

Transform your Network Automation Apps with GenAI

CISCO Live !

Gabi Zapodeanu
Principal Technical Marketing Engineer
Cisco Systems
@zapodeanu

Cisco Webex App

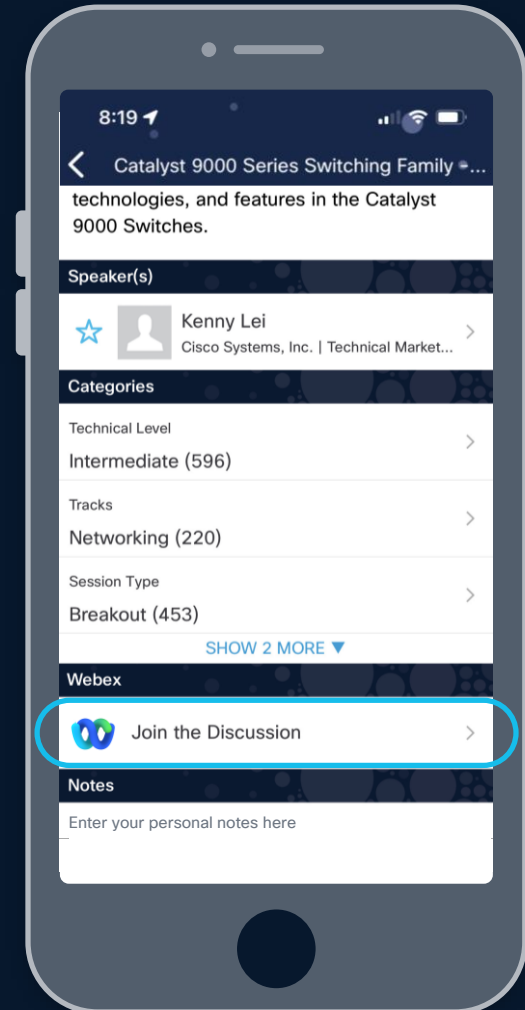
Questions?

Use Cisco Webex App to chat with the speaker after the session

How

- 1 Find this session in the Cisco Live 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 June 13, 2025.



Agenda

- 01 Catalyst Center Platform
- 02 Integrations
- 03 GenAI Essentials
- 04 AI Code Generation
- 05 GenAI Network Automation
- 06 Developer Resources
- 07 Summary

Agenda

- 01 Catalyst Center Platform
- 02 Integrations
- 03 GenAI Essentials
- 04 AI Code Generation
- 05 GenAI Network Automation
- 06 Developer Resources
- 07 Summary

Catalyst Center Platform Vision



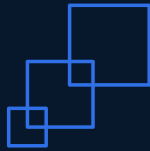
Efficient

Managing infrastructure as code while lowering OPEX



Consistent

Repeatable and precise intent-based outcomes



Scalable

Able to reduce time and effort with large network automation requirements



Secure

Secure APIs and integrations



Sustainable

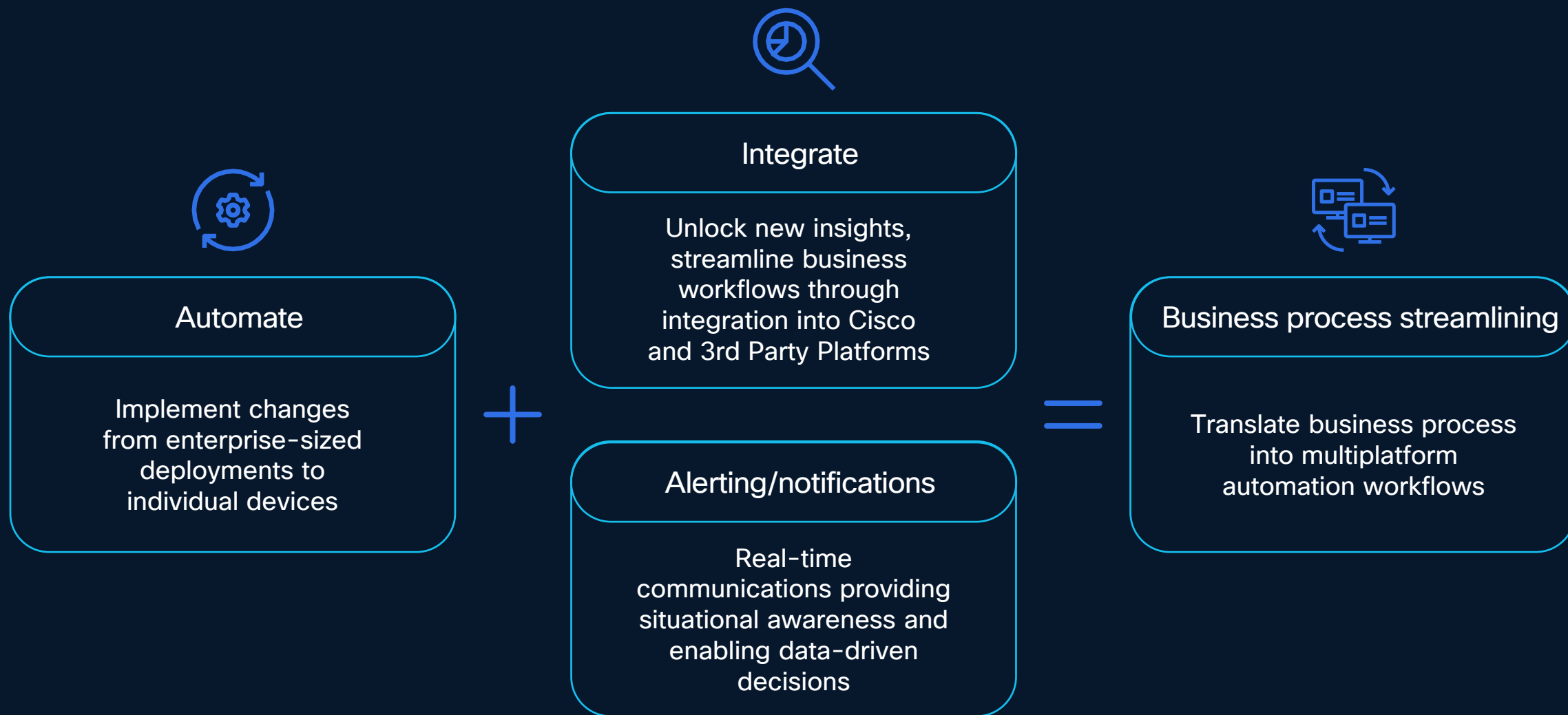
Integrate sustainable devices and systems



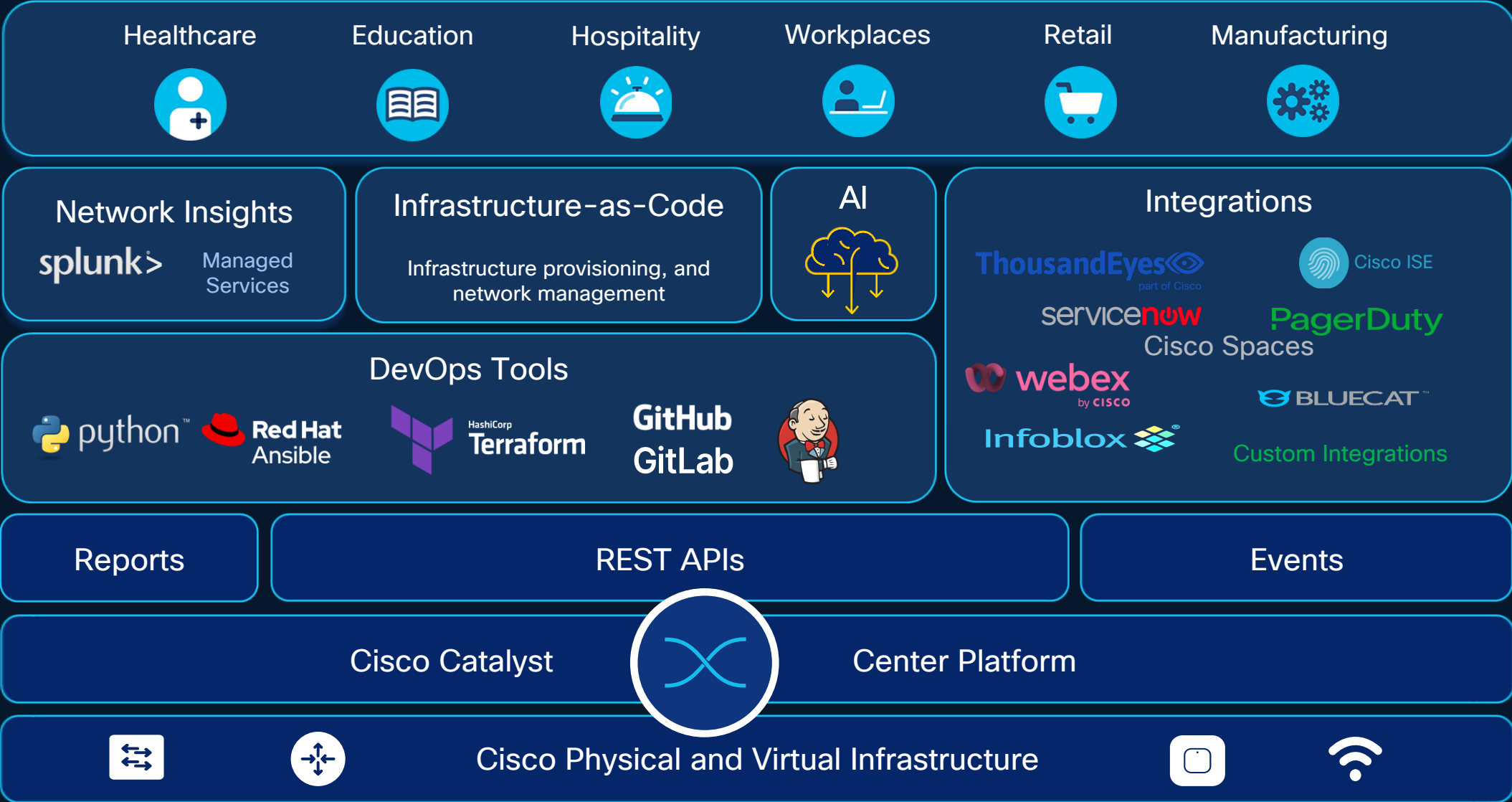
Smart

270+ real time event notifications and AI/ML capabilities

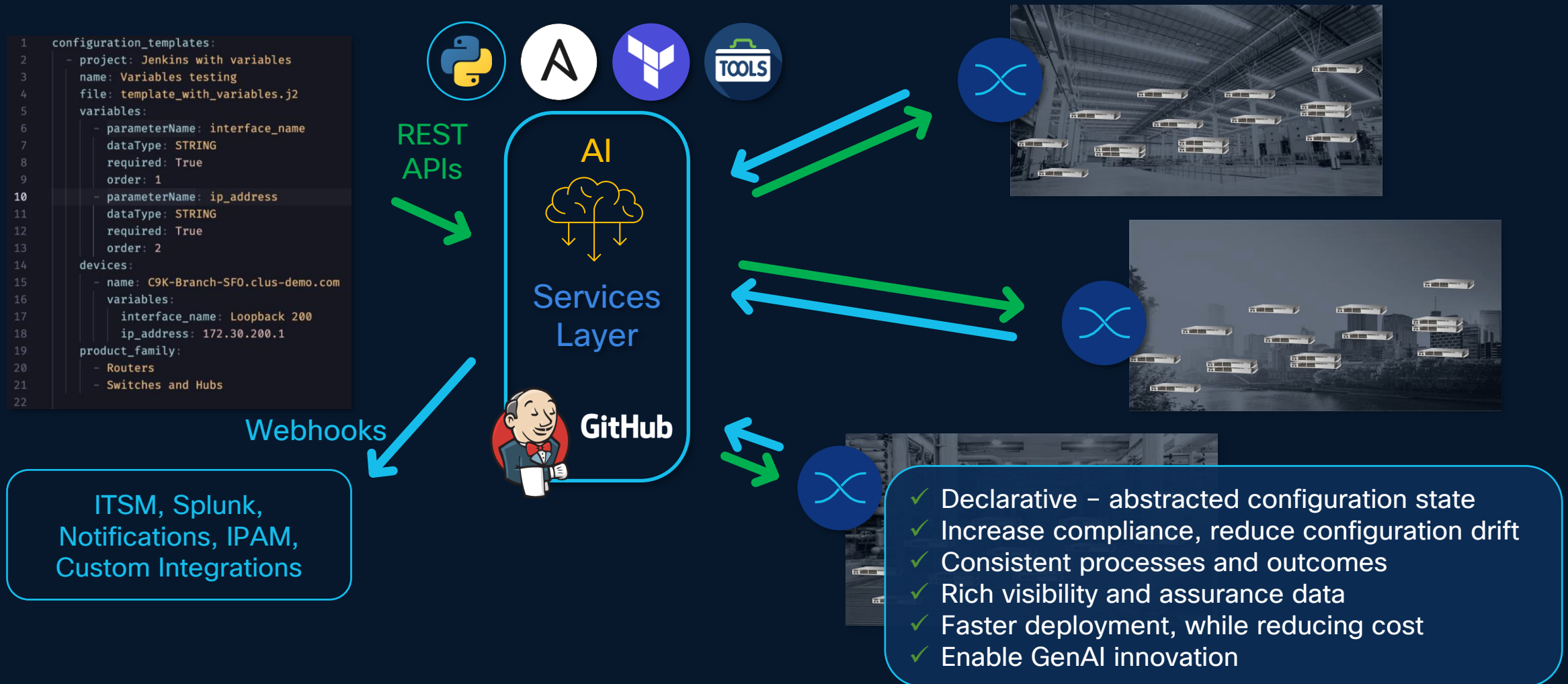
The Value of Catalyst Center Platform



Cisco Catalyst Center Platform



Multi Catalyst Center/Controller Managed Networks



Catalyst Center Platform

Event Notifications

- Assurance Issues
- AI/ML Insights
- System Health
- Integration Connectivity
- License Management
- Webhooks
- PagerDuty
- Webex
- Syslog
- SNMP

Northbound REST APIs

- Network Inventory
- Network Topology
- Network Design
- Provisioning
- SWIM, PnP
- Compliance
- Assurance
- SDA
- Templates
- RMA
- Config Archive
- Sensors

IT Ecosystem Integrations

- IT Service Management
- IP Address Management
- Reporting
- Wireless Planning
- Alerting
- Network Insights

Developer Resources

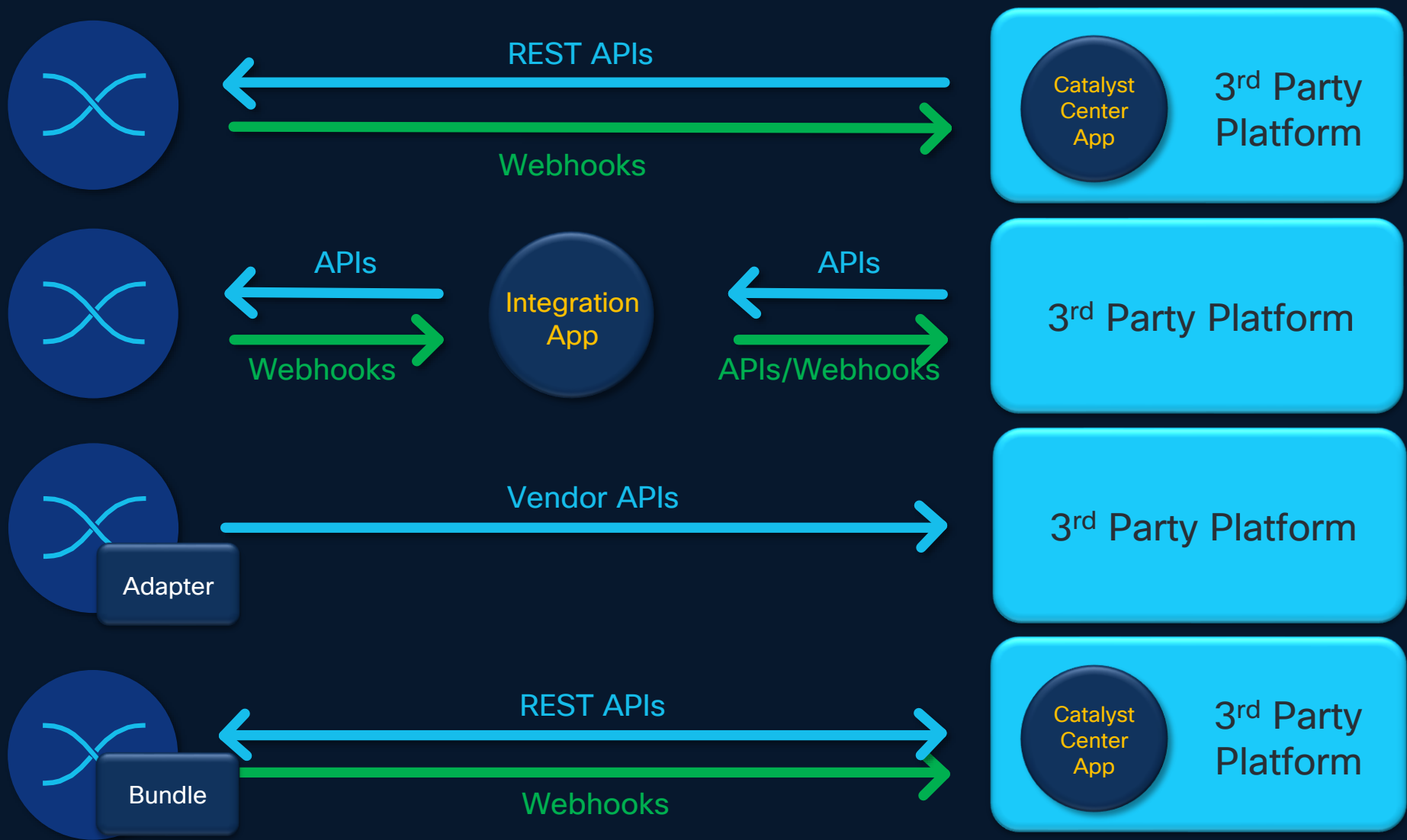
- Sample Code, Videos
- Python SDK, Ansible, Terraform
- GenAI Use Cases
- Cisco DevNet
 - Sandboxes, Learning Labs
 - Developer Guides, Community



