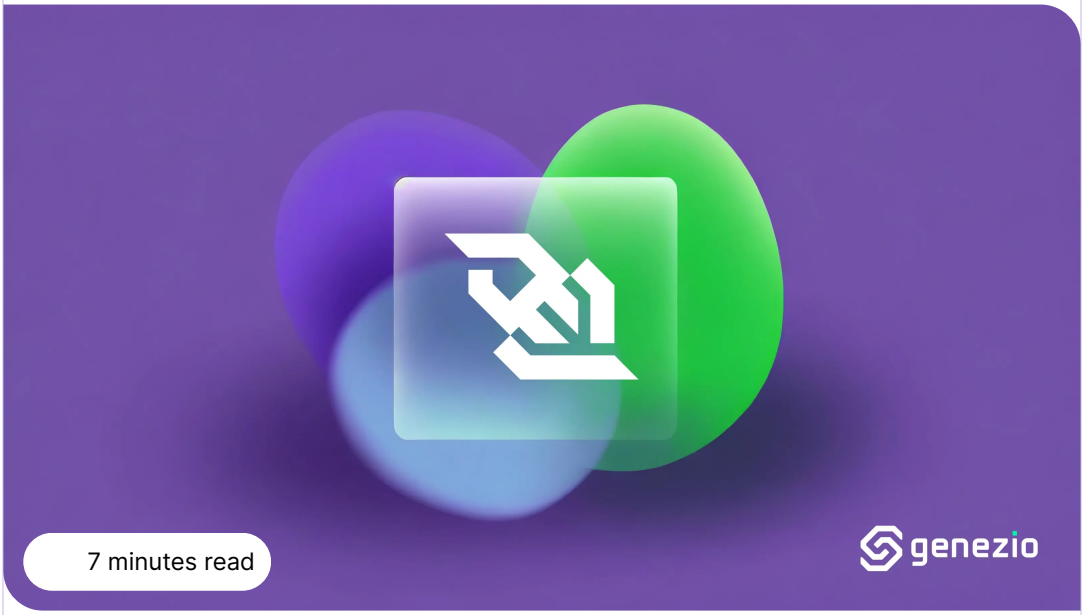
Blog

Run Scalable Real-Time WebSocket Applications on Genezio



Bogdan Vlad
Dec 11, 2024

Share on:

At Genezio, we've solved one of the biggest challenges in serverless technology: deploying auto-scalable WebSocket applications on Function-as-a-Service (FaaS) platforms.

After minimizing cold start times without compromising security, we turned our focus to real-time applications. To make WebSocket connections work seamlessly in a serverless environment, we had

to rethink our approach to enable persistent, real-time communication while maintaining the scalability and flexibility of serverless architecture.

A More Flexible Execution Environment

We introduce a new execution environment tailored for real-time use cases. Here's what it offers:

- **Direct HTTP connection:** Instead of wrapping your application in a function, you can listen directly on a port, like any regular HTTP server. This approach provides out-of-the-box support for WebSockets and HTTP servers.
- **Parallel HTTP connections in the same execution environment:** Compared to AWS Lambda, where parallel requests spawn individual functions, a single Genezio Execution Environment can serve parallel HTTP connections, minimizing idle time and the number of cold starts.
- **Auto-Scalability:** WebSockets are no longer a bottleneck. The architecture scales automatically up and down, accommodating any level of demand without performance degradation.
- **No Hidden Limitations:** Performance and security remain uncompromised, with no unexpected trade-offs or constraints.
- **Simplified Architecture:** We eliminate the need for over-engineered designs, such as complex and non-standard gateway APIs.

From Functions to WebSocket Connections

When we first launched, our platform prioritized full compatibility with AWS Lambda, given the extensive codebase and tooling already developed for this environment. This strategy has proven successful, enabling us to onboard developers with existing applications who are seeking an AWS Lambda alternative that offers better performance and greater cost efficiency.

AWS Lambda operates in a discrete manner - each invocation triggered independently, receiving HTTP request data as input and returning the HTTP response as output. While the response can be streamed, there is no way to maintain a direct connection between the client and the server hosted on an AWS Lambda function.

Developers told us they needed something more: the ability to deploy WebSocket servers, maintain long-living connections, and handle parallel requests within the same function.

