REPÚBLICA FEDERATIVA DO BRASIL

MINISTÉRIO DA ECONOMIA

INSTITUTO NACIONAL DA PROPRIEDADE INDUSTRIAL

DIRETORIA DE PATENTES, PROGRAMAS DE COMPUTADOR E TOPOGRAFIAS DE CIRCUITOS INTEGRADOS

Certificado de Registro de Programa de Computador

Processo Nº: **BR512019001657-2**

O Instituto Nacional da Propriedade Industrial expede o presente certificado de registro de programa de computador, válido por 50 anos a partir de 1° de janeiro subsequente à data de 30/07/2019, em conformidade com o §2°, art. 2° da Lei 9.609, de 19 de Fevereiro de 1998.

Título: MAPI - API para manipulação e consulta para base de dados georreferenciados

Data de publicação: 30/07/2019

Data de criação: 10/07/2019

Titular(es): ERIC AMARAL FERREIRA; HITALO VINICIUS ALVES DE ANDRADE

Autor(es): VICTOR MARCELO RODRIGUES SILVA; HITALO VINICIUS ALVES DE ANDRADE; SEBASTIÃO EMIDIO ALVES FILHO

Linguagem: JAVA SCRIPT

Campo de aplicação: AH-06; GC-02; IF-01; TP-01; UB-04

Tipo de programa: AP-02; FA-01; GI-01; TC-01

Algoritmo hash: SHA-512

Resumo digital hash:
25043413514b8ebdac7f1c653486ba713ecaadd051c33448302f0998603a0f99e1a86385876ca390d6dc7195b6754d57a0b5baeed8eecd7f6ce801d37bf3bfa7

Expedido em: 06/08/2019

Aprovado por:
Helmar Alvares
Chefe da DIPTO - Portaria/INPI/DIRPA Nº 09, de 01 de julho de 2019

UNIVERSIDADE DO ESTADO DO RIO GRANDE DO NORTE – UERN

FACULDADE DE CIÊNCIAS EXATAS E NATURAIS – FANAT

DEPARTAMENTO DE INFORMÁTICA – DI

Hitalo Vinicius Alves de Andrade

MAPI - Uma API REST para manipulação e consulta para base de dados georreferenciados

MOSSORÓ - RN
2019

HITALO VINICIUS ALVES DE ANDRADE

MAPI - Uma API REST para manipulação e consulta para base de dados georreferenciados

Monografia apresentada como pré-requisito para a obtenção do título de Bacharel em Ciência da Computação da Universidade do Estado do Rio Grande do Norte – UERN, submetida à aprovação da banca examinadora composta pelos seguintes membros:

Aprovada em: 26 / 09 / 2019

Banca Examinadora

Prof. Dr. Sebastião Emídio Alves Filho
Universidade do Estado do Rio Grande do Norte - UERN

Prof. Dr. Marcelino Pereira dos Santos Silva
Universidade do Estado do Rio Grande do Norte - UERN

Prof. Dr. Francisco Chagas de Lima Júnior
Universidade do Estado do Rio Grande do Norte - UERN

RESUMO

Com o crescente volume de dados associados ao termo Big Data e imensos conjuntos de dados variados que auxiliam as tomadas de decisões, um tipo de ferramenta vem ganhando cada vez mais espaço no âmbito da administração pública: os sistemas de informações geográficas. Nesse contexto, este trabalho apresenta uma API REST para manipulação e consulta para bases de dados georreferenciados, idealizada a partir da observação de como a Prefeitura Municipal de Mossoró - RN armazena os dados de acidentes de trânsito e ocorrências do SAMU. A API desenvolvida compõe um SIG que pode auxiliar nas tomadas de decisões, assim como permitir o armazenamento de dados georreferenciados de forma mais eficaz, além disso a API desenvolvida permite que aplicações implementadas em diferentes plataformas e linguagens de programação consumam os seus serviços. A API utiliza o Node.js como principal tecnologia e faz uso do framework Express. O banco de dados utilizado na aplicação trata-se do ArangoDB, um banco de dados NoSQL multi modelo que traz por padrão funções voltadas a dados georreferenciados. A validação foi feita através de testes utilizando a ferramenta Postman e a aplicação MAPP, onde foram obtidos os resultados esperados da eficiência da API.

Palavras-chave: Sistemas de Informações Geográficas, Georreferenciamento, API REST, web services, sistemas distribuídos.

Sumário

1. Introdução

Um dos motivos que torna a administração municipal uma tarefa árdua é o rápido processo de urbanização das cidades, além do uso de técnicas obsoletas de armazenar, organizar e manipular informações. Neste contexto, um grande número de municípios está voltado à modernização, buscando ferramentas mais eficazes para eliminar as deficiências administrativas.

Os sistemas de informações geográficas (SIG) vem para modernizar o armazenamento das informações, processar grandes volumes de dados e auxiliar nas tomadas de decisões. As aplicações são variadas abordando atividades como uso e ocupação do solo, planejamento urbano, saúde, transportes e trânsito, infra-estrutura urbana (redes de energia elétrica, abastecimento de água, esgotamento sanitário), policiamento, entre outras.

A partir da observação de como a Prefeitura Municipal de Mossoró armazena os dados de acidentes de trânsito e ocorrências do SAMU, foi proposta uma API REST para manipulação e consultas em bases de dados georreferenciados, esta API compõe um SIG que pode auxiliar nas tomadas de decisões, como posicionamento de veículos do SAMU em locais estratégicos e intervenções na malha viária para diminuição de acidentes de trânsito.

2. Referencial Teórico

2.1. SIG

Os Sistemas de Informações Geográficas (SIG) são ferramentas de apoio à tomada de decisões e à manipulação de informações planejadas. Burrough e McDonnell (2015) retratam SIG como uma tecnologia computacional, desenvolvida com o intuito de capturar, armazenar, manipular e visualizar dados georreferenciados.

