

WebSocket

Workshop

Presented by: Mo El-Gendy & Ayman Mohsen



Today's agenda

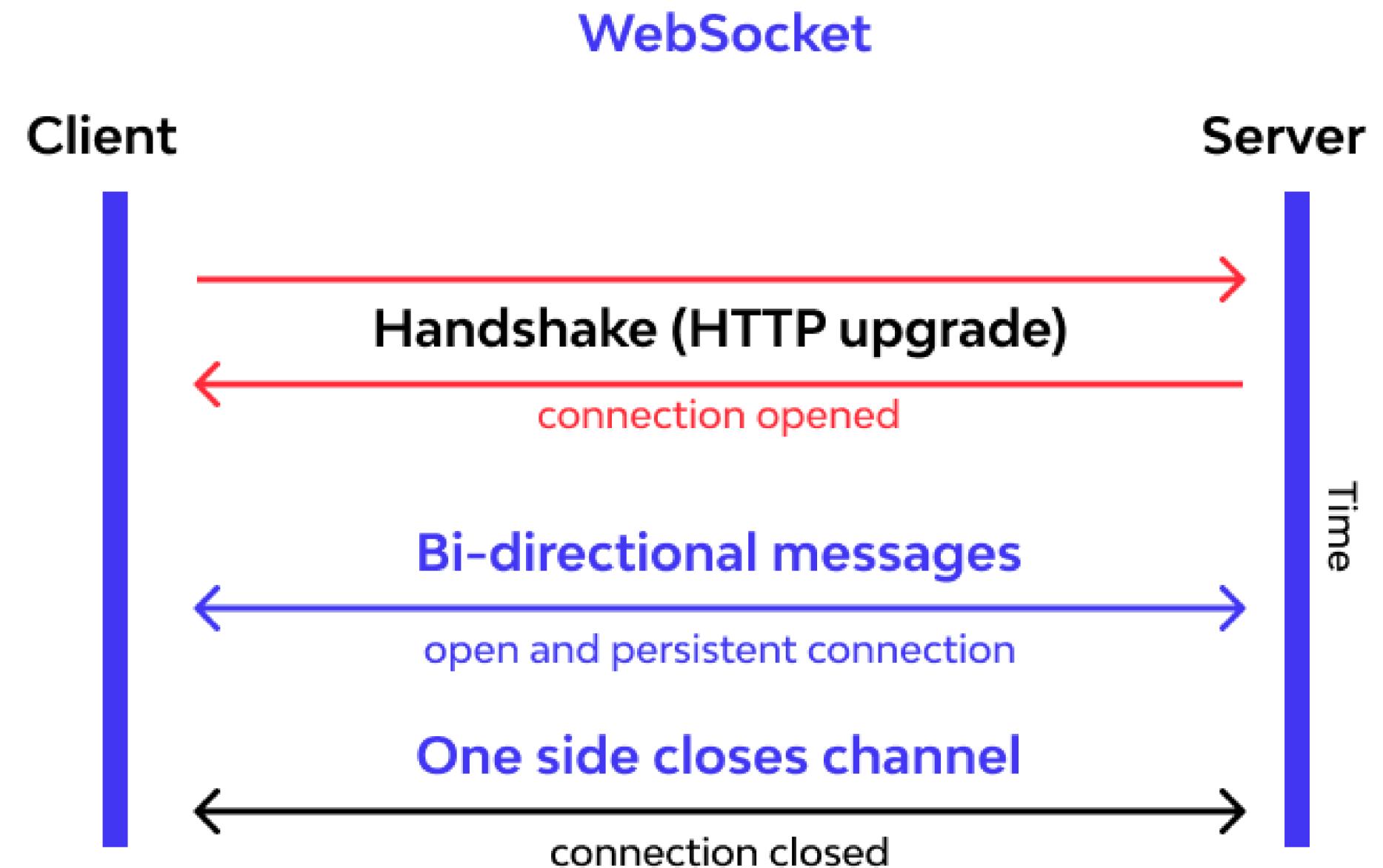
- Introduction
- Real time communication methods
- Real-world Applications
- Project: Chat App
- Set-up the project & Exercises
- Wrap up KAHOOT

