Análise Comparativa entre os *Frameworks* Django e Spring Boot

João Victor Justo Calça, Prof. Dr. Carlos Magnus Carlson Filho (Orientador)

e-mail: jvcalca@gmail.com; carlos.carlson@fatec.sp.gov.br

Resumo: O presente artigo realiza uma análise comparativa entre os *frameworks* Django e Spring Boot. O objetivo do estudo é indicar qual biblioteca é mais completa, de acordo com a necessidade apresentada pelo desenvolvedor na busca de agregar mais valor ao seu projeto. Para tal finalidade, foram expostos aspectos de grande importância no decorrer do desenvolvimento de um projeto para servirem de critérios de comparação. Após estudar diversas publicações, artigos e outros estudos sobre o tema, foram definidos alguns parâmetros como popularidade do *framework* entre os profissionais, empresas que mantêm as bibliotecas, quantidade de estrelas no GitHub, fluxo de perguntas no site Stack Overflow, dentre outros. Como conclusão da análise, foram obtidas comparações favoráveis, que podem auxiliar os desenvolvedores a definir o melhor *framework* para o seu projeto.

Palavras-chave: Análise Comparativa**,** *Framework*, Django, Spring Boot.

Abstract: *This article performs a comparative analysis between the Django and Spring Boot frameworks. The objective of the study is to indicate which library is more complete, according to the needs presented by the developer to add more value to his project. To this end, aspects of great importance during the development of a project were exposed to serve as models of comparison. After studying several publications, articles and other studies on the subject, some parameters were defined as the popularity of the framework among professionals, companies that maintain the libraries, stars on GitHub, question flow on the Stack Overflow site, and among other parameters that will be presented throughout the article. As a conclusion of the analysis, favorable comparisons were obtained, which can help developers to define the best framework for their projects.*

Keywords: *Comparative Analysis, Framework, Django, Spring Boot.*

1 Introdução

Segundo Stack Overflow (2020), o maior site de perguntas e respostas específicas para desenvolvedores, o Spring Boot hoje é um dos *frameworks* web mais amados do mundo, com 57,7% da preferência entre os profissionais da área, e o *framework* Django, listado também entre os mais amados do mundo, apresenta 55,3% de preferência.

As empresas, diante das novas necessidades de mercado, estão se preocupando progressivamente com sua figura no mundo digital, passando a trocar seus sistemas locais por aplicações *web* e aplicativos *mobile*.

O processo de desenvolvimento de aplicações *web* sempre foi um procedimento complexo, maçante e lento, porém, atualmente com a utilização dos *frameworks*, esse processo se alterou nos últimos anos. O uso de uma biblioteca específica para determinada linguagem diminui a complexidade de produção, auxiliando o desenvolvedor em seu projeto.

Para Barro (2022), os *frameworks* são estruturas compostas por um conjunto de códigos genéricos que permitem o desenvolvimento de sistemas e aplicações. Funcionam como uma espécie de *template* ou modelo que, quando utilizado, oferece certos artifícios e elementos estruturais básicos para a criação de alguma aplicação ou *software*.

Com tantas opções de *frameworks* no mercado, torna-se cada vez mais comum, quando se inicia um projeto, haver dúvidas como: Qual o melhor *framework*? Qual *framework* utilizar no projeto? Quais parâmetros básicos devem ser considerados durante esta decisão? Qual biblioteca possui a menor curva de aprendizado?

É preciso compreender que não é possível definir um *framework* melhor do que o outro, mas deve-se procurar compreender qual biblioteca é a mais completa mediante o cenário aparente no mercado e a ocasião do projeto.

Para definir quais bibliotecas examinar, foram investigadas algumas informações, como quais são as empresas que mantêm as bibliotecas, as favoritas dos profissionais da área, a quantidade de estrelas no GitHub e o fluxo de perguntas no site Stack Overflow.

O Django é um *framework* com 66 mil estrelas no GitHub, 297 mil perguntas feitas no Stack Overflow, é atualmente mantido pela *Django Software Foundation* e um dos mais utilizados pelos profissionais. A biblioteca encoraja um desenvolvimento rápido de aplicações *web*, levando a uma curva de aprendizado menor.

O Spring Boo*t* é um *framework* com 63 mil estrelas no GitHub, 127 mil perguntas feitas no Stack Overflow, foi desenvolvido pela plataforma Java e tem uma grande comunidade de apoiadores, apesar de ter uma curva de aprendizado maior, levando em consideração o Django.

O estudo tem como objetivo apresentar uma análise comparativa entre os *frameworks* Django e Spring Boot, que são duas das mais populares bibliotecas para desenvolvimento de aplicações *web*.

2 Fundamentação Teórica

Para que, a análise comparativa possa ser compreendida, neste tópico serão apresentados alguns fundamentos básicos, como as linguagens de programação Java e Python, *Front-end* e *Back-end*, plataformas GitHub e Stack Overflow, conceito de *Framework*, além de Spring Boot e Django, especificamente.

2.1 Java

Segundo a Amazon Web Services (2022a), Java é uma linguagem de programação amplamente usada para codificar aplicações *web*. Ela tem sido uma escolha popular entre os desenvolvedores há mais de duas décadas, com milhões de aplicações Java em uso hoje. Java é uma linguagem multiplataforma, orientada a objetos e centrada em rede que pode ser usada como uma plataforma em si. É uma linguagem de programação rápida, segura e confiável para codificar tudo, desde aplicações móveis e *software* empresarial até aplicações de *big data* e tecnologias do servidor.

