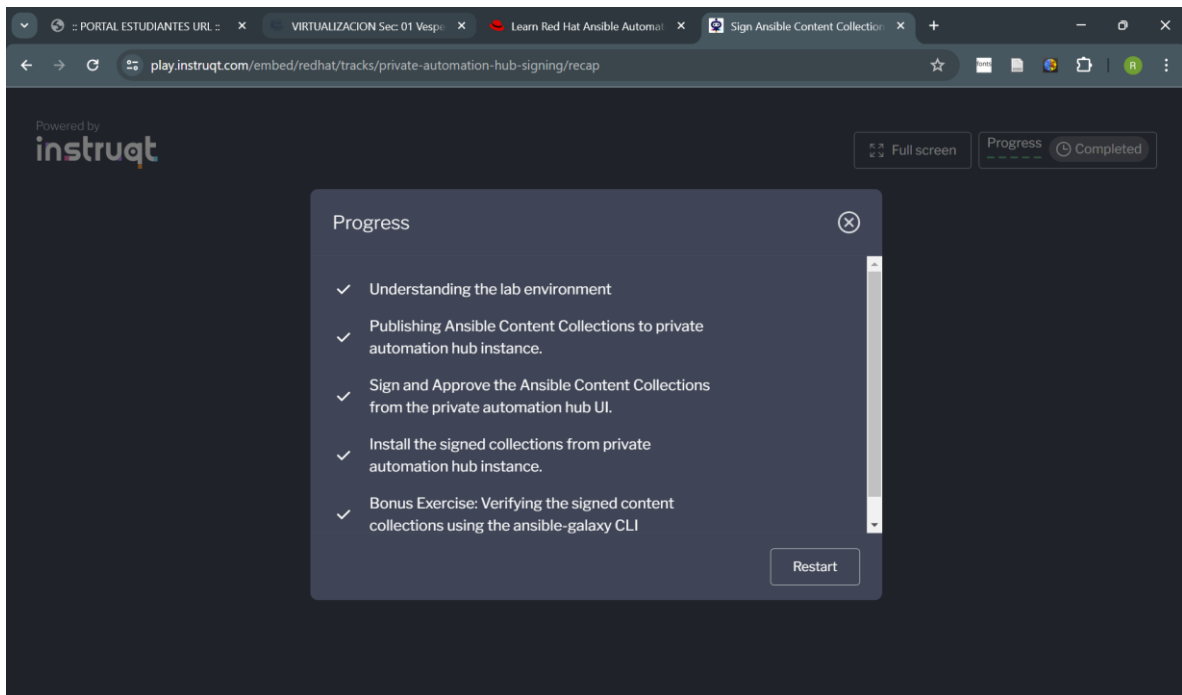


Ansible Completo

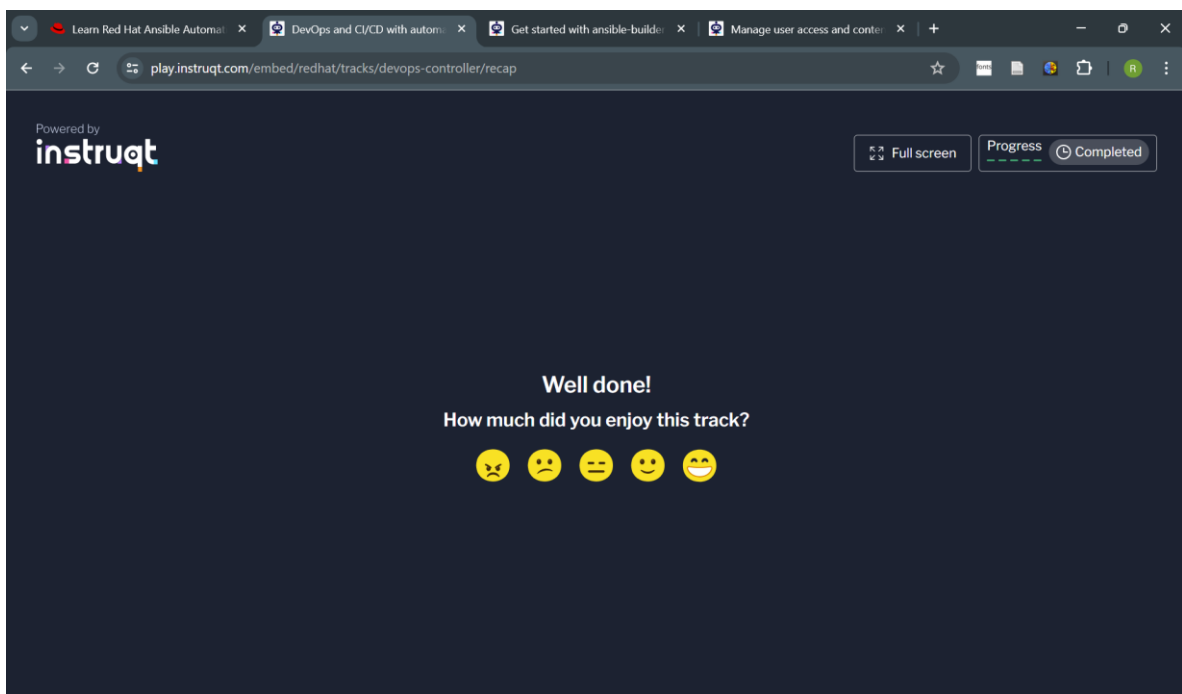
Sign Ansible Content Collections with private automation hub

