CS455: Introduction to Software Engineering

Report, Homework-4: Performance Testing and Reliability Enhancements

Team Information
This assignment is presented by:

1. Aditi Khandelia, Roll No. 220061
2. Kushagra Srivastava, Roll No. 220573

Codebase
The codebase for the application can be found at : `https://github.com/CS455-Assignment-1/StackNServe`

Assignment Information

1. Part 1: Performance Testing
   (a) Client side
      i. Performance tests written for `Leaderboard.razor`, `Home.razor` and `Player_Guide.razor` using Selenium. The tests can be found in 'performance tests' folder.
      ii. The load times of all the pages were tested, with assertions present on the load time.
      iii. Additional tests include the following : Checking of load time when Player Name entered in the home page, checking of load time of select buttons, checking of redirection time from player guide to the main player arena etc.
      iv. The optimisations done include pre-fetching the order lists and prices and storing them, and pre-fetch of ingredient descriptions as soon as game loads.

   (b) Server side
      i. Performance tests are written in `locust` to test the `Leaderboard` and the `User Requests` in the server.
      ii. Tests are executed for 1000 concurrent users, for 5 minutes for each test file.
      iii. The html report generated can be viewed for further statistics such as median, mean, min and max times.

   (c) Nightly Build
      i. The Nightly Pipeline runs the server tests for 1000 (a larger number logs out of render, where the servers are hosted) concurrent users every night, at 00:00 ITC, using a cron job. It also runs tests for the client.
      ii. The tests results are packed into a downloadable zip that can be used to analysis.
      iii. The test can also be triggered manually using the workflow button.

2. Part 2: Reliability and Failover Mechanisms
   (a) Load Balancing
      i. `load_balancer.py` created, which is now the main hosted server receiving the requests from client side.
      ii. Two new server instances created, which now host `server.py` on `https://stacknservesecondserver.onrender.com` and `https://stacknservethirdserver.onrender.com`.
      iii. `load_balancer.py` redirects requests to the two servers in a round robin fashion.

   (b) Automatic Failover
      i. A new instance of the server was hosted on `https://stacknservebackupserver.onrender.com`, which serves as the backup-server.
      ii. A new endpoint `health` was created in the servers, which responds with a the message "Healthy", along with the response code `200` when the server is active and healthy.
      iii. Based on the three servers, a list of active servers are created, and then requests are redirected to these servers.
      iv. In case of both the primary servers failing, all of the requests are passed through the backup server.

Performance Test Reports

These reports can be downloaded from the workflow history provided by GitHub Actions. The performance tests without and with the load balancer are uploaded in the Reports folder.