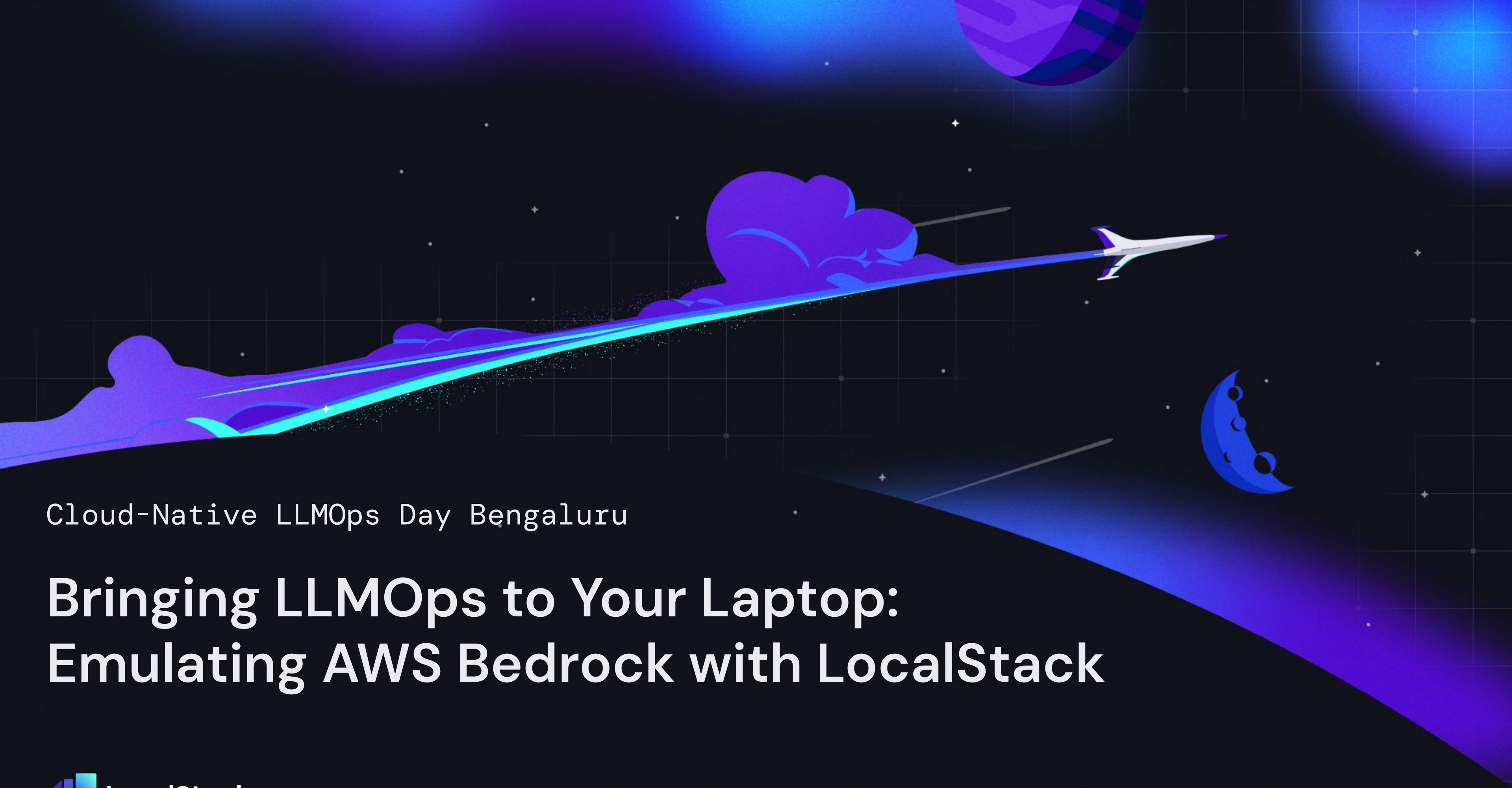


Cloud-Native LLMOps Day Bengaluru

Bringing LLMOps to Your Laptop: Emulating AWS Bedrock with LocalStack

LocalStack

Run Locally, Deploy Globally

www.localstack.cloud

Join at slido.com

Use Code: #4179796

Agenda

- LLMOps on Amazon Web Services
- What is Amazon Bedrock?
- Challenges in Cloud LLM Dev & Testing
- Introduction to LocalStack:
    - General background & overview
    - Quickstart with Bedrock emulator
- Demo: Run a local GenAI conversation summarization workflow
- Quiz & QnA

What is LLMOps?

