Model: GPT-5 (Thinking set to Auto) Prompt level: 1 Candidate ID: gpt5-P1

Base-Prompt

```
Create Terraform for a single-node k3s cluster on Ubuntu to deploy the "Semantic
Segmenter" app with:
- Namespace: `semseg`
- Deployments: ingest-api, results-api, qa-web, convert-ply, part-labeler, redactor,
analytics, redis
- Services: ingest-api, results-api, qa-web, redis

Keep image names as variables and focus on generating the Kubernetes resources via
Terraform. You may assume reasonable service ports, exposure (NodePort vs ClusterIP),
and storage details following best practices for single-node k3s. Apply Terraform best
practices for module structure and code quality (clear modules, variables, outputs,
pinned providers). Use a shared image variable for workers and web UIs
(`shared_image`), a separate image for `ingest-api` (`ingest_image`), and `redis:7-
alpine` for Redis. Focus on a working, single-replica deployment; do not include
autoscaling (HPA/VPA) or replica tuning. Do not include security hardening (RBAC,
PodSecurityPolicy/SCC, NetworkPolicy, Pod/Container securityContext); focus only on
the provided information.
At the end, include example Terraform commands to run (fmt, init, validate, apply).
For apply, include -var assignments for any variables your solution requires.
```

Provided structure

```
semseg-terraform/
├─ versions.tf
├─ providers.tf
├─ variables.tf
├─ main.tf
├─ outputs.tf
└─ modules/
   ├─ k8s-namespace/
   │  └─ main.tf
   ├─ k8s-app/
   │  ├─ main.tf
   │  └─ variables.tf
   └─ k8s-redis/
      ├─ main.tf
      └─ variables.tf
```

Extra prompts needed

Prompt 1

```
Error: Invalid single-argument block definition on modules\k8s-redis\variables.tf line
5, in variable "storage_gi": 5: variable "storage_gi" { type = number, default = 1 }.
Please fix error.
```

Prompt 2

```
Extra information for providing terraform commands: I am testing in a local docker
desktop cluster, the shared image is semantic-segmenter:latest, the ingest-api image
is ingest-api:latest. Common env might not be the best choice as we have different env
vars for each service:
ingest-api: INGEST_OUT_DIR
results-api: SEGMENTS_DIR, REDIS_URL, REDIS_STREAM_FRAMES_CONVERTED,
REDIS_STREAM_PARTS_LABELED, REDIS_STREAM_REDACTED_DONE, REDIS_STREAM_ANALYTICS_DONE
qa-web: RESULTS_API_URL
convert: REDIS_URL, REDIS_STREAM_FRAMES_CONVERTED, PREVIEW_OUT_DIR
labeler: REDIS_URL, REDIS_STREAM_PARTS_LABELED, REDIS_STREAM_FRAMES_CONVERTED,
REDIS_GROUP_PART_LABELER
redactor: REDIS_URL, REDIS_STREAM_PARTS_LABELED, REDIS_STREAM_REDACTED_DONE,
REDIS_GROUP_REDACTOR
analytics: REDIS_URL, REDIS_STREAM_PARTS_LABELED, REDIS_STREAM_ANALYTICS_DONE,
REDIS_GROUP_ANALYTICS
Please only respond with necessary fixes, and where they should be implemented.
```

Prompt 3

```
Please finish writing the env variables with this information below:
Streams: s_frames_converted, s_parts_labeled, s_redacted_done, s_analytics_done
Groups: g_part_labeler, g_redactor, g_analytics
/sub-pc-frames → PVC sub-pc-frames-pvc
   - Producers/consumers: ingest-api (writes), convert-ply (reads)
/pc-frames → PVC pc-frames-pvc
   - Producers/consumers: convert-ply (writes), part-labeler (reads), redactor (reads)
/segments → PVC segments-pvc
   - Producers/consumers: convert-ply (writes previews), part-labeler (writes labels
and colorized), redactor (writes), analytics (writes), results-api (reads)
/segments/labels (subdirectory under /segments)
   - Producer: part-labeler (colorized outputs)
```