2.2 Python

De acordo com a Amazon Web Services (2022b), o Python é uma linguagem de programação amplamente usada em aplicações *web*, desenvolvimento de *software*, ciência de dados e *Machine Learning* (ML). Os desenvolvedores usam Python porque é eficiente e fácil de aprender e pode ser executado em muitas plataformas diferentes. O *software* Python pode ser baixado gratuitamente, integra-se bem a todos os tipos de sistema e agiliza o desenvolvimento.

2.3 *Front-end*

O *front-end* está muito relacionado com a interface gráfica do projeto. É o ambiente em que se desenvolve a aplicação com a qual o usuário irá interagir diariamente, seja em *software*,

sites, aplicativos etc. Em vista disso, é primordial que o desenvolvedor se preocupe com a experiência do usuário.

Apesar do *front-end* envolver a interface do projeto, o programador não precisa ser formado em *design*. Na verdade, o desenvolvedor *front-end* irá mexer com o código do projeto, enxergando quais ferramentas o usuário precisará interagir. (EQUIPE TOTVS, 2021)

2.4 *Back-end*

O *back-end* está relacionado com a estrutura por trás das aplicações desenvolvidas na programação. Resumindo, tudo aquilo que dá estrutura e apoio às ações do usuário na máquina é chamado de *back-end*.

Quando acessamos um site, por exemplo, por detrás da interface gráfica projetada de maneira esteticamente amigável, há uma comunicação das informações trocadas entre banco de dados e navegador. Portanto, o *back-end* está sempre agindo. (EQUIPE TOTVS, 2020)

2.5 GitHub

De acordo com Ramos (2021), o GitHub é uma "rede social para programadores", sendo também um CVS (controlador de versão) e um serviço de publicação e compartilhamento de códigos para programação. A plataforma foi lançada em 2008, é utilizada mundialmente e, desde 2018, pertence à empresa Microsoft.

Conforme o seu Slogan "*Social Code Host*" (hospedeiro social de códigos, em tradução livre), a base do *site* é armazenar códigos de programação, produzidos por programadores em nível mundial, e compartilhá-los como se fosse uma rede social. Portanto, qualquer usuário cadastrado na plataforma pode divulgar seu trabalho para que outros membros possam visualizar ou contribuir com o seu projeto.

A plataforma pode funcionar como um serviço de colaboração de projetos pessoais e até comerciais. Grandes corporações, como Google e WordPress, usufruem dos recursos do GitHub no que diz respeito às possibilidades de suporte a problemas e até novos desenvolvimentos para suas plataformas.

2.6 Stack Overflow

Melo (2020) define o Stack Overflow como uma plataforma de perguntas e respostas para programadores, sejam ele experientes ou aspirantes na área. Qualquer usuário pode publicar uma dúvida para ser respondida por outros membros da plataforma. As melhores respostas são votadas pela comunidade e exibidas com destaque na questão apresentada.

O *site* originado em 2008, desenvolvido pelos programadores Jeff Altwood e Joel Spolsky, é a principal plataforma da rede de perguntas e respostas *Stack Exchange*, empresa fundada pelos programadores. O *site* também proporciona uma rede de classificados de empregos na área da tecnologia, e uma versão privada para empresas. No ano de 2013, o Stack Overflow lançou uma versão oficial em português.

2.7 *Framework*

Para Barro (2022), os *frameworks* são estruturas compostas por um conjunto de códigos genéricos que permitem o desenvolvimento de sistemas e aplicações. Funcionam como um *template* ou modelo que, quando utilizado, proporciona alguns recursos e elementos estruturais básicos para a geração de uma aplicação ou *software*.

Possibilitam que o processo de desenvolvimento de um projeto não precise ser iniciado totalmente "do zero" pelos desenvolvedores, devido às ferramentas pré-prontas e soluções personalizáveis que os *frameworks* introduzem no desenvolvimento da aplicação.

Por serem desenvolvidos por programadores experientes, os *frameworks* costumam ser seguros e eficientes, além de versáteis – já que podem ser utilizados em aplicações de diferentes tipos.

2.8 Django

Django é um *framework web* Python de alto nível que impulsiona o desenvolvimento rápido e um *design* limpo e pragmático (ACHARYA, 2021). Desenvolvido por desenvolvedores experientes, serve como solução para grande parte do incômodo do desenvolvimento *web*, pois os desenvolvedores podem focar em seu aplicativo sem que precisem "reinventar a roda". (DJANGO, 2022a)

O *framework* originado em 2005 no estado do Kansas, nos EUA, desenvolvido pelos programadores Adrian Holovaty e Simon Willinson, tem como objetivo resolver os problemas mais comuns do processo de desenvolvimento de aplicações *web*, como por exemplo, autenticação, rotas, *object relational mapper* (ORM) e até *migrations.* (ROVEDA, 2021)

2.9 Spring Boot

Para Rossalli (2021), o Spring Boot é um *framework* Java, gratuito e de código aberto, que tem como objetivo facilitar os processos em aplicações Java. Para tal finalidade, ele proporciona agilidade no processo de desenvolvimento, uma vez que os desenvolvedores conseguem reduzir o tempo gasto com as configurações iniciais.

O Spring Boot utiliza um conceito chamado "convenção sobre configuração", o que significa que o *framework* decide a melhor alternativa de se fazer algo. Denomina-se como uma ferramenta opinativa, ou seja, ele toma as decisões no lugar do desenvolvedor, baseando-se em convenções, aplicando padrões e facilitando o desenvolvimento do projeto.

