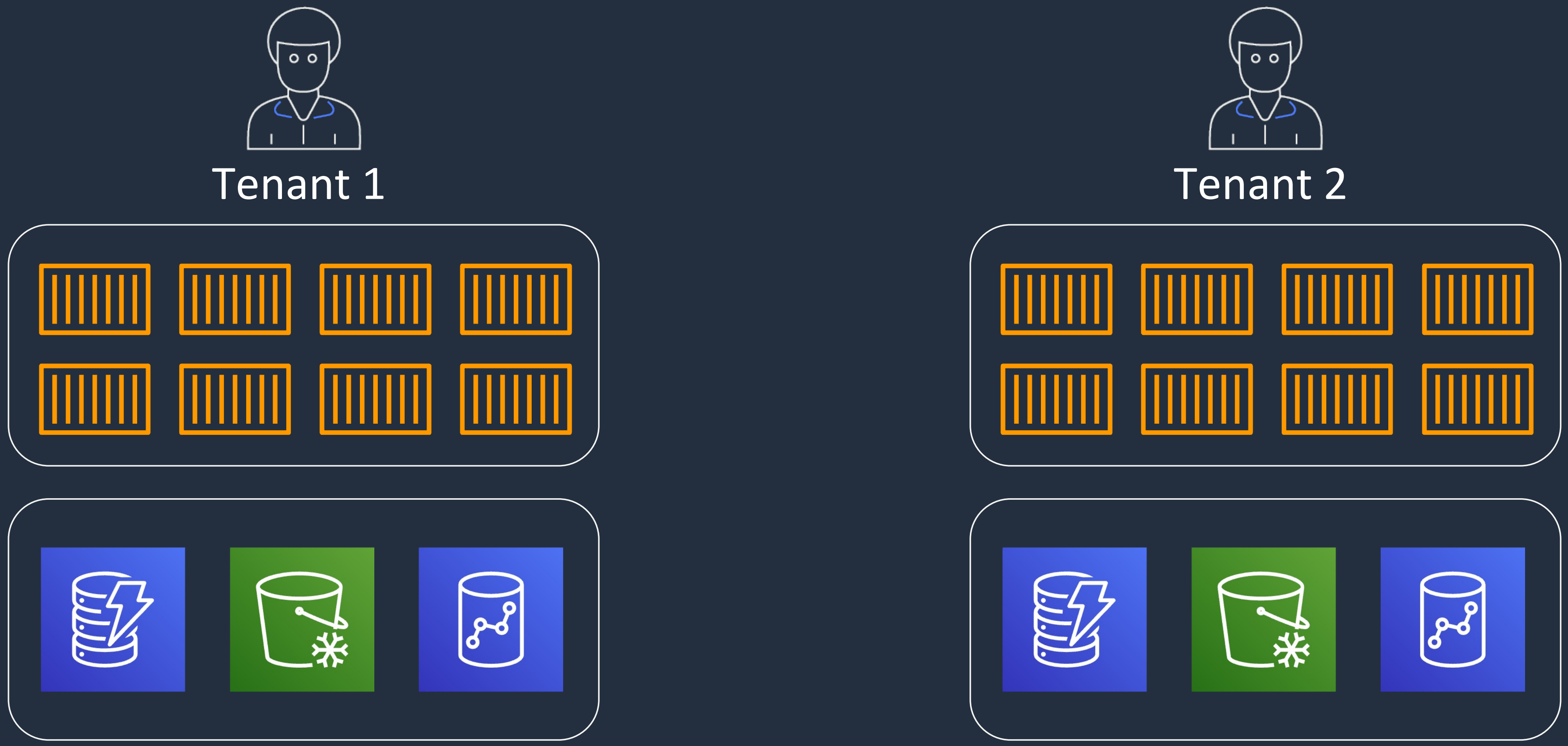
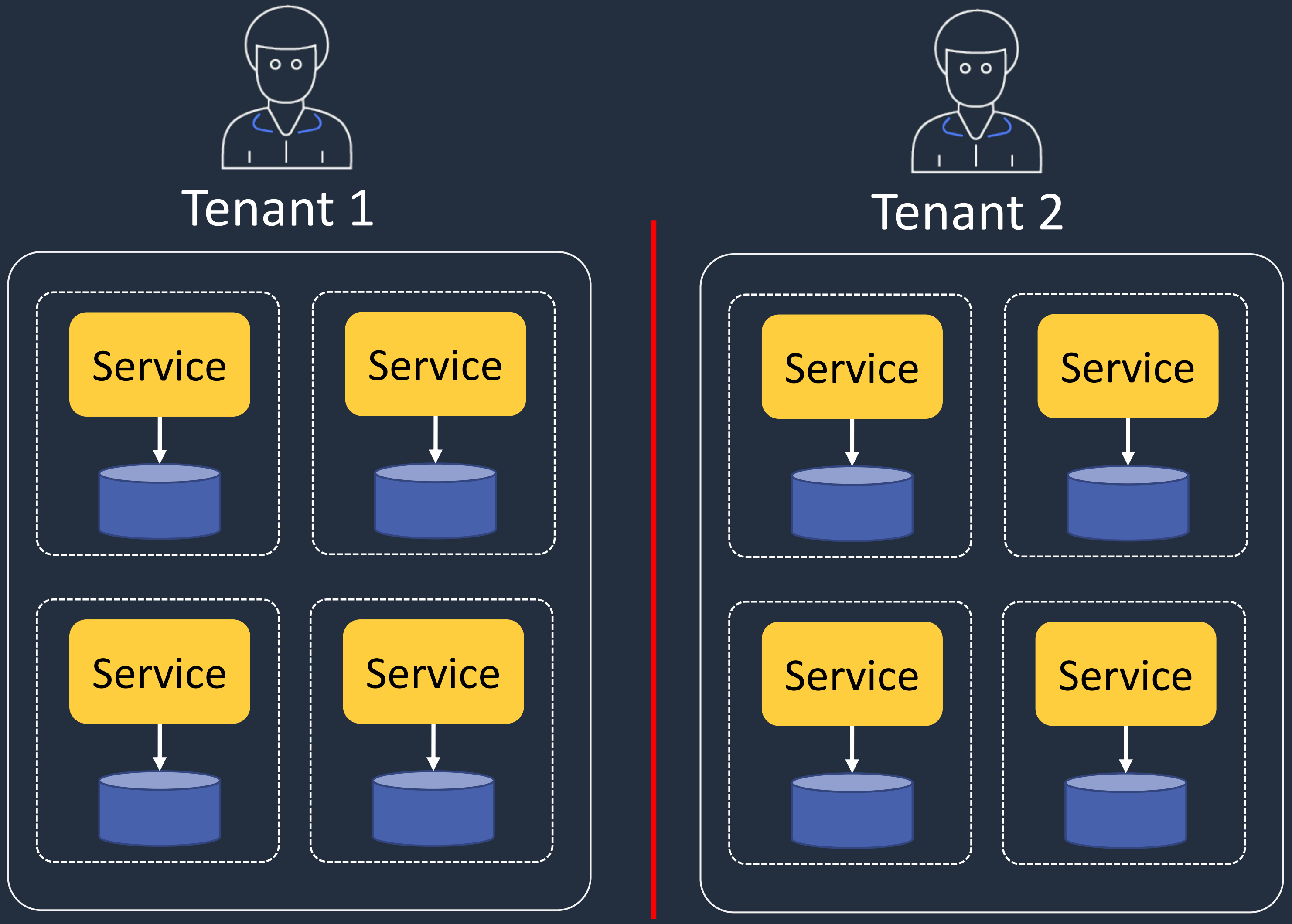
Build Serverless Multi-tenancy Service

