

AWS Transform, hands-on

Two live demos: VMware migration planning and mainframe modernization

David Ernst · Tampa Bay AWS User Group · September 30, 2026

What you'll see

1

Setup

Turn on AWS Transform, open the web app, and organize work into workspaces.

2

Demo 1 • VMware

Ask Transform to plan a VMware-to-EC2 move and get a 7R strategy report per server.

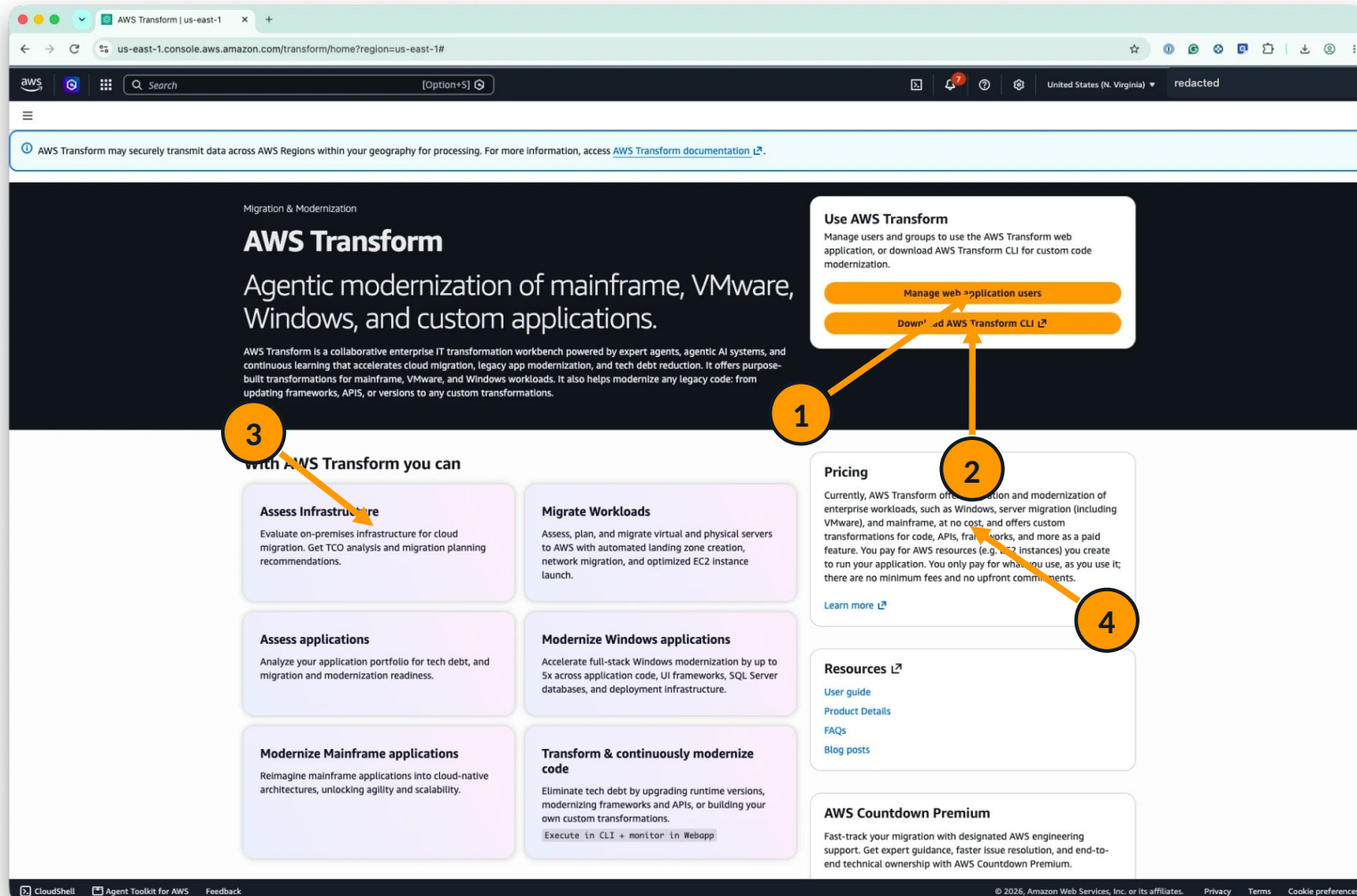
3

Demo 2 • Mainframe

Run a COBOL modernization job, hit real permission failures, fix them, and explore the results.

Start in the AWS console

AWS Transform is one service covering infrastructure, VMware, Windows, mainframe and custom code.



1 Turn on the web app and choose who can sign in.

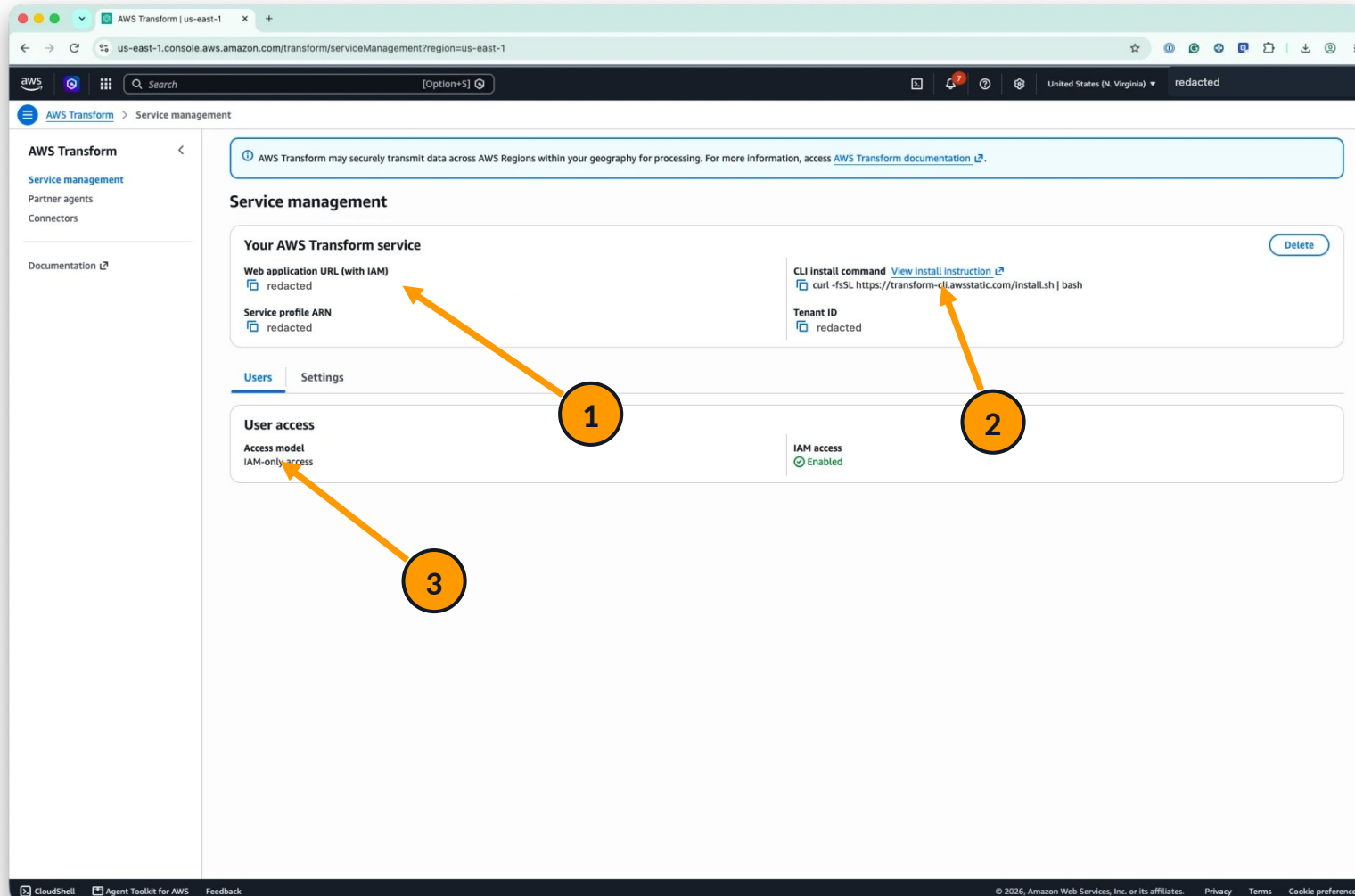
2 Or download the CLI to run custom code transformations from a terminal.

