Bristol City Council

Performance Testing Case Study

www.etesting.com

Overview

e-testing, a registered G-Cloud organisation providing specialist IT services to the public sector, was selected to design and build a performance testing capability for use on the Bristol City Council (BCC) Digital Services Programme. The Programme itself is a drive to deliver joined up digital services that the Council's customers need and want, with the ultimate aim of providing online "Digital Services so good that people prefer to use them". The platform is based on an integrated set of technologies and systems using open data, open standards and open-source (where possible). Information must be shared seamlessly whilst reducing overall cost.

The system is a Java Frontend, running on a Java Portal server, with TIBCO middleware and linking to several internal and external applications, including Liferay, Salesforce and other third party infrastructure / applications, including Capita Payments.

Our brief was to develop an open-source performance testing framework to initially test the performance of the system at the current stage in the project, and also to integrate it with the Agile development methodology in use, allowing regular performance tests to be run, aligned with the continuous integration deployment of code. We recommended JMeter as the open-source performance testing tool, alongside Jenkins' Continuous Integration Server to deliver the assignment.

Objectives

Performance testing was focused on ensuring that the desired number of users could simultaneously access the platform, via the typical user journeys expected through the systems. The main objectives were to determine whether the existing infrastructure could accommodate the initial externally facing modules for public access; Parking Renewals, Concessionary Travel and Complaints, while handling the anticipated levels of traffic and delivering an acceptable level of performance.

The overall system under test included several constituent systems with multiple interfaces. It was important for BCC to identify where bottlenecks occurred, in order to progress problems with a number of third parties.

While developing and executing the initial performance tests, a secondary objective was to deliver a robust and scalable performance testing framework into the Agile project delivery process, allowing early and frequent testing throughout the development process - as more functionality is added to the core products over time. The transferring of skills and knowledge were also important deliverables for BCC – to allow internal staff to run the tests unaided (and automatically) once e-testing had completed the initial project.

Approach

The process followed at BCC is detailed below:

Test Preparation

Planning: The first stage was to develop a comprehensive test plan to agree; scope, transactions, user numbers, environments, testing objectives, entry & exit criteria, test data, reporting structure, defect management processes, schedule and contact details for all personnel on the project. This included obtaining agreement on all key scenarios to be tested as a part of the project, including:

- Unregistered Users Getting Resident and Visitor Parking Permits
- Unregistered Users Getting Only Visitor Permits
- Registered Users Getting Resident and Visitor Parking Permits
- Registered Users Getting Only Visitor Permits
- User Creating New Accounts only
- Unregistered User Applying for Concessionary Travel Bus Pass
- Parking Renewals
- Complaints

Test data volumes were then calculated and obtained - in this case, a list of addresses and postcodes. All of the scenarios were dependent on postcode and address data, which were listed in text files to be accessed by JMeter during the tests.

Once the test plan had been agreed and the data had been provided, automated scripts were generated for all scenarios in JMeter, before being integrated into the overall Framework. In parallel with this activity, the test environment and load generators were set up and tested.

The final preparation step was to ensure that Server Monitors were in place and working, in order to identify any problems and pinpoint bottlenecks. JMeter performance monitors and shell scripts were put in place on key servers to gather the necessary statistics. Ant was integrated with JMeter to generate summary reports, to obtain additional data like page hits, transactions and error codes. JMeter listeners were used and Server monitoring was done by running scripts on the Linux server, and for Windows servers, perfmon counters were configured and run.

Text Execution

A set of tests were run initially on the application, a subset of which could be run automatically overnight via the Jenkins server.

1. **Debug Test:** 10 users over 15 minutes – a smoke test to ensure the scripts remain valid and that all interfaces were in place. Without this test running successfully, the other tests would not be executed.
2. **Normal Load, Parking Module:** 50 users over 30 minutes – a mix of pre-registered and unregistered users carrying out normal user activities on the system, applying for and renewing parking permits.
3. **Normal Load, Concessionary Travel Module:** 50 users over 30 minutes – a peak load of users requesting concessionary travel vouchers only, in order to check this module of the system individually.
4. **Normal Load, Parking Module:** 50 users over 30 minutes – pre-registered users only applying for new and renewal parking permits. This scenario was developed to distinguish between the full activity of users registering from scratch and then applying for parking permits, with applying for parking permits alone, using pre-registered accounts.
5. **Peak Load, All Scenarios:** 200 Users, 1 hour – All scenarios combined together and tested simultaneously, in order to attain a real world scenario of system use at peak volumes.
6. **Soak Test, All Scenarios:** 120 users, 2 hours – All scenarios combined together at a realistic volume and run over a longer period in order to identify any issues that occur over time, particularly memory leaks.

