by ACI.

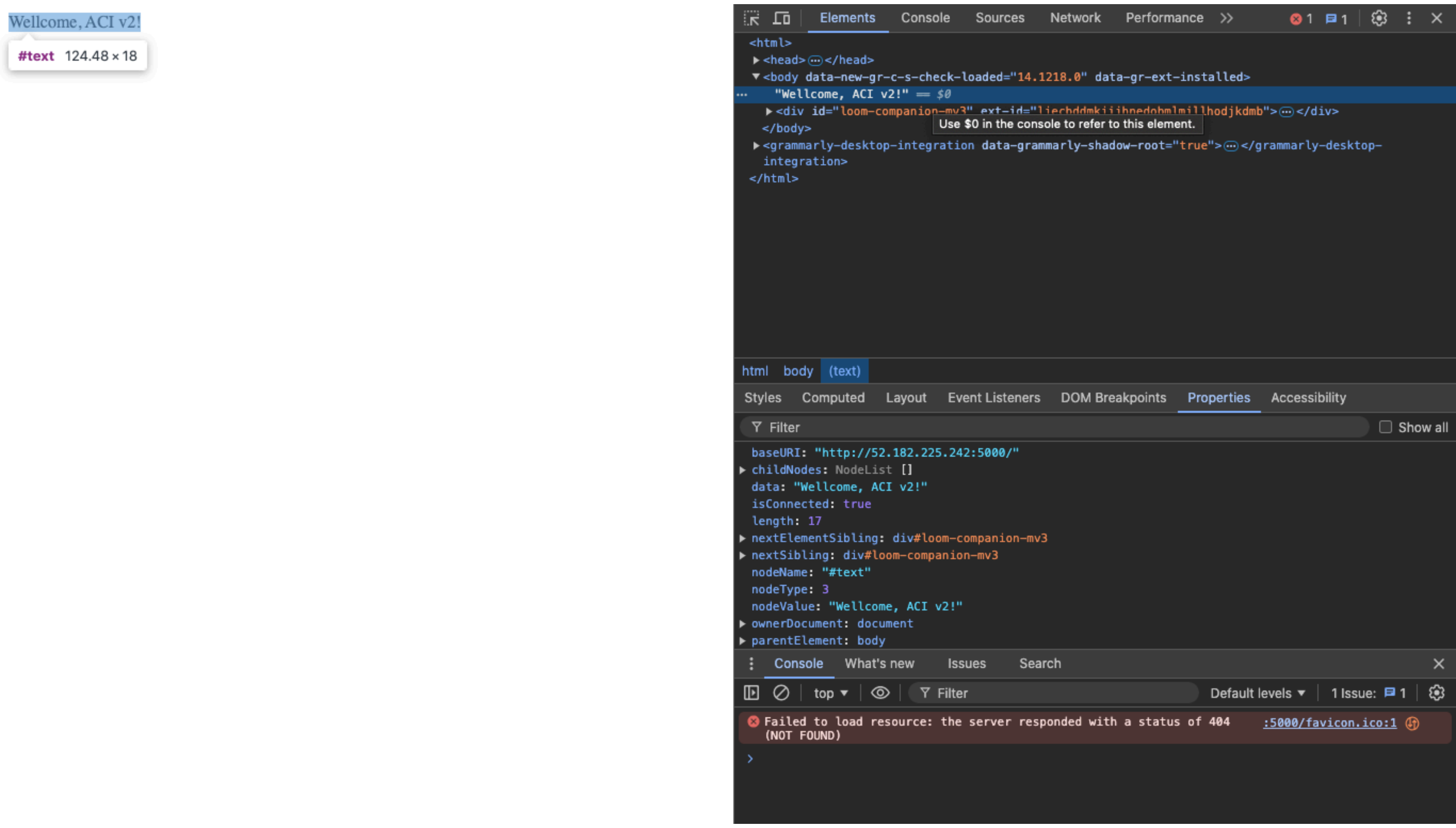
5. Remove the ACI container after verifying the deployment to stop billing.

Practical Task 2: Configure Environment Variables in ACI via Azure Portal

Requirements:

1. Modify your Docker image to read configuration values from environment variables, ensuring minimal environmental complexity.
2. Reuse the ACI instance from Task 1 to deploy the container and specify the necessary environment variables.
3. Verify that the application is correctly using the environment variables by checking its output.
4. Remove the ACI container after verifying that the application correctly uses the environment variables.

