UbuCon Korea 2023

퍼블릭 클라우드에서 Ubuntu Image 버전 관리

박 준 상 (JS Park, jsp@hashicorp.com)

실습 목표

- 클라우드 환경에서 사용되는 가상 서버 이미지(Ubuntu Image) 관리 방안을 다음 솔루션을 사용하여 알아봅니다.
  - 가상 서버 이미지 생성 자동화, 가상 서버 이미지 아티팩트 저장소, 클라우드 인프라 생명 주기 관리
- 위 도구들을 사용, 클라우드 환경에서 Ubuntu Image를 복수의 클라우드 환경에서 사용하는 방법을 실습합니다.

목차

- Who Am I
- #1. 클라우드 서버 이미지 생명 주기 관리
- #2.서버 이미지와 인프라 생명 주기 관리 연계
- #3. 실습

Who Am I

메인 프레임에서 클라우드 까지

Ops로 시작해 Dev를 만난 20년차 Pre-Sales 엔지니어

Posdata (POSCO EX)

Fujitsu FACOM System Admin (JCL)
HP-UX System Admin
WAS (OC4J) Admin, Java Developer
- MainFrame, HP-UX, AIX, Solaris

Megazone

AWS Pre-Sales for Samsung Electronics and subsidiaries
- **Ubuntu, RHEL**

HashiCorp Korea

Senior Solutions Engineer for Public Sector & Telco
- Ubuntu, Debian, RHEL, CentOS

Oracle

Pre-Sales for IAM Solutions, Collaboration Suite, Exalogic, Oracle Cloud Machine
Join Openstack Korea Community
- **Solaris, Oracle Linux, Ubuntu, Elementary**

RedHat

Account Solution Architect for Samsung Electronics and subsidiaries
- RHEL, Fedora

#1. 클라우드 서버 이미지 생명 주기 관리

수 작업 기반, 개별 클라우드 이미지 관리

AMI — Image artifact on AWS

VM Image — Image artifact on Azure

Machine image — Image artifact on VMware

Virtual Machine image — Image artifact on Openstack

Packer

모든 클라우드의 모든 이미지를 **구축**, **거버넌스** 및 **관리**하기 위한 인프라 자동화 표준

코드형 이미지 (Images as Code)

고객화

Ansible
Bash
Powershell
...

다양한 클라우드

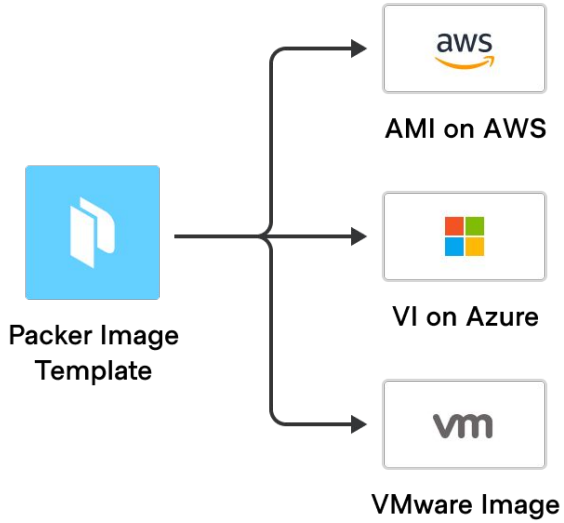
AWS
Azure
GCP
...

프라이빗 DC

VMware
VirtualBox
Vagrant
...

Packer, 단일 워크 플로우 기반 클라우드 이미지 통합 관리

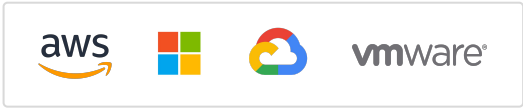
- **사용 편의성**
  HCL을 사용, Terraform처럼 손쉬운 연동.
- **확장성**
  사용 환경과 무관하게 단일 언어로 이미지 관리.
  타 구성 관리 도구와 연계.
- **Community 기반**
  기존 커뮤니티 템플릿을 활용 및 직접 작성.



Packer, 핵심 기능

1 Builder

다양한 클라우드 및 온프리미엄
인프라에 대한 이미지 생성을 담당



2 Provisioner

기본 제공 소프트웨어 및 타사 소프트웨어를
사용, 이미지를 설치하고 구성할 수 있도록
시스템을 준비

3 Post-Processor

이미지 생성 후 작업 실행.
아티팩트를 업로드하거나 다시 패키지.

4 Data Source

구성 시 외부 데이터 사용.

서버 이미지 관리 표준화를 위한 사전 준비 사항은?

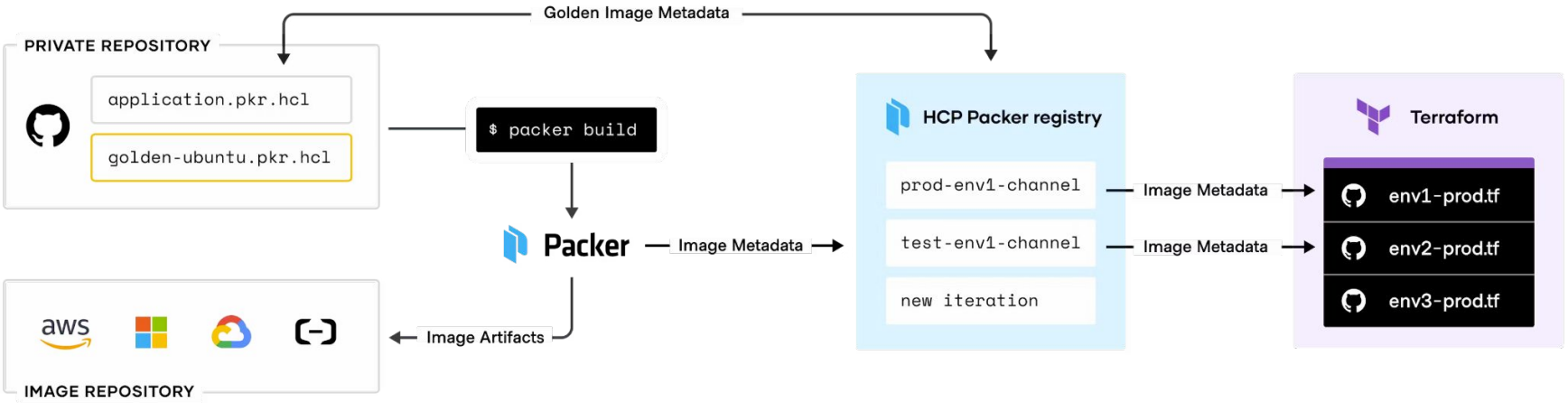
어떤 것이 "골드" 이미지이고 어떤 버전을 사용해야 하는가?

이미지 하드닝, 보안 및 규제 준수를 위한
이상적인 워크플로우는?
클라우드 전반에 걸쳐 복제할 수 있는가?

현재 프로비저닝 워크플로우에 **이 프로세스를 통합할 수 있는가?**

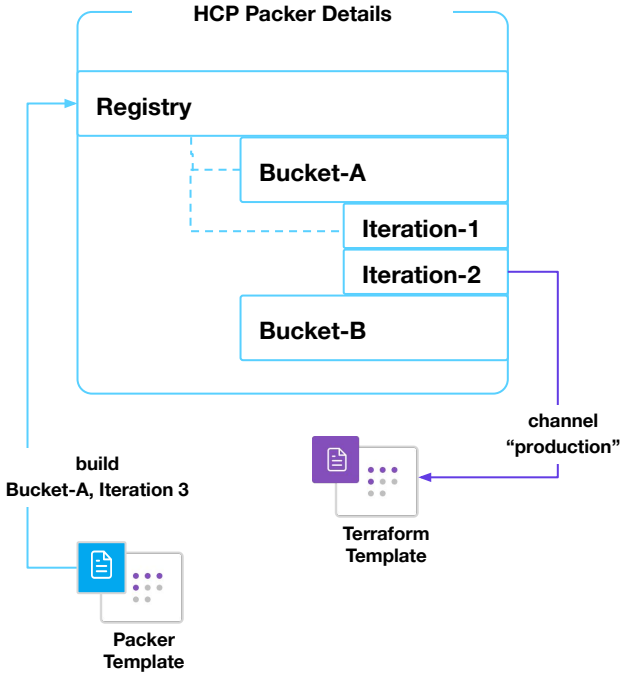
HCP Packer, 코드 기반 표준화를 통한 자동화

- **HCP Packer**는 프로비저닝 파이프라인에서 **이미지 업데이트를 표준화, 보안 및 자동화**하기 위한 멀티 클라우드 이미지 레지스트리입니다.
- **멀티 클라우드 골든 이미지 파이프 라인의 토대를 제공**

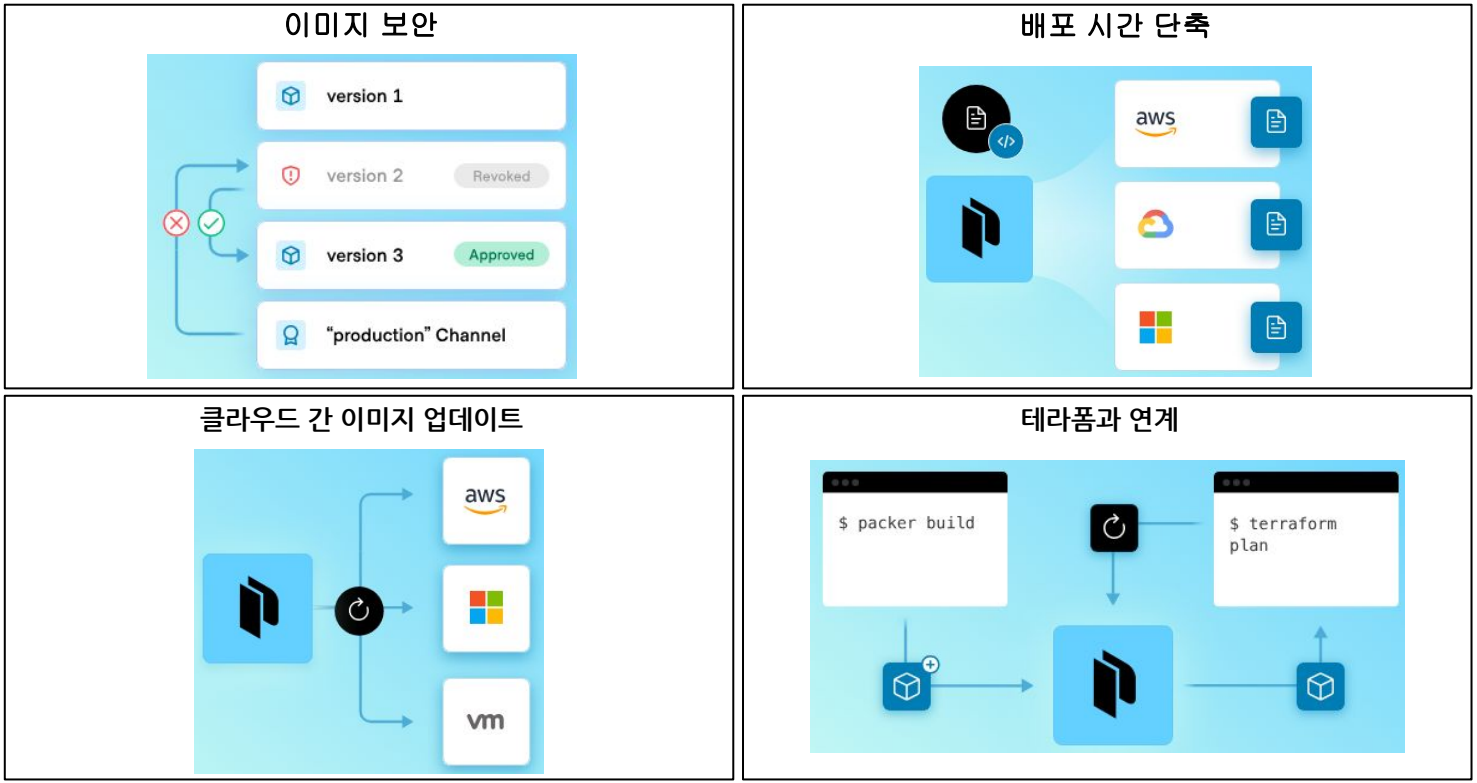


HCP Packer 구성 요소

- **Registry** : 이미지 관련 메타 데이터 저장 서비스. 모든 이미지 버킷 조회 가능
- **Bucket** : 단일 **Packer 탬플릿** 이미지 메타데이터 저장소
- **Iteration** : 성공한 개별 Packer 이미지 빌드 정보를 저장. **(=버전)**.
- **Channel** : 버킷 내 사용하고자 하는 이미지 의 기본 버전**(Iteration)**을 지정. Packer 템플릿과 Terraform Template에서 사용 가능. 버전 관리 표준화.



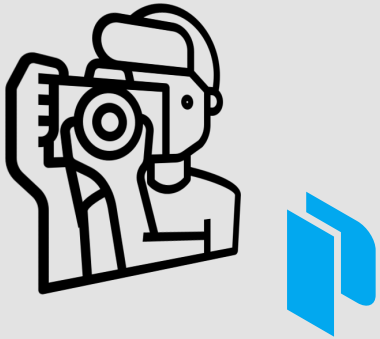
HCP Packer, 주요 쓰임새



Packer vs. HCP Packer

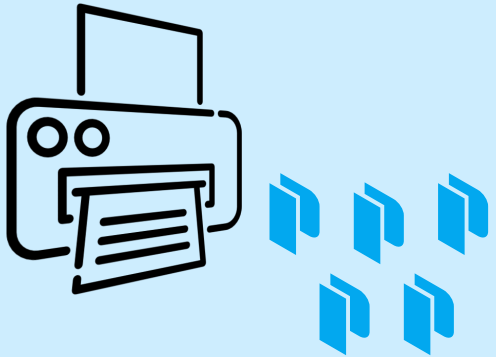
Packer

- 조직에서 준수해야할 요구 사항을 충족하는 이미지 생성/구축
- 필요 시 업데이트를 통한 이미지 유지 관리



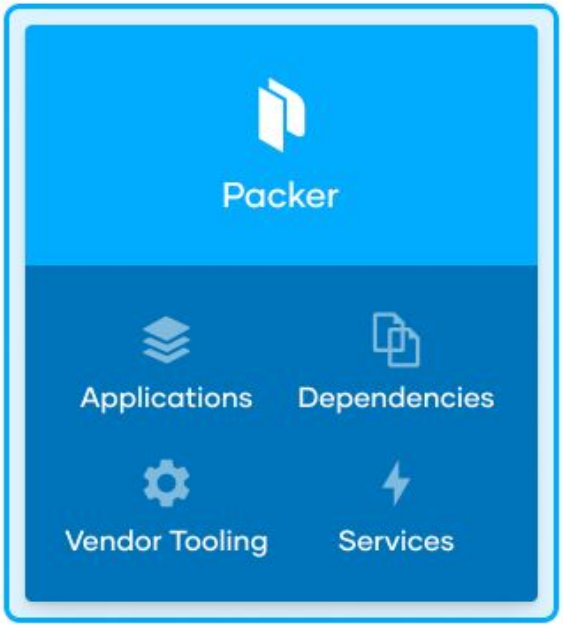
HCP Packer

- Packer로 빌드한 이미지 메타 데이터 저장
- 단일 소스 템플릿을 사용, 여러 플랫폼에 동일한 머신 이미지의 빠른 배포

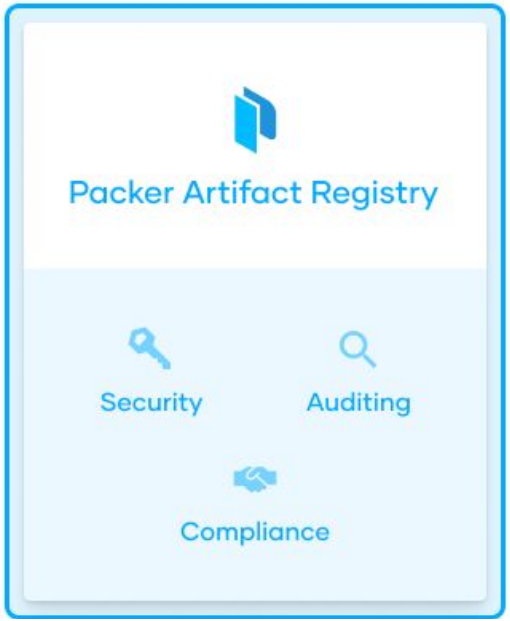


#2. 서버 이미지와 인프라 생명 주기 관리 연계

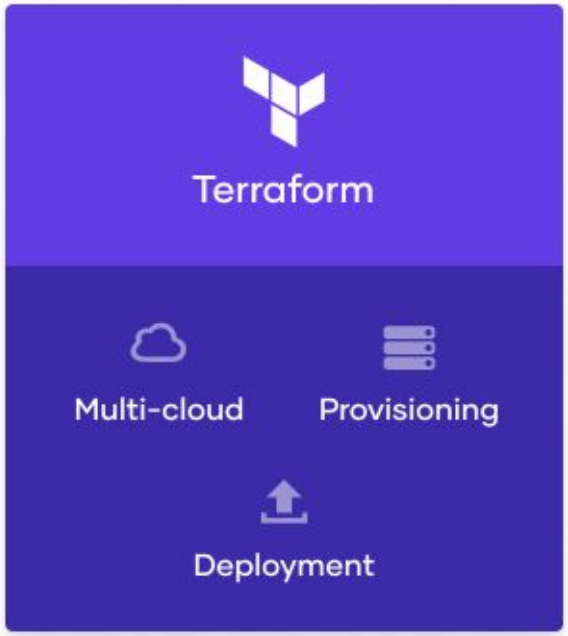
서버 이미지 배포 파이프 라인



Packer 커뮤니티 버전
서버 이미지 아티팩트 생성



HCP Packer
다양한 클라우드 환경에서 서버 이미지 생성, 추적 관리 및 거버넌스 과정의 복잡도 완화



Terraform(*)
서버 이미지 아티팩트를 이용,
자원 생명 주기 관리(배포/변경/삭제)

테라폼, 인프라 생명 주기 관리 자동화

코드형 인프라(IaC)를 통한 클라우드 인프라 구성, 규제 준수 및 관리 자동화 솔루션

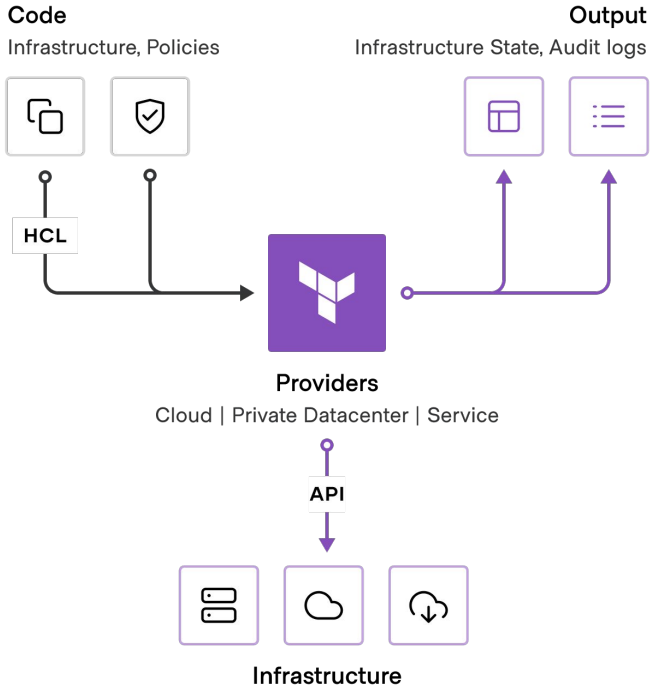
- **클라우드 관리 및 규정 적용**
  단일 Workflow를 통해 인프라 프로비저닝과 Compliance 관리를 실현

- **인프라 셀프 서비스 기반**
  사용자(개발자)가 직접 승인된 인프라 모듈 라이브러리를 사용, 시의적절하게 인프라를 쉽게 프로비저닝 할 수 있도록 지원

- **코드 기반 서비스 재해 복구 (DR)**
  추가 비용없는 RTO 준수 및 빠른 장애 복구

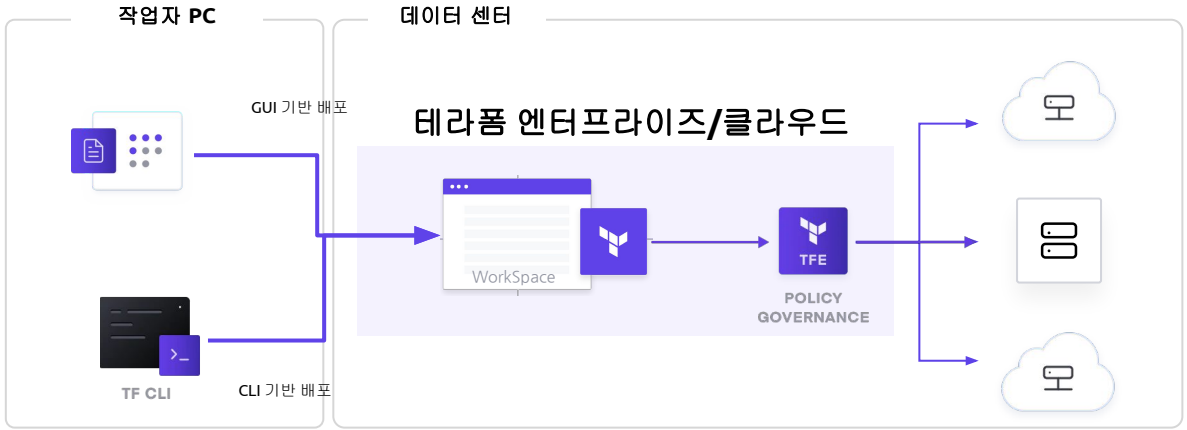
- **Business benefit**

  - 자동화를 통한 장애 최소화 / 운영 효율성 향상
  - 개발자 생산성 향상 / Time-to-market
  - 자원의 효율적인 활용을 통한 비용 절감



테라폼 엔터프라이즈(설치형, 관리형) 동작 방식

GUI/API 또는 CLI 사용



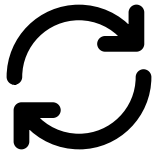
1. 코드 작성	2. 작업 준비	3. 구축 계획 검토	4. 정책 적용	5. 인프라 구축	6. 구축 결과 저장
configuration template	terraform init	terraform plan	코드화된 정책 적용	terraform apply	terraform.tfstate

Terraform Cloud + HCP Packer

개선된 클라우드 기반 서버 이미지 관리 방안 제공

HCP Packer

- **Build** 서버 이미지 생성
- **Publish** 메타 데이터 저장
- **Promote** 채널 기반 관리
- **Revoke** 원복 및 삭제
- **View** 참조 및 이력 조회

