SWE-455 Homework 1: Cloud Application Deployment

This repository contains a Flask web application that serves as both the deployed application and the homework documentation.

Live Demo

http://swe455-hw1-env.eba-zz3samri.us-east-1.elasticbeanstalk.com/ - *AWS Elastic Beanstalk URL*

Project Overview

This project demonstrates the deployment of a cloud-based web application, fulfilling the requirements of SWE-455 Homework 1. The application serves as both the deliverable and its own documentation, featuring:

- A Flask web application with Tailwind CSS for modern styling
- AWS Cloud deployment with auto-scaling configuration
- AWS CloudWatch monitoring and metrics visualization
- Comprehensive documentation of the deployment process

Technical Stack

- **Backend**: Python Flask
- **Frontend**: HTML with Tailwind CSS
- **Cloud Platform**: Amazon Web Services (AWS)
  - AWS Elastic Beanstalk for application deployment
  - Auto Scaling for handling traffic
  - CloudWatch for monitoring
- **Monitoring**: AWS CloudWatch

Project Structure

- `app.py` - Main Flask application

- `templates/` - HTML templates with Tailwind CSS
- `requirements.txt` - Python dependencies
- `README.md` - Project documentation

Local Development Setup

1. Clone this repository:

   ```
   git clone https://github.com/MR-Alyousif/SWE455-HW1.git
   cd SWE455-HW1
   ```

2. Create and activate a virtual environment:

   ```
   python -m venv .venv
   # Windows
   .\.venv\Scripts\activate
   # Unix/MacOS
   source .venv/bin/activate
   ```

3. Install dependencies:

   ```
   pip install -r requirements.txt
   ```

4. Run the application:

   ```
   python app.py
   ```

5. Visit `http://localhost:5000` in your browser

Cloud Deployment Details

AWS Services Used

- **AWS Elastic Beanstalk**: For hosting the Flask application
- **AWS CodePipeline**: For continuous integration and deployment
- **AWS CodeBuild**: For building the application
- **AWS CloudWatch**: For monitoring application metrics

Deployment Configuration

- **Platform**: Python 3.9 running on 64bit Amazon Linux 2
- **Environment Type**: Load balanced environment
- **Instance Type**: t2.micro
- **Health Reporting**: Enhanced