Practical Task 3: Scale Out with Azure Container Instances via Azure Portal



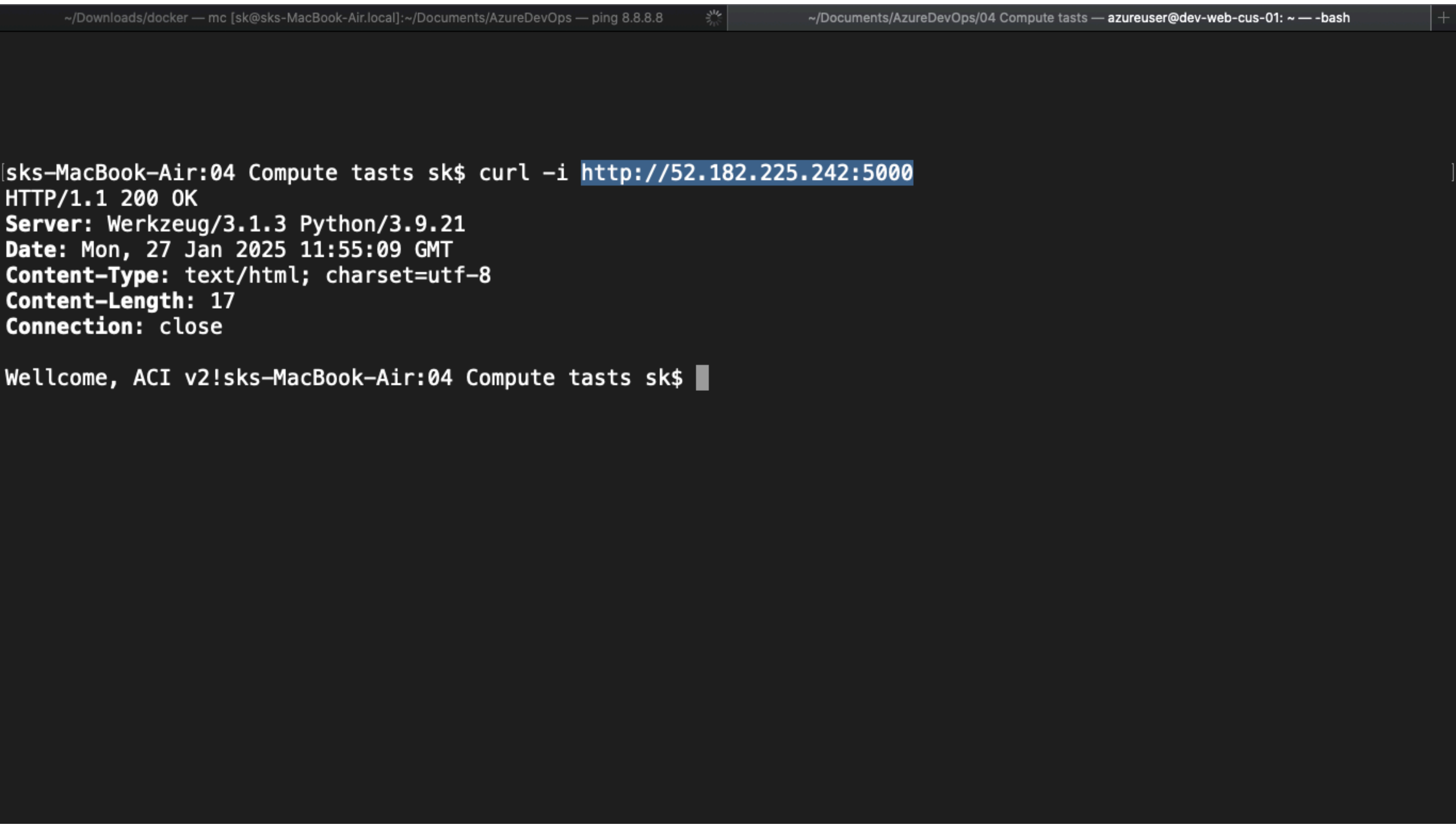
Wellcome, ACI v2!

