Platform as a Service

Writeup:-

- Platform as a service

AWS Elastic Beanstalk is the PaaS offering from AWS that makes it easy to deploy and manage applications in the cloud without worrying about the infrastructure. Developers can simply upload their code and Elastic Beanstalk automatically handles provisioning, load balancing, auto-scaling and monitoring.

Platform as a service (PaaS) is a complete development and deployment environment in the cloud, with resources that enable you to deliver everything from simple cloud-based apps to sophisticated, cloud-enabled enterprise applications.

- Elastic Beanstalk

Elastic Beanstalk is a PaaS service offered by AWS to deploy and scale web applications quickly without worrying about the infrastructure. It automatically handles capacity provisioning, load balancing, scaling and application health monitoring. Developers just have to upload their code and Elastic Beanstalk will deploy it on AWS infrastructure like EC2, auto scale it and monitor it.

Why Elastic BeanStalk

i. It supports multiple languages like Java, Python, Go etc. and platforms like Docker.
ii. Beanstalk integrates well with other AWS services like EC2, S3, RDS etc.
iii. The main benefits are fast and automated application deployment and management, multiple environments, auto scaling, and cost efficiency.

- Components of beanstalk

i. Application: This is the actual web application code packaged into a zip and uploaded to Elastic Beanstalk.
ii. Application Version: Each deployment of code is an application version. Rollbacks can be done to previous versions.
iii. Environment: This is a version of the application running on AWS resources. We can create multiple environments like dev, test, prod etc from the same application.
iv. Configuration Templates: These allow customization of the AWS resources powering an environment like EC2 instance type, autoscaling settings etc.
v. Events: Important lifecycle events like deployments, scaling etc are logged for debugging.

- IAM

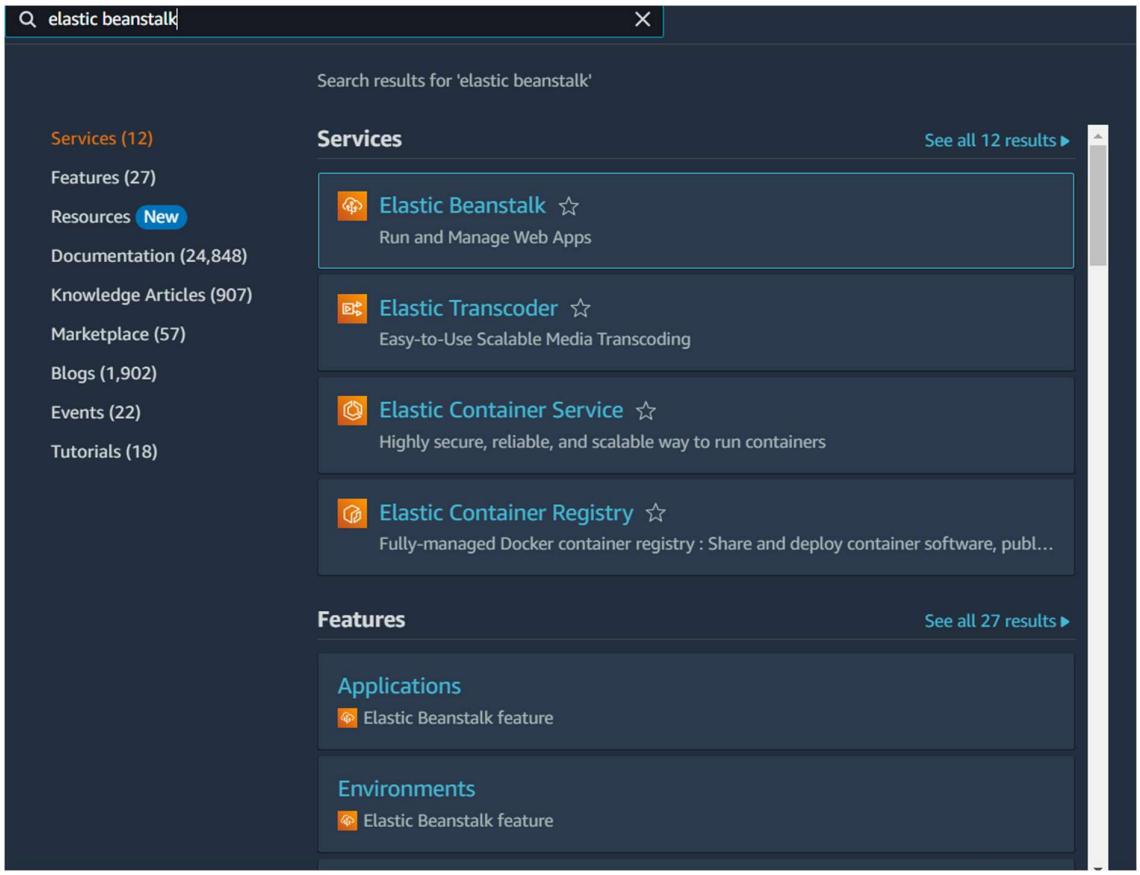
i. IAM allows managing users, roles and permissions to access AWS services and resources.
ii. Users can be created and assigned granular permissions policies.
iii. Roles can be created with permissions and then assigned to AWS resources like EC2 instances.
iv. Policies define the permissions like which AWS actions can be performed on which resources. v. IAM is important for security, access control and compliance in AWS.

v. • Implement PAAS using elastic beanstalk for the following.

