Step 1: Clone the Repository

```bash
git clone https://github.com/ndeepakprasanth/dapr-eks-pubsub-ready.git
cd dapr-eks-pubsub-ready
```

Step 2: Set execute permissions

```
chmod +x oneclick.sh
chmod +x test.sh
chmod +x scripts/*.sh
```

Step 3: One-Click deployment

```
# Replace with your AWS Account ID
ACCOUNT_ID=946248011760 ./oneclick.sh
```

Step 4: Verify deployment

```
# Check pods (should show 2/2 Ready)
kubectl -n dapr-apps get pods -o wide
# Check services
kubectl -n dapr-apps get svc
# Check Dapr components
kubectl -n dapr-apps get components
```

Step 5: Test Pub/Sub functionality

```
./test.sh
```

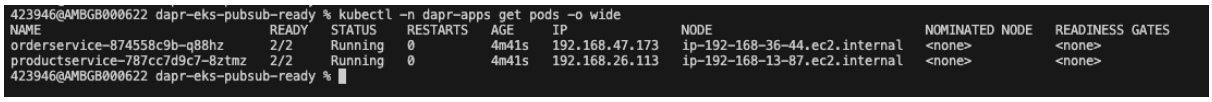
Step 6: View ECR repositories

```
aws ecr describe-repositories --region us-east-1
```

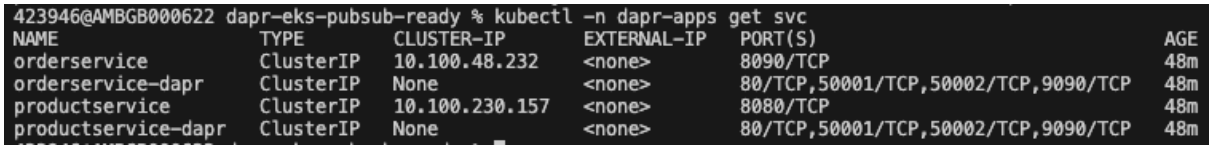
Step 7: Clean up when completed

```
./destroy.sh
```

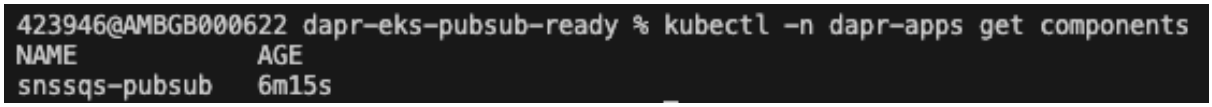
1. Running pods:



2. Services:



3. Dapr compoents



4. Test Pub/Sub

```
423946@AMBGB000622 dapr-eks-pubsub-ready % kubectl -n dapr-apps run test-curl --rm -i --restart=Never --image=curlimages/curl:8.10.1 -- curl -X POST http://productservice:8
080/publish -H 'Content-Type: application/json' -d '{"orderId": 999, "item":"laptop", "price": 199.99}'


If you don't see a command prompt, try pressing enter.
warning: couldn't attach to pod/test-curl, falling back to streaming logs: unable to upgrade connection: container test-curl not found in pod test-curl_dapr-apps
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1400  100  1350  100    50  34119   1263 --:--:-- --:--:-- --:--:-- --:--:-- 35897
{"ok":false,"error":"Request failed with status code 500","details":{"errorCode":"ERR_PUBSUB_PUBLISH_MESSAGE","message":"error when publishing to topic orders in pubsub sns
sqs-pubsub: error publishing to topic: orders with topic ARN : operation error SNS: Publish, https response error StatusCode: 400, RequestID: 5d77f1b3-72a7-5cb5-82b6-90704f
1f5006, InvalidParameter: Invalid parameter: TopicArn Reason: An ARN must have at least 6 elements, not 1","details":[{"@type":"type.googleapis.com/google.rpc.ResourceInfo"
,"description":"error when publishing to topic orders in pubsub snssqs-pubsub: error publishing to topic: orders with topic ARN : operation error SNS: Publish, https respon
se error StatusCode: 400, RequestID: 5d77f1b3-72a7-5cb5-82b6-90704f1f5006, InvalidParameter: Invalid parameter: TopicArn Reason: An ARN must have at least 6 elements, not 1
","owner":"","resource_name":"snssqs-pubsub","resource_type":"pubsub"},{"@type":"type.googleapis.com/google.rpc.ErrorInfo","domain":"dapr.io","metadata":{"error":"error pub
lishing to topic: orders with topic ARN : operation error SNS: Publish, https response error StatusCode: 400, RequestID: 5d77f1b3-72a7-5cb5-82b6-90704f1f5006, InvalidParame
ter: Invalid parameter: TopicArn Reason: An ARN must have at least 6 elements, not 1","topic":"orders"},"reason":"DAPR_PUBSUB_PUBLISH_MESSAGE"}]}}pod "test-curl" deleted
423946@AMBGB000622 dapr-eks-pubsub-ready %
```