- Set of practices and processes designed specifically for deploying & managing app using Large Language Models (LLMOps).
- LLMOps leverages LLMs to build apps, APIs, and tools for human-like text, image, video, and audio generation in business applications.
- LLMOps automates and enhances GenAI workflows, with a focus on security, transparency and optionality to fully utilize the LLM ecosystem.

LLMOps on AWS

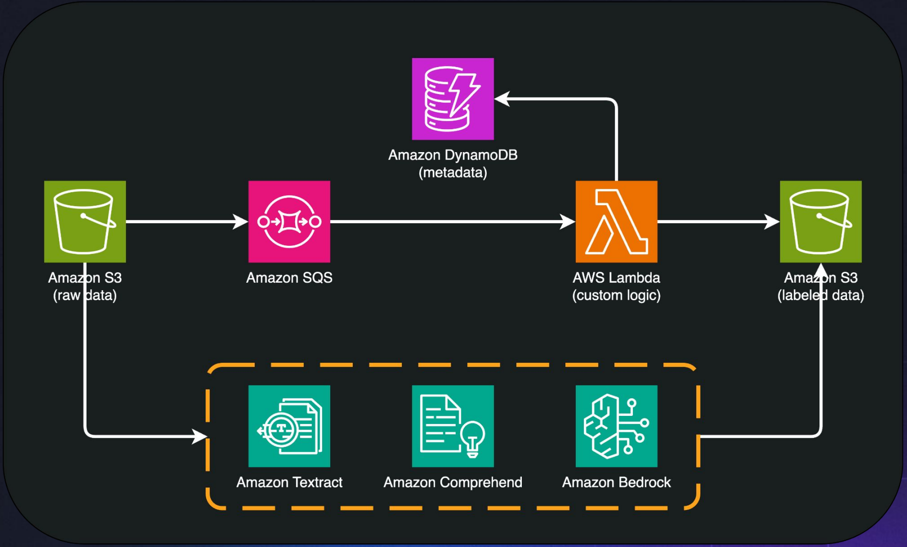
- AWS has one of the widest ranges of large language models available through its Bedrock platform.
  - → Model optionality is key to ensure you point the right model, at the right problem.
- Your data stays private and secure, never leaving your managed perimeter on AWS or being used for LLM training without your knowledge.
- AWS has the scale, capacity & resilience with an API-centric design, model customization for your needs, and enterprise security & compliance features.

Step #1: Data Preparation

Improving the quality of the data, requires a bunch of operations, such as:

- Text cleaning and normalization
- Data deduplication
- Data augmentation
- Data filtering

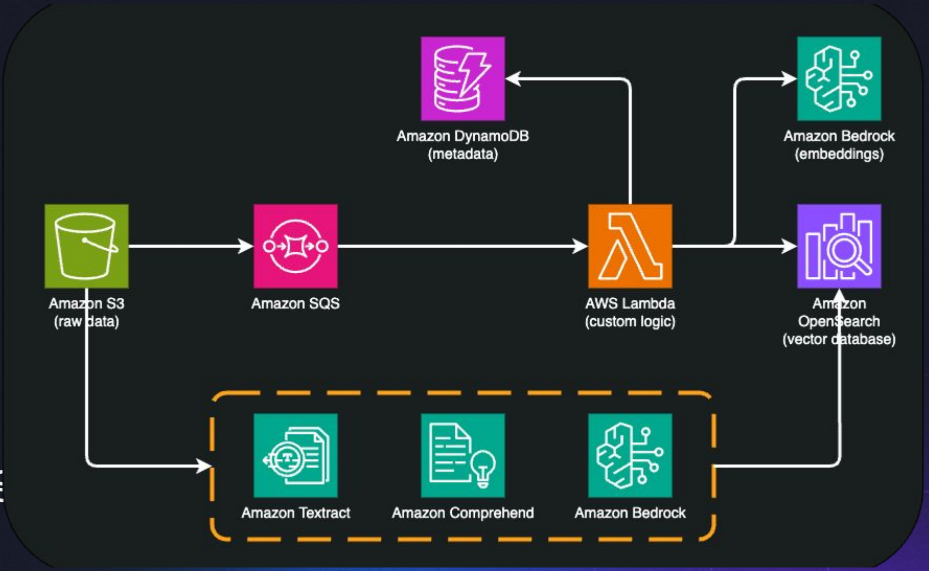
AWS provides many services that help you filter, clean, and augment your data.