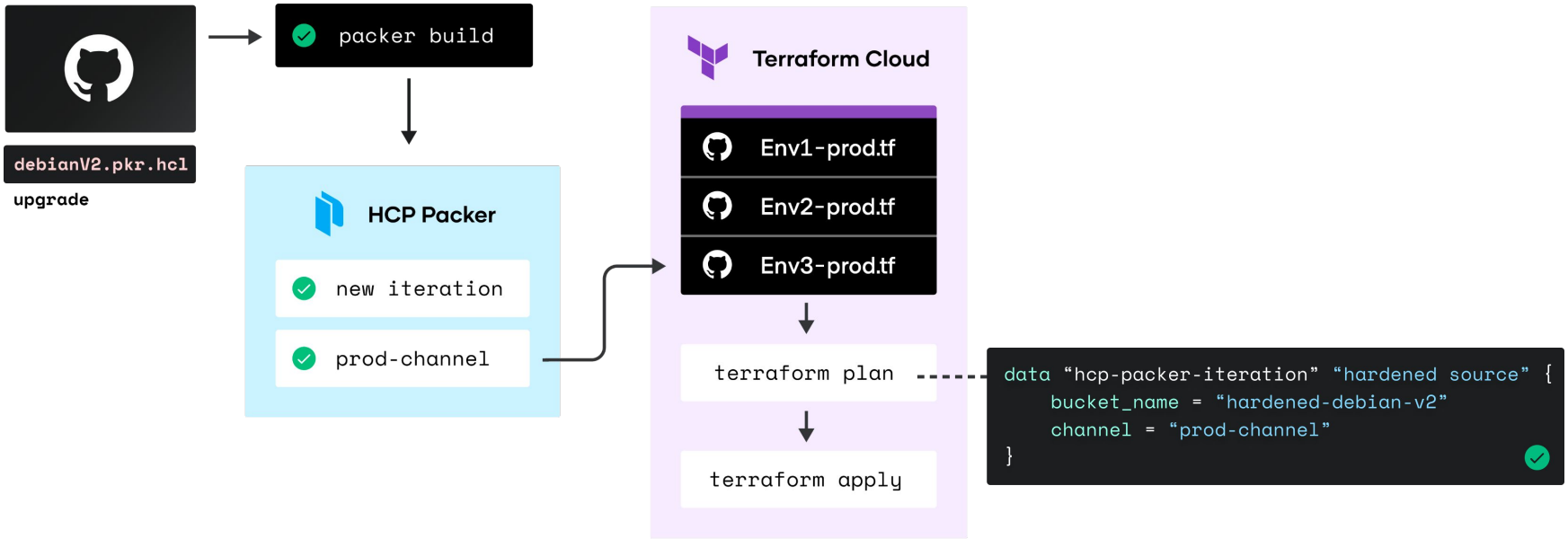


Terraform Cloud

- **Discover** 대상 이미지 검색
  Data source
- **Provision** 인프라 배포
- **Validate** 이미지 상태 검증

Terraform Cloud + HCP Packer

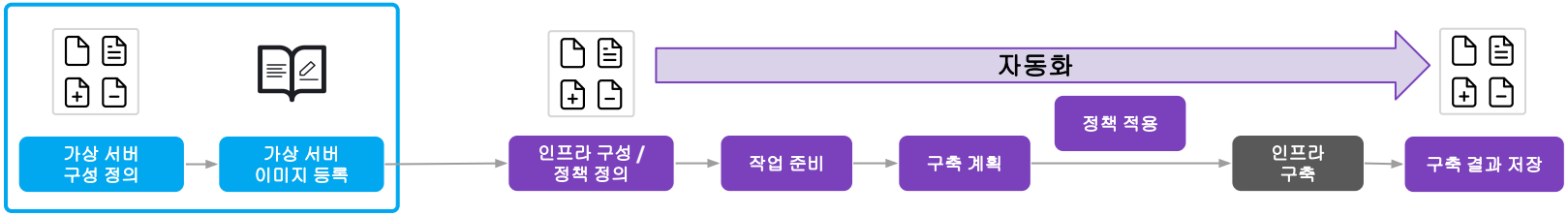
통합된 서버 이미지 생성, 배포 워크 플로우 제공

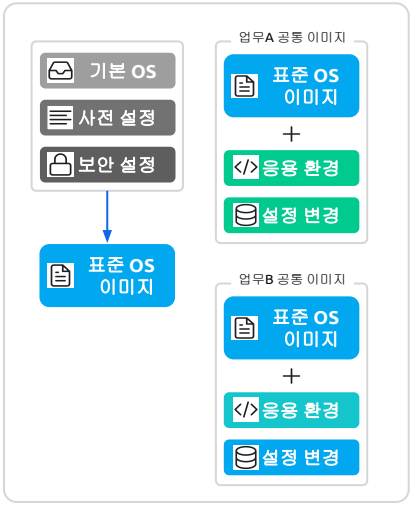


기대 효과

- 표준화된 인프라 형상 관리
- 인프라 관리 작업 방식 표준화
- 운영 표준 및 정책 적용 자동화

#1. 표준화된 인프라 형상 관리

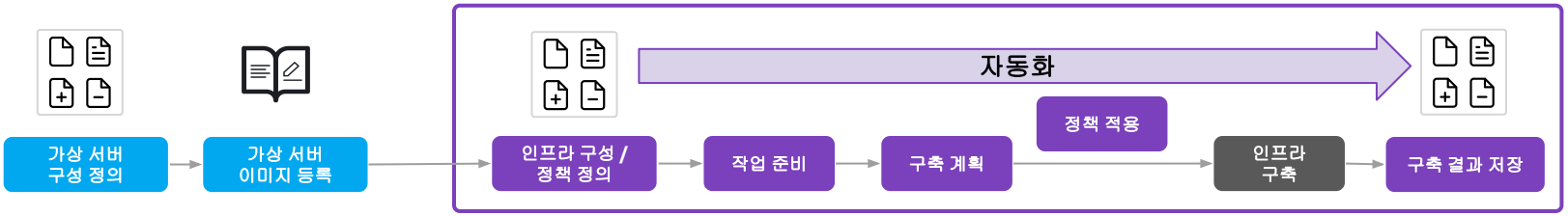




- 조직별/업무별 가상 **서버 형상을 코드로** 표준화
- 필요한 소프트웨어 설치, 보안 조치 등의 작업을 가상 **서버 배포 전** 완료
- 보안 및 OS 패치처럼 이미지 변경 시, 공통 이미지를 변경 후 적용

  → 구성이 상이하여 발생할 수 있는 **서비스 중단 및 장애 예방**

#2. 인프라 관리 작업 방식 표준화



인프라 구축 작업

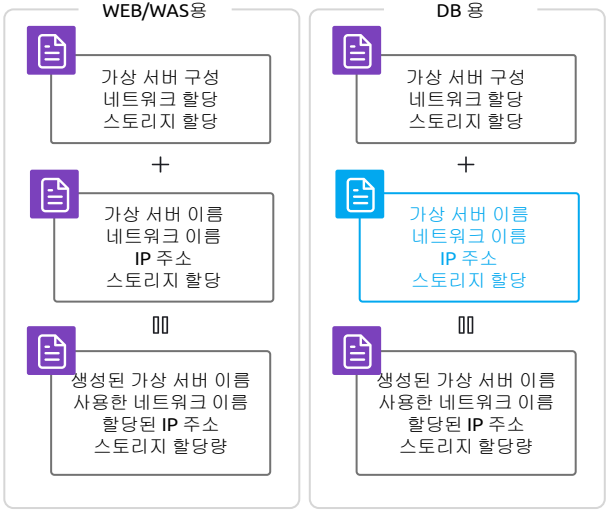
전체 코드의 **90% 이상**

→ 인프라 규모와 무관하게
재사용성이 높음

변수값 지정

환경 별로 다른 값을 변수화

전체 코드의 **10% 이내**

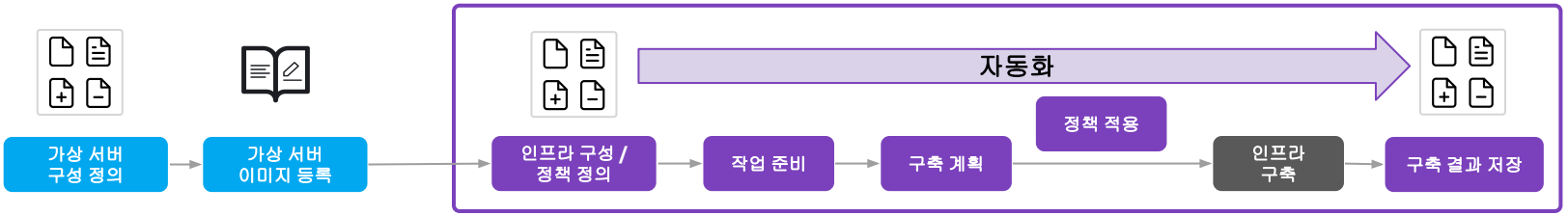


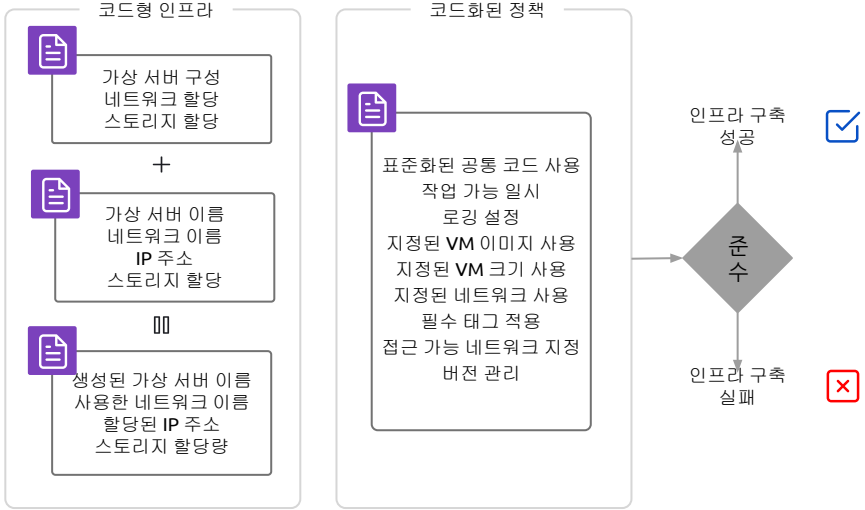
- 인프라 구성을 모두 코드로 관리.
- 수작업 최소화

  → 구성 오류로 인한 서비스 장애 최소화
- 단순 반복 작업 코드화

  → 작업 시간 및 인건비 절감

#3. 운영 표준 및 정책 적용 자동화



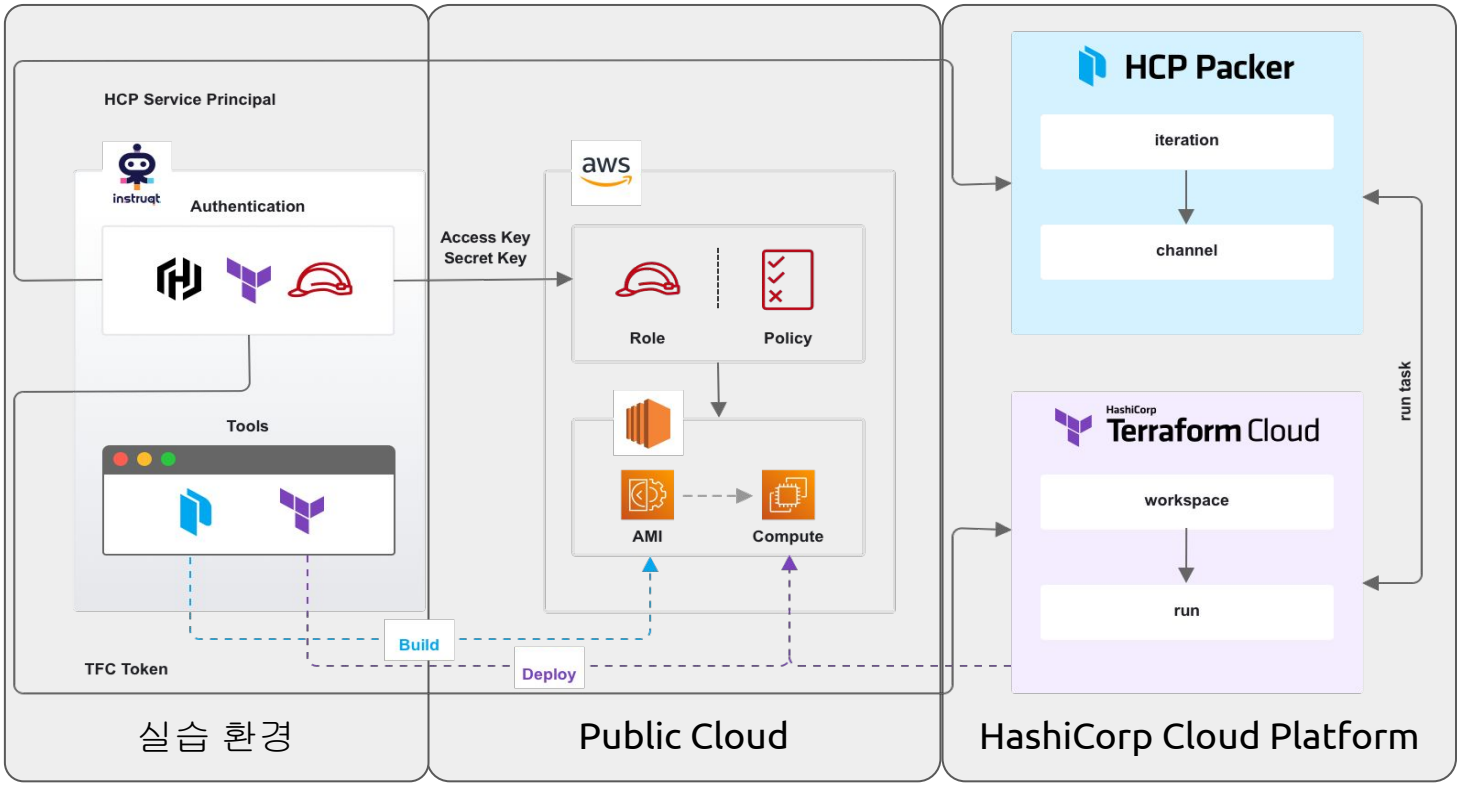


- 인프라 구성 시 보안 적용 강화
- 인프라 배포 시 코드화된 정책을 적용

  → 보안 표준 및 운용 표준에 부합하는 경우에만 인프라를 구성.

#3. 실습

실습 아키텍처


HCP Service Principal
instruqt
Authentication
Tools
TFC Token
Access Key
Secret Key
aws
Role
Policy
AMI
Compute
Build
Deploy
HCP Packer
iteration
channel
HashiCorp
Terraform Cloud
workspace
run
run task
실습 환경
Public Cloud
HashiCorp Cloud Platform

실습 흐름

1	**Runbook**을 패커 템플릿으로	• 작업 절차가 정리된 Runbook을 패커 템플릿화 • 이미지 생성 작업 시간 단축 • 작업 표준화로 오류 최소화
2	골든 이미지 기반 배포 **(AWS → Azure)**	• 골든 이미지를 AWS AMI로 저장 • 저장된 이미지를 사용, AWS에 가상서버 배포 • 이미지를 재사용, Azure로 서비스 확장
3	실습 환경 정리	• CSP 내 가상 서버 삭제 • 필요 시 HCP 내 환경 삭제 및 정리 • Shadow IT 예방

#0. 사전 준비 작업

1. 실습 환경 계정 생성 및 준비
2. HashiCorp Cloud Platform 계정 생성 및 환경 준비
3. Terraform Cloud 계정 생성 및 환경 준비

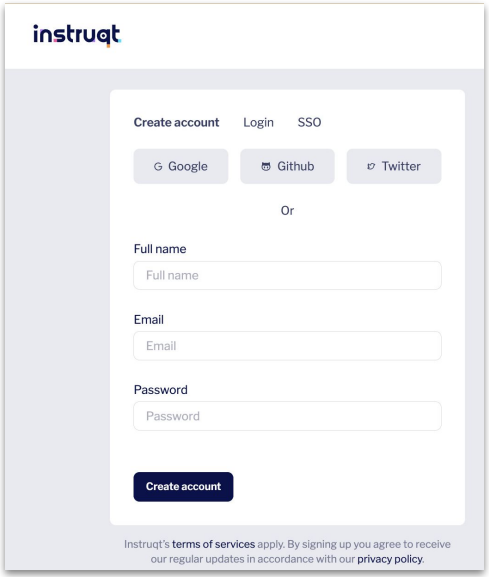
#0. 사전 준비 작업

1. 실습 환경 계정 생성 및 준비
2. HashiCorp Cloud Platform 계정 생성 및 환경 준비
3. Terraform Cloud 계정 생성 및 환경 준비