Auto-scaling Settings

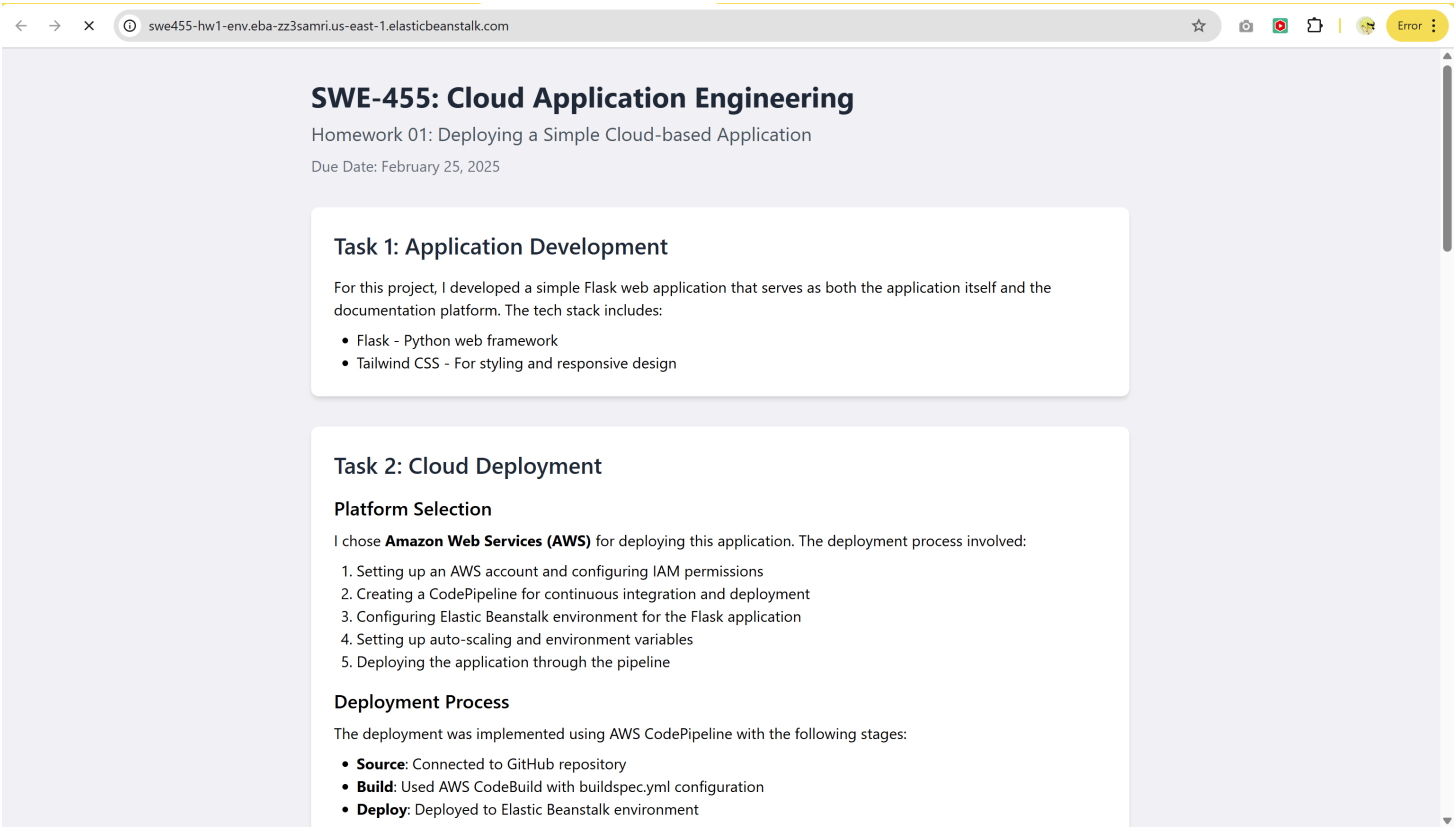
```
aws:autoscaling:asg:
  MinSize: 2
  MaxSize: 5
aws:autoscaling:trigger:
  MeasureName: CPUUtilization
  Statistic: Average
  Unit: Percent
  Period: 300
  BreachDuration: 300
  UpperThreshold: 70
  LowerThreshold: 30
  UpperBreachScaleIncrement: 1
  LowerBreachScaleIncrement: -1
```

Monitoring Setup

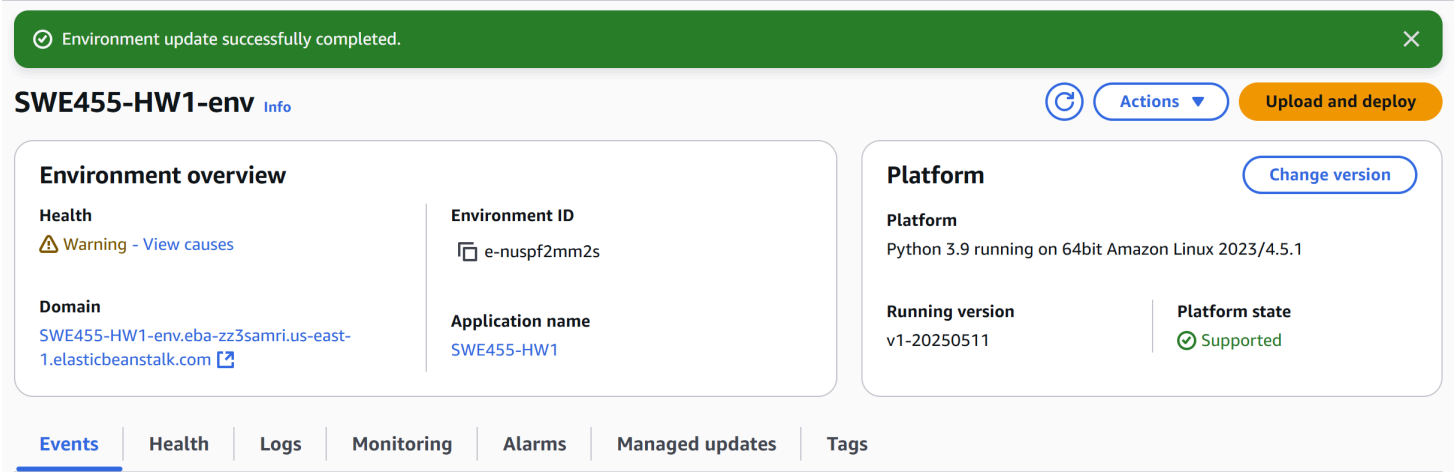
- **CloudWatch Metrics**: CPU Utilization, Network I/O, Request Count
- **Alarms**: Set for high CPU utilization (>80% for 5 minutes)
- **Logs**: Application logs sent to CloudWatch Logs

