Capstone Project Architecture Overview

I decided to go with Azure as my cloud service provider because they provide $200 of free services and out of the big three cloud service providers their interface was the most intuitive for me. The production application is split up into two containers. The first container is a react application that runs on an Nginx server in production. I chose to containerize the application because it makes it easy to deploy and to work within multiple environments. Additionally, with the help of an orchestrator like Kubernetes it will be easy to scale my frontend should the need arise. Another option would be to deploy the front end to a CDN. I chose the first option because it is also easy to run the application locally and thus will make it easier for others to run my application in their local env.

The second container is a fast API server that wraps around the yolov3 model. That's run via open cv2. I chose to run the model this way because it's three times faster when run on CPU vs the original darknet implementation, this proved to be extremely important since getting a GPU via azure has been nearly impossible. I've been trying for the last four weeks. Containerizing this application provides pretty much the same benefits: it's easier to run, deploy, and scale.

For file storage I'm using azure blob storage because it auto-scales and is accessible from anywhere. Furthermore, in the future, it can also be segmented so that only certain users can upload to a specific folder. Additionally, the API is extremely easy to use. For our current use case, I believe this setup is more than sufficient. It provides easy deployment and will be easier to scale in the future.

One modification I would like to make would be to separate the second container into microservices, one for the model and another for the API that uploads files to the azure containers. This will allow us to scale the model separately and faster since Yolo v3 is the bottleneck of our application. We can have a pool of yolov3 containers connected to an azure data bus and assign new videos to process as they come in. The API for uploading and downloading files from azure blob will be able to handle exponentially more requests than the model can. And thus does not need to scale with the model.