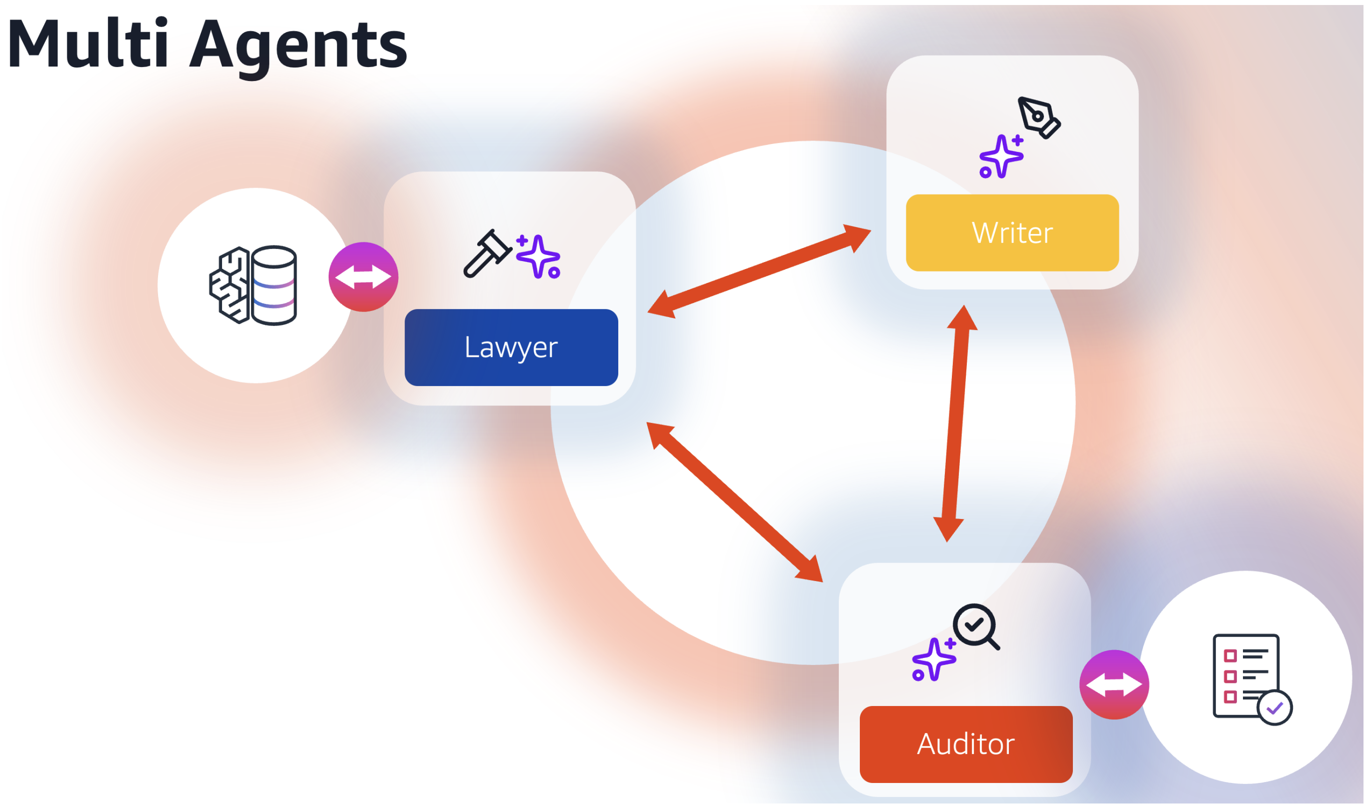
Compliance Analysis Agents

This component contains three specialized agents for compliance analysis and reporting:

1. **Auditor Agent** - Performs compliance assessments
2. **Lawyer Agent** - Provides legal analysis and Q&A
3. **Writer Agent** - Generates compliance reports



Prerequisites

- **AWS CLI** configured with appropriate permissions
- **Docker** installed and running
- **Python 3.12+** with pip
- **A CDK bootstrapped** AWS account
- **A Python virtual environment** with the dependencies installed
- **A Bedrock AgentCore execution role ARN** stored in the environment variable $AGENT_CORE_ROLE_ARN

Agent Overview

The system utilizes three specialized AI agents, each with distinct roles and capabilities:

Auditor Agent

- **Primary Role**: Compliance assessment and validation
- **Responsibilities**:
  - Evaluates the completeness of information for compliance assessment