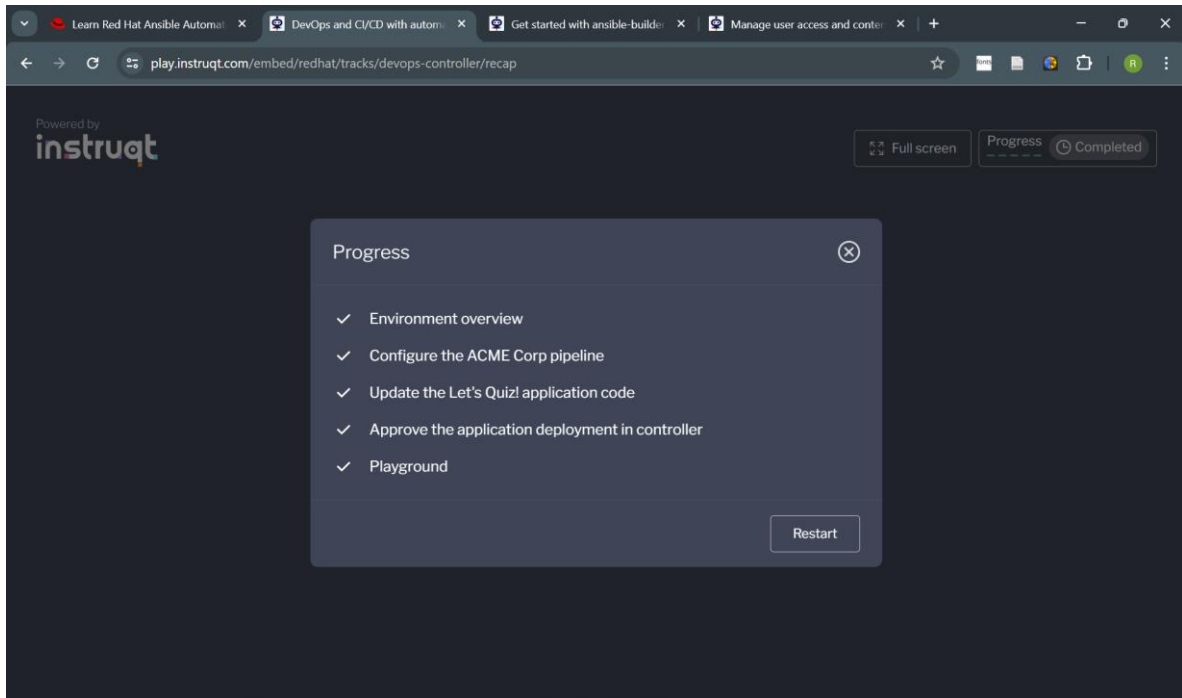
The screenshot shows a web browser window with the URL `play.instruqt.com/embed/redhat/tracks/private-automation-hub-signing/challenges/understanding-lab-environment/assignment`. The interface is split into two main sections. On the left, there is a terminal window titled `automationhub-terminal` showing a failed login attempt: `Last failed login: Tue Jun 7 14:43:19 UTC 2022 from 184.147.217.109 on ssh:notty. There was 1 failed login attempt since the last successful login. [rhel@privatehub-01 ~]$`. On the right, there is a panel titled `automationhub-web` displaying the 'Understanding the lab environment' challenge. The challenge text explains the importance of understanding the lab environment and lists two tabs: `automationhub-terminal` and `automationhub-web`. It provides the username `admin` and password `ansible123!`. A progress bar at the top right shows the challenge is partially completed. A green banner at the bottom right says 'Well done! Loading your next challenge...'.

The screenshot shows a web browser window with the URL `play.instruqt.com/embed/redhat/tracks/private-automation-hub-signing/recap`. The interface is dark-themed. At the top left, it says 'Powered by instruqt'. At the top right, there are buttons for 'Full screen' and 'Progress', with a 'Completed' status indicator. In the center, it says 'Well done!' and 'How much did you enjoy this track?'. Below this text are five yellow emoji icons: a sad face, a neutral face, a slightly smiling face, a happy face, and a very happy face. The progress bar at the top right shows the challenge is fully completed.