Solution Architect , Sanghee Lee

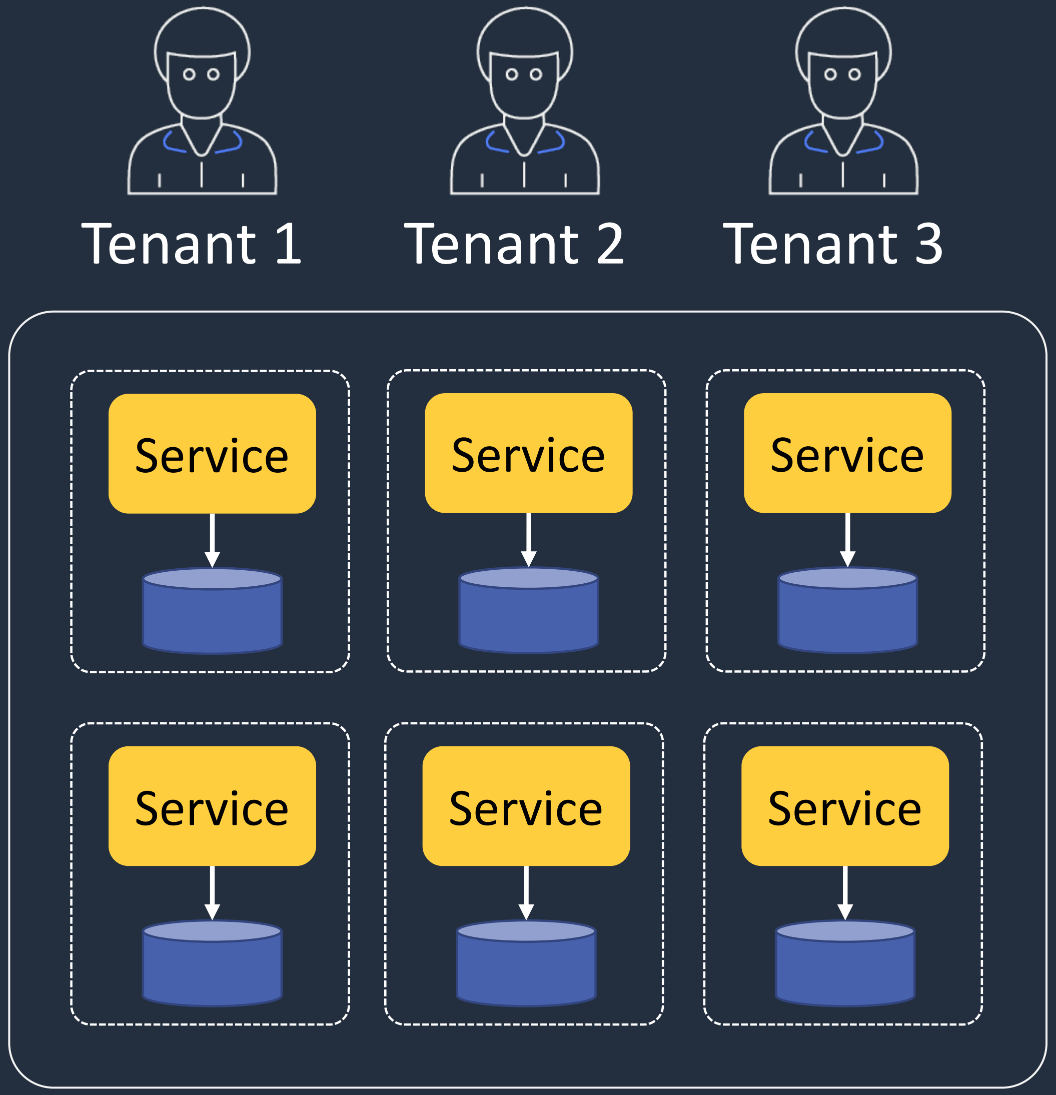
Tenant 분리는 왜 해야 하나요?