3 Six jobs it can do: assess infrastructure and apps, migrate, and modernize Windows, mainframe and custom code.

4 Pricing: migration and the built-in modernization jobs cost nothing; custom transformations are billed per agent minute.

Enable the service and the web app

Service management gives you the web app address, the CLI install command, and who has access.



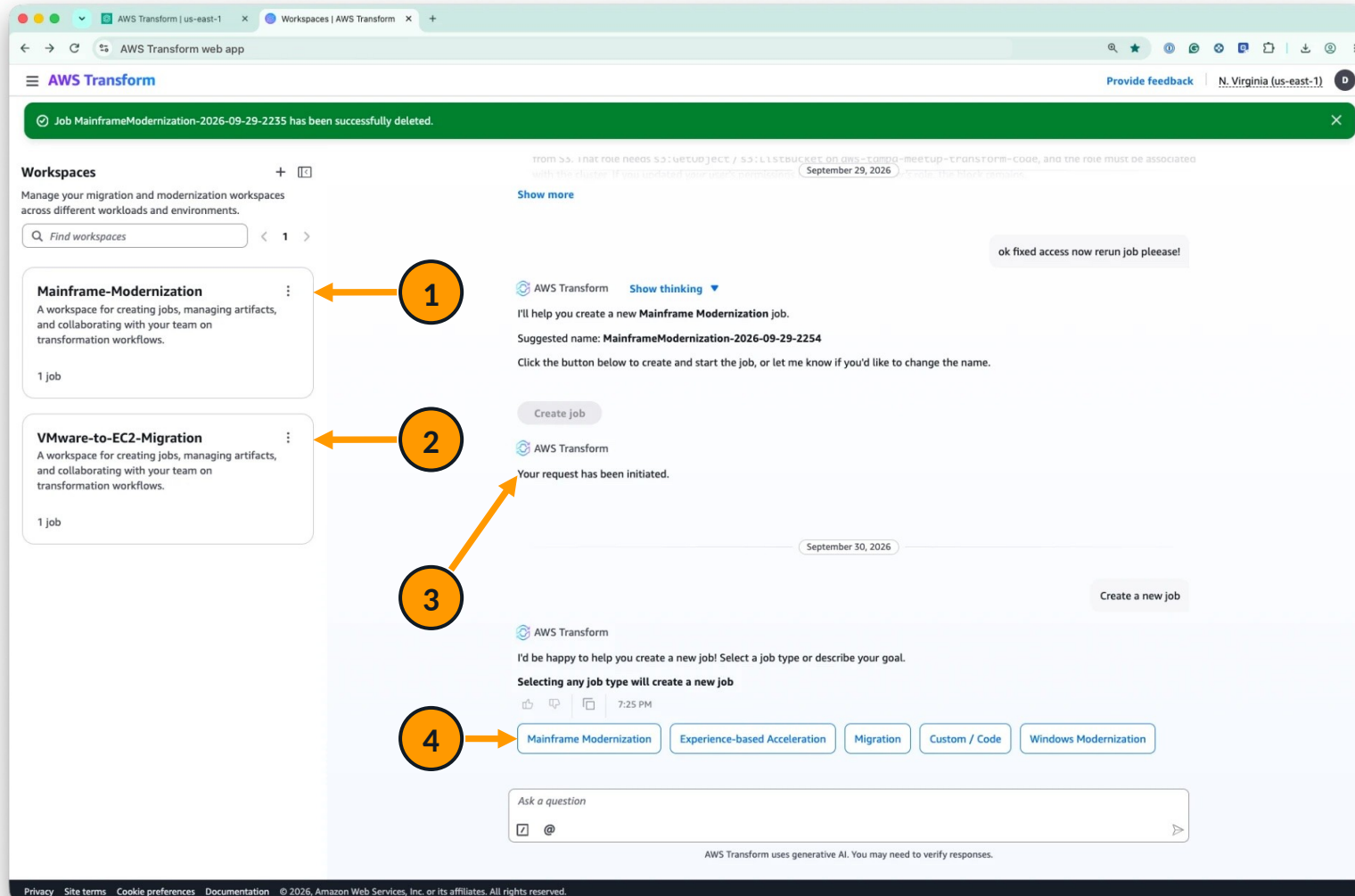
1 Web app URL for this account's Transform tenant (redacted here).

2 One-line CLI install for custom transformations.

3 IAM-only access, enabled: the quickest setup for a demo account. IAM Identity Center works for teams.

Workspaces keep each initiative separate

One workspace per effort: one for the mainframe and one for VMware. Each holds its jobs, artifacts and chat history.



1 Mainframe-Modernization workspace (demo 2).

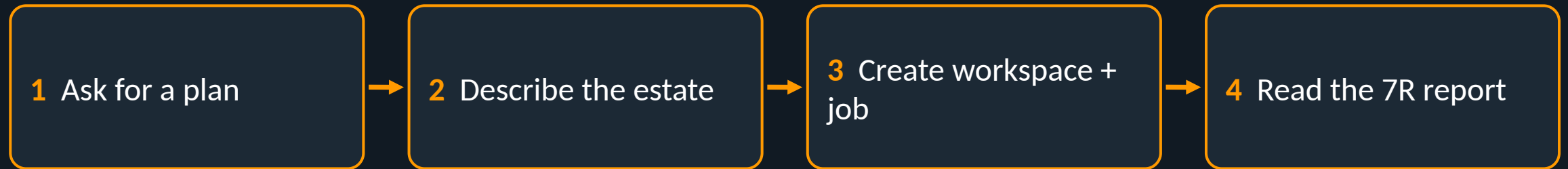
2 VMware-to-EC2-Migration workspace (demo 1).

3 Everything is driven from chat: ask in plain English, approve with a click.

4 Pick a job type: Mainframe, Experience-based Acceleration, Migration, Custom/Code, or Windows.

Demo 1 • VMware to EC2

From "help me plan" to a per-server migration strategy, in one conversation.



Ask Transform to plan the migration

Start with a plain question. Transform explains how it runs migrations, then asks what you have.

The screenshot shows the AWS Transform web application interface. At the top, there's a navigation bar with the AWS Transform logo and a 'Provide feedback' link. Below this, a green banner indicates a successful job deletion. The main content area is titled 'Mainframe-Modernization' and includes tabs for 'Jobs', 'Artifacts', 'Collaborators', 'Connectors', and 'Settings'. A 'Create job' button is visible. On the left, a 'Jobs (1)' section shows a job named 'MainframeModernization-2026-09-29-2254' in a 'Running' state. The right side of the interface features a chat window with a date separator for 'September 28, 2026'. The chat history includes a user prompt (1) asking for help planning a migration, an AI response (2) explaining the migration process and supported paths (3), and a user response (4) stating they are migrating VMware VMs to EC2. The AI response lists supported paths: Rehost (VMware to EC2), .NET modernization, Mainframe modernization, SQL Server to Aurora PostgreSQL, and Language & SDK upgrades. It also asks for more details to provide a concrete sequence. The chat input field at the bottom contains the text 'I'm migrating VMware VMs to EC2'.

1 Help me plan my workload migration to AWS

2 How AWS Transform approaches a migration

3 Every workload follows the same arc: assess → plan → execute → monitor, with human checkpoints along the way. What differs is the workload type, and that determines which agent does the work:

- Rehost (VMware → EC2) — lift-and-shift of VMs, with discovery and wave planning.
- .NET modernization — port .NET Framework apps to cross-platform .NET 8/10 (Linux-ready).
- Mainframe modernization — convert COBOL to Java.
- SQL Server → Aurora PostgreSQL — database migration and schema/code conversion.
- Language & SDK upgrades — Java, Python, Node.js version bumps and AWS SDK migrations.

To point you at the right path, tell me:

1. What are you migrating? (e.g., a fleet of VMware VMs, .NET apps, a mainframe, SQL Server databases, or code that needs version upgrades)
2. Where is it today, and roughly how large? (number of servers / apps / databases)
3. What's the goal and timeline? (cost reduction, get off end-of-life hardware, Linux/open-source targets, a specific cutover date)

Once I know that, I can recommend a concrete sequence — including whether to start with a discovery/assessment pass first, how to think about waves, and which job to create first. If you'd rather just dive in, I can also set up a workspace to hold the work.

