ESTUDO SOBRE O SOFTWARE DE MICRO-FRAMEWORK WEB FLASK

Luís Fernando Alves da Silva

Pedro Ramires da Silva Amalfi Costa

Reginaldo Donizeti Cândido

Mogi Guaçu SP

DOI:

RESUMO

Este artigo científico tem como objetivo explicar a função do software de função do software de micro-framework web Flask, partindo do conceito dos sistemas gerenciadores de banco de dados a linguagem SQL surgimento da web, seguindo pelo conceito de tecnologias de frontend e backend, sites estáticos e dinâmicos assim como as abordagens de desenvolvimento web (abordagem programática, abordagem de modelo, abordagem híbrida e finalmente abordagem de framework mvc) finalizando com o conceito de micro-framework e do Flask assim como de suas principais bibliotecas o Werkzeug e o Jinja2.

Palavras-chave: FLASK; ORM; MICRO-FRAMEWORK WEB; WERKZEUG; JINJA2.

1 INTRODUÇÃO

Os aplicativos Web estão entre os softwares mais importantes atualmente.

Um aplicativo web mostra seu negócio para o mundo, oferece uma plataforma otimizada para dispositivos móveis e traz tráfego para seu site. Se você tiver clientes em potencial acessando seu website, isso aumentará seus lucros (MULTIMEDIA DESIGN STUDIO, 2019)

Existem diversas formas de se programar um aplicativo web, das maneiras mais arcaicas a as mais recentes. Dentre elas o Micro-Framework Web Flask se destaca como uma alternativa eficiente e veloz.

Embora a velocidade não tenha sido muito considerada durante a fase de desenvolvimento do Flask, para o usuário, é uma característica fundamental. O Flask segue uma abordagem de design minimalista e, portanto, tem um desempenho mais rápido em relação ao Django. Ele pode suportar centenas de consultas por segundo sem nenhum impacto negativo. (FATIMA, 2018)

Sua natureza minimalista propicia uma curva de aprendizado leve e altamente customizável.

Devido ao fato de ser minimalista, o Flask fornece ao desenvolvedor os recursos necessários para criar um protótipo para sua aplicação web facilmente. Ele oferece aos desenvolvedores controle e flexibilidade para o desenvolvimento. Existem vários templates disponíveis com a biblioteca de linguagem padrão Jinja2. Além disso, o aplicativo vem com a biblioteca WSGI toolkit e um servidor web embutido para que você não precise depender de servidores externos, como o Apache, para testar seu aplicativo. (FATIMA, 2018)

Esse artigo busca apresentar a evolução tecnológica da Web assim como o princípio de funcionamento do Flask.

2 TECNOLOGIAS DE PROGRAMAÇÃO WEB

Neste capítulo será abordado de forma resumida quais são as tecnologias envolvidas na programação de aplicativos WEB.

2.1 A WORLD WIDE WEB (WWW)

A World Wide Web ou WWW é um sistema de documentos interligados por hipertexto que se tornou sinônimo de internet. Tendo sido lançado em 1991 por Tim Berners-Lee ela consiste em um sistema de páginas (documentos) ao qual o usuário acessa através de um software tipo cliente chamado navegador ligado a um sistema servidor por um protocolo simples de troca de documentos (HTTP) através da internet.

Os documentos são estruturados com uma linguagem de marcação (HTML) e possuem hiperlinks que ao serem selecionados mudam a pagina para outra do mesmo servidor (link interno) ou mesmo de um site localizado em outro servidor (link externo), utilizando para isso um endereçador composto de um domínio que indica o site e um caminho que indica o recurso desejado naquele site (URL).

Conforme nos é mostrado por (Shklar e Rosen, 2003) Havia três componentes básicos criados por Tim Berners-Lee, compreendendo a essência da tecnologia da Web: 1. Uma linguagem de marcação para formatar documentos de hipertexto; 2. Um esquema de notação uniforme para endereçar recursos acessíveis pela rede; 3. Um protocolo para transportar mensagens pela rede.

