Fatec Sorocaba José Crespo Gonzales

Centro Paula Souza

GOVERNO DO ESTADO DE SÃO PAULO

Tecnologia em Análise e Desenvolvimento de Sistemas

LINGUAGENS PARA BACK-END

ATIVIDADE 2

Prof.º Denilce de Almeida Oliveira Veloso

Disciplina: Programação Web

Marcio Ken Matumoto Sakai 0030481921024

Sorocaba

Fevereiro/2021

1. Introdução

Com a popularização da internet muitas linguagens de programação foram criadas ou adaptadas para fazer o serviço de servidor. Muitas das linguagens BACK-END atuais não foram propriamente desenvolvidas para fazê-lo, mas o fazem muito bem.

Linguagens muito populares como C++, Java Script e Python são hoje, muito utilizadas no que se diz respeito a programação web, principalmente quando falamos de BACK-END.

2. O que é BACK-END

Antigamente os sites da web eram apenas um plugar para se visualizar informações, apenas se preocupando no que mostrar e de que forma mostrar, sem muito processo por trás. Hoje as coisas já são muito diferentes, as aplicações web mais modernas não se baseiam em apenas design, mas sim em processo, no que a aplicação faz, quais informações ela guarda, como os elementos interagem com o usuário. Tudo isso só é possível graças ao BACK-END.

Para facilitar o entendimento, podemos ver uma aplicação web como um restaurante.

O salão principal, onde os clientes sentam, fazem o pedido, realizam sua refeição pode ser considerado o FRONT-END, é a parte que o usuário interage, a parte que ele vê, a parte onde ele tem a percepção de tudo o que lhe interessa.

Já o BACK-END então, seria a cozinha, o lugar onde são feitos os processos, a parte onde o cliente em si não interage, afinal nenhum cliente vai ao restaurante, e ao pedir um prato (neste exemplo digamos que o pedido seja um simples ovo frito), o cliente não vai até a cozinha, pega o ovo na dispensa, frita o ovo, o coloca no prato, volta para o salão e faz sua refeição. O que acontece na realidade é, o cliente ao visualizar o menu, pede o ovo, aguarda a cozinha o prepara-lo e quando recebido faz sua refeição.

Portanto podemos considerar o BACK-END como a parte da aplicação que realiza os processos, a parte que o cliente não interage, a parte que faz a conexão com o banco de dados, resumindo, a parte que é responsável por fazer e não aparecer.

3. Exemplos de linguagens e frameworks

Podemos então reconhecer algumas das linguagens utilizadas no desenvolvimento do BACK-END de uma aplicação. Apesar de haver várias linguagens com diversos frameworks, trabalharemos com duas, Java Script e Python.

3.1 Mas, o que é um framework? E qual a diferença entre biblioteca

Como citado no site tableless.github.io, o framework, “tem como principal objetivo resolver problemas recorrentes com uma abordagem genérica, permitindo ao desenvolvedor focar seus esforços na resolução do problema em si, e não ficar reescrevendo software.”.

Ou seja, o framework é um código já escrito que permite o desenvolvedor se preocupar mais com o desenvolvimento das atividades necessárias para resolver o problema do usuário do que as ferramentas para que o desenvolvimento aconteça.

Voltando ao nosso exemplo do restaurante, podemos ver o framework como as ferramentas a disposição do cozinheiro. Ao receber o pedido de fritar um ovo, o cozinheiro não vai se preocupar em por exemplo, produzir a frigideira, isso faz parte das ferramentas que ele tem para realizar o pedido do cliente, e não do pedido em si. Logo o cozinheiro utilizaria a frigideira(framework), para realizar o pedido do cliente, sem precisar cria-la, focando seu tempo em questões em como o ovo deve ser frito, ou que tipo de ovo deve ser utilizado.

Porém, pode surgir a dúvida, já que isso parece muito com uma biblioteca. E realmente ambas têm muitos aspectos comuns.

Podemos considerar por exemplo um framework como uma coleção de bibliotecas que servem de base para a criação da aplicação.