#text 124.48 × 18

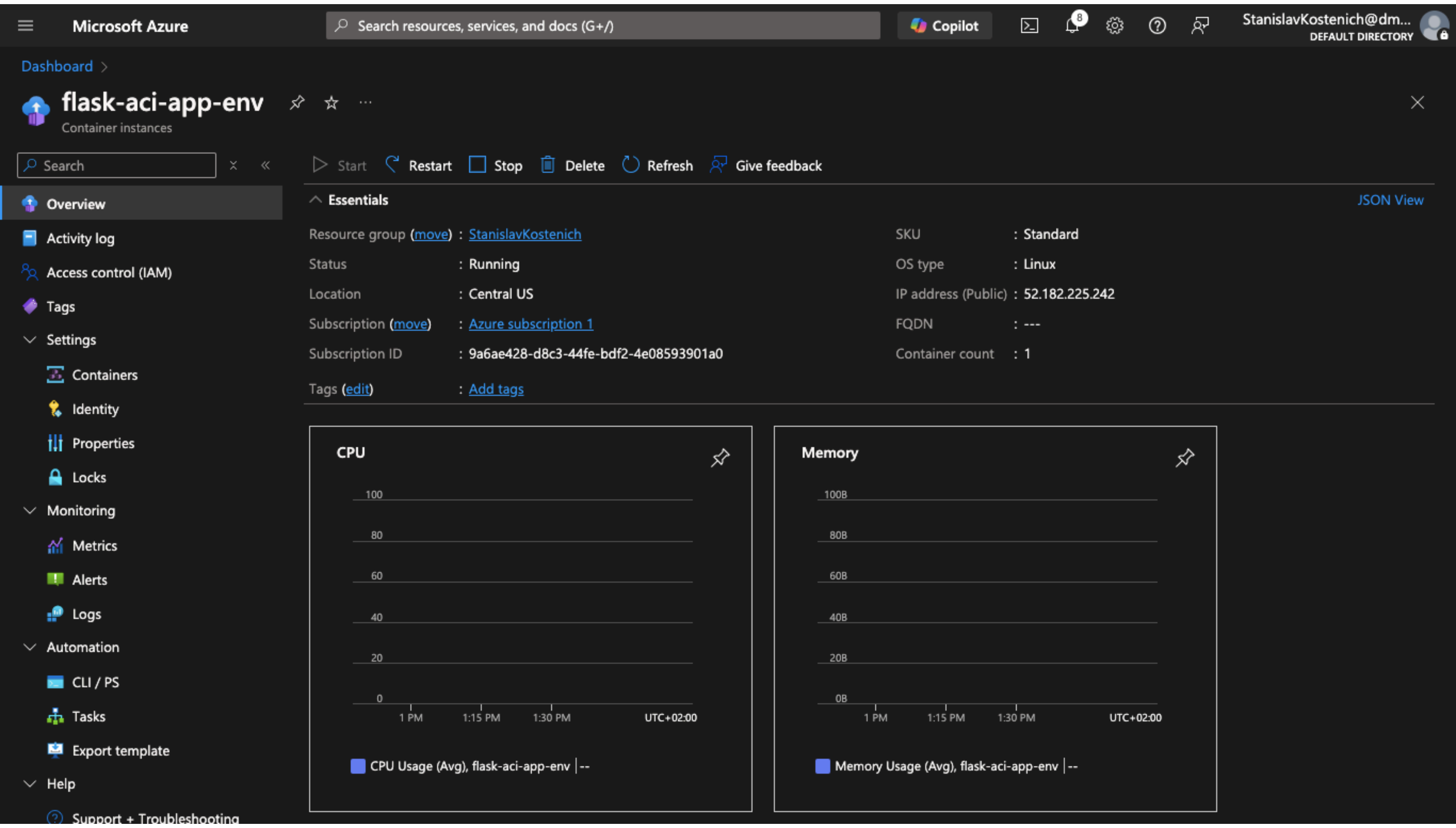
```
<html>
 ▸<head>⋯</head>
 ▾<body data-new-gr-c-s-check-loaded="14.1218.0" data-gr-ext-installed>
    "Wellcome, ACI v2!" == $0
   ▸<div id="loom-companion-mv3" ext-id="liecbddmkiiihnedobmlmillhodjkdmb">⋯</div>
  </body>
 ▸<grammarly-desktop-integration data-grammarly-shadow-root="true">⋯</grammarly-desktop-
  integration>
</html>
```

```
  baseURI: "http://52.182.225.242:5000/"
▸ childNodes: NodeList []
  data: "Wellcome, ACI v2!"
  isConnected: true
  length: 17
▸ nextElementSibling: div#loom-companion-mv3
▸ nextSibling: div#loom-companion-mv3
  nodeName: "#text"
  nodeType: 3
  nodeValue: "Wellcome, ACI v2!"
▸ ownerDocument: document
▸ parentElement: body
```

```
Failed to load resource: the server responded with a status of 404     :5000/favicon.ico:1
(NOT FOUND)
>
```


~/Downloads/docker — mc [sk@sks-MacBook-Air.local]:~/Documents/AzureDevOps — ping 8.8.8.8
~/Documents/AzureDevOps/04 Compute tasts — azureuser@dev-web-cus-01: ~ — -bash

```
sks-MacBook-Air:04 Compute tasts sk$ curl -i http://52.182.225.242:5000
HTTP/1.1 200 OK
Server: Werkzeug/3.1.3 Python/3.9.21
Date: Mon, 27 Jan 2025 11:55:09 GMT
Content-Type: text/html; charset=utf-8
Content-Length: 17
Connection: close

Wellcome, ACI v2!sks-MacBook-Air:04 Compute tasts sk$
```



