Learn Network Automation with Github Copilot

Gowtham Tamilselvan

Praveen Poojary

BRKOPS-1726

About Us

Praveen Poojary

Principal Architect

13 Years in Cisco

#3xCCIE

#CCDE

Gowtham Tamilselvan

Software Architect

13 Years in Cisco

Automation Engineer

Distinguished Speaker

Agenda

- AI Primer
- Introduction to Github Copilot
- Network Automation Use-Cases
- Conclusion

slido

Please download and install the Slido app on all computers you use

Join at slido.com
#20251726

ⓘ Start presenting to display the joining instructions on this slide.

slido

Please download and install the Slido app on all computers you use

How do you rate your proficiency level with using AI?

ⓘ Start presenting to display the poll results on this slide.

slido

Please download and install the Slido app on all computers you use

Which LLM Models are you using today?

ⓘ Start presenting to display the poll results on this slide.

AI Primer

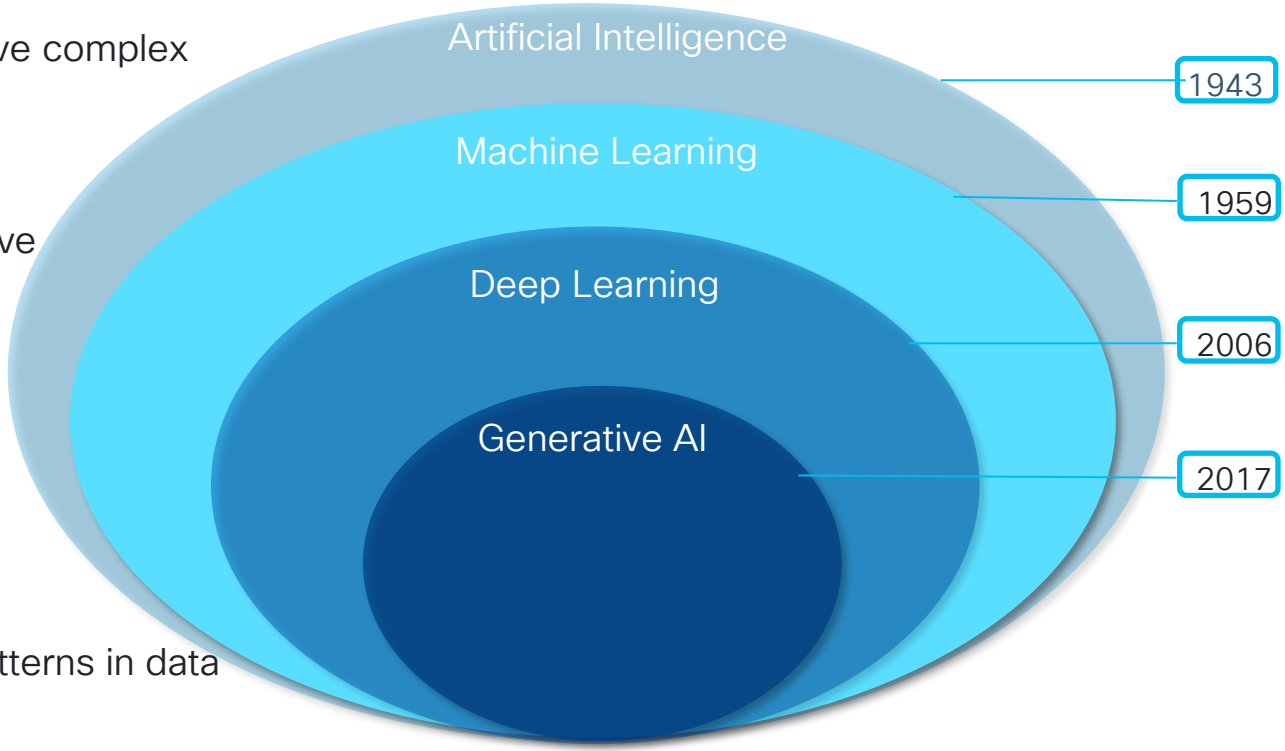
Defining Artificial Intelligence

Artificial Intelligence
Simulates human intelligence to solve complex tasks

Machine Learning
Enables systems to learn and improve from data

Deep Learning
Processes unstructured data using neural networks

Generative AI
A branch of deep learning
Creates new content by learning patterns in data



What is GPT?

Generative	AI models designed to create new content, such as text, images, or videos.
Pre-Trained	Models trained on vast datasets before application-specific fine-tuning.
Transformer	A neural network architecture focusing on contextual relationships in data (introduced by Google, 2017)

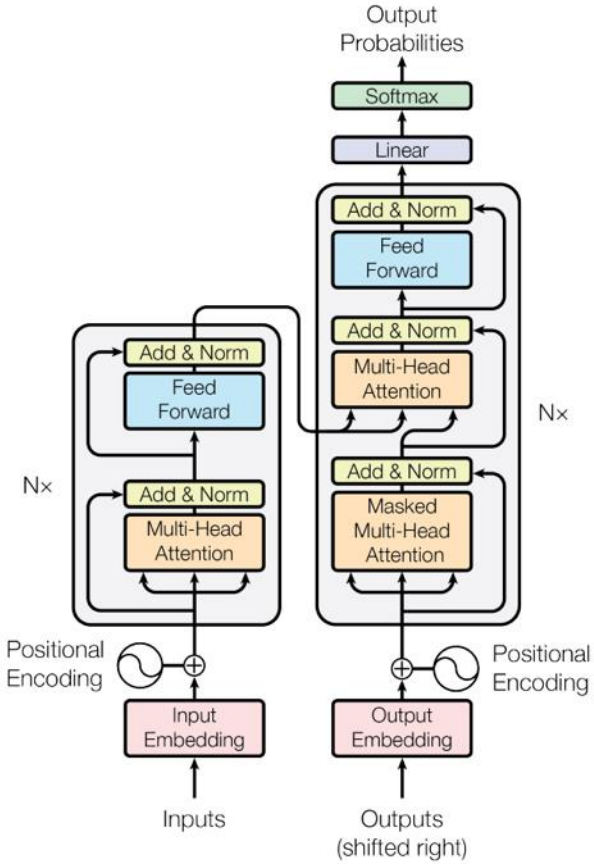


Figure 1: The Transformer - model architecture.

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com
Noam Shazeer* Google Brain noam@google.com
Niki Parmar* Google Research nikip@google.com
Jakob Uszkoreit* Google Research usz@google.com
Llion Jones* Google Research llion@google.com
Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu
Łukasz Kaiser* Google Brain lukaszkaiser@google.com
Illia Polosukhin* ‡ illia.polosukhin@gmail.com

arXiv:1706.03762 **[cs.CL]**

What makes GenAI different?

Deterministic...

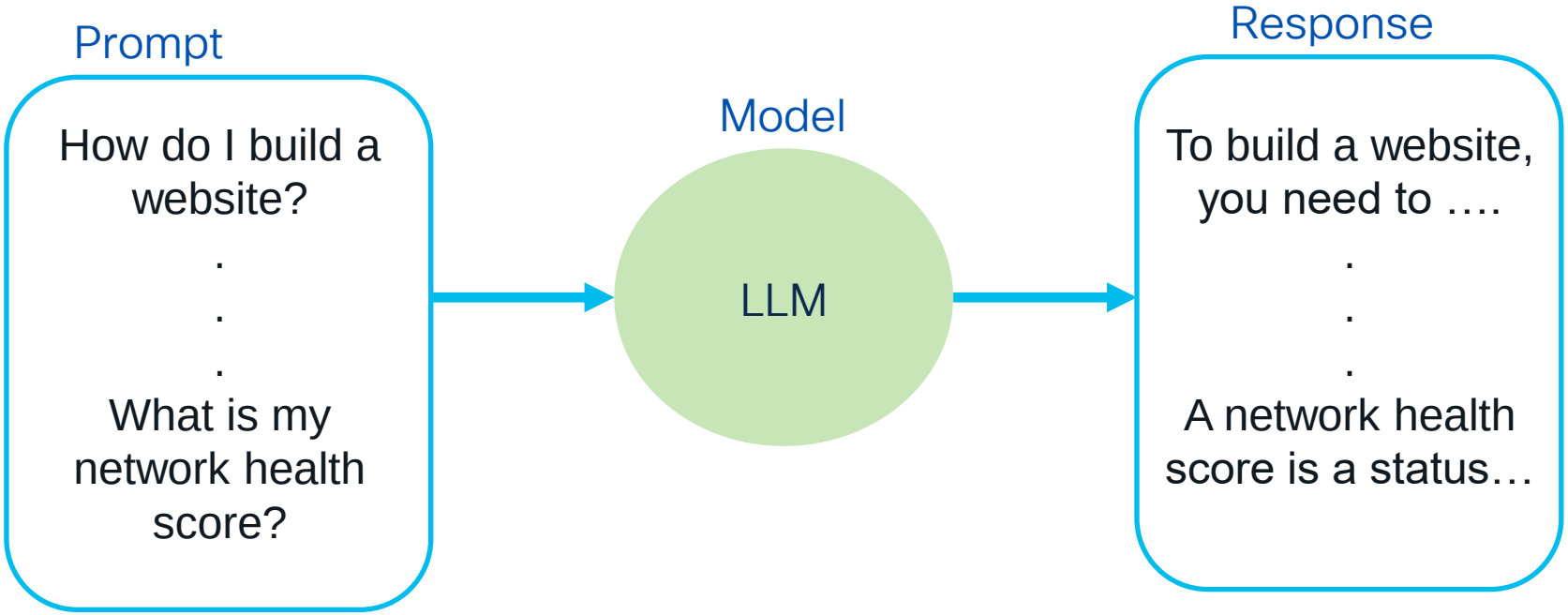
Vs

Probabilistic...

