Lab 7

1. Latency cannot be eliminated completely because it is an inherent physical constraint of data transmission.
   Reasons:
   1. Data cannot travel faster than light even with fiber optics, communication over long distances makes delay.
   2. Data must pass through multiple routers creating delay per hop
   3. Packet handling + CPU/GPU rendering also produce delays
   4. Wireless signals can also experience interference, delaying the data

2. For a shooting around corners problem, say Player A is behind a wall and player B shoots at them, because of the network latency, the game has to balance consistency so both players feel justified, making the game more responsive.

Example without latency compensation:
Player A moves behind a wall at T=0ms
Player B fires at T=50ms
Due to 100ms network latency, Player A's position is outdated for player B
Player B still sees Player A even though he is behind a wall and registers a hit
Server may decide that the hit is registered even is A is behind a wall

Both players are seeing different things so higher latency leads to more inconsistency.

3. There are 2 categories of consistency management:
Optimistic (Client Side Prediction)
Conservative (Dumb Client)

Dumb Client - Conservative

-The client has no authority and only renders what the server tells it.

-The server is the single source of truth, preventing inconsistencies.

Example: An MMORPG where movement and attacks are only executed after confirmation from the server.

Pros: Guarantees consistency.

Cons: Introduces high input lag, making the game feel unresponsive.

Client-Side Prediction - Optimistic

-The client predicts the outcome of actions before receiving confirmation from the server.

-When the server response arrives, the client corrects itself.

-Example: In an FPS game, when you press "W" to move forward, your character moves instantly (prediction), even though the server confirmation comes later.

-Pros: Reduces perceived latency, making gameplay smoother.

-Cons: If predictions are wrong, the client must "snap" back to the correct position, causing visual glitches (rubber-banding).