

Cap 10 - Aplicações em Cloud Computing - Atividade 2

Integrantes

- Abigail Lima - RM96351
- Kaique Bernardo - RM96395

Links:

- Repositorio: <https://github.com/abigailmvlima/traveller>
- Publicação Site: <https://abigailmvlima.github.io/gullivertravell/>
- Tripadvisor: <https://tripadvisor-content-api.readme.io/reference/searchforlocations>

Escolha de API

Lambda AWS + Tripadvisor

APIs UTILIZADAS

Tripadvisor API

A API do Tripadvisor é uma interface de programação de aplicativos (API, na sigla em inglês) fornecida pelo Tripadvisor, que permite que desenvolvedores integrem dados e recursos do site do Tripadvisor em seus próprios aplicativos e sites.

A API fornece acesso aos dados do Tripadvisor, incluindo informações sobre hotéis, restaurantes, atrações turísticas, bem como comentários, avaliações e fotos de

usuários. Com a API, os desenvolvedores podem criar aplicativos e serviços personalizados, como agregadores de avaliações, guias turísticos, aplicativos móveis de reservas de viagens e muito mais.

Documentação Tripadvisor

Tripadvisor

 <https://www.tripadvisor.com/developers>

Utilização dos endpoint e a função de cada um

A utilização dos endpoint do tripadvisor, foi para pesquisa de localidades na apresentação da home no site. Tambem a utilização do endpoint de pesquisa das avaliações da experiencia dos usuarios "Classificação, texto de avaliação e etc ...".

Outra para colher os detalhes da localidade podendo ser "Hotel, Restaurante, Ponto turistico e etc ...".

Já por outro lado a lambda com o objetivo de ocultar os dados mais sensíveis do usuário e credenciais de acesso.

AWS Function Lambda

Lambda é um serviço de computação sem servidor da AWS que permite executar código em resposta a eventos sem precisar gerenciar servidores ou infraestrutura. Ao usar o Lambda, não precisa se preocupar com capacidade, escalabilidade, disponibilidade ou manutenção de servidores. Em vez disso, a AWS cuida de tudo isso para você.

Razões pelas quais usar o Lambda na AWS:

1. Custo: Com o Lambda, você paga apenas pelo tempo que sua função leva para ser executada, sem se preocupar com o tempo de inatividade ou capacidade não utilizada. Isso pode ser muito mais econômico do que manter sua própria infraestrutura ou usar servidores em nuvem tradicionais.
2. Escalabilidade: O Lambda se dimensiona automaticamente para lidar com a carga de trabalho, portanto, você não precisa se preocupar com a capacidade ou escalabilidade de seus servidores.
3. Sem servidor: Ao não precisar gerenciar servidores, você pode se concentrar em escrever e melhorar o código de sua aplicação, em vez de se preocupar com a infraestrutura.
4. Integração com outros serviços da AWS: O Lambda se integra perfeitamente com outros serviços da AWS, permitindo que você construa aplicativos altamente escaláveis e resistentes.

Resumindo, o Lambda é uma excelente opção se você deseja executar código em resposta a eventos de maneira eficiente, econômica e sem se preocupar com a administração de servidores.