Multi-tenancy의 두가지 종류

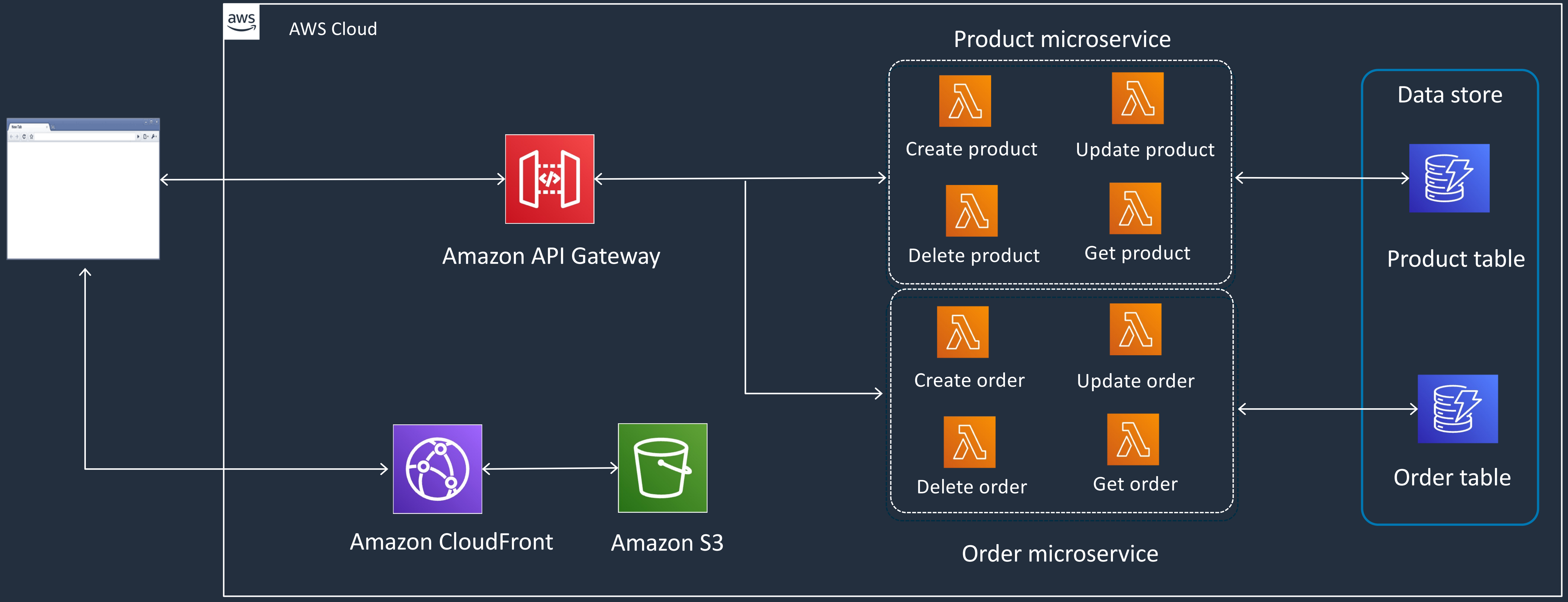


silo model

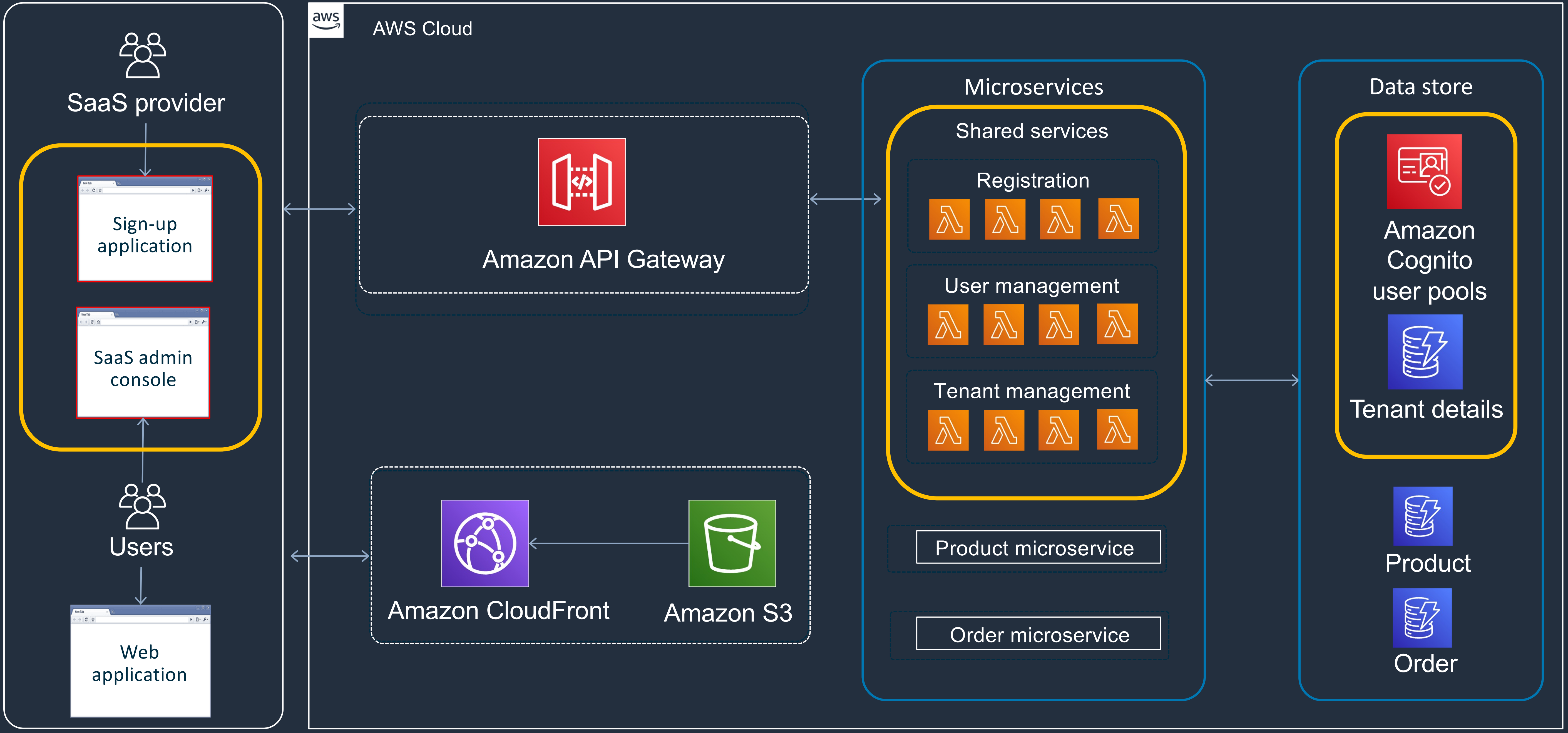


pool model

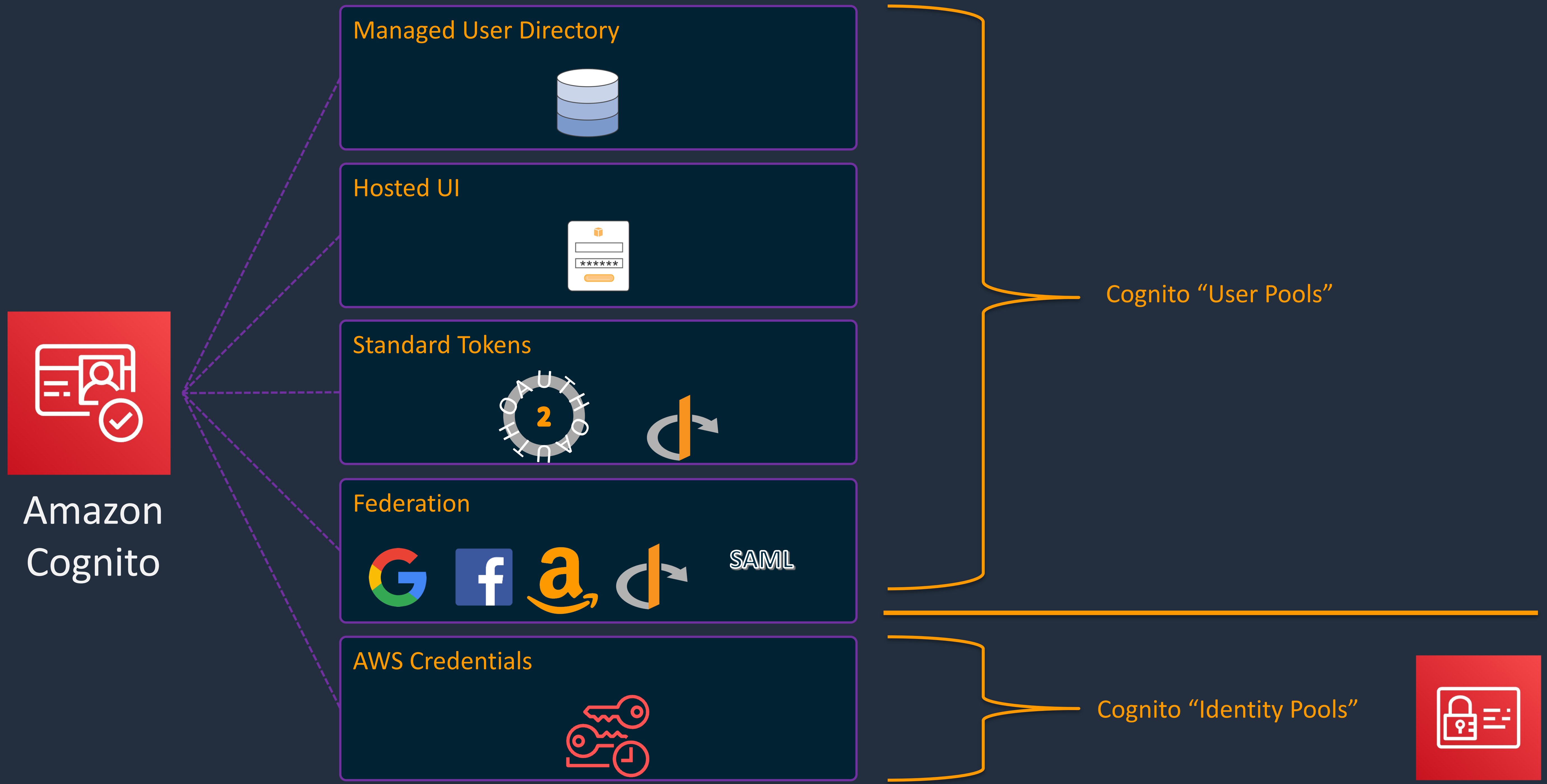
가장 일반적인 Serverless Architecture



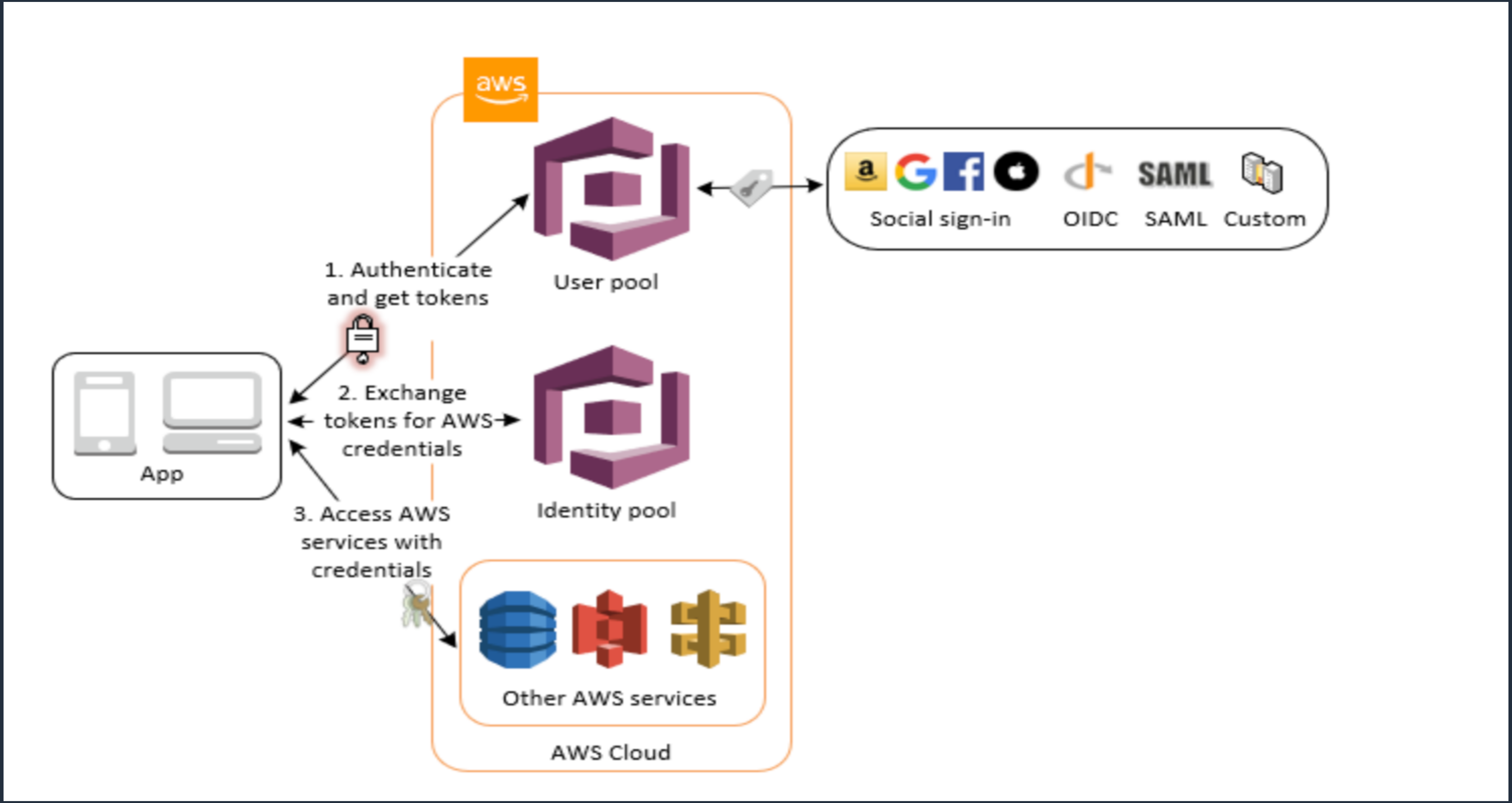
SaaS 를 한 숟가락 추가한다면



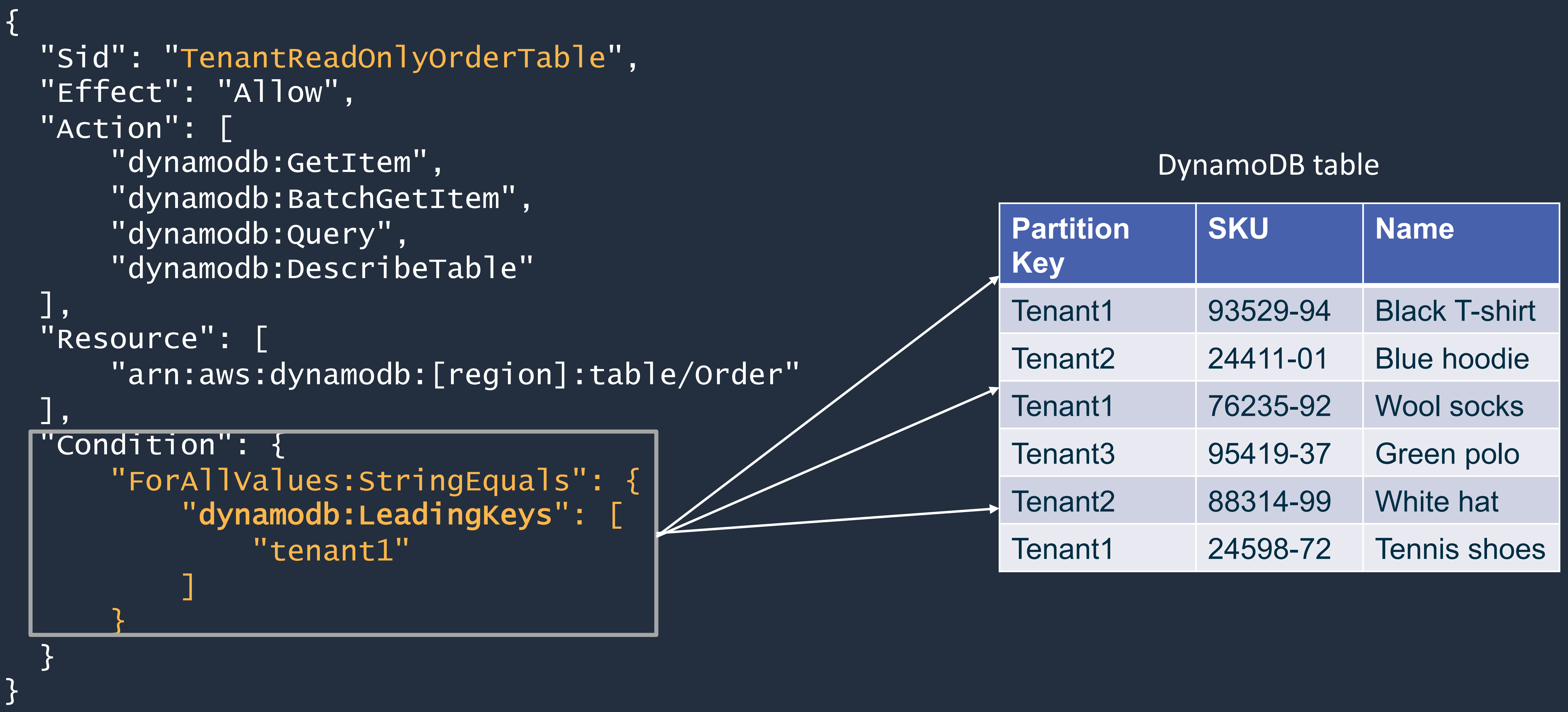
Cognito는 무엇일까요?



IAM 과 함께하는 Cognito


aws
User pool
Social sign-in
OIDC
SAML
Custom
1. Authenticate and get tokens
2. Exchange tokens for AWS credentials
Identity pool
App
3. Access AWS services with credentials
Other AWS services
AWS Cloud

DynamoDB 예제

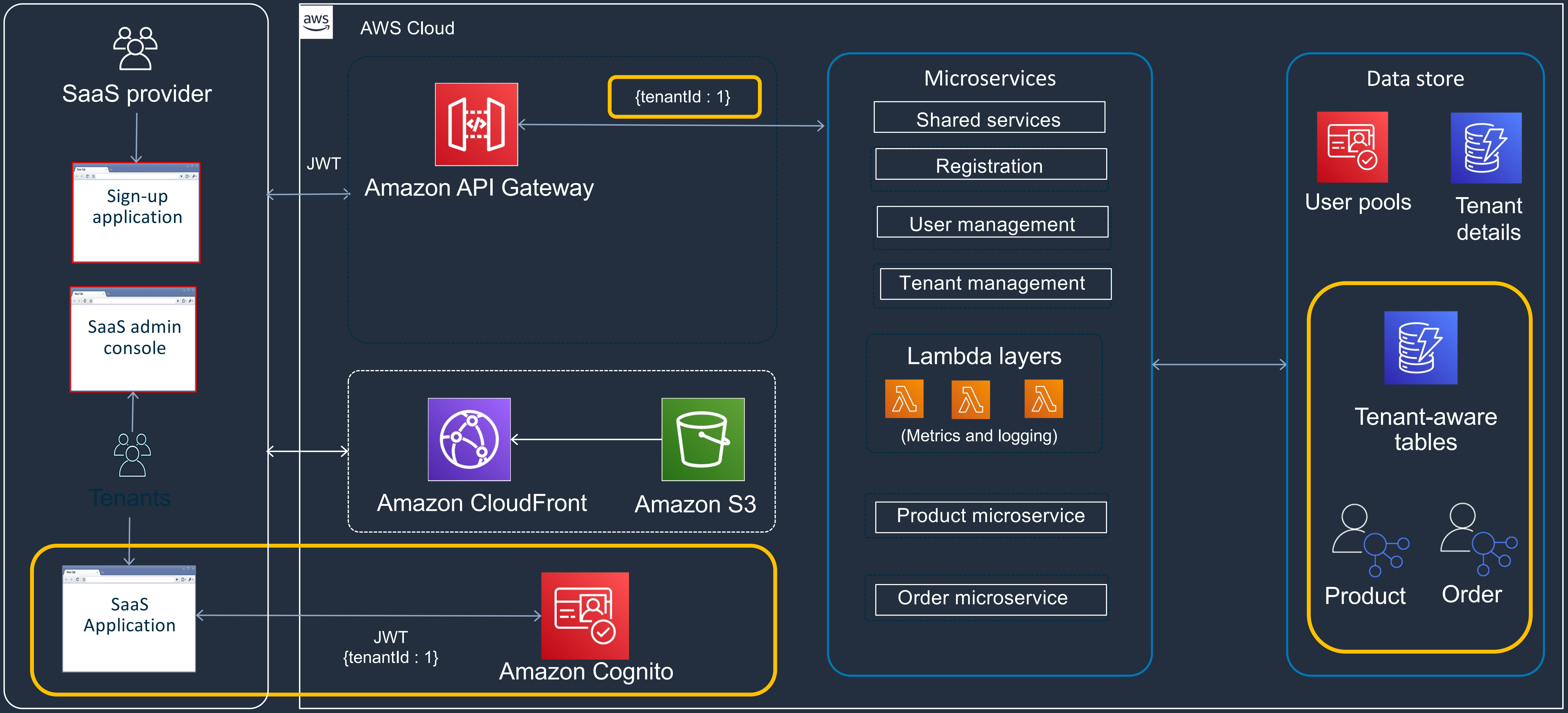


```
{
  "Sid": "TenantReadOnlyOrderTable",
  "Effect": "Allow",
  "Action": [
      "dynamodb:GetItem",
      "dynamodb:BatchGetItem",