De acordo com Câmara e Medeiros (1998), Um SIG pela sua formação, possibilita abranger diversas aplicações em vários segmentos como: atualizações florestais; administração municipal e planejamento urbano; mapeamento de solos; monitoramento de bacias hidrográficas; gestão de redes de distribuição de água e coleta de esgoto.

Para Câmara (2005), o termo sistemas de informação geográfica (SIG) é aplicado para sistemas que realizam o tratamento computacional de dados geográficos. A principal diferença de um SIG para um sistema de informação convencional é sua capacidade de armazenar tanto os atributos descritivos como as geometrias dos diferentes tipos de dados geográficos.

2.2. API REST

API (Application Programming Interface) é um conjunto de rotinas e padrões estabelecidos e documentados por uma aplicação, para que outras aplicações consigam utilizar as funcionalidades desta aplicação sem precisar conhecer detalhes da implementação do software.

REST (Representational State Transfer) trata-se de uma abstração da arquitetura da Web. Resumidamente, o REST consiste em princípios e regras que permitem a criação de

um projeto com interfaces bem definidas. Desta forma, permitindo, por exemplo, que aplicações se comuniquem.

REST trata qualquer tipo de serviço ou informação como um recurso. Os recursos são identificados por um Identificador Uniforme de Recurso do inglês Uniform Resource Identifier (URI) que são acessados por meio do Protocolo de Transferência de Hipertexto do inglês Hypertext Transfer Protocol (HTTP) (SILVA, 2017).

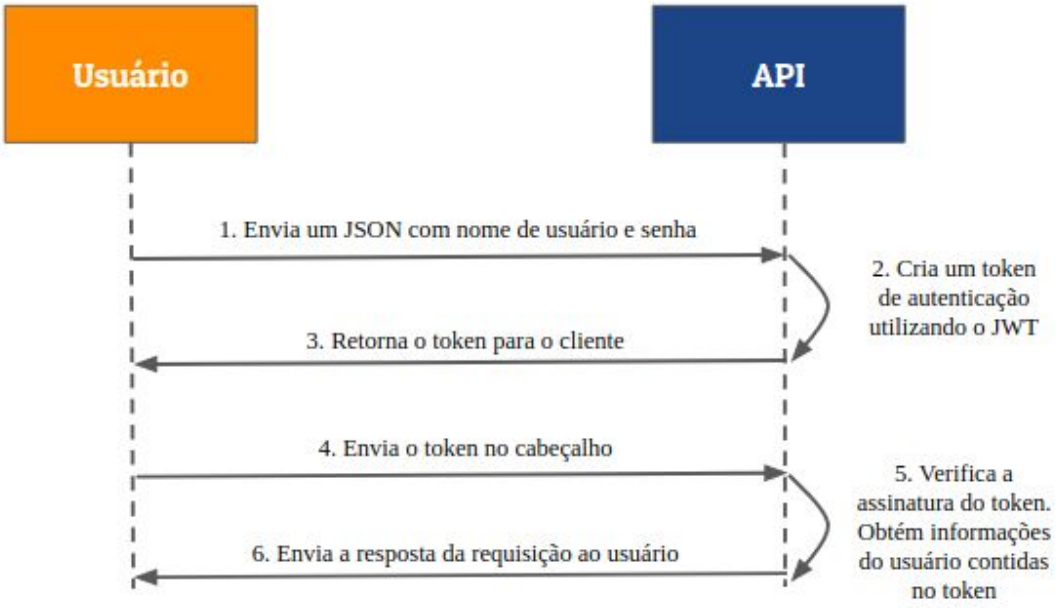
3. Descrição geral da API

MAPI trata-se de uma API que faz uso do modelo REST para manipular dados georreferenciados, assim como efetuar consultas utilizando métodos de filtragem. O principal objetivo da aplicação é facilitar a disponibilização de dados relevantes para uma cidade. A aplicação, em um contexto geral, faz parte de um SIG voltado ao uso de dados georreferenciados da cidade de Mossoró-RN, como ocorrências médicas, acidentes de trânsito e informações de ruas.

A Figura 1 exemplifica como é feita a autenticação do usuário utilizando o padrão *Json Web Token*, onde as credenciais do usuário são mescladas com uma palavra chave para criar um Token de autenticação. Também é apresentado como é permitido ao usuário obter uma resposta a sua requisição.

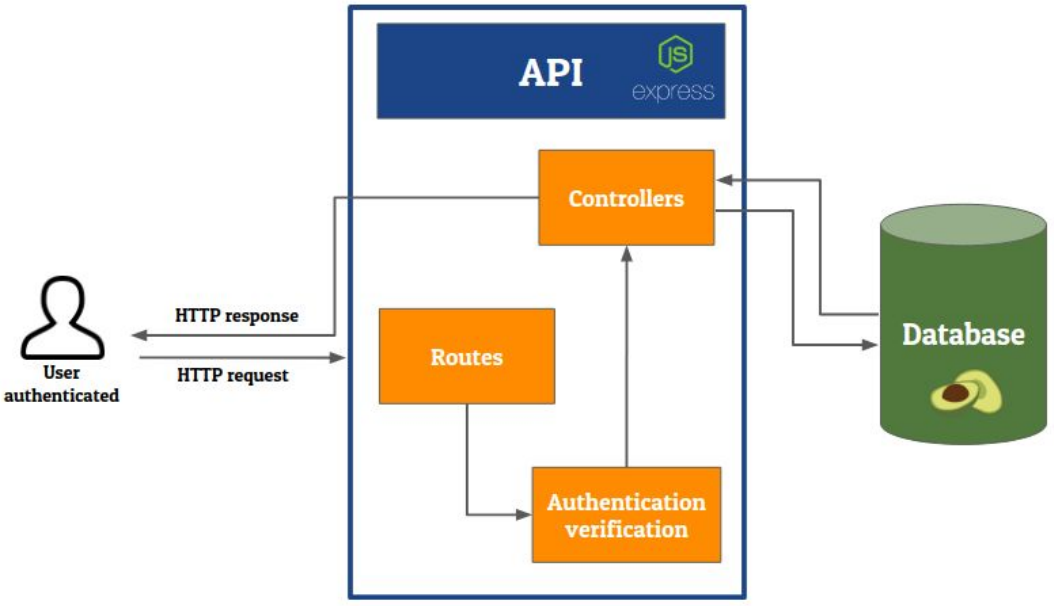
A Figura 2 apresenta a visão geral do funcionamento da API, onde um usuário já autenticado envia uma requisição HTTP para a API. É feita a verificação da rota, se a rota for válida é efetuada a verificação da autenticação do usuário. Após isso o *controller* é responsável por verificar se os parâmetros passados na requisição são válidos, uma busca é feita na base de dados com base nos parâmetros passados, ao obter um resultado o *controller* envia uma resposta ao usuário.

