The eG Enterprise Web App Simulator

eG Innovations Product Documentation

www.eginnovations.com

Total Performance Visibility

Table of Contents

Table of Figures

Chapter 1: Introduction

As IT infrastructures evolve into being business-critical, high availability and peak performance of the IT infrastructure is becoming as critical as reliability is to a mobile network. In such infrastructures, it is imperative to determine in real-time when service/web application failures or slowdown problems occur, and to accurately pin-point which transaction(s) in the service access / web application access are impacting the end-user experience. Speedy problem detection and accurate diagnosis can reduce service / web application downtime and the consequent business impact of critical IT services.

eG's client session simulation and monitoring capability goes well beyond basic protocol-level up/down testing. IT administrators can record typical user accesses to mission-critical infrastructure services / web applications, and later have eG agents periodically playback the recorded accesses using the exact same client applications that users employ to access a service / web application. The eG client session simulator effectively simulates multi-step user interactions with a service / web application (e.g., login, browse, submit data, fill-in forms, etc.). For this purpose, eG Enterprise integrated with Itexis AppsMon for Windows which is a general purpose tool.

Using Itexis AppsMon for Windows to collect the availability and responsiveness of the target service / web application has its own limitation. The first and foremost limitation was that the tool was not HTML-aware. In modern times where transactions performed are mostly browser based, it becomes difficult to track the transaction availability and responsiveness with a tool that is not HTML-aware. Also, if eG integrates with Itexis AppsMon for Windows for monitoring, customers of eG Enterprise are required to purchase a separate license for Itexis AppsMon for Windows. This increased the operational cost of the monitored environment. To reduce the cost incurred by eG's customers and to monitor the web applications with ease, eG Enterprise has developed a purpose-built, in-house tool called Web App Simulator.

The Web App Simulator is a standalone desktop tool that is packaged with the eG agent and is installed on the computer/VM from where the simulation will happen. The Web App Simulator is used to simulate the required web transactions and record the entire simulation to be played back at custom intervals later on. A specialized monitoring model named *Web App Simulation* has been built by the eG Enterprise to periodically play back the simulations and collect the required metrics.

1.1 Benefits of the Web App Simulator

- Emulate real user interactions to Web site from multiple locations and record website availability and end-to-end response times;

- Identify which transaction(s) of a web application/web site interaction are causing slowdowns;
- Provide instantaneous alerts on service outages and slowdowns;
- Facilitate root-cause diagnosis and infrastructure optimization through correlation with other network, server, application performance indicators;
- Offers a reliable way of automating routine service health checks;
- Easy to use application. Web App Simulator is independent of image recognition technique unlike Itexis AppsMon for Windows.
- Recorded simulation is independent of screen resolution. Administrators can record the simulation in a machine and play it on a different machine which has a different screen resolution.

Chapter 2: How does the Web App Simulator Work?

Typically, the first step in using the Web App Simulator is to record a sequence of user activities when accessing the web site(s) to be monitored. For this purpose, eG Enterprise offers a purpose-built, in house Web App Simulation Recorder. The end result of the recording process is a script that can be later played back by the Web App Simulation Playback Engine to emulate the transaction activity.

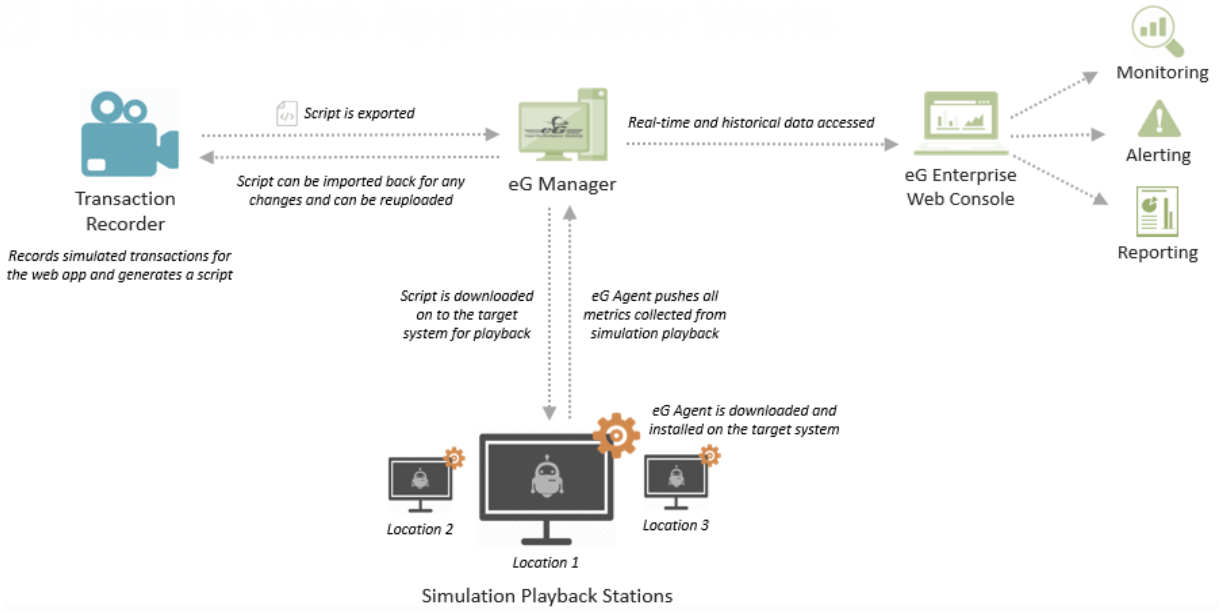

Figure 2.1: How the Web App Simulator works

The recorded script not only includes the sequence of transactions necessary to emulate the transaction activity , but also has information on those transactions - each transaction corresponds to a single user interaction (e.g., logging in, adding to shopping cart, etc).

The Web App Simulation Playback Engine is an executable that is bundled along the eG agent and is installed on remote locations (playback stations) from where the simulation playback happens. The recorded script is played back on a dedicated system by the Web App Simulation Playback Engine, and the results are reported as availability and response time measures for the overall playback activity. In addition, response times of each of the transactions involved can be obtained so as to assist the IT service manager in pin-pointing which of the transactions of the multi-step user interaction with the web application could be causing a slow-down of the overall website /web application.

The eG agent is provided with the location of the recorded script that it has to execute to emulate a user activity. Based on the pre-specified frequency of the test, the agent executes the script, analyzes the script results, and reports availability and response time information to the eG manager. The eG manager receives these web app simulation reports and correlates them in real-time with critical in-depth resource usage and server-side processing metrics that it receives from the other agents, to report on potential bottlenecks in the target IT infrastructure.

The eG external agent implements the integration with the Web App Simulation Recorder. Since the recorder requires a dedicated system to operate on, an external agent can either perform client simulation tests or the other eG protocol simulation tests, but not both. When adding a new external agent, depending on the eG license available, the administrator can specify whether the agent must be allowed to perform client session simulation activity or not.

As stated earlier, a dedicated eG external agent drives the web app simulation. This agent periodically runs a **Simulated Web Transactions** test that emulates the entire process of a user logging into a website and performing a transaction. Since the test is what performs the simulation, let's call it the **Web App Simulator**. To perform this simulation, the Web App Simulation Recorder has to be configured with the URL of the target website.

There are a certain pre-requisites that need to be fulfilled prior to recording the transactions. For this, refer to Section **2.1** .

To know how to record the transaction, refer to Section **3.1**.

Once the sequence of transactions are recorded as a script, it runs at the configured frequency. Every time it runs, it simulates the sequence of transactions recorded in the script and collects the required metrics.

The metrics so collected are reported to the eG manager. The manager captures these metrics into a **Web App Simulation** component and presents them in the eG monitoring console for analysis. Refer to **Analyzing the Simulation Results** for a detailed discussion on the **Web App Simulation** model.

2.1 Pre-requisites for Using Web App Simulator

Before attempting to use the simulator, make sure that the following pre-requisites are fulfilled:

Category	Pre-requisites
Web App Simulator Agent / Simulation Endpoint	• Client Session Simulation capability should be enabled on the eG license.

	• The web app simulator agent / external agent should be installed on a dedicated virtual machine or a physical server running Windows 2008/2012/2016/2019 or Windows 7/8/10 operating system. • The web app simulator agent / external agent should not be used to monitor any other component in the target environment. • Microsoft .NetFramework 4 (or above) should pre-exist on the system hosting the web app simulator agent/external agent. • The simulator also requires a user account with local administrator rights on the simulation endpoint - i.e., on the system hosting the web app simulator agent. This user should be logged in at all times for the simulator to run continuously. Also, make sure that this session window is **not minimized** because this may cause problems in the web app simulation. **Note:** • The web app simulation will not work if the session is closed. • The web app simulation will not work if the screen is locked on the simulation endpoint. • The web app simulator will not work if the screensaver appears on the simulation endpoint.
Browser	Chrome browser v81 (and above) should be available on the dedicated endpoint. **Note:** In some environments where browsers are automatically updated to their latest versions, incompatibility is cited between the browser version and the Chrome drivers. This may sometimes lead to the non start of simulation. Therefore, ensure that the Chrome drivers are also updated whenever the browser is updated to the latest version.

Note:

To ensure that all the pre-requisites of the Web App Simulator is fulfilled, you can execute the **WebAppSimulatorChecks.exe** which is available in the **<eG_INSTALL_DIR>\lib\Synthetic** folder. This executable should be executed by a user with administrator privileges from the command prompt of the target agent host. If any pre-requisite has not been fulfilled, failure will be highlighted in Red (as shown by Figure 2.2).

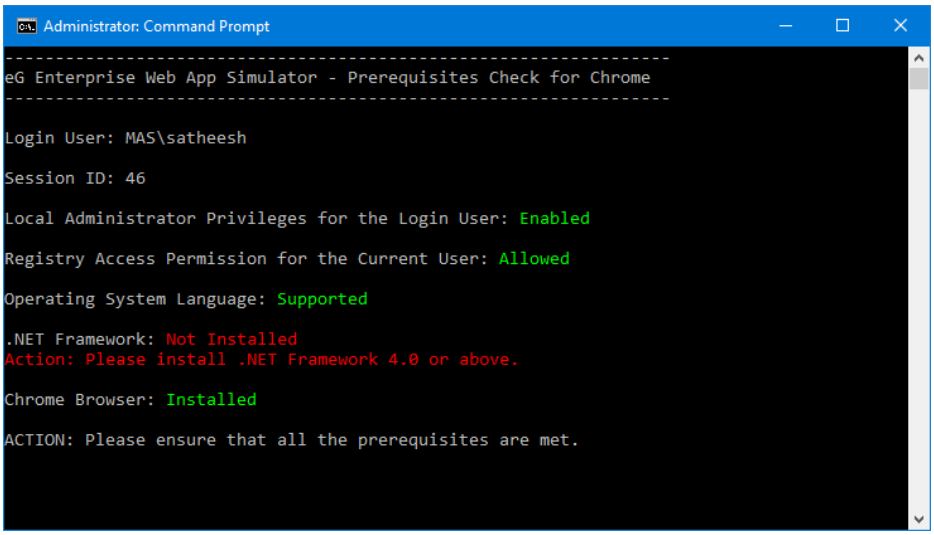

Figure 2.2: Setup script where a pre-requisite has failed

Use the pointers provided in Figure 2.2, just below the failed pre-requisite, to know how to fulfill that requirement. Then, rerun the **WebAppSimulatorChecks.exe** to make sure that all pre-requisites are fulfilled, and then proceed.

If all pre-requisites are fulfilled, then Figure 2.3 will appear.

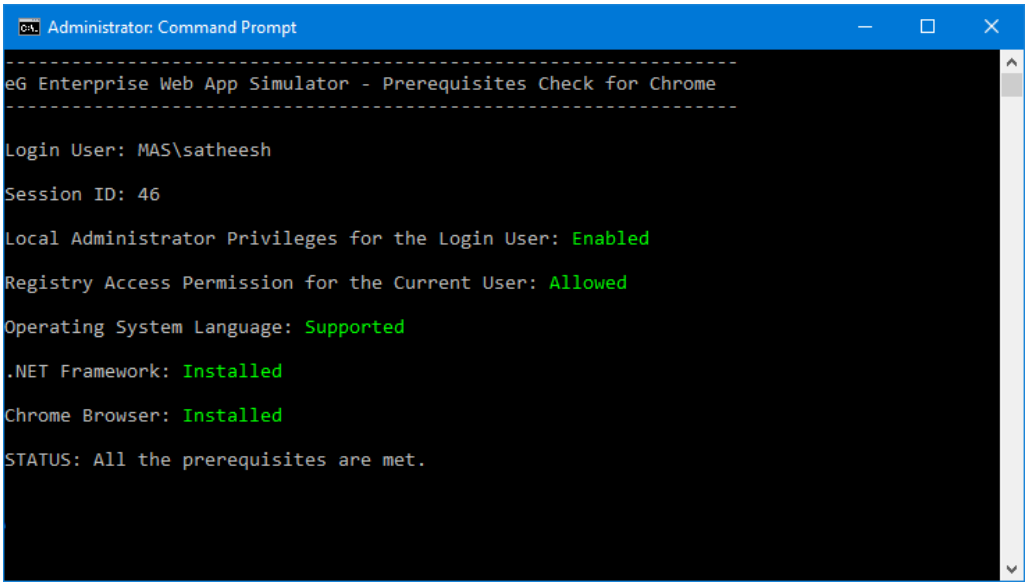

Figure 2.3: All pre-requisites are fulfilled

Troubleshooting

- **The web app simulation was being captured well up until a new Chrome browser update. What should I do in my environment to resume Web App simulation monitoring using the latest Chrome browser?**

  Chrome is capable of automatically applying updates and upgrading itself to higher versions. Sometimes, when Chrome auto-upgrades, some drivers that the eG Web App Simulation Agent uses may suddenly be rendered incompatible with Chrome. This can cause problems in simulation. To avoid this, the Web App Simulator, by default, prevents Chrome upgrades/updates (both automatic and manual) from being applied at the simulation endpoint.