      "dynamodb:Query",
      "dynamodb:DescribeTable"
  ],
  "Resource": [
      "arn:aws:dynamodb:[region]:table/Order"
  ],
  "Condition": {
      "ForAllValues:StringEquals": {
          "dynamodb:LeadingKeys": [
              "tenant1"
          ]
      }
  }
}
```

DynamoDB table

Partition Key	SKU	Name
Tenant1	93529-94	Black T-shirt
Tenant2	24411-01	Blue hoodie
Tenant1	76235-92	Wool socks
Tenant3	95419-37	Green polo
Tenant2	88314-99	White hat
Tenant1	24598-72	Tennis shoes

SaaS Serverless Architecture

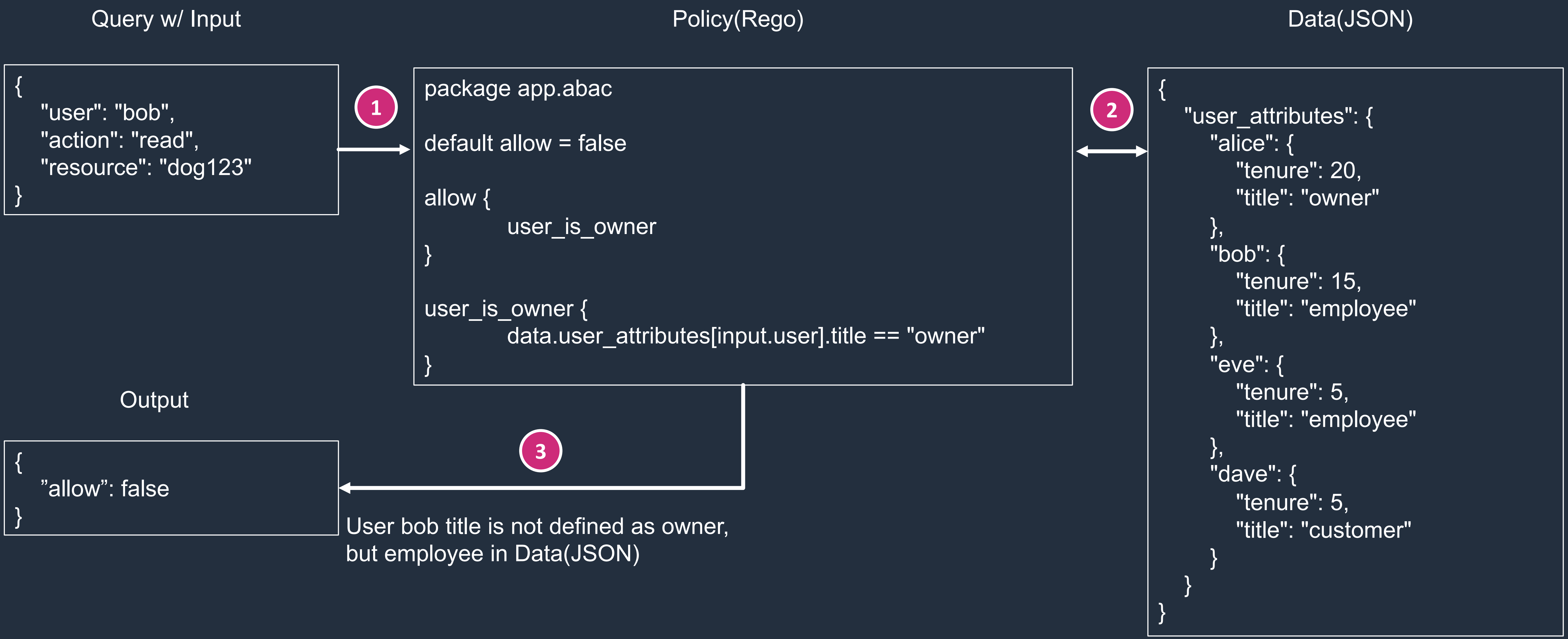


OPA(Open Policy Agent)

- AWS 외부 리소스도 통합 가능
- IAM 의 RBAC 방식이 아닌, PBAC 방식을 사용
- 코딩을 하듯, 로직을 넣어서 만들 수 있음

https://www.openpolicyagent.org/docs/latest/

의사결정은 어떻게 이루어질까요?