4 I'm migrating VMware VMs to EC2

- 1 The prompt: "Help me plan my workload migration to AWS."
- 2 Every migration follows assess, plan, execute, monitor, with human checkpoints.
- 3 It lists the paths it supports: rehost, .NET, mainframe, SQL Server, and version upgrades.
- 4 Our answer: "I'm migrating VMware VMs to EC2."

The rehost journey, end to end

Transform lays out a five-step VMware plan before anything is created.

Jobs (1)
A job is a long-running process that AWS Transform executes to fulfill a specific objective defined by you.

Find jobs

MainframeModernization-2026-09-29-2254
Running

The rehost journey, end to end

- 1. Discovery** — inventory your VMs, utilization, and dependencies. Three ways to feed this in:
 - Assisted discovery — deploy the AWS Transform discovery tool in your vCenter (richest data).
 - RVTools export — upload an RVTools export from vCenter (plus an NSX export if you want network migration).
 - Manual inventory — upload a server inventory file if you can't run a tool.
- 2. Wave planning** — AI groups applications and proposes migration waves; you review and adjust the groupings, then confirm.
- 3. Target account + network** — connect the target AWS account, import network data, and pick a topology (isolated VPCs or hub-and-spoke); AWS Transform generates the VPC config for review.
- 4. Service setup** — initialize the MGN-based replication service and apply MAP tags if you're under a MAP program.
- 5. Execute waves** — deploy replication agents, launch test instances, mark apps ready, then launch cutover instances and finalize.

Human checkpoints sit at every review/approval point, so nothing cuts over without your sign-off.

Two things that shape your plan

- Where do the VMs live — on-premises vCenter, or VMware Cloud on AWS (VMC-A)? Both are supported and follow the same flow.

Show more

Create a workspace for this

AWS Transform

I'll set that up. What would you like to name the workspace? Something like VMware-Rehost or VMware-

Ask a question

AWS Transform uses generative AI. You may need to verify responses.

1 A rehost (lift-and-shift) plan built for VMware workloads.

2 Discovery three ways: the assisted discovery tool, an RVTools export, or a manual inventory.

3 AI wave planning groups apps into migration waves for you to review.

4 Service setup uses MGN replication and can apply MAP tags if you're in the program.

5 Then: "Create a workspace for this."

Create the workspace and the job

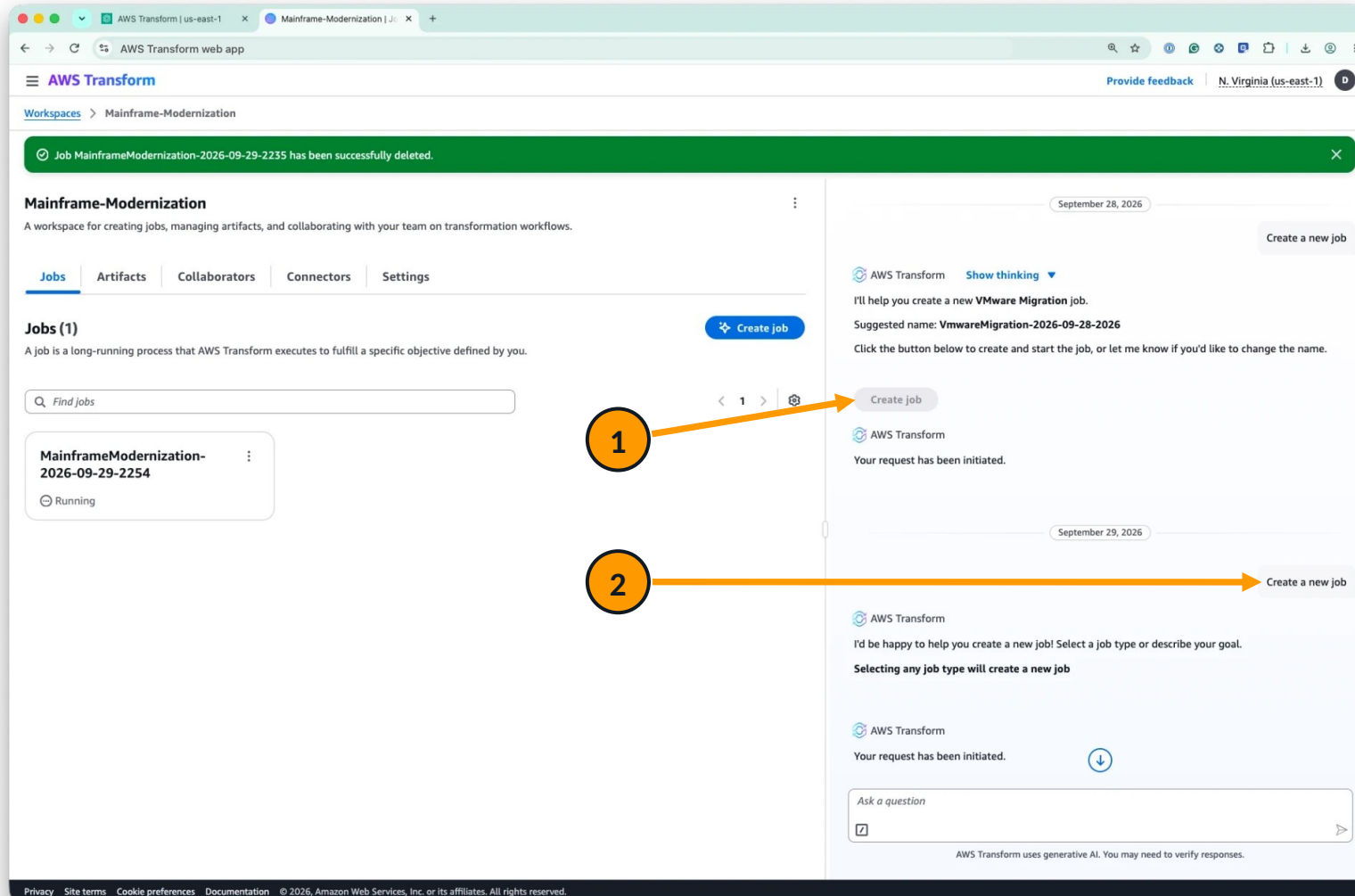
Two short replies and Transform has created a workspace and proposed a VMware migration job.

The screenshot shows the AWS Transform web application interface. At the top, there's a navigation bar with 'AWS Transform' and a 'Provide feedback' link. Below this, a green banner indicates a job deletion. The main content area is titled 'Mainframe-Modernization' and includes tabs for 'Jobs', 'Artifacts', 'Collaborators', 'Connectors', and 'Settings'. The 'Jobs' tab is active, showing a list of jobs with one job, 'MainframeModernization-2026-09-29-2254', in a 'Running' state. To the right, a chat interface shows a conversation with 'AWS Transform'. The chat history includes a message from the user asking for a workspace name and a response from AWS Transform suggesting a name. Four numbered orange circles are overlaid on the chat, with arrows pointing to specific elements: 1 points to the user's input 'Name it VMware-to-EC2-Migration', 2 points to the 'Create workspace' button, 3 points to the 'Create a new job' button, and 4 points to the suggested job name 'VmwareMigration-2026-09-28-2026'.

- 1 Name it VMware-to-EC2-Migration.
- 2 Transform confirms the name and creates the workspace.
- 3 "Create a new job."
- 4 It suggests a VMware Migration job with a dated name.

The migration job starts

One click on Create job, and the VMware job runs in the background while you keep working.