flask-aci-app-env

Container instances

Resource group (move) : StanislavKostenich

Status : Running

Location : Central US

Subscription (move) : Azure subscription 1

Subscription ID : 9a6ae428-d8c3-44fe-bdf2-4e08593901a0

Tags (edit) : Add tags

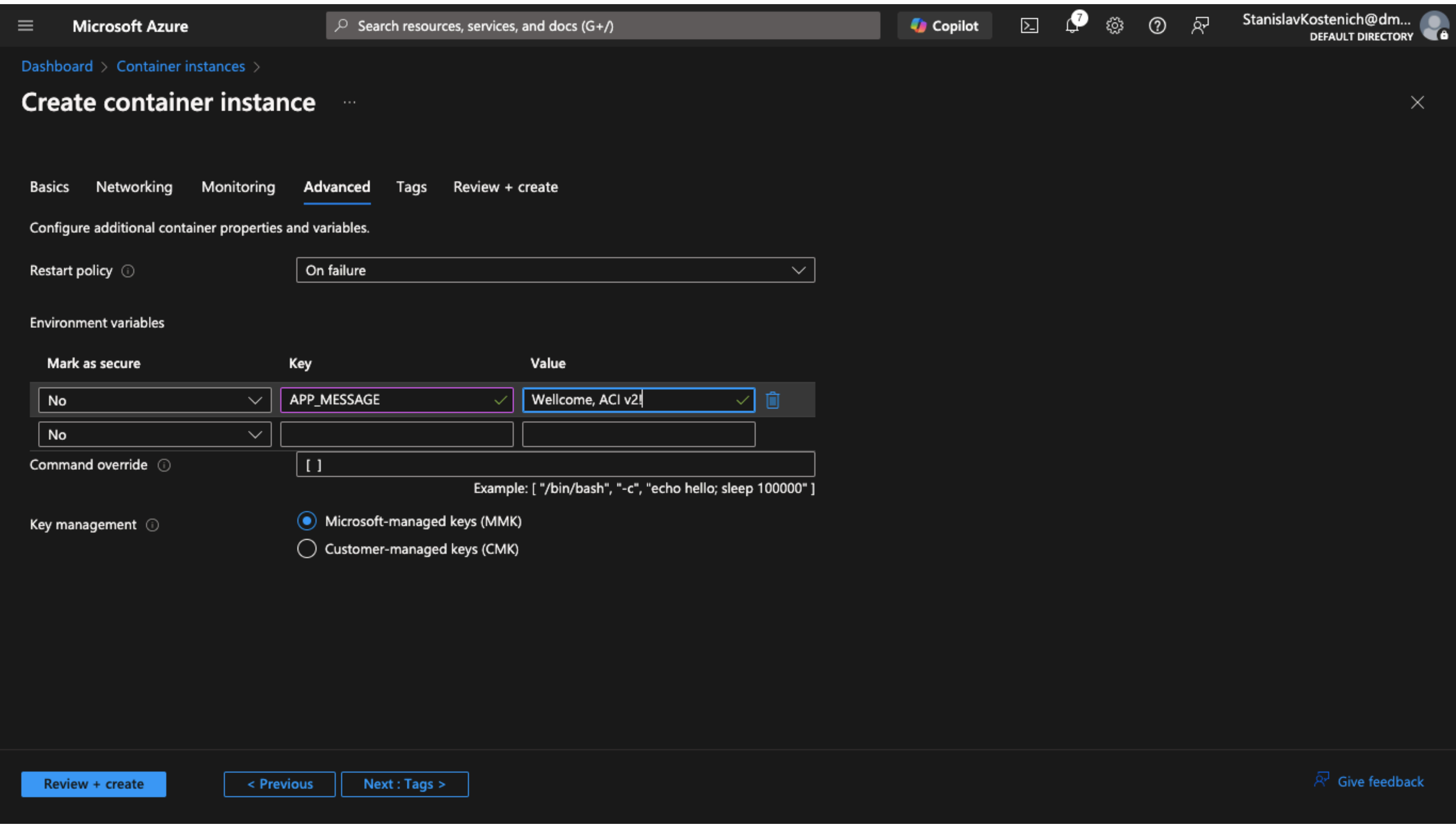
SKU : Standard

OS type : Linux

IP address (Public) : 52.182.225.242

FQDN : ---

Container count : 1



Create container instance

Configure additional container properties and variables.