Query w/ Input

```
{
    "user": "bob",
    "action": "read",
    "resource": "dog123"
}
```

Policy(Rego)

```
package app.abac

default allow = false

allow {
        user_is_owner
}

user_is_owner {
        data.user_attributes[input.user].title == "owner"
}
```

Data(JSON)

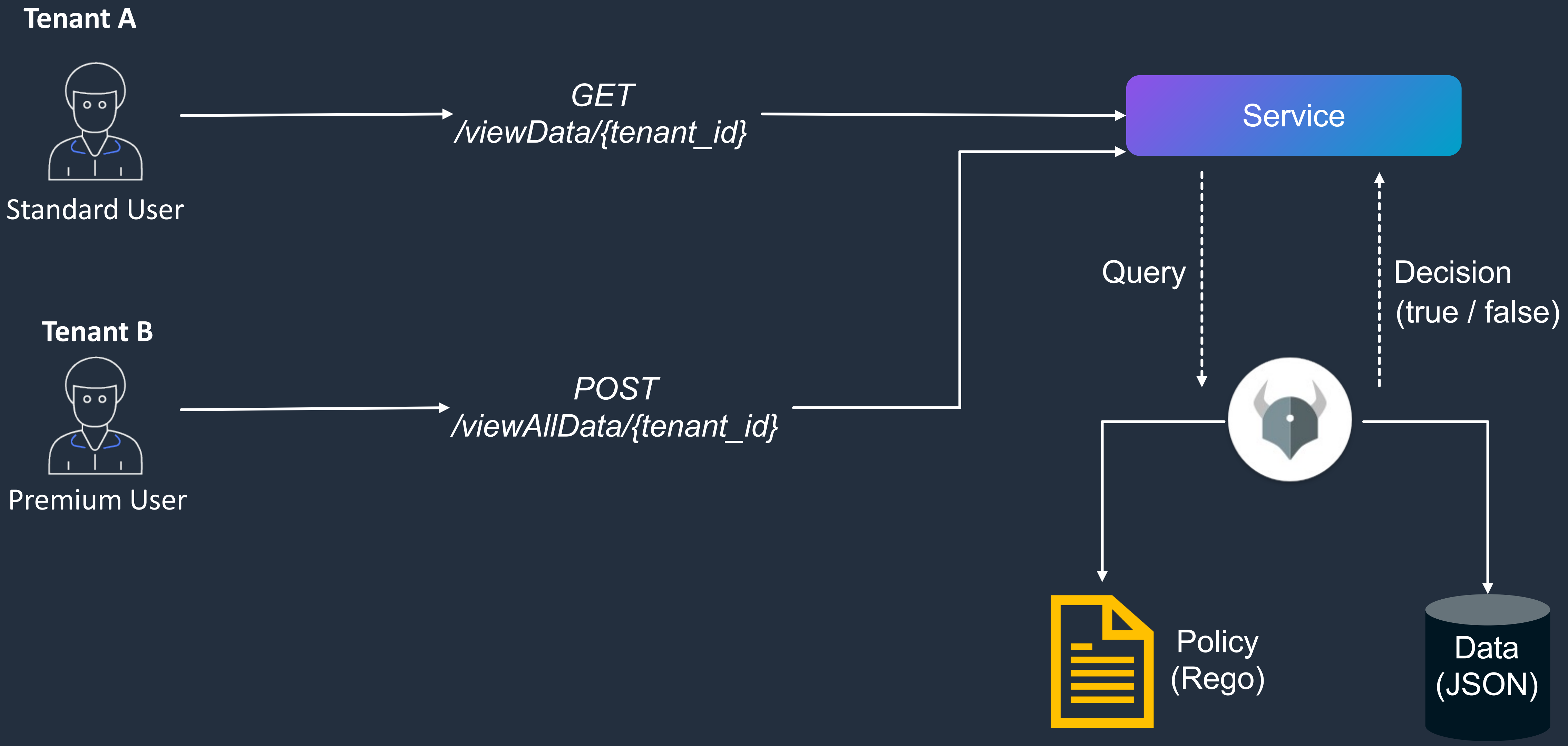
```
{
    "user_attributes": {
        "alice": {
            "tenure": 20,
            "title": "owner"
        },
        "bob": {
            "tenure": 15,
            "title": "employee"
        },
        "eve": {
            "tenure": 5,
            "title": "employee"
        },
        "dave": {
            "tenure": 5,
            "title": "customer"
        }
    }
}
```

Output

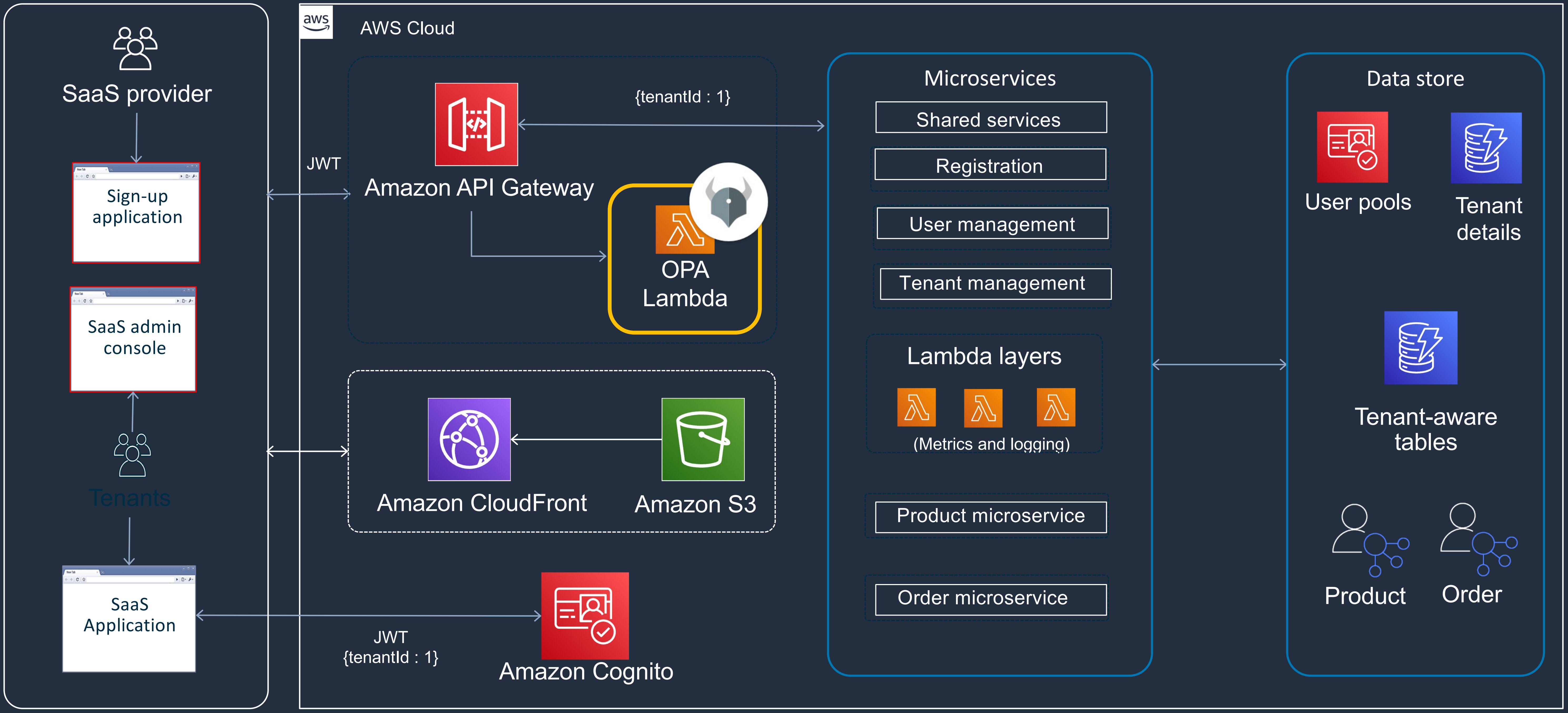
```
{
    ”allow”: false
}
```

User bob title is not defined as owner,
but employee in Data(JSON)

