Load Testing Report

Overview

This document outlines the load testing procedures and results for a script that logs into the Odoo application, prints user data, adds employees to the employee portal, and logs attendance for the employees. The testing was conducted using Locust to simulate user load on the system. Three main test cases were executed to evaluate the system's performance under varying loads and to identify and address gateway timeouts (502 and 504 errors).

Test Environment

- **Application**: Odoo
- **Testing Tool**: Locust
- **Infrastructure**:
    - **ECS Tasks**: Initially 1 task, auto-scaled up to 6 tasks (each with 1 CPU and 2 GB RAM)
    - **RDS**: PostgreSQL Aurora, r5.large instance
- **Duration**: Total testing time exceeded 3 hours across all test cases
- **RDS CPU Utilization**: Peaked at 93%

Test Cases

Test Case 1: Gradual Ramp-Up to 1000 Users

Objective: To evaluate system performance with a gradual increase in user load from 100 to 1000 users.

- **Initial Users**: 100 users
- **Ramp-Up Rate**: 1 user per second
- **Maximum Users**: 1000 users
- **Duration**: Over 2 hours
- **Total Requests Sent**: Over 30,000 requests

Results:

- **Initial ECS Tasks**: 1 task
- **Auto-Scaling Trigger**: ECS tasks increased due to CPU usage reaching 90% while kickstart.
- **Maximum ECS Tasks**: 6 tasks

- RDS CPU utilization reached a peak of 93%.
- Gateway timeouts (502 and 504 errors) occurred initially due to the application receiving more requests than it could handle.
- The system handled the gradual increase in load, but high CPU utilization indicated significant strain on the database.

Test Case 2: Immediate Ramp-Up to 1000 Users

Objective: To evaluate system performance with an immediate increase in user load to 1000 users.

- **Initial Users**: 1000 users
- **Ramp-Up Rate**: 1 user per second
- **Duration**: 30+ minutes
- **Total Requests Sent**: Over 22,000 requests

Results:

- **Initial ECS Tasks**: Maintained from the previous test (6 tasks).
- RDS CPU utilization remained high, peaking at 93%.
- Gateway timeouts (502 and 504 errors) were observed due to the high initial load.
- The system managed the immediate load increase, but the sustained high CPU utilization suggested potential bottlenecks at the database level.

Test Case 3: Controlled Ramp-Up with Revised Auto-Scaling Policy

Objective: To evaluate system performance with a controlled ramp-up of user load and an updated auto-scaling policy.

- **Initial Users**: 100 users
- **Ramp-Up Rate**: 1 user per second
- **Maximum Users**: 1000 users
- **Duration**: Continued testing up to 700+ users
- **Auto-Scaling Policy**: Scale out when CPU reaches 50% (previously 80%)

Results:

- **Initial ECS Tasks**: 1 task
- **Auto-Scaling Trigger**: ECS tasks increased as CPU usage reached 50%.
- **Maximum ECS Tasks**: 6 tasks (service quota limitation)
- Initial gateway timeouts (502 and 504 errors) occurred due to the spike in requests before auto-scaling kicked in.
- Once auto-scaling was active, timeouts were reduced.
- As testing continued up to 700+ users, ECS attempted to provision more tasks, but was limited to 6 due to service quota limitations, causing occasional timeouts.

Observations

- **ECS Auto-Scaling**: The updated auto-scaling policy (triggering at 50% CPU) helped reduce initial gateway timeouts, but service quota limitations prevented further scaling beyond 6 tasks.
- **Database Performance**: The PostgreSQL Aurora r5.large instance reached 93% CPU utilization, indicating that the database is a critical point of strain. Scaling the RDS instance might be necessary to improve performance.
- **Request Handling**: The system successfully handled a large number of requests in all scenarios, but performance degraded under high loads due to both ECS task limitations and database strain.

Recommendations

1. **Database Optimization**:
    - Explore scaling options for the Aurora instance (e.g., larger instance type, read replicas).
2. **Application Scaling**:
    - Evaluate ECS task definitions to ensure optimal resource allocation.
    - Request an increase in the service quota for ECS tasks to allow more scaling flexibility.

By addressing these recommendations, the system can be better prepared to handle higher loads and ensure stable performance for end-users.