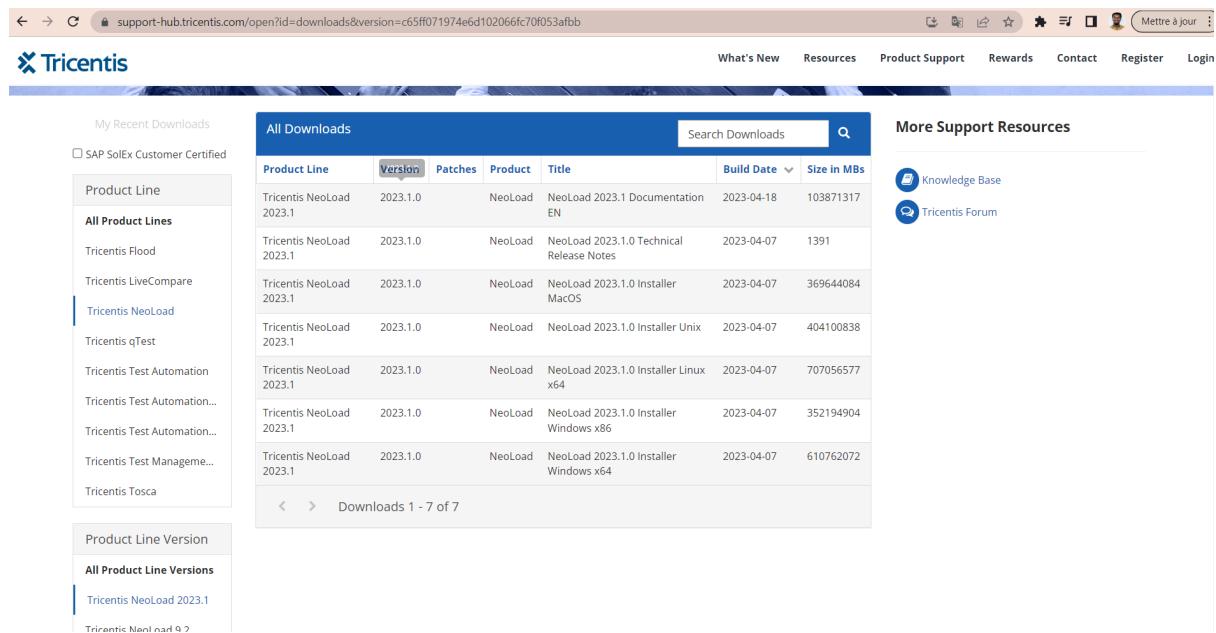


Partie 1 : Téléchargement et Installation NeoLoad

Lien de téléchargement :

<https://support-hub.tricentis.com/open?id=downloads&version=c65ff071974e6d102066fc70f053afbb>



My Recent Downloads

SAP SolEx Customer Certified

Product Line

- All Product Lines
- Tricentis Flood
- Tricentis LiveCompare
- Tricentis NeoLoad**
- Tricentis qTest
- Tricentis Test Automation
- Tricentis Test Automation...
- Tricentis Test Automation...
- Tricentis Test Manageme...
- Tricentis Tosca

