

# Amazon Q Developer and Kiro Introduction

**Michael Lin**

Sr. Solutions Architect  
Amazon Web Services



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

# Agenda

- Introducing Q Developer and Kiro
- Customizing for Code Recommendations
- Modernizing .NET and Java Applications
- Cloud Hosting with Infrastructure as Code
- Getting Started

# AI coding assistants' adoption

By 2028, 75% of enterprise software engineers will use AI code assistants, up from less than 10% in early 2023.

Gartner® Press Release, Gartner Says 75% of Enterprise Software Engineers Will Use AI Code Assistants by 2028, April 11, 2024,  
<https://www.gartner.com/en/newsroom/press-releases/2024-04-11-gartner-says-75-percent-of-enterprise-software-engineers-will-use-ai-code-assistants-by-2028>. GARTNER is a registered trademark and service mark of Gartner, Inc. and/or its affiliates in the U.S. and internationally and is used herein with permission. All rights reserved.

# Amazon Q Developer



Helps developers and IT professionals build faster across the entire software development lifecycle (SDLC)

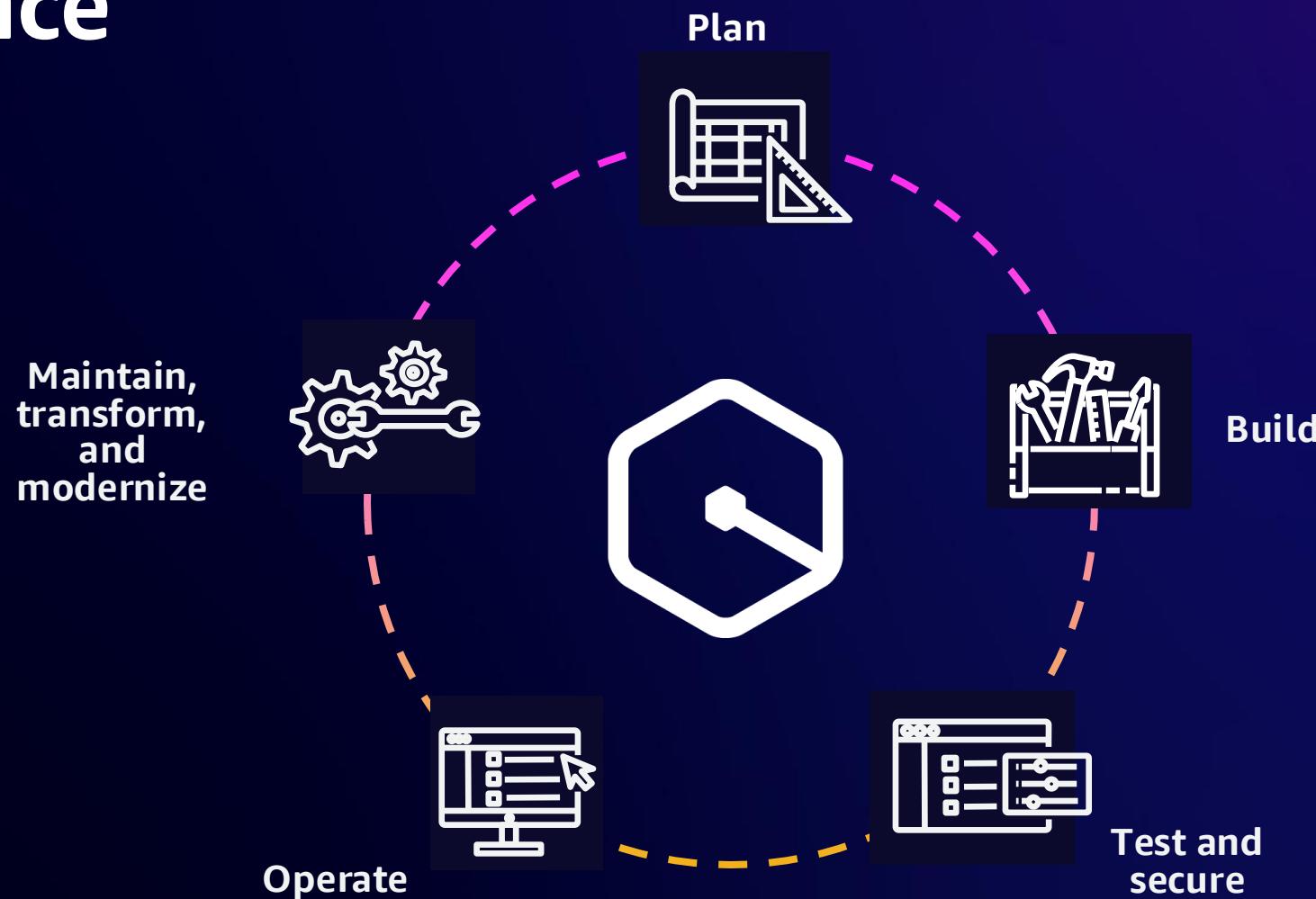
Most accurate coding recommendations

Agents can autonomously help you implement features, refactor code, perform software upgrades and more

Best-in-class security vulnerability scanning and remediation

**Amazon Q is built with security and privacy in mind from the start, making it easier for organizations to use generative AI safely.**

# Amazon Q Developer transforms the development experience



# Amazon Q dashboard

## METRICS WE TRACK

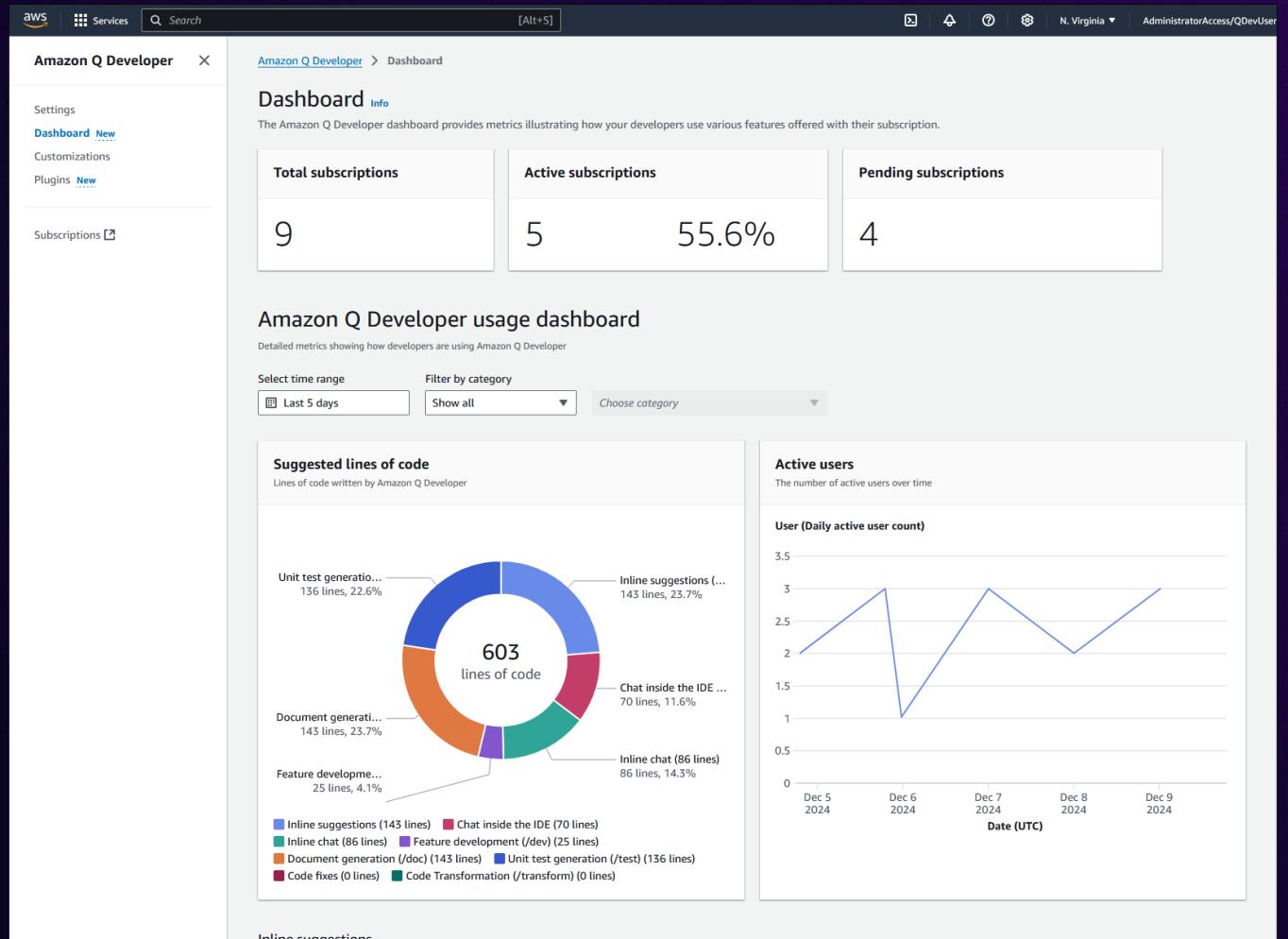
User activity

Lines of code generated

Acceptance rate

Percentage of code written  
by Amazon Q Developer

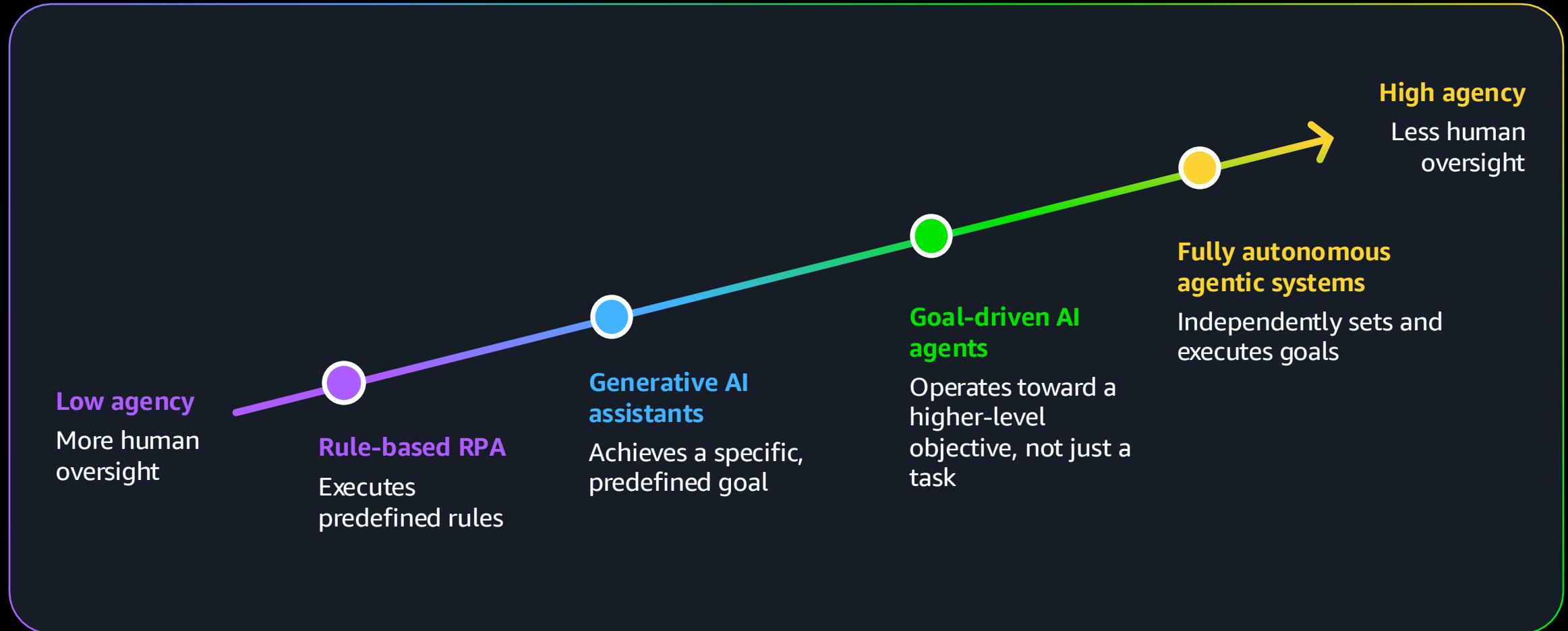
Accepted recommendations  
with references



# Recent Updates in Q Developer IDE, CLI, and Kiro



# Agentic AI Maturity – From Rules to Reasoning



# Recent Updates on Agentic Coding in CLI and IDE

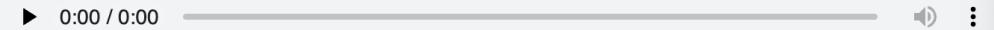
## A lightning fast, new agentic coding experience within the Amazon Q Developer CLI

by Brian Beach | on 06 MAR 2025 | in [Amazon Q](#), [Amazon Q Developer](#), [Announcements](#) | [Permalink](#) | [Share](#)

Earlier today, [Amazon Q Developer](#) announced [an enhanced CLI agent](#) within the [Amazon Q command line interface \(CLI\)](#). With this announcement, Q Developer brings the latest agentic experience to the CLI that provide a more dynamic, interactive coding experience that works with you, and iteratively makes changes based on your feedback. Amazon Q Developer can now use the information in your CLI environment to help you read and write files locally, query AWS resources write code, or automatically debug issues.

