AWS re:Invent

DECEMBER 2 – 6, 2024 | LAS VEGAS, NV

DOP210

Accelerate multi-step SDLC tasks with Amazon Q Developer agents

Doug Clauson
Principal Product Manager, Amazon Q
AWS

Dr. Johnna Powell
Head of Technology, Research, and Innovation
DTCC

Manikandan Srinivasan
Sr. Manager, Product, Amazon Q
AWS

Speakers

Doug Clauson

Principal Product Manager, Amazon Q
AWS

Dr. Johnna Powell

Head of Technology, Research, and Innovation
DTCC

Manikandan Srinivasan

Sr. Manager, Product, Amazon Q
AWS

Amazon Q Developer

Reimagines the experience across the entire software development lifecycle (SDLC)

Helps developers and IT professionals build and manage secure, scalable, and highly available applications

Accelerate software development with autonomous agents that plan and execute multi-step tasks

DTCC’s journey

We bring economies of scale: 2023 by the numbers

$3 QUADRILLION IN SECURITIES PROCESSED

FIXED INCOME CLEARING

Processed an average of **$6 trillion** per day in U.S. Government Securities and a monthly average of almost **7 trillion** Mortgage-Backed Securities transactions

INSTITUTIONAL TRADE PROCESSING

Processed **989 million** securities transactions

DERIVATIVES REPORTING

Covers all asset classes and processes **24 billion** messages annually for **5,350+ firms globally**

74+ regulators across the globe have access to our data from across **38 countries**

ASSET SERVICES

Worlds largest depository holds **1.40 million** active U.S issues worth **$85 trillion**

WEALTH MANAGEMENT

Processed **266 million** Fund/SERV transactions worth **$9.2 trillion**

17 billion insurance and retirement transactions with a processed settlement value of **$243 billion**

10,280+ unique alternative funds

DATA SERVICES

Settles **99%** of cash corporate, cash settled equity and muni debt.

EQUITIES CLEARING & SETTLEMENT

Clears an average of **197 million** broker-to-broker transactions per day worth **$1.93 trillion** for

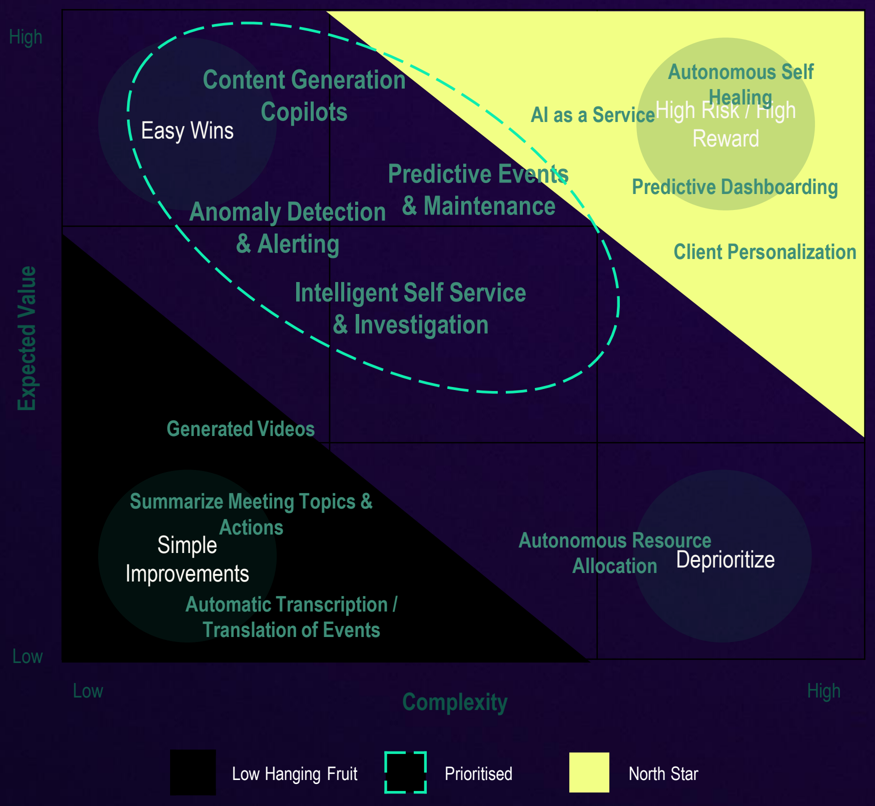
50+ Exchanges and Trading Venues settling **953 million** U.S. Transactions per year

Our unique capabilities and decades of experience deliver a network no other market infrastructure can match

6,000+ CLIENT FIRMS IN 93 COUNTRIES.

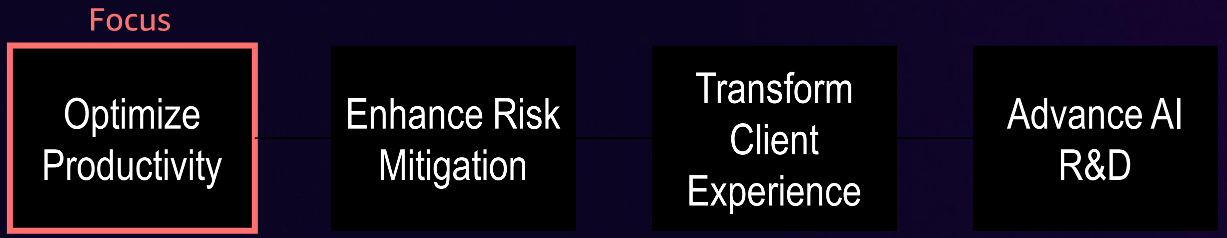
Priority Use case themes were identified through survey, interviews and external research

- Based on a comprehensive survey and insights from leadership, the prioritized set of use cases were aggregated and narrowed down to four key themes as a starting point towards a north star based on value provided and complexity to execute.

Strategic Pillar	Theme	Prioritization
Optimize Productivity	**Content Generation Copilots**: Providing capabilities to generate content from simple text summarization to complicated technical documentation, code and data to **improve developer and employee efficacy**	
Enhance Risk Mitigation	**Anomaly Detection & Alerting:** Identifying new unknown patterns that deviate significantly from historical data **to provide more context-aware notifications** as compared to traditional detection based on defined thresholds **Predictive Events & Maintenance: Predict future potential failures or incidents** to allow remediation to prevent them from occurring	
Transform the Client Experience	**Intelligent Self Service & Investigation**: Streamline and enhance the process of exploring, analyzing and **understanding complex structured & unstructured information from data via a prompt**	

DTCC CONDUCTED A PAIR PROGRAMMING PILOT TO ADDRESS DTCC'S AI STRATEGIC PILLARS & PRIORITIES

DTCC AI Strategy



Focus

Optimize Productivity

Enhance Risk Mitigation

Transform Client Experience

Advance AI R&D

Key Theme

Content Generation: Providing capabilities to generate content from simple text summarization to complicated technical documentation, code, and data to improve developer efficacy.

Use Case

Can pair programming tools help increase developer productivity or technical documentation via generative code.

Pilot Objectives

Faster Development
Automation of routine tasks, such as code generation, freeing up software developers

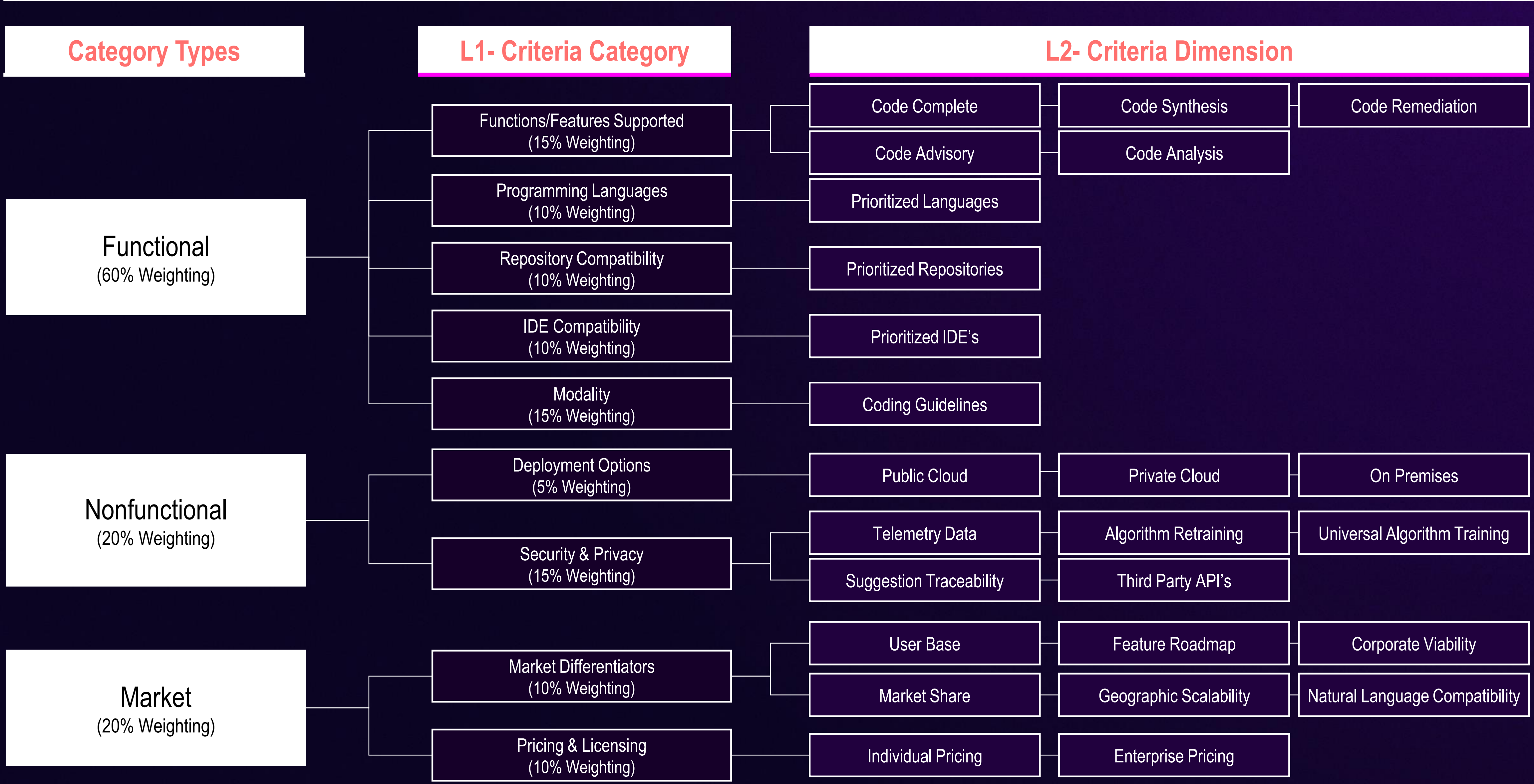
Improve Code Quality
Ensure that code is consistent, efficient, and follows best practices, reducing the likelihood of bugs

Enhance Productivity
Help increase productivity of software developers with relevant information and suggestions in real-time

Increase Innovation
Help software developers to be more creative by freeing up their time from routine tasks

Enhance Collaboration
Facilitate collaboration between software developers by enabling them to share information

WE DEFINED A WEIGHTED FRAMEWORK COVERING FUNCTIONAL, NONFUNCTIONAL, & MARKET CATEGORIES, WITH AN EMPHASIS ON FUNCTIONAL CAPABILITIES



Category Types	L1- Criteria Category	L2- Criteria Dimension
Functional (60% Weighting)	Functions/Features Supported (15% Weighting)	Code Complete Code Synthesis Code Remediation Code Advisory Code Analysis
	Programming Languages (10% Weighting)	Prioritized Languages
	Repository Compatibility (10% Weighting)	Prioritized Repositories
	IDE Compatibility (10% Weighting)	Prioritized IDE’s
	Modality (15% Weighting)	Coding Guidelines
Nonfunctional (20% Weighting)	Deployment Options (5% Weighting)	Public Cloud Private Cloud On Premises
	Security & Privacy (15% Weighting)	Telemetry Data Algorithm Retraining Universal Algorithm Training Suggestion Traceability Third Party API’s
Market (20% Weighting)	Market Differentiators (10% Weighting)	User Base Feature Roadmap Corporate Viability Market Share Geographic Scalability Natural Language Compatibility
	Pricing & Licensing (10% Weighting)	Individual Pricing Enterprise Pricing

THE AMAZON Q DEVELOPER PILOT WAS A REPRESENTATIVE SAMPLE OF DTCC'S DEV SQUADS, LANGUAGES, AND INTEGRATED DEVELOPMENT ENVIRONMENTS (IDES)

- Amazon Q Developer was **provisioned to 30+ developers across 4 teams** (Team SP, Stress Testing, Team U, and ITP-Settlement Instruction Manager)
- The teams covered **71% of coding languages** and **50% of IDEs** used at DTCC
- Participants used the Pair Programming tool as **part of their usual workflow for 17 weeks** (3-5 sprints)
- **Key Performance Indicators across developer productivity and code quality** were analyzed after every sprint and compared against the pre-pilot baseline

- **Success Criteria**
- **Efficiency** (throughput, pull requests, build failure rate)
- **Quality** (code vulnerabilities, bugs, test coverage)
- **Experience** (developer satisfaction and productivity perception)

Primary Development Ecosystem At DTCC

Primary Programming Languages

5 of 7 used

SQL | Java | python | JavaScript JS | TS TypeScript | COBOL | Terraform

Primary IDEs

2 of 4 used

VS Code | IJ IntelliJ IDEA | PC PyCharm | *eclipse

*Eclipse on AWS roadmap for Q4 compatibility

ANALYSIS OF PILOT RESULTS INDICATE DTCC WILL SEE IMPROVED METRICS AND NO LOSS IN QUALITY WITH AMAZON Q DEVELOPER

Analysis Highlights

Throughput:

- Average increase in throughput per participant was 40%
- Amazon Q Developer's positive effect on throughput is statistically significant
- Given the variance in workload throughout development cycles, we expect the sustained increase in throughput to be within 10-20%

Code Quality:

- No adverse impact of Amazon Q Developer seen on code quality
- We saw minor changes in CQMA, test coverage and build failures
- Code quality metrics stayed within normal bounds and these changes should not be tied to Amazon Q Developer