#0. 사전 준비 작업

1.실습환경 계정생성 및 준비

- 실습 환경 계정 생성

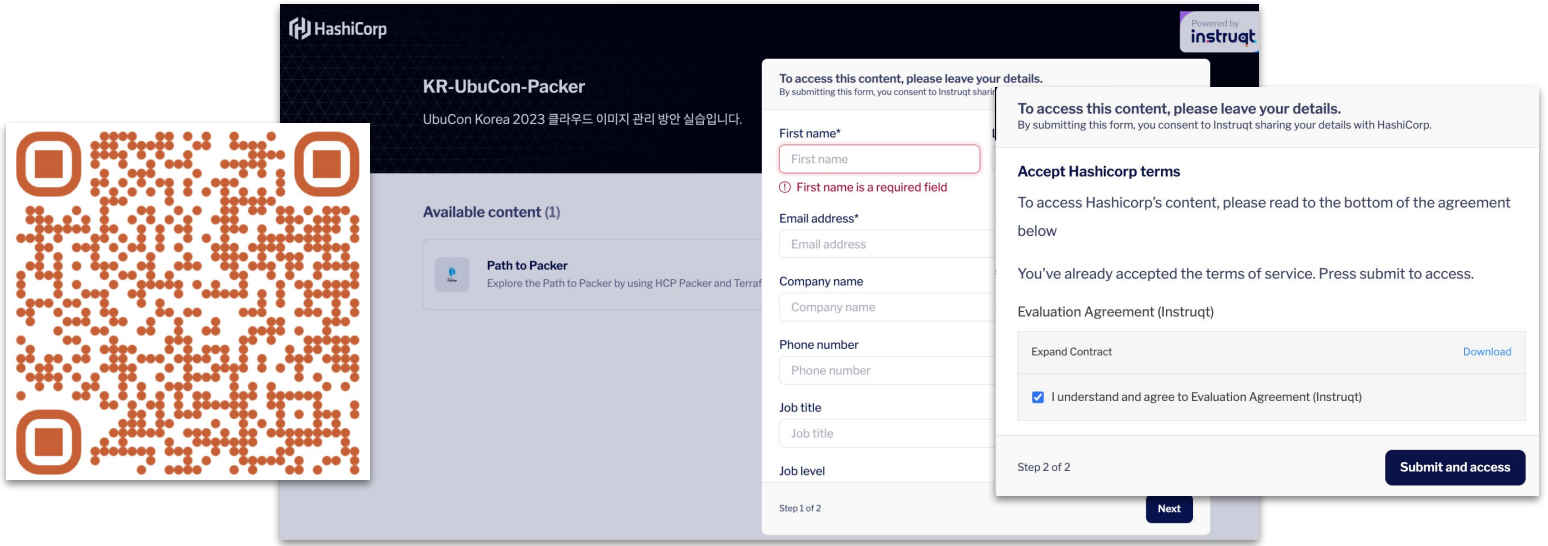


https://play.instruqt.com/signup

#0. 사전 준비 작업

1.실습환경 계정생성 및 준비

- 실습 환경 준비 : 다음 URL에 접속, 실습 환경 사용을 위한 동의서 제출

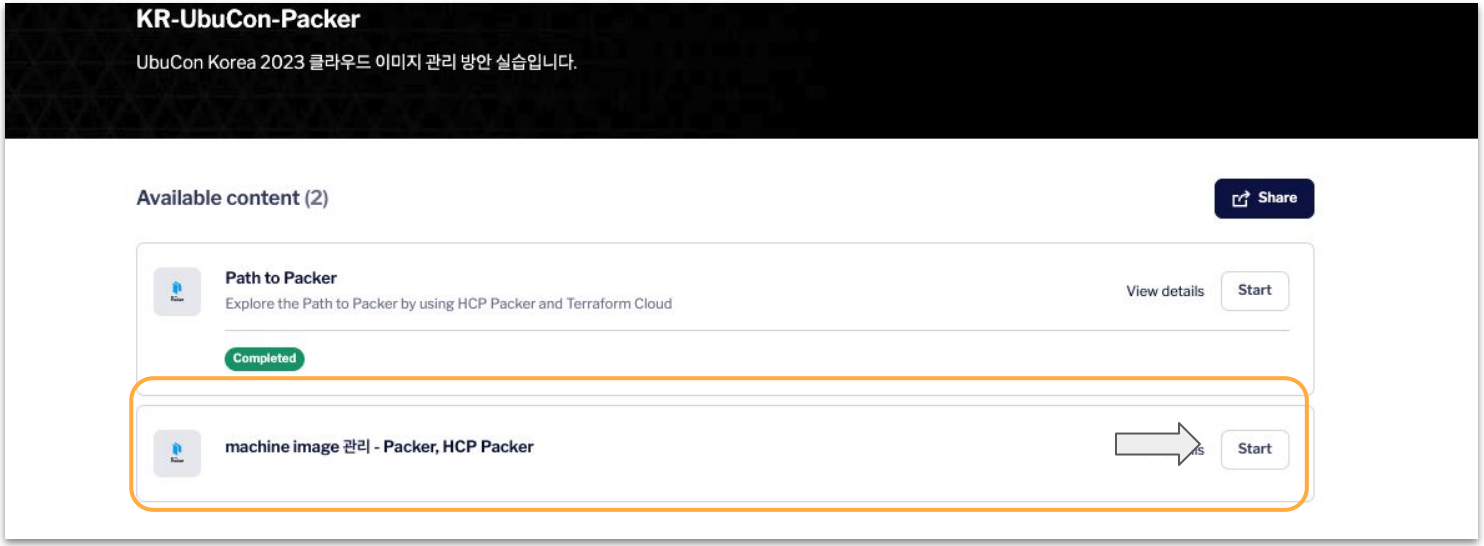


https://play.instruqt.com/hashicorp/invite/4njvh83uzqo7

#0. 사전 준비 작업

1.실습환경 계정생성 및 준비

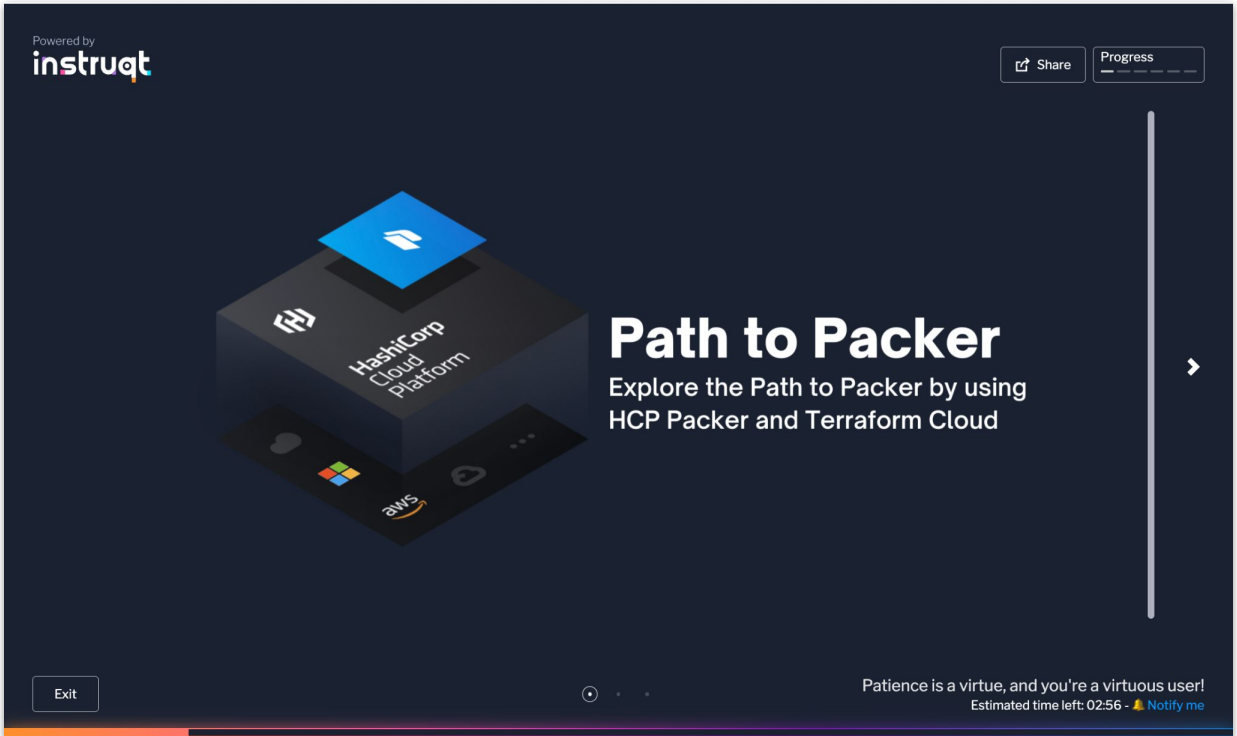
- 실습 환경 준비 : ‘Start’ 버튼 클릭



#0. 사전 준비 작업

1.실습환경 계정생성 및 준비

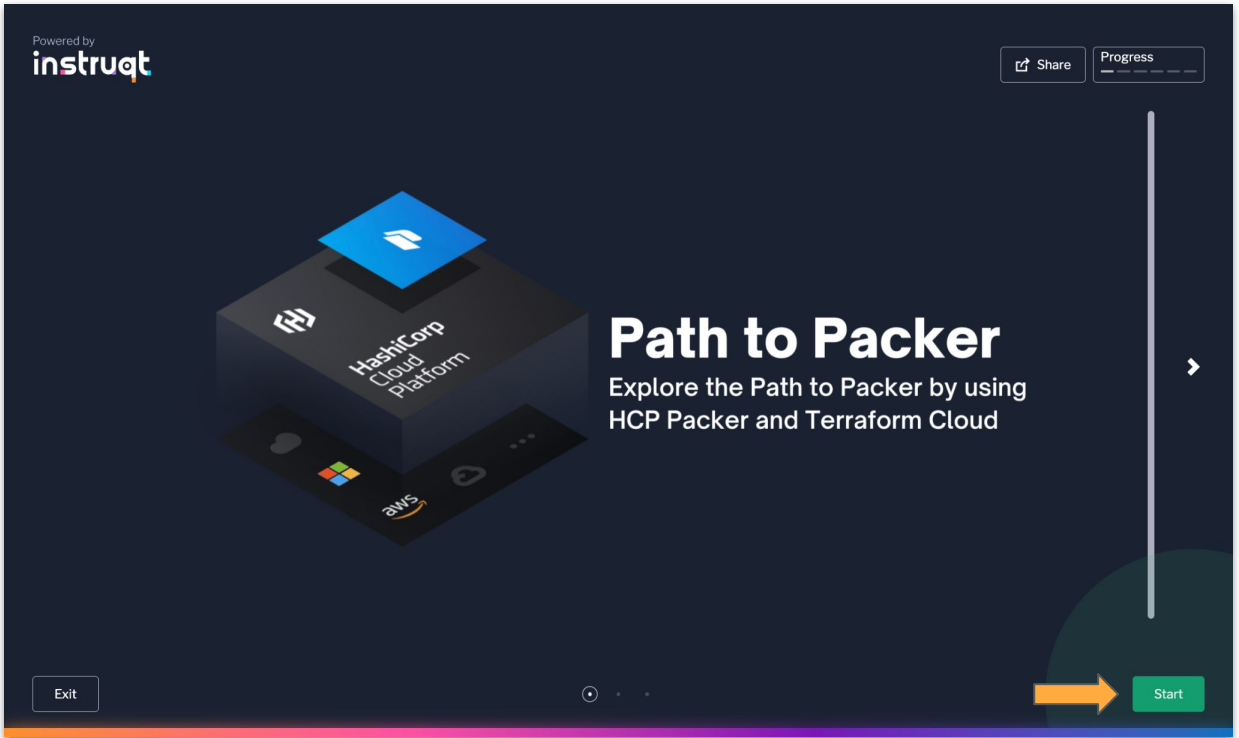
- 실습 환경 준비 :



#0. 사전 준비 작업

1.실습환경 계정생성 및 준비

- 실습 환경 준비 완료 : “Start” 버튼이 활성화 (잠시 대기). 브라우저 다른 탭/창 열기



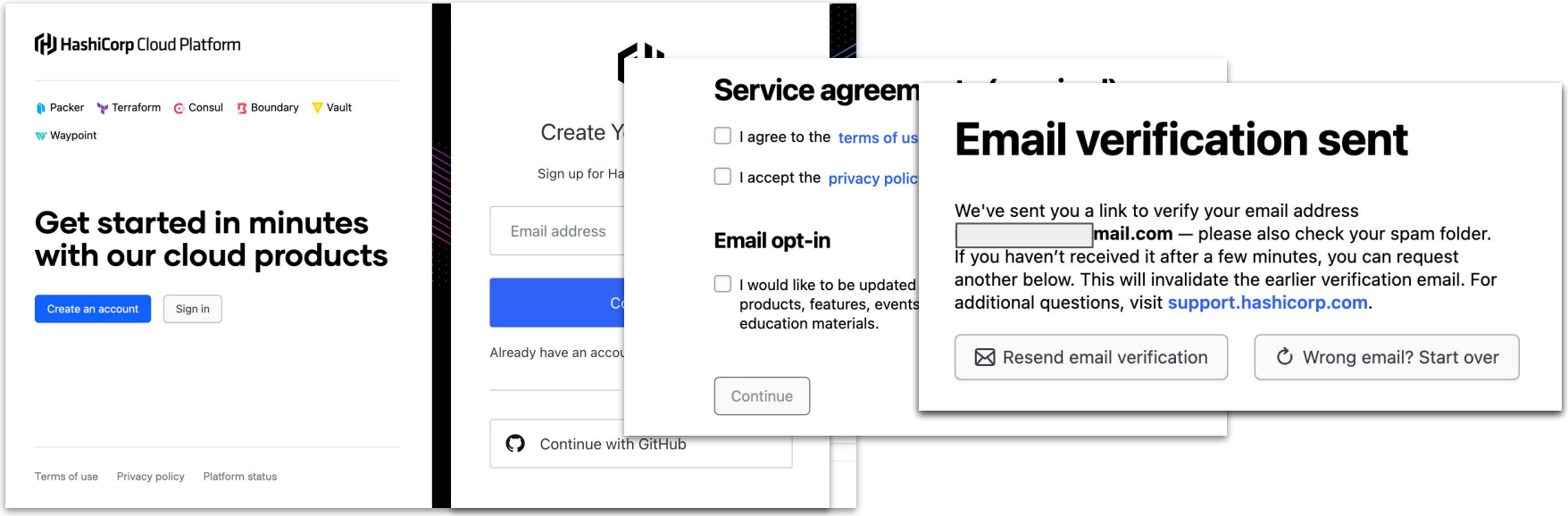
#0. 사전 준비 작업

1. 실습 환경 계정 생성 및 준비
2. HashiCorp Cloud Platform 계정 생성 및 환경 준비
3. Terraform Cloud 계정 생성 및 환경 준비

#0. 사전 준비 작업

2.HashiCorp Cloud Platform 계정 생성 및 환경 준비

- HashiCorp Cloud Platform 계정 생성

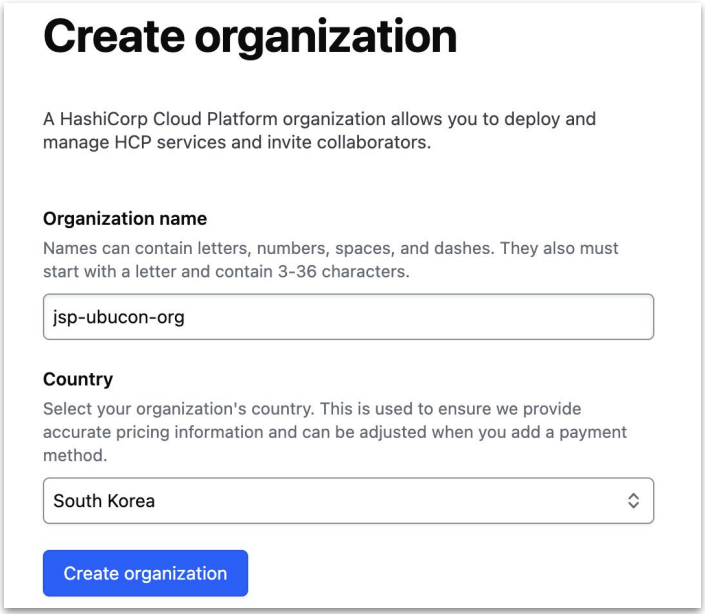


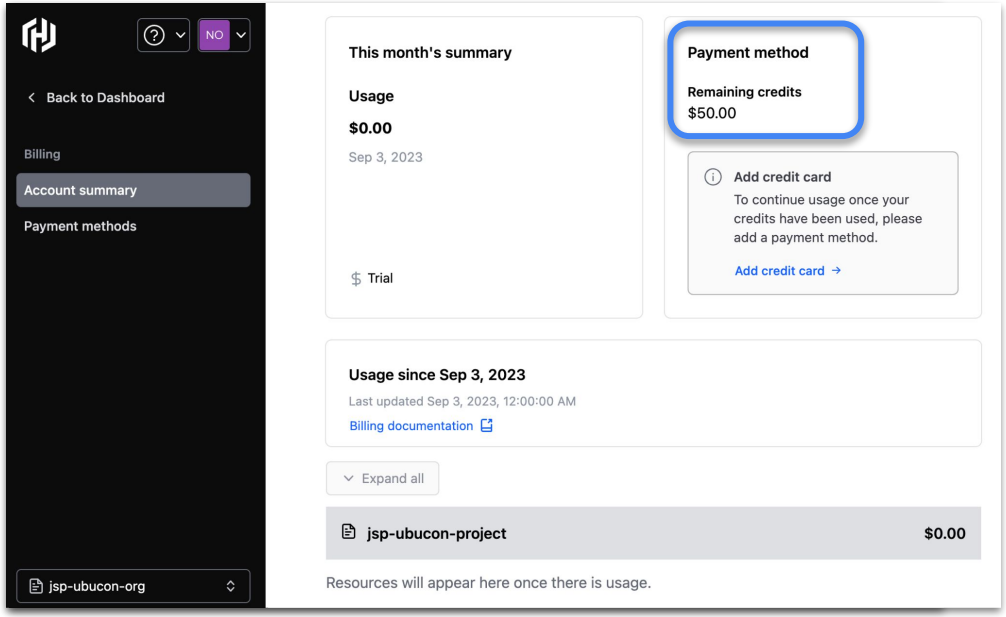
https://portal.cloud.hashicorp.com/sign-up

#0. 사전 준비 작업

2.HashiCorp Cloud Platform 계정 생성 및 환경 준비

- 이메일 인증 링크를 클릭하여 접속

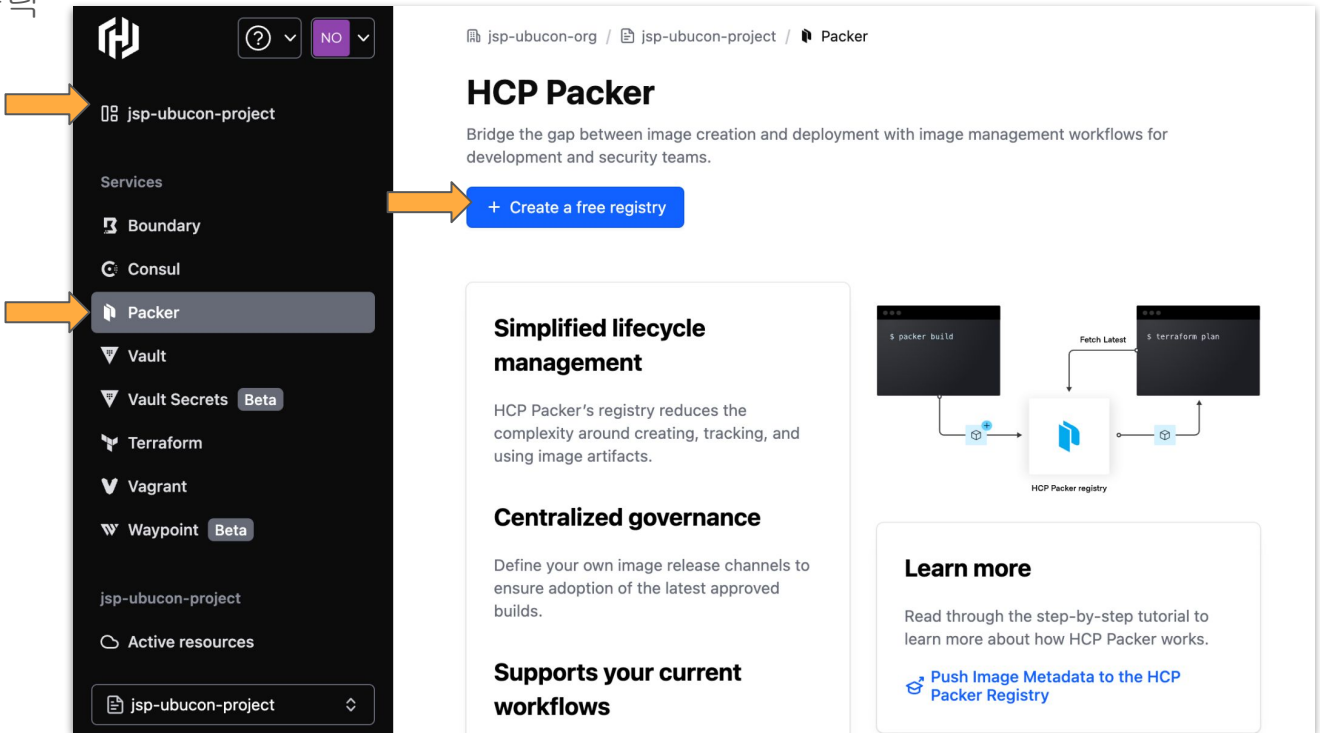




#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비

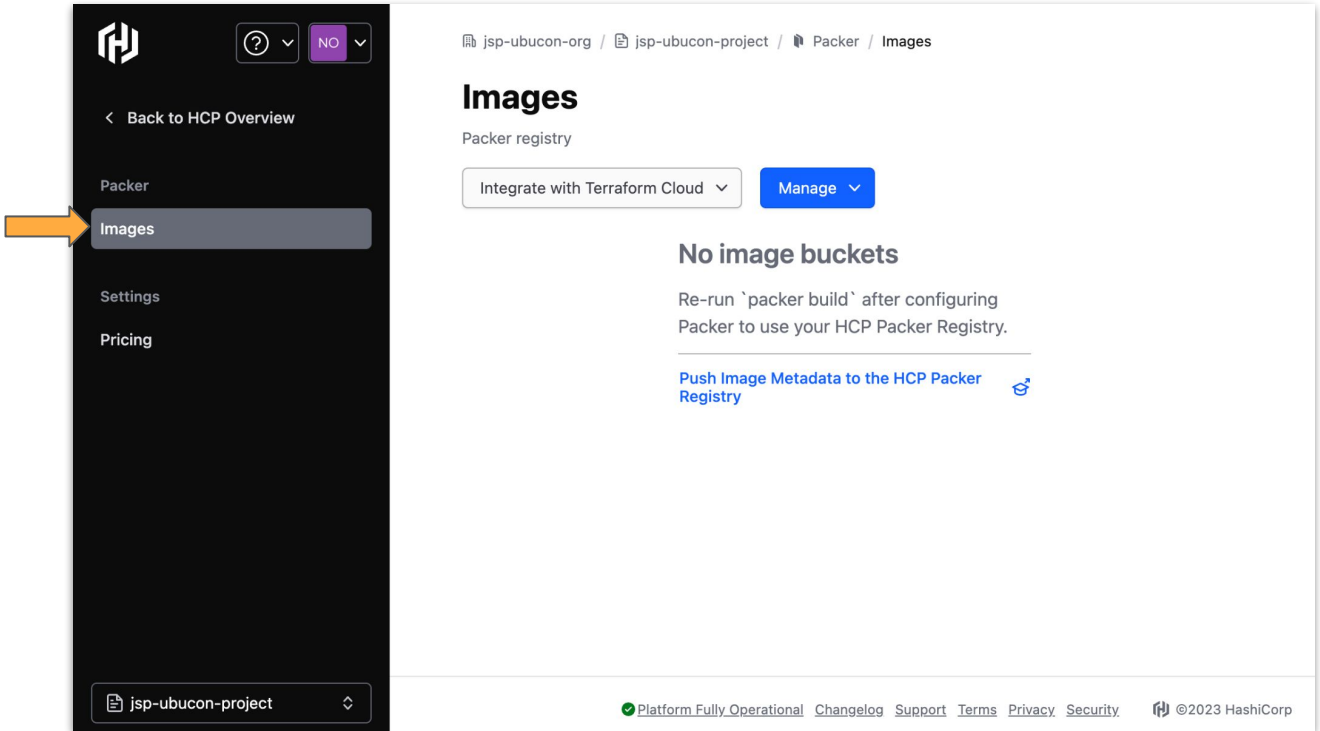
- HashiCorp Packer Registry 생성 : HCP 내 프로젝트 접속 > Packer 선택 > ‘Create a free registry’ 클릭



#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비

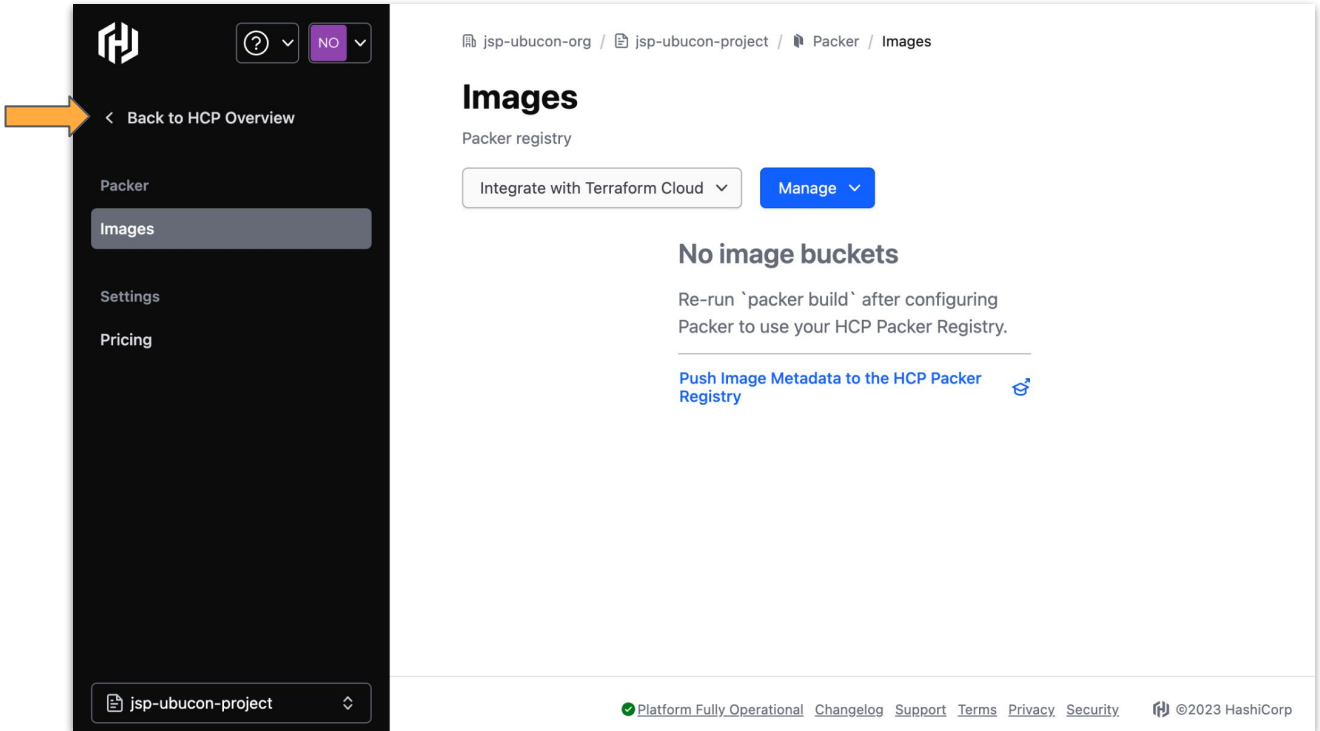
- Image Registry 생성 완료



#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비

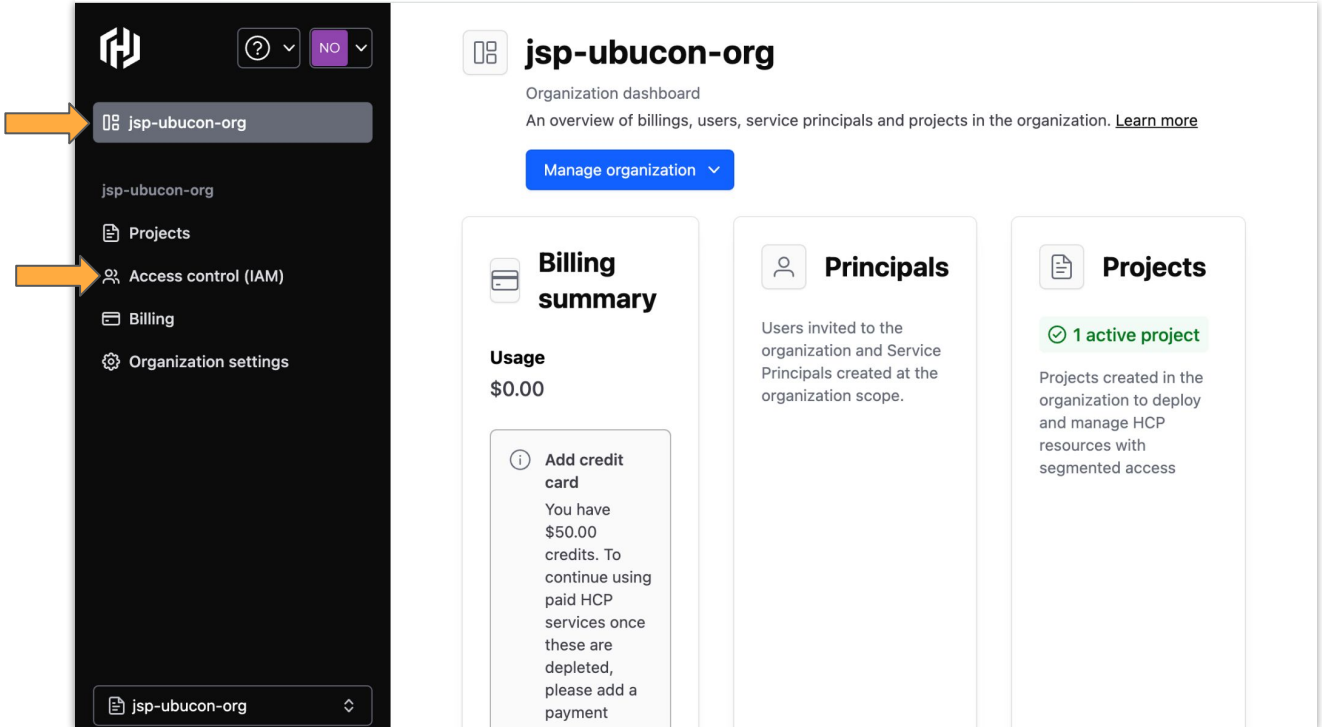
- Service Principal 생성 : ‘Back to HCP Overview’ 클릭



