Starting documentation

EPAM Feedback Portal v. 2019Q4

Performance Testing Strategy

1. **Introduction**

The purpose of this document is to specifically describe how the performance requirements for Feedback Portal web application will be tested and verified on test environment (System Under Test, SUT). This document describes the strategy of performance testing for the Feedback Portal. It consists of outlines for the following items:

* Items to be / not to be tested;
* Approach;
* NFRs;
* Environmental needs;
* Entry and Exit criteria;
* Suspension and Resumption Criteria;
* Responsibilities;
* Schedule (test milestones and item transmittal events);
* Risks.

1. **Items to be tested**

Mainly, Feedback Portal application will be tested via UI, as server side and as client side checking such main features:

|  |  |
| --- | --- |
| # | **As employee** |
|  | Give feedback (to colleague/ to himself, full / piece); |
|  | Request feedback (from colleague / customer); |
|  | View feedback (employees/ for employee); |

|  |  |
| --- | --- |
| # | **As manager** |
|  | View feedback for team member; |
|  | Approve / reject feedback request for customer; |
|  | Rate feedback for team member |
|  | Send summary review on team member |

1. **Items not to be tested**
   * + - Other EPAM internal systems;
       - Integration of application with other EPAM internal systems;
       - “Help” functionality;
       - “Ask” functionality.
2. **Approach**
   1. **Test types assumed for conducting**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Test type** |  | **Period** | **Notes** |
|  | Smoke test | Mandatory | Regular |  |
|  | Capacity | Mandatory | After significant changes |  |
|  | Load test | Mandatory | Regular |  |
|  | Stress | Mandatory | Rare |  |
|  | Volume | Mandatory | Rare |  |
|  | Durable | Mandatory | Rare |  |
|  | Scalability | Optional | Once+ |  |

* + 1. **Smoke testing**

Should be performed every time when functionality of the application and the script need to be checked. Also, if needed can be used as warming up test before main testing step.

* + 1. **Capacity testing**

Should be performed to find the number of virtual users which the application support in stable state. The test can be performed as one of first main tests, and should be performed after significant changes in the application or its configuration.

* + 1. **Load test**

Load test is a kind of the most regular tests to check benchmark of the application and its components. Usually, is running after finding capacity.

* + 1. **Stress testing**

Stress testing supposed to run occasionally to check application’s stability under high load. As from experience high load is not already expected to occur, because the principle of using this application was changed, perform stress test is mandatory, but not so often as load.

* + 1. **Scalability testing**

Can be performed once or more in order to get multiplier(s) for different number of front-end/other servers.

* + 1. **Volume testing**

Is to run with small/planned/huge amount of data with regular load to get indicators on application’s responsiveness/metrics change. Should be performed each time as well as stress test.

* + 1. **Durable (Stability) testing**

Supposed long time running the test with the load lower than average. Should be performed occasionally after significant code changes or by special request to make sure the application’s responsiveness and key performance indicators do not change significantly after long time running, and to check on memory leak as well.

1. **Non-Functional requirements**

|  |  |
| --- | --- |
| Users metrics | |
| Total Users | 40614 |
| Concurrent Users | 2000 users / day, 500 / peak hour (2 PM CET) |
| Concurrent Employees | 65 % |
| Concurrent Managers | 35% |

|  |  |  |  |
| --- | --- | --- | --- |
| **Action / Response time** | **Average** | **90th percentile** | **99th percentile** |
| “Open Log in page” request | 1500 ms | 1800 ms | 2000 ms |
| “Log in” request | 2300 ms | 2600 ms | 3000 ms |
| “Open Request feedback page” request | 2000 ms | 2200 ms | 2700 ms |
| “Open Give feedback page” request | 2000 ms | 2200 ms | 2500 ms |
| “Open Give feedback for person page” request | 800 ms | 1000 ms | 1100 ms |
| “Send feedback” request | 1000 ms | 1200 ms | 1500 ms |
| “Request feedback” request | 600 ms | 900 ms | 1100 ms |
| “Open Previously provided feedback” request | 1100 ms | 1300 ms | 1500 ms |
| “Save as draft” request | 1000 ms | 1200 ms | 1500 ms |
| “Open My team page” request | 2100 ms | 2200 ms | 2600 ms |
| “Send approve status on feedback to customer” request | 600 ms | 900 ms | 1100 ms |
| “Send reject status on feedback to customer” request | 600 ms | 900 ms | 1100 ms |
| “Open Rate for left feedback page” request | 900 ms | 1100 ms | 1300 ms |
| “Rate left feedback” request | 700 ms | 900 ms | 1200 ms |
| “Open Leave summary review page” request | 800 ms | 1000 ms | 1100 ms |
| “Send Summary review” request | 1100 ms | 1300 ms | 1600 ms |
| “Log out” request | 2300 ms | 2600 ms | 3000 ms |

1. **Environmental needs**
   1. **Test environment(s)**

For performance testing a dedicated environment is needed. The configurations of the servers should be as much as close to production’s ones.