Metrics Across Pilot Teams & Participants

40% Average **increase in throughput** per pilot participant

5% Average **increase in CQMA security scores** across code repositories in scope

0% Average change in **build failure rate** across 4 pilot teams

30% Average **reduction in code defects**

2% Average **increase in test coverage**

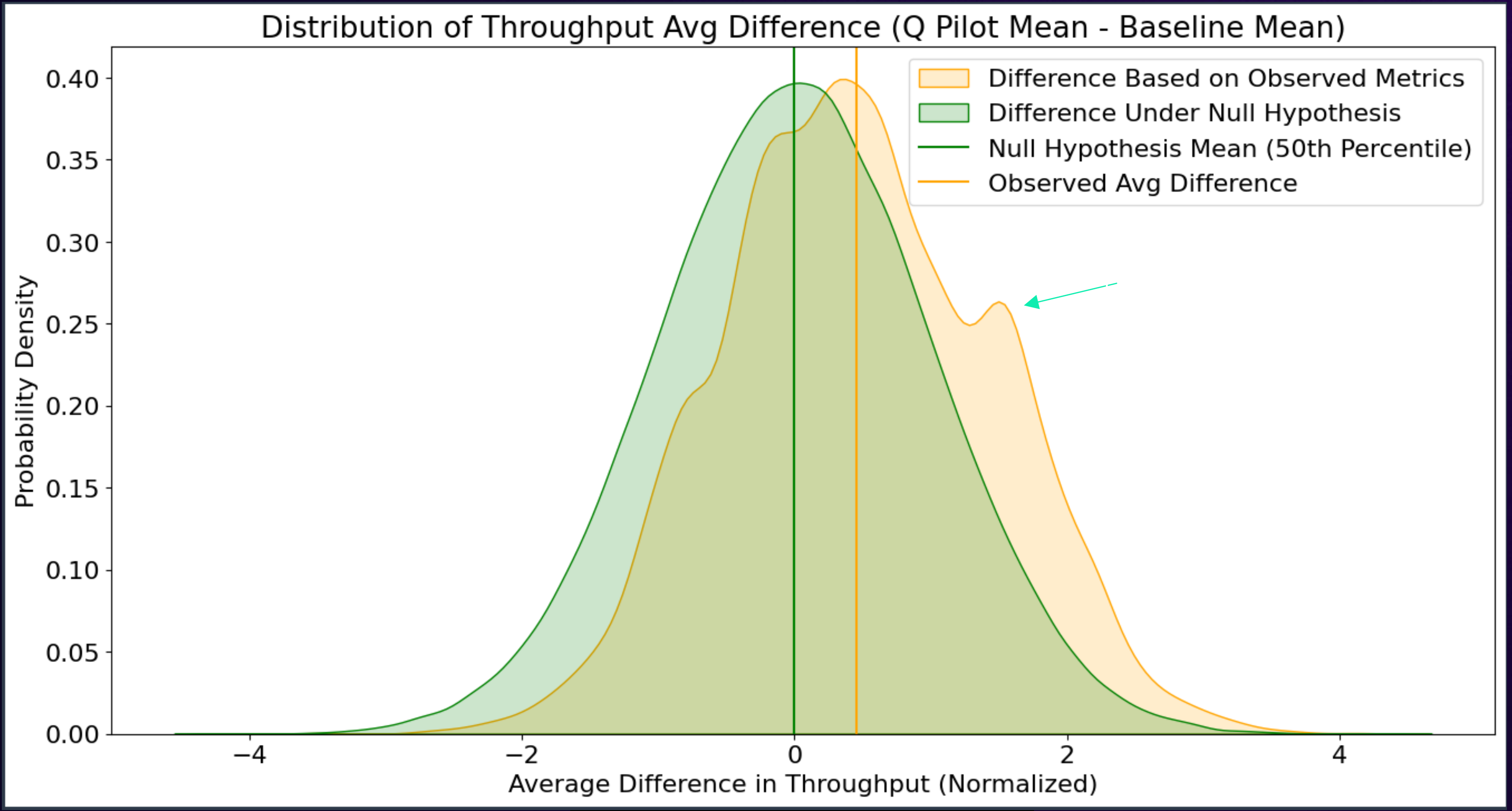
THE INCREASE IN THROUGHPUT OBSERVED WAS STATISTICALLY SIGNIFICANT

The observed average difference in throughput between participants during pilot and baseline periods was **18 percentile points (0.5σ)** above the expected difference under no effect and indicates that Amazon Q Developer has a positive effect with a high level of confidence

- **Green plot** represents the distribution of the difference of throughput means likely to be observed between baseline and pilot if each experiment were run many times under the null hypothesis (no effect seen)

- **Orange plot** shows the distribution of the differences likely to be observed based on observed metrics

- **T-test** indicates that the difference in means between the pilot and baseline distributions is **statistically significant**



Note: Q refers to Amazon Q Developer

CODE QUALITY MATURITY ASSESSMENT (CQMA) REMAINED STABLE WITH AMAZON Q DEVELOPER

CQMA is an aggregated score that reflects the robustness, security, and maintainability of code, providing a reliable measure of software quality.

CQMA = (1/3)*(Coverage + Complexity + Security)

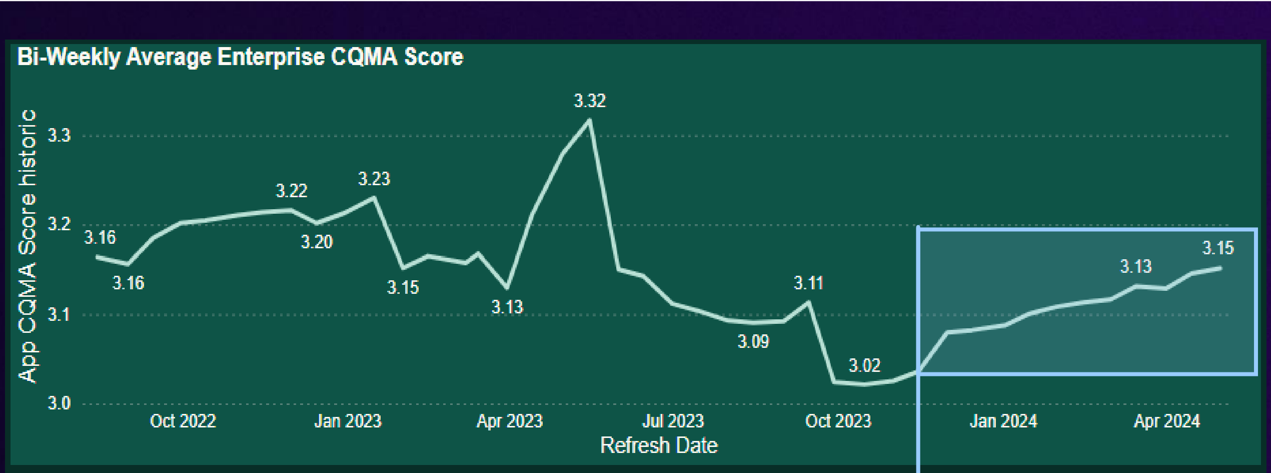
1. Code **Coverage** Quality Metric**:** evaluates the extent of automated test coverage

2. Code **Complexity** Fortify: assesses the structural complexity of the code

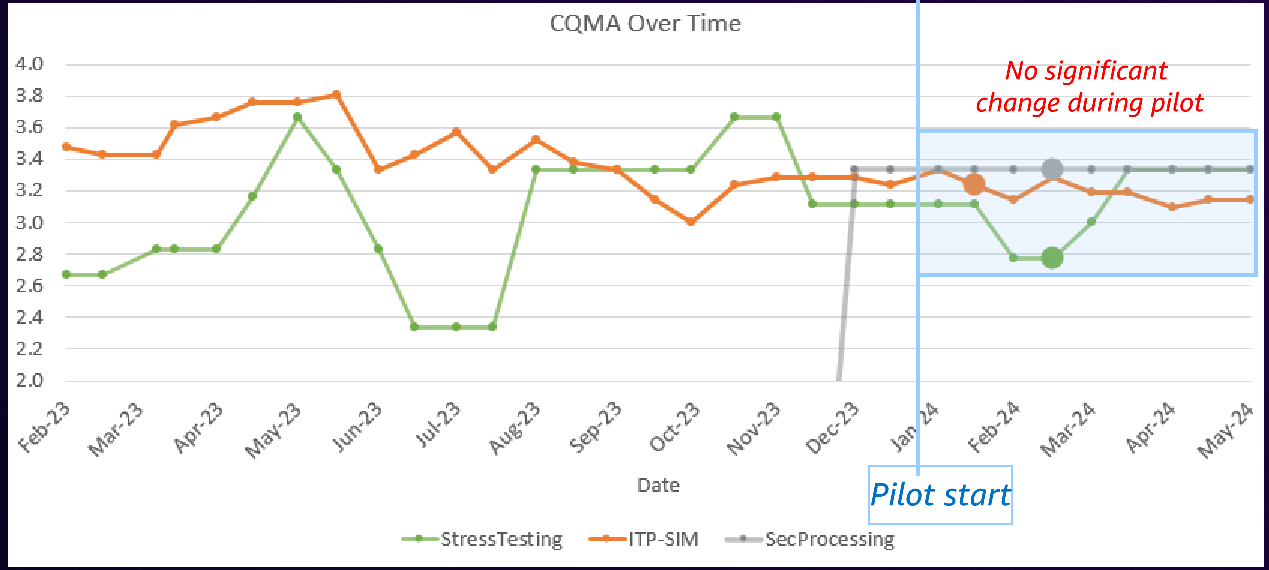
3. Code **Security** FOSS: reviews the security of open-source components used in the code

Baseline CQMA: Bi-weekly (sprint) average across baseline time periods

Pilot CQMA: Bi-weekly (sprint) average during pilot



Bi-Weekly Average Enterprise CQMA Score



CQMA Over Time

No significant change during pilot

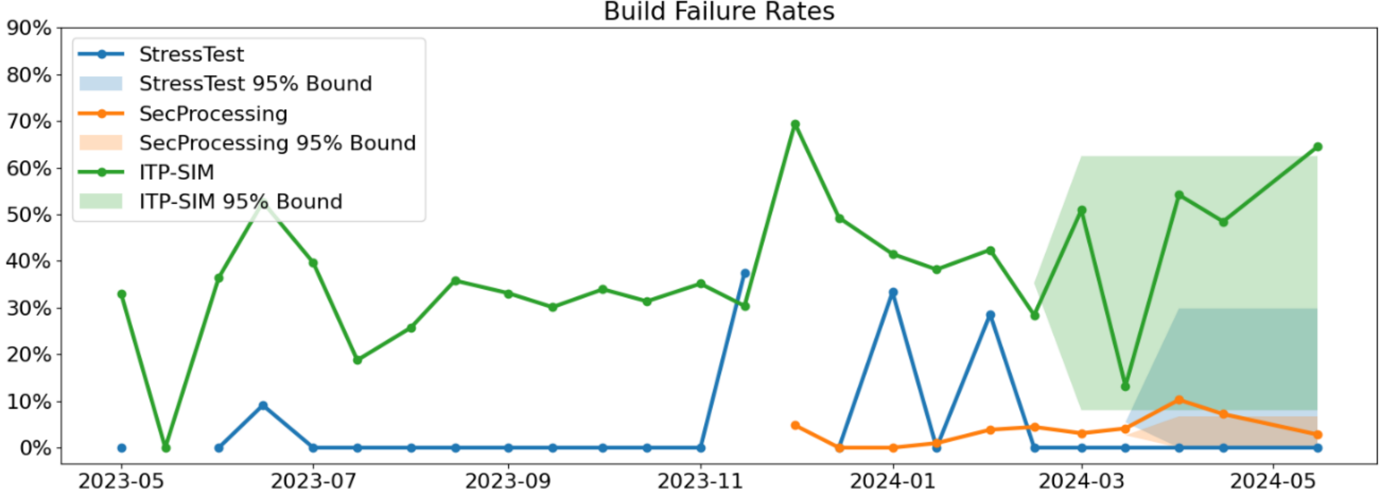
Pilot start

*Large dot = pilot start

BUILD FAILURE RATES AND CODE COVERAGE REMAINED IN RANGE WITH AMAZON Q DEVELOPER

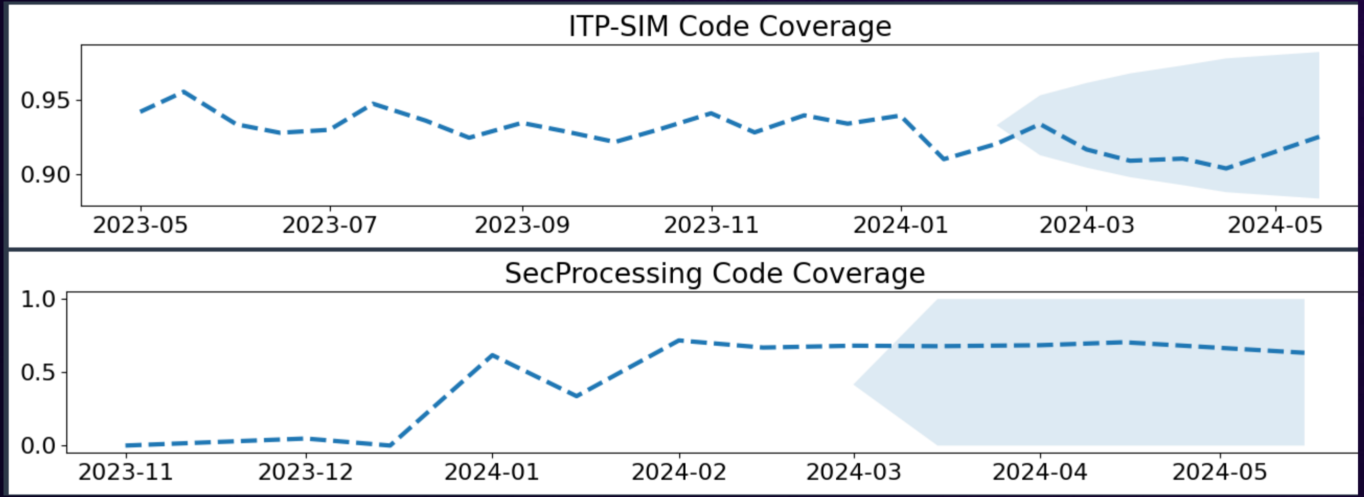
Build failure rate indicates frequency of issues in code integration and testing, which can serve as an indicator of code quality

- Fluctuations seen during the pilot were not statistically significant, indicating Amazon Q Developer has no affect on the number of build failures
- ITP-Settlement Instruction Manager has been experiencing an ongoing infrastructure issue with their pipeline builds failing last couple months (unrelated to Amazon Q Developer)



Code coverage measures the extent to which the automated tests cover the codebase, ensuring that more code is tested for defects, which can lead to higher quality and more reliable software

- New projects begin with low coverage
- Mature projects should have higher coverage
- New releases and code will affect coverage
- Fluctuations in coverage during pilot were not significant



AMAZON Q DEVELOPER INCREASED SATISFACTION AND DEVELOPERS SAW VALUE

High-Level Themes

Feedback on Amazon Q Developer was positive

- Developers highlighted benefits in explaining blocks of code in plain English, writing unit tests, code refactoring, and providing recommendations on complex problems
- Code refactoring is not logged in story points and hence not reflected in throughput although heavily used

Amazon Q Developer fills in technical gaps for developers

- Code explain capabilities and Q&A chat enable self-learning and reduce strain on senior developers
- Significantly reduces time to understanding in legacy environments for all developer levels

Project level contextualization will only increase the value of inline suggestions

- Python team found benefit from inline code generation capabilities, but most teams found less value due to dependencies on code context elsewhere in legacy codebase
- Human in the loop development still required as anticipated

Voice of Pilot Participants

‘Q has improved my productivity significantly while creating functional implementations and tests.’