#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비

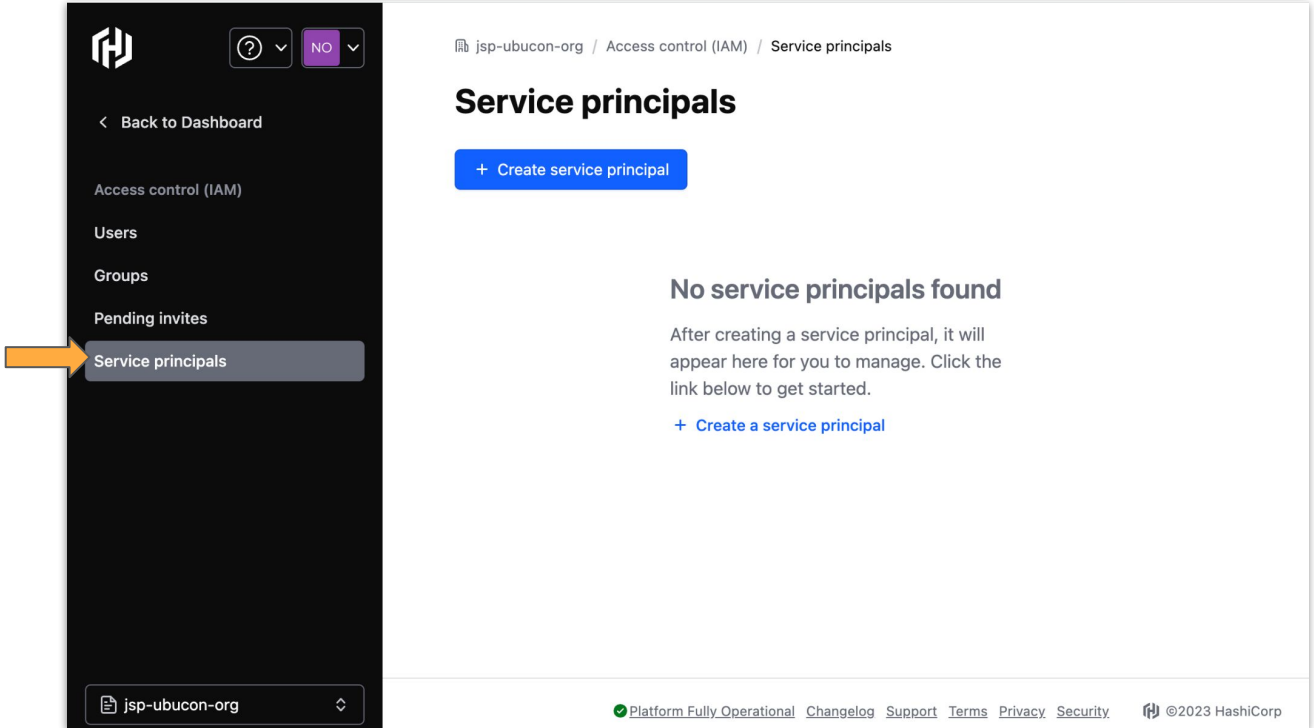
- Service Principal (Org Level) 생성 : “Org 선택 후" ‘Access Control (IAM)’ 선택



#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비

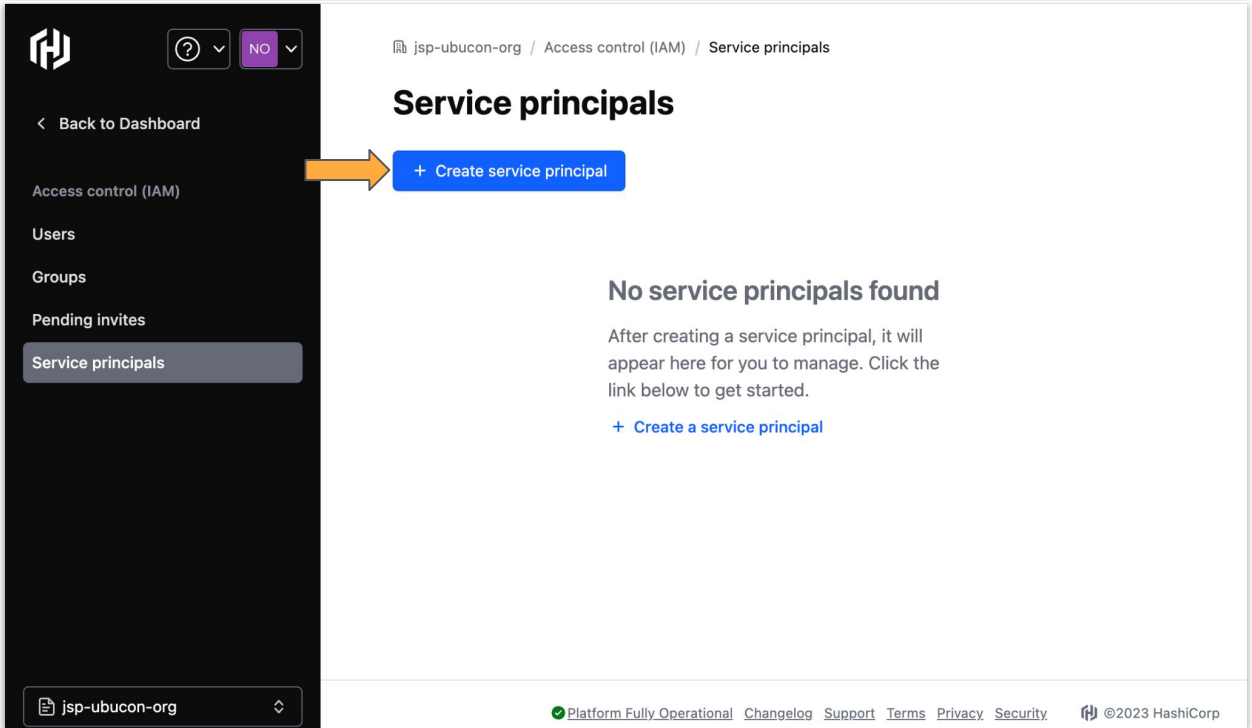
- Service Principal (Org Level) 생성 : ‘Service Principals’ 선택



#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비

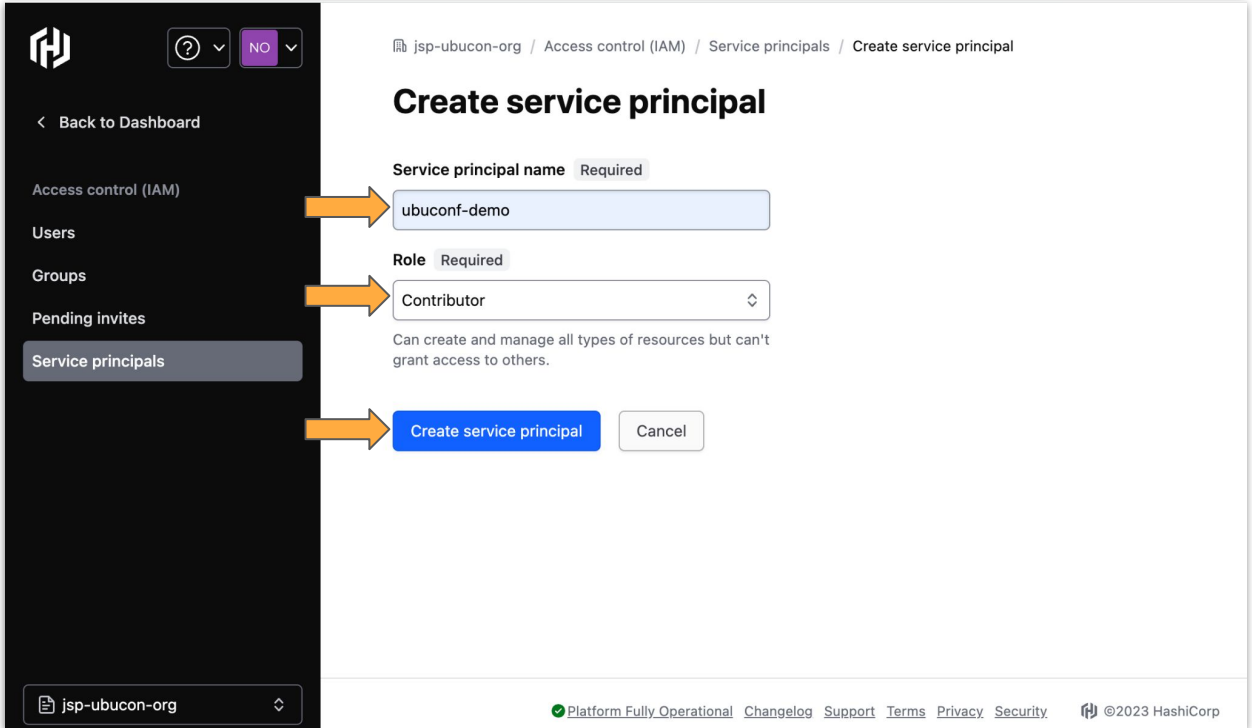
- Service Principal (Org Level) 생성 : '+ Create service principal' 선택



#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비

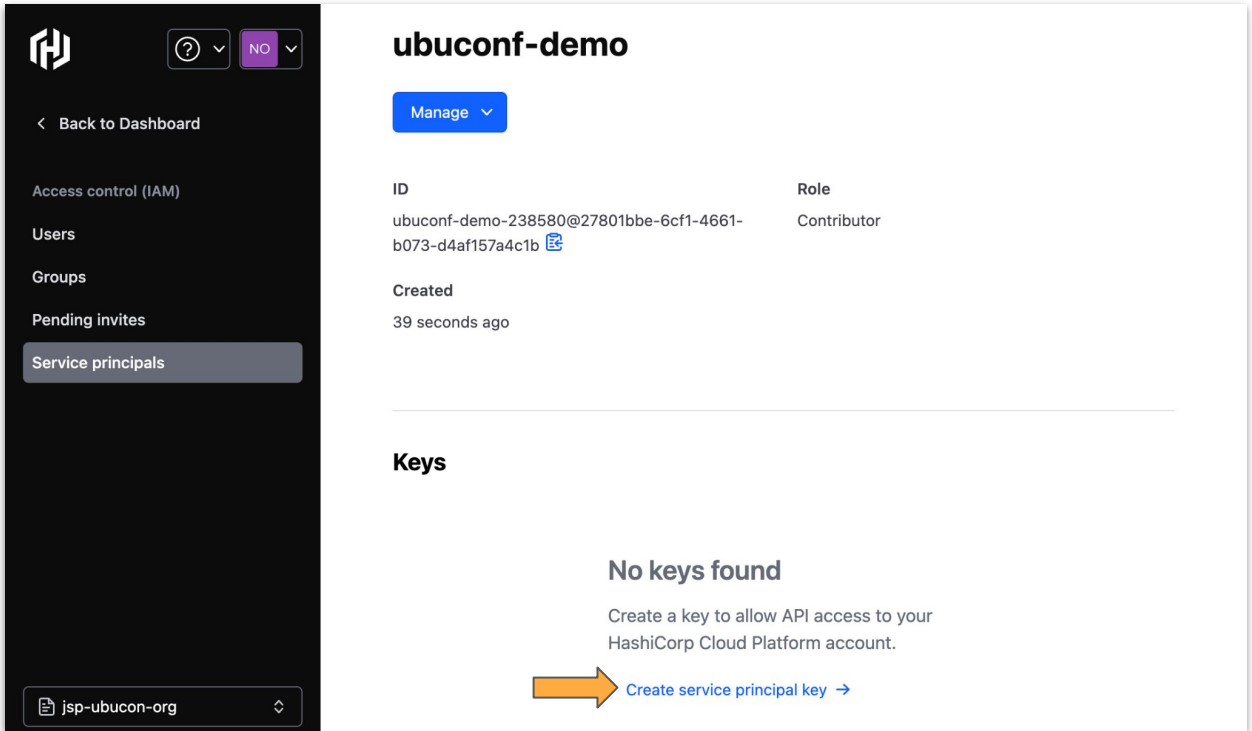
- Service Principal (Org Level) 생성 : ‘Role’은 ‘Contributor’로, 이름은 동일하거나 원하는 이름 입력 후 생성



#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비

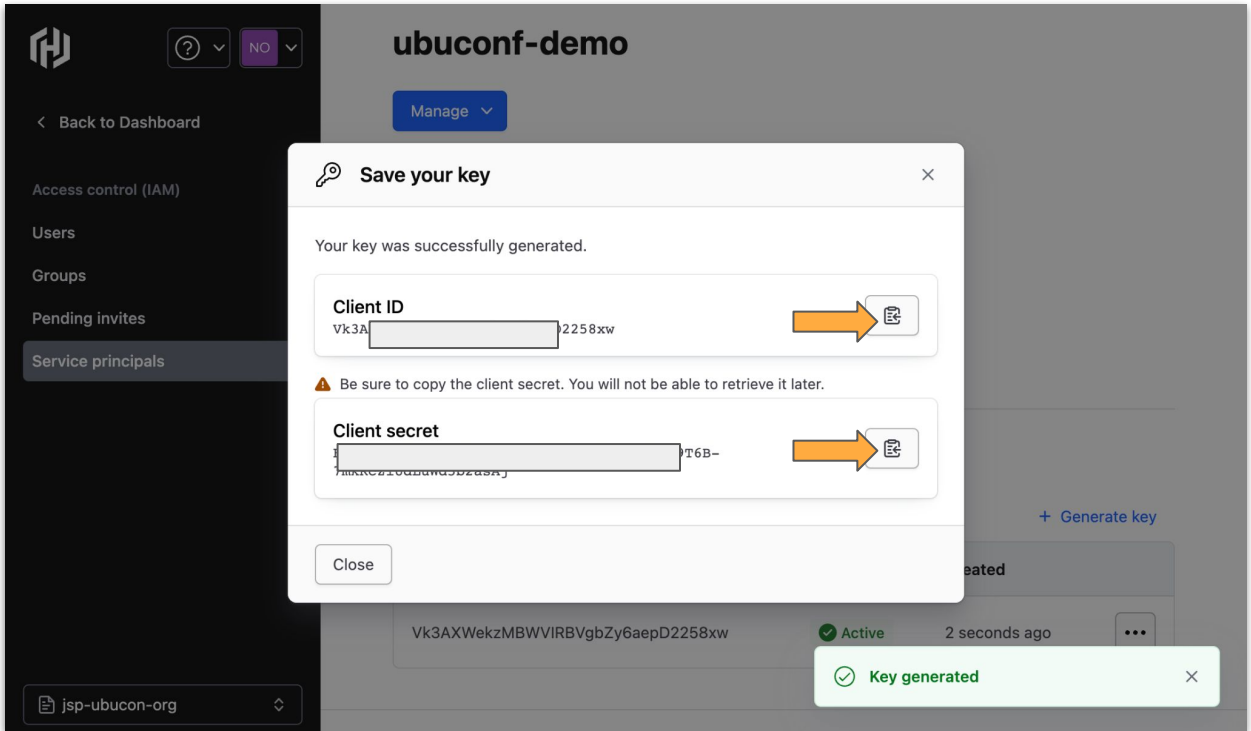
- Service Principal 생성 : ‘Create service principal key’ 클릭하여 key 생성



#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비

- Service Principal 생성 : ‘Client ID’ 와 ‘Client Secret’ 정보 화면 출력, 해당 정보 복사 후 별도 보관 후 ‘Close’

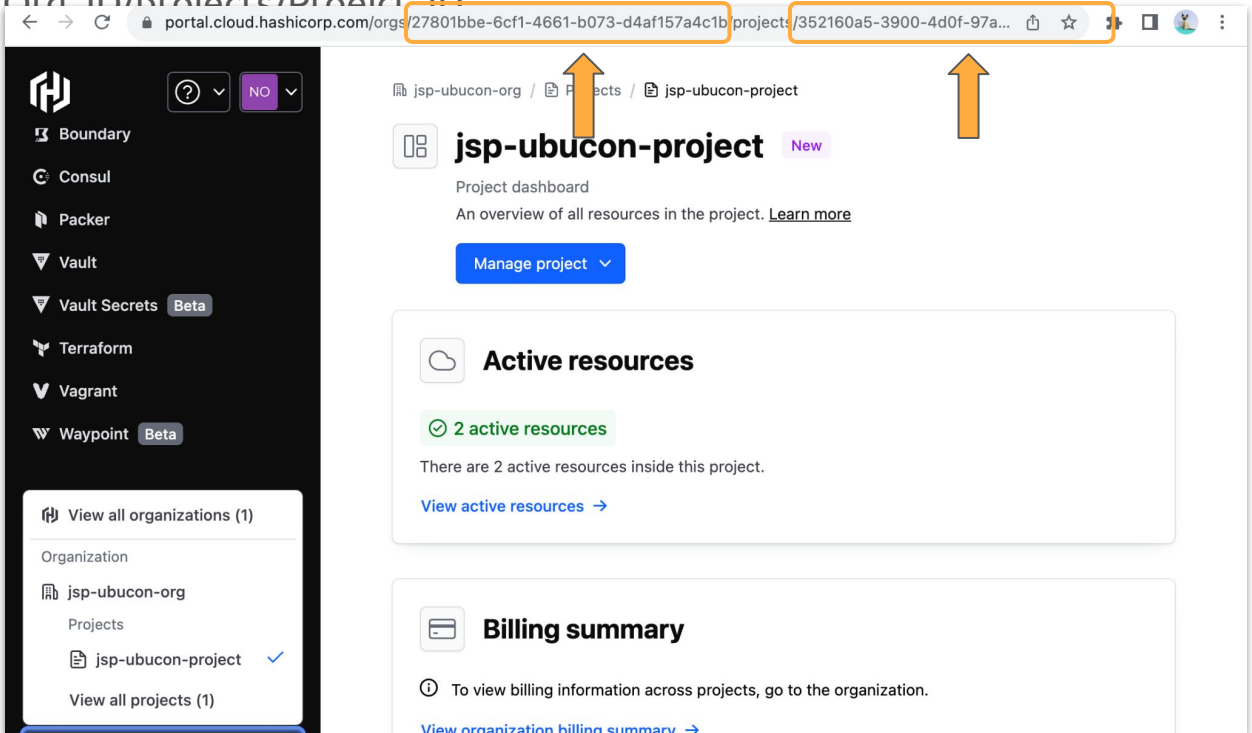


#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비

- Org ID, Project ID 확인 : Project 선택 시, URL 창 에서 해당 정보 복사

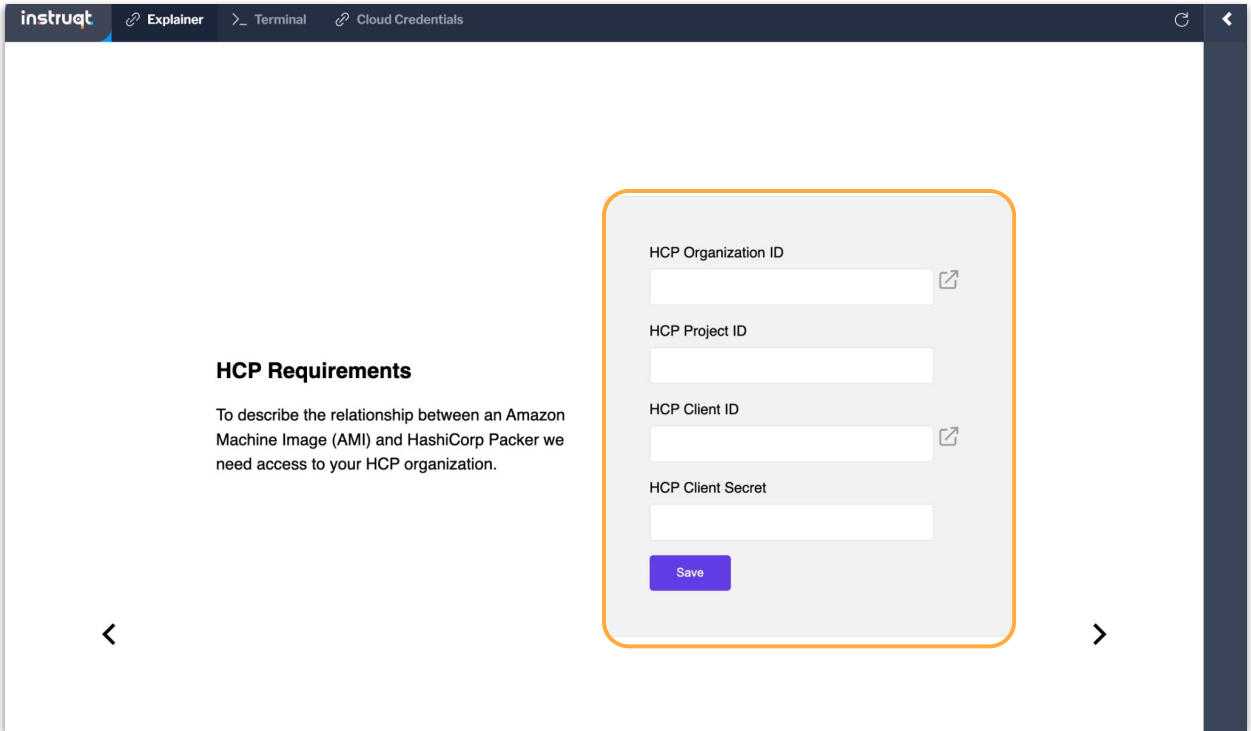
  'Orgs/Org_ID/projects/Proeict_ID'



#0. 사전 준비 작업

2.HashiCorp Cloud Platform(HCP) 계정 생성 및 환경 준비

- HCP 필요 정보 확인 완료 : 복사 후 별도 보관



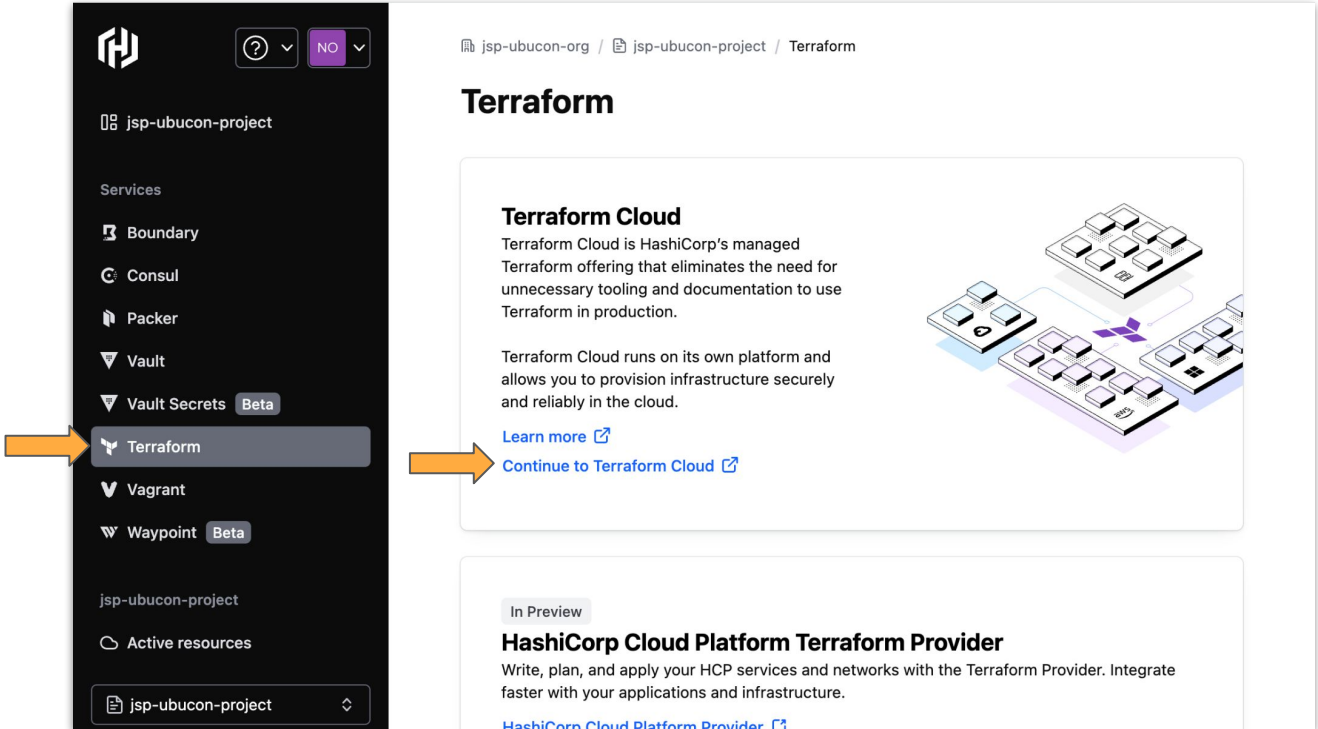
#0. 사전 준비 작업

1. 실습 환경 계정 생성 및 준비
2. HashiCorp Cloud Platform 계정 생성 및 환경 준비
3. Terraform Cloud 계정 생성 및 환경 준비