Expected numbers of the servers:

* Web servers - 4
* Application servers - 2
* Database servers - 2

To perform partial and full end-to-end performance testing it’s recommended to have a separated database and other services, i.e. they should not be located on the same server where related services are being tested by another team.

* 1. **Testing tools**

|  |  |  |  |
| --- | --- | --- | --- |
| **Module** | **Software** | **Supported OS** | **Description** |
| [Continuous integration](https://en.wikipedia.org/wiki/Continuous_integration) software | Jenkins | Linux/ Windows | Module that starts and control execution of your performance tests. |
| Monitoring agent | Telegraf | Linux/ Windows | Module that gets all performance and health metrics from host where it installed (including Load Generator) to get more accurate picture of your performance tests impact |
| Load Generator | JMeter | Linux/ Windows | Module that generates load on your application, currently supported only JMeter 5.2.1 |
| Bug tracking application | JIRA | Linux/ Windows | Tool for tracking all defects and performance issues found during performing tests. |
| Documentation storage | Confluence | Linux/ Windows | Application needed for saving all documentation in one place as these test strategy and test plan, as reports created after each test run. |
| Profiling | JProfiler | Linux/ Windows | Tool that helps to resolve performance bottlenecks, pin down memory leaks and understand threading issues. |

1. **Test data**

To have whole cycle of performance testing test data for Feedback Portal should be:

1. Reusable.
2. Generated in necessary amount for different stage of testing at any time:
   1. Users with different roles(employees, managers);
   2. Ready feedbacks;
   3. Other.
3. Cleanable.
4. **Performance Entry, Exit, Suspension and Resumption Criteria**

**8.1 Entry Criteria**

* Test plan is completed and approved.
* Test environment is updated with needed system configurations.
* Test data is completed and present on the performance testing environment in sufficient time to allow test scripts to be completed.
* Test scripts are created.
* All assigned resources are available to monitor the test.
  1. **Exit Criteria**
* All test scripts completed successfully
* No critical problems encountered
* All non-critical problems are logged
* All test logs are captured
* Report based on testing is prepared and accepted.
  1. **Suspension Criteria**
* Not all test scripts completed
* Critical problems are encountered and logged
* Software/Hardware errors prevented the completion of the test
* Test data issues (no test user or wrong credentials, incomplete data)
* Significant changes in workflow of functionality of the application which require updates in the test plan or scripts/scenarios.
  1. **Resumption criteria**

Resumption will only occur when the problem(s) that caused the suspension have been resolved.

1. **Responsibilities**

Person 1 – Performance Engineer from team working on Server-side performance and integration.

Person 2 – Automation Tester from team working on Client-side performance.

1. **Schedule (test milestones and item transmittal events)**

|  |  |  |  |
| --- | --- | --- | --- |
| **#** | **Stage** | **Period** | **Notes** |
| 1. | Test strategy design |  |  |
| 2. | Test plan creation |  |  |
| 3. | Test cases(scenarios) design |  |  |
| 4. | Setting up test environment |  |  |
| 5. | Test data preparation |  |  |
| 6. | Scripts design |  |  |
| 7. | Automation test running from CI/CD |  |  |
| 8. | Basic tests round: Smoke, Capacity |  |  |
| 9. | Test results analysis |  |  |
| 10. | Test reporting |  |  |
| 11. | Basic tests round (with plugged-in services): Smoke, Capacity |  |  |
| 12. | Test results analysis |  |  |
| 13. | Test reporting |  |  |
| 14. | NFR update, load level definition |  |  |
| 15. | Updating scripts/scenarios |  |  |
| 16. | Starting end-to-end regular load test running |  |  |
| 17. | Running special tests (ex. Scalability) | By request |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. **Risks**

* Performance testing results can be essentially different even in case of minor difference in think times, arrival rate and test duration.
* During the execution of the tests, some major performance or functional problems due to changes in Feedback Portal configurations may occur and in that case it may be necessary to repeat the load test from the beginning.
* Performance testing tool is not capable of identically reproducing real life scenarios - so results could only be trusted as having limited reliability level.
* Network/systems latency issues.
* Environment’s unavailability.
* Some applications may be running at the same time as script, so they may produce extra load.
* Unexpected shutdown of the system.