Restart policy: On failure

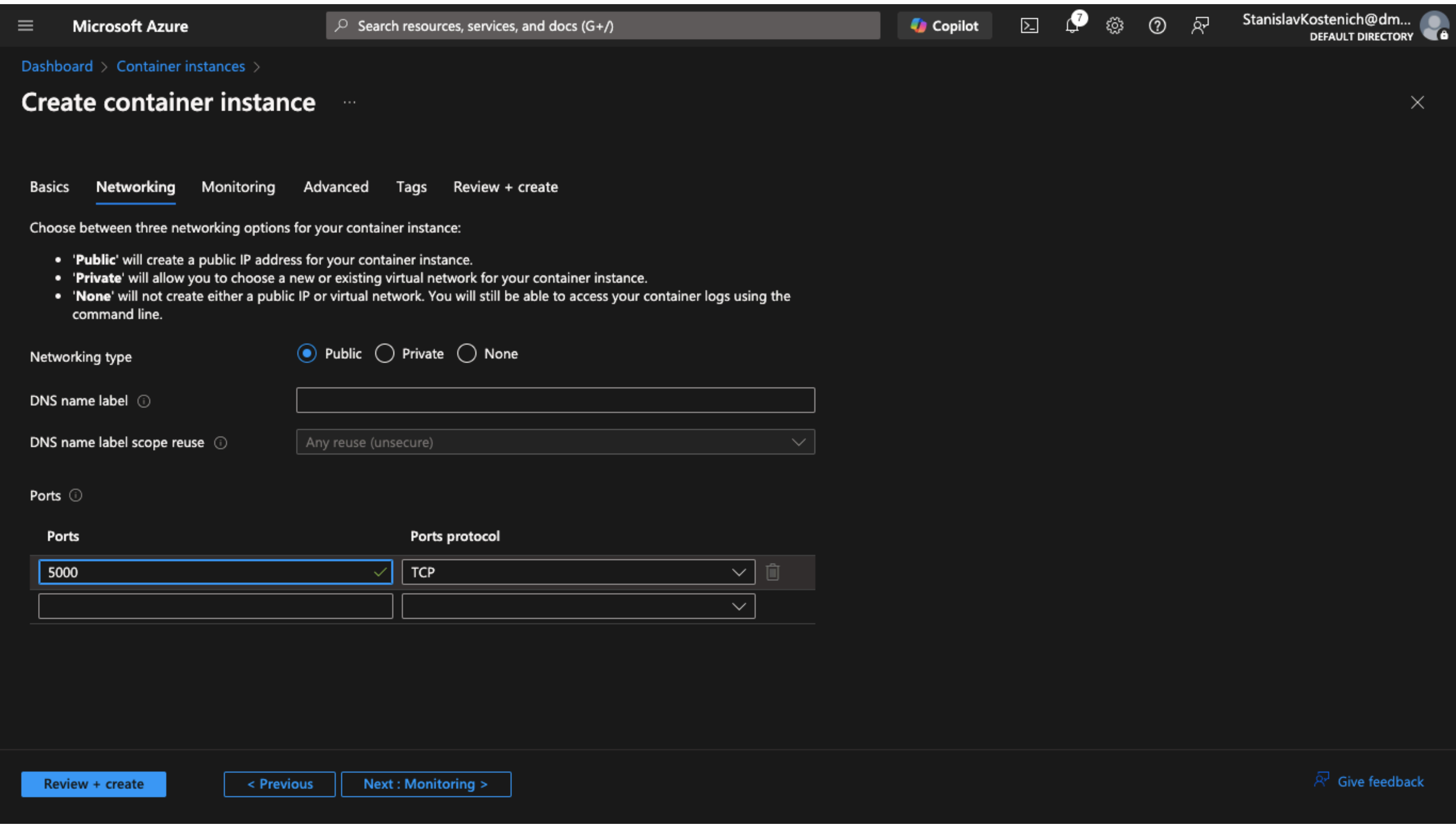
Environment variables

Mark as secure	Key	Value
No	APP_MESSAGE	Wellcome, ACI v2!
No		

Command override: [ ]

Example: ["/bin/bash", "-c", "echo hello; sleep 100000"]

Key management:
● Microsoft-managed keys (MMK)
○ Customer-managed keys (CMK)



Create container instance

Choose between three networking options for your container instance:

- '**Public**' will create a public IP address for your container instance.
- '**Private**' will allow you to choose a new or existing virtual network for your container instance.
- '**None**' will not create either a public IP or virtual network. You will still be able to access your container logs using the command line.

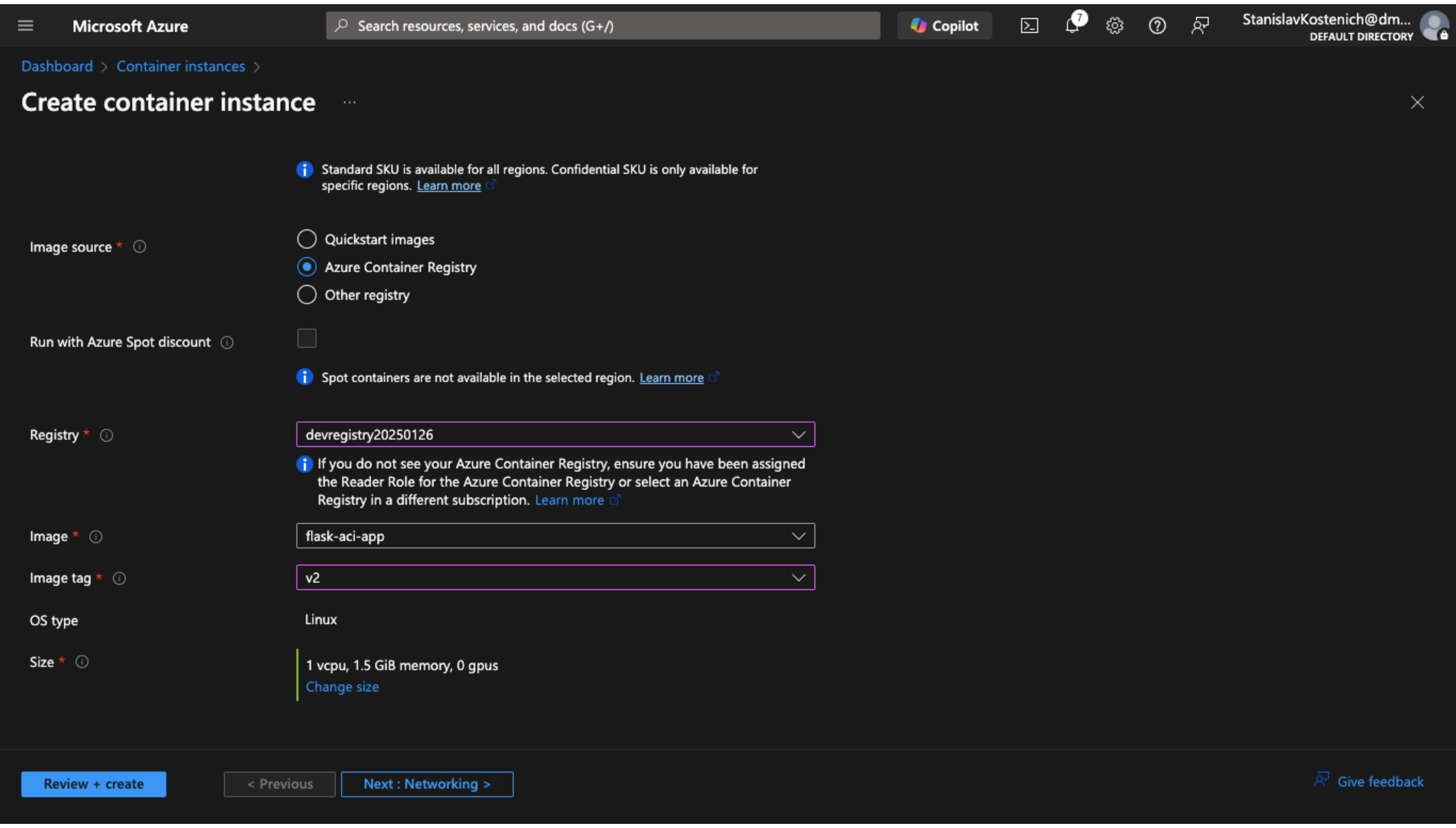
Networking type ● Public ○ Private ○ None

