ARQUITETURA DE SOFTWARE

João Choma Neto
joao.choma@unicesumar.edu.br
https://github.com/JoaoChoma/arquitetura-software
Unicesumar – Maringá

2 BIMESTRE

- Arquitetura monolítica
- Arquitetura em camadas
- Arquitetura cliente-servidor
- Arquitetura baseada em serviços
- Arquitetura de microserviços

ARQUITETURA MONOLÍTICA

MONOLÍTICA

- A arquitetura monolítica é um padrão de arquitetura de software em que uma aplicação é construída como uma única unidade de implementação e implantação.

COMPOSIÇÃO

- **Interface do Usuário (UI):**
  - A camada de interface do usuário na arquitetura monolítica é responsável por interagir com os usuários finais.
- **Lógica de Negócios:**
  - A camada de lógica de negócios contém as regras e processos de negócios da aplicação. Essa camada é responsável por manipular as solicitações do usuário.
- **Acesso a Dados:**
  - A camada de acesso a dados é responsável por interagir com o banco de dados ou outros sistemas de armazenamento de dados.

MONOLÍTICO

A arquitetura monolítica refere-se à forma como uma aplicação é estruturada e implantada.

Em uma arquitetura monolítica, todos os componentes da aplicação são agrupados e implantados juntos como uma única unidade.

Uma aplicação monolítica pode ou não seguir o padrão MVC

MVC

- O padrão MVC é uma forma de estruturar o código dentro de uma aplicação
- O padrão MVC é independente da forma como a aplicação é implantada.
- O MVC pode ser aplicado tanto em arquiteturas monolíticas quanto em arquiteturas distribuídas, como microserviços

BLOG

- Um exemplo comum de aplicação implementada com arquitetura monolítica é um blog
- Um blog monolítico é uma aplicação web que permite aos usuários criar, editar, publicar e visualizar postagens de blog.
- No caso de um blog monolítico, todas as funcionalidades são implementadas como parte de uma única aplicação. Isso inclui o frontend (interface de usuário), backend (lógica de negócios) e acesso a dados (gerenciamento de banco de dados). Todas essas partes são desenvolvidas, testadas e implantadas juntas como uma unidade coesa.

ARQUITETURA EM CAMADAS

CAMADAS

- A arquitetura em camadas é um padrão de design de software que organiza um sistema em camadas distintas, onde cada camada tem uma responsabilidade específica e se comunica apenas com camadas adjacentes.
- Essa abordagem visa separar as preocupações e promover a modularidade, flexibilidade e reusabilidade do código.

CAMADAS

- Uma das principais vantagens da arquitetura em camadas é o isolamento de funcionalidades relacionadas em camadas distintas.
- Isso facilita a reutilização de código, uma vez que as funcionalidades podem ser encapsuladas em módulos ou componentes que podem ser facilmente compartilhados entre diferentes partes do sistema.

CAMADAS

- A arquitetura em camadas está intimamente relacionada a outros padrões de design de software, como MVC (Model-View-Controller), MVVM (Model-View-ViewModel)

REDE DE COMUNICAÇÃO

- Um dos modelos mais conhecidos que utiliza a arquitetura em camadas é o Modelo de Referência OSI (Open Systems Interconnection) e o Modelo TCP/IP.

CLIENTE-SERVIDOR

CLIENTE-SERVIDOR

- A arquitetura cliente-servidor é um modelo de computação distribuída em que as responsabilidades e funcionalidades de um sistema de software são divididas entre dois tipos de entidades: o cliente e o servidor.
- O cliente e o servidor são programas ou dispositivos de computação que interagem entre si por meio de uma rede, como a internet.

CLIENTE-SERVIDOR

- A comunicação entre o cliente e o servidor ocorre por meio de protocolos de comunicação padrão, como HTTP, TCP/IP, WebSocket, etc.
- O cliente envia uma requisição ao servidor, especificando o serviço desejado e quaisquer parâmetros necessários.
- O servidor processa a requisição e retorna uma resposta adequada, que pode incluir dados solicitados, confirmação de operações, mensagens de erro, etc.

EMAIL

- Um exemplo comum de aplicação implementada com arquitetura cliente-servidor é um sistema de correio eletrônico (e-mail)
- O cliente seria o programa de e-mail instalado no dispositivo do usuário, como um aplicativo de e-mail em um computador ou dispositivo móvel. O cliente permite ao usuário enviar, receber, ler e gerenciar e-mails.
- O servidor é responsável por armazenar, gerenciar e distribuir os e-mails entre os clientes. Ele executa aplicativos de servidor de e-mail que lidam com o envio, recebimento, armazenamento e entrega de e-mails

BASEADA EM SERVIÇOS

BASEADA EM SERVIÇOS

- A Arquitetura Orientada a Serviços (SOA) é um estilo arquitetural que preconiza a criação de sistemas compostos por serviços independentes
- Esses serviços são projetados para serem autônomos, autocontidos e altamente reutilizáveis.

BASEADA EM SERVIÇOS

- Na SOA, os serviços geralmente são granulares e podem ser implementados de várias maneiras, como por meio de APIs, componentes, bibliotecas ou aplicativos
- REST (Representational State Transfer) segue o modelo baseado em serviços

COMERCIO ELETRÔNICO

- Um exemplo de aplicação implementada com arquitetura baseada em serviços (SOA - Service-Oriented Architecture) é um sistema de comércio eletrônico.
  1. Serviços de Autenticação
  2. Serviço de Catálogo de Produtos
  3. Serviço de Carrinho de Compras
  4. Serviço de Pagamento
  5. Serviço de Envio
  6. Serviço de Análise de Dados

MICROSERVIÇOS

MICROSERVIÇOS

- Microsserviços são uma abordagem específica para implementar arquiteturas baseadas em serviços, mas com algumas diferenças fundamentais em relação à SOA.

MICROSERVIÇOS

- Em uma arquitetura de microsserviços, um sistema é composto por vários serviços independentes e autônomos, cada um representando uma única funcionalidade ou conjunto de funcionalidades relacionadas.
- Cada microsserviço é implantado e dimensionado de forma independente, e se comunica com outros serviços por meio de protocolos de comunicação padrão, geralmente HTTP/HTTPS

UBER EATS

- Um exemplo de aplicação implementada com arquitetura de microserviços é um sistema de entrega de alimentos online, como o Uber Eats
- **Serviço de Autenticação**
- **Serviço de Gerenciamento de Restaurantes**
- **Serviço de Pedidos**
- **Serviço de Pagamento**
- **Serviço de Entrega**

UBER EATS

- Cada serviço seria desenvolvido, implantado e escalado independentemente.
- Eles se comunicariam entre si por meio de APIs (Application Programming Interfaces) RESTful
- Seria possível utilizar protocolos de comunicação como HTTP/JSON

ARQUITETURA DE SOFTWARE

João Choma Neto
joao.choma@unicesumar.edu.br
https://github.com/JoaoChoma/arquitetura-software
Unicesumar – Maringá