Q 1.

Latency is the measure in milliseconds of the time it takes for the command to be sent and the data request to be serviced and returened to the user, or the Round Trip Time as it is more properly known. Latency should always be minimised, but although data moves around at the speed of light on fibre optic enabled networks, the distance between the user and the server, along with delays introduced by internet infrastructure equipment, mean latency can never be eliminated completely. There are also propagation delays due to speed of communications link and speed of light delays. Queuing and processing at routers and on source and destination nodes. Transcodingh delays due to data manipulation.

Q 2.

The latency of different players can affect how the result of a gunfight can appear on different players screens. If player 1 has latency and can see player 2 standing in front of them they may try to shoot at the player. However, on players 2 screen they may have already moved around a corner so that they are no longer in the line of sight of player 1 so from their perspective they should not be shot. The result of if player 2 is actually shot or not depends on when each of the players information is sent to the server. If player 2's position is not sent to the server in time they may still be in the position that player 1 can see them in when player 1 shoots meaning that they will be shot according to the server as their movement has not been fully sent over yet. There is also a chance that player 1 shooting may take extra time to send over to the server instead of player 2's position information depending on their internet speed as well as how close to the server they are which will instead result in player 1 thinking that they shot player 2 but in reality the server will have registered player 2 moving first and will take extra time to see that player 1 tried to shoot.

Q.3

Dumb client: conservative

Client-side prediction: optimistic

Dumb client: Client sends input to the server, the server calculates the

sate and sends it to the client, client just renders the state to it

from the server

Client-side prediction: sends out inputs to the server, also applies

them immediately locally to update your own state. server can override

player input if simulation on server side has a constraint player does not

know about yet. when you recieve the update from the server, check if it

differs from local state. if different, correct local state.