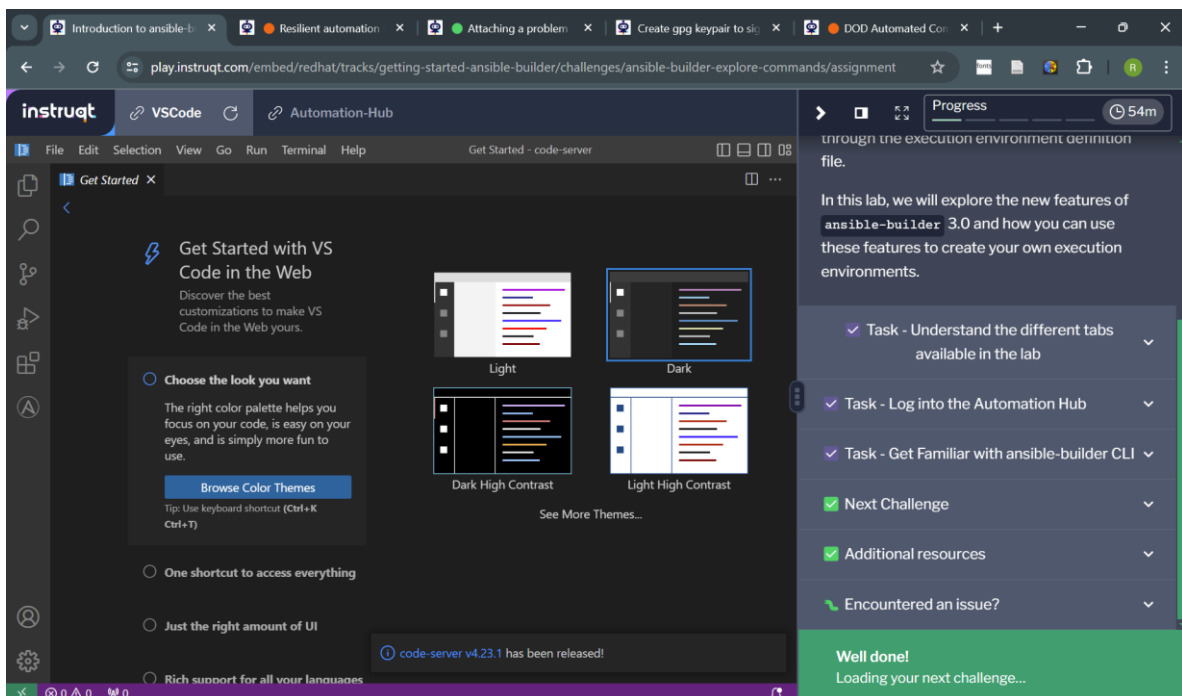


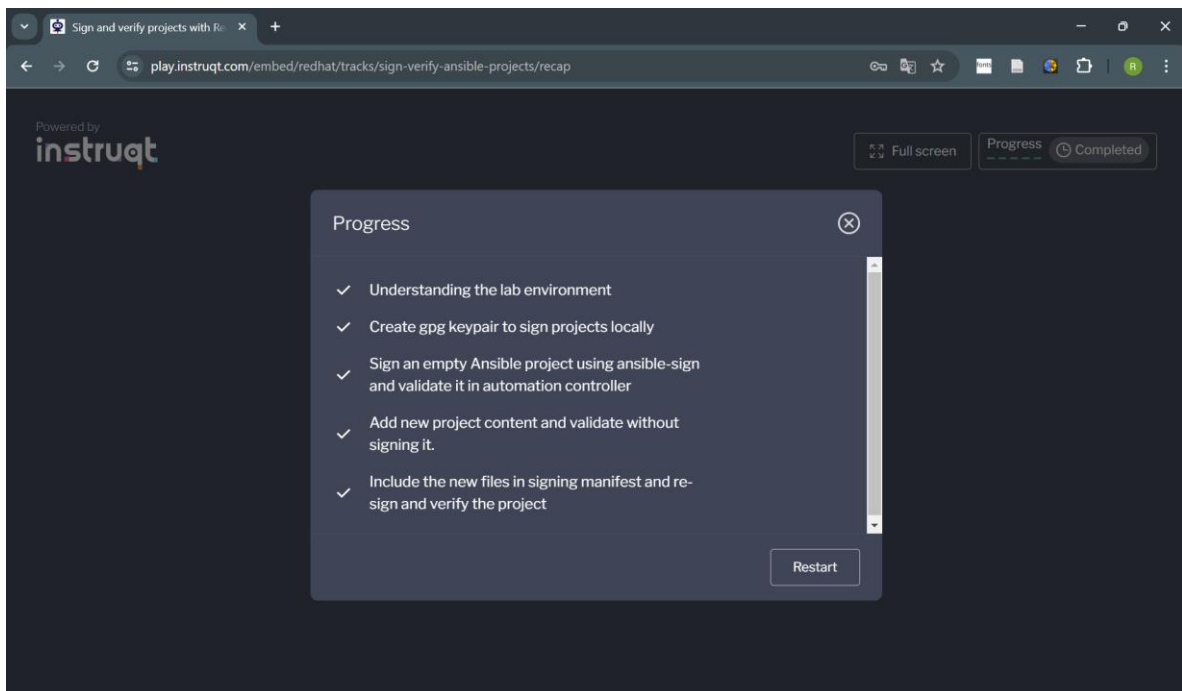