Agenda

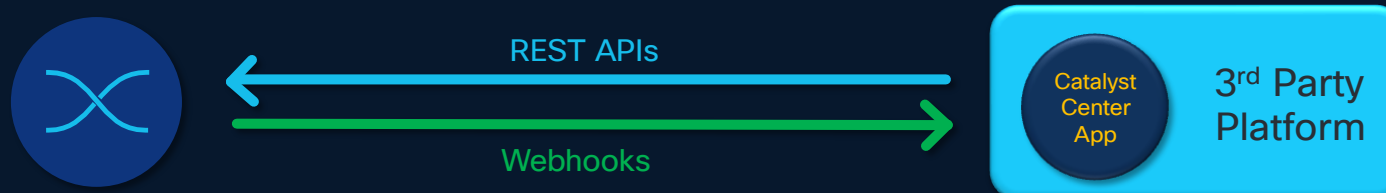
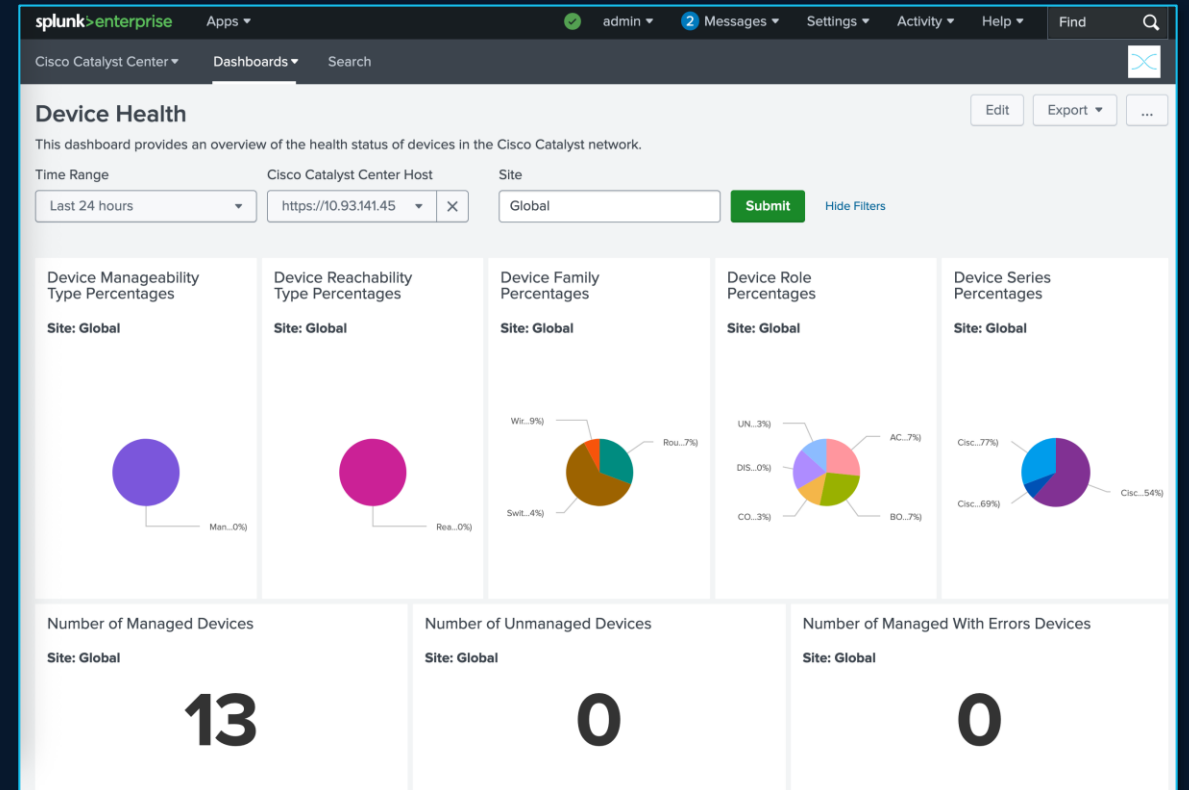
- 01 Catalyst Center Platform
- 02 Integrations
- 03 GenAI Essentials
- 04 AI Code Generation
- 05 GenAI Network Automation
- 06 Developer Resources
- 07 Summary

Catalyst Center Integration Options



Direct Integrations

- Catalyst Center notifications sent using webhooks
- App may call Catalyst Center REST APIs to:
 - Enrich the issue, client, device
 - Collect additional information: health, compliance
 - Run other automations workflows
- Log events and build dashboards
- Correlate with other platforms, data retention, trends, audits, and forward notifications



Adapter Integrations

- Catalyst Center software adapters developed for interoperability with other platforms
- The adapter will transform the event notification auth, header, data, and payload structure to match the destination
- Simplified Catalyst Center workflows to integrate with 3rd Party Platforms
- Ready to use out-of-the-box

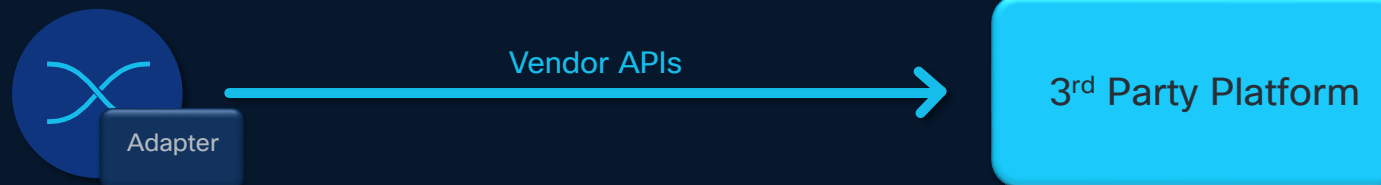
Catalyst Center Lab Notifications ★

Messages People (3) Content Meetings + Apps

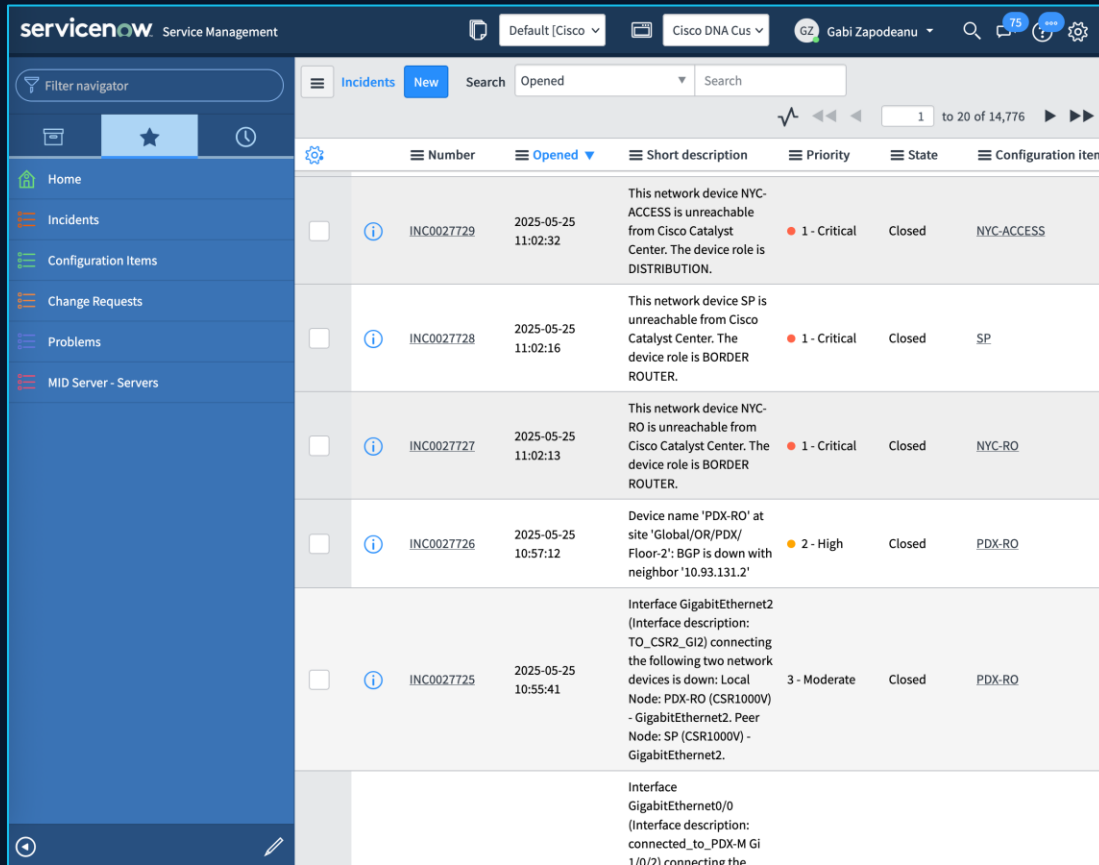
Cisco Catalyst Center Notification

Source Catalyst Center IP:	10.93.141.45
Severity:	1
Category:	ALERT
Timestamp:	2025-05-26 11:10:32
Issue Name:	Interface GigabitEthernet0/0 (Interface description: connected_to_PDX-M Gi 1/0/2) is Down on Network Device 10.93.141.18
Issue Description:	Interface GigabitEthernet0/0 (Interface description: connected_to_PDX-M Gi 1/0/2) connecting the following two network devices is down: Local Node: PDX-CORE, Peer Node: PDX-MGMT
Issue Status:	active

Cisco Catalyst Center Issue Details



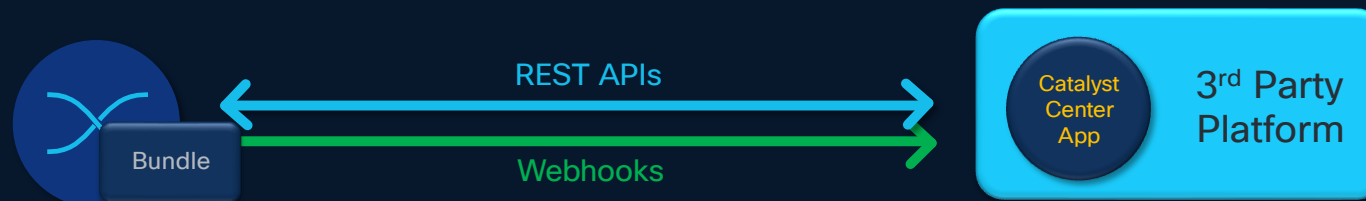
Bundle Integrations



The screenshot displays the ServiceNow 'Incidents' list. The table includes columns for Number, Opened date, Short description, Priority, State, and Configuration item. Five incidents are listed, all with a priority of 1 - Critical or 2 - High, and a state of Closed.

Number	Opened	Short description	Priority	State	Configuration item
INC0027729	2025-05-25 11:02:32	This network device NYC-ACCESS is unreachable from Cisco Catalyst Center. The device role is DISTRIBUTION.	1 - Critical	Closed	NYC-ACCESS
INC0027728	2025-05-25 11:02:16	This network device SP is unreachable from Cisco Catalyst Center. The device role is BORDER ROUTER.	1 - Critical	Closed	SP
INC0027727	2025-05-25 11:02:13	This network device NYC-RO is unreachable from Cisco Catalyst Center. The device role is BORDER ROUTER.	1 - Critical	Closed	NYC-RO
INC0027726	2025-05-25 10:57:12	Device name 'PDX-RO' at site 'Global/OR/PDX/Floor-2'; BGP is down with neighbor '10.93.131.2'	2 - High	Closed	PDX-RO
INC0027725	2025-05-25 10:55:41	Interface GigabitEthernet2 (Interface description: TO_CSR2_GI2) connecting the following two network devices is down: Local Node: PDX-RO (CSR1000V) - GigabitEthernet2. Peer Node: SP (CSR1000V) - GigabitEthernet2.	3 - Moderate	Closed	PDX-RO

- High business value bi-directional, closed-loop integrations between Catalyst Center and 3rd Party Platforms
- Catalyst Center bundles and custom Integration Apps developed for interoperability between platforms
- Complex end-to-end Catalyst Center workflows, processes and automations
- Integration App and bundle compatibility certifications required
- Ready to use out-of-the-box, and customizable



Integration Apps

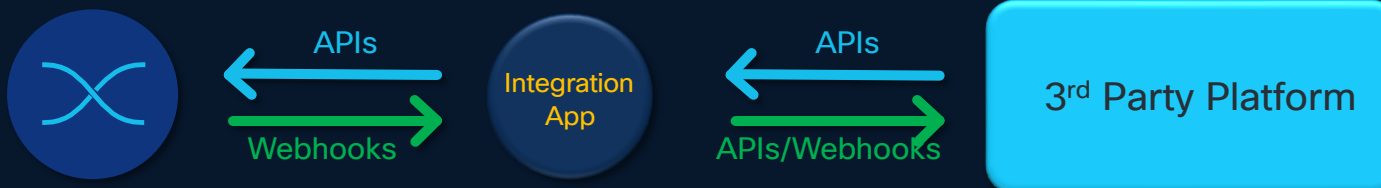
- Rich features, innovation and maximum extensibility
- Supports non-matching API architectures, auth options, data encoding and data structures
- Develop custom integrations with any platforms
- May be required as a proxy between Catalyst Center and cloud platforms
- **Enable GenAI use cases**

```
Your input: Share details about the device PDX-R0
IssuePilot: Certainly! Here is the detailed information about the device PDX-R0:

- **Hostname**: PDX-R0
- **Location**: Global/OR/PDX/Floor-2
- **Device Role**: BORDER ROUTER
- **Device Id**: 01f7cdf2-2298-42c7-bb74-dc68e3c3a051
- **Reachability**: REACHABLE
- **Health**: 8.0
- **Management IP Address**: 10.93.141.23
- **Serial Number**: 92ML86IWCBN
- **Family**: CSR1000V
- **Software**: 17.3.8a

**Compliance Status**:
- NETWORK_SETTINGS: COMPLIANT
- RUNNING_CONFIG: NON_COMPLIANT
- EOX: COMPLIANT_WARNING
- IMAGE: COMPLIANT
- PSIRT: COMPLIANT
- NETWORK_PROFILE: NON_COMPLIANT

**Connected Devices**:
- PDX-MGMT, IP address: 10.93.141.29
- PDX-RN, IP address: 10.93.141.22
- C9800-CL, IP address: 10.93.141.38
- SP, IP address: 10.93.141.24
```



Develop your Integration App

- Could be part of your Network Services Layer, running scheduled automations or triggered by APIs and Webhooks
- Hosted in your Data Center or DMZ, as a VM or container
- Cloud hosted or Software-as-a-Service
- Will likely require to make API calls to Catalyst Center and other Platforms
- Highly desirable to support your existing IaC tooling developed in Python, Ansible, or Terraform



```
INFO:root: App "Network Troubleshooting.py" run start, 2025-05-08 22:22:22
INFO:root: The Assurance issue Id received is: 9a97867f-8082-4668-b666-fd505faaeac9
INFO:root:
-----

INFO:root: The issue details
INFO:root:   Severity: HIGH
INFO:root:   Priority: P1
INFO:root:   Issue name: EIGRP_Peering
INFO:root:   Summary: EIGRP Adjacency Failed on Device "PDX-R0" Interface GigabitEthernet2
(Interface description: T0_CSR2_GI2) with Neighbor 10.93.131.2
INFO:root:   Description: EIGRP adjacency failed on device name:'PDX-R0',
interface:'GigabitEthernet2' (Interface description: T0_CSR2_GI2) at site:'Global/OR/PDX/Floor-2'
with neighbor '10.93.131.2'
INFO:root:   Timestamp: Thu May  8 15:52:34 2025
INFO:root:   Issue Details: https://10.93.141.45/dna/assurance/issueDetails?issueId=9a97867f-8082-4668-b666-fd505faaeac9
INFO:root:
-----

INFO:root: The device details
INFO:root:   Hostname: PDX-R0
INFO:root:   Location: Global/OR/PDX/Floor-2
INFO:root:   Device Role: BORDER ROUTER
INFO:root:   Device Id: 01f7cdf2-2298-42c7-bb74-dc68e3c3a051
INFO:root:   Reachability: REACHABLE
INFO:root:   Health: 10.0
INFO:root:   Management IP Address: 10.93.141.23
INFO:root:   Serial Number: 92ML86IWCBN
INFO:root:   Family: CSR1000V
INFO:root:   Software: 17.3.8a
```

Running an Automation Workflow

- Create and activate a new Python virtual environment, or a Docker container
- Clone, or pull, code from Version Control Platforms (GitHub or GitLab)
- Install or upgrade Python Libraries, example: Catalyst Center SDK
- Provide the required params
- Run the application

We do this every time we start development or run the app

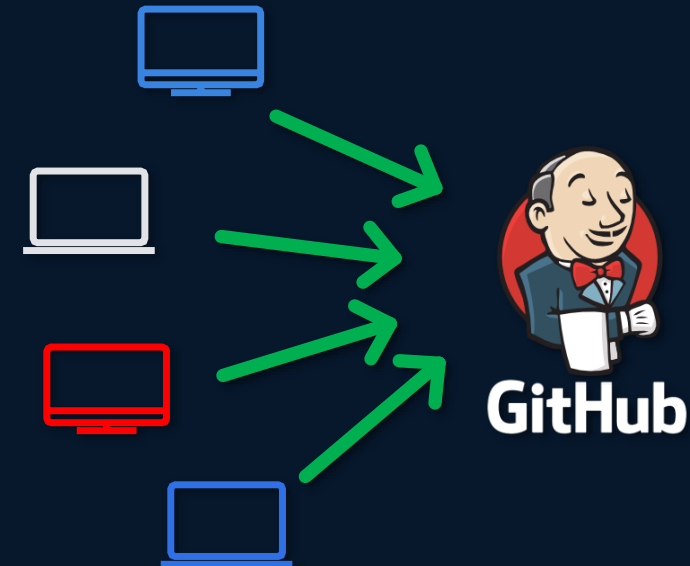


Runbook environments drift between developers, using different libraries and code versions

Pipeline Build

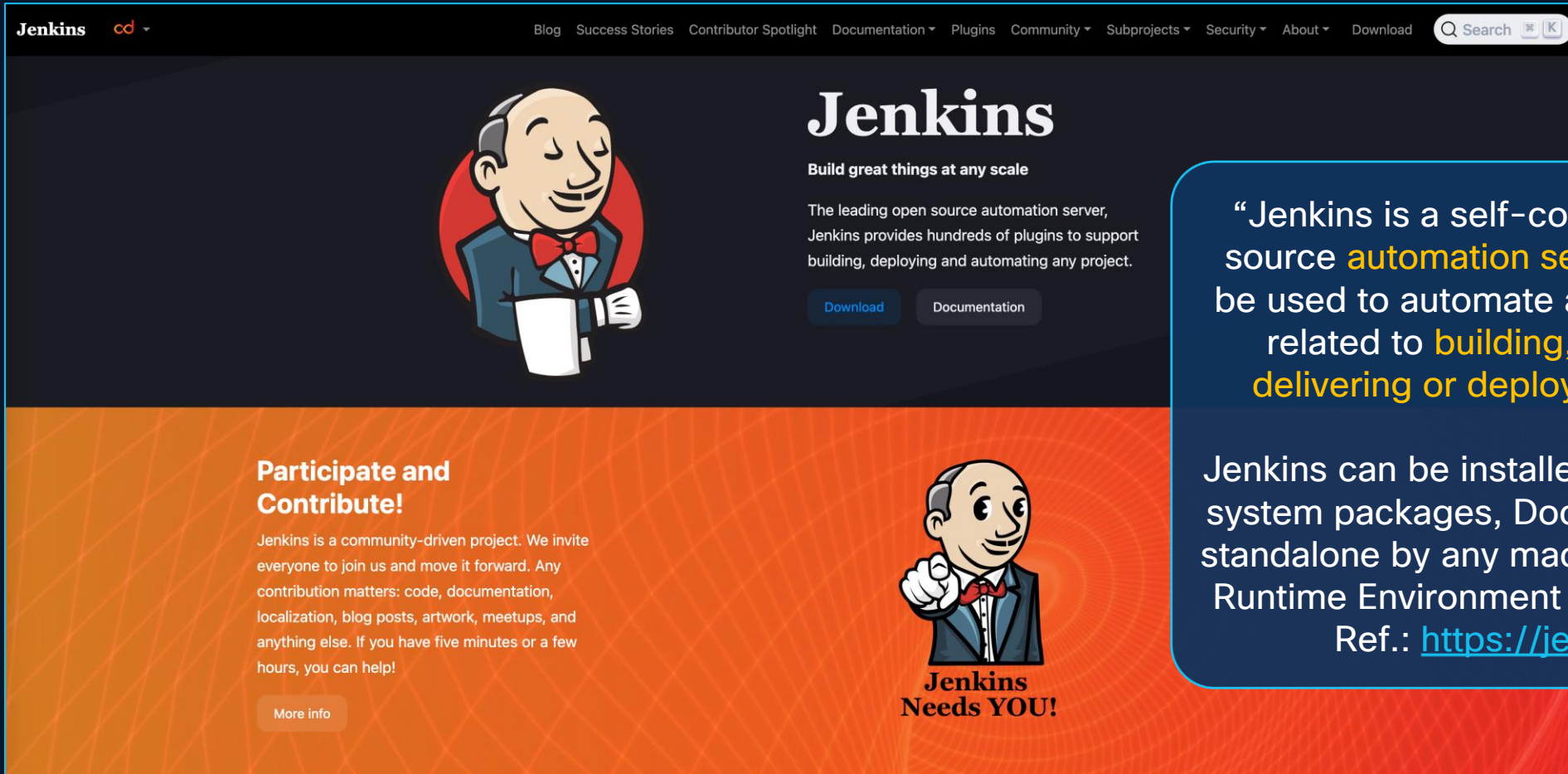
- Trigger Pipeline build manually, periodically, remotely (APIs or actions)
- Create a Docker container
- Clone, or pull, code from Version Control Platforms: GitHub or GitLab
- Install or upgrade Libraries – Catalyst Center SDK, Ansible, Terraform
- Provide the required params, or pull them from Version Control
- Run the application, while enforcing build concurrency, and time of the day policies

All steps are completely automated



Runbook environments identical, always using the latest code versions and libraries
Rich scheduling/triggering options

What is Jenkins?



