Building Agentic Generative AI Apps with MongoDB Atlas on AWS

Aychin Gasimov

Senior Partner Solutions Architect
Amazon Web Services (AWS)

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Better Together: AWS and MongoDB

Competencies/Programs

- Automotive Software Competency
- Data & Analytics Software Competency
- Financial Services Software Competency
- Generative AI Competency
- Government ISV Competency
- Small and Medium Business Software Competency
- AWS Marketplace Seller
- AWS Public Sector Partner
- AWS PrivateLink Ready Product

Joint Integrations

AWS PrivateLink

Amazon EventBridge

AWS Key Management Service

AWS Lambda

AWS IAM Identity Center

Amazon Bedrock

Amazon Kinesis Data Firehose

Amazon Simple Storage Service

Amazon SageMaker

Amazon Managed Streaming for Apache Kafka

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Harness the power of generative AI with MongoDB and AWS

Speed up application development with MongoDB's flexible developer data platform, allowing you to stay at the forefront of AI innovation, deliver unparalleled user experiences, and remain agile in a rapidly changing technological landscape, all while harnessing the capabilities of AWS and MongoDB Atlas.

Amazon Q Developer with MongoDB
Streamlines development workflow, integrates with popular IDEs, bolsters developer productivity by offering intuitive code snippets, pinpoint bugs, and upholding best practices for effective coding process.

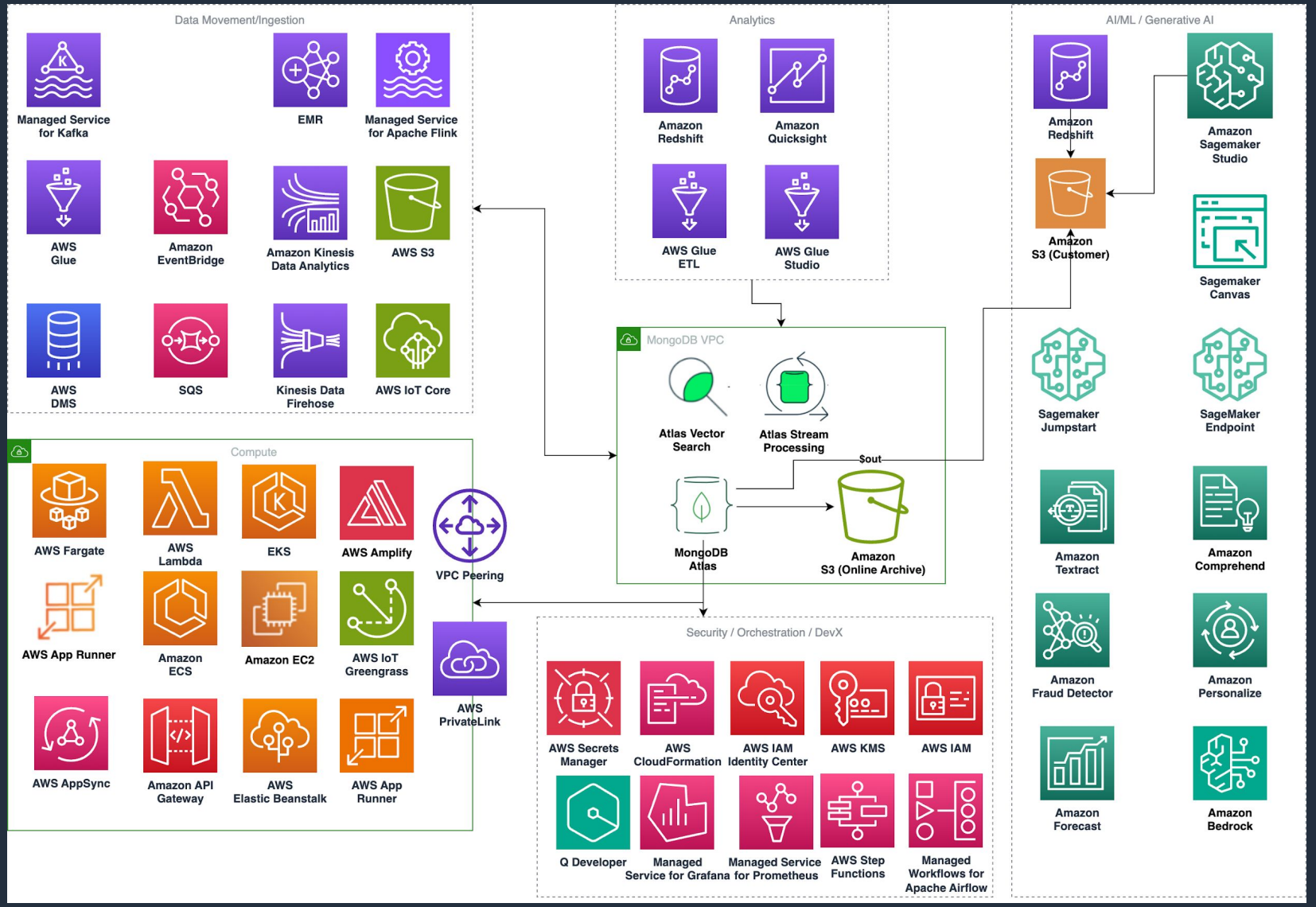
MongoDB Vector Search and Amazon SageMaker Jumpstart
Deploy variety of open-source, public, and private models to power your GenAI applications with MongoDB Atlas Vector Search on AWS.

MongoDB Vector Search and Amazon BedRock
Build your GenAI applications easily with Amazon Bedrock and MongoDB Atlas Vector Search on AWS. Take advantage of Bedrock's privately customizable FMs to power you generative AI applications. Use Titan Embeddings and MongoDB Atlas for personalization and semantic search.

aws

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

MongoDB Atlas on AWS



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Generative AI: Trends through the years

2023
The year of POCs

- What is generative AI?
- Is this secure?
- Do I need to become a prompt engineer?
- How do I choose a model?
- Where do I get started?

2024
The year of Production

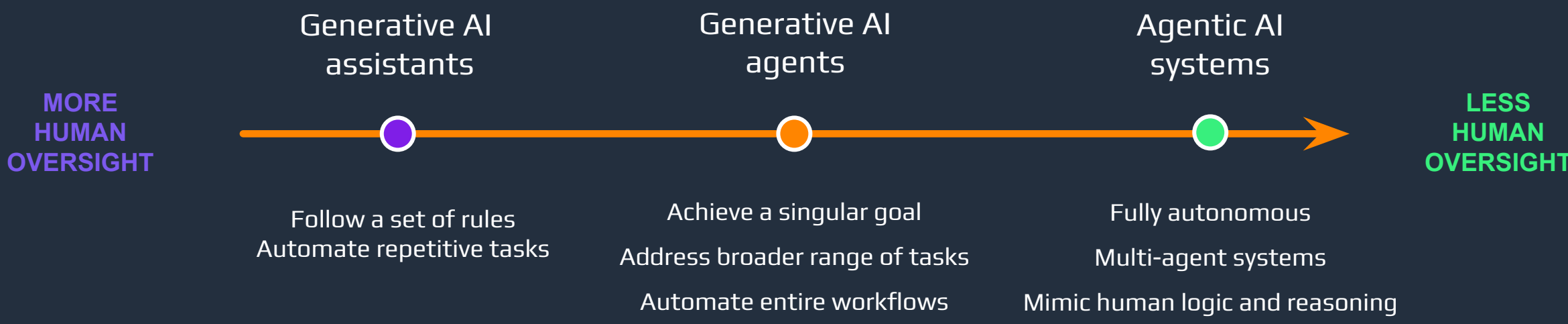
- How do I prioritize my projects?
- How can I lower my costs?
- How do I make this real?
- What customization method should I use?

2025
The year of Business Value

- How can I leverage agents?
- How can we go deeper *and* wider?
- How can we transform our entire business?
- How can I get *all* my employees to embrace AI?

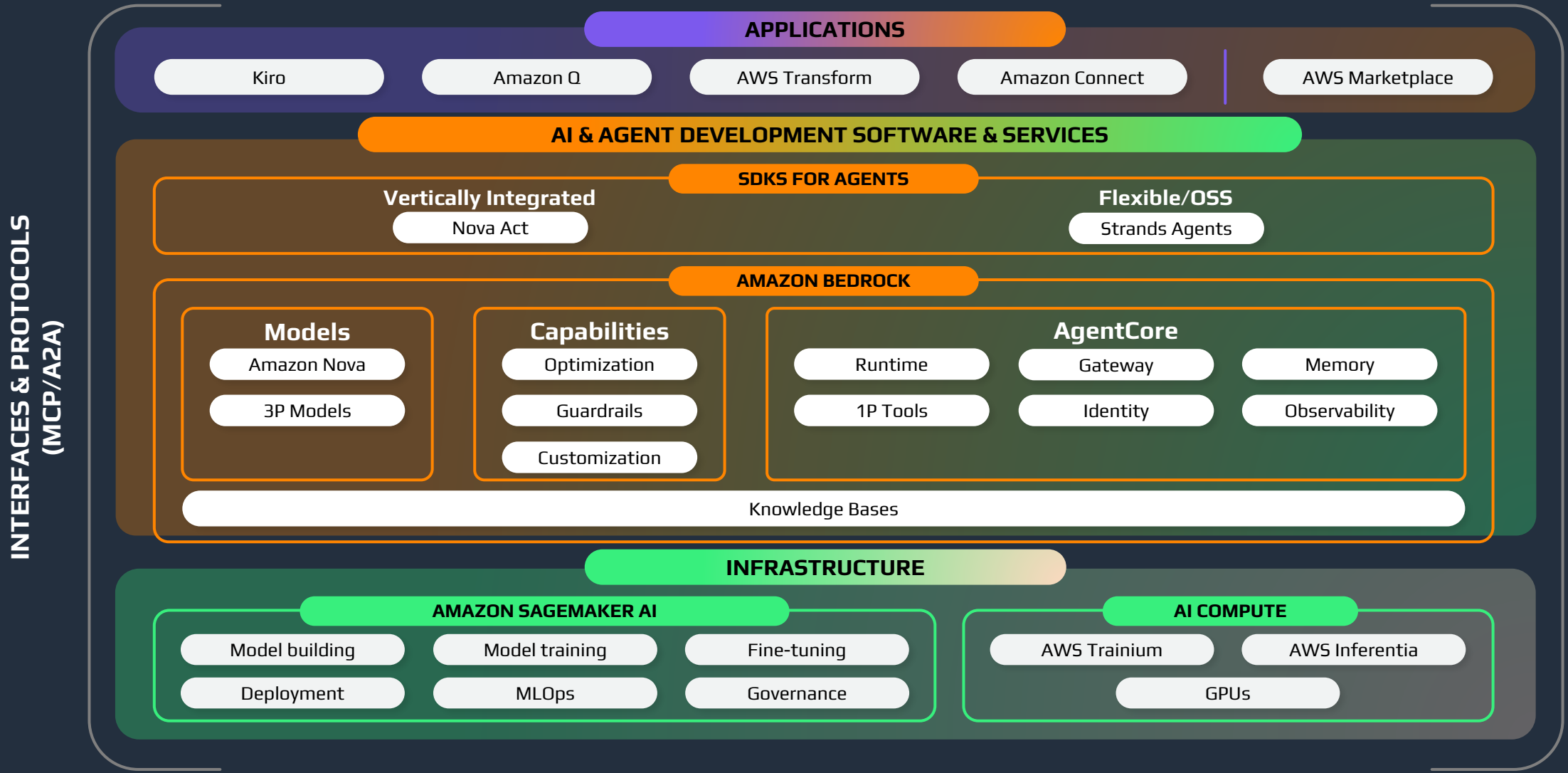
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

