WebSocket

Workshop

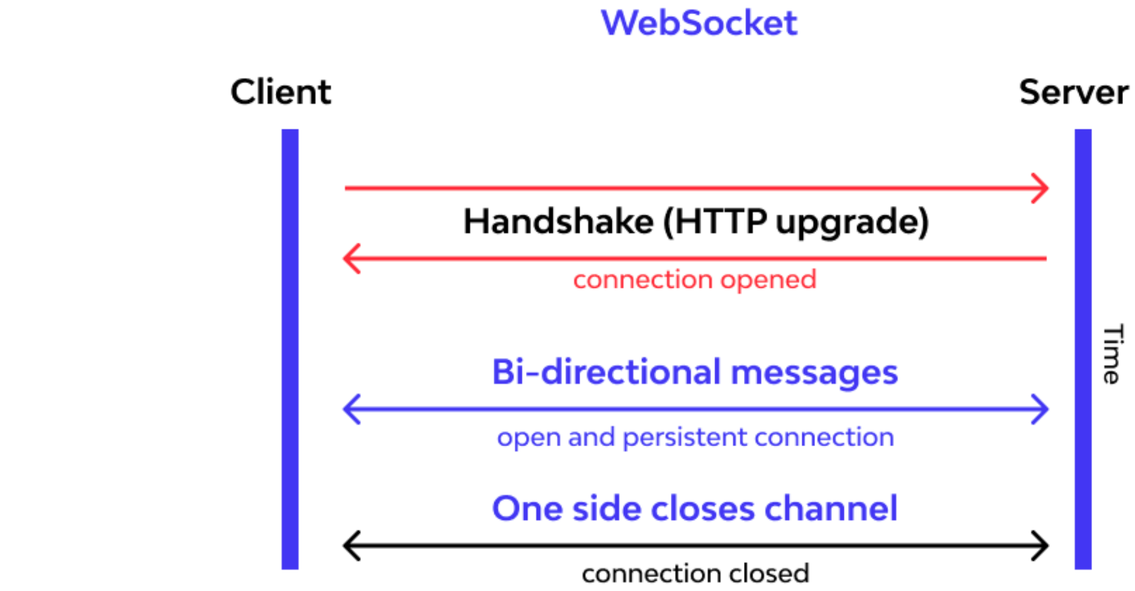
Presented by: Mo El-Gendy & Ayman Mohsen

Today's agenda

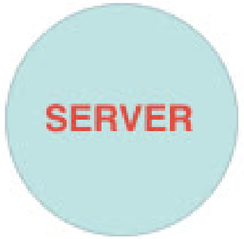
- Introduction
- Real time communicaition methods
- Security issues - and protection
- Real-world Applications
- Project: Chat App
- Set-up the project & Exersices
- Wrap up QUIZIZ
- Q&A

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.**



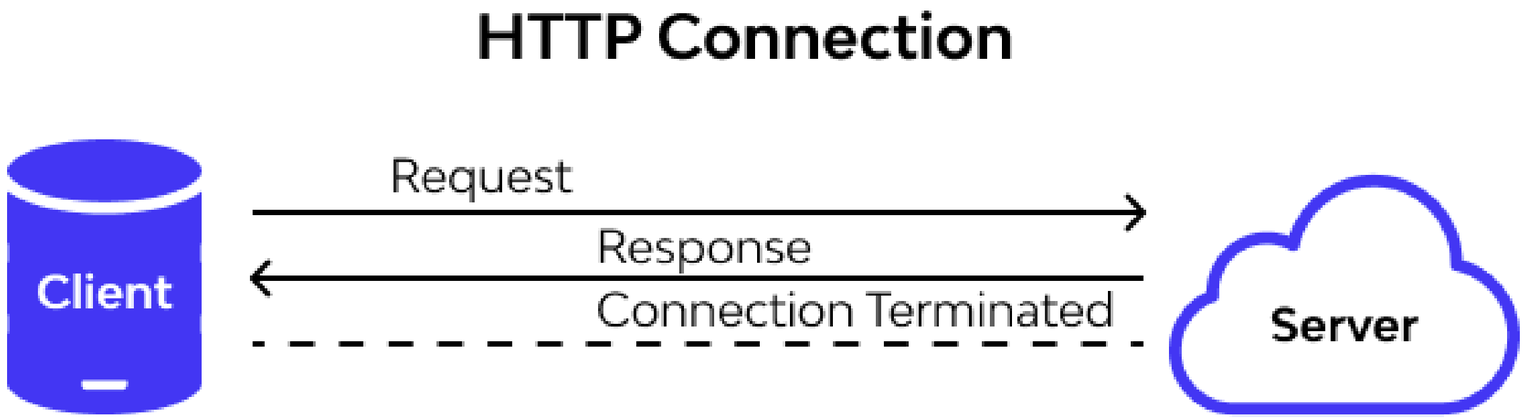
Credits:https://www.wallarm.com/what/a-simple-explanation-of-what-a-websocket-is