## Amazon Q Developer elevates the IDE experience with new agentic coding experience

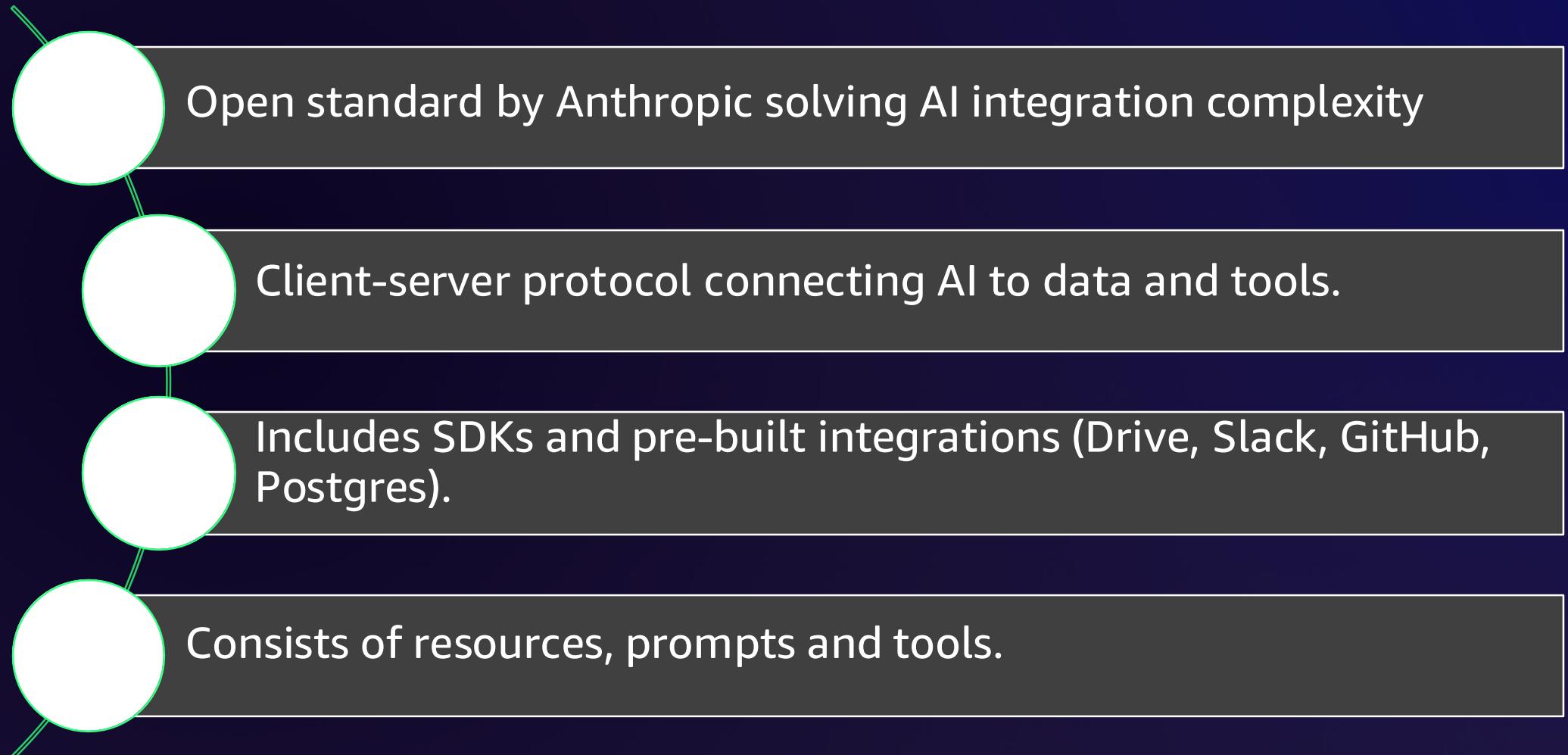
by Elizabeth Fuentes | on 02 MAY 2025 | in [Amazon Q Developer](#), [Announcements](#), [AWS CLI](#), [Developer Tools](#), [Featured](#), [Launch](#), [News](#) | [Permalink](#) | [Comments](#) | [Share](#)



Today, [Amazon Q Developer](#) introduces a new, interactive, agentic coding experience that is now available in the [integrated development environments \(IDE\)](#) for [Visual Studio Code](#). This experience brings interactive coding capabilities, building upon existing prompt-based features. You now have a natural, real-time collaborative partner working alongside you while writing code, creating documentation, running tests, and reviewing changes.

<https://aws.amazon.com/blogs/devops/introducing-the-enhanced-command-line-interface-in-amazon-q-developer/>  
<https://aws.amazon.com/blogs/aws/amazon-q-developer-elevates-the-ide-experience-with-new-agentic-coding-experience/>

# Model Context Protocol (MCP)



# Latest MCP Support in CLI and Context-Aware IDE

## Extend the Amazon Q Developer CLI with Model Context Protocol (MCP) for Richer Context

by Brian Beach | on 29 APR 2025 | in [Amazon Aurora](#), [Amazon Q Developer](#), [Announcements](#), [PostgreSQL compatible](#), [RDS for PostgreSQL](#) | [Permalink](#) |  Share

Earlier today, [Amazon Q Developer](#) announced [Model Context Protocol \(MCP\) support](#) in the command line interface (CLI). Developers can connect external data sources to Amazon Q Developer CLI with MCP support for more context-aware responses. By integrating MCP tools and prompts into Q Developer CLI, you get access to an expansive list of pre-built integrations or any MCP Servers that support `stdio`. This extra context helps Q Developer write more accurate code, understand your data structures, generate appropriate unit tests, create database documentation, and execute precise queries, all without needing to develop custom integration code. By extending Q Developer with MCP tools and prompts, developers can execute development tasks faster, streamlining the developer experience. At AWS, we're committed to supporting popular open source protocols for agents like Model Context Protocol (MCP) proposed by Anthropic. We'll continue to support this effort by extending this functionality within the Amazon Q Developer IDE plugins in the coming weeks.

## Use Model Context Protocol with Amazon Q Developer for context-aware IDE workflows

by Ritik Khatwani | on 12 JUN 2025 | in [Amazon Q](#), [Amazon Q Developer](#), [Announcements](#), [Developer Tools](#), [Generative AI](#) | [Permalink](#) |  Share

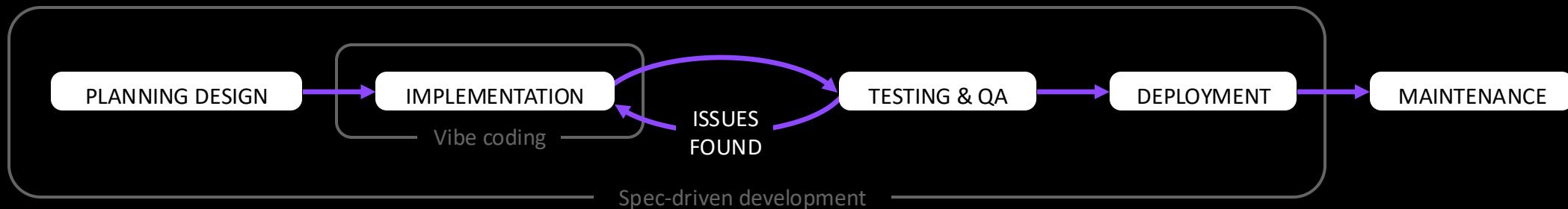
Earlier today, [Amazon Q Developer](#) announced Model Context Protocol (MCP) support in their Integrated Development Environment (IDE) plugins for Visual Studio Code and JetBrains. This allows developers to connect external tools or MCP servers to Q Developer, enabling more context-aware responses and complex workflows. MCP support has already been available in [Amazon Q Developer for Command Line](#) since April 29, 2025.

<https://aws.amazon.com/blogs/devops/extend-the-amazon-q-developer-cli-with-mcp/>  
<https://aws.amazon.com/blogs/devops/use-model-context-protocol-with-amazon-q-developer-for-context-aware-ide-workflows/>



# Spec-Driven Development

Vibe coding ships the prototype,  
traditional SDLC practices ship the product



Kiro helps developers and engineering teams  
ship high quality software with AI agents

# Kiro: The AI IDE for Prototype to Production

- Kiro turns your prompt into clear requirements, system design, and discrete tasks
- Iterate with Kiro on your spec and architecture
- Kiro agents implement the spec while keeping you in control

The screenshot shows the Kiro IDE interface. On the left, there's a sidebar with sections for 'SPECs' (containing 'event-categories' and 'kiro-events-api'), 'AGENT HOOKS' (with 'Translation Monitor'), 'AGENT STEERING' (with 'product', 'structure', and 'tech'), and 'MCP SERVERS' (with 'fetch Connected' and 'jira Connected'). The main workspace is a code editor for a file named 'requirements.md' under the path '.kiro > specs > event-categories'. The code is a Requirements Document:

```
1 # Requirements Document
2
3 ## Introduction
4
5 This feature adds event categorization capabilities to the KiroEvents API, allowing events to be organized by predefined and custom categories. The system will support both built-in categories (Meeting, Social, Training) and administrator-managed custom categories, with full CRUD operations for category management and the ability to assign categories to events.
6
7 ## Requirements
8
9 ### Requirement 1
10
11 **User Story:** As an event organizer, I want to assign categories to events, so that events can be organized and filtered by type.
12
13 #### Acceptance Criteria
14
15 1. WHEN creating a new event THEN the system SHALL allow assignment of a category from available categories
16 2. WHEN updating an existing event THEN the system SHALL allow changing the event's category
17 3. WHEN retrieving events THEN the system SHALL include the category information in the response
18 4. IF no category is specified during event creation THEN the system SHALL allow the event to have no category (optional field)
19
20 ### Requirement 2
21
22 **User Story:** As a system user, I want to view available event
```

At the top of the code editor, there are three tabs: 'Requirements' (highlighted with a red box), 'Design', and 'Task list'. To the right of the code editor, there's a 'Kiro' panel with a chat history:

- I'll help you create a spec for adding event categories to the backend API. First, let me search for the Jira issue in the "Kiro Demo" project to understand the requirements.
- > Called MCP tool jira\_search
- Now let me get my user profile to assign the issue to myself:
- > Called MCP tool jira\_get\_user\_profile
- Let me assign the issue to you:
- > Called MCP tool jira\_update\_issue
- Perfect! I've assigned the Jira issue KIRO-7 to you. Now I'll create a spec for adding event categories to the backend API based on the requirements from the Jira story.
- Created requirements.md
- Do the requirements look good? If so, we can move on to the design.

At the bottom of the Kiro panel, there's a text input field: 'Ask a question or describe a task...' and a button: 'Move to design phase'.

# Q Developer IDE



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

Create a Flask API for managing a to-do list. The API should have endpoints to add a new task, retrieve all tasks, update a task's status, and delete a task. Each task should have an ID, a description, and a status (completed or not completed). Implement basic error handling for invalid requests.

Ok, let me create a plan. This may take a few minutes.