(near) realtime time communication methods

- Short Polling (HTTP)
- Long Polling (HTTP)
- Websockets (real time)

Introduction

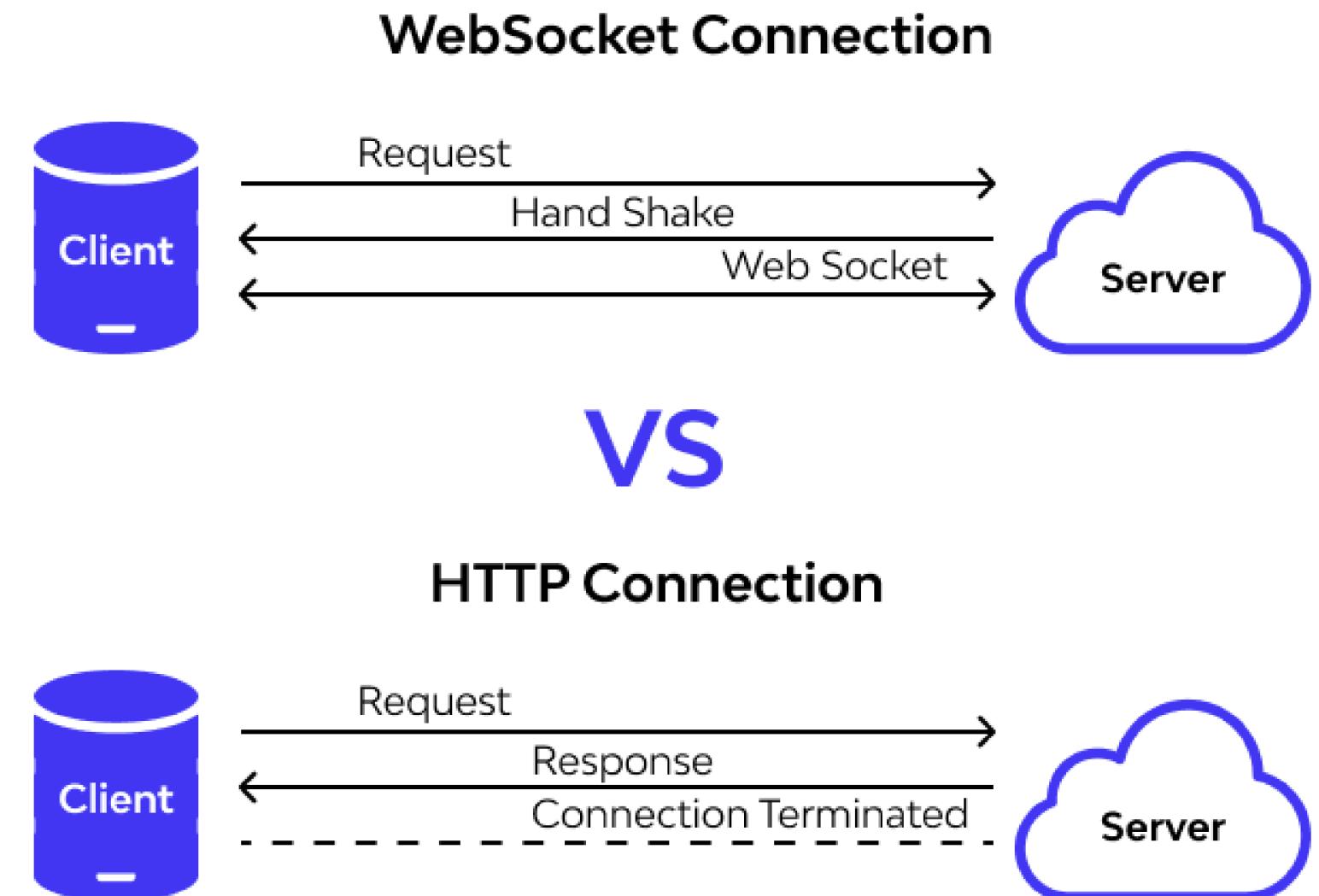
- WebSocket is a **duplex protocol** used mainly in the client-server communication channel. It's **bidirectional** in nature and enables **real-time communications**.
- The **connection remains "live/connected"** until one party **breaks the connection**.



