A EVOLUÇÃO DOS FRAMEWORKS DE BACK END: UMA JORNADA HISTORICA DESDE OS PRIMORDIOS ATE OS FRAMEWORKS MODERNOS

grupo:
Bruno Morais - 01604282
Eytor Bezerra - 01610281
Evânnio Barbosa maia de Sant'ana JR - 01603302
Ícaro Vinicius do Nascimento Silva - 01599214
João Paulo Guerra - 01606505
Rodrigo Roberto de Lima Pereira - 01597883

Sumário

frameworks de back-end:

- **Definição**: Plataformas que facilitam o desenvolvimento do back-end, oferecendo uma estrutura pré-definida para a construção de aplicações robustas.
- **Função**: Gerenciar requisições, rotas, segurança, conexão com bancos de dados e lógica do servidor.

Vantagens:

- Acelera o desenvolvimento.
- Facilita a manutenção e escalabilidade.
- Fornece segurança integrada.

Os primeiros passos :

Surgimento da Programação Estruturada e Paradigmas Iniciais

- Década de 1960: Desenvolvimento de paradigmas estruturados
- Programação baseada em procedimentos e controle de fluxo
- Linguagens pioneiras: Fortran, Cobol, Pascal

Desenvolvimento de Aplicações Sem Frameworks

- Código manual e repetitivo
- Desafios na organização e manutenção do código
- Falta de abstração: cada desenvolvedor criava suas próprias soluções do zero.

Os primeiros passos :

Primeiras Linguagens de Suporte ao Back-End

- C (1972): Alta performance e flexibilidade
- Cobol (1959): Usado em sistemas empresariais e bancários
- Pascal (1970): Projetado para boa estruturação de código

Desafios da Era Pré-Framework

- Complexidade no desenvolvimento e escalabilidade
- Falta de padronização entre projetos
- Longos tempos de desenvolvimento

Anos 1990- iNÍCIO DA WEB DINÂMICA

Nos anos 1990, a web era predominantemente estática, com HTML simples. No entanto, o surgimento de linguagens de script do lado do servidor, como CGI (Common Gateway Interface) com Perl, permitiu a criação de conteúdo dinâmico.

- **Principais Tecnologias: CGI (com Perl ou C), ASP (Active Server Pages) da Microsoft, e PHP, que ganhou popularidade no final da década como uma solução fácil e de código aberto para a construção de sites dinâmicos.**

- **CGI (Common Gateway Interface):**
  - Um dos primeiros métodos para executar scripts no servidor (década de 1990)
  - Permitia a interação entre servidor e navegador através de linguagens como Perl e C
  - Desafios: scripts independentes, performance limitada, segurança precária

Anos 1990- iNÍCIO DA WEB DINÂMICA

Linguagens Emergentes para Web Dinâmica

- Perl (1987):
  - Forte em manipulação de texto, amplamente usado para CGI
  - Base para muitos scripts dinâmicos no início da web

- PHP (1994):
  - Facilita a criação de páginas dinâmicas sem a necessidade de scripts CGI complexos
  - Cresceu rapidamente devido à sua simplicidade e integração com HTML

Meados dos Anos 2000 - Adoção em Massa de Frameworks MVC

Com o crescimento das aplicações web, a arquitetura MVC (Model-View-Controller) se tornou o padrão de fato para organizar a separação de responsabilidades.

- **Django (2005)**: Um framework web baseado em Python que se destacou por sua simplicidade, segurança e foco na "reutilização de componentes". Muito popular para desenvolvimento rápido.

Meados dos Anos 2000 - Adoção em Massa de Frameworks MVC

- **Spring (2002)**: Surgiu no ecossistema Java, oferecendo uma estrutura completa para o desenvolvimento empresarial e altamente escalável. Tornou-se um dos padrões no desenvolvimento corporativo.

- **ASP.NET (2002)**: A Microsoft introduziu o ASP.NET, uma estrutura poderosa para criar aplicações web dinâmicas na plataforma .NET, que evoluiu ao longo dos anos para o moderno ASP.NET Core.

- **Ruby on Rails (2005)**:baseado em Ruby, introduziu a filosofia "Convention over Configuration", reduzindo a necessidade de configurações manuais. . Sua simplicidade e agilidade revolucionaram o desenvolvimento web.

Final dos Anos 2000 e Início dos Anos 2010 - Explosão de Frameworks JavaScript

Com o aumento do uso de JavaScript no front-end e a popularização de técnicas como AJAX, frameworks voltados para o back-end começaram a integrar mais a comunicação entre cliente e servidor.