The evolution into Agentic AI



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS AI Stack



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon Q

Reinvent work with AWS' generative AI–powered assistant

Unites your data sources, repositories, and enterprise systems

Available where you work

Delivers industry-leading assistance

Reinvent work with Amazon Q

BUSINESS

Amazon Q Business

KNOWLEDGE SEARCH

SUMMARIZATION

CONTENT CREATION

EXTRACT INSIGHTS

RESEARCH & ANALYSIS

Amazon Q in QuickSight

UNDERSTAND DATA

BUILD & REFINE VISUALS

BUILD CALCULATIONS

EXECUTIVE SUMMARIES

CREATE DATA STORIES

DEVELOPERS

Amazon Q Developer

PLAN APPLICATION

CODE GENERATION

UNIT TESTING

SECURITY SCANNING

CODE REMEDIATION

CODE MIGRATION

TROUBLESHOOTING

DEVELOPER KNOWLEDGE

SPECIALIZED USERS

Amazon Q in Connect

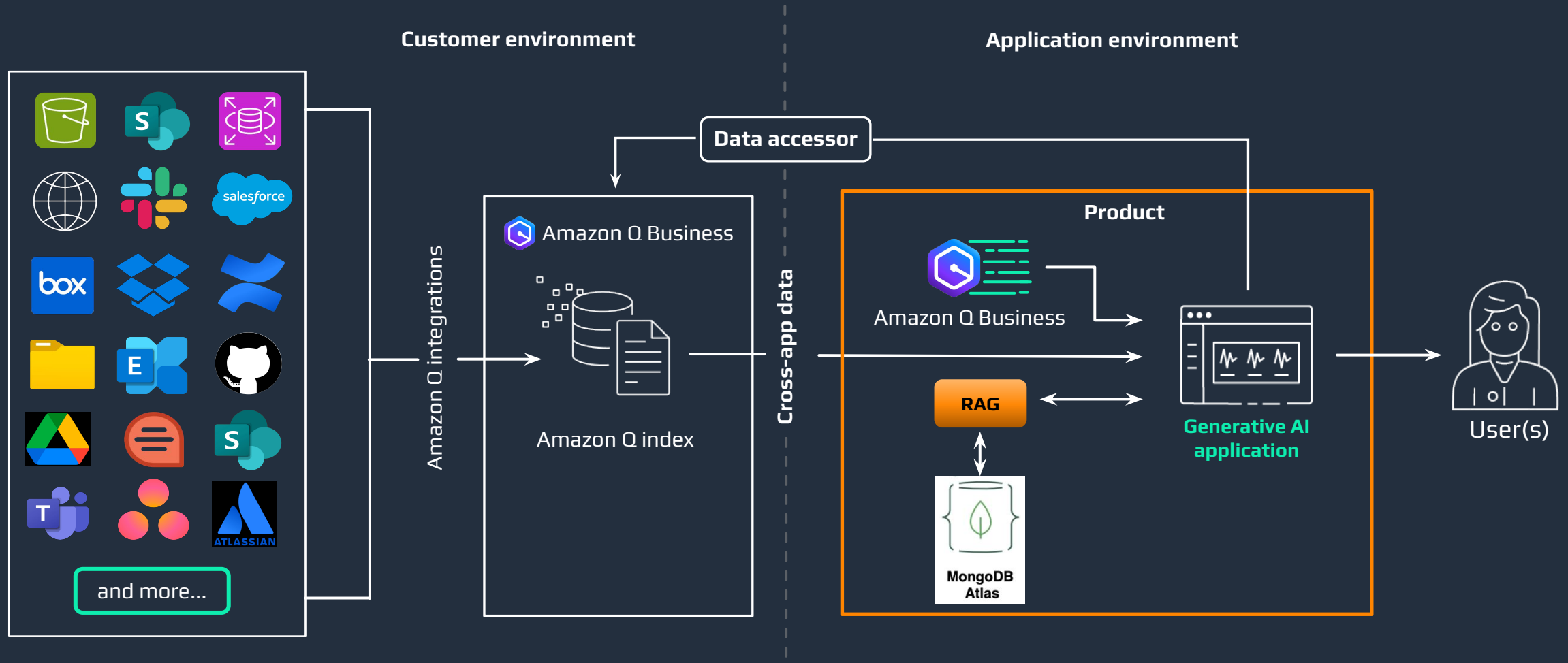
AGENT ASSIST

Amazon Q in AWS Supply Chain

SUPPLY CHAIN

The Amazon Q index

ADD CROSS-APP INSIGHTS TO YOUR GENERATIVE AI APPLICATIONS



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

The broadest choice for building and deploying agents

Agent Powered Applications

For Builders

For Business Users

Tools for Building AI Agents

Marketplace

Bedrock Agents

Strands

Nova Act SDK

Agentic AI Foundation

Amazon Bedrock AgentCore

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon Bedrock

The easiest way to build and scale generative AI applications with powerful tools and foundation models

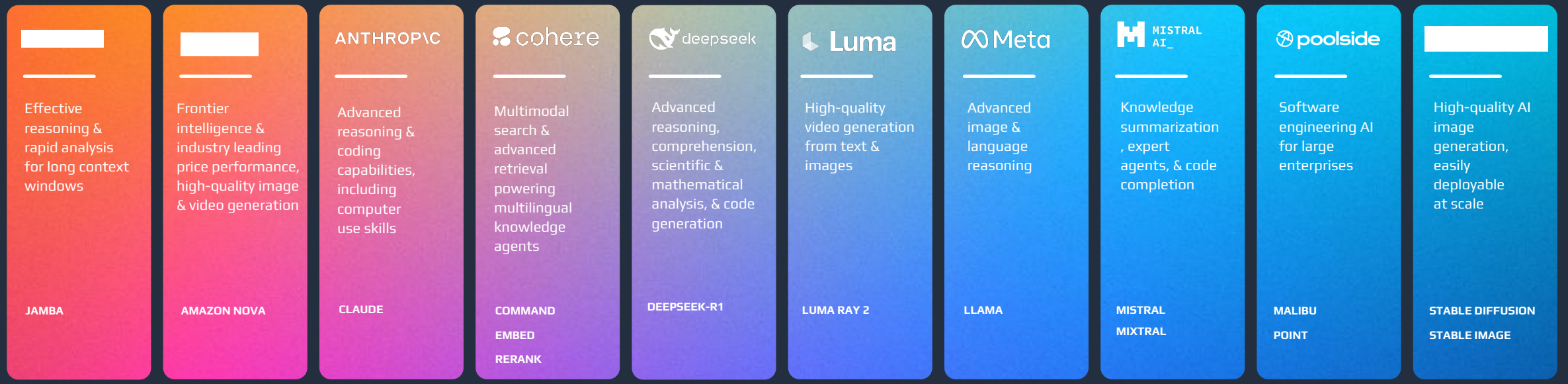
- Choice of leading FMs through a single API
- Optimization for cost, latency and accuracy
- Customization with your data
- Safety and responsible AI checks
- Agents that execute complex tasks

aws

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon Bedrock

BROAD CHOICE OF MODELS



Coming soon

Amazon Bedrock Marketplace enables developers to discover, test, and use over 100 popular, emerging, and specialized foundation models (FMs) alongside the current selection of industry-leading models in Amazon Bedrock**. DeepSeek-R1** model is now available in Amazon Bedrock Marketplace.

aws

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon Nova Foundation Models

State-of-the-art foundation models that deliver frontier intelligence and industry leading price performance.

Understanding models

Amazon Nova Micro

Our text only model that delivers the lowest latency responses at very low cost

GENERALLY AVAILABLE

Amazon Nova Lite

Our lowest cost multimodal model that is lightning fast for lightweight tasks

GENERALLY AVAILABLE

Amazon Nova Pro

Our highly capable multimodal model with best combination of accuracy, speed, and cost for a wide range of tasks

GENERALLY AVAILABLE

Amazon Nova Premier

Our most capable multimodal model for complex reasoning tasks and for use as the best teacher for distilling custom models

COMING SOON

Lower Cost & Latency

Increasing Intelligence

Creative content generation models

Amazon Nova Canvas

State-of-the-art image generation model

GENERALLY AVAILABLE

Amazon Nova Reel

State-of-the-art video generation model

GENERALLY AVAILABLE

Amazon Nova Sonic

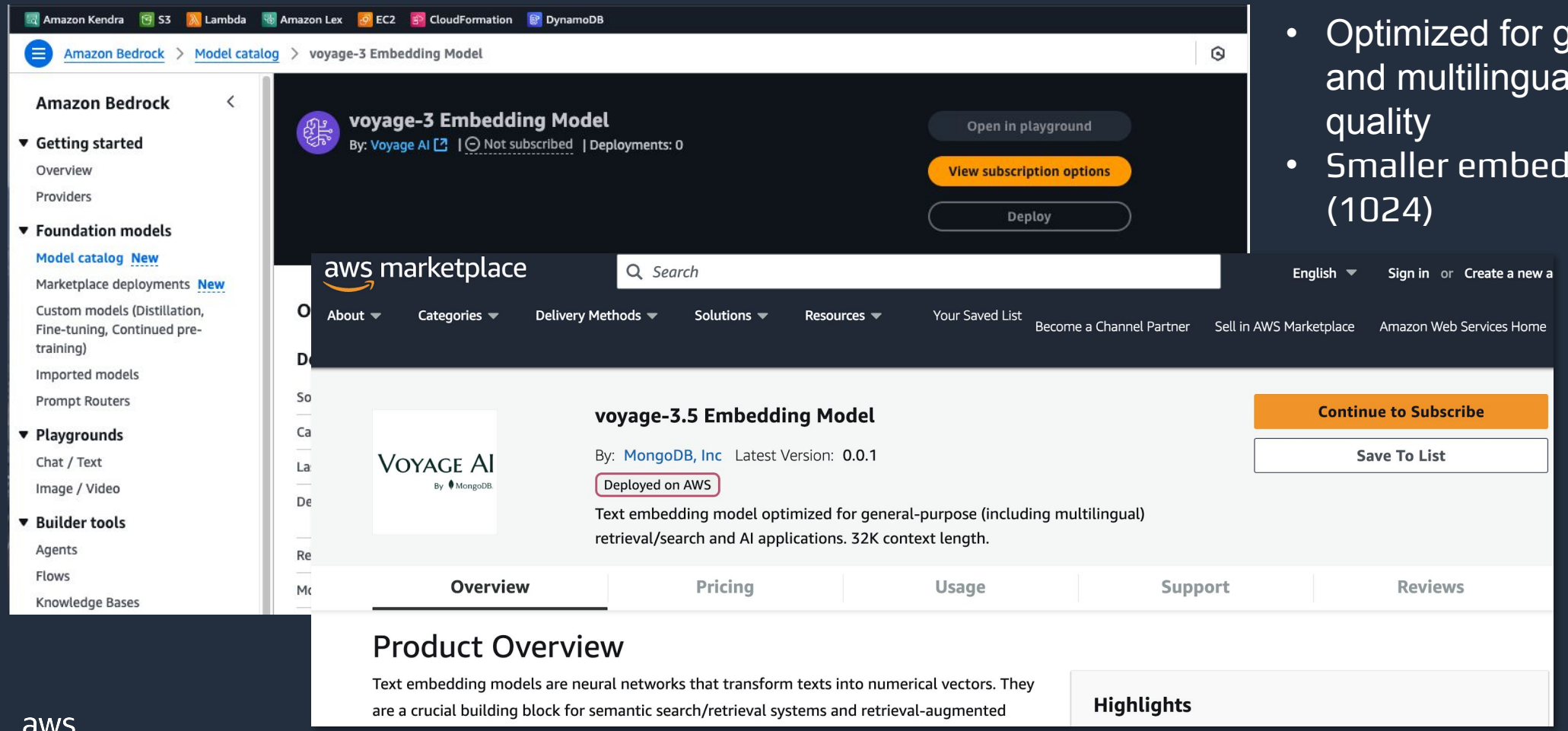
State-of-the-art speech-to-speach model

GENERALLY AVAILABLE

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Voyage.ai Embeddings Model in Bedrock Marketplace

AMAZON BEDROCK MARKETPLACE INCREASES AMAZON BEDROCK MODEL CHOICE WITH 100+ PUBLICLY AVAILABLE AND PROPRIETARY EMERGING, POPULAR, AND SPECIALIZED MODELS THROUGH A SERVERFUL OFFERING