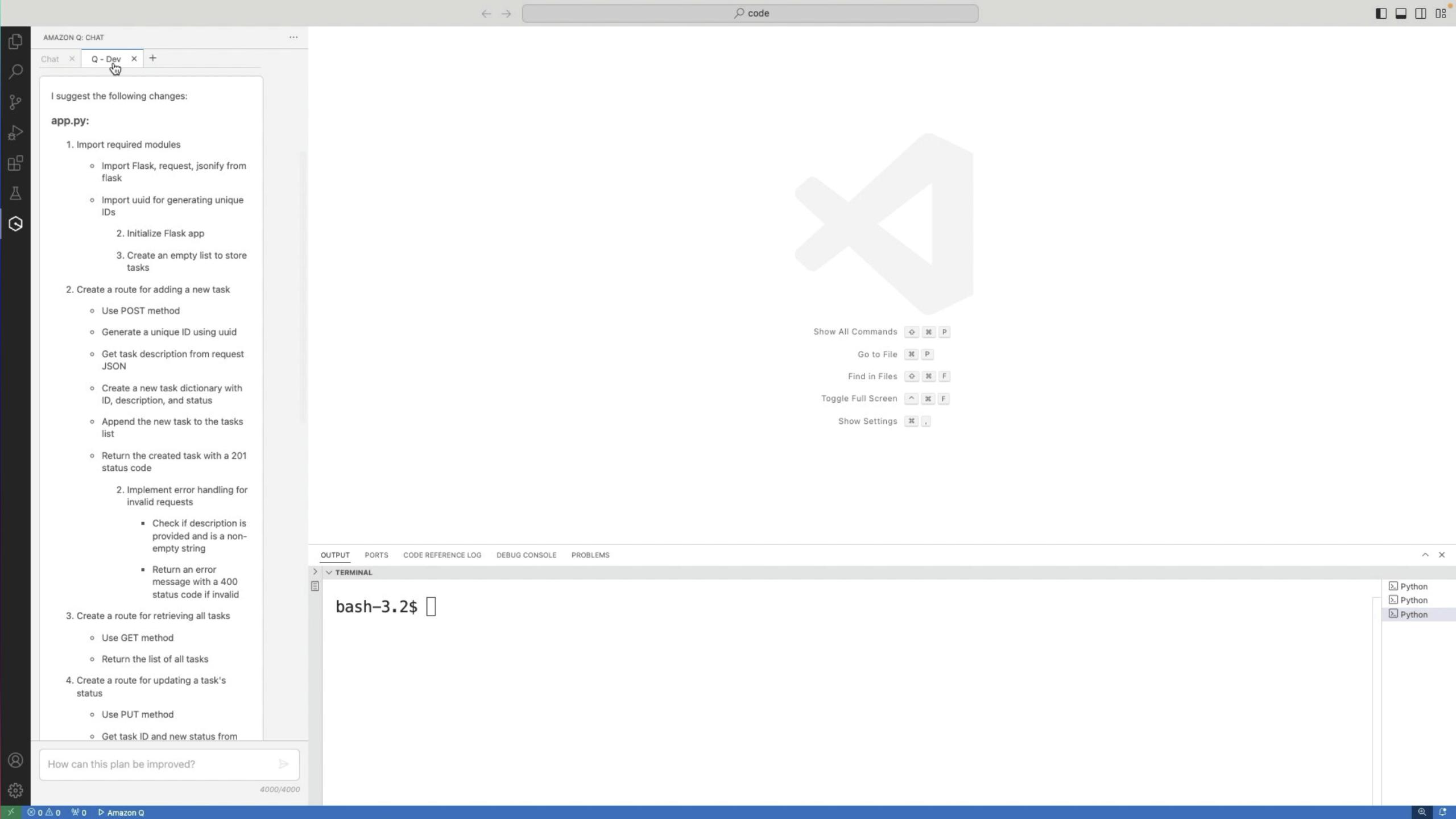
Generating plan ...

OUTPUT PORTS CODE REFERENCE LOG DEBUG CONSOLE PROBLEMS

> < TERMINAL

bash-3.2\$ □





AMAZON Q: CHAT

Chat X Q - Dev X +

I suggest the following changes:

app.py:

1. Import required modules

- o Import Flask, request, jsonify from flask

o Import uuid for generating unique IDs

2. Initialize Flask app

3. Create an empty list to store tasks

2. Create a route for adding a new task

o Use POST method

o Generate a unique ID using uuid

o Get task description from request JSON

o Create a new task dictionary with ID, description, and status

o Append the new task to the tasks list

o Return the created task with a 201 status code

2. Implement error handling for invalid requests

- Check if description is provided and is a non-empty string

- Return an error message with a 400 status code if invalid

3. Create a route for retrieving all tasks

o Use GET method

o Return the list of all tasks

4. Create a route for updating a task's status

o Use PUT method

o Get task ID and new status from

How can this plan be improved? ➤

4000/4000

Show All Commands ⌘ ⌥ P

Go to File ⌘ P

Find in Files ⌘ ⌥ F

Toggle Full Screen ⌘ ⌥ F

Show Settings ⌘ ,

OUTPUT PORTS CODE REFERENCE LOG DEBUG CONSOLE PROBLEMS

> V TERMINAL

bash-3.2\$

Python  
Python  
Python

The image shows a screenshot of a code editor interface, likely Visual Studio Code, displaying a Python file named `app.py`. The code implements a simple REST API for managing tasks using the Flask framework.

```
1 from flask import Flask, request, jsonify
2 import uuid
3
4 app = Flask(__name__)
5
6 tasks = []
7
8 @app.route('/tasks', methods=['POST'])
9 def add_task():
10     data = request.json
11     if not data or 'description' not in data or not isinstance(data['description'], str) or data
12         ['description'].strip() == '':
13             return jsonify({"error": "Invalid task description"}), 400
14
15     task = {
16         "id": str(uuid.uuid4()),
17         "description": data['description'],
18         "status": "not completed"
19     }
20     tasks.append(task)
21     return jsonify(task), 201
```

Below the code editor, there is a terminal window titled "TERMINAL" showing a bash prompt: `bash-3.2$`. To the right of the terminal, there are three small Python icon buttons.

The status bar at the bottom of the screen displays various system and application status indicators, including battery level, signal strength, and the current file path: `Amazon Q`.

← → Home Workspaces API Network

Search Postman

Invite ⚙️ Bell Upgrade

My Workspace New Import GET Get Tasks POST Add Task +

HTTP Amazon Q LiL / Add Task

POST http://127.0.0.1:5000/tasks

Params Authorization Headers (8) Body **Body** Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
1 {
2   "description": "check in code to repo",
3   "status": "not started"
4 }
```

Status: 201 CREATED Time: 5 ms Size: 292 B Save as example

Collections Environments History ChatGPT ChatGPT Action NASA APIs NDC OpenAI API

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize JSON

```
1 {
2   "description": "check in code to repo",
3   "id": "d84309ef-8d00-4a1d-940a-8e987ac93b0f",
4   "status": "not completed"
5 }
```

Online Find and replace Console Postbot Runner Start Proxy Cookies Vault Trash

# Q CLI



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

```
~/aws/amazonq » q chat
✓ calculator loaded in 1.14 s
✓ sqlite loaded in 1.32 s
⚠ tavily_mcp has loaded in 2.01 s with the following warning:
The following tool names are changed:
- tavily_mcp___tavily-search -> tavily_mcp___tavilysearch
- tavily_mcp___tavily-extract -> tavily_mcp___tavilyextract
✓ awslabsaws_documentation_mcp_server loaded in 2.80 s
✓ awslabscore_mcp_server loaded in 2.92 s
```

mba@mbas-MacBook-Air



#### Did you know?

You can use `/compact` to replace the conversation history with its summary to free up the context space

`/help` all commands • `ctrl + j` new lines • `ctrl + s` fuzzy search

🤖 You are chatting with **claude-4-sonnet**

>

> find average salary by department using sqlite database on my system. use tool at your disposal.

> I'll help you find the average salary by department using the SQLite database. Let me start by exploring what tables are available in your database.

🛠 Using tool: list\_tables from mcp server sqlite

● Running list\_tables with the param:

```
{  
  "name": "list_tables",  
  "arguments": {}  
}
```

Allow this action? Use 't' to trust (always allow) this tool for the session. [y/n/t]:

> t

● Completed in 0.54s

> Great! I can see there are two tables: `emp` and `dept`. Let me examine the structure of these tables to understand the data better.

🛠 Using tool: `describe_table` (trusted) from mcp server `sqlite`

- Running `describe_table` with the param:

```
{  
  "name": "describe_table",  
  "arguments": {  
    "table_name": "dept"  
  }  
}
```

- Completed in 0.5s

> Perfect! Now I can see the structure:

- `emp` table has employee data including `sal` (salary) and `deptno` (department number)
- `dept` table has department data including `deptno` and `dname` (department name)

Let me query the average salary by department:

🛠 Using tool: `read_query` from mcp server `sqlite`

- Running `read_query` with the param:

```
{  
  "name": "read_query",  
  "arguments": {  
    "query": "SELECT \n      d.dname as department_name,\n      ROUND(AVG(e.sal), 2) as average_salary,\n      COUNT(e.empno) as employee_count\nFROM emp e\nJOIN dept d ON e.deptno = d.deptno\nWHERE e.sal IS NOT NULL\nGROUP BY d.deptno, d.dname\nORDER BY average_salary DESC"  
  }  
}
```

> Excellent! Here are the results for average salary by department:

## ## Average Salary by Department

Department	Average Salary	Employee Count
ACCOUNTING	\$2,916.67	3 employees
RESEARCH	\$2,175.00	5 employees
SALES	\$1,566.67	6 employees

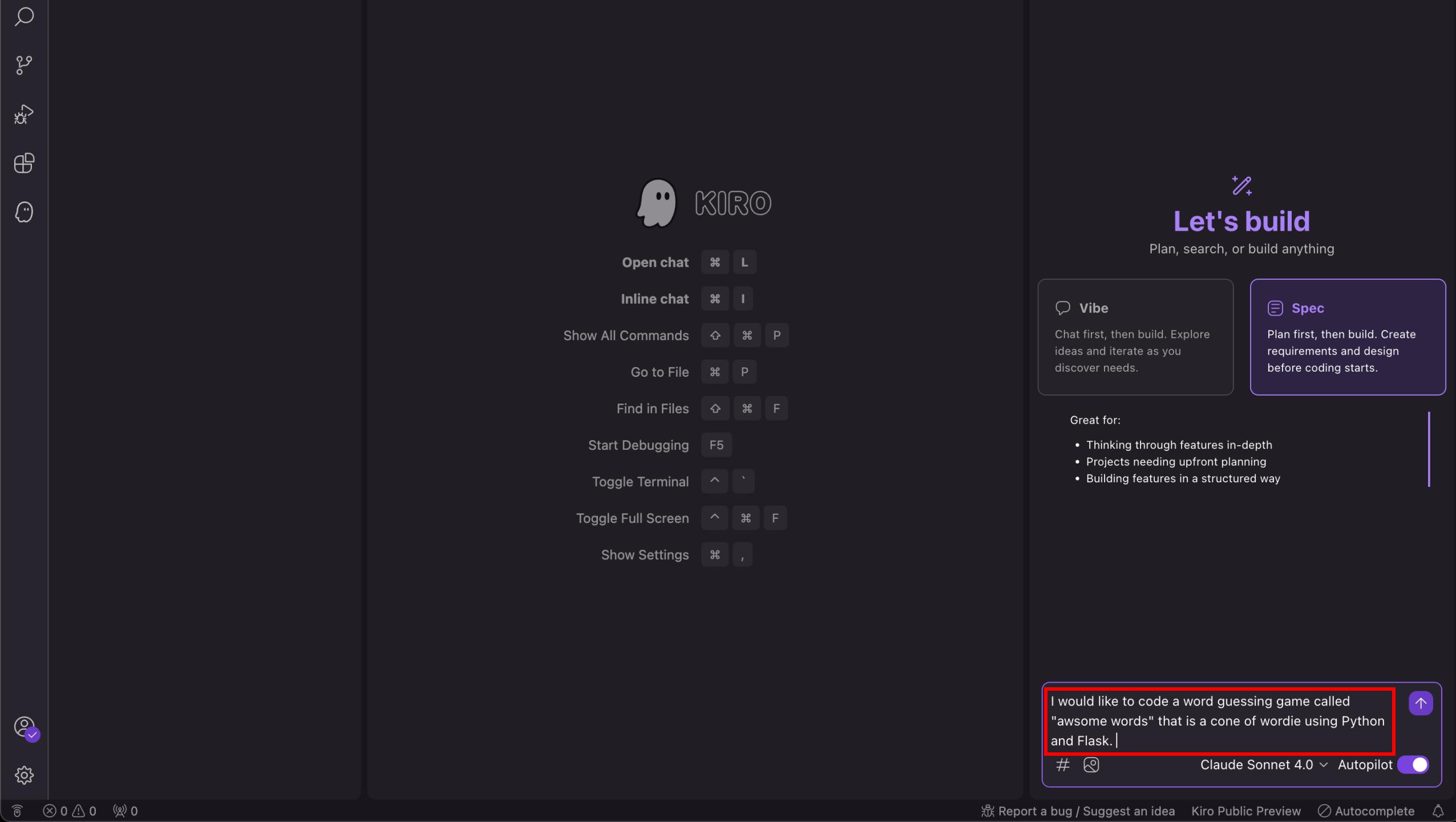
### Key Insights:

- ACCOUNTING department has the highest average salary at \$2,916.67
- SALES department has the lowest average salary at \$1,566.67
- RESEARCH department falls in the middle with \$2,175.00
- The SALES department has the most employees (6), while ACCOUNTING has the fewest (3)

# Kiro



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



EXPLORER: KIRO\_SUMMIT... + ⌂ C ⌂ ...

requirements.md

.kiro > specs > awesome-words > requirements.md > # Requirements Document > ## Requirements

Spec: awesome-words 1 Requirements 2 Design 3 Task list Refine

1 # Requirements Document

2

3 ## Introduction

4

5 Awesome Words is a web-based word guessing game inspired by Wordle, built using Python and Flask. Players attempt to guess a 5-letter word within 6 attempts, receiving feedback after each guess to help them narrow down the correct answer. The game provides an engaging, interactive experience with visual feedback and maintains game state throughout a session.

6

7 ## Requirements

8

9 ### Requirement 1

10

11 \*\*User Story:\*\* As a player, I want to start a new word guessing game, so that I can challenge myself with a fresh puzzle.

12

13 ##### Acceptance Criteria

14

15 1. WHEN a player visits the game page THEN the system SHALL display a clean game board with 6 empty rows for guesses

16 2. WHEN a new game starts THEN the system SHALL randomly select a valid 5-letter word from a predefined word list

17 3. WHEN a new game starts THEN the system SHALL

I would like to code a word guessing game called "awsome words" that is a clone of wordle using Python and Flask.

Kiro

Great idea! Let's create a spec for your "Awesome Words" game - a Wordle clone using Python and Flask. I'll start by creating the requirements document based on your concept.

Created requirements.md

Do the requirements look good? If so, we can move on to the design.

Move to design phase

Ask a question or describe a task...

# ⌂ Claude Sonnet 4.0 Autopilot

requirements.md

Spec: awesome-words Requirements Design Task list Refine