Figura 1: Autenticação do usuário



Fonte: Autoria própria

Figura 2: Visão geral do funcionamento da API



Fonte: Autoria própria

3.1. Rotas

A Tabela 1 apresenta as rotas pertencentes a API, onde as rotas destacadas são acessíveis apenas aos usuários administradores.

Tabela 1: Rotas da API

Método HTTP	Rota	Descrição
GET	.../accidents	Retorna todos os acidentes
GET	.../accidents/*:key*	Retorna um acidente específico
POST	.../accidents	Insere acidente
POST	.../accidents/point	Retorna acidentes usando filtros
POST	.../accidents/heat	Retorna acidentes usando filtros (Padrão de mapa de calor do Leaflet)
GET	.../bus-routes	Retorna todas as rotas de ônibus
GET	.../bus-routes/*:key*	Retorna rota de ônibus específica

GET	.../bus-stops	Retorna todos os pontos de ônibus
GET	.../bus-stops/*:key*	Retorna ponto de ônibus específico
GET	.../streets	Retorna todas as ruas
GET	.../streets/*:key*	Retorna rua específica
GET	.../traffic-lights	Retorna todos os semáforos
GET	.../traffic-lights/*:key*	Retorna semáforo específico
GET	.../medical-occurrences	Retorna todas as ocorrências médicas
GET	.../medical-occurrences*/:key*	Retorna ocorrência médica específica
POST	.../medical-occurrences	Insere ocorrência médica
POST	.../medical-occurrences/point	Retorna ocorrências médicas usando filtros
POST	.../medical-occurrences/heat	Retorna ocorrências médicas usando filtros (Padrão de mapa de calor do Leaflet)
GET	.../user	Retorna todos os usuários
GET	.../user/*:key*	Retorna um usuário específico
POST	.../user	Insere usuário
POST	.../login	Retorna um token válido para a autenticação

3.2. GeoJson

O formato dos dados utilizados na aplicação seguem o padrão de uma variação do JSON (JavaScript Object Notation), o GeoJSON. GeoJSON é um formato de dados projetado para representar características geográficas simples.

As Figuras 3 e 4 apresentam exemplos de dados no formato GeoJSON, onde pode-se observar que um geoJSON pode suportar diferentes tipos de dados geográficos. O

GeoJSON suporta os seguintes tipos de geometria: *Point* , *LineString* , *Polygon* , *MultiPoint* , *MultiLineString* e *MultiPolygon* .

Figura 3: Exemplo de GeoJSON com dados de uma ocorrência médica


```
{
  "type": "Feature",
  "geometry": {
    "type": "Point",
    "coordinates": [
      -37.33814000000001,
      -5.178889
    ]
  },
  "properties": {
    "data": "2017-02-03",
    "endereco": "Rua Epitácio Pessoa, 416 - Barrocas, Mossoró - RN, 59618-730, Brasil",
    "hora": "07:50",
    "latitude": "-5.178889",
    "longitude": "-37.33814000000001",
    "motivo": "Socorro",
    "numero": "00903022017",
    "tipo": "Psiquiátrico"
  }
}
```


Fonte: Autoria própria

Figura 4: Exemplo de GeoJSON com dados de uma rua em Mossoró-RN


```
{
  "type": "Feature",
  "properties": {
    "nomeUsual": "Rua Professor Mauricio De Oliveira",
    "nomeOficial": "Rua Professor Mauricio De Oliveira",
    "meioFio": "No Existe",
    "tipoPavimentacao": "Nenhum",
    "sentidoVia": "Duplo",
    "velocidade": 30,
    "distancia": 77.544
  },
  "geometry": {
    "type": "MultiLineString",
    "coordinates": [
      [
        [
          -37.3215285477527,
          -5.203981260720716
        ],
        [
          -37.321147868492986,
          -5.20339298660845
        ]
      ]
    ]
  }
}
```


Fonte: Autoria própria

4. Tecnologias utilizadas

4.1. Node.js

Node.js é uma plataforma construída sobre o motor JavaScript do Google Chrome para facilmente construir aplicações de rede rápidas e escaláveis. O Node.js foi criado por Ryan Dahl com objetivo de resolver os problemas das arquiteturas bloqueantes. Uma

arquitetura bloqueante enfileira as requisições e posteriormente processa uma por uma, não possibilitando o processamento de várias delas ao mesmo tempo.

O Node utiliza o NPM (node package manager) como gerenciador de pacotes e bibliotecas, que por sua vez é o maior ecossistema de bibliotecas open source do mundo.