2.2 SITES ESTÁTICOS

No início a Web era composta de páginas ou sites (conjunto de páginas) simples, com poucos recursos, o sistema se baseava no paradigma cliente

servidor, através do qual os clientes (navegadores Web) recebiam documentos HTML através dos servidores Web. Nesse sistema para criar um site o programador precisava conhecer somente a linguagem HTML pois os outros componentes (HTTP e URL) fazem parte da infraestrutura da Web (navegadores e servidores).

Esse sistema é classificado como site estático, assim chamado pois todos os documentos recebidos pelo navegador já estavam criados no momento de sua solicitação.

Figura 1 – Fluxograma de um site estático

Fonte: SILVA (2019)

2.2.1 TECNOLOGIAS DE FRONTEND

Embora armazenado no servidor Web nos definimos o HTML como uma tecnologia de Frontend, assim chamada pois ela é executada no navegador do lado do cliente (cliente-side).

Durante a evolução da Web várias outras tecnologias de Frontend foram criadas, destas três se destacam pela sua importância e serão discutidas a seguir.

2.2.1.1 HTML

O HTML Hypertext Markup Language ou em português Linguagem de Marcação de Hipertexto, é a mais antiga e ao mesmo tempo a mais essencial das tecnologias de Frontend sendo obrigatório o seu uso para a criação de um site.

A função do HTML é estruturar o site através de marcadores chamados tags, através dele o navegador consegue saber o significado semântico de cada parte do site e isso contribui para o processo de renderização da página.

2.2.1.2 CSS

A CSS Cascading Style Sheets que, traduzido para o português, significa Folha de Estilo em Cascatas, é a tecnologia de Frontend criada para a estilização de um site embora não essencial o seu uso em sites modernos é cada vez mais difundido, pois no decorrer da evolução do HTML as tags de formatação foram depreciadas em favor do uso da CSS .

A função da CSS é permitir a formatação e o design de um site através de seletores e regras que orientam o processo de renderização da página.

2.2.1.3 JAVASCRIPT (ECMAScript)

O JavaScript (ECMAScript) é uma linguagem de programação que pode ser inserida nas páginas HTML permitindo assim que pequenos programas (scripts) sejam executados no cliente-side dando-lhe capacidades programáticas capazes de alterar o comportamento e estrutura de um site.

Diferente do HTML e CSS que foram desenvolvidos pelo W3C (a instituição criada por Tim Berners-Lee para o desenvolvimento da Web) o JavaScript surgiu por iniciativa de um fabricante de navegador (Netscape) embora o mesmo tenha por iniciativa própria entrado em contato com a

organização de padronização ECMA International para sua padronização no (padrão) que foi chamado de ECMAScript.

2.3 SITES DINÂMICOS

Conforme a Web evoluiu as páginas e sites simples e estáticos não eram mais suficientes para acompanhar a escala e a complexidade necessária aos novos sistemas como e-commerce e portais, estes sites podem ser considerados como aplicativos Web.

Um site simplesmente entrega conteúdo de arquivos estáticos. Um aplicativo da Web apresenta conteúdo adaptado dinamicamente com base em parâmetros de solicitação, comportamentos de usuário rastreados e considerações de segurança (SHKLAR E ROSEN, 2003).

Isso envolve o uso de diversas tecnologias e outros componentes além do servidor Web que colaboram entre si para a criação dinâmica do conteúdo.

Figura 2 – Fluxograma de um site dinâmico

Fonte: SILVA (2019)

2.3.1 TECNOLOGIAS DE BACKEND

O conjunto de tecnologias que é responsável por armazenar e gerar dinamicamente o conteúdo no lado do servidor é definido como tecnologias de

Backend, assim chamada pois ela é executada na pilha de servidores, também conhecida como (server-side).