Embora seja uma ferramenta opinativa, ela não é inflexível, permitindo uma configuração diferente do *default* caso o desenvolvedor assim deseje.

O Spring Boot também trouxe uma grande vantagem no desenvolvimento de aplicações, na qual toda a configuração não necessita ser realizada pelos temidos XMLs. A configuração é realizada de forma pragmática por meio de anotações.

3 Trabalhos Similares

No trabalho de Camargos *et al.* (2019), os autores abordaram uma análise comparativa entre *frameworks* da linguagem de programação Javascript, sendo eles o React e o Angular. O objetivo da escrita é apontar qual biblioteca é a mais completa, mostrando dentre as opções descritas a que agrega mais valor em um projeto. Os autores levantaram alguns aspectos de importância primordial, para no decorrer de um projeto servirem de parâmetros de comparação. Realizaram a análise de diversas publicações e estudos sobre o tema, chegando a 16 referências que contribuíram para o desenvolvimento do artigo. Concluiu-se que, apesar de semelhantes, o *framework* React é o mais completo e proporciona um melhor resultado dentro de projetos *web*.

Mais similar ao tema deste trabalho é o texto encontrado em IT GURU (2021), onde também se propõe uma comparação entre os *frameworks* Django e Spring Boot. No entanto, trata-se mais de listar vantagens e desvantagens de cada um deles, bem como das linguagens de

programação que os embasam. Embora o texto seja de fácil leitura e traga informações interessantes, não há comparativos efetivos baseados em critérios que permitam análise mais profunda.

4 Metodologia

A análise comparativa entre os *frameworks* Django e Spring Boot levantou parâmetros importantes que devem ser estudados durante o processo de escolha da biblioteca a utilizar durante o desenvolvimento de uma aplicação *web*.

Alguns parâmetros foram considerados como fundamentais para a análise comparativa dos *frameworks* e escolha do mesmo para o desenvolvimento de um projeto. Esses parâmetros são: curvas de aprendizagem, ambiente de desenvolvimento/instalação, construção de *template*, recursos, facilidade no desenvolvimento, flexibilidade, tempo de carregamento e desempenho da aplicação final.

5 Desenvolvimento

A análise conduzida neste artigo constitui-se em um estudo entre as documentações dos *frameworks* e artigos de desenvolvedores, pesquisadores e outros autores especialistas na área de desenvolvimento *web.*

Expondo-se os aspectos a serem analisados neste trabalho, será identificado qual o *framework* apresentará o melhor resultado para cada cenário.

Inicia-se então examinando a curva de aprendizagem, termo citado pela primeira vez em 1936, por Theodore Wright, que definiu como resultado uma representação gráfica a partir de uma equação matemática que calcula o tempo que a pessoa precisa para adquirir um nível maior de conhecimento em determinado assunto ou tarefa. (PÓS EDUCAÇÃO UNISINOS, 2022)

Sobre a curva de aprendizagem, existem alguns aspectos que podem influenciar o progresso de um desenvolvedor, como por exemplo a popularidade da linguagem de programação utilizada pelo *framework*, os conhecimentos prévios necessários para utilização e a facilidade de desenvolvimento.

O índice *TIOBE Programming Community* (Figura 1) é um indicador da popularidade das linguagens de programação. As classificações são baseadas no número de engenheiros qualificados em todo o mundo, cursos e fornecedores terceirizados. Motores de busca populares como Google, Bing, Yahoo!, Wikipedia, Amazon, YouTube e Baidu são usados para calcular as classificações. O índice pode ser usado para verificar se suas habilidades de programação ainda estão atualizadas ou para tomar uma decisão estratégica sobre qual linguagem de programação deve ser adotada ao começar a construir um novo sistema de *software*. (TIOBE, 2022)

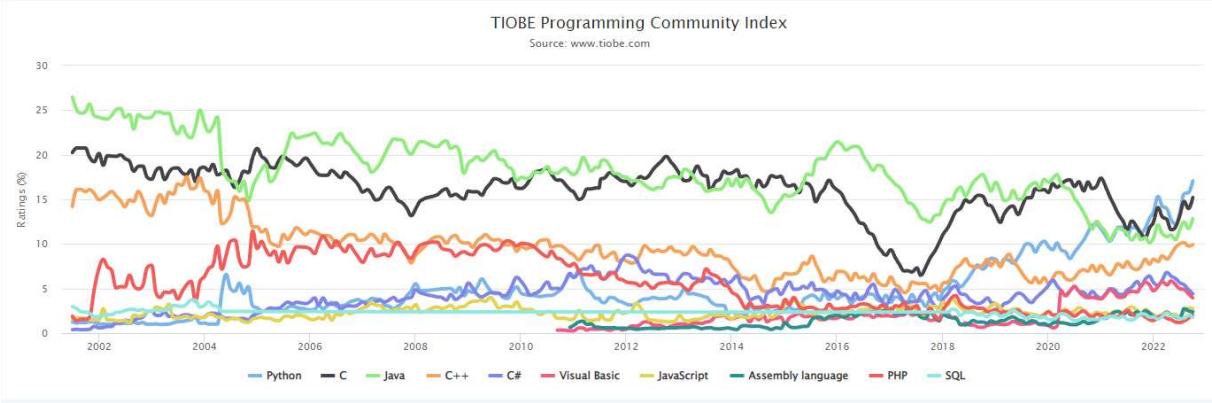

Figura 1 Índice de popularidade *TIOBE Programming Community Index*, outubro de 2022.
Fonte: TIOBE, 2022.