Product Line Version

All Product Line Versions

Tricentis NeoLoad 2023.1

Tricentis NeoLoad 9.2

All Downloads

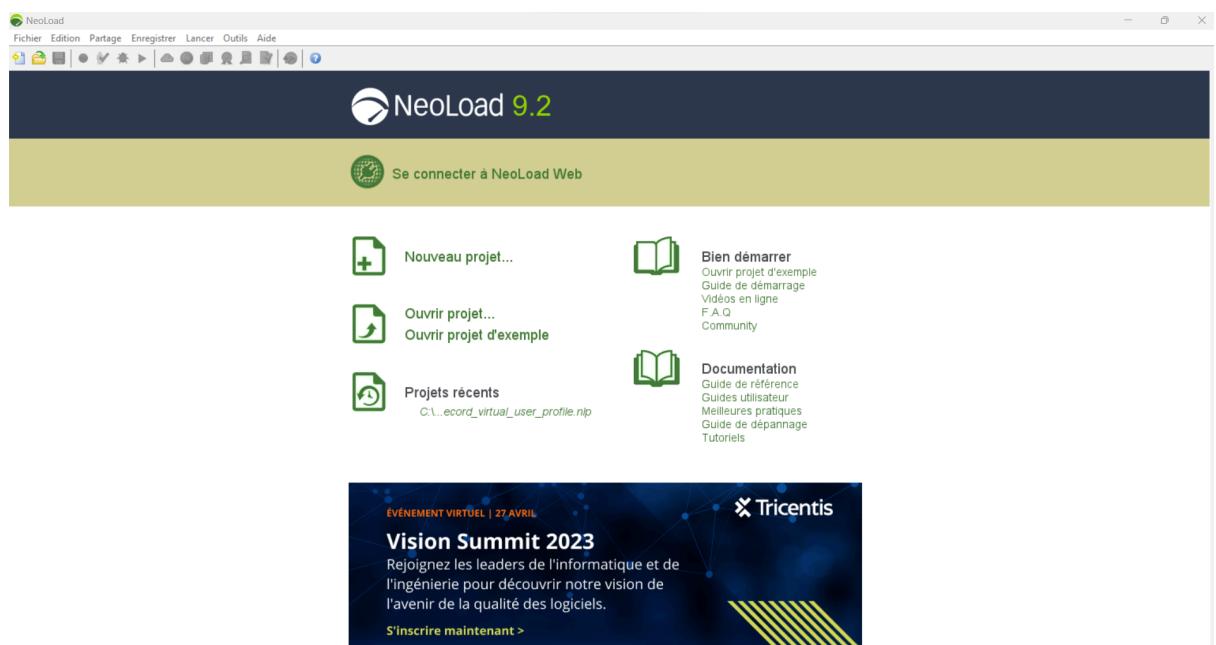
Product Line	Version	Patches	Product	Title	Build Date	Size in MBs
Tricentis NeoLoad 2023.1	2023.1.0		NeoLoad	NeoLoad 2023.1 Documentation EN	2023-04-18	103871317
Tricentis NeoLoad 2023.1	2023.1.0		NeoLoad	NeoLoad 2023.1.0 Technical Release Notes	2023-04-07	1391
Tricentis NeoLoad 2023.1	2023.1.0		NeoLoad	NeoLoad 2023.1.0 Installer Mac OS	2023-04-07	369644084
Tricentis NeoLoad 2023.1	2023.1.0		NeoLoad	NeoLoad 2023.1.0 Installer Unix	2023-04-07	404100838
Tricentis NeoLoad 2023.1	2023.1.0		NeoLoad	NeoLoad 2023.1.0 Installer Linux x64	2023-04-07	707056577
Tricentis NeoLoad 2023.1	2023.1.0		NeoLoad	NeoLoad 2023.1.0 Installer Windows x86	2023-04-07	352194904
Tricentis NeoLoad 2023.1	2023.1.0		NeoLoad	NeoLoad 2023.1.0 Installer Windows x64	2023-04-07	610762072

Search Downloads

What's New Resources Product Support Rewards Contact Register Login

More Support Resources

- Knowledge Base
- Tricentis Forum



NeoLoad

Fichier Edition Partage Enregistrer Lancer Outils Aide

NeoLoad 9.2

Se connecter à NeoLoad Web

Nouveau projet...
Ouvrir projet...
Ouvrir projet d'exemple
Projets récents
C:\...ecord_virtual_user_profile.nlp

Bien démarrer
Ouvrir projet d'exemple
Guide de démarrage
Vidéos en ligne
F.A.Q
Community

Documentation
Guide de référence
Guides utilisateur
Meilleures pratiques
Guide de dépannage
Tutoriels

EVÉNEMENT VIRTUEL | 27 AVRIL

Vision Summit 2023

Rejoignez les leaders de l'informatique et de l'ingénierie pour découvrir notre vision de l'avenir de la qualité des logiciels.