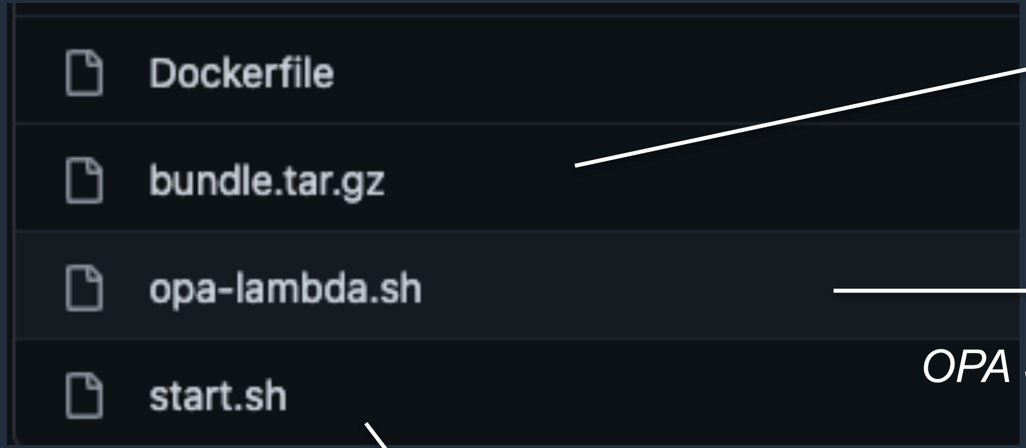
실제 서비스 흐름


Tenant A
Standard User
GET /viewData/{tenant_id}
Tenant B
Premium User
POST /viewAllData/{tenant_id}
Service
Query
Decision (true / false)
Policy (Rego)
Data (JSON)

Advanced SaaS Serverless Architecture



OPA 예제

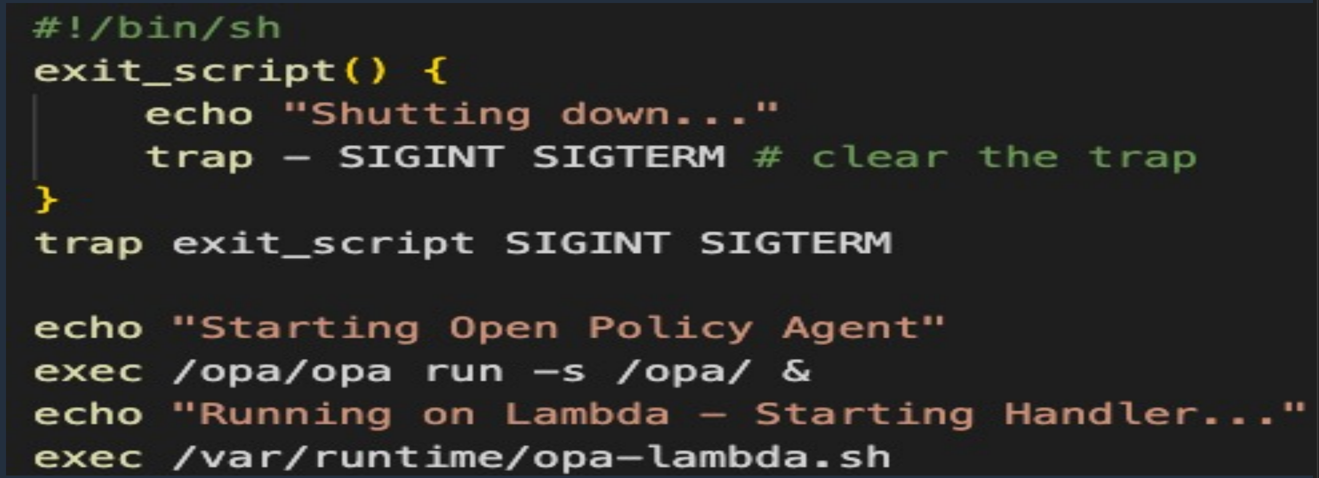


```
Dockerfile
bundle.tar.gz
opa-lambda.sh
start.sh
```