‘Q has reduced the time it takes me to understand and solve problems.’

‘Q was very useful when working on non framework related tasks and unit tests.’

‘Q features help to improve code quality.’

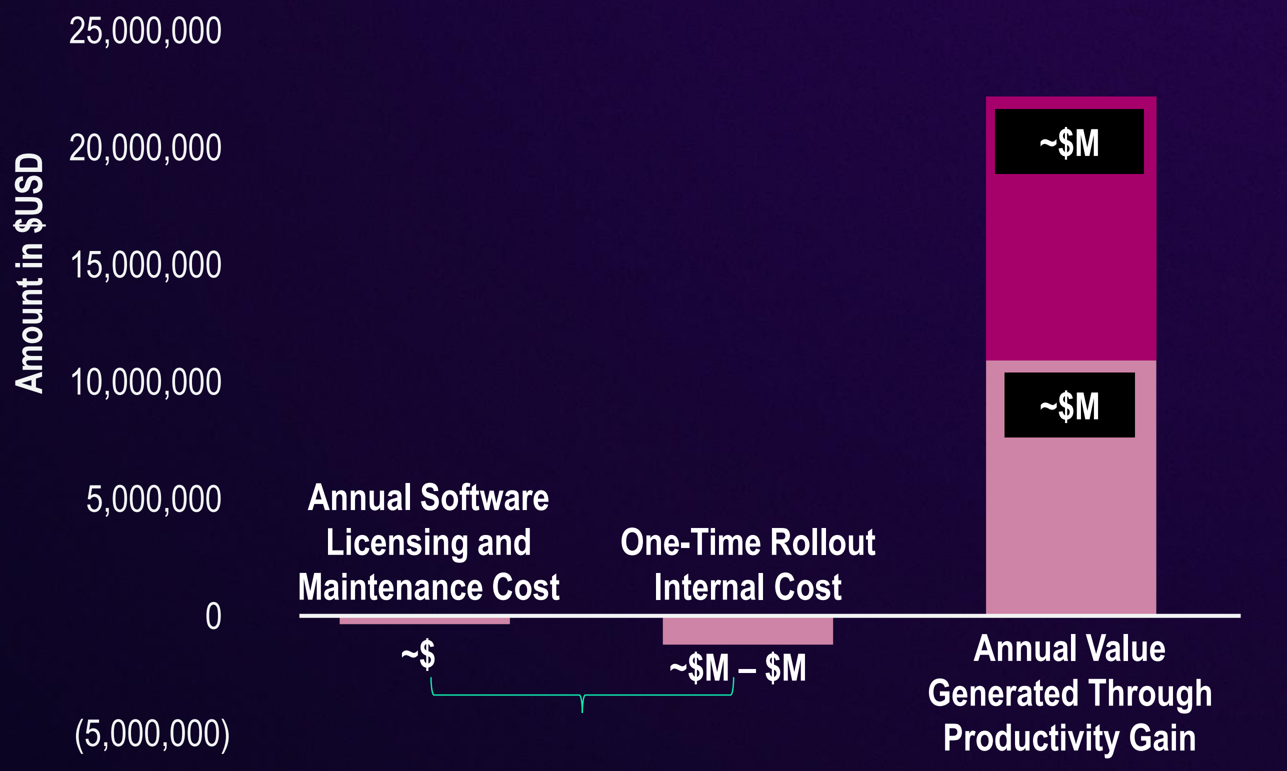
‘Q is helpful with code refactoring and unit testing, looking forward to explore more.’

‘Q has removed the dependency on more tenured engineers for assistance

WE RECOMMENDED AMAZON Q DEVELOPER DUE TO VALUE GENERATED

2,000 hours	50%	10-20%	$	$	$-$
Annual hours worked per developer	Percentage of total annual hours spent coding	Projected productivity boost	Annual cost per FTE	Annual Q cost per developer	Net annual value generated per developer

- ✓ Generally, productivity boost is expected to be lower in full scale rollout relative to pilot results.
- ✓ If we assume a rollout would achieve **10-20% productivity boost** (reduced over 50% from pilot results), that equates to **100-200 hours saved per developer** annually (assuming 50% of time spent on coding).
- ✓ Assuming an annual cost of ~$150k per FTE, this equates to **$7.2k - $14.8k in net value generated per developer**, net of product licensing cost.
- ✓ Assuming **1.5k developers, this equates to 150k-300k hours saved per year, or ~$M - $M of net value annually.**
- ✓ A full rollout of the product is expected to take 7 months, and **considers costs of risk reviews and approvals, training, and general set-up and support.**



Additional notes:

- 1,500 developers based on conservative active Jira users assigned story points
- $1M - 1.5M includes the internal cost associated with effort for approvals, set up, and change management.

Q DEVELOPER AGENTS BEING EVALUATED AT DTCC FOR ADOPTION

Area	DTCC's Use Case
Code upgrades	Java version upgrade from Java 8 to Java 17
Test coverage	Unit test generation, test cycle acceleration
Code comprehension	Technical specification and release notes
Code reviews	Scan code vulnerabilities, identify and upgrade older libraries

Amazon Q Developer agents

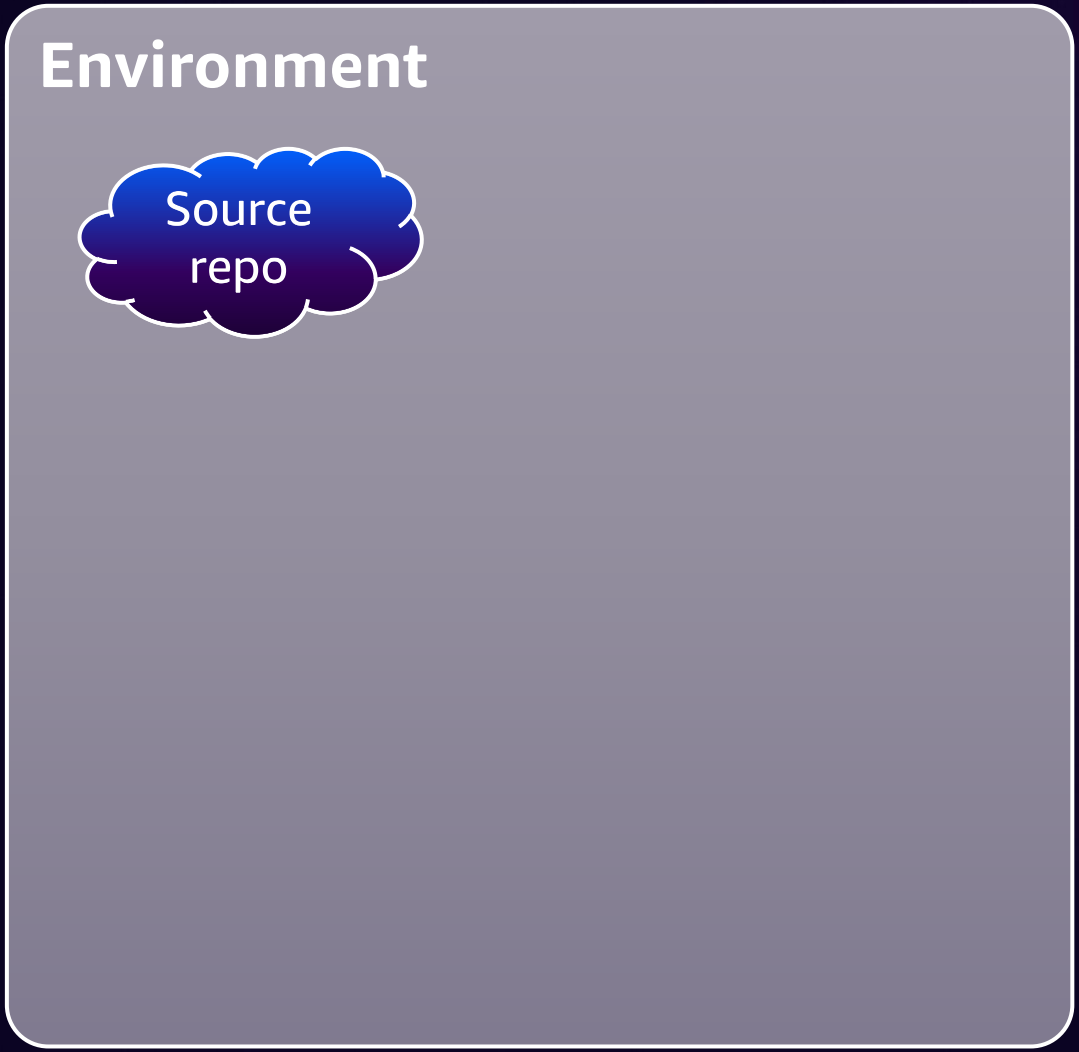
> ***An agent is a software program or system that perceives its environment through input data, makes decisions through some decision-making model or logic, and then takes actions to achieve its designed goals or objectives.***