Input A equals output B

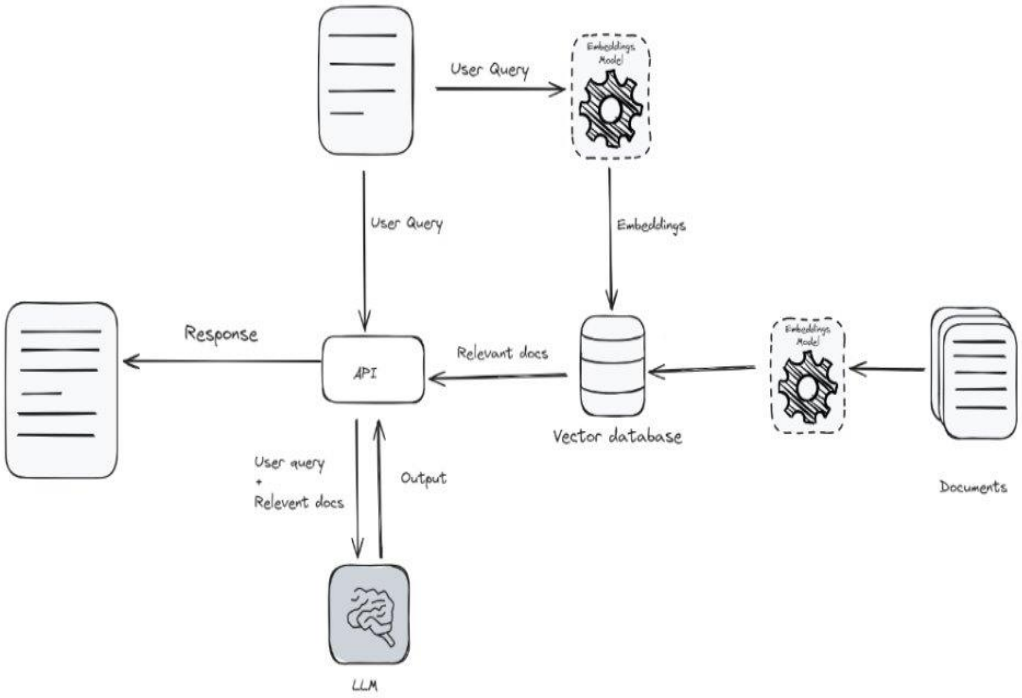
Input A could equal B, C, D, etc

GenAI Prompt and Response



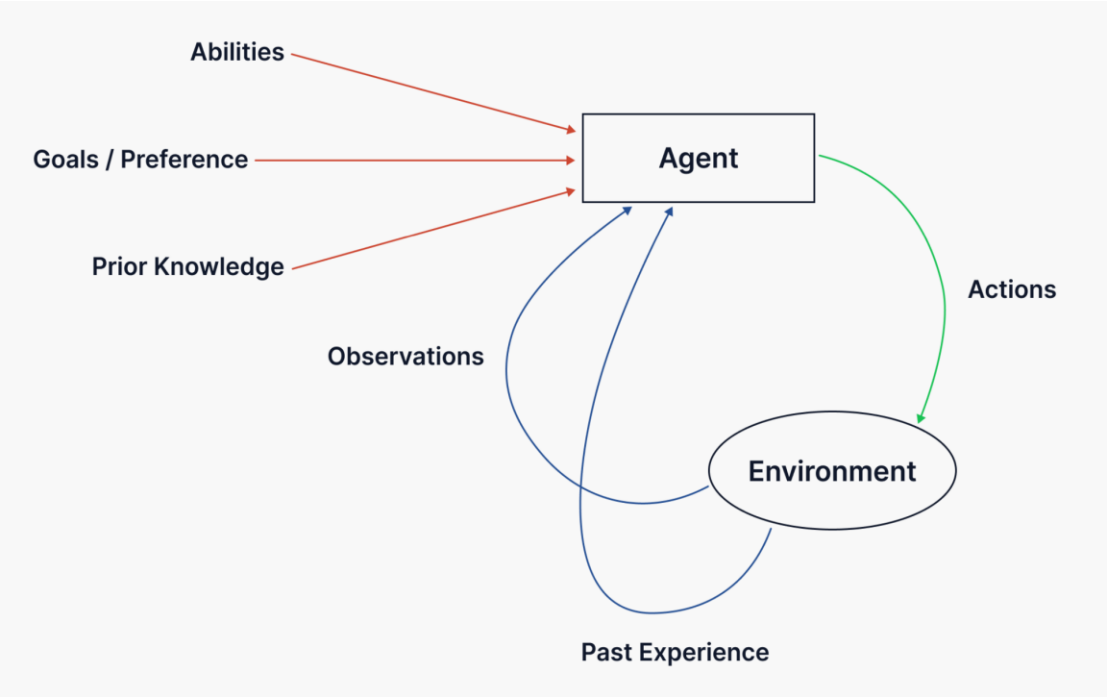
What is Retrieval Augmented Generation (RAG)

- The process of optimizing the output of a large language model, so it references an authoritative knowledge base outside of its training data sources before generating a response.



What is an AI Agent?

A software program that uses artificial intelligence (AI) to perform tasks, answer questions, and solve problems.



Introduction to GitHub Copilot

slido

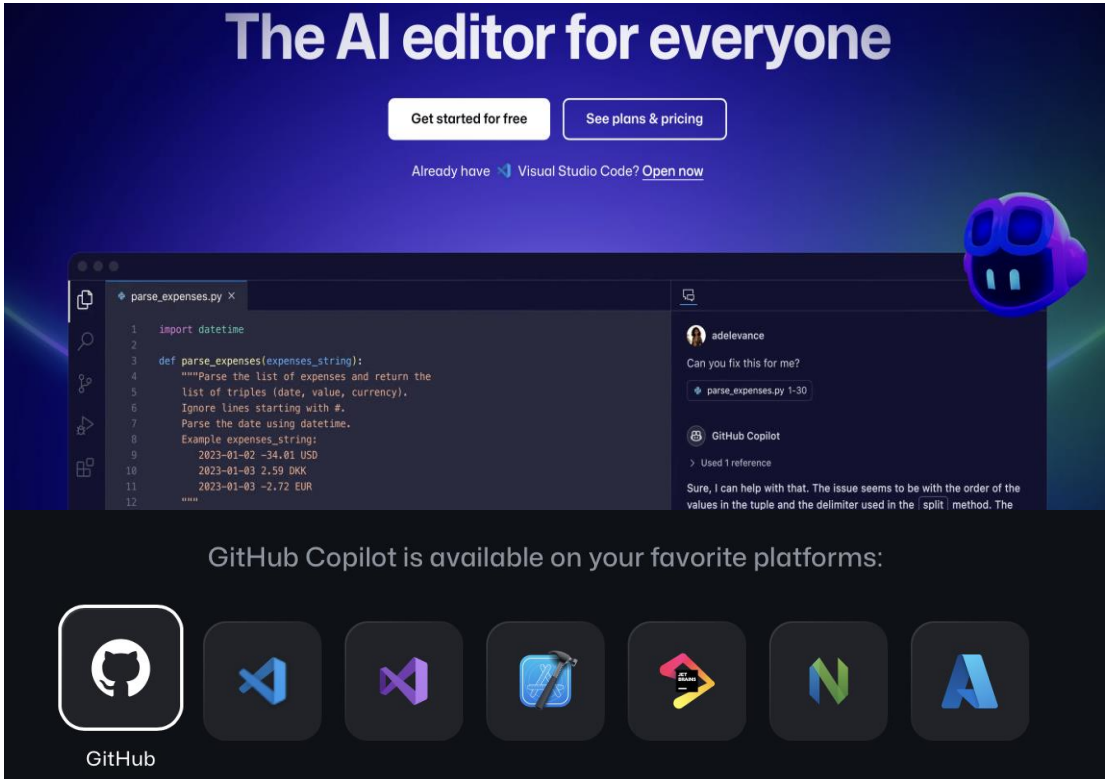
Please download and install the Slido app on all computers you use

Rate your experience level using Github Copilot?

ⓘ Start presenting to display the poll results on this slide.