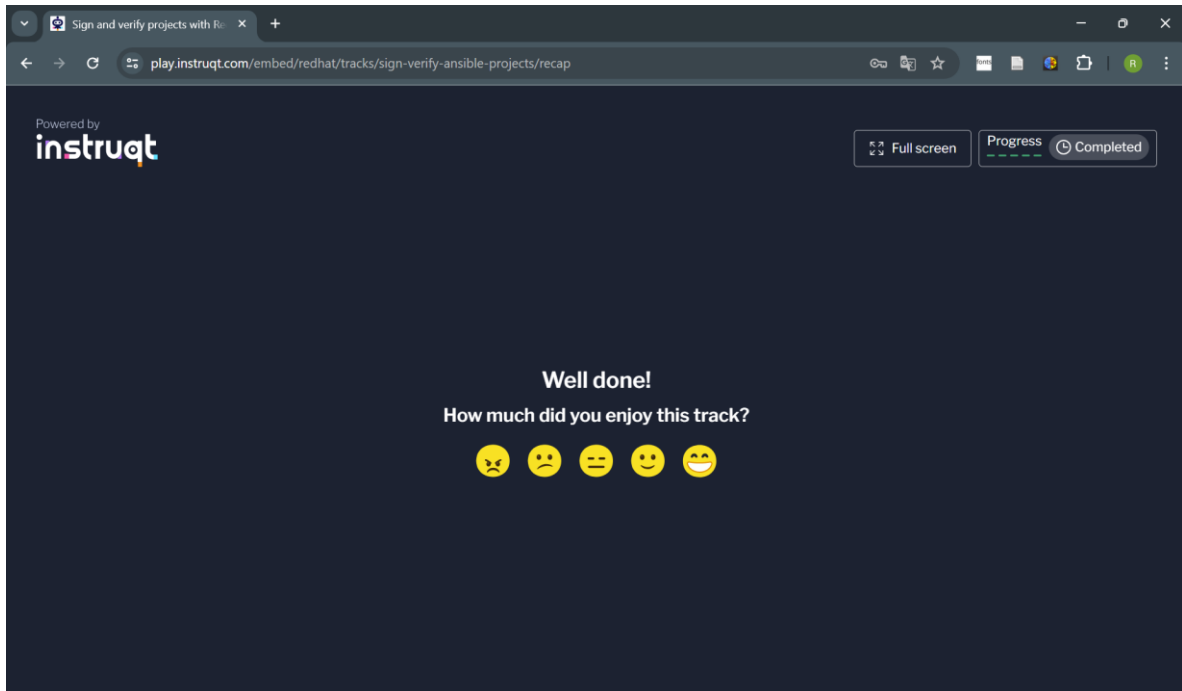
DevOps and CI/CD with automation controller