Eu gosto de definir a principal diferença entre framework e biblioteca pela maneira como se é utilizado cada uma. Uma biblioteca por exemplo é utilizada chamando seus métodos, por exemplo:

```
import random

num = random.randint(1, 11)

print(num)
```

```
11
```

Neste caso utilizamos a biblioteca do python ”random“, e chamamos seu metodo, ”randint”.

Porem no caso de um Framework é comum utilizarmos suas ferramentas assim:

```
from flask import Flask, render_template

app = Flask(__name__)

@app.route('/')
def home():
    return render_template("index.html")

if __name__ == "__main__":
    app.run(debug=True)
```

```
 * Serving Flask app "main" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a pro
duction deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 288-766-521
```

O Framework Flask de BACK-END em Python neste exemplo foi criado para criar renderizar uma página da web, o index.html.

Resumindo, o que o Framework está fazendo é, ao acessar o url desta página, na route (“/”) que é o próprio endereço, a página é carregada. Veja que diferente da biblioteca, não chamamos diretamente o código que escrevemos para realizar algo, neste caso, programamos o comportamento da aplicação quando for pedido o que se encontra no “/”.

Portanto, uma das diferenças mais marcantes, é que na biblioteca utilizamos seus métodos, suas funções de forma mais ativa, as chamando e recebendo seu resultado.

Em um Framework, definimos o comportamento das respostas, enquanto o Framework se encarrega de chama-las quando é necessário.

3.2 Java Script e Express

Diretamente do https://expressjs.com/, podemos utilizar facilmente um dos melhores frameworks em Java Script, o Express.

Ele necessita de Node.JS. E é uma excelente feramente para o desenvolvimento rápido e fácil de aplicações web.

Pode ser usado tanto para aplicações como APIs e também é muito popular na criação de outros frameworks.

Um bom lugar para entender como funciona, e talvez até começar a desenvolver uma aplicação em Express é o MDN Web Docs que pode ser lido em português. https://developer.mozilla.org/pt-BR/docs/Learn/Server-side/Express_Nodejs/Introdu%C3%A7%C3%A3o

Como exemplo deixo uma lista de afazeres on-line, desenvolvida em Express. https://powerful-depths-23298.herokuapp.com/ (Desenvolvido por mim no tutorial do curso Web da App Brewery).

3.3 Python e Flask

Outro Framework bem popular é o Flask, desenvolvido em Python que apesar de ser muito utilizada em Data Science, também está ganhando mercado com desenvolvimento web, é uma linguagem BACK-END fácil de se utilizar e bastante poderosa.

Sua Documentação se encontra em https://flask.palletsprojects.com/en/1.1.x/.

Um exemplo básico de uma aplicação em Flask, pode ser encontrada em https://dashboard.heroku.com/apps/ken-matsumoto-blog (Desenvolvido por mim no tutorial do curso Web da App Brewery), um simples blog, com sistema de login, que permite usuários comentarem no post do seu dono.

4. Conclusão

Existem diversas linguagens de programação quando se trata de BACK-END, o que é preciso é saber quais suas necessidades e quais ferramentas que melhor as atendem.

Deve-se levar em conta também a familiaridade da equipe com a linguagem, as vezes é mais fácil utilizar o que está no conhecimento comum do que aprender uma linguagem e um framework do zero.

Referencias

Express 4.17.1 Fast, unopinionated, minimalist web framework for Node.js. Disponível em: < https://expressjs.com/ >. Acesso em: 15 fev. 2020.

Flask. Disponível em: < https://flask.palletsprojects.com/en/1.1.x/ >. Acesso em: 15 fev. 2020.

O que é um Framework?. Disponível em: <https://tableless.github.io/iniciantes/manual/js/o-que-framework.html >. Acesso em: 15 fev. 2020.

Personal Blog Example. Disponível em: < https://ken-matsumoto-blog.herokuapp.com/ >. Acesso em: 15 fev. 2020.

To-DO List. Disponível em: < https://powerful-depths-23298.herokuapp.com/ >. Acesso em: 15 fev. 2020.