Web Sockets

What are web sockets?

Imagine you are watch football game on your mobile device and your favorite team scores a goal. The score board automatically updates on your screen. This happens with the help of web sockets.

Key characteristics of web sockets

1- **Full-duplex communication**
   It allows both client and server to send messages simultaneously.

2- **Low Latency**
   Transmit data quickly for real time updates.

3- **Lightweight**
   Reduce data overhead compared to HTTP.

4- **Persistent connection**
   Keep the connection open(until client or server breaks it) for continuous data flow

How do web sockets work?

1- **Connecting State**
   Client sends an HTTP request to the server if the server supports the web sockets it will response forming an opening handshake.

2- **Open State**
   Once the connection is established both client and server can send messages (these message can be in the form of text of binary format)

3- **Closing State**

The Client or Server can close the connection by sending a closing frame indicating that no further messages will be sent forming a closing handshake.

4- **Closed State**
Indicates the top connection is terminated

Real-life use cases of web-sockets in Mobile apps

1- Chat Applications (deliver instant messages)
2- Live Sports Updates (notify scores instantly)
3- Collaborative Editing (sync changes in real time such as Google Docs)
4- Online Gaming (Enable real time player interaction)
5- Finance Apps (Update market data in real time)

Advantages of Web Sockets

1- Efficiency (reduce data overheads)
2- Real-time Communication (enable instant updates)
3- Scalability (handle large user base)

Challenges with web sockets

1- Network Relaibilty (lose connection during drops)
2- Security (Face attack risks)
3- Server load (strain server resources)

Web Socket support in Flutter	
WebSocket (dart::io)	Enable WebSocket connection with dart built in SDK support
Flutter_websocket	Simplify WebSocket message handling for Flutter Apps