove um livro através do seu *_id*

/books/:id (DELETE)

Parâmetro:

id – corresponde ao id do livro

Sucesso (200):

```
1 ✓ {
2   |   "acknowledged": true,
3   |   "deletedCount": 0
4   | }
```

404: Couldn't find that book

500: Server error

8 – Remover user pelo <i>id</i> (Maria)
Este endpoint remove um user através do seu <i>_id</i>
/users/:id (DELETE)
Parâmetro: id – corresponde ao id do user
Sucesso (200): <pre> 1 { 2 "acknowledged": true, 3 "deletedCount": 0 4 }</pre>
Erros: 404: Couldn't find that user 500: Server error

9 – Update livro (Maria)
Este endpoint faz update do livro pelo <i>_id</i>
/books/:id (PUT)
Parâmetro: id – corresponde ao id do livro
Sucesso (200): <pre> 1 { 2 "acknowledged": true, 3 "modifiedCount": 1, 4 "upsertedId": null, 5 "upsertedCount": 0, 6 "matchedCount": 1 7 }</pre>
Erros: 404: Couldn't find that book 500: Server error

10 – Update user (Maria)
Este endpoint faz update do user pelo <i>_id</i>
/user/:id (PUT)
Parâmetro: id – corresponde ao id do user
Sucesso (200): <pre> 1 { 2 "acknowledged": true, 3 "modifiedCount": 1, 4 "upsertedId": null, 5 "upsertedCount": 0, 6 "matchedCount": 1 7 }</pre>

Erros:

404: Couldn't find that user

500: Server error

11- Lista de livros com maior score (pela média) (Loice)

Este endpoint apresenta lista de livros com maior score (pela média), por ordem descendente.

/books/top/:limit (GET)

Parâmetro: limit – corresponde ao limite que o utilizador deseja filtrar a sua pesquisa

Sucesso (200):

```
1  [
2    {
3      "_id": 23,
4      "avg_score": 3.216494845360825,
5      "info": [
6        {
7          "_id": 23,
8          "title": "Hibernate in Action (Chinese Edition)",
9          "pageCount": 400,
10         "publishedDate": "1999-06-01T07:00:00.000Z",
11         "thumbnailUrl": "https://s3.amazonaws.com/AKIAJC5RLADLUMVRPFDQ.book-thumb-images/
            bauer-cn.jpg",
12         "status": "PUBLISH",
13         "authors": [
14           "Christian Bauer",
15           "Gavin King"
16         ],
17         "categories": [
18           "Java"
19         ],
20         "price": 8.51
21       }
22     ]
23   }
```

Erros:

404: Couldn't find that book

500: Server error

12- Lista de livros ordenado pelo número total de reviews (Mariana)

Este endpoint apresenta a lista de livros ordenado pelo número total de reviews

/books/ratings/:order (GET)

Parâmetro: **order**: É a ordem em que o utilizador deseja apresentar os livros

Sucesso (200):

```
1  [
2    {
3      "_id": 16,
4      "book_title": [
5        "Brownfield Application Development in .NET"
6      ],
7      "number_reviews": 76
8    },
9    {
10     "_id": 21,
11     "book_title": [
12       "3D User Interfaces with Java 3D"
13     ],
14     "number_reviews": 76
15   }
```


Erros:
404: Couldn't find that book
500: Server error

13- Lista de livros com mais 5 estrelas. (Mariana/Maria)
Este endpoint apresenta a lista de livros com mais 5 estrelas.
/books/star (GET)
Parâmetro: Não tem.
Sucesso (200):
<pre> 1 [2 { 3 "_id": 16, 4 "book_title": [5 "Brownfield Application Development in .NET" 6], 7 "total_5_reviews": 18 8 }, 9 { 10 "_id": 21, 11 "book_title": [12 "3D User Interfaces with Java 3D" 13], 14 "total_5_reviews": 9 15 }, 16 { 17 "_id": 36,</pre>
Erros:
404: Couldn't find that book
500: Server error