1. Server
2. Java
3. Python
4. Node.js

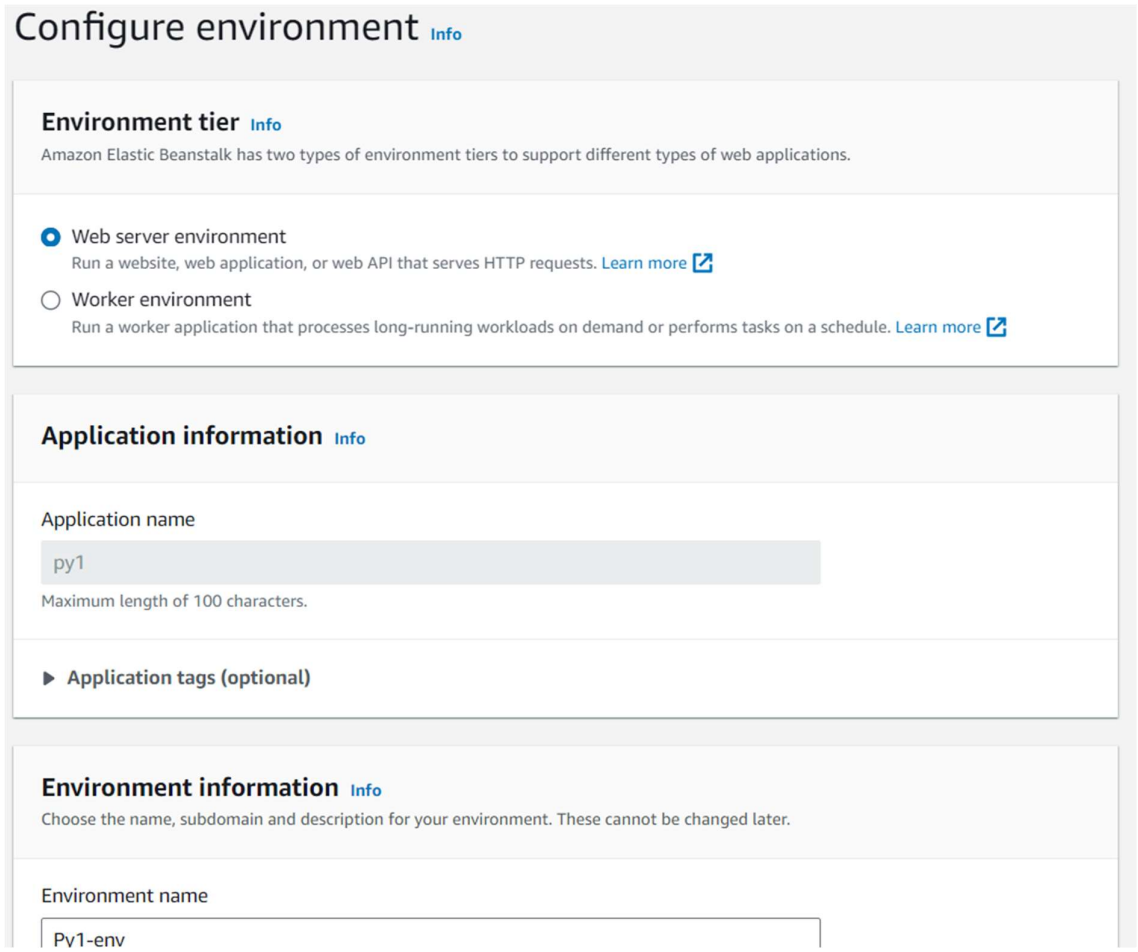
For Python

Step 1: Search for elastic beanstalk

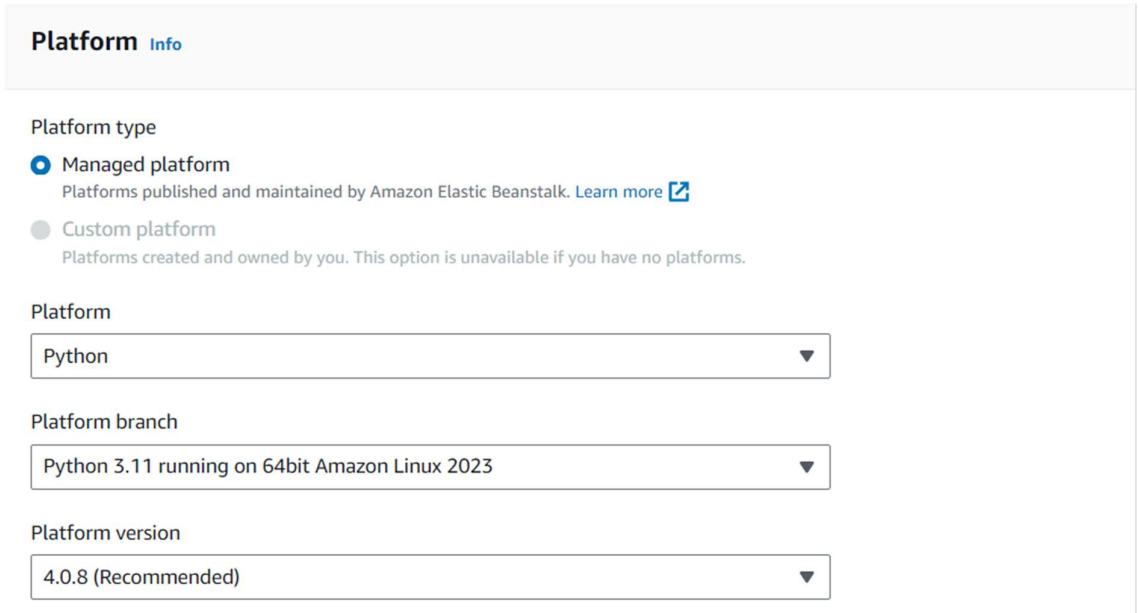


Step 2: Create a new application

