




# The Ultimate Online Gaming Experiences with CockroachDB





In today's ever-evolving digital age, gaming as a sector has emerged as a powerful medium that enables a global yet connected experience. One of the key pillars of this experience is gametech that fosters innovation and transformation centered around giving gamers a near real-time experience no matter where they are playing from.

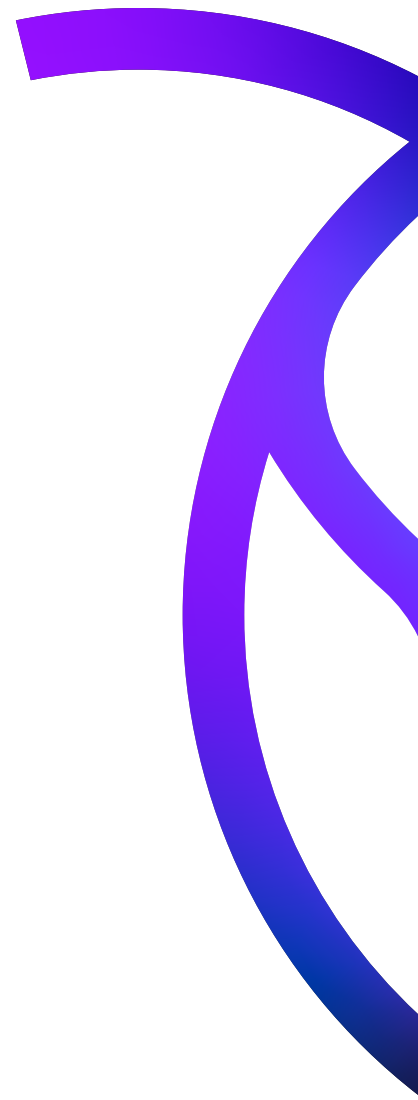
The gametech sector is a large and rapidly growing market that encompasses the development, production, and distribution of video games, as well as the technology and infrastructure that supports the gaming industry.

With mainstream acceptance of e-gaming to increase in engagement driven by global demand for online gaming, the gaming industry is expected to [grow at 12.9% YOY to \\$584B by 2030](#).

Gametech organizations are supporting this growth. As the backbone of this industry, they're challenged with improving their systems and providing an experience that can scale and is reliable to all their users. While there are various technical challenges to solve, one of the most pressing is databases. The database supports gaming servers that enable the multiplayer online experience and can often be a bottleneck to scale, reliability, and performance amongst other things.

It's thus paramount that organizations choose a new type of database that can support these modern gametech requirements for a global gaming community: a distributed SQL database design that unlocks performance and provides scale, reliability and resilience for real-time gaming.

Main challenges for gametech today	4
Architectures for multiplayer online games	8
Designing a real-time multiplayer game architecture	11
<i>"The Cockroach Lab"</i> game example	12
Use cases of CockroachDB in gametech	13
CockroachDB case studies	13
Ready, set, match	14



# Main challenges for gametech today

## **Challenge:** High game latency = Gamers logging off

Gamers thrive on lightning-fast response time. In a competitive gaming landscape, even a slight delay could mean the difference between victory and defeat. Low latency is the holy grail for gamers, and they expect a smooth, uninterrupted experience no matter where they are on the planet. For developers of such games, delivering a low-latency experience to a global gaming audience is critical and among the very top challenges.

With the onset of multiplayer global online games, that includes a globally distributed gaming community. Managing and synchronizing data while providing a low-latency experience can be a Herculean task, fraught with complexity for a game with gamers spanning multiple regions.

The solution, though, is pretty straightforward: locate gaming services closer to the gamers' geographical locations.

## **Solution:** A distributed global database with built-in geo-location

One of the best ways to minimize latency is to minimize the distance data has to travel over the network, which means data services for the game need to be located as close as possible to a gamer's region. Let's say you have a gamer in New York, USA, who wants to purchase an upgrade to their character in a game to play their next online match. You wouldn't want them to make the purchase from a service running in Europe, which would add to a delayed experience. The goal is to provide a service that runs within the US Region — perhaps us-east-1 if you use AWS — to minimize latency during both in-game purchases and in the game itself.

With CockroachDB, game data can be intelligently distributed across multiple regions and placed as close as possible to the players. This ensures lightning-fast response times and a seamless gaming experience, no matter where the players are located. Its globally distributed architecture is specially designed to enable use cases centered around geo-location.

