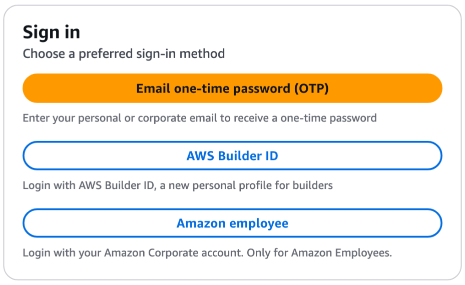
Generative AI and Data Day Workshop (London)

Generative AI Use Cases with Aurora PostgreSQL and pgvector

1. One Click link to access the workshop - https://catalog.us-east-1.prod.workshops.aws/join

2. Select one-time passcode if you don't have AWS Builder ID



Sign in
Choose a preferred sign-in method

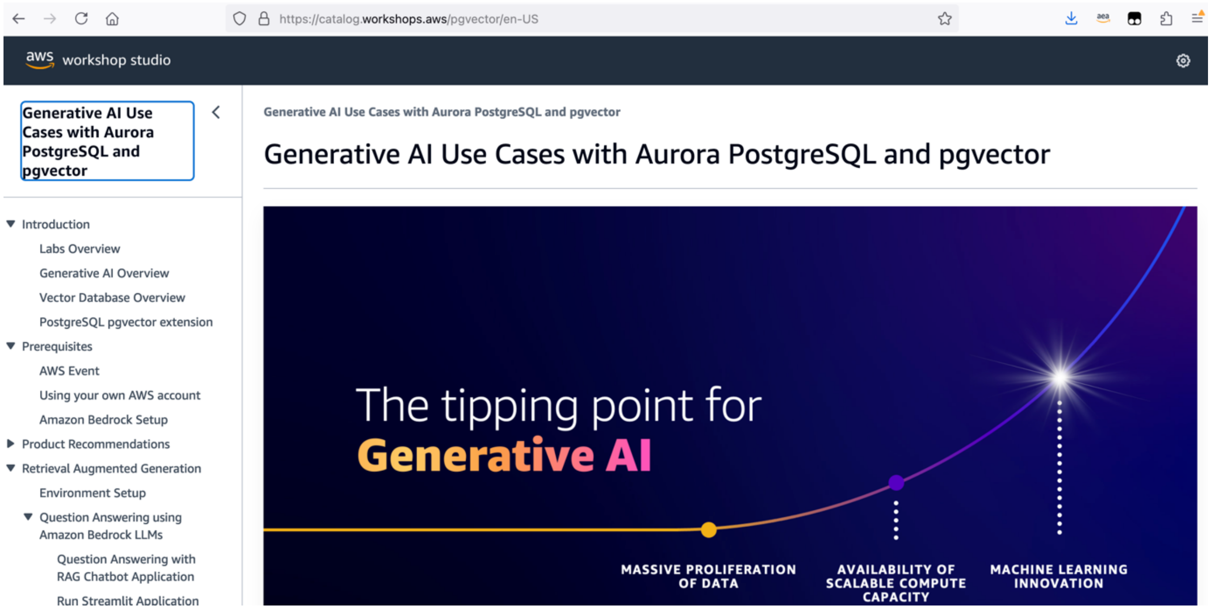
Email one-time password (OTP)
Enter your personal or corporate email to receive a one-time password

AWS Builder ID
Login with AWS Builder ID, a new personal profile for builders

Amazon employee
Login with your Amazon Corporate account. Only for Amazon Employees.

3. Event Access code will be provided separately. Please ask the AWS staff if you can't get the access code.

After accepting the agreement, you should see the workshop environment



https://catalog.workshops.aws/pgvector/en-US

aws workshop studio

Generative AI Use Cases with Aurora PostgreSQL and pgvector

- Introduction
  - Labs Overview
  - Generative AI Overview
  - Vector Database Overview
  - PostgreSQL pgvector extension
- Prerequisites
  - AWS Event
  - Using your own AWS account
  - Amazon Bedrock Setup
- Product Recommendations
- Retrieval Augmented Generation
  - Environment Setup
  - Question Answering using Amazon Bedrock LLMs
    - Question Answering with RAG Chatbot Application
    - Run Streamlit Application