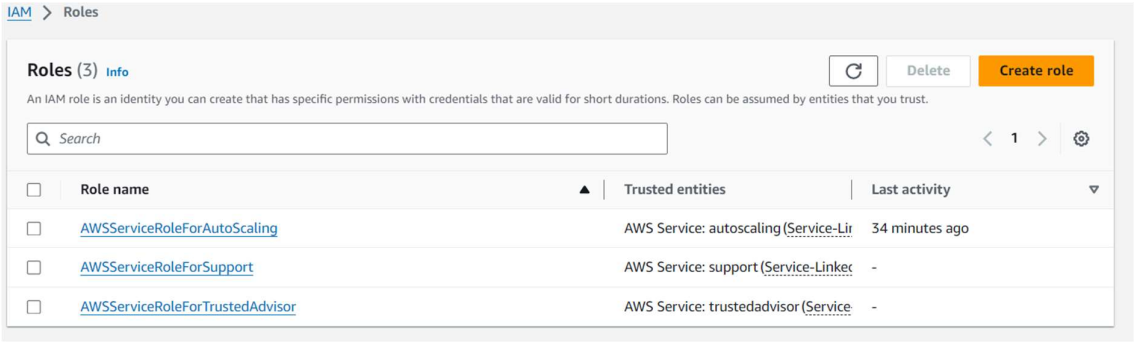
Step 3: Create a new environment



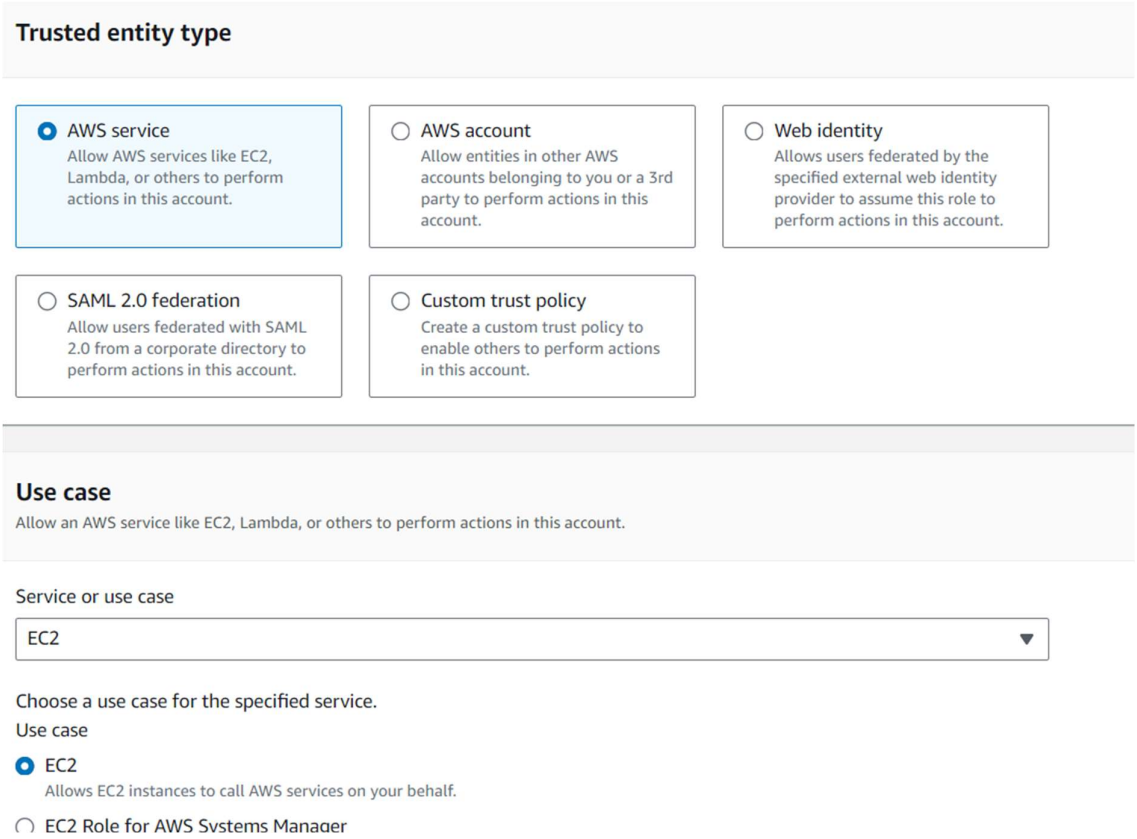
Step 4: Choose python as platform



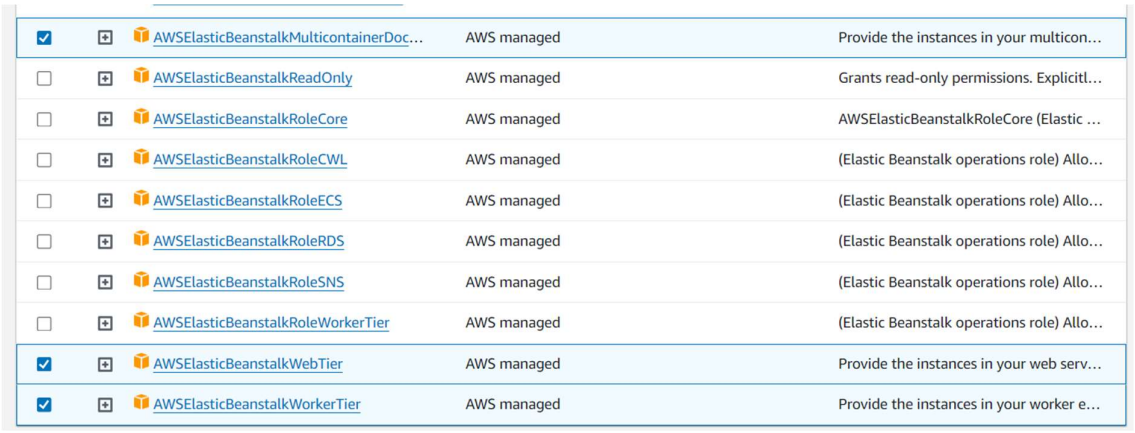
Step 5: Search IAM and create a new role



