Distribution Plan For Car Finder Application

By: Ian Estevez ,Daniel Tribaldos, Lucas Hardeman, Amisaday Santiago, Elizabeth Sauer, Benjohn Shung

Modern web applications are increasingly adopting containerization for various reasons, including ease of deployment, scalability, and consistency across different environments. Our team recently worked on a car finder web application that needed to be containerized.

The database we decided to use was Google’s Cloud Firestore, a NoSQL cloud database. As a result of using Google’s database, the search for the cloud platform became quite simple. It made the most sense to choose Google’s Cloud Platform. To containerize I believe Google Kubernetes Engine would be the best bet. In searching for this method of deployment we thought about using Amazon EKS (Elastic Kubernetes Service) instead of the GKE (Google Kubernetes Engine) however the Amazon EKS pairs better with the AWS ecosystem and we had been using a GCP (Google Cloud Products with our database. Another option was the Azure Kubernetes Service, but I have worked with docker deployment in Azure, and it didn’t make sense if we had a Google Database.

To begin we would need to create a Google Cloud account and within the directory of our flask app we would need to begin to configure the Dockerfile. The Dockerfile is the file that contains commands to create the docker image with our python code and necessary libraries within. I think this may be the most crucial step of containerization. The docker file must have the correct python image, directory, requirements, installations, files, port numbers, and commands just to be able to containerize properly. After finishing that file, we would need to build the image and push the image to the Google Container Registry. Next, we would need to Create and configure a Kubernetes cluster. To configure the deployment, we would need to have YAML file to run on the GKE. This would containerize and deploy the web app. It is important to note that because our database is cloud based it saves the need for management within the GKE cluster. Although we did not actually deploy with any of this, we chose docker because some members of the group had experience with it and it is the most widely used container engine.

Containerizing and deploying a web application on GCP, with integrated services like Firestore and GKE, made this approach quite linear. This method simplifies the deployment process while taking advantage of Googles cloud and engine.