

# **Aplicação de *Automatic Number Plate Recognition* (ANPR) no controle de acesso de veículos**

**Maurício de A. Cordeiro<sup>1</sup>**

<sup>1</sup>Instituto Federal de Educação, Ciência e Tecnologia da Bahia  
Avenida Sérgio Vieira Melo, 3150. Bairro Zabelê - Vitória da Conquista - BA - Brasil  
CEP 45078-900

mauriciocordeiro@live.com

**Abstract.** *soon...*

**Resumo.** *Este trabalho consiste no desenvolvimento de sistema para controle de acesso de veículos baseado em ANPR (Automatic Number Plate Recognition), com largo potencial de sua aplicação em locais ou espaços físicos que exigem algum nível de segurança, quando da entrada e saída de pessoas (a título de exemplo, citam-se aqui condomínios e estacionamentos privados).*

## **1. Introdução**

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus tristique augue in diam accumsan, ac pellentesque tortor rutrum. Maecenas sollicitudin lorem nec turpis egestas consequat. Ut ornare, lorem sed aliquam tincidunt, massa erat luctus nulla, id malesuada velit risus quis leo. Nulla facilisi. Maecenas non sodales lectus, eu lacinia nibh. Vestibulum tortor ligula, pellentesque sed vestibulum quis, suscipit nec augue. Ut ac augue non ex dictum laoreet ut sed diam. Vestibulum pretium tellus quis nisl iaculis, interdum viverra felis fermentum. Aenean rhoncus vitae neque id tempor. Nulla ac turpis cursus, fermentum ante et, dignissim dui. Suspendisse et odio dapibus, lacinia justo vel, fermentum nibh. Nam a ligula in nisi pretium aliquam. Aenean volutpat justo est, et dignissim tellus tempor non. Praesent imperdiet erat non quam efficitur, a tincidunt diam mollis. Nam convallis lacus nec nisl dignissim efficitur.

Quisque cursus quis odio a tincidunt. Maecenas vel purus sagittis, suscipit lectus a, pellentesque magna. Morbi sagittis, quam quis suscipit suscipit, lectus justo egestas felis, sed vehicula elit lorem vitae ante. Nam lacinia interdum ullamcorper. Nulla lorem est, placerat id convallis id, consequat eu lorem. Aliquam est enim, viverra ac lacinia in, lacinia at risus. Mauris maximus urna ornare enim interdum ornare a id metus. Sed non aliquet diam, sed tristique erat. Vestibulum congue mi et ligula faucibus malesuada.

Donec ornare venenatis magna. Aenean facilisis tincidunt vestibulum. Quisque at lorem vitae mauris sollicitudin sodales interdum et diam. Suspendisse potenti. Vestibulum eget rutrum tortor. Nulla vel scelerisque justo, tincidunt consectetur mauris. Cras vulputate rhoncus justo, facilisis elementum magna pretium et. Curabitur mollis massa risus, non efficitur neque ullamcorper in. Etiam imperdiet pellentesque tortor et efficitur. Fusce sed nisi in turpis cursus vehicula. Nulla pellentesque, purus vitae interdum sodales, mauris ipsum fermentum sem, ac vestibulum augue lorem in nulla. Donec quis ligula nibh. Fusce pretium mauris quis turpis convallis mollis.

## 1.1. Trabalhos correlatos

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus tristique augue in diam accumsan, ac pellentesque tortor rutrum. Maecenas sollicitudin lorem nec turpis egestas consequat. Ut ornare, lorem sed aliquam tincidunt, massa erat luctus nulla, id malesuada velit risus quis leo. Nulla facilisi. Maecenas non sodales lectus, eu lacinia nibh. Vestibulum tortor ligula, pellentesque sed vestibulum quis, suscipit nec augue. Ut ac augue non ex dictum laoreet ut sed diam. Vestibulum pretium tellus quis nisl iaculis, interdum viverra felis fermentum. Aenean rhoncus vitae neque id tempor. Nulla ac turpis cursus, fermentum ante et, dignissim dui. Suspendisse et odio dapibus, lacinia justo vel, fermentum nibh. Nam a ligula in nisi pretium aliquam. Aenean volutpat justo est, et dignissim tellus tempor non. Praesent imperdiet erat non quam efficitur, a tincidunt diam mollis. Nam convallis lacus nec nisl dignissim efficitur.