Step 6: Select EC2 as use case

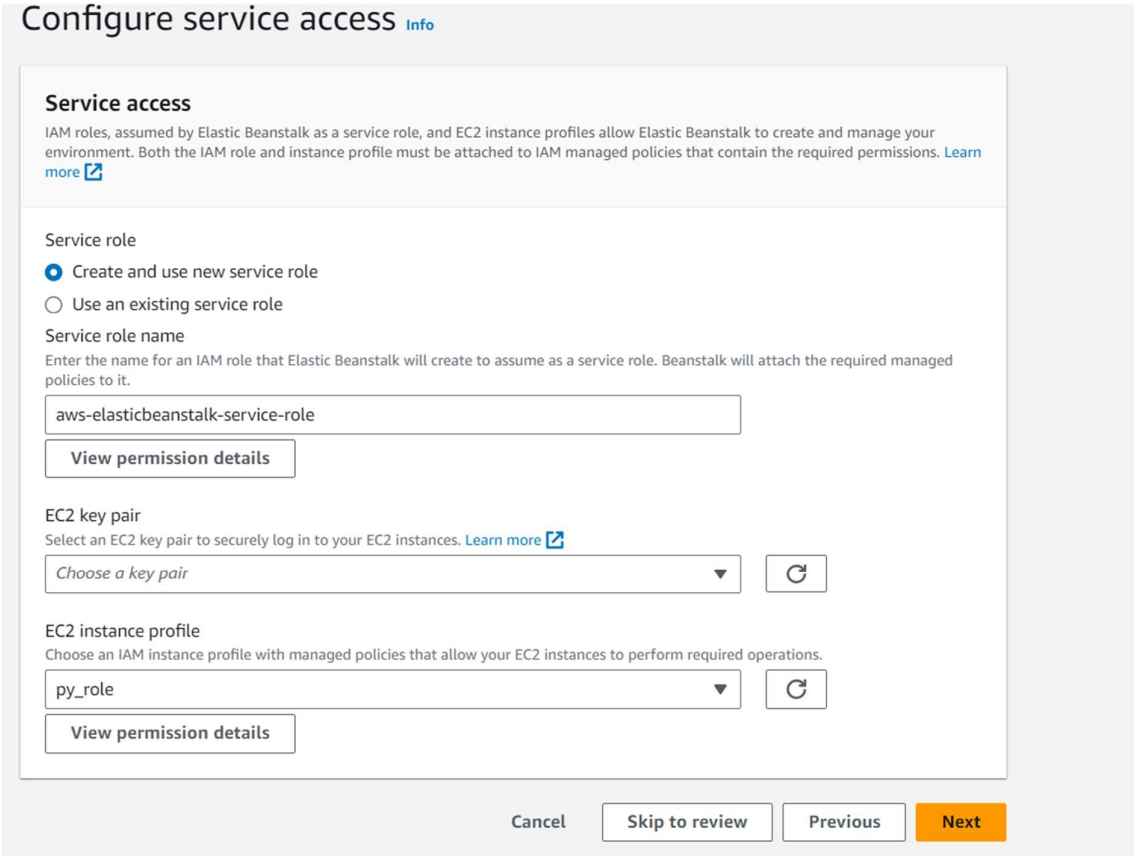


Step 7: Select the following permissions and then create the role

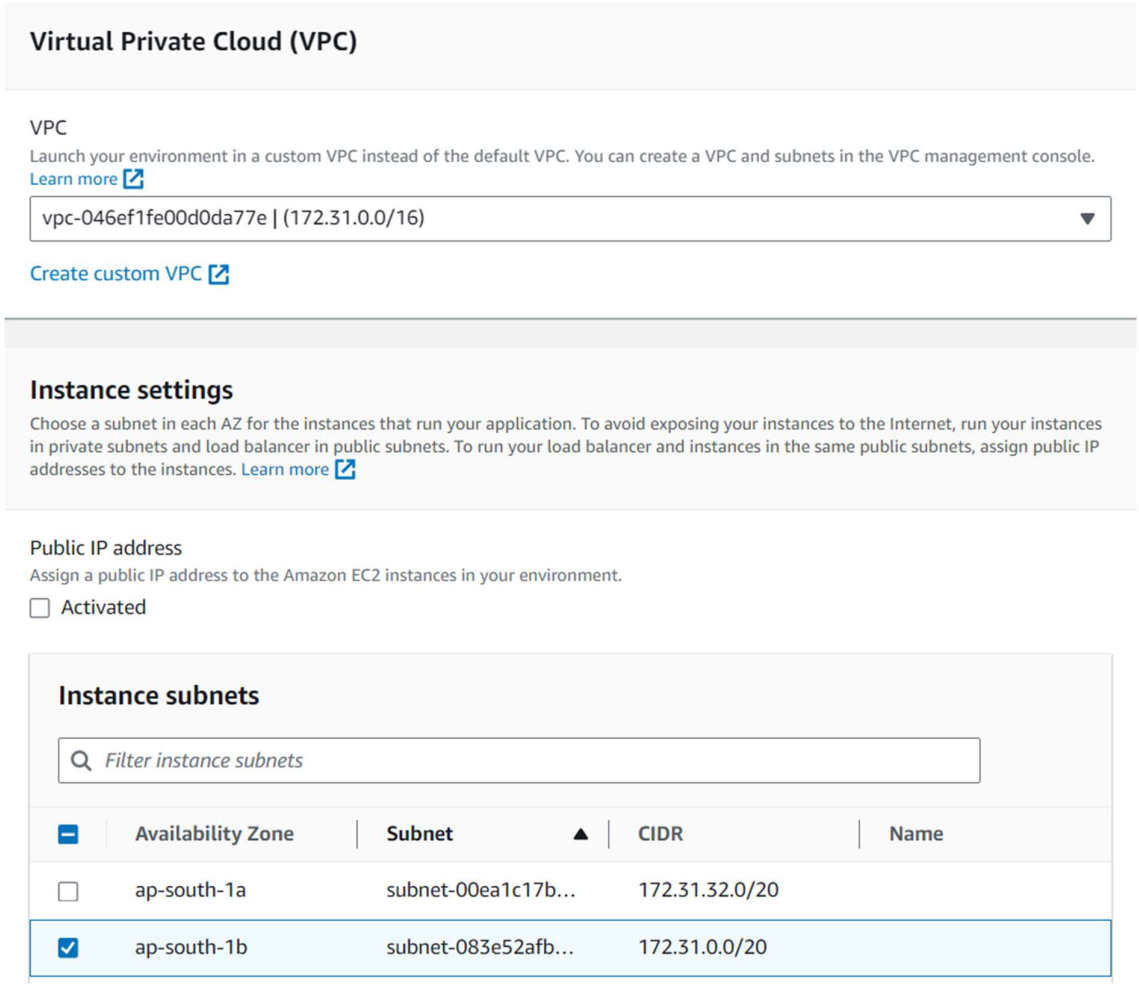


AWSElasticBeanstalkMulticontainerDoc…	AWS managed	Provide the instances in your multicon…
AWSElasticBeanstalkReadOnly	AWS managed	Grants read-only permissions. Explicitl…
AWSElasticBeanstalkRoleCore	AWS managed	AWSElasticBeanstalkRoleCore (Elastic …
AWSElasticBeanstalkRoleCWL	AWS managed	(Elastic Beanstalk operations role) Allo…
AWSElasticBeanstalkRoleECS	AWS managed	(Elastic Beanstalk operations role) Allo…
AWSElasticBeanstalkRoleRDS	AWS managed	(Elastic Beanstalk operations role) Allo…
AWSElasticBeanstalkRoleSNS	AWS managed	(Elastic Beanstalk operations role) Allo…
AWSElasticBeanstalkRoleWorkerTier	AWS managed	(Elastic Beanstalk operations role) Allo…
AWSElasticBeanstalkWebTier	AWS managed	Provide the instances in your web serv…
AWSElasticBeanstalkWorkerTier	AWS managed	Provide the instances in your worker e…