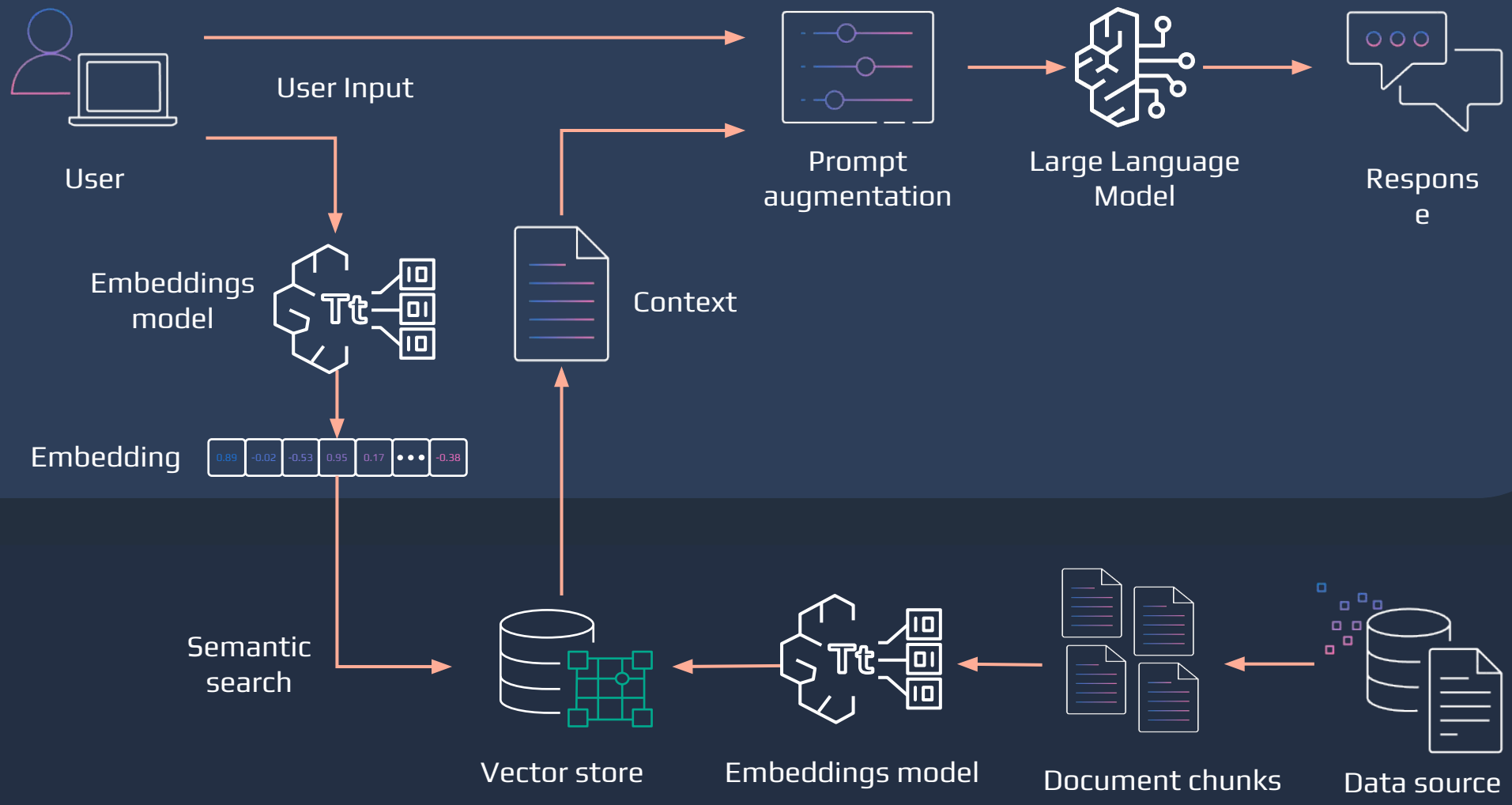
- Optimized for general-purpose and multilingual retrieval quality
- Smaller embedding dimension (1024)

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Everybody loves RAG!

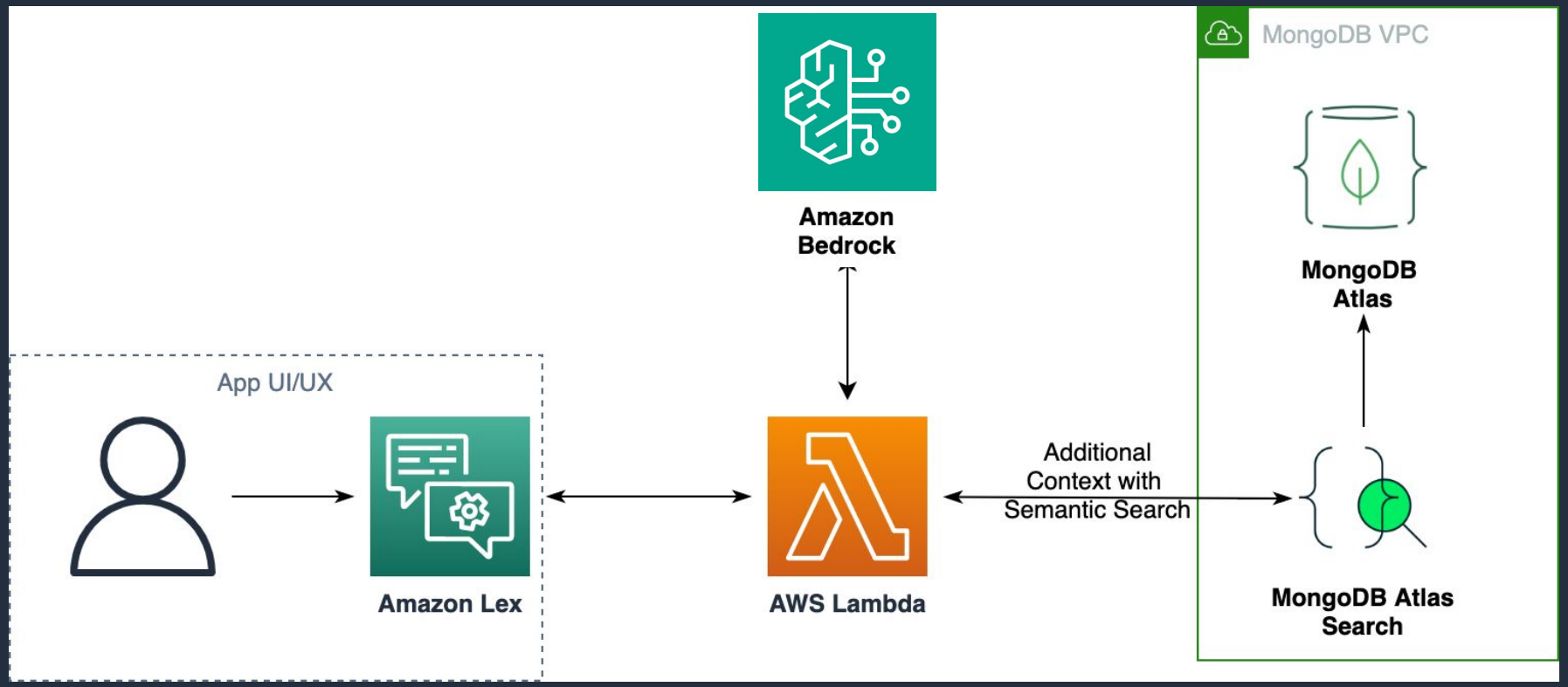
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

RAG in action



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

MongoDB Atlas and Amazon Bedrock architecture



- Simple bot
- Orchestration in Lambda
- Interaction with LLMs via Bedrock APIs
- Ingestion flow is a separate process

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Code example for Simple Bot

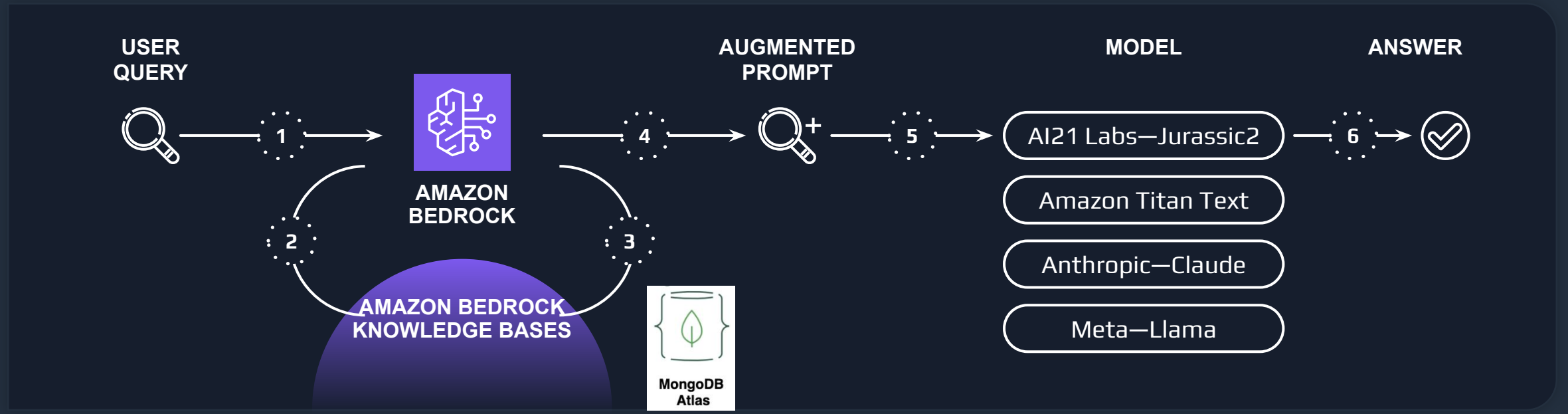
```python
def mdb_query(mdbclient, query, kcount):
    text_as_embeddings = embeddings.embed_documents([query])
    embedding_value = text_as_embeddings[0]
    print("embedding size: " + str(len(embedding_value)))

    # get the vector search results based on the filter conditions.
    response = collection.aggregate([
        {
            "$vectorSearch": {
                "index": "vector_index",
                "path": "eg_vector",
                "queryVector": text_as_embeddings[0],
                "numCandidates": 20,
                "limit": 4
            }
        }, {
            '$project': {
                'score': {'$meta': 'searchScore'},
                field_name_to_be_vectorized : 1,
                'title': 1,
                'claim_id' : 1,
                '_id':0
            }
        }
    ])
```

- Explicit Embeddings generation
- MongoDB API to search
- Need to handle ingestion

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon Bedrock Knowledge Bases

NATIVE SUPPORT FOR RAG



Securely connect FMs to data sources for RAG to deliver more relevant responses

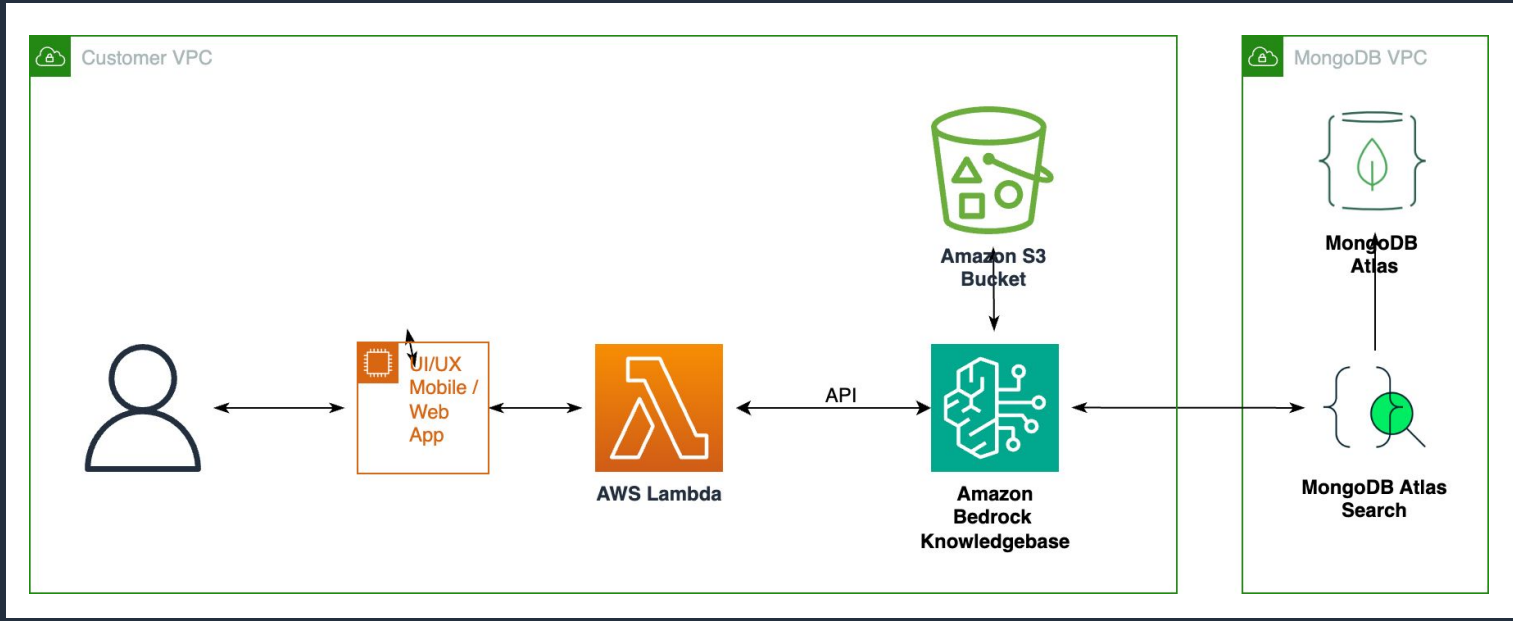
Fully managed RAG workflow including ingestion, retrieval, and augmentation

Built-in session context management for multiturn conversations

Automatic citations with retrievals to improve transparency

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

MongoDB Atlas Bedrock—Reference architecture KB Only



- Content/Knowledge Ingestion handled by Bedrock
- UX: Web or Mobile App
- Lambda interacts w/ Bedrock APIs
- Semantic Search using Atlas Vector Search
- Text and Hybrid Search with Atlas!