Step #2: Data Ingestion

Data ingestion usually involves sending data to some kind of storage. Common examples include:

- Classic SQL/NoSQL databases.
- Vector databases for semantic searches
- Cold storage for less frequently used data
- Metadata database for tracking & tagging

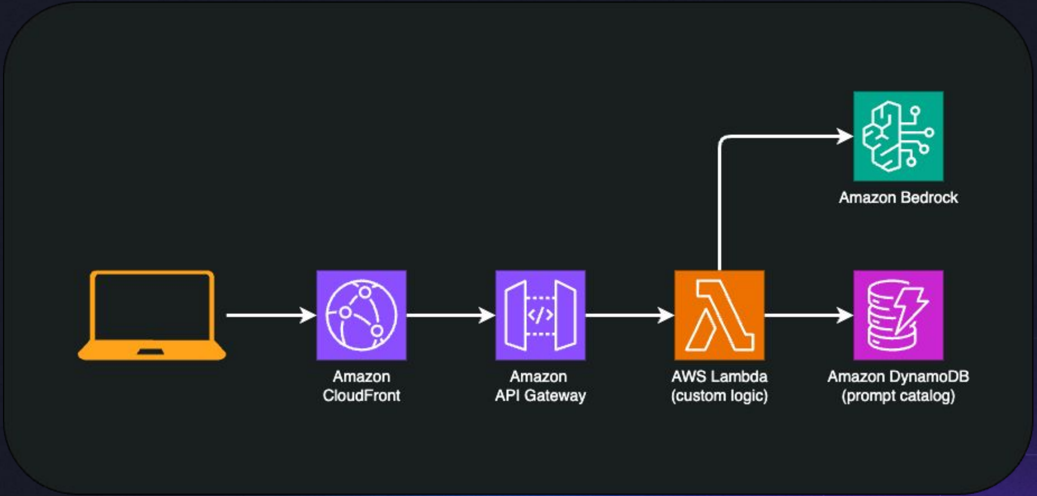

Amazon DynamoDB (metadata)
Amazon Bedrock (embeddings)
Amazon S3 (raw data)
Amazon SQS
AWS Lambda (custom logic)
Amazon OpenSearch (vector database)
Amazon Textract
Amazon Comprehend
Amazon Bedrock

Step #3: Model Evaluation & Deployment

Whatever you are doing with LLMs, you must use some evaluation to test if the LLM answers properly to your prompts.

Amazon Bedrock even provide evaluation against your own prompts.

When your model is ready, it's time to deploy and serve your model, and this is exactly it!



Next Steps

- Monitoring is essential in every software workflow.
  - → How do you ensure your LLMs are up and running?
  - → How do you know that your LLMs are not being jailbroken?
  - → How do you track your LLMs sentiment or response quality?
- Cost Optimization: What strategies (caching, RAG, prompt compression) do you adopt to mitigate costs in your LLMs?
- Prompt Engineering: How do you evaluate prompt techniques, security defenses, prompt optimization, and more?
- Extra points: Model Governance, security/compliance, lifecycle management, etc.

Introduction to Amazon Bedrock

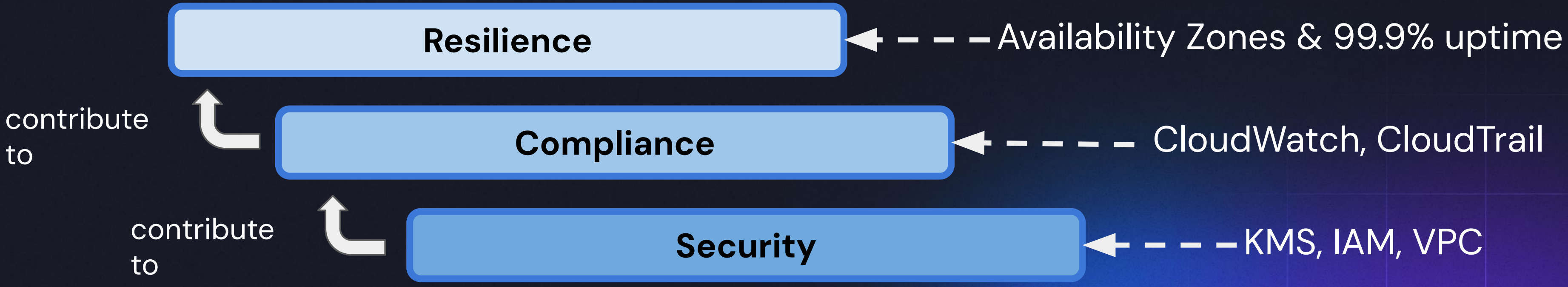
- Fully Managed service for GenAI apps with foundation models (FMs) from Amazon & other providers, accessible via API/SDK.