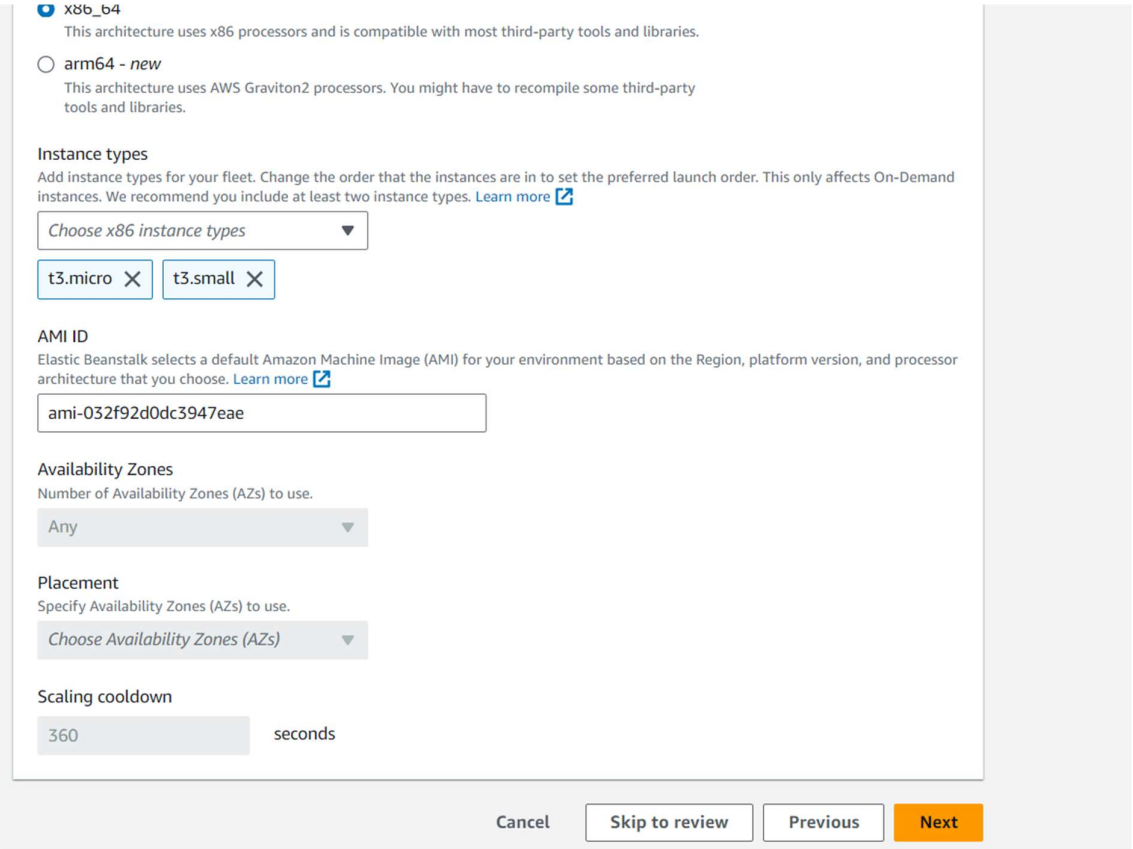
Step 8: Configure service access



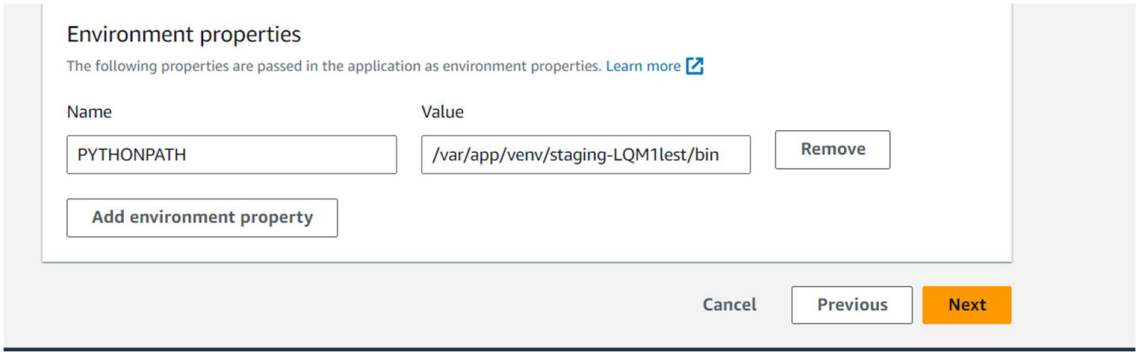
Step 9: Configure VPC



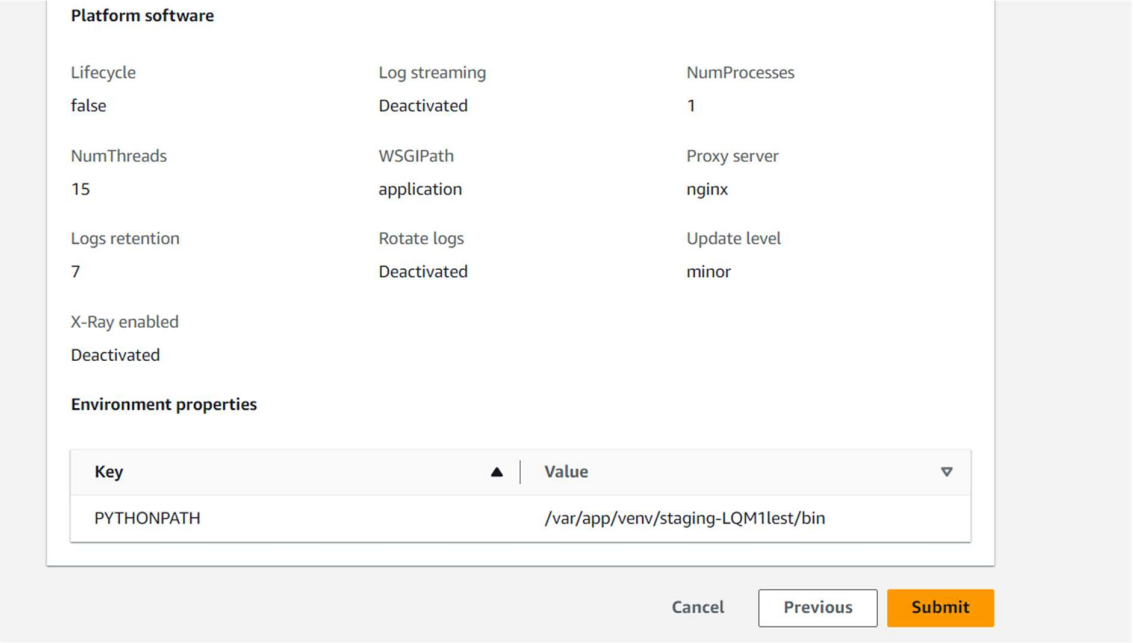
Step 10: Skip the next step without making changes