The screenshot shows the Jenkins website homepage. At the top, there's a navigation bar with links like Blog, Success Stories, Contributor Spotlight, Documentation, Plugins, Community, Subprojects, Security, About, and Download. A search bar is also present. The main header features the Jenkins logo (a cartoon man in a tuxedo) and the text "Jenkins" in a large font, followed by the tagline "Build great things at any scale". Below this, a paragraph describes Jenkins as the leading open source automation server. There are buttons for "Download" and "Documentation". The lower section has an orange background with the heading "Participate and Contribute!" and text inviting community participation. Another Jenkins logo is shown with the text "Jenkins Needs YOU!".

Jenkins 

Blog Success Stories Contributor Spotlight Documentation Plugins Community Subprojects Security About Download

Jenkins

Build great things at any scale

The leading open source automation server, Jenkins provides hundreds of plugins to support building, deploying and automating any project.

[Download](#) [Documentation](#)

Participate and Contribute!

Jenkins is a community-driven project. We invite everyone to join us and move it forward. Any contribution matters: code, documentation, localization, blog posts, artwork, meetups, and anything else. If you have five minutes or a few hours, you can help!

[More info](#)



**Jenkins
Needs YOU!**

“Jenkins is a self-contained, open-source **automation server** which can be used to automate all sorts of tasks related to **building, testing, and delivering or deploying software**.

Jenkins can be installed through native system packages, Docker, or even run standalone by any machine with a Java Runtime Environment (JRE) installed.”

Ref.: <https://jenkins.io/>

Why Catalyst Center and Jenkins?



+



Jenkins is a self-contained, open-source **automation server** which may be used to:

- Support Infrastructure-as-Code
- Integrate with Version Control for validated and approved apps, intent configurations, profiles and policies
- Pipelines may include lab testing, approval, production deployment, validation, and reporting

Jenkins 

Blog Success Stories Documentation ▾ Plugins Community ▾ Subprojects ▾ Security ▾ About ▾ Download

> User Documentation Home

User Handbook

- [User Handbook Overview](#)
- [Installing Jenkins](#)
- [Using Jenkins](#)
- [Pipeline](#)
- [Blue Ocean](#)
- [Managing Jenkins](#)
- [Securing Jenkins](#)
- [System Administration](#)
- [Scaling Jenkins](#)
- [Troubleshooting Jenkins](#)
- [Glossary](#)

Tutorials

- [Guided Tour](#)
- [Jenkins Pipeline](#)
- [Using Build Tools](#)

Resources

- [Pipeline Syntax reference](#)
- [Pipeline Steps reference](#)
- [LTS Upgrade guides](#)

Jenkins User Documentation

Welcome to the Jenkins user documentation - for people wanting to use Jenkins's existing functionality and plugins.

If you want to extend the functionality of Jenkins by developing your own plugins, please refer to the [Jenkins](#) (developer documentation).

What is Jenkins?

Jenkins is a self-contained, open source automation server used to automate all sorts of tasks related to building, testing, and delivering or deploying software.

Jenkins can be installed through native system packages. Docker images are also available. All you need is a Java Runtime Environment (JRE) installed.

About this documentation

This documentation begins with a [Guided Tour](#) to help you get up and running with Jenkins and introduce you to Jenkins's main feature, Pipeline.

There are also [tutorials](#) geared to developers who want to orchestrate and automate building their project in Jenkins using Pipeline and Blue Ocean.

If you've never used Jenkins before or have limited Jenkins experience, then the Guided Tour and introductory tutorials are good places to start.

If you are looking for more detailed information about using Jenkins, please refer to the [User Handbook](#).

Includes everything you need to know:

- How to install, upgrade, configure
- How to get started
- Jenkins management, credentials and plugins
- Pipeline syntax and tutorials
- Security features

Jenkins Dashboard

 **Jenkins** / OpsView

+ New Item

 Build History

 Edit View

 Delete View

 Project Relationship

 Check File Fingerprint

Build Queue

No builds in the queue.

Build Executor Status 0/3

All LabView OpsView +

S	W	Name	Last Success ↑	Last Failure	Last Duration
✓	☀	Trigger Assurance Issues	10 hr #1436	N/A	11 min ▶
✓	☀	Device Inventory	16 hr #2335	N/A	1 min 3 sec ▶
✓	☀	GitHub Sync	17 hr #875	N/A	29 sec ▶
✓	☀	Catalyst Center State Sync	18 hr #588	N/A	58 sec ▶
✓	☁	CLI Templates Deployment	1 day 22 hr #24	2 days 20 hr #22	2 min 50 sec ▶
✗	☁⚡	Intent Device Compliance	4 days 18 hr #547	18 hr #551	42 sec ▶
✓	☀	Provision Device	20 days #94	N/A	35 sec ▶
✓	☀	Add Device	20 days #15	N/A	39 sec ▶
✓	☁	Network Troubleshooting	26 days #192	26 days #190	1 min 21 sec ▶
🕒	☁	Software Distribution	3 mo 22 days #21	26 days #22	52 sec ▶

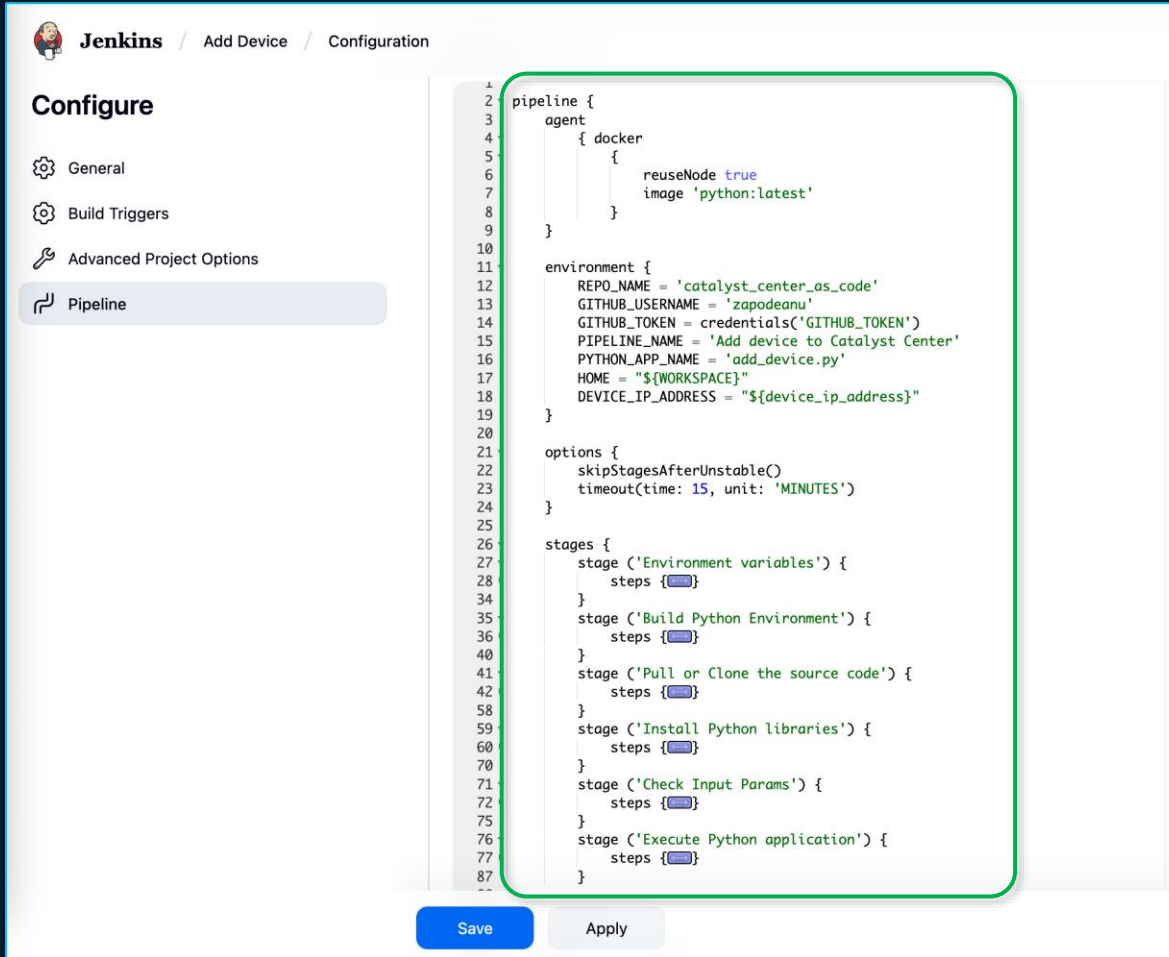
- Jenkins preferences
- Group pipelines in views
 - Build queue and execution status
 - Pipeline stats – health, number of runs, success and failure, ...

Jenkins Server – Enable Remote API Access

- Add new API Auth token
- Configure pipeline to use input params or input file
- Configure the build trigger for remote access API
- **REST API URL call to Jenkins:**
`/Jenkins_server/job/pipeline_name/buildWithParameters/{parameters}`

The image shows two overlapping screenshots of the Jenkins web interface. The top screenshot displays the 'Security' configuration page, which includes a sidebar with navigation links (Status, Builds, Favorites, My Views, Account, Appearance, Preferences, Security, Experiments) and a main content area titled 'Security'. In the 'Security' section, there is a box for 'API Token' containing two tokens, 'AI_TOKEN' and 'API_TOKEN', both marked as 'Created over a year ago'. A green box highlights this section. The bottom screenshot shows the 'Build Triggers' configuration page. It features a list of checkboxes for various build triggers. The 'Trigger builds remotely (e.g., from scripts)' checkbox is checked and highlighted with a green box. Below this checkbox, there is a field for 'Authentication Token' with the value 'AI_TOKEN'. A description below the token field provides the REST API URL: 'JENKINS_URL/job/Add%20Device/build?token=AI_TOKEN or /buildWithParameters?token=AI_TOKEN'. A 'String Parameter' dialog box is also visible, showing a parameter named 'device_ip_address' with a default value and a description 'The device management IP address'. The dialog box has a green border and a close button.

Jenkins Pipeline – Add Device



The screenshot shows the Jenkins 'Add Device' configuration page. On the left, the 'Configure' sidebar has tabs for General, Build Triggers, Advanced Project Options, and Pipeline (which is selected). The main area displays a Jenkinsfile code editor with a green border. The code defines a pipeline with an agent, environment variables, options, and several stages. At the bottom, there are 'Save' and 'Apply' buttons.

```
1 pipeline {
2   agent
3     { docker
4       {
5         reuseNode true
6         image 'python:latest'
7       }
8     }
9
10  environment {
11    REPO_NAME = 'catalyst_center_as_code'
12    GITHUB_USERNAME = 'zapodeanu'
13    GITHUB_TOKEN = credentials('GITHUB_TOKEN')
14    PIPELINE_NAME = 'Add device to Catalyst Center'
15    PYTHON_APP_NAME = 'add_device.py'
16    HOME = "${WORKSPACE}"
17    DEVICE_IP_ADDRESS = "${device_ip_address}"
18  }
19
20  options {
21    skipStagesAfterUnstable()
22    timeout(time: 15, unit: 'MINUTES')
23  }
24
25  stages {
26    stage ('Environment variables') {
27      steps {
28        ...
29      }
30    }
31    stage ('Build Python Environment') {
32      steps {
33        ...
34      }
35    }
36    stage ('Pull or Clone the source code') {
37      steps {
38        ...
39      }
40    }
41    stage ('Install Python libraries') {
42      steps {
43        ...
44      }
45    }
46    stage ('Check Input Params') {
47      steps {
48        ...
49      }
50    }
51    stage ('Execute Python application') {
52      steps {
53        ...
54      }
55    }
56  }
57 }
```

- Jenkinsfile is a text file representing the definition of a pipeline. It includes multiple stages and steps.
- Steps may be constructed using:
 - Shell commands
 - Run code developed using the Python SDK, Ansible or Terraform
 - Jenkins Plugins
- Environment includes all params required by the pipeline

Details: Cisco Live US 2023– Jenkins Automations for Cisco DNA Center – DEVNET-2151

Add Device – Pipeline Parameters

☒ This project is parameterized ?

☰ **String Parameter** ?

Name ?

device_ip_address

Default Value ?

Description ?

The device management IP address

Plain text [Preview](#)

☒ Trim the string ?

- Integration Apps execution may require parameters
- Params could be entered manually, at the time of the run, retrieved with the application files, or included with API call
- Adding a device to Catalyst Center inventory requires the management IP address of the device

Add Device- Pipeline Triggers

Pipelines build could be:

- Manual, or after another pipeline completes
- **Scheduled**
- GitHub Integrations
- Triggered by an API call to Jenkins. Parameters to be provided when using the option “/buildWithParameters”

Build Triggers

- ☐ Build after other projects are built ?
- ☐ Build periodically ?
- ☐ Build whenever a SNAPSHOT dependency is built ?
- ☐ GitHub Branches
- ☐ GitHub Pull Requests ?
- ☐ GitHub hook trigger for GITScm polling ?
- ☐ Poll SCM ?
- ☐ Quiet period ?

- ☒ Trigger builds remotely (e.g., from scripts) ?

Authentication Token

AI_TOKEN

Use the following URL to trigger build remotely: JENKINS_URL/job/Add%20Device/build?token=TOKEN_NAME or /buildWithParameters?token=TOKEN_NAME

Optionally append &cause=Cause+Text to provide text that will be included in the recorded build cause.

Demo

Add Device Pipeline

Camtasia 2024

File Edit Modify Text View Export Window Help

Home - Cisco Catalyst Center

Dashboard [Jenkins]

https://10.93.141.45/dna/home120%

Cisco Systems, IncAPIsToolsLAB_AccessCiscoLiveMeraki Dashboard ...SFEH-NAS - Synolo...CDETSDNACaaPSharePointAPIs and Owners - ...JIRA Engineering SJ...Catalyst Center API ...

☰

Catalyst Center

☆

🔍

🟢

?

🔔

|

Gabi

▼

Welcome to Catalyst Center!

Explore

Assurance Summary

Health ⓘ

Healthy as of Feb 5, 2025 6:11 PM

77%

Network Devices

--%

Wireless Clients

89%

Wired Clients

View Details

Critical Issues

Last 24 Hours

1

P1

0

P2

View Details

Trends and Insights

Last 30 Days

0

AP Performance Advisories

0

Trend Deviations

View Details

Network Snapshot

Sites

As of Feb 5, 2025 6:11 PM

16

DNS Servers : 1

NTP Servers : 1

Add Sites

Network Devices

As of Feb 5, 2025 6:11 PM

12

Unclaimed: 3

Unprovisioned: 1

Unreachable: 0

Find New Devices

Application QoS Policies

As of Feb 5, 2025 6:11 PM

0

Successful Deploys: 0

Errored Deploys: 0

Stale Policies: 0

Add New Policy

Agenda

- 01 Catalyst Center Platform
- 02 Integrations
- 03 **GenAI Essentials**
- 04 AI Code Generation
- 05 GenAI Network Automation
- 06 Developer Resources
- 07 Summary

AI, ML, LLM, Function Calling, GenAI, RAG, ... ?

- **Artificial Intelligence (AI)** – use of data and algorithms to allow systems to perform tasks similar with human intelligence
 - **Machine Learning (ML)** – AI subset, focused on performing steps without being explicitly programmed. Learn from data patterns and insights to create recommendations
 - **Large Language Model (LLM)** – neural network model designed to understand and generate human language
- **Function calling** – Use LLM models to interface with external platforms (usually with API calls), to identify and perform tasks or retrieve data
 - **Generative AI (GenAI)** – Use patterns in existing data to create new content. This content could be text, numbers or multimedia
 - **Retrieval-Augmented Generation (RAG)** Retrieves matches in provided dataset and uses NLPs to create high-confidence content

Large Language Models (LLMs)

LLMs:

- **GPT (OpenAI)**
- Gemini (Google)
- Llama 3 (Meta)
- Ollama (framework)
- Claude (Anthropic)
- ...