Sobre o código fonte

```

serverless.yml • serverless.yml — api
serverless Framework Configuration - Schema for serverless framework configuration files (reference.json)
1 org: traveller
2 app: aws-node-express-api-app
3 service: traveller
provider:
4   name: aws
5   timeout: 30
6   memorySize: 128
7   runtime: nodejs14.x
8   region: us-east-1
9   environment:
10     TRIPADVISOR_URL: 'https://api.content.tripadvisor.com/api/v1'
11     TRIPADVISOR_KEY: 'A0A244D2D5D8441CBE2DEFCCF1958378'
12
13   iamRoleStatements:
14     - Effect: 'Allow'
15       Action:
16         - 'codedeploy:*'
17         - 's3:}'
18       Resource: '*'
19
20
21 functions: ${file(serverless/functions.yml)}
22
23 resources:
24   Resources:
25     GatewayResponseDefault4XX:
26       Type: 'AWS::ApiGateway::GatewayResponse'
27       Properties:
28         ResponseParameters:
29           gatewayresponse.header.Access-Control-Allow-Origin: "*"
30           gatewayresponse.header.Access-Control-Allow-Headers: "*"
31         ResponseType: DEFAULT_4XX
32         RestApiId:
33           Ref: 'ApiGatewayRestApi'
34
35
36   plugins:
37     - serverless-offline

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL CODEWHISPERER REFERENCE LOG COMMENTS

dashboard: <https://app.serverless.com/elioglima/apps/aws-node-express-api-app/traveller/dev/us-east-1>

endpoints:

- GET - <https://p7167519uh.execute-api.us-east-1.amazonaws.com/dev/search>
- GET - <https://p7167519uh.execute-api.us-east-1.amazonaws.com/dev/search/photos>
- GET - <https://p7167519uh.execute-api.us-east-1.amazonaws.com/dev/search/nearby>
- GET - <https://p7167519uh.execute-api.us-east-1.amazonaws.com/dev/location/reviews>
- GET - <https://p7167519uh.execute-api.us-east-1.amazonaws.com/dev/details>

functions:

- search: traveller-dev-search (56 MB)
- searchPhotos: traveller-dev-searchPhotos (56 MB)
- searchNearby: traveller-dev-searchNearby (56 MB)
- locationReviews: traveller-dev-locationReviews (56 MB)
- details: traveller-dev-details (56 MB)

1 deprecation found: run 'serverless doctor' for more details

- * Done in 106.73s.
- elioglima in startup/traveller/api took 1m 46.9s >

Estrutura de dados dos lambdas

The screenshot shows the VS Code interface with the Serverless IDE extension open. On the left, the Explorer sidebar displays the project structure:

```

    API
      .serverless
      .vscode
      node_modules
      postman
      serverless
      src
        database
          controller.js
        lambda
          controller.js
        services
          controller.js
        utils
          db.js
          httpHelper.js
          index.js
          lambda.js
          returnDb.js
        package.json
        README.md
        serverless.yml
        yarn.lock
  
```

In the center, the Open Editors section shows two files:

- `serverless.yml`
- `functions.yml`

The `functions.yml` file is selected and its content is displayed on the right:

```

serverless > functions.yml > Serverless IDE > {} searchPhotos > [ ] events
  1  search:
  2    handler: src/lambda/controller.search
  3    events:
  4      - http:
  5        path: /search
  6        method: GET
  7        cors: false
  8
  9  searchPhotos:
 10    handler: src/lambda/controller.searchPhotos
 11    events:
 12      - http:
 13        path: /search/photos
 14        method: GET
 15
 16  searchNearby:
 17    handler: src/lambda/controller.searchNearby
 18    events:
 19      - http:
 20        path: /search/nearby
 21        method: GET
 22
 23  locationReviews:
 24    handler: src/lambda/controller.locationReviews
 25    events:
 26      - http:
 27        path: /location/reviews
 28        method: GET
 29
 30  details:
 31    handler: src/lambda/controller.details
 32    events:
 33      - http:
 34        path: /details
 35        method: GET
  
```

Implementação da Lambda x Tripadvisor