  However, whenever a new version of the eG agent with updated drivers is released, you will have to manually upgrade Chrome to ensure continued compatibility. In this case therefore, you will have to make sure that the simulation endpoint allows Chrome upgrades. To achieve this, before manually upgrading Chrome, follow the steps below:

  - Login to the eG agent host.
  - Open the Windows command prompt as Administrator.
  - Switch to the <EG_AGENT_INSTALL_DIR>\lib directory, and issue the following command:

    ChromeUpgradeHandler.exe enable

- **In our environment, web app simulation is performed via RDP sessions. Your pre-requisites specifies that the session window should not be minimized. How should we ensure that the simulation is not impacted if the RDP session is minimized?**

  If the web app simulation is performed via an RDP session, then, you can make sure that the simulation is not impacted even if the RDP session window is minimized. For this, execute the **RDPSessionInteractiveTask.exe** on the system from which the user has launched the RDP session. This executable is available in the **<eG_INSTALL_DIR>\lib** folder. Once you download and extract the package into any location, you will find the RDPSessionInteractiveTask.exe within.

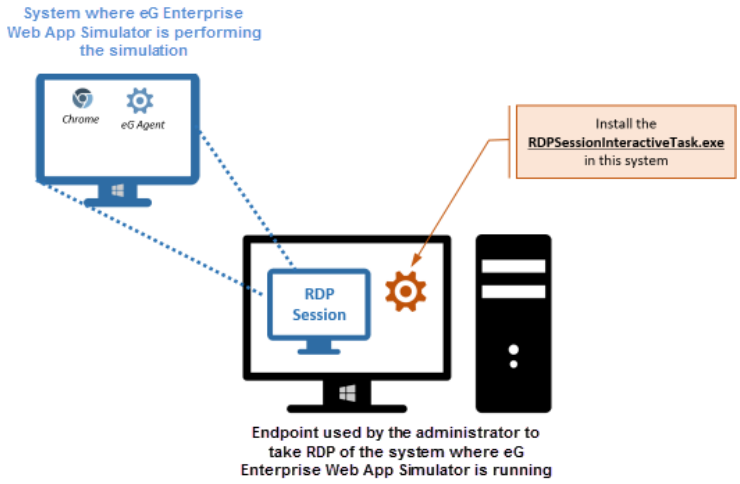

Figure 2.4: Web App Simulation performed via an RDP session

To execute the **RDPSessionInteractiveTask.exe**, do the following:

- Open the command prompt of the system from which the RDP session was launched as an administrator.
- Execute the RPSessionInteractiveTask.exe file. Upon successful execution, a message to that effect will appear.
- Reconnect the RDP session.

Chapter 3: How to Use the Web App Simulator?

The following sections will describe in detail on how to record a transaction and configure the Web App Simulation component for monitoring.

3.1 How to Record a Transaction?

The Web App Simulation Recorder is a standalone desktop tool that is packaged with the eG agent and is installed on the computer / VM from where the simulation will happen. This is used to simulate the required web transactions and record the entire simulation to be played back at custom intervals later on.

To record a transaction, do the following:

1. Login to the host on which the eG agent is installed.
2. Execute the **WebAppSimulationRecorder.exe** from the **<EG_INSTALL_DIR>\lib\Synthetic** folder.
3. Figure 3.1 will then appear.

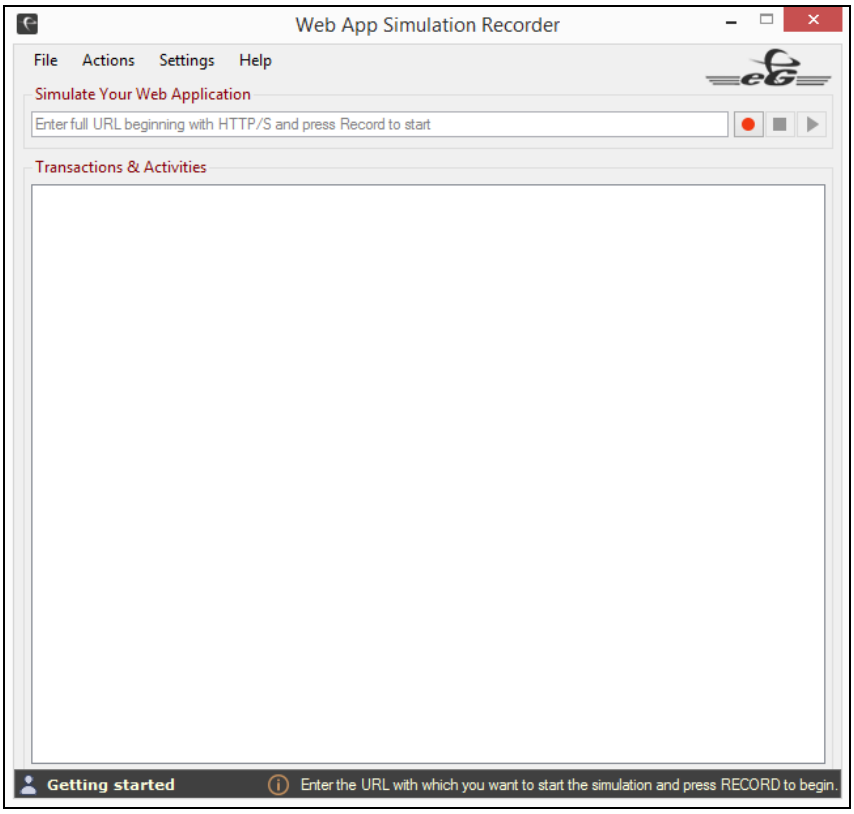

Figure 3.1: The Web App Simulation Recorder

4. Here, specify the URL of the website using which you wish to record a sequence of transactions to build a script. Let us take http://teleportal.acme.com as an example.

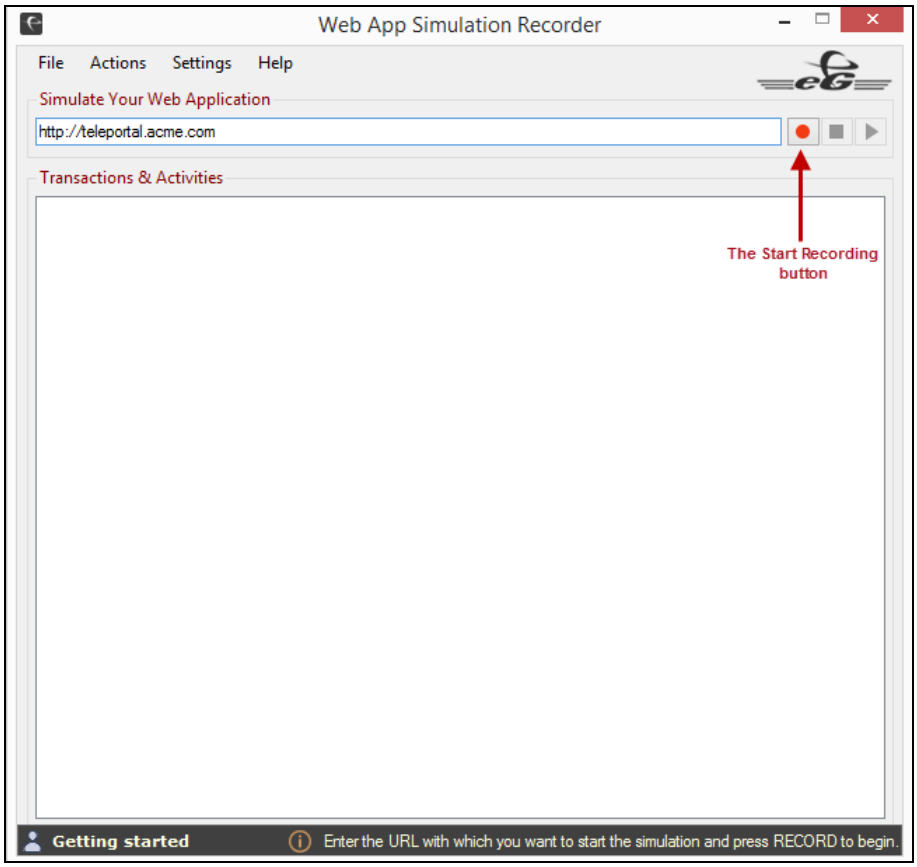

Figure 3.2: Entering the URL for simulation

5. In Figure 3.2, once you enter the URL, click the **Start Recording** button. Google Chrome browser opens automatically and the simulator hits the specified URL in Figure 3.2. Figure 3.3 will then appear prompting you to start the recording.

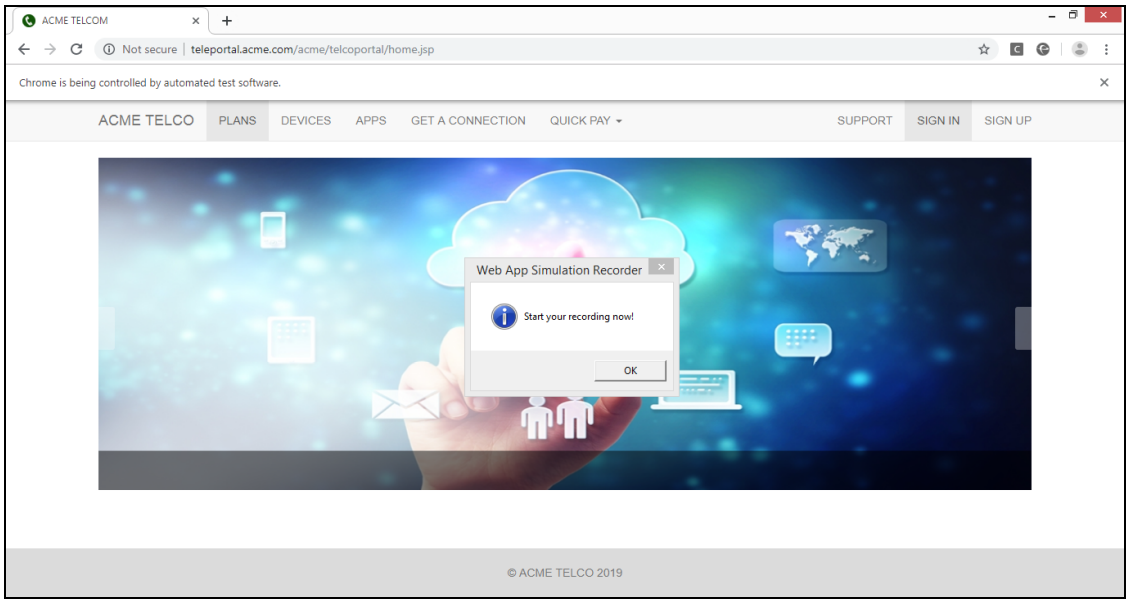

Figure 3.3: Starting the recording

6. Click the **OK** button in Figure 3.3 to start the recording.

7. The next step towards building the script is to start the recording. The primary purpose of the recording is to start capturing the user transactions that are to be emulated.

8. Each activity that you do on the website will be recorded. For example, let us say take the case of recharging a mobile with a pre-paid plan. For this, in our example, we login into the website, ascertain the best pre-paid plan, proceed to recharge by making the payment. These transactions and activity pertaining to the transactions will be recorded in sequence.