DNS name label

DNS name label scope reuse: Any reuse (unsecure)

Ports

Ports	Ports protocol
5000	TCP


Microsoft Azure
Search resources, services, and docs (G+/)
Copilot
StanislavKostenich@dm...
DEFAULT DIRECTORY
Dashboard > Container instances >

Create container instance

Standard SKU is available for all regions. Confidential SKU is only available for specific regions. Learn more

Image source *
○ Quickstart images
● Azure Container Registry
○ Other registry

Run with Azure Spot discount □

Spot containers are not available in the selected region. Learn more

Registry * devregistry20250126

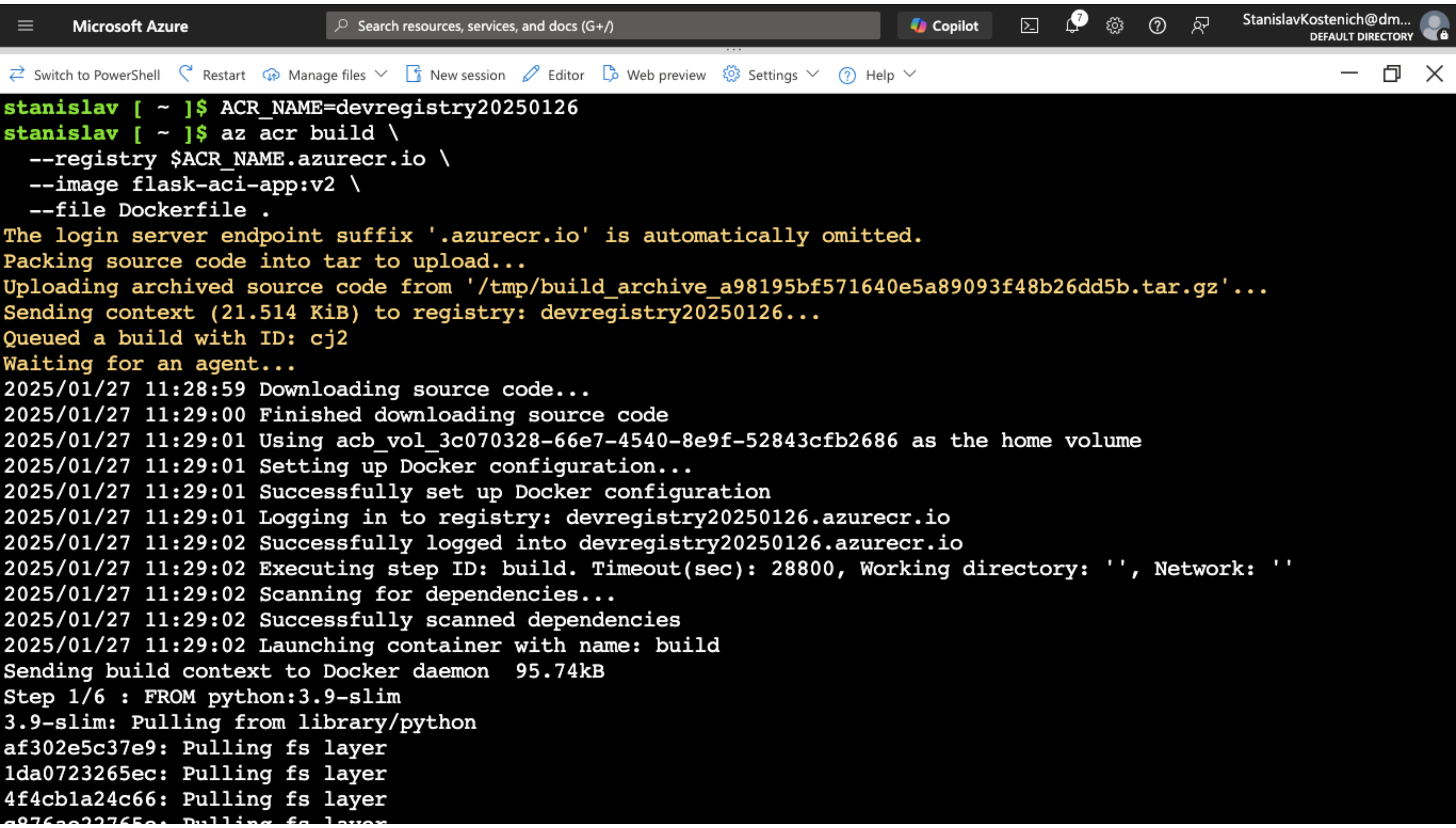
If you do not see your Azure Container Registry, ensure you have been assigned the Reader Role for the Azure Container Registry or select an Azure Container Registry in a different subscription. Learn more