## 2. Arquitetura

A solução desenvolvida neste trabalho é baseada na arquitetura de microsserviços, com módulos virtuais containerizados, que se comunicam via HTTP, além da aplicação de tecnologias baseadas em ANPR. Esta seção se dedica a detalhar esses conceitos.

### 2.1. Arquitetura de Microsserviços

Arquitetura de Microsserviços (*Microservices Architecture* - MSA) é um modelo arquitetural onde processos de *software* são realizados por componentes fracamente acoplados, que possuem funcionalidades específicas e bem definidas, e que se comunicam através de interfaces padronizadas [Viggiato et al. 2018].

A MSA pode ser considerada como a segunda iteração da Arquitetura Orientada a Serviços (*Service Oriented Architecture* - SOA), onde serviços complexos são decompostos em partes mais flexíveis, especializadas e de fácil manutenção, de modo que cada uma é responsável por uma única e pequena funcionalidade com o intuito de realizá-la bem [Hoday et al. 2019].

### 2.2. Virtualização baseada em contêiner

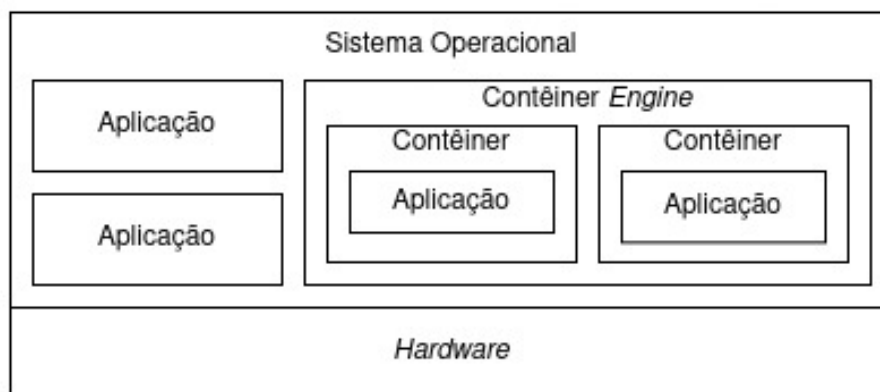
A Virtualização baseada em contêiner [Eder 2016] consiste na utilização de recursos de *hardware* e do *kernel* de um sistema hospedeiro a fim de criar ambientes isolados para a execução de determinados processos, o chamado **contêiner**, esquematizado na Figura 1.

Nessa abordagem, o contêiner aloca apenas os recursos necessários para executar sua aplicação, enquanto trabalha isolado de outros contêineres que possam estar em execução em um mesmo hospedeiro. Essa característica corrobora com sua utilidade quanto a aplicação em soluções baseadas em MSA, uma vez que os contêineres são capazes de se comunicar entre si.