De acordo com Melo (2019), o Python é uma das linguagens de programação mais acessíveis, simples e rápidas de aprender: com poucas linhas de código é possível escrever seus *scripts*. Diante disso, o desenvolvimento utilizando o *framework* Django possui uma facilidade na curva de aprendizagem.

Segundo Souza Júnior (2022), uma das vantagens mais significativas do Java é a sua capacidade de adaptação entre as diversas plataformas. A execução de aplicações em muitos sistemas é essencial para um software *web* ou *desktop*, e o Java consegue isso por ser independente de plataforma nos níveis de código fonte e binário.

Além disso, a linguagem tem uma boa curva de aprendizagem, apesar de ser considerada como "verbosa". Ao longo dos anos surgiram vários *frameworks* que facilitam cada vez mais o desenvolvimento da linguagem. O Spring Boot pode ser citado como um desses *frameworks*, expressando a sua facilidade também na curva de aprendizagem.

Conectada de maneira direta com a curva de aprendizagem está a facilidade no desenvolvimento, que se refere aos recursos que o *framework* disponibiliza ao desenvolvedor objetivando o menor custo possível para o desenvolvimento de uma aplicação.

Seguindo a documentação da ferramenta Django, foi estruturada a Tabela 1 com o objetivo de apresentar os recursos e suas vantagens oferecidas ao desenvolvedor.

Tabela 1 Relação de Recursos e Vantagens da ferramenta Django
Fonte: Django Software Foundation (2022a)

Recursos	Vantagens
Mapeador Objeto-Relacional	Modelos de dados inteiramente definidos em Python, obtendo uma API rica e dinâmica de acesso ao banco de dados, não excluindo a possibilidade de uso da escrita em SQL.
URLs e Visualizações	O Django encoraja um *design* agradável de URL e não coloca nenhuma "sujeira" em URLs, como .php ou .asp. O módulo Python chamado URLconf projeta URLs para aplicativos. Como um índice para seu aplicativo, ele contém um mapeamento simples entre padrões URL e suas visualizações.

Modelos	A linguagem de *template* do Django é projetada para encontrar um equilíbrio entre poder e facilidade. Ele foi projetado para ser confortável e fácil de aprender para aqueles acostumados a trabalhar com HTML, como *designers* e desenvolvedores *front-end*. Mas também é flexível e altamente extensível, permitindo que os desenvolvedores aumentem a linguagem do modelo conforme necessário.
Formulários	Poderosa biblioteca de formulários que trabalham com a renderização de formulários como HTML, validando dados enviados pelo usuário e os convertendo em tipos nativos do Python.
Autenticação	A ferramenta possui um sistema de autenticação seguro e completo, trabalhando com contas de usuários, grupos, permissões e sessões de usuários baseadas em *cookies*. O usuário pode criar facilmente *sites* que permitem aos seus clientes criar contas e fazer *login*/ *logout* com segurança.
Administrador	Mecanismo de administração automática, realiza a leitura de metadados em seus modelos fornecendo uma interface poderosa e pronta para produção, para que os produtores de conteúdo imediatamente possam usar para gerenciar o conteúdo em seu *site*. Fácil de configurar e com vários recursos para personalização.
Internacionalização	A ferramenta permite que os desenvolvedores e autores de modelos especifiquem quais partes de seus aplicativos devem ser traduzidas ou formatadas para idiomas e culturas locais e usa esses ganchos para localizar aplicativos *web* para usuários específicos de acordo com suas preferências.
Segurança	O Django fornece várias proteções contra *clickjacking* (técnica maliciosa para induzir um usuário a clicar em algo diferente do que o usuário percebe), *script* entre *sites*, falsificação de solicitações entre sites (CSRF), injeção SQL e execução remota de código.

Diferentemente da documentação apresentada pela ferramenta Django, o Spring Boot exibe em sua documentação os seus recursos e alguns guias para outras funcionalidades do *framework*.

Os principais recursos disponibilizados pelo Spring Boot são (SPRING, 2022):

a) criação de aplicativos Spring autônomos;
b) incorporação direta do Tomcat, Jetty ou Undertow (sem necessidade de implantar arquivos WAR);
c) fornecimento de dependências “iniciais” opinativas para simplificar a configuração de compilação;

d) configuração automática de bibliotecas Spring e de terceiros sempre que possível;
e) fornecimento de recursos prontos para produção, como métricas, verificações de integridade e configuração interna; e
f) absolutamente nenhuma geração de código e nenhum requisito para configuração XML.

Alguns guias fornecidos pelo Spring Boot são (SPRING, 2022):

a) Spring Boot com Docker;
b) Acessando dados com MySQL;
c) Testando a camada *web*;
d) Criando um projeto de vários módulos; e
e) Implantando um aplicativo Spring Boot no Azure.

Ainda procedendo aspectos relacionados ao início da experiência *Dev-Framework*, foi realizado o estudo das ferramentas quanto à facilidade de instalação e criação do ambiente de desenvolvimento.

Segundo o guia de início rápido do Spring, encontrado na documentação da plataforma, preparar o ambiente com Spring Boot é simples e rápido graças à ferramenta Spring Initializr (Figura 2). É necessário dispor de um ambiente de desenvolvedor integrado (IDE), tal como Visual Studio Code, Eclipse, IntelliJ, Spring Tools ou outros, além da instalação do JDK, o *kit* de desenvolvimento Java (SPRING, 2022).