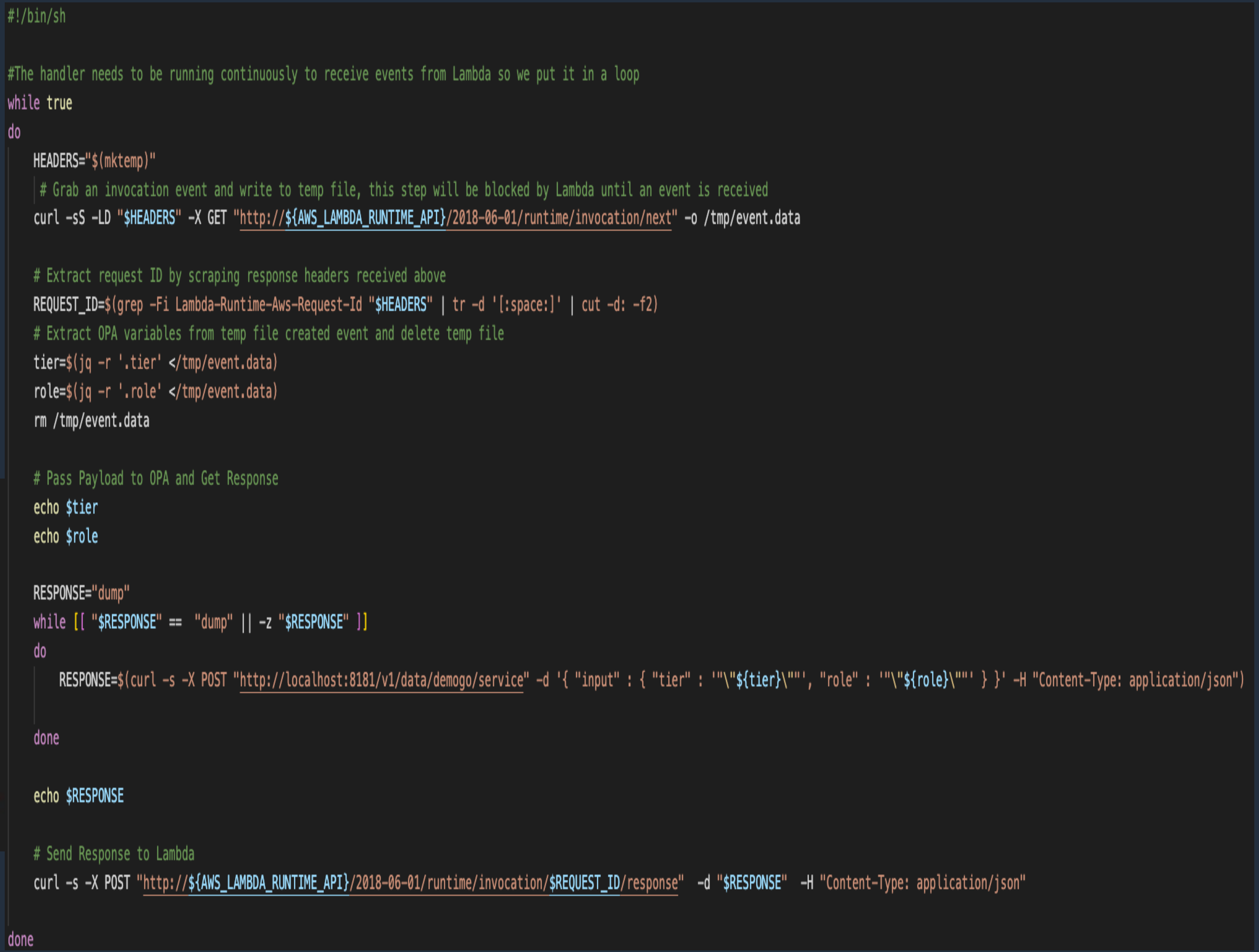
OPA Policy 와 Data 파일이 들어있음

OPA Service

OPA start command

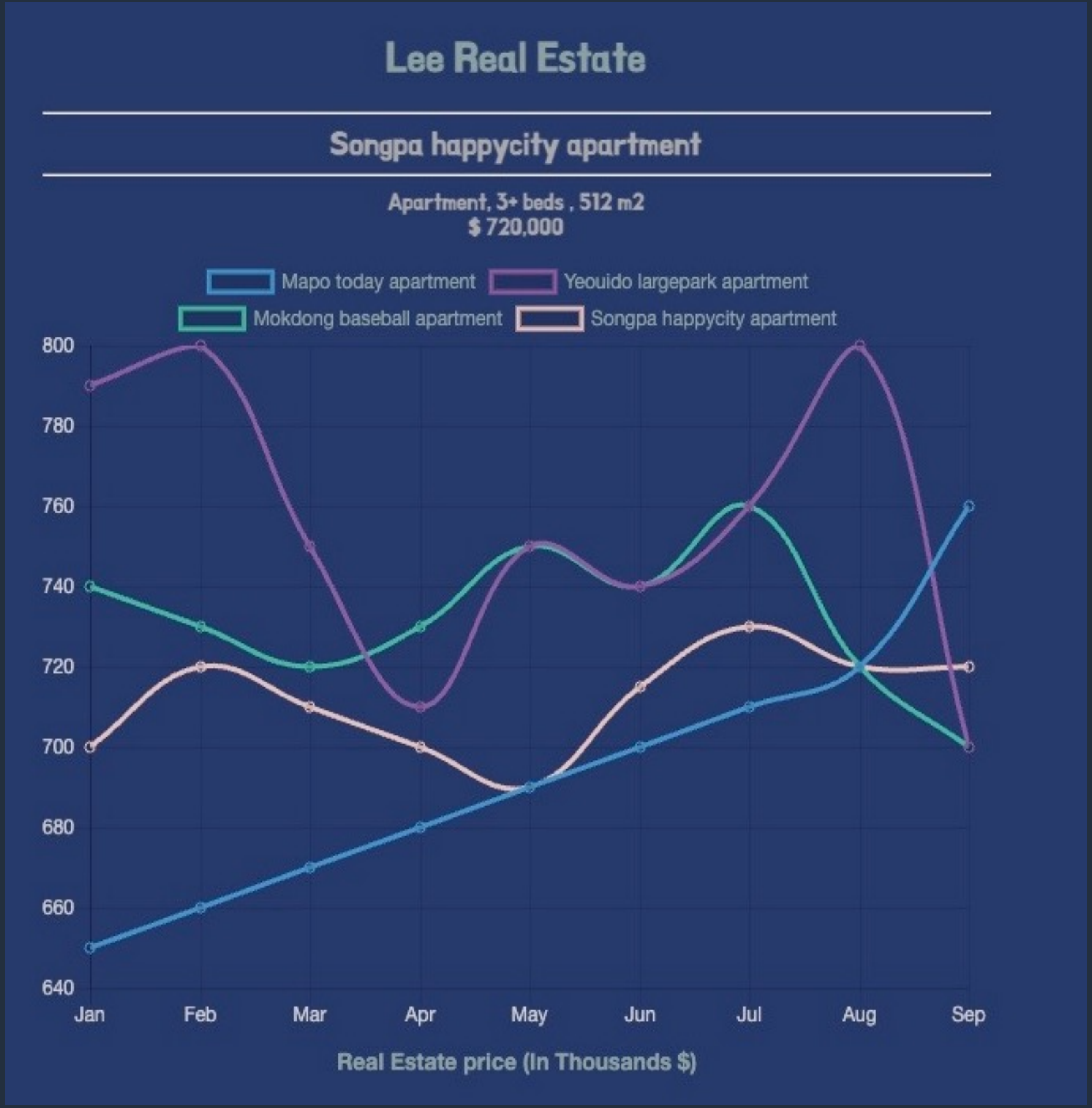


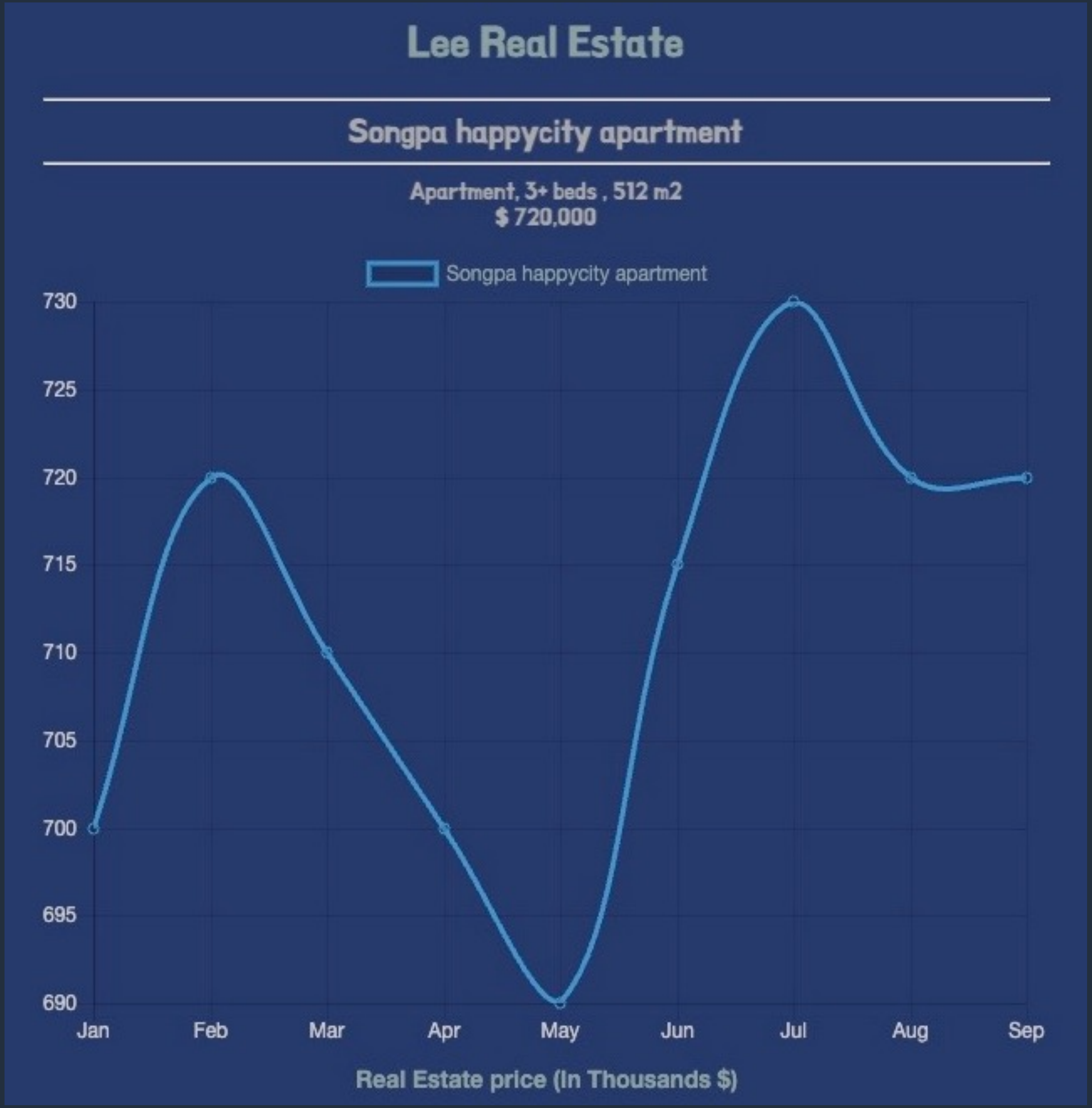
```
#!/bin/sh
exit_script() {
    echo "Shutting down..."
    trap - SIGINT SIGTERM # clear the trap
}
trap exit_script SIGINT SIGTERM

echo "Starting Open Policy Agent"
exec /opa/opa run -s /opa/ &
echo "Running on Lambda - Starting Handler..."
exec /var/runtime/opa-lambda.sh
```



```
#!/bin/sh

#The handler needs to be running continuously to receive events from Lambda so we put it in a loop
while true
do
  HEADERS="$(mktemp)"
   # Grab an invocation event and write to temp file, this step will be blocked by Lambda until an event is received
  curl -sS -LD "$HEADERS" -X GET "http://${AWS_LAMBDA_RUNTIME_API}/2018-06-01/runtime/invocation/next" -o /tmp/event.data

  # Extract request ID by scraping response headers received above
  REQUEST_ID=$(grep -Fi Lambda-Runtime-Aws-Request-Id "$HEADERS" | tr -d '[:space:]' | cut -d: -f2)
  # Extract OPA variables from temp file created event and delete temp file
  tier=$(jq -r '.tier' </tmp/event.data)
  role=$(jq -r '.role' </tmp/event.data)
  rm /tmp/event.data

  # Pass Payload to OPA and Get Response
  echo $tier
  echo $role

  RESPONSE="dump"
  while [[ "$RESPONSE" ==  "dump" || -z "$RESPONSE" ]]
  do
    RESPONSE=$(curl -s -X POST "http://localhost:8181/v1/data/demogo/service" -d '{ "input" : { "tier" : '"\"${tier}\""', "role" : '"\"${role}\""' } }' -H "Content-Type: application/json")

  done

  echo $RESPONSE

  # Send Response to Lambda
  curl -s -X POST "http://${AWS_LAMBDA_RUNTIME_API}/2018-06-01/runtime/invocation/$REQUEST_ID/response"  -d "$RESPONSE"  -H "Content-Type: application/json"

done
```