Claude 3.5 Sonnet v2
LLM model

Amazon Q agent vision

Environment

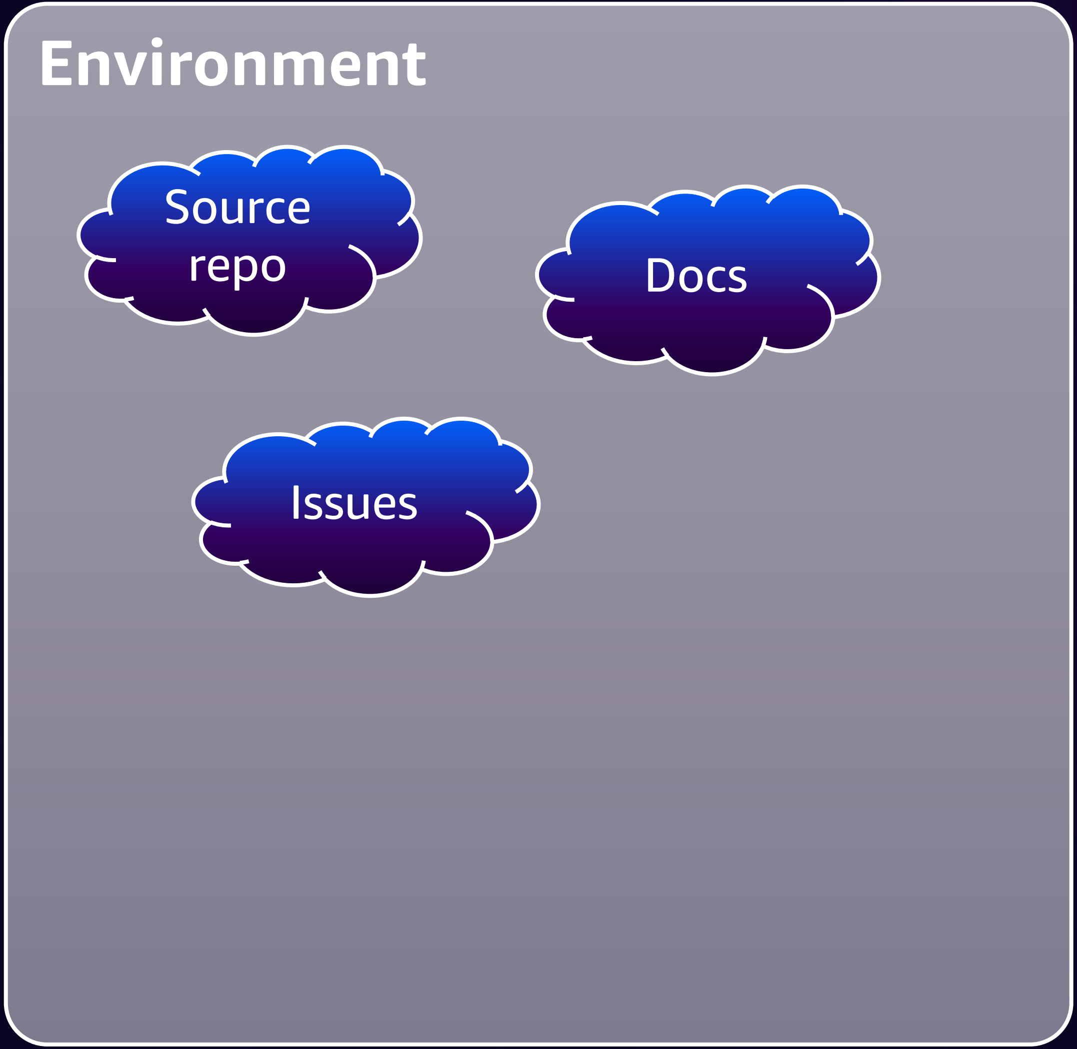
Amazon Q agent vision



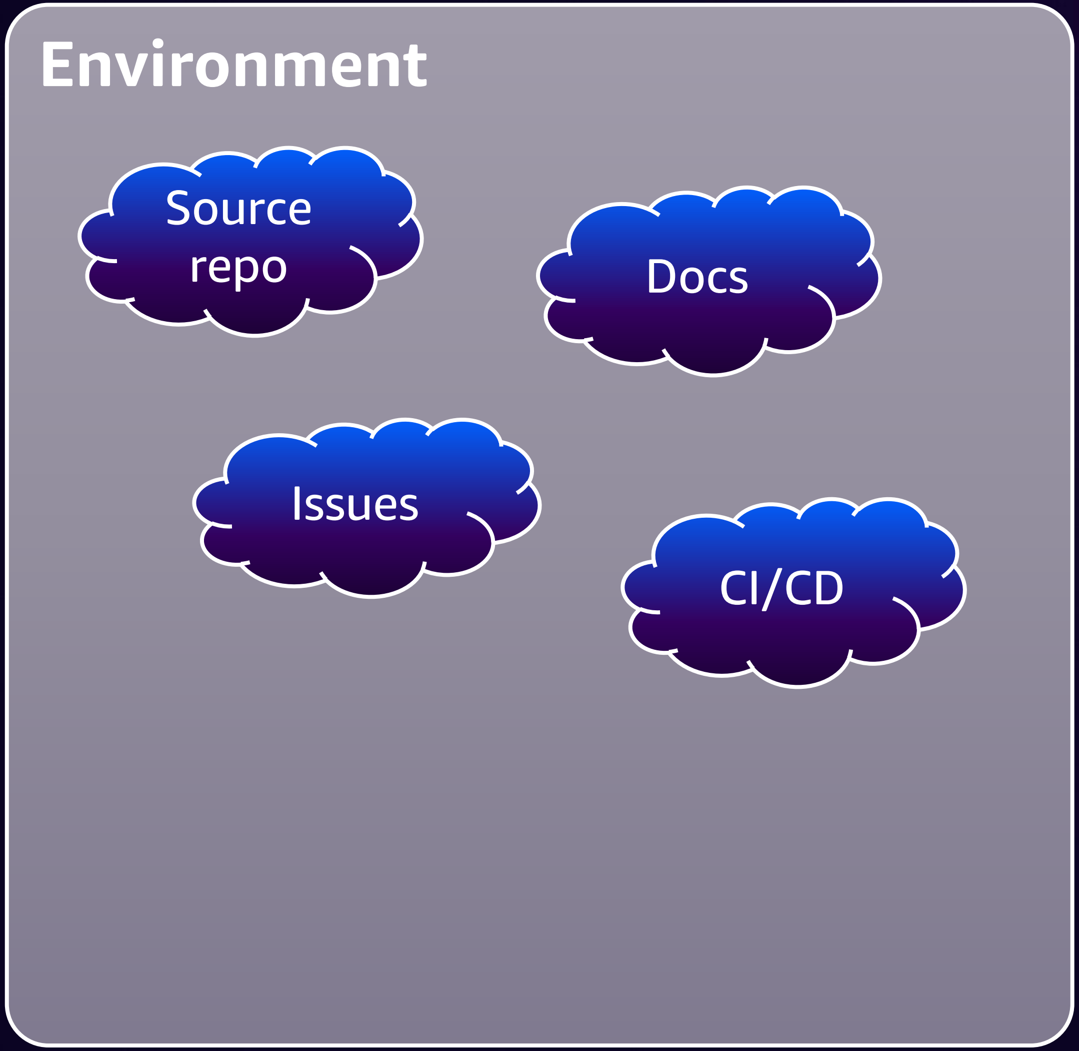
Environment

Source repo

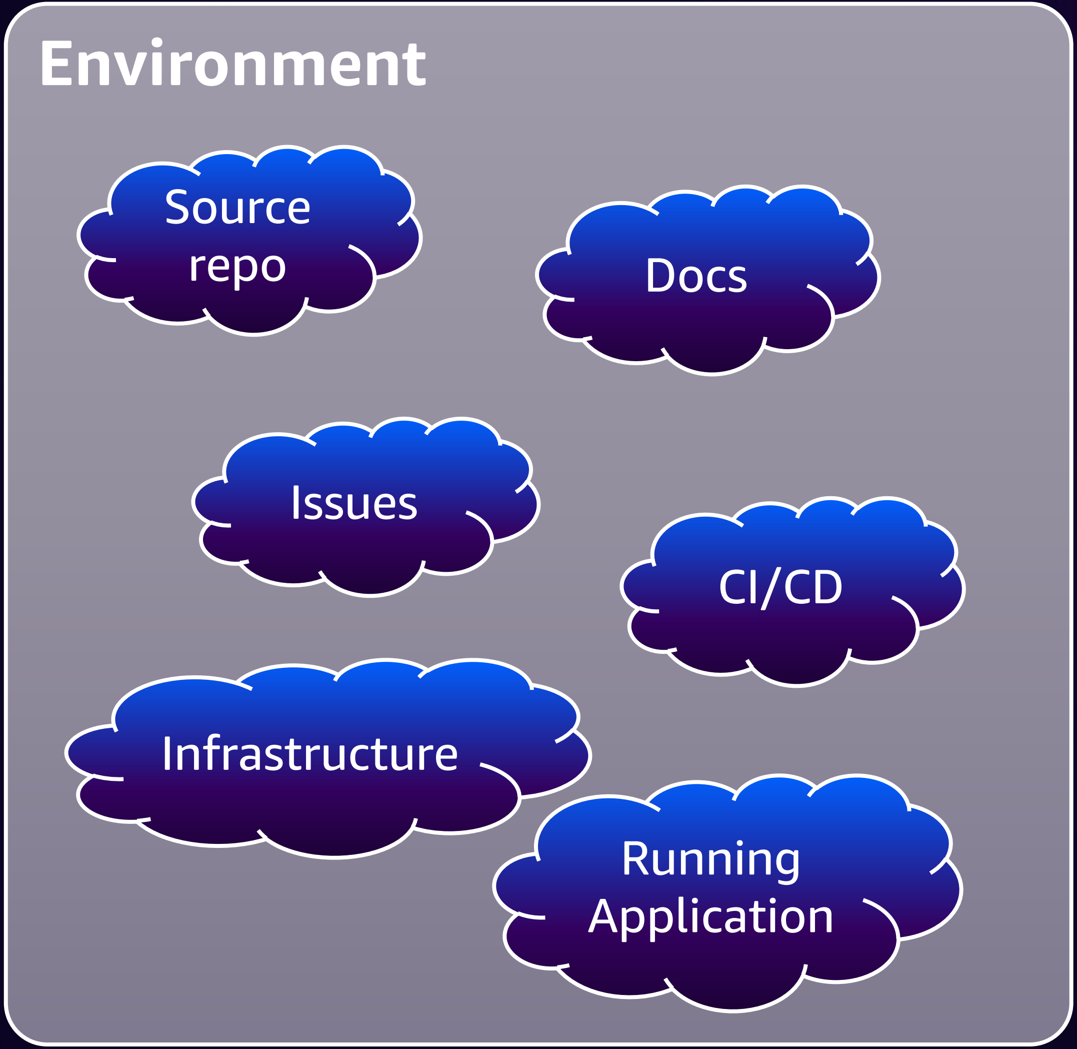
Amazon Q agent vision



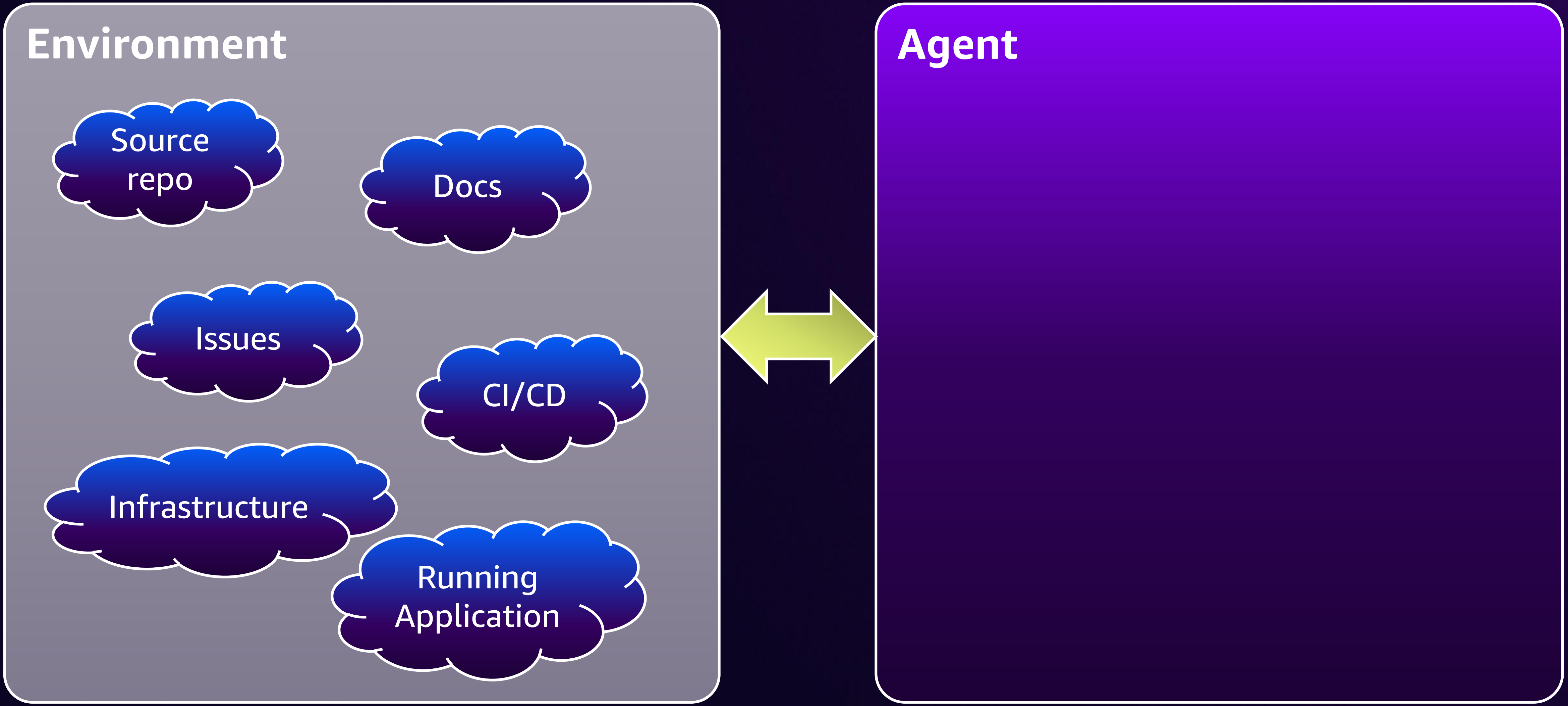
Amazon Q agent vision



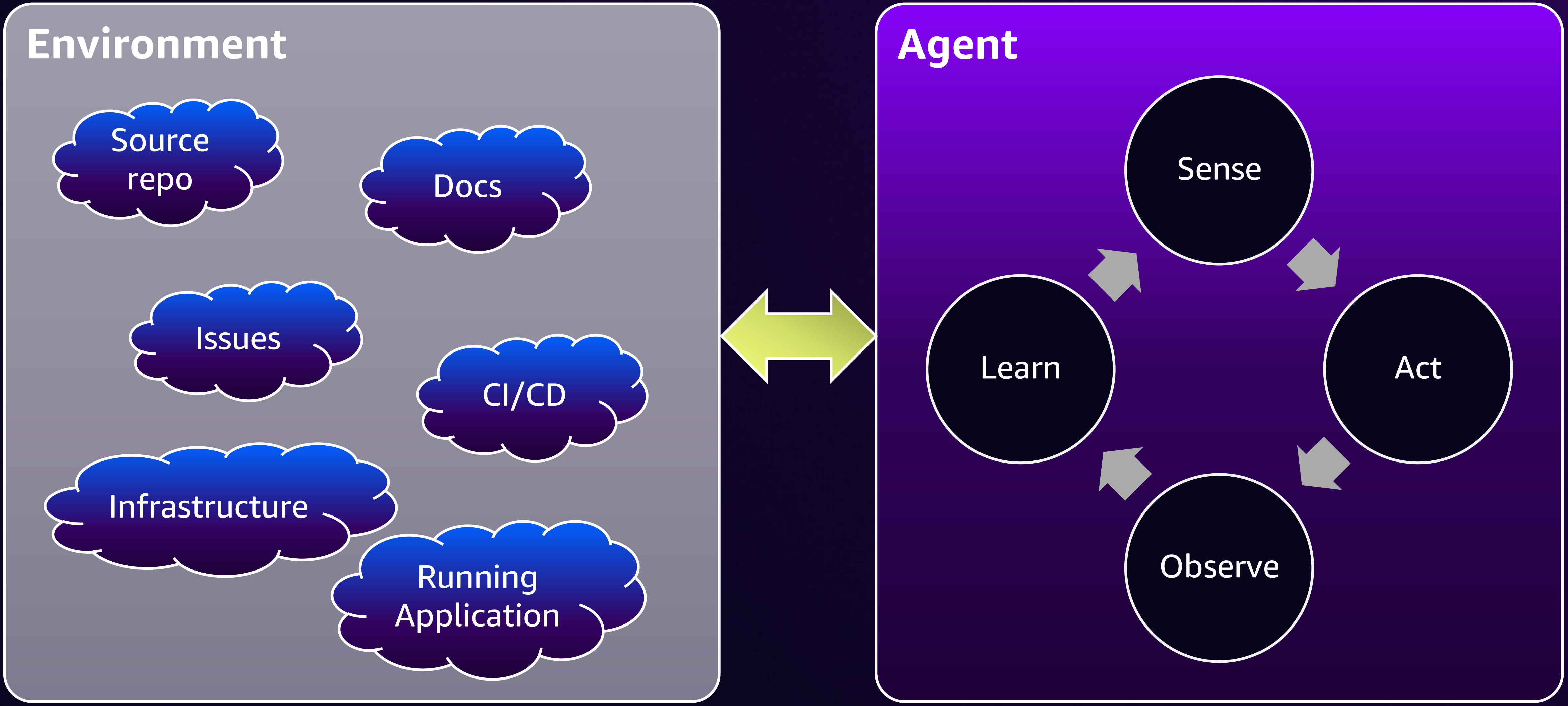
Amazon Q agent vision



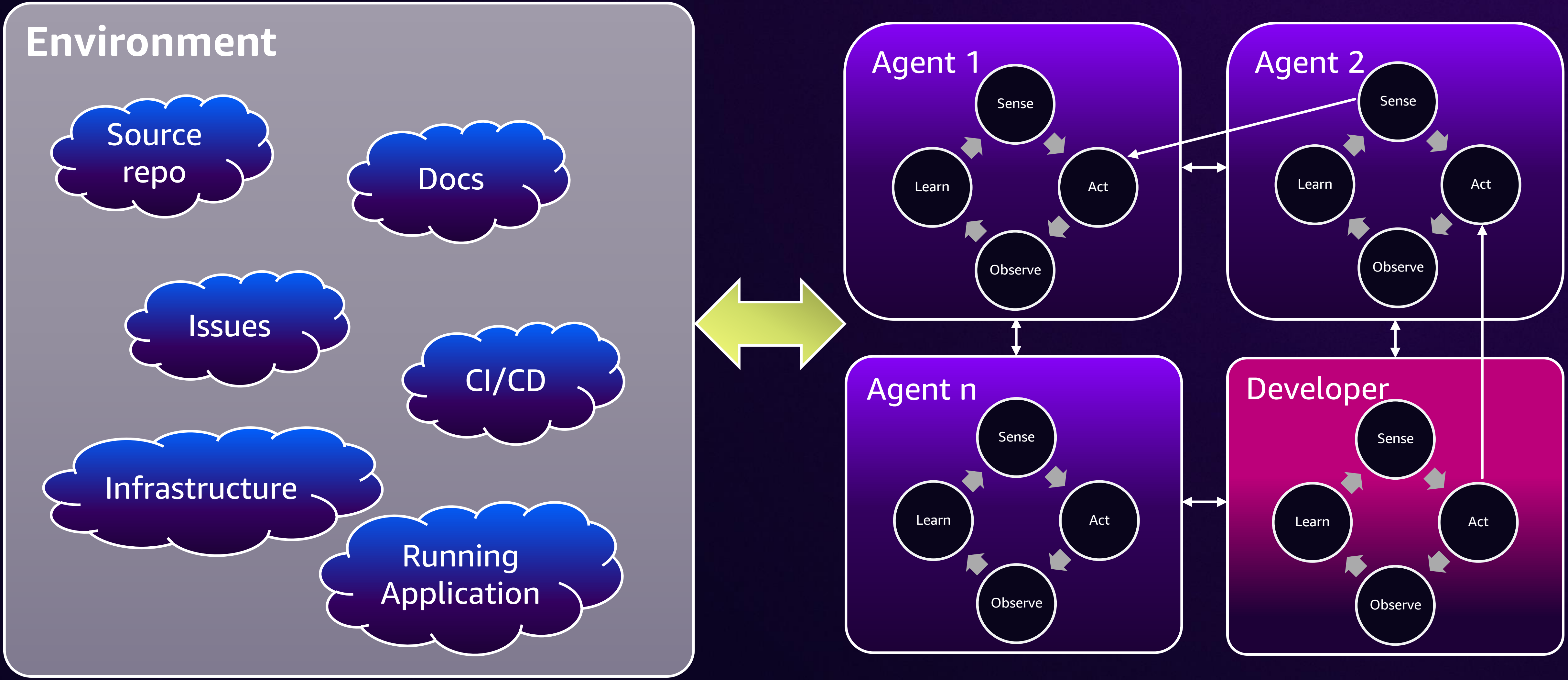