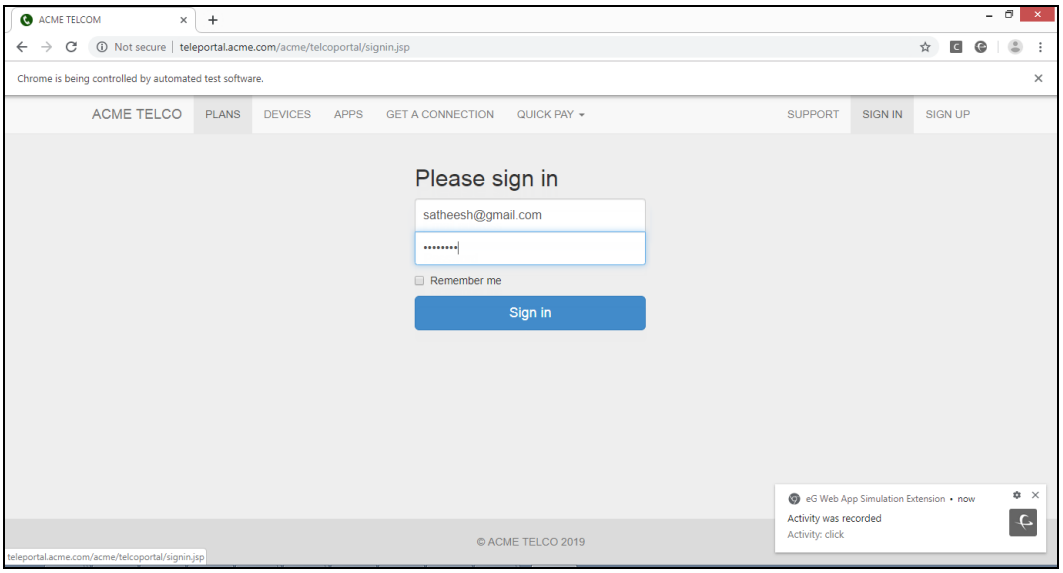

Figure 3.4: Recording an activity

9. Figure 3.4 captures each URL hit on the target web app as a transaction and every action (mouse click, keystroke etc) as an Activity.

10. While you are recording a simulation, you are allowed to verify the page title or a text in the web page. This helps in maintaining a smooth navigation between web pages while replaying the simulation. To verify the page title, right click on the Google Chrome browser and follow the menu sequence: *eG Web App Simulation Extension -> Assert Title* (see Figure 3.5).

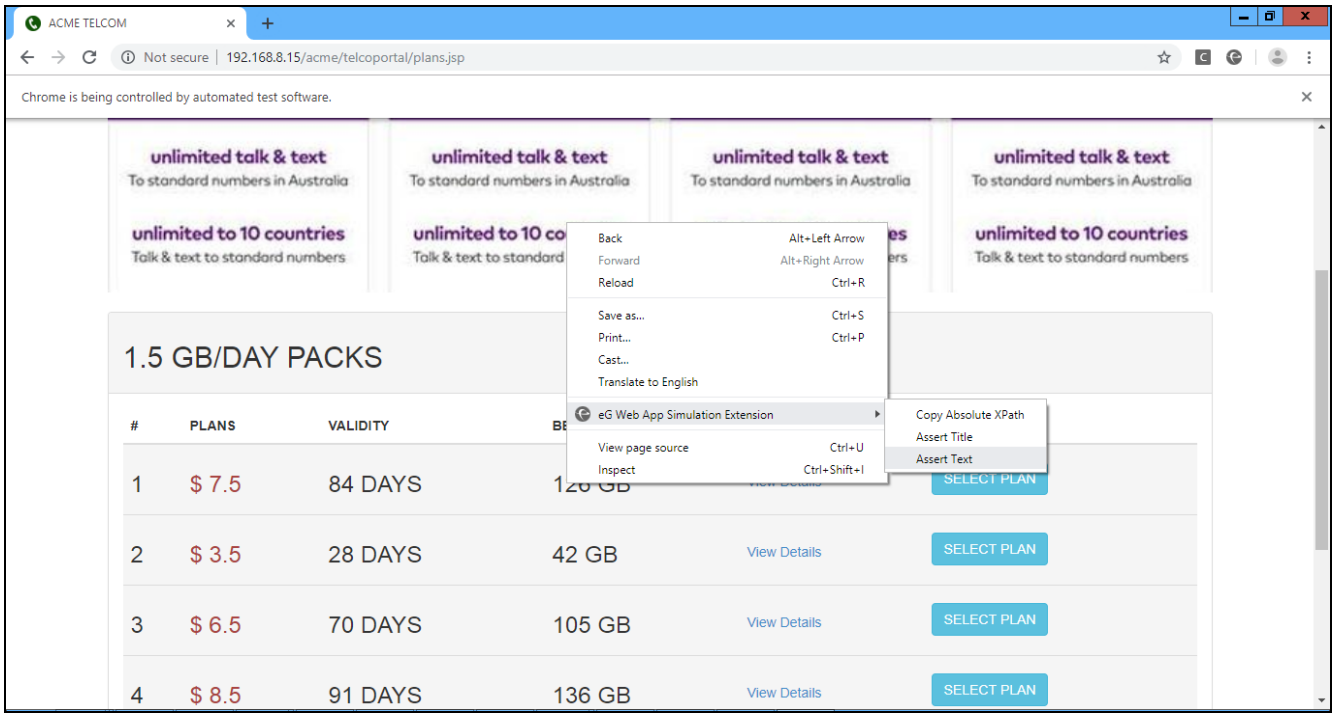

Figure 3.5: Verifying the title of the web page

This will ensure that the title of the web page is verified. In our example, the page title is ACME TELCOM (see Figure 3.5) and the same will be verified while replaying the simulation. This will be added as an activity while performing the simulation (see Figure 3.6). Similarly, you can even verify a text in the web page. For example, you can verify the *Payment Successful* message in the web page while performing the simulation.

11. Once you have performed all the required transactions, you can stop the recording by selecting the **Stop Recording** button that appears in Figure 3.6.

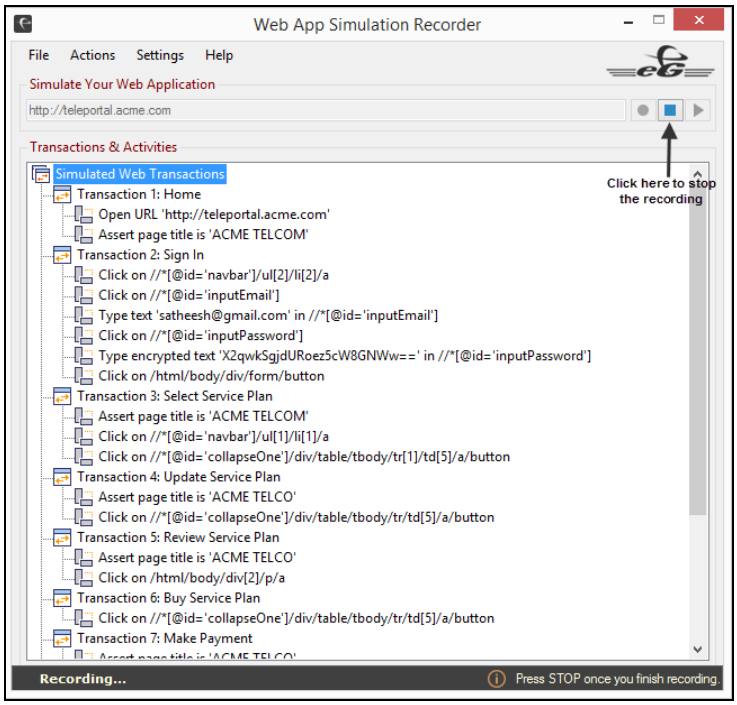

Figure 3.6: Stopping the recording

12. Once you have stopped the recording, you can even replay the recording to verify if you have perfectly recorded the script. For this, press the **Play** button as shown in Figure 3.7.

Figure 3.7: Clicking the Play button to replay the recorded script

13. You are even allowed to modify / delete a transaction at any point of time. For this, you need to right click on a Transaction. Figure 3.8 will then appear using which you can provide your own name for that transaction. **Note that you are not allowed to delete the very first transaction that you have recorded.**

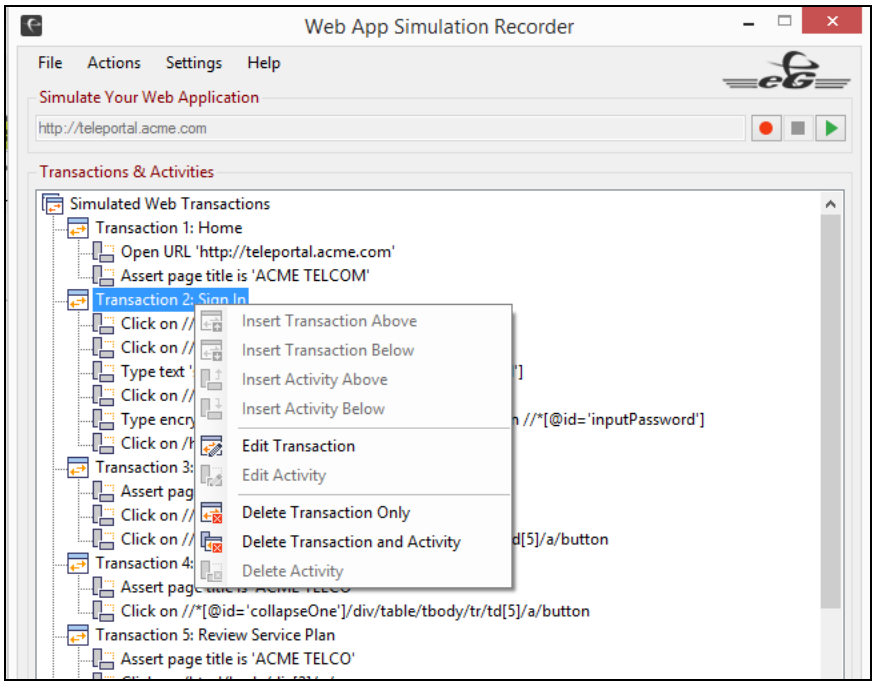

Figure 3.8: Editing a transaction

14. When you click the **Edit Transaction**, Figure 3.9 will appear prompting you to rename the transaction.

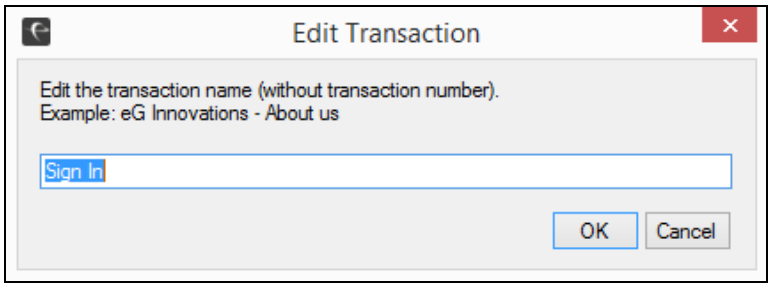

Figure 3.9: Renaming of the transaction

15. Similarly, you can delete an individual transaction alone or delete the transaction along with the activities performed in the chosen transaction.

16. You are even allowed to modify / delete an activity too. For this, you need to right click on an **Activity** and click the **Edit Activity** option. Figure 3.10 will then appear using which you can select the **Activity** and the *Target:Value* pair of the chosen activity.

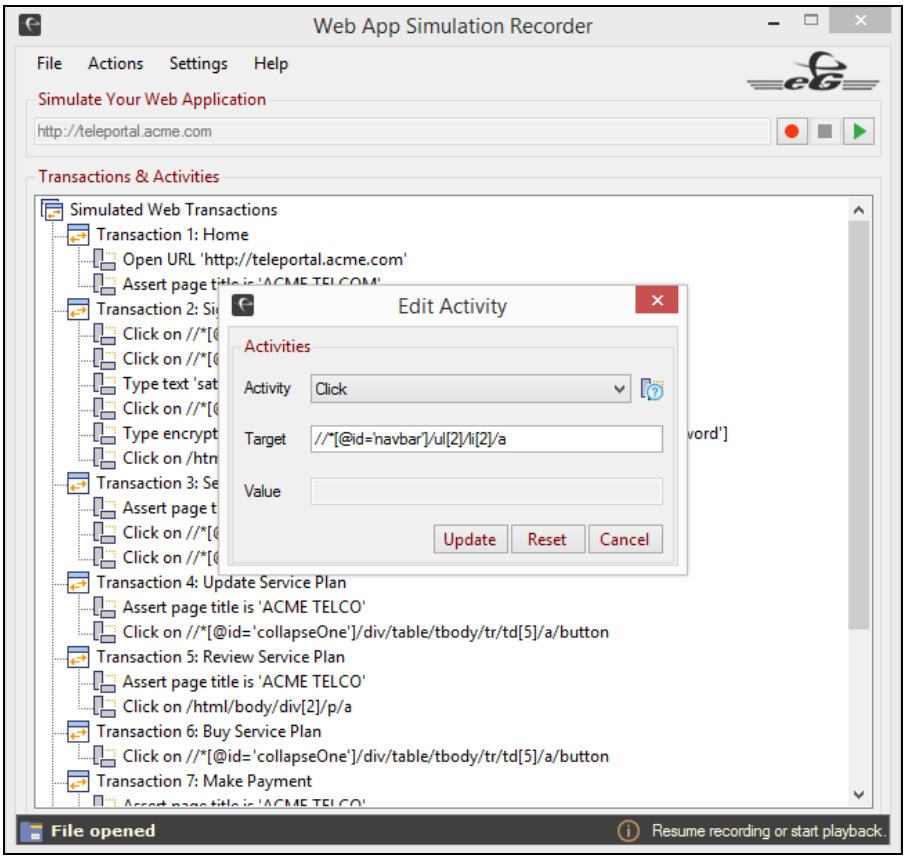

Figure 3.10: Editing an Activity

The below table lists the purpose of each **Activity** in detail.