```

src > services > JS controller.js > ...
1  const axios = require('axios');
2  const utils = require('../utils');
3
4  const search = async ({ queryStringParameters }) => {
5    try {
6      const params = utils.httpHelper.objectToUri({ key: process.env.TRIPADVISOR_KEY, language:'pt', ...queryStringParameters})
7      const url = `${process.env.TRIPADVISOR_URL}/location/search${params}`;
8      const response = await axios.get(url)
9      return utils.httpHelper.ok(response.data)
10 } catch (error) {
11   return utils.httpHelper.badRequest(error.message)
12 }
13 }
14 const searchPhotos = async ({ queryStringParameters }) => {
15   try {
16     const location_id = queryStringParameters.location_id
17     delete queryStringParameters.location_id
18
19     const params = utils.httpHelper.objectToUri({ key: process.env.TRIPADVISOR_KEY, language:'pt', ...queryStringParameters})
20     const url = `${process.env.TRIPADVISOR_URL}/location/${location_id}/photos${params}`;
21     console.log(url)
22     const response = await axios.get(url)
23     return utils.httpHelper.ok(response.data)
24   } catch (error) {
25     return utils.httpHelper.badRequest(error.message)
26   }
27 }
28 const searchNearby = async ({ queryStringParameters }) => {
29   try {
30     const params = utils.httpHelper.objectToUri({ key: process.env.TRIPADVISOR_KEY, language:'pt', ...queryStringParameters})
31     const url = `${process.env.TRIPADVISOR_URL}/location/nearby_search${params}`;
32     const response = await axios.get(url)
33     return utils.httpHelper.ok(response.data)
34   } catch (error) {
35     return utils.httpHelper.badRequest(error.message)
36   }
37 }
38 const locationReviews = async ({ queryStringParameters }) => {
39   try {
40     const location_id = queryStringParameters.location_id
41     delete queryStringParameters.location_id
42     const params = utils.httpHelper.objectToUri({ key: process.env.TRIPADVISOR_KEY, language:'pt', ...queryStringParameters})
43     const url = `${process.env.TRIPADVISOR_URL}/location/${location_id}/reviews${params}`;
44     const response = await axios.get(url)
45     return utils.httpHelper.ok(response.data)
46   } catch (error) {
47     return utils.httpHelper.badRequest(error.message)
48   }
49 }
50 const details = async ({ queryStringParameters }) => {
51   try {
52     const location_id = queryStringParameters.location_id
53     delete queryStringParameters.location_id

```

Efetuando Deploy AWS

```

Deploying traveller to stage dev (us-east-1)

✓ Service deployed to stack traveller-dev (101s)

dashboard: https://app.serverless.com/elioglima/apps/aws-node-express-api-app/traveller/dev/us-east-1
endpoints:
  GET - https://p7167519uh.execute-api.us-east-1.amazonaws.com/dev/search
  GET - https://p7167519uh.execute-api.us-east-1.amazonaws.com/dev/search/photos
  GET - https://p7167519uh.execute-api.us-east-1.amazonaws.com/dev/search/nearby
  GET - https://p7167519uh.execute-api.us-east-1.amazonaws.com/dev/location/reviews
  GET - https://p7167519uh.execute-api.us-east-1.amazonaws.com/dev/details
functions:
  search: traveller-dev-search (56 MB)
  searchPhotos: traveller-dev-searchPhotos (56 MB)
  searchNearby: traveller-dev-searchNearby (56 MB)
  locationReviews: traveller-dev-locationReviews (56 MB)
  details: traveller-dev-details (56 MB)