```
1 # Requirements Document
7 ## Requirements
50 ### Requirement 5
54 ##### Acceptance Criteria
57 2. WHEN a player uses all 6 attempts without
guessing correctly THEN the system SHALL display a
game over message and reveal the correct word
58 3. WHEN the game ends THEN the system SHALL
provide an option to start a new game
59
60 ### Requirement 6
61
62 **User Story:** As a player, I want an intuitive
web interface, so that I can easily interact with
the game.
63
64 ##### Acceptance Criteria
65
66 1. WHEN the game loads THEN the system SHALL
display a responsive web interface that works on
desktop and mobile devices
67 2. WHEN interacting with the game THEN the system
SHALL provide clear visual indicators for
interactive elements
68 3. WHEN using the interface THEN the system SHALL
support both keyboard input and on-screen controls
for letter entry
69 4. WHEN the page loads THEN the system SHALL
display the game title "Awesome Words" prominently
```

I would like to code a word guessing game called "awsome words" that is a clone of wordle using Python and Flask.

Kiro

Great idea! Let's create a spec for your "Awesome Words" game - a Wordle clone using Python and Flask. I'll start by creating the requirements document based on your concept.

Created requirements.md

Do the requirements look good? If so, we can move on to the design.

Move to design phase

I would like to support multiple languages. for start, let's use English, French, and German.

# Claude Sonnet 4.0 Autopilot

The image shows a developer's workflow integrated with AI assistance. On the left, the VS Code interface displays a requirements document for a game named "Awesome Words". The document includes sections like "# Requirements Document", "## Introduction", and "## Requirements". A specific section of the introduction is highlighted in green, detailing the game's mechanics: "Awesome Words is a web-based word guessing game inspired by Wordle, built using Python and Flask. Players attempt to guess a 5-letter word within 6 attempts, receiving feedback after each guess to help them narrow down the correct answer. The game provides an engaging, interactive experience with visual feedback and maintains game state throughout a session." This highlighted text is also repeated below it in a larger, semi-transparent box. A red arrow points from the bottom of this box towards the AI interface on the right.

EXPLORER: KIRO\_SUMMIT... ⌂ ⌃ ⌄ ⌅ ⌆ ⌇ ⌈ ⌉ ⌊ ⌋

requirements.md ↗ requirements.md ./kiro/specs/awesome-words ⌂ ⌃ ⌄ ⌅ ⌆ ⌇ ⌈ ⌉ ⌊ ⌋

.kiro > specs > awesome-words > requirements.md # Requirements Document ## Introduction

1 1 # Requirements Document  
2 2  
3 3 ## Introduction  
4 4  
5 - Awesome Words is a web-based word guessing game  
- inspired by Wordle, built using Python and Flask.  
- Players attempt to guess a 5-letter word within 6  
- attempts, receiving feedback after each guess to  
- help them narrow down the correct answer. The game  
- provides an engaging, interactive experience with  
- visual feedback and maintains game state  
- throughout a session.  
5+ Awesome Words is a web-based word guessing game  
+ inspired by Wordle, built using Python and Flask.  
+ Players attempt to guess a 5-letter word within 6  
+ attempts, receiving feedback after each guess to  
+ help them narrow down the correct answer. The game  
+ supports multiple languages (English, French, and  
+ German) and provides an engaging, interactive  
+ experience with visual feedback and maintains game  
+ state throughout a session.  
6 6  
7 7 ## Requirements  
8 8  
9 9 ### Requirement 1  
10 10  
11 11 \*\*User Story:\*\* As a player, I want to start a new  
word guessing game, so that I can challenge myself  
with a fresh puzzle.  
12 12  
13 13 ##### Acceptance Criteria  
14 14

New Session × + ⌂ ⌃ ⌄ ⌅ ⌆ ⌇ ⌈ ⌉ ⌊ ⌋

Kiro

Retry

Kiro

Great idea! Let's create a spec for your "Awesome Words" game - a Wordle clone using Python and Flask. I'll start by creating the requirements document based on your concept.

Created requirements.md ⌂ ⌃

Do the requirements look good? If so, we can move on to the design.

I would like to support multiple languages. for start, let's use English, French, and German.

I'll update the requirements to include multi-language support for English, French, and German.

3 edits to file requirements.md ⌂ ⌃

Accepted edits to requirements.md ⌂ ⌃

Accepted edits to requirements.md ⌂ ⌃

Accepted edits to requirements.md ⌂ ⌃

Do the updated requirements with multi-language support look good? If so, we can move on to the design.

Move to design phase

Ask a question or describe a task...

# ⌂ Claude Sonnet 4.0 ⌂ Autopilot ⌂

EXPLORER: KIRO\_SUMMIT... ⌂ ⌃ ⌄ ⌅ ⌆ ⌇

requirements.md design.md

.kiro > specs > awesome-words > design.md > # Design Document > ## Testing Strategy >

Spec: awesome-words 1 Requirements 2 Design 3 Task list Refine

1 # Design Document

2

3 ## Overview

4

5 Awesome Words is a Flask-based web application that implements a Wordle-style word guessing game with multi-language support. The application uses server-side session management to maintain game state, provides a responsive web interface, and supports English, French, and German languages with appropriate word lists for each.

6

7 ## Architecture

8

9 ### High-Level Architecture

10

11 ````mermaid

12 graph TB

13 A[Web Browser] --> B[Flask Application]

14 B --> C[Game Logic Module]

15 B --> D[Language Manager]

16 B --> E[Word Lists]

17 B --> F[Session Storage]

18

19 C --> G[Game State]

20 C --> H[Guess Validator]

21 C --> I[Feedback Generator]

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

64

65

66

67

68

69

70

71

72

73

74

75

76

77

78

79

80

81

82

83

84

85

86

87

88

89

90

91

92

93

94

95

96

97

98

99

100

101

102

103

104

105

106

107

108

109

110

111

112

113

114

115

116

117

118

119

120

121

122

123

124

125

126

127

128

129

130

131

132

133

134

135

136

137

138

139

140

141

142

143

144

145

146

147

148

149

150

151

152

153

154

155

156

157

158

159

160

161

162

163

164

165

166

167

168

169

170

171

172

173

174

175

176

177

178

179

180

181

182

183

184

185

186

187

188

189

190

191

192

193

194

195

196

197

198

199

200

201

202

203

204

205

206

207

208

209

210

211

212

213

214

215

216

217

218

219

220

221

222

223

224

225

226

227

228

229

230

231

232

233

234

235

236

237

238

239

240

241

242

243

244

245

246

247

248

249

250

251

252

253

254

255

256

257

258

259

260

261

262

263

264

265

266

267

268

269

270

271

272

273

274

275

276

277

278

279

280

281

282

283

284

285

286

287

288

289

290

291

292

293

294

295

296

297

298

299

300

301

302

303

304

305

306

307

308

309

310

311

312

313

314

315

316

317

318

319

320

321

322

323

324

325

326

327

328

329

330

331

332

333

334

335

336

337

338

339

340

341

342

343

344

345

346

347

348

349

350

351

352

353

354

355

356

357

358

359

360

361

362

363

364

365

366

367

368

369

370

371

372

373

374

375

376

377

378

379

380

381

382

383

384

385

386

387

388

389

390

391

392

393

394

395

396

397

398

399

400

401

402

403

404

405

406

407

408

409

410

411

412

413

414

415

416

417

418

419

420

421

422

423

424

425

426

427

428

429

430

431

432

433

434

435

436

437

438

439

440

441

442

443

444

445

446

447

448

449

450

451

452

453

454

455

456

457

458

459

460

461

462

463

464

465

466

467

468

469

470

471

472

473

474

475

476

477

478

479

480

481

482

483

484

485

486

487

488

489

490

491

492

493

494

495

496

497

498

499

500

501

502

503

504

505

506

507

508

509

510

511

512

513

514

515

516

517

518

519

520

521

522

523

524

525

526

527

528

529

530

531

532

533

534

535

536

537

538

539

540

541

542

543

544

545

546

547

548

549

550

551

552

553

554

555

556

557

558

559

560

561

562

563

564

565

566

567

568

569

570

571

572

573

574

575

576

577

578

579

580

581

582

583

584

585

586

587

588

589

590

591

592

593

594

595

596

597

598

599

600

601

602

603

604

605

606

607

608

609

610

611

612

613

614

615

616

617

618

619

620

621

622

623

624

625

626

627

628

629

630

631

632

633

634

635

636

637

638

639

640

641

642

643

644

645

646

647

648

649

650

651

652

653

654

655

656

657

658

659

660

661

662

663

664

665

666

667

668

669

670

671

672

673

674

675

676

677

678

679

680

681

682

683

684

685

686

687

688

689

690

691

692

693

694

695

696

697

698

699

700

701

702

703

704

705

706

707

708

709

710

711

712

713

714

715

716

717

718

719

720

721

722

723

724

725

726

727

728

729

730

731

732

733

734

735

736

737

738

739

740

741

742

743

744

745

746

747

748

749

750

751

752

753

754

755

756

757

758

759

760

761

762

763

764

765

766

767

768

769

770

771

772

773

774

775

776

777

778

779

780

781

782

783

784

785

786

787

788

789

790

791

792

793

794

795

796

797

798

799

800

801

802

803

804

805

806

807

808

809

810

811

812

813

814

815

816

817

818

819

820

821

822

823

824

825

826

827

828

829

830

831

832

833

834

835

836

837

838

839

840

841

842

843

844

845

846

847

848

849

850

851

852

853

854

855

856

857

858

859

860

861

862

863

864

865

866

867

868

869

870

871

872

873

874

875

876

877

878

879

880

881

882

883

884

885

886

887

888

889

890

891

892

893

894

895

896

897

898

899

900

901

902

903

904

905

906

907

908

909

910

911

912

913

914

915

916

917

918

919

920

921

922

923

924

925

926

927

928

929

930

931

932

933

934

935

936

937

938

939

940

941

942

943

944

945

946

947

948

949

950

951

952

953

954

955

956

957

958

959

960

961

962

963

964

965

966

967

968

969

970

971

972

973

974

975

976

977

978

979

980

981

982

983

984

985

986

987

988

989

990

991

992

993

994

995

996

997

998

999

1000

1001

1002

1003

1004

1005

1006

1007

1008

1009

1010

1011

1012

1013

1014

1015

1016

1017

1018

1019

1020

1021

1022

1023

1024

1025

1026

1027

1028

1029

1030

1031

1032

1033

1034

1035

1036

1037

1038

1039

1040

1041

1042

1043

1044

1045

1046

1047

1048

1049

1050

1051

1052

1053

1054

1055

1056

1057

1058

1059

1060

1061

1062

1063

1064

1065

1066

1067

1068

1069

1070

1071

1072

1073

1074

1075

1076

1077

1078

1079

1080

1081

1082

1083

1084

1085

1086

1087

1088

1089

1090

1091

1092

1093

1094

1095

1096

1097

1098

1099

1100

1101

1102

1103

1104

1105

1106

1107

1108

1109

1110

1111

1112

1113

1114

1115

1116

1117

1118

1119

1120

1121

1122

1123

1124

1125

1126

1127

1128

1129

1130

1131

1132

1133

1134

1135

1136

1137

1138

1139

1140

1141

1142

1143

1144

1145

1146

1147

1148

1149

1150

1151

1152

1153

1154

1155

1156

1157

1158

1159

1160

1161

1162

1163

1164

1165

1166

1167

1168

1169

1170

1171

1172

1173

1174

1175

1176

1177

1178

1179

1180

1181

1182

1183

1184

1185

1186

1187

1188

1189

1190

1191

1192

1193

1194

1195

1196

1197

1198

1199

1200

1201

1202

1203

1204

1205

1206

1207

1208

1209

1210

1211

1212

1213

1214

1215

1216

1217

1218

1219

1220

1221

1222

1223

1224

1225

1226

1227

1228

1229

1230

1231

1232

1233

1234

1235

1236

1237

1238

1239

1240

1241

1242

1243

1244

1245

1246

1247

1248

1249

1250

1251

1252

1253

1254

1255

1256

1257

1258

1259

1260

1261

1262

1263

1264

1265

1266

1267

1268

1269

1270

1271

1272

1273

1274

1275

1276

1277

1278

1279

1280

1281

1282

1283

1284

1285

1286

1287

1288

1289

1290

1291

1292

1293

1294

1295

1296

1297

1298

1299

1300

1301

1302

1303

1304

1305

1306

1307

1308

1309

1310

1311

1312

1313

1314

1315

1316

1317

1318

1319

1320

1321

1322

1323

1324

1325

1326</

design.md  
requirements.md

Spec: awesome-words 1 Requirements 2 Design 3 Task list Refine