GitHub Copilot

- Developed by GitHub and OpenAI in 2021
- Supports Python, JavaScript, TypeScript, Ruby, Go, C# and C++
- Developers are using it to write code 55% faster

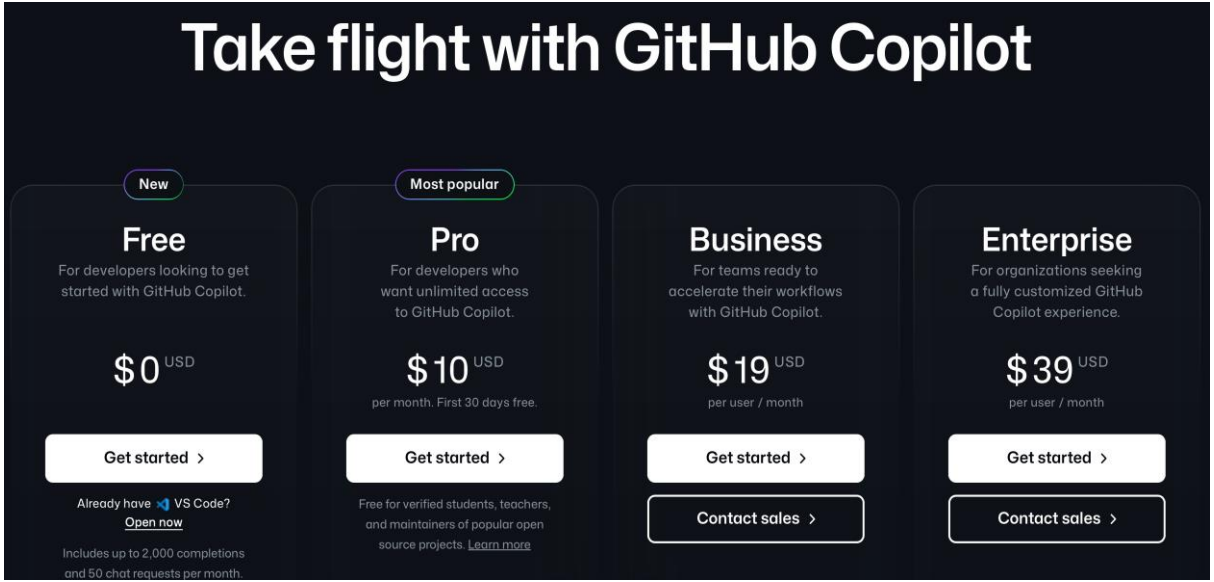


Why Choose Copilot Over ChatGPT for Coding?

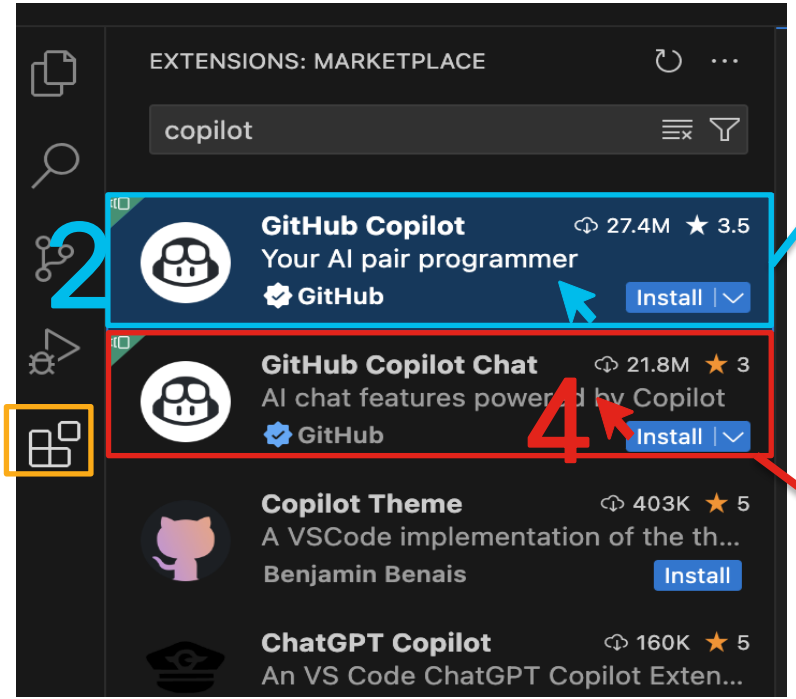
GitHub Copilot	ChatGPT
Purpose-Built for Developers	General Purpose
More Accurate Code	Less Accurate Code
Context-Aware	Limited Code Context
Seamless IDE Integration	No IDE Integration
Developer focused Interface	Familiar/Friendly Interface

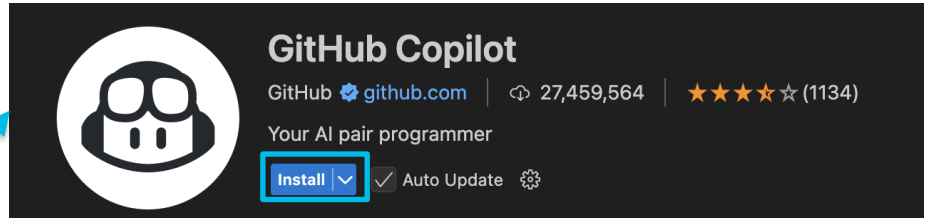
Prerequisites To Using GitHub Copilot

- Active GitHub account
- Subscription to GitHub Copilot
- Supported IDE
- Internet connection

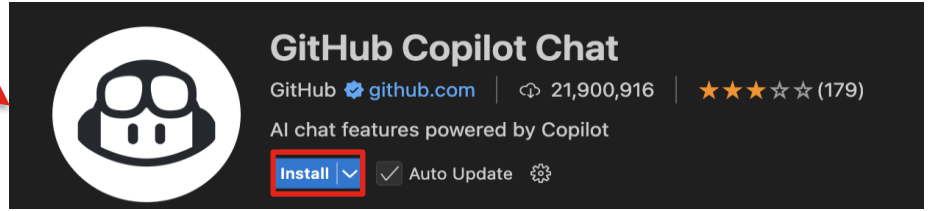


Install GitHub Copilot with Chat in VS Code



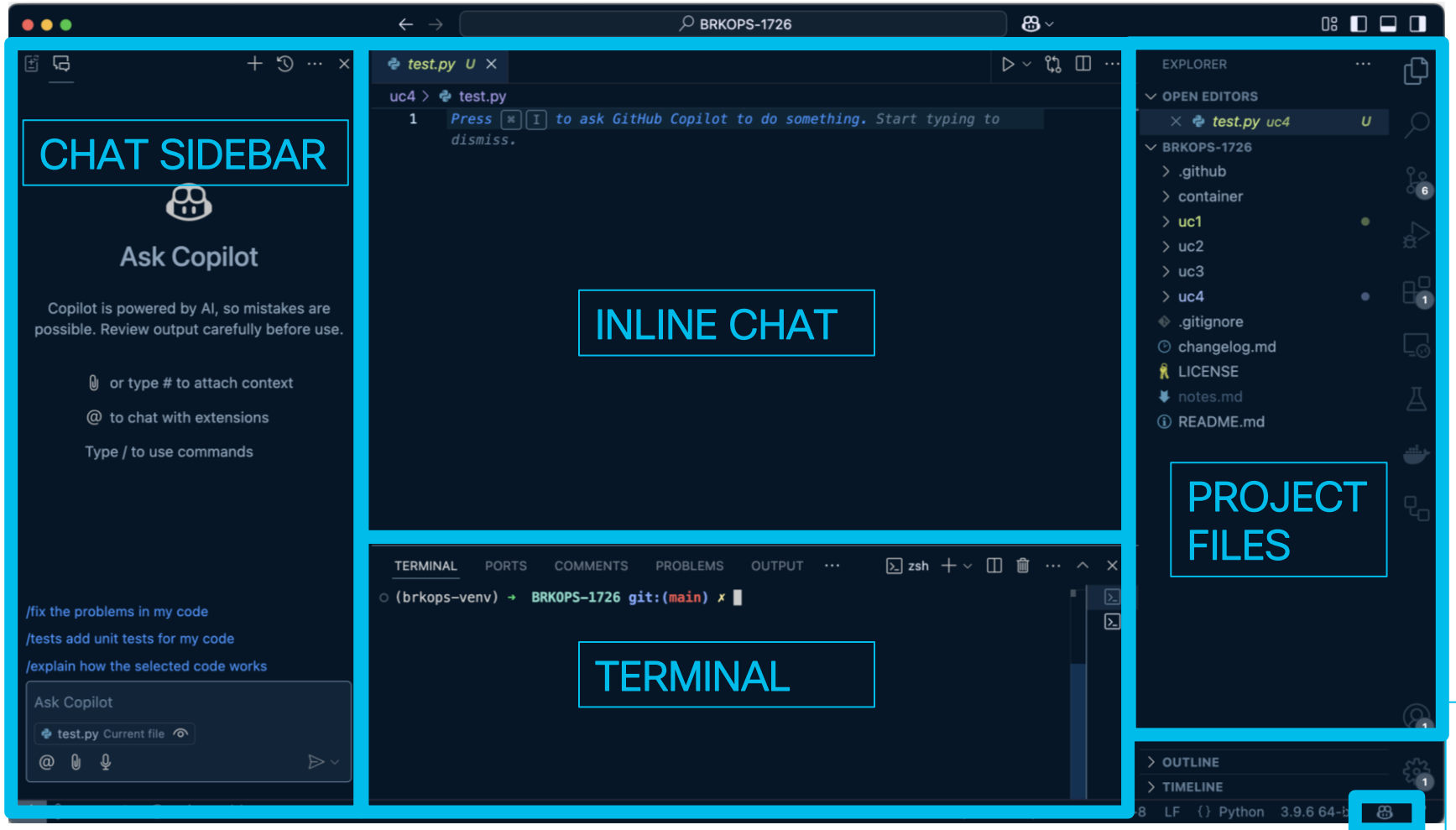


3 Click Install button for Copilot
(shows Uninstall when already installed)



5 Click Install button for Copilot Chat
(shows Uninstall when already installed)

VS Code – Copilot View



Copilot Status <-

Github Copilot Prompt Engineering

Three “S” Principle

1. SIMPLE
2. SPECIFIC
3. SHORT

Github Copilot Prompt Engineering

Three “S” Principle

1. SIMPLE

   a) Breakdown the code to smaller steps and prompt Copilot one step at a time

   b) Copilot performs better with smaller steps, reduces Hallucinations

   c) Can help to use the word “SIMPLE” in your prompt

Github Copilot Prompt Engineering

Three “S” Principle

2. SPECIFIC

   a) Provide Specific Context to Copilot by leveraging in-build Agents

   b) Use @workspace agent to have Copilot review all of the project files when answering questions