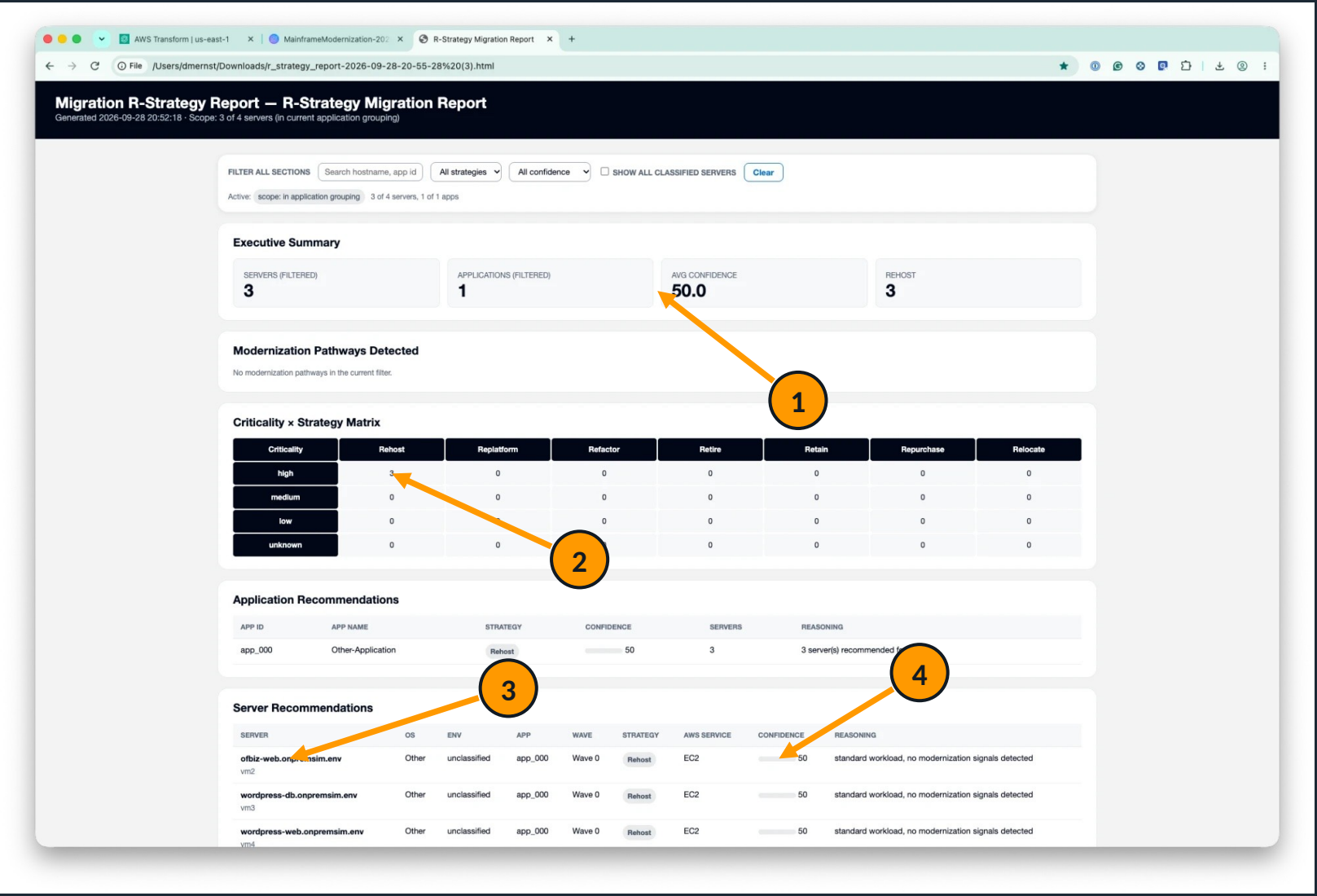


1 Create job, then "Your request has been initiated."

2 The next day we came back to the same chat to start the mainframe demo.

Result: a 7R strategy report

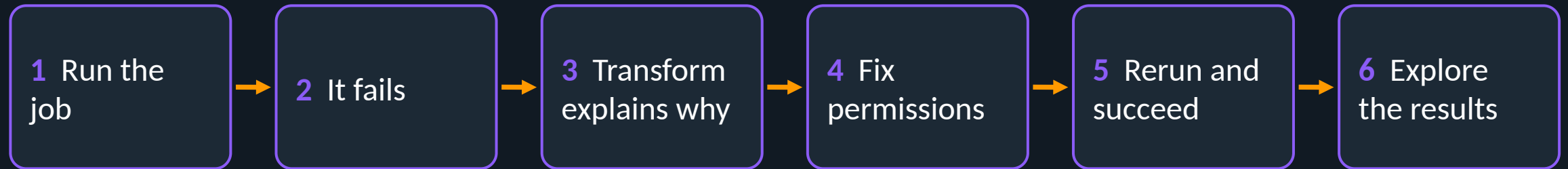
For our simulated on-prem lab, Transform recommends a strategy, a target service and a wave for each server.



- 1 Summary: 3 servers, 1 application, all recommended for Rehost.
- 2 Criticality by strategy matrix: 3 high-criticality servers go to Rehost.
- 3 Per server: ofbiz-web, wordpress-db and wordpress-web go to EC2 in Wave 0.
- 4 Confidence is 50: a starting point. Richer discovery data sharpens the recommendation.

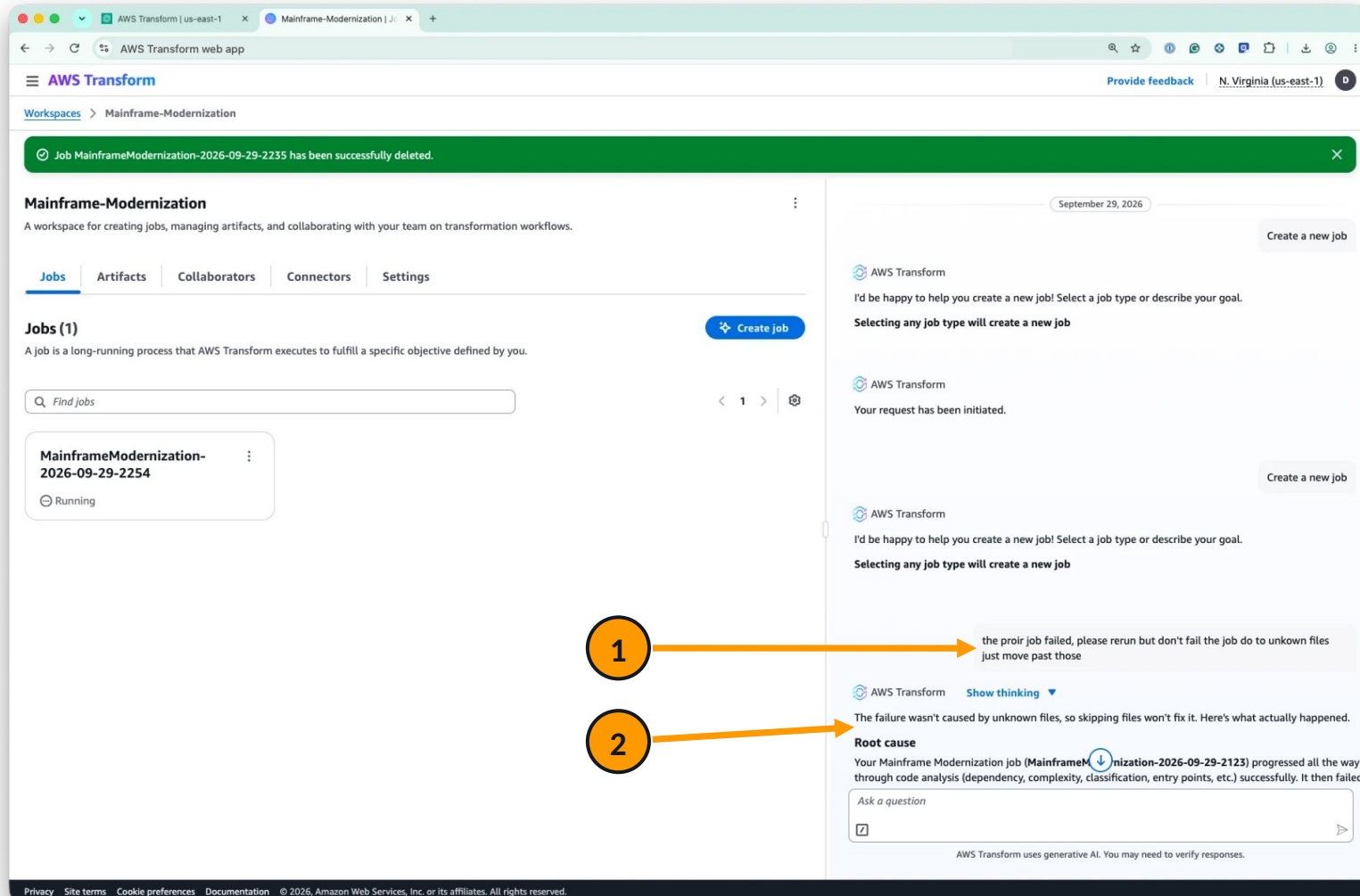
Demo 2 • Mainframe modernization

A COBOL credit-card application (AWS CardDemo), including two real failures and how we fixed them.



First run fails, and Transform pushes back

We asked it to skip unknown files. Transform pointed out that wasn't the real problem.