1 deprecation found: run 'serverless doctor' for more details
  Done in 106.73s.

```

Configurações AWS Lambdas

The screenshot shows the AWS Lambda console interface. On the left, there's a sidebar with navigation links like Painel, Aplicativos, Funções, Recursos adicionais, and Recursos relacionados da AWS. The main area is titled "Funções (6)" and lists the following Lambda functions:

Nome da função	Descrição	Tipo de pacote	Tempo de execução	Última modificação
traveller-dev-locationRatings	-	Zip	Node.js 14.x	há 2 horas
traveller-dev-search	-	Zip	Node.js 14.x	há 2 horas
traveller-dev-searchNearby	-	Zip	Node.js 14.x	há 2 horas
traveller-dev-details	-	Zip	Node.js 14.x	há 2 horas
traveller-dev-searchPhotos	-	Zip	Node.js 14.x	há 2 horas
traveller-dev-custom-resource-apigw-cw-role	-	Zip	Node.js 16.x	há 2 horas

The screenshot shows the AWS Lambda function configuration page for 'traveller-dev-search'. The top navigation bar includes the AWS logo, Services, a search bar, and account information for 'Norte da Virgínia' and 'Abigail'. The main content area displays the function details:

- Visão geral da função**: Shows the function name 'traveller-dev-search' and its ARN: arnaws:lambda:us-east-1:342061705349:function:traveller-dev-search. It also lists 'Layers (0)' and an 'API Gateway' trigger.
- Descrição**: Empty.
- Última modificação**: Há 2 horas.
- ARN da função**: arnaws:lambda:us-east-1:342061705349:function:traveller-dev-search
- Aplicativo**: traveller-dev
- URL da função**: Informações

Below the general view, there are tabs for Código, Testar, Monitor, Configuração, Aliases, and Versões. The Código tab is selected. The Origem do código section contains a note: "O pacote de implantação da função do Lambda 'traveller-dev-search' é muito grande para habilitar a edição de código em linha. Contudo, você ainda pode invocar a função." A 'Fazer upload de' button is available for uploading code.

Testando Lambdas e Tripadvisor

Consulta personalizada

The screenshot shows the Postman application interface. On the left, there's a sidebar with sections like 'Traveller', 'Collections', 'APIs', 'Environments', 'Mock Servers', 'Monitors', and 'History'. The main area is titled 'Explore' and shows an 'API / Consulta' section. A 'GET Consulta' request is selected, with the URL being <https://p7167519uh.execute-api.us-east-1.amazonaws.com/dev/search?searchQuery=hotel&category=hotels&latLong=-23.537430585909647%252C%2520-46.5166998134...>. The 'Params' tab is active, showing query parameters: 'searchQuery' (value: 'hotel'), 'category' (value: 'hotels'), 'latLong' (value: '-23.537430585909647%252C%2520-46.5166998134...'), and 'language' (value: 'pt-br'). Below this, the 'Body' tab is selected, showing a JSON response with two data objects. The first object has a 'location_id' of '2406720', a name of 'Shangrila Park Hotel', a distance of '1.7533415654952575', a bearing of 'north', and an address object with street 1 as 'Avenida Sao Miguel 1745', street 2 as 'Penha', city as 'São Paulo', state as 'Estado de São Paulo', country as 'Brasil', postal code as '03619-100', and address string as 'Avenida Sao Miguel 1745 Penha, São Paulo, Estado de São Paulo 03619-100 Brasil'. The second object has a 'location_id' of '6746069', a name of 'Ceasar Palace Hotel', a distance of '1.8287825950304384', a bearing of 'southeast', and an address object with street 1 as 'Avenida Professor Edgar Santos 780', street 2 as 'Vl Nhocene', city as 'São Paulo', state as 'Estado de São Paulo', country as 'Brasil', postal code as '03560-080', and address string as 'Avenida Professor Edgar Santos 780 Vl Nhocene, São Paulo, Estado de São Paulo 03560-080 Brasil'. At the bottom, there are buttons for 'Cookies', 'Capture requests', 'Runner', 'Trash', and a refresh icon.

Consulta de fotos por localidade

The screenshot shows the Postman application interface. On the left, there's a sidebar with sections like 'Traveller', 'Collections', 'APIs', 'Environments', 'Mock Servers', and 'Monitors'. The main area is titled 'API / Consulta Foto' and shows a GET request to 'https://p7167519uh.execute-api.us-east-1.amazonaws.com/dev/search/photos?location_id=2406720'. The 'Params' tab is selected, showing a query parameter 'location_id' with value '2406720'. Below the request, the 'Body' tab displays a JSON response with code highlighting. The response body is:

```
1  "data": [
2    {
3      "id": 155720408,
4      "is_blessed": false,
5      "caption": "Quarto 311",
6      "published_date": "2015-10-21T16:15:32.405Z",
7      "images": {
8        "thumbnail": {
9          "height": 50,
10         "width": 50,
11         "url": "https://media-cdn.tripadvisor.com/media/photo-t/09/48/1a/d8/quarto-311.jpg"
12       },
13       "small": {
14         "height": 150,
15         "width": 150,
16         "url": "https://media-cdn.tripadvisor.com/media/photo-l/09/48/1a/d8/quarto-311.jpg"
17     },
18     "medium": {
19       "height": 205,
20       "width": 154,
21       "url": "https://media-cdn.tripadvisor.com/media/photo-f/09/48/1a/d8/quarto-311.jpg"
22   },
23   "large": {
24     "height": 450,
25     "width": 338,
26     "url": "https://media-cdn.tripadvisor.com/media/photo-s/09/48/1a/d8/quarto-311.jpg"
27 },
28   "original": {
29     "height": 2048,
30     "width": 1536,
31     "url": "https://media-cdn.tripadvisor.com/media/photo-o/09/48/1a/d8/quarto-311.jpg"
32 }
33 },
34   "album": "Quarto/suite",
35 ]
```

Consulta de avaliações

The screenshot shows the Postman application interface. On the left, there's a sidebar with sections for Collections, APIs, Environments, Mock Servers, and Monitors. The main area shows a 'Traveller' collection with several API endpoints listed under 'API'. One endpoint, 'GET location-reviews', is selected. The request details show a GET method to the URL https://p7167519uh.execute-api.us-east-1.amazonaws.com/dev/location/reviews?location_id=2406720. The 'Params' tab is active, showing a single parameter 'location_id' with the value '2406720'. The 'Body' tab shows a JSON response with a single data item. The response body is as follows:

```

1
2   "data": [
3     {
4       "id": 502743568,
5       "lang": "pt",
6       "location_id": 2406720,
7       "published_date": "2017-07-17T06:51:32Z",
8       "rating": 1,
9       "helpful_votes": 3,
10      "rating_image_url": "https://www.tripadvisor.com/img/cdsi/img2/ratings/traveler/s1.0-66827-5.svg",
11      "url": "https://www.tripadvisor.com.br/
12        ShowUserReviews-x303631-d2406720-r502743568-Reviews-Shangrila_Park_Hotel-Sao_Paulo_State_of_Sao_Pa
13        ulo.html?m=66827#review502743568",
14      "text": "A senha do wifi nao funciona, o ar condicionado \u00e9 virado para fora, com a janela aberta os
15      pernilongos avancaram para o quarto. Formigas no banheiro e teto. \n\nO cafe no primeiro dia tinha
16      suco e no segundo nao.",
17      "title": "Sem ar condicionado, sem wifi, muito pernilongo e formigas.",
18      "trip_type": "Familia",
19      "travel_date": "2017-07-31",
20      "user": {
21        "username": "gabriellam315",
22        "user_location": {
23          "id": "303631",
24          "name": "Sao Paulo, State of Sao Paulo"
25        },
26        "avatar": {
27          "thumbnail": "https://media-cdn.tripadvisor.com/media/photo-t/1a/f6/df/2b/
28            default-avatar-2020-39.jpg",
29          "small": "https://media-cdn.tripadvisor.com/media/photo-l/1a/f6/df/2b/default-avatar-2020-39.
30            jpg",
31          "medium": "https://media-cdn.tripadvisor.com/media/photo-f/1a/f6/df/2b/default-avatar-2020-39.
32            jpg",
33          "large": "https://media-cdn.tripadvisor.com/media/photo-p/1a/f6/df/2b/default-avatar-2020-39.
34            jpg",
35          "original": "https://media-cdn.tripadvisor.com/media/photo-o/1a/f6/df/2b/
36            default-avatar-2020-39.jpg"
37        }
38      }
39    }
40  ]

```

Pesquisa loclidades nas proximidade de uma determinada geolocalização

The screenshot shows the Postman application interface. On the left, there's a sidebar with sections for Collections, APIs, Environments, Mock Servers, Monitors, and History. The main area is titled "API / Pesquisa Proximidades". A GET request is selected with the URL: <https://p7167519uh.execute-api.us-east-1.amazonaws.com/dev/search/nearby?latLong=-23.53752894777355%2C%20-46.51669981349255&language=pt>. The "Params" tab is active, showing two parameters: "latLong" with value "-23.53752894777355%2C%20-46.51669981349255" and "language" with value "pt". Below this, the "Body" tab is selected, showing a JSON response with two objects under "data":