#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비

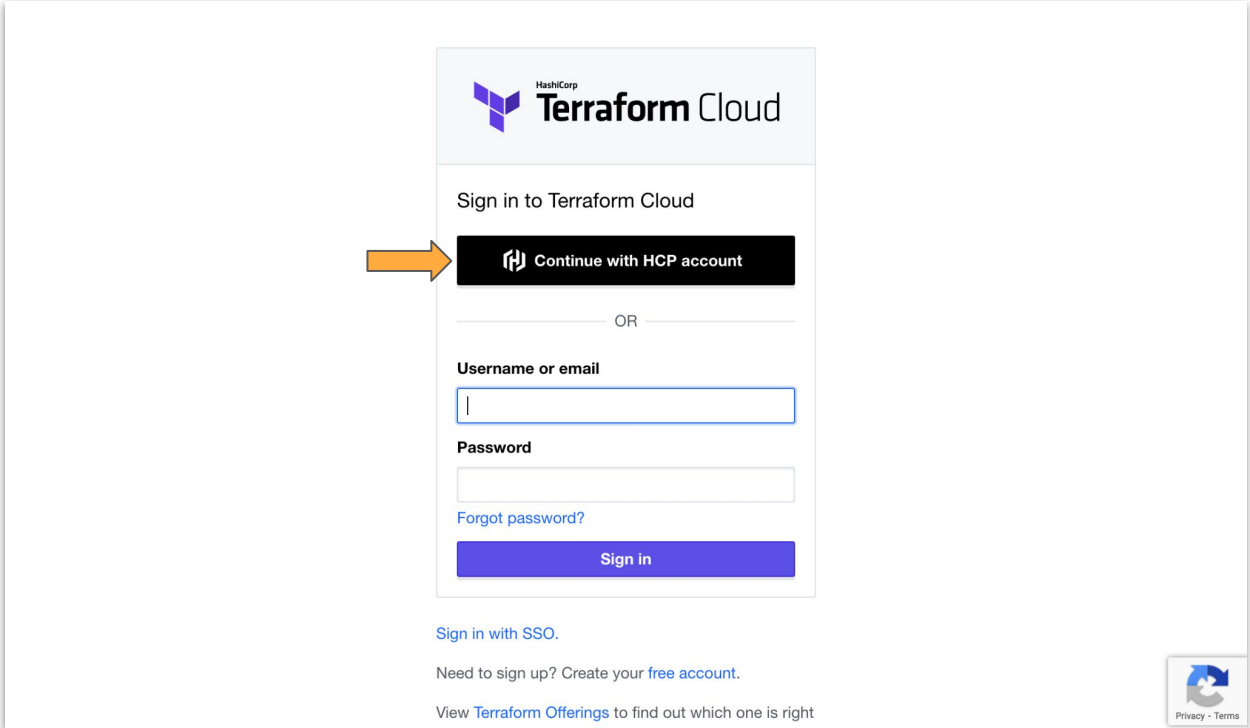
- Terraform Cloud 계정 생성 : ‘Terraform’ 선택 후 ‘Continue to Terraform Cloud’ 선택



#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비

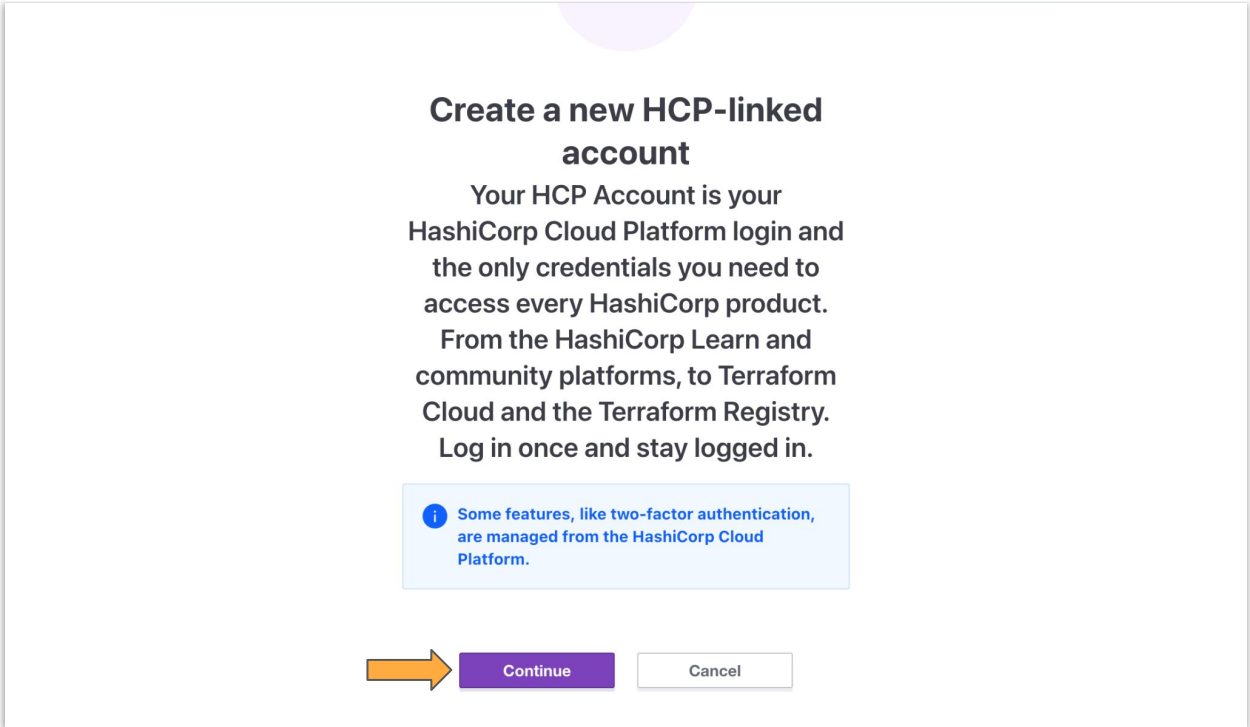
- Terraform Cloud 계정 생성 : ‘Continue with HCP Account’ 선택



#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비

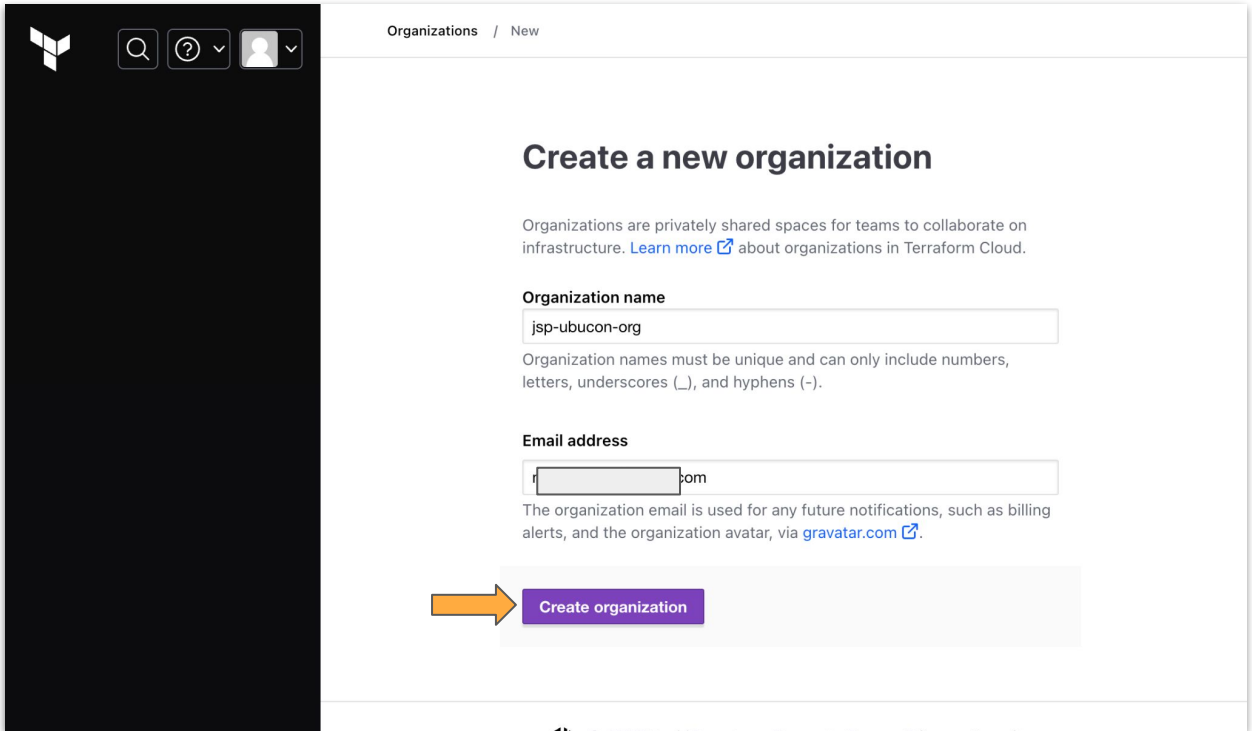
- Terraform Cloud 계정 : ‘Continue ’ 선택



#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비

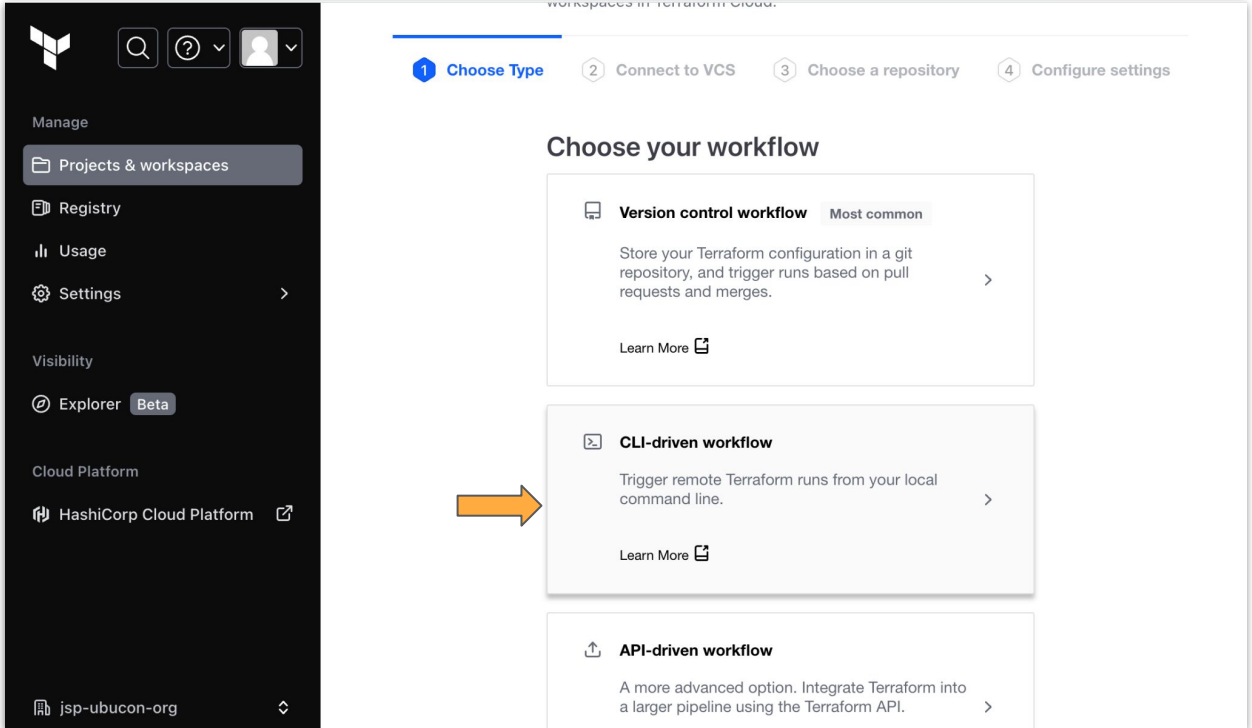
- Terraform Cloud 환경 준비 : '이름약자3자-ubucon-org'를 조직명으로 설정 후, 'Create organization' 선택



#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비

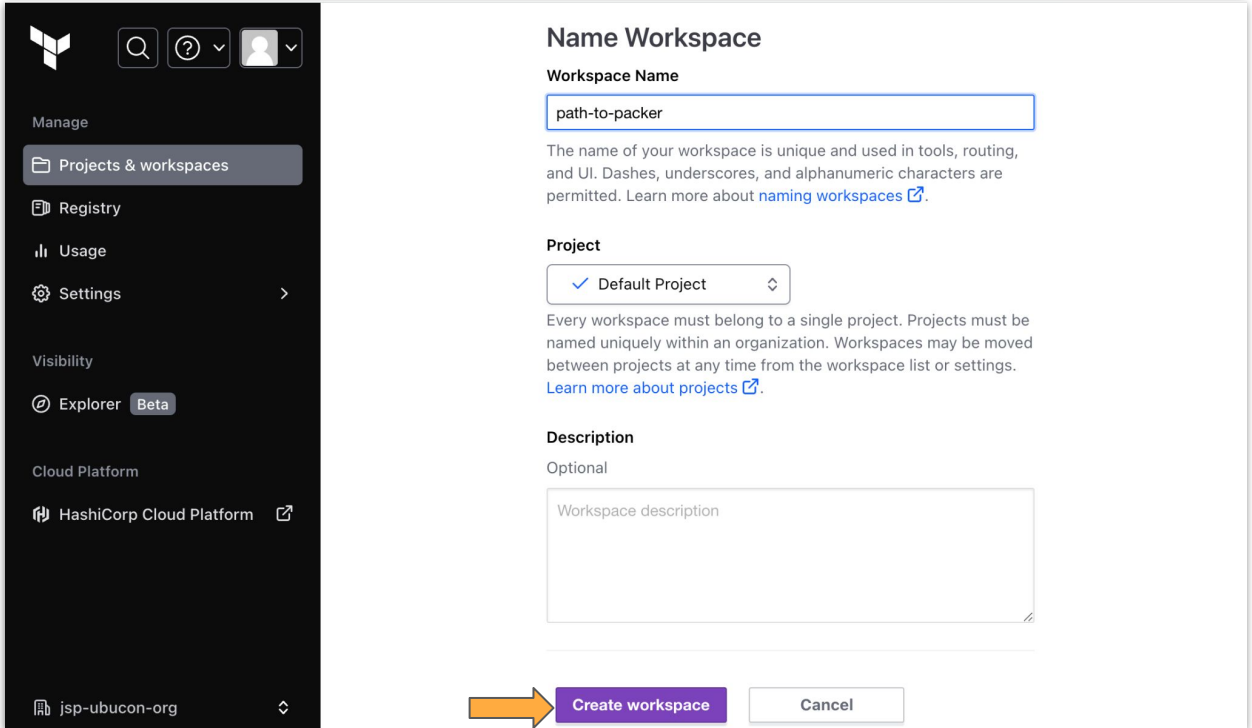
- Terraform Cloud 환경 준비 : ‘CLI-driven Workflow’ 선택



#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비

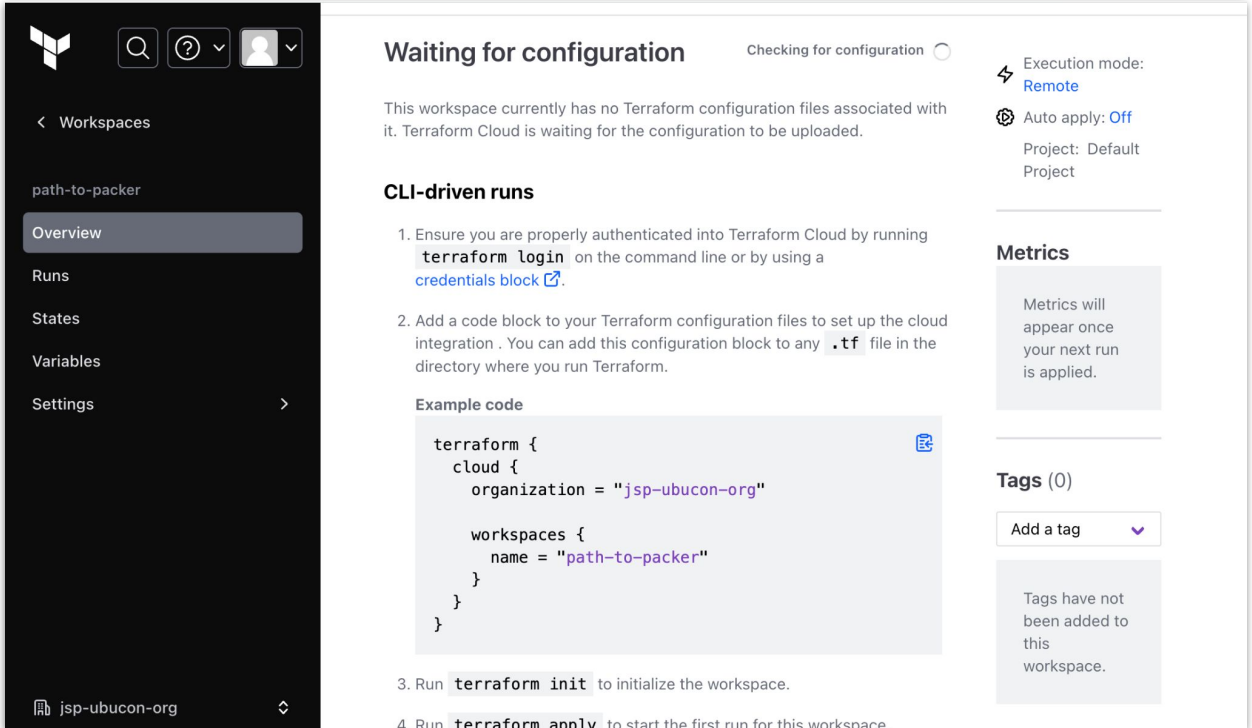
- Terraform Cloud 환경 준비 : ‘path-to-packer’로 Workspace 이름 지정 후, ‘Create Workspace’ 선택



#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비

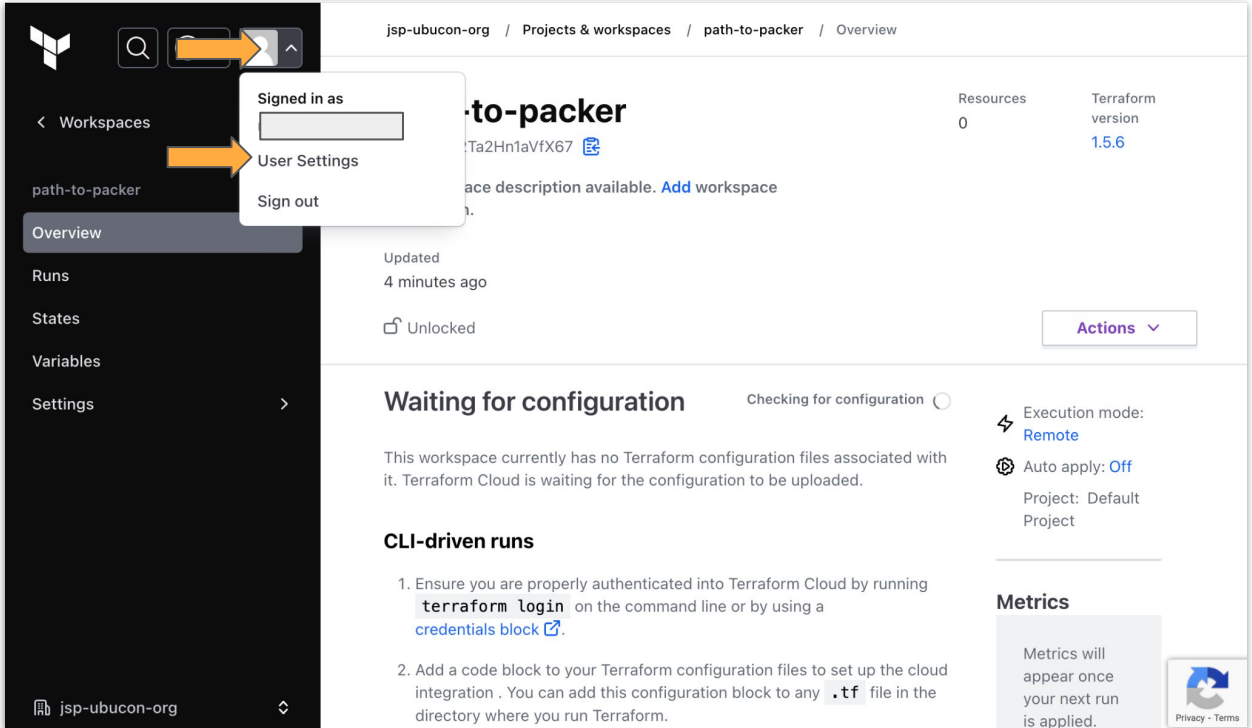
- Terraform Cloud 환경 준비 : 아래와 같이 Workspace 생성 완료



#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비

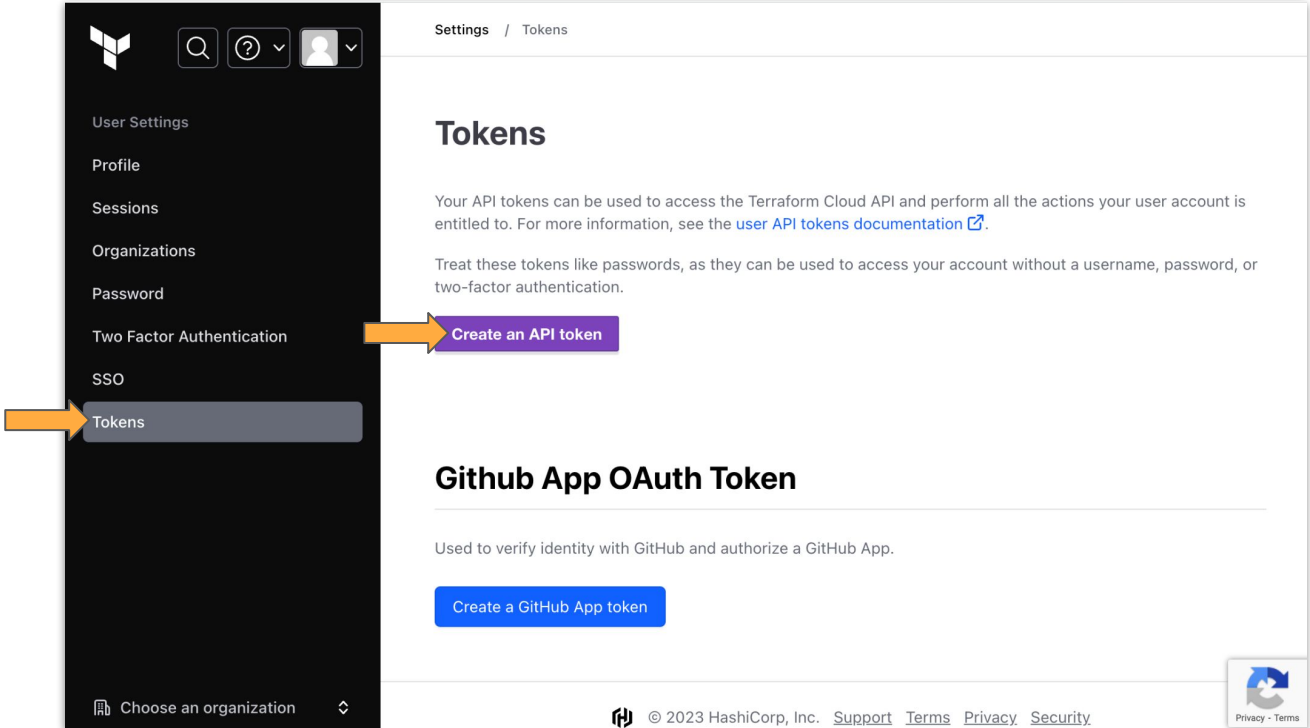
- Terraform Cloud 환경 준비 : 좌측 상단 사람 모양 클릭 후 'User Settings' 선택