Essa estrutura pode se organizada com diversos componentes a depender da complexidade do sistema, entretanto a pilha de softwares (conjunto de programas) chamada LAMP (as iniciais de Linux, Apache, MySQL e PHP) é a base mais comum no desenvolvimento Web moderno, como característica comum esses programas são de código aberto e gratuitos e além de estar presente na maioria das hospedagens Web, de maneira didática o seu uso será brevemente explanado a seguir.

2.3.1.1 SISTEMA OPERACIONAL

O Sistema Operacional é o software responsável por gerenciar as funções e recursos do sistema, ele é uma parte básica da maioria dos sistemas computacionais.

O Linux é o sistema operacional presente na maioria dos sistemas Web, isto se deve por ter sido uma opção gratuita que surgiu e cresceu junto da Web, além disso ele é um sistema altamente customizável e sua administração remota favoreceu a sua utilização

2.3.1.2 SERVIDOR WEB

O Servidor Web é o componente responsável pela interação entre o navegador e o servidor, é ele que recebe as requisições através do protocolo HTTP e as responde diretamente em um site ou arquivo estático (buscando e encaminhando os arquivos HTML) ou encaminha para a linguagem de programação responsável em um site dinâmico.

O Apache é o servidor Web mais utilizado, assim como o Linux sua popularidade é devido a um misto de pioneirismo e gratuidade, nos últimos anos outros servidores Web vem se destacando como o Lighttpd e o Nginx.

2.3.1.3 LINGUAGEM DE PROGRAMAÇÃO

Em um sistema Web a Linguagem de Programação é responsável por gerar dinamicamente o conteúdo, seja através das informações enviadas pelo usuário ou da interação com o Banco de Dados.

Embora a maioria das linguagens de programação possa ser utilizada para isso, existe um destaque das linguagens de script para essa tarefa, como exemplos podemos citar as linguagens Pearl e PHP (que geralmente representam a letra P da pilha LAMP) e mais recentemente Python e Ruby.

2.3.1.4 BANCO DE DADOS

Embora não seja estritamente necessário para se fazer um site dinâmico o uso de Banco de Dados é extremamente difundido para isso pois fornece uma fonte de dados que podem ser utilizados pelos scripts para gerar as páginas Web.

Dentre os diversos Banco de Dados existentes se destacam o MySQL e o seu fork (software derivado) MariaDB que (que representa a letra M da pilha LAMP).

3 O SOFTWARE DE MICRO-FRAMEWORK WEB FLASK

Este capítulo abordará qual é o princípio de funcionamento de um software de Micro-Framework Web especificamente o Flask.

3.1 ABORDAGENS DE DESENVOLVIMENTO WEB

Projetar um aplicativo Web envolve diversas tarefas como interpretar as requisições, autenticar usuários, buscar informações nos Banco de Dados e renderizar as respostas.

Existem diversas abordagens diferentes para fazer isso, (SHKLAR E ROSEN, 2003) definem quatro diferentes.

3.1.1 ABORDAGEM PROGRAMÁTICA

A Abordagem Programática é a mais antiga e consiste em um script que gera o código HTML através dos comandos de saída da linguagem em questão.

Um exemplo dessa abordagem é o CGI (*Common Gateway Interface* – interface de porta de entrada comum). O CGI permite que um script tenha acesso as informações do servidor web como os parâmetros da requisição, em seguida o script trata essas informações e gera o arquivo HTML através dos seus comandos de saída.

Essa é uma abordagem muito utilizada por programadores, entretanto ela dificulta o papel dos designers pois qualquer mudança que o designer queira implementar deve ser feita pelo programador (SHKLAR E ROSEN, 2003).

3.1.2 ABORDAGEM DE MODELO

Esta consiste em um modelo que contém a formatação do site, formado principalmente de código HTML e de construções incorporadas limitadas que adicionam poder programático.