Consider cost and data privacy

- Large Language Models are AI Models designed to understand and generate human language
- Trained on massive data sets (billions of text chunks)
- Understand context and are general-purpose
- Will answer queries, based on the similarity search results, selecting only the semantically relevant chunks
- Provide other features, like **function calling**
- Support **AI code generation** for network automation and Infrastructure-as-Code, such as writing Ansible Playbooks

Claude

<https://claude.ai>



Features

Pricing

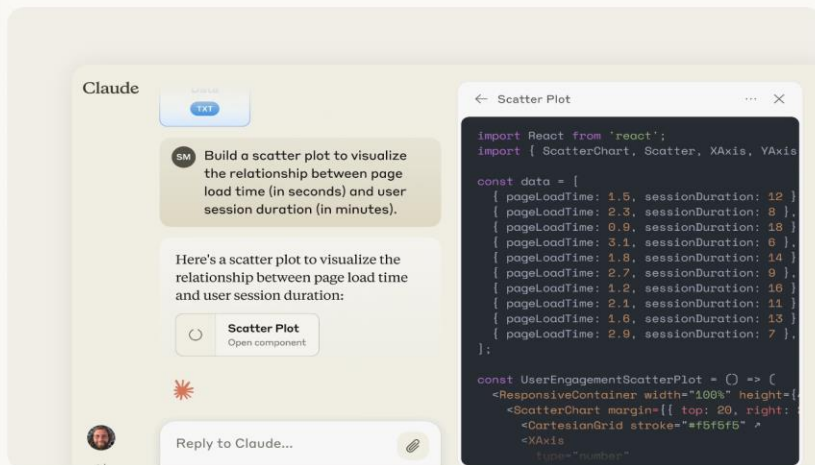
FAQ

Sign up

Sign in

Meet Claude

Claude is a next generation AI assistant built by Anthropic and trained to be safe, accurate, and secure to help you do your best work.



Create with Claude

Draft and iterate on websites, graphics, documents, and code alongside your chat with Artifacts.

Bring your knowledge

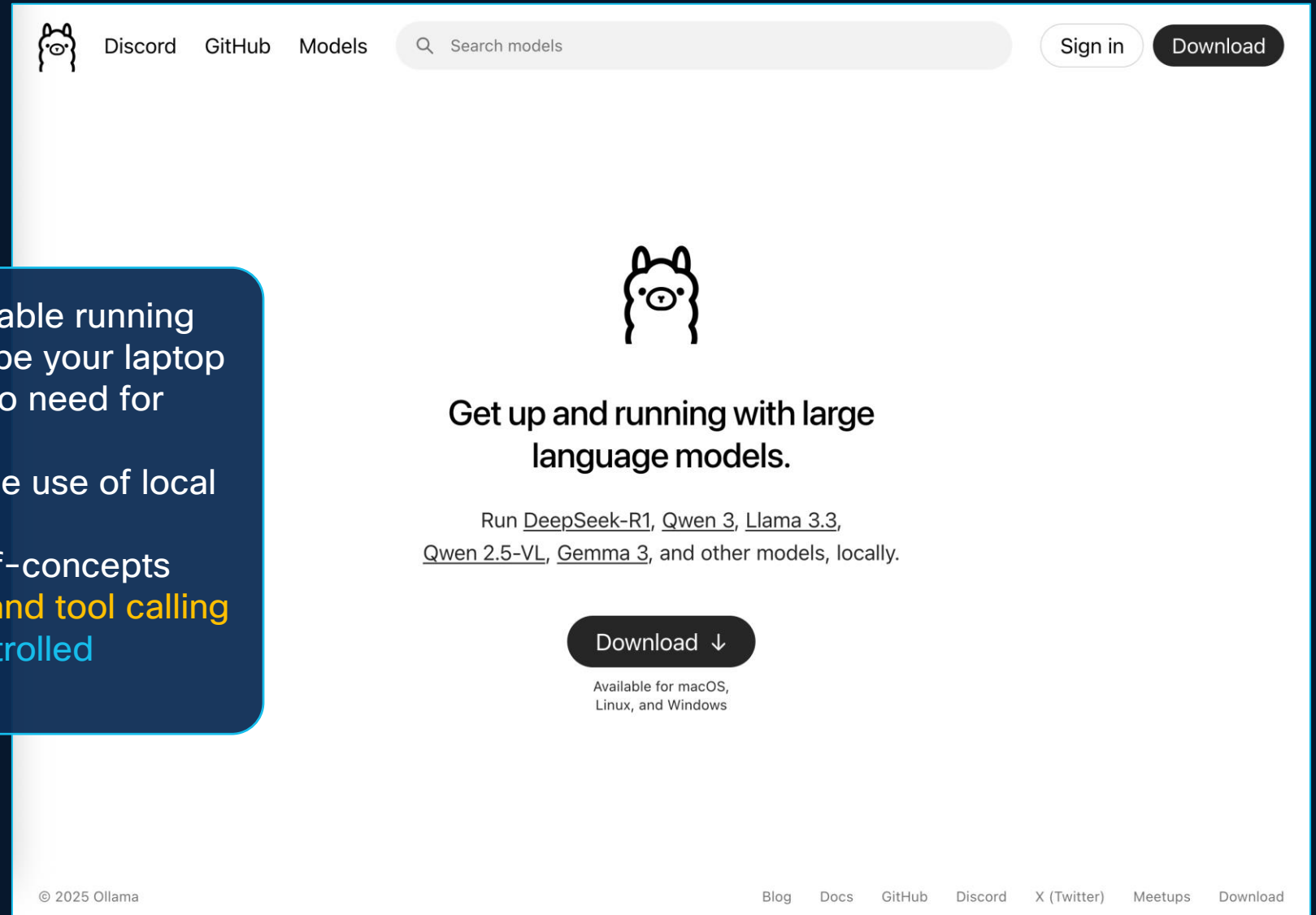
Share and collaborate with your team

- Very versatile, general-purpose LLM model, excellent for networking use cases
- High quality conversation, maybe less answers variation
- Could be a little slower, and more expensive
- Supports AI Code generation and tool calling
- Be aware of data privacy when exposing private data
- Consider cost when used at scale

Llama3 Model

<https://ollama.com/>

- Open-source framework to enable running LLMs on your server, and maybe your laptop
- Very easy to start, learn, and no need for licensing fees
- Ollama framework simplifies the use of local installed Llama3 model
- GPUs not required for proof-of-concepts
- Supports AI Code generation and tool calling
- Stronger data privacy and controlled environment for your data



The screenshot shows the Ollama website homepage. At the top, there is a navigation bar with links for Discord, GitHub, and Models, along with a search bar labeled 'Search models'. On the right side of the navigation bar are 'Sign in' and 'Download' buttons. The main content area features the Ollama logo (a stylized llama face) and the text 'Get up and running with large language models.' Below this, it says 'Run DeepSeek-R1, Qwen 3, Llama 3.3, Qwen 2.5-VL, Gemma 3, and other models, locally.' A prominent 'Download' button with a downward arrow is displayed, with a note below it stating 'Available for macOS, Linux, and Windows'. The footer contains copyright information '© 2025 Ollama' and a series of links: Blog, Docs, GitHub, Discord, X (Twitter), Meetups, and Download.

OpenAI Platform

<https://platform.openai.com/>

- Very easy to use and learn, massive number of examples available, multiple models, cloud based
- Versatile, very good results for networking cases, without any specific training
- Supports AI Code generation and tool calling
- Be aware of data privacy when exposing private data
- Cost may be a challenge when used at scale

OpenAI developer platform

Developer quickstart

Make your first API request in minutes. Learn the basics of the OpenAI platform.

🕒 5 min

javascript ↕

```
1 import OpenAI from "openai";
2 const client = new OpenAI();
3
4 const response = await client.responses.create({
5   model: "gpt-4.1",
6   input: "Write a one-sentence bedtime story about a unicorn.",
7 });
8
9 console.log(response.output_text);
```



Browse models

View all

GPT-4.1

o4-mini

o3

GPT-4.1

Flagship GPT model for complex tasks

o4-mini

Faster, more affordable reasoning model

o3

Our most powerful reasoning model

Agenda

- 01 Catalyst Center Platform
- 02 Integrations
- 03 GenAI Essentials
- 04 AI Code Generation**
- 05 GenAI Network Automation
- 06 Developer Resources
- 07 Summary

Infrastructure-as-Code (IaC) Solutions

- Most customers use Python SDK
- Ansible Collection and Terraform Provider are very well adopted
- Customers develop network-as-code workflows in Go SDK, JavaScript, and other programming languages
- Pipelines and Runbook Platforms are very common

Ansible Collection - cisco.catalystcenter

Ansible Modules for CATALYST Center

The catalystcenter-ansible project provides an Ansible collection for managing and automating your Cisco CATALYST Center environment. It consists of a set of modules and roles for performing tasks related to CATALYST Center.

This collection has been tested and supports Cisco CATALYST Center 2.3.7.6.

Other versions of this collection have support for previous Cisco CATALYST Center versions. The recommended versions are listed below on the [Compatibility matrix](#).

Compatibility matrix

The following table shows the supported versions.

Cisco CATALYST Center version	Ansible "cisco.catalystcenter" version	Python "catalystcentersdk" version
2.3.7.6	1.0.0	2.3.7.6.2
2.3.7.7	^2.1.2	^2.3.7.9.4
2.3.7.9	^2.1.3	^2.3.7.9.4

If your Ansible collection is older please consider updating it first.

Will AI help with new IaC use cases development?

Why AI Code Generation for Catalyst Center



AI may be able to:

- Suggest recommendations and search documentation
- Accelerate code development for your IaC workflows
- Multi programming language support
- Know available APIs, and able to use Swagger files
- Understand Catalyst Center order of operations

Always verify the recommended code, check the order of operations and test the recommendations before using them.

ChatGPT and Catalyst Center API Docs

ChatGPT o4-mini-high ▾

What APIs do I need to get a list of devices managed by Catalyst Center?

Thought for 15 seconds >

To retrieve the inventory of devices managed by Catalyst Center, you will use the Intent API's "network device" endpoints. In summary:

1. Authenticate
 - Endpoint:
 - swift
 - `POST /dna/system/api/v1/auth/token`
 - Description: Obtain an authentication token (X-Auth-Token) to include in subsequent requests. [developer.cisco.com](https://developer.cisco.com/docs/catalyst-center-api/1.10/#/api-authentication)
2. Get the full list of devices
 - Endpoint:
 - bash
 - `GET /dna/intent/api/v1/network-device`
 - Description: Returns an array of all devices currently managed (discovered) by Catalyst Center. You can optionally supply query parameters (e.g. `hostname`, `macAddress`, `platformId`).

- I asked for API docs, answer includes docs and sample code
- The example provided is using the requests library

ChatGPT o4-mini-high ▾

Example Workflow (Python with `requests`):

```
python
import requests
from requests.auth import HTTPBasicAuth

BASE_URL = "https://<CATALYST_CENTER_IP>"
AUTH_URL = "/dna/system/api/v1/auth/token"
DEVICES_URL = "/dna/intent/api/v1/network-device"

# 1. Authenticate
resp = requests.post(
    BASE_URL + AUTH_URL,
    auth=HTTPBasicAuth("<username>", "<password>"),
    verify=False
)
token = resp.json()["Token"]
headers = {
    "X-Auth-Token": token,
    "Content-Type": "application/json"
}

# 2. Get full device list
resp = requests.get(BASE_URL + DEVICES_URL, headers=headers, verify=False)
devices = resp.json()["response"]
for device in devices:
    print(device["id"], device["hostname"], device["managementIpAddress"])

• Replace <CATALYST_CENTER_IP>, <username>, <password> with your own values.
```

Using CatalystCenterSDK

ChatGPT o4-mini-high ▾ Share GA PLUS

Can you rewrite this code using the CatalystCenterSDK?

Thought for 43 seconds >

Below is the same "list all devices" workflow rewritten to use the **CatalystCenterSDK** (instead of `dnacentersdk`). In this example, we assume you have Catalyst Center version 2.3.7.6 (adjust the `version` string as needed).

python Copy Edit

```
from catalystcentersdk import api

# 1. Create a CatalystCenterAPI client
# - `version` should match your Catalyst Center release (e.g., "2.3.7.6").
# - `verify=False` skips TLS verification (set to True in production).
catalyst = api.CatalystCenterAPI(
    username="admin",
    password="password",
    base_url="https://<CATALYST_CENTER_IP>",
    version="2.3.7.6",
    verify=False
) :contentReference[oaicite:0]{index=0}

# 2. Retrieve the full device inventory
# - `get_device_list()` returns a native Python object whose `.response`
#   attribute is a list of device records. Each record has attributes
#   like .hostname, .managementIPAddress, .platformId, etc.
devices = catalyst.devices.get_device_list() :contentReference[oaicite:1]{index=1}
```

ChatGPT o4-mini-high ▾ Share GA PLUS

attribute is a list of device records. Each record has attributes like .hostname, .managementIPAddress, .platformId, etc.
devices = catalyst.devices.get_device_list() :contentReference[oaicite:1]{index=1}

3. Iterate and print basic details
for dev in devices.response:
 print(f"{dev.hostname} | {dev.managementIPAddress} | {dev.platformId}")

Notes:

- The call to `catalyst.devices.get_device_list()` accepts the same query-parameter filters as the REST endpoint (e.g., `family`, `hostname`, `managementIPAddress`, `platformId`, etc.). In the snippet above, no filters are applied, so you'll get every node in Inventory. [github.com](#)
- Each item in `devices.response` is a Python object—access its properties directly (e.g., `dev.hostname`, `dev.upTime`, `dev.serialNumber`, etc.). [github.com](#)
- If you only need a count, you can call:

python Copy Edit

```
count = catalyst.devices.get_device_count()
print(f"Total devices: {count.response}")
```

(mirrors `/dna/intent/api/v1/network-device/count`) [developer.cisco.com](#)

Feel free to adjust the `version` parameter (e.g., "2.3.7.9" or whichever release you're running) and any filter kwargs to narrow down the results.

Copy Like Share Feedback Help Sources

Interesting the reference to 2.3.7.9 version, from a model with a cutoff in 2024

Python Code Generation – Model “04-mini-high”

ChatGPT o4-mini-high Temporary Chat

Can you write a Python script that deploys a CLI template to a device managed by Catalyst Center?

Thought for 17 seconds

- I think the best course of action here is to search for resources on deploying the Catalyst Center API via CLI with Python templates. I'll start with a search query like "Cisco Catalyst Center REST API CLI template Python" to find relevant developer docs. This should give me the most accurate, up-to-date information to guide my response. Once I get the results, I'll review them to ensure I'm providing the best possible advice.
- The user is asking for a Python script to deploy a CLI template to a device managed by Catalyst Center, presumably using its API. Cisco Catalyst Center is a cloud-based management tool for Cisco devices, and it likely offers APIs for automation. To help, I may need to look up API endpoints and authentication details, as these would be crucial for writing the script. I'll stick with focusing on the Catalyst Center specifically, even though using Cisco's DNA Center might offer some similarities. Let me look into the API!

Searched the web

Q Cisco Catalyst Center REST API deploy CLI template Python

[developer.cisco.com](#) [community.cisco.com](#) [www.mteke.com](#) [5 more](#)

- I'll be looking for documentation on the Catalyst Center API — specifically how to deploy a CLI template using Python. "Cisco Catalyst Center REST API CLI template Python" seems like a good search. The top result likely refers to the API path for deploying templates. Now, I need to figure out the authentication details and the specific API endpoints.

Searched the web

Q Cisco Catalyst Center API Python authentication

- To write the Python script for deploying a CLI template to a device managed by Catalyst Center, I need to find the right API documentation. A key resource seems to be the "Deploy Template - Cisco Catalyst Center API". I also need to figure out the authentication method, likely OAuth2 or basic auth, similar to Cisco's DNA Center. After that, I'll focus on getting the auth token and identifying the template ID, either provided by the user or input into the script.
- Since the request is general, I'll write a parameterized Python script to handle base URL, credentials, template IDs, and device identifiers. It'll include functions like `get_token()` and `deploy_template()` using Python's `requests` module. The script will also support templated device deployment—single or composite. I'll send a POST request to deploy the template with necessary details (like device UUID or hostname). I'm still considering how to handle the `targetType`, which seems implied but not explicitly stated in the API docs.

Done

Below is a self-contained Python script that:

1. Authenticates to Catalyst Center via the token-based API (using Basic Auth).
2. Uses that token to call the "Deploy Template" endpoint and push a CLI template to a managed device.

Prerequisites

- "Python 3.x"
- "The `requests` library (install with `pip install requests`)"
- "The IP or hostname of your Catalyst Center cluster, plus a valid username/password."
- "A pre-created CLI template in Catalyst Center (you'll need its UUID)."
- "The UUID of the target device (as shown in Catalyst Center's inventory)."

The ask is to write Python code to deploy templates to devices
Pre-requisites are stated, and interesting to see the AI process writing the code

Python Code Generation – continued

ChatGPT o4-mini-high

Temporary Chat

python

Copy Edit

```
#!/usr/bin/env python3
import requests
from requests.auth import HTTPBasicAuth
import json
import urllib3

# -----
# CONFIGURATION SECTION
# -----
# Replace with your Catalyst Center IP
BASE_URL = "https://<CATALYST_CENTER_IP>"
USERNAME = "<YOUR_USERNAME>"
PASSWORD = "<YOUR_PASSWORD>"

# The UUID of the CLI template
TEMPLATE_ID = "<TEMPLATE_UUID>"

# The UUID of the target device
DEVICE_UUID = "<DEVICE_UUID>"

# If your template has variable parameters
TEMPLATE_PARAMS = {
    # e.g. "hostname": "Switch-1",
    # "domain_name": "example.com"
}

# If you want to force-push even if already deployed
FORCE_PUSH = False
```

```
def deploy_cli_template(token, template_id, device_uuid, params):
    """
    Call the 'Deploy Template' API to push a CLI template to a device.
    """
    url = f"{BASE_URL}/dna/intent/api/v1/template-programmer/template/deploy"
    headers = {
        "Content-Type": "application/json",
        "X-Auth-Token": token
    }

    payload = {
        "forcePushTemplate": force_push,
        "isComposite": False,
        "mainTemplateId": template_id,
        "targetInfo": [
            {
                "id": device_id,
                "params": params or {}
            }
        ]
    }

    response = requests.post(url, headers=headers, json=payload, verify=False)
```

```
def main():
    print("Authenticating to Catalyst Center...")
    token = get_token()
    print("Token obtained.")

    print(f"Deploying template {TEMPLATE_ID} to device {DEVICE_UUID}")
    result = deploy_cli_template(token=token,
                                  template_id=TEMPLATE_ID,
                                  device_id=DEVICE_UUID,
                                  params=TEMPLATE_PARAMS,
                                  force_push=FORCE_PUSH)

    # Pretty-print the JSON result
    print(json.dumps(result, indent=2))

if __name__ == "__main__":
    main()
```

How it works

1. Authentication

The `get_token()` function POSTs to

bash

Copy Edit

```
https://<CATALYST_CENTER_IP>/dna/system/api/v1/auth/token
```

using HTTP Basic Auth (`USERNAME:PASSWORD`) and receives a JSON payload containing `"Token"`.