Credits:<https://www.wallarm.com/what/a-simple-explanation-of-what-a-websocket-is>

WebSocket VS HTTP

- HTTP is a **unidirectional protocol** functioning above the TCP protocol.
- HTTP, the connection is built at **one end**, making it a bit more **sluggish** than WebSocket.

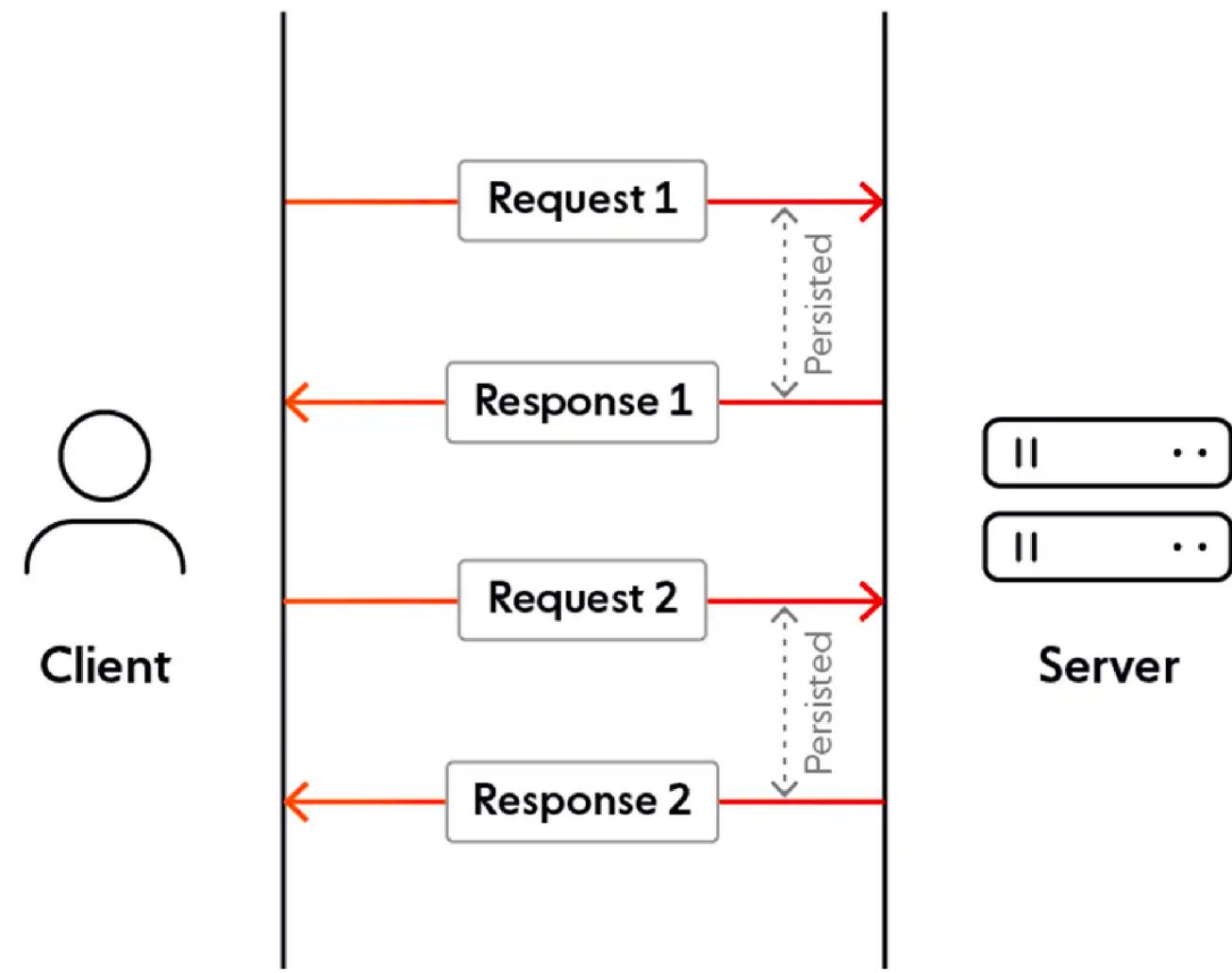


Credits:<https://www.wallarm.com/what/a-simple-explanation-of-what-a-websocket-is>

WebSocket VS Long polling

- HTTP Long Polling is a technique used to push information to a client as soon as possible on the server. As a result, the server does not have to wait for the client to send a request.
- Long polling is more resource intensive on the server than a WebSocket connection. Long polling can come with a latency overhead because it requires several hops between servers and devices.

