The BLOG

Test plan

1. Introduction
   1. Purpose of this Test plan

The purpose of this document is to specify performance requirements and conditions for TheBlog website. The document will outline the Soak Test scenarios, test cases, parameters and data used in evaluating the capacity of the included features.

* 1. Test plan identification

|  |  |
| --- | --- |
| Application Name | TheBlog |
| Test Cases | Practice: Task 1 running from Performance Testing Mentoring Program 2022 Q1Q3 |
| Practice: Task 2 running from PTMP 2022 |
| Practice: Task 3 running from PTMP 2022 |
| Practice: Task 4 running from PTMP 2022 |
| Practice: Task 5 running from PTMP 2022 |
| Practice: Task 6 running from PTMP 2022 |
| Practice: Task 7 running from PTMP 2022 |
| Practice: Task 8 running from PTMP 2022 |
| Practice: Task 9 running from PTMP 2022 |
| Practice: Task 10 running from PTMP 2022 |
| Practice: Task 11 running from PTMP 2022 |

1. Test description
   1. Pre-requisites
      1. Execution of Performance testing follow after development stories mark as "Done" to avoid functionality bugs
      2. The performance testing process uses the Agile principles as the development process
      3. Performance testing is a continuous process that integrated into Cl tools.
      4. Check-in's TheBlog is done regularly (scheduled on the CI)
      5. The final code should be promoted to the Production environment after performance e2e testing is completed and results meet expectations.
      6. Performance testing jobs on the CI should follow the performance testing purpose and type

In an attempt to detect issues as early as possible Application performance testing will start as early as possible with basic performance measurements being taken as part of the dev process and final end-2-end performance testing being performed after all development for the increment has been completed but before implementation.

Early stage tests will be incorporated into the CI process allowing basic performance tests to be executed for each build whilst the more extensive end-2-end testing will be instigated in the appropriate performance test tool on Non-functional environment

* 1. Test objectives
     1. Ensure the performance of the application according to the stated requirements
     2. Check for compliance with ESLAY, NFR, acceptance criteria
     3. Early discovery of bottlenecks related to the application system, infrastructure, network,
     4. Early detection of performance degradations associated with performance metrics
  2. Items not to be tested
     1. Functionality of the whole website
     2. Features which are not implemented by the start of the performance testing
  3. Test data

Test data must be generated before starting testing

1. Suspension criteria and resumption requirements
   1. Functional issues
      1. Test environment issues (not ready, any trouble with accessing to it etc.)
      2. Not stable version of the application
      3. Test data issues (no test user or wrong credentials, incomplete data)
      4. Significant changes in workflow of functionality of the application which require updates in the test plan or scripts/scenarios
      5. Testing tools issues (load generation, monitoring)
   2. Resumption criteria
      1. Test plan is complete/updated and approved by Company and the client.
      2. Correct version is installed in performance testing environment, i.e. the version previously functionally tested and fixed if needed
      3. Test data is complete and in the performance testing environment in sufficient time to allow test scripts to be completed.
      4. Test accounts have been created in the performance testing environment in sufficient time to allow test scripts to be completed.
      5. Test scripts complete.
      6. All assigned resources are available to monitor the test.
      7. All parameter sets used in the script are generated based on the Database values.
2. Testing tasks
   * 1. Task 1. Simple script

Description:

Create simple script to open Home page, open Contacts.

Number of users: up to 10

User role: anonymous Task 1. Simple script

Goals:

Check availability of the site from the load tool

Make sure the site is configured for multiuser mode

Setup basic metrics and make sure they can be collected

Setup ramp-up mode in the scenario

* + 1. Task 2.

Prerequisites:

Using Admin user create Editor user and assign him appropriate permissions.

Description:

Create the script and the scenario to generate test data (posts) with any text (like Loren ipsum…) of random length between 50 and 1000 characters.

The number of the posts should be parametrized and consist at least 3 values:

100, 1000, 2000

For 100 posts dates should be set from the list of particulate dates which will be used in following regular tests (at least 10 values). For other posts dates can be generated randomly within particular range.

Also, get measured time for generation 100 posts, and is possible for 1000 and 2000 posts.

Number of users: up to 2

User role: editor

Goals:

* Get data generator
* Script login procedure.
* Get experience of test data preparation
* Get estimated time needed for data generation.
  + 1. Task 3

Description:

Implement Admin user scenario (see Admin script algorithm).

* Perform Smoke testing.
* Perform load test with 2 admin users.
* Document results.

Names of created users: User\_*nn* (e.g. User\_15, User\_07)

User role: admin

Goals:

* Design admin script and scenario
* Get base line on performance of particular admin’s actions.
  + 1. Task 4

Prerequisites:

* Created 2 Editors
* Generated 100 posts

Description:

Implement Editor user scenario (see Editor script algorithm).

* Perform Smoke testing.
* Perform load test with 2 editor users.
* Document results.

User role: editor

Goals:

* Design editor script and scenario
* Get base line on performance of particular editors’ regular actions.
  + 1. Task 5

Prerequisites:

* Created 2 Editors
* Generated 100 posts

Description:

Implement Editor user scenario (see Editor script algorithm).

* Perform Smoke testing.
* Perform load test with 2 editor users.
* Document results.