14-Lista de livros avaliados no ano (Alexandre)
Este endpoint lista livros avaliados num ano específico.
/books/year/:year (GET)
Sucesso (200):
<pre> 1 [2 { 3 "_id": 20, 4 "title": "Taming Jaguar", 5 "isbn": "1884777686", 6 "pageCount": 362, 7 "publishedDate": "2000-07-01T07:00:00.000Z", 8 "thumbnailUrl": "https://s3.amazonaws.com/AKIAJ3C5RLADLUMVRPFDQ.book-thumb-images/barlotta3.jpg", 9 "longDescription": "Taming Jaguar is part of the PowerBuilder Developer's series, which includes Distributed Application Development with PowerBuilder 6 and Jaguar Development with PowerBuilder 7. An application server is the heart of your enterprise computing architecture, centralizing your web content, business logic, and access to your data and legacy applications. Sybase's application server, Jaguar CTS, delivers performance, scalability, and flexibility running CORBA , COM, Java/EJB, C++, and PowerBuilder components. If you are looking to adopt Jaguar in your enterprise, look no further. Taming Jaguar shows you how to solve the real-world problems of installing, trouble-shooting, designing, developing, and maintaining a Jaguar application. Topical chapters are organized in a Q & A format making it easy for you to quickly find the solution to your problem. They also provide foundational and background information as well as detailed technical how-tos. Although designed so you can find your problems easily, this book is meant to be read cover-to-cover with each chapter discussing its topic exhaustively. What's inside: J2EE development Java Servlets Jaguar administration & code balancing EJBs Web development with PowerDynamo Advanced component design ", 10 "status": "PUBLISH", 11 "authors": [</pre>

Erros:
500: Erro de servidor
404: Não encontrou utilizador

15-Lista de livros ordenado pela quantidade de comentários (Alexandre)

Este endpoint lista livros com comentários, ordenados pelo número de comentários

/books/comments (GET)

Sucesso (200):

```
[
  {
    "_id": 23,
    "title": "Hibernate in Action (Chinese Edition)",
    "book_comments": [
      {
        "_id": 21,
        "user_id": 72,
        "book_id": 23,
        "comment": "Integer pede justo, lacinia eget, tincidunt eget, tempus vel, pede. Morbi porttitor lorem id ligula. Su",
        "date": "1679453400000"
      },
      {
        "_id": 50,
        "user_id": 201,
        "book_id": 23,
        "comment": "Morbi sem mauris, laoreet ut, rhoncus aliquet, pulvinar sed, nisl.",
        "date": "1679122136000"
      },
      {
        "_id": 64,
        "user_id": 119,
        "book_id": 23,
        "comment": "Mauris enim leo, rhoncus sed, vestibulum sit amet, cursus id, turpis. Integer aliquet, massa id lobortis",
        "date": "1679398361000"
      }
    ]
  }
]
```

Erros:
500: Erro de servidor
404: Não encontrou utilizador

16 – Número de reviews por trabalho (Mariana)

Este endpoint devolve o número de reviews de livros filtrado pelo trabalho introduzido pelo utilizador

/books/job/:job (GET)

Parâmetros: **Job** – Indica o trabalho pelo qual o utilizador quer filtrar

Sucesso (200):

```
1  [
2    {
3      "Recruiter": 23
4    }
5  ]
```

Erros:
404: Não encontrou as reviews

500: Erro de servidor

17 – Listar livros por diferentes parâmetros (preço, autor e categoria) (Mariana)

Este endpoint devolve o número de livros filtrado pelo preço, autor, categoria introduzidos pelo utilizador.

/books/category/:category/price/:price/author/:author (GET)

Parâmetros:

Categoria – Indica o tipo de categoria do livro.

Preço – Indica o preço do livro. Procura livros com preços mais baratos do que o utilizador colocou.

Autor – Indica o autor do livro. Utiliza REGEX, portanto o utilizador pode só colocar “D” e encontrará o autor.

Sucesso (200):

```
1  [
2    {
3      "_id": 9,
4      "title": "Griffon in Action",
5      "isbn": "1935182234",
6      "pageCount": 375,
7      "thumbnailUrl": "https://s3.amazonaws.com/AKIAJC5RLADLUMVRPFDQ.book-thumb-images/almiray.jpg",
8      "longDescription": "longDescription",
9      "status": "PUBLISH",
10     "categories": "Java",
11     "authors": "Danno Ferrin",
12     "price": 91.48
13   }
]
```

Erros:

404: Não encontrou os livros

500: Erro de servidor

18 – Criar um novo comentário (Mariana)

Este endpoint cria um novo comentário.