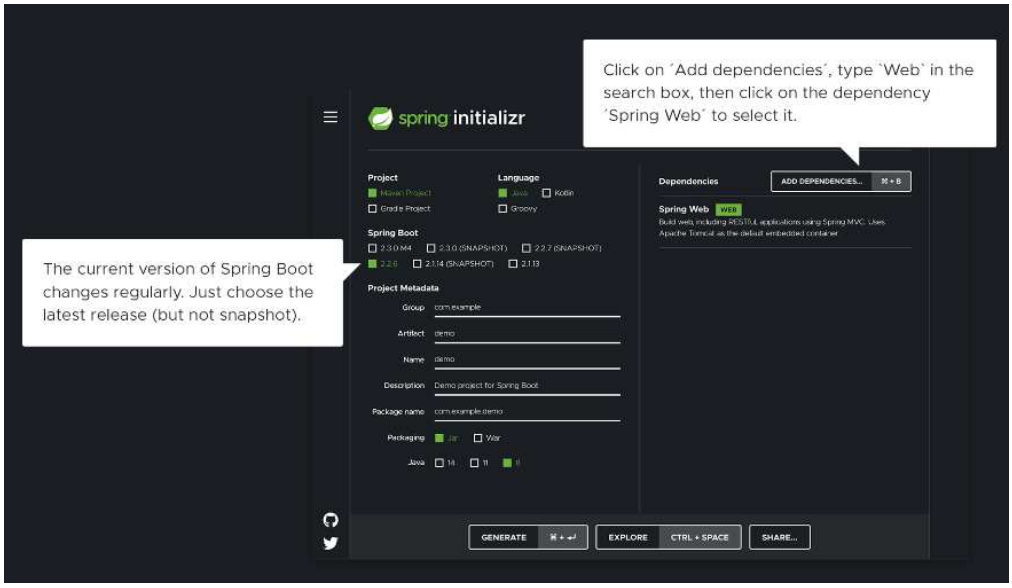

Figura 2 Tela de início da ferramenta Spring Initializr
Fonte: https://start.spring.io/

Já para o Django, segundo a sua documentação oficial, é preciso realizar a sua instalação através das linhas de comando. O requisito prévio para instalação do *framework* é possuir um ambiente de trabalho com o Python e seus componentes pip e venv instalados. (DJANGO SOFTWARE FOUNDATION, 2022b)

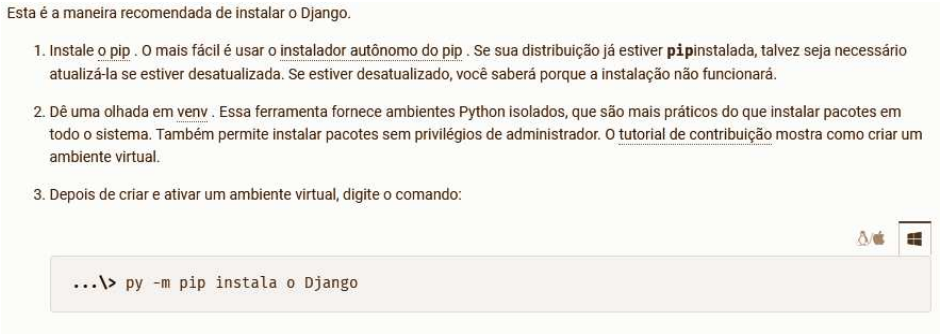

Figura 3 *Print* do método de instalação da ferramenta Django
Fonte: https://docs.djangoproject.com/en/4.1/topics/install/#installing-official-release

Outro recurso importante para o desenvolvimento de uma aplicação é a reutilização de código. E um dos melhores modos para se utilizar este recurso é trabalhando com a construção de *templates*, o que é considerado outro aspecto importante para análise neste artigo.

De acordo com Souza (2020), um *template* é utilizado para fazer com que o *site* tenha a cara da empresa e as funções aplicadas por ela na internet. Além do conteúdo, o que chama a atenção do usuário pela primeira vez é a forma como estão disponibilizados os elementos na tela, assim como cores, formatos e fontes.

O Spring Boot utiliza-se do padrão MVC (*Model View Controller*) para construção de *templates*, proporcionando como benefício isolar as regras de negócios da lógica de apresentação, a interface com o usuário (Figura 4). Assim, várias interfaces com o usuário podem ser modificadas sem que haja a necessidade da alteração das regras de negócios, proporcionando muito mais flexibilidade e oportunidades de reuso das classes. (MÁRCIO, 2011)

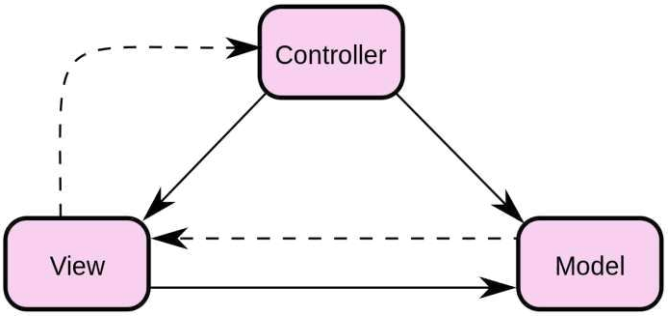

Figura 4 Padrão MVC (*Model View Controller*)
Fonte: https://pt.wikipedia.org/wiki/MVC

O *Django Template Language* (DTL) é o *template engine* padrão do Django, com o objetivo de auxiliar na criação de páginas HTML em suas aplicações. O DTL faz com que a troca de informações entre uma aplicação Django e suas páginas HTML seja realizada de forma mais simples e intuitiva. Um *template engine*, se comparado com o HTML puro, apresenta maior velocidade na criação de *template*, melhor reaproveitamento do código HTML, uso de tipos de dados nativos em páginas HTML e uso de recursos de linguagens de programação em páginas HTML. (ANDRADE, 2021)

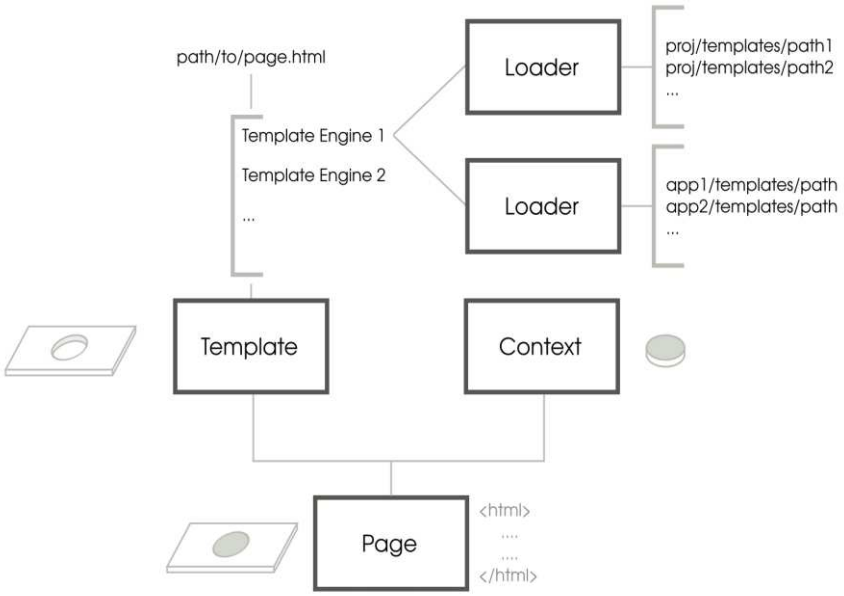

Figura 5 Padrão DTL (*Django Template Engine*)
Fonte: https://subscription.packtpub.com/book/programming/9781788831345/5/ch05lvl1sec40/how-templates-work

Outro aspecto significativo a levar em consideração na decisão de sua ferramenta é a flexibilidade, destacada como um dos pilares da qualidade de um *software*. Este aspecto representa a capacidade de um *software* realizar alterações para trabalhar com grandes cargas.

Do ponto de vista do *framework* Django, o *hardware* é barato se comparado com o tempo de desenvolvimento. Portanto a ferramenta foi projetada para aproveitar o máximo de *hardware* possível.

O Django usa uma arquitetura *shared-nothing*, o que significa que se pode adicionar *hardware* em qualquer nível – servidores de banco de dados, servidores de *cache* ou servidores *web*/aplicativos. A estrutura separa componentes de forma clara, como sua camada de banco de dados e camada de aplicativo. E vem com uma estrutura de *cache* simples, porém poderosa. (DJANGO SOFTWARE FOUDATION, 2022)

O Spring Boot baseia-se no Spring, tratando de pegar as partes existentes do Spring *framework*, configurando-as e empacotando-as com o mínimo de trabalho de desenvolvimento necessário. (MICROSOFT AZURE, 2022)

De certa forma, o Spring Boot contém alguns princípios apresentados na ferramenta Spring, como a flexibilidade que suporta uma ampla gama de necessidades de aplicação com diferentes perspectivas, a forte compatibilidade com versões anteriores suportando uma gama de versões de JDK e bibliotecas de terceiros. Um dos objetivos declarados do Spring é criar APIs que sejam intuitivas e que se sustentem em muitas versões e muitos anos. (VERSIANI, 2022)

Examinando o desempenho, foram ponderados alguns aspectos fundamentais, como o tempo de carregamento, uso da CPU, alocação de memória e espaço de armazenamento. Foram desenvolvidos dois projetos referente a cada *framework* apresentado. Ambos os projetos foram estruturados com o objetivo de apresentar um esquema CRUD (*Create, Read, Update and Delete*) básico. O ambiente aplicado para testes é um Desktop Windows 10 Pro (Intel(R) Core (TM) i3-9100F CPU @ 3.60GHz, 8GB RAM, 64 bits) e as versões dos *frameworks* são Django 4.1.3 e Spring Boot 2.7.2. A Tabela 2 resume os resultados comparativos de desempenho nestas condições.

Tabela 2 Amostra de desempenho dos *frameworks*

	Frameworks	
Aspectos	**Django**	**Spring Boot**
Tempo de Carregamento (s)	3.19	4.656
Uso da CPU	6,9%	6,7%
Alocação de Memória	651,7 MB	1.739,5 MB
Espaço de Armazenamento	1.129,5 MB	740,5 MB

6 Resultados e Discussões

O presente artigo, elaborado através do estudo realizado em pesquisas utilizando *sites*, documentações e outros artigos, exibiu como resultados algumas tabelas, gráficos e imagens. Esses resultados têm como objetivo auxiliar o desenvolvedor na escolha do melhor *framework* a ser utilizado em suas aplicações.

Com base nos resultados, foram determinadas algumas semelhanças e diferenças entre as duas ferramentas. Pode-se destacar, entre as semelhanças identificadas, que ambas são gratuitas e de código aberto, são populares, possuem uma ampla documentação e proporcionam agilidade e rapidez no desenvolvimento de aplicações *web*. Entre as diferenças, é possível identificar que o Spring Boot é superior nos aspectos de segurança, e consegue atender múltiplas solicitações, diferente do Django que tem como superioridade a construção de *templates*.

Para construir a Tabela 2, foram realizados dois projetos para análise de desempenho dos *frameworks*. Como resultado, é possível definir (nas condições descritas) que o *framework* Django apresentou um tempo de carregamento mais rápido e uma alocação de memória muito menor, e o Spring Boot apresentou menor uso da CPU e a utilização de um espaço de armazenamento menor.

A Tabela 3 resume, de maneira tangível, os resultados encontrados ao longo do trabalho, considerados os critérios que foram elencados no Capítulo 4 (Metodologia). Cada critério é valorado da seguinte maneira: um indicativo de sua importância relativa (o “peso” do critério na avaliação) e uma “nota” alcançada pelo framework com respeito a esse critério. Tanto o “peso” quanto a nota são subjetivos e baseados na análise específica aqui realizada. A multiplicação da nota pelo peso produz a nota ponderada.

Tabela 3 Pontuação subjetiva dos resultados

		Frameworks			
		Django		**Spring Boot**	
Critério	**Peso**	**Nota**	**Nota Ponderada**	**Nota**	**Nota Ponderada**
Curva de aprendizagem	20%	4	80	3	60
Ambiente de desenvolvimento/ Instalação	15%	2	30	5	75
Construção de *template*	10%	5	50	3	30
Recursos e facilidade no desenvolvimento	15%	5	75	4	60
Flexibilidade	15%	4	60	5	75
Desempenho da aplicação final	25%	4	100	5	125
Totais	**100%**	**(inaplicável)**	**395**	**(inaplicável)**	**425**

* Escala de notas: {1 (péssimo), 2 (ruim), 3 (regular), 4(bom), 5 (excelente)}

De acordo com a Tabela 3, verifica-se que o *framework* Spring Boot demonstrou superioridade nos critérios de Ambiente de desenvolvimento/Instalação, Flexibilidade e Desempenho da aplicação final. E a ferramenta Django apresentou predominância nos aspectos de Curva de aprendizagem, construção de *template* e Recursos e facilidade no desenvolvimento.

7 Conclusão

Compreende-se que o Django e Spring Boot são *frameworks* famosos, cada um com o seu conjunto único de vantagens e desvantagens. Se o critério principal é a linguagem de programação que embasa um projeto, é esperado que desenvolvedores experientes em Java adotem o Spring Boot, enquanto o Django será preferido por aqueles experientes em Python.

Para desenvolvedores iniciantes, ambas as linguagens são atraentes e excelentes para o aprendizado: Java tem como pontos fortes a segurança, confiabilidade e flexibilidade, mas o Python tem ganhado muito espaço em aplicações envolvendo análise de dados e aprendizado de máquina.

O presente trabalho comprometeu-se a indicar qual o *framework* mais completo para o desenvolvimento *web*. Naturalmente, mesmo com todas as informações aqui reunidas não é possível apontar um "vencedor" absoluto, principalmente porque qualquer critério de comparação envolve grande parcela de intangibilidade. Para contornar esta condição, realizaram-se testes de desempenho e desenvolveu-se uma matriz de pontuação que atribuiu

notas aos *frameworks*. Esta matriz, mesmo simplificada e recorrendo a valores numéricos estipulados arbitrariamente, é coerente com o trabalho de pesquisa realizado,

A partir dos resultados obtidos nas Tabelas 2 e 3, pode-se concluir que o Spring Boot apresentou uma superioridade em relação ao Django. Essa predominância está apontada no fato de que o Spring Boot mostrou um rendimento superior nos aspectos de Ambiente de desenvolvimento/Instalação, Flexibilidade e Desempenho da aplicação final, além de apresentar uma baixa inferioridade nos critérios de Curva de aprendizagem e Recursos e facilidade no desenvolvimento. Concluindo, nas circunstâncias aqui observadas, o *framework* Spring Boot mostra-se mais completo.

Referências

ACHARYA, Durga Prasad. **Django vs Laravel**: qual é o melhor framework em 2022? 2021. Blog elaborado para a empresa KINSTA, tag "Django é um framework web, mantido pela Django Software Foundation". Disponível em: https://kinsta.com/pt/blog/django-vs-laravel/#:~:text=Django%20é%20um%20framework%20web,mantido%20pela%20Django%20Software%20Foundation. Acesso em: 16 set. 2022.

AMAZON WEB SERVICES. **O que é Java?**: guia de Enterprise Java para iniciantes. 2022. Disponível em: https://aws.amazon.com/pt/what-is/java/#:~:text=Java%20é%20uma%20linguagem%20multiplataforma,data%20e%20tecnologias%20do%20servidor.. Acesso em: 16 set. 2022.

AMAZON WEB SERVICES. **O que é Python?**: guia de Python para iniciantes na nuvem. 2022. Disponível em: https://aws.amazon.com/pt/what-is/python/#:~:text=O%20Python%20é%20uma%20linguagem,executada%20em%20muitas%20plataformas%20diferentes.. Acesso em: 16 set. 2022.

ANDRADE, Ana Paula de. **O que é Django Template Language?** 2021. Artigo elaborado para TreinaWeb. Disponível em: https://www.treinaweb.com.br/blog/o-que-e-django-template-language. Acesso em: 31 out. 2022.

BARRO, Bruna B. **O Que São Frameworks e Quais os Mais Utilizados**. 2022. Tutorial elaborado para a empresa HOSTINGER. Disponível em: https://www.hostinger.com.br/tutoriais/frameworks. Acesso em: 16 set. 2022.