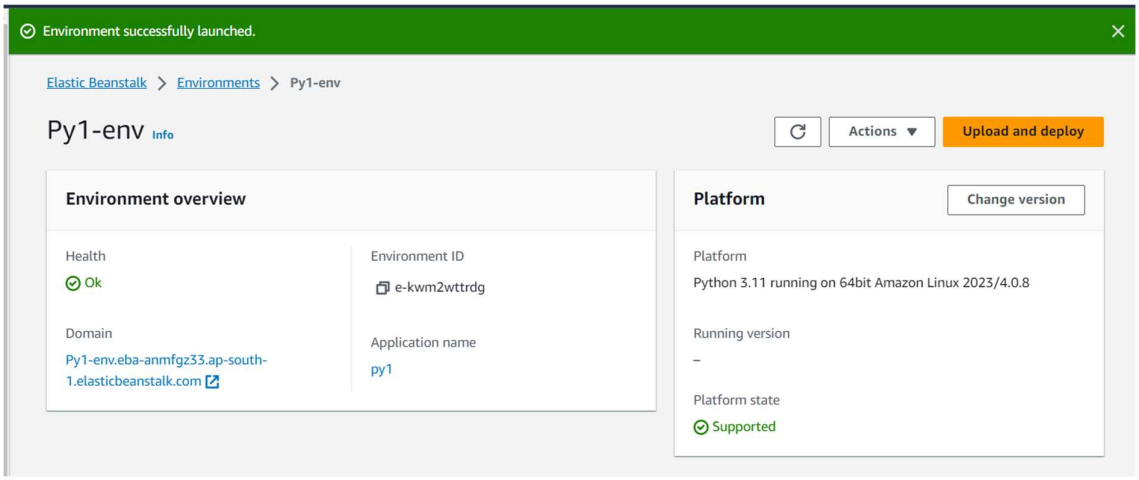
Step 11: Skip the next step too



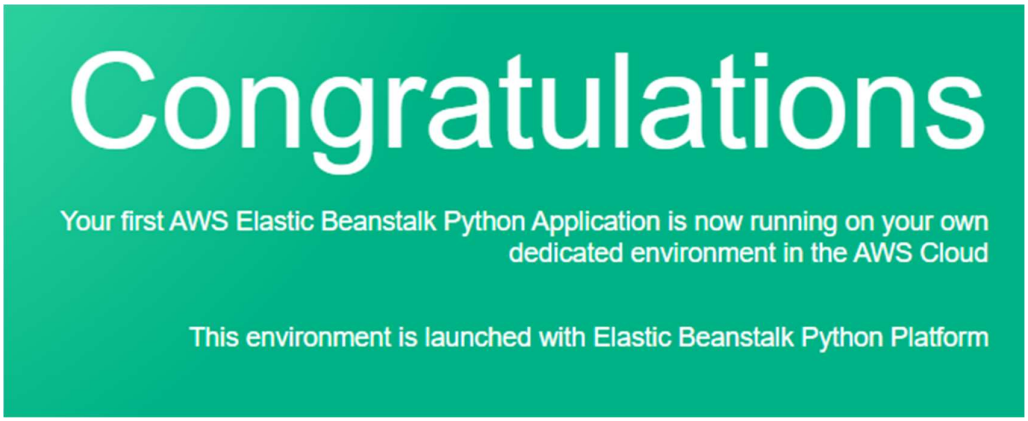
Step 12: Proceed and submit



Step 13: Wait and let the environment launch



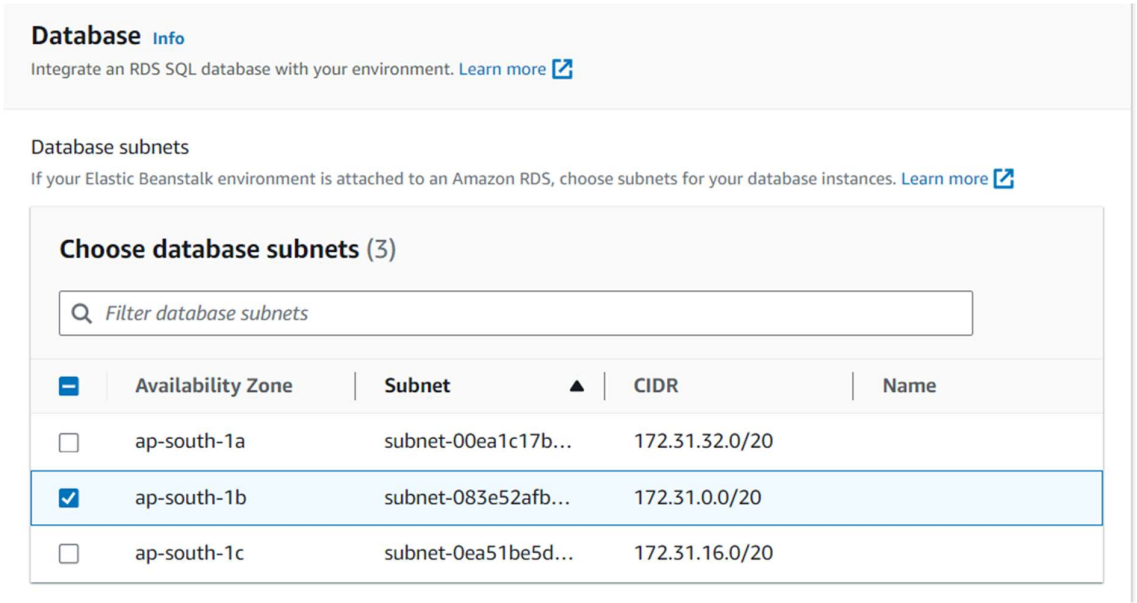
Step 14: Click on the domain link to check whether the environment is launched properly



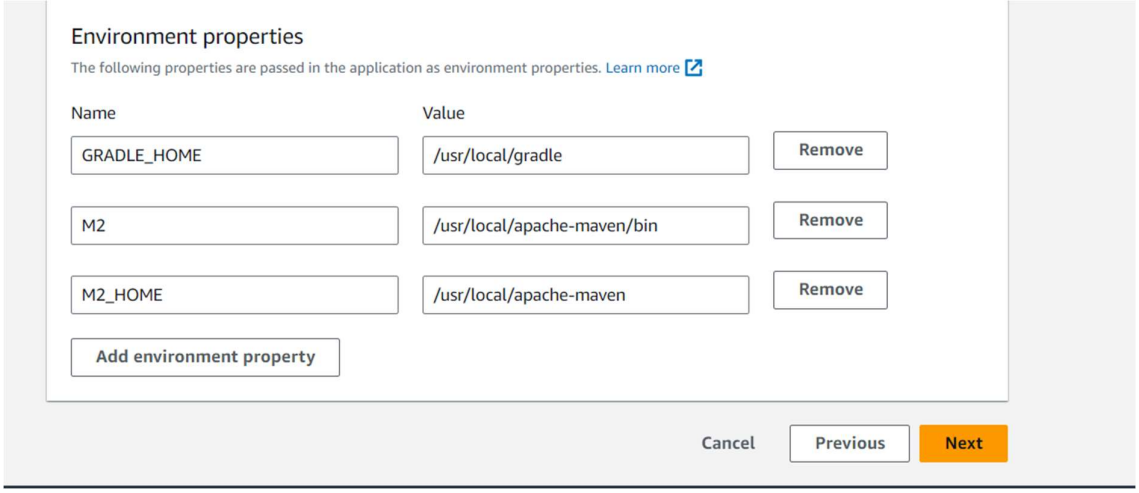
For Java

From Step 1 to Step 8 we repeat the process and from Step 4 we select Java and continue the steps