- **Node.js (2009)**: Uma das maiores revoluções no back-end. Com o Node.js, foi possível executar JavaScript no lado servidor, abrindo caminho para a criação de aplicações em tempo real e frameworks baseados em event-driven. Node popularizou o non-blocking I/O, tornando-o altamente escalável.

- **Frameworks como Express.js (2010) e Meteor (2012), ambos baseados em Node.js, começaram a surgir, proporcionando estruturas leves e rápidas para desenvolvimento.**

Meados dos Anos 2010 - Microserviços e Frameworks Mais Leves

O conceito de microserviços começou a dominar a arquitetura de sistemas, movendo os desenvolvedores de grandes monólitos para aplicações compostas por vários serviços menores.

- **Spring Boot (2014)**: No ecossistema Java, o Spring Boot simplificou a criação de aplicações baseadas em Spring, tornando o desenvolvimento de microserviços muito mais rápido e eficiente.

- **Flask (2010)**: Para desenvolvedores Python, o Flask surgiu como uma alternativa leve ao Django, permitindo maior flexibilidade e controle sobre a aplicação, especialmente útil para microserviços.

Final dos Anos 2010 e Início dos Anos 2020 - Serverless e Frameworks Full Stack

Com a chegada de plataformas serverless, como AWS Lambda e Google Cloud Functions, desenvolvedores começaram a construir aplicações sem precisar gerenciar a infraestrutura do servidor.

- **Frameworks Serverless**: Ferramentas como o Serverless Framework começaram a facilitar o desenvolvimento e a implantação de funções serverless em nuvens públicas.

- **NestJS (2017)**: Um framework TypeScript para Node.js, baseado em módulos e inspirado em padrões do Angular, que rapidamente ganhou popularidade devido ao seu foco em arquitetura escalável e orientada a microserviços.

2020 e Além - APIs, Microserviços e Edge Computing

Atualmente, frameworks focam em alta performance, integração com API REST e GraphQL, além de edge computing, com uma crescente demanda por baixa latência e distribuição global.

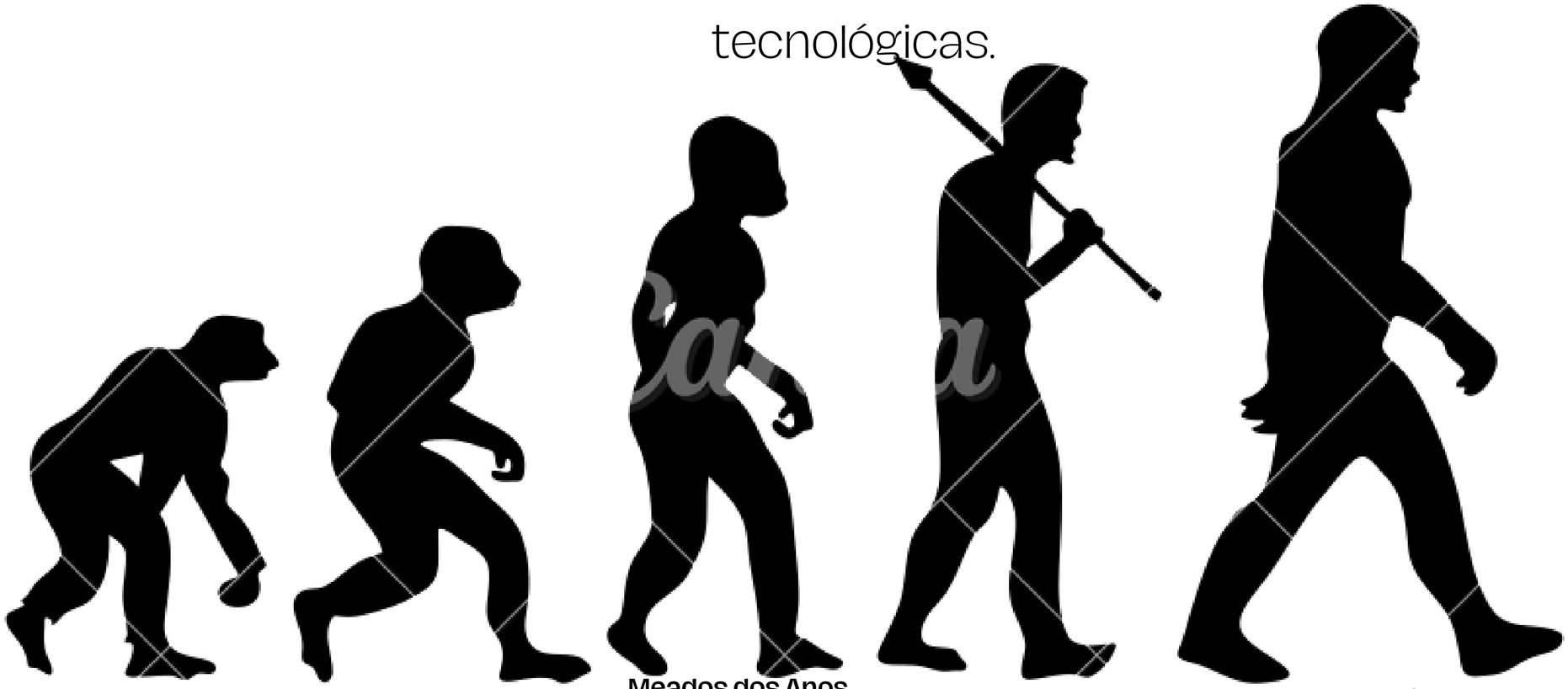
- **FastAPI (2018):** Um framework Python que cresceu rapidamente em popularidade devido ao seu desempenho (usando asyncio) e suporte nativo a OpenAPI para a criação de APIs RESTful de alta performance.

