Serverside

Table of Contents

Client Server Approach

- there is the client, which renders the application, and the server that, has the logic within the application
  - there is good insulation but also more complexity within these types of approaches

Disadvantages of Client Server Approach

- With the client-server approach, there are performance issues regarding throughput and latency
  - throughput means how much data we can get through the network or the server (bits/second)
  - latency is the delay per seconds
  - how long will the request take to fulfill (send request, process request, send the result back from the server, render result)
  - the latency is impacted by factors out of the control of the developer, such as networking connection over long distances and basic networking requirements that vary by clients

Fixing Throughput

- out of order execution can speed up the processing by allowing the server to do the small task while doing the fast tasks out of order
  - downside means the execution might cause logic errors

Fixing Latency

- caching allows for not having to process the request because the answer is already there
  - might have stale caches causing incorrect data within the server

Peer-to-Peer Approach

- the client is split across many different servers, and all of the peers are connected and communicate with each other
  - when one peer disconnects, the rest of the peers can communicate and survive

Primary-Secondary Approach

- the primary part is in charge of the application, which looks over all of the secondary workers
  - the primary nodes are in charge of the request and know where the state is, but the worker bees are in charge of the details of the stuff