Amazon Q agent vision



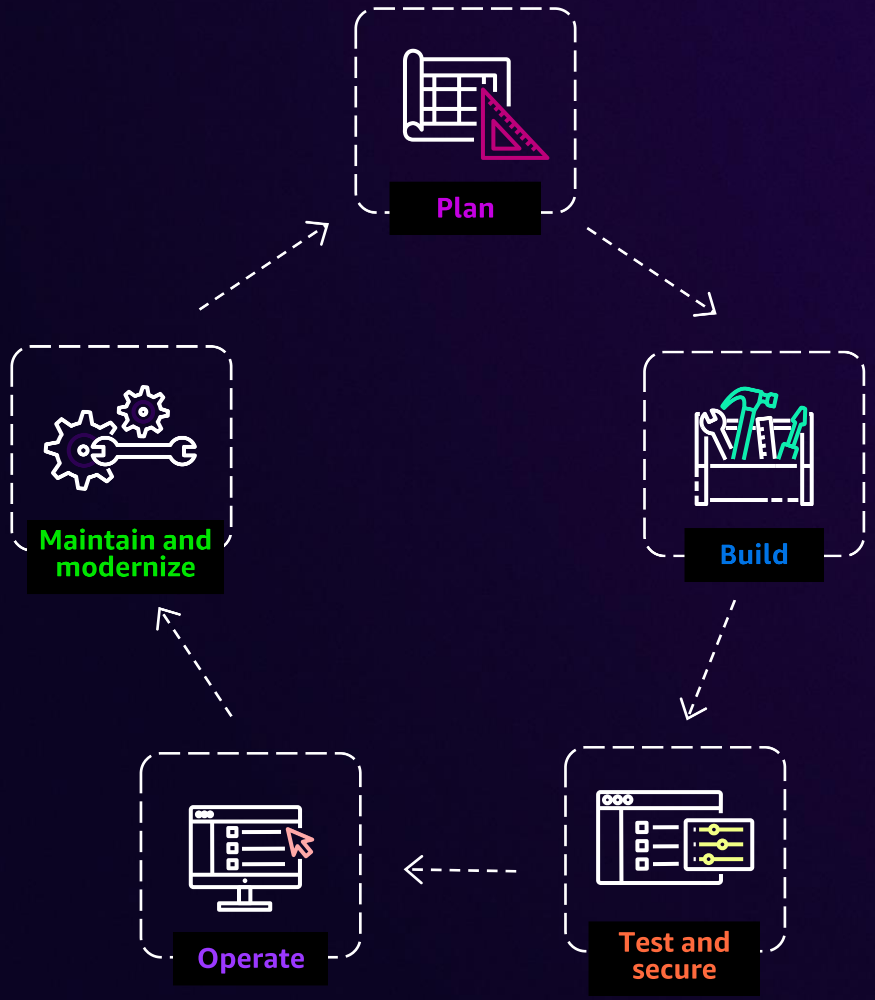
Amazon Q agent vision



Amazon Q agent vision



Amazon Q Developer agents (QDA)



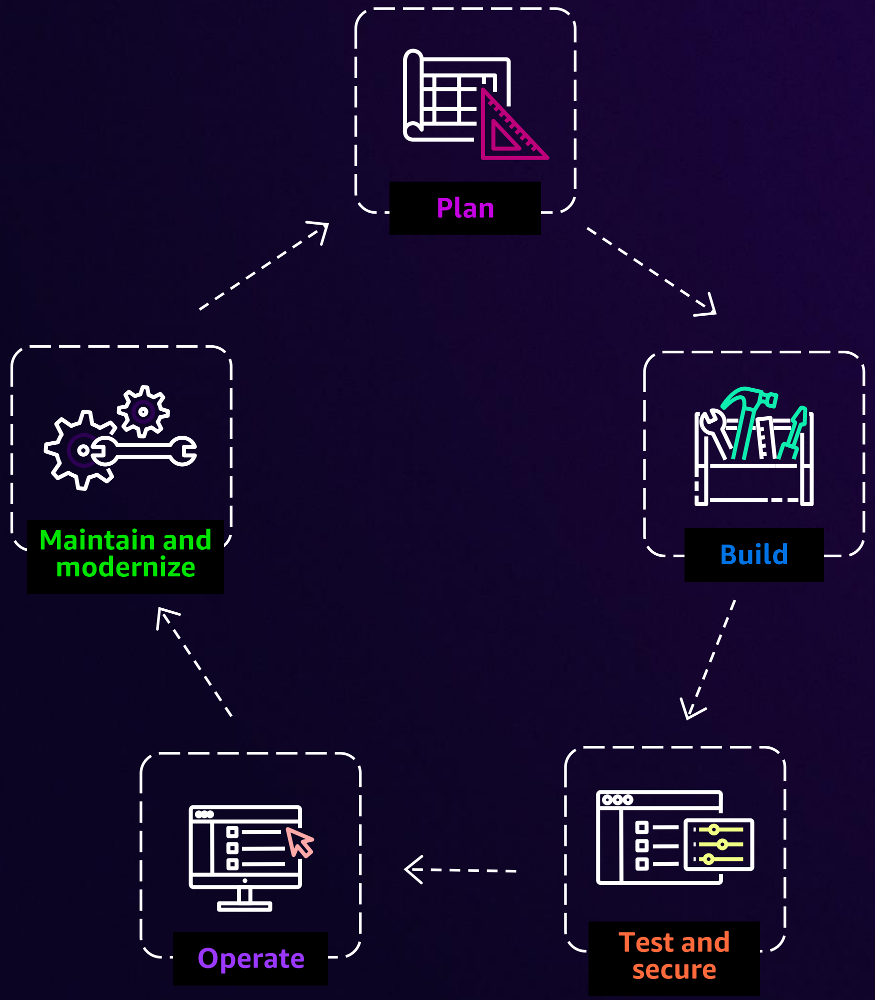
Code transformation

Complete Java language upgrades in a fraction of the time (/transform)

Amazon Q Developer agents (QDA)

Code transformation

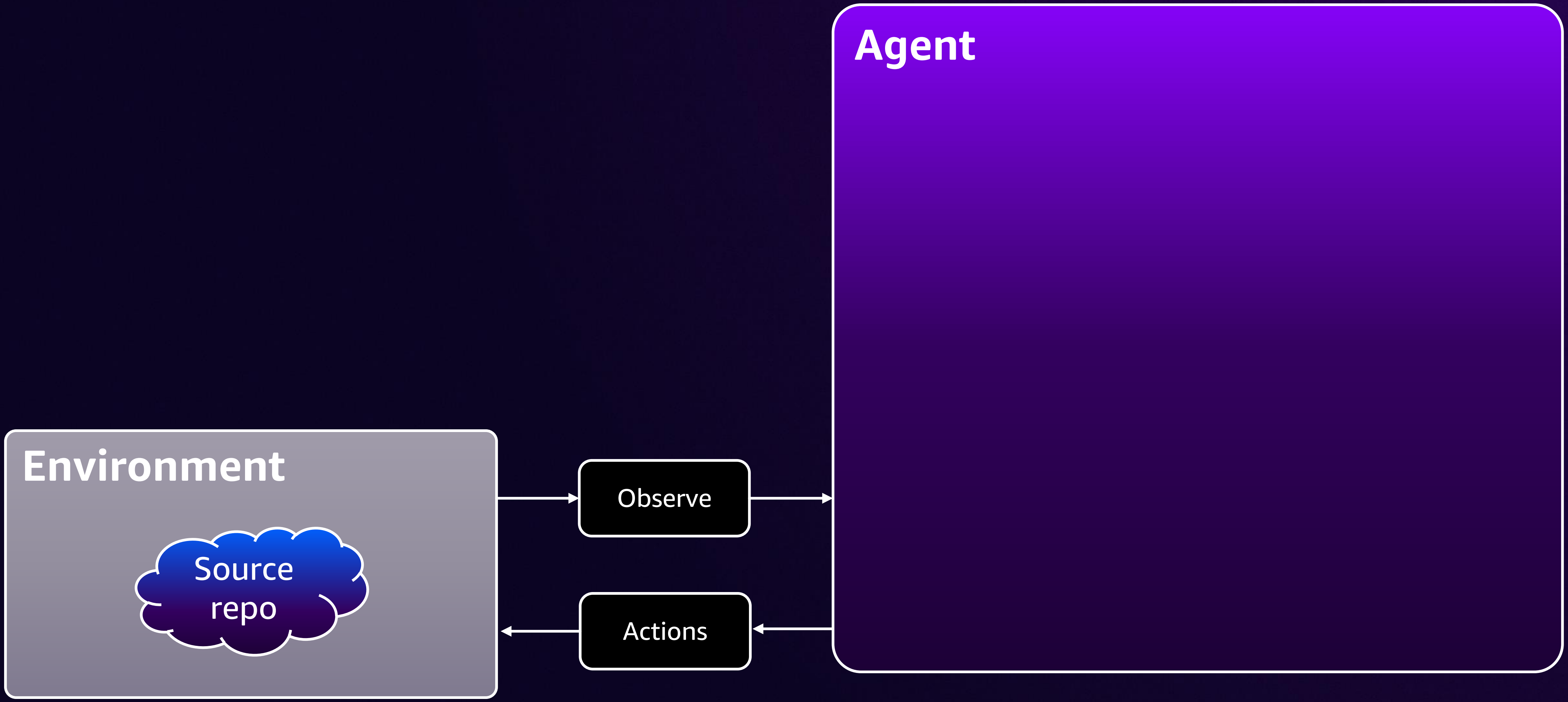
Complete Java language upgrades in a fraction of the time (/transform)