S'inscrire maintenant >

Tricentis

Outil NeoLoad

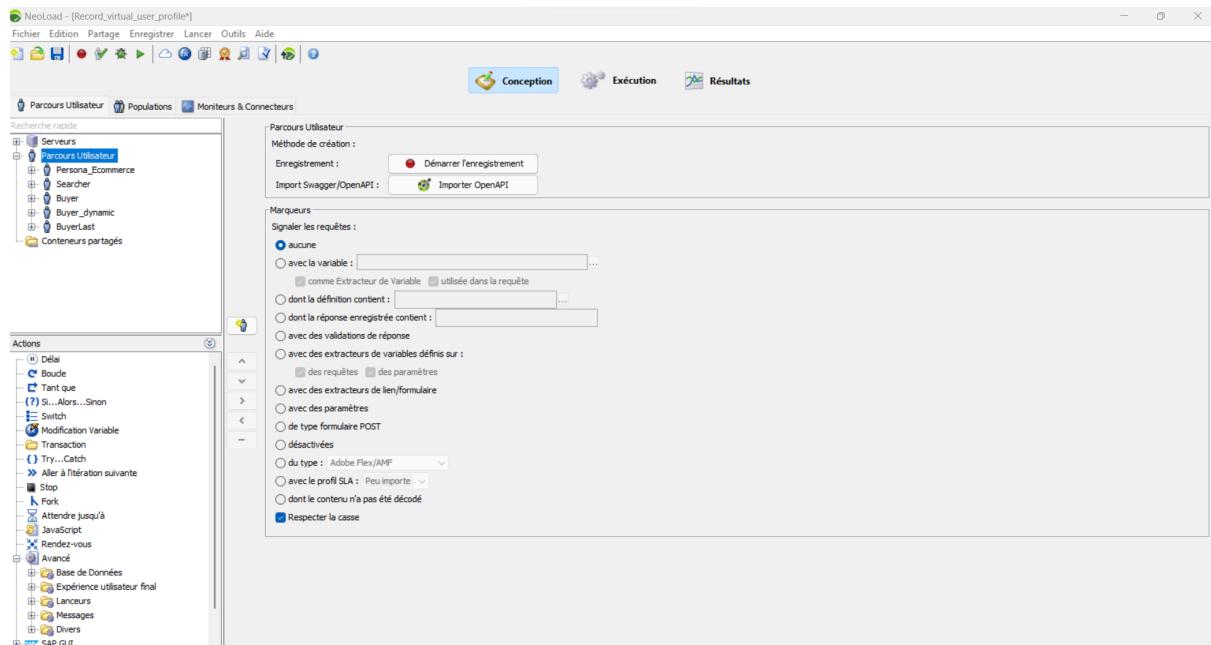
Partie 2 : Neoload Overview

Les différentes parties de Neoload :

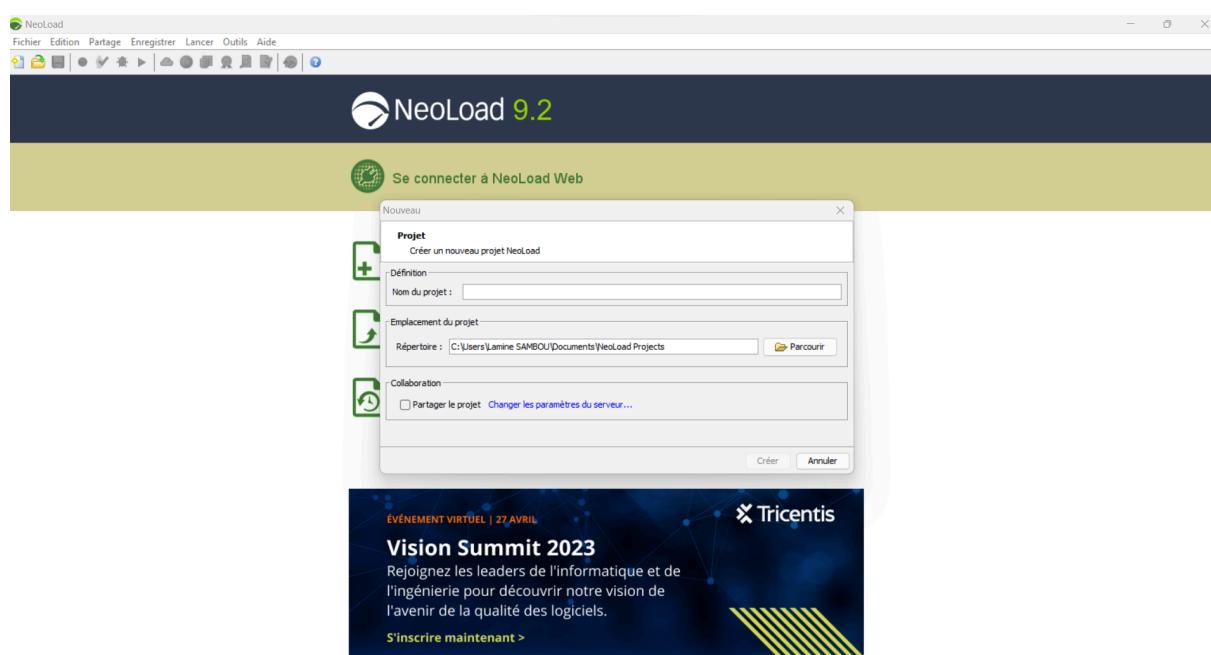
Design : Cette partie permet d'écrire des scripts, des cas d'utilisations, des conditions logiques, etc

