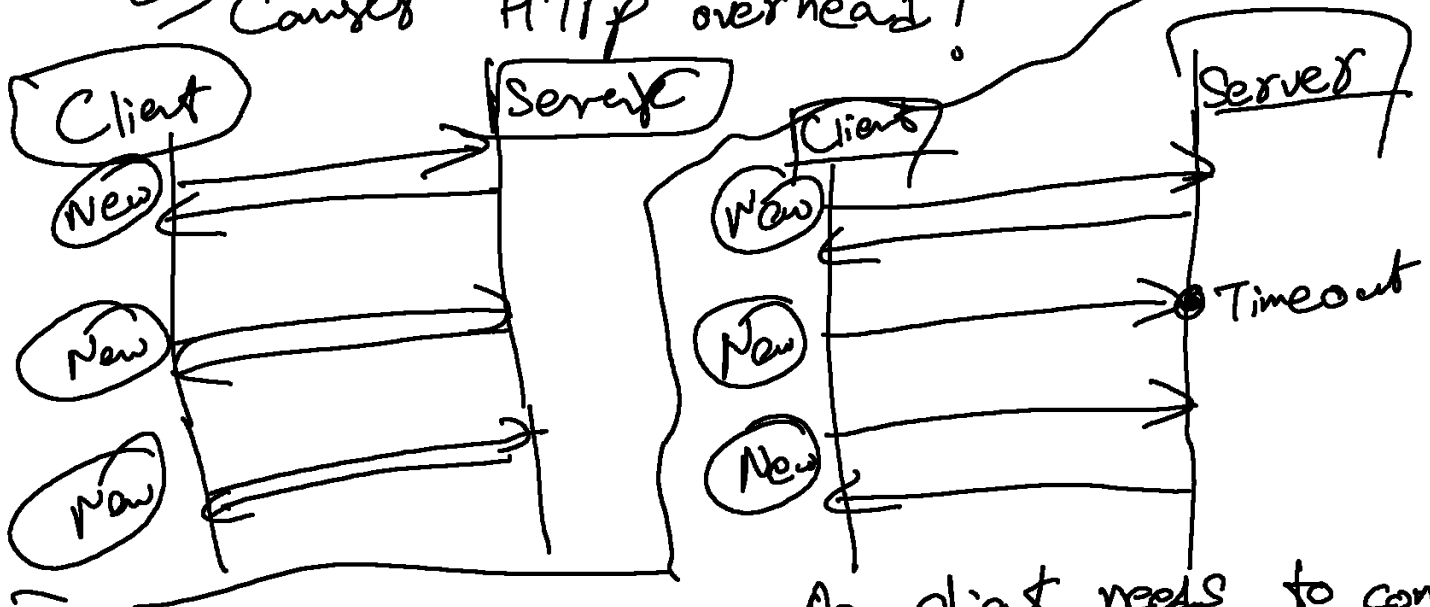- Bi-directional communication in Real-time over TCP connection

Polling

- Makes it so the client keeps sending requests

- Server sends a blank response when nothing to report

- Leads to a lot of requests being made
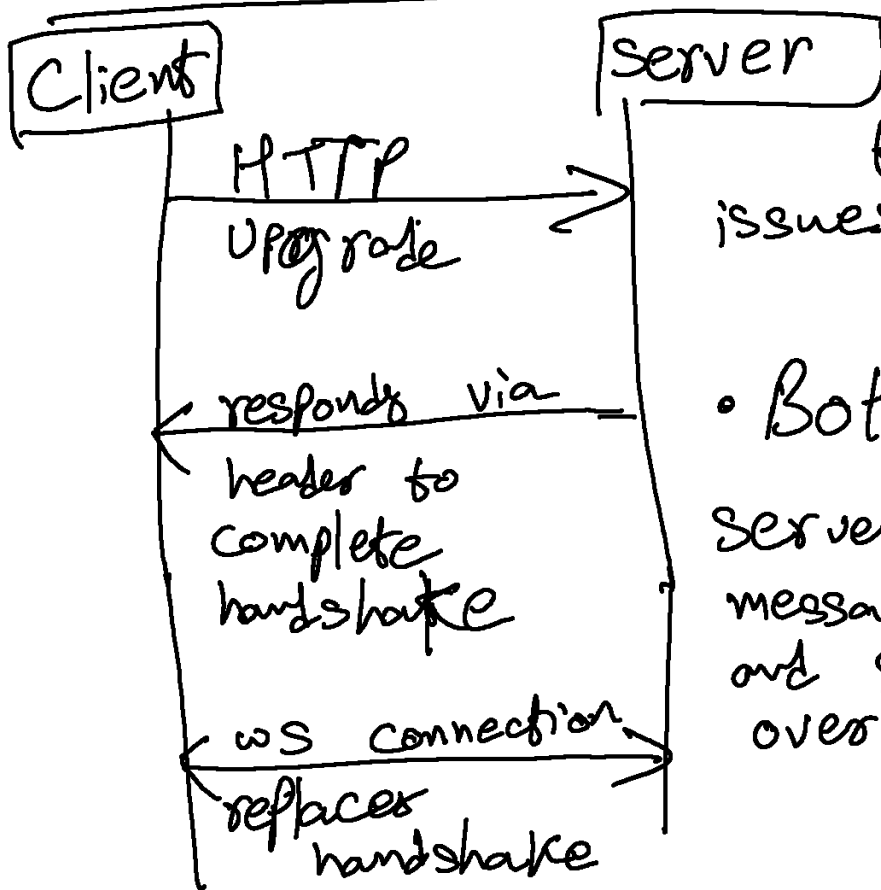 ↳ Causes HTTP overhead!

HTTP long polling

- As client needs to connect periodically, long polling can be more resource-intensive

- Client waits with a hanging GET request to a server and wait until timeout

- If timeout, client will send another hanging GET request

Websockets

Client →(HTTP Upgrade)→ Server

Client ←(responds via)← Server
header to
complete
handshake

Client ←(ws connection)→ Server
replaces
handshake

- Unique connection to reduce latency issues from long-polling

- Both client and server can stream messages independently and simultaneously over TCP

- A ws handler keeps an open connection with active users

- Server responds with HTTP status '101 switching protocols' with a bashed string Sec-WebSocket-Accept of client ws-key
  ↳ This helps ensure secure/authenticated key

- Masking helps distinguish websocket data from HTTP requests
  ↳ Helps stop caching from proxies

- Fragmentation helps prevents exceeding of buffer limitations and stops overwhelming connections from large messages
  ↳ prevents buffer overflow
  ↳ FIN (Final) bit is set to '0' for all but last frames of the message. Set to '1' for final frame to signal final fragment of message to reciever.