Github Copilot Prompt Engineering

Three “S” Principle

3. SHORT
    1. You don’t need proper grammar, spelling, or complete sentences when Prompting Copilot
    2. Keep Prompt short and to the point for better results

Demo 1 – Create a testbed File

Single Prompt:

```
create a testbed yaml file with two Cisco devices

1) device name CPS-FE1; type switch; os iosxe;
credentials: username admin, password Cisco;
connections set to CLI: ip 10.1.1.5, protocol ssh;


2) device name dnac; type controller; os linux;
credentials: username admin, password Cisco;
connections set to rest: ip 10.1.1.1, protocol
https;
```

Demo 1 – Create a testbed.yml File

brkops1726v

testbed.yml

testbed.yml

1 Press ⌘ I to ask GitHub Copilot to do something. Start typing to dismiss.

Ask Copilot

Copilot is powered by AI, so mistakes are possible. Review output carefully before use.

or type # to attach context

@ to chat with extensions

Type / to use commands

/fix the problems in my code

/tests add unit tests for my code

/explain how the selected code works

Ask Copilot

testbed.yn Current

TERMINAL PORTS COMMENTS PROBLEMS OUTPUT DEBUG CONSOLE

zsh

(brkops-venv) → brkops1726v touch testbed.yml

(brkops-venv) → brkops1726v

EXPLORER

OPEN EDITORS 1 unsaved

testbed.yml

BRKOPS1726V

uc1

uc2

uc3

testbed.yml

OUTLINE

TIMELINE

0 0 0

Ln 1, Col 1 Spaces: 2 UTF-8 LF YAML

Demo-1

Network Automation Use-Cases

Network Automation with Copilot and GenAI

Explore use-cases with Github Copilot and GenAI

- UC1: Network Device Data Extraction
- UC2: Network Device Workflow Creation
- UC3: Gen AI Agent based Automation

UC1: Network Device Data Extraction

UC1: Network Device Data Extraction

Objective: Connect to Cisco Switch and extract a list of “show commands” using Github Copilot

Functional Steps:

1. Utilize testbed file from DEMO-1 with device connectivity details
2. Create “show_cmds.py” python file
3. Use python pyats libraries to connect to device via SSH
4. Retrieve a list of show commands [*show version, show ip int bri, show vlan*] from the device
5. Format the output and save it to a file

UC1: Network Device Data Extraction

MULTIPE PROMPTS:

```
Import genie testbed, logging, and datetime
libraries, then load the testbed.yml file

Connect to device “FE1” and set prompt_recovery=true

Run following commands and parse the output: “show
version”, “show ip interface brief”, ”show vlan”
Then save that output to file with current date

Disconnect from device

Simplify and improve overall code debuggability
```

Demo 2 – Network Device Data Extraction

brkops1726v

prompt.md ● sh_cmds.py ●

uc1 > sh_cmds.py

1 Press ⌘ I to ask GitHub Copilot to do something. Start typing to dismiss.

Ask Copilot

Copilot is powered by AI, so mistakes are possible. Review output carefully before use.

or type # to attach context

@ to chat with extensions

Type / to use commands

/fix the problems in my code

/tests add unit tests for my code

/explain how the selected code works

Ask Copilot

sh_cmds.py Current file

TERMINAL PORTS COMMENTS PROBLEMS OUTPUT

zsh - uc1

(brkops-venv) → uc1

EXPLORER

OPEN EDITORS 2 unsaved

prompt.md uc1

sh_cmds.py uc1

BRKOPS1726V

uc1

prompt.md

sh_cmds.py

testbed.yml

OUTLINE

TIMELINE

Ln 1, Col 1 Spaces: 4 UTF-8 LF Python 3.9.6 64-bit

Demo-2

UC1: Network Device Data Extraction Recap

- Learned how to breakdown a project into smaller steps to better utilize Github Copilot capabilities
- Importance of Prompting with the Three “S” principle
- Show-cased a simple example of Network automation use-case using Copilot

Okay that's great, but what-if I didn't have to trigger my script manually every time!

UC2: Network Device Workflow Creation

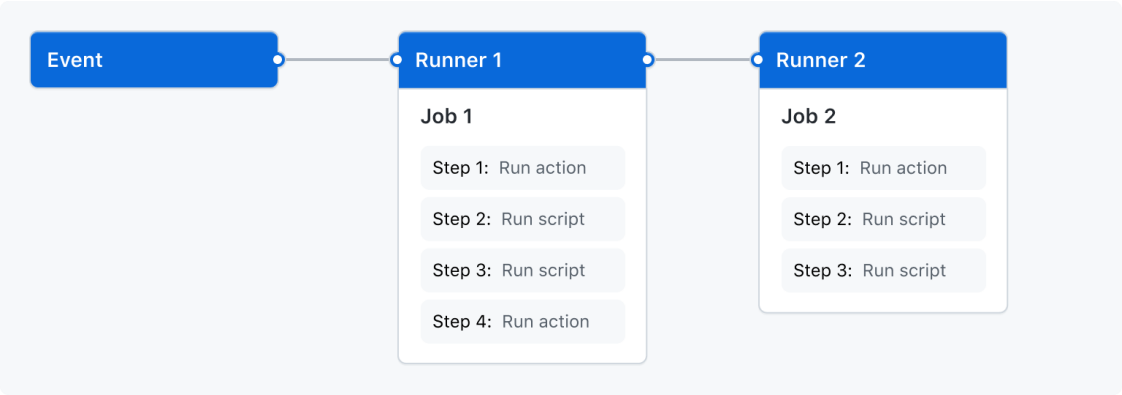
Github Actions - Workflows

Workflows are configurable automated process that will run one or more jobs.

Workflows will run when triggered by an Event in your repository, or they can be triggered manually, or at a defined schedule.

Workflows YAML files are defined in the .github/workflows directory.

A Runner is a server that runs your workflows when they're triggered.



Github Actions – Runner Setup

gtamilse1 / BRKOPS-1726

Type / to search

Code | Issues | Pull requests | Actions | Projects | Security | Insights | Settings

General

Access

Collaborators

Code and automation

Branches

Tags

Rules

Actions

General

Runners

Webhooks

Runners

New self-hosted runner

Host your own runners and customize the environment used to run jobs in your GitHub Actions workflows. Learn more about self-hosted runners.

Runners	Status
GT-MAC1 self-hosted macOS X64 gtmac	Idle
WIN-Q73ABTFIH81 self-hosted X64 Windows win	Idle

Github Actions – Runner Setup

Code | Issues | Pull requests | Actions | Projects | Security | Insights | Settings

General

Access

Collaborators

Code and automation

Branches

Tags

Rules

Actions

General

Runners

Webhooks

Codespaces

Pages

Security

Code security

Deploy keys

Secrets and variables

Integrations

Runners / Add new self-hosted runner · gtamilse1/BRKOPS-1726

Adding a self-hosted runner requires that you download, configure, and execute the GitHub Actions Runner. By downloading and configuring the GitHub Actions Runner, you agree to the GitHub Terms of Service or GitHub Corporate Terms of Service, as applicable.

Runner image

macOS | Linux | Windows

Architecture

x64

Download

```
# Create a folder
$ mkdir actions-runner && cd actions-runner
# Download the latest runner package
$ curl -o actions-runner-osx-x64-2.321.0.tar.gz -L https://github.com/actions/runner/releases/download/v2.321.0/actions-runner-osx-x64-2.321.0.tar.gz
# Optional: Validate the hash
$ echo "b2c91416b3e4d579ae69fc2c381fc50dbda13f1b3fcc283187e2c75d1b173072  actions-runner-osx-x64-2.321.0.tar.gz" | shasum -a 256 -c
# Extract the installer
$ tar xzf ./actions-runner-osx-x64-2.321.0.tar.gz
```

UC2: Network Device Workflow Creation

Objective: Create a workflow to automate a method of procedure (MOP) for device provisioning

Workflow automation steps:

1. Connect to Cisco switch
2. Collect pre-check commands
3. Configure new Vlan
4. Collect post-check commands
5. Ensure configuration was successful

UC2: Network Device Workflow Creation

Functional Steps:

1. Utilize “testbed.yml” file from DEMO-1 with device connectivity details
2. Utilize “show_cmds.py” file from DEMO-2 to collect pre and post check commands
3. Create “create_vlan.py” file to provision new VLAN 1111
4. Create “diff_compare.sh” file to confirm VLAN creation
5. Create a “github-ci.yml” workflow file with all steps in sequence to execute as single flow

UC2: Network Device Workflow Creation

Prompts for “create_vlan.py”:

```
Import genie testbed, logging, and datetime
libraries, then load the testbed.yml file

Define function configure_vlan which will create and
configure a vlan_id and vlan_name

Connect to device “FE1” and set prompt_recovery=true

Configure VLAN 1111 with name test

Disconnect from the device

Improve overall code debuggability
```

UC2: Network Device Workflow Creation

Prompts for “diff_compare.sh”:

create a shell script that diff compares two files that start with the name command_outputs. And remove unnecessary lines in diff output