Runtime : Cette partie permet de configurer et de lancer l'exécution des scénarios

Result : Cette partie permet de visualiser, d'analyser et de faire un reporting des résultats

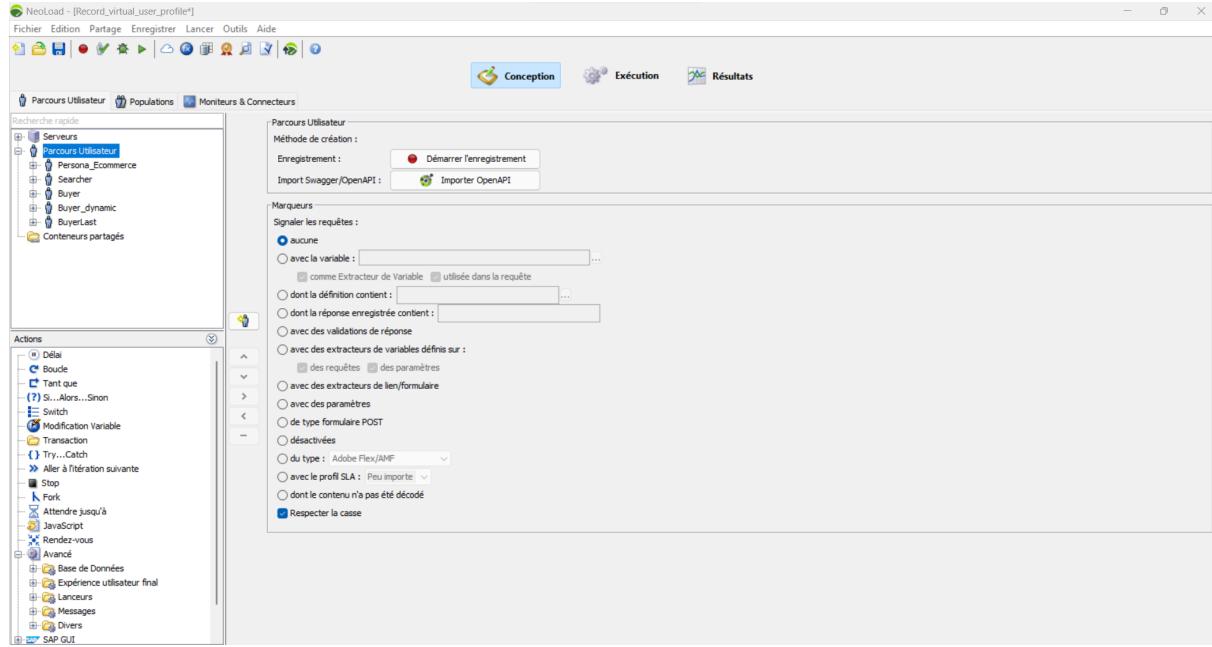


Partie 3 : Création de projet



Partie 4 : Design module

Cette partie permet d'écrire des scripts, des cas d'utilisations, des conditions logiques, des corrélations, des paramétrisations, etc.