Screenshots

Application Running in Browser



AWS Dashboard



Monitoring Metrics

Log groups (12)

Actions ▼ | View in Logs Insi

By default, we only load up to 10000 log groups.

Filter log groups or try prefix search | Exact match

Log group	Log class	Anomaly d...
/aws/elasticbeanstalk/SWE455-HW1-env/environment-health.log	Standard	Configure
/aws/elasticbeanstalk/SWE455-HW1-env/var/log/eb-engine.log	Standard	Configure
/aws/elasticbeanstalk/SWE455-HW1-env/var/log/eb-hooks.log	Standard	Configure
/aws/elasticbeanstalk/SWE455-HW1-env/var/log/httpd/access_log	Standard	Configure
/aws/elasticbeanstalk/SWE455-HW1-env/var/log/httpd/error_log	Standard	Configure
/aws/elasticbeanstalk/SWE455-HW1-env/var/log/nginx/access.log	Standard	Configure
/aws/elasticbeanstalk/SWE455-HW1-env/var/log/nginx/error.log	Standard	Configure
/aws/elasticbeanstalk/SWE455-HW1-env/var/log/web.stdout.log	Standard	Configure

Deployment Process

Environment Configuration

Step 1
Configure environment

Step 2
Configure service access

Step 3 - *optional*
Set up networking, database, and tags

Step 4 - *optional*
Configure instance traffic and scaling

Step 5 - *optional*
Configure updates, monitoring, and logging

Step 6
Review

Configure environment Info

Environment tier Info

Amazon Elastic Beanstalk has two types of environment tiers to support different types of web applications.

- (•) Web server environment
  Run a website, web application, or web API that serves HTTP requests. Learn more
- ( ) Worker environment
  Run a worker application that processes long-running workloads on demand or performs tasks on a schedule. Learn more