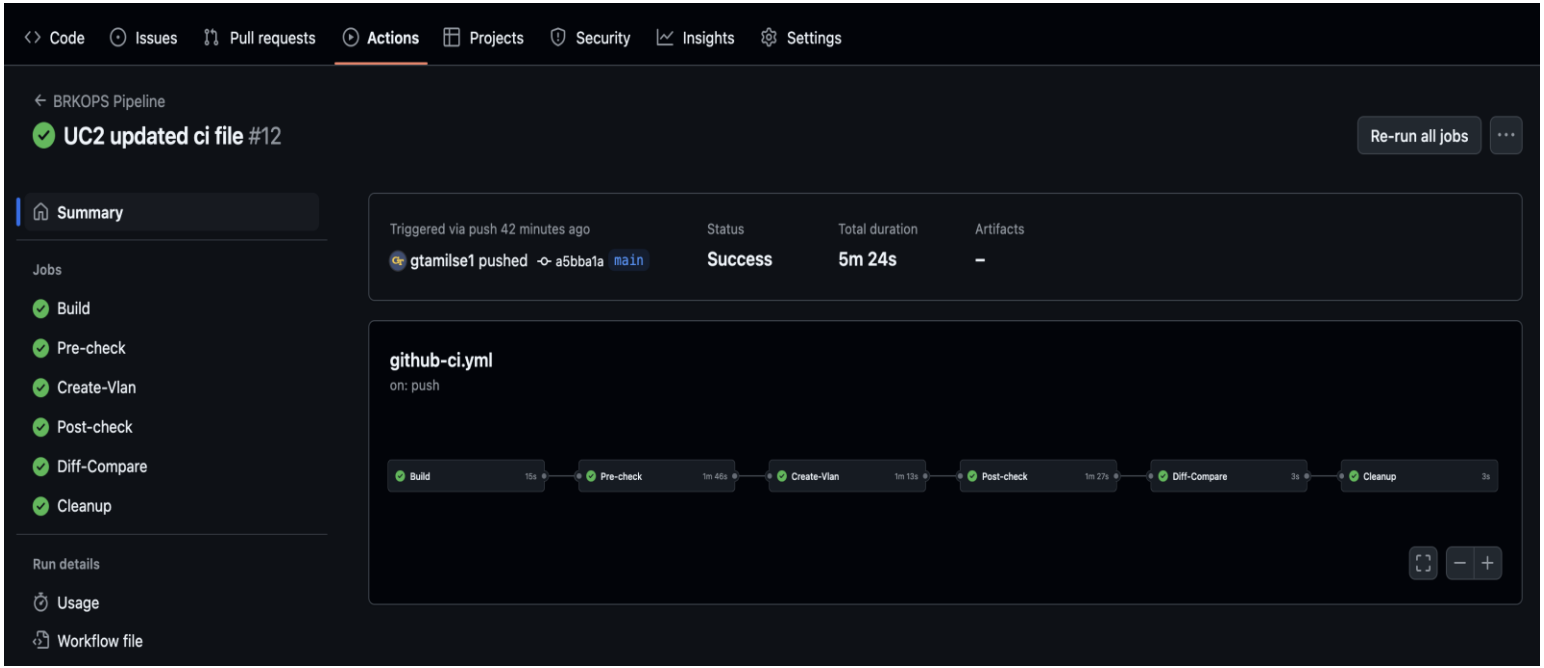
```
files=(command_outputs*)
if [ ${#files[@]} -eq 2 ]; then
    diff -u "${files[0]}" "${files[1]}" | grep -E "^\+|^-"
else
    echo "There are not exactly two files matching the pattern."
fi
```

UC2: Network Device Workflow Creation

- Github-ci.yml – Pipeline File ->
- Located - .github/workflows/ dir
- Triggers on git push/pull request
- Contains 6 Jobs:
  - Build, Pre-Check, Create-Vlan,
  - Post-Check, Diff-Compare, Cleanup
- Executes the individual scripts in sequence

```yaml
name: BRKOPS Pipeline
on:
  push:
    branches: [ main ]
  pull_request:
    branches: [ main ]
jobs:
  Build:
    runs-on:
      - macos
    steps:
    - uses: actions/checkout@v2
    - name: Set up Python 3.12
      uses: actions/setup-python@v3
      with:
        python-version: "3.12"
    - name: Install dependencies
      run: |
        python3 -m pip install --upgrade pip
        pip3 install flake8 pytest pyats genie
        if [ -f requirements.txt ]; then pip install -r requirements.txt; fi
  Pre-check:
    needs: Build
    runs-on:
      - macos
    steps:
    - name: Run Pre-Checks
      run: |
        pwd
        ls -ltr
        cd uc2/demo/
        python sh_cmds.py
        ls -ltr
```

UC2: Network Device Workflow Creation



Demo 3 – Network Device Workflow Creation

brkops1726v

create_vlan.py 1 | diff_compare.sh | github-ci.yml

uc2 > create_vlan.py > ...

```python
import logging
from genie.testbed import load
from datetime import datetime

# Configure logging
logging.basicConfig(level=logging.DEBUG, format='%(asctime)s - %(levelname)s - %(message)s')

def configure_vlan(device, vlan_id, vlan_name):
    try:
        # Enter configuration mode
        device.configure([
            f'vlan {vlan_id}',
            f'name {vlan_name}'
        ])
        logging.info(f"Configured VLAN {vlan_id} with name {vlan_name}")
    except Exception as e:
        logging.error(f"Failed to configure VLAN {vlan_id}: {e}")

def main():
    try:
        # Load the testbed file
        testbed = load('/Users/gtamilse/Downloads/GITHUB/BRKOPS-1726/uc1/demo/testbed.yml')
        logging.info("Testbed loaded successfully")

        # Connect to the device "FE1"
        device = testbed.devices['FE1']
        device.connect(prompt_recovery=True)
        logging.info(f"Connected to {device.name} with prompt_recovery set to True")

        # Configure VLAN
        vlan_id = 1111
        vlan_name = "Test_VLAN"
```

Ask Copilot

Copilot is powered by AI, so mistakes are possible. Review output carefully before use.

or type # to attach context

@ to chat with extensions

Type / to use commands

/fix the problems in my code

/tests add unit tests for my code

/explain how the selected code works

Ask Copilot

create_vlar Current

EXPLORER

OPEN EDITORS

create_vlan.py... 1

diff_compare.sh uc2

github-ci.yml uc2

BRKOPS1726V

uc1

uc2

create_vlan.py 1

diff_compare.sh

github-ci.yml

sh_cmds.py

OUTLINE

TIMELINE

Ln 18, Col 1 | Spaces: 4 | UTF-8 | LF | Python 3.9.6 64-bit

Demo-3

UC2: Network Device Workflow Creation Recap

- Learned about Github Actions feature to create workflows that can trigger automatically
- Utilized Github Copilot to create sequential scripts that achieve specific tasks
- Show-cased a simple example of Network Method of Procedure (MOP) automation

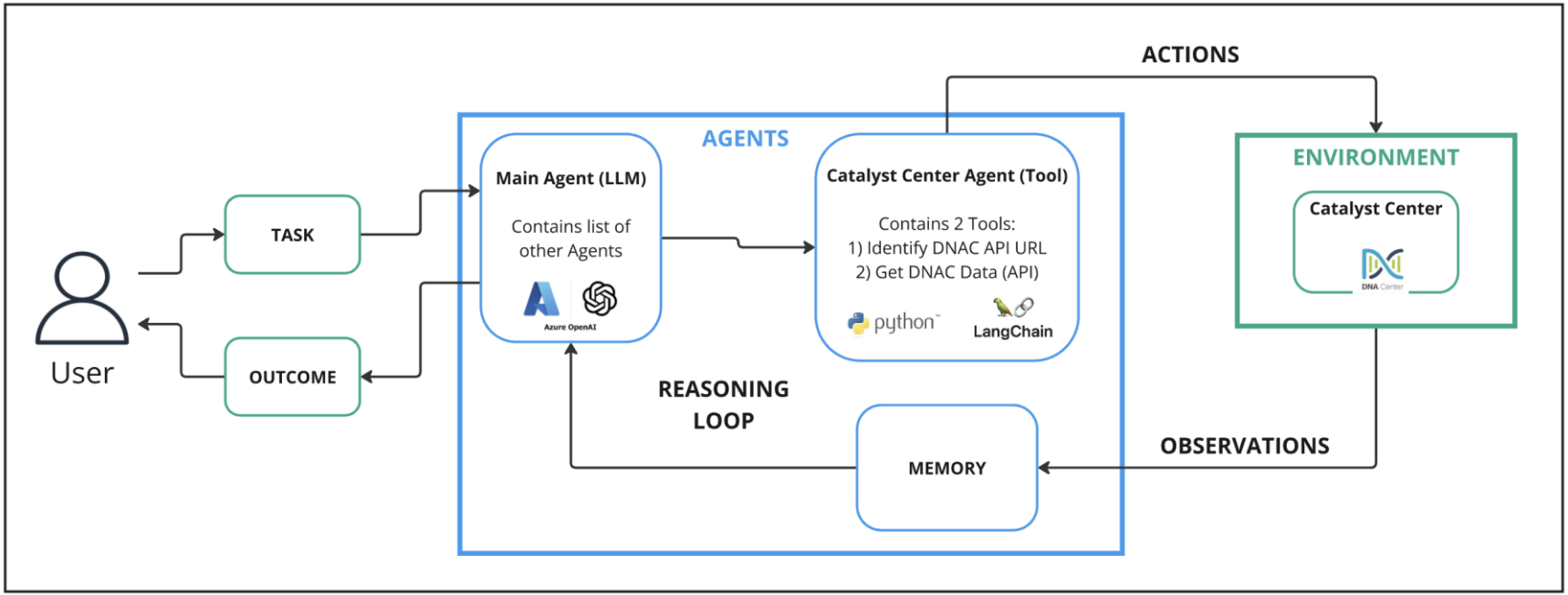
Hey that’s cool, but what about automation using Controllers and can GenAI help?