O Server Side Includes, ou SSI é uma tecnologia semelhante ao CGI que implementa essa abordagem, ela fornece uma maior facilidade aos designers e pode ser integrada com o CGI, entretanto ela não é robusta o suficiente, especialmente para aplicativos orientados a banco de dados (SHKLAR E ROSEN, 2003).

3.1.3 ABORDAGEM HIBRIDA

A abordagem híbrida combina elementos de script com estruturas de modelo. Ela tem mais poder programático do que modelos puros porque permitem blocos incorporados contendo 'scripts'. Isso parece oferecer o benefício de uma estrutura orientada a página combinada com poder programático adicional (SHKLAR E ROSEN, 2003).

Por misturar scripts com formatação essa abordagem viola o princípio de separação de responsabilidades, por colocar tanto programadores como designers para trabalhar com um mesmo arquivo.

Uma grade exemplo dessa abordagem é o PHP, em um script PHP o texto é processado normalmente como HTML (abordagem de modelo) entretanto nos trechos onde é necessário de processamento computacional é adicionada uma tag especial (a super tag do PHP) e nestes trechos o código é processado como um script (abordagem programática).

3.1.4 ABORDAGEM DE FRAMEWORK MVC

Por fim, nenhuma dessas abordagens atende a um dos principais requisitos de uma boa estrutura de aplicativos da Web: a verdadeira separação de conteúdo da apresentação (SHKLAR E ROSEN, 2003).

Essencialmente, tudo se resume a entender que (1) existe conteúdo ou dados (geralmente chamado de modelo), (2) existe a maneira pela qual esses dados são apresentados (geralmente chamados de visão) e (3) essas são duas coisas separadas (SHKLAR E ROSEN, 2003).

Para escabecearmos essa separação podemos utilizar um padrão de projeto chamado MVC.

3.1.4.1 PADRÃO MVC

O padrão MVC, Modelo-Visão-Controlador (em inglês: Model-View-Controller) separa as responsabilidades do software em três camadas.

O Modelo (Model) consiste nos dados da aplicação, regras de negócios, lógica, funções, entidades e camada de acesso a dados.

A Visão (View) renderiza a resposta, pode ser qualquer saída de representação dos dados em aplicações web são (templates, html, css, javascript), em aplicações desktop (formulários, tabela, diagrama) enfim tudo relativo a visualizações.

Controlador (Controller) controla o fluxo de informações, ele faz a mediação da entrada, convertendo-a em comandos para o modelo ou visão, também coordena o fluxo de dados, recebe informações do usuário, e as requisições.

Figura 3 – Fluxograma do padrão MVC

Fonte: RAMOS ALLAN (2015)

3.1.4.2 FRAMEWORK WEB

O MVC é somente um padrão de projeto, e como o tal ela não especifica como o software deve ser implementado, para isso utilizamos um outro conceito de programação chamado Framework.

Framework é um conjunto de códigos abstratos e/ou genéricos, geralmente classes, desenvolvidos em alguma linguagem de programação, que relacionam-se entre si para disponibilizar funcionalidades específicas ao desenvolvedor de software. Em outras palavras, é como uma caixa de ferramentas, um kit que possui diversas funcionalidades devidamente implementadas, testadas e prontas para serem utilizadas na construção de softwares, poupando ao desenvolvedor tempo e trabalho na elaboração de operações básicas como acesso a banco de dados, sistema de templates, mapeamento de rotas e validação de dados. (JAQUES, 2016)

Muitos Frameworks Web implementam o padrão MVC, geralmente utilizando um software de ORM Mapeamento objeto-relacional para lidar com o Modelo.

Figura 4 – Camadas de um ORM

Fonte: DEVMEDIA (2019)

Um ORM age como uma interface entre a linguagem de programação e o Banco de Dados permitindo que o programador use a mesma linguagem de programação para criar o sistema e interagir com o BD, facilitando o seu uso gerenciamento e migração.