```
1  "data": [
2    {
3      "location_id": "23047222",
4      "name": "Call Center Burguer",
5      "distance": "0.3284669829858449",
6      "bearing": "east",
7      "address_obj": {
8        "street1": "Praça Nelson Sales De Abreu, 161",
9        "street2": "",
10       "city": "São Paulo",
11       "state": "Estado de São Paulo",
12       "country": "Brasil",
13       "postalcode": "03547-100",
14       "address_string": "Praça Nelson Sales De Abreu, 161, São Paulo, Estado de São Paulo 03547-100 Brasil"
15     }
16   },
17   {
18     "location_id": "5330699",
19     "name": "Estacao Das Pizzas",
20     "distance": "0.3102157975760826",
21     "bearing": "east",
22     "address_obj": {
23       "street1": "Praca Nelson Sales de Abreu 103",
24       "street2": "Centro",
25       "city": "São Paulo",
26       "state": "Estado de São Paulo",
27       "country": "Brasil",
28       "postalcode": "03547-100",
29       "address_string": "Praca Nelson Sales de Abreu 103 Centro, São Paulo, Estado de São Paulo 03547-100 Brasil"
30     }
31   }
32 ]
```

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'Traveller' selected under 'Collections'. Below it are sections for 'APIs', 'Environments', 'Mock Servers', 'Monitors', and 'History'. The main area is titled 'API / Detalhes' and shows a 'GET' request to 'https://p7167519uh.execute-api.us-east-1.amazonaws.com/dev/details?location_id=2406720'. The 'Params' tab is active, showing a single parameter 'location_id' with value '2406720'. The 'Body' tab displays the JSON response:

```

1
2   "location_id": "2406720",
3   "name": "Shangrila Park Hotel",
4   "web_url": "https://www.tripadvisor.com.br/
5     Hotel_Review-g303631-d2406720-Reviews-Shangrila_Park_Hotel-Sao_Paulo_State_of_Sao_Paulo.html?m=66827",
6   "address_obj": {
7     "street1": "Avenida Sao Miguel 1745",
8     "street2": "Penha",
9     "city": "S\u00e3o Paulo",
10    "state": "Estado de S\u00e3o Paulo",
11    "country": "Brasil",
12    "postalcode": "03619-100",
13    "address_string": "Avenida Sao Miguel 1745 Penha, S\u00e3o Paulo, Estado de S\u00e3o Paulo 03619-100 Brasil"
14  },
15  "ancestors": [
16    {
17      "level": "City",
18      "name": "S\u00e3o Paulo",
19      "location_id": "303631"
20    },
21    {
22      "level": "State",
23      "name": "Estado de S\u00e3o Paulo",
24      "location_id": "303598"
25    },
26    {
27      "level": "Country",
28      "name": "Brasil",
29      "location_id": "294280"
30    }
31  ],
32  "latitude": "-23.51219",
33  "longitude": "-46.51415",
34  "timezone": "America/Sao_Paulo",
35  "write_review": "https://www.tripadvisor.com.br/

```

The status bar at the bottom indicates 'Status: 200 OK Time: 2.75 s Size: 3.29 KB'. The bottom navigation bar includes links for 'Online', 'Find and Replace', 'Console', 'Cookies', 'Capture requests', 'Runner', 'Trash', and 'Help'.

Sobre o código do website

<https://github.com/abigailvlima/traveller>

<https://github.com/abigailvlima/traveller>

The screenshot shows a GitHub repository page for the 'traveller' repository. The main content includes:

- Code** tab: Shows 1 branch (main) and 0 tags. A list of commits from 'abigailvlima' is displayed, including updates to README.md, .vscode, api, css, images, js, .gitignore, and index.html.
- About** section: No description, website, or topics provided. Statistics: 0 stars, 1 watching, 0 forks.
- Releases** section: No releases published. Create a new release.
- Packages** section: No packages published. Publish your first package.
- Environments** section: 1 environment, 'github-pages' (Active).
- Languages** chart: SCSS 37.0%, Less 36.5%, JavaScript 9.5%, CSS 9.4%, HTML 7.6%.

Below the main repository page, there is a smaller screenshot of the 'Settings' tab for the same repository, specifically showing the GitHub Pages configuration. It indicates that the site is live at <https://abigailvlima.github.io/traveller>.

The screenshot shows the Visual Studio Code (VS Code) interface. The left sidebar contains icons for file operations, GitHub integration, AWS services, and Kubernetes resources. The main area has three tabs: 'load.js' (selected), 'services.js', and 'loadListCards'. The code editor displays two files: 'load.js' and 'services.js'. The 'load.js' file contains a function to fetch data from an AWS Lambda endpoint. The 'services.js' file contains functions for editing cards and loading lists of cards. Below the editor is a terminal window showing a git commit message and a successful push to a GitHub repository.