CockroachDB also partitions data into smaller chunks and replicates data across its globally distributed architecture. This optimizes data retrieval and ensures that players enjoy snappy, responsive gameplay without delays or hiccups even when a service in a particular region may not be available for some unforeseen reason. In the world of gaming, every millisecond counts, and CockroachDB makes sure you stay ahead of the pack.

## **Challenge: A gamer streamed your game and it just went viral**

The world of gaming is as unpredictable as it gets. Games can go mainstream for a variety of reasons, from global e-gaming tournaments to online gamers streaming your game on Twitch or YouTube. One moment, you're handling a steady stream of players — and the next, your game goes viral. Now you're swamped with a massive influx of gamers.

Not only do you have to accommodate the sudden surge in demand, but you also need to ensure that the performance remains top-notch and that players aren't left hanging in the virtual world. It's not just about handling spikes in user demand; you've got to be prepared for the long haul.

While game developers who design the games intend for their games to have as many players as possible, not considering elastic scale as a part of deployment strategy can be an unnecessary blocker. On the flip side, as the player base for an older game dwindles, back-end architecture and data services must gracefully scale down to adapt to these changes.

Such a dynamic landscape demands elastic scalability to easily expand and contract.

## **Solution: Scale is the name of the game**

CockroachDB is designed to tackle the unpredictable gaming world head-on. With CockroachDB, scaling your game is a breeze. Its distributed architecture allows for smooth horizontal scaling, making it easy to add or remove nodes as needed to accommodate your game's ever-changing requirements.

This flexibility ensures that your game can handle both sudden surges in user demand and the gradual decline of older games with ease. CockroachDB is also designed to work on any cloud platform of your choice while also being compatible with Kubernetes to easily automate and deliver an elastic scale.

## **Challenge: Achieving lightning-fast global consistency**

Gamers live for the thrill of unlocking and purchasing in-game items. They crave instant gratification, and the last thing they want is to wait around for their shiny new avatar, token, or weapon to appear. Such transactions enable a range of experiences, from styling an avatar to getting an advantage in the next battle. And it's critical for a back-end system to be able to manage these millions of transactions while ensuring that the data is correct and consistent.

Not only is it important for the data to be consistent, it's also expected to be immediately available. Every second counts in the world of gaming, and delays can lead to dissatisfaction and even rage-quitting.

Game developers have to account for this when designing and deploying their games. They have to first ensure that data is available across multiple regions immediately while also ensuring that this data is consistent so that gamers get a single, yet powerful experience.

## **Solution: Consistent multi-region, active-active architecture**

The ability to guarantee highly performant, correct transactions is difficult. Doing this at scale is another challenge. Such a capability is critical in solving the modern challenges in gametech.

CockroachDB is an ACID-compliant database, which means it adheres to the principles of atomicity, consistency, isolation, and durability. It provides serializable isolation, which ensures that data remains consistent throughout the database — even in distributed environments — and that transactions are executed in a way that produces the same result as if they were executed sequentially. This makes it the perfect ally for gaming developers who want to create a consistent experience for gamers. With CockroachDB, the moment gamers unlock or purchase in-game items, they can instantly access them, regardless of their location.

The other benefit to developers is that cockroachDB simplifies the developer's life by eliminating the need for complex, location-aware code, since all of this can be handled within the database layer and not the application layer.

The ability to automatically replicate data and rebalance while guaranteeing that data is always consistent, available, and accessible across different regions — even in the face of network failures or other unexpected issues — only makes the solution more robust.

## **Challenge: Tackling unplanned outages and data loss like a pro**

Gamers are a passionate bunch, and any hiccup in their gaming experience could unleash their fury. Unplanned outages? You better brace yourself for an inbox flooded with angry messages/tweets/posts on social media! And heaven forbid any data loss occurs during those outages. Unlocking an in-game item just to find it lost forever due to a service interruption is every gamer's worst nightmare. In the gaming world, a highly available back end with zero data loss is not just a desire — it's an absolute necessity.

Modern gaming is a complex ecosystem, with multiple services working together to create an immersive experience. As the stakes get higher, a single outage could spell disaster, impacting not just one game but the entire portfolio. Designing a system that's resilient, always available, and keeps data safe is the ultimate challenge in the gaming realm.