Activity	Description	Target	Value
AssertText	Asserts an expected page text. Verifies whether the text in the Target location is the same as the Value specified	The location (XPath) of the text that should be matched against the Value.	The string that is expected in the Target location specified.
AssertTitle	Asserts an expected page title. Gets the title of a website and checks it against the provided Target text	The expected title.	Not applicable
ClearandType	Clears and sets the value of an input field, as though you typed it in.	The location (XPath) where the Value should be typed.	The input string to be typed in the Target location.

Click	Clicks on an element (button, link, check box etc).	The location (XPath) of the element to be clicked at.	Not applicable
ClickAt	Moves mouse inside the element in the Target location and clicks it.	The location (XPath) of the element to be clicked at.	Not applicable
CloseWindow	Closes the tabs and windows	Index of the tab to be selected. 0 is the main window/tab, 1 the first tab to the right, 2 the second, and so on.	Not applicable
DoubleClick	Double-clicks on the element in the Target location.	The location (XPath) of the element to be double-clicked.	Not applicable
ExitFrame	Exits the current frame and moves to the top of the web page.	Not applicable	Not applicable
Pause	Delays the execution of the simulation by a specified time.	The duration (in secs) of the delay.	Not applicable
Scroll	Scrolls the web page up or down.	X,Y coordinates of the web page to be scrolled. Example: 0,400.	Not applicable
Select	Selects the specified Value from the element (drop-down/combo box/list box) in the Target location.	The location (XPath) of the element.	The value to be selected from the element in the Target location.
SelectFrame	Switches to the frame that uses the Target index.	Pass the frame index. 0 is the first frame, 1 is the frame below it, and so on. For nested iframe, the format is 0,1,2,3,…n. Example: 0 for an iframe (OR) 0,5,2 for a nested iframe.	Not applicable
Sendkeys	The location (XPath) of the element from which the keystrokes have to be sent.	The location (XPath) of the element from which the keystrokes have to be sent.	Key or Value to be sent. Refer the table below for the format in which the Value needs to be provided:

			<table><tr><th>Key</th><th>Value</th></tr><tr><td>Enter</td><td>${KEY_ ENTER}</td></tr><tr><td>Escape</td><td>${KEY_ESC}</td></tr><tr><td>Tab</td><td>${KEY_TAB}</td></tr><tr><td>Delete</td><td>${KEY_ DELETE} or ${KEY_DEL}</td></tr><tr><td>Backspace</td><td>${KEY_ BACKSPACE} or ${KEY_ BKSP}</td></tr><tr><td>Up Arrow</td><td>${KEY_UP}</td></tr><tr><td>Down Arrow</td><td>${KEY_ DOWN}</td></tr><tr><td>Left Arrow</td><td>${KEY_LEFT}</td></tr><tr><td>Right Arrow</td><td>${KEY_ RIGHT}</td></tr></table>
Submit	Submits the element in the Target location.	The location (XPath) of the element to be submitted.	Not applicable
Type	Sets the value of an input field, as though you typed it in.	The location (XPath) of the element in which the Value is to be typed.	The value to be typed in the element in the Target location.
TypeWithEncrypt	Sets the Value of an input field, as though you typed it in and encrypts that Value.	The location (XPath) of the element in which the Value is to be typed.	The value to be typed in the element in the Target location.

17. By default, there are certain pre-built settings for the Web App Simulation Recorder. To access these settings, navigate through the *Settings -> Recorder Settings* menu sequence in the Web App Simulation Recorder. Figure 3.5 then appears.

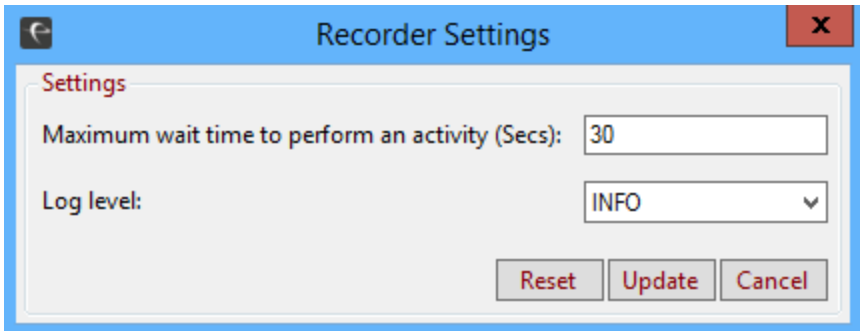

Figure 3.11: The default settings available in the Web App Simulation Recorder

18. The settings mentioned in Figure 3.11 are explained below:

    - **Maximum wait time to perform an activity (Secs):** By default, the transactions recorded in the script are replayed in sequence. In the process, during playback or simulation, the maximum wait time to perform an activity in a transaction is set to 30 seconds. This implies that the script will wait for the transaction activity to complete within 30 seconds beyond which will exit the simulation. If you prefer to set the value for this parameter, you can override the default value.
    - **Log level:** By default, **INFO** is chosen from this list. This indicates that whenever the simulation is performed, information level logs are logged for further processing. You can choose to disable the logging activity by selecting the **OFF** option. You can also enable debug mode by choosing the **DEBUG** option. If you choose **WARNING** or **ERROR** options, the warning messages and error messages (respective to the option chosen) will be logged.
    - Clicking the **Reset** option will set the default values to the fields mentioned above.
    - Clicking the **Update** button will effect the changes made. Clicking the **Cancel** button will cancel the changes made, if any.

19. Once you have performed all the transactions, export the recorded transactions as a script to the eG manager. To do this, click the **Export To eG Manager** option that appears when you click the **File** menu in the Web App Simulation Recorder (see Figure 3.12).

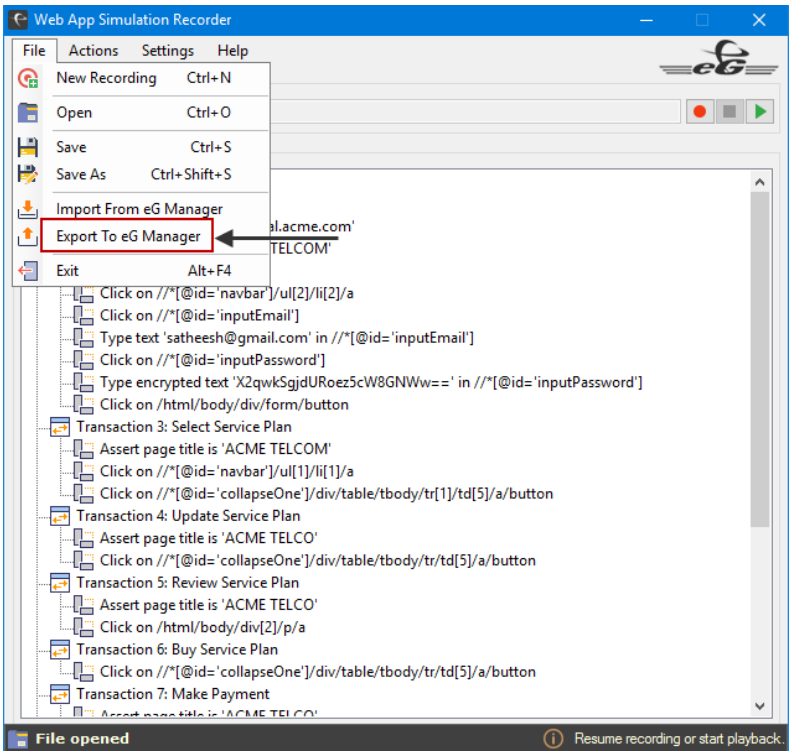

Figure 3.12: Choosing the Export To eG Manager option for exporting the script

20. In Figure 3.13 that appears, specify the following:

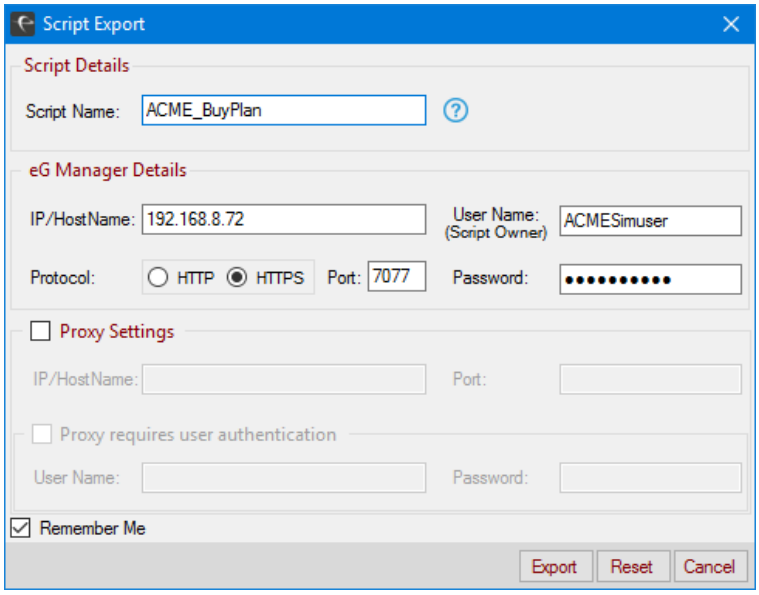

Figure 3.13: Specifying the credentials to export the script

- Specify the name of the script in the **Script Name** text box.
- Specify the IP address or the host name of the eG manager against the **IP/Host Name** text box.
- Against the **User Name (Script Owner)** and **Password** text boxes, specify the credentials of the user who is authorized to export the script and initiate the simulation process.
- If the eG manager is SSL-enabled, then set the **Protocol** flag to **HTTPS**. If the eG manager is not SSL-enabled, then set the flag to **HTTP**.
- Specify the port at which the eG manager listens to against the **Port** text box.
- Alternately, if you wish to export the script via a proxy server to the eG manager, select the check box against the **Proxy Settings**.
    - Specify the IP address or the host name of the proxy server against the **IP/Host Name** text box.
    - Specify the port through which the communication to the proxy server is established against the **Port** text box.
- If the proxy server requires to be authenticated by a user for connecting to the eG manager, then select the check box against the **Proxy requires user authentication** check box.
    - Specify the credentials of the user who is authorized to access the proxy server against the **User name** and **Password** text boxes.
- If you wish to store the credentials that you have provided for future use, then select the **Remember Me** check box.
- Once you have provided all the required credentials, click the **Export** button.
- If you wish to abort the operation click the **Cancel** button.
- If you wish to reset the credentials that you have entered, then, click the **Reset** button.

21. Once the script is exported successfully, a message to that effect will appear as shown in Figure 3.14.

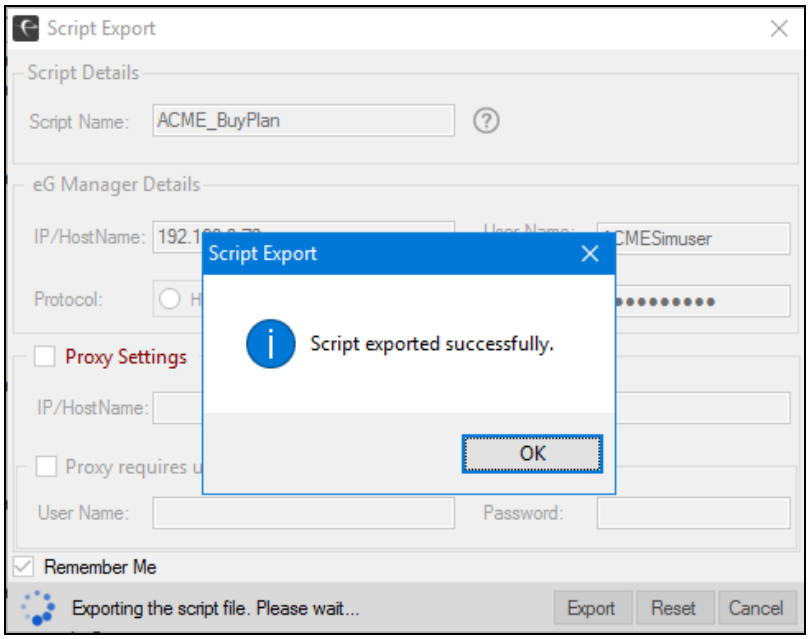

Figure 3.14: Message that appears when the script is exported successfully

22. If you want to add/modify/delete a transaction in an exported script, eG Enterprise provides an option to import the exported script. For this, click the **Import from eG Manager** option that is available in the **File** menu (see Figure 3.12).