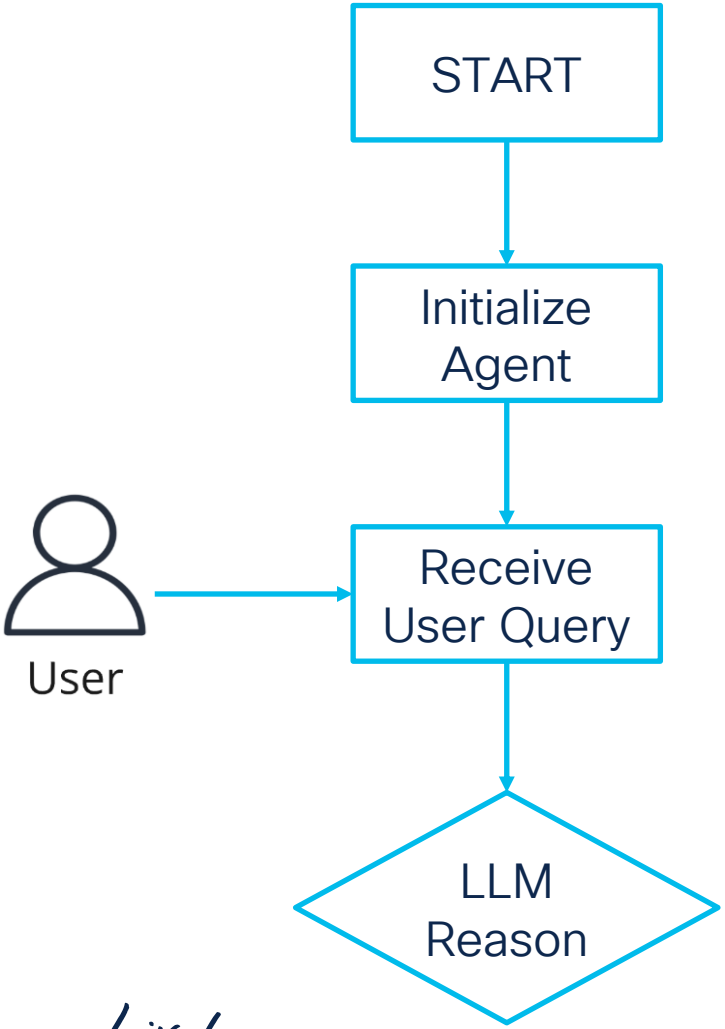
UC3: Gen AI Agent based Automation

GEN AI – ReAct Framework

- *Reasoning and Acting (ReACT):*
  - ReAct agents utilize LLMs as centralized components (brain) that concurrently reason about the environment and determine appropriate actions.
  - The agent processes generate plans (Thought), execute Actions, and Observe the response in an iterative cycle.
  - ReAct agents can integrate with external Tools (scripts/code) and APIs, selecting and employing them based on the current context and objectives.