3.2 MICRO-FRAMEWORK WEB

Frameworks Web comuns também chamados de Frameworks Full-Stack são estruturas que buscam fornecer soluções para as diversas tarefas presentes em uma aplicação Web, conforme foi ocorrendo a evolução dessas aplicações alguns destes Frameworks foram incorporando cada vez mais soluções, crescendo assim em complexidade.

Por definição um Framework é um conjunto de funcionalidades para uma tarefa específica e para funcionar como tal este deve possuir um alto nível de integração, sendo assim ao mesmo tempo que um Framework prove soluções

para problemas de desenvolvimento comuns ele também exige para funcionar em contra partida um nível considerável de conhecimento do programador, além de reduzir a flexibilidade das escolhas do programador pois esse deve seguir algumas convenções adotadas pelo Framework em questão.

Buscando manter os benefícios dos Frameworks Full-Stack mas fornecendo uma maior flexibilidade e simplicidade surgiram os Micro-Frameworks Web.

Os Micro-Frameworks são Frameworks modularizados que possuem uma estrutura inicial muito mais simples quando comparado a um Framework convencional. Pense em um Micro-Framework como uma peça de LEGO. Inicialmente, um projeto criado com o micro-framework possui apenas o básico para funcionar, (normalmente, sistema de rotas), porém, ao decorrer do projeto, podem haver necessidades para utilização de outros recursos como conexão de banco de dados, sistemas de templates, envio de email, etc. A partir desta necessidade, novas bibliotecas são “encaixadas” no projeto, como uma estrutura de um LEGO. (ANDRADE, 2019)

3.3 O MICRO-FRAMEWORK WEB FLASK

O Flask é um Micro-Framework Web para a linguagem de programação Python, ele foi construído com o objetivo de ser simples e flexível, sendo de menor complexidade do que outros Frameworks Full-Stack em Python como o Django.

Flask é um micro-framework para Python, segundo a documentação... “Micro” não significa que a sua aplicação web inteira tenha que se encaixar em um único arquivo Python, embora isso seja totalmente possível. Também não quer dizer que falte funcionalidade ao Flask. “Micro” em microframework significa que o Flask procura manter o núcleo simples, mas extensível. O Flask não toma muitas decisões por você, tipo, que banco de dados usar. Por outro lado, é fácil mudar as coisas que já decidiu, tipo, o mecanismo de templates. Todo o resto fica por sua conta decidir, de modo que você pode obter do Flask tudo o que precisa e dispensar tudo o que não quer. (DEVFURIAa, 2019)

O Flask é baseado no Werkzeug uma biblioteca para WSGI (a especificação padrão do Python para a comunicação entre aplicações web e

servidores HTTP) e no Jinja2 (um mecanismo de renderização de templates), mas diferente do Django ele não conta com um ORM padrão.

Por padrão, o Flask não inclui uma camada de abstração de banco de dados, validação de formulário, ou qualquer outra coisa para a qual já existam diferentes bibliotecas. Em vez disso, suporta extensões para adicionar uma determinada funcionalidade à sua aplicação como se ela tivesse sido implementada no Flask em si. Muitas extensões fornecem integração de banco de dados, validação de formulários, manipulação de upload, diversas tecnologias de autenticação abertas, entre outras coisas. O Flask pode ser “micro”, mas está pronto para uso em produção, preenchendo uma gama variada de necessidades. (DEVFURIAa, 2019)

3.3.1 O TOOLKIT WSGI WERKZEUG

O Werkzeug é um toolkit para a especificação do Python WSGI Web Server Gateway Interface (em português Interface de Porta de Entrada do Servidor Web), trata-se de uma forma para servidores conversarem com frameworks, e vice-versa.