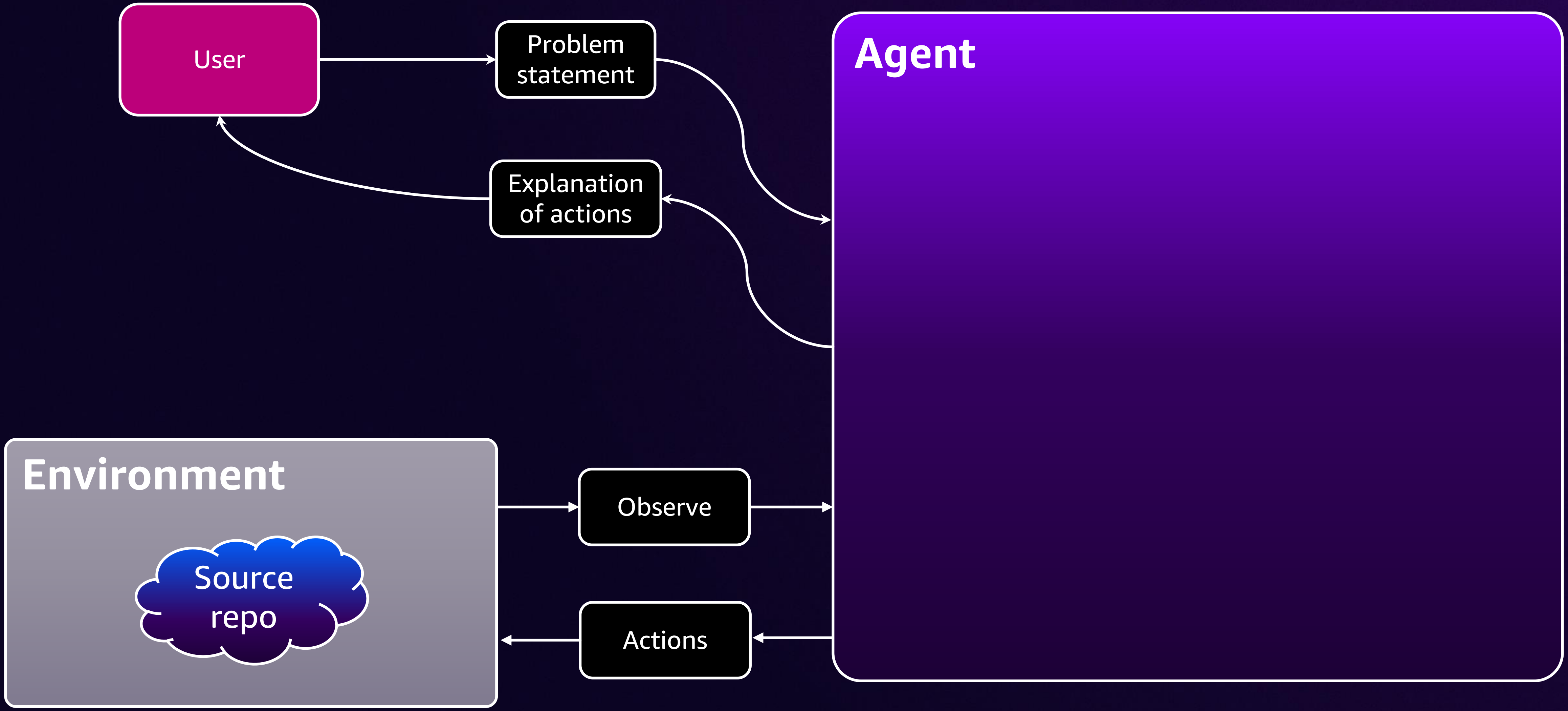
Software development

Feature development from natural language input to merge-ready code across multiple files (/dev)

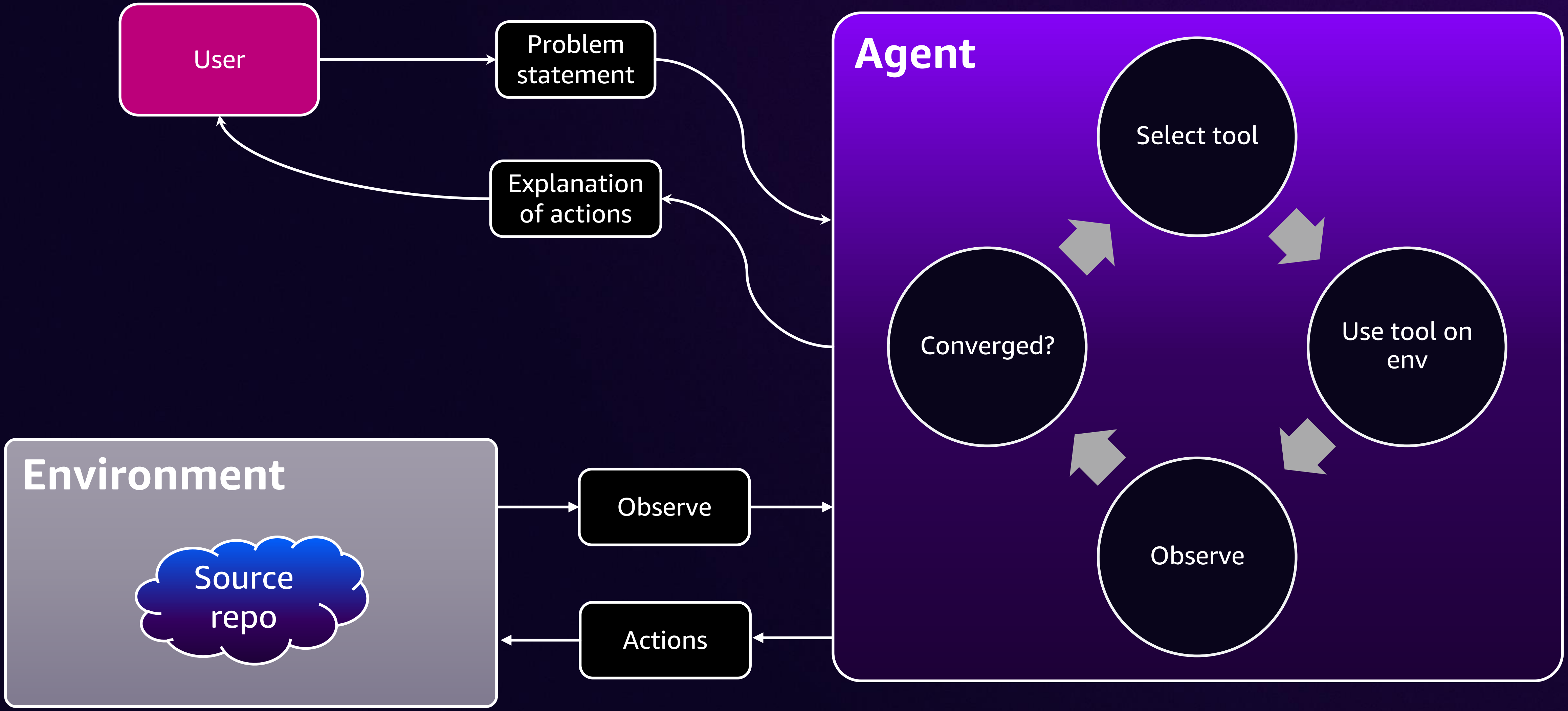
Q software development agent architecture



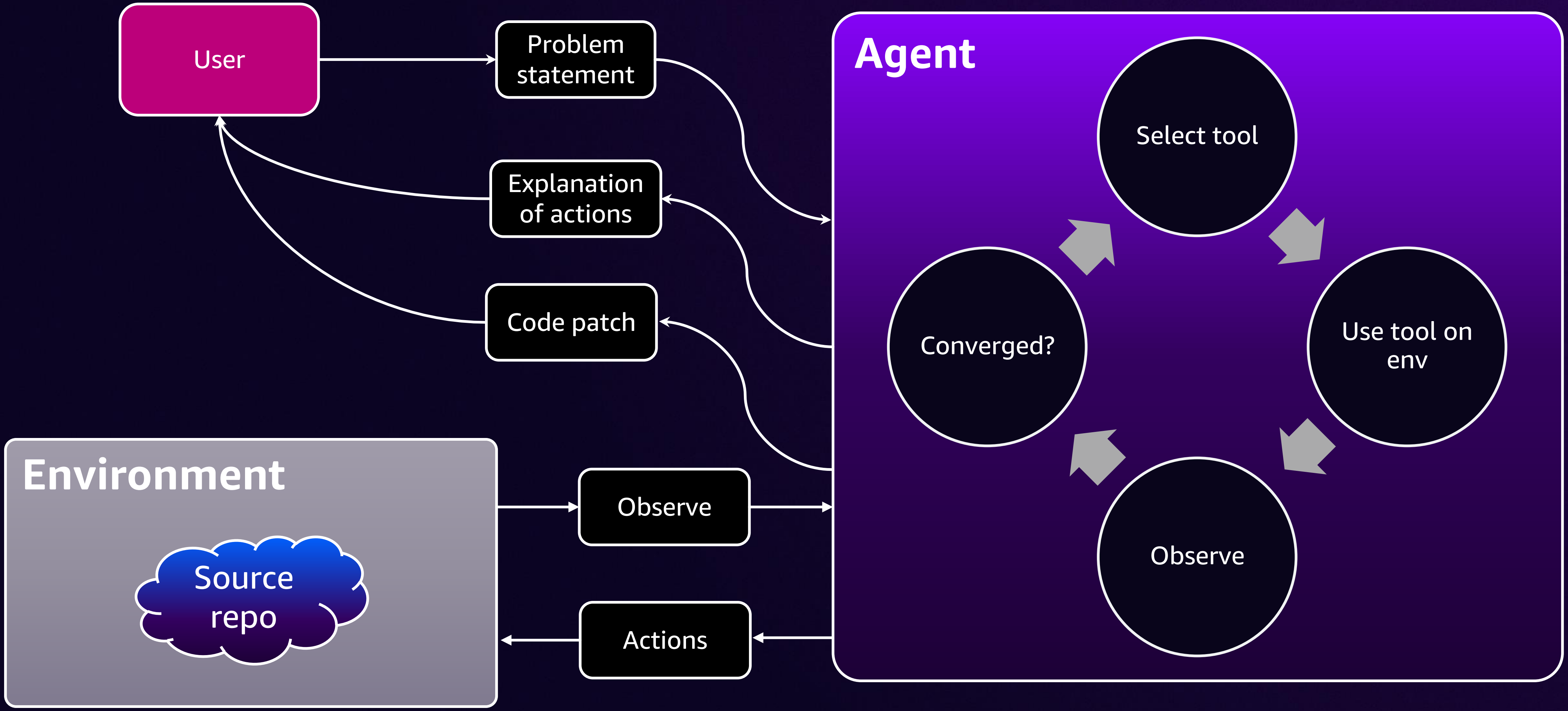
Q software development agent architecture

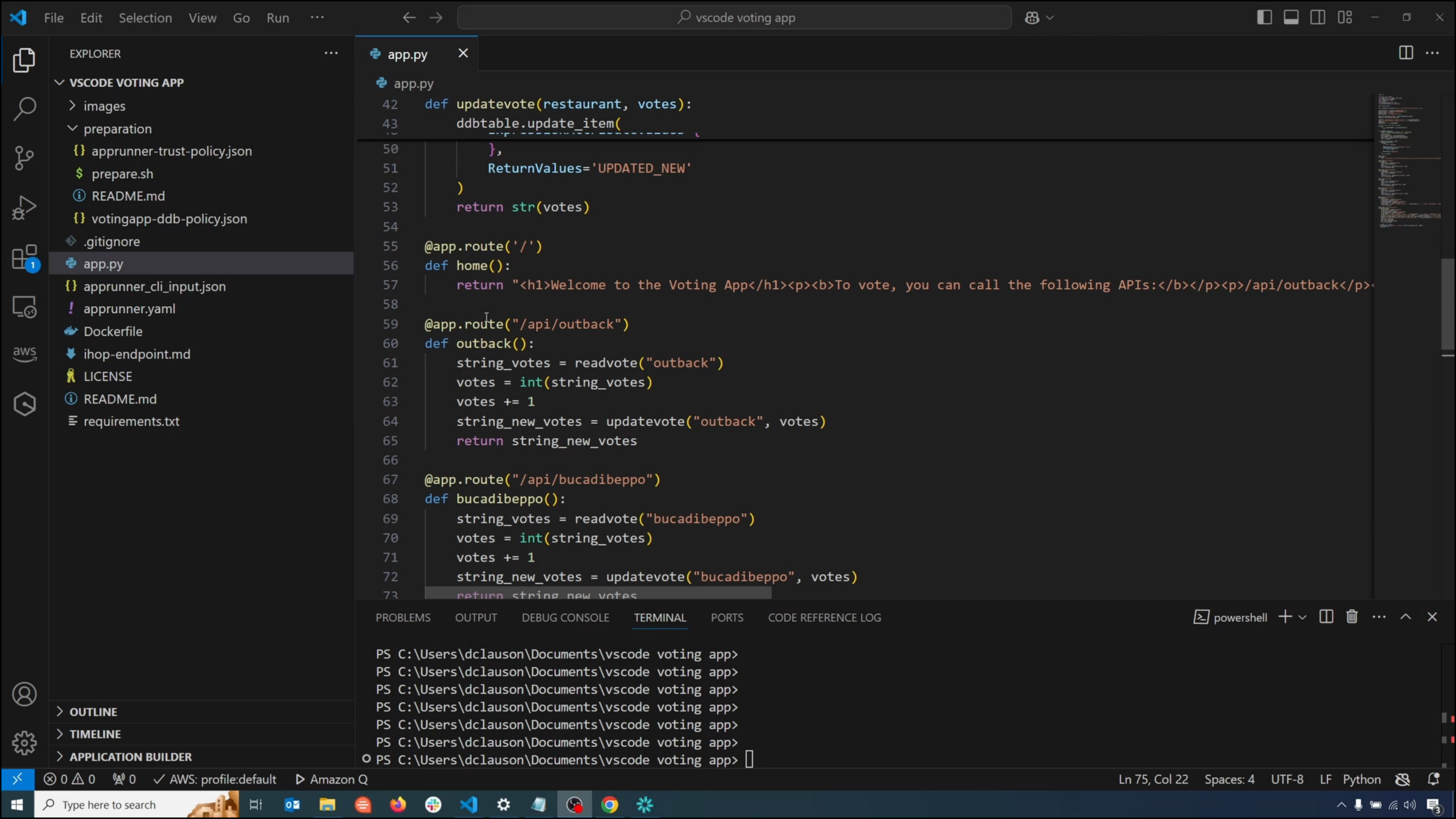


Q software development agent architecture



Q software development agent architecture





```
def updatevote(restaurant, votes):
    ddbtable.update_item(
        },
        ReturnValues='UPDATED_NEW'
    )
    return str(votes)

@app.route('/')
def home():
    return "<h1>Welcome to the Voting App</h1><p><b>To vote, you can call the following APIs:</b></p><p>/api/outback</p>

@app.route("/api/outback")
def outback():
    string_votes = readvote("outback")
    votes = int(string_votes)
    votes += 1
    string_new_votes = updatevote("outback", votes)
    return string_new_votes

@app.route("/api/bucadibeppo")
def bucadibeppo():
    string_votes = readvote("bucadibeppo")
    votes = int(string_votes)
    votes += 1
    string_new_votes = updatevote("bucadibeppo", votes)
    return string_new_votes
```

```
PS C:\Users\dclauson\Documents\vscode voting app>
PS C:\Users\dclauson\Documents\vscode voting app>
PS C:\Users\dclauson\Documents\vscode voting app>
PS C:\Users\dclauson\Documents\vscode voting app>
PS C:\Users\dclauson\Documents\vscode voting app>
PS C:\Users\dclauson\Documents\vscode voting app>
PS C:\Users\dclauson\Documents\vscode voting app>
```