Generative AI Use Cases with Aurora PostgreSQL and pgvector

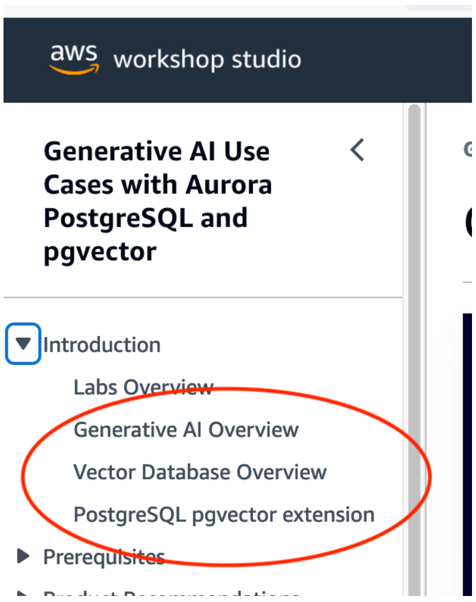
Generative AI Use Cases with Aurora PostgreSQL and pgvector

The tipping point for Generative AI
MASSIVE PROLIFERATION OF DATA
AVAILABILITY OF SCALABLE COMPUTE CAPACITY
MACHINE LEARNING INNOVATION

README First: There are 5 major labs in this workshop but you only need to do the first use case of 'Retrieval Augmented Generation' (RAG) which is **Question Answering using Amazon Bedrock LLMs** and the lab of **Knowledge Bases for Amazon Bedrock with Aurora PostgreSQL.** Each of these labs will take you roughly 30-45 mins apart from the introduction and Prerequisites. You can also just complete one of these labs based on your preference (basic RAG with LangChain or managed RAG with Bedrock Knowledge Bases) as there is no dependency between the 2 labs.

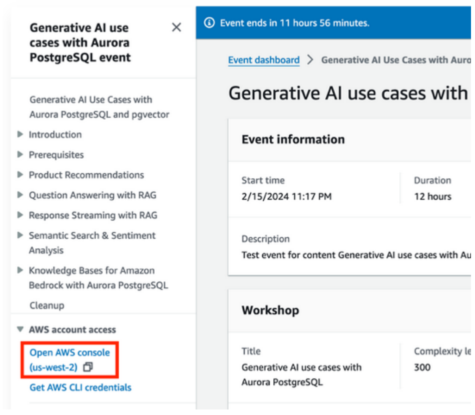
Task 1 - Introduction (~15 minutes)

Go through the introduction and basic concept of GenAI, Vector Database and pgvector extension if you are not familiar with these topics. **You can skip it if you think you have fair amount of knowledge on these topics**.



Task 2 Prerequisites (5-10 minutes)

- Click **Open AWS console** on the left navigation bar to log into the AWS Management Console.



Generative AI use cases with Aurora PostgreSQL event

Generative AI Use Cases with Aurora PostgreSQL and pgvector
- Introduction
- Prerequisites
- Product Recommendations
- Question Answering with RAG
- Response Streaming with RAG
- Semantic Search & Sentiment Analysis
- Knowledge Bases for Amazon Bedrock with Aurora PostgreSQL
- Cleanup

AWS account access
- Open AWS console (us-west-2)
- Get AWS CLI credentials

Event ends in 11 hours 56 minutes.

Event dashboard > Generative AI Use Cases with Auro

Generative AI use cases with

Event information

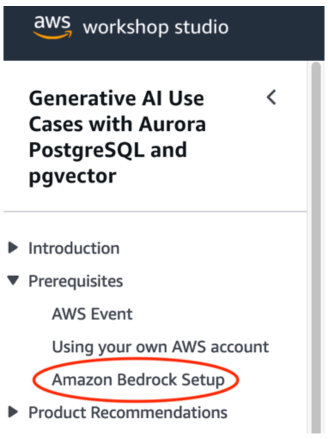
Start time: 2/15/2024 11:17 PM
Duration: 12 hours