Include that token as an `X-Auth-Token` header in subsequent calls. [developer.cisco.com](https://developer.cisco.com/docs/developer-center/1-10/authentication/)

2. Deploying the CLI Template

The `deploy_cli_template()` function issues a POST to:

ruby

Copy Edit

```
https://<CATALYST_CENTER_IP>/dna/intent/api/v1/template-programmer/template/deploy
```

with a payload of the form:

json

Copy Edit

```
{
  "forcePushTemplate": false,
  "isComposite": false,
  "mainTemplateId": "<TEMPLATE_UUID>",
  "targetInfo": [
    {
      "id": "<DEVICE_UUID>",
      "params": {
        /* any template variables */
      }
    }
  ]
}
```

Provides Python Code, mostly accurate

Python Code Generation (continued)

Can you rewrite using the Catalyst Center SDK?

Using Catalyst Center SDK version 2.3.7.9,
released in 2025

Thought for 33 seconds >

Below is a Python script that uses the **Catalyst Center** template to a managed device. It follows the same workflow as the previous script, but leverages the SDK's `configuration_templates.deploy` method.

Prerequisites

1. Install the SDK

```
bash
```

```
pip install catalystcenter
```

This package provides the Catalyst Center Configuration Template API client.

2. Gather the following

- **BASE_URL:** Your Catalyst Center base URL
- **USERNAME/PASSWORD:** Your Catalyst Center credentials
- **VERSION:** The Catalyst Center SDK version (e.g., 2.3.7.9)
- **TEMPLATE_ID:** The ID of the template to deploy
- **DEVICE_ID:** The ID of the managed device
- **TEMPLATE_PARAMS:** The parameters for the template (e.g., variables)

How It Works

1. Initialize the SDK "client"

```
python

from catalystcentersdk import api

catalyst = api.CatalystCenterClient(
    username=USERNAME,
    password=PASSWORD,
    base_url=BASE_URL,
    version=VERSION,
    verify=False # Change to True if you have a valid certificate
)
```

This establishes a connection to the Catalyst Center API (using the Token for you) and exposes the `configuration_templates` module.

2. Call the "deploy template" method

```
python

result = catalyst.configuration_templates.deploy_template_v2(
    forcePushTemplate=False, # Set True to override any pending changes
    isComposite=False,      # False for a single (non-composite) template
    mainTemplateId=TEMPLATE_ID,
    targetInfo=[{
        "id": DEVICE_ID,
        "type": "MANAGED_DEVICE_UUID",
        "params": TEMPLATE_PARAMS
    }]
)
```

Under the covers, this sends a `POST` to `/dna/intent/api/v1/template-programmer/template/deploy` with the same JSON structure we built by hand previously. The `deploy_template_v2` call returns a SDK "response" object, whose `.response` field contains the task ID you can poll to track deployment status. mteke.com

Code Explanation

Explanation of Key Points

- `CatalystCenterAPI`
Creates a connection object that handles token retrieval and exposes all Intent API domains as Python attributes (`devices`, `configuration_templates`, `task`, etc.). [github.com](#) [github.com](#)
- `deploy_template_v2(...)`
Corresponds to the "Deploy Template" Intent API (`POST /dna/intent/api/v1/template-programmer/template/deploy`).
 - `forcePushTemplate` (boolean): set to `False` to avoid overwriting in-flight changes, or `True` to force-push.
 - `isComposite` (boolean): `False` for a single CLI template; set `True` only when you're deploying a composite template that bundles multiple member templates.
 - `mainTemplateId` (string): the UUID of the versioned template you want to push.
 - `targetInfo` (list): one entry per device. Each entry is a dict with:
 - `"id"`: the device's UUID (i.e. `DEVICE_ID`)
 - `"type"`: `"MANAGED_DEVICE_UUID"` (tells Catalyst Center you're targeting by UUID)
 - `"params"`: a dict of any template variable names→values, if your template defines parameters. [mteke.com](#)

- Details about composite templates, using APIs v2, SSL verification, polling for task completion
- Good understanding of Catalyst Center order of operations

• Polling for Task Status

The SDK's response typically contains one or more objects with a `.taskId` field. You can later call:

```
python
```

[Copy](#) [Edit](#)

```
catalyst.task.get_task_by_id(task_id=<taskId>)
```

to check whether deployment succeeded or failed. [github.com](#)

Note:

- If you're using a self-signed certificate on Catalyst Center, keep `verify=False`. Otherwise, point `verify` at your CA bundle or set `verify=True`.
- If you need to push the **same** template to multiple devices, simply append more entries to the `targetInfo` list:

```
python
```

[Copy](#) [Edit](#)

```
targetInfo=[  
    {"id": "device-uuid-1", "type": "MANAGED_DEVICE_UUID", "params": {...}},  
    {"id": "device-uuid-2", "type": "MANAGED_DEVICE_UUID", "params": {...}}  
]
```

That's it! Save this file (e.g. `deploy_cli_template_sdk.py`), edit the top-section variables, and run:

```
bash
```

[Copy](#) [Edit](#)

```
python3 deploy_cli_template_sdk.py
```



Review Generated Code

```
python
Copy Edit

#!/usr/bin/env python3
"""
deploy_cli_template_by_name.py

Uses the Catalyst Center Python SDK (catalystcentersdk) to:

1. Find a CLI template by its NAME and pick the latest committed version.
2. Find a managed device by its HOSTNAME.
3. Deploy that template version to that device.

Prerequisites:
    pip install catalystcentersdk

Variables to set below:
    BASE_URL      - e.g. "https://<CATALYST_CENTER_HOST>"
    USERNAME      - Catalyst Center username
    PASSWORD      - Catalyst Center password
    VERSION       - Catalyst Center API version (e.g., "2.3.7.9")
    TEMPLATE_NAME - The exact name of the CLI template in Catalyst Center
    DEVICE_HOSTNAME - The exact hostname of the managed device
    TEMPLATE_PARAMS - dict of any template-variable values (if your template uses variables)
"""
```

Be aware of cutoff dates, libraries
and references,
and always test in lab.

```
def find_template_version_id(catalyst, template_name):
    """
    Find the latest committed versionId for the CLI template named 'template_name'.
    Returns the version-UUID (a string).
    """
    print(f"→ Searching for template named '{template_name}'...")
    try:
        # The SDK method get_templates(...) calls GET /dna/intent/api/v1/template-program
        # and returns a MyDict with a key 'response' containing a list of matching templates
        response = catalyst.configuration_templates.get_templates(name=template_name,
                                                                includeVersionDetails=True)

    except ApiError as e:
        print(f"[ERROR] Failed to retrieve template list: {e}")
        sys.exit(1)

    templates = response.response # This is a list of template-dicts :contentReference[0]
    if not templates:
        print(f"[ERROR] No template found matching name '{template_name}'.")
        sys.exit(1)

    # Assume the first match is the one we want (names must be unique within a project).
    template = templates[0]
    print(f"    ✓ Found template: id={template.get('templateId')} (Project: {template.get('projectName')})

    # Each template-dict has a 'versionList' key with 'versions': a list of version-dicts
    version_list = template.get("versionList", {}).get("versions", [])
    if not version_list:
        print(f"[ERROR] Template '{template_name}' has no versions available.")
        sys.exit(1)
```

Review Generated Code – continued

```
def find_device_id_by_hostname(catalyst, hostname):
    """
    Look up the managed device that exactly matches 'hostname' and return its UUID.
    """
    print(f"→ Searching for device with hostname '{hostname}'...")
    try:
        # SDK method get_device_list(...) calls GET /dna/intent/api/v1/network-device?hostname={hostname}
        response = catalyst.devices.get_device_list(hostname=hostname)
    except ApiError as e:
        print(f"[ERROR] Failed to retrieve device list: {e}")
        sys.exit(1)

    devices = response.response # List of device-dicts :contentReference[oaicite:3]{index}
    if not devices:
        print(f"[ERROR] No device found with hostname '{hostname}'.")
        sys.exit(1)

    # If there are multiple, pick the first exact match.
    for dev in devices:
        if dev.get("hostname") == hostname:
            device_id = dev.get("id")
            print(f"✓ Found device: hostname={hostname}, id={device_id}, platform={dev.get('platform')}")
            return device_id

    # If none match exactly (e.g. wildcard returned partial matches), bail out.
    print(f"[ERROR] No device with exact hostname '{hostname}' found among {len(devices)}")
    sys.exit(1)
```



```
def deploy_template_to_device(catalyst, template_version_id, device_id, params):
    """
    Deploys the specified template version to the given device UUID.
    Returns the raw SDK response (which includes task-ID(s)).
    """
    print(f"→ Deploying templateVersionId={template_version_id} to deviceId={device_id}..")
    try:
        deploy_resp = catalyst.configuration_templates.deploy_template_v2(
            forcePushTemplate=False,
            isComposite=False,
            mainTemplateId=template_version_id,
            targetInfo=[{
                "id": device_id,
                "type": "MANAGED_DEVICE_UUID",
                "params": params
            }]
        )
    except ApiError as e:
        print(f"[ERROR] Deployment API call failed: {e}")
        sys.exit(1)

    return deploy_resp
```

Review Generated Code

```
def main():  
    print("=== Catalyst Center SDK: Deploy CLI Template by Name → Hostname ===")  
  
    # Step 1: Initialize the CatalystCenterAPI client  
    try:  
        print("1) Connecting to Catalyst Center...")  
        catalyst = api.CatalystCenterAPI(  
            username=USERNAME,  
            password=PASSWORD,  
            base_url=BASE_URL,  
            version=VERSION,  
            verify=False # Change to True if you have a valid TLS certificate  
        )  
        print("    ✓ Connected (token retrieved).")  
    except ApiError as e:  
        print(f"[ERROR] Unable to initialize SDK client: {e}")  
        sys.exit(1)  
  
    # Step 2: Look up templateId/versionId by TEMPLATE_NAME  
    template_version_id = find_template_version_id(catalyst, TEMPLATE_NAME)  
  
    # Step 3: Look up deviceId by DEVICE_HOSTNAME  
    device_id = find_device_id_by_hostname(catalyst, DEVICE_HOSTNAME)  
  
    # Step 4: Deploy the template version to that device  
    deploy_response = deploy_template_to_device(  
        catalyst,  
        template_version_id,  
        device_id,  
        TEMPLATE_PARAMS  
    )
```

Many data validations, error exceptions, and understanding of operations. Very good start.

```
# SDK wraps the raw JSON under deploy_response.response (often a list of task IDs).  
print("→ Raw SDK response object:")  
print(deploy_response)  
  
# Attempt to extract any taskId from the response  
task_ids = []  
# The SDK often returns a list of objects under response.response  
data = deploy_response.response  
if isinstance(data, list):  
    for entry in data:  
        if isinstance(entry, dict) and entry.get("taskId"):  
            task_ids.append(entry["taskId"])  
elif isinstance(data, dict) and data.get("taskId"):  
    task_ids.append(data["taskId"])  
  
if task_ids:  
    print("→ Task ID(s) returned by Catalyst Center (you can poll these under the Task ID(s) section)")  
    for tid in task_ids:  
        print(f"    • {tid}")  
else:  
    print("→ No taskId found in response. Check the full response above to debug.")  
  
print("\nDone. If you wish to track status, call:")  
print("    catalyst.task.get_task_by_id(task_id=<taskId>)")  
  
if __name__ == "__main__":  
    main()
```

AI Generated Code Required Updates

<pre>✓ print(f"→ Deploying templateVersionId={template_version_id} to deviceId={device_id}") try: deploy_resp = catalyst.configuration_templates.deploy_template_v2(forcePushTemplate=False, isComposite=False, mainTemplateId=template_version_id, targetInfo=[{ "id": device_id, "type": "MANAGED_DEVICE_UUID", "params": params }]) except ApiError as e: print(f"[ERROR] Deployment API call failed: {e}") sys.exit(1) return deploy_resp def main(): print("=== Catalyst Center SDK: Deploy CLI Template by Name → Hostname") # Step 1: Initialize the CatalystCenterAPI client try: print("1) Connecting to Catalyst Center...") catalyst = api.CatalystCenterAPI(username=USERNAME, password=PASSWORD, base_url=BASE_URL, version=VERSION, verify=False # Change to True if you have a valid TLS certificate)</pre>	<pre>120 101 121 102 122 103 123 104 124 105 125 106 126 107 127 108 128 109 129 110 130 111 131 112 132 113 133 114 134 115 135 116 136 117 137 118 138 119 139 120 140 121 141 122 142 123 143 124 144 125 145 126 146 127 147 128 148 129 149 130 150 131 151 132</pre>	<pre>print(f"→ Deploying templateVersionId={template_version_id} to deviceId={device_id}") try: deploy_resp = catalyst.configuration_templates.deploy_template_v2(forcePushTemplate=False, isComposite=False, templateId=template_version_id, targetInfo=[{ "id": device_id, "type": "MANAGED_DEVICE_UUID", "params": params }]) except ApiError as e: print(f"[ERROR] Deployment API call failed: {e}") sys.exit(1) return deploy_resp # Step 1: Initialize the CatalystCenterAPI client try: print("1) Connecting to Catalyst Center...") catalyst = api.CatalystCenterAPI(username=USERNAME, password=PASSWORD, base_url=BASE_URL, version=VERSION, verify=False # Change to True if you have a valid TLS certificate)</pre>
---	--	---

- Code requires small changes:
- Incorrect parameters (a couple of functions)
 - Removed the search for the last version template Id

App Run Using Catalyst Center 2.3.7.9

App Run

```
=== Catalyst Center SDK: Deploy CLI Template by Name
1) Connecting to Catalyst Center...
  ✓ Connected (token retrieved).
→ Searching for template named 'logging_profile'...
  ✓ Found template: id=73aab375-30ef-4a41-90de-a110a072ea12
→ Searching for device with hostname 'PDX-RN'...
  ✓ Found device: hostname=PDX-RN, id=657008e6-6160-481e-aa7d-a5b3513b1f7c...
  platform=CSR1000V
→ Deploying templateVersionId=73aab375-30ef-4a41-90de-a110a072ea12
  deviceId=657008e6-6160-481e-aa7d-a5b3513b1f7c...
  ✓ Deployment request submitted.
→ Raw SDK response object:
{'response': {'taskId': '01974212-a663-748d-9a30-5f4ef2a49267', 'versionId': '01974212-a663-748d-9a30-5f4ef2a49267'}}
→ Task ID(s) returned by Catalyst Center (you can poll these task IDs):
  • 01974212-a663-748d-9a30-5f4ef2a49267

Done. If you wish to track status, call:
catalyst.task.get_task_by_id(task_id=<taskId>)
```

Task Status API call

Try 'Get tasks'

Method: **GET** Public URL : <https://10.93.141.45/dna/intent/api/v1/tasks>

- ☐ order ⓘ
asc
- ☐ startTime ⓘ
- ☐ endTime ⓘ
- ☐ parentId ⓘ
- ☒ rootId ⓘ
e9-80b6-4fed7bbc84ca
- ☐ status ⓘ

Reset

Run

Response

Headers

Status Code: 200

```
1 {
2   "response": [
3     {
4       "endTime": 1749082455726,
5       "rootId": "01973d70-3d52-70e9-80b6-4fed7bbc84ca",
6       "status": "SUCCESS",
7       "startTime": 1749082455378,
8       "resultLocation": "/dna/intent/api/v1/tasks/01973d70-3d52-70e9-80b6-4fed7bbc84ca/detail",
9       "id": "01973d70-3d52-70e9-80b6-4fed7bbc84ca"
10    }
11  ],
12  "version": "1.0"
13 }
```

Demo AI Generated Python Code

DEVNET-3419

Version control

Local History: deploy_templates_ai_o4-mini-high.py

4 differences

Before 6/5/25, 2:19 PM

Current

Change 6/5/25, 2:41 PM

Change 6/5/25, 2:40 PM

Change 6/5/25, 2:40 PM

Change 6/5/25, 2:38 PM

Change 6/5/25, 2:36 PM

Change 6/5/25, 2:31 PM

Change 6/5/25, 2:31 PM

Change 6/5/25, 2:29 PM

Change 6/5/25, 2:29 PM

Change 6/5/25, 2:28 PM

Change 6/5/25, 2:27 PM

Change 6/5/25, 2:24 PM

Change 6/5/25, 2:21 PM

Change 6/5/25, 2:19 PM

=== End of user-configurable variables ===

def find_template_version_id(catalyst, template_name):

"""

Find the latest committed versionId for the CLI template named 'templ

Returns the version-UUID (a string).

"""

print(f"→ Searching for template named '{template_name}'...")

try:

The SDK method get_templates(...) calls GET /dna/intent/api/v1/t

and returns a MyDict with a key 'response' containing a list of

response = catalyst.configuration_templates.get_templates(name=tem

except ApiError as e:

print(f"[ERROR] Failed to retrieve template list: {e}")

sys.exit(1)

templates = response.response # This is a list of template-dicts :cor

if not templates:

print(f"[ERROR] No template found matching name '{template_name}'.

sys.exit(1)

Assume the first match is the one we want (names must be unique with

template = templates[0]

print(f" ✓ Found template: id={template.get('templateId')} (Project: {template

Each template-dict has a 'versionList' key with 'versions': a list c

version_list = template.get("versionList", {}).get("versions", [])

if not version_list:

print(f"[ERROR] Template '{template_name}' has no versions availat

sys.exit(1)

SDK method get_device_list(...) calls GET /dna/intent/api/v1/network-device?hostname=<hostname>

38 38 # === End of user-configurable variables ===

39 39

40 40

41 41 def find_template_version_id(catalyst, template_name):

42 42 """

43 43 Find the latest committed versionId for the CLI template named 'template_n

44 44 Returns the version-UUID (a string).

45 45 """

46 46 print(f"→ Searching for template named '{template_name}'...")

47 47 try:

48 48 # The SDK method get_templates(...) calls GET /dna/intent/api/v1/templ

49 49 # and returns a MyDict with a key 'response' containing a list of mat

50 50 response = catalyst.configuration_templates.get_templates_details_v2(n

51 51 except ApiError as e:

52 52 print(f"[ERROR] Failed to retrieve template list: {e}")

53 53 sys.exit(1)

54 54

55 55 templates = response.response # This is a list of template-dicts :content

56 56 if not templates:

57 57 print(f"[ERROR] No template found matching name '{template_name}'.

58 58 sys.exit(1)

59 59

60 60 # Assume the first match is the one we want (names must be unique within a

61 61 template = templates[0]

62 62 print(f" ✓ Found template: id={template.get('id')} (Project: {template.

63 63

64 64 return template.get('id')

65 65

66 66

67 67 def find_device_id_by_hostname(catalyst, hostname):

68 68 """

69 69 Look up the managed device that exactly matches 'hostname' and return its

DEVNET-3419 > deploy_templates_ai_o4-mini-high.py 119:1 LF UTF-8 4 spaces Python 3.12 (DEVNET-3419)

Ansible Code Generation

Write an Ansible Playbook while providing a reference to the latest collection

can you rewrite using this collection: <https://galaxy.ansible.com/ui/repo/published/cisco/catalystcenter/>

Thought for 1m 7s >

Below is an Ansible playbook that uses the **cisco.catalystcenter** 2.3.7.9) to:

1. Find a CLI template by **name**, pick its latest commit
2. Find a managed device by **hostname**, extract its UUID
3. Deploy that template version to the device.