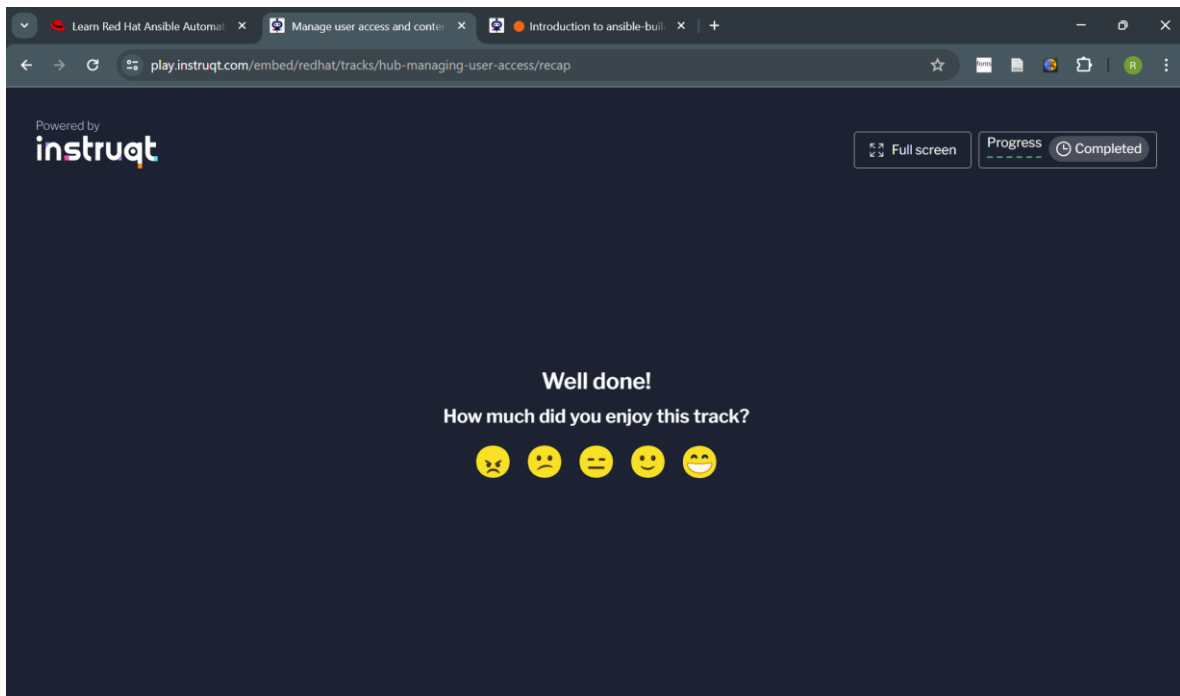


Get started with ansible-builder

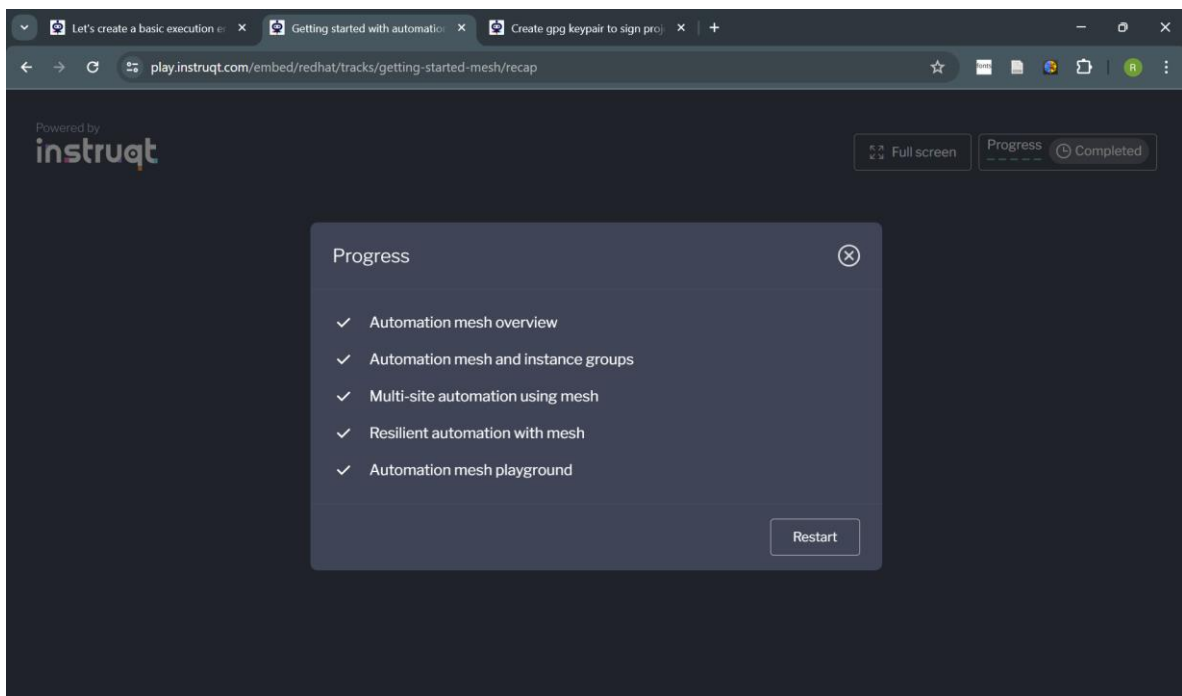
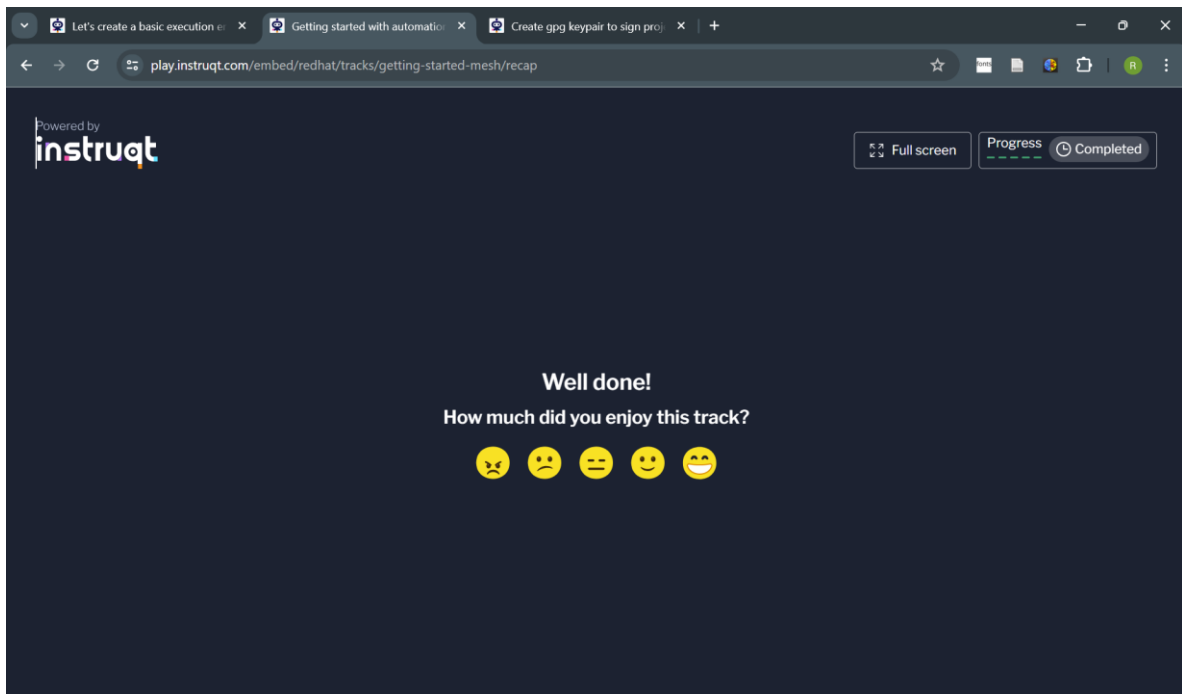