5. ECR repositories with pushed images

```
423946@AMBGB000622 dapr-eks-pubsub-ready % aws ecr describe-repositories --region us-east-1 --profile Deepak
{
    "repositories": [
        {
            "repositoryArn": "arn:aws:ecr:us-east-1:946248011760:repository/orderservice",
            "registryId": "946248011760",
            "repositoryName": "orderservice",
            "repositoryUri": "946248011760.dkr.ecr.us-east-1.amazonaws.com/orderservice",
            "createdAt": "2025-12-22T11:41:09.817000+00:00",
            "imageTagMutability": "MUTABLE",
            "imageScanningConfiguration": {
                "scanOnPush": false
            },
            "encryptionConfiguration": {
                "encryptionType": "AES256"
            }
        },
        {
            "repositoryArn": "arn:aws:ecr:us-east-1:946248011760:repository/productservice",
            "registryId": "946248011760",
            "repositoryName": "productservice",
            "repositoryUri": "946248011760.dkr.ecr.us-east-1.amazonaws.com/productservice",
            "createdAt": "2025-12-22T11:41:06.238000+00:00",
            "imageTagMutability": "MUTABLE",
            "imageScanningConfiguration": {
                "scanOnPush": false
            },
            "encryptionConfiguration": {
                "encryptionType": "AES256"
            }
        }
    ]
}
423946@AMBGB000622 dapr-eks-pubsub-ready %
```

6. Bedrock analysis

```
==> Bedrock GenAI Analysis Prompts for Assignment

📋 Copy these prompts to Amazon Bedrock Console:

1️⃣ TELEMETRY ANALYSIS PROMPT:
----------------------------------------
Given the attached Node.js microservices (ProductService and OrderService) running on EKS with Dapr sidecars for pub/sub messaging via AWS SNS/SQS, suggest missing telemetr
y points:

- Custom application logs for business events
- Distributed trace spans and attributes
- Metrics for publish latency, delivery success/failure rates
- Dead letter queue monitoring
- Dapr sidecar performance metrics

Output OpenTelemetry configuration snippets for Node.js applications and Kubernetes manifests.

2️⃣ RESILIENCY ANALYSIS PROMPT:
----------------------------------------
Recommend retry, backoff, and circuit-breaker policies for event-driven architecture using:
- Dapr pub/sub with AWS SNS/SQS
- EKS microservices with potential network failures
- Application-level idempotency handling

Include:
- Dapr resiliency policies YAML configuration
- Sample deduplication logic for Node.js
- Error handling patterns for pub/sub failures

3️⃣ SECURITY & PERFORMANCE ANALYSIS PROMPT:
----------------------------------------
Analyze the following artifacts for security and performance issues:
- Dockerfiles (Node.js Alpine-based containers)
- Kubernetes Deployment YAML (ProductService, OrderService)
- Dapr SNS/SQS component configuration

Flag issues like:
- Root user usage in containers
- Missing resource requests/limits
- Absent liveness/readiness probes
- Message concurrency settings
- Security contexts and capabilities

Provide specific fixes and improved configurations.

4️⃣ SCALING PATTERNS PROMPT:
----------------------------------------
For SQS + Dapr pub/sub on Amazon EKS, propose:
- Horizontal Pod Autoscaling based on SQS queue length using KEDA
- Dapr bulkSubscribe settings for high throughput
- Resource optimization for cost-effective scaling
- Monitoring and alerting for scaling events

Include complete Kubernetes manifests for KEDA ScaledObject and HPA configurations.
```