OPA 예제

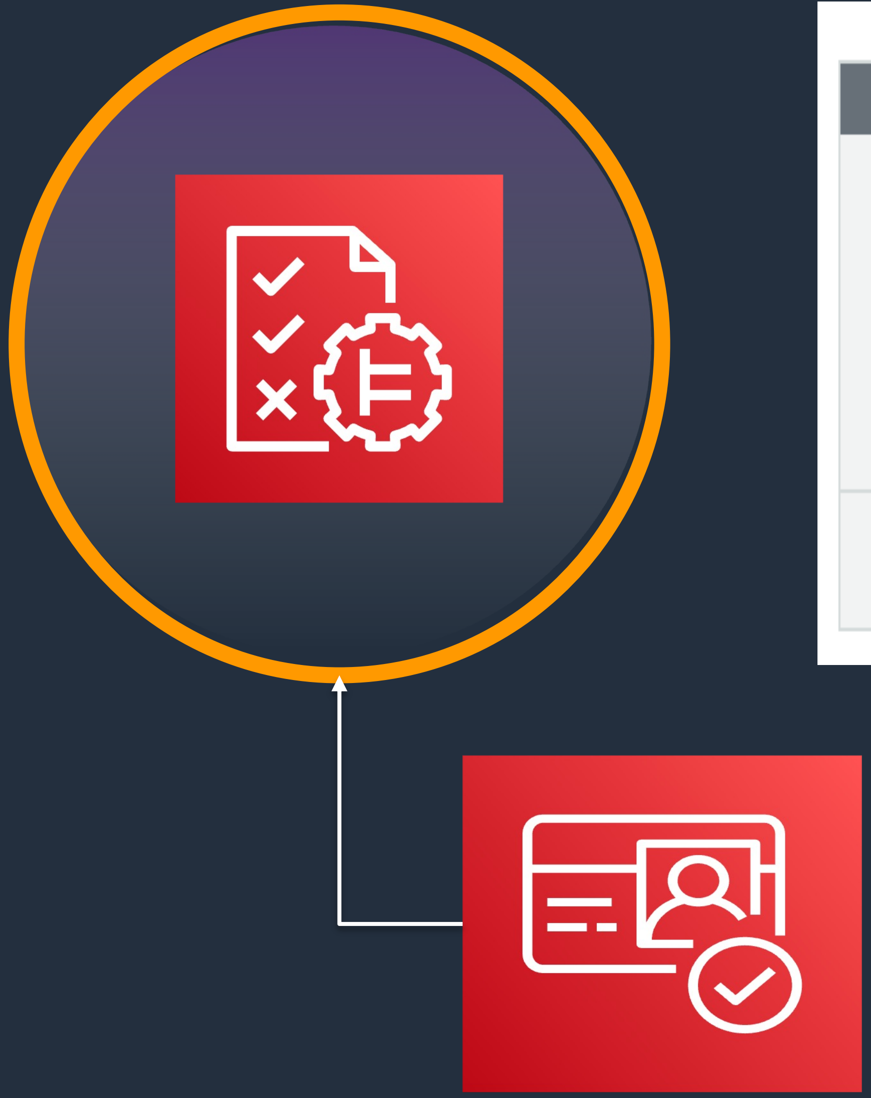

Lee Real Estate
Songpa happycity apartment
Apartment, 3+ beds , 512 m2
$ 720,000
Mapo today apartment
Yeouido largepark apartment
Mokdong baseball apartment
Songpa happycity apartment
800
780
760
740
720
700
680
660
640
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Real Estate price (In Thousands $)

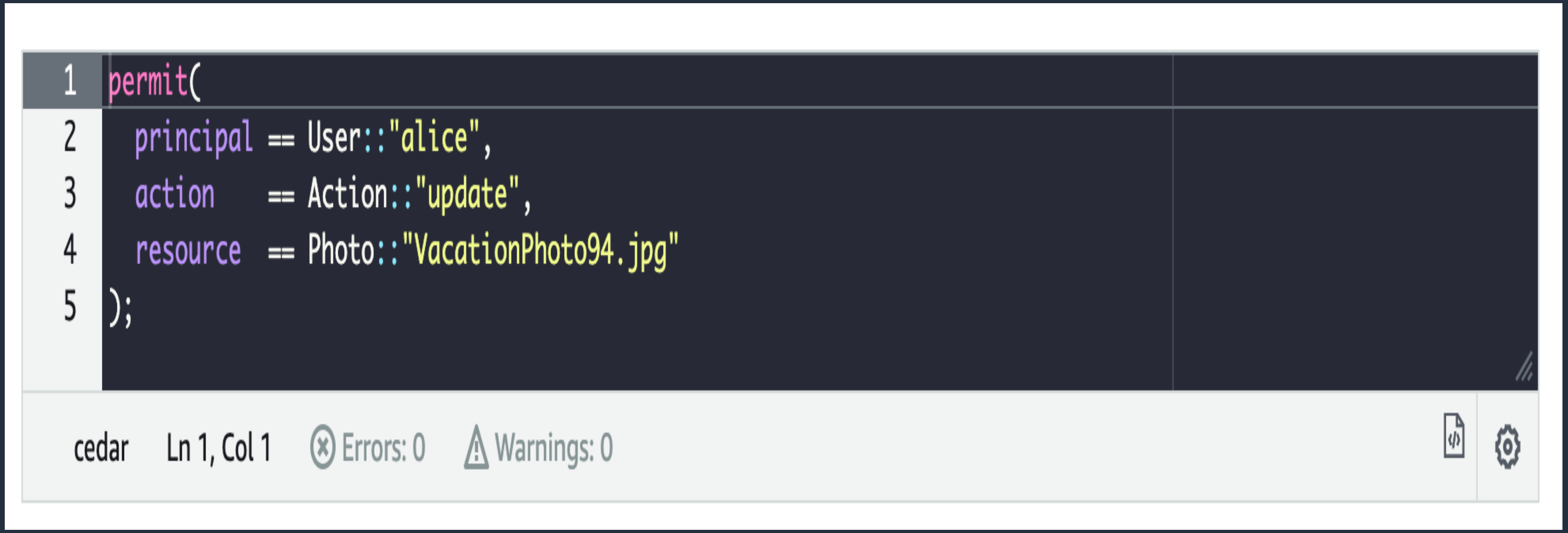

Lee Real Estate
Songpa happycity apartment
Apartment, 3+ beds , 512 m2
$ 720,000
Songpa happycity apartment
730
725
720
715
710
705
700
695
690
Jan
Feb
Mar
Apr
May
Jun
Jul
Aug
Sep
Real Estate price (In Thousands $)

Bonus track

Introducing Amazon Verified Permissions

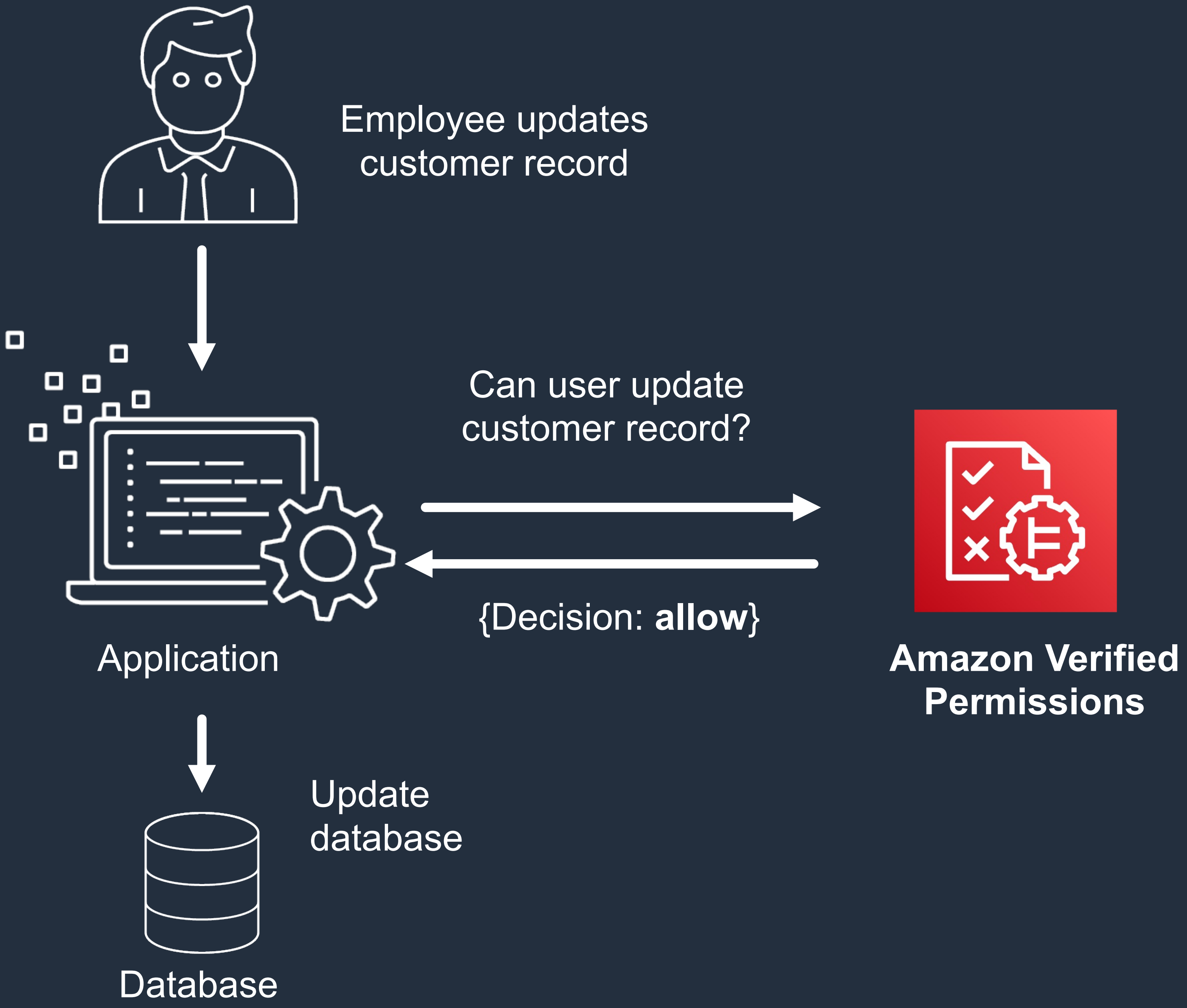
FINE-GRAINED PERMISSIONS AND AUTHORIZATION FOR YOUR CUSTOM APPLICATIONS





```
permit(
  principal == User::"alice",
  action    == Action::"update",
  resource  == Photo::"VacationPhoto94.jpg"
);
```

동작 방식


Employee updates customer record
Can user update customer record?
{Decision: **allow**}
Application
Amazon Verified Permissions
Update database
Database

감사합니다!