1 Our ask: rerun, but don't fail on unknown files.

2 "The failure wasn't caused by unknown files, so skipping files won't fix it." Then it goes straight to the root cause.

The root cause: an S3 to Neptune permission block

Code analysis worked. The failure came later, when loading the knowledge graph.

The screenshot shows the AWS Transform console interface. At the top, a green banner indicates a job was successfully deleted. Below, the 'Mainframe-Modernization' workspace is shown with tabs for Jobs, Artifacts, Collaborators, Connectors, and Settings. A job named 'MainframeModernization-2026-09-29-2254' is listed as 'Running'. To the right, the job details are displayed, including a message from AWS Transform: 'the prior job failed, please rerun but don't fail the job do to unkown files just move past those'. The 'Root cause' section explains that the job progressed through code analysis but failed at the 'Knowledge Graph ingestion' step. Three numbered callouts (1, 2, 3) point to specific parts of the error message: 1 points to 'Unable to connect to s3 endpoint', 2 points to 'Please verify your S3 configuration', and 3 points to 'Failure reason: NEPTUNE_CUSTOMER_ERROR'. The console also lists likely causes: an S3 VPC endpoint policy or bucket policy denying access, a missing/misconfigured S3 VPC endpoint, or an SCP/account guardrail restricting connectivity. A note states the job is not restartable.

1

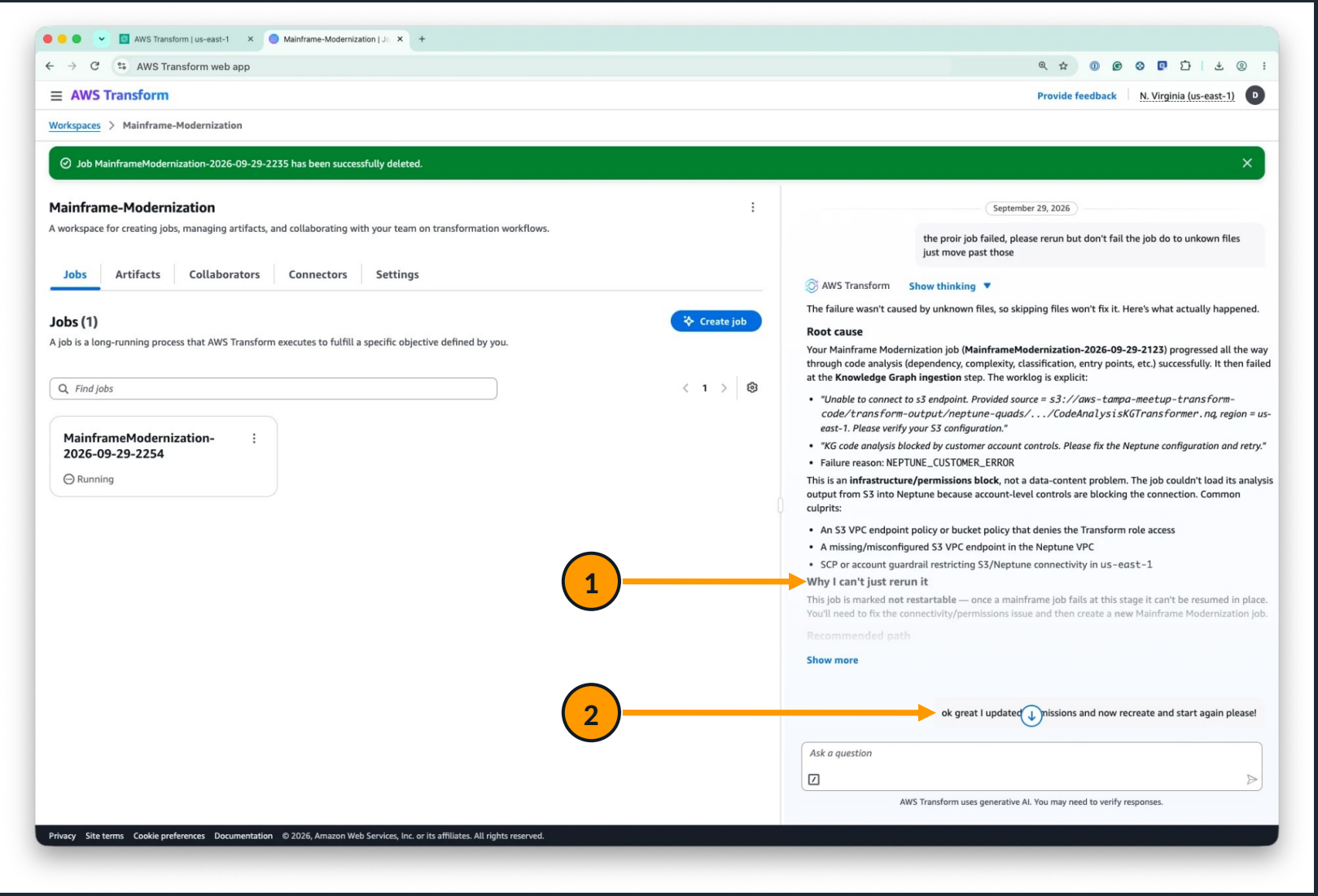
2

3

- 1 Code analysis succeeded (dependencies, complexity, entry points), then it failed at Knowledge Graph ingestion.
- 2 The work log: unable to connect to the S3 endpoint, a NEPTUNE_CUSTOMER_ERROR.
- 3 Likely causes: a VPC endpoint or bucket policy, a missing S3 endpoint, or an SCP.

Fix the permissions and create a fresh job

Failed mainframe jobs at this stage can't resume, so the fix is a new job.

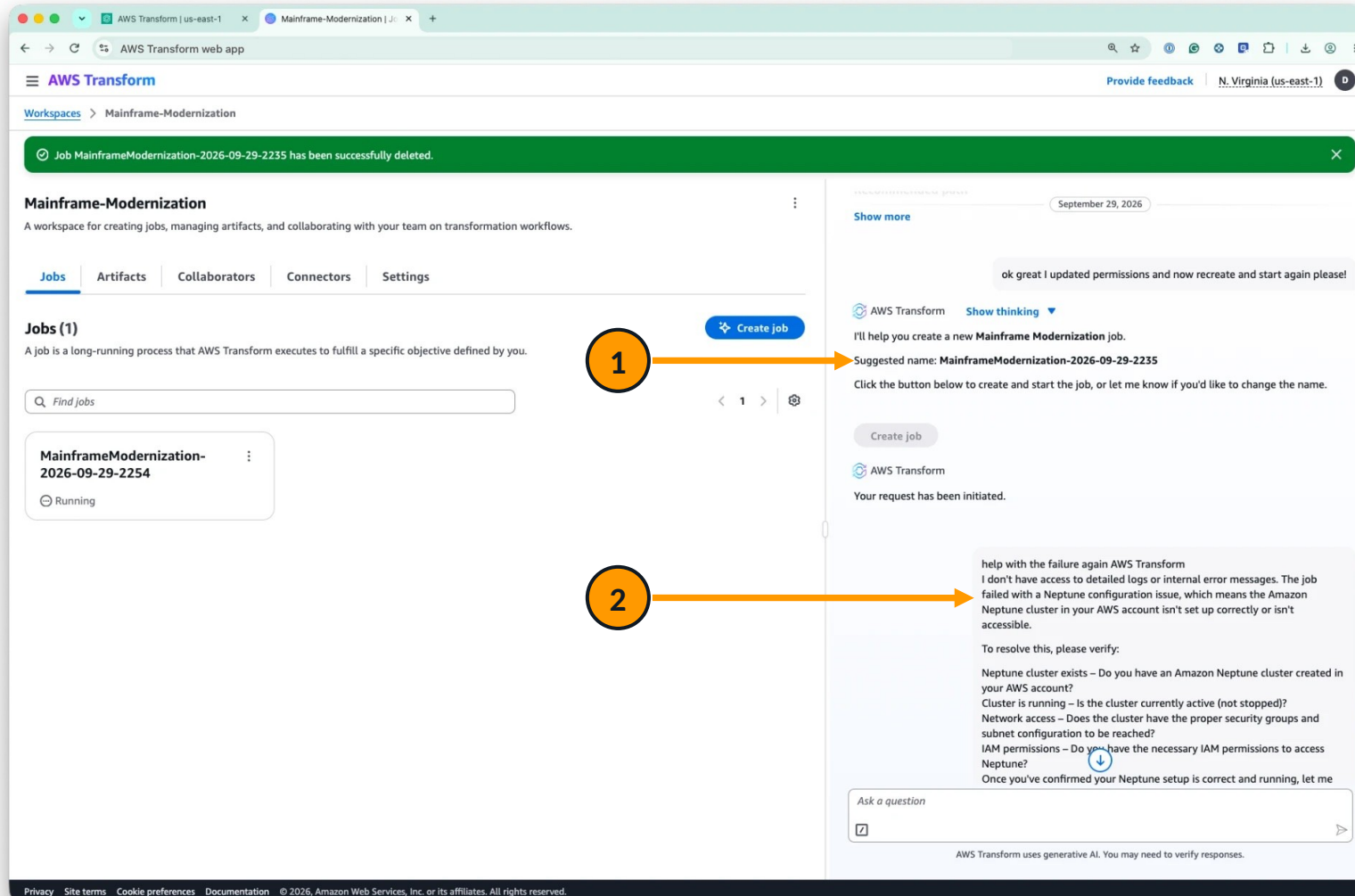


1 Why it can't just rerun: the job is marked not restartable.

2 "OK, I updated permissions. Now recreate and start again."

