|  |  |  |
| --- | --- | --- |
| Section 1 | Click Instructions | Talking Points |
| Beacon.pngimage.png  **Click Here** | 1. Ensure PartsUnlimited is loaded into Visual Studio 2017. |  |
| Beacon.pngimage.png  **Click Here** | 1. Right-click the solution, and select Add -> New Project. | We're going to add a new project to hold our web and load tests. |
| Beacon.pngimage.png  **Click Here** | 1. Select Web Performance and Load Test Project, name the project PartsUnlimited.LoadTests and click OK. |  |
| Beacon.pngimage.png  **Click Here** | 1. Click the Record button. | We now need to record some web tests that reflect human behavior. We'll start with a simply search pattern, with no actual purchase. |
| Beacon.pngimage.png  **Click Here** | 1. Browse to the PartsUnlimited web site. |  |
| Beacon.pngimage.png  **Click Here** | 1. Search for batteries. |  |
| Beacon.pngimage.png  **Click Here** | 1. Stop the recording. | We'll assume this is a very short search session and stop the recording now. |
| Beacon.pngimage.png  **Click Here** | 1. Point out failure of test. | Notice that the test is failing. This is partially because of some extraneous requests. Let's get rid of those. |
| Beacon.pngimage.png  **Click Here** | 1. Delete all steps reaching out to visualstudio.com, or for theglyphicons. |  |
| Beacon.pngimage.png  **Click Here** | 1. Rename WebTest1 to Search.webtest |  |
| Beacon.pngimage.png  **Click Here** | 1. Click the button to Parameterize Web Servers | We're going to want to potentially run this against many different URLs, for instance Dev, Test and Production sites. To do that we need to parameterize the web server name. |
| Beacon.pngimage.png  **Click Here** | 1. Click OK. |  |
| Beacon.pngimage.png  **Click Here** | 1. Run the web test. |  |
| Beacon.pngimage.png  **Click Here** | 1. Right-click the Search request, and Add Validation Rule. |  |
| Beacon.pngimage.png  **Click Here** | 1. Choose Find Text, and set the text to find as Battery. 2. Click OK. |  |
| Beacon.pngimage.png  **Click Here** | 1. Run the test. |  |
| Beacon.pngimage.png  **Click Here** | 1. Click on the 2nd request |  |
| Beacon.pngimage.png  **Click Here** | 1. Click the Details tab. | As you can see, we're validating that we can find the text 'battery'. |
| Beacon.pngimage.png  **Click Here** | 1. Add a new Web Performance Test. | At this point, we need to add another user behavior. Let's add some functionality for people actually making a purchase. |
| Beacon.pngimage.png  **Click Here** | 1. Browse to the PartsUnlimited web site, and click Log In. |  |
| Beacon.pngimage.png  **Click Here** | 1. Log in with an appropriate user. |  |
| Beacon.pngimage.png  **Click Here** | 1. Click Brakes. |  |
| Beacon.pngimage.png  **Click Here** | 1. Select the first set of brakes returned. |  |
| Beacon.pngimage.png  **Click Here** | 1. Click Add to Cart. | Let's add these brakes to our shopping cart. |
| Beacon.pngimage.png  **Click Here** | 1. Click Checkout. |  |
| Beacon.pngimage.png  **Click Here** | 1. Enter appropriate details and click Submit Order. |  |
| Beacon.pngimage.png  **Click Here** | 1. Highlight the Thanks for your order! response, and copy it. | We're going to want this text later, so let's grab it now. |
| Beacon.pngimage.png  **Click Here** | 1. Stop the recording. |  |
| Beacon.pngimage.png  **Click Here** | 1. CS29: Click Here |  |
| Beacon.pngimage.png  **Click Here** | 1. Right-click the last request and select Add Validation Rule. | Let's add a new validation rule to the last request. |
| Beacon.pngimage.png  **Click Here** | 1. Paste the "Thanks for your order!" text into the Find Text area and press OK. |  |
| Beacon.pngimage.png  **Click Here** | 1. Rename the webtest to But.webtest. |  |
| Beacon.pngimage.png  **Click Here** | 1. Parameterize the web server. |  |
| Beacon.pngimage.png  **Click Here** | 1. Run the test. |  |