Credits:https://stanko.io/do-you-really-need-websockets-343aed40aa9b

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/websocket-vs-http-how-are-these-2-different

Real-time communication methods

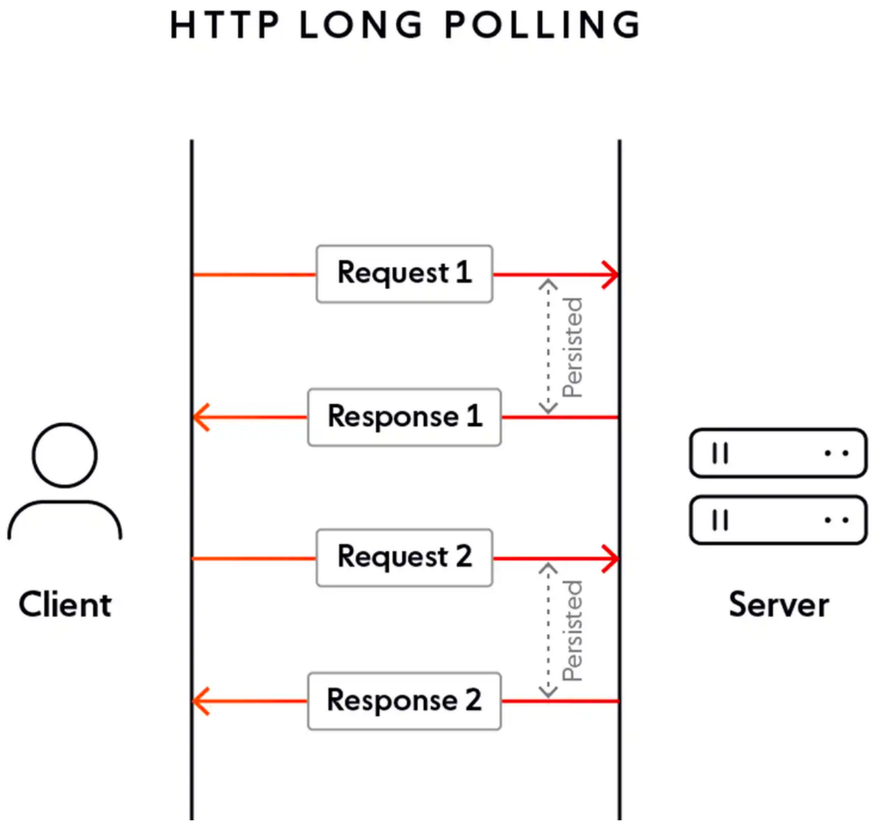
- Websockets

(near) real-time communication methods

- Short Polling (HTTP)
- Long Polling (HTTP)