Image * flask-aci-app

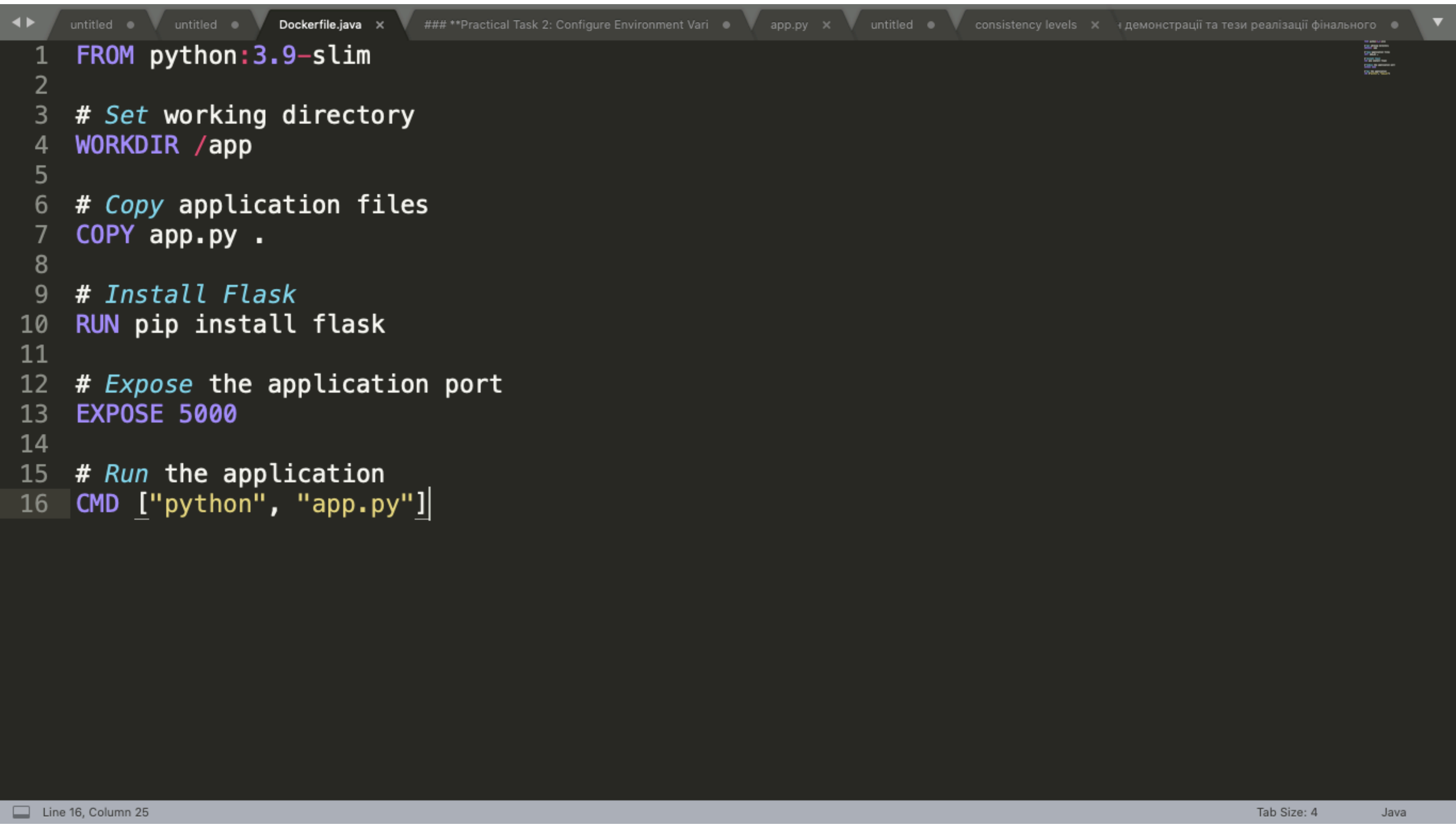
Image tag * v2

OS type Linux

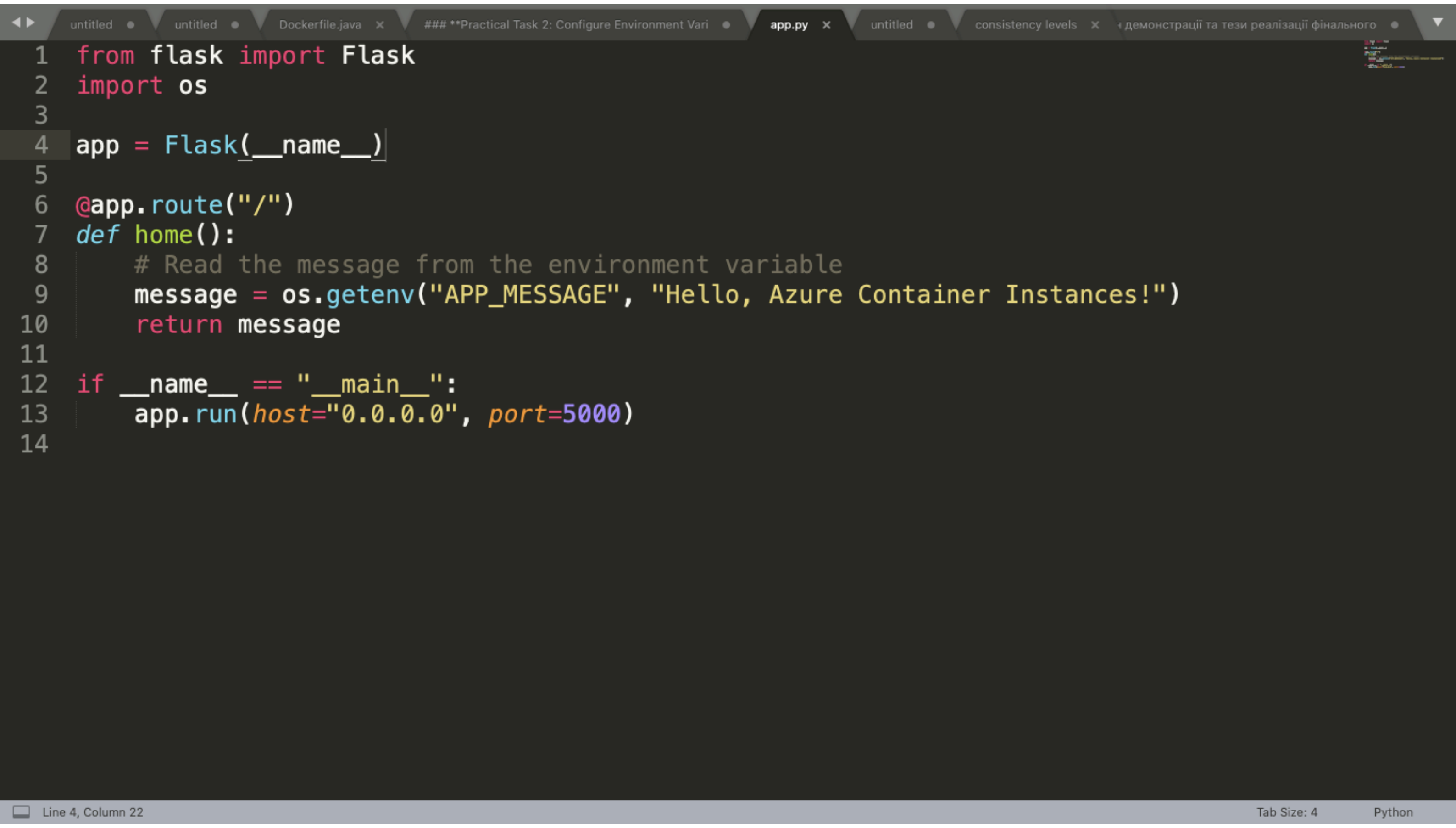
Size * 1 vcpu, 1.5 GiB memory, 0 gpus
Change size



```
stanislav [ ~ ]$ ACR_NAME=devregistry20250126
stanislav [ ~ ]$ az acr build \
  --registry $ACR_NAME.azurecr.io \
  --image flask-aci-app:v2 \
  --file Dockerfile .
The login server endpoint suffix '.azurecr.io' is automatically omitted.
Packing source code into tar to upload...
Uploading archived source code from '/tmp/build_archive_a98195bf571640e5a89093f48b26dd5b.tar.gz'...
Sending context (21.514 KiB) to registry: devregistry20250126...
Queued a build with ID: cj2
Waiting for an agent...
2025/01/27 11:28:59 Downloading source code...
2025/01/27 11:29:00 Finished downloading source code
2025/01/27 11:29:01 Using acb_vol_3c070328-66e7-4540-8e9f-52843cfb2686 as the home volume
2025/01/27 11:29:01 Setting up Docker configuration...
2025/01/27 11:29:01 Successfully set up Docker configuration
2025/01/27 11:29:01 Logging in to registry: devregistry20250126.azurecr.io
2025/01/27 11:29:02 Successfully logged into devregistry20250126.azurecr.io
2025/01/27 11:29:02 Executing step ID: build. Timeout(sec): 28800, Working directory: '', Network: ''
2025/01/27 11:29:02 Scanning for dependencies...
2025/01/27 11:29:02 Successfully scanned dependencies
2025/01/27 11:29:02 Launching container with name: build
Sending build context to Docker daemon  95.74kB
Step 1/6 : FROM python:3.9-slim
3.9-slim: Pulling from library/python
af302e5c37e9: Pulling fs layer
1da0723265ec: Pulling fs layer
4f4cb1a24c66: Pulling fs layer
```



```dockerfile
FROM python:3.9-slim

# Set working directory
WORKDIR /app

# Copy application files
COPY app.py .

# Install Flask
RUN pip install flask

# Expose the application port
EXPOSE 5000

# Run the application
CMD ["python", "app.py"]
```



```python
from flask import Flask
import os

app = Flask(__name__)

@app.route("/")
def home():
    # Read the message from the environment variable
    message = os.getenv("APP_MESSAGE", "Hello, Azure Container Instances!")
    return message

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=5000)

```