Bristol City Council

Performance Testing Case Study

www.etesting.com

Initial tests generated a lot of 500 errors and slow transaction times when requesting permits or travel vouchers, even at relatively low levels of load. Configuration changes were initially made to the Parking Module and performance did improve, before the Concessionary Travel (CT) module was integrated into the system. On implementation, the CT module performed well at normal levels of use, giving the team confidence that the integration could proceed. Once this element was proven in isolation, load scenarios on both the Parking and CT modules could be executed. Initial tests again generated 500 errors, although a lower number which indicated a bottleneck with a third party component, the postcode lookup.

These problems continued into the stress tests, with the initial tests being ramped up to 300 concurrent users which also resulted in a number of both 500 and subsequently 503 errors. At this point the decision was taken to scale down the stress test to 200 users, as this was in line with the expected level of users who would be accessing the system once made live. The 503 errors were as a result of a proxy server going into 'protective mode' once the number of 500 errors increased, in order to protect downstream services. Changes to the proxy server were made at this point, to reduce these errors and streamline the process, allowing slightly longer response times for transactions including postcode and vehicle lookup.

The Soak test was then run, which initially did indicate a high but successful average and maximum response time for key transactions.

After each test an Interim Test report was generated, which included details of transaction summaries, server stats, transactions per/second, hits per/second and graphs, with a summary of the test result and any observations/ recommendations.

Once all the tests were completed, a final report was produced, summarising the performance test phase as a whole.

Jenkins

Jenkins Integration

The JMeter framework, originally built on Windows was customised to Linux and deployed to the Linux Jenkins build server. Jenkins was configured to run the JMeter file and was scheduled to run every day at 2300Hrs for the selected scenario which was usually based on a low number of users for 15-13 minutes as a performance smoke test. If any issues were identified by the overnight test, it was then possible the next morning to run additional scenarios, to further isolate the problems so that fixes could be put in place.

Bristol City Council

Performance Testing Case Study

www.etesting.com

Handover

Once the tests were complete, BCC took ownership of the automated scripts and scenarios used in the testing, as well as the integration with Jenkins in the development environment. The tests are kept as a re-usable resource so they can be re-run when necessary, for example if a change is made to the environment.

The lead e-testing consultant on site developed a user guide and general documentation about the performance tests and Jenkins integration. This was uploaded to the central test repository, to be used as a reference for BCC staff to run the tests or make changes to the configuration. The guide included standards, best practice, naming conventions, location of scripts, monitoring and results reporting format / templates.

Conclusion

The implementation of JMeter and Jenkins at BCC allowed a series of tests to be built and run against an evolving platform to provide the Council's key services online. The tests were structured to gradually prove each element of the system individually, at increasing volumes, before modules were integrated together and the performance of the solution as a whole could be measured. Enhancements and configuration changes were made along the way, so that BCC could be confident that the new platform would stand up to the expected load, at normal and peak volumes.

The integration of the performance testing framework together with Jenkins Continuous Integration Server allowed the automatic scheduling of tests from the development environment on a nightly basis, to allow BCC staff to immediately see the impact of any changes made to the code via an automated solution, in line with Agile principles, releasing and testing often. The performance testing framework is completely open-source, in line with the Council's ethos as performance is to be monitored automatically on a regular basis, with no associated tools costs. The framework was designed to be modular, as well as easy to maintain and enhance, as the Digital Services Programme progresses into moving other areas of the Council's services online.

e-testing are a UK based provider of specialist software testing services including; QA & testing services, training courses and recruitment. With over 15 years of experience in software testing and QA, our team of experts have assisted organisations of all shapes and size with delivering quality software projects of the highest order.

Government Procurement Service Supplier

CENTRE FOR ASSESSMENT LTD ISO 9001 UKAS MANAGEMENT SYSTEMS 120

CONTACT

For more information, please contact us via:

Info@etesting.com | 020 8905 2761

www.etesting.com