4.2. Express

Express é um framework para Node.js minimalista e flexível que fornece um conjunto robusto de recursos para aplicativos web e móvel. Com uma miríade de métodos utilitários HTTP e middlewares a seu dispor, o Express facilita a criação de APIs robustas fornecendo uma camada fina de recursos fundamentais para aplicativos da web, sem conflitar com os recursos nativos do Node.js.

4.3. ArangoDB

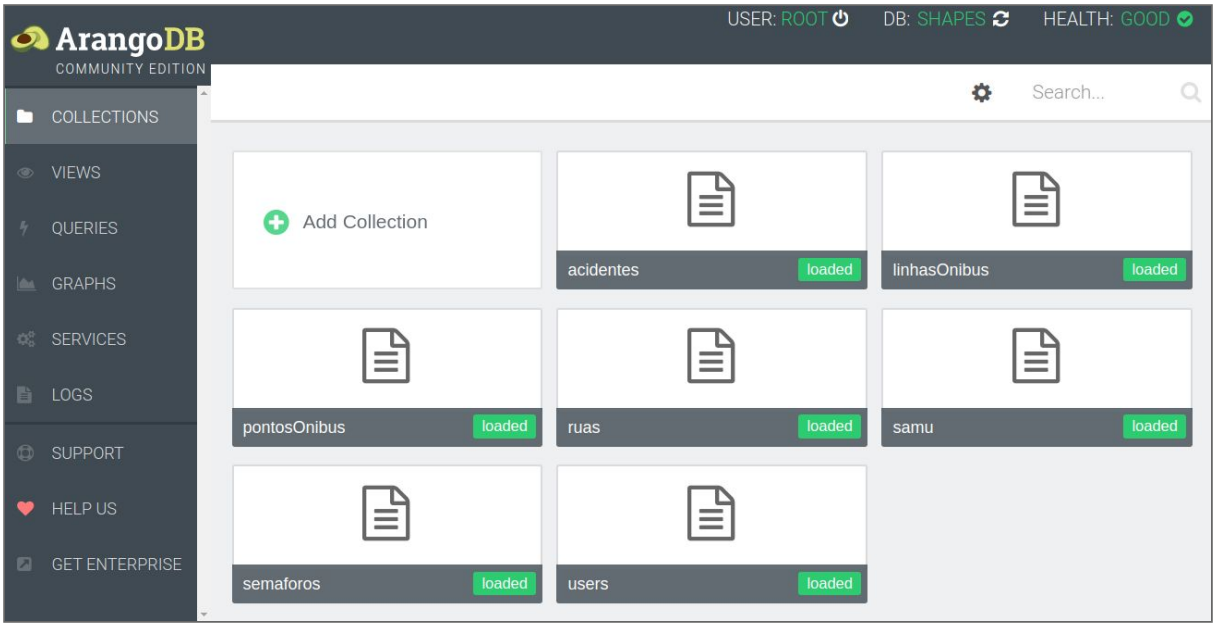
ArangoDB é um sistema de banco de dados NoSQL multi-modelo e livre de esquema que suporta três modelos de dados importantes, chave-valor, orientado a documentos e orientados a grafos. O ArangoDB possui uma linguagem de consulta unificada, a AQL. O idioma da consulta é declarativo e permite a combinação de diferentes padrões de acesso a dados em uma única consulta.

A estrutura dos dados no ArangoDB é organizada em coleções, que podem ser de dois tipos distintos: documentos e arestas. Cada coleção pode conter documentos, que por sua vez podem conter atributos, os atributos podem receber valores dos seguintes tipos: number, boolean, string, null, array e object. Arrays e objetos podem conter qualquer um dos tipos citados, permitindo assim que estruturas de dados possam ser representadas em um único documento.

A escolha do ArangoDB foi motivada pela sua versatilidade, além de ser multi-modelo o ArangoDB, a partir da versão 3.4, passou a dar suporte completo ao GeoJSON, disponibilizando funções geográficas que encaixaram-se perfeitamente com a proposta da aplicação. Além de retornar dados nos formatos JSON e em tabela, o ArangoDB passou a renderizar dados baseados no GeoJSON utilizando a biblioteca Leaflet e OpenStreetMap.

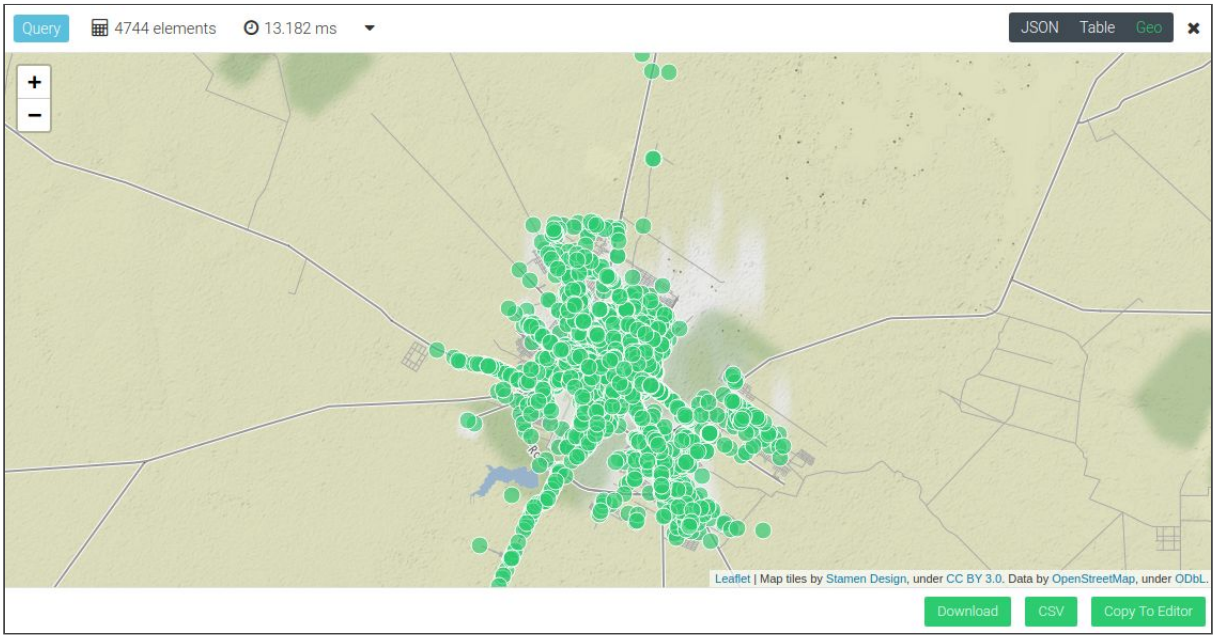
A Imagem 5 apresenta a interface web do ArangoDB, assim como as coleções utilizadas. Nas imagens 6 e 7 é apresentada a renderização de dados baseados no GeoJSON.