/comments (POST)

Parâmetros:

Body do comentário

```
○ none ○ form-data ○ x-www-form-urlencoded ● raw ○ binary ○ GraphQL JSON Beautify
1  {
2    "user_id":1,
3    "book_id":1,
4    "comment": "comentário"
5  }
```

Sucesso (200):

```
1  {
2    "acknowledged": true,
3    "insertedId": "673788f8c7d2ffd78814e758"
4  }
```

Erros:

500: Erro de servidor

19 – Elimina um comentário (Mariana)
Este endpoint cria um novo comentário.
/comments/:id (DELETE)
Parâmetros: id - Identifica o comentário
Sucesso (200): <pre> 1 { 2 "acknowledged": true, 3 "deletedCount": 0 4 } </pre>
Erros: 404: Não encontrou o comentário 500: Erro de servidor

#1 – Adicionar um livro a uma livraria (Alexandre)
Estes endpoint devolve o número de livrarias perto de uma localização dada
/livrarias/:id (POST)
Parâmetros: id – identifica a livraria
Sucesso (200): <pre> 1 { 2 "message": "Livros adicionados à livraria com sucesso" 3 } </pre>
Erros: 404: Não existem livrarias perto 500: Server Error

#2 – Pesquisar livros numa livraria (Alexandre)

Estes endpoint devolve o número de livros numa livraria

/livrarias/id/:id (GET)

Parâmetros:

Sucesso (200):

```
1 [
2   {
3     "_id": 4,
4     "title": "Flex 3 in Action",
5     "isbn": "1933988746",
6     "pageCount": 576,
7     "publishedDate": "2009-02-02T00:00:00.000Z",
8     "thumbnailUrl": "https://s3.amazonaws.com/AKIAJCSRLAOLUMVRPFDQ.book-thumb-images/ahmed.jpg",
9     "longDescription": "New web applications require engaging user-friendly interfaces and the cooler, the better. With Flex 3, web
    developers at any skill level can create high-quality, effective, and interactive Rich Internet Applications (RIAs) quickly and
    easily. Flex removes the complexity barrier from RIA development by offering sophisticated tools and a straightforward
    programming language so you can focus on what you want to do instead of how to do it. And now that the major components of Flex
    are free and open-source, the cost barrier is gone, as well! Flex 3 in Action is an easy-to-follow, hands-on Flex tutorial.
    Chock-full of examples, this book goes beyond feature coverage and helps you put Flex to work in real day-to-day tasks. You'll
    quickly master the Flex API and learn to apply the techniques that make your Flex applications stand out from the crowd.
    Interesting themes, styles, and skins It's in there. Working with databases You got it. Interactive forms and validation
    You bet. Charting techniques to help you visualize data Bam! The expert authors of Flex 3 in Action have one goal to help
    you get down to business with Flex 3. Fast. Many Flex books are overwhelming to new users focusing on the complexities of
    the language and the super-specialized subjects in the Flex eco-system; Flex 3 in Action filters out the noise and dives into
    the core topics you need every day. Using numerous easy-to-understand examples, Flex 3 in Action gives you a strong foundation
    that you can build on as the complexity of your projects increases.",
10    "status": "PUBLISH",
11    "authors": [
12      "Tariq Ahmed with Jon Hirschi",
13      "Faisal Abid"
14    ],
15  },
16 ]
```

Erros:

404: Não existem livrarias perto

500: Server Error

#3 – Lista de livrarias perto de uma localização (Maria)

Este endpoint mostra livrarias perto de uma localização dada

/livrarias/nearLocation (GET)

Parâmetro:

Sucesso (200):

```
1 [
2   {
3     "_id": 1,
4     "type": "Feature",
5     "geometry": {
6       "type": "Point",
7       "coordinates": [
8         -9.14421890064127,
9         38.7105419551935
10      ]
11    },
12     "properties": {
13       "OBJECTID": 1,
14       "COD_SIG": "506",
15       "INF_NOME": "Livraria Loreto",
16       "INF_MORADA": "Rua do Loreto, 15",
17       "FREGUESIA": "Misericórdia",
18       "INF_TELEFONE": "+351 210 998 295",
19       "INF_DESCRICAO": "Esta sala de cinema, aberta desde 1904, conheceu ao longo dos anos diversas designações - Salão Ideal, Cinema
        Ideal, Cine Camões e Cine Paraíso. O espaço sofreu uma grande remodelação nos anos 50, mas conservou todo o charme de um
        cinema de bairro, com o seu foyer, balcão e plateia. Em agosto de 2014 reabriu como Cinema Ideal, depois de um profundo
        trabalho de renovação e recuperação, de arquitetura e de equipamento de projeção de imagem e som. ",
20       "INF_FONTE": "-",
21       "INF_ACTIVADO": 1,
22       "GlobalID": "45afe625-02d7-4712-b6b8-7aa0c2116ed6"
23     }
24   },
25 ]
```

```

23  |
24  |   },
25  |   {
26  |     "_id": 2,
27  |     "type": "Feature",
28  |     "geometry": {
29  |       "type": "Point",
30  |       "coordinates": [
31  |         -9.14217296415889,
32  |         38.7155597377788
33  |       ]
34  |     },
35  |     "properties": {
36  |       "OBJECTID": 2,
37  |       "COD_SIG": "587",
38  |       "INF_NOME": "Livraria Júnior",
39  |       "INF_MORADA": "Praça dos Restauradores, Palácio da Foz",
40  |       "FREGUESIA": "Santa Maria Maior",
41  |       "INF_TELEFONE": "+351 213 462 157 | +351 213 476 129",
42  |       "INF_DESCRICAO": "A Cinemateca Júnior, instalada no Palácio Foz, é um serviço da Cinemateca Portuguesa-Museu do Cinema
43  |         direccionado para os espectadores infantis e juvenis.",
44  |       "INF_FONTE": "http://www.cinemateca.pt/",
45  |       "INF_ACTIVIVO": 1,
46  |       "GlobalID": "1d3ca135-5a24-478c-95f4-dd2dc87a65c9"
47  |     },
48  |     "books": [

```

Erros:

404: Nenhuma livraria próxima

500: Server error

#4 – Lista de livrarias perto do caminho de uma rota (Maria/Mariana)

Este endpoint devolve uma lista de livrarias numa rota dada

/livrarias/pertoRota (GET)

Parâmetros:

Sucesso (200):

```

1  | [
2  |   {
3  |     "_id": 1,
4  |     "type": "Feature",
5  |     "geometry": {
6  |       "type": "Point",
7  |       "coordinates": [
8  |         -9.14421898864127,
9  |         38.7185419551935
10  |       ]
11  |     },
12  |     "properties": {
13  |       "OBJECTID": 1,
14  |       "COD_SIG": "586",
15  |       "INF_NOME": "Livraria Loreto",
16  |       "INF_MORADA": "Rua do Loreto, 15",
17  |       "FREGUESIA": "Misericórdia",
18  |       "INF_TELEFONE": "+351 219 998 295",
19  |       "INF_DESCRICAO": "Esta sala de cinema, aberta desde 1904, conheceu ao longo dos anos diversas designações - Salão Ideal, Cinema
20  |         Ideal, Cine Camões e Cine Paraíso. O espaço sofreu uma grande remodelação nos anos 50, mas conservou todo o charme de um
21  |         cinema de bairro, com o seu foyer, balcão e plateia. Em agosto de 2014 reabriu como Cinema Ideal, depois de um profundo
22  |         trabalho de renovação e recuperação, de arquitetura e de equipamento de projecção de imagem e som. ",
23  |       "INF_FONTE": "-",
24  |       "INF_ACTIVIVO": 1,
25  |       "GlobalID": "45afe625-02d7-4712-b6b8-7aa8c2116ed6"
26  |     }
27  |   }
28  | ]

```

Erros:

404: Não existem livrarias perto da rota

500: Server error

#5 – Retornar número de livrarias perto de uma localização (Maria)
Estes endpoint devolve o número de livrarias perto de uma localização dada
/livrarias/quantasPerto (GET)
Parâmetros:
Sucesso (200): <pre> 1 { 2 "quantidade": 4 3 }</pre>
Erros: 404: Não existem livrarias perto 500: Server Error

#6 – Verifica se um utilizador está dentro da feira do livro (Mariana)
Estes endpoint
/livrarias/feiralivro (GET)
Parâmetros: coordenadas
Sucesso (200): <pre> 1 Está na feira do livro</pre>
Erros: 404: Não existem livrarias perto 500: Server Error