Performance Test Plan

1. **Introduction**

The purpose of this document is to specify performance requirements and conditions for EPAM Feedback Portal application. The document will outline test objectives, test cases, test data, user roles and testing tasks. This document has clearly identified what the test deliverables will be, and what is deemed in and out of scope.

1. **Test description**
   1. **Test objectives**
      1. Check test scripts workability
      2. Check application’s availability on selected environment
      3. Define capacity in term of number of concurrent virtual users
      4. Perform basic test rounds for different kinds of tests
      5. Define regular load
      6. Perform end-to-end regular load tests
   2. **Items to be tested**

|  |  |  |
| --- | --- | --- |
| **Item** | **Action** | **Notes** |
| Open Log in page | Page load request | - |
| Log in | Send request with data on authorization server | From Log in page |
| Open Request feedback page | Page load request | From Log in page |
| Open Give feedback page | Page load request | From Request feedback page |
| Open Give feedback for person page | Page load request | From Give feedback page |
| Send feedback | Send request with data on server | From Give feedback for person page |
| Request feedback | Send request with mail on server | From Request feedback page |
| Open Previously provided feedback | Page load request | From Give feedback page and Request feedback page (if feedback shared) |
| Save as draft | Send request with data on server | From Give feedback for person page |
| Open My team page | Page load request | From Log in page |
| Send approve status on feedback request to customer | Send TRUE value in request on server | From My team page |
| Send reject status on feedback request to customer | Send FALSE value in request on server | From My team page |
| Open Employees feedbacks page | Page load request | From My team page |
| Send Rate for left feedback | Send request with data on server | From Employees feedbacks page |
| Open Leave summary review page | Page load request | From Rate for left feedback |
| Send Summary review | Send request with data on server | Leave summary review page |
| Log out | Send request with value on server | From any page |

* 1. **Items not to be tested**
     1. Other EPAM internal systems;
     2. Integration of application with other EPAM internal systems;
     3. “Help” functionality;
     4. “Ask” functionality;
     5. Features which are not implemented by the start of the performance testing.
  2. **Test data**

Test data should be generated by special tools outside of application and should include integrated data:

1. Set of users:

* Employees;
* Managers.

1. Feedbacks:

* Received feedbacks(employee);
* Sent feedbacks(employee);
* Received feedbacks(manager);

Test data should be submitted to performance testing group for running load tests.

* 1. **Test users roles**
     + - Employee;
       - Manager.

1. **Non-Functional requirements**

NFR are provided in EPAM Feedback Portal strategy document.

1. **Suspension criteria and resumption requirements**

Suspension and resumption criterias described in EPAM Feedback Portal strategy document.

1. **Test deliverables**

Main expected test deliverables are:

1. Test plan.
2. Test strategy.
3. Basic test scripts and scenarios.
4. Baseline of main metrics: response times for the pages, system resources consuming etc.
5. Analysis on gathered KPI`s.
6. **Testing tasks**

6.1. Basic scripting

6.2. Basic scenarios creation

6.3. Setting up load generation tools

6.4. Setting up monitoring and collecting tools

6.5. Deployment stable version to environment

6.6. Smoke testing

6.7. Capacity testing

6.8. Running set of performance testing:

6.8.1. Regular load test (after definition of capacity)

6.8.2. Volume

6.8.3. Stress

6.8.4. Durable (stability)

6.8.5. Scalability

1. **Test environment**

Test environment described in EPAM Feedback Portal strategy document.

1. **Risks**

Main risks are described in EPAM Feedback Portal strategy document.

Test cases

As all main actions performed by users are described in Test Plan in section “Items to be tested” was decided to create test scenarios on which test scripts will be based and created in Jmeter tool. Overall, there will be 2 main scenarios: employee scenario and manager scenario which are showed below.

**Employee scenario**

|  |  |  |  |
| --- | --- | --- | --- |
| **Scenario** | **Test Script Steps** | | |
| Employee scenario | Open Feedback portal (Log in) start page | | |
| Login | | |
| Open Request feedback page | | |
| X10 | | Request feedback |
| If possible | | Open Previously provided feedback for employee |
| Open Give feedback page | | |
| X3 | Random or first | Open Give feedback for person page |
| Random | Save as Draft |
| Send feedback |
| Random | | Open Previously provided feedback by employee |
| Logout | | |

**Manager scenario**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Scenario** | **Test Script Steps** | | | |
| Manager scenario | Open Feedback portal (Log in) start page | | | |
| Login | | | |
| Open My team page | | | |
| For all possible feedback requests | Random | Send reject status on feedback request to customer | |
| Send approve status on feedback request to customer | |
| X3 | Open Employees feedbacks page | | |
| For all possible feedbacks | | Send Rate for left feedback |
| Open Leave summary review page | | |
| Send Summary review | | |
| Logout | | | |