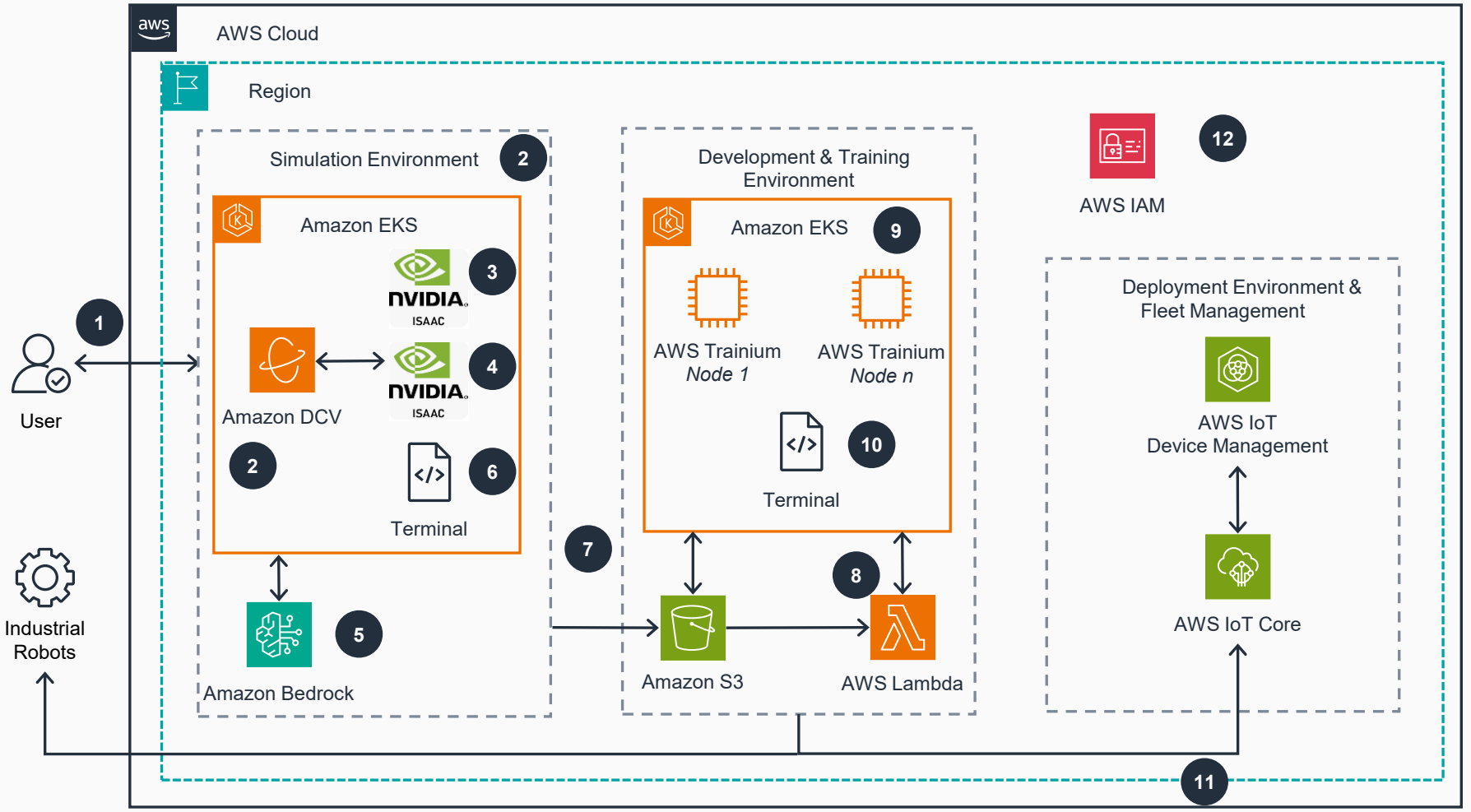
Guidance for Using Bedrock FM & AWS Trainium in AI Robot Fleet Training

Imitation Learning Simulation Environment

This architecture diagram shows a robotic learning system integrating the intelligence of foundation models with ML and mathematical algorithms, accelerated by AWS Trainium/GPU infrastructure and managed through cloud-native technologies.



1. To teach the robot to perform tasks like pushing a T-bar into a T-slot on a workbench, connect to the simulation environment and choose between a single environment for simple tasks or multiple parallel environments for complex training scenarios.

2. Connect to Isaac Sim simulation environments running in **Amazon Elastic Kubernetes Service (Amazon EKS)** through **Amazon DCV**. Deploy the DCV server as a DaemonSet to provide visualization capabilities for simulation pods. Run multiple Isaac Sim environments in parallel across **Amazon EKS** nodes to enable concurrent processing for simulations and dataset generation.

