Integração com APIs Remotas: REST e GraphQL

Ensaio teórico

Data: 03/11/2025 — Unidade Curricular: Programação de Dispositivos Móveis — Grupo: 7

Introdução

Este ensaio discute, de forma concisa, duas abordagens amplamente utilizadas para integrar aplicações com serviços externos: REST e GraphQL. O objetivo é dar critérios objetivos de escolha e apontar práticas que aumentam clareza do contrato, desempenho e robustez de integrações em projetos académicos e profissionais.

API: o que é?

Uma API, ou Interface de Programação de Aplicações, é um conjunto de regras, protocolos e ferramentas que permite que diferentes softwares se comuniquem entre si, funcionando como um intermediário para a troca de informações e funcionalidades de forma segura e eficiente.

REST: conceito

REST, ou Representational State Transfer, é um estilo arquitetural para sistemas distribuídos, como a web, que se baseia em um conjunto de princípios e restrições para a comunicação entre cliente e servidor. Cada recurso expõe operações através dos verbos HTTP — tipicamente GET para ler, POST para criar, PUT/PATCH para atualizar e DELETE para remover — e o servidor devolve códigos de estado (200, 201, 404, 500…) e um corpo de resposta, quase sempre em JSON

GraphQL: conceito

GraphQL é uma linguagem de consulta para APIs e um ambiente de execução que permite que os clientes solicitem exatamente os dados de que precisam, sem sobrecarga ou necessidade de múltiplas requisições.

Critérios de escolha

Usamos REST para aplicações mais simples, com operações de criar, ler, atualizar e apagar bem definidas e quando a cache simples é vantajosa.

Já o GraphQL é ideal para aplicações que precisam de dados flexíveis ou que envolvem múltiplos pedidos de dados, como aplicativos móveis, pois permite ao cliente solicitar exatamente o que precisa em um único pedido.

Boas práticas gerais em APIs

- ✓ **Resiliência:** limites de tempo, novas tentativas controladas e pausas temporárias
- ✓ **Erros claros:** quando dá erro é fácil entender o quê e porquê, códigos e mensagens consistentes.
- ✓ **Segurança:** autenticação por tokens, regras de acesso no navegador, validação/filtragem de entradas.
- ✓ **Observabilidade:** registos organizados, identificador por pedido, métricas essenciais.
- ✓ **Paginação:** listas por partes e marcadores, limites e regras claras.
- ✓ **Versionamento:** REST: só criamos nova versão quando há quebras; indicamos a versão e mantemos coexistência com um guia de migração. GraphQL: evitamos novas versões; avisamos e retiramos campos com prazo, mantendo compatibilidade.

Resumo comparativo

Critério	REST	GraphQL
Quantidade de dados	Pode acontecer receber dados a mais ou a menos	Cliente pede apenas o que precisa
Cache	Tira bom partido da cache web	Cache mais usada no lado do cliente
Descrição da API	Descrito através de ferramentas externas	Faz parte do próprio sistema
Erros	Códigos de resposta + mensagem de erro	Dados parciais + lista de erros
Paginação	Páginas simples ou por marcadores	Paginação incremental com indicação de continuação
Riscos	Muitos pontos de acesso	Consultas pesadas se não houver limites
Versionamento	Criação de novas versões quando há mudanças grandes	Evita-se criar versões; campos são descontinuados com aviso

Referências

- https://pt.wikipedia.org/wiki/REST
- https://www.redhat.com/pt-br/topics/api/what-is-graphql
- https://developer.mozilla.org/pt-BR/docs/Web/HTTP/Caching
- https://docs.github.com/pt/graphql/guides/introduction-to-graphql