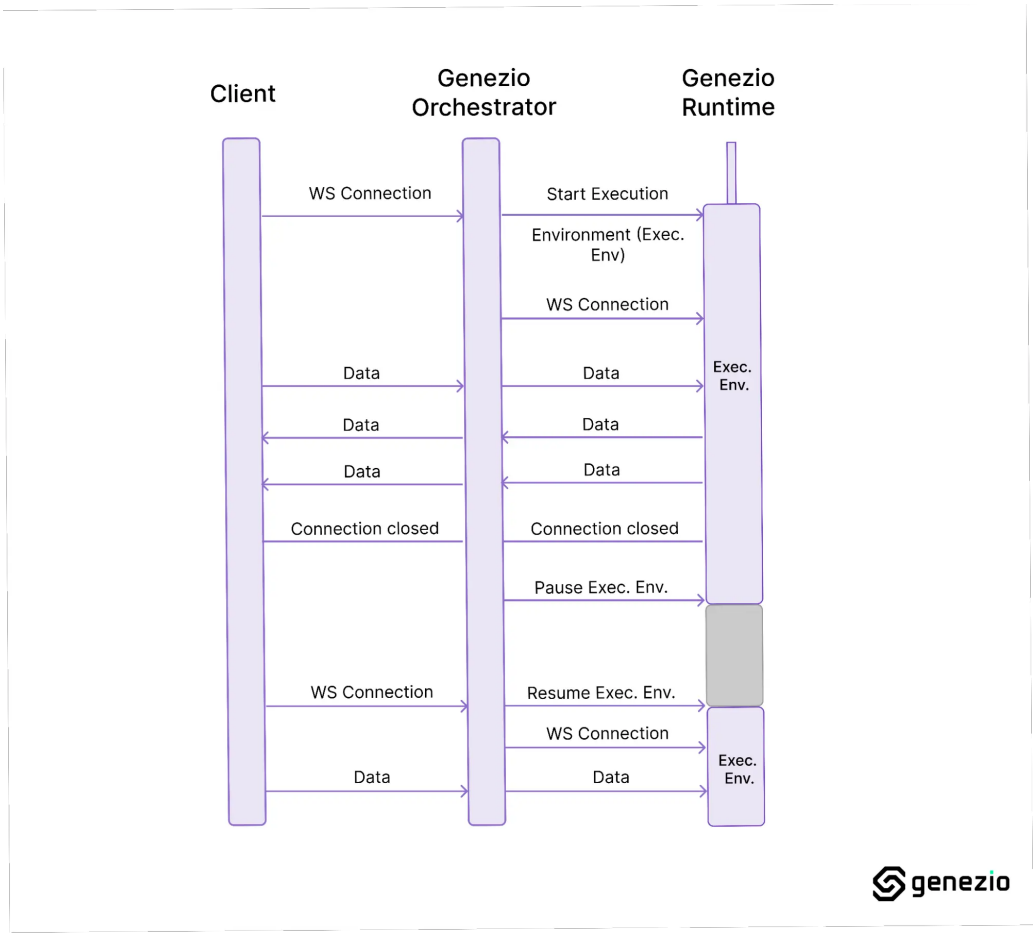
So, we rethought how our platform interacts with user code. Instead of following the traditional “function” paradigm, we now **forward HTTP connections directly to the execution environment**. The execution environment remains active as long as the HTTP connections are in use. When no active HTTP connections exist, the execution environment is either paused or terminated, ensuring compliance with the scale-to-zero promise.

How It Works

Let’s break it down with an example: a basic Node.js HTTP server application:

- **Cold start:** When the first request arrives, the execution environment starts, and the HTTP connection is passed to it.
- **Pause:** After the response is sent back to the client, the execution environment is paused.
- **Resume:** For subsequent requests, the execution environment is resumed. Multiple concurrent requests can be served by the same execution environment to minimize the CPU idle time and the number of cold starts.
- **Scaling:** If traffic spikes, additional execution environments are spawned to handle the increased load.

This model also works seamlessly with WebSocket connections. When the first WebSocket connection is received, the application starts and remains active as long as data flows through the connection or until a configurable timeout is reached.



Backwards Compatibility

Your existing Genezio applications will continue to run as they always have. A compatibility layer translates HTTP requests into function invocations. Additionally, this compatibility layer can also be utilized by new applications that require the function invocation interface.

Most importantly, this paradigm shift hasn't impacted performance. Cold start times remain low, and we continuously monitor response times to ensure no degradation across scenarios.

Why This Matters

We've redefined the concept of serverless functions by passing HTTP connections directly to the execution environment. This breakthrough eliminates traditional limitations, enabling developers to build dynamic, real-time applications like WebSocket servers while:

- Maintaining the core benefits of serverless (auto-scaling, scale-to-zero, no maintenance).
- Handling keep-alive connections and parallel requests effortlessly.
- Simplifying deployment with a straightforward architecture.

Ready to Build?

It's time to take your serverless applications to the next level. Experience the freedom of building real-time apps with Genezio.