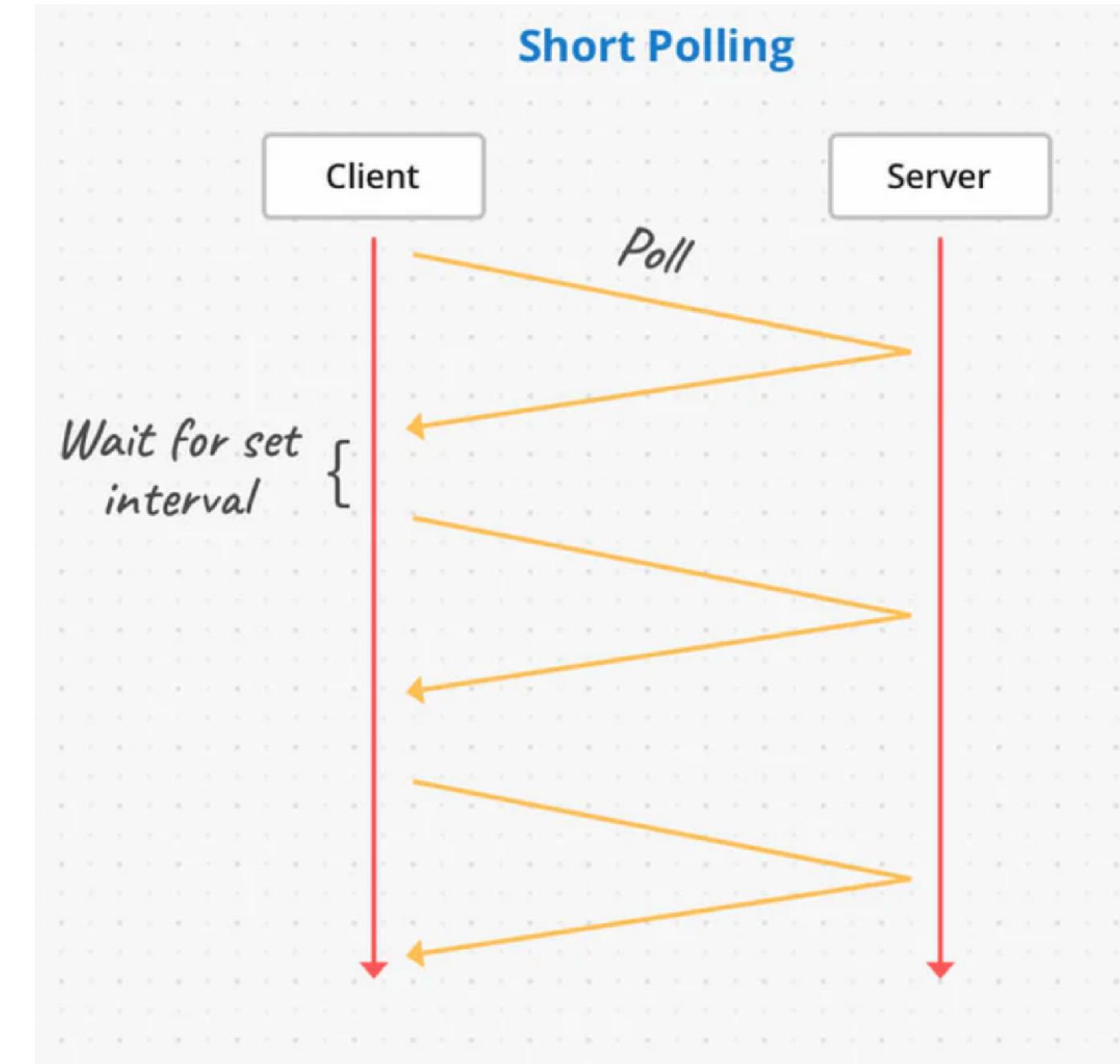
HTTP LONG POLLING



<https://ably.com/blog/websockets-vs-long-polling#:~:text=Long%20polling%20is%20more%20resource,hops%20between%20servers%20and%20device>
s.

WebSocket VS Short polling

- Short Polling is a technique in which the client sends a request to the server asking for data at fixed delays after getting a response from the previously sent request.
- The client sends a request to the Server. The server responds with an empty response or data.



<https://levelup.gitconnected.com/understand-and-implement-long-polling-and-short-polling-in-node-js-94334d2233f3>

WebSocket

Pros

- It allows for **two-way communication**.
