Experimentation System

Gili Chiko, Sean Naor, Gilad Chen, Shahar Morad

Tests Document

Functional Requirements:

we preformed a large amount of unit and acceptance testing on our logic & data server in order to get full coverage and making sure that each requirement is handled sufficiently.
We will describe our tests according to this table:

requirement	description	layer	test name	test directory/class	pass/fail

Requirement - the id of the requirement that the function is related to.
Description – a short explanation about the function.
Layer – the layer of the function (Service, Business, Entities).
Test name.
Directory/Class – a more specific info about the environment of the function.
Pass/Fail- simply to see if the test passes of fails.

Our unit tests on the entire code reached at least 90% for each class as you can see:

Coverage Breakdown

Package	Class, %	Method, %	Line, %
com.example.demo.BusinessLayer.Entities	100% (8/ 8)	92.6% (87/ 94)	92.5% (222/ 240)
com.example.demo.BusinessLayer.Entities.GradingTask	100% (5/ 5)	88.6% (39/ 44)	90.5% (114/ 126)
com.example.demo.BusinessLayer.Entities.Results	100% (9/ 9)	89.1% (49/ 55)	95.1% (117/ 123)
com.example.demo.BusinessLayer.Entities.Stages	100% (10/ 10)	91% (81/ 89)	91.5% (226/ 247)

our acceptance tests were divided into 2 parts:

First processing the request and the data from the user in the Service Layer:

Here we reached 100% of coverage:

Coverage Breakdown

Package	Class, %	Method, %	Line, %
com.example.demo.ServiceLayer	100% (3/ 3)	100% (38/ 38)	100% (211/ 211)

Secondly, going throw the request, involving the relevant data from the data entities and making the logical examination to check if and how the request should be executed, this was all preformed in the business Layer:

Here we reached 99.5% of coverage:

Coverage Breakdown

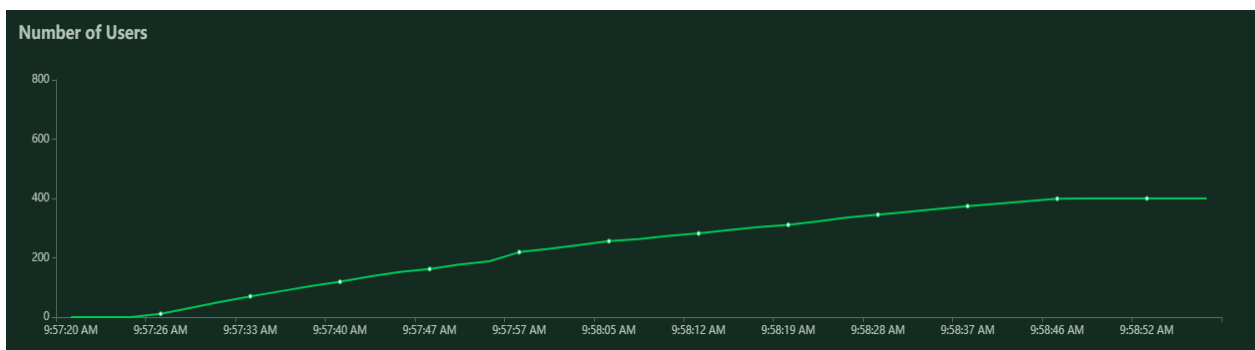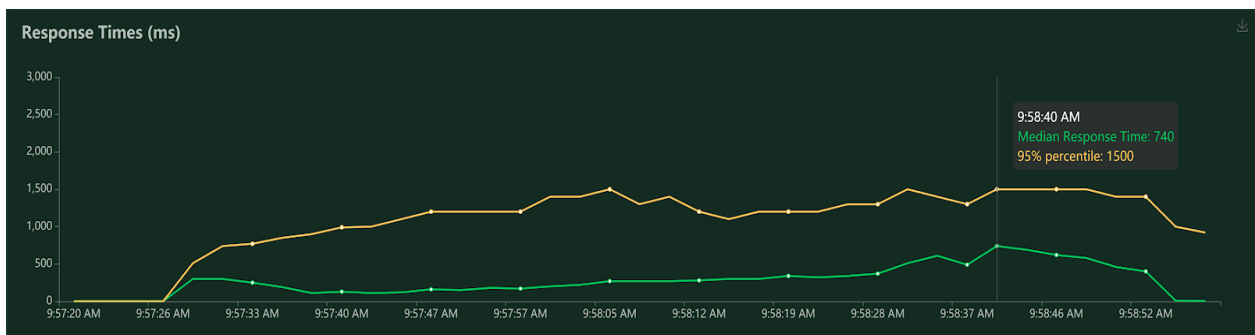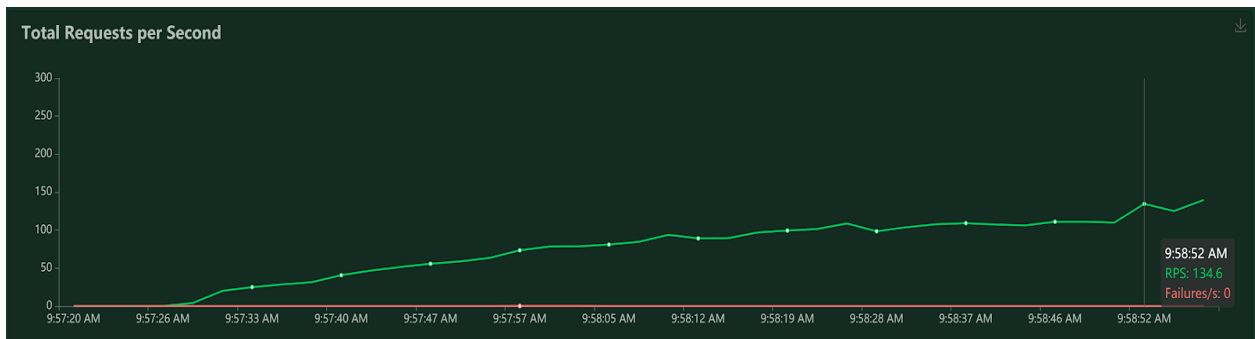| Package	Class, %	Method, %	Line, %
com.example.demo.BusinessLayer	100% (3/ 3)	100% (44/ 44)	99.5% (218/ 219)

we added a list of all the unit tests and the acceptance tests at the end of the document.

User Interface:

We did not perform any tests regarding the user interface.

Load Testing:

We didn't had the time to do load testing on the entire system but we took the action that has will probably occur most of the time which is when an experimentee submits a stage. We used locust to perform the load testing, we generated 400 users that logged in to start participation in an experiment and made them submit stages concurrently, we can see the results below:

Total Requests per Second

9:58:52 AM
RPS: 134.6
Failures/s: 0

Response Times (ms)

9:58:40 AM
Median Response Time: 740
95% percentile: 1500

Number of Users

The following link is to list is a list of all the tests we performed on the data & logic server:

https://docs.google.com/spreadsheets/d/13UIYKW180tCECaJCfu6uzrOhemFu_M3tzFIq8zaAFzg/edit?usp=sharing