Figura 5: Coleções utilizadas na aplicação



Fonte: Autoria própria

Figura 6: Acidentes de trânsito registrados na cidade de Mossoró-RN entre os anos de 2014 e 2017



Fonte: Autoria própria

Figura 7: Ruas da cidade de Mossoró

Fonte: Autoria própria

5. Prova de conceito

5.1. Postman

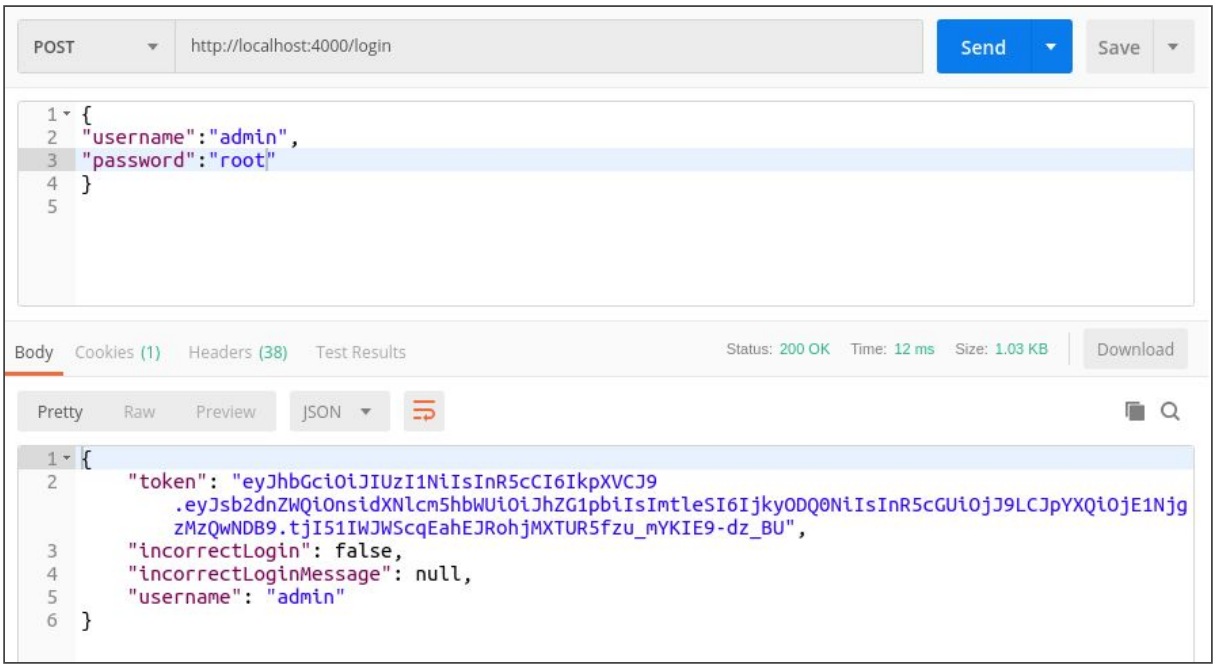
Todas as funcionalidades da API foram testadas e validadas utilizando a ferramenta Postman. O Postman é uma ferramenta que tem como objetivo efetuar testes em serviços REST (Web APIs) através do envio de requisições HTTP e por meio da análise do seu retorno. Com o Postman é possível consumir de forma simples serviços locais, assim como serviços na internet, enviando requisições e efetuando testes sobre suas respostas.

A Figura 8 apresenta o funcionamento do envio de requisição para a obtenção do token de autenticação, onde é enviada uma requisição POST com os parâmetros necessários para efetuar a verificação de validação para a rota */login*, caso o usuário seja válido a resposta da requisição conterá o token de autenticação junto a outros parâmetros que indicam o sucesso ou fracasso da requisição.

Após estar autenticado, o usuário tem permissão para enviar requisições referentes aos dados geográficos. A Figura 9 apresenta uma requisição GET enviada a rota */accidents/691820* onde *"691820"* é o identificador do GeoJSON que foi a resposta da requisição.