You can start from one of our templates:

- A very simple echo websocket application:
  

- A full stack chat socket-io application that uses MongoDB adapter for scaling capabilities:
  

Subscribe

Getting Started with Django: Build Your First Web Hosted App in 10 Minutes

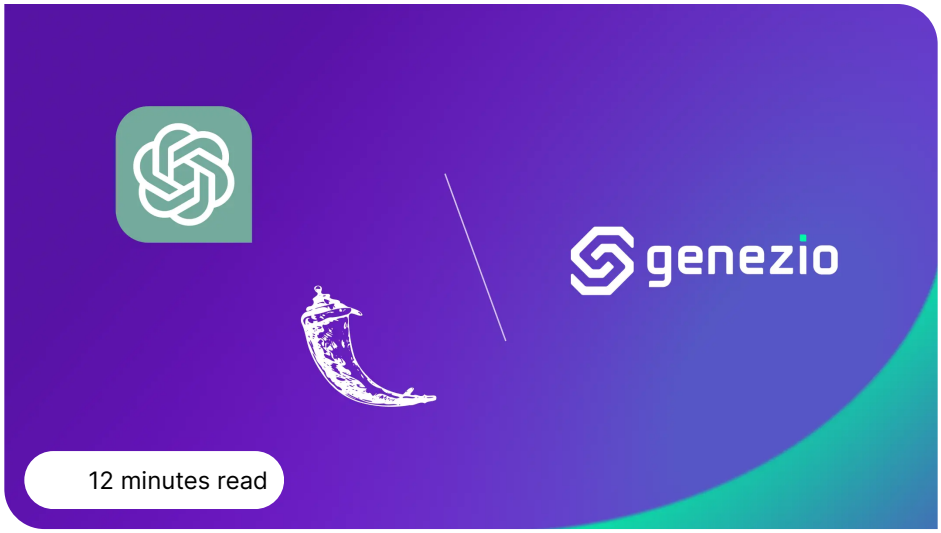
Build a ChatGPT Chatbot App with Flask: A Step-by-Step Guide

Related articles

Getting Started with Django: Build Your First ...

Hi, I'm Cristi Miloiu, and in this tutorial, I'll guide you through building your first web application using Django, one o...

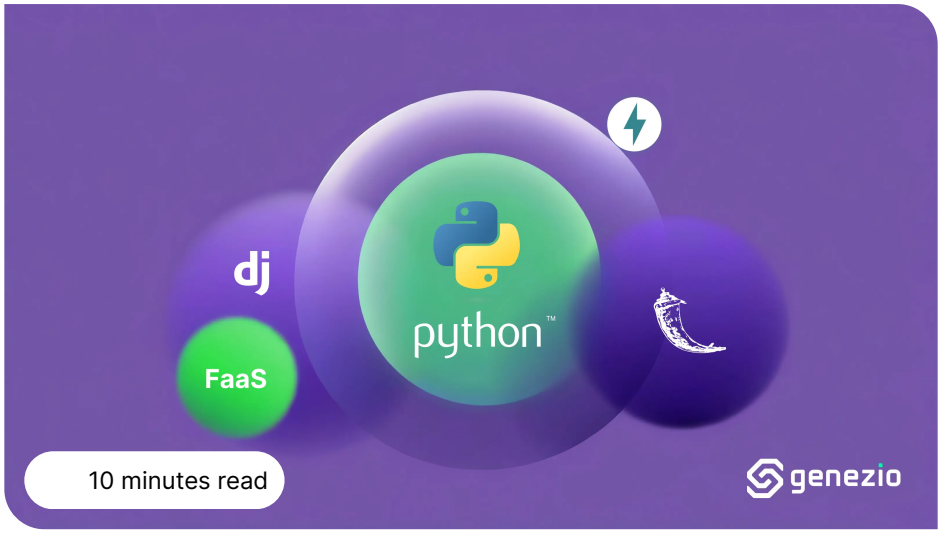
Cristi Miloiu
Dec 12, 2024



Build a ChatGPT Chatbot App with Flask: A Step...

Hi! I'm Cristi Miloiu, and in this article, I'll walk you through building a ChatGPT-powered chat app using Flask, Python, ...

Cristi Miloiu
Dec 10, 2024



Genezio Brings Serverless Python Support: Djan...

We're thrilled to announce that Genezio now fully supports deploying Python popular frameworks like Django, Flask, FastAPI....

Cristi Miloiu
Dec 09, 2024

More from News

Case Study: How Genezio Helped BwareLabs Automate Their Workflows

Luis Minvielle
Oct 07, 2024

Genezio V1.0 is here

Paula Cionca
Mar 19, 2024

Collaboration, Queues and New Project Wizard - genezio v0.7

Bogdan Vlad
Dec 18, 2023

Introducing a new configuration file for your projects

Costin Sin
Mar 20, 2024

Integrate Redis and Postgres plus much more - genezio v0.6

Andra Pitis
Nov 09, 2023

Genezio Partners with MongoDB Atlas: Effortless NoSQL Database Creation

Costin Sin

Oct 22, 2024

Services

Genezio Typesafe
Authentication
Databases
Cron Jobs

Frameworks

Express.js
Next.js

Resources

Blog
Docs
Github

Support

Contact Us
Privacy Policy
EULA
Brand Assets
Status
Careers

Subscribe to our newsletter

Genezio is a serverless platform for building full-stack web and mobile applications in a scalable and cost-efficient way.

Enter your email

Subscribe