## **Solution: Outage or downtime = Game over**

In today's fast-paced gaming landscape, if your multiplayer game is lagging or stuck with the infamous "loading" icon, gamers are just a click or press away from finding another game to play — or worse, taking to social media to vent their frustrations. Building a highly available online gaming platform means eliminating the possibility of downtime, whether planned or unplanned. Distributing data across multiple regions increases fault tolerance, ensuring that your gaming services never go offline. CockroachDB's resilient multi-active architecture enables your database to survive node, availability zone, and even entire cloud region failures, keeping gamers immersed in your virtual world.

The next time a critical server goes down or an unexpected power outage strikes a data center, your game can still be up and running — while your competitors scramble to recover. Unforeseen downtime aside, planned downtime is also a thing of the past. CockroachDB's live schema changes mean that you can update your database in production without disrupting the gaming experience.

Let's face it: gamers flock to online multiplayer games for the thrill of competition and camaraderie. If they can't play because the game is slow — or, worse, not working at all — they'll quickly grow frustrated and abandon the game. They might never return, costing you both players and revenue. In the world of online gaming, reliability is key.

By leveraging CockroachDB's resilient architecture, you can ensure that gamers enjoy a seamless, uninterrupted experience, fostering loyalty and positive word of mouth. Keep your players engaged and coming back for more, while your competitors struggle to catch up.

# Architectures for multiplayer online games

## Key infrastructure considerations

Running a multiplayer online game requires a combination of technologies and architecture that can handle a large number of players and game assets, while providing a smooth and seamless gameplay experience. These are the key technologies required for running a multiplayer online game:



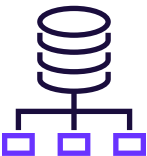
**Game server:** The game server encompasses the hardware and software components necessary for running the servers that manage game logic, game state, player data, and matchmaking. These servers can run on cloud provider infrastructure such as AWS, Google, or Azure, and need to be optimized for performance, ensuring that they can handle high volumes of concurrent players while providing a seamless, lag-free experience.



**Database:** A robust database is crucial for storing essential game data, such as player profiles, game state, leaderboards, and in-game assets. This information must be readily accessible and easily managed, enabling developers to make updates and improvements as needed. Additionally, the database should be highly scalable to accommodate the growth of the player base and the game itself.



**Networking:** Networking plays a pivotal role in the success of multiplayer online games, as it facilitates communication between players and game servers. This involves implementing efficient protocols for data transfer, mitigating packet loss, and reducing latency. High-quality networking ensures a smooth, enjoyable gaming experience for players, fostering a strong sense of immersion and connection with their online counterparts.



**Load balancer:** Load balancers are essential components for managing traffic across multiple game servers, ensuring scalability and availability. By evenly distributing player connections and requests, load balancers help maintain optimal server performance, prevent bottlenecks, and minimize the risk of downtime. This results in a more stable, responsive gaming environment for players, enhancing their overall experience.



Now let's talk about the architecture of a typical multiplayer online game.

**1. Client-server architecture:** This architecture involves clients (players) connecting to a central server that manages game logic, game states, and communication between players. The server can either be authoritative or nonauthoritative. An authoritative server manages all game states, whereas a nonauthoritative server delegates game states to clients.