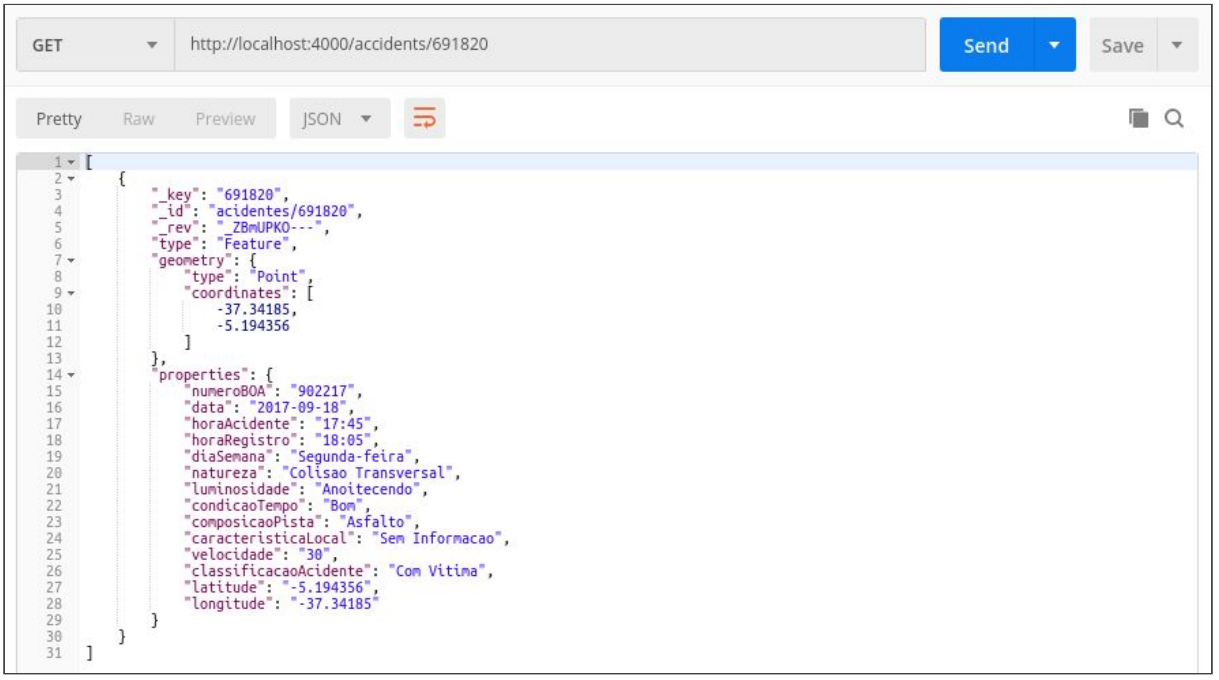
Requisições podem também ser feitas com a utilização de filtros, a Figura 10 apresenta um exemplo onde um corpo é enviado em uma requisição para a rota */accidents/point* contendo as especificações do GeoJSON que será obtido, com base nesse corpo a API retornará um GeoJSON contendo apenas os acidentes que possuem as características definidas no corpo da requisição.