23. Figure 3.15 then appears. Specify the following in Figure 3.15.

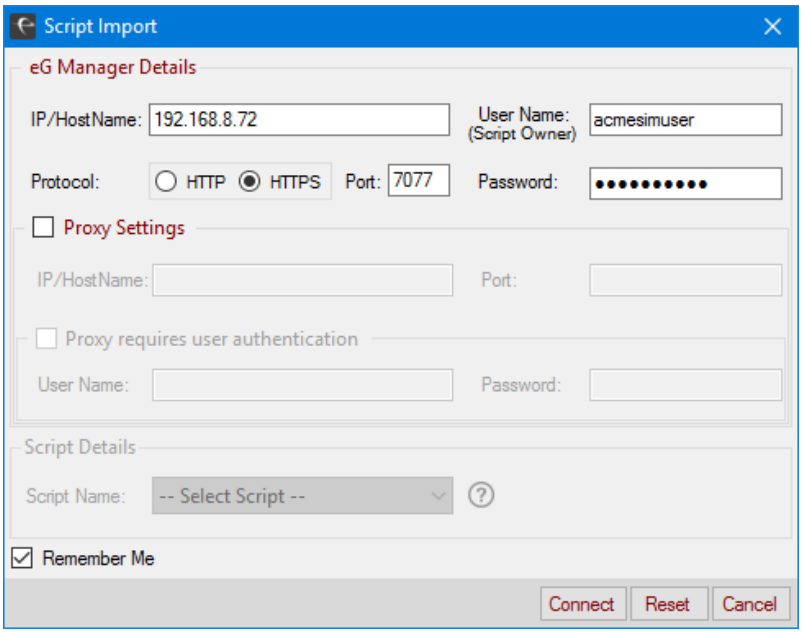

Figure 3.15: Specifying the credentials to import the script from eG manager

- Specify the IP address or the host name of the eG manager against the **IP/Host Name** text box.
- Against the **User Name (Script Owner)** and **Password** text boxes, specify the credentials of the user who is authorized to export the script and initiate the simulation process.
- If the eG manager is SSL-enabled, then set the **Protocol** flag to **HTTPS**. If the eG manager is not SSL-enabled, then set the flag to **HTTP**.
- Specify the port at which the eG manager listens to against the **Port** text box.
- Alternately, if you wish to import the script via a proxy server from the eG manager, select the check box against the **Proxy Settings**.
    - Specify the IP address or the host name of the proxy server against the **IP/Host Name** text box.
    - Specify the port through which the communication to the proxy server is established against the **Port** text box.
- If the proxy server requires to be authenticated by a user for establishing a connection between the eG manager and the machine on which you wish to import the script, then, select the check box against the **Proxy requires user authentication** check box.

- Specify the credentials of the user who is authorized to access the proxy server against the **User Name** and **Password** text boxes.

- If you wish to save the credentials for future use, then select the **Remember Me** check box.
- Once you have provided the above-mentioned credentials, click the **Connect** button in Figure 3.15.
- Clicking the **Connect** button in Figure 3.15 will enable and populate the **Script Name** list as shown in Figure 3.16. Choose a script from this list and click the **Import** button.

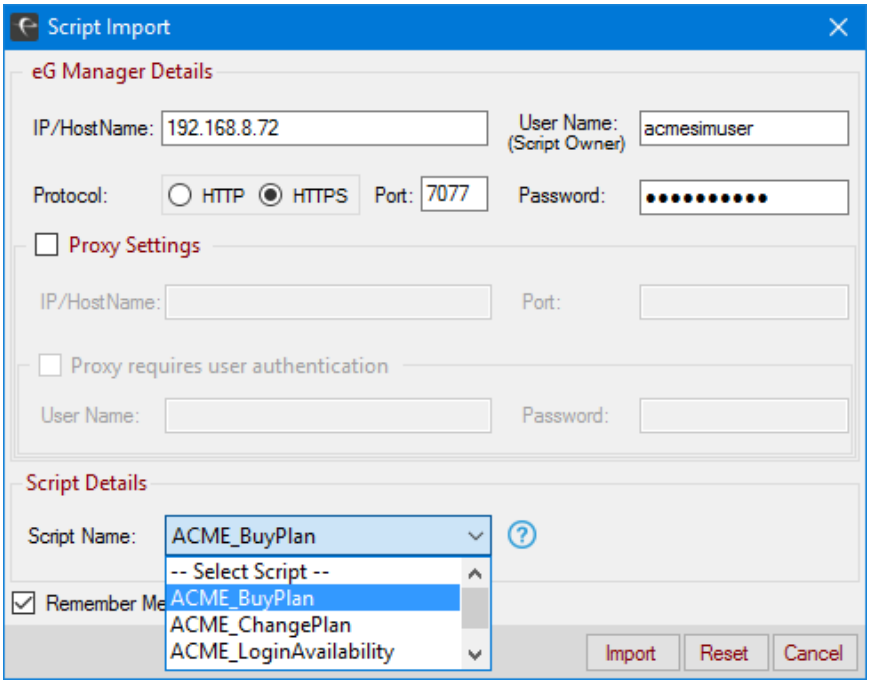

Figure 3.16: Choosing the script that you wish to import from the eG manager

- If you wish to abort the operation click the **Cancel** button.
- If you wish to reset the credentials that you have entered, then, click the **Reset** button.

24. Once the import is successful, the script will automatically open in the Web App Simulation Recorder.
25. In case you have recorded only a partial script and wish to save the script in the local machine, then, you can click the **Save As** option available in the **File** menu of the Web App Simulation Recorder.

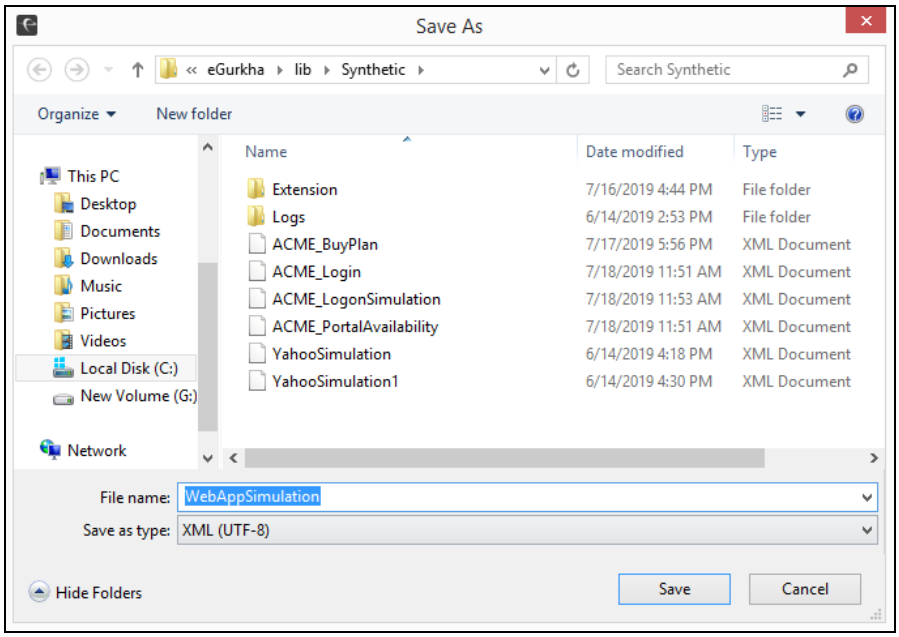

Figure 3.17: Saving the script

26. Figure 3.17 will then appear prompting you to locally save the script. By default, you will be prompted to save the script with a default **File name**. You can either carry on with that default file name or specify a new file name for the script. By default, the script will be saved in .xml format.

3.2 Configuring the Web App Simulator to Perform the Simulation

Once the pre-requisites mentioned in Section **2.1** are fulfilled and a script is recorded and exported to the eG manager, follow the steps detailed below to get the simulator up and running.

1. Log into the eG administrative interface.
2. Add a dedicated external agent for the purpose of the simulation. For that, follow the *Agents -> External Agents* menu sequence and click the **Add New Agent** button. Then, specify the IP address/host name of the system that is hosting the dedicated external agent, and also provide a **Nick name** for the agent (see Figure 3.18).

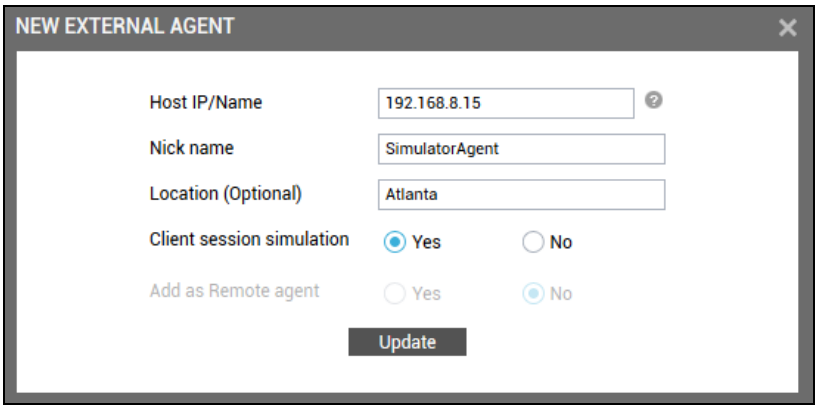

Figure 3.18: Adding a dedicated external agent for the simulation

3. Also, make sure that the **Client session simulation** flag is set to **Yes** for the agent.
4. Finally, click the **Update** button in Figure 3.18 to save the changes.
5. Once this external agent is started, it simulates the entire simulation process by periodically running a **Simulated Web Transactions** test. It is this test that serves as the **Web App Simulator**. Since this test is mapped to a Web App Simulation component, you now need to manage a component of that type. For this, follow the *Infrastructure -> Components -> Add/Modify* menu sequence, and then pick **Web App Simulation** from the list of **Component types**. Then, click **Add New Component**. When Figure 3.19 appears, add a Web App Simulation component using any nick name you want.

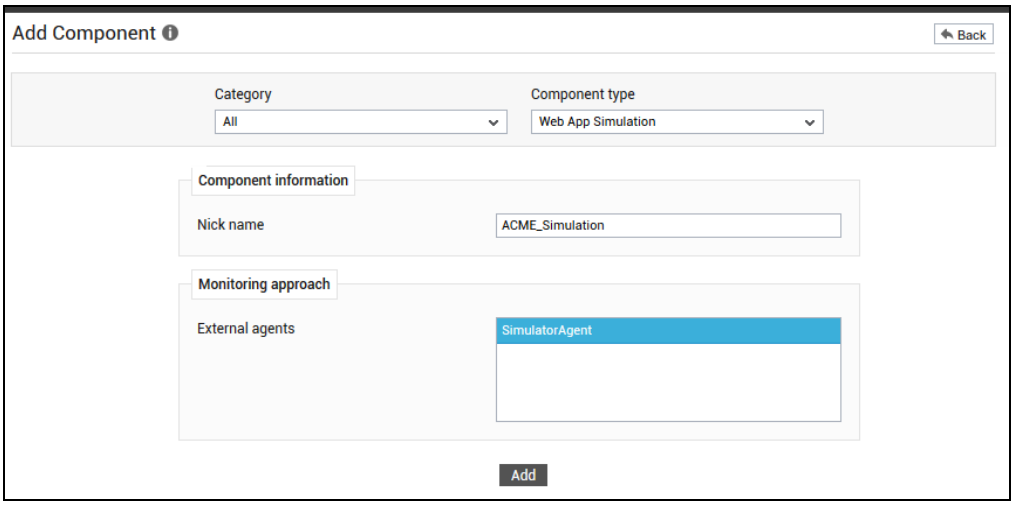

Figure 3.19: Adding a Web App Simulation component

6. When adding, make sure you assign the dedicated external agent, which you had previously installed and configured for the sole purpose of this simulation, to the Web App Simulation component.
7. After clicking the **Add** button in Figure 3.19, proceed to sign out of the eG administrative interface. You will then be prompted to configure the **Simulated Web Transactions** test for this component. Click on the test to configure it.
8. Figure 3.20 will then appear.

