

AWS **Sioux Falls** User Group

Amazon Kiro for Test/Spec Driven Development



JUNE 30th | 4:30 PM



Please scan and RSVP for attendance



Sioux Falls Area AWS User Group

Meetup #2: 30 June 2026

Welcome everyone! Let's build a local AWS community.

What's on the Agenda?

5:00 PM



Introductions & Meetup Details

5:15 PM



Amazon Kiro for Test/Spec Driven Development (Intro)

6:00 PM



Open Discussion & Q&A

6:15 PM



Community Growth

Meet Your Leader

- Hi, I'm your meetup leader. I'm passionate about building the AWS community in Sioux Falls.
- Why Sioux Falls? We have a vibrant tech scene that deserves a dedicated space for cloud discussion.
- Our Goal: Learn from each other, share cloud experiences, and build a strong professional network.
Shape Sioux Falls Startups with {any} cloud

AWS Management & Governance Consultant

10+ years in AWS / 5 Certs / Quiverstone / Sentri Cloud

6-year AWS Community Builder / 1st Year UG Leader / 2025 R2R



Ross Wickman



What Is the Sioux Falls AWS User Group?



Developers, architects, ops engineers, founders — anyone building on or curious about AWS.



“Monthly” meetups, third week of each month. ~1.5 hours each.
Cloud / Coffee / Conversation



This is a space to share, learn, and connect with local AWS practitioners.



Hands-on topics, real-world scenarios, guest speakers, and community-driven discussions.

<https://www.siuoxfallsawsuser.group>

contact@siouxfallsawsuser.group

TONIGHT · ~45 MINUTES

What we'll cover



Why spec-driven?

Moving beyond "vibe coding"

5 min



Hooks

Automate quality on every save & task

6 min



Kiro IDE basics

Specs, the editor, your first feature

8 min



Custom agents

Add a specialist agent in minutes

6 min



Kiro CLI basics

Chat, agents, and headless runs

6 min



Agent Toolkit for AWS

Wire AWS into Kiro + live demo

10 min

From “vibe coding” to agentic engineering

Vibe coding

- Prompt → hope → fix → reprompt
- Agent guesses intent and architecture
- Hard to review, hard to hand off
- Regressions sneak back in
- Great for a demo — risky for production

Spec / test-driven

- Write the requirements & design first
- Agent works from an approved plan
- Acceptance criteria = the tests
- Tasks are tracked, reviewable, shareable
- Repeatable — built for production

What is Kiro?

An agentic IDE (and CLI) that works alongside you from prototype to production — built on spec-driven development, hooks, steering, MCP, and custom agents.



IDE / Web

Full editor (VS Code-based) with a Kiro agent panel: specs, hooks, steering.



CLI

Same agent in your terminal — interactive chat or headless in CI/CD.



Specs

requirements → design → tasks, tracked as living artifacts.



AWS-native

From Amazon; pairs with the Agent Toolkit for AWS.

Download both the IDE and CLI at kiro.dev

Spec-driven development: three artifacts



1 · requirements.md

User stories + acceptance criteria. Define WHAT to build (or bug analysis for fixes).



2 · design.md

Architecture, sequence diagrams, data flow, error handling & testing strategy.



3 · tasks.md

Discrete, trackable tasks. Run one at a time or all at once with live status.

Three-phase workflow: Requirements → Design → Tasks, each with an approval gate. “Run all Tasks” builds a dependency graph and runs independent tasks concurrently in waves.

Specs vs. Vibe — and where tests fit



Reach for a Spec when...

- Building a real feature that needs a plan
- Fixing a bug where regressions are costly
- You want tests written from acceptance criteria
- Work is shared across a team



Reach for Vibe when...

- Quick exploratory coding
- Prototyping without clear goals
- Throwaway scripts or spikes
- You'll promote it to a spec once it sticks

Test-driven tip: Put “generate unit/integration tests from the acceptance criteria, then implement until they pass” right in design.md — Kiro will scaffold tests as tasks.

Kiro IDE basics

Install & sign in

Download from kiro.dev, open your project folder (it's VS Code–based).

Find the Kiro panel

Left sidebar: Specs, Agent Hooks, Steering, MCP — plus a chat pane.

Start your first spec

Click + under Specs (or choose Spec in chat) → Feature or Bug → describe it.