Figura 8: Requisição para a obtenção do token de autenticação



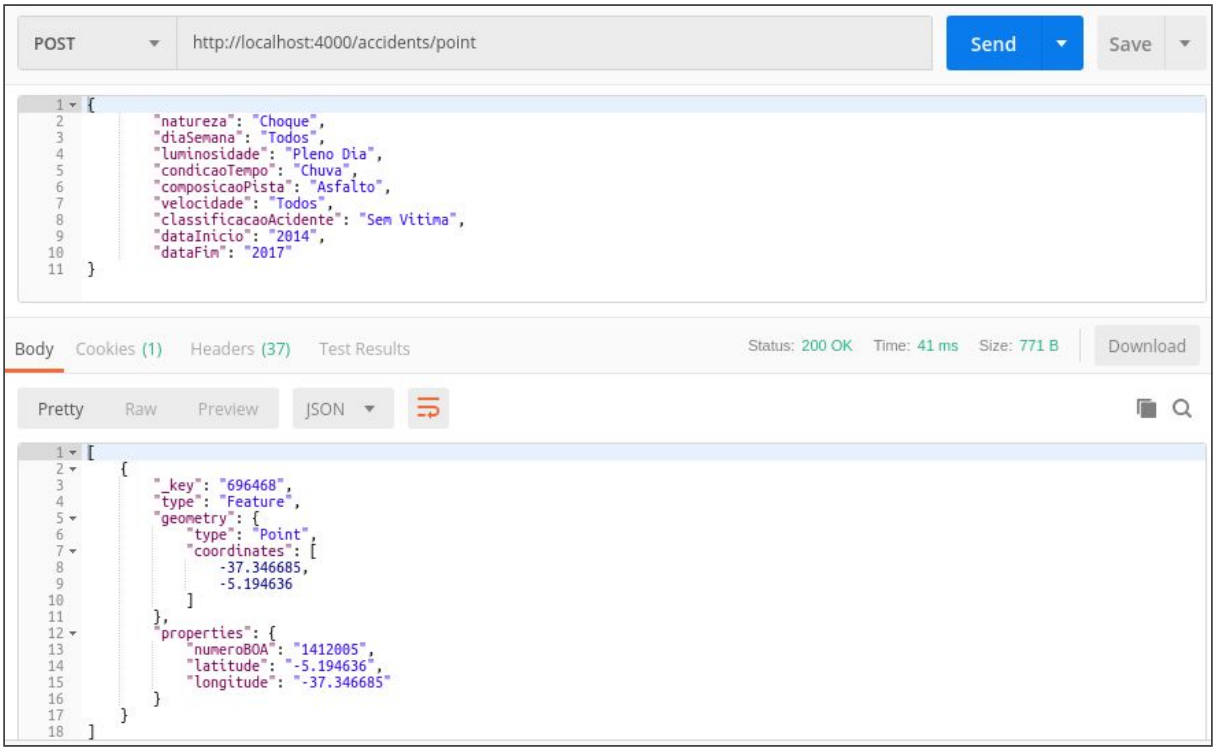
Fonte: Autoria própria

Figura 9: Requisição para a obtenção de um acidente específico



Fonte: Autoria própria

Figura 10: Requisição para a obtenção de acidentes utilizando filtros



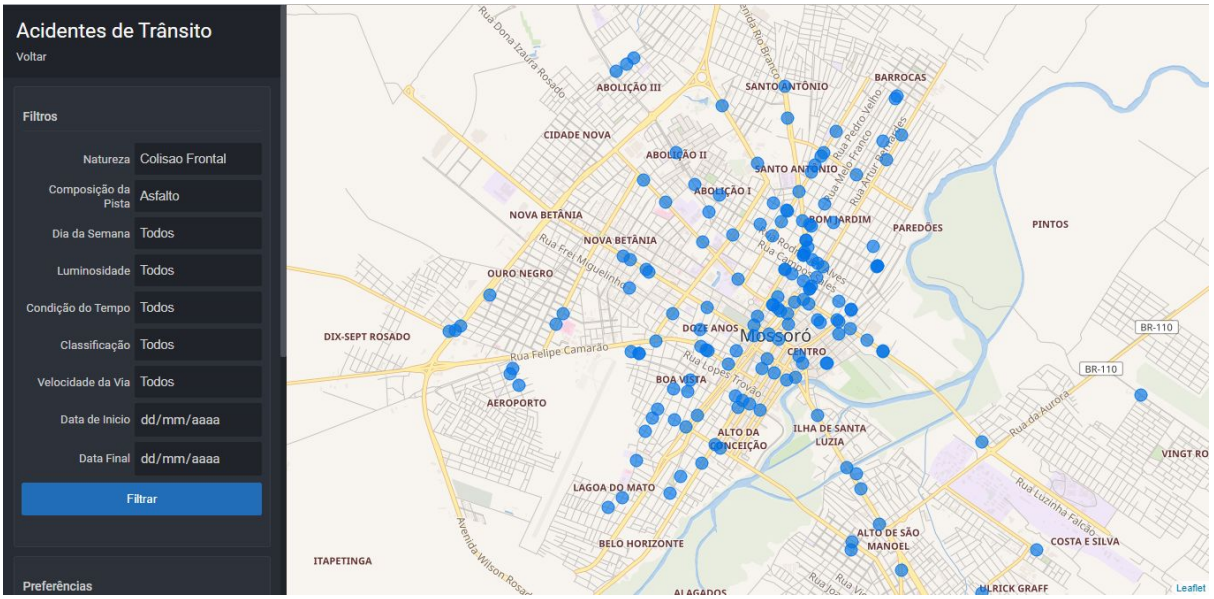
Fonte: Autoria própria

5.2. MAPP

Além da validação utilizando o Postman uma aplicação fruto de outro trabalho foi desenvolvida para consumir a API. A aplicação MAPP trata-se de uma plataforma para visualização de dados georreferenciados, em sua primeira versão ela tem como objetivo consumir os dados relacionados aos acidentes de trânsito e ocorrências médicas da cidade de Mossoró-RN.

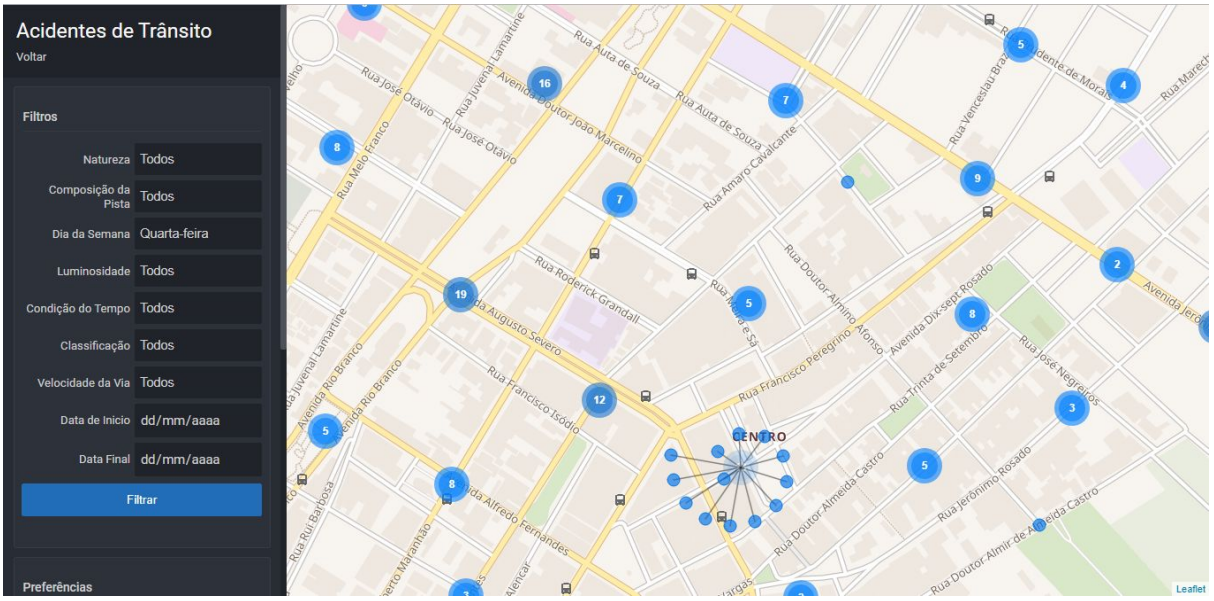
As Figuras 11, 12 e 13 apresentam a forma como a aplicação MAPP lida com os GeoJSONs retornados pela API e mostram de forma gráfica os menus de filtros utilizados nas aplicação.