- Offers several features, such as Text/Image playground, RAG, fine-tuning, and AI agent orchestration, with pay-as-you-go pricing.

- Integrates seamlessly with other AWS services: S3, Lambda, CloudWatch, etc.

Why use Amazon Bedrock?

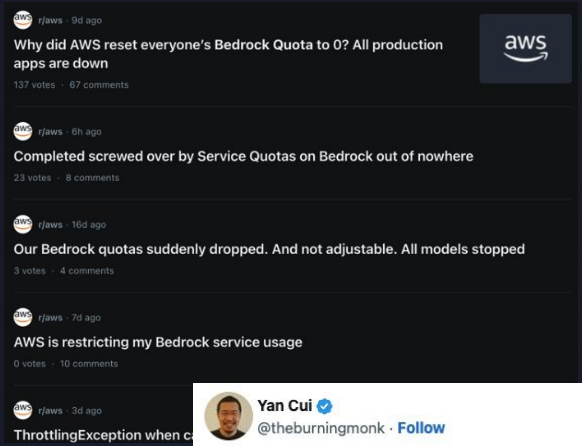
- Bedrock offers a robust LLMOps platform with a multi-layered approach towards securing & commoditizing the FMs.
- GenAI POCs often require testing multiple LLMs — switching models adds complexity. Bedrock simplifies this with a unified API for all top vendors.



Challenges in Cloud LLM Dev & Testing

Devs face several challenges:

- **High Costs**
  - Difficult to experiment/test extensively without incurring significant spend.
- **Limited Offline Iteration**
  - Hard to prototype quickly if each test requires an active cloud environment.
- **Integration Testing**
  - Debugging/validating LLMOps workflows is complex and resource-intensive



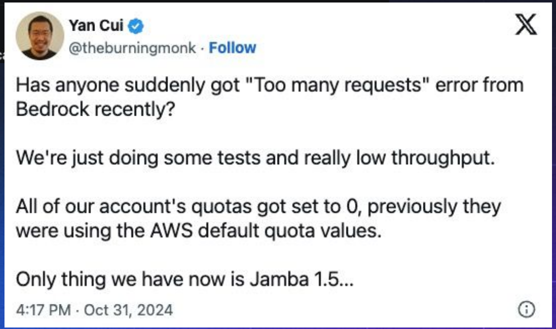
r/aws · 9d ago
Why did AWS reset everyone's Bedrock Quota to 0? All production apps are down
137 votes · 67 comments

r/aws · 6h ago
Completed screwed over by Service Quotas on Bedrock out of nowhere
23 votes · 8 comments

r/aws · 16d ago
Our Bedrock quotas suddenly dropped. And not adjustable. All models stopped
3 votes · 4 comments

r/aws · 7d ago
AWS is restricting my Bedrock service usage
0 votes · 10 comments

r/aws · 3d ago
ThrottlingException when c



Yan Cui
@theburningmonk · Follow

Has anyone suddenly got "Too many requests" error from Bedrock recently?

We're just doing some tests and really low throughput.

All of our account's quotas got set to 0, previously they were using the AWS default quota values.

Only thing we have now is Jamba 1.5...

4:17 PM · Oct 31, 2024

Challenge Scenario

How would you develop & test the following LLMOps workflow without having to deal with:

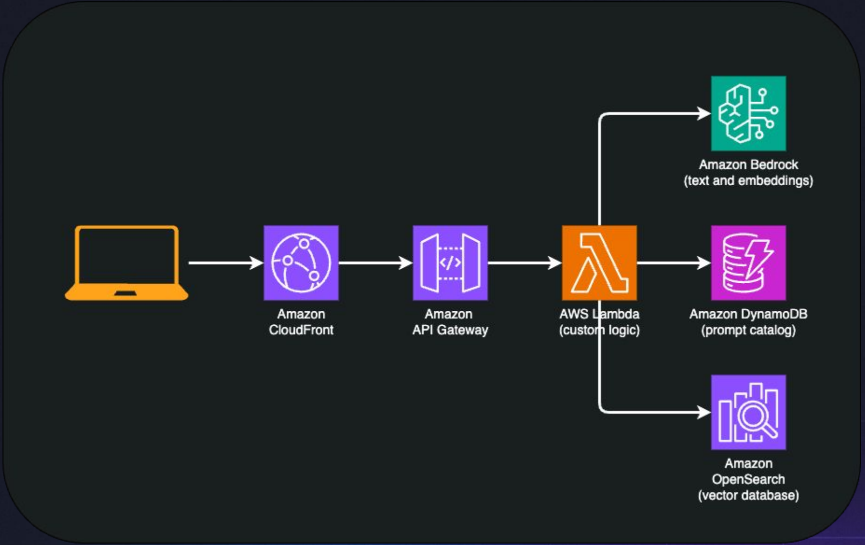
→ High API Costs

→ Network Latency

→ API rate limits & quotas

→ Integration Testing complexity

→ Limited debugging visibility


Amazon CloudFront
Amazon API Gateway
AWS Lambda (custom logic)
Amazon Bedrock (text and embeddings)
Amazon DynamoDB (prompt catalog)
Amazon OpenSearch (vector database)

What is LocalStack?

