ECE750-T37: Engineering Self-Adaptive Software Systems
Assignment #1

Due Date: Friday, October 6, 2023 @11pm

Overview

The following document outlines the tasks for Assignment #1 (A1). This assignment is the first of three assignments based on the **M**onitor **A**nalyze **P**lan **E**xecute-**K**nowledge (MAPE-K) adaptation loop. As monitoring is the first component of the adaptation loop, it will also be the focus of this assignment. First, you must set up the necessary software modules (both locally and in the cloud), deploy a Java-based benchmark application, and then monitor application metrics. The metrics to be collected include various Java runtime performance indicators including memory footprint, response time, and CPU usage in addition to metrics from JMX Monitoring.

Open-Source Technologies

Before working on A1, you should obtain some basic knowledge regarding the list of following concepts:

- High-level Kubernetes – Concepts
- Git – Basic Usage
- Docker – Concepts and Basic Usage
- Apache Maven – Basic Usage

An important aspect of this assignment involves the utilization of various open-source runtime technologies. Table 1 provides an overview of the roles and some relevant documentation regarding the proposed technologies to be used in this course.

Technologies	Description	Relevant Documentation
OpenShift	Provides the encompassing environment for hosting the deployed application	OpenShift CLI OpenShift Local Guide
Eclipse OpenJ9	Provides Java execution environment to run the deployed application	OpenJ9 Docs
OpenLiberty	Provides server runtime for the deployed applications	OpenLiberty Guides
IBM Cloud Monitoring with Sysdig	Provides method to collect metrics from the deployed application	Sysdig for OpenShift
Apache JMeter	Provide method to simulate workload for deployed application	JMeter Docs

Table 1. Overview of Open-Source Technologies

The above technologies will be relevant for certain components in A1.

The Workflow

The workflow of this assignment is classified in three sections:

- *Environment Setup:* To start, various software tools and frameworks need to be installed and set up.
- *Deployment:* The application to be deployed is a fictitious flight booking service named Acme Air which is implemented with Java-based microservices. The high-level architecture of Acme Air is shown in Figure 1.
- *Monitoring:* The metrics should be pulled under different workload conditions (*e.g.*, high/low microservice workload). A small set of potential metrics (organized by type) to be monitored is listed in Table 2.

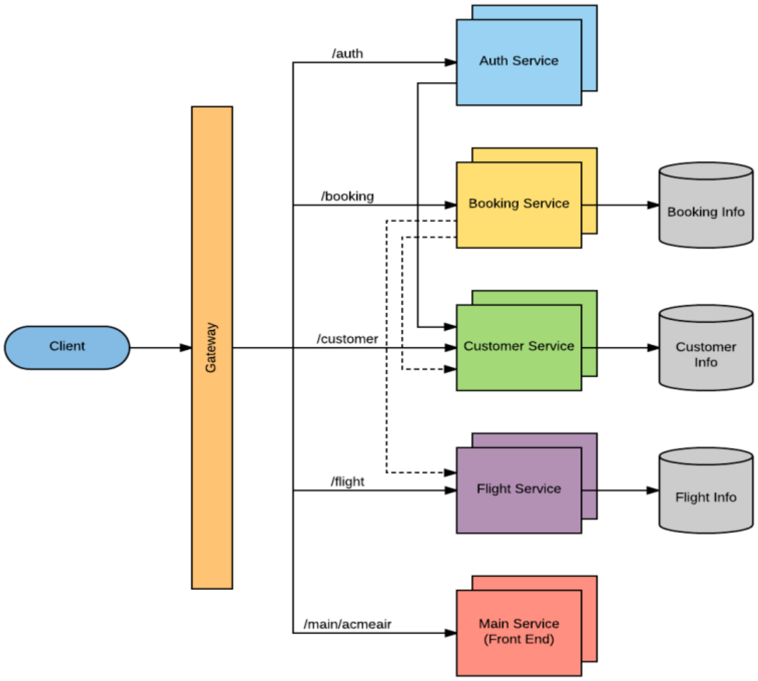

Figure 1. Conceptual Architecture of Acme Air

Deliverables

As the primary deliverable for A1, *a driver program* should be implemented to periodically pull metrics from Sysdig. The metrics should be pulled under different workload and included as *dataset files* in the final deliverable. In addition, you are asked to provide a typed *report* (maximum 2 pages) summarizing your driver program implementation. Finally, as a part of the report, *some graphs* should be plotted for two metrics from each of the metric types in Table 2, and *an explanation of the trend* observed in each of the plots should also be provided.

Metric type	**Examples (More on Sysdig)**
Java Runtime	● Jvm.class.loaded ● Jvm.class.unloaded ● Jvm.gc.global.count ● Jvm.gc.global.time ● Jvm.heap.used ● Jvm.nonHeap.used ● Jvm.thread.count
Application Specific	● Minimum response times ● Maximum response times ● Transactions per second ● Transactions over time
System Resources	● CPU % ● Memory Usage %

Table 2. Potential Metrics

Note: You should create a compressed file containing the above three deliverables. The name of this file should be **G*n*-A1.gzip** where ***n*** is the number assigned to your group. Upload your file on a designated Dropbox for A1 on LEARN.