```
1 # Design Document
7 ## Architecture
27
28 ### Technology Stack
29
30 - **Backend**: Python 3.8+ with Flask framework
31 - **Frontend**: HTML5, CSS3, JavaScript (vanilla)
32 - **Session Management**: Flask-Session with server-side storage
33 - **Styling**: CSS Grid/Flexbox for responsive layout
34 - **Data Storage**: JSON files for word lists, Flask sessions for game state
35
36 ## Components and Interfaces
37
38 ### 1. Flask Application (`app.py`)
39
40 Main application entry point that handles routing and request processing.
41
42 **Routes:**
43 - `GET /` - Main game page
44 - `POST /guess` - Process player guess
45 - `POST /new-game` - Start new game
46 - `POST /set-language` - Change language preference
47
48 ### 2. Game Logic Module (`game_logic.py`)
49
50 Core game mechanics and state management.
```

3 edits to file requirements.md

Accepted edits to requirements.md

Accepted edits to requirements.md

Accepted edits to requirements.md

Do the updated requirements with multi-language support look good? If so, we can move on to the design.

Yes, the requirements looks good. let's proceed to design phase.

Perfect! Now I'll create the design document. Let me first do some research on Flask web applications and multi-language word game implementations to inform the design.

Searched workspace

Researching Flask session management patterns for maintaining game state

Now I'll create the comprehensive design document based on the requirements and best practices for Flask applications:

Created design.md

Does the design look good? If so, we can move on to the implementation plan.

Move to implementation plan

Ask a question or describe a task...

# Claude Sonnet 4.0 Autopilot

Ln 245, Col 50 Spaces: 4 UTF-8 LF Markdown Report a bug / Suggest an idea Kiro Public Preview Autocomplete

.kiro > specs > awesome-words > tasks.md > # Implementation Plan

Spec: awesome-words    1 Requirements    2 Design    3 Task list    Update tasks

1 # Implementation Plan

92    ↳ Start task

93    - [ ] 6. Integrate components and test end-to-end functionality

94       ↳ Start task

95       - [ ] 6.1 Wire together backend and frontend

96          - Connect Flask routes with game logic classes

97          - Integrate session management with game state

98          - Connect language manager with UI localization

99          - Test complete game flow from start to finish

100          - Requirements: All requirements integration

101       ↳ Start task

102       - [ ] 6.2 Implement complete game scenarios

103          - Write integration tests for winning game scenarios

104          - Write integration tests for losing game scenarios

105          - Test language switching during active games

106          - Test session persistence across page refreshes

107          - Requirements: 5.1, 5.2, 6.4

108       ↳ Start task

109       - [ ] 6.3 Add final polish and optimization

110          - Implement loading states and user feedback

111          - Add input sanitization and security measures

112          - Optimize word list loading and caching

113          - Test cross-browser compatibility

114          - Requirements: 7.1, 7.2

Do the updated requirements with multi-language support look good? If so, we can move on to the design.

Yes, the requirements looks good. let's proceed to design phase.

Perfect! Now I'll create the design document. Let me first do some research on Flask web applications and multi-language word game implementations to inform the design.

Searched workspace

Researching Flask session management patterns for maintaining game state

Now I'll create the comprehensive design document based on the requirements and best practices for Flask applications:

Created [design.md](#)

Does the design look good? If so, we can move on to the implementation plan.

ok. let's move to implementation.

Perfect! Now I'll create the implementation plan with actionable coding tasks based on the requirements and design.

Created [tasks.md](#)

Do the tasks look good?

Finalize task list

Ask a question or describe a task...

#    Claude Sonnet 4.0 Autopilot

Kiro\_Summit\_Demo\_A

EXPLORER... ⌂ ⌂ ⌂ ⌂ ⌂ ⌂

requirements.md design.md tasks.md

.kiro > specs > awesome-words > tasks.md > @@ # Implementation Plan

Spec: awesome-words 1 Requirements 2 Design 3 Task list Update tasks

1 # Implementation Plan

2 **Task in progress** | View changes

3 - [ ] 1. Set up project structure and core dependencies

- Create Flask application directory structure with templates, static, and data folders
- Create requirements.txt with Flask and necessary dependencies
- Initialize basic Flask app with configuration
- Requirements: 7.1, 7.4

4 ↴ Start task

5 - [ ] 2. Implement core game logic classes

6 ↴ Start task

7 - [ ] 2.1 Create GameState class with state management

- Write GameState class to track word, guesses, attempts, and completion status
- Implement methods for adding guesses and checking game completion
- Create unit tests for GameState functionality
- Requirements: 1.3, 4.1, 4.2, 5.1, 5.2

8 ↴ Start task

9 - [ ] 2.2 Implement GuessValidator class

- Write GuessValidator to validate 5-letter alphabetic input
- Implement word existence checking against word lists
- Create unit tests for input validation scenarios

10 11 12 13 14 15 16 17 18 19

Would like to code... × Execute task: 1. Set up project structure and core dependencies 1. Set up project structure and core dependencies

Kiro

Working

Ask a question or describe a task...

# Best Practices for Customization

# Customize code recommendations

FoodDeliveryFulfillment.java

```
1 // Process a list of unassigned food deliveries around the driver's current location
2
3 package anycompany.fooddelivery.fulfillment;
4 import anycompany.delivery.Delivery;
5 import anycompany.delivery.DeliveryService;
6 import anycompany.driver.Driver;
7 import anycompany.driver.DriverLocationService;
8
9 public class FoodDeliveryFulfillment {
10     private DeliveryService deliveryService;
11     private DriverLocationService driverLocationService;
12
13     //Process all the unassigned deliveries
14     public void processUnassignedDeliveries() {
15         List<Delivery> unassignedDeliveries = deliveryService.getUnassignedDeliveries();
16         //Iterate over all the unassigned deliveries and assign them to a driver
17         for (Delivery delivery : unassignedDeliveries) {
18             //Get the nearest drivers for delivery location
19             List<Driver> drivers = driverLocationService.getDriver(delivery.getLocation());
20             for (Driver driver : drivers) {
21                 //Assign delivery to driver and send notification
22                 boolean isAssigned = deliveryService.assignDeliveryToDriver(delivery, driver);
23                 if (isAssigned) {
24                     deliveryService.notifyDelivery(delivery);
25                     driverLocationService.notifyDriver(driver);
26                     break;
27                 }
28             }
29         }
30     }
31     ...
32 }
```

Natural language prompt

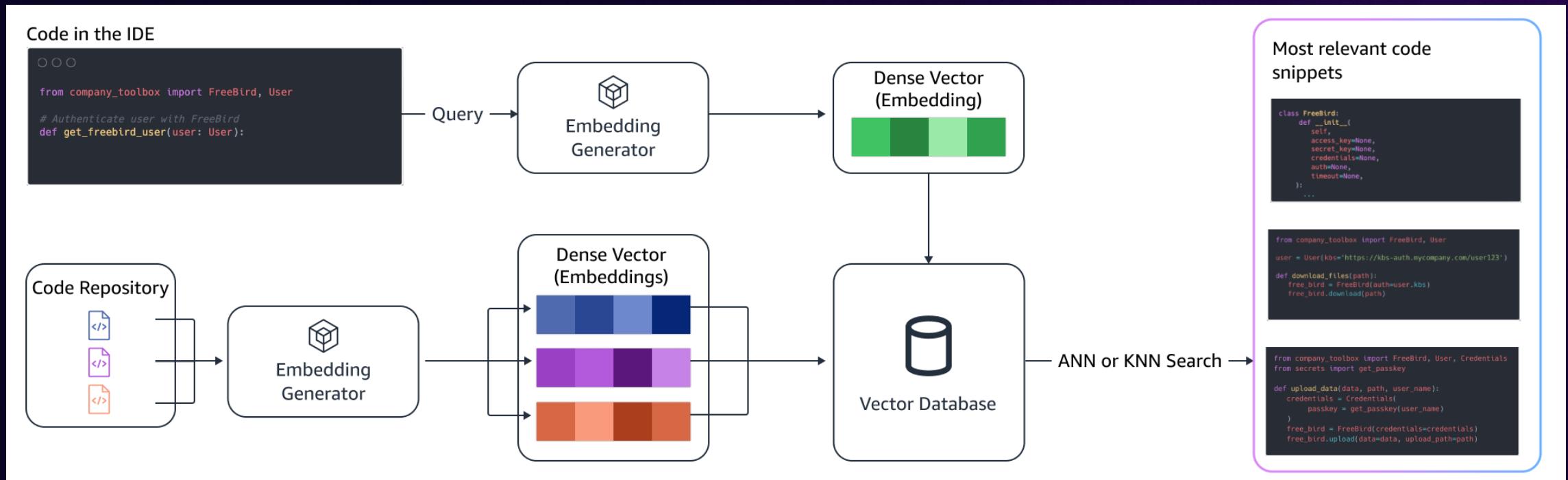
Generated in-line code based on your company's internal code bases

Generate code based on your private repositories

Use Amazon Q Developer chat in the IDE to ask questions about your internal code base

Help understand your internal code base

# Amazon Q customization is based on RAG



<https://aws.amazon.com/blogs/aws/customize-amazon-q-developer-in-your-ide-with-your-private-code-base/>

# Best practice: Comment your code



```
def add_new_movie_review(movie_id: int, customer_review: str):  
    # Implementation
```



```
def add_new_movie_review(movie_id: int, customer_review: str) -> int:
```

"""

Adds a new movie review to an existing movie.

Args:

movie\_id (int): The id of an existing movie.  
 customer\_review (str): movie review consisting  
 of up to 4096 characters.

Returns:

int: The id created for the new movie review.

"""

# Best practice: Include usage examples