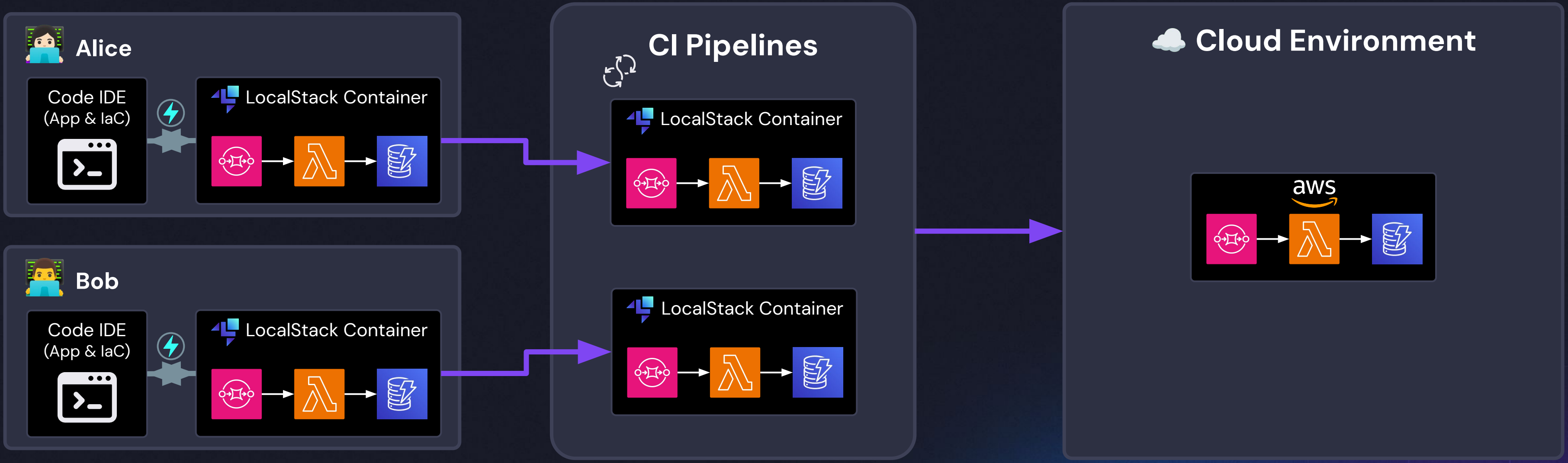
- LocalStack is a local cloud development platform
  - LocalStack core emulator: high-fidelity AWS cloud emulator (100+ services)
  - LocalStack SaaS platform: additional features to support the SDLC

- It provides an easy onramp into AWS
  - Allows teams to get started locally with low setup effort
  - Easy to "flip the switch" to AWS cloud at any time

- Enables fast local development cycles
  - Quickly iterate on your code logic (e.g., Lambda, ECS, ...)
  - Easily test integrations with your IaC framework (Terraform, CDK, etc)

- Shouldn't replace, but **complements** staging/test envs in AWS!