Second run, second failure: the Neptune setup

The new job hit the next gap, so we worked through the Neptune checks.

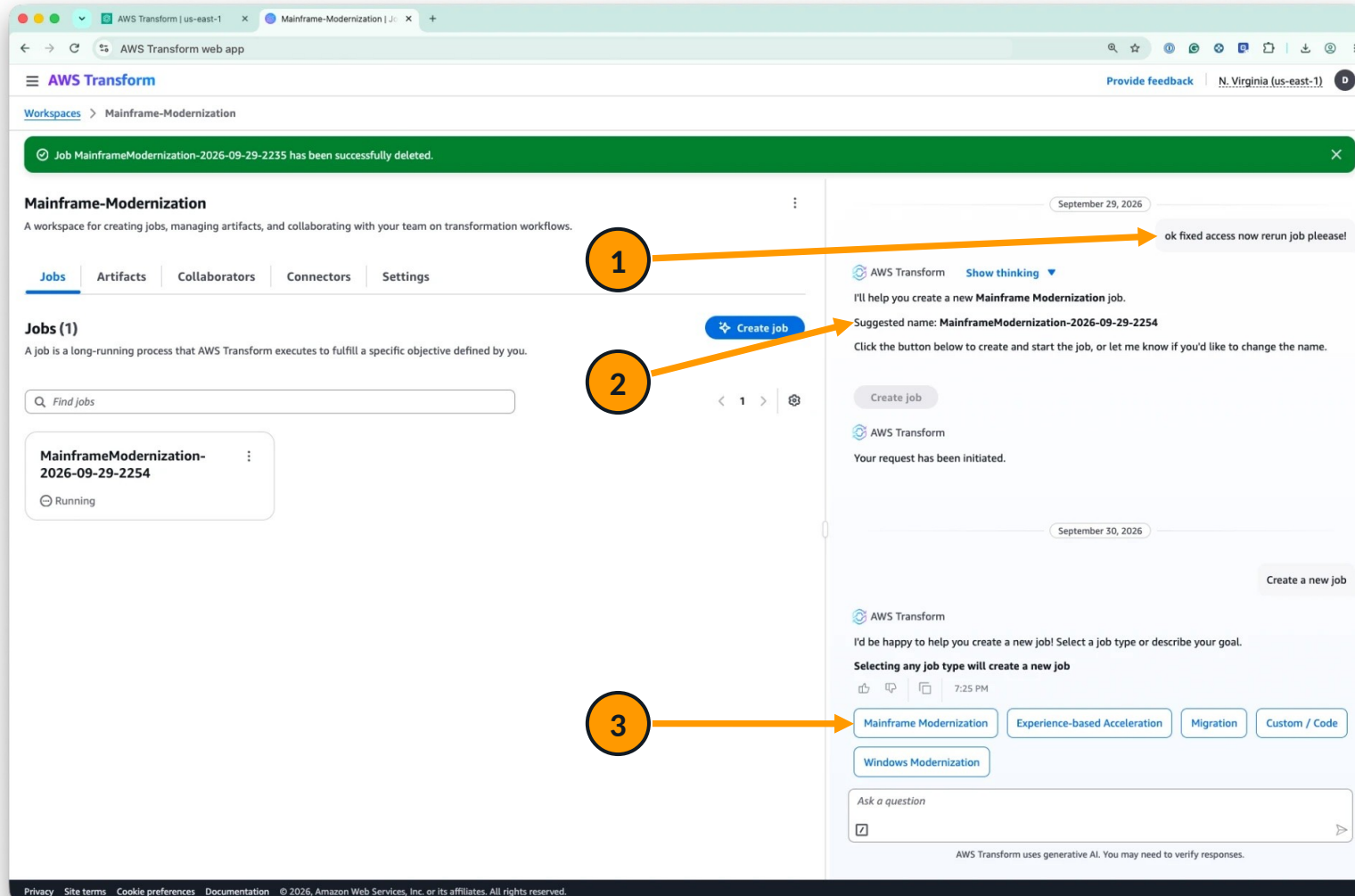


1 Transform creates and starts the new job.

2 The checklist: does the Neptune cluster exist, is it running, can it be reached, and do the IAM permissions allow it.

Access fixed, run it again

With S3 and Neptune access in place, a one-line request starts the third run.



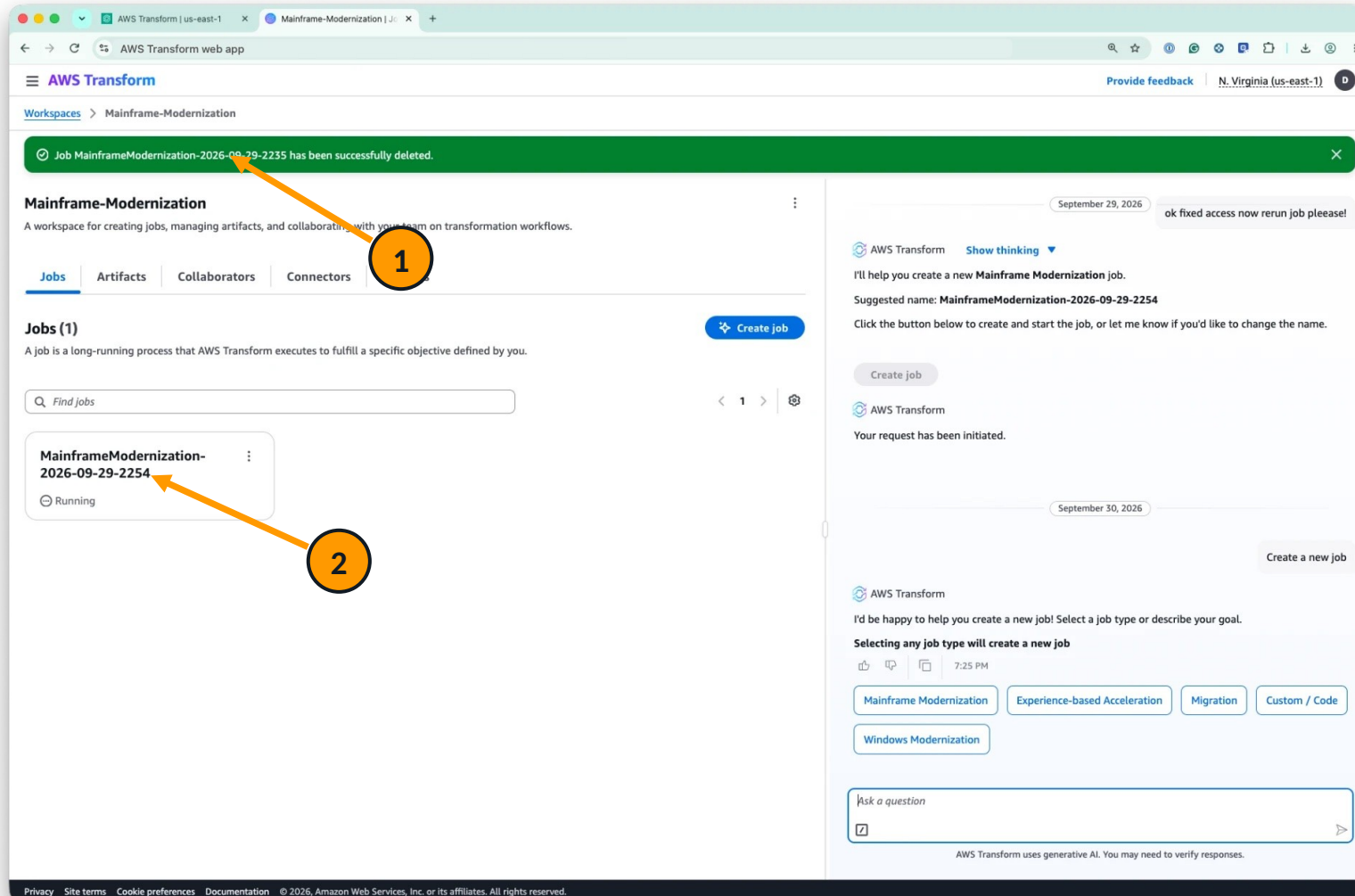
1 "OK, fixed access. Now rerun the job, please!"