### 2.3. REST

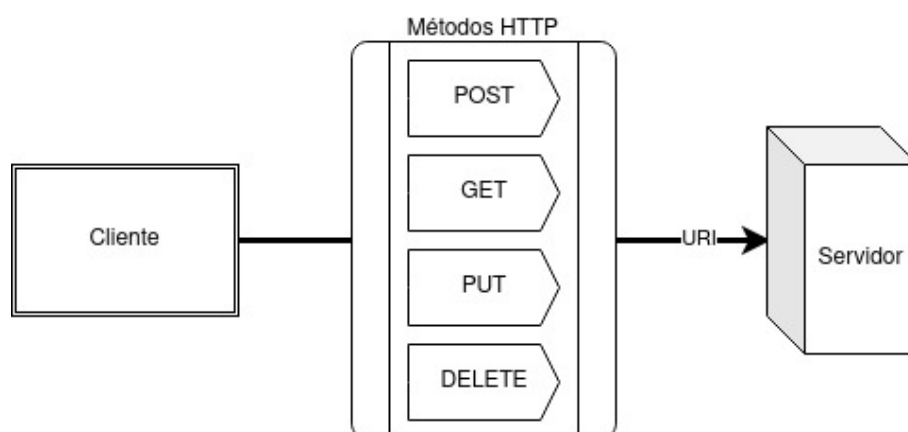
O REST (*REpresentational State Transfer*) é uma implementação da SOA sobre o HTTP para transferência de representações de um determinado recurso entre cliente e servidor [Mumbaikar et al. 2013].

De forma geral, o REST estabelece semântica para métodos e URI (*Universal Resource Identifier*) no HTTP, padronizando a forma como recursos são solicitados e



**Figure 1. Virtualização baseada em contêiner**

disponibilizados na web (Figura 2). Nesse contexto, leia-se "recurso" como um item acessível via URI e a "semântica do método" padroniza o modo de interação com o recurso, como, por exemplo, POST para criar, GET para solicitar, PUT para editar e DELETE para remover [Adamczyk et al. 2011].



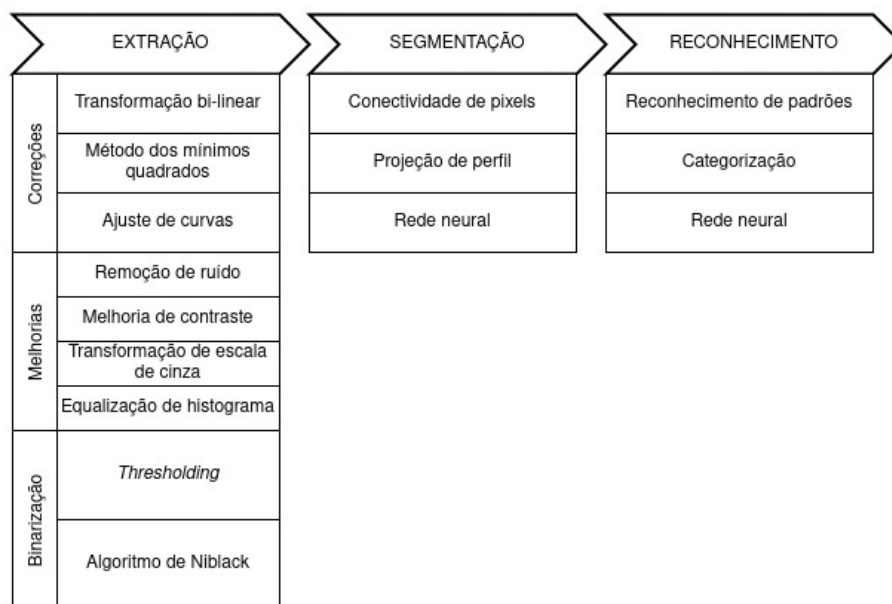
**Figure 2. Comunicação cliente-servidor com REST**

## 2.4. ANPR

O ANPR (*Automatic Number Plate Recognition*), que possui papel central na solução desenvolvida neste trabalho, é uma tecnologia que utiliza de visão computacional e processamento de imagem para identificar e reconhecer caracteres em placas de veículos.

Existem diferentes técnicas e algoritmos que podem ser usados no ANPR, mas, de forma geral, todos eles seguem um processo com etapas bem definidas [Mufti et al. 2021]: (i) Extração da placa, (ii) Segmentação de caracteres e (iii) Reconhecimento de caracteres.