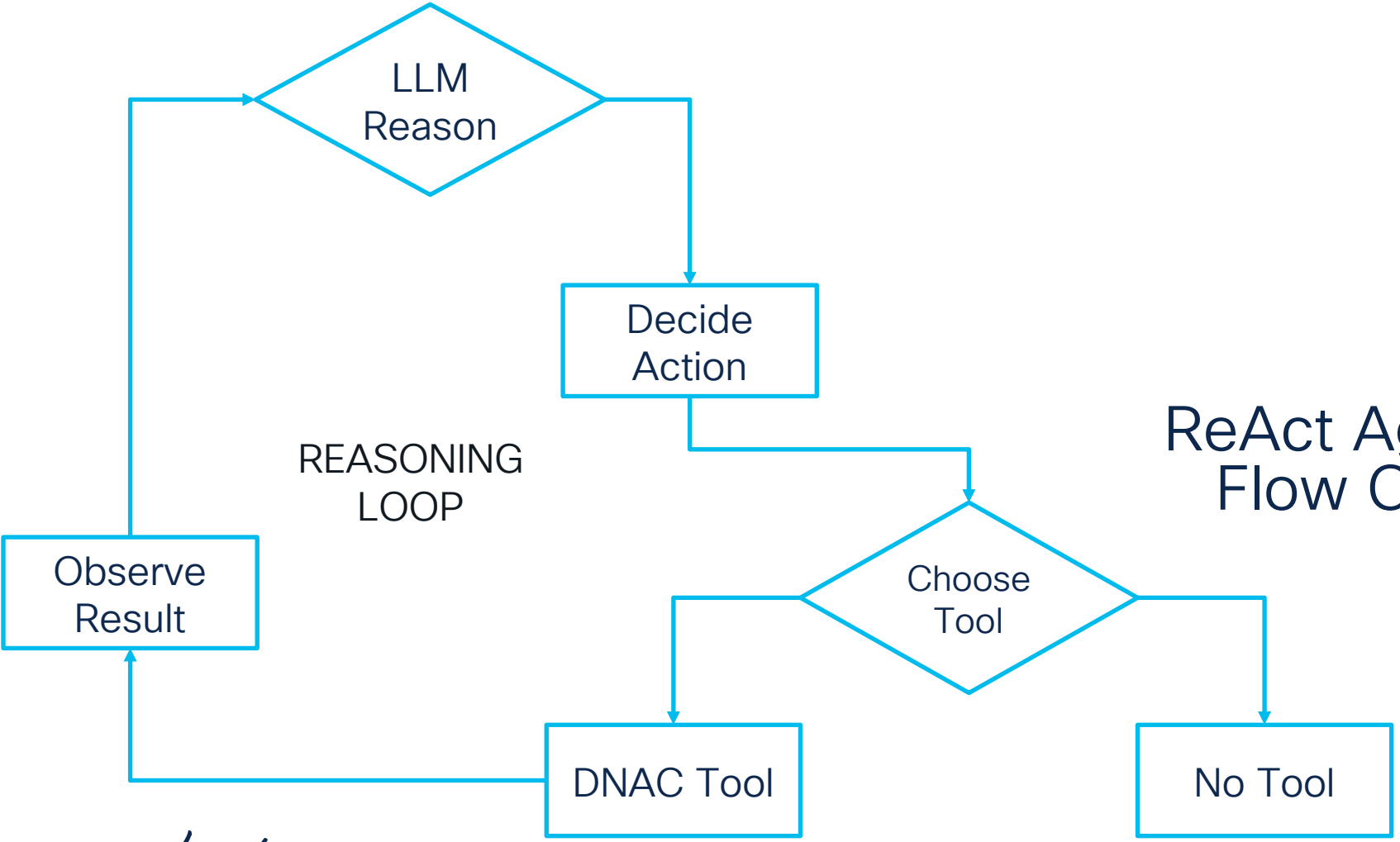
ReAct Agent Architecture



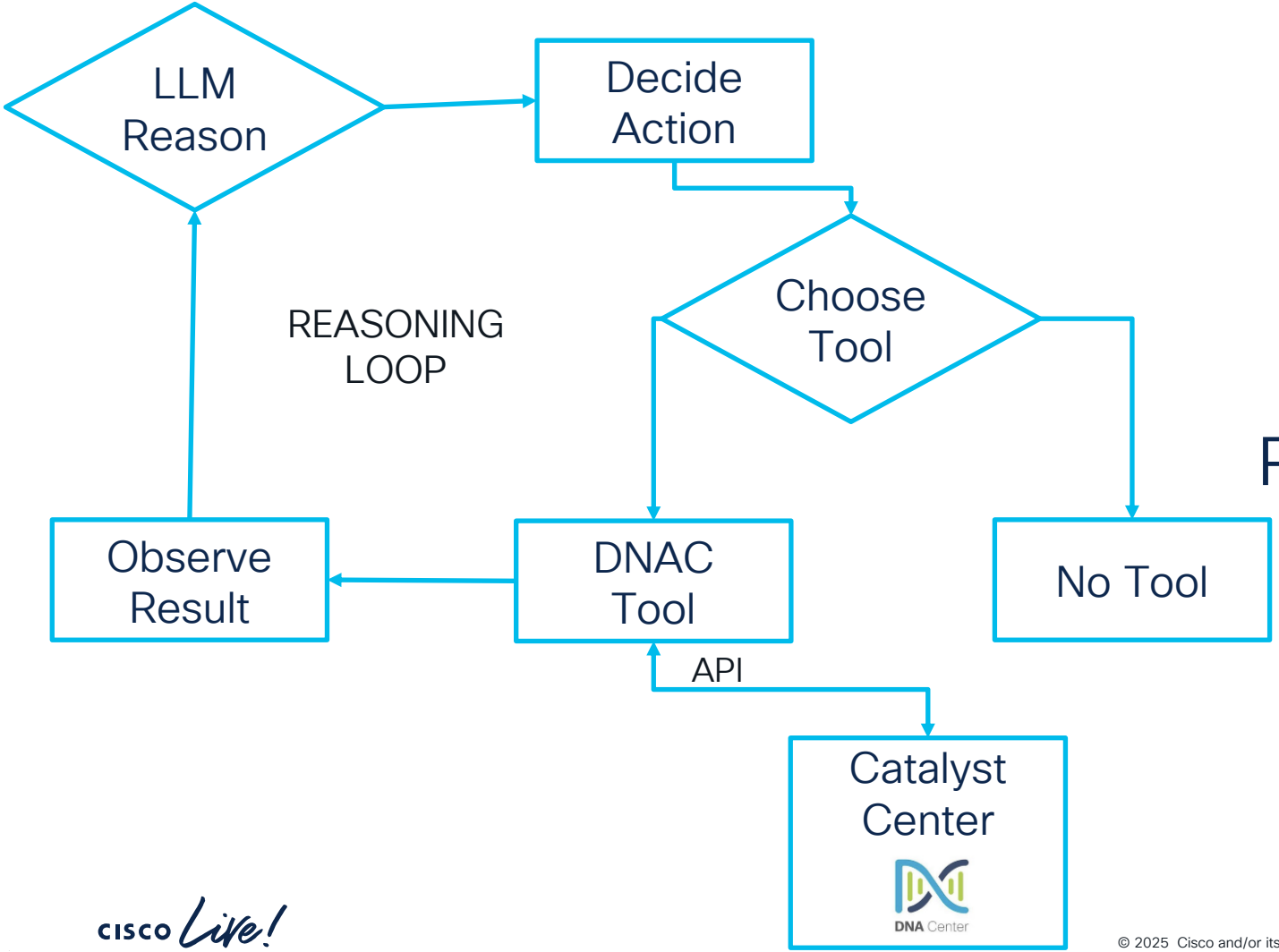


ReAct Agent Flow Chart

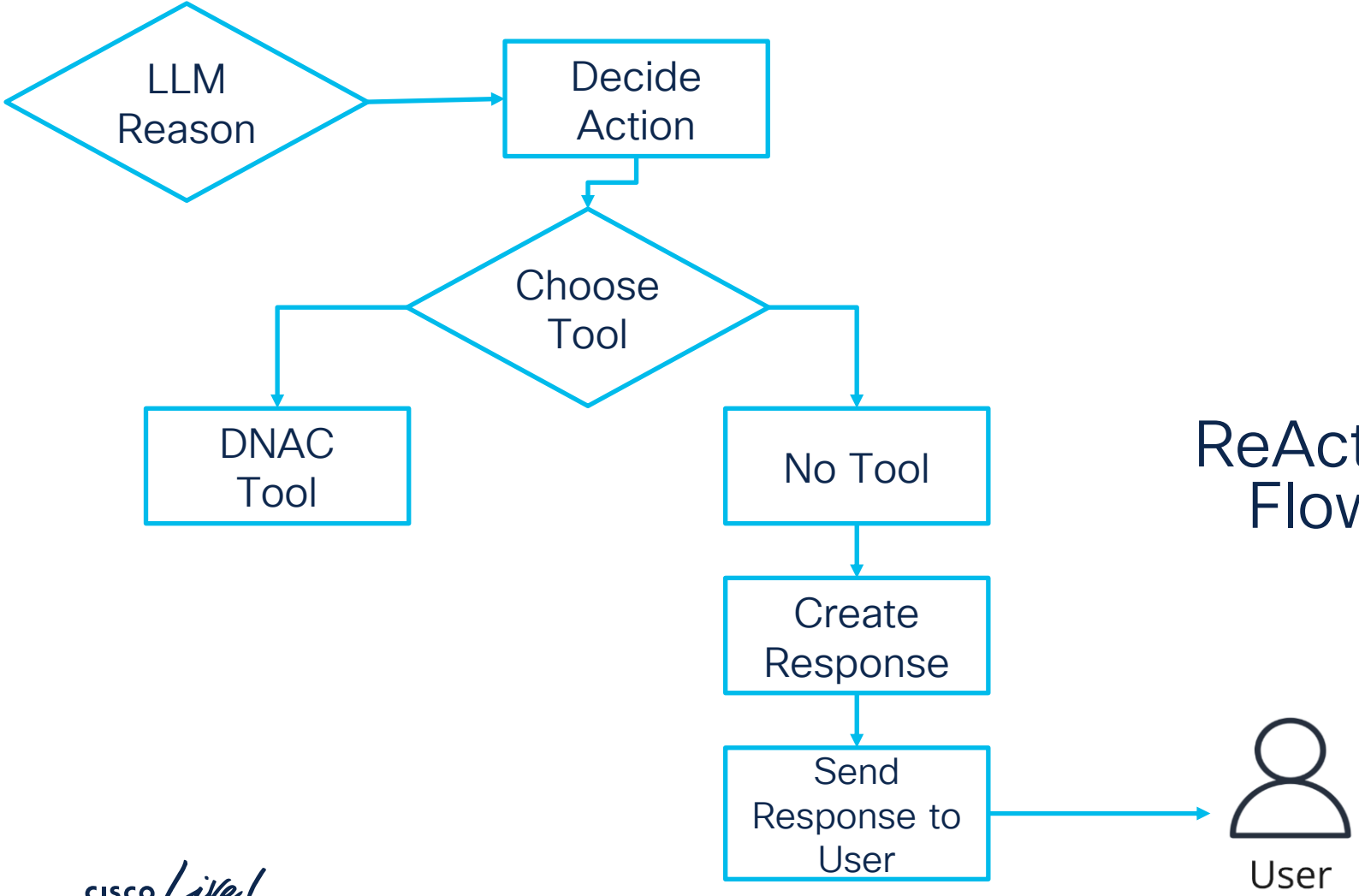
ReAct Agent Flow Chart



ReAct Agent Flow Chart



ReAct Agent Flow Chart



ReAct Agent Crafting a Prompt Template

ReAct prompt typically includes four components:

1. Current user query
2. Any previous reasoning steps and observations (memory)
3. Available tools
4. Output format instructions

ReAct Agent Prompt Template

template = """

Agent is a network assistant with the capability to manage data from Cisco Catalyst Center controllers using API Requests.

Network Instructions:
Assistant is designed to retrieve information from the Cisco Catalyst Center controller using provided tools. You MUST use these tools for checking available data and fetching that data.
Assistant has access to a list of API URLs and their associated Names provided in a ‘dnac_urls.json' file. You can use the 'Name' field to find the appropriate API URL to use.

ReAct Agent Prompt Template cont'd

Important Guidelines:

1. **If you are certain of the API URL or the Name of the data you want, use the 'get_dnac_data_tool' to fetch data.**

2. **If you are unsure of the API URL or Name, or if there is ambiguity, use the 'check_supported_url_tool' to verify the URL or Name or get a list of available ones.**

3. **If the 'check_supported_url_tool' finds a valid URL or Name, then use the appropriate tool to perform the action.**

4. **Do NOT use any unsupported URLs or Names.**

ReAct Agent Prompt Template cont'd

To use a tool, follow this format:
Thought: Do I need to use a tool? Yes

Action: the action to take, should be one of [{tool_names}]

Action Input: the input to the action

Observation: the result of the action

If the first tool provides a valid URL or Name, you MUST immediately run the correct tool for the operation (fetch, create, update, or delete) without waiting for another input.
"""

UC3: Gen AI Agent based Automation

Objective: Utilize OpenAI ChatGPT and Copilot to create a ReAct Agent that performs “Get” requests to Catalyst Center via API

5 Components:

1. Env File – which contains the DNAC Server IP, Username, and Password
2. Main Agent –the ReAct Agent that utilizes LLM (OpenAI GPT-4o-mini)
3. DNAC Agent – which contains python code to check and make API calls to DNAC; as known as the “tool agent”
4. DNAC APIs List – contains a list of Catalyst Center APIs in JSON format for DNAC agent tool to use
5. Browser based Chat interface – handles user queries and displays the response as well as past conversation history

UC3: Gen AI Agent based Automation

Functional Steps:

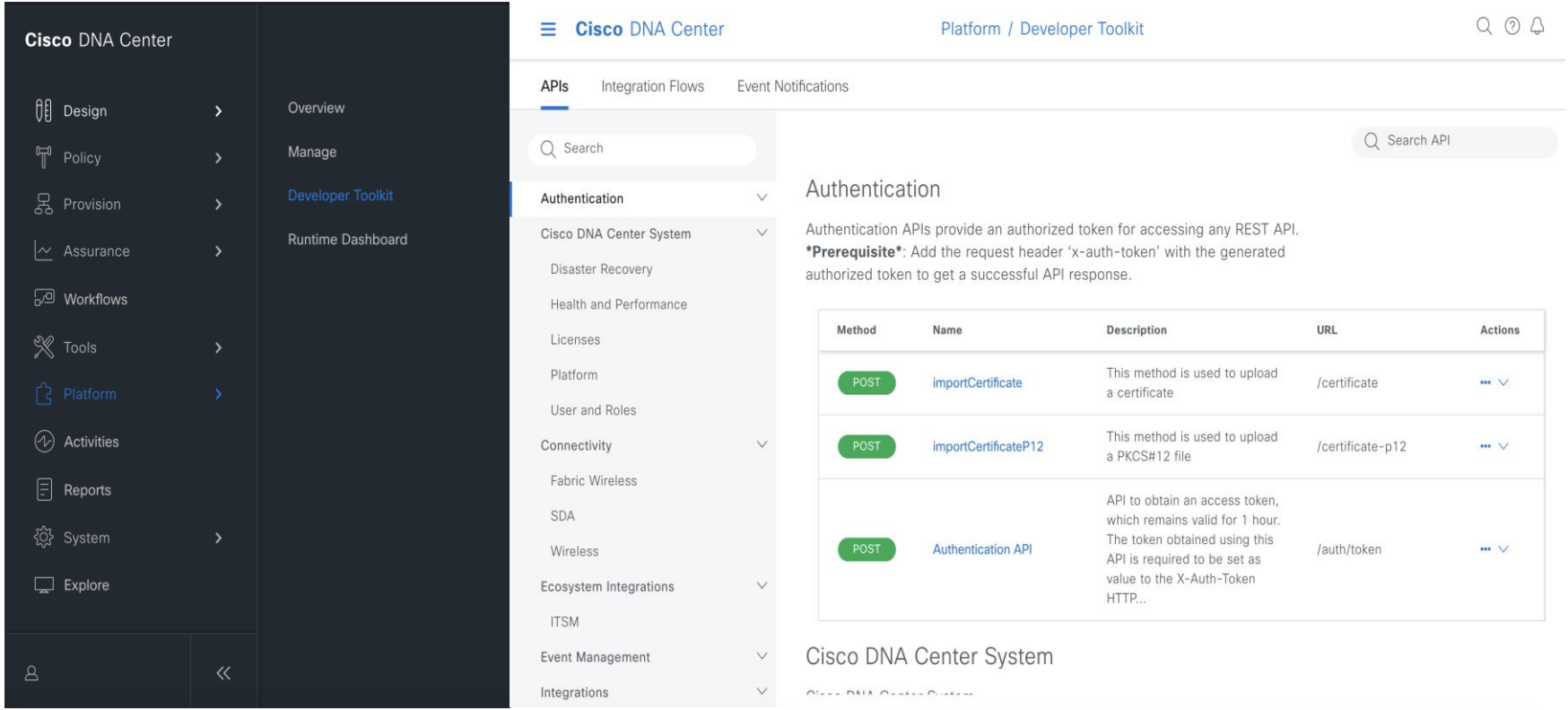
1. Create an “dnac_urls.json” file that contains list of Catalyst Center APIs and their names.

