Sanity Matrix

Layer	What we need	Your current state	✅ / ❌	Notes / CLI to double-check
VPC & Sub-nets	• Private-A subnet-0fc666… • Private-B subnet-05210f… • Both routed **0.0.0.0/0 → NAT nat-0c1bf41f…**	Confirmed in *podinsight-private-rt* (rt 0c4da16…)	✅	aws ec2 describe-route-tables --route-table-ids rtb-0c4da16c… --query 'RouteTables[0].Routes'
Interface VPC endpoints	S3, ECR (API & DKR), STS, CloudWatch Logs	5 endpoints present (vpce-07b5f0…, etc.)	✅	Endpoints sit in both private sub-nets
Security group for Batch traffic	SG must allow **egress ALL** and **ingress TCP 443** *from itself*	SG sg-0e769c5121c9fe9c7 now has self-referencing 443 rule	✅	aws ec2 describe-security-groups --group-ids sg-0e769c5121c9fe9c7 --query 'SecurityGroups[0].IpPermissions'
NAT GW	One public NAT servicing the two private sub-nets	nat-0c1bf41f… (Elastic IP 18.132.154.20) = Available	✅	
Service-linked role	AWSServiceRoleForBatch present	Exists & attached	✅	
Instance profile (hosts)	ecsInstanceRole – contains **ECR-RO**, **ECS-for-EC2**, **AmazonS3ReadOnly**	Present & PassRole allowed	✅	
Job role (inside container)	podinsight-task-role – inline "access" policy + **ECR-RO**, **S3-RO**, **Dynamo**, **CloudWatch**	Present & PassRole allowed	✅	
AMI (GPU builder)	Launch template **pod-insight-dlami-largev3 ver 1** built from Deep-Learning AL2023, 60 GiB gp3, **Whisper Large-v3 weights pre-cached**	Launch template resolves to AMI ami-0eab4… (owned by you)	✅	(AMI does *not* show in the console's "Owned by me" list because it was baked **after** the LT; totally normal)
ECR image	594331569440.dkr.ecr.eu-west-2.amazonaws.com/podinsight/transcribe:latest	Digest sha256:3bbff2…, size ≈ 4 GB	✅	aws ecr describe-images --repository-name podinsight/transcribe --image-ids imageTag=latest
CloudWatch log group	/aws/batch/podinsight exists	Auto-created	✅	
Compute Env	podinsight_gpu_smoke_ce type **EC2**, max vCPU 32, template above	State **ENABLED, VALID**	✅	aws batch describe-compute-environments --compute-environments podinsight_gpu_smoke_ce
Job Queue	podinsight_smoke_q points *only* to the CE	State **ENABLED, VALID**	✅	
Job Definition	podinsight_smoke_jd:3• platform = **EC2**• vCPU 4, Mem 16 GiB, GPU 1• command pulls tier1_smoke.csv limit 1, --model_size large-v3	Latest rev = 3	✅	aws batch describe-job-definitions --job-definition-name podinsight_smoke_jd --status ACTIVE
Buckets & table	pod-insights-manifests, pod-insights-raw, pod-insights-stage, pod-insights-insights/raw_entities …, Dynamo podinsights-status — all in eu-west-2	Present	✅	S3 default encryption = AES-256 (no KMS key needed)

Area	What we checked	Result	Notes / next move
VPC & sub-nets	Private-A/B → route-table points *0.0.0.0/0 → nat-0c1bf41f…*Public-A/B → *0.0.0.0/0 → igw-05dbca6d…*	✅	NAT/IGW paths in place, so the Batch instance can reach the Internet if an interface-endpoint is ever unavailable.
Interface endpoints	*ecr.api*, *ecr.dkr*, *s3*, *logs*, *sts* exist in podinsight-vpc and are attached to the two private sub-nets.	✅	Minimises outbound $ and keeps image-pulls inside the VPC.
Security-group mesh	Self-referencing **HTTPS-ingress** rule on sg-0e769c5121c9fe9c7 (Batch + endpoints)	✅	You added tcp/443 ← sg-0e769c5121c9fe9c7 so ECR calls are no longer blocked.