Each task includes a reference to the underlying Catalyst

Citations:

- "The `configuration_template_v1` module supports finding a template whose `templateName` matches the given name"
- "The `network_devices_query` module supports finding a device whose `hostname` matches."
- "The deployment task invokes the `Deploy Ten` (under `template-programmer/template/deploy`) under

yaml

```
---
- name: Deploy a CLI Template
  hosts: localhost
  gather_facts: no
```

vars:

```
# === Catalyst Center
cc_host: "10.10.20.30"
cc_username: "admin"
cc_password: "YourPassword"
cc_port: 443
cc_version: "2.3.7.9"
cc_verify: false
```

```
# === Deployment parameters
template_name: "My-CLI-Template"
device_hostname: "sw1"
template_params:
  interface_name: "Lo1"
  ip_addr: "192.168.1.1"
  mask: "255.255.255.0"
```

collections:

```
- cisco.catalystcenter
```

tasks:

```
- name: > Retrieve template details
  cisco.catalystcenter.configuration_template_v1:
    host: "{{ cc_host }}"
    username: "{{ cc_username }}"
    password: "{{ cc_password }}"
    port: "{{ cc_port }}"
    version: "{{ cc_version }}"
    verify: "{{ cc_verify }}"
    name: "{{ template_name }}"
  register: template_list
  failed_when:
```

```
- template_list.response | length == 0
# Under the covers: GET /dna/intent/api/v1/template-programmer/template
```

```
- name: > Compute the latest commit
  set_fact:
    version_id: >-
    {{
      (template_list.response[0]
      | selectattr("isCommit", "equalto", true)
      | sort(attribute="version")
      | last
      ).versionId
    }}
  when: template_list.response[0].versionList.versions is defined
```

```
cisco.catalystcenter.network_devices_query.
```

```
host: "{{ cc_host }}"
username: "{{ cc_username }}"
password: "{{ cc_password }}"
port: "{{ cc_port }}"
version: "{{ cc_version }}"
verify: "{{ cc_verify }}"
hostname: "{{ device_hostname }}"
register: device_list
failed_when:
```

```
- device_list.response | length == 0
```

```
# Under the covers: GET /dna/intent/api/v1/network-device?hostname=<device_hostname>
```

```
- name: > Extract the exact-match device UUID
```

```
set_fact:
  device_id: >-
  {{
    (device_list.response
    | selectattr("hostname", "equalto", device_hostname)
    | list
    )[0].id
  }}
when: >
```

```
(device_list.response
| selectattr("hostname", "equalto", device_hostname)
| list
) | length > 0
```

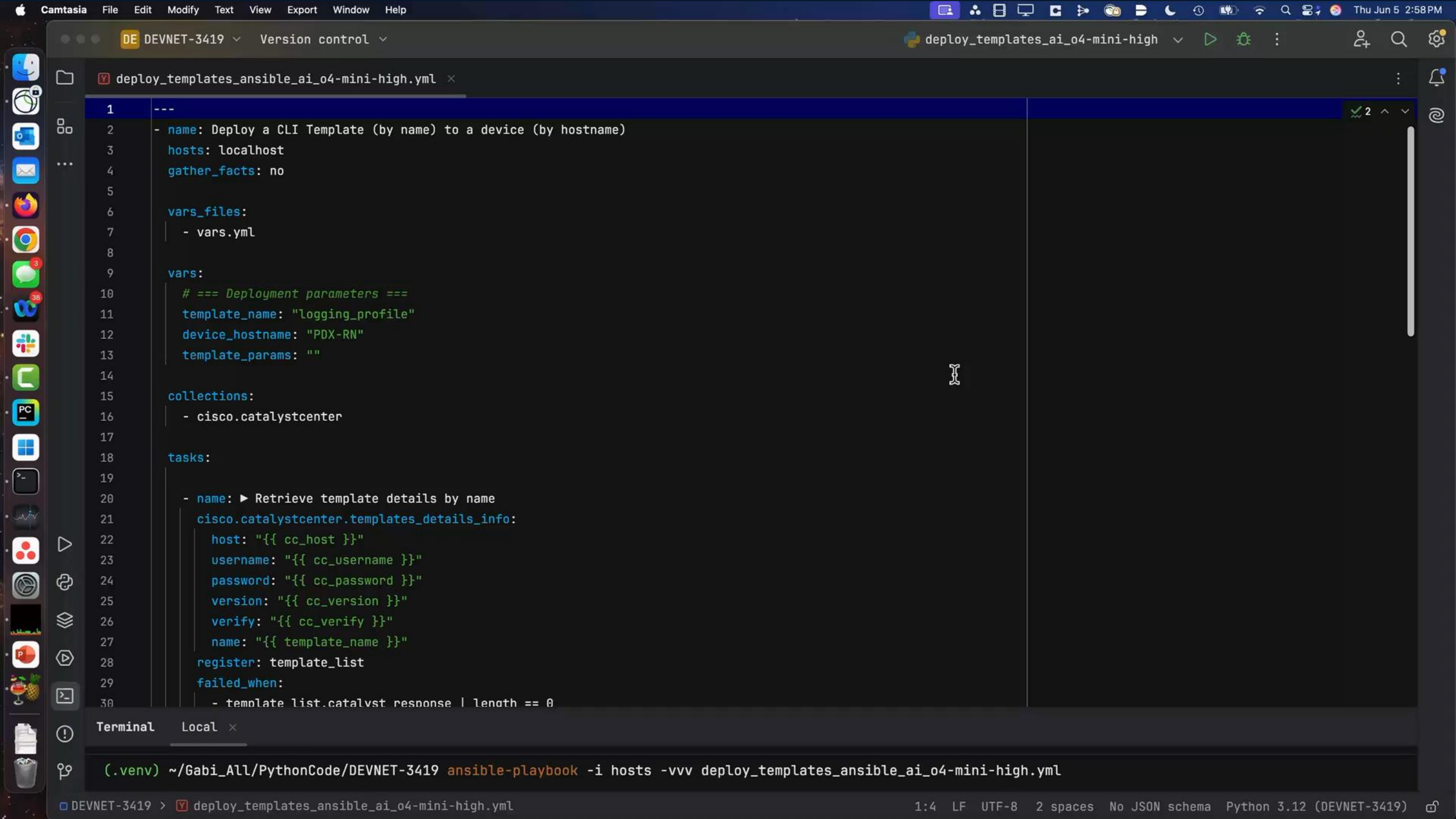


Generated Ansible Code Required Updates

```
17
18 tasks:
19
20 - name: ► Retrieve template details by name
21   cisco.catalystcenter.templates_details_info:
22     host: "{{ cc_host }}"
23     username: "{{ cc_username }}"
24     password: "{{ cc_password }}"
25     version: "{{ cc_version }}"
26     verify: "{{ cc_verify }}"
27     name: "{{ template_name }}"
28   register: template_list
29   failed_when:
30     - template_list.catalyst_response | length == 0
31   # Under the covers: GET /dna/intent/api/v1/template-programmer/template?name=<template_name> :cont
32
33 - name: ► Look up device UUID by hostname
34   cisco.catalystcenter.network_device_info:
35     host: "{{ cc_host }}"
36     username: "{{ cc_username }}"
37     password: "{{ cc_password }}"
38     version: "{{ cc_version }}"
39     verify: "{{ cc_verify }}"
40     hostname: "{{ device_hostname }}"
41   register: device_list
42   failed_when:
43     - device_list.catalyst_response | length == 0
44   # Under the covers: GET /dna/intent/api/v1/network-device?hostname=<device_hostname> :contentReference[aaicite:4]{index=4}
45
46
47 - name: ► Deploy the template version to the device
48   cisco.catalystcenter.configuration_template_deploy_v2:
```

- Good start, but will not run as written
- There are few errors, easy to fix:
 - Updated to parse the data using "catalyst_response"
 - Few modules not identified correctly, or the correct version (v2)
 - Removed some data validations to simplify playbook
- Quality is lower due to using libraries unknown to the LLM model (post cut off), and with little other open-source Ansible code available for this use case, with the new Ansible collection

Demo AI Generated Ansible Playbook



deploy_templates_ansible_ai_o4-mini-high.yml

```
1 ---
2 - name: Deploy a CLI Template (by name) to a device (by hostname)
3   hosts: localhost
4   gather_facts: no
5
6   vars_files:
7     - vars.yml
8
9   vars:
10     # === Deployment parameters ===
11     template_name: "logging_profile"
12     device_hostname: "PDX-RN"
13     template_params: ""
14
15   collections:
16     - cisco.catalystcenter
17
18   tasks:
19     - name: ► Retrieve template details by name
20      cisco.catalystcenter.templates_details_info:
21        host: "{{ cc_host }}"
22        username: "{{ cc_username }}"
23        password: "{{ cc_password }}"
24        version: "{{ cc_version }}"
25        verify: "{{ cc_verify }}"
26        name: "{{ template_name }}"
27        register: template_list
28        failed_when:
29          - template_list.catalyst response | length == 0
```

Terminal Local

```
(.venv) ~/Gabi_All/PythonCode/DEVNET-3419 ansible-playbook -i hosts -vvv deploy_templates_ansible_ai_o4-mini-high.yml
```

Improve AI Generated Code Quality

- Build your AI Code Generation platform using Retrieval Augmented Generation (RAG) with current versions of the:
 - Swagger Files and API Specs
 - Catalyst Center Libraries documentation and examples
 - Open-Source code, your existing IaC libraries, and your enterprise policies
- Try various LLM models



Thank you for sharing the updated code! It looks good, and it follows the proper use of the **Python SDK for Cisco DNA Center version 2.3.7.6**. Let me walk you through the key components and some minor optimizations for clarity:

Key Features:

1. **Device Retrieval:** The code correctly fetches the device information by **management IP address** using `api.devices.get_device_list`.
2. **Template Fetching:** The correct API method is used to retrieve available templates with `api.configuration_templates.gets_the_templates_available()`, and the template ID is selected based on the template name.
3. **Template Deployment:** You are using `deploy_template_v2` with the `forcePushTemplate` option and passing the device ID in the deployment data.
4. **Task Monitoring:** The code checks the deployment task status using `api.task.get_task_by_id` and continues to monitor it in a loop until completion.

Agenda

- 01 Catalyst Center Platform
- 02 Integrations
- 03 GenAI Essentials
- 04 AI Code Generation
- 05 GenAI Network Automation**
- 06 Developer Resources
- 07 Summary

AI Enable your Infrastructure-as-Code Workflows

- Re-use existing investment in:
 - Network automation workflows written in Python, Ansible, Terraform, ...
 - CI/CD pipelines and services layer
- Multi-controller support
- Multi-channel interface in your native speaking language – Bots, ITSM, Voice
- Accelerate Infrastructure-as-Code adoption

```
functions = [  
    {  
        "name": "add_device",  
        "description": "Add a new network device, switch our router to Catalyst Center. Call this function when  
            'user asks to add a network device, device, node, host, to the inventory or Catalyst'  
            'Center while providing an IP address of device. The device must be up and'  
            'reachable by Catalyst Center, and Catalyst Center will use the provided IP address'  
            'to manage the device. For example: an user asks for 'add device'",  
        "parameters": {...}  
    },  
    {  
        "name": "provision_network_device_jenkins",  
        "description": "Provision a device to a site. Call this whenever you need to provision a device, "  
            "or network device, to a location, for example: an user asks "  
            "'Provision a network device to a site'",  
        "parameters": {...}  
    },  
    {  
        "name": "software_distribution",  
        "description": "Start a new software upgrade to a device. Call this whenever you need to start a "  
            "software upgrade or image distribution to a device or network device, to a location, "  
            "for example: an user asks for 'software distribution'",  
        "parameters": {...}  
    }  
]  
]
```

LLM Function Calling

- Connects Large Language Models to other platforms to:
 - Retrieve data with API calls
 - Perform tasks or trigger processes
 - Create new workflows and AI assistants
- The **LLM does not call the function**, it identifies the function, and it returns:
 - function name
 - required parameters

```
functions = [  
  {  
    "name": "provision_network_device_jenkins",  
    "description": "Provision a device to a site. Call this whenever you need to provision a device, "  
                  "or network device, to a location, for example: an user asks "  
                  "'Provision a network device to a site'",  
    "parameters": {  
      "type": "object",  
      "properties": {  
        "hostname": {  
          "type": "string",  
          "description": "This is the device name or hostname, or the network device name or hostname"  
        },  
        "siteHierarchy": {  
          "type": "string",  
          "description": "The site hierarchy, or site name, where the device will be provisioned"  
        }  
      },  
      "required": ["hostname", "siteHierarchy"],  
      "additionalProperties": False  
    }  
  },  
]
```

OpenAI Function Calling

<https://platform.openai.com/docs/guides/function-calling>

OpenAI Platform

Docs API reference Log in Sign up

Search % K

CAPABILITIES

Text generation

Image generation

Vision

Audio generation

Text to speech

Speech to text

Vector embeddings

Moderation

Reasoning

GUIDES

Function calling

Structured outputs

Evaluations

Fine-tuning

Distillation

</> Cookbook

Forum

Help

Function calling

Learn how to connect large language models to external tools.

Introduction

Function calling allows you to connect models like `gpt-4o` to external tools and systems. This is useful for many things such as empowering AI assistants with capabilities, or building deep integrations between your applications and the models.

In August 2024, we launched Structured Outputs. When you turn it on by setting `strict: true`, in your function definition, Structured Outputs ensures that the arguments generated by the model for a function call exactly match the JSON Schema you provided in the function definition.

As an alternative to function calling you can instead constrain the model's regular output to match a JSON Schema of your choosing. [Learn more](#) about when to use function calling vs when to control the model's normal output by using `response_format`.

Example use cases

Function calling is useful for a large number of use cases, such as:

1 **Enabling assistants to fetch data:** an AI assistant needs to fetch the latest customer data from an internal system when a user asks "what are my recent orders?" before it can generate the response to the user

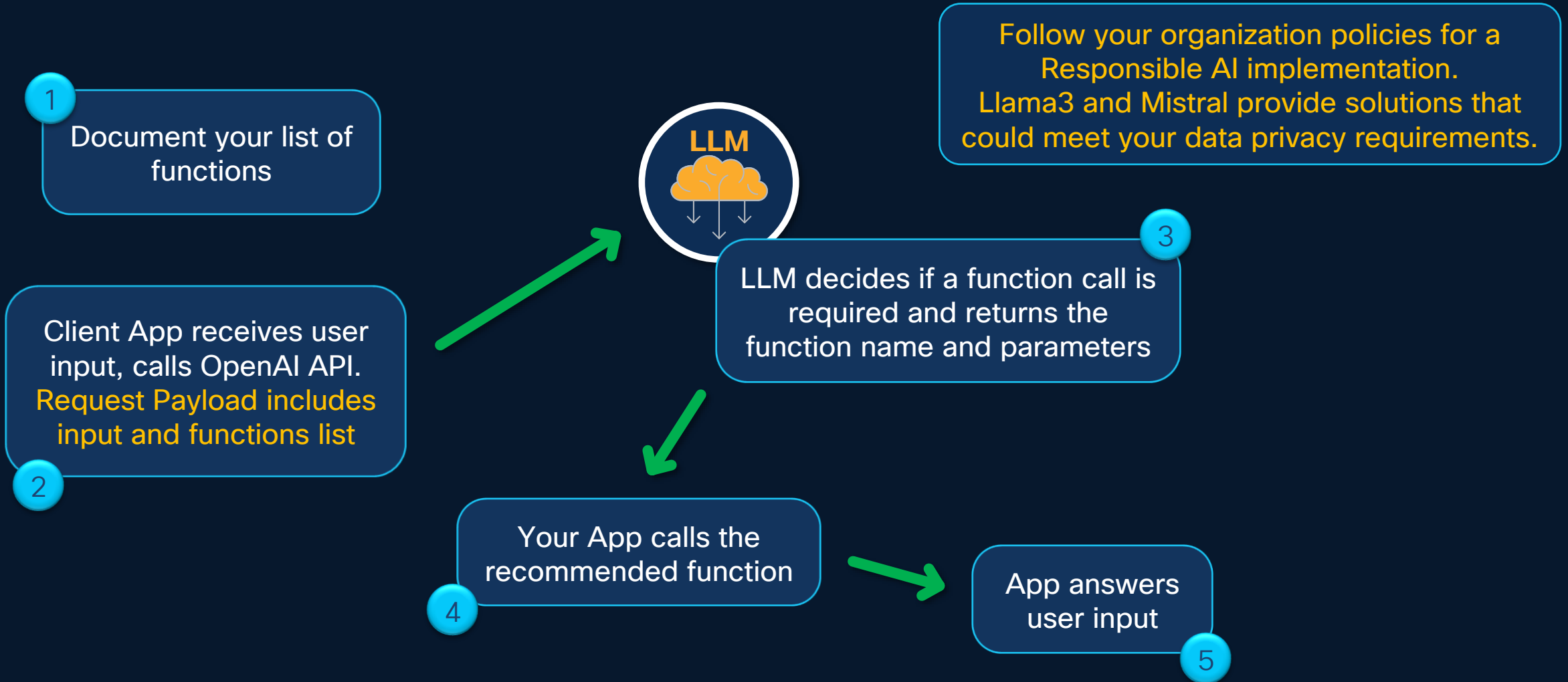
2 **Enabling assistants to take actions:** an AI assistant needs to schedule meetings based on user preferences and calendar availability.

3 **Enabling assistants to perform computation:** a math tutor assistant needs to perform a math computation.