| Beacon.pngimage.png  **Click Here** | 1. Note the first failure. Click to expand it. | Looks like we have a failure. Let's drill in a bit deeper. |
| Beacon.pngimage.png  **Click Here** | 1. Click on the last request, which is failing. | We're trying to download some jquery stuff. That's not really relevant, since people will only need to download the first time, so lets delete that request. |
| Beacon.pngimage.png  **Click Here** | 1. Open the Buy.webtest tab. |  |
| Beacon.pngimage.png  **Click Here** | 1. Open up the Properties page. |  |
| Beacon.pngimage.png  **Click Here** | 1. Change Parse Dependent Requests to False. |  |
| Beacon.pngimage.png  **Click Here** | 1. Run the test. |  |
| Beacon.pngimage.png  **Click Here** | 1. Notice you have more failures. Correct those, too. | We need to correct the dependent requests issue for the next set of errors, too. |
| Beacon.pngimage.png  **Click Here** | 1. Once you've corrected the last dependent request failures, run the test again. |  |
| Beacon.pngimage.png  **Click Here** | 1. Note we still have a failure in the last request. |  |
| Beacon.pngimage.png  **Click Here** | 1. Click on the error. | It looks like Order Number is being used as part of the URL, but that changes every time. That's why it's breaking. |
| Beacon.pngimage.png  **Click Here** | 1. Open the Buy.webtest file. |  |
| Beacon.pngimage.png  **Click Here** | 1. Remove the Response URL from the Validation Rules. 2. Rerun the test. |  |
| Beacon.pngimage.png  **Click Here** | 1. Highlight that everything is passing. | OK. We finally have a good test that's passing, and is able to be used to generate a load. |
| Beacon.pngimage.png  **Click Here** | 1. Right-click the LoadTests project, and select Add -> Load Test. | At this point, we have a small number of web tests. Let's group them into a load test. First we need to add a load test to the solution. |
| Beacon.pngimage.png  **Click Here** | 1. Select On-premise Load Test | We can run load tests either in the cloud, or on premises. We're going to use the built in Load Test in VS Enterprise for this demo. |
| Beacon.pngimage.png  **Click Here** | 1. Click Next. |  |
| Beacon.pngimage.png  **Click Here** | 1. Click Next. | We're going to use a random distribution of think times related to the think times we recorded. |
| Beacon.pngimage.png  **Click Here** | 1. Click Next. | We'll use 25 users. |
| Beacon.pngimage.png  **Click Here** | 1. Click Next. | And we'll take the default testing distribution. |
| Beacon.pngimage.png  **Click Here** | 1. Click Next. Then click Add. |  |
| Beacon.pngimage.png  **Click Here** | 1. Add both Buy and Search to the selected Tests. | We're going to generate the load with these two tests. |
| Beacon.pngimage.png  **Click Here** | 1. Set the distribution so more people are Searching than Buying. 2. Click Next. | More people are likely searching than buying. Let's reflect that in the load test. |
| Beacon.pngimage.png  **Click Here** | 1. Click Next. |  |
| Beacon.pngimage.png  **Click Here** | 1. Add Chrome, distribute percentages, and click Next. |  |
| Beacon.pngimage.png  **Click Here** | 1. Add a Computer. Change name to localhost. |  |
| Beacon.pngimage.png  **Click Here** | 1. Monitor the ASP.NET counter set and click Finish. |  |
| Beacon.pngimage.png  **Click Here** | 1. Run the test, it should take 2 minutes. | Now, we're finally ready to run the test and start getting feedback. |
| Beacon.pngimage.png  **Click Here** | 1. When the test is done running, look at the resulting tabs. |  |
| Beacon.pngimage.png  **Click Here** | 1. Explore down to the problem issue. In this case perf. |  |
| Beacon.pngimage.png  **Click Here** | 1. Show the failure in the graphs. | Looks like we've got some heavy CPU utilization going on. We might want to look into this at some point. |