- Websockets allow you to **send and receive data much faster than HTTP**.
They're also faster than AJAX.
- Compatibility between platforms
(web, desktop, mobile)

Cons

- Websockets, unlike HTTP, do **not provide intermediary/edge caching**.
- WebSockets **don't automatically recover** when **connections are terminated** – this is something you need to implement yourself

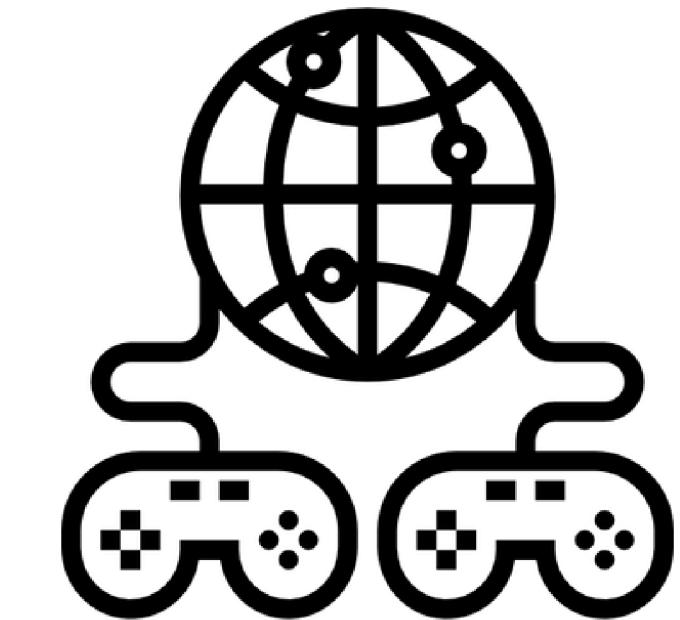
Socket.io

- In nodejs, Socket.IO is a **webSocket library**
- It is an **event-driven library for real-time web applications.**
- It enables **low-latency, bidirectional and event-based communication** between a client and a server.

