Lars Erik Schonander

For market research, I would plan to talk to a variety of different types of developers, to see what pain points they face, and how Cloudflare Workers could be helpful. I would also do research on other serverless products to see if they have been implemented in the context of video games to get a better understanding of how serverless code can be uniquely helpful for game development.

One issue that video game developers face is server management. If one is renting a server or has one On-Prem, an issue is that these servers are being used all the time, even if current traffic is low. On the other hand, when traffic spikes, a company likely needs to use additional servers, so the game does not suffer under the strain of many users. The problem that Cloudflare Workers for gamers is solving for is writing code for functions that need to be scaled for traffic, but also ensure that the developer does not need to care about scaling and deploying their code across the world, as Cloudflare via the CDN's and automatic scaling can handle that.

Product changes that might be helpful could include additional documentation and examples on how to use Cloudflare Workers for mobile devices, so mobile game developers could be a part of the market as well.

For improving the quality of the offering, a closed beta with video game developers of various types (mobile, MMORPG, action) could be done. A closed beta would be helpful to see how video game developers would use the product, but also be a source of information to understand not only what they found helpful, but also find ways we could improve Cloudflare workers for video games.

A deliverable for the "Cloudflare Workers for gamers product" could be tooling to use workers within Unity, a game development framework. In 2016, there were 5.5 million developers, and likely more now, providing a large market to work with. While Unity does not support C++, C, or Rust, it still supports JavaScript, which is one of the languages Cloudflare Workers also supports. Tutorials would be required to explain the benefits of using Cloudflare workers. My example would be to create a Cloudflare Workers function for a daily update for a given user or for global events that require certain prerequisites. Instead of maintaining a function running in the background which costs money due to server costs, one could just write it as a Cloudflare Workers function that only runs when the conditions for it being trigged are fulfilled. Beyond the actual code to create compatibility between Cloudflare workers and Unity, the day the product comes out should also come with tutorials, sample repositories on GitHub and FAQ's to make it as easy as possible for a developer to get started without getting frustrated and abandoning the product because they are unsure how to proceed.

To measure the deliverable's success, I would measure if the Unity collaboration occurred how many people downloaded the library/SDK and track that to see if the number of users increases vs decreases over time. For usage beyond the library, I would want to track how much the new tutorials for Cloudflare Workers for gamers would be talked about by for exampling, seeing how often the tutorials were mentioned on Twitter or how often the sample code is forked on GitHub.

A risk that could lead to the failure of the product is if it turns out the way the product was marketed/viewed did not quite fit the needs of what actual video game developers needed. A way to recover from that type of potential failure state, however, would be to collect success stories like Cloudflare does for its other products. It will be useful to see Cloudflare workers as used by actual video game developers to potentially pivot the marketing and various tutorials to fit the use case in the wild and maximize how many developers could use the product.