    - Determines whether sufficient evidence exists to make regulatory determinations
    - Generates follow-up questions when additional information is needed
    - Produces final compliance assessments and recommendations
- **Decision Logic**: Uses regulatory frameworks to assess whether available information meets compliance requirements

Lawyer Agent

- **Primary Role**: Legal research and information retrieval
- **Responsibilities**:
    - Answers compliance-related questions using the organizational knowledge base
    - Provides legal interpretation and regulatory context
    - Retrieves relevant documentation and evidence
- **Available Tools**:
    - **Knowledge Base Tool**: RAG (Retrieval-Augmented Generation) system that searches the knowledge base from specific perspectives or viewpoints
    - **Query Classifier Tool**: Categorizes and routes queries to the most appropriate knowledge base perspective for optimal answer generation

Writer Agent

- **Primary Role**: Report compilation and documentation
- **Responsibilities**:
    - Synthesizes question-answer pairs into comprehensive compliance reports
    - Generates structured Markdown-formatted documentation
    - Ensures clarity and readability of compliance findings
    - Maintains consistent reporting standards across assessments

Agent Interaction Flow

1. **Auditor Agent** receives compliance documents and performs initial assessment
2. **Lawyer Agent** provides legal context and answers specific compliance questions
3. **Writer Agent** compiles findings into comprehensive reports

Deployment Instructions

Setup

Before running the deployment scripts, ensure you have:

1. **Environment Variables**: Set the following environment variables:

    ```
    export AWS_DEFAULT_REGION="us-east-1" #Change the region according to your needs
    export AWS_DEFAULT_REGION="us-east-1" #Change the region according to your needs
    ```

2. **Agent-specific Configuration**: Each agent directory contain a `.env` file with agent-specific environment variables that will be loaded during deployment. Make sure to modify the values according to your deployment needs

Deploy All Agents

Use the provided `deploy_agents.sh` script to deploy all three agents at once:

```
# Set required environment variables
export AWS_DEFAULT_REGION="us-east-1"

# Make the script executable and run it
chmod +x deploy_agents.sh
./deploy_agents.sh
```

The script will:

- Automatically detect your AWS account ID
- Build and push Docker images to ECR
- Create or update all three agent runtimes
- Handle environment variables from `.env` files in each agent directory
- Output the ARNs for all deployed agents

Verify Deployment

After deployment, test each agent using the provided test scripts. Each agent directory contains a `test_invoke_agent.py` file with sample payloads.

Important: You must update the `agentRuntimeArn` in each test file with the ARN output from the deployment script.

Test Agents

```
cd lawyer_agent
# Edit test_invoke_agent.py and update the agentRuntimeArn with your deployed agent ARN
python test_invoke_agent.py

cd auditor_agent
# Edit test_invoke_agent.py and update the agentRuntimeArn with your deployed agent ARN
python test_invoke_agent.py

cd writer_agent
# Create or edit test_invoke_agent.py and update the agentRuntimeArn with your deployed agent ARN
python test_invoke_agent.py
```

Updating Test Files

After deployment, the `deploy_agents.sh` script outputs the ARNs for all agents. Update each test file:

1. Copy the appropriate agent runtime ARN from the deployment output
2. Replace the `agentRuntimeArn` value in the test file
3. Run the test script to verify the agent responds correctly

Example ARN format:

```
arn:aws:bedrock-agentcore:us-east-
1:YOUR_ACCOUNT_ID:runtime/compliance_analysis_lawyer_agent-UNIQUE_ID
```

Troubleshooting

Common Issues

1. **Permission Errors**: Ensure your AWS credentials have sufficient permissions for Bedrock AgentCore
2. **Docker Issues**: Verify Docker is running and you can pull images
3. **Role Not Found**: Make sure the IAM role was created successfully

Logs and Monitoring

- Observability is enabled for all agents
- Check CloudWatch logs for detailed execution information
- Use `aws logs tail <agent_log_group> --follow` to view recent logs

Cleanup

```
# Delete agents (using AgentCore CLI)
bedrock-agentcore-control delete-agent-runtime
compliance_analysis_writer_agent
bedrock-agentcore-control delete-agent-runtime
compliance_analysis_auditor_agent
bedrock-agentcore-control delete-agent-runtime
compliance_analysis_lawyer_agent
```