Real-world Applications



Chat App



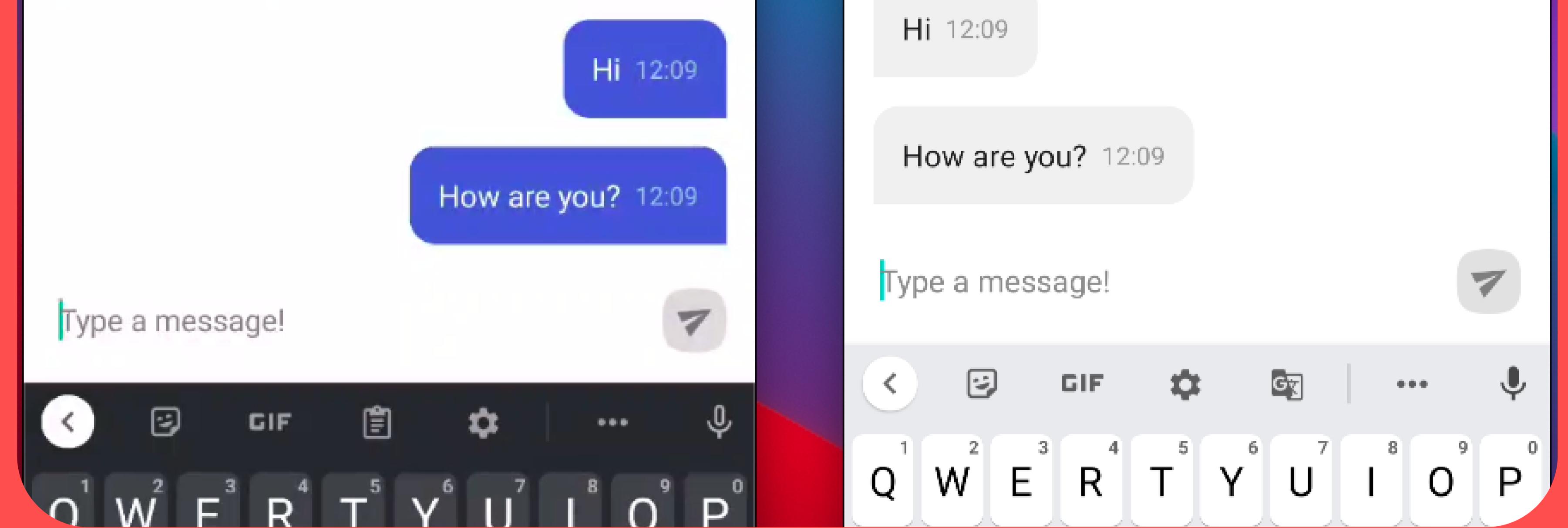
Multiplayer
gaming



Social media
feed

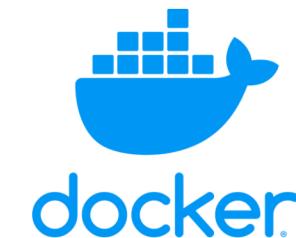


Location
tracker



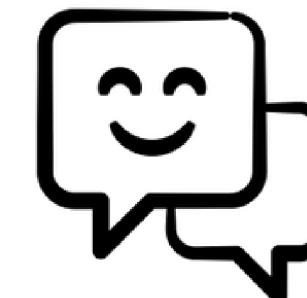
Project chat App

Setup using Docker



1. Open Docker
2. Open VS code or your fav IDE
3. Open terminal and write "docker build -t chatapp ."
4. Then write "docker run -dp 3000:3000 chatapp"
5. Then go to "localhost:3000" in your browser

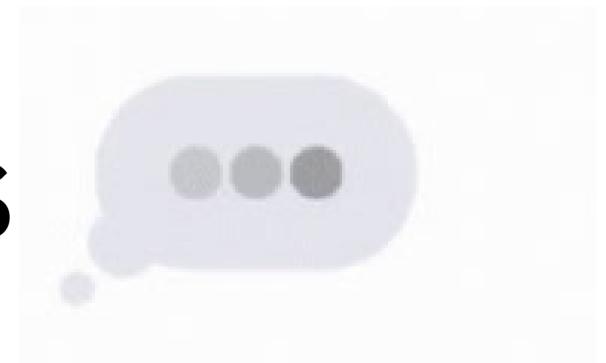
HAPPY CHATTING!



Exercises

Implement using socket.io:

1. Typing Indicator when someone types



2. Notify users when a new user joins the chat



Let's wrap up with Kahoot!

Any
questions!

