CLOUD LEARNERS INC.

IAM SECURITY TRAINING CHATBOT

AWS RE/START PROGRAM
GROUP 6: AWS LEX CHATBOT PROJECT

DATE: 18 DECEMBER 2025

THE CHALLENGE WE FACED

THE PROBLEM

- Students confused **Users** **vs.** **Roles**.
- Struggled with **AuthN** **vs.** **AuthZ** (Login vs. Permission).
- Labs often resulted in "Admin Access" over-permissioning.
- Traditional lectures were passive and low-retention.

OUR MISSION

Build an **engaging, interactive quiz bot** that clarifies these exact confusion points through real-world scenarios.

"Don't just define IAM. Make them use it."

WHY WE CHOSE IAM (OVER S3)

THE STATISTIC

40% of cloud security breaches stem from IAM misconfiguration.

THE RELEVANCE

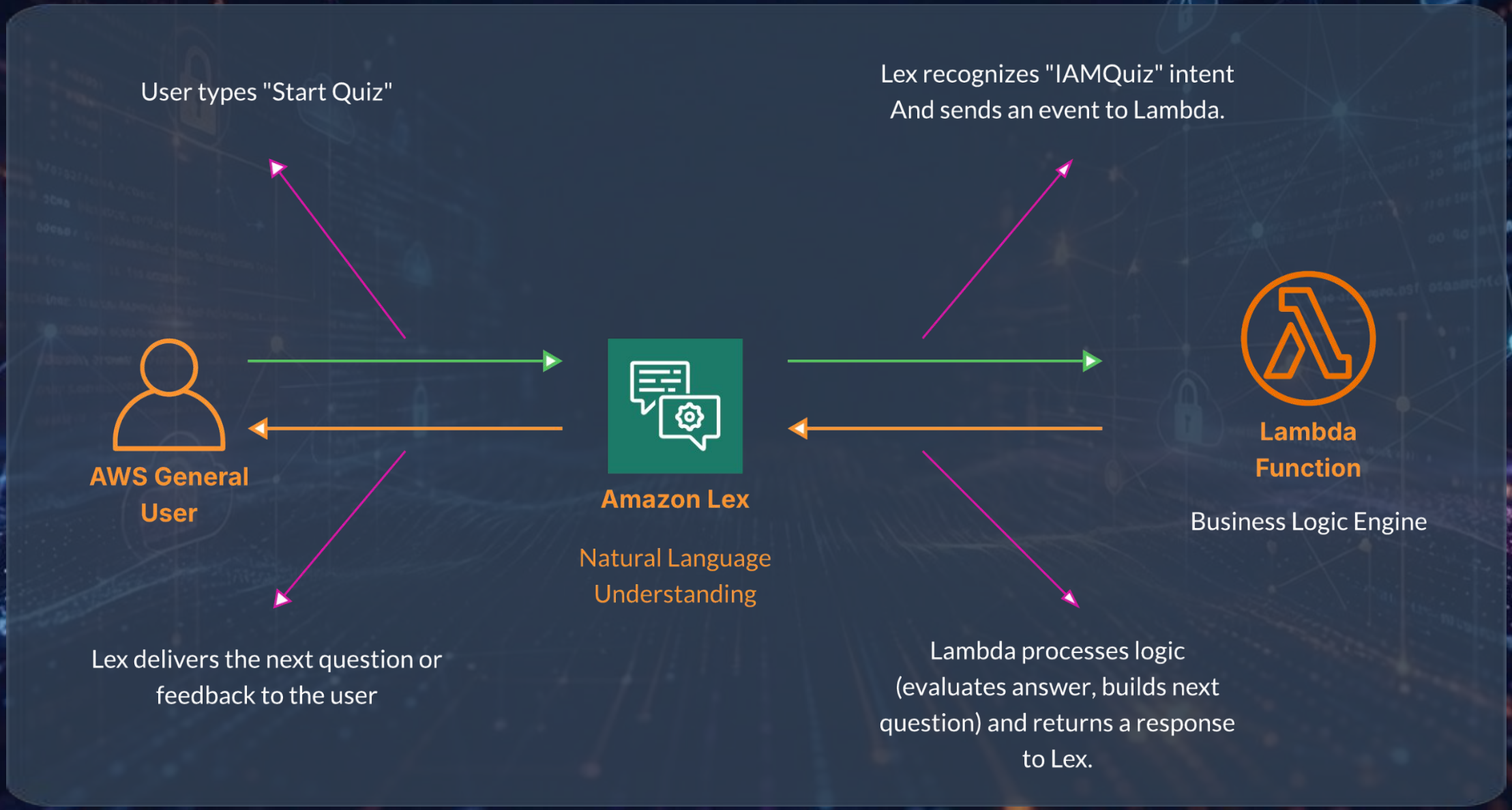
Every single AWS user touches IAM daily. You can avoid S3, but you cannot avoid Identity.