```

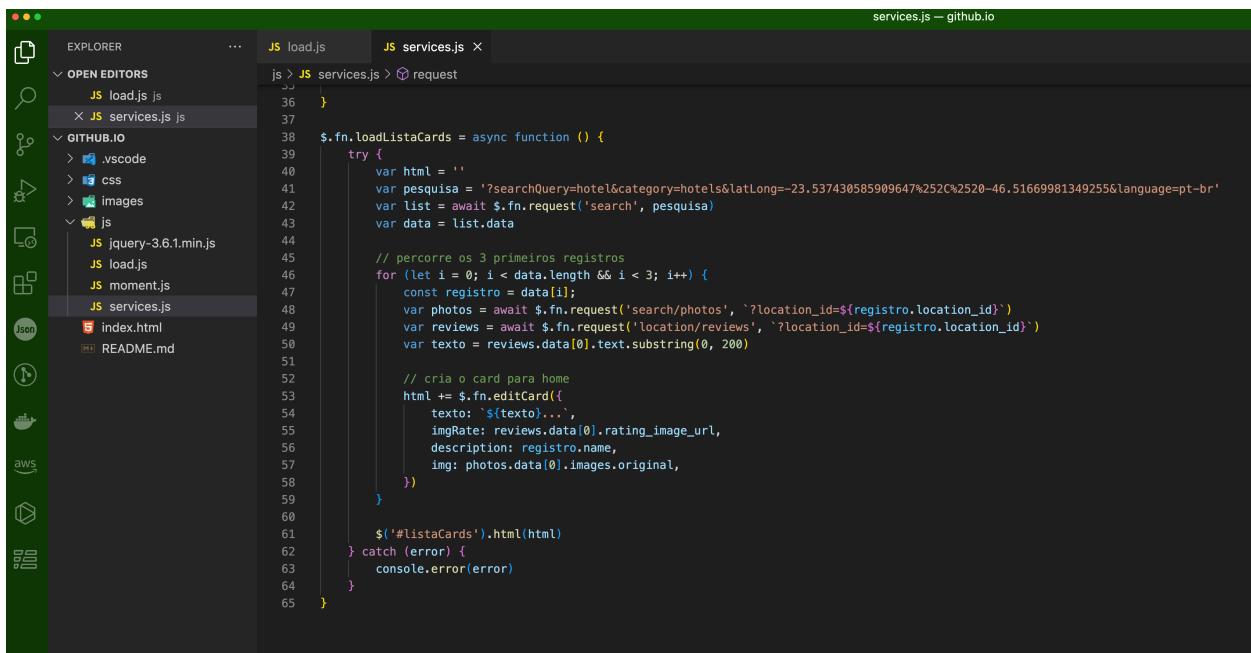
load.js
js > JS services.js > loadListCards
You, 4 minutes ago | 1 author (You)
1  $.url_service = "https://p7167519uh.execute-api.us-east-1.amazonaws.com/dev";
2
3  $fn.request = async function (uri, dataString) {
4      const url = `${$.url_service}/${uri}${dataString}`;
5      const response = await fetch(url, {
6          method: 'GET',
7          withCredentials: true,
8          crossorigin: true,
9      })
10
11     if (!response.ok) {
12         console.error('error', response, await response.json())
13         return;
14     }
15
16     return await response.json()
17 }
18
19
20 $.fn.editCard = function ({ texto, description, img, imgRate }) {
21     return `
22         <div class="box-hotel">
23             
24             <div class="stars">
25                 
26             </div>
27             <div class="foto">
28                 <div>
29                     <span>${description}</span>
30                     <p>${texto}</p>
31                 </div>
32             </div>
33         `;
34
35
36 }
37
38 $.fn.loadListaCards = async function () {
39     try {

```