Drive the workflow

Approve requirements → design → tasks. Run tasks individually or all at once.

```
# Vibe vs Spec, side by side
Vibe:  just chat in the pane
Spec:  + under "Specs"
```

```
# Files Kiro creates in repo
.kiro/specs/<feature>/
  requirements.md
  design.md
  tasks.md
```

Pro tips

- Command Palette (Cmd/Ctrl+Shift+P) → “Kiro:” for everything
- Commit the .kiro/ folder — specs are team docs
- “Analyze Requirements” catches gaps before design

Kiro CLI basics

```
# Install from kiro.dev, then:
cd my-project
kiro-cli                # interactive chat

# One-shot question
kiro-cli chat "how do I list files?"

# Use a specific agent
kiro-cli --agent backend-specialist

# Handy flags
kiro-cli --list-models
kiro-cli --resume      # resume last chat

# In-session slash commands
> /agent create    /agent swap
> /spec           /hooks
```



Interactive chat Natural conversation in your terminal.



Custom agents Specialized agents per workflow.



MCP integration Connect external tools & data.



Smart hooks Pre/post command automation.



Headless / CI API-key auth for pipelines.

Agent Hooks: automate quality

Event-driven triggers that run an agent prompt or a shell command automatically — keep quality consistent without remembering to.

1 · Event detection

Kiro watches for a specific event in your workspace.

2 · Automated action

It runs a shell command or an agent prompt in response.

Common triggers

Trigger	Fires when...	Can block?
PostFileSave	Agent saves a file (matcher = path regex)	No
PostTaskExec	A spec task finishes	No
PreToolUse	Before a tool runs (matcher = tool name)	Yes
UserPromptSubmit	You submit a prompt	Yes
SessionStart / Stop	Session begins / agent finishes its turn	No

EXAMPLE

A hook you can copy tonight

Stored as JSON in `.kiro/hooks/` (workspace) or `~/.kiro/hooks/` (user).

Lint on every TypeScript save

```
{
  "version": "v1",
  "hooks": [{
    "name": "lint-on-save",
    "trigger": "PostFileSave",
    "matcher": "\\\\.ts$",
    "action": {
      "type": "command",
      "command": "npm run lint"
    },
    "timeout": 30,
    "enabled": true
  }]
}
```

Run tests after every spec task (TDD)

```
{
  "name": "test-after-task",
  "trigger": "PostTaskExec",
  "action": {
    "type": "agent",
    "prompt": "Run the test suite.  
If anything fails, fix it  
before moving on."
  }
}
```

Exit codes: 0 = pass · 2 = block (Pre* hooks), stderr goes back to the agent · other = warning.

Custom agents: add one in minutes

A custom agent = a named persona with its own prompt, tool permissions, context (resources), MCP servers, and model. Great for scoping what the agent can touch.

Create it (AI-assisted)

```
> /agent create
name:      backend-specialist
description: backend coding expert
scope:     Local (workspace)
MCP servers: code-analysis, aws

# or inline / from terminal:
kiro-cli agent create backend-specialist
```

Saved to: .kiro/agents/ (workspace) or ~/.kiro/agents/ (global). Swap anytime with /agent swap.

What it looks like (JSON)

```
{
  "name": "backend-specialist",
  "description": "Backend expert",
  "tools": ["read", "write"],
  "allowedTools": ["read"],
  "resources": [
    "file://README.md",
    "file://.kiro/steering/**/* .md"
  ],
  "prompt": "You are a careful\n  backend engineer...",
  "model": "claude-sonnet-4"
}
```

Steering + MCP = the bridge to AWS



Steering (rules)

- Persistent markdown context the agent always reads
- Lives in `.kiro/steering/**/* .md`
- Encode standards: style, security, “use our AWS tags”
- Replaces the old “manual hooks”



MCP

- Model Context Protocol — connect external tools
- Configured in `.kiro/settings/mcp.json`
- One server can expose 300+ AWS services
- **This is how the Agent Toolkit plugs in →**

THE AGENT TOOLKIT FOR AWS

Give your coding agent the tools, knowledge, and guardrails to build on AWS

Official & AWS-supported. Works with the agents you already use — Kiro, Claude Code, Cursor, Codex.



Picks the right service

Stops agents choosing the wrong tool for the job



Avoids misconfiguration

Curated skills follow AWS best practices



Current docs, on demand

Searches live AWS docs — no stale training data

Three components



AWS MCP Server

- Managed, remote endpoint
- Full AWS API (300+ services)
- Sandboxed script execution
- Live documentation search
- IAM, CloudWatch, CloudTrail



Agent Skills

- Curated, step-by-step guidance
- Decision guides + references
- Loaded on demand by the agent
- End-to-end evaluated
- npx skills add ...



Plugins

- One-install bundles
- = MCP config + skills
- aws-core, aws-agents, ...
- Claude Code, Codex, Cursor
- Kiro: configure MCP + skills

Toolkit plugins

Plugin	What it covers	For
aws-core ★	Service selection, CDK/CloudFormation, serverless, containers, storage, observability, billing, SDKs, deploy	App developers — start here
aws-agents	Building AI agents on AWS with Amazon Bedrock & AgentCore	AI engineers
aws-data-analytics	Data lake, analytics & ETL with S3 Tables, AWS Glue, Athena	Data analysts
aws-agents-for-devsecops	Incident investigation, code review/UAT, vuln scans, pen tests	DevSecOps teams

★ **Start with aws-core.** The agent auto-discovers the right skills from your request — you don't pick skills by hand.