IAM – instance profile	ecsInstanceRole attached to CE, has *AmazonEC2ContainerServiceforEC2Role* + *AmazonEC2ContainerRegistryReadOnly*	✅	EC2 host can register with ECS & pull from ECR.
IAM – job role	podinsight-task-role attached to JD revision 3, inline *access* policy for S3 + Dynamo + CW metrics	✅	Container will see AWS_REGION, AWS_ACCESS_KEY_ID, etc.
CloudWatch Logs	/aws/batch/podinsight group exists; job definition uses awslogs driver.	✅	Once a container *starts* you'll see streams under that group. (Currently nothing because the job never leaves RUNNABLE.)
Launch template	pod-insight-dlami-largev3 points at AMI that already has **Whisper-Large v3** & Docker layers cached.	✅	Warm launch: no first-run weight download cost.
Batch plumbing	CE podinsight_gpu_smoke_ce → JQ podinsight_smoke_q → JD podinsight_smoke_jd:3	✅	Wiring is correct.
Service-quotas (blocking)	**Running On-Demand G & VT instances – quota = 0** (code L-DB2E81BA)	✅	You submitted an increase to **20 vCPU** – currently **PENDING**. Until AWS approves (usually < 24 h in London) *no* On-Demand g5.xlarge can be started, so jobs stay RUNNABLE. Spot quota is fine (90 vCPU).

ami-0457d1bb9a9b5c796

Below is a **pure-console run-book** that starts with quick sanity-checks, then walks straight through spinning up **one g5.xlarge node in eu-west-2**, running a single Whisper-Large job, watching it succeed, and finally tearing the node back down.

(Every CLI action you've been doing is available in the console UI – this guide shows you exactly where.)

---

0 Fast sanity checks (5 min)

What to verify	Where (console)	What "good" looks like
Service quota	*Service Quotas → Amazon EC2*	**Running On-Demand G and VT instances** shows **20 vCPUs** (the raise you requested).
VPC routing	*VPC → Route tables*	podinsight-private-rt has 0.0.0.0/0 → **nat-0c1bf41f...**.Public RT has IGW.
Interface endpoints	*VPC → Endpoints*	One each for **S3, ECR API, ECR DKR, STS, Logs** → *Available*.
Batch SG rule	*VPC → Security groups → sg-0e769c... (podinsight-batch-sg)*	Inbound rule **HTTPS (443)** *source* **sg-0e769c...** (self-referencing).
ECR image	*ECR → Repositories → podcast-ingestor → latest*	Pushed **28 May 2025 22:52 (4 GB)**.
Launch template	*EC2 → Launch templates → pod-insight-dlami-largev3*	Uses **AMI ID shown**; Description notes "Whisper Large-v3 cached".

If any row differs, fix that one piece first (ping me if unsure).

---

1 Create / verify the

compute environment

1. **AWS Batch → Compute environments → Create**
   - Name: podinsight_gpu_smoke_ce (or select the one you already have)
   - **Provisioning model**: *EC2*
   - **Instances**
     - *Max vCPUs*: **4** (one g5.xlarge)
     - *Allowed instance types*: g5.xlarge only
     - *Allocation strategy*: **BEST_FIT_PROGRESSIVE**
   - **Launch template**
     - Choose **pod-insight-dlami-largev3**, Version 1
   - **Networking**

- VPC: **podinsight-vpc**
- Subnets: **subnet-0fc666bb52c5ec92b** & **subnet-05210fec5a5db91c8** (private-a / private-b)
- Security groups: **sg-0e769c5121c9fe9c7** (batch-sg) + **sg-053d014a5fc5e1ef6** (S3 endpoint)

- **Instance role**: ecsInstanceRole instance-profile
- **Service role**: leave as ***AWSServiceRoleForBatch***
- Click **Create** → wait until **Status = VALID**.