Manage user access and content policies using private automation hub



Getting started with automation mesh



Get started with ServiceNow automation

The screenshot shows the Instruqt interface for the 'Incident creation' challenge. The browser tabs include 'Introduction to ansible-l...', 'Multi-site automation us...', 'Incident creation - instru...', 'Create gpg keypair to sig...', and 'DOD Automated Cor...'. The URL is `play.instruqt.com/embed/redhat/tracks/getting-started-servicenow-automation/challenges/incident-creation/assignment`. The interface features a top bar with 'instruqt', 'VS Code', 'Automation Controller', and 'ServiceNow' tabs. A progress bar at the top right shows 'Progress' and a timer at '55m'. The main content area is titled 'Introduction' and contains the following text:

In ITIL, an incident refers to an unplanned outage or reduction in quality of an IT service or application. ServiceNow implements technology mapped to ITIL terminology and is accepted as an industry standard for incident management.

The servicenow.itsm certified collection allows organizations to leverage incident management within Ansible Automation Platform workflows.

A playbook has been created in the `vs code` tab called `incident-create.yml`.

- Inspect this playbook and review the in-line comments to understand how the collection is being leveraged.

Below the text is a button labeled 'Create incident'. At the bottom, a green banner reads 'Well done! Loading your next challenge...'.

The screenshot shows the Instruqt interface for the 'Attach a problem' challenge. The browser tabs include 'Let's create a basic execu...', 'Resilient automation', 'Attaching a problem - in...', 'Create gpg keypair to sig...', and 'DOD Automated Cor...'. The URL is `play.instruqt.com/embed/redhat/tracks/getting-started-servicenow-automation/challenges/attach-problem/assignment`. The interface features a top bar with 'instruqt', 'VS Code', 'Automation Controller', and 'ServiceNow' tabs. A progress bar at the top right shows 'Progress' and a timer at '54m'. The main content area is titled 'Introduction' and contains the following text:

A problem in ServiceNow represents the cause of one or more incidents. The root cause of the problem may not be known at the time of creation and may require root cause analysis through the problem management process.

Another playbook has been added to your working directory in VS Code and a template has been created to attach a problem to the incident previously created.

- First, inspect the playbook called `problem-attach.yml` in the `vs code` tab.

Below the text is a box with the following text:

Notice that this playbook will first query for existing incident numbers that you have created and will use the value returned in the task that creates the problem.

At the bottom, a green banner reads 'Well done! Loading your next challenge...'.

Let's create a basic execu... x Resilient automation with... x Query and update Servi... x Create gpg keypair to sig... x DOD Automated Cor... x + -

play.instruct.com/embed/redhat/tracks/getting-started-service-now-automation/challenges/configuration-items/assignment

instruct VS Code Automation Controller ServiceNow Progress 53m

Introduction

In ITIL, a configuration management database (CMDB) is a database that an organization uses to track hardware and software assets. ServiceNow CMDB is typically a critical service within an organization and is often seen as the single source of truth for tracking IT assets.