WebSocket VS Long polling

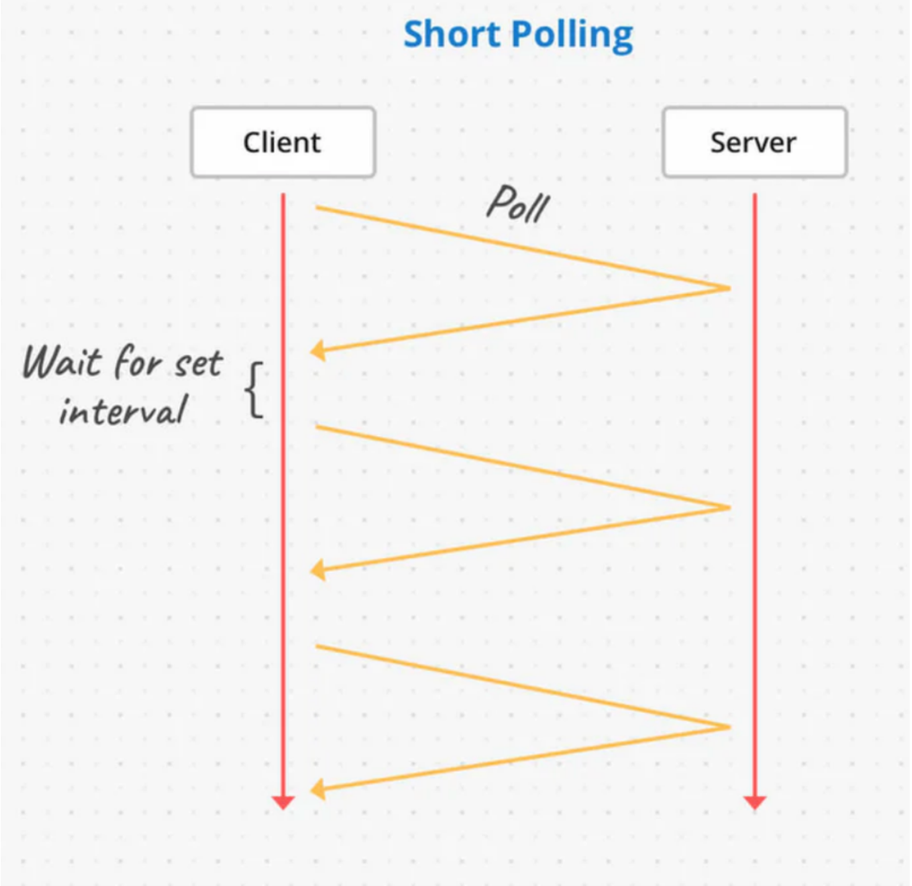
- HTTP Long Polling is a technique used to push information to a client as soon as possible on the server. As a result, the server does not have to wait for the client to send a request.
- Long polling is more resource intensive on the server than a WebSocket connection. Long polling can come with a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



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

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.



Credits: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

WebSocket security issues

- SQL injections
- Manipulating WebSocket messages
- Cross-Site WebSocket hijacking (CSWSH)
- DOS and sniffing attacks

Securing WebSockets

01 Verifying the Origin Header: Only requests coming from trusted origins are to be accepted.

02 Using CSRF protections: CSRF protections such as Anti-CSRF tokens will help in preventing cross-site websocket hijacking attacks

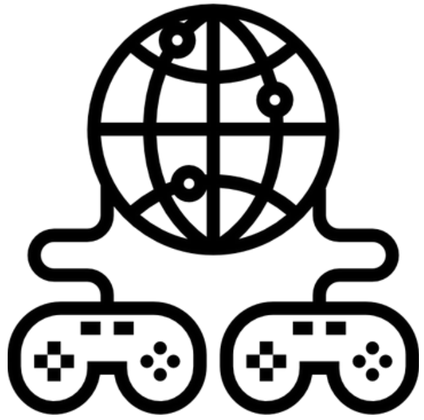
03 Using WebSocket Secure (WSS): Using encrypted communications will prevent anyone from sniffing the communication traffic. It protects against man-in-the-middle attacks

