

EEX5362 Performance Modelling

Mini Project

Deliverable 01

Name : P.D.K. Sulakshana

Student ID : S22010213

Registration No : 322513797

Auto Scaling Learning Management System (LMS) on Cloud

The system that I selected is an auto scaling LMS web service on a cloud platform. This type of system is used by universities and educational institutions. Students use it to login, access course content, and submit quizzes.

Many students use it at the same time, like during peak hours when quizzes start. The system has a load balancer that sends requests to backend virtual machines (VMs). Each VM handles requests but can get busy with CPU, memory, and network use.

An auto scaler watches metrics like CPU or queue length. It adds VMs when load is high or removes them when load drops. But adding a VM takes time, called provisioning delay. This delay can make response slow during sudden peaks. Each VM costs money based on run time.

If scaling is poor, requests take longer to answer, queues grow, or cost goes up. Good scaling keeps response fast, handles more requests, and saves money.

Dataset:

The dataset contains over 3,000 records that simulate real-world usage of an LMS like the one used by OUSL. Each record shows what happened in one minute of system operation. The dataset includes the following information:

- Timestamp: the exact minute the data was recorded
- Requests in Last Minute: how many student requests arrived in that minute
- Active VMs: how many VMs were running at that time
- Avg CPU %: average CPU usage across all active VMs
- Avg Queue Length: average number of requests waiting to be processed
- VM Provisioning Delay (seconds): how long it took to add a new VM
- Notes: short description of what happened.

It shows normal load, peaks, scaling events. From this, we see how arrival rate changes, VM count shifts, and delays affect performance.

Performance Objectives

The main goal of this study is to understand how well the auto scaling LMS performs under different loads and scaling settings. By focusing on key performance metrics, we can find ways to make the system faster, more reliable, and cost-effective.

- Reduce average response time

Students should get quick replies when they log in or submit work. Long delays hurt their experience, especially during peak times.

- Increase system throughput

The system should handle as many requests per second as possible without slowing down or failing.

- Use resources efficiently

Balance CPU, memory, network across VMs. Avoid overload on one VM.

- Reduce delays from scaling

When a surge happens, the system must scale quickly. If new VMs take too long to start, response times spike. We want to measure and reduce this effect.