2 A new Mainframe Modernization job, created and started.

3 The same job-type buttons are always one click away.

The third run is underway

The failed run is cleaned up, and the new job shows as Running in the workspace.



1 The failed job is deleted, so nothing stale is left in the workspace.

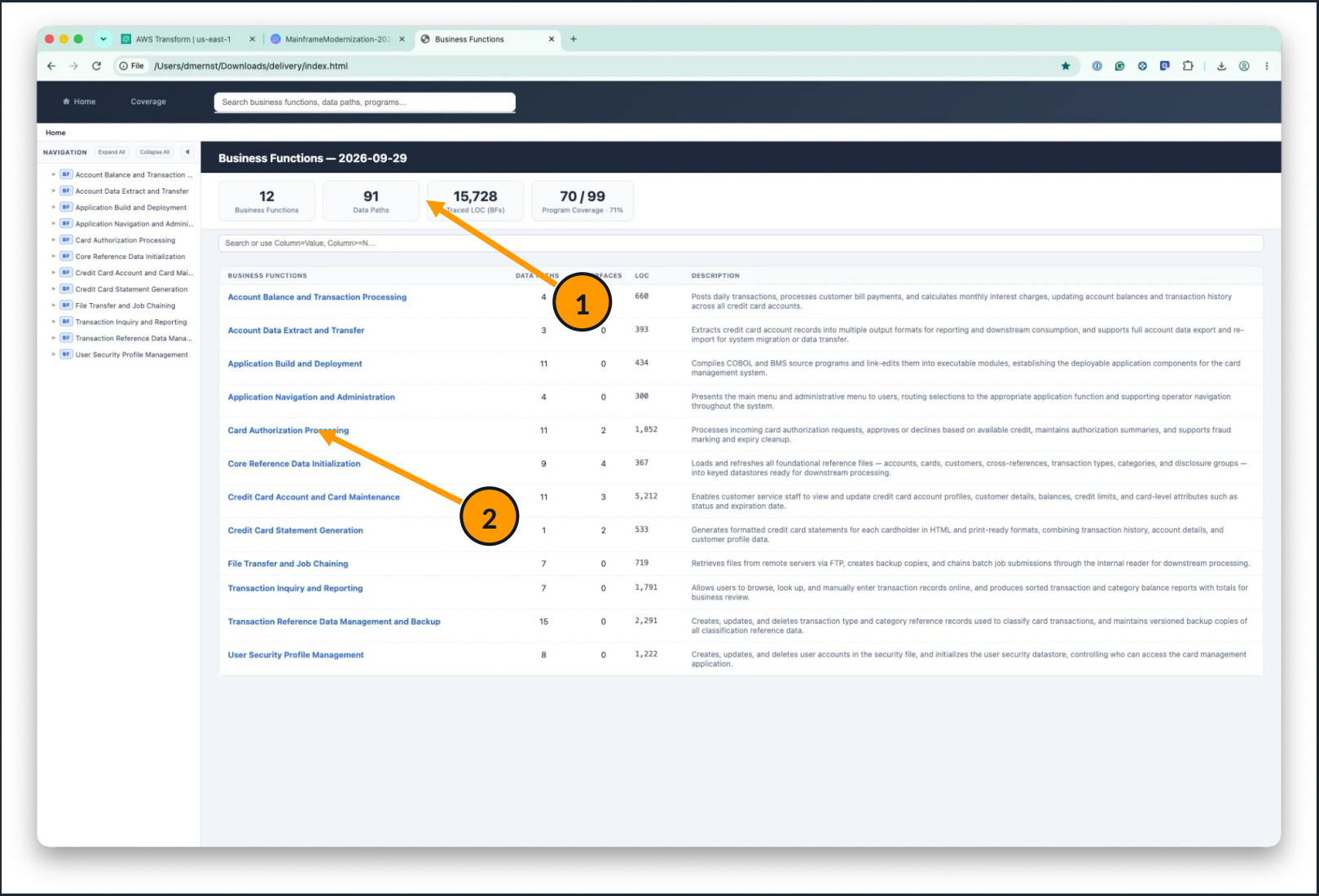
2 MainframeModernization (third run): Running.

Every step succeeded

The workspace summary shows the whole plan, from kickoff to generated requirements.

What it found: 12 business functions

Transform turns the code into a map of what the application does, in business terms.



1 12 business functions, 91 data paths, 15,728 traced lines of code, 71% program coverage.

2 Each function is named and described in plain English, for example Card Authorization Processing.

Drill into one business function

Account Balance and Transaction Processing breaks down into the batch and online flows behind it.

Home

Coverage

Search business functions, data paths, programs...

Home

Account Balance and Transaction Processing (BF)

NAVIGATION

Expand All

Collapse All

Account Balance and Transaction Processing

Monthly Credit Card Interest Posting

Daily Credit Card Transaction Posting

Credit Card Bill Payment Processing

Card Transaction Consolidation and Master Load

Account Data Extract and Transfer

Application Build and Deployment

Application Navigation and Administration

Card Authorization Processing

Core Reference Data Initialization

Credit Card Account and Card Management

Credit Card Statement Generation

File Transfer and Job Chaining

Transaction Inquiry and Reporting

Transaction Reference Data Management

User Security Profile Management

Business Function - Account Balance and Transaction Processing

Posts daily transactions, processes customer bill payments, and calculates monthly interest charges, updating account balances and transaction history across all credit card accounts.

4

1

660

Data Paths

DP Connections

Interfaces

Search or use Column=Value, Column=N...

NAME	TYPE	CATEGORY	ENTRY	EXIT	PROGRAMS	CHAIN	LOC	DESCRIPTION
Monthly Credit Card Interest Posting DP-CBACT04C-666f6f349	BATCH	jcl	INTCALC.jcl	AMS_M2_CARDDEMO_SYSTRAN this flow writes 2 data stores; one is shown	CBACT04C.cbl	1	127	This batch job calculates monthly interest charges for credit card accounts by reading transaction category balances, applying interest rates, and computing...
Daily Credit Card Transaction Posting DP-CBTRN02C-6843847f	BATCH	jcl	POSTTRAN.jcl	AMS_M2_CARDDEMO_TRANSACTION_VSAM_KSDS this flow writes 4 data stores; one is shown	CBTRN02C.cbl	1	173	This batch job processes daily credit card transactions by validating each transaction against the card cross-reference and account records, then posting...
Credit Card Bill Payment Processing DP-CBTL08C-c7ea55f	ONLINE	action	CB00	AMS_M2_CARDDEMO_TRANSACTION_VSAM_KSDS this flow writes 2 data stores; one is shown	CBTL08C.cbl		334	This data path processes a bill payment transaction initiated by a customer through an online CICS screen.
Card Transaction Consolidation and Master Load DP-COBTRAN-578444f	BATCH	jcl	COBTRAN.jcl	AMS_M2_CARDDEMO_TRANSACTION_COMBINED this flow writes 2 data stores; one is shown	COBTRAN.jcl		26	This batch job consolidates card transaction records by merging the current transaction file with system-generated transactions into a single combined file.