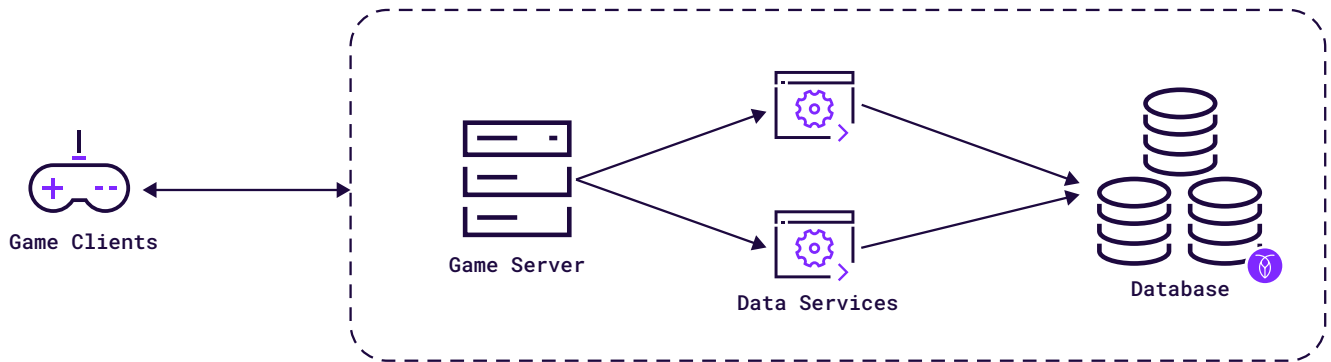


Figure 1. Client-server architecture

**2. Peer-to-peer architecture:** This architecture involves players connecting to each other directly without the need for a central server. However, this architecture is not commonly used for online games as it is difficult to implement and prone to cheating.

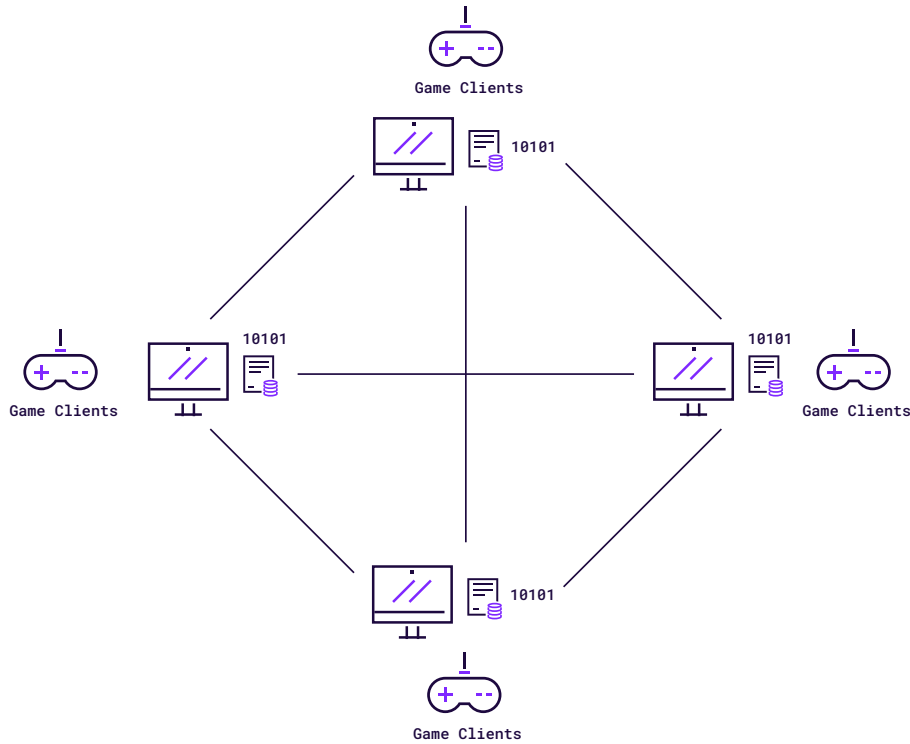


Figure 2. Peer-to-peer architecture

**3. Distributed server architecture:** This architecture involves multiple game servers that manage different parts of the game world, such as different regions or game modes. A load balancer distributes traffic across these servers.