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Code example for Bedrock KB

```python
from langchain_aws import AmazonKnowledgeBasesRetriever

# initialize AmazonKnowledgeBaseRetriever
retriever = AmazonKnowledgeBasesRetriever(
    knowledge_base_id=os.environ.get("BEDROCK_KB_ID"),
    region_name="us-west-2",
    retrieval_config={
        "vectorSearchConfiguration": {
            "numberOfResults": 5,
            "overrideSearchType": "SEMANTIC",
        }
    },
)

# Streamed response emulator
def response_generator(query_string):
    res = retriever.invoke(query_string, region_name="us-west-2")

    print("finished search...")
```

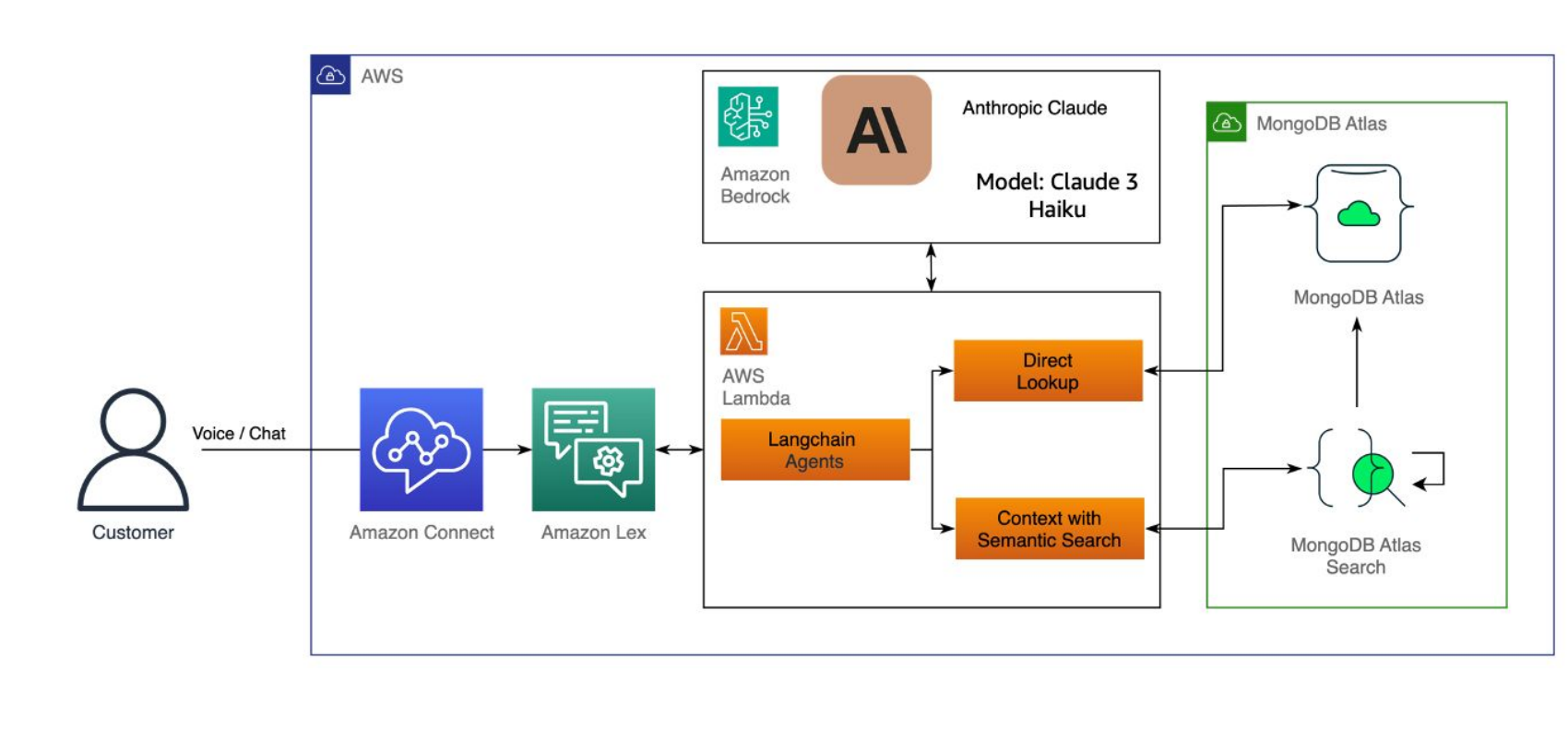
- No need for explicit Embeddings generation
- Simple API to search

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Let’s Build with Agents!

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

MongoDB Atlas, Amazon Bedrock, Langchain, and Anthropic



- More complex bot
- Orchestration in Lambda
- Langchain Agent/Tools
- Direct/Lexographical/Semantic Searches
- Interaction with LLMs via Bedrock APIs
- Ingestion flow is a separate process

https://github.com/mongodb-partners/QS-Anthropic-1

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Code Example for Langchain Agents

```python
def mongodb_place_lookup_by_name(query_str: str) -> str:
    """
    Retrieve place information by place name from the MongoDB database.

    This function connects to the MongoDB Atlas database and searches for a place
    using the provided name query. The search checks both the place name and country,
    using case-insensitive regex matching. The function returns the first matching
    document without the ID field.

    Args:
        query_str (str): The place name to search for. It may include partial matches.

    Returns:
        str: A string representation of the first matching document's details without the ID field.
        """
    res = ""
    filter = {
        "$or": [
            {"Place Name": {"$regex": query_str, "$options": "i"}},
            {"Country": {"$regex": query_str, "$options": "i"}},
        ]
    }
    project = {"_id": 0,"details_embedding":0}

    res = collection.find_one(filter=filter, projection=project)
    return str(res)
```

- Langchain constructs the tools for an agent from the functions

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

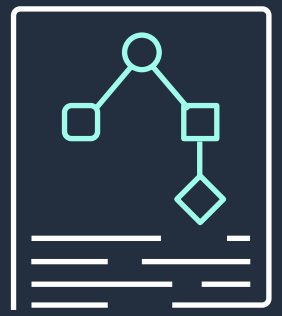
Amazon Bedrock Agents

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon Bedrock Agents

ENABLE GENERATIVE AI APPLICATIONS TO EXECUTE MULTISTEP TASKS USING COMPANY SYSTEMS AND DATA SOURCES

Decompose into steps using available actions and Amazon Bedrock Knowledge Bases



Execute action or search knowledge base

Observe results

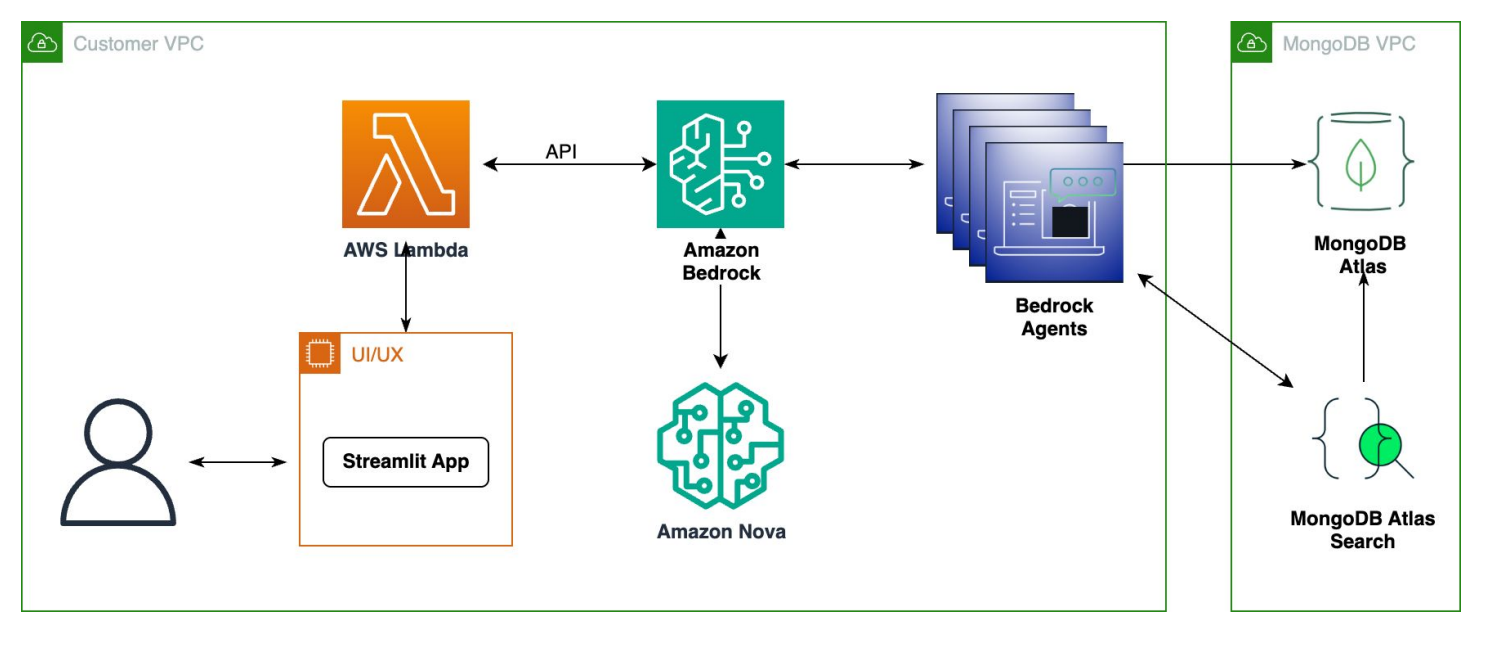
Think about next step

Until final answer

aws

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

MongoDB Atlas Bedrock—Reference architecture Agents Only



- Content/Knowledge Ingestion handled by Bedrock
- UX: Web or Mobile App
- Lambda interacts w/ Bedrock APIs
- Semantic Search using Atlas Vector Search

