Cloudflash

Dynamic IaaS Resource Allocation System

Arjun Christopher S, 2201112007

Gowtham M, 2 2201112017

Raja Hariharan K, 2201112033

Syed Khizr Tahseen, 2201112047

# Overview

CloudFlash is a dynamic, real-time Infrastructure-as-a-Service (IaaS) simulation system built using Python (Flask), HTML/CSS/JavaScript, and WebSockets. It enables automated cloud resource management through virtual machine (VM) scaling, SLA-prioritized job scheduling, and interactive monitoring via a professional-grade web UI.

# Key Features

* **Dynamic Cloudlet Submission:** Cloudlets (user jobs) can be submitted with resource demands, SLA priority, and deadlines.
* **Auto Scaling:** Automatically creates or removes VMs based on average system utilization and idle time thresholds.
* **SLA-Aware Scheduling:** Cloudlets are prioritized and escalated based on their SLA level and proximity to deadline.
* **Live Monitoring Dashboard:** Interactive dashboard with real-time updates using Flask-SocketIO.
* **Resource Presets:** Quick setup for Small/Medium/Large VMs and Light/Moderate/Heavy Cloudlets.
* **Countdown Timers:** Visual countdown to each cloudlet’s deadline.

# System Components

## Backend (Flask)

* app.py: Defines all API endpoints for VM and Cloudlet operations, renders the dashboard, and emits real-time updates.
* core.py: Contains the ResourceManager, VM, and Cloudlet classes. Manages allocation, scaling, deadlines, and metrics.

## Frontend

* index.html: Main dashboard with interactive forms and resource charts.
* dashboard.js: Handles user input, charts (Chart.js), real-time updates (Socket.IO), logs, and animations.
* style.css: Custom UI styling following a light modern aesthetic.

## Real-Time Metrics

* Charts for CPU, RAM, Storage, Bandwidth, GPU utilization.
* SLA Compliance pie chart.
* Cloudlet status doughnut chart.
* Auto-scaling status and average utilization bar.

# Auto-Scaling Logic

* **Scale Up**:
  + Triggered when average resource utilization exceeds 80%.
  + Creates a new Medium VM and attempts to allocate pending Cloudlets.
* **Scale Down**:
  + Triggered when average utilization falls below 30%.
  + Removes idle VMs not used in the past 5 minutes.
* **Cooldown Period**:
  + Enforced 60-second delay between scaling actions to avoid flapping.

# Cloudlet Scheduling Logic

* Prioritized by:
  + SLA (High = 3, Medium = 2, Low = 1).
  + Creation time (earlier submitted = higher priority).
* Escalation:
  + If time left < 10s before deadline, SLA is automatically boosted to High.

# Presets

* **VM Presets**:
  + Small: 2 CPU, 4GB RAM, 40GB Storage
  + Medium: 4 CPU, 8GB RAM, 100GB Storage
  + Large: 8 CPU, 16GB RAM, 200GB Storage
* **Cloudlet Presets**:
  + Light: 1 CPU, 2GB RAM, 5GB Storage
  + Moderate: 2 CPU, 4GB RAM, 10GB Storage
  + Heavy: 4 CPU, 8GB RAM, 20GB Storage + GPU

# Technologies Used

* **Backend:** Python Flask, Flask-SocketIO
* **Frontend:** HTML5, CSS3, JavaScript (ES6), Chart.js, Socket.IO
* **3D UI (optional):** Zdog (icon animation), Three.js (3D objects)

# Deployment & Hosting

* Recommended: Railway, Render, Heroku, or PythonAnywhere
* Requirements: Python 3.8+, Flask, Flask-SocketIO

# Future Enhancements

* Add user authentication for multi-tenant simulation.
* Integrate Redis or Celery for distributed workload.
* Add workload analytics and historical visualizations.
* GPU-aware VM selection algorithm.
* Dark mode toggle for dashboard.

# Conclusion

CloudFlash delivers a highly interactive and dynamic platform for simulating IaaS-based resource provisioning and scheduling. With a focus on SLA-aware prioritization, automated VM scaling, and modern UI/UX features, the system is ideal for demonstrating cloud infrastructure behavior under varying workloads. Its modular design and extensibility make it suitable for academic learning, research, and prototype development in cloud computing.