04 Validating client input: Sanitize and validate user input. You never know what input the client will send.

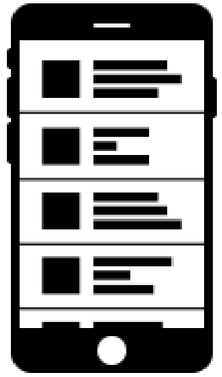
Credits:https://payatu.com/blog/manash.saikia/websocketsecurity

Real-world Applications

Chat App



Multiplayer gaming

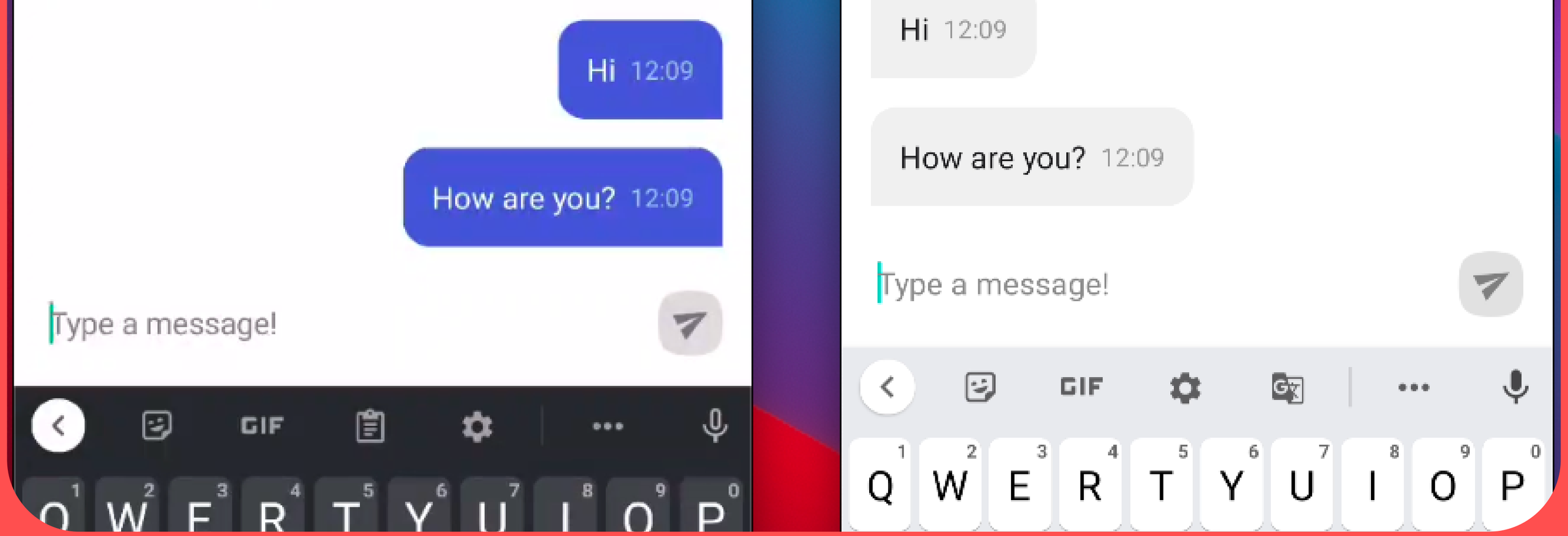


Social media feed

Location tracker

Socket.io

- In JS, 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.



Project chat App

In the chat App

You can:

- Navigate to a room and join it
- Chat with others
- Leave the room

Setup using Docker

1. Open **Docker**
2. Open **VS code** or your favorite **IDE**
3. Pull repo from: **/sebivenlo/esde_2022_websockets**
4. **Cd** to **/socket-server**
5. Run "**docker build . -t chat**"
6. Run "**docker run -dp 3000:3000 chat"**
7. Then go to "**localhost:3000**" in your browser

HAPPY CHATTING!

Exercises

Go To Github -> **README.md**

Let's wrap up with Quizizz

Go to joinmyquiz.com

Any questions!