10,000 Foot View of LocalStack



⚡ Instant local feedback loops
🕵️ Reducing debug time from hrs to mins

✅ Isolated test sandboxes for fully repeatable tests

🎯 No cloud waste – 100% efficiency & predictability

Growing support for integrations

CI/CD Systems

GitLab HARBOR circleci
Shipyard GitHub Actions

IaC Tools

aws Cloud Development Kit HashiCorp Terraform
Pulumi

Local Dev Tools

docker Compose AWS CLI AWS CloudFormation

App Development FrameWorks

serverless Testcontainers AWS SAM
namespace

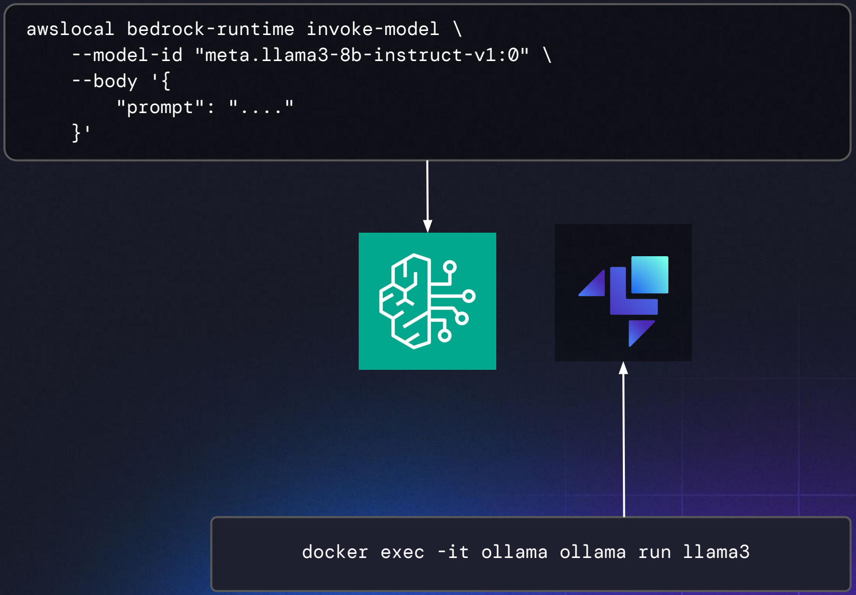
Programming Languages SDKs

spring python™ .NET
JUnit 5 JS GO

New Bedrock emulator

Initial Support for Bedrock APIs & FMs:

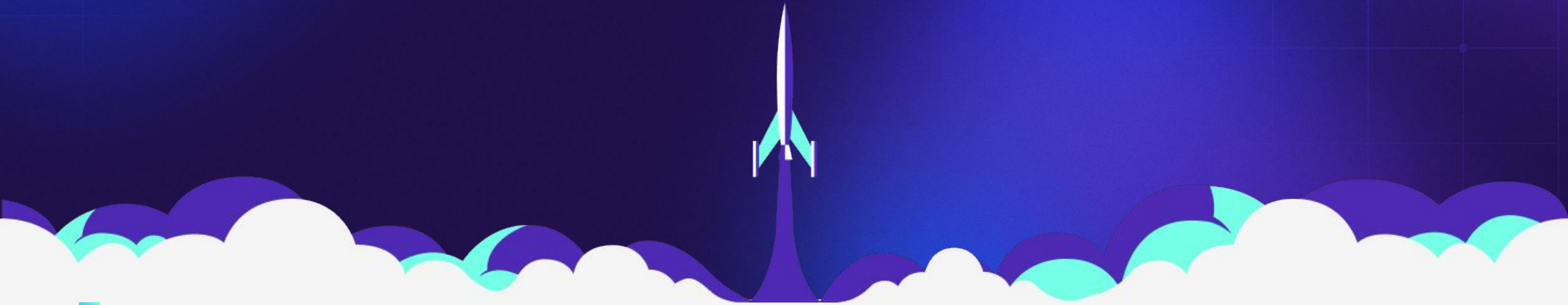
- Support for Invoke and Converse endpoints.
- Run and invoke LLMs locally with public Ollama models.
- Utilize advanced features such as Batch Streaming,
- Fully emulate LLMOps workflows on your local machine before moving to production.



```
awslocal bedrock-runtime invoke-model \
    --model-id "meta.llama3-8b-instruct-v1:0" \
    --body '{
        "prompt": "...."
    }'
```

```
docker exec -it ollama ollama run llama3
```

Quickstart



Using Bedrock emulator with AWS SDKs

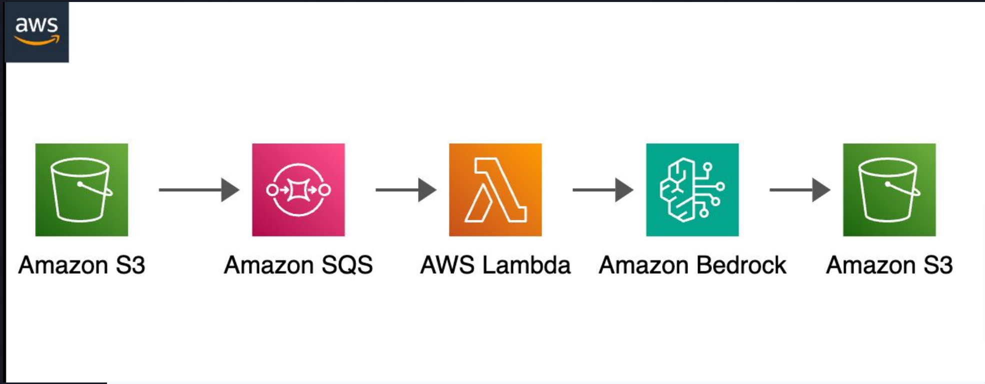
- Install the LocalStack CLI and pull the LocalStack for AWS Docker image.