Description: Test event for content Generative AI use cases with Au

Workshop

Title: Generative AI use cases with Aurora PostgreSQL
Complexity le: 300

- Go to Amazon Bedrock setup and requested the required models



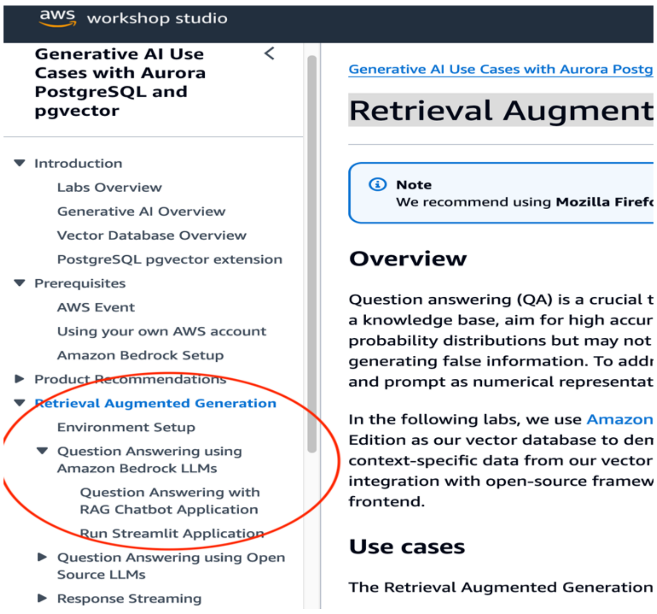
aws workshop studio

Generative AI Use Cases with Aurora PostgreSQL and pgvector

- Introduction
- Prerequisites
  - AWS Event
  - Using your own AWS account
  - Amazon Bedrock Setup
- Product Recommendations

Task 3 - Retrieval Augmented Generation (RAG) - Question Answering using Amazon Bedrock LLMs (30-45 minutes)

- For this lab, you will experience how to build Generative AI application with Retrieval Augmented Generation (RAG) using opensource framework LangChain and popular UI Streamlit connecting with LLM hosted in Amazon Bedrock.
- From this example app, you will experience the code required to synchronise the data to a Vector stored in PostgreSQL and how to use LangChain framework to initiate the search against the Vector and pass the result to LLM to provide the result and manage the conversational state.
- You **don't need** to complete the labs for Question Answering using Open Source LLMs and Response Streaming but if you want to do so, you may need another 30-45 minutes.


aws workshop studio
Generative AI Use Cases with Aurora PostgreSQL and pgvector
Introduction
Labs Overview
Generative AI Overview
Vector Database Overview
PostgreSQL pgvector extension
Prerequisites
AWS Event
Using your own AWS account
Amazon Bedrock Setup
Product Recommendations
Retrieval Augmented Generation
Environment Setup
Question Answering using Amazon Bedrock LLMs
Question Answering with RAG Chatbot Application
Run Streamlit Application
Question Answering using Open Source LLMs
Response Streaming
Generative AI Use Cases with Aurora Postg

Retrieval Augment

Note
We recommend using **Mozilla Firefc**

Overview

Question answering (QA) is a crucial t
a knowledge base, aim for high accur
probability distributions but may not
generating false information. To addr
and prompt as numerical representat