2. Create the “dnac_agent.py” file which contains 2 Tools (python code):
   a) Check the dnac_urls.json file to find the closest matching API call to user query
   b) Executes the api call against Catalyst Center

3. Create the “main_agent.py” that initializes the LLM Agent (OpenAI) and connects it to the dnac tools. Also adds frontend chat interface.

4. Initiate user request in chat window and observe ReAct Agent retrieve results from Catalyst Center.

Cisco DNA Center APIs



Demo 4 – Gen AI Agent based Automation

BRKOPS-1726

dnac_agent.py 8, U | main_agent.py 5, U | {} dnac_urls.json U

uc4 > test > {} dnac_urls.json > {} 2 > URL

```
[
    {
    "URL": "/dna/system/api/v1/auth/token",
    "Name": "Authentication"
    },
    {
    "URL": "/dna/intent/api/v1/task/{task_id}",
    "Name": "Task Status"
    },
    {
    "URL": "/dna/intent/api/v1/network-health",
    "Name": "Network Health"
    },
    {
    "URL": "/dna/intent/api/v1/client-health",
    "Name": "Client Health"
    },
    {
    "URL": "/dna/intent/api/v1/site-health",
    "Name": "Site Health"
    },
    {
    "URL": "/dna/intent/api/v1/site/count",
    "Name": "Site Count"
    },
    {
```

EXPLORER

OPEN EDITORS
- dnac_a... 8, U
- main_a... 5, U
- {} dnac_urls.... U

BRKOP...
- .github
- container
- uc1
- uc2
- uc3
- uc4
- .gitignore
- changelog.md
- LICENSE
- notes.md
- README.md

OUTLINE

TIMELINE

TERMINAL | PORTS | COMMENTS | PROBLEMS 13 | OUTPUT | DEBUG CONSOLE

Python - test

```
To install langchain-community run `pip install -U langchain-community`.
  warnings.warn(
INFO:__main__:Initialized OpenAI LLM successfully.
INFO:__main__:Initialized DNAC agent successfully.
INFO:__main__:Initialized main agent successfully.
```

main*+ ⊗ 0 ⚠ 13 0 | Ln 11, Col 39 (28 selected) | Spaces: 4 | UTF-8 | LF | {} JSON

Demo-4

UC3: Gen AI Agent Based Automation Recap

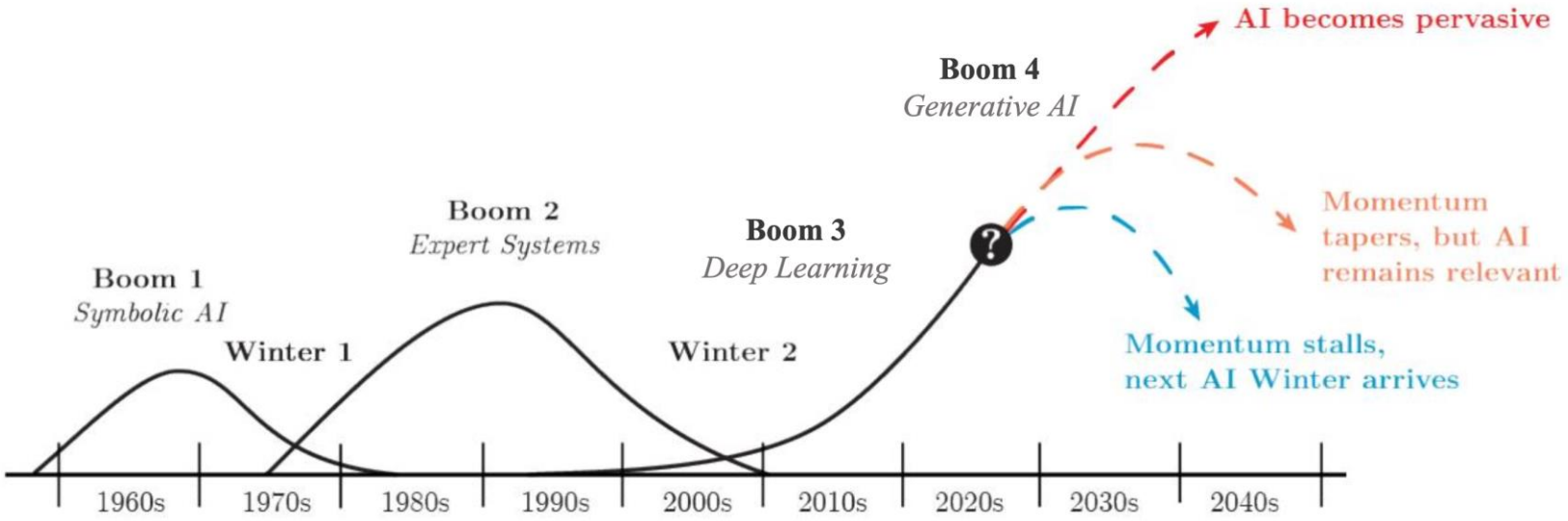
- Learned about ReAct Framework and how to integrate LLM models into automation projects
- Utilize the Thought-Action-Observation loop via Prompt Engineering to influence LLM behavior
- Show-cased how agent based architecture allows LLMs to utilize Tools to act on behalf of the user

Conclusion

Github Copilot Best Practice Recommendations

1. You are the main architect and Copilot is your assistant.
2. Outline the project pseudo-code before starting and break down the steps.
3. Keep prompts simple, specific, and short.
4. Use Inline Chat to build code in small steps.
5. Use available Copilot features (/functions) for quick support.

Where do we go with AI?



slido

Please download and install the Slido app on all computers you use

On a scale of 1 to 5, how excited are you to go try Copilot and AI Automation now?

ⓘ Start presenting to display the poll results on this slide.

Webex App

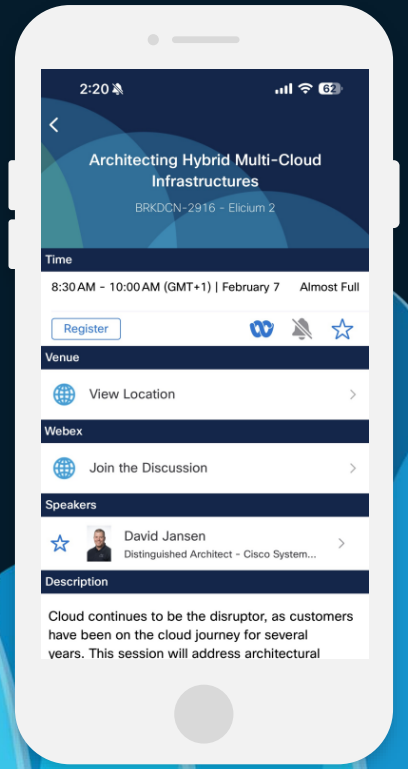
Questions?

Use the Webex app to chat with the speaker after the session

How

1. Find this session in the Cisco Events mobile app
2. Click “Join the Discussion”
3. Install the Webex app or go directly to the Webex space
4. Enter messages/questions in the Webex space

Webex spaces will be moderated by the speaker until February 28, 2025.



Fill Out Your Session Surveys

Participants who fill out a minimum of 4 session surveys and the overall event survey will get a unique Cisco Live t-shirt.

(from 11:30 on Thursday, while supplies last)

All surveys can be taken in the Cisco Events mobile app or by logging in to the Session Catalog and clicking the ‘Participant Dashboard’

Content Catalog

Continue your education

- Visit the Cisco Showcase for related demos
- Book your one-on-one Meet the Engineer meeting
- Attend the interactive education with DevNet, Capture the Flag, and Walk-in Labs
- Visit the On-Demand Library for more sessions at ciscolive.com/on-demand. Sessions from this event will be available from March 3.

Contact us at: BRKOPS-1726 Webex Space

Thank you

cisco Live!

GO BEYOND