Frameworks de aplicação web em Python têm sido um problema para novos usuários de Python pois a escolha do framework web pode limitar a escolha de servidores web úteis, e vice-versa. Aplicações em Python eram frequentemente projetadas para apenas um dos servidores CGI, FastCGI, mod_python ou alguma outra API personalizada de um servidor web específico. O WSGI foi criado como uma interface de baixo-nível entre servidores web e aplicações web ou frameworks para promover uma base comum para o desenvolvimento de aplicações web portáteis. (DEVFURIAb, 2019)

As funcionalidades do Werkzeug incluem:

- Um depurador interativo
- Um objeto para lidar com requisições incluindo:
 - cabeçalhos
 - query strings
 - dados de formulário
 - arquivos
 - cookies

- Um objeto para lidar com respostas e manipular dados de streaming
- Um sistema de roteamento para lidar com recursos e endpoints de URLs além capturar variáveis de URLs
- Um utilitário HTTP para:
 - controle de cache
 - agentes de usuário
 - cookies
 - arquivos
- Um servidor de desenvolvimento WSGI local
- Um cliente para simular solicitações HTTP

Figura 5 – Site em Flask utilizando os mecanismos de objeto de requisição e rota do Werkzeug

```
from flask import Flask, escape, request

app = Flask(__name__)

@app.route('/')
def hello():
    name = request.args.get("name", "World")
    return f'Hello, {escape(name)}!'
```

Fonte: THE PALLETS PROJECTS (2019)

3.3.2 O MECANISMO DE TEMPLATE JINJA2

O Jinja2 é um mecanismo de template (modelo) em Python, ele permite separar a lógica do modelo do site de uma forma semelhante a abordagem de modelo citada anteriormente.

Jinja é um mecanismo de templates para Python. Ele facilita a tarefa de usar HTML dentro de nossas aplicações Python, tem completo suporte unicode, e para aplicativos

que necessitam de maior segurança, adiciona execução em área restrita e escape automático. (OLIVEIRA, 2019)

Ele torna a sintaxe da aplicação Web mais clara pois assim como na abordagem de modelo a maior parte do código é composta de HTML e nos locais onde é necessário que conteúdo seja inserido dinamicamente nos utilizamos tags especiais que podem ter diversas funcionalidades, de passagem de variáveis, filtros personalizados até estruturas de decisão e repetição.

O HTML embebido no código gera uma grande confusão e torna difícil fazer manutenção do mesmo, o melhor é usar um sistema de templates, onde o HTML é mantido em arquivos separados com uma sintaxe especial para a introdução de alguma lógica e manipulação dos dados vindos das aplicações. (SILVA, 2014)

A diferença em relação a abordagem de modelo é que essa tem um poder programático limitado poucas instruções, já com o Jinja2 essa limitação não existe pois os dados são processados pelo Flask e após isso as variáveis são passadas para o Jinja2 que age como a camada de visão do modelo MVC renderizando a resposta no formato desejado. Além disso o Jinja2 possui mecanismos de herança e composição que tornam mais simples a criação de sites.

O Jinja2 permite a herança de templates, isto é, podemos ter um template base onde fazemos o design geral da página, podemos incluir um cabeçalho e um rodapé, um menu horizontal ou vertical que depois poderá ser reutilizado por todos os templates que herdem do template base.html. (SILVA, 2014)

Figura 6 – Mecanismo de Template do Jinja2

```
{% extends "layout.html" %}
{% block body %}
    <ul>
        {% for user in users %}
            <li><a href="{{ user.url }}">{{ user.username }}</a></li>
        {% endfor %}
    </ul>
{% endblock %}
```

Fonte: THE PALLETS PROJECTS (2019)

4 CONCLUSÃO

A Web teve uma grande evolução desde quando foi criada até os dias atuais.

Essas mudanças foram significativas tanto em termos de conteúdo, a evolução de páginas Web para Web sites e finalmente aplicações Web. Quanto em termos de tecnologia, de Web sites estáticos a Web sites dinâmicos e os diversos paradigmas de programação de aplicativos Web.

