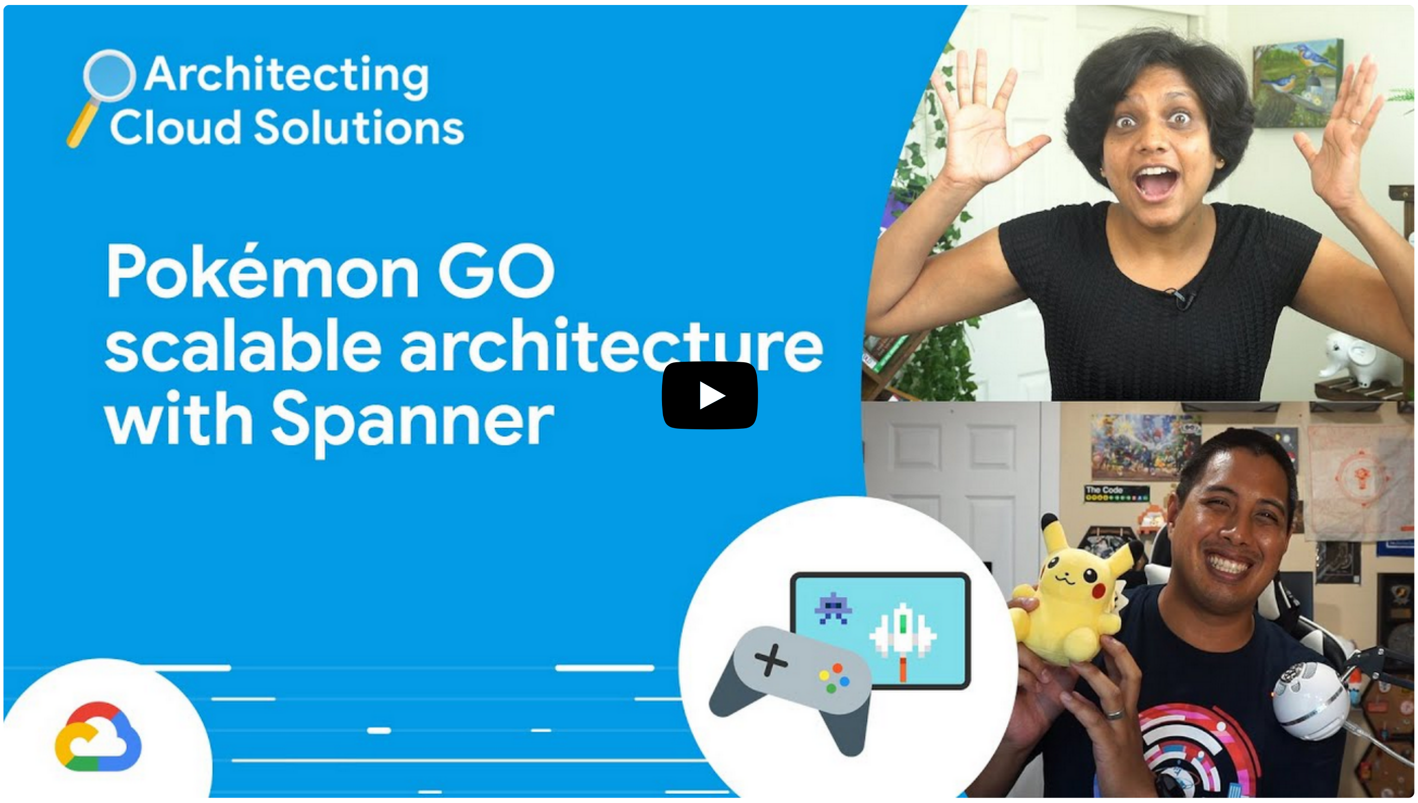
Google Cloud

Blog

Developers & Practitioners

How Pokémon GO scales to millions of requests?

October 27, 2021


Architecting Cloud Solutions
Pokémon GO scalable architecture with Spanner

Priyanka Vergadia
Staff Developer Advocate

James Prompanya
Senior Engineering Manager, Pokémon GO

asked him about the architecture that powers this extremely popular game. Checkout the video!

Priyanka: What is Pokémon GO?

James: It's not your typical mobile game. It's a game that involves walking around to catch these little Pokémon creatures that are appearing all around you in the real world. It encourages you to go outside, explore, and discover things using augmented reality.

A big part of that is the community aspect of it. When the game first came out, we hadn't built community features into the game yet, but players still met with others in real life, played together, and pointed out when rare, powerful Pokémon would appear. Everyone sees the same Pokémon, and shares the same virtual world, so when someone points out a Pokémon, you'd just see crowds of people running out after it. Nowadays, we make this a major part of the game by hosting regular live events such as community days, raid hours, all culminating in GO Fest, our annual celebration during the summer and our biggest event of the year.