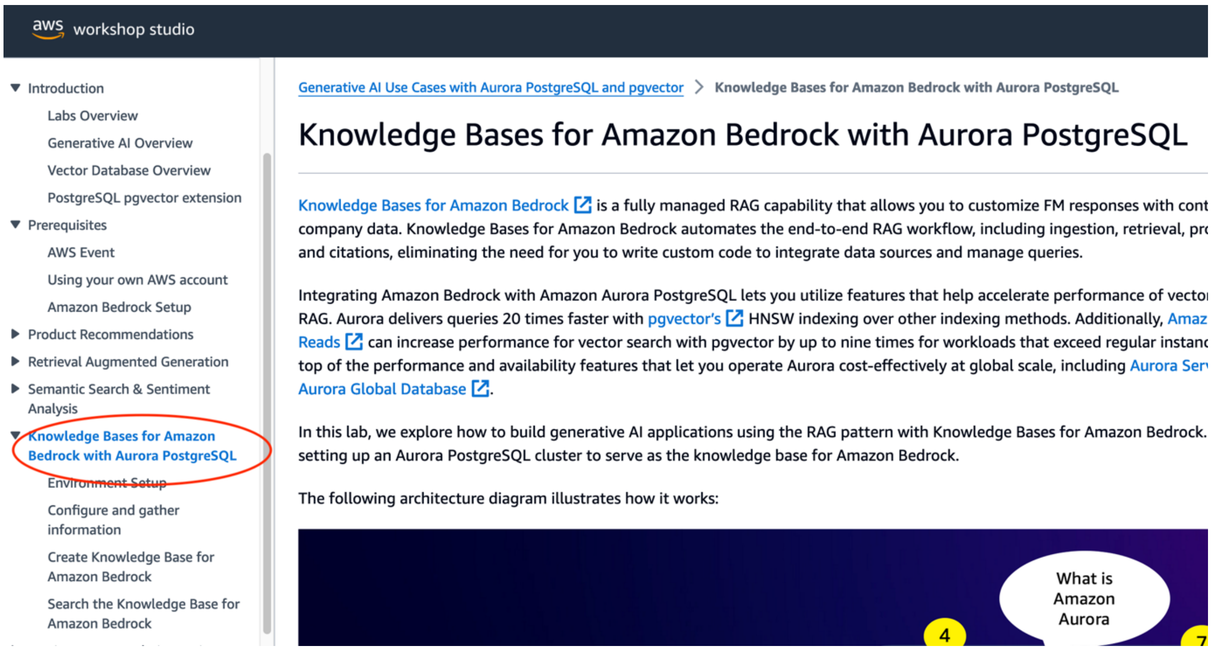
In the following labs, we use Amazon
Edition as our vector database to den
context-specific data from our vector
integration with open-source framew
frontend.

Use cases

The Retrieval Augmented Generation

Task 4 - Knowledge Bases for Amazon Bedrock with Aurora PostgreSQL (30-45 minutes)

- For this lab, you will also build a RAG Generative AI app using Vector data stored in PostgreSQL. The major difference from Task 3 is it will leverage the managed RAG capability via Amazon Bedrock and you don't need to rely on other frameworks like LangChain to build the RAG app and you don't need to deal with the complexity or write custom code for Vector Data synchronisation and queries management. The lab will use Amazon Bedrock native test interface instead of a web GUI like StreamLit but this can be easily built with any popular UI like Streamlit or Gradio.


aws workshop studio
Introduction
Labs Overview
Generative AI Overview
Vector Database Overview
PostgreSQL pgvector extension
Prerequisites
AWS Event
Using your own AWS account
Amazon Bedrock Setup
Product Recommendations
Retrieval Augmented Generation
Semantic Search & Sentiment Analysis
Knowledge Bases for Amazon Bedrock with Aurora PostgreSQL
Environment Setup
Configure and gather information
Create Knowledge Base for Amazon Bedrock
Search the Knowledge Base for Amazon Bedrock
Generative AI Use Cases with Aurora PostgreSQL and pgvector > Knowledge Bases for Amazon Bedrock with Aurora PostgreSQL
What is Amazon Aurora
4
7

Knowledge Bases for Amazon Bedrock with Aurora PostgreSQL

Knowledge Bases for Amazon Bedrock is a fully managed RAG capability that allows you to customize FM responses with cont
company data. Knowledge Bases for Amazon Bedrock automates the end-to-end RAG workflow, including ingestion, retrieval, pr
and citations, eliminating the need for you to write custom code to integrate data sources and manage queries.

Integrating Amazon Bedrock with Amazon Aurora PostgreSQL lets you utilize features that help accelerate performance of vecto
RAG. Aurora delivers queries 20 times faster with pgvector's HNSW indexing over other indexing methods. Additionally, Amaz
Reads can increase performance for vector search with pgvector by up to nine times for workloads that exceed regular instanc
top of the performance and availability features that let you operate Aurora cost-effectively at global scale, including Aurora Ser
Aurora Global Database.

In this lab, we explore how to build generative AI applications using the RAG pattern with Knowledge Bases for Amazon Bedrock.
setting up an Aurora PostgreSQL cluster to serve as the knowledge base for Amazon Bedrock.

The following architecture diagram illustrates how it works: