Systems in Cloud Computing - Project 2 Report

Francisco Freitas
60313

Teresa Ribeiro
60756

December 2024

1 Introduction

This report documents the adaptation and deployment of the Tukano application, originally developed for the first assignment, to explore Azure's IaaS facilities, specifically using Docker and Kubernetes. The solution maintains the core features while replacing Azure PaaS services with Kubernetes-based alternatives.

2 Implementation

2.1 Code Changes for Hibernate and PostgreSQL Integration

To prepare the application for deployment with Hibernate and PostgreSQL, some modifications were applied to the code. The most relevant being that the `hibernate.cfg.xml` configuration file was updated to adapt PostgreSQL settings for Kubernetes, replacing the configurations originally used for the Azure Resource in the first project. Additionally, the `PostgreDBLayer` class was refactored to utilize Hibernate to perform Create, Read, Update and Delete (CRUD) operations.

Another significant improvement was the addition of authentication for the blob resource. This was implemented using session cookies. Users are now required to authenticate through a dedicated login endpoint at `http://tukano-dns.northeurope.azurecontainer.io:8080/tukano-1/rest/login`, after which a session cookie is retrieved. This cookie is used for all subsequent interactions with the blob service, such as uploading, downloading, or deleting blobs.

2.2 Docker Usage

The Tukano application was containerized using the following commands:

```
docker build -t tukano .
docker tag tukano franciscojrfreitas/tukano
docker push franciscojrfreitas/tukano
```

This created a Docker image, tagged it for Docker Hub, and pushed it to a public repository for deployment.

2.3 Azure Container Instances

The following commands were used to manage Azure Container Instances:

```
az container create --resource-group tukano-container --name tukano-container-cluster \
--image franciscojrfreitas/tukano:latest --ports 8080 --dns-name-label tukano-dns \
--location northeurope --registry-username franciscojrfreitas --registry-password Pass1234! \
--registry-login-server index.docker.io
```

This ensured that the container was deployed with the correct image and accessible via a DNS label.

2.4 Kubernetes Deployment

The application and its dependencies were deployed to the Azure Kubernetes Service (AKS). The commands used include:

```
kubectl apply -f postgres-pvc.yaml
kubectl apply -f postgres.yaml
kubectl apply -f blob-pvc.yaml
kubectl apply -f blobs.yaml
kubectl apply -f webapp.yaml
```

These commands configured Persistent Volumes, PostgreSQL, and the Tukano web application.

2.5 PostgreSQL in Kubernetes

PostgreSQL was deployed using Kubernetes with the following configurations:

- The environment variables for the credentials of the database were defined in `postgres.yaml`.
- Persistent Volumes were used for data storage.

3 Performance Testing

To evaluate the performance of the system, three main test scenarios were performed using Artillery. Each test scenario was designed to simulate user interactions with the Tukano application and measure its performance in terms of throughput, latency, and resource utilization.

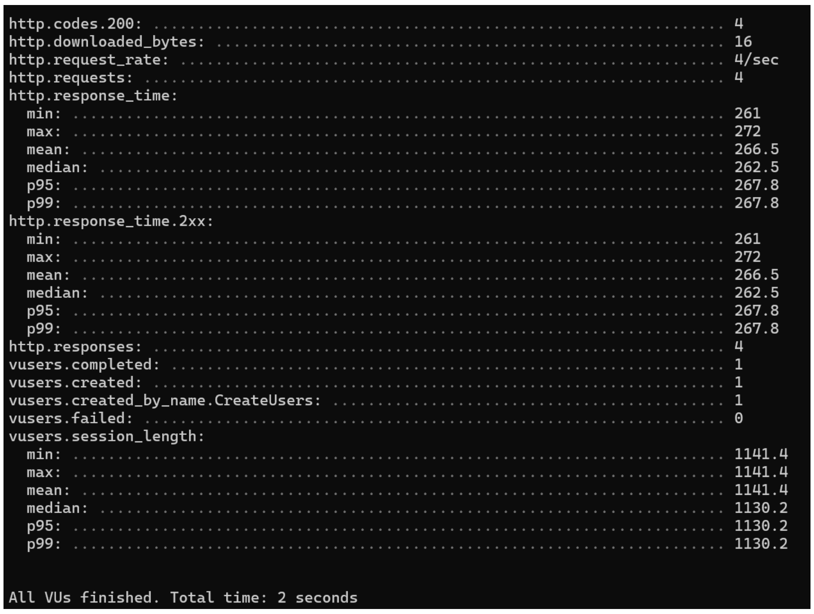
3.1 Test Setup

The performance tests targeted the deployed application at the endpoint
`http://tukano-dns.northeurope.azurecontainer.io:8080/tukano-1/rest`.

The tests included scenarios for creating, deleting, and interacting with user resources and blob data. They were configured to use the `helpers.js` file, which implemented session authentication, token extraction, and binary payload loading. Three primary scenarios were tested: users and shorts creation and deletion, blob upload and download operations.

3.2 User Creation and Deletion Test

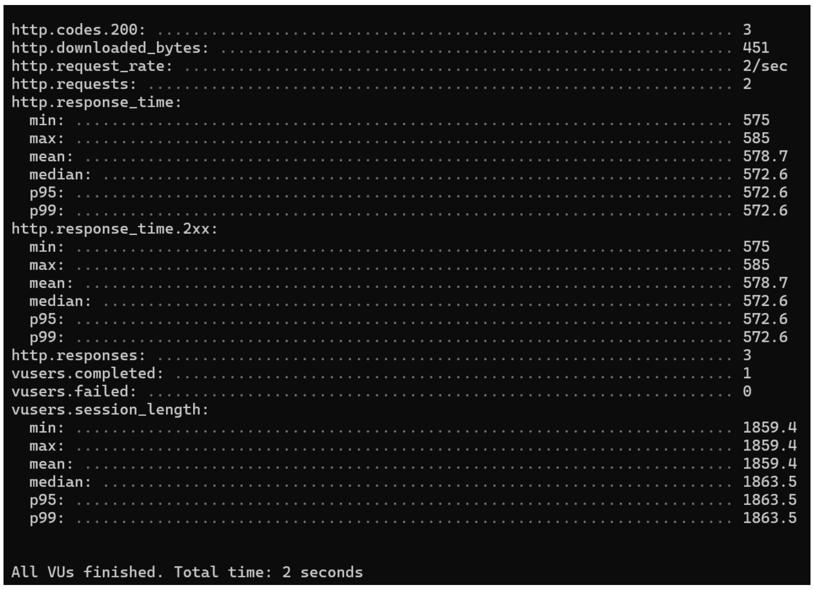
This test focused on creating users, validating their credentials, and subsequently deleting them. The Artillery configuration captured response times and throughput for each operation. The test verified that the application correctly handled session cookies and database operations for user management.



```
http.codes.200: ............................................................... 4
http.downloaded_bytes: ........................................................ 16
http.request_rate: ............................................................ 4/sec
http.requests: ................................................................ 4
http.response_time:
  min: ........................................................................ 261
  max: ........................................................................ 272
  mean: ....................................................................... 266.5
  median: ..................................................................... 262.5
  p95: ........................................................................ 267.8
  p99: ........................................................................ 267.8
http.response_time.2xx:
  min: ........................................................................ 261
  max: ........................................................................ 272
  mean: ....................................................................... 266.5
  median: ..................................................................... 262.5
  p95: ........................................................................ 267.8
  p99: ........................................................................ 267.8
http.responses: ............................................................... 4
vusers.completed: ............................................................. 1
vusers.created: ............................................................... 1
vusers.created_by_name.CreateUsers: ........................................... 1
vusers.failed: ................................................................ 0
vusers.session_length:
  min: ........................................................................ 1141.4
  max: ........................................................................ 1141.4
  mean: ....................................................................... 1141.4
  median: ..................................................................... 1130.2
  p95: ........................................................................ 1130.2
  p99: ........................................................................ 1130.2


All VUs finished. Total time: 2 seconds
```

Figure 1: Response Time for User Creation



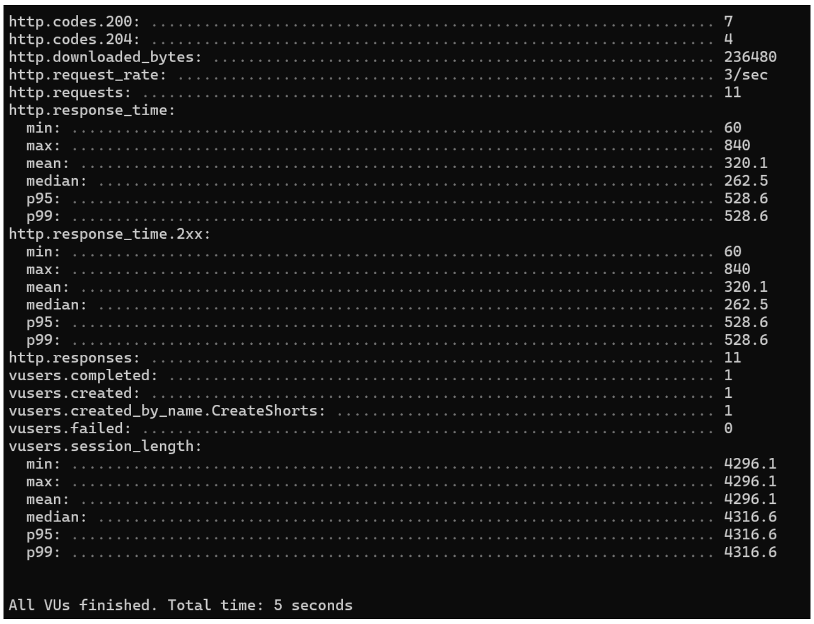
```
http.codes.200: ............................................................... 3
http.downloaded_bytes: ........................................................ 451
http.request_rate: ............................................................ 2/sec
http.requests: ................................................................ 2
http.response_time:
  min: ........................................................................ 575
  max: ........................................................................ 585
  mean: ....................................................................... 578.7
  median: ..................................................................... 572.6
  p95: ........................................................................ 572.6
  p99: ........................................................................ 572.6
http.response_time.2xx:
  min: ........................................................................ 575
  max: ........................................................................ 585
  mean: ....................................................................... 578.7
  median: ..................................................................... 572.6
  p95: ........................................................................ 572.6
  p99: ........................................................................ 572.6
http.responses: ............................................................... 3
vusers.completed: ............................................................. 1
vusers.failed: ................................................................ 0
vusers.session_length:
  min: ........................................................................ 1859.4
  max: ........................................................................ 1859.4
  mean: ....................................................................... 1859.4
  median: ..................................................................... 1863.5
  p95: ........................................................................ 1863.5
  p99: ........................................................................ 1863.5


All VUs finished. Total time: 2 seconds
```

Figure 2: Response Time for Users Deletion

3.3 Shorts and Blobs Operations Test

This test evaluated the performance of uploading and downloading blobs using the persistent volume configured in the Kubernetes deployment. During the test, blobs were uploaded to the server with tokens extracted during the session, and subsequent downloads were made using session cookies. Specific blobs were deleted after these operations to validate all functionalities.



```
http.codes.200: ............................................................... 7
http.codes.204: ............................................................... 4
http.downloaded_bytes: ........................................................ 236480
http.request_rate: ............................................................ 3/sec
http.requests: ................................................................ 11
http.response_time:
  min: ........................................................................ 60
  max: ........................................................................ 840
  mean: ....................................................................... 320.1
  median: ..................................................................... 262.5
  p95: ........................................................................ 528.6
  p99: ........................................................................ 528.6
http.response_time.2xx:
  min: ........................................................................ 60
  max: ........................................................................ 840
  mean: ....................................................................... 320.1
  median: ..................................................................... 262.5
  p95: ........................................................................ 528.6
  p99: ........................................................................ 528.6
http.responses: ............................................................... 11
vusers.completed: ............................................................. 1
vusers.created: ............................................................... 1
vusers.created_by_name.CreateShorts: .......................................... 1
vusers.failed: ................................................................ 0
vusers.session_length:
  min: ........................................................................ 4296.1
  max: ........................................................................ 4296.1
  mean: ....................................................................... 4296.1
  median: ..................................................................... 4316.6
  p95: ........................................................................ 4316.6
  p99: ........................................................................ 4316.6


All VUs finished. Total time: 5 seconds
```

Figure 3: Response Time for Shorts Creation, Blob Media Upload and Download

4 Conclusion

This project adapted the Tukano application to take advantage of Azure IaaS facilities, demonstrating the use of Docker and Kubernetes. Kubernetes-based alternatives provide more control over resource allocation, scaling, and infrastructure configuration, allowing the application to better meet specific performance and reliability requirements. Performance testing validated the deployment, demonstrating the system's ability to handle concurrent operations with efficient results.