Ansible Automation Platform is able to leverage modules within the `servicenow.itsm` collection to both query and update configuration items within ServiceNow's CMDB.

For this challenge, two Red Hat Enterprise Linux (RHEL) servers have automatically been provisioned and added to an inventory on automation controller. Their hostnames are `node1` and `node2`. Two additional playbooks have been created in the `vs code` tab.

Well done!
Loading your next challenge...

Let's create a basic execu... x Resilient automation with... x Query and update recor... x Create gpg keypair to sig... x DOD Automated Compl... x + -

play.instruct.com/embed/redhat/tracks/getting-started-service-now-automation/challenges/query-and-update-records/assignment

instruct VS Code Automation controller ServiceNow Progress 53m

Introduction

It's now time to cleanup the records that were created as a part of this learning track. To do this, we're going to leverage different modules to locate records created by your ServiceNow user.

To do this, inspect the playbook `close-records-by-user.yml` in VS Code. There are a few new modules here. These `*_info` modules are being used to query for different record types (incident, problem, and change_request) for active records created by your username. Ansible then transforms these returned records into simple lists of objects and passes the lists to their respective modules to update/close the records. For any fields not implemented by the module itself, there is a module parameter called `other` that can be used to specify any other

Well done!
Loading your next challenge...

Do you trust the authors of the files in this folder?

code-server provides features that may automatically execute files in this folder.

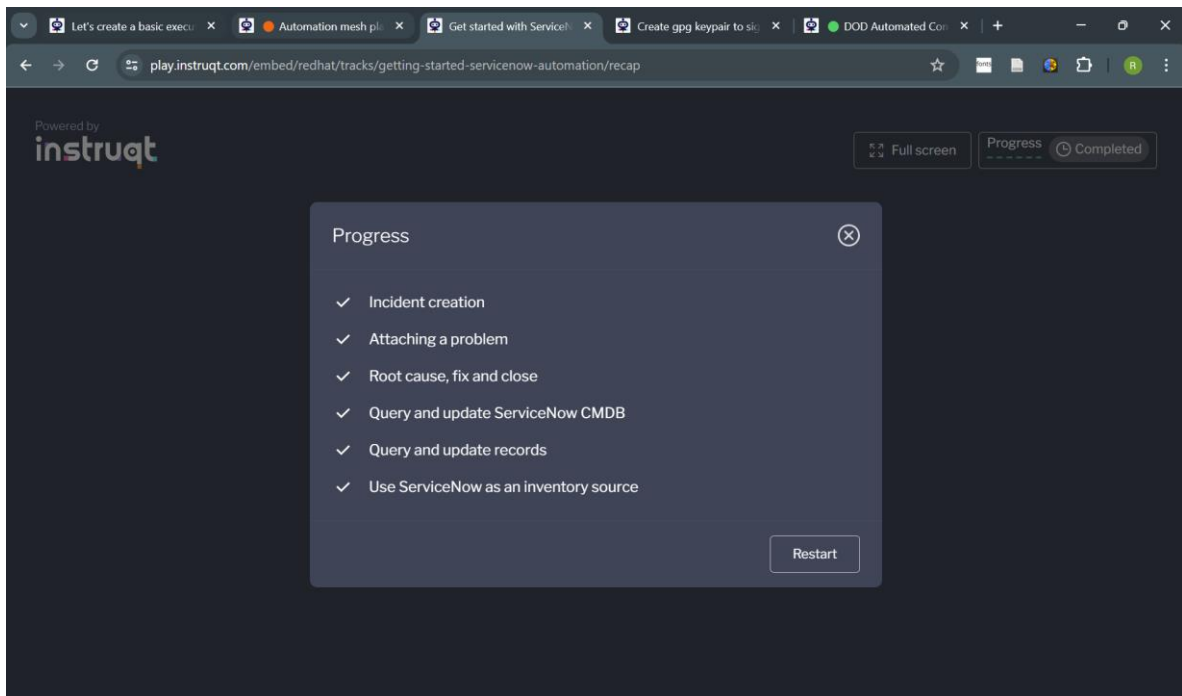
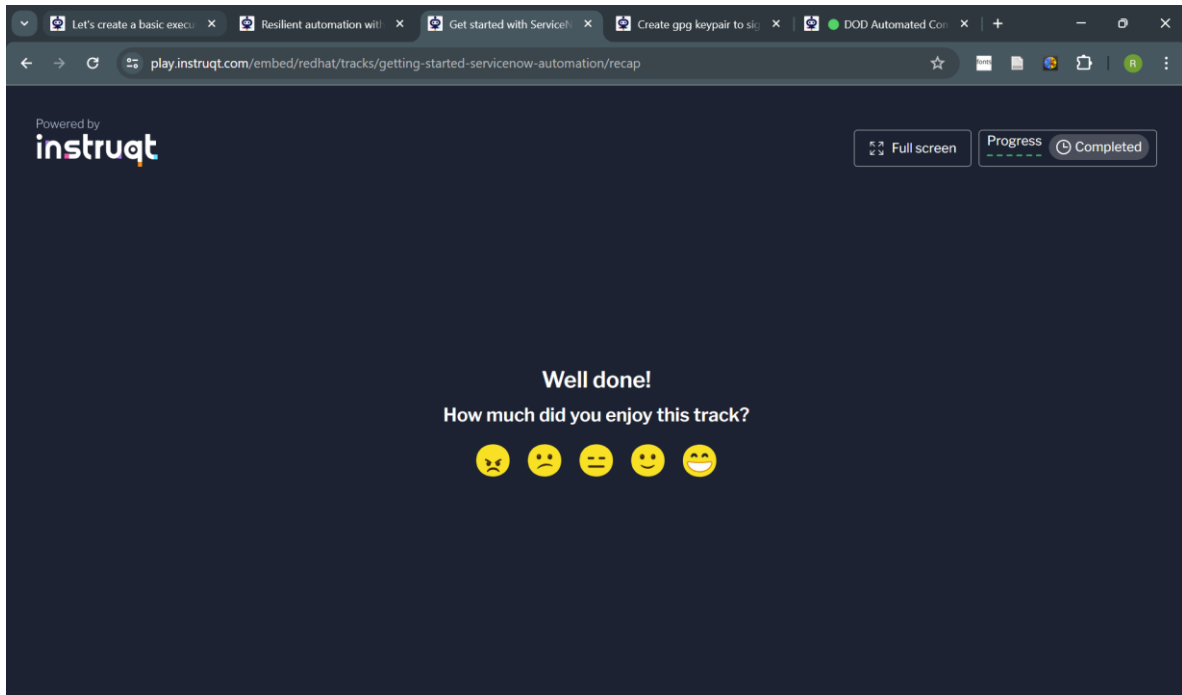
If you don't trust the authors of these files, we recommend to continue in restricted mode as the files may be malicious. See [our docs](#) to learn more.