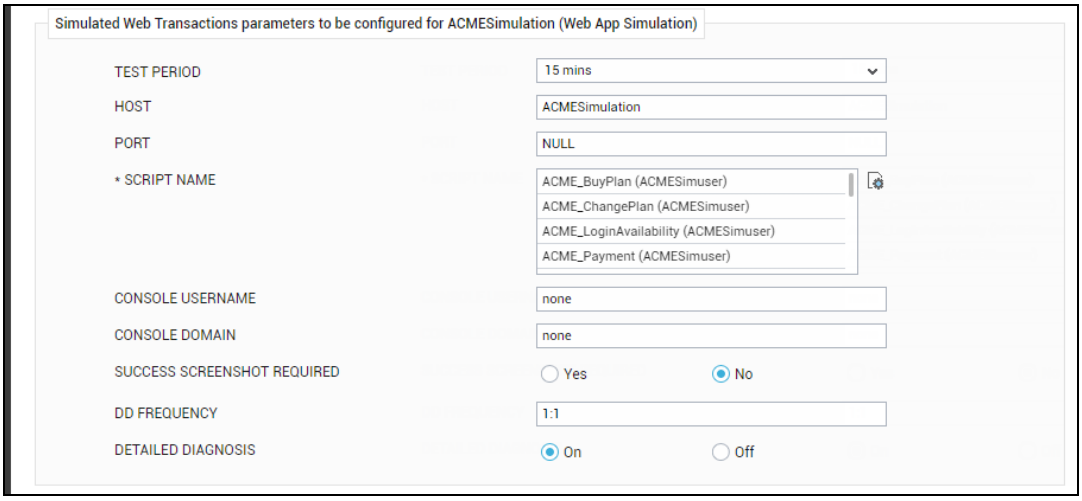

Figure 3.20: Configuring the Simulated Web Transactions test

9. To know how to configure the test, refer to Section **4.1** .
10. Once all parameters are configured, click the **Update** button to save the configuration.

Chapter 4: Analyzing the Simulation Results

Once the simulation ends, the simulator - i.e., the Simulated Web Transactions test - sends the availability and duration measures it collects to the eG manager. Using a specialized **Web App Simulation** monitoring model, the eG manager captures these metrics and publishes them in the eG monitoring console for analysis.

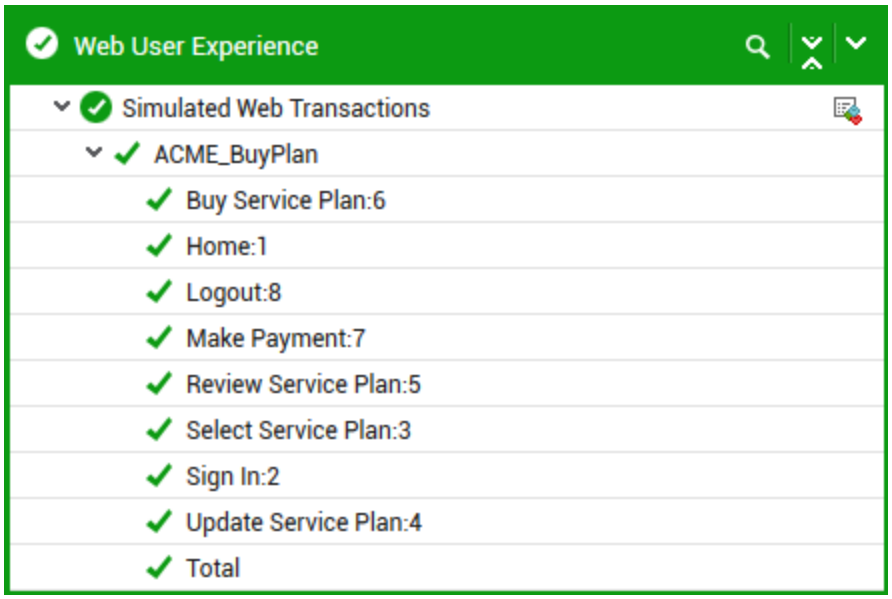

Figure 4.1: The layer model of a Web App Simulation component

As can be inferred from Figure 4.1, this monitoring model consists of a single **Web User Experience** layer, to which the **Simulated Web Transactions** test is mapped. The following section describes how this test works and the measures it reports.

4.1 Simulated Web Transactions Test

This test emulates a user logging into a web application being monitored by the Web App Simulator tool. In the process, the test reports the availability and responsiveness of the web application. The activities that were successful and that failed are measured and reported along with the total simulation time. Additionally, the test also captures failures (if any) at each step of the simulation. Using the insights provided by this test, administrators can proactively detect logon slowness/failures and precisely pinpoint the root-cause of the anomaly. This way, web administrators are enabled to isolate the probable pain-points of their web application, even before users begin to actively use the web application.

Target of the test : A web site or a web application

Agent deploying the test : An external agent

Outputs of the test : One set of results for every transaction in each recorded script

Configurable parameters for the test

Parameters	Description
Test period	How often should the test be executed. The default is 5 minutes. The default is 15 minutes.
Host	The host for which the test is to be configured
Port	Refers to the port used by the target web application
SCRIPT NAME	By default, the scripts that are available for simulation will be listed in this list box. You can select one/more scripts from this list for monitoring. eG Enterprise is also capable of monitoring the scripts that are recorded and exported by more than one user using a single component. For this purpose, a separate icon is provided against this parameter. Clicking the icon will lead you to Figure 4.2.
XML Location	Specify the exact path to the script that has been recorded. By default, the script should be placed in **<eG_INSTALL_DIR>\lib\Synthetic** folder. For example, if a script named *ACME_AppSimulation* has been recorded, then you should specify the location as **<eG_INSTALL_DIR>\lib\Synthetic\ACME_AppSimulation.xml**.

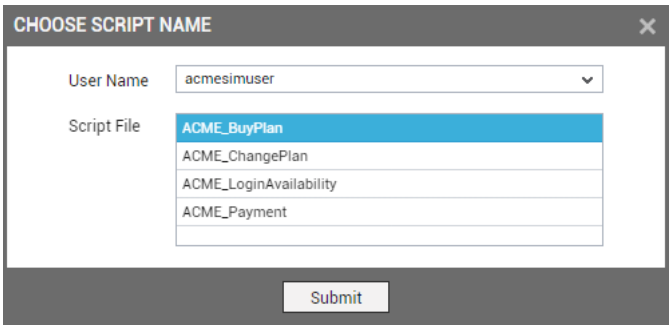

Figure 4.2: Choosing the Script file

In Figure 4.2, select the user who has recorded one/more scripts to be monitored from the **User Name** list.

The scripts corresponding to that user will be populated in the **Script File** list box. Select the script and click the **Submit** button. You are allowed to choose more than one *User Name:Script File* combination.

You can even specify a comma-separated list of scripts for monitoring.

Parameters	Description
Console Username	The simulator needs to run in the account of a user who has local administrator rights on the simulation end point - i.e., the system on which the external agent is installed. Specify the name of this user here. This user should also be logged in at all times for the simulator to run continuously.
Console Domain	Specify the domain to which the Console Username belongs to. By default, *none* is specified against this parameter.
Success Screenshot Required	By default, this flag is set to **No**. If this flag is set to **Yes**, then , if all the transactions for the script were completed successfully, an additional screenshot will be offered for the *Transaction availability* measure of the *Total* descriptor.
DD Frequency	Refers to the frequency with which detailed diagnosis measures are to be generated for this test. The default is *1:1*. This indicates that, by default, detailed measures will be generated every time this test runs, and also every time the test detects a problem. You can modify this frequency, if you so desire. Also, if you intend to disable the detailed diagnosis capability for this test, you can do so by specifying *none* against DD frequency.
Detailed Diagnosis	To make diagnosis more efficient and accurate, the eG Enterprise suite embeds an optional detailed diagnostic capability. With this capability, the eG agents can be configured to run detailed, more elaborate tests as and when specific problems are detected. To enable the detailed diagnosis capability of this test for a particular server, choose the **On** option. To disable the capability, click on the **Off** option. The option to selectively enable/disable the detailed diagnosis capability will be available only if the following conditions are fulfilled: • The eG manager license should allow the detailed diagnosis capability • Both the normal and abnormal frequencies configured for the detailed diagnosis measures should not be 0.

Measurements made by the test

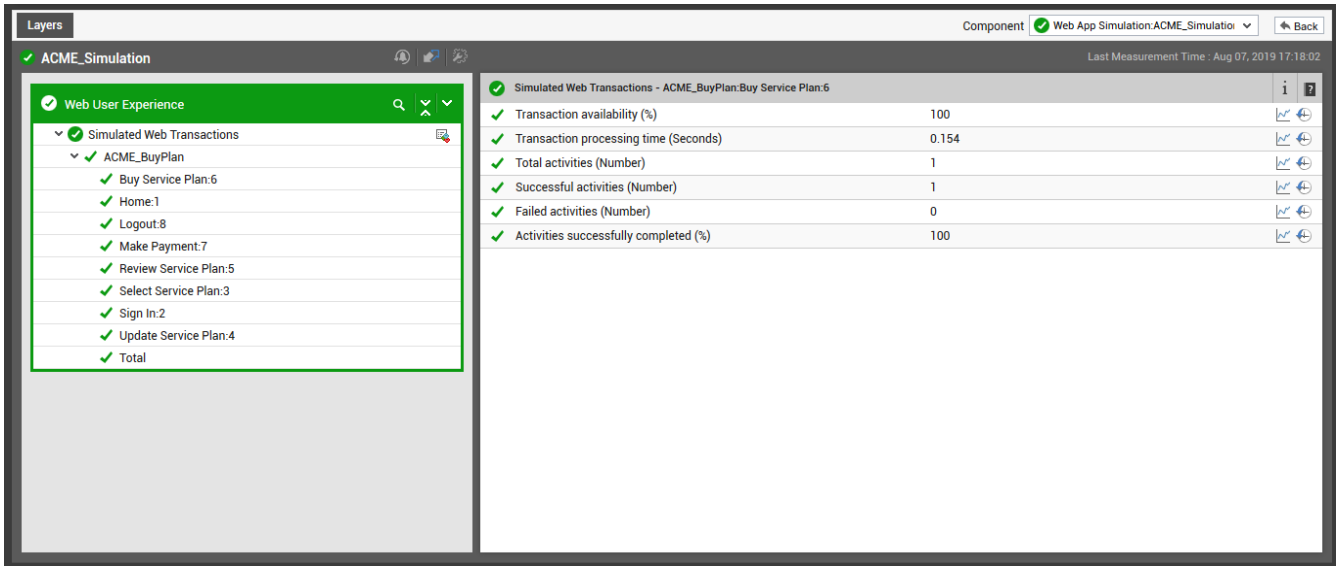

Figure 4.3: The measures reported by the Simulated Web Transactions test

Measurement	Description	Measurement Unit	Interpretation
Transaction availability	Indicates whether/not this transaction is completed successfully.	Percent	The value 100 for this measure indicates that the transaction was completed successfully, and the value 0 indicates that transaction failed. If this measure reports the value 0, then no other measures will be reported for that transaction. You can also use the detailed diagnosis of this measure to view the output of the simulation script, scrutinize it, and isolate the failure and problem points of the web application at first glance.
Transaction processing time	Indicates the time taken by this transaction for execution.	Secs	Compare the value of this measure across the transactions to figure out the transaction that is taking too much time for execution.
Total activities	Indicates the total number of activities performed by	Number	

Measurement	Description	Measurement Unit	Interpretation
	this transaction.		
Successful activities	Indicates the number of activities that were successfully performed by this transaction.	Number	Compare the value of this measure across transactions to figure out the transaction for which least number of activities were successful.
Failed activities	Indicates the number of activities that failed for this transaction.	Number	Compare the value of this measure across transactions to figure out the transaction for which the maximum number of activities failed.
Activities successfully completed	Indicates the percentage of activities successfully completed for this transaction.	Percent	The value 100 for this measure indicates that the activities were successfully completed, and the value 0 indicates that the activities failed to complete. By comparing the value of this measure across transactions, you can quickly identify the transactions for which the activities failed to complete successfully.
Total simulation time	Indicates the total time taken to simulate this transaction.	Secs	This measure is applicable only for the *Total* descriptor. An abnormally high value for this measure could indicate a logon slowness.

The detailed diagnosis of the *Transaction availability* measure will appear only when a value of 0 is reported. Clicking the icon will lead you to the screenshot which visually indicates the exact transaction at which the script failed during simulation. Clicking the icon will directly open the screenshot at which the transaction failed.

