https://excalidraw.com/#json=X3ZR37EvG9niNJMTMBygB,N_EZGHnS485f94qQO2aHRA

https://www.youtube.com/watch?v=YcKf7BT8v8Y&t=495s&ab_channel=Jordanhasnolife
https://www.youtube.com/watch?v=K3Z1PY2vr3Q&ab_channel=TechDummiesNarendraL

Functional Requirement

Online battle royale game like pubg

Non Functional Requirement

Low latency
Reliable
Fault tolerant

Capacity Estimates
1000 players in a lobby
10,000 statelful objects per game

API Design

-findGame(userid)
-doAction(userId, Action action)

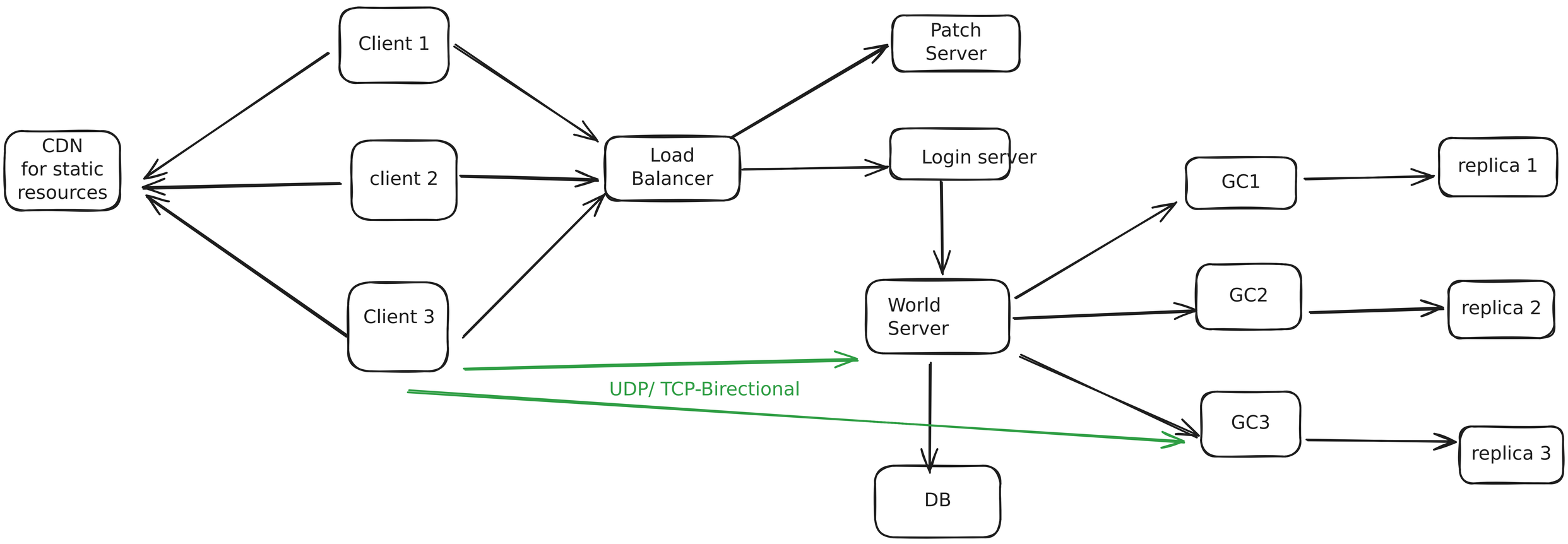
Database Tables
playerLocations
playerItems
playerHealth
ObjectLocations
ObjectsHealth

Architecture

1. Why not use peer-to-peer communication?
Problems being -> cheating. If we are communication directly, we will not know if the users are cheating.
Central server can verify and validate if the move is valid.

2. We will be storing the game state in-memory to make sure as low latency as possible. Persistence needs to be taken care.
3. UDP can be used as underling transport protocol to further reduce the latency. Bidirectional communication will not be supported or we need to custom implement it. Because it is also not reliable and prone to packet loss we can do one optimization is to send P1 and then P1,P2 and then P1,P2,P3. In case P1 lost, because next set of packet is P1,P2 it'll work.
4.Do the serlization of data from client e.g proto or avro to further reduce the latency due to network.
5. Because the ordering of the events matter over here in case of every action, we also need to send the timestamp.

HLD



Login Server
Authenticate the valid user and sent a login confirmation to client.

Patch Server.
Before the login, connect to patch server to make sure the client has the updated version. If not, sent the info to client.
Also login will not happen before the client has updated (just like in pubg).

World server
Login server will sent the request to world server. Responsibilites will include

1. Matches the player to the game based on the region he is closest at.
2. Stores the current states of the player including health , ammuntion , where the player is etc. for a particular game session.
3. Also manages the game server like which map is loaded on which game server etc.
4. World server will return the following information to client (let's say c1). game server IP's and port for which the current player belongs as well as the nearest game server along with all the information about the player.
Now the client will connect to the respective game server to know about the state of the objects and render the information.
5.Will also manages the scaling of game server.

Game server
1. Why does a particular map is dividied into submap's ? In the game where lot of objects and player is involve
handling all the information on a single server will become difficult and hence the map is divided into submap's.
2. We can do this intelligently ? The world server can associate 2 different game map to same machine intelligently.
e.g low activity of one game + high activity of another game on single machine.
3. There are some cases where the client or player is trying to cross from one region to another ? How to handle
this scenario.
The region for which the user belongs to needs to broadcast the information to adjacent region's that , you need to broadcast any update in the object state to this player. Also same goes with the above region. This can happen till the point of intrest near the border are overlapping with the user information. In case the user crosses the border, send the broadcast to world server also to update the information.