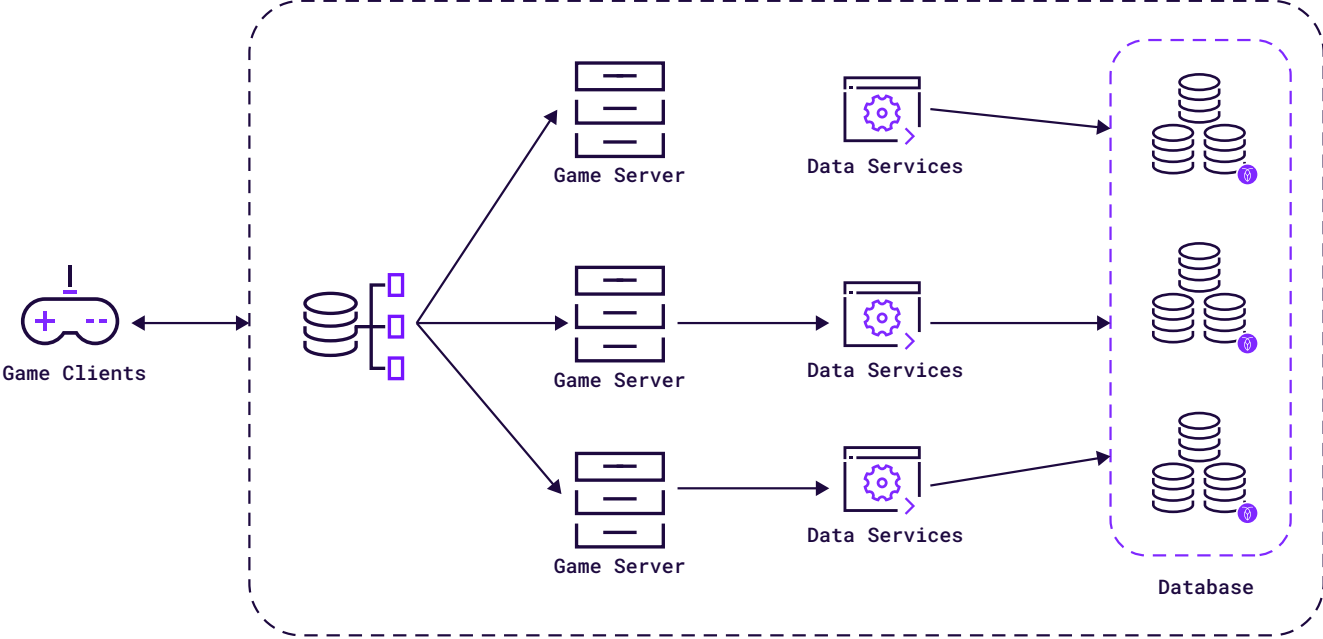


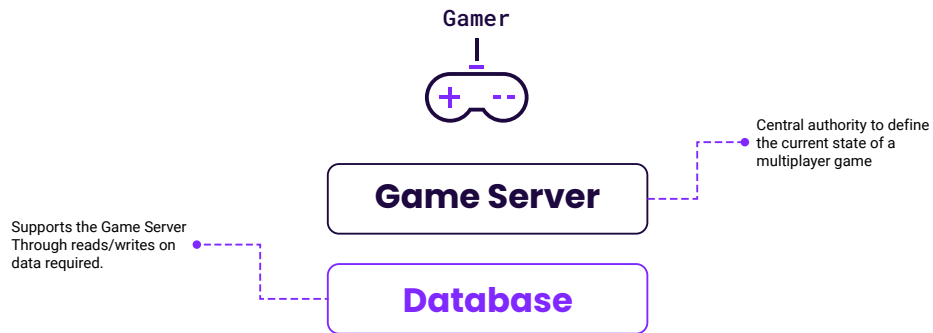
Figure 3. Distributed server architecture

For today’s gametech initiatives, requirements, and industry challenges that we’ve highlighted in this paper, distributed architecture with an equally scalable, consistent, and distributed database is an acceptable solution.

# Designing a real-time multiplayer game architecture

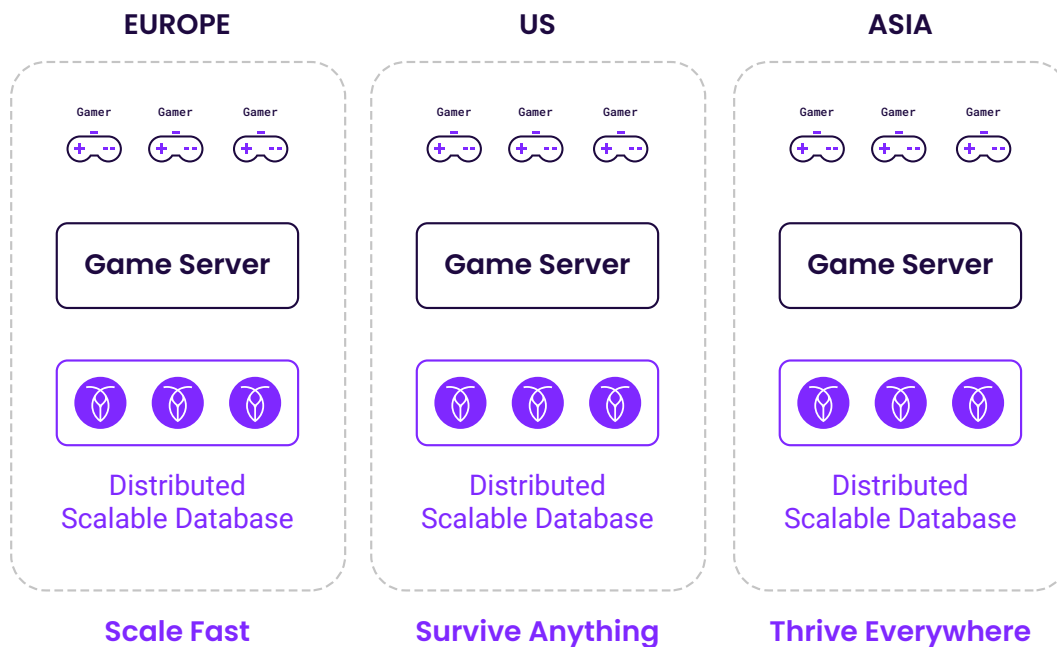
As mentioned above, designing and architecting an online multiplayer gaming experience and its supporting infrastructure and environment can be a demanding and intricate endeavor. This is particularly true when aiming to incorporate numerous essential attributes such as resilience, low-latency performance, consistency, and scalability.