```

[main f75577f] feat(Fix): ajust coment
  1 file changed, 5 deletions(-)
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 10 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 359 bytes | 359.00 KiB/s, done.
Total 4 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
remote: This repository moved. Please use the new location:
remote:   https://github.com/abigailvlima/traveller.git
To https://github.com/abigailvlima/github.io.git/
  0daf757..f75577f  main -> main

```



```
services.js — github.io

OPEN EDITORS
  JS load.js
  JS services.js
GITHUB.IO
  .vscode
  css
  images
  js
    jquery-3.6.1.min.js
    load.js
    moment.js
    services.js
  index.html
  README.md

services.js
js > JS services.js > request
35
36  }
37
38  $.fn.loadListaCards = async function () {
39    try {
40      var html = ''
41      var pesquisa = '?searchQuery=hotel&category=hoteis&latLong=-23.537430585909647%2520-46.51669981349255&language=pt-br'
42      var list = await $.fn.request('search', pesquisa)
43      var data = list.data
44
45      // percorre os 3 primeiros registros
46      for (let i = 0; i < data.length && i < 3; i++) {
47        const registro = data[i];
48        var photos = await $.fn.request('search/photos', `?location_id=${registro.location_id}`)
49        var reviews = await $.fn.request('location/reviews', `?location_id=${registro.location_id}`)
50        var texto = reviews.data[0].text.substring(0, 200)
51
52        // cria o card para home
53        html += $.fn.editCard({
54          texto: `${texto}...`,
55          imgRate: reviews.data[0].rating_image_url,
56          description: registro.name,
57          img: photos.data[0].images.original,
58        })
59      }
60
61      $('#listaCards').html(html)
62    } catch (error) {
63      console.error(error)
64    }
65  }


```

Repositório github site

- <https://github.com/abigailvlima/traveller>

Publicação site no github.io

- <https://abigailvlima.github.io/gullivertravell/>

The screenshot shows the GitHub Pages settings page for the repository `abigailvlima/traveller`. The left sidebar contains navigation links for General, Access, Collaborators, Moderation options, Code and automation (Branches, Tags, Actions, Webhooks, Environments, Codespaces, Pages), Security (Code security and analysis, Deploy keys, Secrets and variables), and Integrations (GitHub Apps, Email notifications). The main content area is titled "GitHub Pages" and includes sections for "Build and deployment" (Source: Deploy from a branch, Branch: main, / (root), Save), "Custom domain" (Custom domain: abigailvlima.github.io, Save, Remove), and "Visibility" (Enforce HTTPS checked, Required for your site because you are using the default domain (abigailvlima.github.io)). A note states: "With a GitHub Enterprise account, you can restrict access to your GitHub Pages site by publishing it privately. A privately published site can only be accessed by people with read access to the repository the site is published from. You can use privately published sites to share your internal documentation or knowledge base with members of your enterprise." Buttons for "Try GitHub Enterprise risk-free for 30 days" and "Learn more about the visibility of your GitHub Pages site" are also present.

Site Publicado

O que os hóspedes estão falando sobre as acomodações

Mais de 490.000 avaliações com uma média de 4.8 de 5 estrelas



●○○○○

Shangrila Park Hotel

A senha do wifi não funciona, o ar condicionado é virado para fora, com a janela aberta os pernilongos avançaram para o quarto. Formigas no banheiro e teto. O café no primeiro dia tinha suco e no se...



●●●●●

Cesar Palace Hotel

Eu classifico como ótimo clássico de Vegas, Quarto com espaço para casal, com ante-sala e banheiro espaço... Índico com certeza para família, ótimos restaurantes no complexo do hotel de fácil acesso, ...



●○○○○

OYO Hotel Park Leste

Hotel é ruim, não dá para família hospedar. Fizemos reserva, quando chegamos, tivemos que ir para outro local, ainda tivemos que pagar taxa de cancelamento. Infelizmente consta no site como hotel.....

Reserve no Traveller e viaje com tranquilidade