What’s next for Amazon Q Developer agents

NEW

Amazon Q Developer streamlines unit test generation

Automate the end-to-end process of generating unit tests

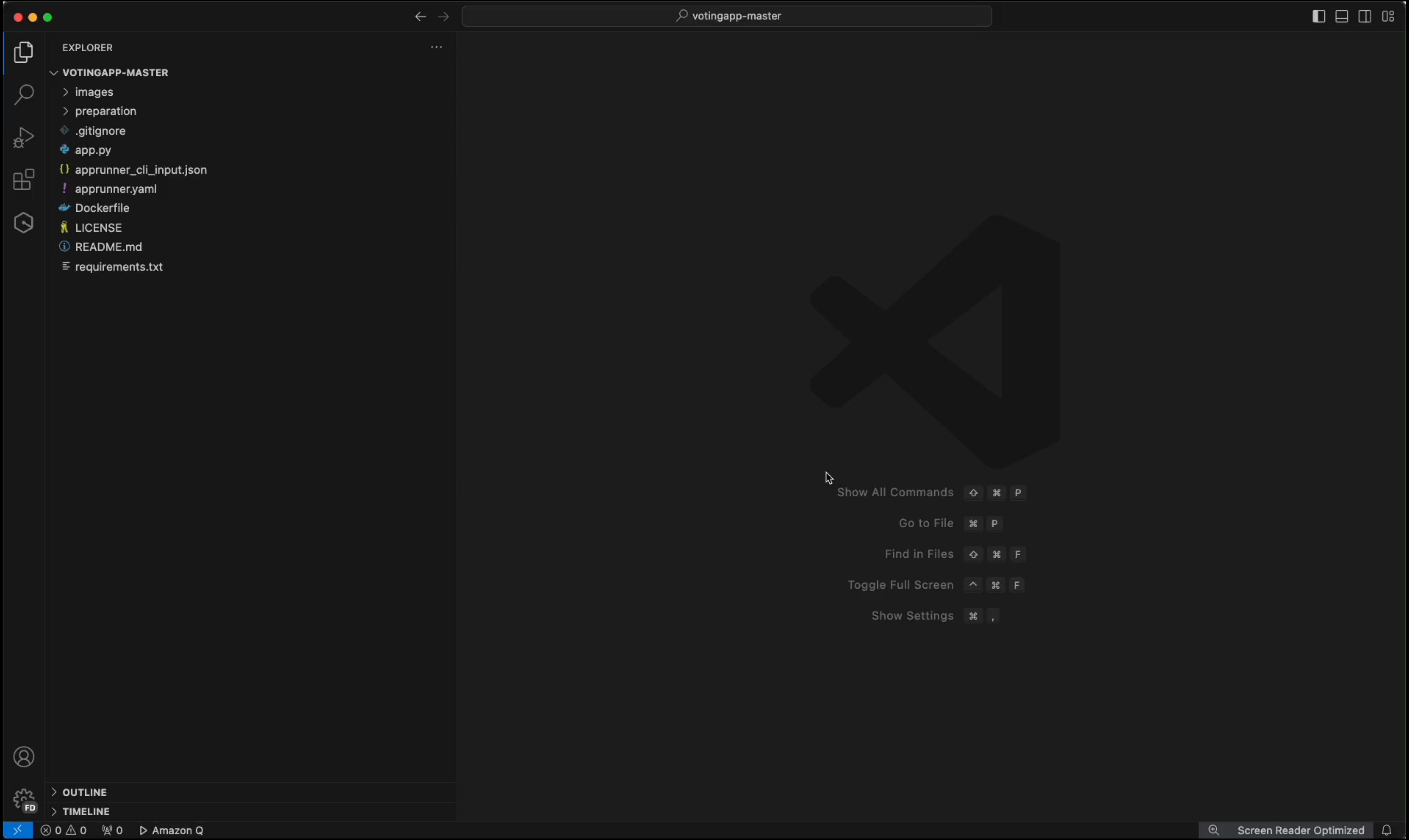
GENERALLY AVAILABLE

Reduce developer time and effort

Ship code more reliably with better unit test coverage

Improve code reliability, maintainability, and effectiveness

aws



votingapp-master
EXPLORER
VOTINGAPP-MASTER
images
preparation
.gitignore
app.py
apprunner_cli_input.json
apprunner.yaml
Dockerfile
LICENSE
README.md
requirements.txt
Show All Commands ⇧ ⌘ P
Go to File ⌘ P
Find in Files ⇧ ⌘ F
Toggle Full Screen ^ ⌘ F
Show Settings ⌘ ,
OUTLINE
TIMELINE
FD
⊗ 0 ⚠ 0
0
Amazon Q
Screen Reader Optimized

NEW

Amazon Q Developer generates documentation

Speed up understanding of your code base

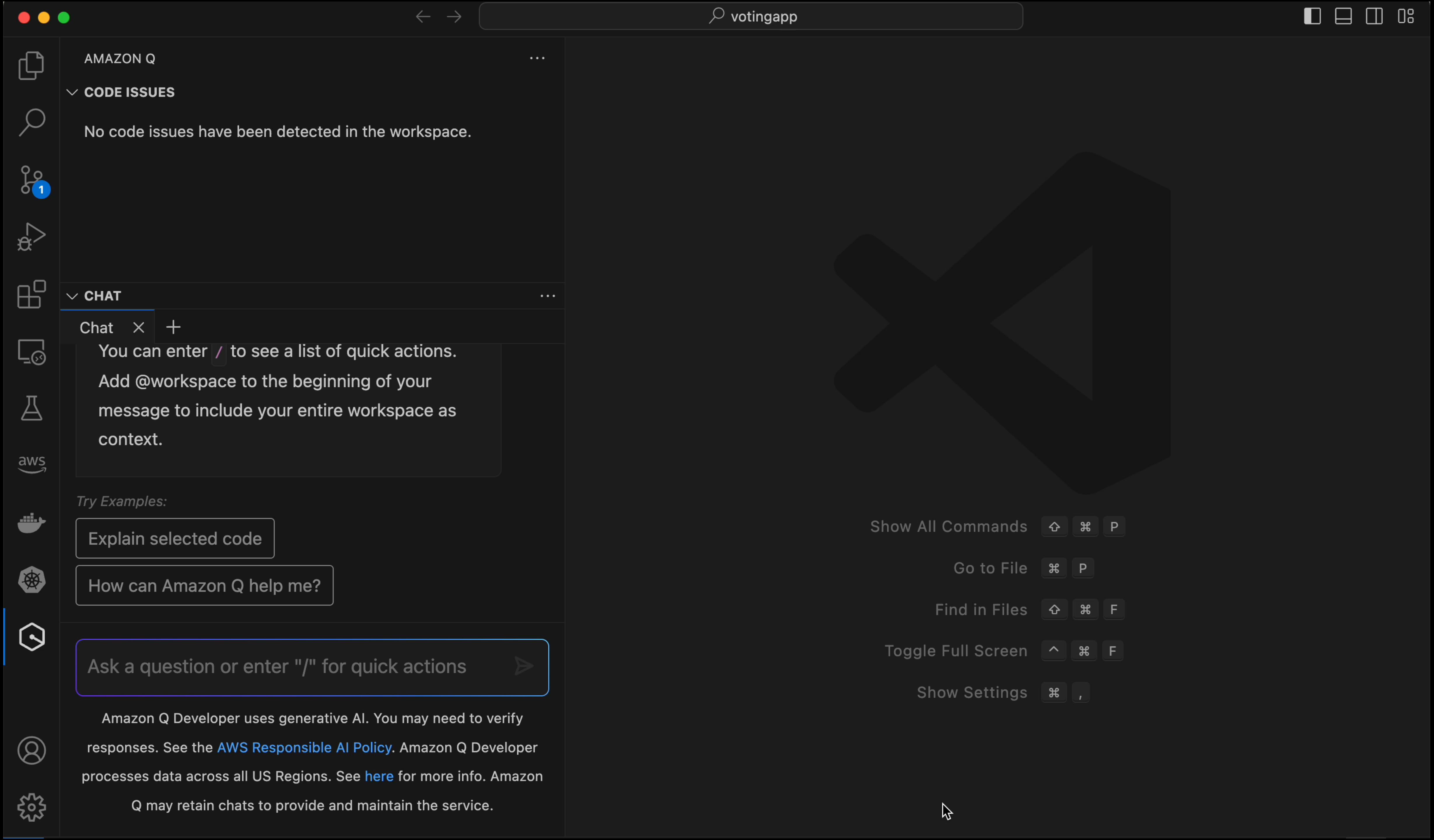
GENERALLY AVAILABLE TODAY

Easily keep projects well documented for team mates

Onboard to new code bases in a fraction of the time

Save hours while following code quality best practices

aws



No code issues have been detected in the workspace.

You can enter / to see a list of quick actions. Add @workspace to the beginning of your message to include your entire workspace as context.

Amazon Q Developer uses generative AI. You may need to verify responses. See the AWS Responsible AI Policy. Amazon Q Developer processes data across all US Regions. See here for more info. Amazon Q may retain chats to provide and maintain the service.

NEW

Amazon Q Developer automates code reviews

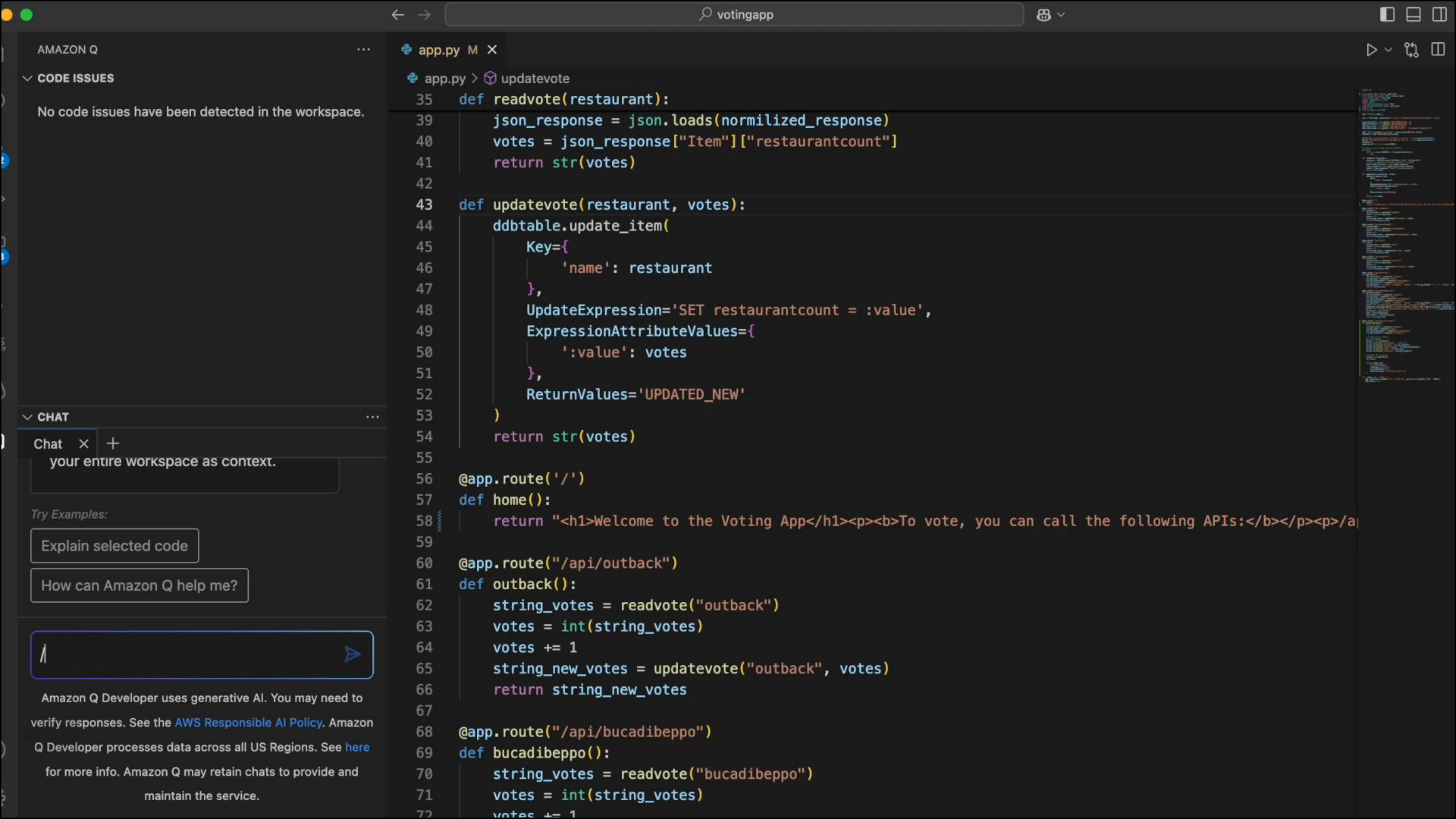
Save time, improve code quality, and ship with confidence

GENERALLY AVAILABLE TODAY

Automatically detect security vulnerabilities and code quality issues

Resolve issues in two clicks instead of hours

Assess and remediate deployment risk



No code issues have been detected in the workspace.

your entire workspace as context.