While it's important to use technologies to develop the right game with the right physics to give gamers the experience, we also need the infrastructure backbone to support the particular game. This includes selecting a scalable back-end game server and an equally capable and scalable distributed database to meet the requirements of gametech.



## CockroachDB as the backbone of the game server

CockroachDB is a distributed SQL database that provides ACID (atomicity, consistency, isolation, and durability) transactions and horizontal scalability. It can be easily deployed on-premises or hybrid cloud (on-premises + private/public cloud) or multi-cloud on providers such as AWS, GCP, and Azure across multiple regions. CockroachDB uses a distributed architecture that allows it to be highly available and resilient to failures.



## “The Cockroach Lab” game example



Figure 4. Gameplay from The Cockroach Lab game

Our team designed a multiplayer game called “The Cockroach Lab” that reflects and demonstrates how CockroachDB would really work in gametech and address the challenges we discussed above. The game focuses on Craig, the cockroach protagonist competing for supremacy against other global players in the Cockroach Lab, built on scalable servers.

We are utilizing Heroic Labs Nakama, a versatile, open source server specifically designed for the development of real-time multiplayer games, social gaming experiences, and various other applications. This powerful tool empowers us to harness features like multiplayer matchmaking, player profiles, real-time chat, leaderboards, and much more.

By integrating Nakama with CockroachDB, we can use CockroachDB as the primary database for preserving essential game data such as player profiles, game states, matchmaking, and leaderboards. CockroachDB’s advantages – such as its geo-distributed capabilities, ACID compliance, high availability, and seamless scalability – provide a rock-solid foundation for this gaming infrastructure. This ensures that the gaming experience remains consistent and fast, regardless of the player’s location.

Nakama can function as an additional layer on top of CockroachDB, delivering real-time features like instantaneous chat, notifications, and matchmaking, further enhancing the gaming experience. The combination of Nakama’s real-time capabilities and CockroachDB’s robust, scalable, and resilient architecture creates a powerful synergy that can handle the demanding needs of modern multiplayer gaming, ensuring a seamless and enjoyable experience for players worldwide.