4 **Building rich workflows:** a data extraction pipeline that fetches raw text, then converts it to structured data and

If the LLM of your choice does not have a native function calling, one could be developed for your use case

OpenAI Function Calling Workflow



GenAI Network Operations

```
I am a network assistant running network automation workflows. What network configuration task are you interested in? Can you start the software upgrade for the device PDX-ACCESS?

Workflow name: software_distribution
Params identified:
{"hostname": "PDX-ACCESS"}

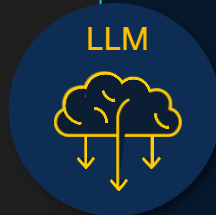
Do you want to continue or not (y/n)? n

I am a network assistant running network automation workflows. What network configuration task are you interested in? Provision this switch BEAV-CN at this location Global/OR/BEAV/Floor-2

Workflow name: provision_network_device_jenkins
Params identified:
{"hostname": "BEAV-CN", "siteHierarchy": "Global/OR/BEAV/Floor-2"}

Do you want to continue or not (y/n)? y

Network Assistant: Device provisioning started, see status here: https://10.93.141.47:8443/job/Provision%20Device/
```



API call to trigger pipeline run



GitHub

Single source of truth, version controlled, data validation, compliant

Pull source-code and intent configs

Pipeline build will configure Catalyst Center and network devices

Pipelines may execute Python, Ansible or Terraform workflows



Multi controllers, lab and production



- Multi-channel support
- Webex Bots
- Web Portal
- ITSM – change management
- Voice interface

Function Database

```
functions = [
{
  "name": "add_device",
  "description": "Add a new network device, switch our router to Catalyst Center. Call this function when
    'user asks to add a network device, device, node, host, to the inventory or Catalyst
    'Center while providing an IP address of device. The device must be up and
    'reachable by Catalyst Center, and Catalyst Center will use the provided IP address
    'to manage the device. For example: an user asks for 'add device'",
  "parameters": {
    "type": "object",
    "properties": {
      "device_ip_address": {
        "type": "string",
        "description": "This is the device IP address that will be used to manage the device"
      }
    }
  },
  "required": ["device_ip_address"],
  "additionalProperties": False
},
],
```

- Create a list of all functions for your workflows
- Document functions following the OpenAPI Specs
- The list of functions and query will be sent to the LLM
- The LLM returns the match: function name and required params, or asks for more details

Function to be
called by your App

```
def add_device(arguments):
    """
    This function will call REST APIs to trigger Jenkins pipeline to add a new network device with
    {management_ip_address} to Catalyst Center
    :param arguments: Required arguments as dictionary: {'management_ip_address': '10.93.141.23'}
    :return: task status, and status code
    """

    url = JENKINS_SERVER + '/job/Add%20Device/buildWithParameters'
    response = requests.post(url, params=arguments, auth=JENKINS_AUTH, verify=False)
    if response.status_code == 201:
        response_status = 'Add new device to inventory started, see status here: https://10.93.141.47:8443/job/Add%20Device/'
    else:
        response_status = 'Pipeline not started, something went wrong'
    return response_status, response.status_code
```

Function Database (List)

- Each function documentation includes:
 - Detailed name and description
 - Params type, descriptions, and if required or not
- Be as detailed as you can when providing the function and parameters description
- Function names and descriptions enable the LLM to match user queries with appropriate functions and identify parameters.

```
functions = [  
  {  
    "name": "add_device",  
    "description": "Add a new network device, switch our router to Catalyst Center. Call this function when "  
      "user asks to add a network device, device, node, host, to the inventory or Catalyst"  
      "Center while providing an IP address of device. The device must be up and"  
      "reachable by Catalyst Center, and Catalyst Center will use the provided IP address"  
      "to manage the device. For example: an user asks for 'add device'",  
    "parameters": {  
      "type": "object",  
      "properties": {  
        "device_ip_address": {  
          "type": "string",  
          "description": "This is the device IP address that will be used to manage the device"  
        }  
      },  
      "required": ["device_ip_address"],  
      "additionalProperties": False  
    }  
  },  
  {  
    "name": "provision_network_device",  
    "description": "Provision a device to a site. Call this whenever you need to provision a device, "  
      "or network device, to a location, for example: an user asks "  
      "'Provision a network device to a site'",  
    "parameters": {  
      "type": "object",  
      "properties": {  
        "hostname": {  
          "type": "string",  
          "description": "This is the device name or hostname, or the network device name or hostname"  
        }  
      }  
    }  
  }  
]
```

Chat Completion using Function Calling

```
138
139 # Start a chat that triggers the function
140 messages = [
141     {"role": "system", "content": "I am a network assistant running network automation workflows like devices "
142                                     "provisioning, PnP onboarding, software upgrades, device configuration, ..."},
143     {"role": "user",
144      "content": user_input}
145 ]
146
147 # Make a request to the OpenAI API with function calling
148 response = client.chat.completions.create(model="gpt-4o", messages=messages, functions=functions,
149                                           function_call="auto")
150 print(' Chat completion response: ' + str(response))
151
152 # Check if the assistant decided to make a function call
153 if response.choices[0].finish_reason == 'function_call':
154     function_call = response.choices[0].message.function_call # Access the function call details
155     function_name = function_call.name # Get the function name
156     arguments = json.loads(function_call.arguments) # Load the arguments as a JSON object
157     print('\n Workflow name: ' + function_name)
158     print(' Params identified: \n      ' + json.dumps(arguments))
159     validation_input = input('\n Do you want to continue or not (y/n)? ')
160
```

- Collect the user input
- Call the chat completion API:
 - Specify model – “gpt-4o”
 - API call includes the user input, prompt, and functions list
- The LLM will use the query and function list to identify the best match returning the function and the parameters values

Function Calling Response

Response

I am a network assistant running network automation workflows. What network configuration task are you interested in? *Start the image distribution to the device PDX-RN*

Chat completion response: ChatCompletion(id='chatcmpl-ALfZVf1knvWDmVmiLvUkCQwpurGw', choices=[Choice(finish_reason='function_call', index=0, logprobs=None, message=ChatCompletionMessage(content=None, refusal=None, role='assistant', audio=None, function_call=FunctionCall(arguments='{"hostname": "PDX-RN"}', name='software_distribution'), tool_calls=None)), created=1729728053, model='gpt-4o-2024-08-06', object='chat.completion', service_tier=None, system_fingerprint='fp_9e15ccd6a4', usage=CompletionUsage(completion_tokens=18, prompt_tokens=230, total_tokens=248, completion_tokens_details=CompletionTokensDetails(audio_tokens=None, reasoning_tokens=0), prompt_tokens_details=PromptTokensDetails(audio_tokens=None, cached_tokens=0)))

Workflow name: software_distribution
Params identified:
{"hostname": "PDX-RN"}

Do you want to continue or not (y/n)?

Proposed action, and
parameters values,
identified based on
user input

JSON Formatted Response

```
{
  "id": "chatcmpl-ALfZVf1knvWDmVmiLvUkCQwpurGw",
  "model": "gpt-4o-2024-08-06",
  "created": 1729728053,
  "object": "chat.completion",
  "system_fingerprint": "fp_9e15ccd6a4",
  "service_tier": null,
  "choices": [
    {
      "index": 0,
      "role": "assistant",
      "finish_reason": "function_call",
      "message": {
        "content": null,
        "function_call": {
          "name": "software_distribution",
          "arguments": {
            "hostname": "PDX-RN"
          }
        }
      }
    }
  ],
}
```

Function Calling Error Handling

I am a network assistant running network automation workflows. What network configuration task are you interested in? *Can you create a fabric at this site: Global/OR/PDX/Floor-2?*

Something went wrong, not sure what you asked, Creating a fabric involves a complex process that involves tasks such as provisioning devices, configuring network settings, and possibly coordinating with site-specific infrastructure requirements. While I'm equipped to handle provisioning network devices to specific sites, the creation of a network fabric is beyond my direct capabilities.

However, if you have devices that need provisioning to this site, please provide their hostname, and I can assist you with that!

I am a network assistant running network automation workflows. What network configuration task are you interested in? |

No function call recommended, provides info about the documented functions.

Very interesting that it knows the required steps to configure a fabric. One required step for configuring a fabric is to provision devices, operation offered to be completed by the AI tool.

Function correctly identified, but missing required parameter

I am a network assistant running network automation workflows. What network configuration task are you interested in? *Start the provisioning of this device PDX-R0*

Something went wrong, not sure what you asked, Could you please provide the site hierarchy or site name where the device PDX-R0 should be provisioned?

I am a network assistant running network automation workflows. What network configuration task are you interested in? |

Language Support

Understands queries in other languages,
can answer them in another language

I am a network assistant running network automation workflows. What network configuration task are you interested in? *Poti sa adaugi acest echipament NYC-RO la rețeaua de date?*

Pentru a adăuga un echipament la rețeaua de date,
 respectiv. Te rog să îmi furnizezi adresa IP pentru

I am a network assistant running network automation
are you interested in? *Poti sa adaugi acest echipa
date?*

```
Workflow
Params id
{"de
```

```
I am a network assistant running
configuration task are y
10.93.141.12 zum Datenne I am a
```

Workflow name: add_device

Params identified:

```
{"device_ip_address": "10.10.10.10",
```

Do you want to continue c

I am a network assistant running network automation workflows. What network configuration task are you interested in? ホスト名 J-TOK-SW10 でスイッチのソフトウェア アップグレードを開始します。

Workflow name: software_distribution

Params identified:

```
{"hostname": "J-TOK-SW10"}
```

I am a network assistant running network automation workflows. What can I help you with?

What can you help me with? Answer in Italian

Posso aiutarti con diverse attività di automazione della rete, come ad esempio:

1. Aggiungere nuovi dispositivi di rete al tuo inventario.
2. Effettuare il provisioning di dispositivi di rete in una specifica sede.
3. Aggiornare il software dei dispositivi di rete.

Se hai bisogno di assistenza con una di queste attività, fammi sapere!

Write Function Call Docs using ChatGPT 4o

```
75
76 def main():
77     """
78     This application will collect the inventory using the Cisco Catalyst Center REST APIs.
79     It will collect all devices managed by Catalyst Center and create inventory files for all devices, all APs,
80     non-compliant security advisories and image.
81     App workflow:
82     - create device inventory - hostname, device management IP address, device UUID, software version,
83       device family, role, site, site UUID
84     - create access point inventory - hostname, device management IP address, device UUID, software version,
85       device family, role, site, site UUID
86     - identify device image compliance state and create image non-compliance report
87     - create a report for security advisories non-compliant devices
88     - save all files to local folder, formatted using JSON and YAML
89     - push the inventory files to GitHub repo
90     :param:
91     :return:
92     """
```

Ask LLMs to create the Function Calling documentation for your functions.

ChatGPT 4o ▾

Can you help me write the OpenAI function calling list doc for this function?

device_inventory.py

```
def main():
```

```
    """
```

This application will collect the inventory using the Cisco Catalyst Center REST APIs.

It will collect all devices managed by Catalyst Center and create inventory files for all devices, all APs, non-compliant security advisories and image.



Here's a structured OpenAI function calling list documentation for your `device_inventory.py` function:

Function Call Docs using ChatGPT 4o – continued

Description:

This function collects the device inventory using Cisco Catalyst Center REST APIs. It retrieves all managed devices and generates inventory files for:

- **All Devices:** A complete list of devices.
- **All Access Points (APs):** A subset of devices.
- **Non-Compliant Security Advisor:** Information on security requirements.
- **Image Compliance:** Information on image requirements.

OpenAI Function Signature:

```
json
{
  "name": "get_device_inventory",
  "description": "Collects the device inventory using Cisco Catalyst Center REST APIs. It retrieves all managed devices and generates inventory files for:",
  "parameters": {
    "type": "object",
    "properties": {},
    "required": []
  }
}
```

This is a very good recommendation

Ask to use my existing docs style for new functions, to improve format, and maintain consistency with existing list

Can you redo using this format?

```
{
  "name": "add_device",
  "description": "Add a new network device to Catalyst Center. Call this function with the device name, host, to the inventory or Catalyst Center while providing an IP address. The device must be up and reachable by Catalyst Center. Catalyst Center will use the provided IP address to manage the device. For example, if the user asks for 'add device', the function will be used to manage the device"
  "parameters": {
    "type": "object",
    "properties": {
      "device_ip_address": {
        "type": "string",
        "description": "This is the device IP address that will be used to manage the device"
      }
    },
    "required": ["device_ip_address"],
    "additionalProperties": False
  }
}
```



Here is the function calling list documentation in the requested format:

json

Copy Edit

```
{
  "name": "get_device_inventory",
  "description": "Collects the device inventory using Cisco Catalyst Center REST APIs. It retrieves all managed devices and generates inventory files for:",
  "parameters": {
    "type": "object",
    "properties": {},
    "required": [],
    "additionalProperties": False
  }
}
```

Demo GenAI Network Operations

Chrome

File Edit View History Bookmarks Profiles Tab Window Help

Home - Cisco Catalyst Center

Dashboard - Jenkins

Not Secure

https://10.93.141.45/dna/home

Search Star Bookmarks Extensions

Cisco Systems, Inc

APIs

Tools

LAB_Access

CiscoLive

Meraki Dashboard...

Syno SFEH-NAS - Synol...

CDETS

DNACaaP

SharePoint

APIs and Owners -...

JIRA Engineering...

Catalyst Center A...

API-Executive-Das...

Menu

Cisco

Catalyst Center

Star Search Cloud Help

Notifications

Gabi

Dropdown

Welcome to Catalyst Center!

Explore

Assurance Summary

Health

Healthy as of Jun 5, 2025 4:31 PM

100%

Network Devices

--%

Wireless Clients

89%

Wired Clients

View Details

Critical Issues

Last 24 Hours

0

P1

0

P2

View Details

Trends and Insights

Last 30 Days

0

AP Performance Advisories

0

Trend Deviations

View Details

Network Snapshot

Sites

As of Jun 5, 2025 4:31 PM

15

DNS Servers : 1

NTP Servers : 1

Add Sites

Network Devices

As of Jun 5, 2025 4:31 PM

13

Unclaimed: 0

Unprovisioned: 2

Unreachable: 0

Find New Devices

Application QoS Policies

As of Jun 5, 2025 4:31 PM

0

Successful Deploys: 0

Errored Deploys: 0

Stale Policies: 0

Add New Policy

Ollama Tool Calling Support

<https://ollama.com/blog/tool-support>

- Llama3 Models support tool calling
- Tool Calling requires minimal changes: rename Functions with Tools, few format changes
- Call Ollama Chat

The screenshot shows the Ollama website's 'Models' page with the 'Tools' filter selected. It displays three models:

- llama3.3**: New state of the art 70B model. Llama 3.3 70B offers similar performance compared to the Llama 3.1 405B model. **tools** 70b. 1.2M Pulls, 14 Tags, Updated 2 months ago.
- llama3.2**: Meta's Llama 3.2 goes small with 1B and 3B models. **tools** 1b 3b. 8.6M Pulls, 63 Tags, Updated 4 months ago.
- llama3.1**: Llama 3.1 is a new state-of-the-art model from Meta available in 70B and 405B parameter sizes. **tools** 8b 70b 405b. 23M Pulls, 93 Tags, Updated 2 months ago.

```
tools = [
{
  "type": "function",
  "function": {
    "name": "add_device",
    "description": "Add a new network device, switch our router to Catalyst Center. "
    "Call this function when user asks to add a network device, device, node, host, "
    "to the inventory or Catalyst Center while providing an IP address of device "
    "to manage the device. For example: an user asks for 'add device'",
    "parameters": {
      "type": "object",
      "properties": {
        "device_ip_address": {
          "type": "string",
          "description": "This is the device IP address that will be used to manage the device"
        },
      },
      "required": [
        "device_ip_address"
      ]
    }
  }
}
```

GenAI Network Operations

https://github.com/zapodeanu/catalyst_center_genai_ops

zapodeanu / catalyst_center_genai_ops

Type / to search

<> Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings

zapodeanu

catalyst_center_genai_ops

Public

Pin

Unwatch 1

Fork 1

Star 2

main 1 Branch 0 Tags

Go to file

Add file

<> Code

About

zapodeanu

updated for CL

20e25b2 · now

6 Commits

.gitignore	updated Readme, gitignore, app	9 months ago
CODE_OF_CONDUCT.md	updated for CL	1 minute ago
CONTRIBUTING.md	updated for CL	1 minute ago
LICENSE	updated for CL	1 minute ago
NOTICE	updated for CL	1 minute ago
README.md	updated for CL	now
catalyst_center_genai_config_tools.py	updated for CL	1 minute ago
requirements.txt	updated for CL	1 minute ago

README

Code of conduct

License

Catalyst Center Ops using GenAI

The repo includes the files for GenAI operations using function calling and Catalyst Center REST APIs

Readme

View license

Code of conduct

Activity

2 stars

1 watching

1 fork

Releases

No releases published

Create a new release

Packages

No packages published

Publish your first package

Agenda

- 01 Catalyst Center Platform
- 02 Integrations
- 03 GenAI Essentials
- 04 AI Code Generation
- 05 GenAI Network Automation
- 06 Developer Resources**
- 07 Summary

Catalyst Center API Journey



Cisco API
platform launch



Python SDK



Ansible
collection



Ansible
RedHat certified
Terraform provider
Jenkins open-
source



Ansible community
certification
Workflows Ansible
modules introduced



Python SDK,
Ansible,
Terraform
Support for
Catalyst Center
2.3.5.3



AI enabled use
cases
Support for
Catalyst Center
2.3.7.9+

2018

2019

2020

2021

2022

2023

2024/2025

API adoption



5+ billion API calls/quarter, 54% customers



1130+ APIs available in version 2.3.7.9



271 events and 67 reports

DevOps resources



Python SDK – 20,000 downloads/month



Ansible collection – 101,000 downloads

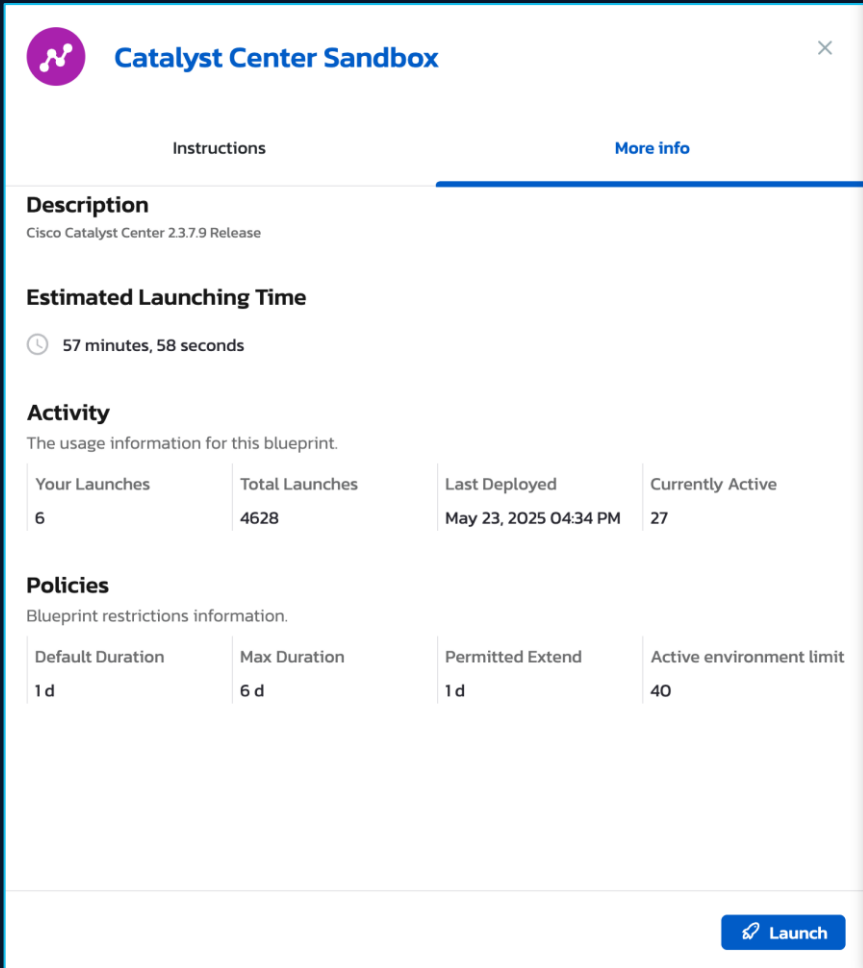


Terraform provider – 50,000 downloads

40 Catalyst Center DevNet sandboxes

https://devnetsandbox.cisco.com/DevNet/catalog/catalyst-center-sandbox_catalyst-center

DevNet Catalyst Center Sandboxes



The screenshot displays the 'Catalyst Center Sandbox' interface. It includes a 'Description' section with the text 'Cisco Catalyst Center 2.3.7.9 Release'. The 'Estimated Launching Time' is shown as '57 minutes, 58 seconds'. The 'Activity' section provides usage information for the blueprint, including a table with 'Your Launches' (6), 'Total Launches' (4628), 'Last Deployed' (May 23, 2025 04:34 PM), and 'Currently Active' (27). The 'Policies' section shows blueprint restrictions information, including a table with 'Default Duration' (1 d), 'Max Duration' (6 d), 'Permitted Extend' (1 d), and 'Active environment limit' (40). A 'Launch' button is visible at the bottom right.