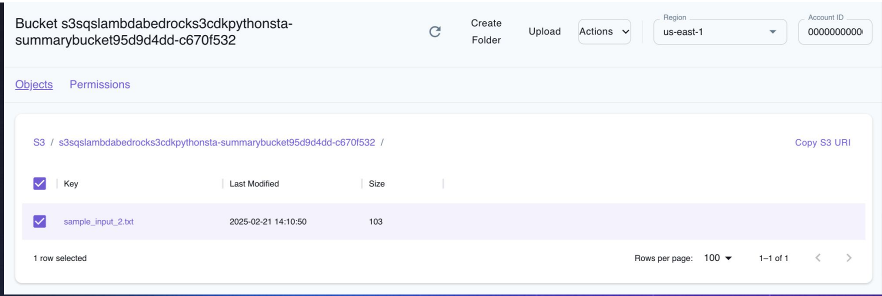
- Use AWS SDK for Python (or any other tool) to point your endpoint configuration towards `localhost:4566` or `localhost.localstack.cloud`

```python
import boto3
import json

# Initialize Bedrock client
bedrock = boto3.client('bedrock-runtime', region_name='us-east-1',
endpoint_url='http://localhost:4566')

# Make API call
response = bedrock.invoke_model(
    modelId='anthropic.claude-v2',
    body=json.dumps({
        "prompt": "Tell me a fact about space.",
        "max_tokens_to_sample": 300,
        "temperature": 0.7
    })
)

# Get response
result = json.loads(response['body'].read())
print(result['completion'])
```

Demo: Run a conversation summarization workflow

- Demo creates a creates an S3 bucket that sends events to an SQS queue.
- Lambda function receives the messages and invokes Bedrock to extract a summary.
- Result is stored in a different S3 bucket for consumption.




Bucket s3sqslambdabedrocks3cdkpythonsta-summarybucket95d9d4dd-c670f532

Create Folder | Upload | Actions | Region: us-east-1 | Account ID: 0000000000

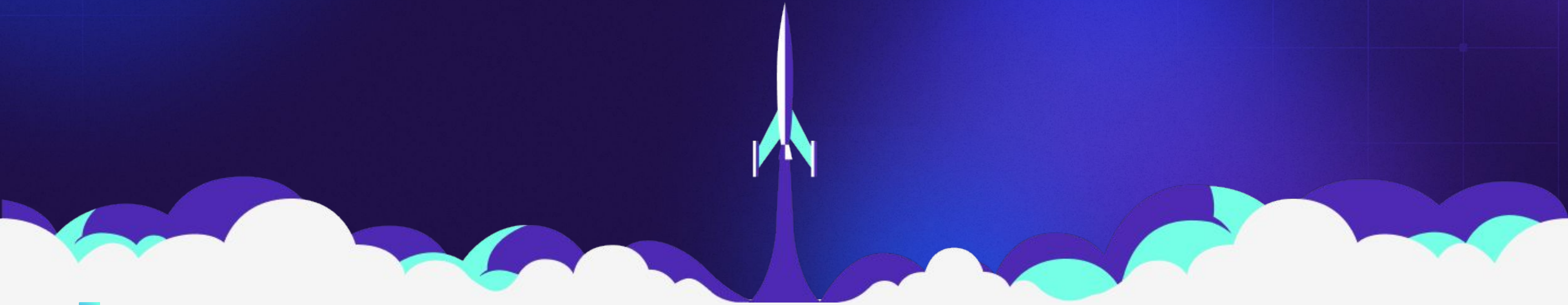
Objects | Permissions

S3 / s3sqslambdabedrocks3cdkpythonsta-summarybucket95d9d4dd-c670f532 / Copy S3 URI

✓	Key	Last Modified	Size
✓	sample_input_2.txt	2025-02-21 14:10:50	103

1 row selected   Rows per page: 100   1–1 of 1

Demo



Using local LLMs with Ollama

- LocalStack Bedrock emulator supports models from the Ollama Models library.
- To use a model, retrieve its ID from Ollama and set `DEFAULT_BEDROCK_MODEL` to that ID.
- You can also set `model-id` parameter to `ollama.<ollama-model-id>` to define it directly.

