## Name attributes of the http protocol that makes it difficult to use for real time systems

The http protocol works with request/response sets, as a request is sent to the server it will respond with data via a response. When the transaction is finished the connection is terminated and afterwards the client and the server don’t know anything about each other. This makes it difficult for a client and server to communicate in real time. So in order to get a real time system to work with HTTP you can use polling, long-polling, HTTP streaming or web sockets.

## Explain polling and long-polling strategies, their pros and cons

Polling is when an application repeatedly pools a server for data. The client makes requests and waits for the server to respond. If there is no data available an empty response is returned.

Long pooling is a slight variation of pooling. In long pooling if the server doesn’t have any data available, the server holds the request open until new data is made available. When data is available the server responds and closes the connection, and then the process is repeated.

## What is HTTP streaming, SSE (Server sent events)

Instead of the client requesting data over and over a server can push data whenever it wants when communicating with SSE. That means updates can be streamed from the server to the client as they happen. The SSE connection is unidirectional, which means that the client will listen if something happens on the server, but the server will not listen to the client.

## What is the WebSocket Protocol, and how is it different from HTTP communication, what advantages has it over HTTP

The web socket protocol is an independent TCP-based protocol. The http protocol relies on polling(new connection every time a client request data) to establish connections between a client and a server. This puts a harder load on the server compared to web sockets, who only have to establish the initial connection.

## Explain what the WebSocket Protocol brings to the Web-world

Instead of an unidirectional connection like SSE, web sockets opens up a bidirectional connection, which is useful for games/messaging apps and in other cases where you need near real time updates in both the server and in the client.

## What’s the advantage of using libraries like Socket.io, Sock.JS, WS, over pure WebSocket libraries in the backend and standard APIs on frontend? Which problems do they solve

They simplify the usage of web sockets and more importantly provides failovers to other protocols in the event that web sockets are not supported on the browser or the server.

## Explain and demonstrate the process of WebSocket communication – From connection client to server, through sending messages, to closing connection

Using the Socket.io library:

* Creates a new socket instance and connects it to localhost on port 8080.
* Whenever a new client logs on it will be written in the console.
* When a server sends out data to the clients this will print a message + the data in the console.
* When a client closes his connection this will write that a client has disconnected in the console.
* Sends a message to the server, who should then send it back to all clients connected to the server.

Also see Sockets\_Chat\_Server for example