Partie 5 : What is workflow

Un workflow est une série d'action effectuée par un utilisateur dans un site ou une application web. Un workflow est donc l'enregistrement d'un parcours utilisateur. Dans neoload on l'appelle **Parcours Utilisateur**.

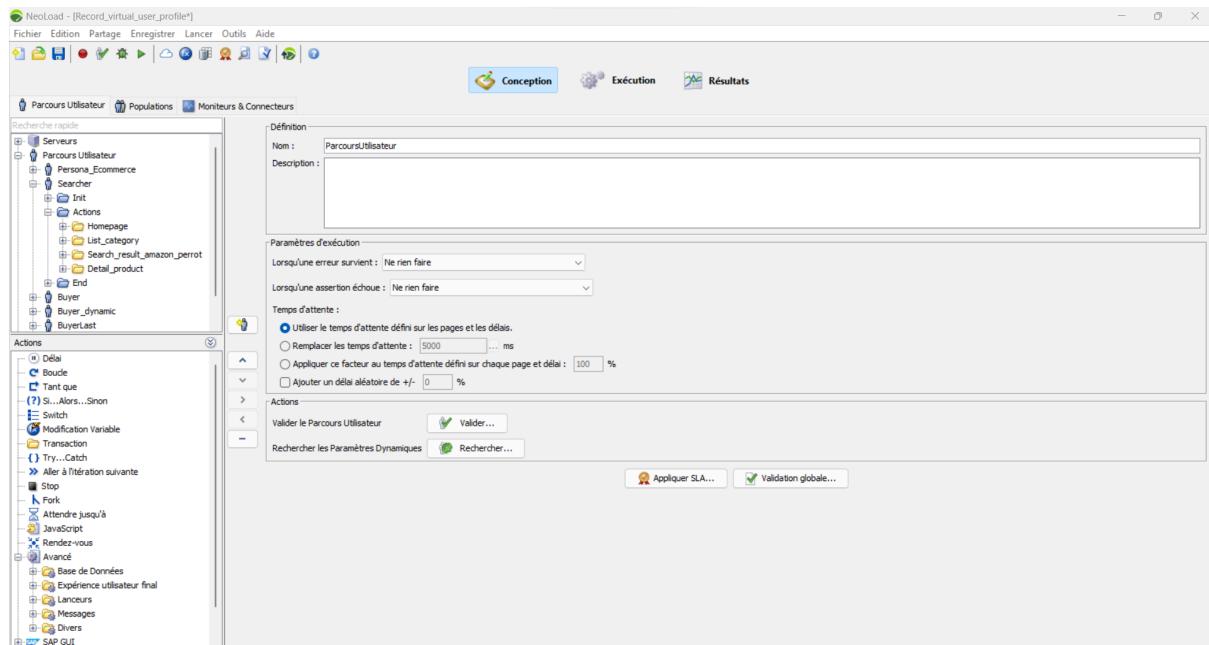
Partie 6 : Record a workflow

Lorsqu'on enregistre un parcours utilisateur via un navigateur web, on ajoute d'abord le nom de la transaction et ensuite on clique ou bien on navigue vers le lien correspondant au nom de la transaction donnée.

L'ensemble des noms de transactions vont constituer un workflow.

Partie 7 : Exploring the user path

A la fin de l'enregistrement de notre workflow, toutes les actions effectuées dans l'application web ou le site sont répertoriées dans le dossier Action, Init, End.



Partie 8 : User Path hierarchy

Dans un parcours utilisateur, nous avons 3 dossiers : **Init, Action, End**

- Init et End peuvent être utilisés pour enregistrer les transactions de logIn et de logOut.
- Dans la partie Action, nous avons d'abord le nom de la transaction ensuite la page enregistrée et enfin toutes les ressources de cette page (image, fichier, session, variables, etc)

Partie 9 : Check User path validity

Après l'enregistrement du parcours utilisateur, il est nécessaire de relancer le parcours dans neoload pour pouvoir détecter les potentielles erreurs et les variables dynamiques, les sessions ID, les login, les mots de passes, etc.