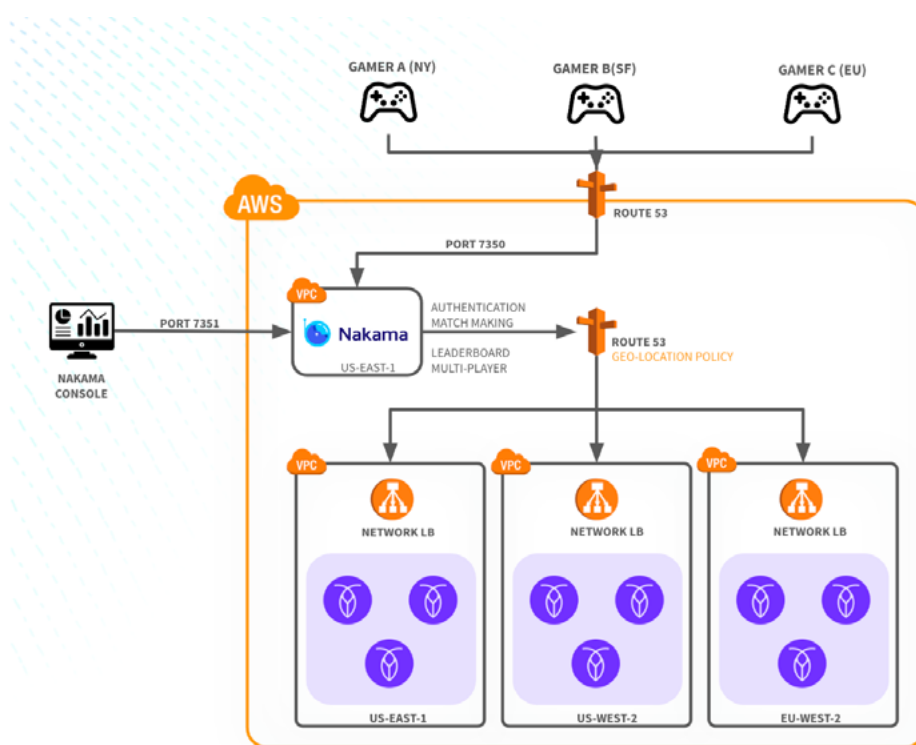


Figure 5. Architecture for “The Cockroach Lab Game” running on AWS with Nakama and CockroachDB

## Use cases of CockroachDB in gametech

CockroachDB is a great choice for gametech or gaming organizations that want the ability to horizontally scale as needed, maintain consistency, support extremely heavy traffic, and support transactions, among other things. Here are some of the use cases CockroachDB enables:

### Gamer system of record

Gametech platforms must manage vast amounts of player data, including user account details for authentication and authorization purposes. CockroachDB offers a dependable and scalable solution for storing this information, boasting geo-distributed capabilities for a global gamer community.

### Game persistence record

To guarantee consistent transactions and ensure that users return to the correct game state, CockroachDB's features play a vital role in maintaining game persistence records. Its ACID-compliant nature ensures that transactions are processed reliably and consistently, reducing the risk of data inconsistencies or lost transactions. CockroachDB's distributed architecture enables seamless data replication across multiple nodes and regions, providing fault tolerance and ensuring that the game state remains accurate and up-to-date even in the face of hardware failures or network issues.

### In-game purchase transactions

When in Game mode, many gamers choose to purchase skins, life, tokens, etc. As more and more gamers join the game, this can turn into a number of transactions. Depending on the traffic, these transactions can affect a lot of writes as well as reads to the database. It's also critical to ensure that these transactions are consistent and available to the game immediately. For such use cases, CockroachDB is ideal as it provides consistent data while enabling scale and low latency.

### Game matchmaking

Multiplayer games played online require an efficient and low-latency matchmaking experience. This is typically managed and authorized by the game server; however, to serve this reliable, scalable, and low-latency experience, CockroachDB is the ideal back end to store and manage matchmaking-related data. Game developers can build a matchmaking service that leverages CockroachDB's features to create an effective matchmaking system. This includes leveraging data stored in CockroachDB such as player skill, geographical location, and in-game preferences to match players. The matchmaking service queries CockroachDB to fetch relevant player data to make these decisions.

## CockroachDB case studies

**DEVSISTERS** [Dev Sisters Case Study](#) 

 **Heroic Labs** [Heroic Labs Case Study](#) 



## Ready, set, match

Get started and get in touch. CockroachDB was built to work the way you work. The familiarity and reliability of SQL combined with the fully global and elastic scale of a distributed cloud database reduces both risk and complexity in building new applications – or modernizing existing services. To operate across multiple locations and cloud providers anywhere in the world, you can self-host open source CockroachDB in your own environment. To make multi-region and multi-cloud (and everything else) easy, choose CockroachDB dedicated, our fully hosted and managed cloud service.

To get up and running right away, try CockroachDB serverless, which delivers free and pay-as-you-go CockroachDB clusters for organizations of any size. It's a managed instance of CockroachDB that lets you start using your database immediately and autoscales based on your application traffic.

For information about all three ways to access CockroachDB, please visit [cockroachlabs.com/get-startedcockroachdb](https://cockroachlabs.com/get-startedcockroachdb). Or, for a personalized demo of CockroachDB or to speak with one of our solution architects about your requirements, contact us at [sales@cockroachlabs.com](mailto:sales@cockroachlabs.com).