**End-to-End Request Flow — No-NAT VPC, Cognito JWT, OpenSearch, Neptune**

*Auth & Query paths; VPC-only egress via VPC Endpoints; API Gateway JWT protection on /query*

Generated: 2025-08-31 18:33 UTC

## /auth/login (username + password)

[Browser / UI]  
 │ 1. POST /auth/login (email, password)  
 ▼  
[API Gateway /auth/login] ───────────────► [Lambda auth\_login (VPC)]  
 │ uses VPCe: cognito-idp  
 ▼  
 [Cognito User Pool]  
 │ issues IdToken  
 ▼  
 (IdToken returned to Browser)

## /query (JWT-protected)

[Browser / UI]  
 │ 2. POST /query (Authorization: Bearer <IdToken>, question)  
 ▼  
[API Gateway /query]  
 │ 3. JWT Authorizer (Cognito): verify issuer/audience → 401 if invalid  
 ▼  
[AWS\_PROXY integration]  
 ▼  
[Lambda query\_handler (VPC, private subnets, SG-λ)]  
 │ 4. Build SPARQL CQs (e.g., CQ1 trims, CQ2 sales)  
 │ 5. SPARQL -> Neptune (HTTPS :8182)  
 │ • if NEPTUNE\_IAM\_AUTH=true → SigV4 (service neptune-db)  
 │ • else direct TLS  
 │ 6. Optional snippet -> OpenSearch (index "docs")  
 │ • SigV4 (service es) via VPC endpoint to VPC domain  
 │ 7. Merge results: {trims, units, evidence}  
 ▼  
[API Gateway] (CORS headers)  
 ▼  
[Browser / UI] 8. Render JSON answer

## Networking / service access (no NAT)

VPC Private Subnets:  
 • Lambda SG -> Neptune SG (tcp 8182)  
 • Lambda SG -> OpenSearch SG (tcp 443)  
 • Lambda → AWS APIs via VPC Endpoints:  
 - Interface VPCe: cognito-idp, logs, sts  
 - Gateway VPCe: s3 (for bulk-loader data path)  
Observability:  
 • Lambda → CloudWatch Logs via VPCe: logs

## Compact sequence view

UI -> APIGW (/auth/login): email, password  
APIGW -> Lambda auth\_login: invoke  
Lambda auth\_login -> Cognito (via VPCe cognito-idp): InitiateAuth  
Cognito -> Lambda auth\_login: IdToken  
Lambda auth\_login -> APIGW: 200 { IdToken }  
APIGW -> UI: 200  
  
UI -> APIGW (/query): Authorization: Bearer IdToken, { question }  
APIGW (JWT Authorizer) -> Cognito keys (cached): validate token  
APIGW -> Lambda query\_handler: invoke (authorized)  
Lambda query\_handler -> Neptune (SPARQL): CQ1 (trims), CQ2 (sales) [SigV4 if IAM auth ON]  
Neptune -> Lambda query\_handler: results  
Lambda query\_handler -> OpenSearch (SigV4): /docs/\_search "Falcon X ACC"  
OpenSearch -> Lambda query\_handler: top snippet  
Lambda query\_handler -> APIGW: 200 { answer, trims, evidence[] }  
APIGW -> UI: 200 payload (CORS ok)

## Minimal request/response example

Request:

POST {api}/query  
Authorization: Bearer <IdToken>  
Content-Type: application/json  
  
{"question":"Which 2023 Falcon X trims have ACC and CA sales on 2023-03-15?"}

Response:

{  
 "answer": {"T101":120,"T102":55},  
 "trims": ["T101","T102"],  
 "evidence": [{"doc\_id":"RND-42","text":"Improved ACC sensor fusion for Falcon X Pro..."}]  
}

*Note: In production, restrict OpenSearch policies, attach a Cognito authorizer to /query (enforced), and enable Neptune IAM auth + SigV4.*