User role: editor

Goals:

* Design editor script and scenario
* Get base line on performance of particular editors’ regular actions.
  + 1. Task 6

Prerequisites:

* Generated 100 posts
* Generated 1000 posts

Description:

* Implement Anonymous user scenario (see Main script, Anonymous script, Open post script algorithms) as in Task 3.
* Update the script/scenario to implement probabilities usage:

- Home Page: 15%

- Open Random Date: 10%

- Open Predefined Date: 30%

- Search by Name: 30%

- Open Large Calendar: 10%

- Open Contacts: 5%

Open Random page (yes/no): 50% / 50%

Open post (yes/no): 80% / 20%

Random or First (yes/no): 65% / 35%

Comment (yes/no): 20% / 80%

* Perform Smoke testing.
* Perform Capacity testing for the test from Task 3 (the same probabilities for all branches) with two sets of generated posts (100 and 1000).
* Perform Capacity testing for Task 6 with two sets of generated posts (100 and 1000).
* Compare results between:
  + Task 3 and Task 6: 100 posts
  + Task 3 and Task 6: 1000 posts
  + Task 6: 100 and 1000 posts
* Perform analysis of the results with summary and detailed explanation of changes in the behavior.
* Document results.

During the testing gather main business, application and system metrics and include them in the comparison.

User role: anonymous

Note: the task is to be done without third party controllers.

Goals:

* Get an experience of smooth update of the script according to new requirements and cases
* Get an experience in regression testing (like after new version uploaded)
* Get an experience in volume testing
* Improve skills in comparison and analysis
  + 1. Task 7  
        Prerequisites:

Generated 1000 posts.

Description:

Prepare combined scenario which is to include the scripts:

* Task 4 (admin)
* Task 5 (editor)
* Task 6 (anonymous)

Implement Anonymous user scenario (see Main script, Anonymous script, Open post script algorithms).

* Perform Smoke testing.
* Perform Capacity testing.
* Document results.

The number of the posts: 1000

User role: (admin, editor, anonymous)

Goals:

* Design combined script and scenario
* Get an experience of combined testing with different user roles  
  + 1. Task 8  
        Prerequisites:

Generated 1000 posts.

Results of Capacity testing from Task 7.

Description:

* Using results of the testing from Task 7 define refular load.
* Define general Key Performance Indicators (KPI), and get their basic value using the results from Task 7.
* Set the size of the memory as bigger as possible for the virtual machine.
* Scaling CPU:
  + Perform load testing for different number of CPUs: 1, 2, 3 (if possible), 4, 6 (if possible).
  + For each test gather all needed metrics.
  + Calculate KPI.
  + Calculate multiplier for scaling CPU.
* Set the number of CPU as big as possible.
* Scaling RAM:
  + Perform load testing for different size of RAM: 2Gb, 3Gb, 4Gb (if possible), 6Gb (if possible).
  + For each test gather all needed metrics.
  + Calculate KPI.
  + Calculate multiplier for scaling RAM.
* Prepare the report on scalability testing.

The number of the posts: 1000

User role: (admin, editor, anonymous)

Goals:

* Get an experience on regular load definition.
* Get an experience on KPI definition and calculation.
* Get an experience of scalability testing.
  + 1. Task 9

Prerequisites:

Defined regular load.

Description:

* Testing with different number of the text posts:
  + Perform combined testing with different number of the posts:
    - 100
    - 1000
    - 2000
    - 5000
  + Gather all needed metrics
  + Calculate KPI.
  + Compare the results
  + Define the threshold for degradation depending on the quantity of the posts
* Testing with media information
  + Prepare 1000 text posts
  + Prepare 1000 posts with a text and attached 1Mb photo
  + Run the regular load test
  + Gather all needed metrics
  + Calculate KPI
  + Compare the results between the results of the testing with 2000 pure text posts and the results of the testing with 2000 mixed posts
* Prepare complex report on volume testing

User role: (admin, editor, anonymous)

Goals:

* Get an experience of advanced volume testing.
* Get an experience on preparation the different kinds of test data.
  + 1. Task 10  
        Prerequisites:

Generated 1000 text posts.

Description:

* Define low load based on the results from Task 7.
* Perform long-time testing.
* Gather all needed metrics.
* Calculate KPI.
* Try to identify any issues based on the results.
* Prepare detailed report.

The number of the posts: 1000

User role: (admin, editor, anonymous)

Goals:

* Get an experience on durable/longevity/stability testing.
* Get an experience on load parameters definition for the long-time testing.
* Learning how to identify bottlenecks and possible issues for long-time running systems.
  + 1. Task 11  
       Description:
* Switch the data source from file system to DB.
* Generate 1000 text posts.
* Perform capacity testing.
* Perform regular load testing with the parameters based on the results of capacity testing.
* Perform regular load testing with the parameters based on the results from Task 8.
* Perform stress testing.
* Repeat volume testing with the parameters of the Task 9.
* Prepare complex report including the comparison between the results with files system data source and DB data source if appropriate results are available.

User role: (admin, editor, anonymous)

Goals:

* Get an experience of configuration testing
* Learn complex testing during migration of test data to new platform.
* Learn complex analysis.

1. Test environment

For performance testing a dedicated environment on premise/cloud/VMs and optional database. The configurations of the servers should be as much as close to production’s ones.

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 that may require code changes, creation of a new build may be discovered 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