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

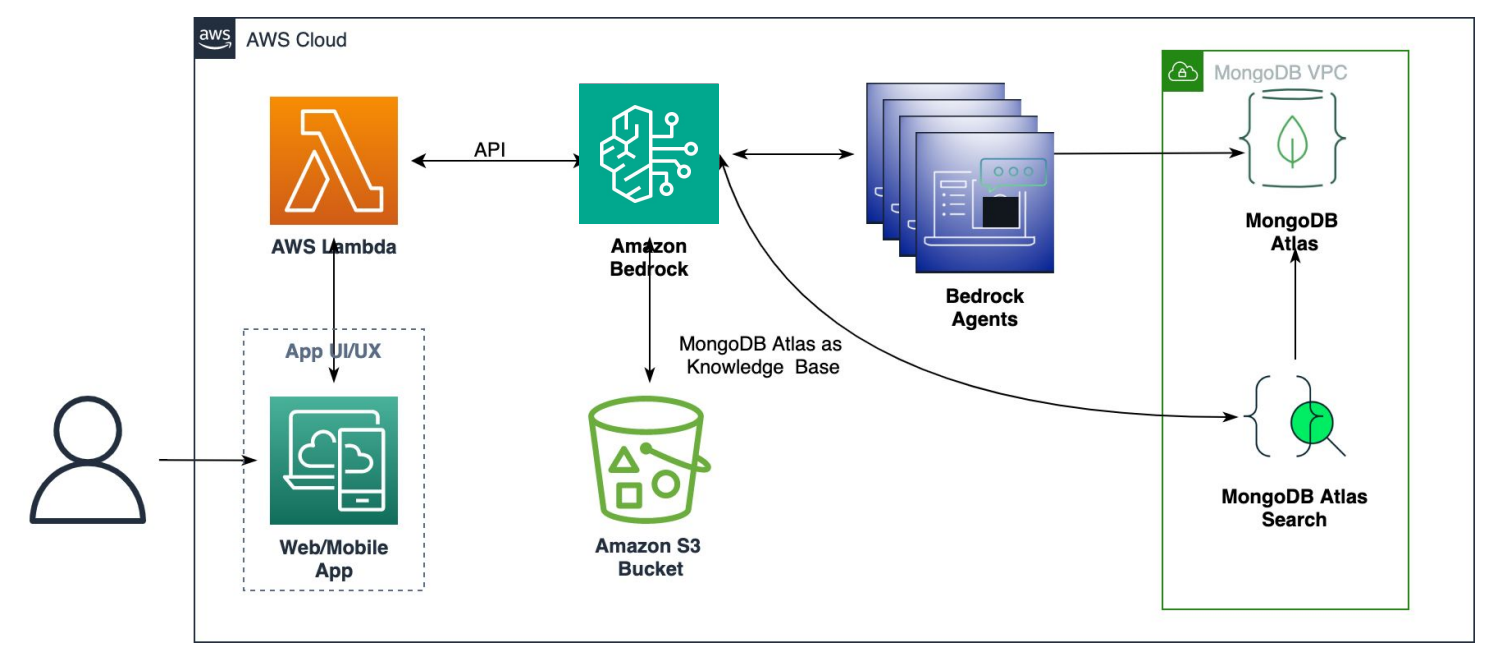
Code Example for Bedrock Agents

```python
@app.get("/get_place_by_country", description="Retrieve places by country name")
@tracer.capture_method
def place_lookup_by_country(query_str: Annotated[str, Query(description="The country name")]
                            ) -> Annotated[str, Body(description="Place Names")]:
    logger.info(f"Looking up places by country: {query_str}")
    client = get_mongo_client()
    # get database and collection
    collection = get_travel_collection(client)
    res = ""
    res = collection.aggregate(
        [
            {"$match": {"Country": {"$regex": query_str, "$options": "i"}}},
            {"$project": {"Place Name": 1}},
        ]
    )
    places = []
    for place in res:
        places.append(place["Place Name"])
    logger.info(f"Found {len(places)} places in country: {query_str}")
    return str(places)
```

- Lambda Power Tools framework to simplify development

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

MongoDB Atlas Bedrock—Reference architecture Combined



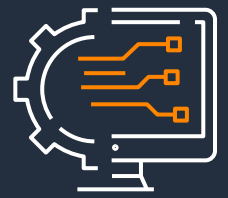
- Content/Knowledge Ingestion handled by Bedrock
- UX: Web or Mobile App
- Lambda interacts w/ Bedrock APIs
- Semantic Search using Atlas Vector Search

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

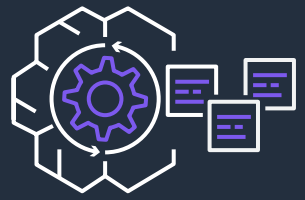
Amazon Bedrock AgentCore

© 2025, Amazon Web Services, Inc. or its affiliates. All rights reserved. Amazon Confidential and Trademark.

Essential tools and capabilities to build highly effective agents



Deploy securely at scale



Enhance with tools and memory

AgentCore Gateway

AgentCore Memory

AgentCore Browser

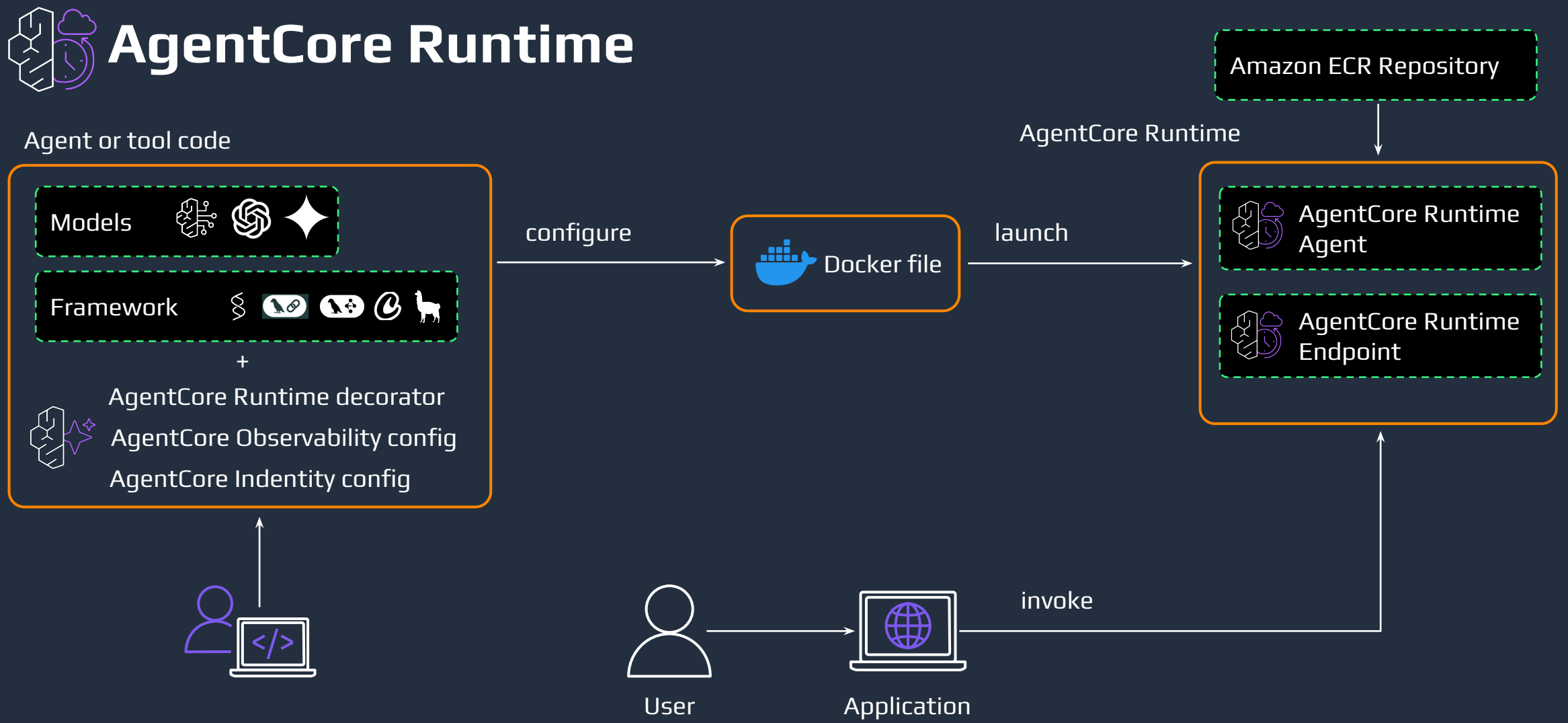
AgentCore Code Interpreter

Monitor

aws

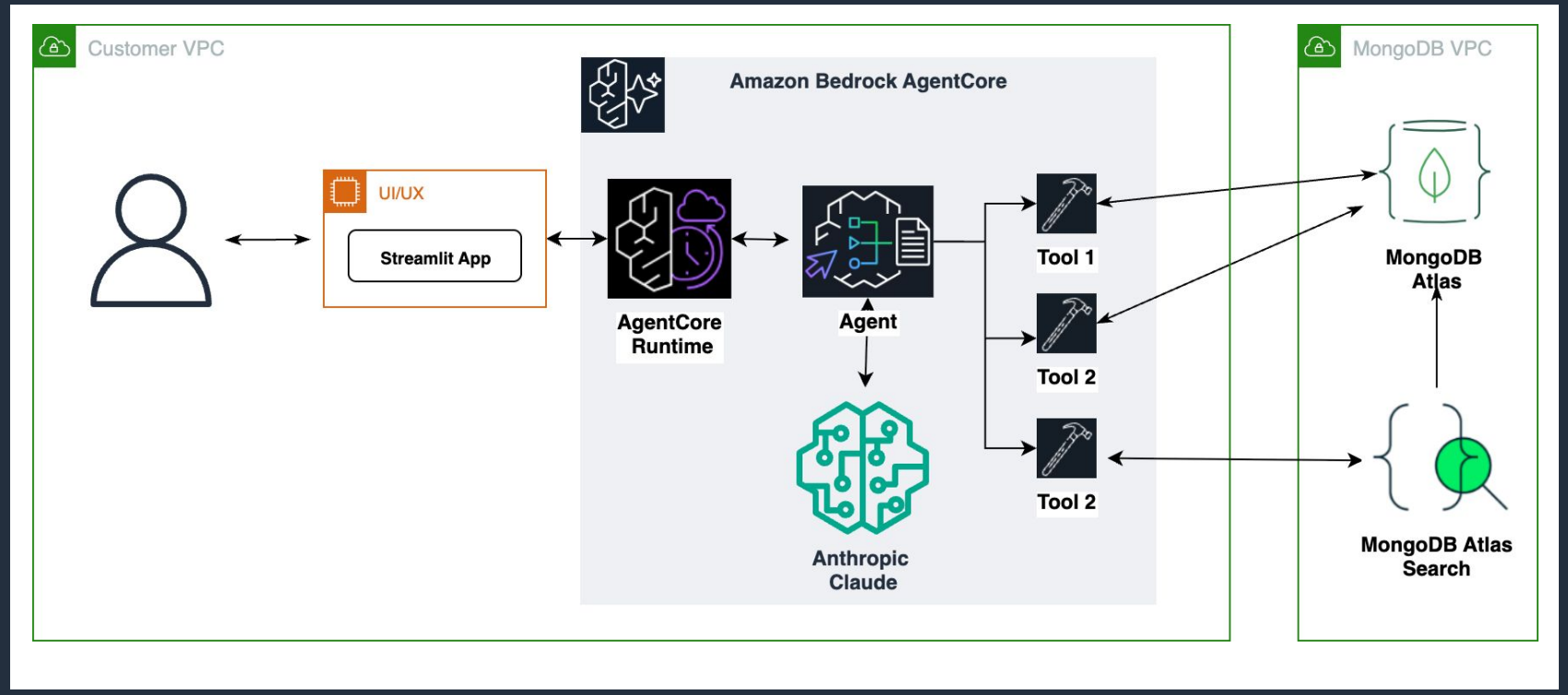
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AgentCore Runtime



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Bedrock AgentCore—Reference architecture



- Compute is managed by AgentCore
- UX: Web or Mobile App

- Semantic Search using Atlas Vector Search

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Code Example for Bedrock AgentCore Strands

```python
@tool
def place_lookup_by_country(query_str: str) -> str:
    """Retrieve places by country name

    Args:
        query_str: The country name to search for

    Returns:
        List of place names in the specified country
    """
    logger.info(f"Looking up places by country: {query_str}")
    client = get_mongo_client()
    # get database and collection
    collection = get_travel_collection(client)
    res = ""
    res = collection.aggregate(
        [
            {"$match": {"Country": {"$regex": query_str, "$options": "i"}}},
            {"$project": {"Place Name": 1}},
        ]
    )
    places = []
    for place in res:
        places.append(place["Place Name"])
    logger.info(f"Found {len(places)} places in country: {query_str}")
    return str(places)
```

- Annotation to define a tool
- No Lambda
- Similar to other orchestrations
- Easy migration