THE LEARNING GAP

Beginners constantly confuse "Who you are" with "What you can do".

Result: A quiz that is immediately applicable to production safety.

SOLUTION ARCHITECTURE



LEX = THE MOUTH | LAMBDA = THE BRAIN

Serverless Design

- **Amazon Lex V2:** The "Mouth" (Natural Language Understanding). Handles intent recognition.
- **AWS Lambda (Node.js):** The "Brain" (Business Logic). Handles scoring, branching, and validation.
- **Scalability:** Zero infrastructure to manage. Infinite scale.

IMPLEMENTATION CHALLENGES

CHALLENGE	THE SOLUTION
Lambda Permission Denied Bot gave "Intent Fulfillment Failed" error.	✓ Configured **Resource-based Policy** to explicitly allow lex.amazonaws.com to invoke the function.
Messy User Input Users typed "answer is B" instead of "B".	✓ Implemented **Input Normalization** (Regex) in Node.js to extract the answer letter.
Debugging Blindness Silent failures in the cloud.	✓ Added deep **CloudWatch Logging** at every logic branch to trace execution.

HOW THE "BRAIN" WORKS

BRANCHING LOGIC

The core requirement was to handle Right/Wrong answers differently. We achieved this via conditional logic in Lambda.

- **IF Correct:** Praise user → Ask next question.
- **IF Incorrect:** Explain *why* it's wrong → Offer retry (Game Over).

```
const ans1 = normalizeAnswer(slots.Answer1.value.interpretedValue);

if (ans1 !== 'B') {
  // WRONG PATH: Explain why and offer a retry
  return close('IAMQuiz',
    '❌ Not quite right. The correct answer was B (IAM Role)...'
  );
} else {
  // RIGHT PATH: Congratulate and ask Question 2
  return elicitSlot('IAMQuiz', slots, 'Answer2',
    '✅ Excellent! IAM Roles are the secure way...'
  );
}
```

LIVE DEMO

Switching to AWS Console...

HONEST REFLECTION

WHAT WE'D KEEP

- Scenario-based questions (Real world context).
- Serverless architecture (Simple, cheap).
- The IAM topic choice.

WHAT WE'D CHANGE

Start with Logs: We added CloudWatch late; should be Day 1.

Test Earlier: Don't wait until Question 3 to test Question 1.

User Testing: Test input variations sooner.

FUTURE ENHANCEMENTS

PHASE 2 — **Expand Content:** Add 10 questions covering MFA, Policies, and Cross-Account access.

PHASE 3 — **Capabilities:** Voice interaction ("Alexa, start quiz") and Certificate generation.

PHASE 4 — **Integration:** Connect to LMS (Canvas/Moodle) via API Gateway for grade tracking.

KEY TAKEAWAYS

CONSTRAINTS ACCELERATE LEARNING:

Part 1's "one utterance" rule forced us to master the basics before adding complexity.

BRANCHING IS THE DIFFERENTIATOR:

Simply answering questions is a search engine. *Teaching* based on wrong answers is an educational tool.

REAL-WORLD > DEFINITIONS:

We didn't ask "What is IAM?". We asked "Your EC2 needs S3 access...". Context creates memory.


THANK
YOU!
ANY QUESTIONS?