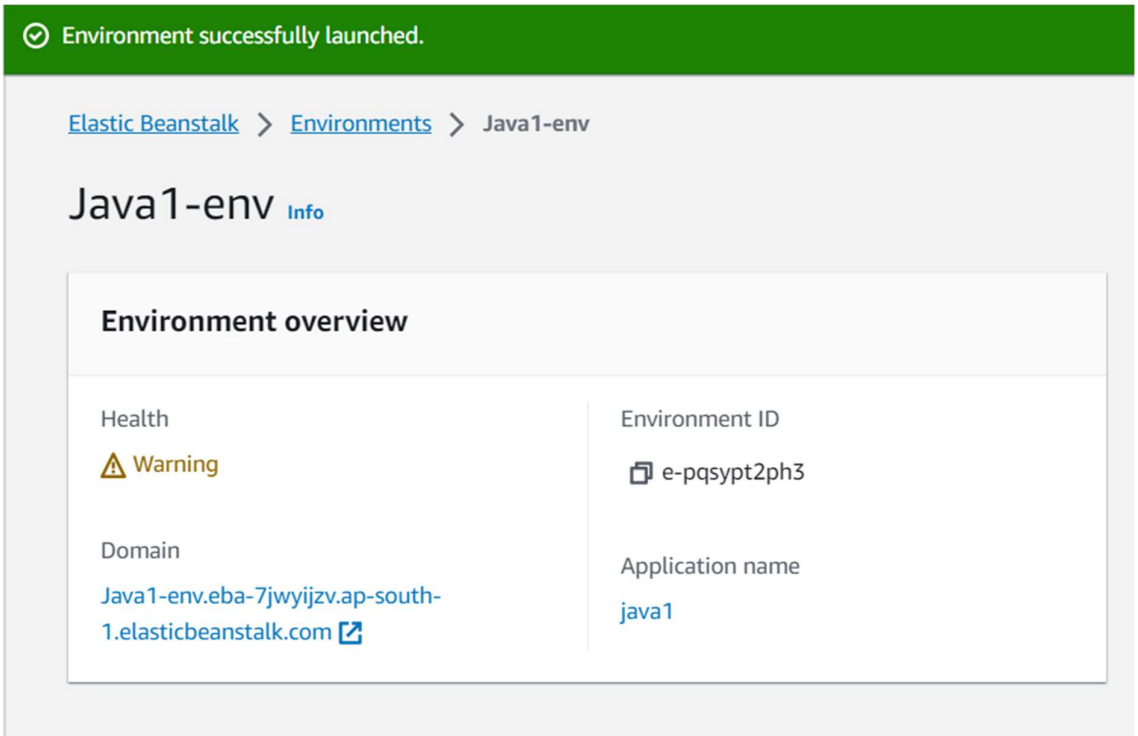
Step 9: Select database subnet



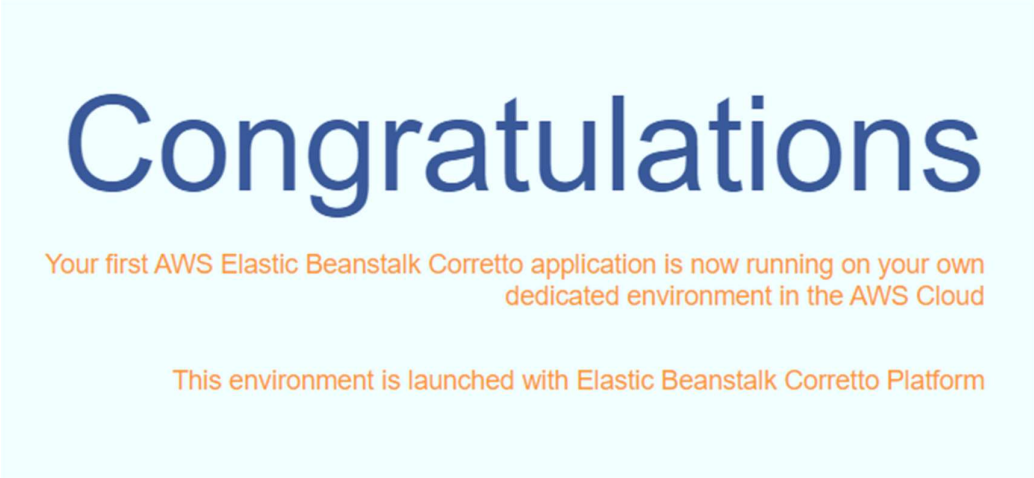
Step 10: Skip the steps as done previously and submit



Step 11: Wait and let the environment launch



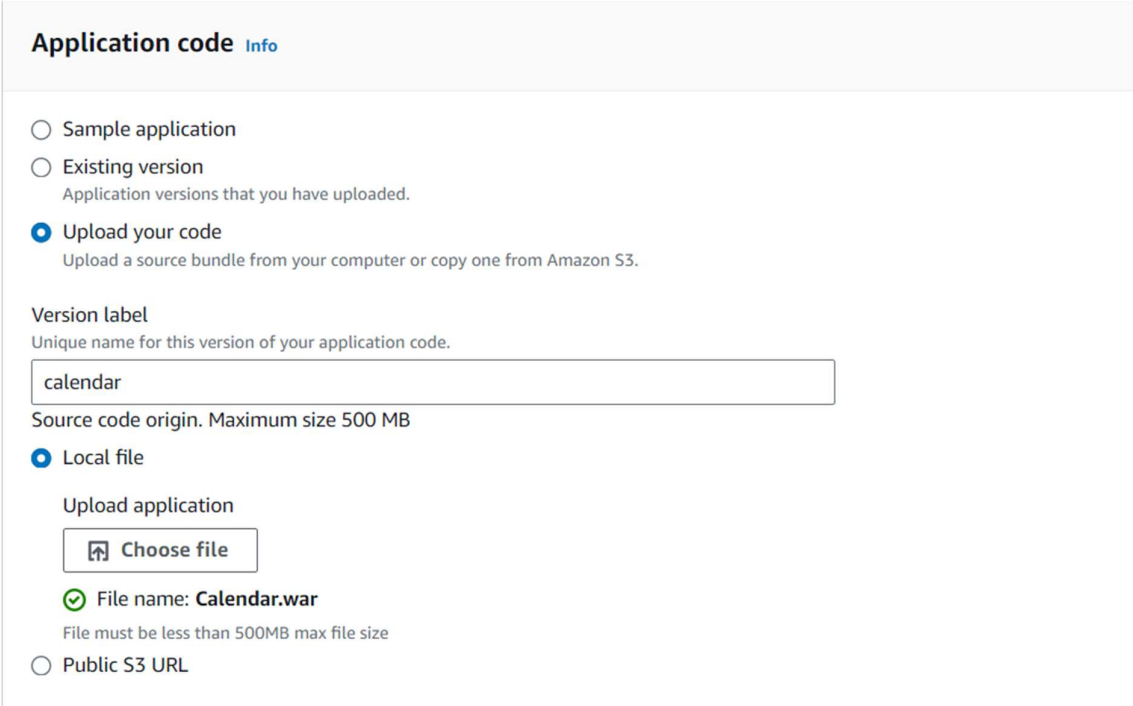
Step 12: Click on domain link to check whether it is getting launched properly



