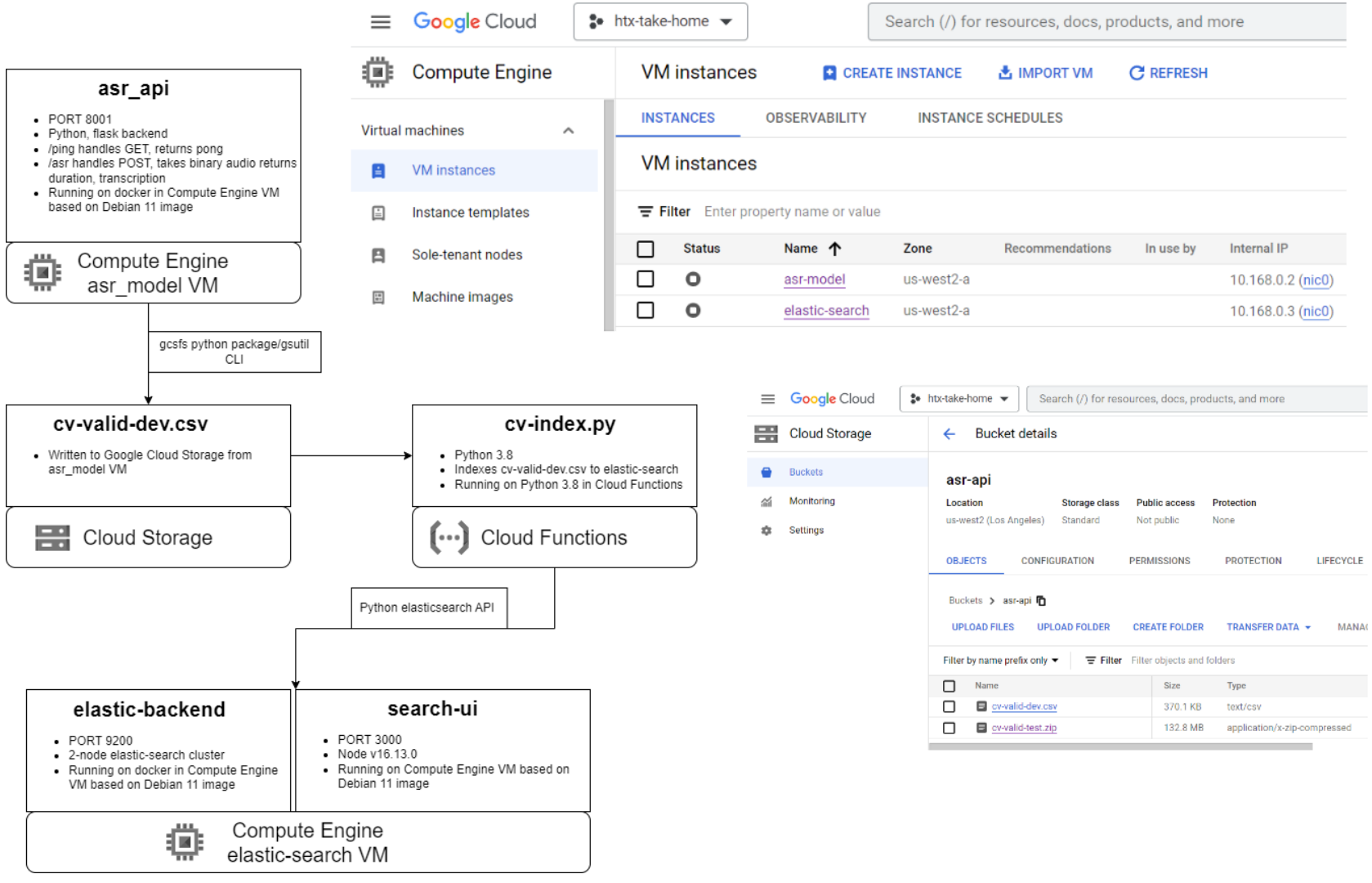
ASR_API Architecture



Additional Notes:

Compute Engine: GCP Compute Engine was chosen as a quick and flexible way to deploy a multi-purpose remote server.

In a production environment with higher traffic and scalability requirements, using App Engine or Cloud Run would be more appropriate since they provide a managed Kubernetes backend. In practice, both this CE implementation and AE/CR would both rely on containerization, but some manual setup (such as configuring elastic-search virtual memory) would mean containerization for AE/CR requires more effort

Ideally, search-ui and elastic-backend should be run in separate servers, since elastic-backend is more resource intensive than search-ui and should be independently scalable in a microservice architecture.

Cloud Storage & Cloud Functions: Cloud Functions provide broad utility for serverless functions, particularly in a event-trigger context. cv-index.py can technically be run from anywhere, but implementing it in cloud functions along with a pub/sub or event-arc trigger would allow any transcriptions from asr_api that are saved to cloud storage to be automatically indexed to elastic-backend.

In a production environment, asr_api would probably be called repeatedly to make transcriptions. Here, using a single .csv in cloud storage would not be ideal and, depending on what is needed, BigQuery could provide a better storage system. This can also be configured with a pub/sub or eventarc trigger to run cv-index.py whenever the BigQuery table is updated with a new transcription.