   Optional: Consider **AWS Batch** as an alternative orchestration service to manage multiple **Amazon Elastic Compute Cloud (Amazon EC2)** instances running Isaac Sim simulations, providing automatic scaling and job management for distributed simulation workloads.

3. In the NVIDIA Isaac Sim simulation environment, step up the environment with the Universal Scene Description (USD) file.

4. Isaac Sim generates robot manipulation scenarios with randomized T-bar and robot positions, publishing scene data (camera images, robot poses, object positions) to Robot Operating System 2 (ROS 2) topics at 10-30 Hz frequency.

5. **Amazon Bedrock** foundation models analyze the robot workspace conditions (object distances, robot positioning, environmental constraints) and return high-level strategy recommendations. For advanced and complex tasks, use **Amazon Bedrock AgentCore** with Strands agents and Model Context Protocol (MCP) server to coordinate and orchestrate the tasks.

6. The application logic in the simulation environment processes AI strategies and uses ML algorithms to generate safe robot positions and velocities, while simultaneously storing all episode data through the HuggingFace LeRobot library for optimized formatting and preprocessing.

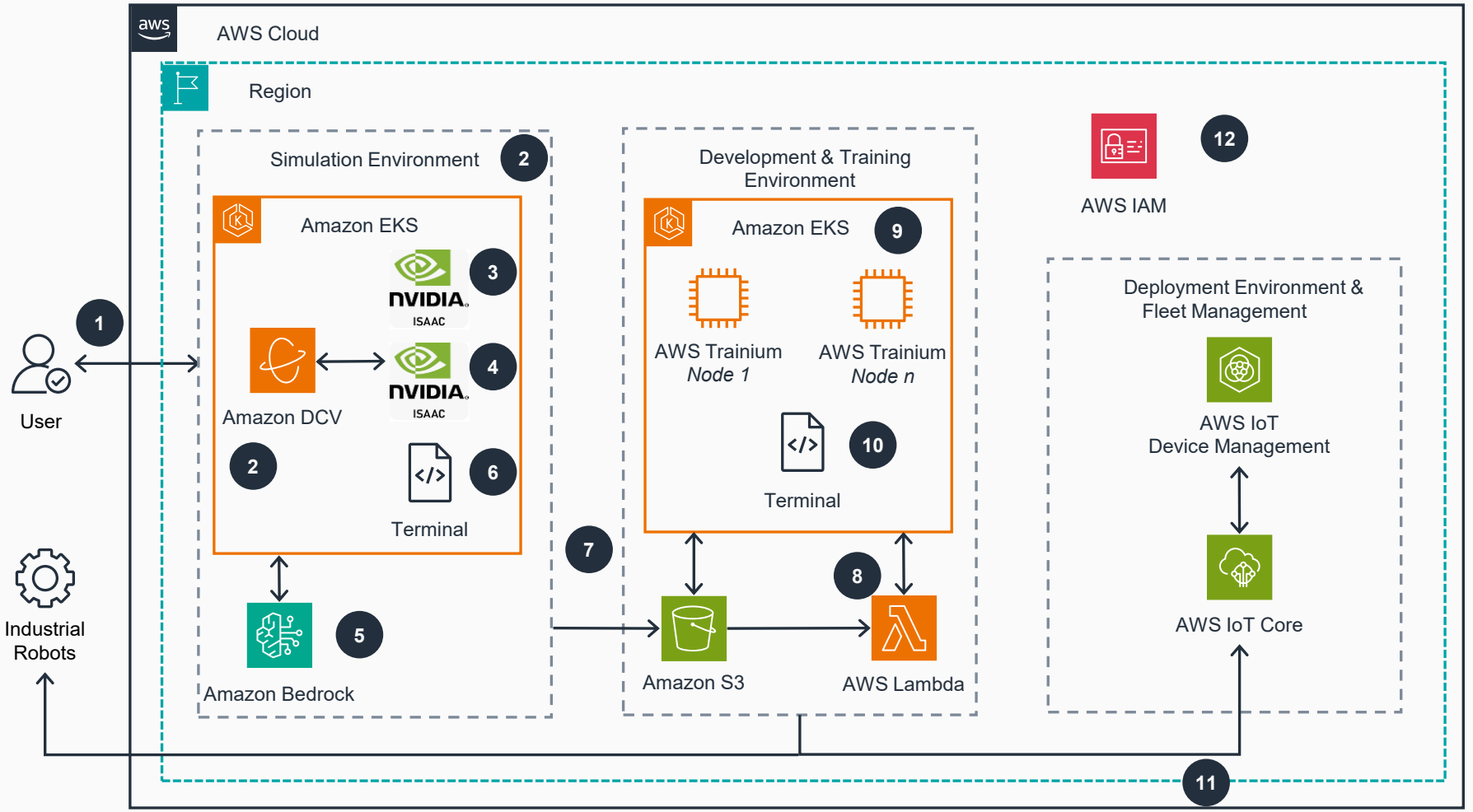
Reviewed for technical October 28, 2025

AWS Reference Architecture

Guidance for Using Bedrock FM & AWS Trainium in AI Robot Fleet Training

Imitation Learning Simulation Environment

This architecture diagram shows a robotic learning system integrating the intelligence of foundation models with ML and mathematical algorithms, accelerated by AWS Trainium/GPU infrastructure and managed through cloud-native technologies.



7 Completed episodes are organized using LeRobot's standardized directory structure organize (data/chunk-000/episode_000000.parquet) and automatically uploaded with metadata files enabling seamless integration with LeRobot training pipelines.

8 The **Amazon Simple Storage Service (Amazon S3)** bucket triggers **AWS Lambda** functions upon new episode uploads to updatethe LeRobot dataset indices and notify the **Amazon EKS** or GPU training orchestration system of available data for processing.

9 The **Amazon EKS** cluster with dedicated **AWS Trainium** node groups (trn1.32xlarge instances) or GPUs receives training job requests and provisions containerized training environments with LeRobot framework.

10 LeRobot's ecosystem processes **Amazon S3** episode data, implementing DiffusionPolicy with GPU/Trainium optimizations. The system executes mixed-precision distributed training and packages final models for **Amazon S3** and HuggingFace Hub. The pipeline uses transformers, accelerator-specific operations, and standardized workflows throughout.

11 **AWS IoT Core** manages robot fleet deployment by creating **AWS IoT Jobs** for model distribution. Robots subscribe to these jobs, download and validate LeRobot models from **Amazon S3**, and execute standardized deployment steps. The AWS IoT device management with IoT jobs monitors deployment progress and can trigger rollbacks if needed. Successfully deployed robots publish operational metrics while performance data streams to **Amazon S3**.

12 The imitation learning pipeline uses **AWS IAM** roles for service-to-service authentication between core components (**Amazon EKS**, **Amazon S3**, **Amazon Bedrock**, **AWS Lambda**, **AWS IoT Core**) and controls access to resources. Robot devices authenticate using IoT thing policies while accessing **Amazon S3** for model downloads, with all cross-service communications enforced through least-privilege **IAM** policies.

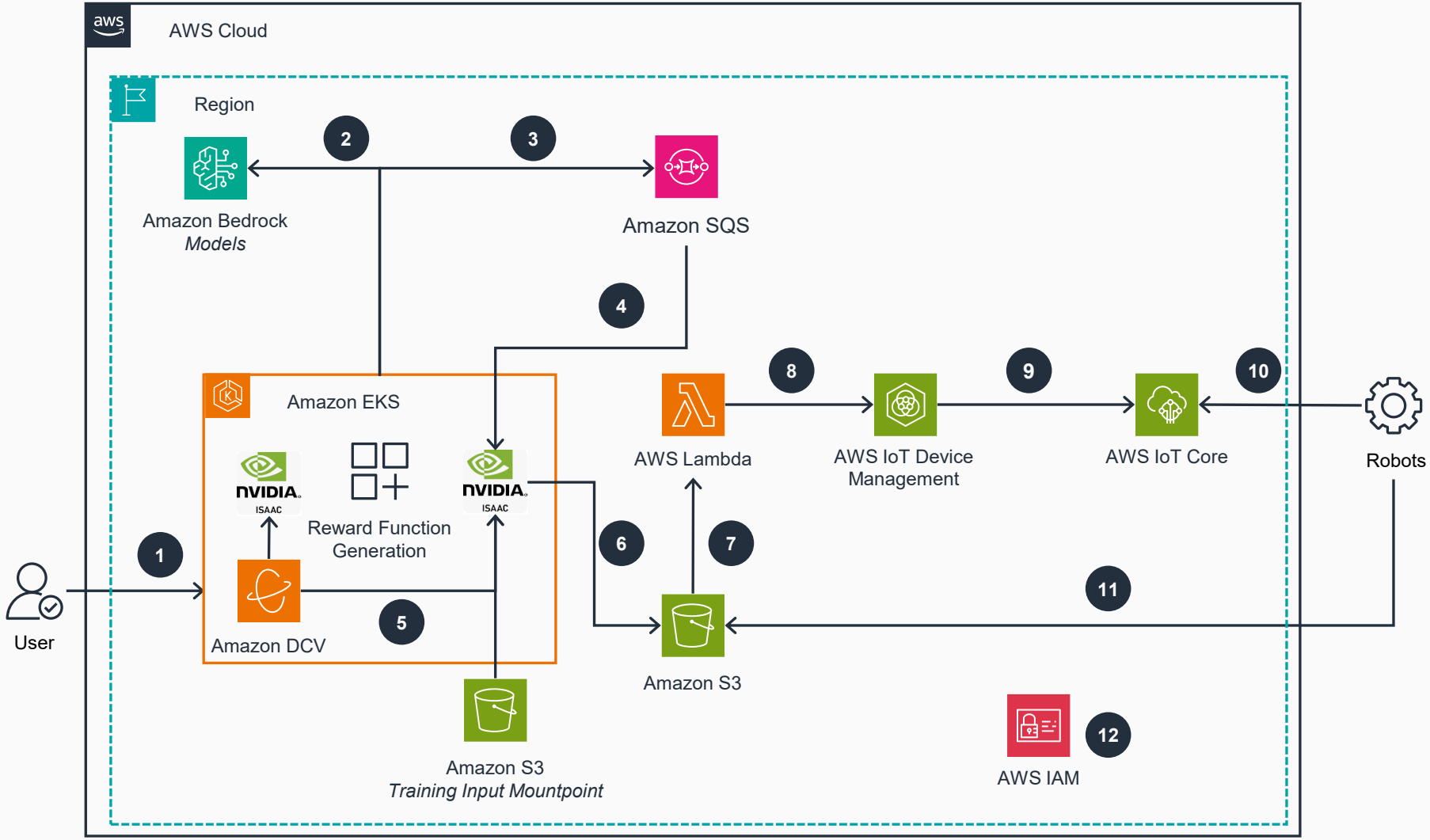
Reviewed for technical October 28, 2025

AWS Reference Architecture

Guidance for Using Bedrock FM & AWS Trainium in AI Robot Fleet Training

Reinforcement Learning Training Environment

This architecture diagram shows developers how to train robotic agents using NVIDIA Isaac Sim on Amazon EKS with LLM-generated reward functions, then automatically deploy trained models to physical robots using AWS IoT services.



1. A developer deploys the NVIDIA Isaac Sim on **Amazon Elastic Kubernetes Service (Amazon EKS)** and accesses its interface via **Amazon DCV**. The developer defines a task (e.g. opening a box) within the Isaac Sim environment, which serves as the simulation environment for the reinforcement learning agent to train on.

2. The Isaac Sim controller pods leverage **Amazon Bedrock** to sample reward functions from the Large Language Model (LLM). This automated process of generating reward functions is a key innovation, as crafting effective reward functions manually can be very challenging.

3. **Amazon SQS** receives the award function along with the training and simulation input data, as a message. The **Amazon SQS** queue acts as the intermediary between the reward function generation and the training execution.

4. The Isaac Sim training pods read the message from the **Amazon SQS** queue and execute the training and simulation processes.

5. The input data files and results are read and stored using an **Mountpoint for Amazon Simple Storage Service (Amazon S3)**, which provides a cost-effective and low-latency file storage solution for the training artifacts.

   The developer interactively observes and debugs the training process in near real-time through the **Amazon DCV** remote visualization interface, gaining valuable insights into the agent's learning progress.

6. The developer stores new model files in an **Amazon S3** bucket for persistence.

7. At the time of model creation, the creation event in Amazon S3 triggers an **AWS Lambda** function. This function deploys the trained model to the physical robot devices.