Application information Info

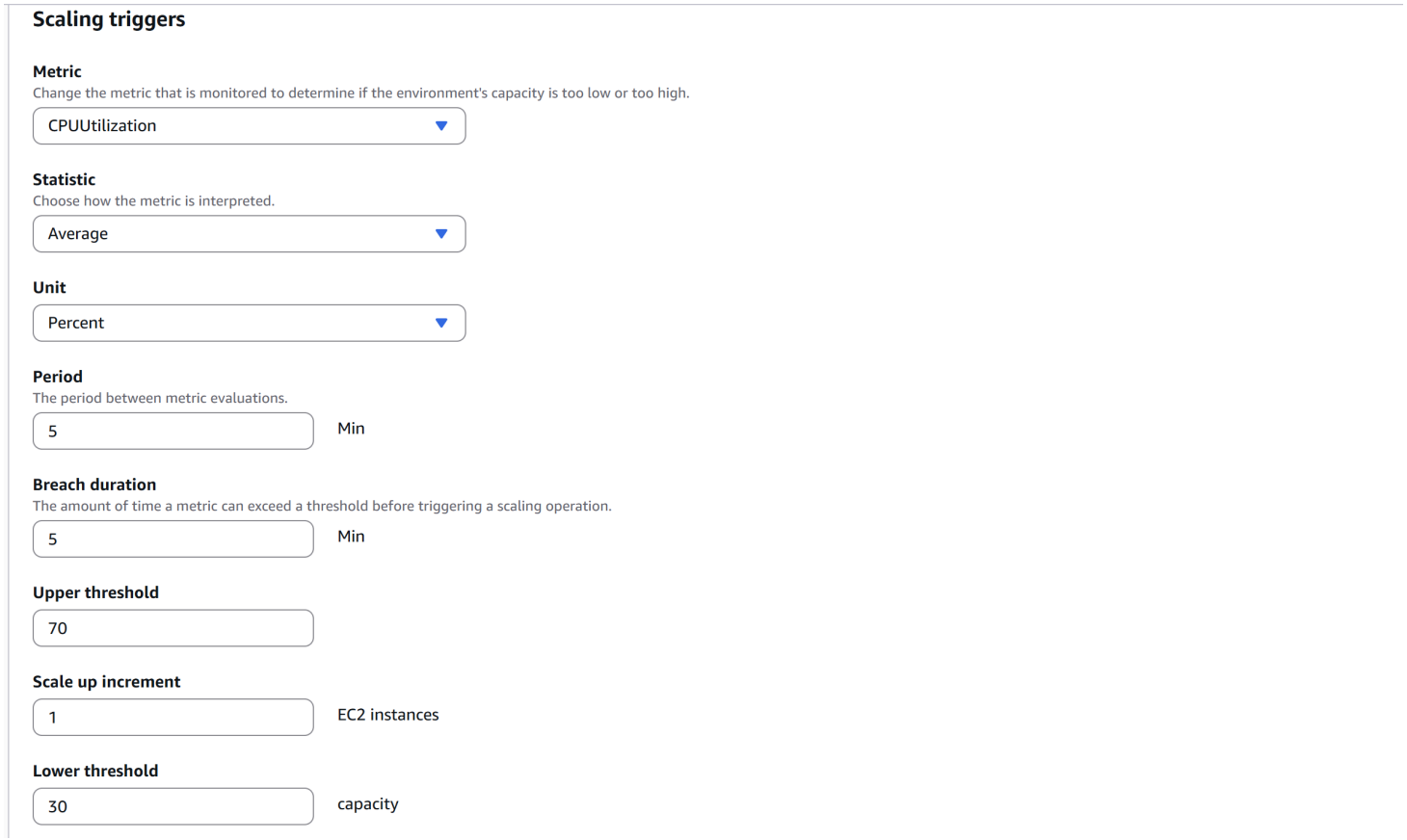
Application name

SWE455-HW1

Maximum length of 100 characters.

▶ Application tags (optional)

Auto-Scaling Configuration



Scaling triggers

Metric
Change the metric that is monitored to determine if the environment's capacity is too low or too high.
CPUUtilization

Statistic
Choose how the metric is interpreted.
Average

Unit
Percent

Period
The period between metric evaluations.
5 Min

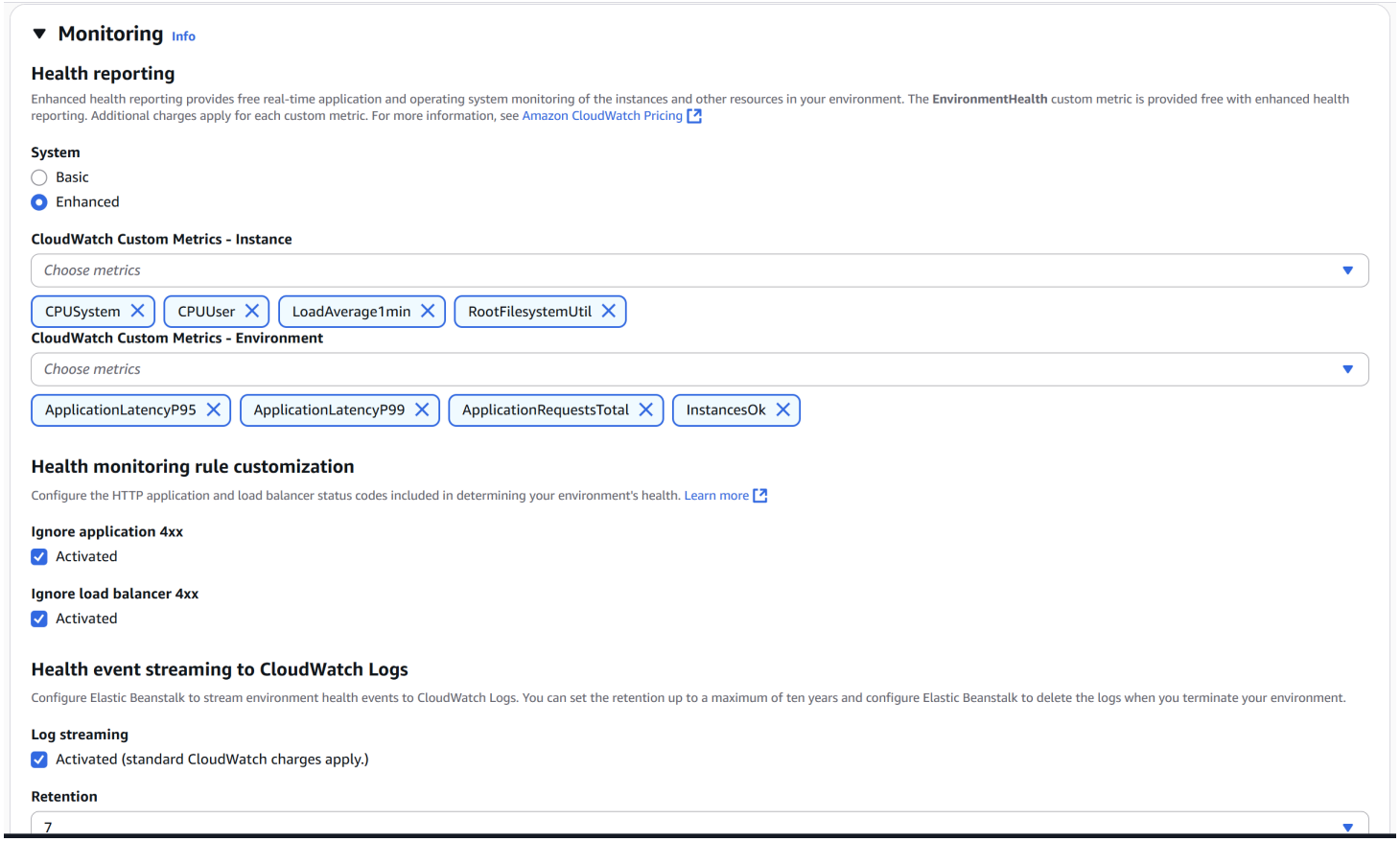
Breach duration
The amount of time a metric can exceed a threshold before triggering a scaling operation.
5 Min