The screenshot shows the 'Valider un Parcours Utilisateur' (Validate User Path) tool. The main window title is 'Vérifier la validité du Parcours Utilisateur'. The left sidebar shows a tree view of the user path 'Searcher' with various actions like 'Init', 'Actions', 'Homepage', 'List_category', 'Search_result_amazon_perrot', and 'Detail_product'. The main area displays a table of validation results:

Date	Action	Code de réponse	Assertions	Déférence
00:00:00.224	Searcher			
00:00:00.229	Init			
00:00:00.235	Actions			
00:00:00.235	Homepage			
00:00:00.243	/actions/Catalog.action			
00:00:00.895	/actions/Catalog.action	200	4 %	
00:00:00.930	List_category			
00:00:00.930	/actions/Catalog.action			

Below the table, there are tabs for 'Détails', 'Variables', 'Serveurs', 'Rendu HTML', 'Rendu HTML enregistré', and 'Rendu HTML comparé'. The 'Détails' tab is selected, showing the following information:

Page : /actions/Catalog.action
Requête /actions/Catalog.action
Référent :
Date : 00:00:00.895 Code de réponse : 200 Durée de réponse : 0,505 s Taille de réponse : 4 Ko
Détails
 Requête Réponse Assertions

```
HTTP/2.0 200
date: Mon, 24 Apr 2023 09:34:45 GMT
content-type: text/html; charset=UTF-8
content-length: 4503
set-cookie: JSESSIONID=7A4756B27393EC63FD16FD2EEFE01519; Path=/; HttpOnly
content-language: en
strict-transport-security: max-age=15724800; includeSubDomains
```

At the bottom right of the main window is a 'Fermer' (Close) button.

La colonne **Action** représente le nom de la transaction ou de la requête

La colonne **Code de réponse** représente le statut de la réponse (200, 404, etc)

La colonne **Assertion** : ?

La colonne **Déférence** représente la différence qui existe entre la réponse du serveur lors de l'enregistrement et la réponse du serveur lors de la validation du check user path

Partie 10 : Google chrome developer Tools

Tous les éléments figurant dans la partie **Network** de **Google Chrome** (qui affiche l'ensemble des ressources d'une page web (image, fichiers, css, html, etc) doit se retrouver dans une des **transactions enregistrées dans Neoload**.

Name	Status	Type	Initiator	Size	Time
Catalog.action/viewCat...	200	docu...	Other	3.9 kB	39 ms
petstore.css	200	styles...	Catalog.act...	(mem...)	0 ms
logo-topbar.gif	200	gif	Catalog.act...	(mem...)	0 ms
cart.gif	200	gif	Catalog.act...	(mem...)	0 ms
separator.gif	200	gif	Catalog.act...	(mem...)	0 ms
sm_fish.gif	200	gif	Catalog.act...	(mem...)	0 ms
sm_dogs.gif	200	gif	Catalog.act...	(mem...)	0 ms
sm_reptiles.gif	200	gif	Catalog.act...	(mem...)	0 ms
sm_cats.gif	200	gif	Catalog.act...	(mem...)	0 ms
sm_birds.gif	200	gif	Catalog.act...	(mem...)	0 ms

Partie 11 : Understanding response code

- 100 : informations
- 200 : success
- 300 : redirection
- 400 : client side error
- 500 : server side error