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

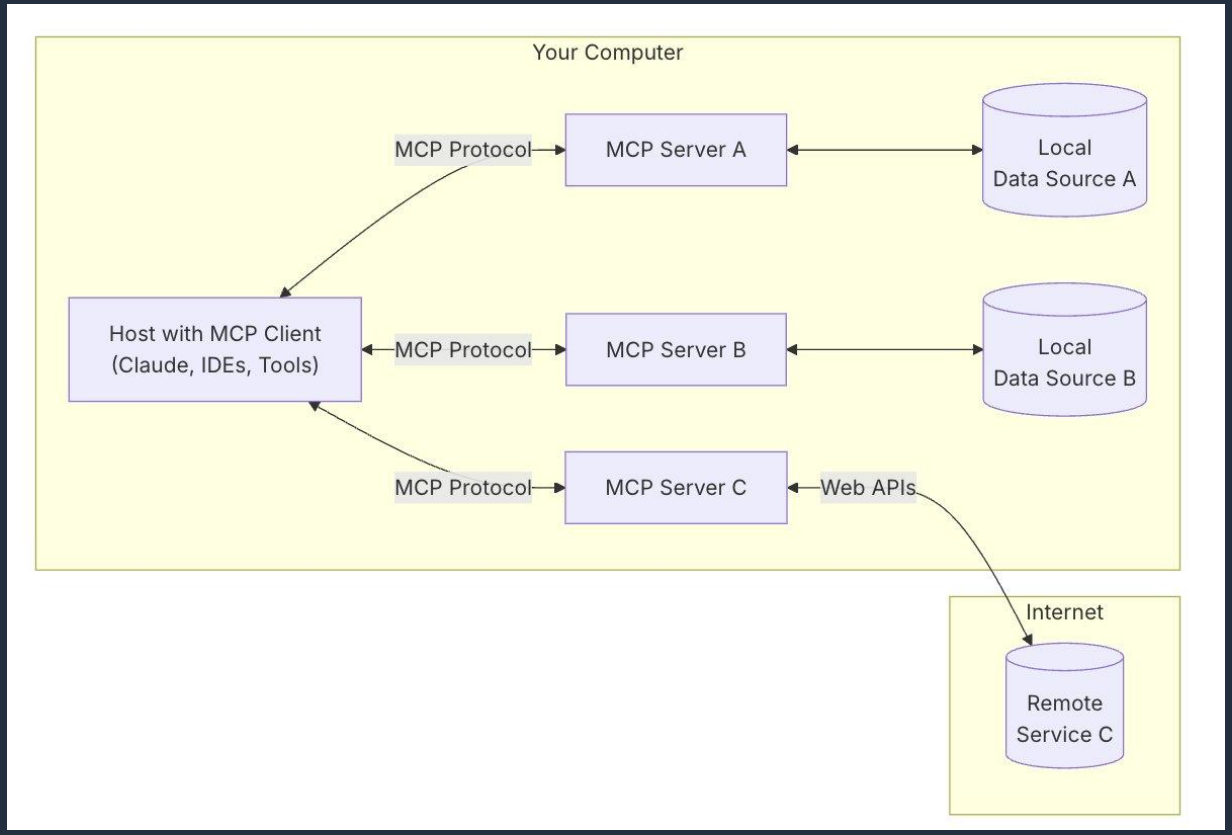
Amazon Q Developer

– Q for Command line （Q Developer CLI)

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

MCP Architecture

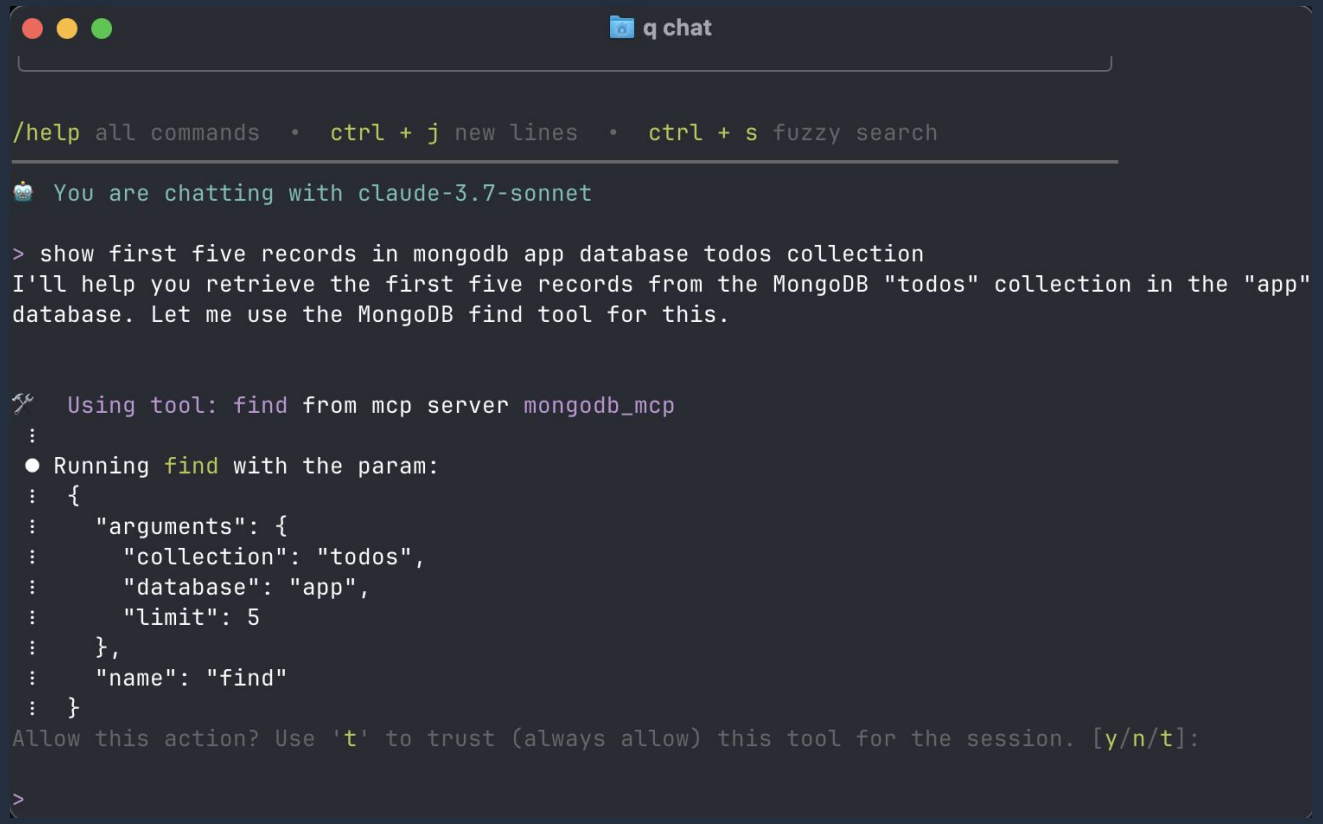
- **MCP Host**: LLM Applications that initiate connection. Example: Cursor, Windsurf, Claude Desktop etc.
- **MCP Client**: Maintain 1:1 connections with servers, inside the host application
- **MCP Server**: provide context, tools, and prompt to clients



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon Q CLI + MongoDB MCP Server

Amazon Q Developer on the command line

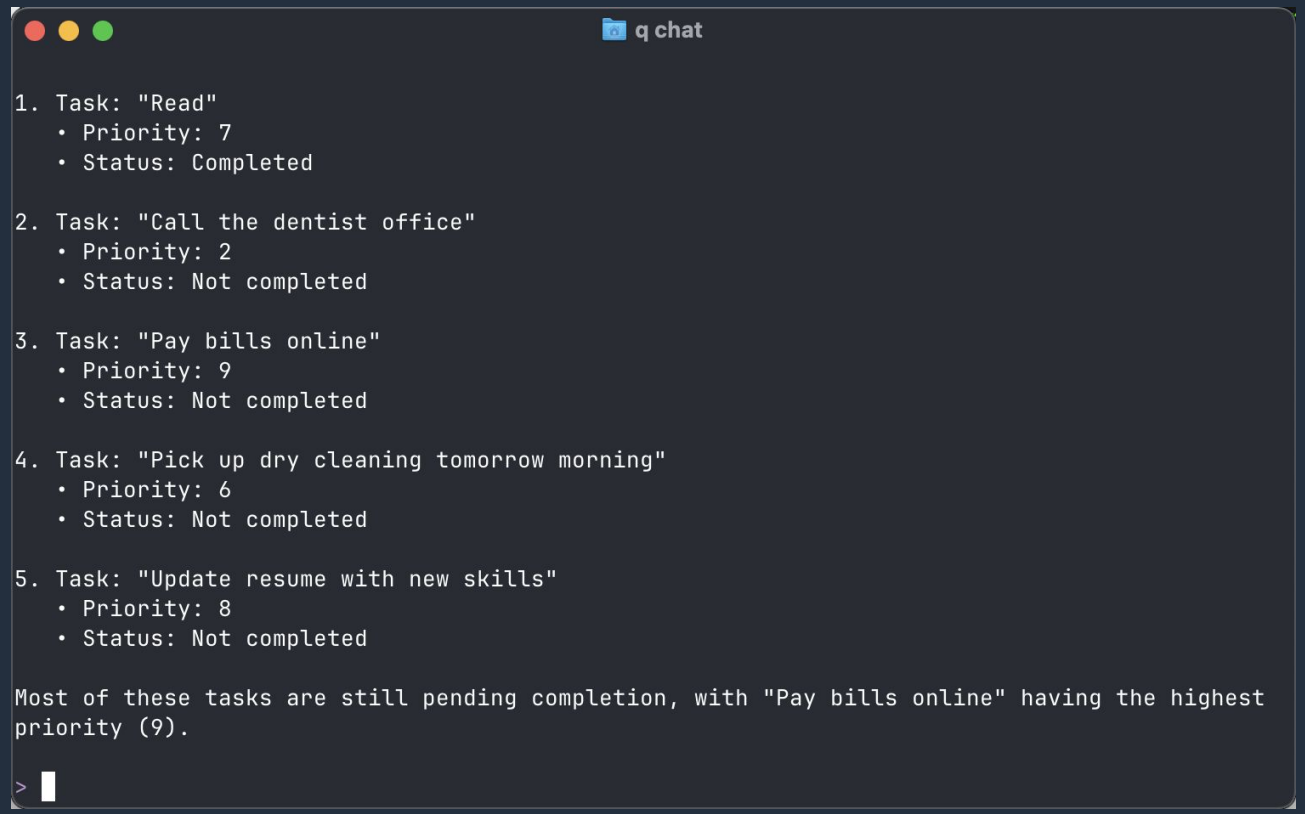


- Command line interface to chat through natural language conversations, questions, and responses within your terminal environment.
- Translate natural language instructions to executable shell code snippets.
- Inline suggestions as you type in your command prompt, with completions to popular CLIs.

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Amazon Q CLI + MongoDB MCP Server

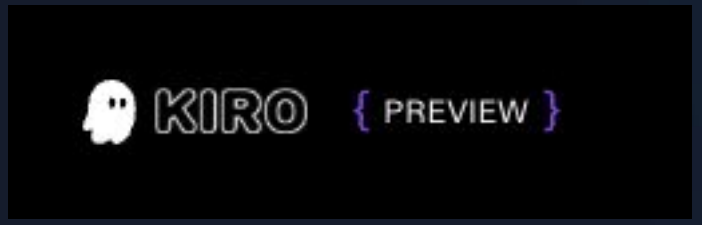
Amazon Q Developer on the command line



- Command line interface to chat through natural language conversations, questions, and responses within your terminal environment.
- Translate natural language instructions to executable shell code snippets.
- Inline suggestions as you type in your command prompt, with completions to popular CLIs.