Your Launches	Total Launches	Last Deployed	Currently Active
6	4628	May 23, 2025 04:34 PM	27

Default Duration	Max Duration	Permitted Extend	Active environment limit
1 d	6 d	1 d	40

- 40 Catalyst Center DevNet sandboxes
- Include virtual Catalyst Center 2.3.7.9, ISE, CML, and 4 Cat9kV switches
- Unrestricted access to all Catalyst Center APIs
- Build and test your IaC workflows
- Develop your custom integrations
- Reserve yours for 6 days
- Free

https://devnetsandbox.cisco.com/DevNet/catalog/catalyst-center-sandbox_catalyst-center

Catalyst Center Python SDK

- Covers all Cisco Catalyst Center REST APIs
- Support for Catalyst Center versions up to 2.3.7.9
- CatalystCenterSDK rebranded library released
- Average downloads/month 20,000+

Installation

Installation

PIP Install

catalystcentersdk is available via PIP and the [Python Package Index \(PyPI\)](#). To install catalystcentersdk, simply run this command from your terminal of choice:

```
$ pip install catalystcentersdk
```

The catalystcentersdk package is distributed as a *source distribution* (no binaries).

PIP Upgrade

To ensure that you have the latest version, check-for and install upgrades via PIP:

```
$ pip install catalystcentersdk --upgrade
```

Get the Source Code

catalystcentersdk is developed on GitHub. If you like and use this package, please take a few seconds to Star the package on the [cisco-en-programmability/catalystcentersdk](#) GitHub page. Your feedback and contributions are always welcome.

Use the following command to download the source code (GIT repository):

```
$ git clone https://github.com/cisco-en-programmability/catalystcentersdk.git
```

latest

<https://catalystcentersdk.readthedocs.io>
<https://dnacentersdk.readthedocs.io>

2 Branches 9 Tags

Go to file

Code

About

No description, website, or topics provided.

Readme

MIT license

Activity

Custom properties

2 stars

3 watching

1 fork

Report repository

Releases 9

v2.3.7.9.5 Latest

on Mar 5

+ 8 releases

Packages

JosePabloOcampo1212	Merge pull request #9 fro...	e60d759 · 2 months ago	24 Commits
catalystcentersdk	Error, correction in the user_and_roles ...	2 months ago	
docs	Error, correction in the user_and_roles ...	2 months ago	
script	First commit, catalystcenter, version 2....	7 months ago	
tests	Alias have been adjusted for backward ...	3 months ago	
.gitattributes	Repository reset	7 months ago	
.gitignore	Repository reset	7 months ago	
.readthedocs.yml	modified version from 2.3.7.6.1 to 2.3.7....	7 months ago	
CHANGELOG.md	Merge branch 'main' into develop	2 months ago	
LICENSE	First commit, catalystcenter, version 2....	7 months ago	
MANIFEST.in	First commit, catalystcenter, version 2....	7 months ago	
Pipfile	First commit, catalystcenter, version 2....	7 months ago	

<https://github.com/cisco-en-programmability/catalystcentersdk>

Cisco Catalyst Center Ansible Collection

<https://galaxy.ansible.com/ui/repo/published/cisco/catalystcenter>

The screenshot shows the 'cisco.catalystcenter' collection page. The breadcrumb trail is 'Namespaces > cisco > catalystcenter'. The collection name 'cisco.catalystcenter' is highlighted with a green box. Below it, the version is '2.1.4 updated 11 days ago (latest)' and it was 'Last updated 11 days ago'. The 'Install' tab is selected, showing the title 'Ansible Modules for Cisco CATALYST Center'. There are tags for 'catalystcenter', 'collection', 'sdn', 'cisco', 'cloud', and 'networking'. The installation command is 'ansible-galaxy collection install cisco.catalystcenter'. A note states: 'Note: Installing collections with ansible-galaxy is only supported in ansible-core >= 2.13.9'. There is a 'Download tarball' link. It also shows 'Requires Ansible >= 2.15.0' and the title 'Ansible Collection - cisco.catalystcenter'.

<https://galaxy.ansible.com/ui/repo/published/cisco/dnac>

The screenshot shows the 'cisco.dnac' collection page. The breadcrumb trail is 'Namespaces > cisco > dnac'. The collection name 'cisco.dnac' is highlighted with a green box. Below it, the version is '6.33.2 updated 9 days ago (latest)' and it was 'Last updated 9 days ago'. The 'Install' tab is selected, showing the title 'Ansible Modules for Cisco DNA Center'. There are tags for 'collection', 'sdn', 'cisco', 'dnac', 'cloud', and 'networking'. The installation command is 'ansible-galaxy collection install cisco.dnac'. A note states: 'Note: Installing collections with ansible-galaxy is only supported in ansible-core >= 2.13.9'. There is a 'Download tarball' link. It also shows 'Requires Ansible >= 2.15.0' and the title 'Ansible Collection - cisco.dnac'.

- Complete set of Ansible modules for all Catalyst Center REST APIs
- Support for Catalyst Center versions up to 2.3.7.9
- 101,000+ downloads
- Libraries may be found

<https://github.com/cisco-en-programmability>

Cisco Developed, TAC/Engineering Supported

Namespaces > cisco > dnac > Content

cisco.dnac

Version 6.33.2 updated 9 days ago (lates... Last updated 9 days ago 101,128 Downloads

Install Documentation **Contents** Import log Dependencies

Docs site Website Issue tracker Repo

Q workflow_manager X

Showing: All (65) Action (0) Doc_fragments (1) **Module (32)** Module_utils (0) Playbook (32)

Name	Type	Description
accesspoint_workflow_manager	module	Automates bulk AP configuration changes.
application_policy_workflow_manager	module	Resource module for managing queuing profiles, applications, app... wired and wireless in Cisco Catalyst Center.
assurance_device_health_score_settings_workflow_manager	module	Resource module for managing assurance Health score settings in
assurance_icap_settings_workflow_manager	module	Configure and manage ICAP (Intelligent Capture) settings in Cisco
assurance_issue_workflow_manager	module	Resource module for managing assurance global profile settings and
device_configs_backup_workflow_manager	module	Device Configs Backup module for taking configuration backups of reachable devices in the Cisco Catalyst Center.

- Support starting with version 2.3.7.6
- Identified with the "workflow_manager"
- Ansible playbooks for sites, PnP, SWI templates, network

- Support starting with Cisco Catalyst Center version 2.3.7.6
- Identified with the module name including “workflow_manager”
- Ansible playbooks for common operations for sites, PnP, SWIM, compliance, templates, network settings, ...

Terraform Provider and Go SDK

<https://registry.terraform.io/providers/cisco-en-programmability/dnacenter>

Providers / cisco-en-programmability / dnacenter / Version 1.3.1-beta ▾ Latest Version

dnacenter 

 **dnacenter**

 Partner by: [cisco-en-programmability](#)

Networking

VERSION	PUBLISHED	SOURCE CODE
1.3.1-beta	3 months ago	cisco-en-programmability/terraform-provider-dnacenter

- Extended coverage of Catalyst Center REST APIs
- Support for Catalyst Center versions up to 2.3.7.9
- 50,000+ downloads
- Libraries may be found here:

<https://github.com/cisco-en-programmability>

 Terraform

Providers / cisco-en-programmability / catalystcenter / Version 1.1.1-beta ▾ Latest Version

catalystcenter 

 **catalystcenter**

 Partner by: [cisco-en-programmability](#)

Networking

VERSION	PUBLISHED	SOURCE CODE
1.1.1-beta	3 months ago	cisco-en-programmability/terraform-provider-catalystcenter

<https://registry.terraform.io/providers/cisco-en-programmability/catalystcenter/>

Catalyst Center Splunk Integration Apps

<https://splunkbase.splunk.com/app/7859>



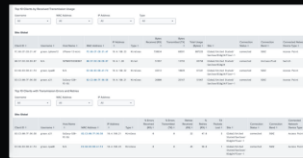
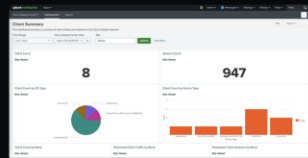
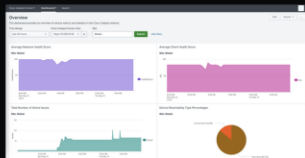
Catalyst Center App

Cisco Catalyst Center and Splunk have partnered to deliver an advanced security reporting and analysis tool. This collaboration provides operational insights and customizable dashboards across Cisco Catalyst Center, empowering administrators with real-time visibility and...

Built by Cisco EN Programmability



Login to Download



- Two Apps:
 - Add-on – data collection
 - Visualizations
- Published on Splunkbase and GitHub
- Support via GitHub issues
- Fully customizable

- Adoption – 1,000+ downloads
- New features:
 - Client dashboards
 - Sites filtering and improved user experience
 - Catalyst Center rebranded



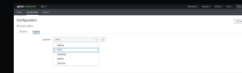
Catalyst Center Add-on

The Cisco Catalyst Center Add-on for Splunk allows a Splunk® Enterprise or Splunk Cloud administrator to collect data from Cisco Catalyst Center APIs.

Built by Cisco EN Programmability



Login to Download



<https://splunkbase.splunk.com/app/7858>

Cisco EN Splunk Integration Apps

- Integrate with ISE, Catalyst SD-WAN, Catalyst Center, and CyberVision
- Apps – Visualizations and Add-on for data collection
- Splunk Enterprise + Cloud certified



Cisco Catalyst Add-on for Splunk

Cisco Catalyst Add-on for Splunk collects data for different Cisco Products - **Cisco Identity Services Engine**, **Cisco Catalyst SD-WAN**, **Cisco Catalyst Center**, and **Cisco CyberVision**. The add-on parses the data from these sources and stores them into the Splunk indexes. *.
Built by [Cisco Systems, Inc.](#)



Login to Download



Latest Version 2.0.0

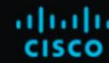
May 29, 2025

[Release notes](#)

Compatibility



Splunk Enterprise
Platform Version: 9.4, 9.3, 9.2,
9.1
CIM Version: 5.X



Cisco Enterprise Networking for Splunk Platform

The **Cisco Enterprise Networking for Splunk Platform** presents visualizations in dashboards for different Cisco Products - **Cisco Identity Services Engine**, **Cisco Catalyst SD-WAN**, and **Cisco Catalyst Center**. The App uses the data collected by "Cisco Catalyst Add-on for...
Built by [Cisco Systems, Inc.](#)



Login to Download



Latest Version 2.0.0

May 29, 2025

[Release notes](#)

Compatibility



Splunk Enterprise, Splunk
Cloud
Platform Version: 9.4, 9.3, 9.2,
9.1
CIM Version: 5.X

Rating

0 ☆☆☆☆☆ (0)
Log in to rate this app

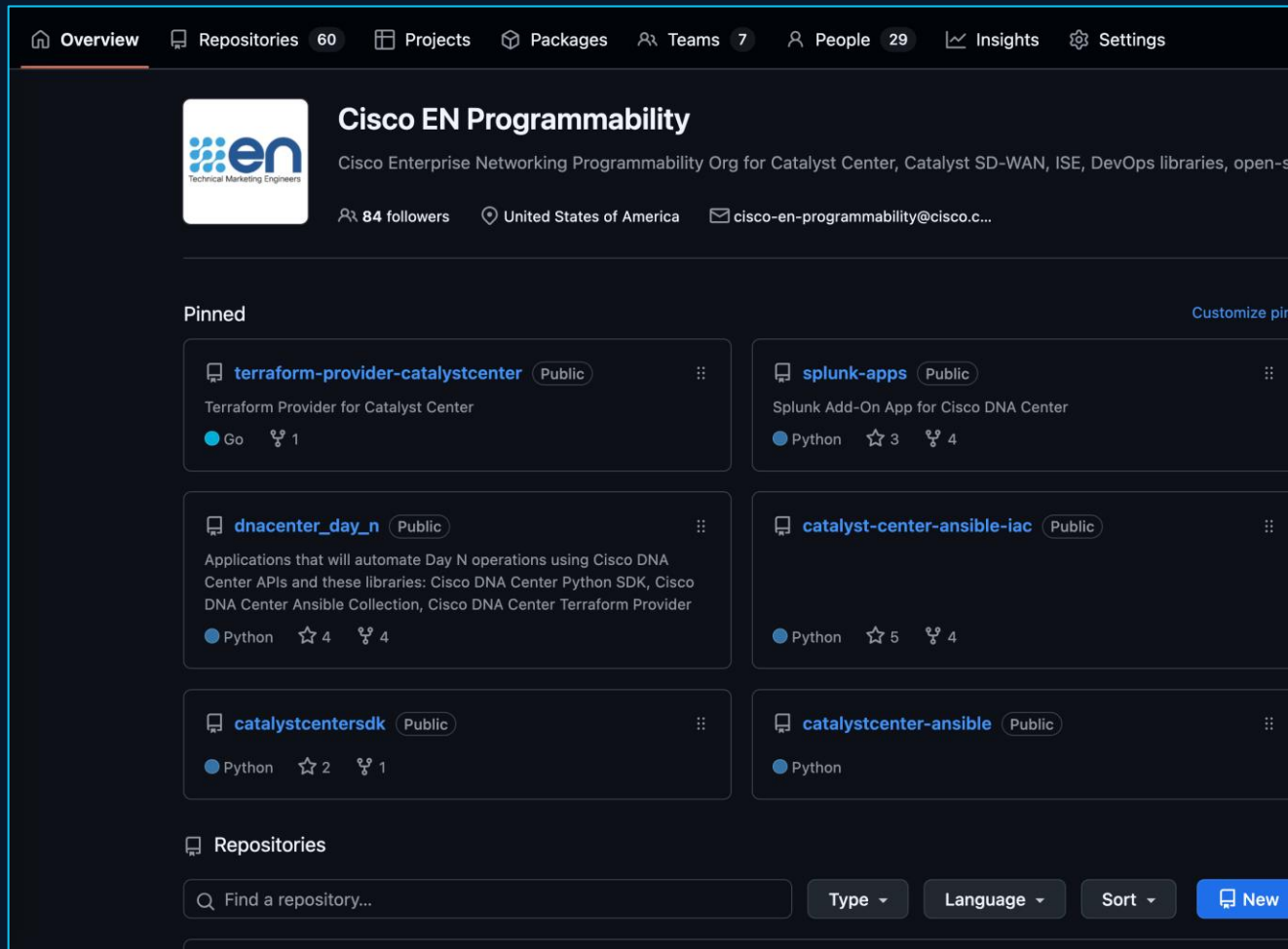
Support

Cisco Supported App
[Learn more](#)

Ranking

#4 In Network Access
Control
#20 In Firewall

Cisco EN Programmability GitHub Org



- Cisco EN DevOps Resources for Catalyst Center, Catalyst SD-WAN
- DevOps Catalyst Center use cases
- Python SDK
- Ansible modules
- Go SDK and Terraform provider
- Splunk Integration Apps
- GenAI use case
- SDA-as-Code
- Report Operations
- Day N use cases – Python SDK, Ansible, Terraform
- Jenkins open-source Integration and pipelines
- GitHub/GitLab open-source Integrations
- Compliance
- Custom Integration App (Jira Service Desk)
- Webhook Receiver
- Other sample code

<https://github.com/cisco-en-programmability>

Agenda

- 01 Catalyst Center Platform
- 02 Integrations
- 03 GenAI Essentials
- 04 AI Code Generation
- 05 GenAI Network Automation
- 06 Developer Resources
- 07 **Summary**

Summary

```
functions = [  
  {  
    "name": "add_device",  
    "description": "Add a new network device, switch our router to Catalyst Center.  
    \"user asks to add a network device, device, node, host, to the i  
    \"Center while providing an IP address of device.\"  
    \"to manage the device. For example: an user asks for 'add device  
    \"  
    \"parameters\": {  
      \"type\": \"object\",  
      \"properties\": {  
        \"device_ip_address\": {  
          \"type\": \"string\",  
          \"description\": \"This is the device IP address that will be used to  
        },  
      },  
      \"required\": [\"device_ip_address\"],  
      \"additionalProperties\": False  
    },  
  },  
  {  
    \"name\": \"provision_network_device\",  
    \"description\": \"Provision a device to a site. Call this whenever you need to pr  
    \"or network device, to a location, for example: an user asks \"  
    \"'Provision a network device to a site'\",  
    \"parameters\": {  
      \"type\": \"object\",  
      \"properties\": {  
        \"hostname\": {  
          \"type\": \"string\",  
          \"description\": \"This is the device name or hostname, or the network device name or hostname\"  
        },  
      },  
    },  
  },  
]
```

- Bridge the APIs and GenAI skills gaps to democratize the use of AI for Network Management
- Transform how we manage networks through GenAI NetOps: configuration, insights, compliance and troubleshooting
- Use AI Code Generation to accelerate development of Infrastructure-as-Code use cases
- Explore Model Context Protocol (MCP) for complex, multistage workflows or production apps

Catalyst Center Platform Sessions

Session	Title	Room	Date
DEVNET-3419	Transform your Network Automation Apps with GenAI	DevNet Theater 2	06/10/2025, 5:30 PM
DEVNET-2660	Catalyst Center Network Operations Essentials using UI and APIs	DevNet Theater 2	06/11/2025, 10:00 AM
BRKOPS-3822	GenAI Your Way: Reimagine Network Troubleshooting	Upper Level, Room 8	06/11/2025, 1:30 PM
DEVNET-3000	Open-Source GenAI Bot for Catalyst Center	DevNet Theater 1	06/12/2025 9:30 AM

Complete your session evaluations



Complete a minimum of 4 session surveys and the Overall Event Survey to be entered in a drawing to win 1 of 5 full conference passes to Cisco Live 2026.



Earn 100 points per survey completed and compete on the Cisco Live Challenge leaderboard.



Level up and earn exclusive prizes!



Complete your surveys in the Cisco Live mobile app.

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 www.CiscoLive.com/on-demand

Contact me at: gabi@cisco.com

Thank you

CISCO Live !