Amazon Q Developer uses generative AI. You may need to verify responses. See the AWS Responsible AI Policy. Amazon Q Developer processes data across all US Regions. See here for more info. Amazon Q may retain chats to provide and maintain the service.

```
def readvote(restaurant):
    json_response = json.loads(normilized_response)
    votes = json_response["Item"]["restaurantcount"]
    return str(votes)

def updatevote(restaurant, votes):
    ddbtable.update_item(
        Key={
            'name': restaurant
        },
        UpdateExpression='SET restaurantcount = :value',
        ExpressionAttributeValues={
            ':value': votes
        },
        ReturnValues='UPDATED_NEW'
    )
    return str(votes)

@app.route('/')
def home():
    return "<h1>Welcome to the Voting App</h1><p><b>To vote, you can call the following APIs:</b></p><p>/a

@app.route("/api/outback")
def outback():
    string_votes = readvote("outback")
    votes = int(string_votes)
    votes += 1
    string_new_votes = updatevote("outback", votes)
    return string_new_votes

@app.route("/api/bucadibeppo")
def bucadibeppo():
    string_votes = readvote("bucadibeppo")
    votes = int(string_votes)
    votes += 1
```

NEW

GitLab Duo with Amazon Q

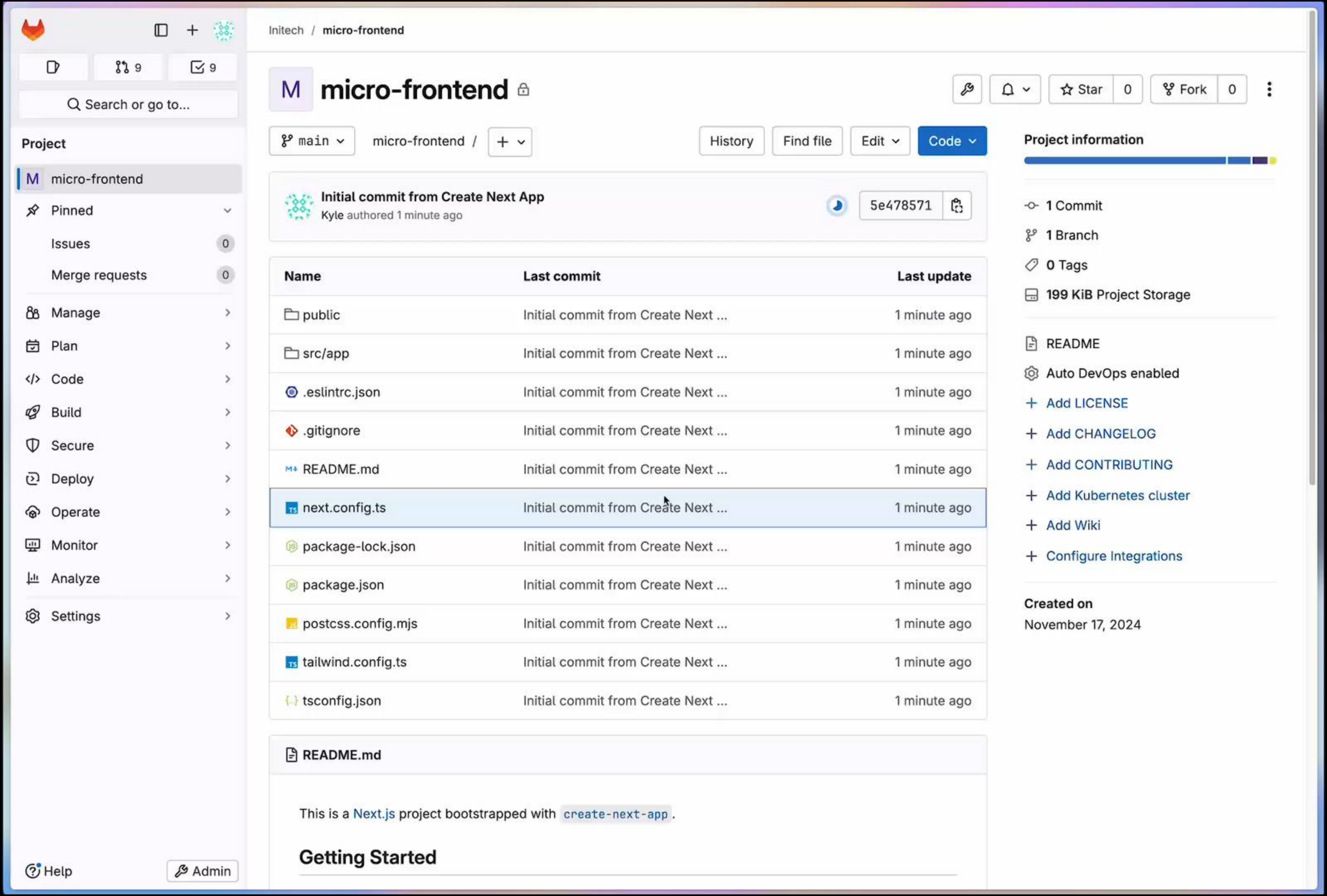
AI-driven DevSecOps

NOW IN PREVIEW

Utilize Amazon Q Developer agents across the entire software development lifecycle right from GitLab

A seamless developer experience

Available for Self Managed Ultimate tier subscriptions



Initech / micro-frontend

micro-frontend

micro-frontend /

Initial commit from Create Next App
Kyle authored 1 minute ago

5e478571

Name	Last commit	Last update
public	Initial commit from Create Next ...	1 minute ago
src/app	Initial commit from Create Next ...	1 minute ago
.eslintrc.json	Initial commit from Create Next ...	1 minute ago
.gitignore	Initial commit from Create Next ...	1 minute ago
README.md	Initial commit from Create Next ...	1 minute ago
next.config.ts	Initial commit from Create Next ...	1 minute ago
package-lock.json	Initial commit from Create Next ...	1 minute ago
package.json	Initial commit from Create Next ...	1 minute ago
postcss.config.mjs	Initial commit from Create Next ...	1 minute ago
tailwind.config.ts	Initial commit from Create Next ...	1 minute ago
tsconfig.json	Initial commit from Create Next ...	1 minute ago

Project information

- 1 Commit
- 1 Branch
- 0 Tags
- 199 KiB Project Storage

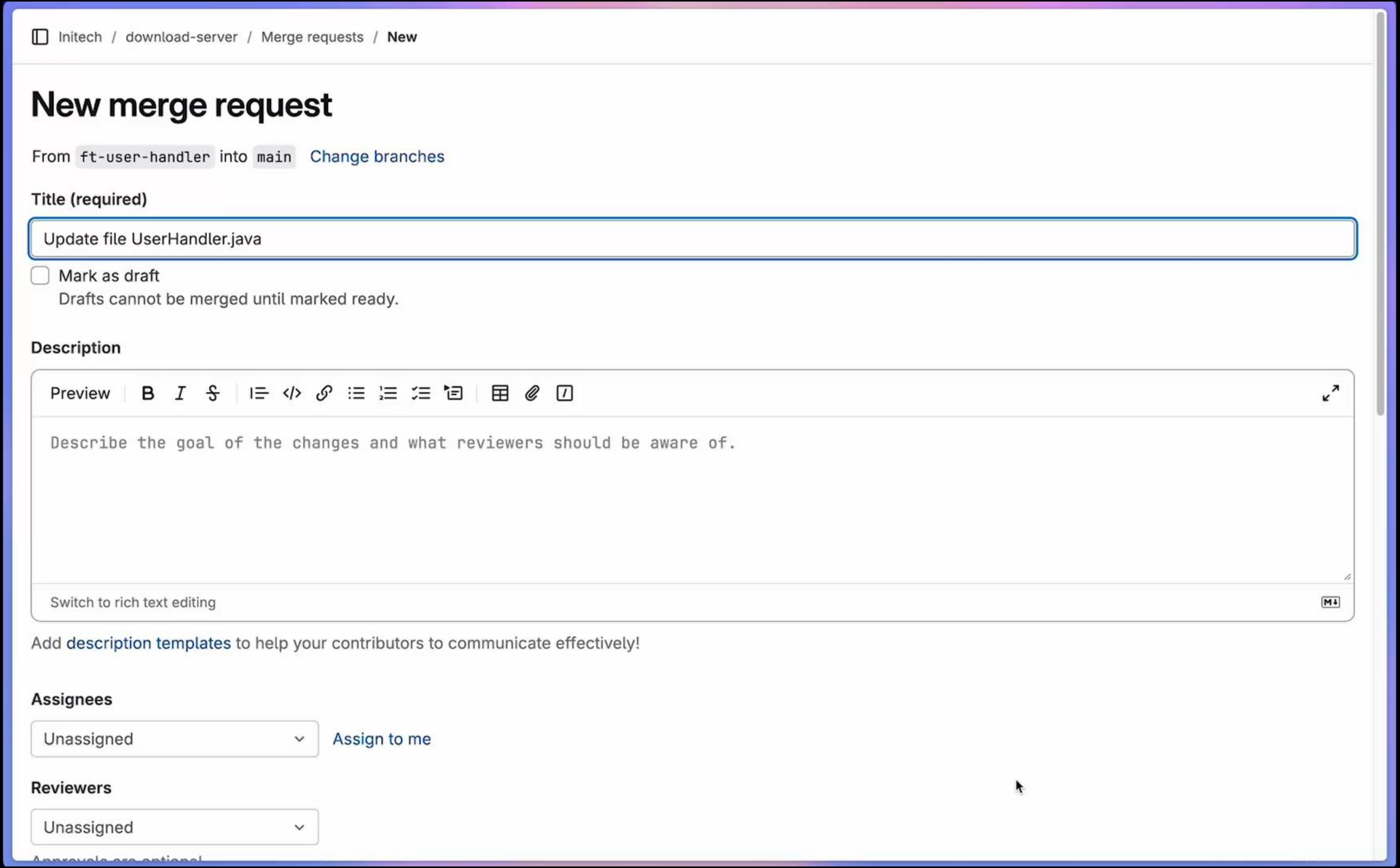
- README
- Auto DevOps enabled
- + Add LICENSE
- + Add CHANGELOG
- + Add CONTRIBUTING
- + Add Kubernetes cluster
- + Add Wiki
- + Configure Integrations

Created on
November 17, 2024

README.md

This is a Next.js project bootstrapped with `create-next-app`.

Getting Started



Initech / download-server / Merge requests / New

New merge request

From `ft-user-handler` into `main` Change branches

Title (required)

Update file UserHandler.java

□ Mark as draft

Drafts cannot be merged until marked ready.

Description

Describe the goal of the changes and what reviewers should be aware of.

Add description templates to help your contributors to communicate effectively!

Assignees

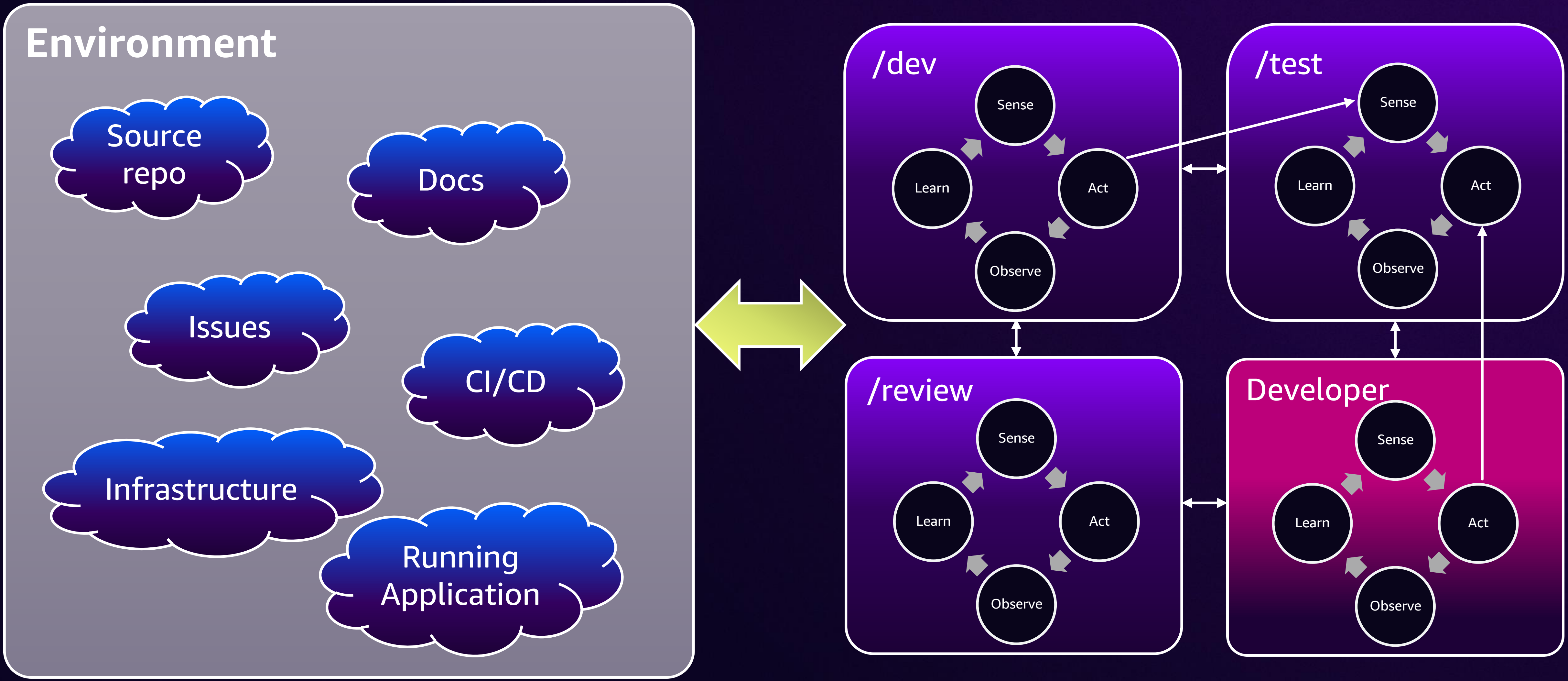
Unassigned

Reviewers

Unassigned

Approvals are optional

Agents working together



Best practices using Amazon Q Developer agents

- Start with a particular use case in mind
- Merge agents into your daily workflows
- Use unit and system test to validate changes
- Measure outcomes

Take the next steps

Install the Q extension inside your IDE or use GitLab

Explore the agent capabilities

Let’s go build!

Thank you!

☆☆☆☆☆ Please complete the session survey in the mobile app

Doug Clauson
linkedin.com/in/doug-clauson/

Johnna Powell
linkedin.com/in/johnnapowell/

Manikandan Srinivasan
linkedin.com/in/srinivm/