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

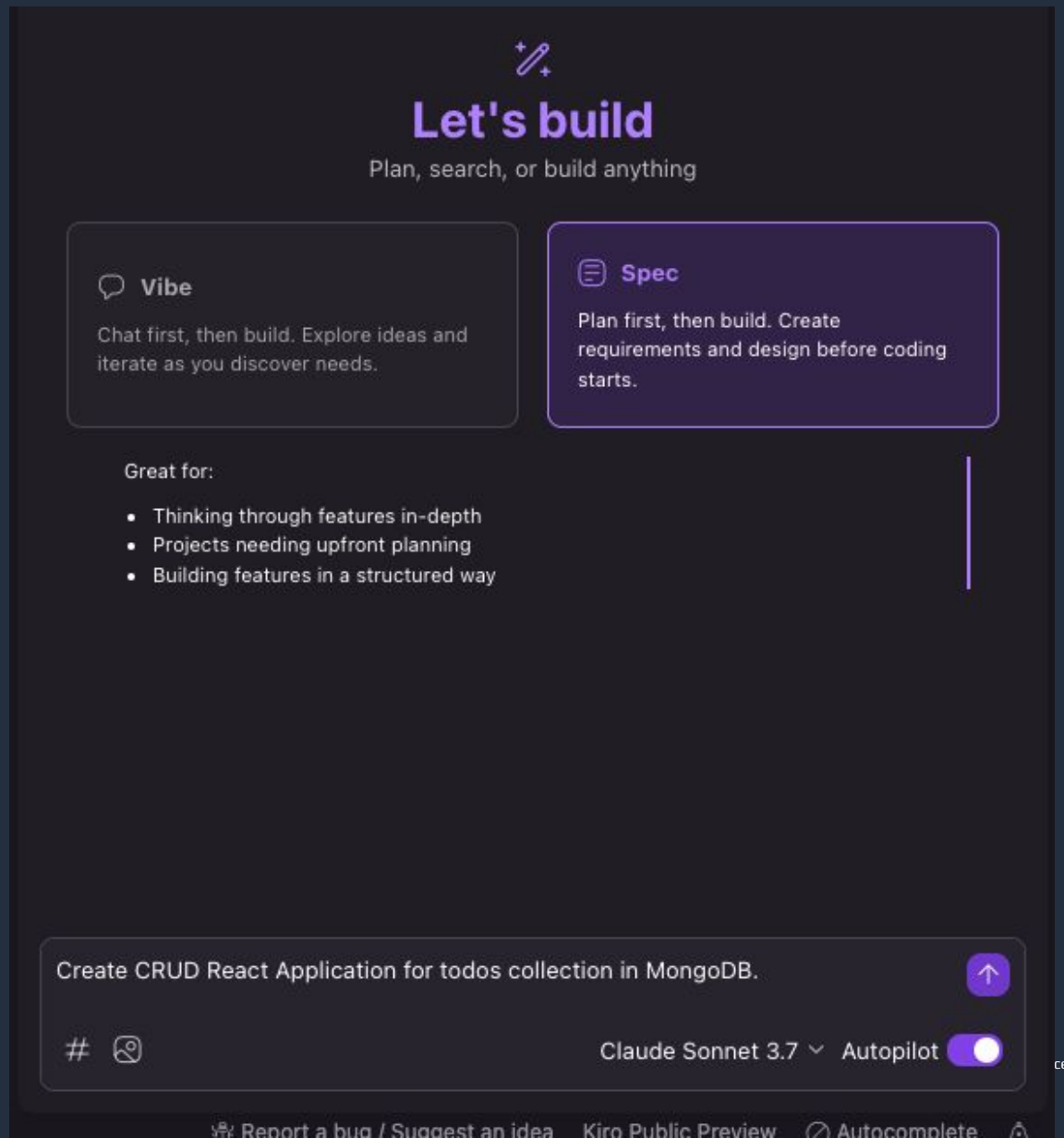
Kiro



aws

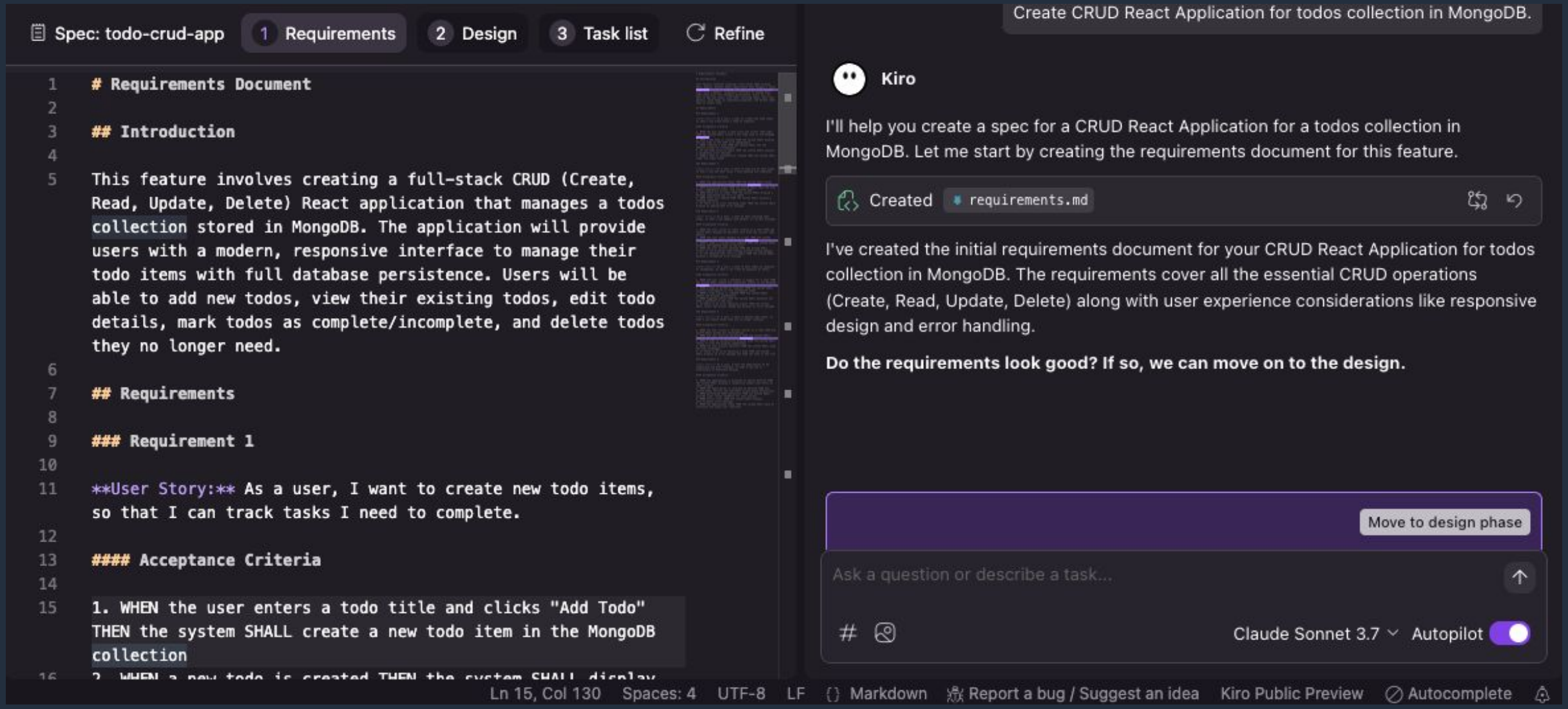
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Starting



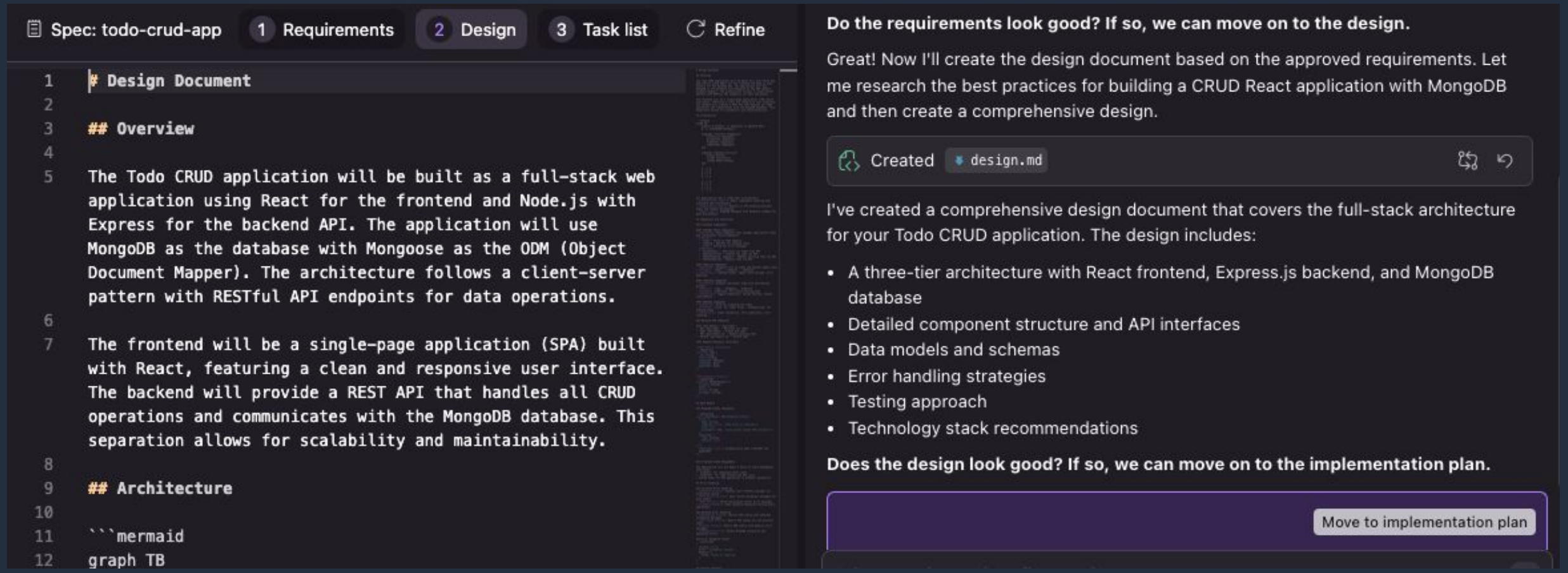
ces, Inc. or its affiliates. All rights reserved.

Spec



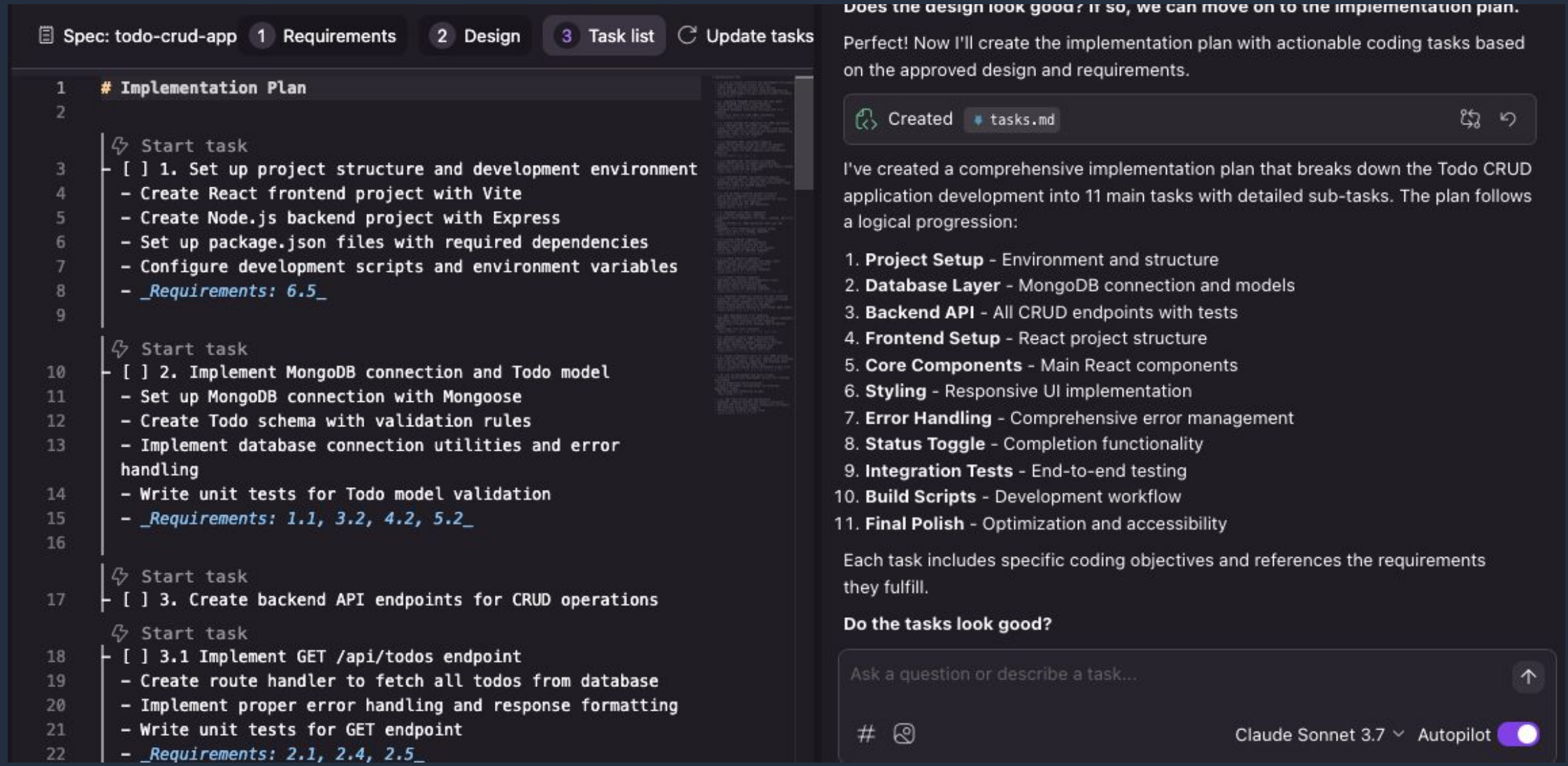
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Design