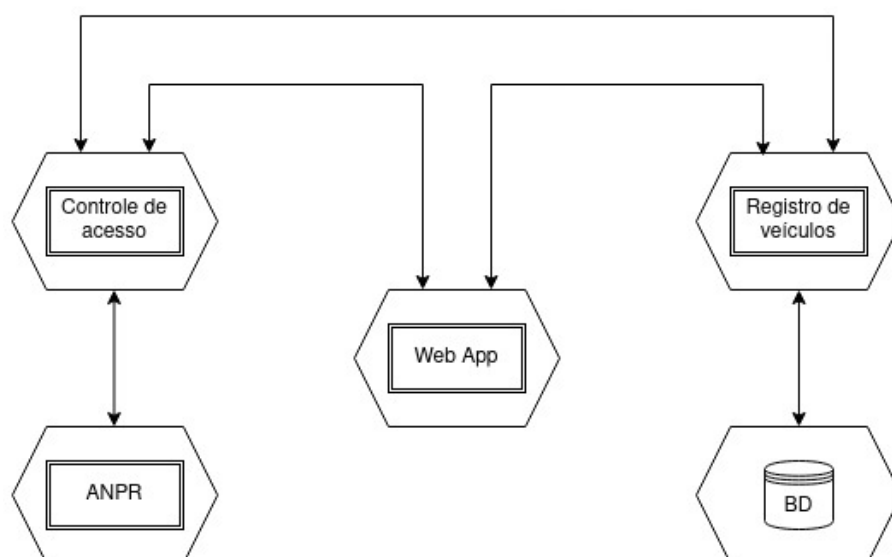
Dentro das etapas para a execução do ANPR, [Shashirangana et al. 2020] pode-se classificar as diferentes técnicas (que vão desde algoritmos de processamento de imagens até aplicação de redes neurais para classificação de objetos) conforme o mostrado na Figura 3:



**Figure 3. Etapas e técnicas aplicadas no ANPR**

### 3. Sistema para controle de acesso de veículos baseado em ANPR

O sistema para controle de acesso de veículos implementado neste trabalho foi organizado em módulos, ou contêineres, cada um com uma funcionalidade específica (segundo os princípios da MSA), que se comunicam direta ou indiretamente entre si, como mostrado na Figura 4. Esses módulos foram implementados com ferramentas tais como: Docker, Java, Python, Angular e MongoDB.



**Figure 4. Esquema da organização e relacionamento entre os contêineres do sistema**

A seguir, será apresentada com detalhes a forma com que cada um dos módulos foi desenvolvida e como eles se interagem.

### 3.1. Composição de contêineres

Todos os módulos do sistema são construídos com contêineres Docker, constituindo assim uma aplicação multi-contêiner, a partir de alguma imagem<sup>1</sup> padrão ou de uma imagem personalizada.

Para definir e executar a aplicação com múltiplos contêineres, é utilizada a ferramenta Docker Compose<sup>2</sup>. Essa ferramenta tem o objetivo de otimizar o processo de construção e execução de múltiplos contêineres, tanto das etapas de desenvolvimento e testes quanto na implantação para produção.

O uso da ferramenta consiste em 3 passos:

1. Definir, no arquivo *Dockerfile*, os recursos necessários de cada um dos contêineres da aplicação, como a imagem docker, dependências, usuários e diretórios, portas de serviços e comandos para execução de aplicações, etc. De forma geral, o *Dockerfile* é um *script* que define **quais** e **como** instalar e utilizar as ferramentas e aplicações de um contêiner;
2. Definir, em um arquivo *.yml*, os serviços necessários e a relação entre os contêineres, como interface de rede, portas de serviços, volumes de disco e dependências entre contêineres;
3. Executar o comando `docker-compose up` para inicializar todos os contêineres descritos no arquivo *.yml*

Vale ressaltar que, apesar do controle ser realizado em um único arquivo, em um cenário de atualização, o Docker Compose apenas reconstruirá o contêiner que possuir alguma modificação.

