Aluno: Kauã Sousa de Oliveira

Questionário de Networking em Flutter

1. Qual é a principal diferença entre requisições HTTP normais e WebSockets no Flutter?

- A) Requisições HTTP são para comunicação unidirecional, enquanto WebSockets são para comunicação bidirecional em tempo real.
- B) WebSockets utilizam polling constante, enquanto HTTP não.
- C) Requisições HTTP são mais rápidas que WebSockets para comunicação em tempo real.
- D) WebSockets são usados apenas para enviar dados, enquanto HTTP é para receber.

2. Qual pacote do Flutter é comumente utilizado para comunicação HTTP, sendo suportado em diversas plataformas como Android, iOS, macOS, Windows, Linux e web?

- A) `web_socket_channel`
- B) `dio_client`
- C) `http`
- D) `firebase_database`

3. No contexto de requisições HTTP em Flutter, qual o "verbo" utilizado especificamente para obter dados da internet?

- A) POST
- B) PUT
- C) DELETE
- D) GET

4. Qual é o intervalo de Status Code que geralmente indica uma resposta bem-sucedida de uma requisição HTTP?

- A) 100 a 199
- B) 200 a 299
- C) 300 a 399
- D) 400 a 499

5. Como você adiciona o pacote `http` como uma dependência ao seu projeto Flutter, usando o terminal?

- A) `flutter add http`
- B) `pub get http`

- C) `flutter pub add http`
- D) `import 'package:http/http.dart';`

6. O que é JSON no contexto de comunicação com APIs em Flutter?

- A) Um tipo de banco de dados local.
- B) Um formato leve e amplamente utilizado na internet para troca de dados entre diferentes sistemas, especialmente com APIs.
- C) Uma linguagem de programação para back-end.
- D) Um pacote Flutter para autenticação.

7. No Flutter, para lidar com eventos assíncronos que podem entregar *muitos* eventos ao longo do tempo (como mensagens contínuas de um WebSocket), qual classe é mais adequada?

- A) `Future`
- B) `Sync`
- C) `Stream`
- D) `Callback`

8. Qual widget é utilizado no Flutter para construir a interface do usuário a partir do resultado de um `Future` (uma operação assíncrona que produz um único resultado)?

- A) `StreamBuilder`
- B) `StatefulWidget`
- C) `FutureBuilder`
- D) `StatelessWidget`

9. Qual tipo de exceção de rede em Flutter ocorre quando uma requisição não recebe nenhuma resposta dentro do tempo esperado, possivelmente devido à falta de conexão com a internet ou problemas no sistema operacional ao usar um socket de rede?

- A) `TimeoutException`
- B) `FormatException`
- C) `SocketException`
- D) `NetworkUnavailableException`

10. Por que é crucial lidar adequadamente com exceções de rede em um aplicativo Flutter?

- A) Para tornar o código mais longo e complexo.
- B) Para evitar que o aplicativo se conecte à internet.
- C) Para identificar problemas, permitir que o aplicativo responda de forma eficiente a falhas (por exemplo, falhando graciosamente ou reportando erros) e, assim, aprimorar a experiência do usuário com mensagens claras.
- D) Para permitir o hardcoding de chaves de API.

11. Qual é a forma recomendada de capturar e tratar exceções (incluindo as de rede) em Dart e, consequentemente, em Flutter?

- A) Usando declarações `if-else` aninhadas.
- B) Através de blocos `try-catch`.
- C) Ignorando exceções e deixando o aplicativo falhar.
- D) Reiniciando o aplicativo sempre que um erro ocorre.

12. Em comparação com o pacote HTTP nativo do Flutter, qual é uma das principais vantagens da biblioteca Dio para comunicação com APIs?

- A) Dio é mais simples e intuitivo para operações básicas, mas não oferece recursos avançados como o pacote nativo.
- B) O pacote HTTP nativo tem mais facilidade no uso e configuração manual simplificada.
- C) Dio oferece uma série de funcionalidades avançadas, como Interceptores, FormData, Cancelamento de requisições, Retry, e um gerenciamento de erros mais detalhado, tornando-o a escolha preferida para projetos mais complexos.
- D) Dio não suporta requisições POST.

13. O que são "Interceptores" na biblioteca Dio?

- A) Classes para converter dados JSON em objetos Dart.
- B) Ferramentas que atuam como "agentes" para modificar e validar requisições, respostas e erros HTTP de forma centralizada antes que sejam processados ou enviados, garantindo segurança e conformidade.
- C) Widgets para exibir o progresso de uma requisição de rede.
- D) Métodos para fechar conexões de rede automaticamente.

14. Qual método de um interceptor Dio é chamado *antes* que uma requisição seja enviada, permitindo a modificação das opções da requisição?

- A) `onResponse`
- B) `onError`
- C) `onCancel`
- D) `onRequest`

15. Para cancelar uma requisição HTTP em andamento com a biblioteca Dio, qual funcionalidade é utilizada, passando-a como parâmetro na requisição e depois chamando seu método `cancel()`?

- A) `Dio.stop()`
- B) `CancelToken()`
- C) `dio.abort()`
- D) `request.dispose()`

16. Em uma aplicação Flutter que suporta "offline-first", qual é o papel principal dos repositórios na arquitetura de dados?

- A) Atuar como a única fonte de dados remotos.
- B) Gerenciar a autenticação do usuário.
- C) Ser a **única fonte de verdade**, responsável por apresentar e modificar dados, combinando diferentes fontes de dados locais e remotas.
- D) Exibir a interface do usuário diretamente.

17. Qual abordagem de escrita de dados em aplicativos offline-first permite que os usuários salvem dados localmente *primeiro*, mesmo sem conexão com a internet, e depois tenta enviá-los ao serviço de API, o que pode levar a uma dessincronização se a rede falhar?

- A) Escrita apenas online.
- B) Escrita apenas offline.
- C) Escrita offline-first.
- D) Escrita otimista.

18. Por que o hardcoding de chaves de API diretamente no código-fonte Flutter é considerado uma má prática e altamente desencorajado?

- A) Porque isso torna o aplicativo mais lento.
- B) Porque expõe as chaves de API no seu codebase (código-fonte), o que permite acesso não autorizado a funcionalidades da sua aplicação.
- C) Porque impede o uso de pacotes de terceiros.
- D) Porque é uma prática complexa e difícil de implementar.

19. Qual das seguintes opções é considerada a mais segura para gerenciar chaves de API em um aplicativo Flutter, garantindo que as chaves fiquem totalmente fora do aplicativo?

- A) Armazenar as chaves em um arquivo `.json` local.
- B) Usar o `.env` file com o pacote ENVied.
- C) Armazenar as chaves diretamente no Firebase Realtime Database.
- D) Implementar um backend layer com Firebase Functions e armazenar as chaves em variáveis de ambiente do projeto Firebase.

20. O que representa uma API (Application Programming Interface) no contexto de comunicação entre o front-end (aplicativo Flutter) e o back-end (servidor/banco de dados)?

- A) O servidor de banco de dados diretamente.
- B) Um meio de comunicação e integração, funcionando como um "telefone" que permite ao front-end requisitar e receber dados do back-end sem interagir diretamente com ele.
- C) A interface gráfica do usuário (UI) do aplicativo.
- D) Um sistema de cache de dados local.

Respostas

1. **A**
2. **C**
3. **D**
4. **B**
5. **C**
6. **B**
7. **C**
8. **C**
9. **C**
10. **C**
11. **B**
12. **C**
13. **B**
14. **D**
15. **B**
16. **C**
17. **C**
18. **B**
19. **D**
20. **B**