---

2 Create / verify the

job queue

1. **AWS Batch → Job queues → Create queue**
   - Name: podinsight_smoke_q
   - Priority: **1**
   - **Connected compute environments** → *Add* → select **podinsight_gpu_smoke_ce** (order = 1).
   - Leave everything else default → **Create** → status **VALID**.

---

3 Register the

job definition

(console version)

1. **AWS Batch → Job definitions → Create**
   - Name: podinsight_smoke_jd
   - **Execution environment**: *ECS*
   - **Platform capability**: *EC2*
   - **Container image**: 594331569440.dkr.ecr.eu-west-2.amazonaws.com/podinsight/transcribe:latest
   - **vCPUs** = **4 Memory (MiB)** = **16000 GPUs** = **1**
   - **Command** (click *Override command* → *Edit JSON* → paste exactly):

```
["--mode","transcribe",
 "--manifest","s3://pod-insights-manifests/tier1_smoke.csv",
 "--limit","1",
 "--model_size","large-v3"]
```

1.
   -
   - **Job role**: podinsight-task-role
   - **Log configuration**:
     - Log driver: **awslogs**
     - Group: /aws/batch/podinsight (“create new” if missing)
   - **Create** → note the new **Revision #** (e.g. :4).

---

4 Submit the smoke job

1. **AWS Batch → Jobs → Submit job**
   - Name: whisper_smoke
   - Job queue: **podinsight_smoke_q**

- Job definition: **podinsight_smoke_jd : (latest rev)**
- Leave overrides blank (definition already has the command).
- **Submit**.

---

5 Watch it run

- **Jobs** page – filter by *RUNNABLE* → should move to **STARTING** within ~30 s, **RUNNING** next.
- **EC2 → Instances** – a **g5.xlarge** appears (pod-insight-dlami-largev3 as the AMI) → *running*.
- **CloudWatch Logs → Log groups → /aws/batch/podinsight**
  - open latest stream, you'll see the container pull, nvidia-smi, then Whisper progress.
- Job finishes **SUCCEEDED**; CE scales back to **0 vCPU** automatically.

Time & cost: one g5.xlarge in eu-west-2 On-Demand ≈ US $ 1.212 / hour.

A single 60 MB MP3 with Whisper-Large typically runs **~9 min**, so ≈ $0.18.

---

6 Clean-up (keeps the artifacts, drops cost to zero)

1. **Terminate** the test job's log tab.
2. **Compute environment** → *Edit* → set **Max vCPUs = 0** → *Save*.

   (Brings ASG to zero, instance gone.)
3. Leave the queue & JD – reuse for later runs.

Where we stand on the sanity-check matrix

Section	✅ Done in guide	Notes
IAM roles / policies	yes	ecsInstanceRole + podinsight-task-role attached.
VPC NAT & SG	yes	private sub-nets → NAT; sg-self-443 rule confirmed.
Interface endpoints	yes	S3 / ECR API / ECR DKR / Logs / STS all *Available*.
Service quotas	yes	20 On-Demand G/VT vCPU; 90 Spot vCPU.
Launch template / AMI warmth	yes	Large-v3 cached; Medium still to add if desired.
CloudWatch Logs group	yes	/aws/batch/podinsight.
S3 buckets, Dynamo table	untouched (all eu-west-2).	

If any other artefact (Dockerfile, updated manifests, etc.) needs a look, feel free to re-upload; a few of the earlier screenshot files have expired.

Happy clicking – shout if any step behaves differently!

2 Job queue →

podinsight_smoke_q

Nothing to change – it already points to *podinsight_gpu_smoke_ce* and is **ENABLED**.

---

3 Job definition

Latest retained, active revision is **podinsight_smoke_jd:3** – that's the one you'll refer to.

(Console → *Job definitions* → click **podinsight_smoke_jd** → the top line shows **Revision 3** and *Platform = EC2*).

Sanity Check 1 June