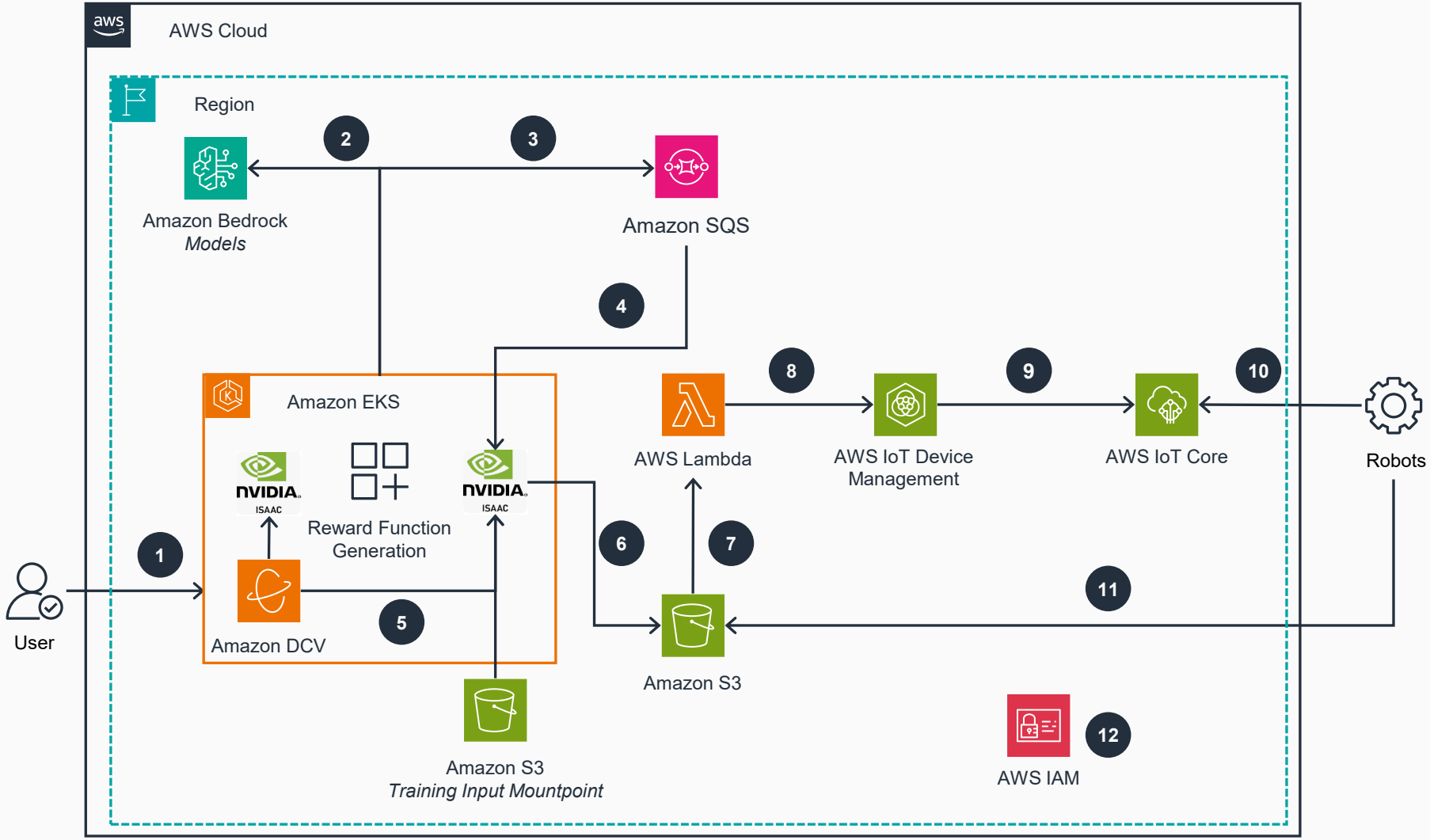
Reviewed for technical October 28, 2025

AWS Reference Architecture

Guidance for Using Bedrock FM & AWS Trainium in AI Robot Fleet Training

Reinforcement Learning Training Environment

This architecture diagram shows developers how to train robotic agents using NVIDIA Isaac Sim on Amazon EKS with LLM-generated reward functions, then automatically deploy trained models to physical robots using AWS IoT services.



8 The **AWS Lambda** function creates an AWS IoT Job in the **AWS IoT Device Management** service. **AWS IoT Device Management** handles scheduling, retrying, and reporting the status of remote operations on the robot devices, ensuring reliable and scalable model deployment.

9 **AWS IoT Device Management** publishes the AWS IoT Job to **AWS IoT Core**, the managed message broker service that distributes the job information to the connected robot devices.

10 The robotic devices subscribe to job notifications from **AWS IoT Core**. The robotics team implements the logic to retrieve and process robot-side job notifications.

11 The robotic devices download the newly trained model from **Amazon S3** and deploy it locally on the robot hardware, completing the loop of simulation-based training and real-world deployment.

12 The reinforcement learning pipeline uses **AWS IAM** roles to secure the training flow between **Amazon EKS**, **Amazon Bedrock** (LLM reward functions), **Amazon SQS**, and **Amazon S3** Mountpoint for data storage. The deployment flow leverages **AWS IAM** to enable **AWS Lambda**'s model deployment pipeline and IoT thing policies for robot authentication when accessing **AWS IoT Core** and **Amazon S3**.

Reviewed for technical October 28, 2025

AWS Reference Architecture