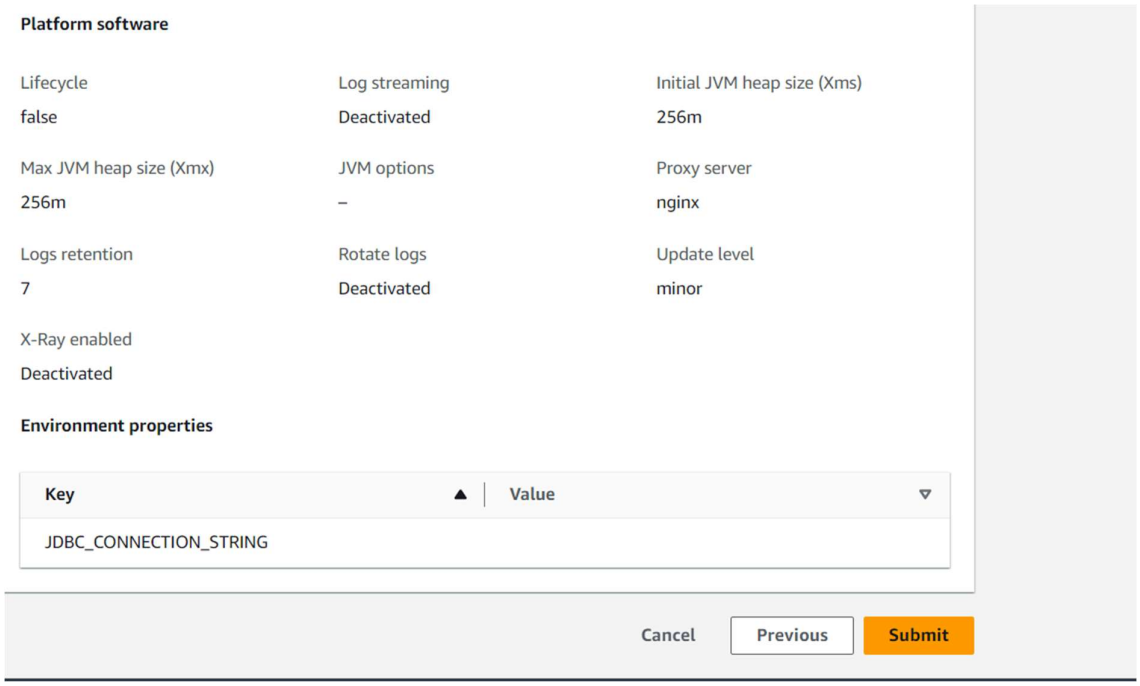
For Tomcat

Repeat the steps by selecting tomcat as platform

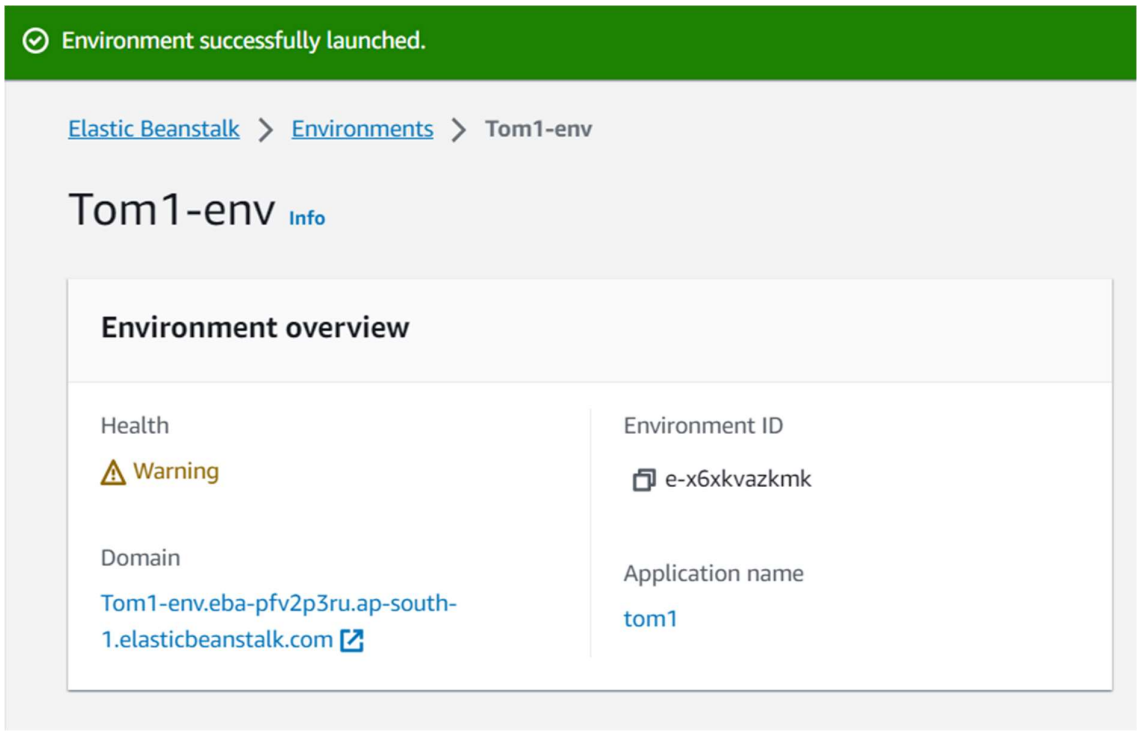
Step1: Upload calendar file in application code



Step 2: Skip the next steps again and submit



Step 3: Wait for the environment to launch



Step 4: Open the domain link to check the application