#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비

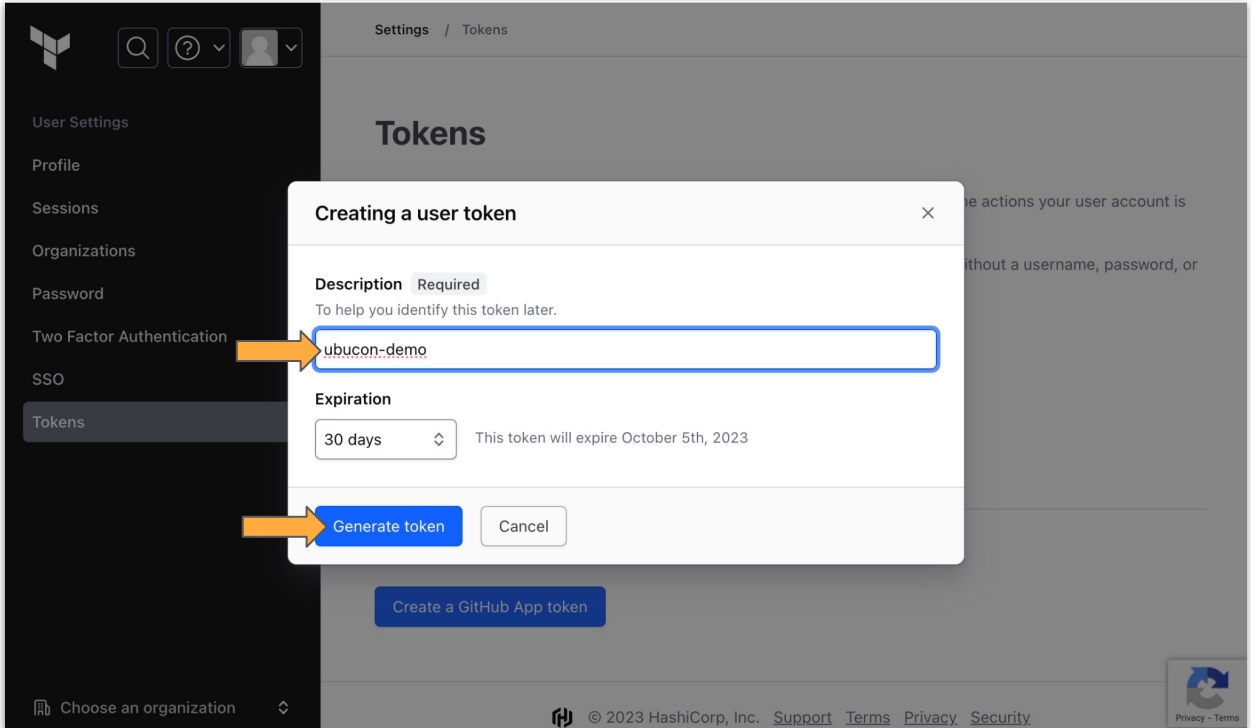
- Terraform Cloud 환경 준비 : Tokens 선택 후 'Create an API token' 선택



#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비

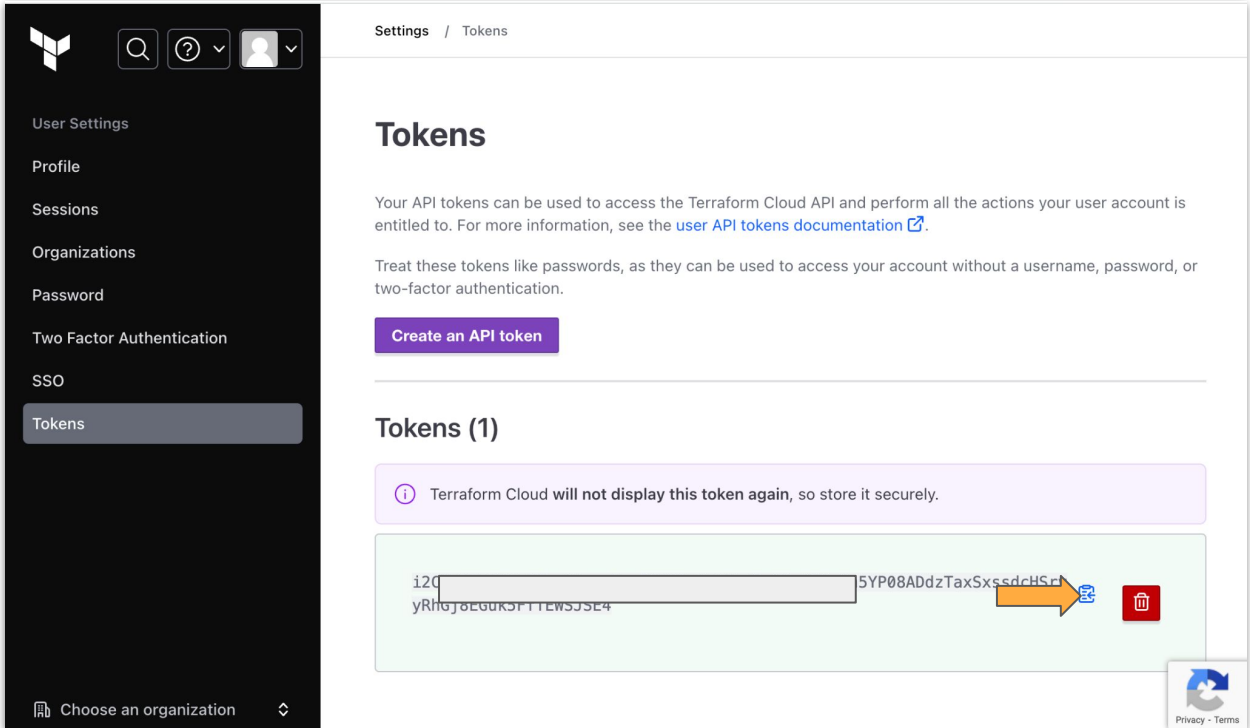
- Terraform Cloud 환경 준비 : ‘ubucon-demo’ 입력 후 , ‘Generate token’ 선택



#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비

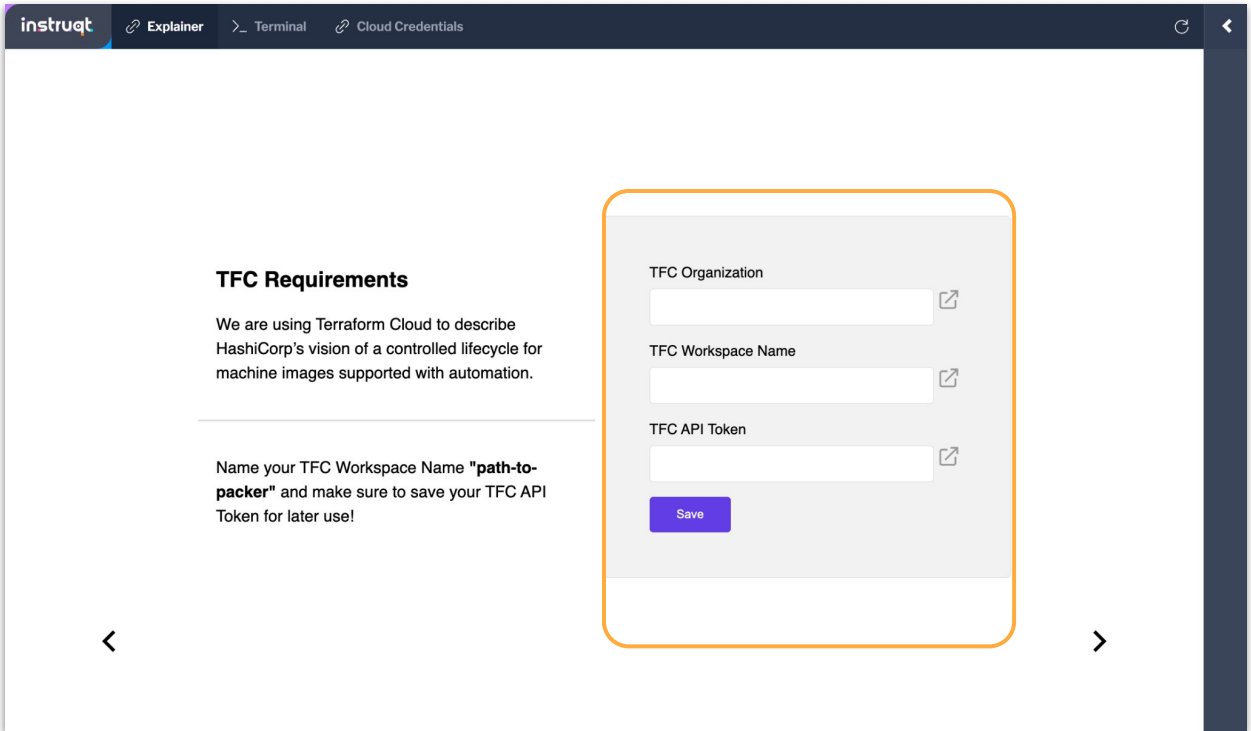
- Terraform Cloud 환경 준비 : 생성된 토큰 복사 후 별도 보관



#0. 사전 준비 작업

2. Terraform Cloud 계정 생성 및 환경 준비

- Terraform Cloud 환경 준비 완료



실습 흐름

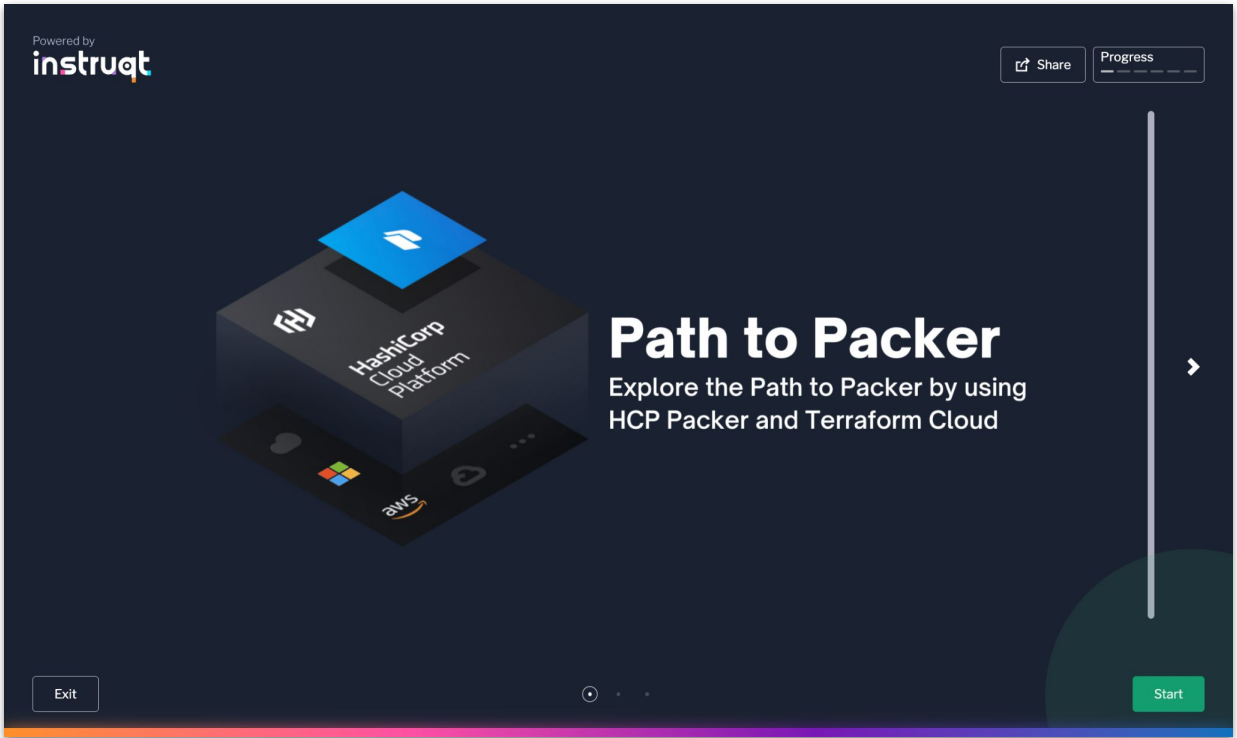
1	**Runbook**을 패커 템플릿으로	• 작업 절차가 정리된 Runbook을 패커 템플릿화 • 이미지 생성 작업 시간 단축 • 작업 표준화로 오류 최소화
2	골든 이미지 기반 배포 **(AWS → Azure)**	• 골든 이미지를 AWS AMI로 저장 • 저장된 이미지를 사용, AWS에 가상서버 배포 • 이미지를 재사용, Azure로 서비스 확장
3	실습 환경 정리	• CSP 내 가상 서버 삭제 • 필요 시 HCP 내 환경 삭제 및 정리 • Shadow IT 예방

실습 흐름

1	**Runbook**을 패커 템플릿으로	• 작업 절차가 정리된 Runbook을 패커 템플릿화 • 이미지 생성 작업 시간 단축 • 작업 표준화로 오류 최소화
2	골든 이미지 기반 배포 **(AWS → Azure)**	• 골든 이미지를 AWS AMI로 저장 • 저장된 이미지를 사용, AWS에 가상서버 배포 • 이미지를 재사용, Azure로 서비스 확장
3	실습 환경 정리	• CSP 내 가상 서버 삭제 • 필요 시 HCP 내 환경 삭제 및 정리 • Shadow IT 예방

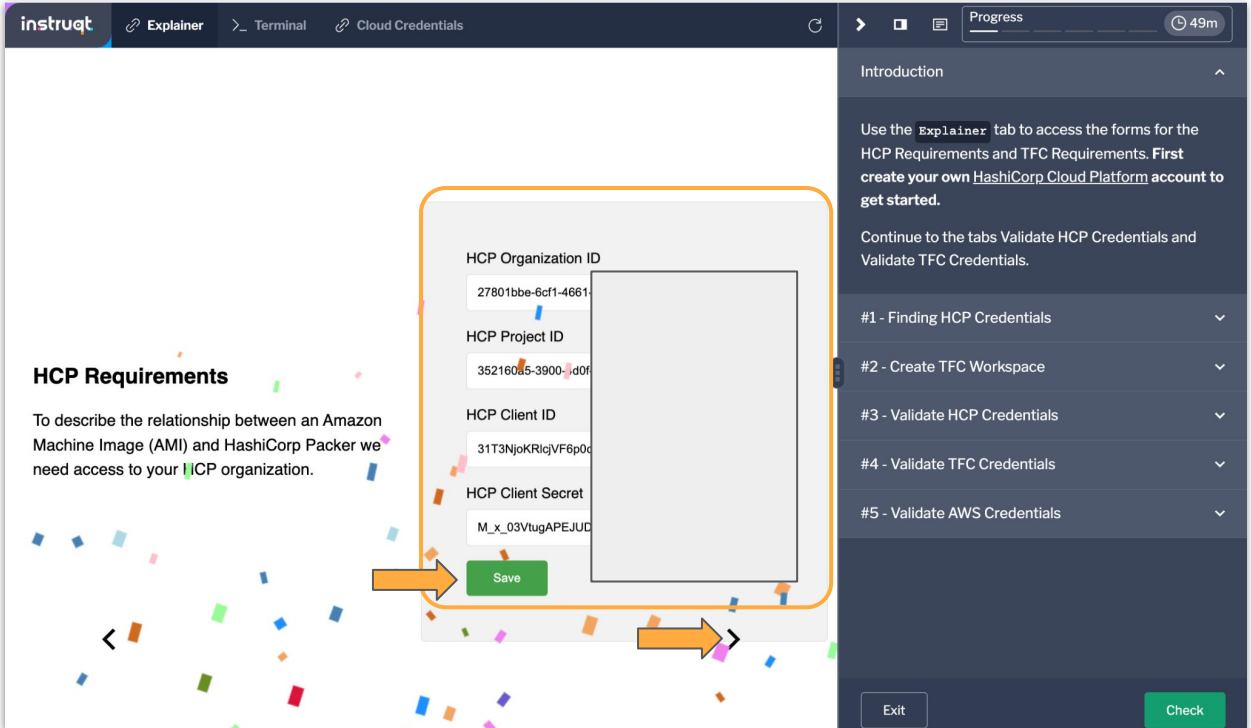
#1. Runbook을 패커 템플릿으로

- 실습환경으로 복귀!

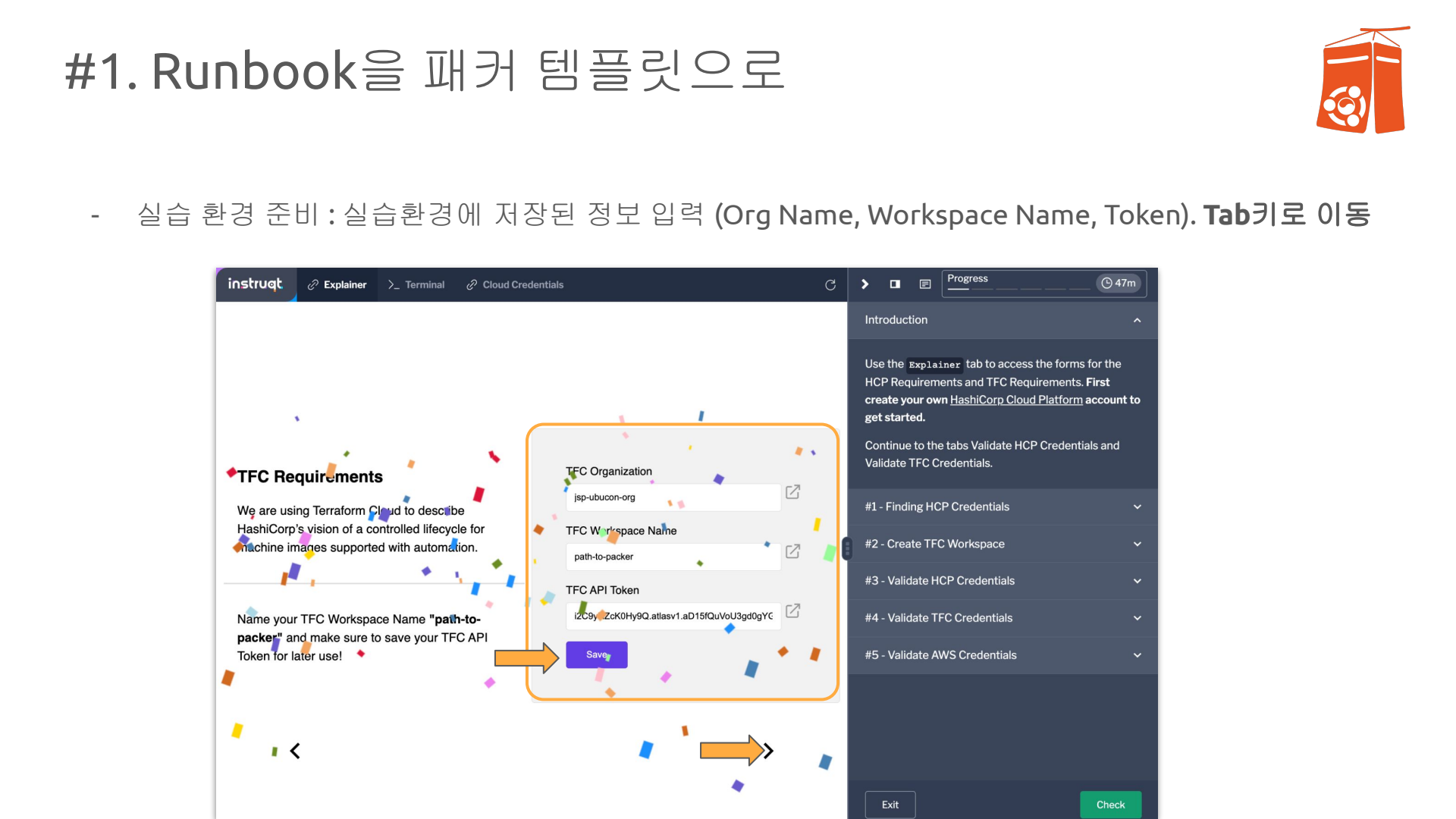


#1. Runbook을 패커 템플릿으로

- 실습 환경 준비 : 실습환경에 저장된 정보 입력 (HCP Org ID, Project ID, Client ID/Secret). **Tab키로 이동**