~/servicenow_project

☐ Trust the authors of all files in the parent folder 'rhel'

Yes, I trust the authors
Trust folder and enable all features

No, I don't trust the authors
Browse folder in restricted mode



Sign and verify projects with Red Hat Ansible Automation Platform

The screenshot shows the Instruct lab interface for the challenge 'Understanding the lab environment'. The browser address bar shows the URL: `play.instruct.com/embed/redhat/tracks/sign-verify-ansible-projects/challenges/understanding-lab-environment/assignment`. The interface has a dark theme. On the left, there is a terminal window with the prompt `[rhel@controller ~]$`. On the right, there is a progress bar and a list of tabs: 'Terminal', 'Automation-Controller', and 'gitea'. The main content area on the right contains the following text:

Understanding the lab environment

It's important to understand the lab environment first before starting the challenges, in this lab you will see three tabs:

- **Terminal** - We will use this to run commands to setup the Ansible project and sign it using `ansible-sign`.
- **Automation-Controller** - This is the WebUI for automation controller and will be used to verify content. You can login to the WebUI using the below username/password combination.
 - Username: `admin`
 - Password: `ansible123!`
- **Gitea** - Gitea is a community managed lightweight code hosting solution written in Go. This will be used to host the Ansible

At the bottom right, there is a green box with the text: **Well done!** Loading your next challenge...

DOD Automated Compliance

The screenshot shows the Instruct lab interface for the challenge 'DOD Automated Compliance'. The browser address bar shows the URL: `play.instruct.com/embed/redhat/tracks/dod-automated-compliance/challenges/dod-cac-linux/assignment`. The interface has a dark theme. On the left, there is a terminal window with the prompt `[rhel@controller ~]$`. On the right, there is a progress bar and a list of tabs: 'Controller UI', 'VS Code', 'node1 web', and 'win1 web'. The main content area on the right contains the following text:

Introduction

In this section we will look at using Ansible for to ensure STIG compliance on Linux.

To get started, login to the Ansible Controller with the following credentials:

username: `admin`

password: `ansible123!`

Find detailed documentation and playbooks from this lab [here](#)

Below the introduction, there is a list of tasks:

- ✓ Task 1 - Setup Linux Demos
- ✓ Task 2 - Run STIG Job Template for Linux
- ✓ Task 3 - (optional) Run Additional Jobs

At the bottom right, there is a green box with the text: **Well done!** Loading your next challenge...

