Intro

We really appreciate your dedication. We're going to give you proper support during the test, so apart from this document you should receive by email an invitation to join a #slack channel in CARTO, where you can ask questions to our engineers.

Asking questions is fine, don't be shy, we'll evaluate that positively. We prefer you to ask questions rather than spending hours on your own research to finally get stuck. See the **Support** section at the end of this doc to know more.

In this assignment we will evaluate the following aspects:

- **Simple and clean code**. You should come up with a clear component-oriented solution (good naming, hierarchy, folder structure and so on...). Avoid over-engineering, the simpler the better.
- **Communication**. During the whole test: in the slack channel, in the code, when delivering...
- The test should be done in a **maximum of 6 hours**. We assume you will produce in this amount of time an incomplete infrastructure. Just focus on the most important things. We will discuss

Part 1 - Local development environment

You have just entered a new project and the developers give you the current code of the API (can be downloaded in CARTO_DevOps_test_API__v1_3.zip). They also transmit to you the next problems that you should solve:

- They need a common local, stable, and uniform development environment for all. They use multiple OS (mainly Linux and Mac) and had multiple problems caused by the use of different technologies between production, CI, and local environments.

- They need a stable table on BQ. Right now, that table is updated frequently causing problems so they would like to have an exactly equal copy in another place to avoid problems. They also would like to have a script or similar to be able to replicate themselves in the future. The table is "carto-demo-data.demo_tilesets.osm_buildings".

Please give, implement, and document the solution for the problems.

Part 2 - CI/CD

Based on part 1, implement a CI/CD in order to check the API and deploy it in a cloud provider of your choice.

We need at least, 3 different environments (including production).

Please implement and explain how this would work, the flows to use the CI/CD that should be followed by developers, and any technology involved in the process.

Part 3 - Deployment

Now you need to implement the deployment. It must be container-based. The preferred choice here is Kubernetes, but it is not mandatory, so if you pick any other solution, please explain your choice and why you chose it.

Requirements:

- 5 x API containers
- 1 x Cache container: it should be deployed as an API cache, so once a response is cached, subsequent requests will be served from cache content. Please elaborate on your technology choice and any knowledge about caches and invalidation strategies.

Please, provide **all** config files, along with instructions on how to use them. Don't forget to also include valid IaC code to create all the infrastructure resources needed to deploy.

Part 4 - Production-ready

Please explain in detail what is needed following your criteria for this project to be production ready.

Notes

- To test the different environments you can use the following jsfiddle.
- Our currently used technologies in production are GitHub, GitHub Actions, and Google Cloud. Anyway, you can use whatever technology of your choice.

Delivery

Once you've finished your work, we would like to have:

- access to the git repo with it
- some kind of README with instructions & any detail you may find relevant for us (eg. reasons to choose a technical approach, things that could be improved, etc.)

Support

You should receive an invitation to join a slack channel (something like #test-support-yourname), where you will be able to ask anything related to this assignment. The people from the **Infrastructure** team will try to help you.