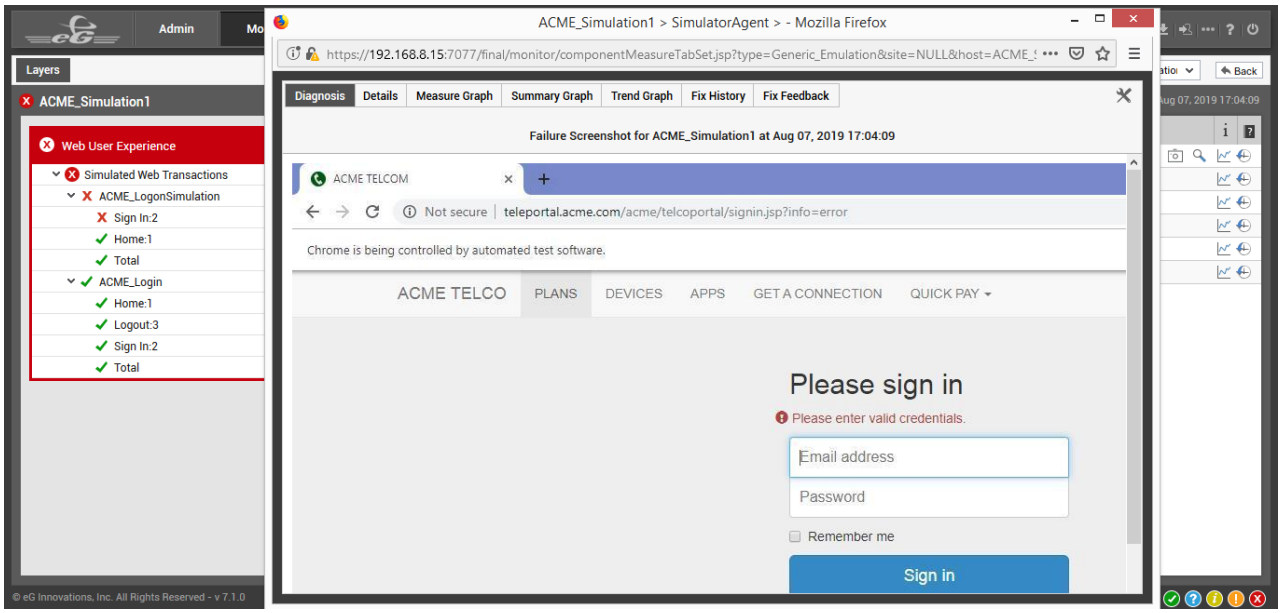

Figure 4.4: The Failure screenshot that was captured

Use the **Details** section of the detailed diagnosis to view the transaction flow, output of the simulation script, scrutinize it, and isolate the failure and problem points of the web application at first look. A summary of the simulation is also provided as part of the detailed diagnostics.

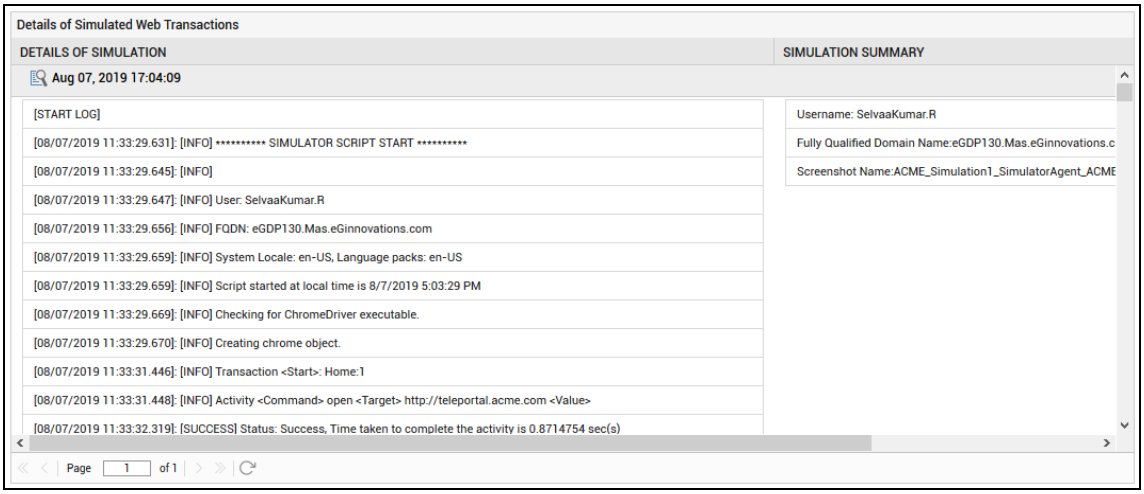

Figure 4.5: The detailed diagnosis of the Transaction availability measure

4.2 Synthetic Monitoring Dashboard

Where two/more Web App Simulation components are managed, clicking on the *Web App Simulation* component-type in the **Components At-A-Glance** section of the Monitor dashboard automatically opens the **Synthetic Monitoring Dashboard**.

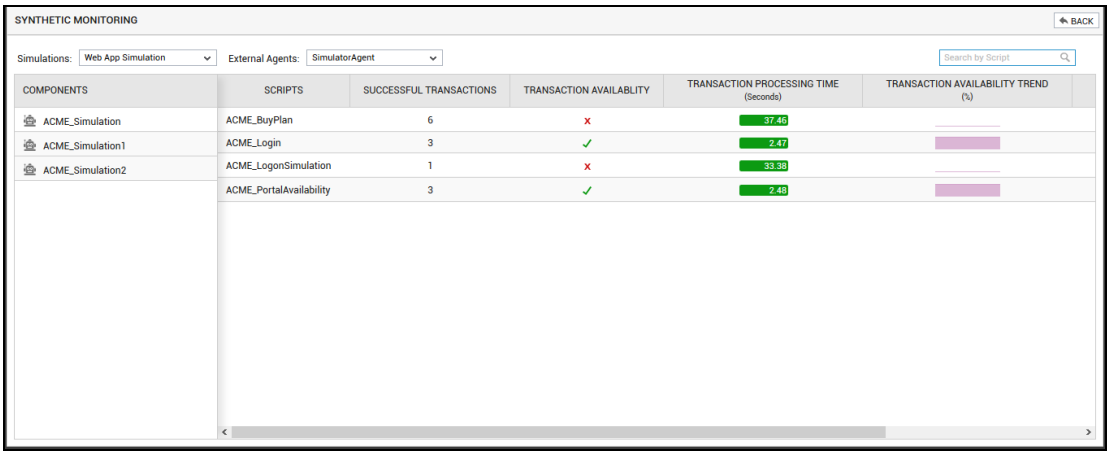

Figure 4.6: The Synthetic Monitoring Dashboard

By default, the dashboard displays all the simulations performed by all the simulators configured in an environment. For each simulation, the dashboard displays the transactions that were successfully simulated, the availability of the transaction, the processing time of the transaction, the transaction availability trend and the processing time trend of each transaction as captured by that simulation. This way, the simulations that failed and the precise failure points (for e.g., availability) of each simulation can be instantly and accurately isolated. Clicking on a simulation in Figure 4.6 will lead you to the transaction flow process as shown in Figure 4.7. Using this graphical representation, administrators can clearly identify the step of the simulation process has failed completely.

Figure 4.7: A graphical view of the transaction

You can click on the icon if failures were detected. This will lead you to the screenshot of the page on which the transaction failed (see Figure 4.8).

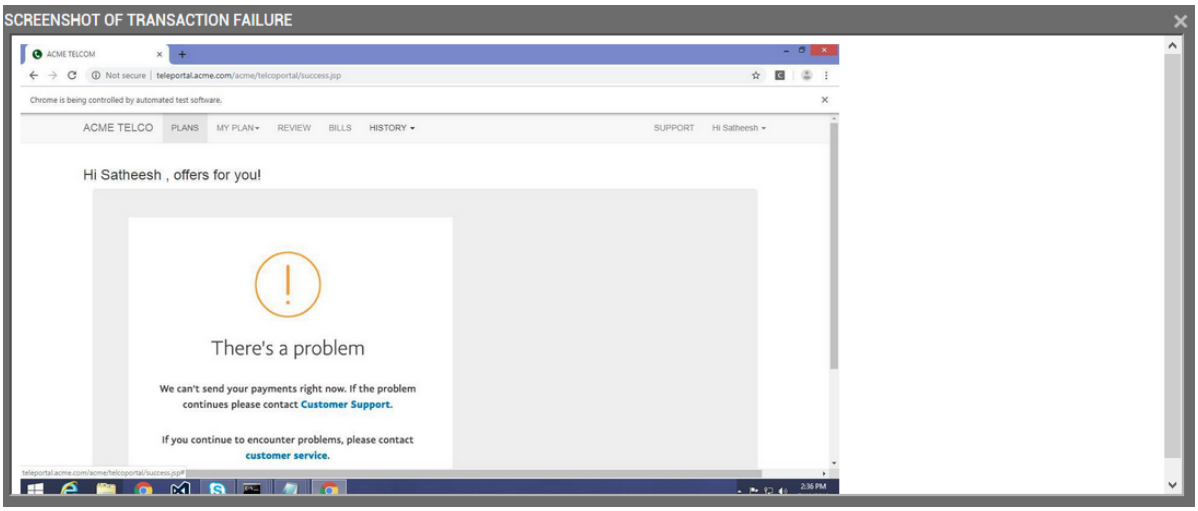

Figure 4.8: The simulation script highlighting the success and failure points of the simulation

Similarly, to view the performance trend of the simulated transaction over a period of time, you can click the icon against the *Transaction processing time* measure (see Figure 4.7). This will lead you to Figure 4.9.

Figure 4.9: Analyzing the performance trend of the simulated transaction

By default, the performance trend of each measure of the simulated transaction is analyzed and graphs are plotted individually. The default, time duration for plotting the graph is 1 hour. Using the graphs that are plotted, administrators can determine how well the simulated transaction has performed in the last 1 hour. If you wish to change time period to analyze the performance trend,

then, you can click the icon. Figure 4.10 will then appear using which you can change the **Timeline** for which the graphs need to be plotted.

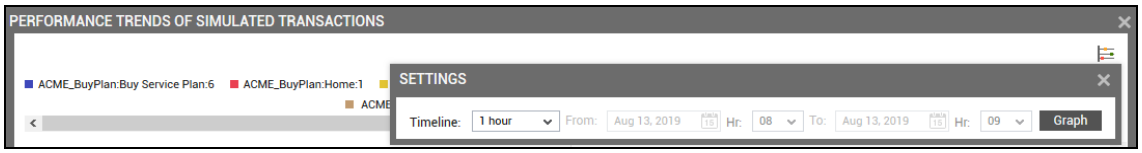

Figure 4.10: Changing the time period for analyzing the simulated transaction graphs

You can even choose an image for each transaction activity that is recorded in the transaction flow process. For this purpose, click on any activity icon that you wish to change in Figure 4.7. Figure 4.11 will then appear.

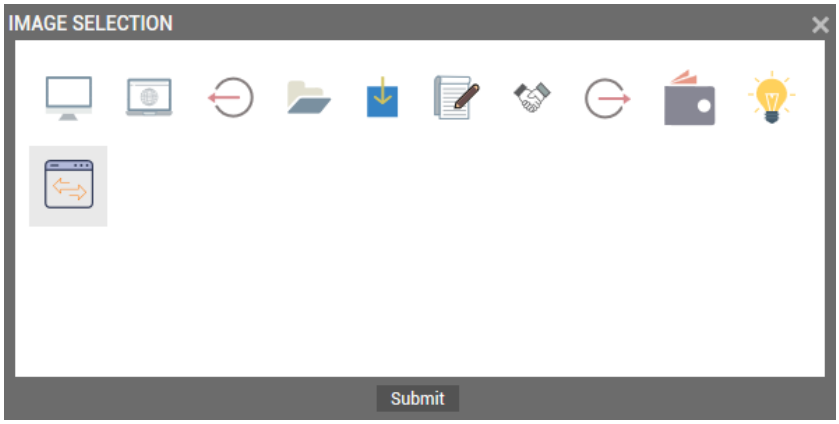

Figure 4.11: Choosing an appropriate image for a transaction

Here, select the icon of your choice and click the **Submit** button.

Clicking on any transaction name or the state of the transaction will lead you to the layer model for the simulation as shown in Figure 4.14.

You can even filter the details displayed in the dashboard by picking the simulator for which you want to view the details. This can be achieved by picking a particular external agent from the **External Agents** drop-down (see Figure 4.12).