#1. Runbook을 패커 템플릿으로

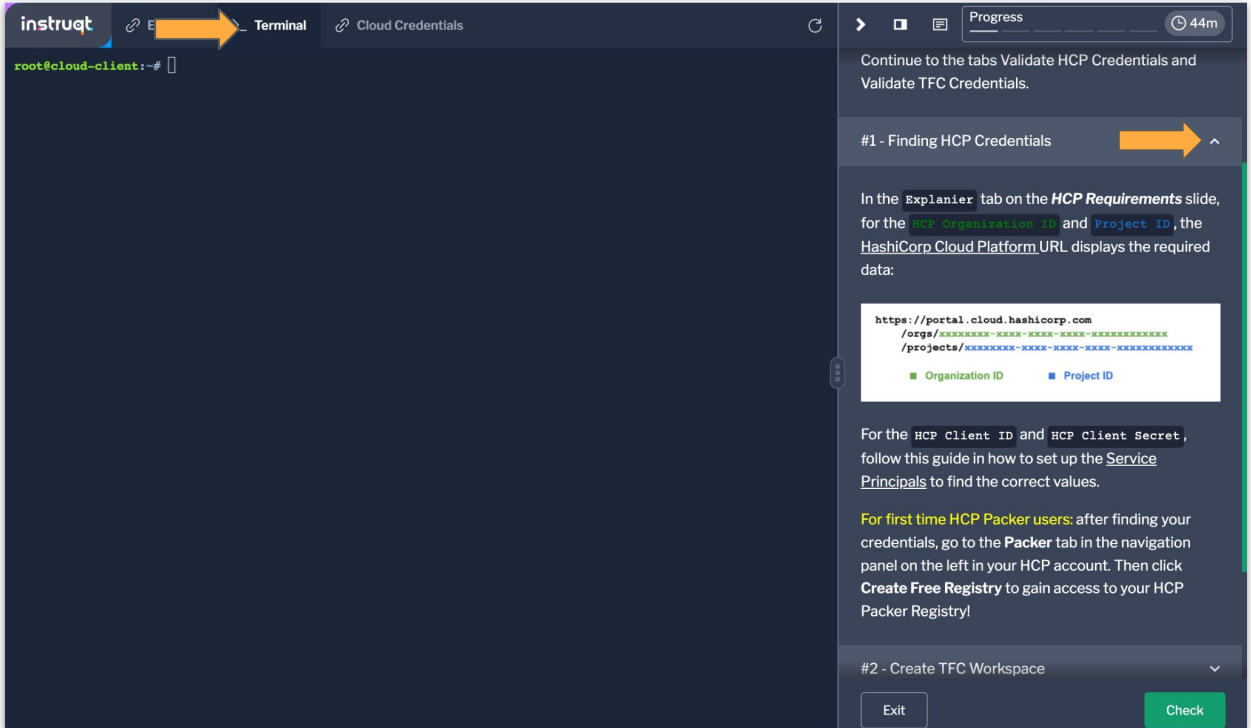


- 실습 환경 준비 : 실습환경에 저장된 정보 입력 (Org Name, Workspace Name, Token). **Tab키로 이동**

#1. Run Book을 패커 탬플릿으로

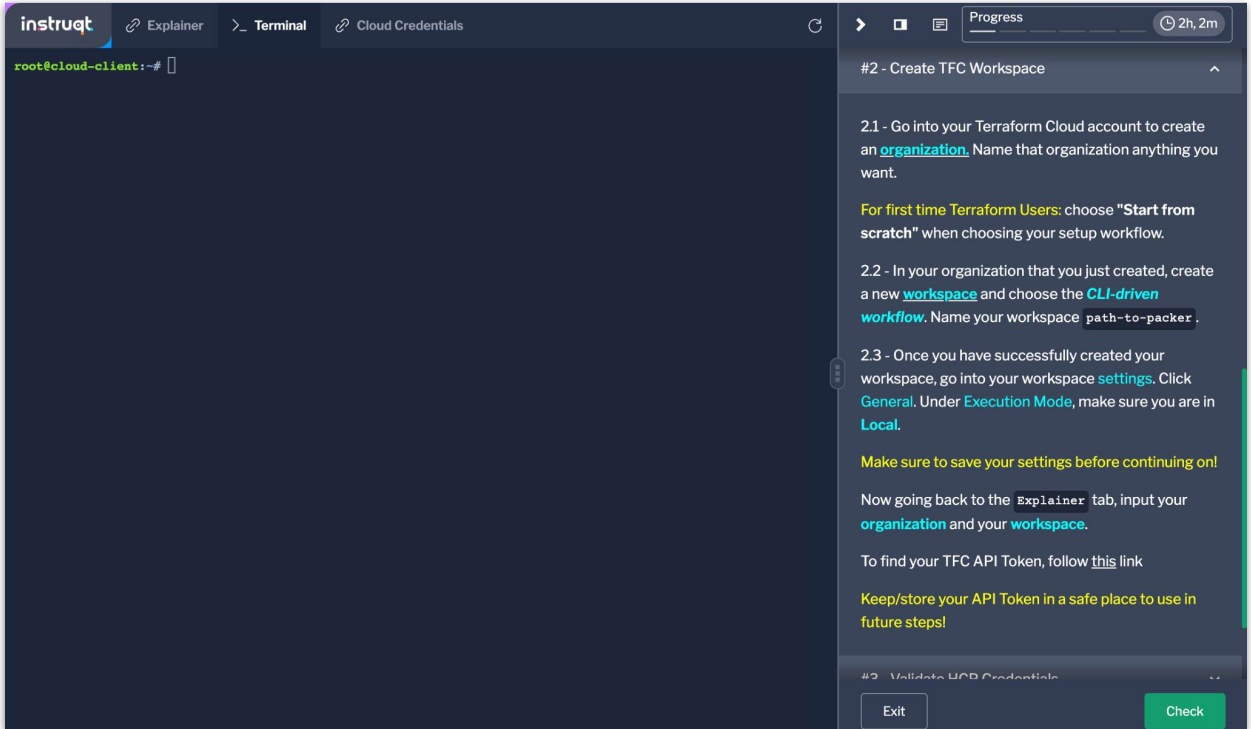
2. Terraform Cloud 계정 생성 및 환경 준

- 실습 환경 준비 : 설정된 정보 확인. #1 ~ #5까지 Task 수행 후 성공 시 'Check' 선택하여 다음 단계로 이동



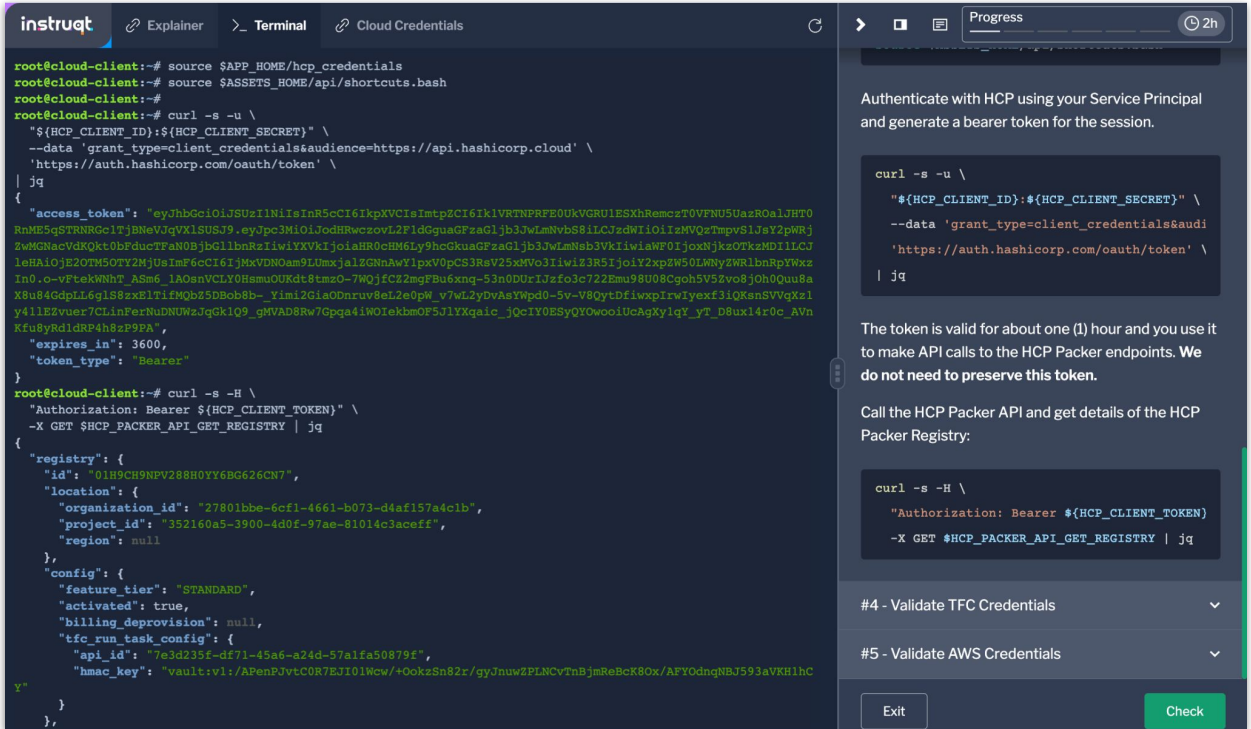
#1. Runbook을 패커 템플릿으로

- 실습 환경 준비 : #1 HCP 정보 확인, #2 Terraform Cloud 구성 확인 단계



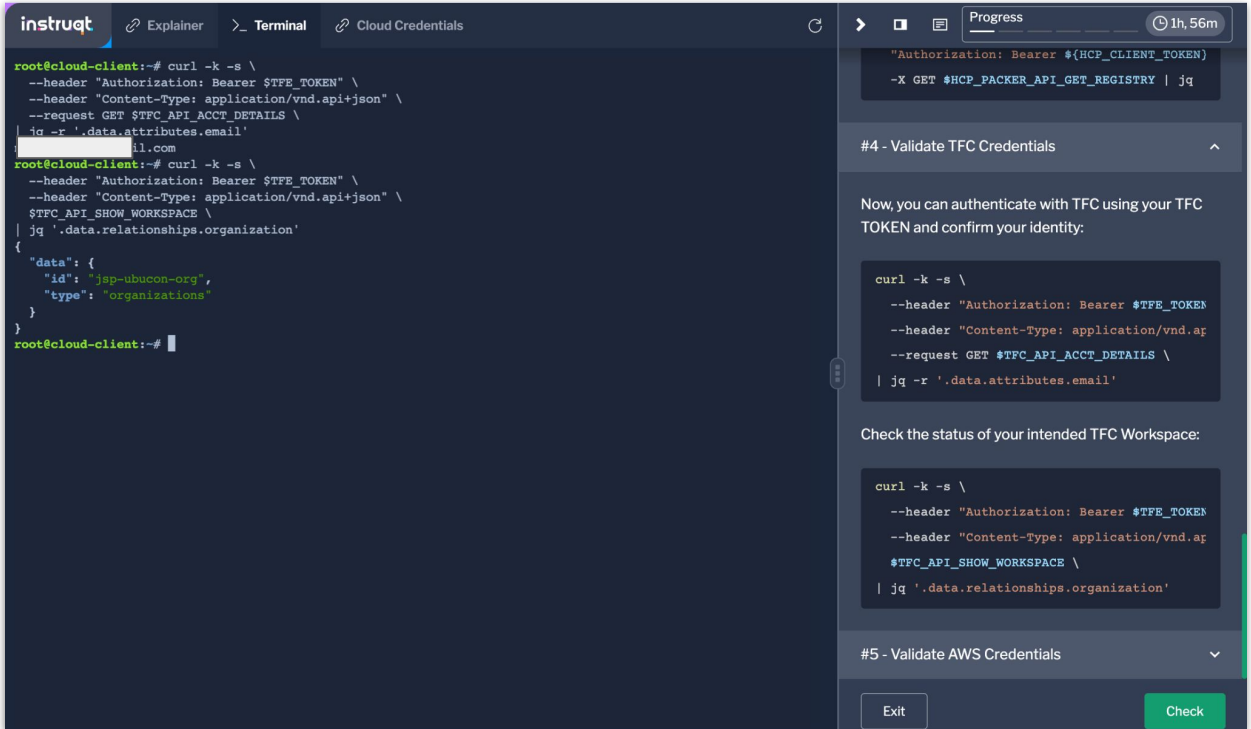
#1. Runbook을 패커 템플릿으로

- 실습 환경 준비 : #3 HCP 정보 확인, Terminal 탭에서 아래와 같이 실행



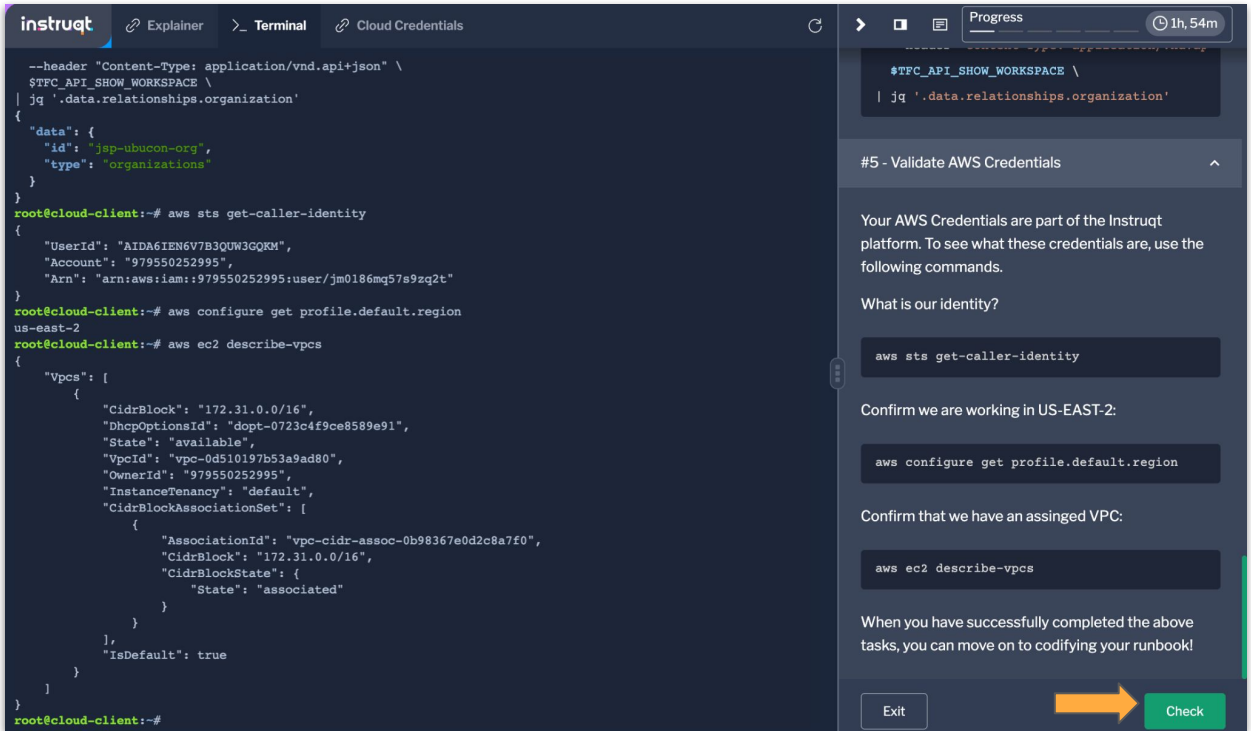
#1. Runbook을 패커 템플릿으로

- 실습 환경 준비 : #4 Terraform Cloud 정보 확인, Terminal 탭에서 아래와 같이 실행



#1. Runbook을 패커 템플릿으로

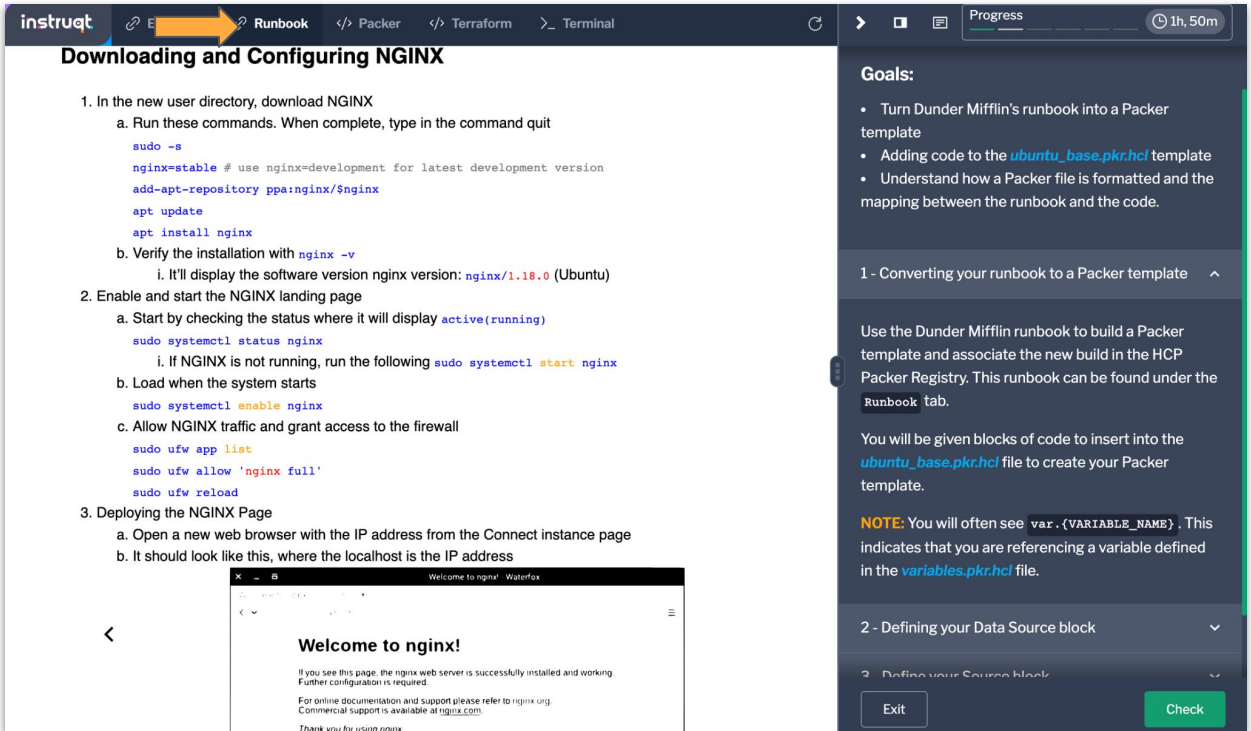
- 실습 환경 준비 : #5 AWS 정보 확인, Terminal 탭에서 아래와 같이 실행. 정상 종료 시 'Check' 선택



#1. Runbook을 패커 템플릿으로

1. Converting your runbook to a Packer Template

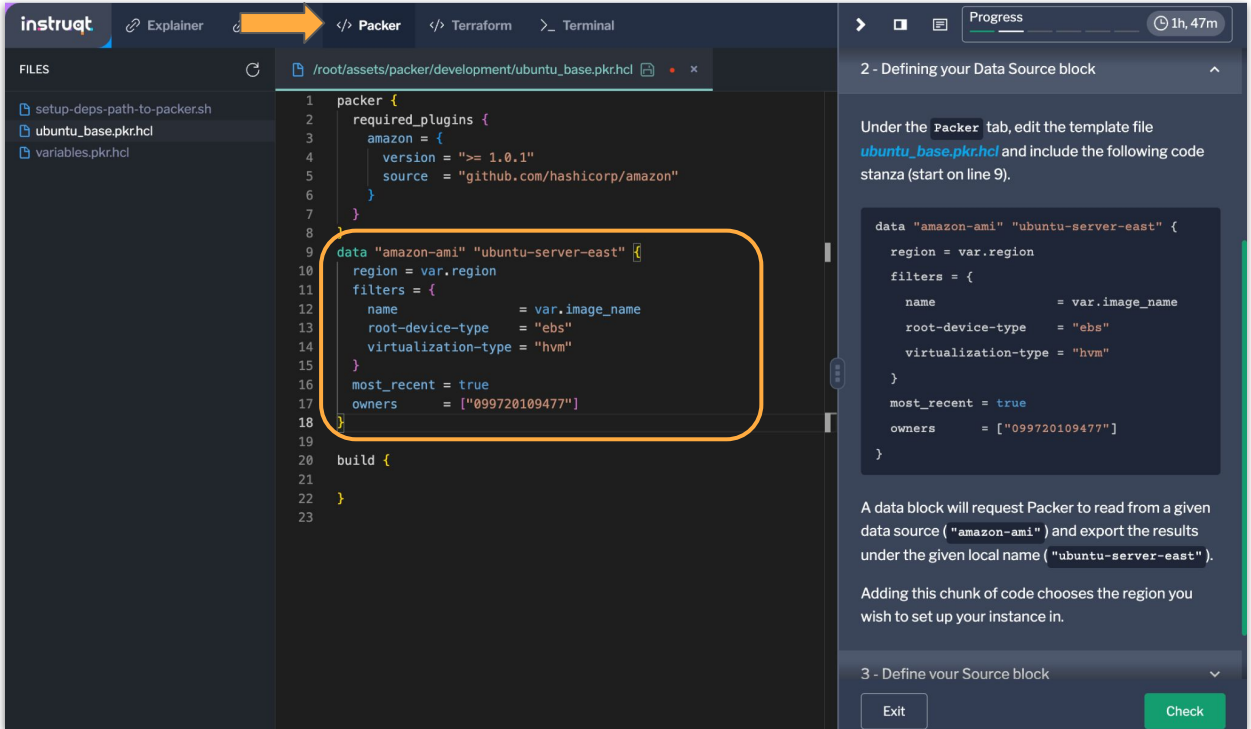
- RunBook 확인 : 수작업으로 진행하는 AWS 인스턴스 배포 및 구성 작업 확인



#1. Runbook을 패커 템플릿으로

2. Defining your Data Source Block

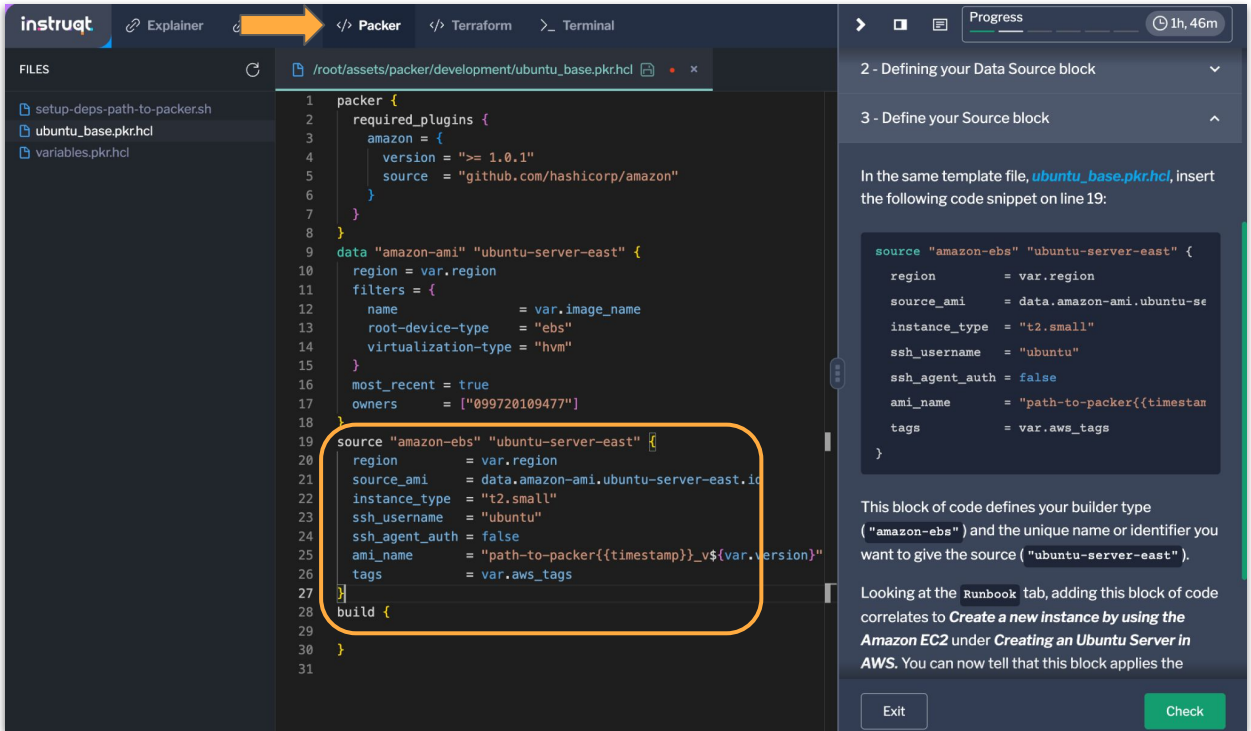
- ubutu_base.pkr.hcl 수정 : ‘Packer’ 탭에서 수행. Data Source 블럭 추가