Dentre os diversos paradigmas existentes os Frameworks Web são os que melhor atendem as necessidades atuais, entretanto dominá-los demanda tempo e conhecimento.

Os Micro-Frameworks Web são uma solução com um footprint menor, mais fáceis de aprender e altamente customizáveis.

O Flask alia essas características com a facilidade de uso da linguagem de programação Python o que o torna uma boa alternativa para essa tarefa.

REFERÊNCIAS

ANDRADE ANA PAULA DE – O que é um Micro-Framework - URL (<https://tableless.com.br/mvc-afinal-e-o-que/>) visto em 18/09/2019 publicado em 06/05/2019.

DEVFURIAa – Python Flask Introdução - URL (<http://www.devfuria.com.br/python/flask/>) acessado em 18/09/2019.

DEVFURIAb – Web Server Gateway Interface - URL (<http://www.devfuria.com.br/python/servindo-paginas-html-com-wsgi/>) acessado em 18/09/2019.

DEVMEDIA - ORM: Object Relational Mapper – URL (<https://www.devmedia.com.br/orm-object-relational-mapper/19056>) acessado em 18/09/2019.

FATIMA HALEE MUNNISA – Flask x Django: como escolher o framework correto para seu aplicativo web - URL (<https://imasters.com.br/back-end/flask-x-django-como-escolher-o-framework-correto-para-seu-aplicativo-web>) visto em 18/09/2019 publicado em 10/04/2018.

INNOVAHOST – O que é Server Side Includes (SSI) - URL (<https://innovahost.com.br/cliente/knowledgebase/73/O-que-e-Server-Side-Includes-SSI.html>) visto em 18/09/2019 publicado em 06/05/2019.

JAQUES RAFAEL – O que é um Framework? Para que serve? - URL (<http://www.phpit.com.br/artigos/o-que-e-um-framework.phpit>) visto em 18/09/2019 publicado em 07/01/2016.

MULTIMEDIA DESIGN STUDIO – A Importância de um Aplicativo Web para a sua Empresa - URL (<https://www.multimediadesignstudio.com.br/web/blog/90-a-importancia-de-um-aplicativo-web.html>) acessado em 18/09/2019.

OLIVEIRA KAMILA DE FÁTIMA SANTOS – O que é Flask e porque ele é considerado um microframework - URL

(<https://imasters.com.br/desenvolvimento/conhecendo-o-jinja2-um-mecanismo-para-templates-no-flask>) visto em 18/09/2019 publicado em 15/07/2019.

PHP – O que é um Framework? Para que serve? - URL (https://www.php.net/manual/pt_BR/intro-what-is.php) visto em 18/09/2019 publicado em 06/05/2019.

RAMOS ALLAN – O que é MVC - URL (<https://tableless.com.br/mvc-afinal-e-o-que/>) visto em 18/09/2019 publicado em 26/02/2015.

SHKLAR LEON, ROSEN RICHARD. Web Application Architecture: Principles, protocols and practices. 1º Edição. West Sussex Inglaterra: John Wiley & Son, 2003.

SILVA SÉRGIO – Tutorial, Blog em Python - Parte 4: Templates - URL (<http://www.sergiofasilva.com/post/tutorial-blog-em-python-parte-4-templates>) visto em 18/09/2019 publicado em 23/01/2014.

STACKOVERFLOW – O que é CGI e qual é sua finalidade - URL (<https://pt.stackoverflow.com/questions/93308/o-que-%C3%A9-cgi-e-qual-%C3%A9-sua-finalidade>) visto em 18/09/2019 publicado em 06/05/2019.

THE PALLETS PROJECTS – Flask - URL (<https://palletsprojects.com/p/flask/>) acessado em 18/09/2019.

THE PALLETS PROJECTS – Jinja - URL (<https://palletsprojects.com/p/jinja/>) acessado em 18/09/2019.