Prompt 4

```
In-cluster Service DNS: Kubernetes resolves Services as <service>.
<namespace>.svc.cluster.local , please fix provied URLs based on this. We also need to
add args to some deployments, please do that based on the information provided below:
results-api: uvicorn services.results_api.app:app --host 0.0.0.0 --port 8080 --workers
1
qa-web: uvicorn services.qa_web.app:app --host 0.0.0.0 --port 3000 --workers 1
convert: --in-dir /sub-pc-frames --out-dir /pc-frames --preview-out-dir /segments --
delete-source --log-level info
labeler: python3 /semantic-segmenter/services/part_labeler/part_labeler.py --log-level
info --out-dir /segments --colorized-dir /segments/labels --write-colorized
redactor: python3 /semantic-segmenter/services/redactor/redactor.py --log-level info -
-out-dir /segments
analytics: python3 /semantic-segmenter/services/analytics/analytics.py --log-level
info --out-dir /segments
```

Prompt 5

```
Please fix convert-ply's command as it has to be a python3 call, similar what you can
find in the other deployments args. Structure is /semantic-
segmenter/services/convert_service/convert-ply
```

Prompt 6

```
PVCs are missing from the implementation, please implement them based on what we
included in the envs.
```

Prompt 7

```
redis:// is missing from the beginning of redis URLs.
```

Prompt 8

```
I need you to expose ingest-api as we have to reach it from outside to be able to send
test files.
```

Notes

- 1st extra prompt provided solution for the Error, but also included removing two dynamic blocks "args" and "command" from modules/k8s-app/main.tf

- 2nd extra prompt provided solution for the env's but it was a bit weird. It provided the URLs for the envs but not the redis groups and streams, for each of those it wrote "set your stream/group", but other than this the fix for this issue looked acceptable.
- 3rd extra prompt provided the correct envs, but still no correct URLs.
- 4th extra prompt provided good URLs, and sme good args, but conver-ply's command was faulty as it was missing the python call.
- 5th extra prompt provided the correct arg for convert-ply.
- I manually changed kubeconfig default path for testing purposes.
- For the 6th extra prompt it provided a correct but bit complex pvc implementation.
- I noticed late the the redis url is still missing redis:// from the beginning, so we needed to provide th 7th extra prompt.
- 8th extra prompt provided good solution.
- Noticed /0 suffix missing from redis URLs causing some problems, I fixed this manually.
- Updated redis args to ["--appendonly", "yes", "--save", ""] manually.
- Changed log-level to debug for debugging purposes manually.

Local test apply command

```
terraform apply -var="shared_image=semantic-segmenter:latest" -
var="ingest_image=ingest-api:latest"
```

Scoring (with notes)

Scores (1..5):

- Correctness (docker): 3 — Works on Docker Desktop after fixes (redis:// prefix, /0 suffix), but required multiple corrections during iteration.
- Kubernetes fit: 3 — Reasonable single-node k3s defaults; however, several services lack probes and some exposure choices are inconsistent with the stated rubric.
- Storage: 4 — PVCs implemented and mapped to the described directories; approach is correct though a bit more complex than necessary.
- Image handling: 4 — Uses shared_image and ingest_image variables as requested; Redis pinned to 7-alpine with appropriate args.
- Networking: 3 — In-cluster DNS fixed; ingest-api exposed as requested, but results-api exposure and NodePort conventions aren’t fully standardized.

- Modularity: 3 — Separate reusable modules could have been created for repeated resources (e.g., deployments and PVCs).
- Observability: 2 — Readiness/liveness probes are largely missing; logging guidance present but limited health checks.
- Reasoning: 3 — Addressed issues through a sequence of prompts; some missteps (e.g., env URLs, command formats) required later fixes.

Overall (avg): 3.1 / 5