```
from movies import MovieRepository

# Create a new Movies repository using the defaults
movies = MovieRepository()

# Add a review to an existing movie
Movie.add_new_movie_review(123, "Acting is
brilliant! Leonardo DiCaprio and Kate Winslet are
two of the best actors of their generation.")
```

# Best practice: Increase volume of examples

- While the minimum is 2MB, you will get better results with much larger codebases
- Avoid duplication of same code (copy/paste) in training set
- Avoid non-unique code (for example, node\_modules, etc.)

# Customized code suggestions with Amazon Q



National Australia Bank

“

At NAB, our developers use Amazon Q Developer's generative AI-powered assistant to generate real-time coding suggestions. NAB has now added specific suggestions using the Amazon Q customisation capability that are tailored to the NAB coding standards. **We have 500 developers using Amazon Q Developer with customisation enabled. We're seeing increased acceptance rates of 60% with customisation, and we expect this to grow with Amazon Q Developer chat, which makes it even easier to interact using natural language.** ”

**Andrew Brydon**

Executive for Enterprise Engineering, National Australia Bank



# Modernization of .NET and Java

# Amazon Q Developer transformation capabilities

## WORKLOADS



**.NET Framework  
applications  
on Windows**

Repository connection  
Repository selection  
Transformation  
Review



**Linux-ready  
cross-platform  
.NET applications**



**Java applications  
in v8, v11, v17, v21**

Build/test v8/v11/v17/v21  
Apply knowledge base  
Build/test v17/v21  
Error fixing using genAI  
Continuous maintenance



**Java applications  
in v17/v21**

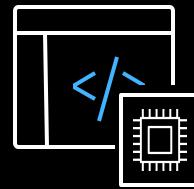
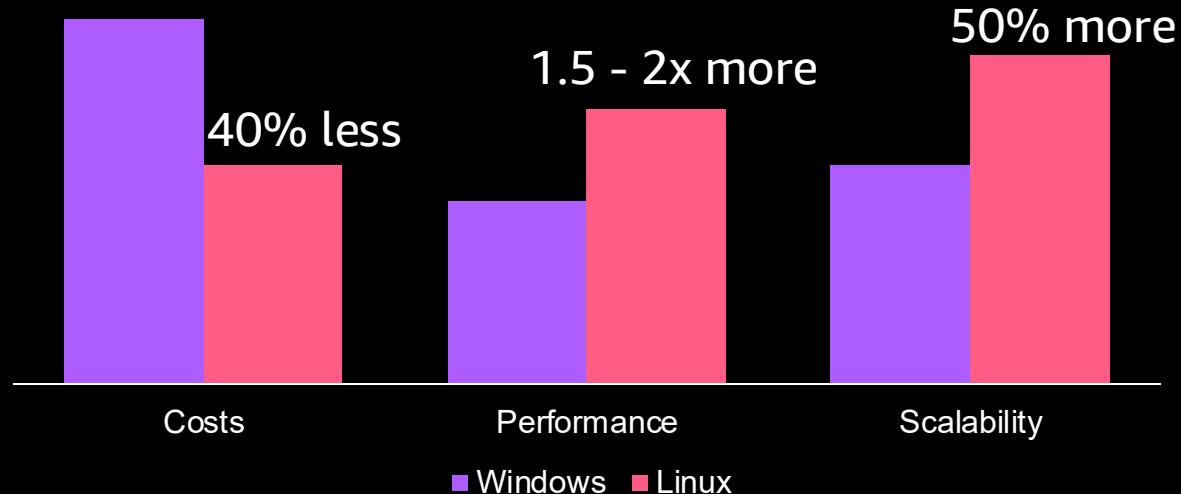
# Amazon Q Developer .NET porting



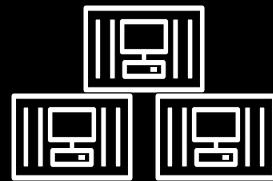
© 2025, Amazon Web Services, Inc. or its affiliates. All rights reserved.

# Why modernize .NET apps from Windows to Linux?

*Customers have reduced their Windows usage by over 70% to adopt Linux*



Compatible with x86-64  
and ARM64



Lightweight containers



Serverless architecture  
with Lambda



# Modernizing can cut compute costs substantially

## Examples of EC2 hourly pricing for the same instance type/size

<u>Instance</u>	<u>Windows</u>	<u>Linux</u>	<u>Savings vs. Windows</u>
m7i.2xlarge	\$0.7712	\$0.4032	<b>47.7% savings</b>
c7i.4xlarge	\$1.450	\$0.714	<b>50.8% savings</b>
r7i.8xlarge	\$3.5888	\$2.1168	<b>41.0% savings</b>

## Additional savings by moving to Graviton instances

<u>Instance</u>		<u>Linux</u>	<u>vs. Intel+Windows</u>	<u>vs. Intel+Linux</u>
m7g.2xlarge	—	\$0.3264	<b>57.7% savings</b>	19% savings
c7g.4xlarge	—	\$0.5781	<b>60.1% savings</b>	19% savings
r8g.8xlarge	—	\$1.88512	<b>47.5% savings</b>	11% savings

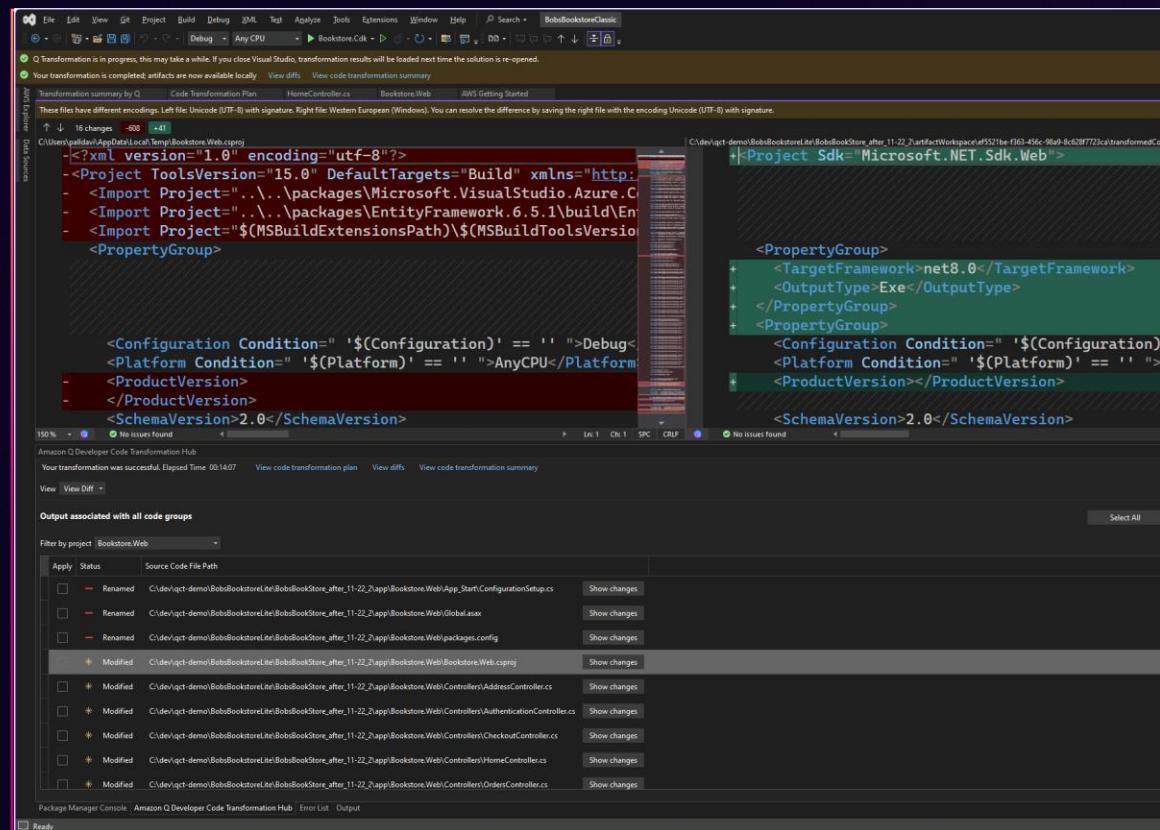
\*Ohio region, July 2024



# Two experiences for .NET porting

THE SAME GENERATIVE AI AGENT CAPABILITIES

## .NET porting in Visual Studio IDE (for developers)

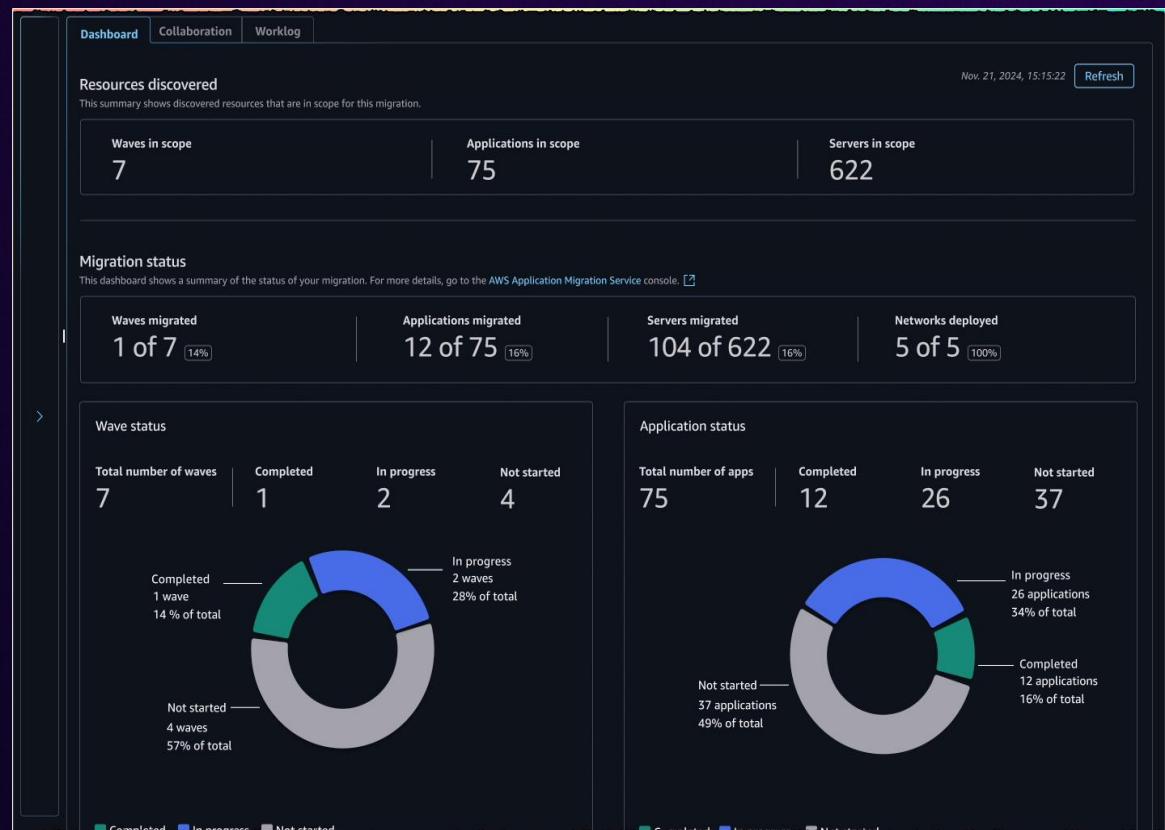


Amazon Q Developer



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

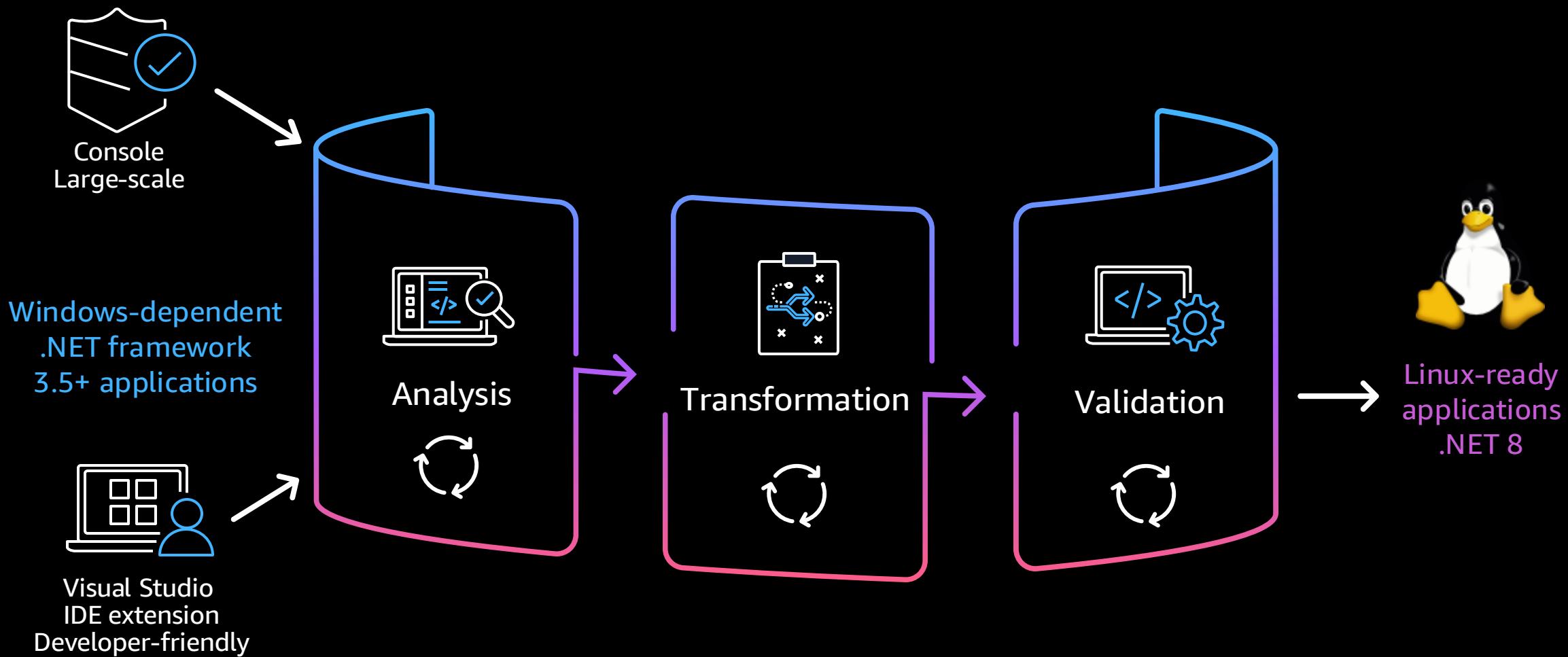
## .NET porting at large-scale using a web experience (for IT and modernization teams)



AWS Transform

# .NET Transformation agent

Agent for .NET application porting to Linux



# Analysis

Understand code incompatibilities

- **Expanded connector support: GitHub, Bitbucket, GitLab**
- **Validates supported project types:** WCF, Web API, Test projects, MVC/SPA BLLs, Class Library, and Console Apps.
- **Console-only: Automated cross-repo code dependency analysis and downloadable assessment metrics (LoCs, projects, dependencies) to aid ProServe and partner estimates.**
- **Integration with 3P tools like CAST**



Amazon Q Developer: Transform

my-first-workspace

.NET Application modernization

Review and start transformation

Rejected In progress

The user [awscustomer] sent you a list of repositories, dependent repositories, and dependent packages that were selected for transformation. Amazon Q can transform a maximum of 100 dependencies and repositories per transformation plan. Review the list, and when you are ready, approve the list to start the transformation. If you reject the plan, please inform the user [awscustomer] to restart the job.

Job summary

Target branch destination: qt-mybranch-10-20-2024

Target version: .NET 8.0

Exclude .NET Standard projects: Yes

Number of repos: 25

Number of dependent repos: 10

Number of Private NuGet packages: 75

Total lines of code for the job: 2,209,304

Repositories selected (25)

Name	Source branch	Supported projects	Lines of code	Project detected	Projects skipped	Dependencies detected
companyApp-repo	Main	Yes	1,000	5	3 of 3	3 of 3
WebApp-framework	Main	Yes	1,000	75	3 of 3	3 of 3
CloudInfrastructure	Develop	Yes	1,000	84	3 of 3	3 of 3

Dependent repositories added (25)

Name	Needed by	Source branch	Supported projects	Lines of code	Projects detected	Projects skipped
companyApp-repo	companyApp-repo	Main	Yes	1,000	5	3 of 3
WebApp-framework	companyApp-repo	Main	Yes	1,000	34	3 of 3

Dependent packages (25)

Name	Associated repositories	Framework version status	Core version status
AzureCore	20	Resolved	Resolved
AWS SDK Core	10	Resolved	Resolved
Autofac	5	Resolved	Resolved
Corecore.core	5	Resolved	Resolved
Repo 4	5	Resolved	Resolved