### 3.2. ANPR

O módulo ANPR é o responsável por realizar o processamento das imagens dos veículo e devolver o texto de sua placa. Para tal, foi utilizado a biblioteca de código-aberto OpenALPR<sup>3</sup> e foi implementado uma API REST com o *framework* Java Spring-Boot para processar as requisições de leitura de placas e devolver uma resposta, no formato JSON<sup>4</sup>, para o contêiner cliente.

A Figura 5 ilustra o fluxo das requisições de leitura de placa para a API do contêiner de ANPR.

A biblioteca OpenALPR, ao receber uma imagem, realiza uma série de etapas para detectar o possível texto escrito na placa do veículo [Hill 2014]:

1. **Deteção:** Busca regiões com possíveis placas na imagem;
2. **Binarização:** Transforma em preto-e-branco as regiões de possíveis placas;
3. **Análise de caracteres:** Busca por possíveis caracteres na imagem;

---

<sup>1</sup>Uma "imagem docker" pode ser entendida como uma base ou um *template* sobre o qual o contêiner será inicializado.

<sup>2</sup>Página para a documentação oficial do Docker Compose: <https://docs.docker.com/compose/>

<sup>3</sup>Endereço do repositório e documentação da biblioteca: <https://github.com/openalpr/openalpr>

<sup>4</sup>JavaScript Object Notation.

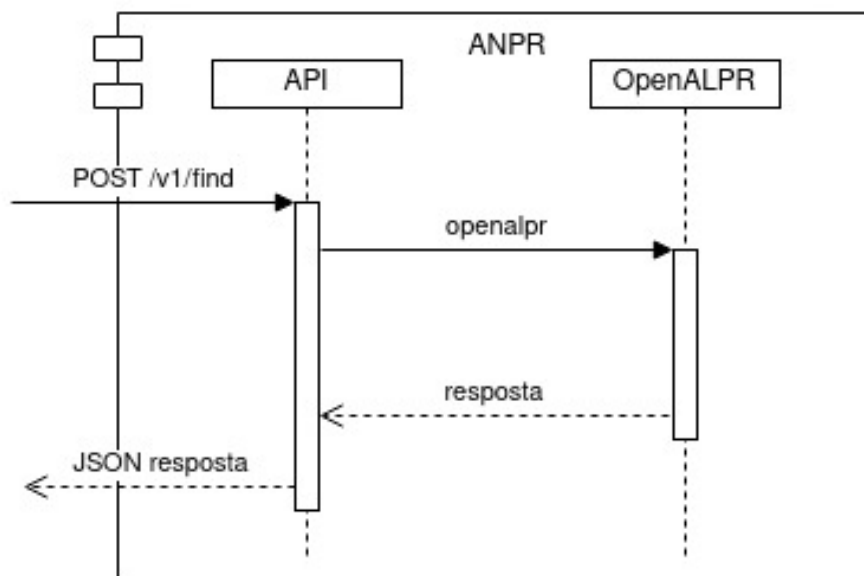


Figure 5. Fluxo de requisições no contêiner de ANPR

4. **Análise de bordas:** Busca pelas bordas da placa na região;
5. **Ajuste de distorção:** Ajusta distorções de perspectiva para a placa ser "vista de frente";
6. **Segmentação:** Segmenta individualmente os caracteres na placa;
7. **OCR<sup>5</sup>:** Analisa a imagem de cada caractere para encontrar a letra ou número correspondente;
8. **Pós-processamento:** Cria um *ranking* de placas detectadas, com base na confiança e expressões regulares (se solicitado).

A resposta do OpenALPR é então capturada pela API e enviada ao contêiner solicitante.

### 3.3. Banco de Dados (BD)

Este é o módulo responsável pela persistência dos dados usados pelo sistema, construído com uma imagem do MongoDB<sup>6</sup>. Ele se encontra em um contêiner próprio para fins de controle de eventuais atualizações e uso de volume de disco, uma vez que a lógica da persistência dos dados se encontra no módulo descrito na sub-seção seguinte.

### 3.4. Registro de veículos

Este é o módulo responsável pela lógica de persistência do sistema e acesso ao banco de dados.