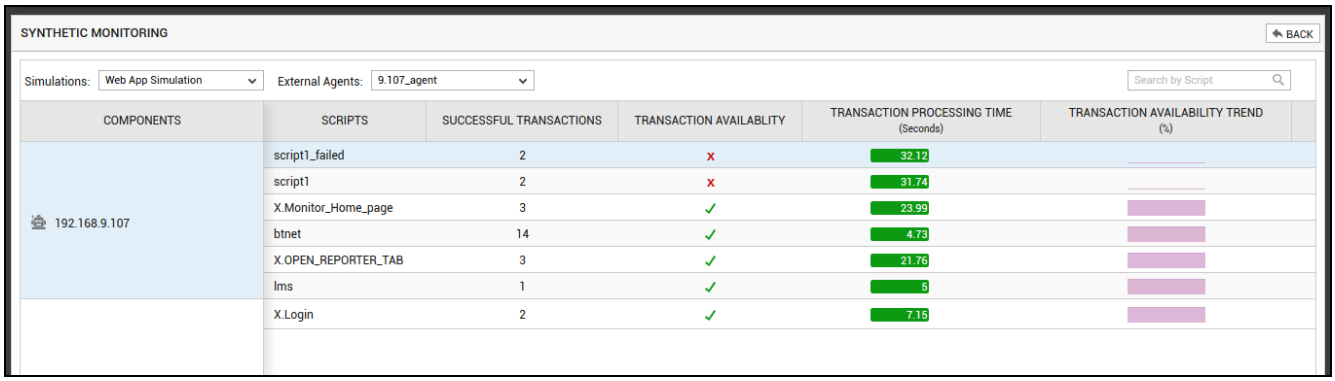

Figure 4.12: Viewing the details of a particular simulation alone

Alternatively, you can filter the dashboard contents on the basis of the scripts that were recorded for the *Web App Simulation* component that you managed. You can specify the whole/part of the script name in the **Search by Script** search text box (see Figure 4.13) and click the icon alongside. This will display the details of only those scripts with names that contain the specified search string.

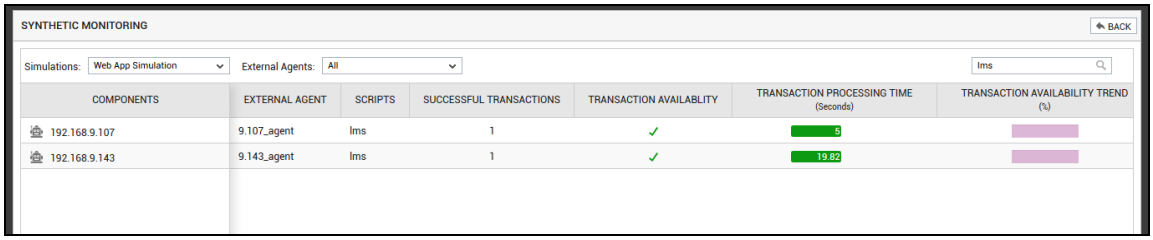

Figure 4.13: Viewing the details of only those scripts that matched the search criteria

Clicking on any value specified against the scripts in the dashboard will lead you to the **Layers** tab page, where you can view the metrics reported by the simulation and the current state of each metric.

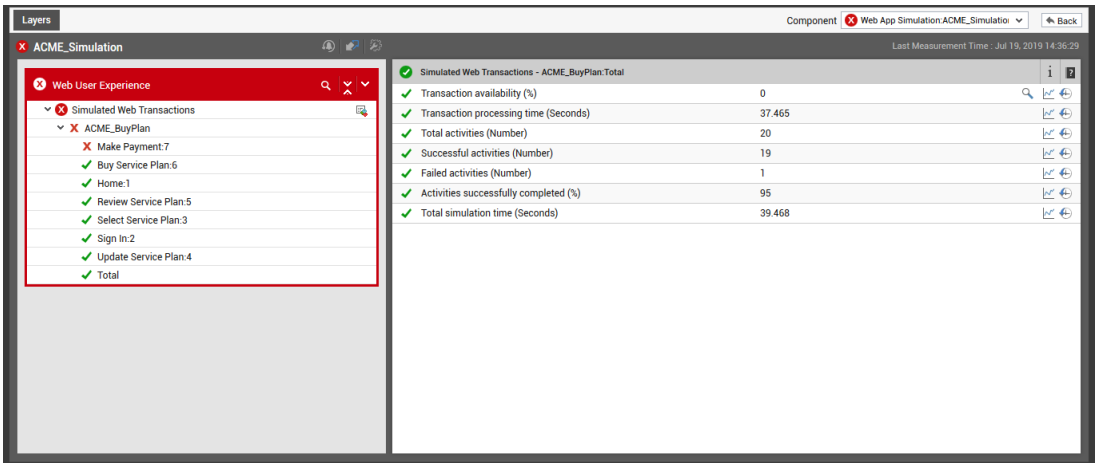

Figure 4.14: The layer model of the Web App Simulation component that was clicked on

Clicking the  icon in the layer model page (see Figure 4.14) will lead you back to the transaction flow graph as shown in Figure 4.7.

For historical analysis of the simulated results, the eG Reporter provides a Web App Simulation Report . This report can be generated for one/all simulations, or for one/all external agents. To know more on how to generate the Web App Simulation report, refer to *Synthetic Monitoring - Web App Simulation Report* of *The eG Reporter* document.

Chapter 5: Fine-tuning the Simulation

One of the key pre-requisites for the simulation is a user account with local administrator rights on the simulation endpoint. This user should also be logged in at all times for the simulator to run continuously. Sometimes however, this user session may get disconnected. For instance, if the simulation endpoint is rebooted due to automatic updates, scheduled reboots, power failure etc., the user session on the simulation endpoint may get disconnected.

Every time a session disconnect occurs owing to reasons cited above, the administrator will have to login to the endpoint by manually providing the user credentials at the login prompt, while the system boots. If this is not done, then the user session will not get up and running; consequently, the simulation will not occur.

To save the time and effort involved in manually typing the login credentials everytime the endpoint reboots, and to make sure that a user is always logged into the endpoint (even when it reboots) for the purpose of the simulation, you can automate a user login at the time of a reboot. To achieve this, you can either run *Autologon.exe* or manually *edit the windows registry*.

5.1 Fine-tuning the simulation using Autologon.exe

If you wish to automate the user logon by executing Autologon.exe, follow the steps below:

1. Download the **Autologon.zip** file from the **Download Autologon** link from the following location:

   https://docs.microsoft.com/en-us/sysinternals/downloads/autologon

2. Extract the contents of the **Autologon.zip** file.
3. Once extracted, run the **Autologon.exe** file.

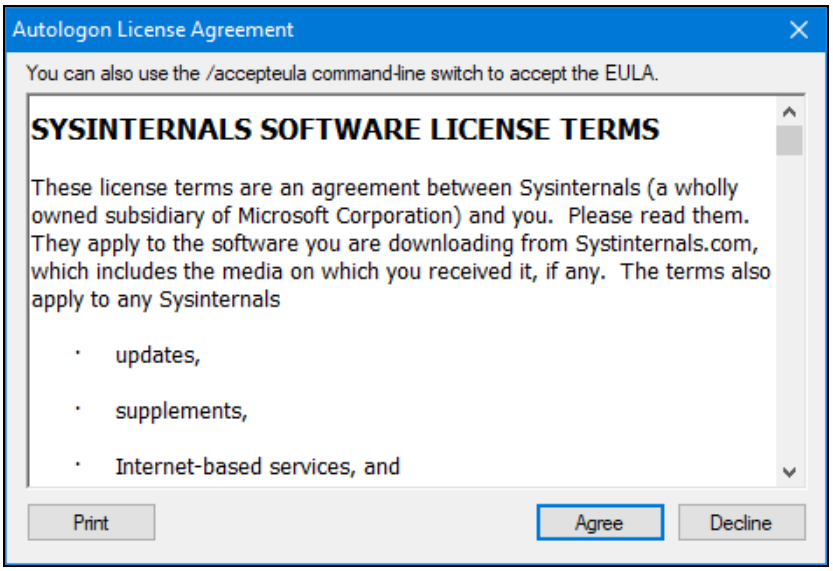

Figure 5.1: Agreeing to the Software License Terms

4. Figure 5.1 then appears. Click **Agree** to accept the Sysinternals Software License Terms.

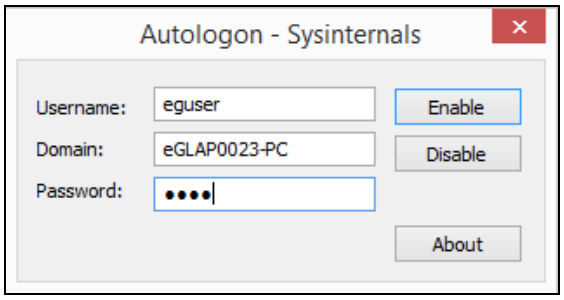

Figure 5.2: Provide the password in this form

5. In Figure 5.2 that appears next, the name of the user and the domain to which the user belongs will be automatically populated against the **Username** and **Domain** fields. Specify the password that should be used for automatic user logon against the **Password** text box.
6. Click the **Enable** button.
7. Ensure that the **eGurkhaAgentServices** are delayed for a period of 5 minutes (using Automatic (Delayed Start) Service properties) before restarting the simulation endpoint.
8. Finally, restart the simulation endpoint.

5.2 Fine-tuning the simulation by editing the windows registry

If you wish to automate the user login by editing the windows registry, follow the steps below:

1. Open the Windows Registry Editor.
2. Locate the following registry entry:

   HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\Current Version\Winlogon
3. In this registry entry, add the following REG_SZ string values:
   - **AutoAdminLogon:** To enable automatic user logon on the simulation endpoint, set this string value to 1.
   - **DefaultUserName:** Specify the name of the user who is authorized to login into the simulation endpoint.
   - **DefaultPassword:** Specify the password for the user mentioned in the DefaultUserName. **Note that the password should be entered in plain text.**
   - **DefaultDomainName:** Specify the domain to which the user belongs to.
4. Ensure that the **eGurkhaAgentServices** are delayed for a period of 5 minutes (using Automatic (Delayed Start) Service properties) before restarting the simulation endpoint.
5. Finally, restart the simulation endpoint.

Chapter 6: Troubleshooting the Web App Simulator

Sometimes, the simulator may not be able to launch the web application automatically. Some of the problems that are caused while launching the web application, and the means to resolve them are outlined here.

6.1 Browser launch hindered due to disabled chrome extensions

In highly secure environments, administrators may not want to load the chrome extensions on the Chrome browser for all users. In such cases, a group policy may be applied to disable these chrome extensions from loading on the Chrome browser. If simulation happens in such environments, the Chrome browser may not be launched and an error message as shown in Figure 6.1 appears.

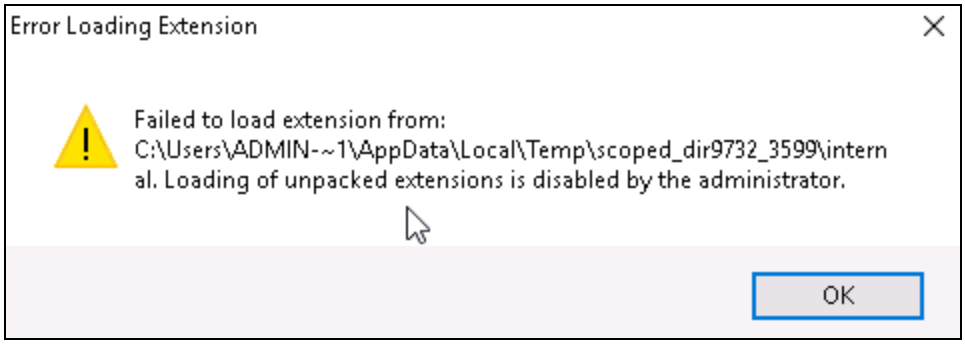

Figure 6.1: Error message that appears when chrome extensions failed to load

For the Web App Simulator to launch the Chrome browser by overriding the group policy settings that disabled the extensions, do the following:

1. Open the Windows Registry Editor.
2. Locate the following registry entry:

   HKLM\Software\Policies\Google\Chrome\ExtensionInstallBlacklist

   In this registry entry, delete all keys and values.

3. Locate the following registry entry:

   HKCU\Software\Policies\Google\Chrome\ExtensionInstallBlacklist

In this registry entry, delete all keys and values.

4. Finally, restart the eG agent.

Ensure that the group policy is disabled on the simulation endpoint so that the Chrome browser can be launched by the Web App Simulator at periodic intervals.

About eG Innovations

eG Innovations provides intelligent performance management solutions that automate and dramatically accelerate the discovery, diagnosis, and resolution of IT performance issues in on-premises, cloud and hybrid environments. Where traditional monitoring tools often fail to provide insight into the performance drivers of business services and user experience, eG Innovations provides total performance visibility across every layer and every tier of the IT infrastructure that supports the business service chain. From desktops to applications, from servers to network and storage, from virtualization to cloud, eG Innovations helps companies proactively discover, instantly diagnose, and rapidly resolve even the most challenging performance and user experience issues.

eG Innovations is dedicated to helping businesses across the globe transform IT service delivery into a competitive advantage and a center for productivity, growth and profit. Many of the world's largest businesses use eG Enterprise to enhance IT service performance, increase operational efficiency, ensure IT effectiveness and deliver on the ROI promise of transformational IT investments across physical, virtual and cloud environments.

To learn more visit www.eginnovations.com.

Contact Us

For support queries, email support@eginnovations.com.

To contact eG Innovations sales team, email sales@eginnovations.com.