During these events, transactions go from 400K per second to close to a million in a matter of minutes as soon as regions come online.

Priyanka: How does the Pokémon GO backend scale to handle peaks in traffic during events such as Pokémon GO Fest?

James: There are lots of services we scale, but Google Kubernetes Engine and Cloud Spanner are the main ones. Our front end service is hosted on GKE and it's pretty easy to scale the nodes there — Google Cloud provides us with all the tools we need to manage our Kubernetes cluster. The Google Cloud console is easy to use, with detailed monitoring graphs, tools, and logging available to use with just a few clicks. The support we get from Google engineers is top notch, At any given time, we have about **5000 Spanner nodes** handling traffic. We also have thousands of Kubernetes nodes running specifically for Pokémon GO, plus the GKE nodes running the various microservices that help augment the game experience. All of them work together to support millions of players playing all across the world at a given moment. And unlike other massively multiplayer online games, all of our players share a single “realm”, so they can always interact with one another and share the same game state.

Priyanka: Were you always using Spanner? Or did you decide to make that architectural decision as the game got popular?

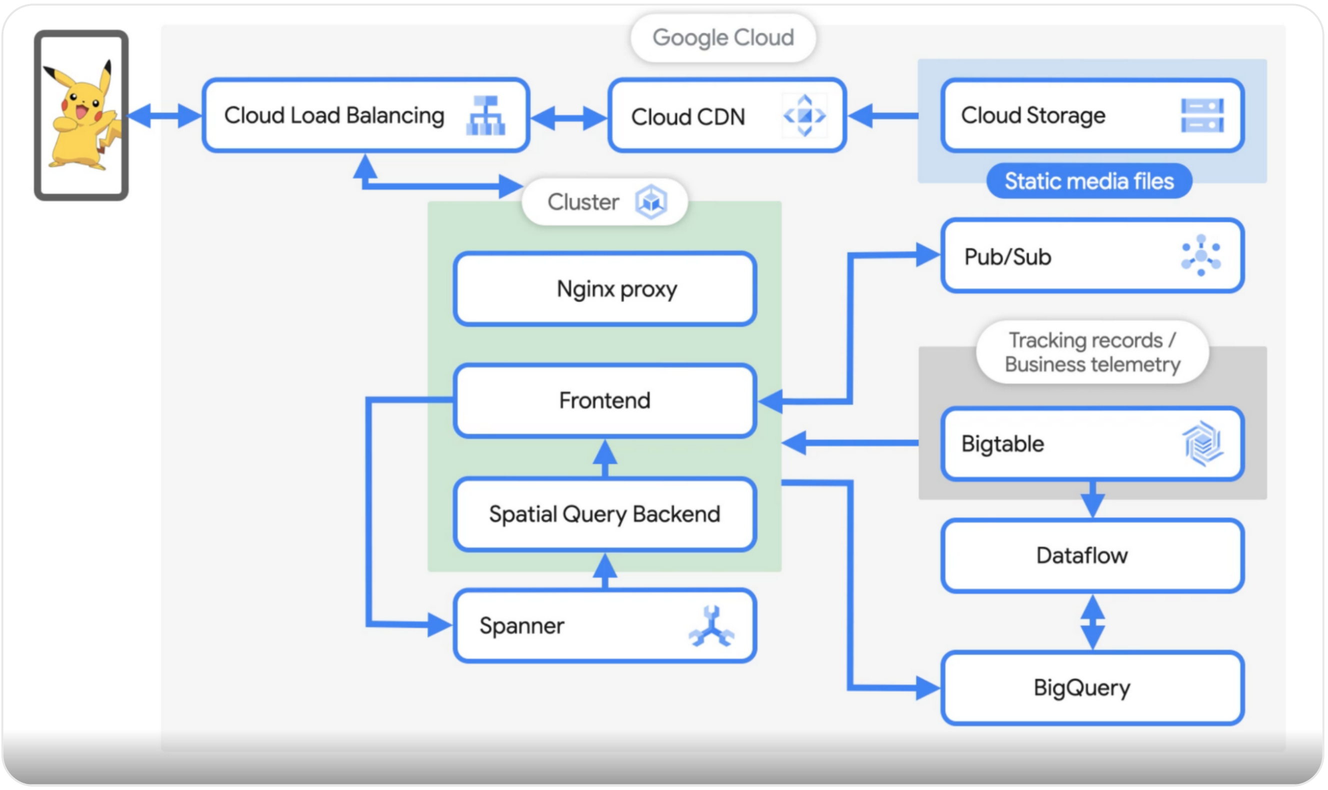
James: We started off using Google Datastore. It was an easy way to get started without having to worry about managing another piece of infrastructure. As the game matured, we decided we needed more control over the size and scale of the database. We also like the consistent indexing that Cloud Spanner provides, which allows us to use more complex database schemas with primary and secondary keys.