Upper threshold
70

Scale up increment
1 EC2 instances

Lower threshold
30 capacity

Monitoring Configuration



Monitoring Info

Health reporting
Enhanced health reporting provides free real-time application and operating system monitoring of the instances and other resources in your environment. The **EnvironmentHealth** custom metric is provided free with enhanced health reporting. Additional charges apply for each custom metric. For more information, see Amazon CloudWatch Pricing

System
- ○ Basic
- ◉ Enhanced

CloudWatch Custom Metrics - Instance
Choose metrics
CPUSystem × CPUUser × LoadAverage1min × RootFilesystemUtil ×

CloudWatch Custom Metrics - Environment
Choose metrics
ApplicationLatencyP95 × ApplicationLatencyP99 × ApplicationRequestsTotal × InstancesOk ×

Health monitoring rule customization
Configure the HTTP application and load balancer status codes included in determining your environment's health. Learn more

Ignore application 4xx
☑ Activated

Ignore load balancer 4xx
☑ Activated

Health event streaming to CloudWatch Logs
Configure Elastic Beanstalk to stream environment health events to CloudWatch Logs. You can set the retention up to a maximum of ten years and configure Elastic Beanstalk to delete the logs when you terminate your environment.

Log streaming
☑ Activated (standard CloudWatch charges apply.)

Retention
7

Author

Mohammed Alyousif

Submission Details

- **Course**: SWE-455 Cloud Application Engineering
- **Assignment**: Homework 01
- **Due Date**: February 25, 2025
- **Submission Date**: February 25, 2025

Deployment Steps

1. AWS Account Setup

- Created AWS account and configured IAM permissions
- Set up access keys for local development
- Created necessary IAM roles for Elastic Beanstalk service and EC2 instances

2. Local Development and Testing

- Developed Flask application with Tailwind CSS
- Tested locally to ensure functionality
- Created necessary configuration files (.ebextensions, buildspec.yml)

3. AWS Elastic Beanstalk Setup

- Created a new Elastic Beanstalk application
- Configured environment settings (Python platform, load balancing)
- Set up auto-scaling configuration with min 2, max 5 instances
- Configured scaling triggers based on CPU utilization (70% upper threshold, 30% lower threshold)