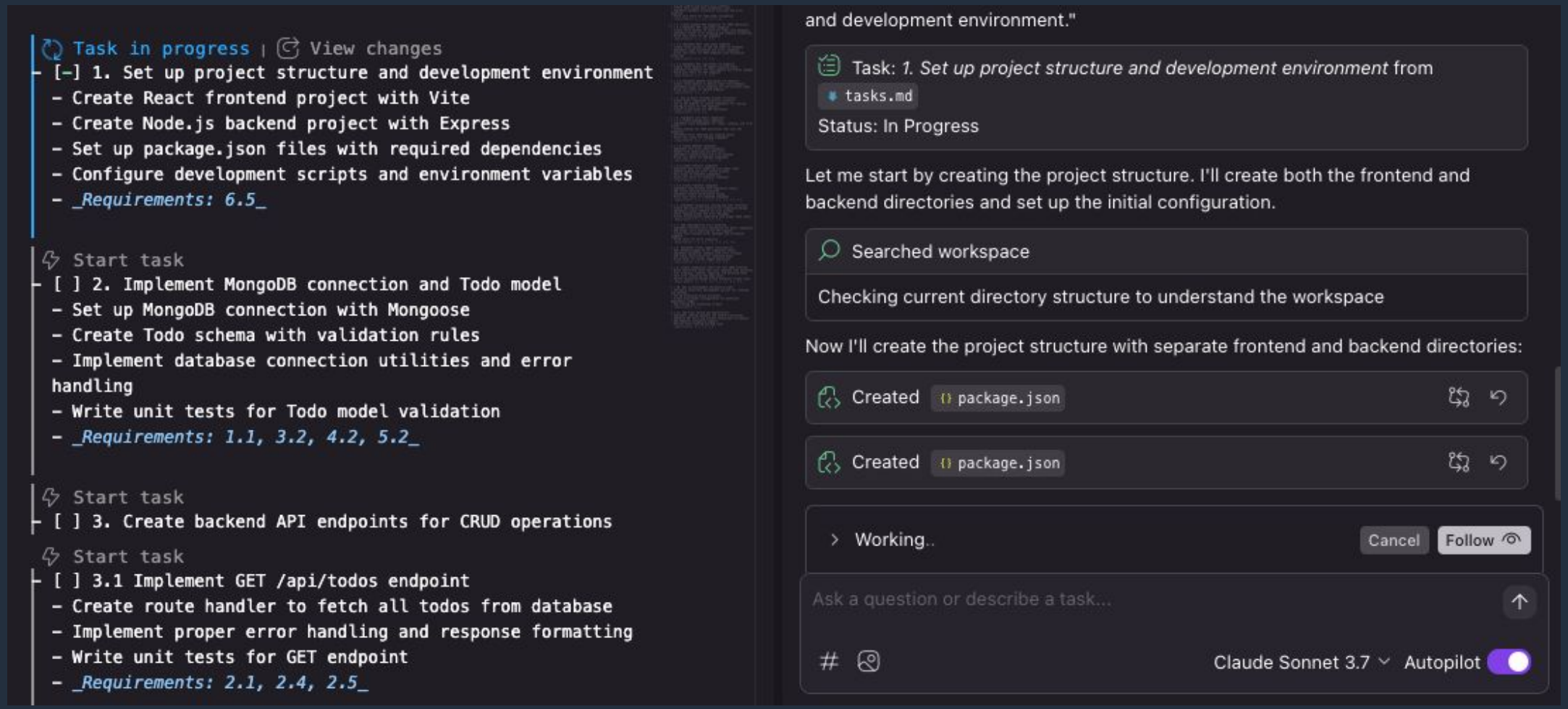
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Tasks



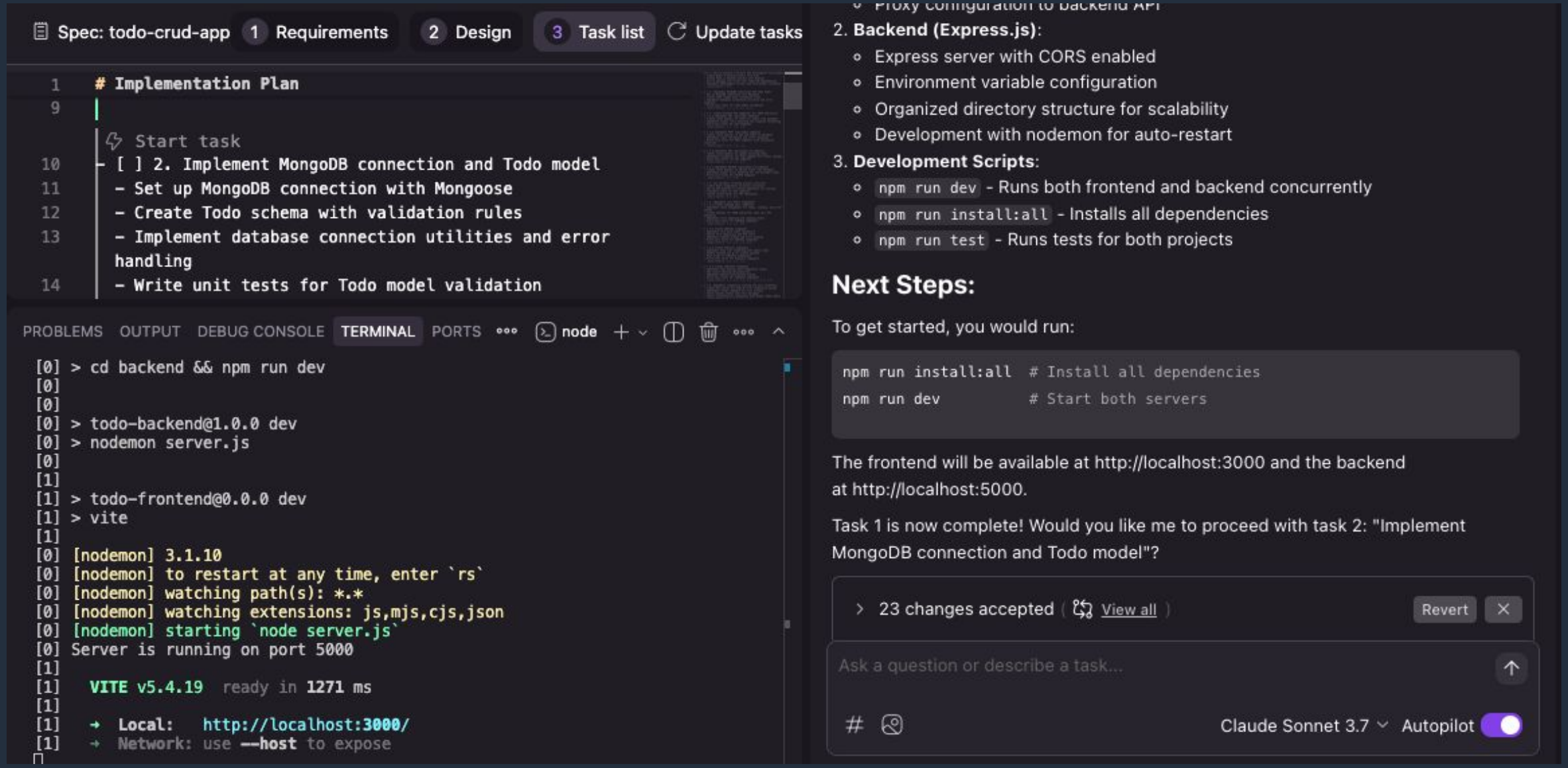
© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Execution



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

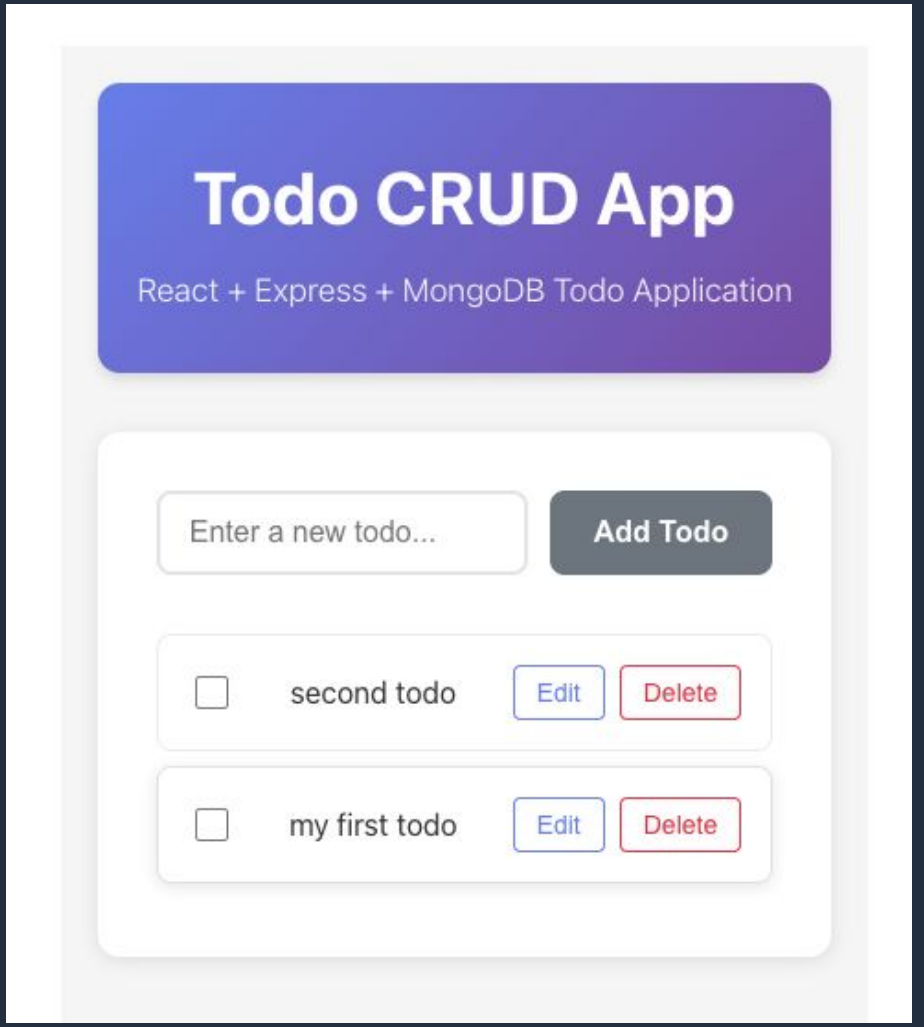
Execute



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Tasks Completed

```
# Implementation Plan

✓ Task completed
- [x] 6. Implement responsive styling and user interface
  - Create CSS styles for mobile-first responsive design
  - Implement visual feedback for user actions
  - Style loading states and error messages
  - Ensure accessibility compliance with proper ARIA labels
  - _Requirements: 6.1, 6.2, 6.3, 6.5_

✓ Task completed
- [x] 7. Implement status toggle functionality
  - Add checkbox/toggle UI for completion status
  - Implement optimistic updates with error rollback
  - Add visual distinction for completed todos
  - Write tests for status toggle operations
  - _Requirements: 4.1, 4.2, 4.3, 4.4, 4.5_

✓ Task completed
- [x] 8. Set up development and build scripts
  - Configure concurrent development servers for frontend and backend
  - Set up production build processes
  - Create environment configuration for different deployment stages
  - Add linting and formatting scripts
  - _Requirements: 6.5_
```



© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Getting **started**

Get started with Amazon Bedrock AgentCore

Discover Kiro

MongoDB Atlas and Amazon Bedrock hands-on Workshop

AI Agents with MongoDB:
https://mdb.link/ai-agent-localnyc

aws

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Thank you!

Igor Alekseev

ialek@amazon.com

© 2024, Amazon Web Services, Inc. or its affiliates. All rights reserved.