- **Deno (2020):** Um sucessor potencial para o Node.js, criado pelo mesmo criador, focado em segurança e modernidade, suportando TypeScript diretamente e com melhorias na execução de módulos.

- **ASP.NET Core:** Evoluiu como uma plataforma open-source, leve e multiplataforma, focando em desempenho e escalabilidade.

exemplo prático

Vamos considerar uma empresa fictícia chamada “TechStore”, que vende produtos online e precisa adaptar seu sistema conforme o crescimento e as mudanças tecnológicas.



Anos 1990: Tecnologias básicas como CGI para um sistema simples

Início dos Anos 2000: Frameworks MVC como CakePHP para melhor estrutura e manutenção.

Meados dos Anos 2000: Microserviços e frameworks como Django para suportar crescimento e complexidade.

Final dos Anos 2010: Node.js e serverless para melhorar performance e escalabilidade.

2020 e Além: FastAPI e edge computing para otimização e inovação contínua.

Resumo:

Anos 1990

Tecnologias primitivas como CGI e ASP.

Anos 2000

Surgem frameworks MVC e linguagens como Ruby e Python crescem com Rails e Django.

Anos 2010

Node.js e a arquitetura de microserviços transformam o desenvolvimento web.

Anos 2020

Serverless, frameworks mais performáticos e o avanço de edge computing dominam o cenário.

Referências:

OLIVEIRA JÚNIOR, Laércio Germano de. ATLOM.JS: um Framework NODE.JS para aplicações Web baseado em componentes. 2017. 81 f. Trabalho de Conclusão de Curso (Bacharelado em Engenharia de Software) – Universidade Federal do Ceará, Quixadá, 2017. Disponível em: http://repositorio.ufc.br/handle/riufc/29556. Acesso em: 16 set. 2024.

CORDEIRO, Alexandre Costa. Dyfocus: Desenvolvimento do back-end de um aplicativo mobile para smartphone. 2014. 71 f. Trabalho de Conclusão de Curso (Graduação em Engenharia de Controle e Automação) – Universidade Federal de Santa Catarina, Centro Tecnológico, Florianópolis, 2014. Disponível em: https://repositorio.ufsc.br/handle/123456789/169957. Acesso em: 16 set. 2024.

Referências:

SANTOS, Giovanni Almeida. Evolução de um framework para a construção de aplicações de gerência de falhas em redes de computadores. 2001. 116 f. Dissertação (Mestrado em Informática) – Universidade Federal da Paraíba, Centro de Ciências e Tecnologia, Departamento de Sistemas e Computação, Campina Grande, 2001. Disponível em: https://dspace.sti.ufcg.edu.br:8080/jspui/handle/riufcg/11958. Acesso em: 16 set. 2024.

Referências:

LEITE, Raquel Machado.
Framework conceitual para a construção de interfaces persuasivas educacionais: estratégia computacional no modelo inovador de ensino. 2018. 101 f. Dissertação de mestrado apresentada
ao Programa de Pós-Graduação em
Computação da Universidade Federal
do Rio Grande, como requisito parcial
para a obtenção do grau de Mestre em
Engenharia de Computação Universidade Federal do Rio Grande (FURG), Rio Grande, 2018. Disponível em: https://www.repositorio.furg.br/handle/1/8453. Acesso em: 16 set. 2024.

Obrigado!