Escrito em Python, este módulo implementa as operações CRUD<sup>7</sup> para veículos e usuários do sistema — utilizando a biblioteca PyMongo<sup>8</sup> —, enquanto implementa uma API — com a biblioteca Flask<sup>9</sup> —, para tratar as requisições de outros módulos.

<sup>5</sup>Optical character recognition (Reconhecimento ótico de caracteres)

<sup>6</sup>Página oficial do MongoDB: <https://www.mongodb.com/>. Página para a imagem Docker do MongoDB: [https://hub.docker.com/\\_/mongo](https://hub.docker.com/_/mongo)

<sup>7</sup>Create, Read, Update e Delete

<sup>8</sup>Documentação oficial do PyMongo: <https://docs.mongodb.com/drivers/pymongo/>

<sup>9</sup>Documentação oficial do Flask: <https://flask.palletsprojects.com/en/2.0.x/>

### 3.5. Controle de acesso

O módulo de Controle de acesso — escrito em Java com Spring-Boot — é onde a principal lógica de negócio do sistema está implementada. Ele é responsável por receber, via requisição REST, uma imagem de uma aplicação cliente qualquer, repassar esta imagem para o módulo ANPR a fim de obter a placa do veículo e então requisitar para o módulo de Registros as informações de cadastro referentes a placa encontrada. Uma vez de posse dessas informações, ele responde a aplicação cliente se o veículo da imagem tem ou não a autorização de acesso.

O diagrama da Figura 6 ilustra a troca de mensagens encabeçada pelo módulo de Controle de acesso.

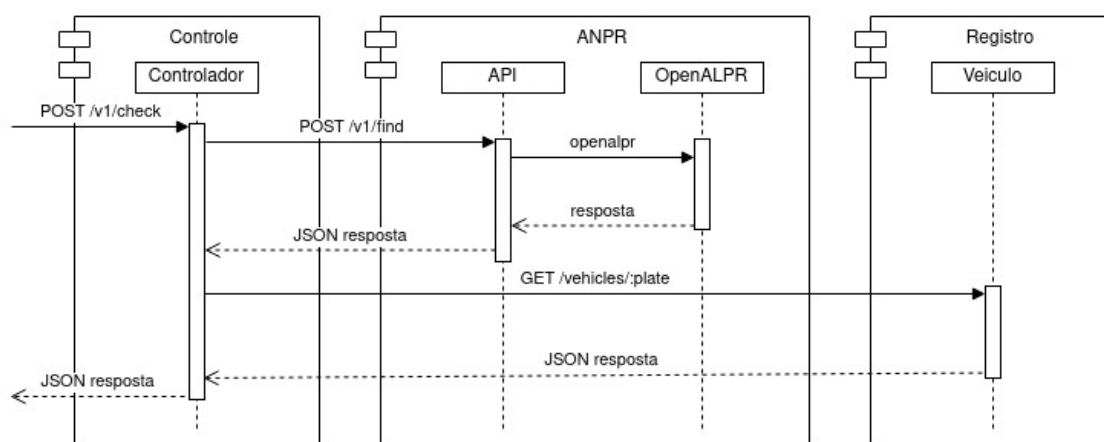


Figure 6. Fluxo de requisições a partir do Controle de acesso

### 3.6. Web App

O módulo *Web App* — escrito em Typescript com o *framework* Angular 9 —, interpreta dois papéis no sistema implementado: (i) é o *front-end* para a manutenção das informações de cadastro de veículos, acessando o módulo Registro de veículos, e (ii) é o *front-end* para prova de conceito do sistema, realizando requisições para o módulo de Controle de acesso.

Além de poder ser executado em um contêiner Docker, este módulo pode ser compilado como um aplicativo móvel multiplataforma<sup>10</sup>, através do *framework* Apache Cordova<sup>11</sup>.