#1. Runbook을 패커 템플릿으로

3. Defining your Source Block

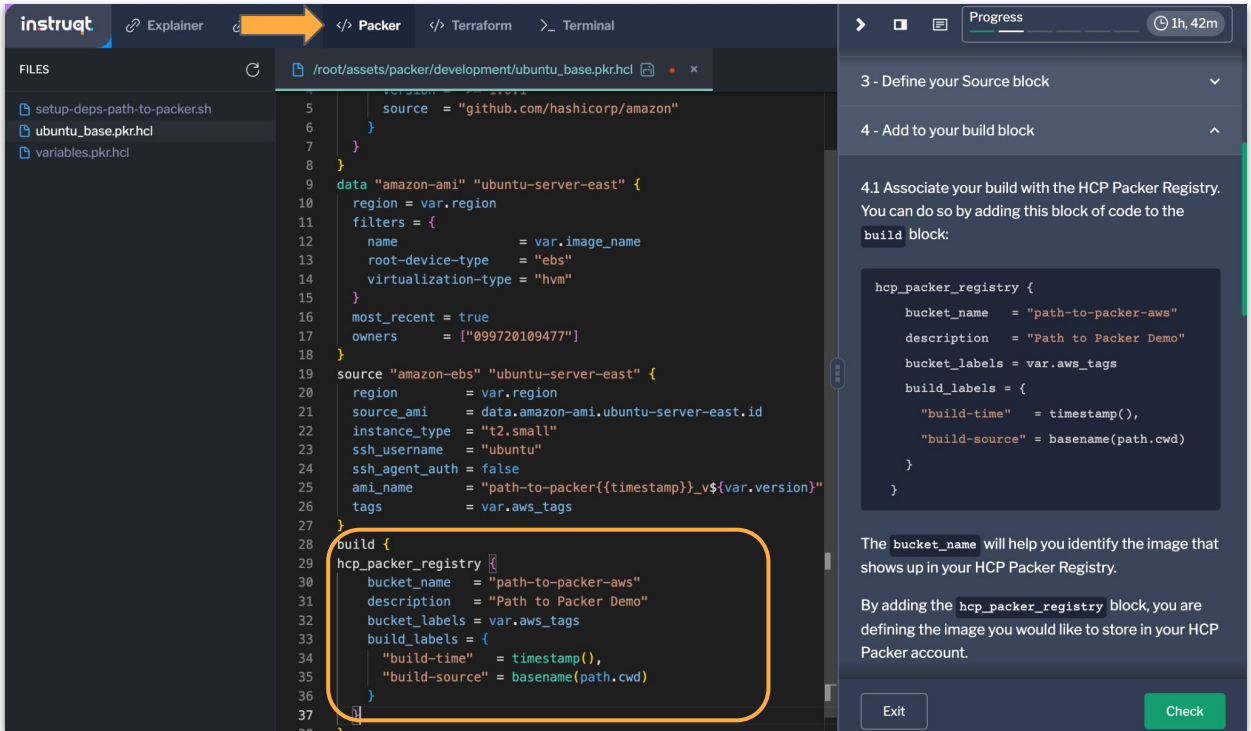
- ubutu_base.pkr.hcl 수정 : Source 블럭 추가



#1. Runbook을 패커 템플릿으로

4. Add your build Block

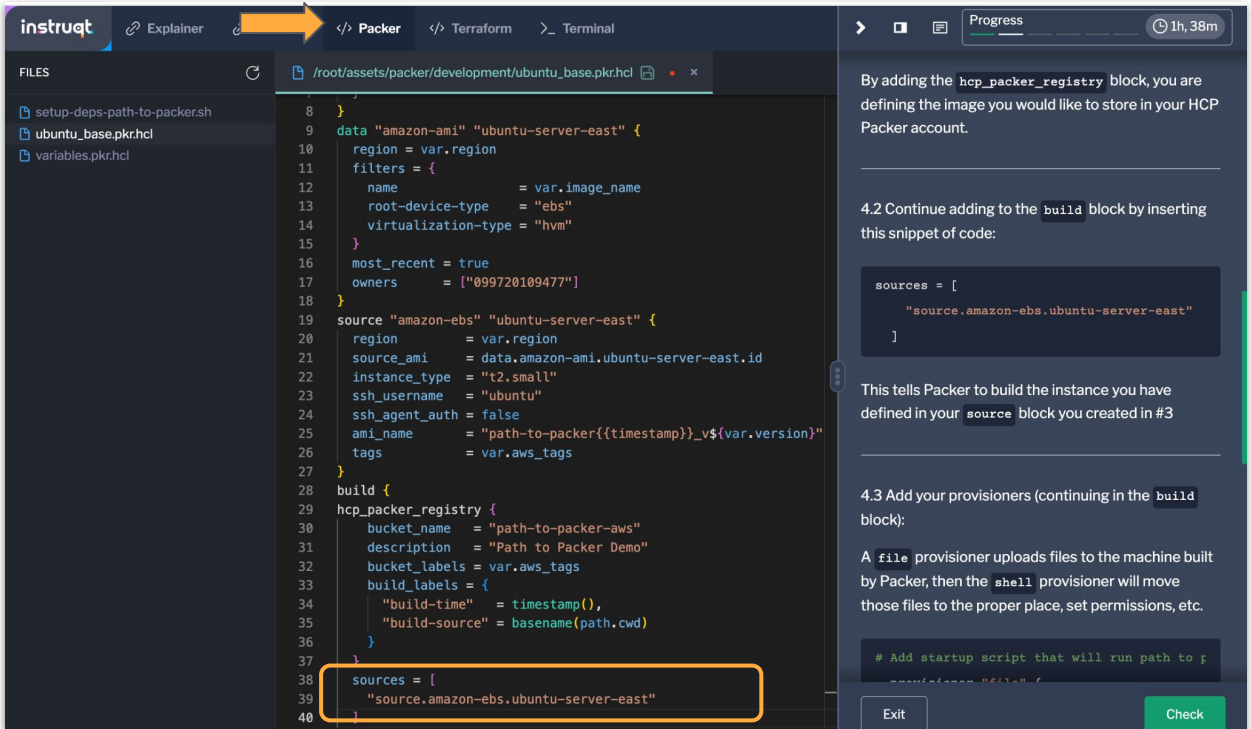
- 4.1 Associate your build with the HCP Packer Registry :



#1. Runbook을 패커 템플릿으로

4. Add your build Block

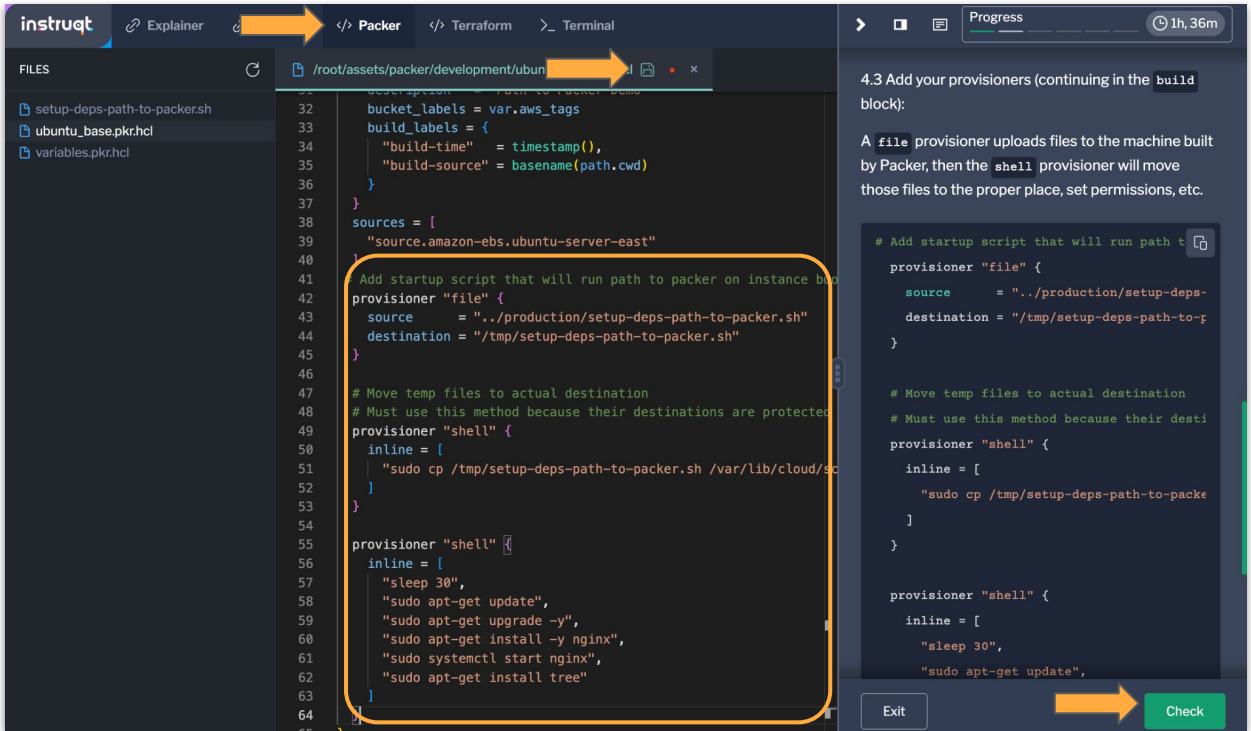
- 4.2 Continue adding to the ‘build’ block :



#1. Runbook을 패커 템플릿으로

4. Add your build Block

- 4.3 Add your provisioner : “디스켓"모양 클릭하여 저장 후, “Check” 선택

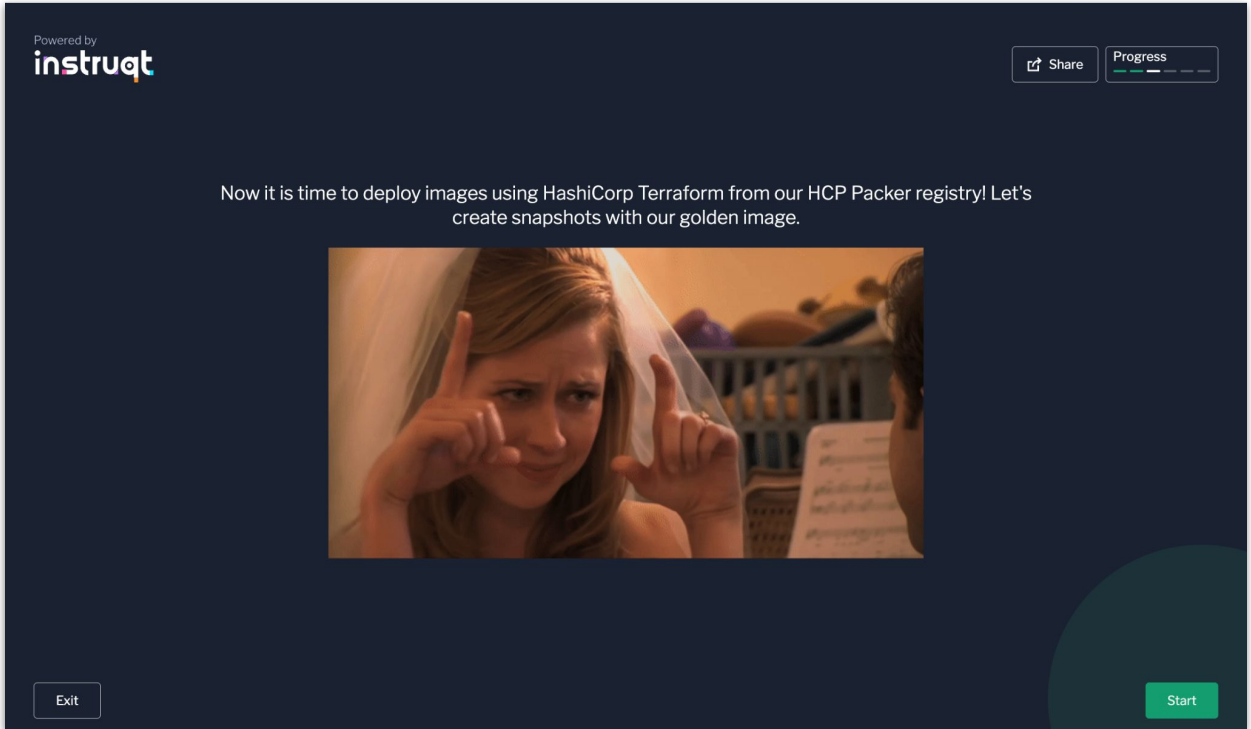


실습 흐름

1	**Runbook**을 패커 템플릿으로	• 작업 절차가 정리된 Runbook을 패커 템플릿화 • 이미지 생성 작업 시간 단축 • 작업 표준화로 오류 최소화
2	골든 이미지 기반 배포 **(AWS → Azure)**	• 골든 이미지를 AWS AMI로 저장 • 저장된 이미지를 사용, AWS에 가상서버 배포 • 이미지를 재사용, Azure로 서비스 확장
3	실습 환경 정리	• CSP 내 가상 서버 삭제 • 필요 시 HCP 내 환경 삭제 및 정리 • Shadow IT 예방

#2. 골든 이미지 기반 배포 (AWS → Azure)

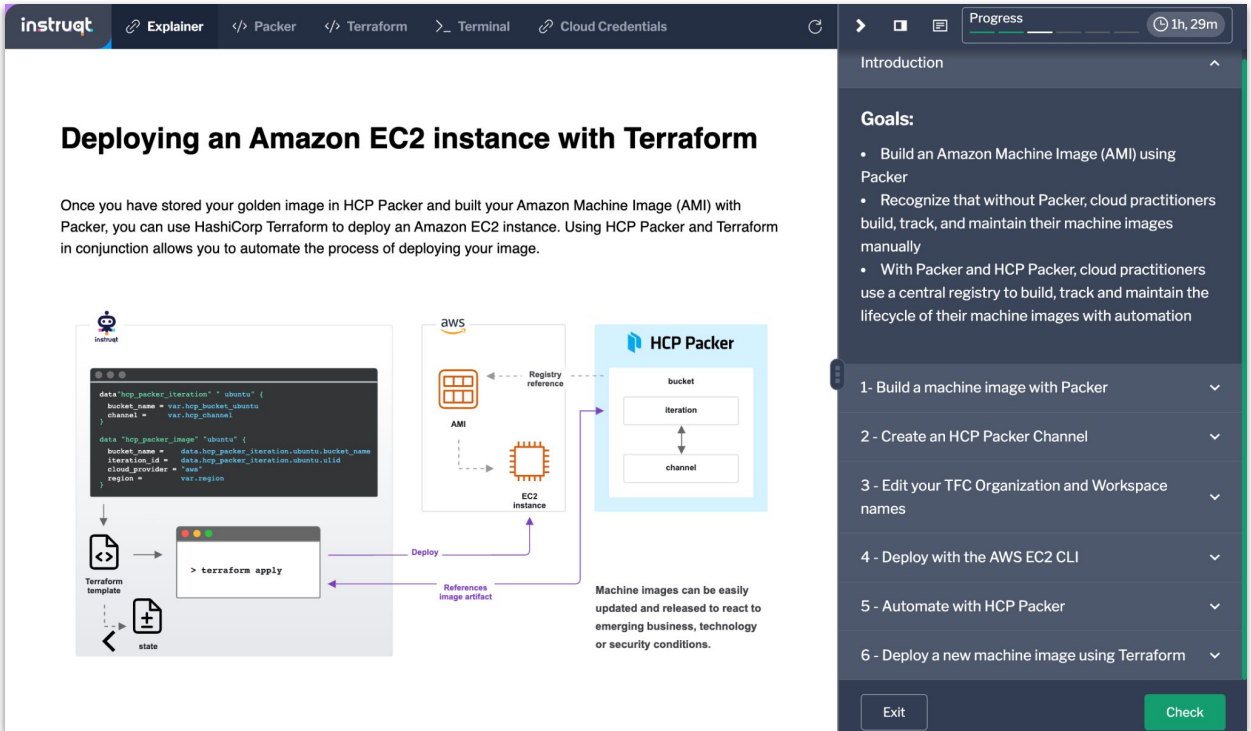
-



#2. 골든 이미지 기반 배포 (AWS)

1. Add your build Block

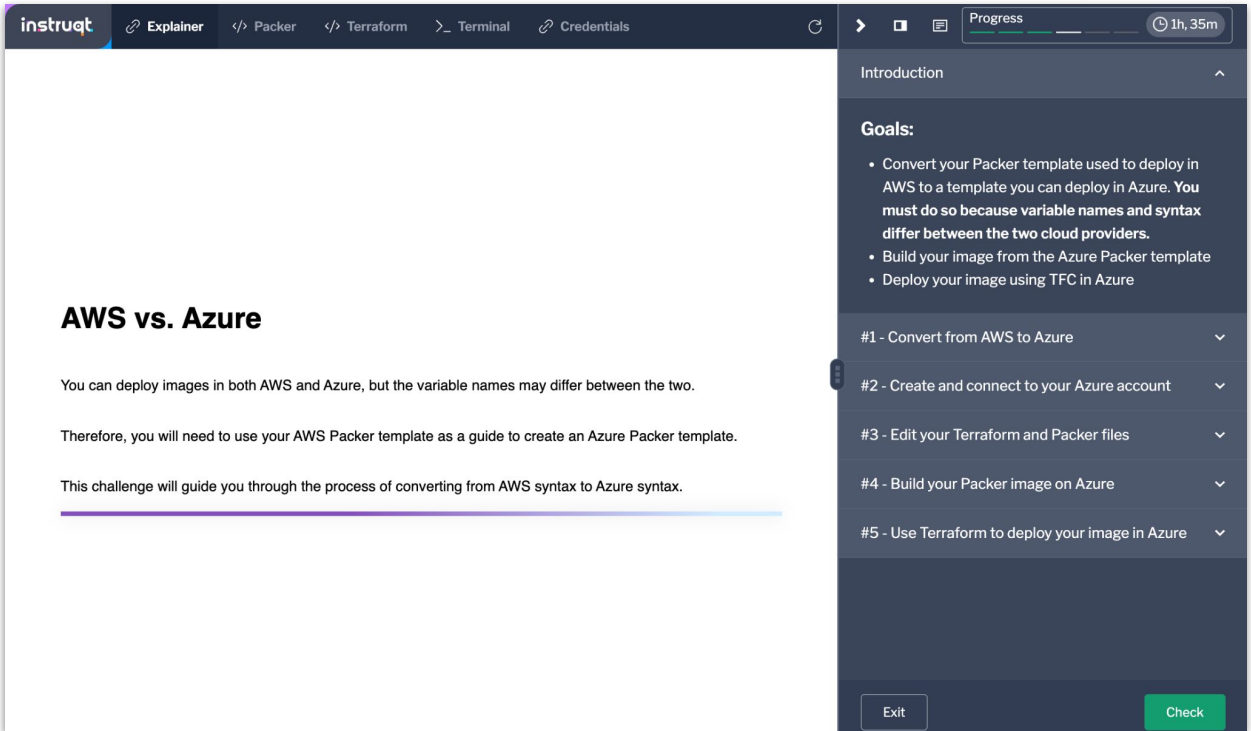
- #1 ~ #6까지 수행. #1 이미지 생성 시 약 4분 30초 ~ 5분 소요



#2. 골든 이미지 기반 배포 (Azure)

1. Add your build Block

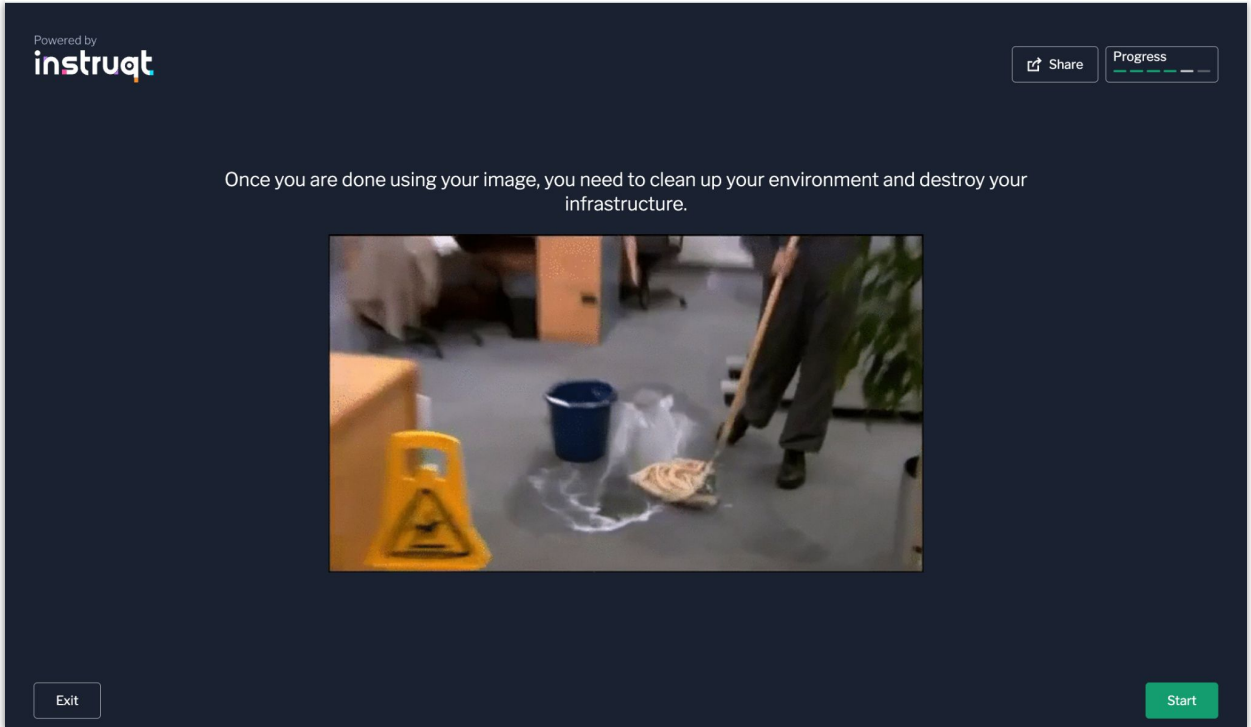
- #1 ~ #5까지 수행



실습 흐름

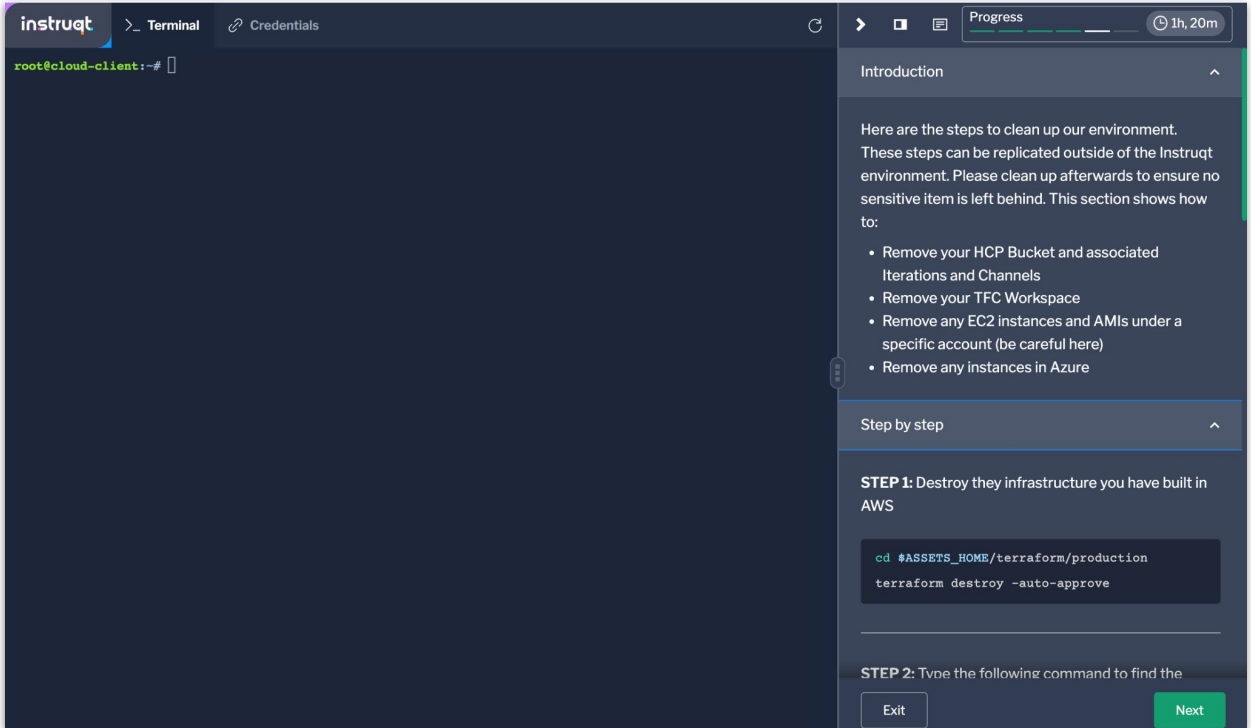
1	**Runbook**을 패커 템플릿으로	• 작업 절차가 정리된 **Runbook**을 패커 템플릿화 • 이미지 생성 작업 시간 단축 • 작업 표준화로 오류 최소화
2	골든 이미지 기반 배포 **(AWS → Azure)**	• 골든 이미지를 **AWS AMI**로 저장 • 저장된 이미지를 사용, **AWS**에 가상서버 배포 • 이미지를 재사용, **Azure**로 서비스 확장
3	실습 환경 정리	• **CSP** 내 가상 서버 삭제 • 필요 시 **HCP** 내 환경 삭제 및 정리 • **Shadow IT** 예방

#3. 실습 환경 정리


Powered by
instruqt
Share
Progress
Once you are done using your image, you need to clean up your environment and destroy your infrastructure.
Exit
Start

#3. 실습 환경 정리

- #1 ~ #8까지 수행.



마무리

- 클라우드 환경, 가상 서버 이미지

  (Ubuntu Image) 관리 방안 실습