4. AWS CodePipeline Configuration

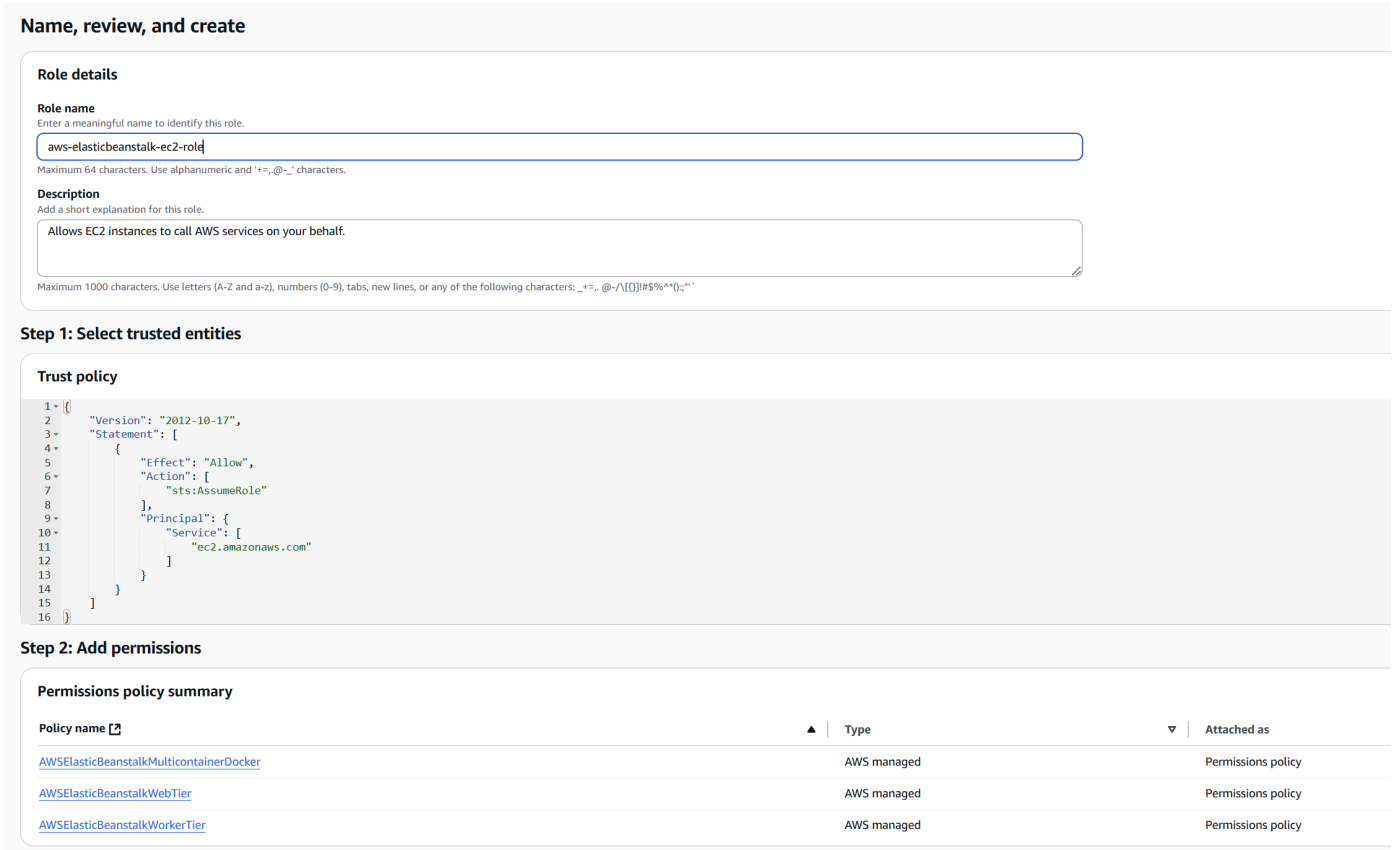
- Created a new pipeline connected to GitHub repository
- Configured source, build, and deploy stages
- Set up webhook for automatic deployments on code changes

5. Monitoring and Testing

- Enabled CloudWatch monitoring with enhanced health reporting
- Configured custom metrics for CPU, memory, requests, and latency
- Set up CloudWatch logs for application monitoring

6. Challenges Encountered

- IAM role configuration issues required creating specific roles for Elastic Beanstalk



- Deployment package configuration needed adjustments for proper static asset handling
- WSGI path configuration required specific settings in the .ebextensions/python.config file