Finally, Datastore is non-relational with Atomic & Durable transactions, but we needed a relational database with full consistency. Spanner provides all of this, plus global ACID transactions.

Priyanka: Let's say I am a player, playing the game right now. I opened the app to catch Pokémon. What is happening behind the scenes - how does the request flow work?

James: When a user catches a Pokémon, we receive that request via Cloud Load Balancing. All static media, which is stored in Cloud Storage, is downloaded to the phone on the first start of the app. We also have Cloud CDN enabled at Cloud Load Balancing level to cache and serve this content. First, the traffic from the user's phone reaches Global Load Balancer which then sends the request to our NGINX reverse proxy. The reverse proxy then sends this traffic to our front-end game service.

interaction with the game, while the spatial query backend handles the map. The front end retrieves information from spatial query backend jobs to send back to the user.



Priyanka: What happens when I hunt a Pokémon down and catch it?

James: When you catch the Pokémon, we send an event from the GKE frontend to Spanner via the API and when that write request from the frontend to spanner is complete. When you do something to update the map like gyms and PokéStops, that request sends a cache update and is forwarded to the spatial query backend.

Spanner is strongly consistent: once the update is received, the spatial data is updated in memory, and then used to serve future requests from the frontend. Then the frontend retrieves information from the spatial query backend and sends it back to the user. We also write the protobuf representation of each user see the same Pokémon data, and keep that relatively in sync? (Especially for events!)

James: It’s actually pretty interesting! Everything on our servers is deterministic. Therefore, even if multiple players are on different machines, but in the same physical location, all the inputs would be the same and the same Pokémon would be returned to both users. There’s a lot of caching and timing involved however, particularly for events. It’s very important that all the servers are in sync with settings changes and event timings in order for all of our players to feel like they are part of a shared world.

Priyanka: A massive amount of data must be generated during the game. How does the data analytics pipeline work and what are you analyzing?

James: You are correct, 5-10TB of data per day gets generated and we store all of it in BigQuery and BigTable. These game events are of interest to our data science team to analyze player behavior, verify features like making sure the distribution of pokemon matches what we expect for a given event, marketing reports, etc.

We use BigQuery - it scales and is fully managed, we can focus on analysis and build complex queries without worrying too much about the structure of the data or schema of the table. Any field we want to query against is indexed in a way that allows us to build all sorts of dashboards, reports, and graphs that we share across the team. We use Dataflow as our data processing engine, so we run a Dataflow batch job to process the player logs stored in Bigtable.

We also have some streaming jobs for cheat detection, looking for and responding to improper player signals. Also for setting up Pokétops and gyms and habitat information all over the world we take in information from various sources, like OpenStreetMap, the US Geological Survey, and WayFarer, where we crowdsource our POI data, and combine them together to build a living map of the world.

more). The only thing that the Niantic SRE team needs to ensure is that they have the right quota for these events, and since these are managed services, there is much less operational overhead for the Niantic team.

Priyanka: With that much traffic, the health of the system is critical. How do you monitor the health of their system during these massive events?

James: We use Google Cloud Monitoring which comes built in, to search through logs, build dashboards, and fire an alert if something goes critical. The logs and dashboards are very extensive and we are able to monitor various aspects and health of the game in real time.

Next up, James and the Pokémon GO engineering team plan to explore managed Agones, Game Servers, stay tuned and checkout our entire customer architecture playlist.

We just took a behind the scenes tour into Pokémon GO’s architecture. How they use GKE and Spanner for scaling to those peaks and how their data science team works with BigQuery, BigTable, Dataflow & Pub/Sub for data analytics!

What did you think about this story? Tell me more about it on Twitter @pvergadia.

Google Cloud

Blog

Google Cloud

Developers & Practitioners

Under the hood: Distributed joins in Cloud Spanner

How do you join two tables when both of them are divided into multiple splits managed by multiple different machines? In this blog entry, we'll describe distributed joins using the Distributed Cross Apply (DCA) operator.

By Campbell Fraser • 4-minute read

Posted in Developers & Practitioners—Google Cloud—Spanner

Related articles

Cost Management

Keep a closer eye on Google Cloud costs with new Budgets for project users

By Mark Mirchandani • 3-minute read

AI & Machine Learning

What is Multimodal Search: "LLMs with vision" change businesses

Google Cloud

Blog

Get your BigQuery production sample, all self-serving

By Gustavo Kuhn Andriotti • 6-minute read

Healthcare & Life Sciences

Manage FHIR Data from Android App with Open Health Stack and Google Cloud

By Abirami Sukumaran • 9-minute read

Follow us

Google Cloud Google Cloud Products Privacy Terms

Help English