**ENVIANDO REQUISIÇÕES**

* **Dependências:**
  + **http:**
* **Armazenamento de dados**
  + **Local (dispositivo):**
    - Apenas disponível no app – aquele usuário q usa a app
    - Dados persistente – conseguimos ter um bd dentro da app
    - Trabalha offline
  + **Remoto(web server):**
    - Disponível para mult user
    - Dados persistente
    - Precisa de net
* **Conexão BD – Flutter**
  + **Web server:** a conexão ocorre por meio de um web server, onde manda as requisições e recebe as respostas
  + **App -**> <- **web server -> <- BD**
* **Firebase**
  + **Realtime Database:** cria um api REST.
* **Requisições HTTP**
  + **Comunicação via Rest API**
  + **Convenção REST: Endpoint HTTP (URL) + método HTTP = Ação**
  + **A** conexão da internet é feita por meio de uma pilha de protocolos, API DE PROTOCOLOS TCP/IP (camada de transport) e essa pilha de protocolos consegue fazer comunicação entre computadores. Dentro deste cenário de pilha de protocolos, temos a CAMADA DE APLICACAO tbm com os protocolos de envio de email , transferência de arquivo e o HTTP - o HTTP (protocolos de transferência de hipertexto (html)) é o q está mais proxumo do usuário. Hj em dia o HTTP e HTTPS é o responsável pelo maior numero de trafero da internet
  + **Métodos** 
    - **GET:**  obter dados
    - **POST:** inserir dados
    - **PATCH:** atualizar dados
    - **PUT:** substituir dados
    - **DELETE:** excluir dados
* **Future<T> - .then -.catchError**
  + Tem tudo haver com código Assicronismo. Quadno precisar disso pode chamar o future par aajudar.
  + O método **post** vai retornar um future <respost> , ele vai agir de forma assíncrona. Se quisermos executar algo depois q retornar a reposta temos o **.then**
  + **Async/Await**
    - É uma forma de tratar o código que é assíncrono de uma forma como se vc estivesse programando um código síncrono
    - **Async:** vai possibilitar programar um método assincrono com sincronismo
    - **Await:** vai esperar a resolução do future
    - **Não** precisa do **return** num future<void > em um método que é async pq vai retornar um void automaticamente
* **FutureBuilder:** tbm ajuda na criação de código assíncrono. É focada tbm para trabalhar com requisições assíncronas future
  + **Future**
  + **Builder**
* **JSON:**
  + **jsonEncode()**
  + **jsonDecode()**