```
{ "version": "1.0", "repositories": [ { "owner": "owner1", "name": "Repo 1", "sourceBranch": "mainline", "projectCount": 3, "projectTypes": [ "Type1", "Type2" ], "privateNuGetPackages": { "count": 5 }, "publicNuGetPackages": { "count": 6 } } ] }
```



# Transformation

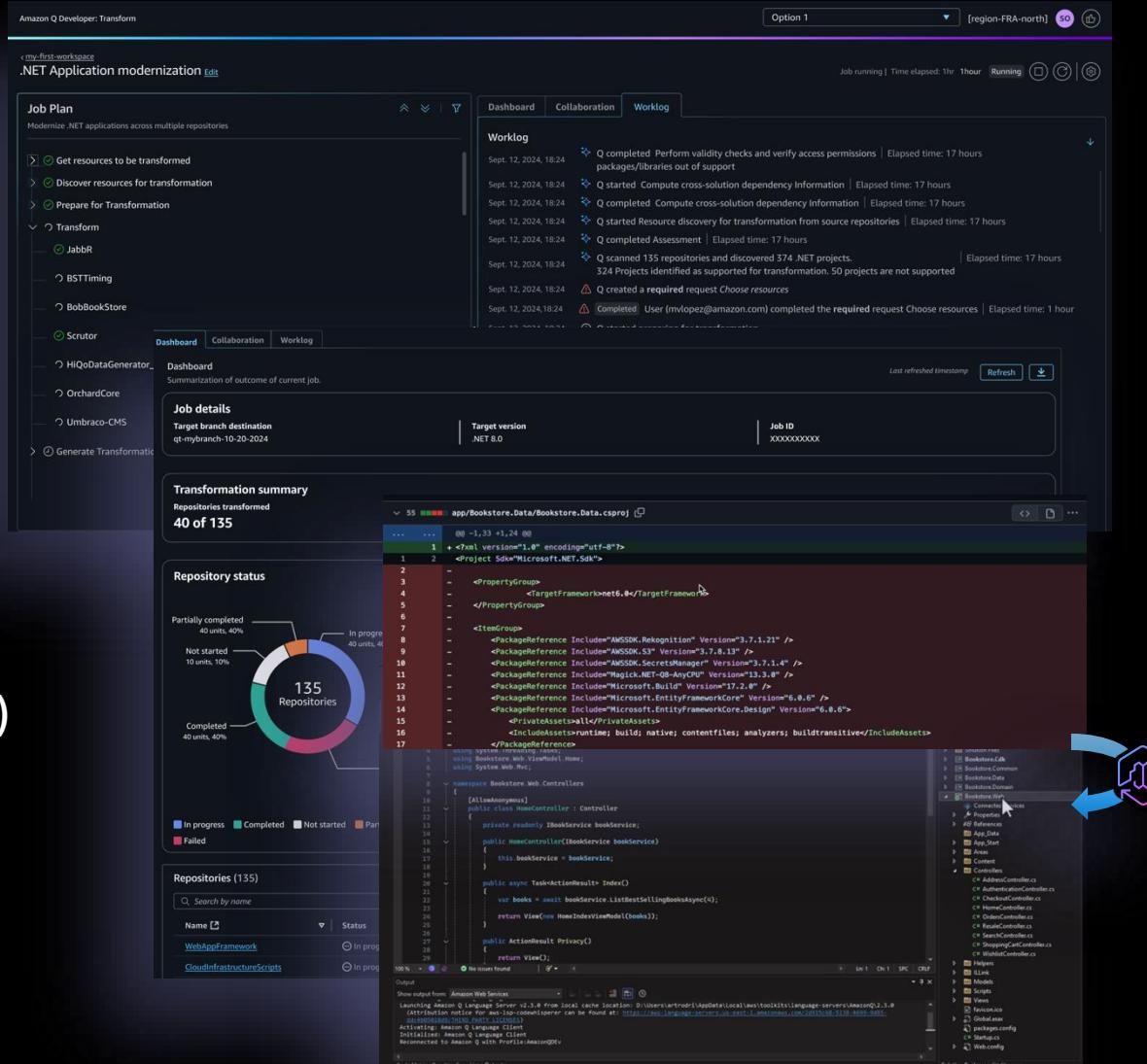
Modernize legacy .NET code to cross-platform

- **Automated code transformation**

- 100s of applications
- Support for 1P and top 35 3P like Newtonsoft, Razor views
- Support for Private NuGet

- **Progress visibility and tracking with** transformation logs and chat support progress check

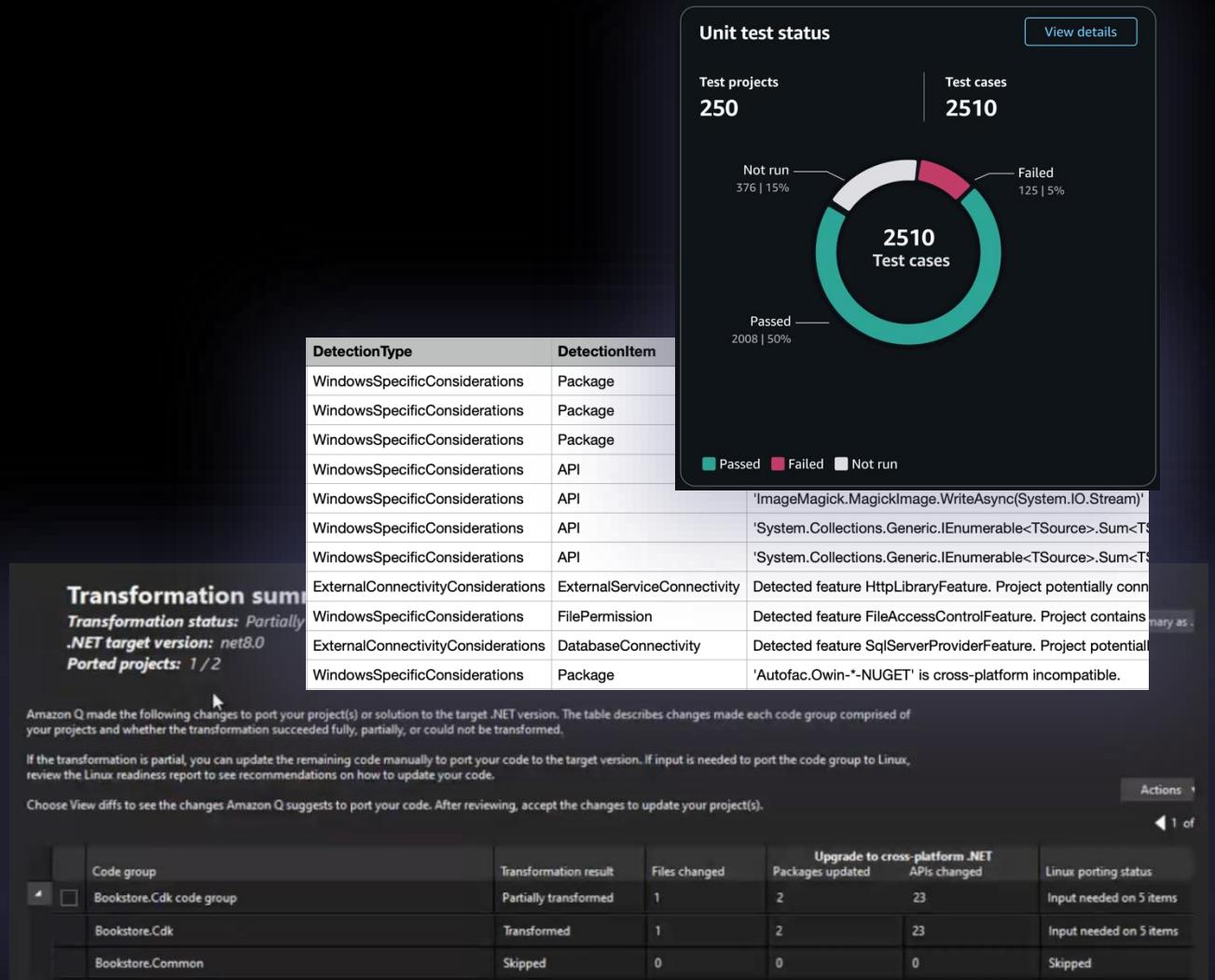
- **Developer-led completions** of remaining work (~20%) with Visual Studio IDE extension



# Validation

Validate transformed code with human oversight

- **Automated full build & unit test execution** to receive results of running transformed test projects
- **Transformation summary** for explainability of LLM-based code changes for validation
- IDE –only: **Linux readiness report** to identify compatibility issues requiring human attention



# Thomson Reuters turbocharged .NET modernization with AWS Transform

## CHALLENGES

Thomson Reuters was struggling with **expensive, time-consuming maintenance** of legacy .NET Framework applications that were impeding innovation and competing with their roadmap priorities.

## SOLUTIONS

The company implemented AWS Transform, an agentic AI-powered tool designed for **large-scale .NET modernization** that enables parallel processing through both **web interface** and **IDE integration**.

## RESULTS

- › Modernized 4x faster: **1.5M lines of code/month**
- › **30% cost savings** moving to Linux
- › 70% reduction in technical debt
- › Transformation time: **2 weeks** vs. months

**"AWS Transform felt like an extension of our team—constantly learning, optimizing, and helping us move faster."**

**Matt Dimich**

VP, Platform Engineering Enablement at Thomson Reuters



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

## CUSTOMER PROFILE



THOMSON REUTERS

INDUSTRY  
Financial Services

REGION  
United States/Global

As a global tech and AI leader powering the legal, tax, and compliance industries, Thomson Reuters is always looking for smarter, faster ways to build. That means staying ahead of the curve—and teaming up with partners like AWS to push what's possible.

# Amazon Q Developer **Java upgrades**



# Q Developer Java upgrades



V8, v11, v17, v21



V17, v21



**Build & test**  
Java 8,11,17,21



**Apply Knowledge Base**



**Build & Test**  
Java 17,21



**Fix Errors with GenAI**



**Continuous Maintenance**

# Amazon's internal use of Amazon Q



CEO Andy Jassy reported in a recent [LinkedIn post](#) that the internal developer tooling team used Q Developer's new code transformation capability—specifically integrating it with internal development systems—to automate the process of upgrading over 30,000 applications from Java 8 or 11 to Java 17, saving over 4,500 developer years compared to manual upgrades.

Source: <https://news.a2z.com/contents/38885081>

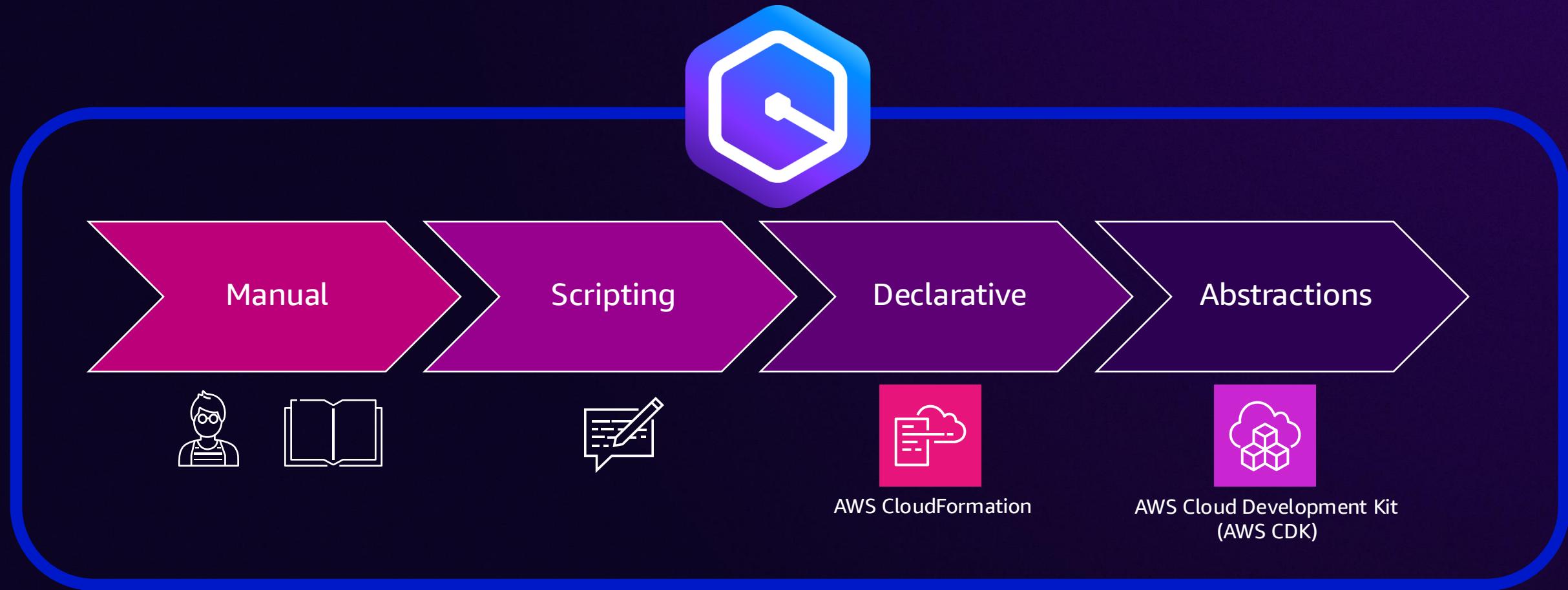
LinkedIn post: [https://www.linkedin.com/posts/andy-jassy-8b1615\\_one-of-the-most-tedious-but-critical-tasks-activity-7232374162185461760-AdSz/?utm\\_source=share&utm\\_medium=member\\_desktop](https://www.linkedin.com/posts/andy-jassy-8b1615_one-of-the-most-tedious-but-critical-tasks-activity-7232374162185461760-AdSz/?utm_source=share&utm_medium=member_desktop)

# Cloud Hosting with Infrastructure as Code



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

# Evolution of building in AWS



# Generate Infrastructure as Code (IaC)

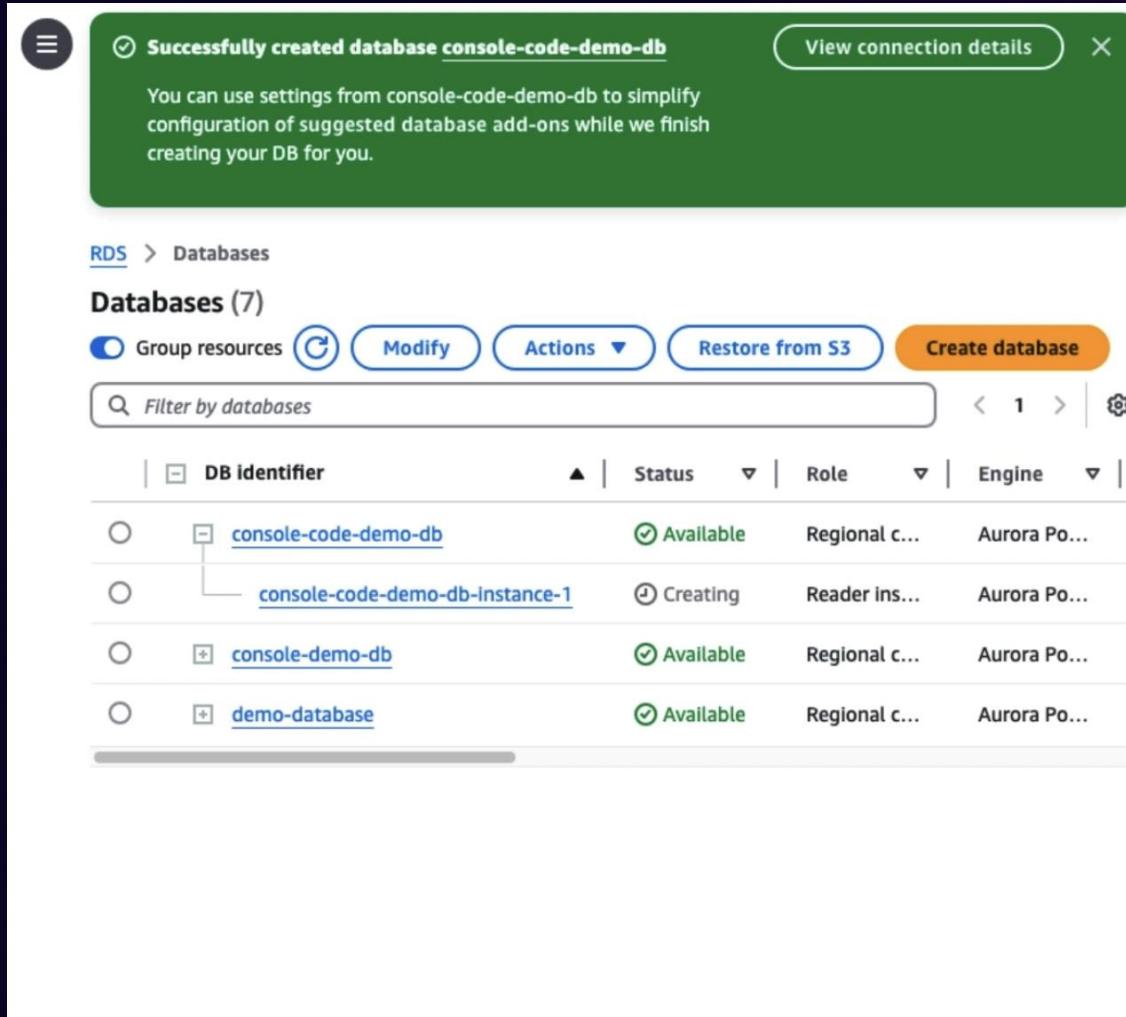
The screenshot shows the AWS RDS Databases page. At the top, a blue banner indicates that a database named "console-code-demo-db" is "Creating". Below the banner, there are buttons for "View credential details", "Create database", "Actions", "Restore from S3", and "Modify". A "Group resources" button is also present. A search bar labeled "Filter by databases" is available. The main table lists seven databases, including "console-code-demo-db" (status: Creating), "console-code-demo-db-instance-1" (status: Creating), "console-demo-db" (status: Available), and "demo-database" (status: Available). The table columns include DB identifier, Status, Role, and Engine.

The screenshot shows the "Console-to-Code" feature in progress. It displays a list of recorded actions (5/86) and a dropdown menu for generating code in different formats: CDK (Generate CDK Java, Generate CDK Python, Generate CDK TypeScript), CloudFormation (Generate CFN JSON, Generate CFN YAML), and RDS / Launch Dbinst... (createDBinstance, createDBCluster, authorizeSecurityGroup, createSecurityGroup, createDBSubnet). The "Generate CDK Python" option is highlighted with a cursor icon. The interface includes "Reset" and "Stop" buttons at the top right.

Accelerates creation of Infrastructure as Code (IaC) - CDK, Terraform, CloudFormation

Console to code (Preview) sets foundation for IaC best practice patterns

# Generate Infrastructure as Code (IaC)



The screenshot shows the AWS RDS Databases page. A green success message box at the top left says "Successfully created database console-code-demo-db". Below it, a note says "You can use settings from console-code-demo-db to simplify configuration of suggested database add-ons while we finish creating your DB for you." The main table lists seven databases:

DB identifier	Status	Role	Engine
console-code-demo-db	Available	Regional c...	Aurora Po...
console-code-demo-db-instance-1	Creating	Reader ins...	Aurora Po...
console-demo-db	Available	Regional c...	Aurora Po...
demo-database	Available	Regional c...	Aurora Po...

To the right, a code editor window titled "CDK Python" displays generated infrastructure code:

```
```python
from aws_cdk import (
    Stack,
    aws_rds as rds,
    aws_ec2 as ec2,
    CfnOutput
)
from constructs import Construct

class RdsStack(Stack):
    def __init__(self, scope: Construct, construct_id: str, **kwargs) -> None:
        super().__init__(scope, construct_id, **kwargs)

        # Create DB Subnet Group
        subnet_ids = ["subnet-0caf3c76ef61bd6a",
                      "subnet-02fdf4d3e6b890c22", "subnet-05be166ae0aed2834", "subnet-01d2af67ffdd05d0c"]
        db_subnet_group = rds.SubnetGroup(
            self, "DefaultVpcSubnetGroup",
            description="Created from the RDS Management Console",
            subnet_group_name="default-vpc-0052c11fff911fa89",
            vpc_id="vpc-0052c11fff911fa89",
            vpc_subnets=ec2.SubnetSelection(subnet_ids=sub
```

```

Accelerates creation of Infrastructure as Code (IaC) - CDK, Terraform, CloudFormation

Console to code (Preview) sets foundation for IaC best practice patterns

# Accelerate IaC Deployment with High Accuracy

AWS DevOps & Developer Productivity Blog

## Accenture Expedites Infrastructure Deployment with Amazon Q Developer

by Vikas Purohit | on 30 SEP 2024 | in Amazon Q, Case Study | Permalink |  Share

*By Priya Mallya, Managing Director – Accenture, Sandeep Singh Bhatia, Sr Manager – Accenture*

*Vikas Purohit – Sr. Solutions Architect – AWS*

Being able to internally setup and manage flexible, efficient infrastructure can be painful. Manually authoring your Infrastructure as Code (IaC) templates is error prone and time consuming. However, adoption of generative AI coding tools is changing the way infrastructure engineers can carry out their day-to-day activities. Accenture utilizing Amazon Q Developer to create IaC templates for one of their US based customer became a game changer.

- Accelerated Terraform code writing time by 30%
- Achieved nearly 99% accuracy in generated Terraform code
- Provided conversational interface for quick AWS service queries
- Helped identify and fix potential errors and edge cases
- Supported deployment of over 50 AWS services via Terraform
- Enabled meeting tight customer timelines while maintaining IaC best practices

<https://aws.amazon.com/blogs/devops/accelerate-iac-deployment-with-high-accuracy/>



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

# Getting Started



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

# Deployment Options for Q Developer

## Free tier



Code faster with code suggestions in the IDE and CLI



Use where you work: Your IDE, CLI, the AWS Management Console, etc.



Limited monthly access of advanced features

## Pro tier

**(\$19/mo. per user)**



Manage users and policies with enterprise access controls



Customize Amazon Q to your code base



High limits of advanced features



Access to Kiro (no wait-list)

# Deployment Options for Kiro

KIRO FREE

COMING SOON

\$0

per month

- ✓ 50 credits

Join Waitlist

KIRO PRO

COMING SOON

\$20

per month

- ✓ 1,000 credits
- ✓ Pay-per-use overage

KIRO PRO+

COMING SOON

\$40

per month

- ✓ 2,000 credits
- ✓ Pay-per-use overage

KIRO POWER

COMING SOON

\$200

per month

- ✓ 10,000 credits
- ✓ Pay-per-use overage

<https://kiro.dev/pricing/>



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

# Developers can be more productive

Accelerate Workflows

Eliminate Repetitive Tasks

Focus on High-Value Activities



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

THE WEB THE API THE ML THE DB THE FS THE CD THE CI/CD THE INFRASTRUCTURE AS A CODE THE SERVERLESS THE SERVERLESS

# Thank you!

**Michael Lin**

[linmicht@amazon.com](mailto:linmicht@amazon.com)



Please complete the session  
survey in the mobile app