Figura 11: Mapa de pontos



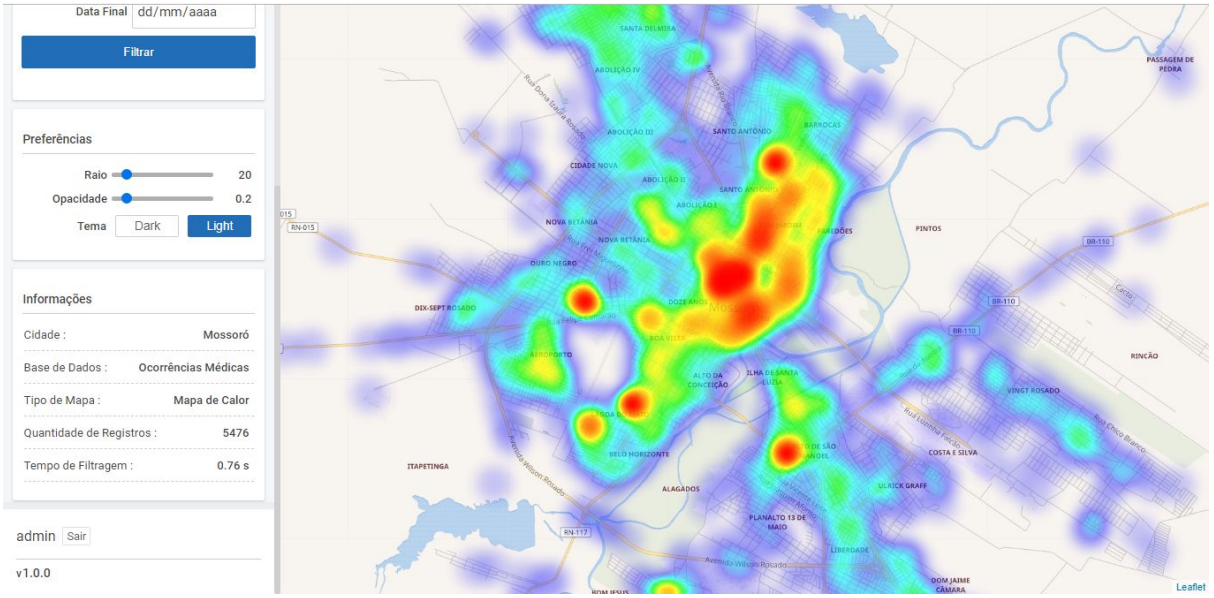
Fonte: Victor Silva

Figura 12: Pontos clusterizados



Fonte: Victor Silva

Figura 13: Mapa de calor



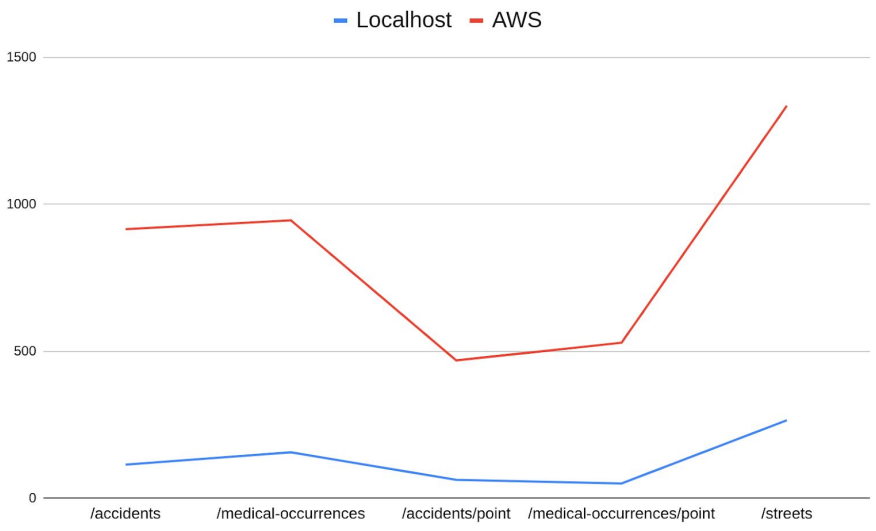
Fonte: Victor Silva

7. Avaliação de desempenho

Os testes ocorreram com a API em execução em dois ambientes distintos, sendo o primeiro com a API executando localmente e o segundo com a API em execução em um servidor em nuvem. Para as execuções em nuvem foi utilizada uma instância EC2 t2.micro, que faz parte da miríade de ferramentas disponíveis no AWS (Amazon Web Services), que trata-se de uma plataforma de serviços de computação em nuvem oferecida pela Amazon.

Foram realizados 30 requisições em cada uma das rotas exemplificadas na figura 14, onde a API, mesmo em um ambiente muito limitado, apresentou um bom desempenho no tempo médio de resposta.

Figura 14: Comparação de desempenho entre execução local e AWS (t2.micro)



Fonte: Autoria própria

8. Conclusão

Ao fim deste trabalho foi desenvolvida uma API REST que, ao fazer parte de um SIG, pode auxiliar na tomada de decisões, modernizar a base de dados e a forma como eles são manipulados pela gestão pública, disponibilizar dados importantes a população em geral e possibilitar a terceiros a realização de análises de pontos críticos em uma cidade.

Futuramente pretende-se melhorar a aplicação para permitir a exportação para outros formatos além do JSON, efetuar testes em outros ambientes, adicionar funções para análise estatística e data science e melhorar o modelo REST para atingir o RESTful.

9. Referências

BURROUGH, Peter A. et al. Principles of geographical information systems. Oxford university press, 2015.

CÂMARA, G. MEDEIROS, J.S. Geoprocessamento para Projetos Ambientais. 1998. Disponível em: <www.dpi.inpe.br/gilberto/tutoriais/gis_ambiente>.

Acesso em: 02 set. 2019.

CASANOVA, M. A. CÂMARA, G. DAVIS, C. VINHAS, L. QUEIROZ, G. Bancos de Dados Geográficos. Editora MundoGEO, 2005.

SILVA, C. B. da. Um Modelo Computacional Para Integração de Problemas de Otimização Utilizando Banco de Dados Orientados a Grafos. Mossoró, RN, 2017. Monografia, Universidade do Estado do Rio Grande do Norte.