## 4. Resultados

```
/anpr-auth
+-- /mongo
+-- /alpr4j
| +-- Dockerfile
```

<sup>10</sup>Android ou iOS

<sup>11</sup>Documentação oficial do Apache Cordova: <https://cordova.apache.org/docs/en/latest/>

```

| +-- ...
+-- /check4j
| +-- Dockerfile
| +-- ...
+-- /vehiclespy
| +-- Dockerfile
| +-- ...
+-- /anpr-ng
| +-- Dockerfile
| +-- ...
+-- docker-compose.yml

```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam at tellus sed diam condimentum tempus. Morbi sagittis ante ex, sed molestie mi euismod sed. Lorem ipsum dolor sit amet, consectetur adipiscing elit. In in elementum dui. Aliquam eget cursus erat. Quisque porttitor vitae metus pulvinar commodo. Vivamus vel ipsum quis diam cursus vulputate vitae at metus. Morbi at quam in mauris lobortis varius. Etiam venenatis lacinia felis quis viverra. Quisque volutpat ac nisl ut gravida. Etiam auctor lacus a nisi fermentum blandit. Fusce at luctus elit, at fringilla est. Nunc nunc nibh, tempor id imperdiet nec, porttitor non tortor.

Sed eros augue, pharetra at dignissim sed, dignissim in odio. Praesent id maximus tellus. Fusce nec condimentum arcu, efficitur sollicitudin dolor. Suspendisse molestie iaculis diam ut tincidunt. Sed ultricies massa sit amet ex convallis, at porttitor nulla eleifend. In ut varius ipsum. Aliquam molestie vestibulum purus, eu rhoncus nisl commodo sit amet. Donec ac ex nisl. Nullam ultrices lectus ipsum, non porta nunc vehicula id.

## 5. Conclusão

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam at tellus sed diam condimentum tempus. Morbi sagittis ante ex, sed molestie mi euismod sed. Lorem ipsum dolor sit amet, consectetur adipiscing elit. In in elementum dui. Aliquam eget cursus erat. Quisque porttitor vitae metus pulvinar commodo. Vivamus vel ipsum quis diam cursus vulputate vitae at metus. Morbi at quam in mauris lobortis varius. Etiam venenatis lacinia felis quis viverra. Quisque volutpat ac nisl ut gravida. Etiam auctor lacus a nisi fermentum blandit. Fusce at luctus elit, at fringilla est. Nunc nunc nibh, tempor id imperdiet nec, porttitor non tortor.

*trabalhos futuros*

## References

- Adamczyk, P., Smith, P. H., Johnson, R. E., and Hafiz, M. (2011). Rest and web services: In theory and in practice. In *REST: from research to practice*, pages 35–57. Springer.
- Eder, M. (2016). Hypervisor-vs. container-based virtualization. *Future Internet (FI) and Innovative Internet Technologies and Mobile Communications (IITM)*, 1.
- Hill, M. (2014). *OpenALPR Design*. OpenALPR, <https://github.com/openalpr/openalpr/wiki/OpenALPR-Design>.



- Homay, A., Zoitl, A., Sousa, M. d., and Wollschlaeger, M. (2019). A survey: Microservices architecture in advanced manufacturing systems. In *2019 IEEE 17th International Conference on Industrial Informatics (INDIN)*, volume 1, pages 1165–1168. IEEE.
- Mufti, N., Shah, S. A. A., et al. (2021). Automatic number plate recognition: A detailed survey of relevant algorithms. *Sensors*, 21(9):3028.
- Mumbaikar, S., Padiya, P., et al. (2013). Web services based on soap and rest principles. *International Journal of Scientific and Research Publications*, 3(5):1–4.
- Shashirangana, J., Padmasiri, H., Meedeniya, D., and Perera, C. (2020). Automated license plate recognition: a survey on methods and techniques. *IEEE Access*, 9:11203–11225.
- Viggiato, M., Terra, R., Rocha, H., Valente, M. T., and Figueiredo, E. (2018). Microservices in practice: A survey study. *arXiv preprint arXiv:1808.04836*.