**Report on results of performance testing: Task8 – Perform scalability testing**

**Purpose**

Carry out a series of tests by changing the amount of available RAM as well as the number of processors of the machine on which the application under test is running. As the number of resources increases, the application should run faster. We need to find out how much, and how this will affect stability.

Types of scalability that are tested during the testing:

Vertical scaling - to increase performance of each individual system component (adding RAM to the server, CPU replacement, etc.) to improve performance of the whole system;

Horizontal scaling - allocation of the system to more servers running in parallel and performing the same functions;

Temporary scaling inside the system by queues, asynchronous requests, etc.

**Environment**

Virtual machine emulated with Oracle VM VirtualBox Manager Version 6.0.10:

|  |  |
| --- | --- |
| OS | Windows 10 |
| Base Memory | 1024 - 6144 MB |
| Processors | 1 - 6 |
| Acceleration | VT-x/AMD-V, Nested Paging, PAE/NX, Hyper-V Paravirtualization |
| Attached to | Bridget Adapter |
| Adapter Type | Intel PRO/1000 MT Desktop (82540EM) |
| Promiscuous Mode | Deny |
| MAC Address | 0800272FEB0A |

Host Machine:

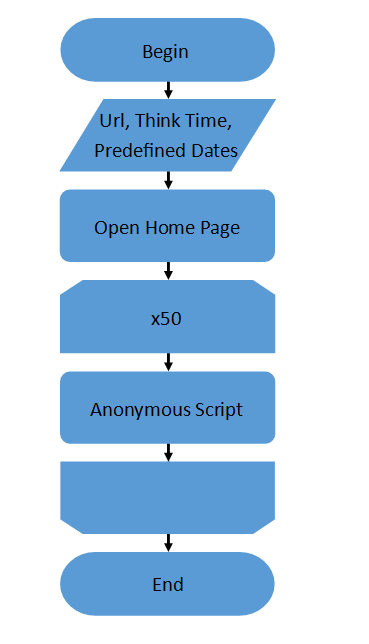
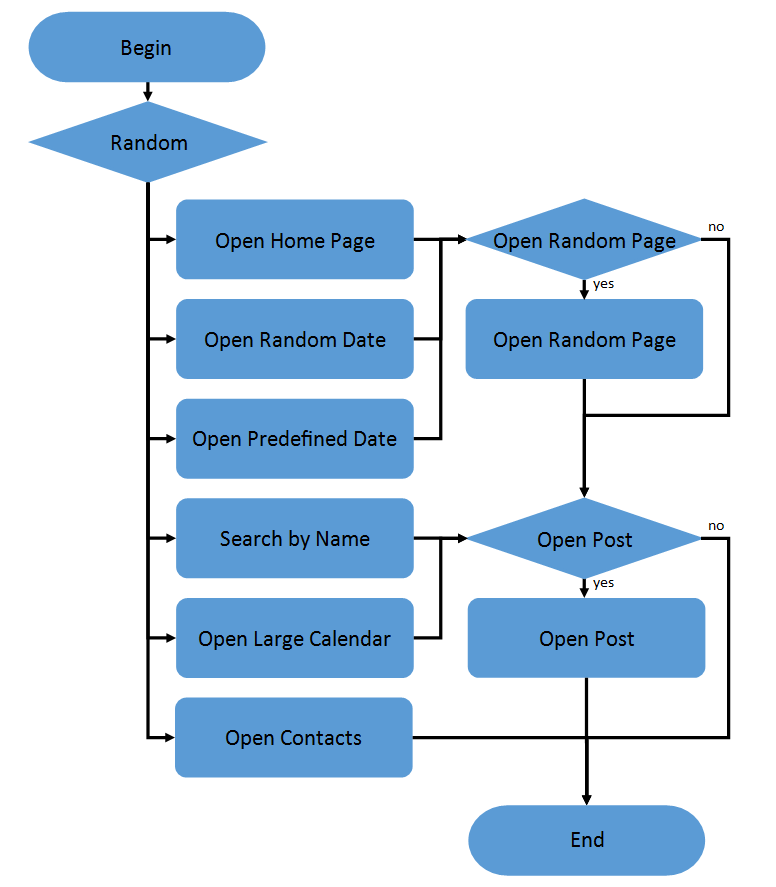
|  |  |
| --- | --- |
| Processor | Intel(R) Core (TM) i7-8700 CPU @ 3.20GHz 3.19 GHz |
| Installed memory(RAM) | 32.0 GM (31.7 GB usable) |
| System Type | 64-bit Operating System, x64-based processor |

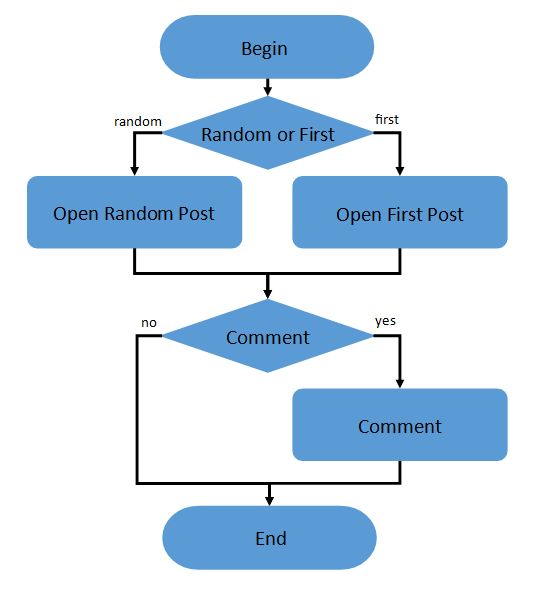
Parameters:

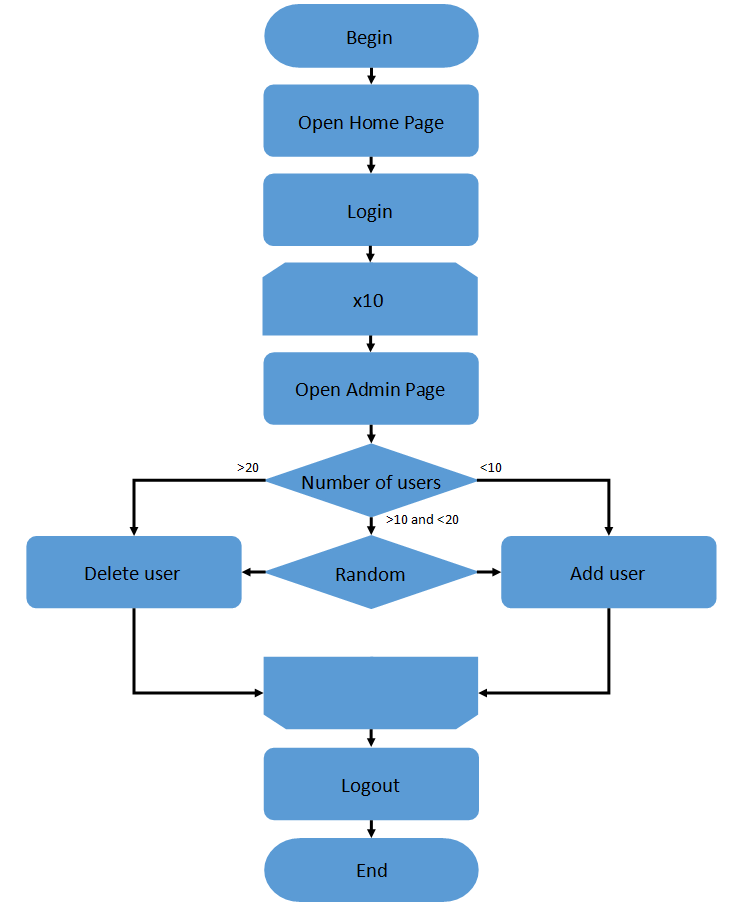
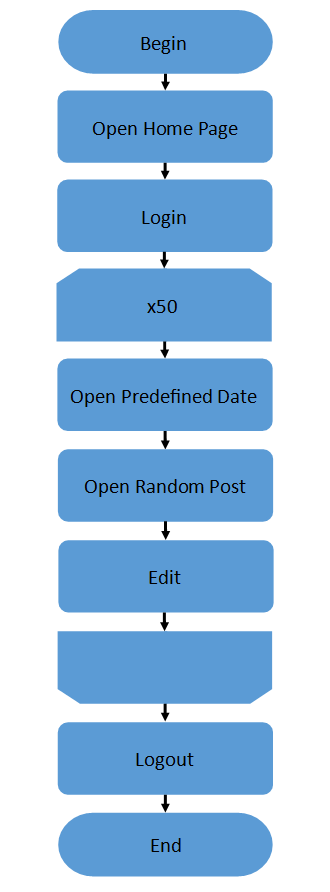
|  |  |
| --- | --- |
| Server Name | 10.66.154.88/blog |
| Protocol | http |
| Timer delay | 3000 ms |
| Timer deviation | 1000 ms |

**Load Model**

**Anonymous script:**

****

**Admin script: Editor script:**

**** 

|  |  |
| --- | --- |
| Number of anonymous users | 33 (85 % from maximum load) |
| Number of editor users | 1 |
| Number of admin users | 1 |
| Ramp-Up-Period | 2.5 minutes |
| Loop Count | Forever |
| Duration | 30 min |

**Tests**

|  |  |
| --- | --- |
| **Memory** | **Cores** |
| **1 Gb** | **6 CPUs** |
| **2 Gb** | **6 CPUs** |
| **3 Gb** | **6 CPUs** |
| **4 Gb** | **6 CPUs** |
| **5 Gb** | **6 CPUs** |
| **6 Gb** | **6 CPUs** |
| **6 Gb** | **1 CPUs** |
| **6 Gb** | **2 CPUs** |
| **6 Gb** | **3 CPUs** |
| **6 Gb** | **4 CPUs** |
| **6 Gb** | **5 CPU** |

**Results**



At first glance, the results can be called controversial, but certain regularities can be found. Let's start by looking at the number of cores tests.

We can confidently say that an application is unable to handle the load having a single core. The median is larger than the standard 0.012 only for this test. Other metrics also differ significantly from the trend. However, if you consider the following tests, you will notice that the values of metrics are increasing. It may seem strange, but it can be explained.

If you look for recommendations for using cores on virtual machines, you will find that you should start with a **single core configuration and increase only as needed**. Using more cores not only increases the number of transmission channels between cores, but also the time it takes to receive data from the physical processor. In addition, it is becoming increasingly difficult to maintain load balance between cores.

**So the increase in metrics is mainly due to the fact that the application has been tested in a virtual machine.**

As far as memory is concerned, it becomes simpler and the application works faster when providing the system with more memory. We can draw a conclusion at least using test results with 2, 5 and 6 gigabytes. Tests with 3 and 4 failed because the application failed. Most likely, there is a correlation between the amount of memory allocated to the system and **fault tolerance which is decreasing**.

From all this we should conclude that vertical scaling has an ambiguous effect on the performance and stability of the application, and we should expect **unpleasant consequences** if we do not pay attention to it from the beginning.