Wiring the toolkit into Kiro

Step 1 — Add the AWS MCP Server to .kiro/settings/mcp.json

```
{
  "mcpServers": {
    "aws": {
      "command": "uvx",
      "args": [
        "mcp-proxy-for-aws@1.6.2",
        "https://aws-mcp.us-east-1.api.aws/mcp",
        "--metadata", "AWS_REGION=us-west-2"
      ]
    }
  }
}
```

Step 2 — Add skills

```
npx skills add \
  aws/agent-toolkit\
  -for-aws/skills
```

Prerequisites

- uv installed (runs the proxy)
- AWS credentials for API calls
- (docs/skill search need no creds)
- Pin the version for safety

Step 3 — Verify: start a new chat and ask *"What AWS Regions are available?"* — a list back means you're connected.

Spec-driven, on AWS, end to end



1. Write a spec

“A serverless URL shortener: API Gateway + Lambda + DynamoDB”



2. Design w/ AWS skills

Toolkit guides service choice & best-practice config



3. Generate tasks

Kiro breaks it into trackable, testable steps



4. Hook runs tests

PostTaskExec hook keeps every step green



5. Deploy via MCP

Agent uses the AWS MCP Server to provision & verify



LIVE DEMO · RUNBOOK

0:00

Open Kiro, show the panel

Specs / Hooks / Steering / MCP. Show mcp.json with the aws server.

1:30

Smoke-test the toolkit

Ask: "What AWS Regions are available?" → proves MCP is live.

3:00

Start a Feature spec

"Serverless URL shortener on AWS." Walk requirements → design.

6:00

Add the test hook

Drop in PostTaskExec "run tests" hook; show it in .kiro/hooks.

7:30

Run a task or two

Let Kiro implement; hook runs tests; show live task status.

10:00

Swap a custom agent

/agent swap to an "aws-reviewer"; ask it to sanity-check IAM.

Wait...! There's more



Bugfix spec

Reproduce a bug, let Kiro write a failing test, then fix to green.



Natural-language hook

"Format Python on save" — let Kiro author the hook for you.



Docs without creds

Ask an AWS "how do I..." question to show live doc search.



Headless in CI

Run kiro-cli with API-key auth in a pipeline step.



Steering file

Add a steering rule ("always tag resources") and watch it apply.



Run all tasks

Show parallel "waves" executing independent tasks at once.

Resources & links

Kiro — IDE & CLI

- **Get the IDE & CLI**
kiro.dev
- **Docs (specs, hooks, agents)**
kiro.dev/docs
- **Hooks reference**
kiro.dev/docs/hooks
- **Custom agents**
kiro.dev/docs/cli/custom-agents

Agent Toolkit for AWS

- **GitHub repo**
github.com/aws/agent-toolkit-for-aws
- **Product page**
aws.amazon.com/products/developer-tools/agent-toolkit-for-aws
- **Quick start (user guide)**
docs.aws.amazon.com/agent-toolkit/latest/userguide/quick-start.html

Let's Talk About You



Your Projects

What are you currently building or migrating to the cloud?

Challenges

What are the biggest AWS hurdles you're facing right now?

Future Topics

What would you like to learn about in our next meetups?

What's Next

Help us grow the Sioux Falls AWS community by:



Speaker Outreach



Venue Coordination



Social Media



Sustenance Management

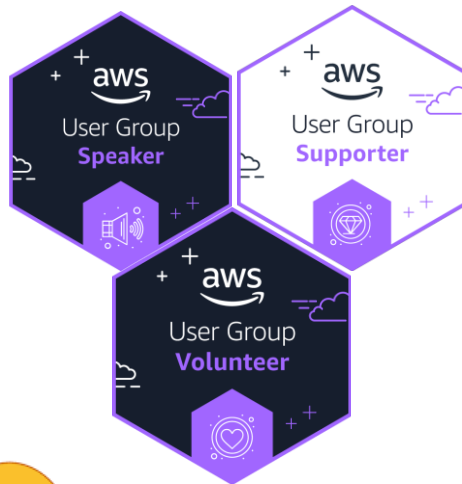
Interested in becoming a Co-Leader?

Come talk to me after the session or message me on Meetup.com.

Let's build something great together!

<https://www.sioxfallsawsuser.group>

contact@siouxfallsawsuser.group





Please scan and RSVP for attendance