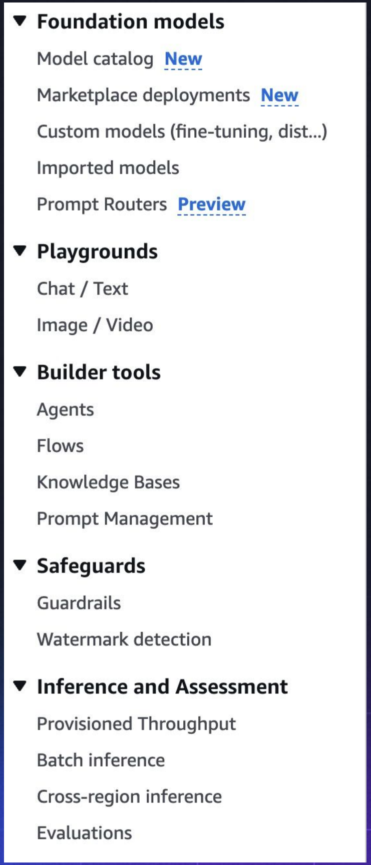
```
awslocal bedrock-runtime converse \
    --model-id "ollama.deepseek-r1" \
    --messages '[{ "role": "user", "content": [{ "text": "Hi!" }] }]'
```

What the future holds?

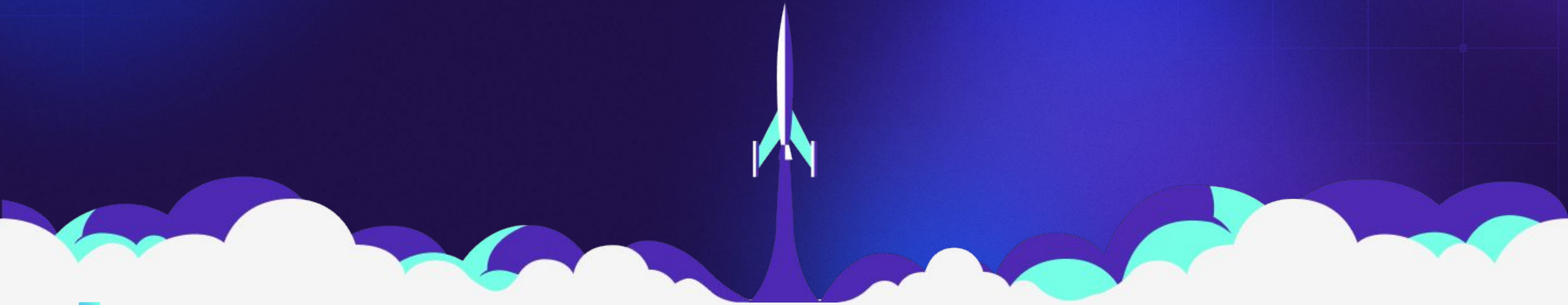
Now: Core Bedrock API support for text-based LLM invocation and batch processing.

Future:

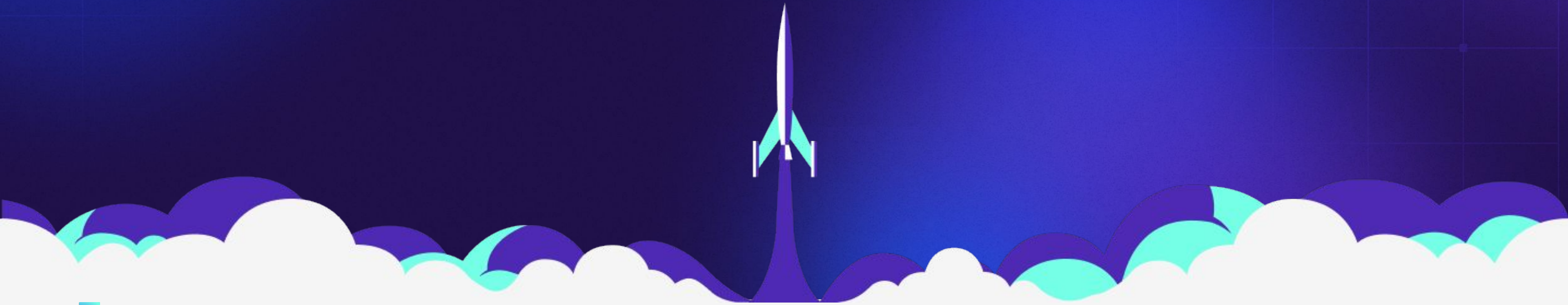
- Improve Invocation support (async)
- Builder tools (Knowledge Bases — RAG — Agents)



Quiz



Q&A



Harsh Mishra

@harsh_casper
harshcasper
/in/harshcasper

Get involved!
localstack.cloud/slack

Thank You