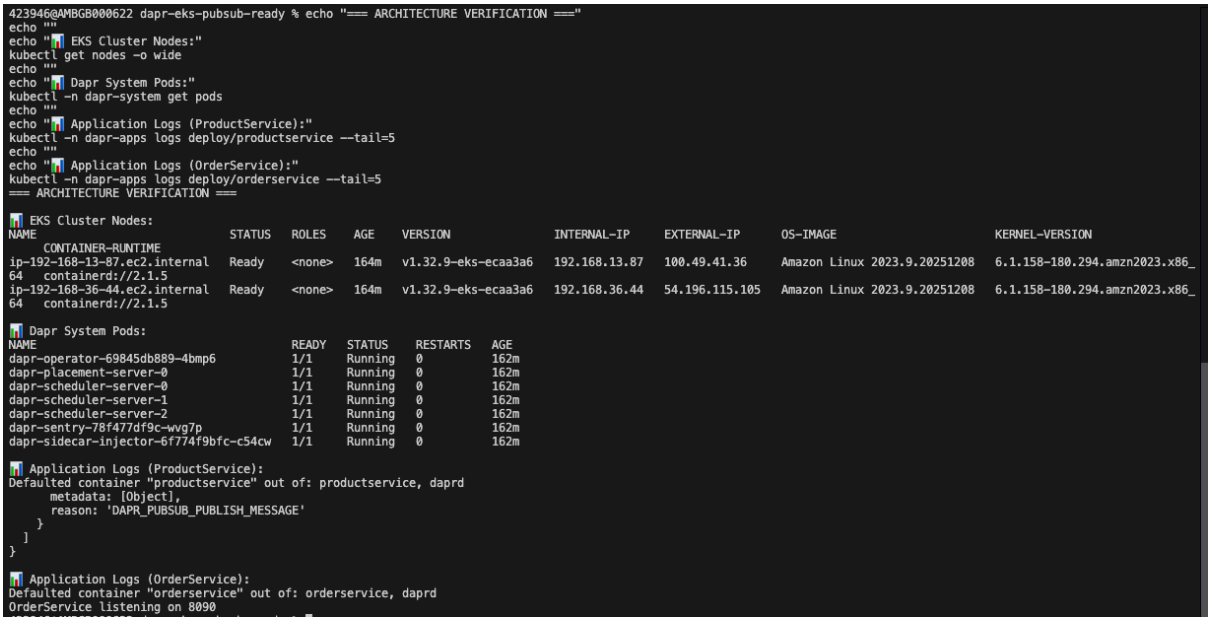
7. Stress test output

```
423946@AMBGB000622 dapr-eks-pubsub-ready % echo "=== STRESS TEST DEMONSTRATION ==="
echo "Sending multiple concurrent requests..."
for i in {1..5}; do
  kubectl -n dapr-apps run load-test-$i --restart=Never --image=curlimages/curl:8.10.1 --command -- curl -X POST http://productservice:8080/publish -H 'Content-Type: applic
ation/json' -d "{\"orderId\": $i, \"item\":\"load-test-$i\", \"price\": $((RANDOM % 1000))}" &
done
echo "Waiting for completion..."
sleep 8
echo ""
echo "=== STRESS TEST RESULTS ==="
kubectl -n dapr-apps get pods | grep load-test
echo ""
echo "=== CLEANUP ==="
kubectl -n dapr-apps delete pods -l run --field-selector=status.phase=Succeeded
=== STRESS TEST DEMONSTRATION ===
Sending multiple concurrent requests...
[2] 7386
[3] 7387
[4] 7388
[5] 7389
[6] 7390
Waiting for completion...
pod/load-test-5 created
pod/load-test-4 created
[6]  + done       kubectl -n dapr-apps run load-test-$i --restart=Never  --command -- curl -X
[5]  + done       kubectl -n dapr-apps run load-test-$i --restart=Never  --command -- curl -X
pod/load-test-1 created
[2]    done       kubectl -n dapr-apps run load-test-$i --restart=Never  --command -- curl -X
pod/load-test-3 created
[4]  + done       kubectl -n dapr-apps run load-test-$i --restart=Never  --command -- curl -X
pod/load-test-2 created
[3]  + done       kubectl -n dapr-apps run load-test-$i --restart=Never  --command -- curl -X

=== STRESS TEST RESULTS ===
load-test-1                       0/1     Completed   0          8s
load-test-2                       0/1     Completed   0          8s
load-test-3                       0/1     Completed   0          8s
load-test-4                       0/1     Completed   0          8s
load-test-5                       0/1     Completed   0          8s

=== CLEANUP ===
pod "load-test-1" deleted
pod "load-test-2" deleted
pod "load-test-3" deleted
pod "load-test-4" deleted
pod "load-test-5" deleted
```

8. Logs screenshot:



9. Bedrock analysis