1

2

3

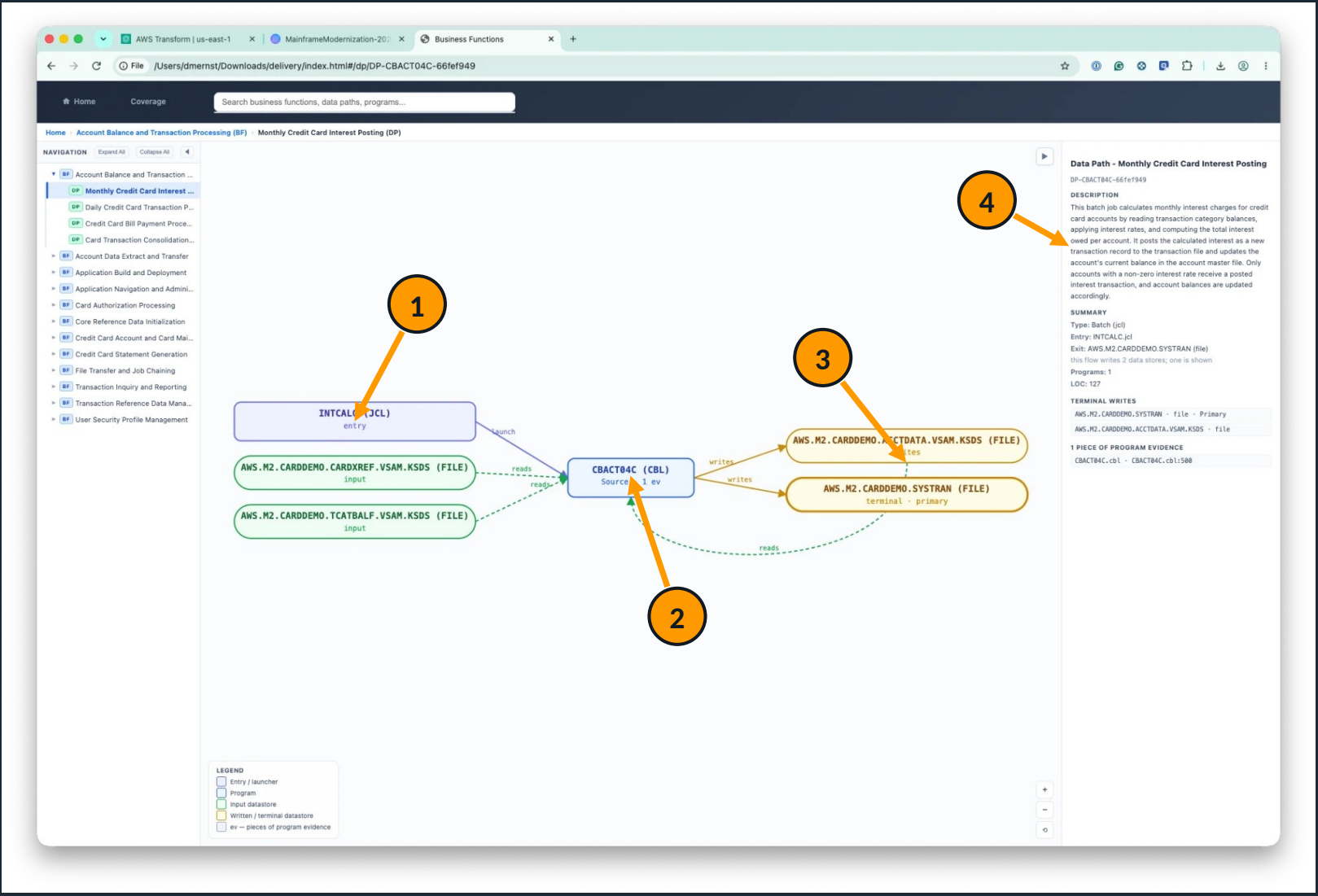
1 The business function and its plain-English description.

2 4 data paths: interest posting, daily transaction posting, bill payment, and consolidation.

3 Each path traced to its JCL entry point and COBOL program.

One data path, as a graph

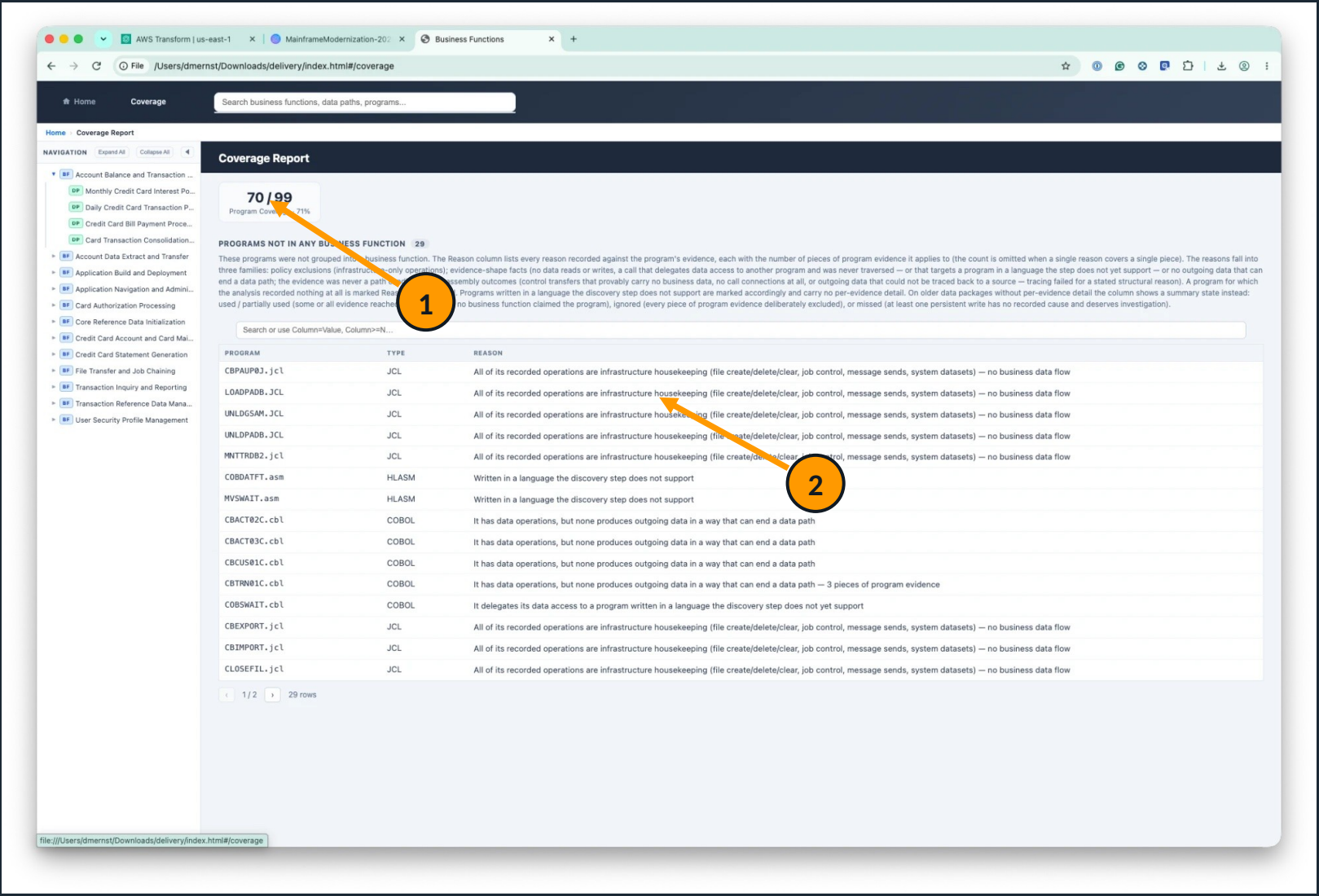
Monthly interest posting: the JCL launches the program, it reads two files and writes two.



- 1 Entry: the INTCALC JCL job.
- 2 Program: CBACT04C (COBOL).
- 3 Writes: account data and the system transaction file.
- 4 A plain-English description, plus the program evidence (the exact COBOL line).

And what it didn't cover, with reasons

70 of 99 programs map to a business function. The other 29 are listed with the reason for each.



1 Program coverage: 70 of 99 (71%).

2 Each gap has a reason: housekeeping-only JCL, assembler it doesn't support yet, or no outgoing data.

What the demos showed

1

Chat-driven, human-approved

Plain-English requests, with a confirm click before anything is created.

2

Failures explained, not hidden

It found the real cause (S3 and Neptune access), not the one we guessed.

3

Output you can act on

A 7R report per server, a business-function map, data-path graphs, and requirements ready for Kiro.

Try it yourself

.NET Framework to Linux

catalog.workshops.aws/transformdotnet

VMware migrations

catalog.workshops.aws/aws-transform-vmware

Mainframe to microservices

catalog.workshops.aws/mainframe-reimagine

Custom transformations

catalog.workshops.aws