CAMARGOS, João Gabriel Colares de *et al.* Uma Análise Comparativa entre os Frameworks Javascript Angular e React. **Computação & Sociedade**, FUMEC, Belo Horizonte (MG), v. 1, p. 101-113, 24 set. 2019. Disponível em: http://201.48.93.203/index.php/computacaoe sociedade/article/view/7307. Acesso em: 16 set. 2022.

DJANGO SOFTWARE FOUNDATION. **FAQ General**: Django documentation. 2022. Página de FAQ, tag "Does Django scale?". Disponível em: https://docs.djangoproject.com/en/4.1/faq/general/#does-django-scale. Acesso em: 31 out. 2022.

DJANGO SOFTWARE FOUNDATION. **Getting started with Django**. 2022. Disponível em: https://www.djangoproject.com/start/. Acesso em: 31 out. 2022.

EQUIPE TOTVS. **Front end**: o que é, como funciona e qual a importância. 2021. Disponível em: https://www.totvs.com/blog/developers/front-end/#:~:text=Front-end%3A%20o%20que%20é,%2C%20sites%2C%20aplicativos%2C%20etc. Acesso em: 16 set. 2022.

EQUIPE TOTVS. **O que é back-end e qual seu papel na programação**. 2020. Blog da categoria "Developers" mantido pela empresa TOTVS. Disponível em: https://www.totvs.com/blog/developers/back-end/#:~:text=Back-end%20se%20relaciona%20com,é%20chamado%20de%20back-end.. Acesso em: 16 set. 2022.

IT GURU. **Spring Boot VS Django**: what's the difference in 2021? 2021. Blog mantido pela empresa IT GURU. Disponível em: https://onlineitguru.com/blog/spring-boot-vs-django-what%27s-the-difference-in-2021. Acesso em: 16 set. 2022.

MÁRCIO. **Padrão MVC (Model-View-Controller)**: tutorial. 2011. Artigo elaborado para DevMedia. Disponível em: https://www.devmedia.com.br/padrao-mvc-java-magazine/21995. Acesso em: 31 out. 2022.

MELO, Carlos. **5 Motivos para você aprender Python hoje**. 2019. Blog elaborado para SIGMOIDAL, tag "Curva de Aprendizado – O Python, a linguagem que mais cresceu nos últimos anos". Disponível em: https://sigmoidal.ai/5-motivos-para-voce-aprender-python/#:~:text=Curva%20de%20Aprendizado%20–%20O%20Python,mais%20cresceu%20nos%20últimos%20anos. Acesso em: 31 out. 2022.

MELO, Diego. **O que é Stack Overflow**. 2020. Texto elaborado para a empresa TECNOBLOG. Disponível em: https://tecnoblog.net/responde/o-que-e-stack-overflow/#:~:text=O%20Stack%20Overflow%20é%20uma%20plataforma%20gratuita%20de%20perguntas%20e,pelos%20outros%20membros%20do%20site.. Acesso em: 16 set. 2022.

MICROSOFT AZURE. **O que é o Java Spring Boot?**: introdução ao Spring Boot. 2022. Disponível em: https://azure.microsoft.com/pt-br/resources/cloud-computing-dictionary/what-is-java-spring-boot. Acesso em: 31 out. 2022.

PÓS EDUCAÇÃO UNISINOS. **O que é a curva de aprendizagem**. 2021. Blog mantido por Pós Educação UNISINOS. Disponível em: https://poseducacao.unisinos.br/blog/curva-aprendizagem. Acesso em: 31 out. 2022.

RAMOS, Guilherme. **O que é o GitHub?**: veja para que serve a ‘rede social de programadores’. 2021. Texto elaborado para a empresa TECHTUDO. Disponível em: https://www.techtudo.com.br/listas/2021/05/o-que-e-o-github-veja-para-que-serve-a-rede-social-de-programadores.ghtml. Acesso em: 16 set. 2022.

ROSSALI, Bárbara. **Spring Boot**: como começar.2021. Disponível em: <https://www.zup.com.br/blog/spring-boot#:~:text=O%20Spring%20Boot%20é%20um,gasto%20com%20as%20configurações%20iniciais.>. Acesso em 16 de setembro de 2022.

ROVEDA, Ugo. **O que é Django, para que serve e como usar este framework**. 2021. Blog da categoria "Tecnologia" elaborado para a empresa KENZIE. Disponível em: https://kenzie.com.br/blog/django. Acesso em: 16 set. 2022.

SOUZA JÚNIOR, Alexandre Marinho de. **Especialista Java**: como se tornar um? 2022. Artigo elaborado para LYNCAS. Disponível em: https://lyncas.net/especialista-java/. Acesso em: 31 out. 2022.

SOUZA, Ivan de. **Template**: o que é, para que serve e como aplicar no site da sua empresa. 2020. Disponível em: https://rockcontent.com/br/blog/template. Acesso em: 31 out. 2022.

SPRING. **Spring**: guides. 2022. Disponível em: https://spring.io/guides. Acesso em: 31 out. 2022.

STACK OVERFLOW. **Stack Overflow Developer Survey 2020**. 2020. Informações atualizadas anualmente. Disponível em: https://insights.stackoverflow.com/survey/2020#most-popular-technologies. Acesso em: 16 set. 2022.

TIOBE. **TIOBE Index for October 2022**. 2022. Disponível em: <https://www.tiobe.com/tiobe-index/>. Acesso em 31 de outubro de 2022.

VERSIANI, Rafael. **Spring Framework**: descubra o que é, seus módulos e exemplos! 2022. Disponível em: https://enotas.com.br/blog/spring-framework. Acesso em: 31 out. 2022.