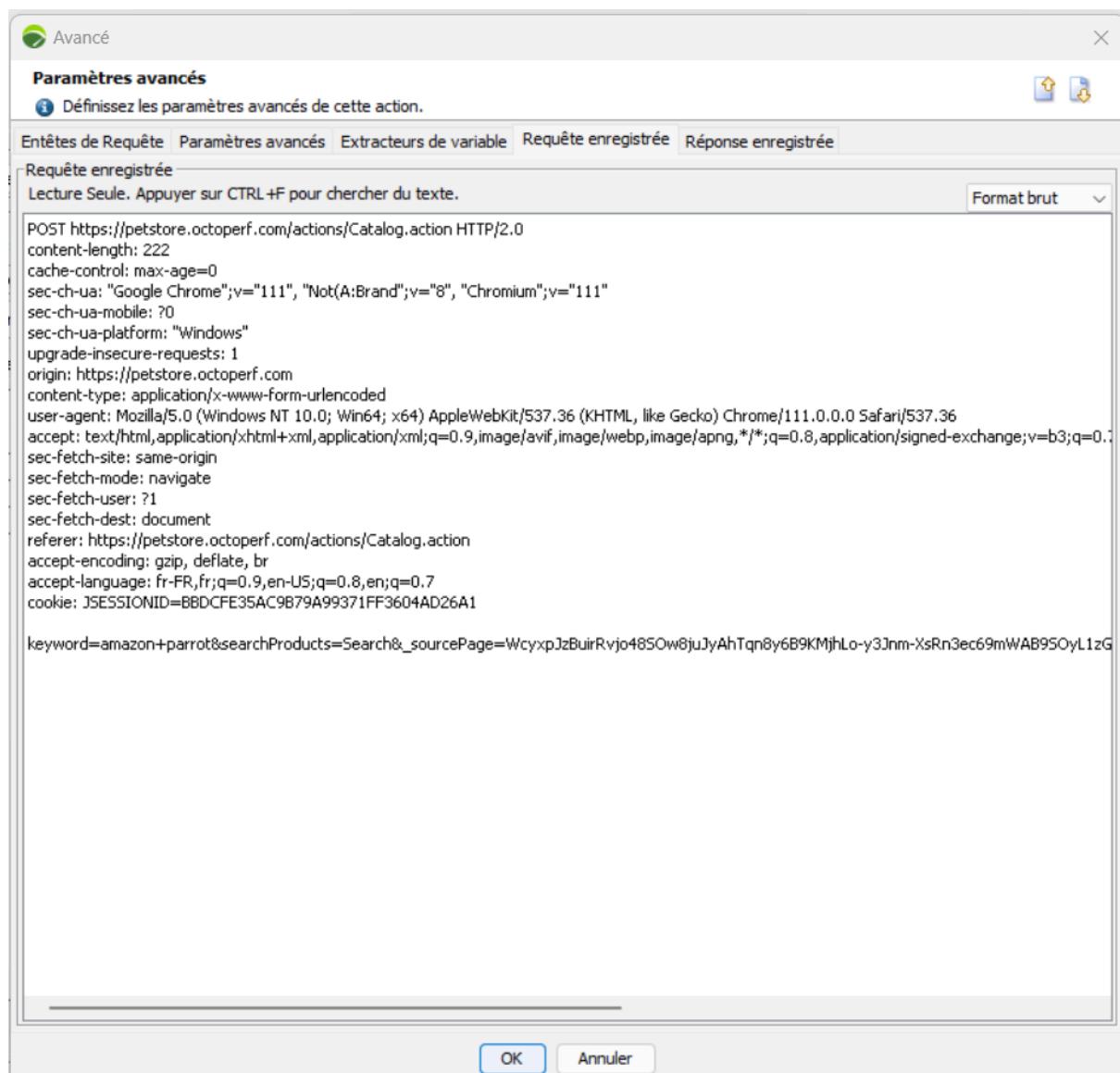
Partie 12 : Components of request & response

Un request component contient :

- une méthode HTTP (POST, GET, DELETE, PUT, PATCH, etc)
- un URL
- un header (content-type : json, xml, etc)
- un body qui est optionnel

Une réponse component contient :

- un code de réponse ou statut (200, 404, 500, etc)
- un header (content-type : json, xml, etc)
- un body qui est optionnel



Request figure

HTTP/2.0 200
date: Mon, 03 Apr 2023 20:53:43 GMT
content-type: text/html; charset=UTF-8
content-length: 3639
content-language: en
strict-transport-security: max-age=15724800; includeSubDomains

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head>
<link rel="StyleSheet" href="..,css/jpetstore.css" type="text/css"
media="screen" />

<meta name="generator"
content="HTML Tidy for Linux/x86 (vers 1st November 2002), see www.w3.org" />
<title>JPetStore Demo</title>
<meta content="text/html; charset=windows-1252"
http-equiv="Content-Type" />
<meta http-equiv="Cache-Control" content="max-age=0" />
<meta http-equiv="Cache-Control" content="no-cache" />
<meta http-equiv="expires" content="0" />
<meta http-equiv="Expires" content="Tue, 01 Jan 1980 1:00:00 GMT" />
<meta http-equiv="Pragma" content="no-cache" />
</head>

<body>

```

Response figure

Partie 13 : Corrélation (extractor variable)

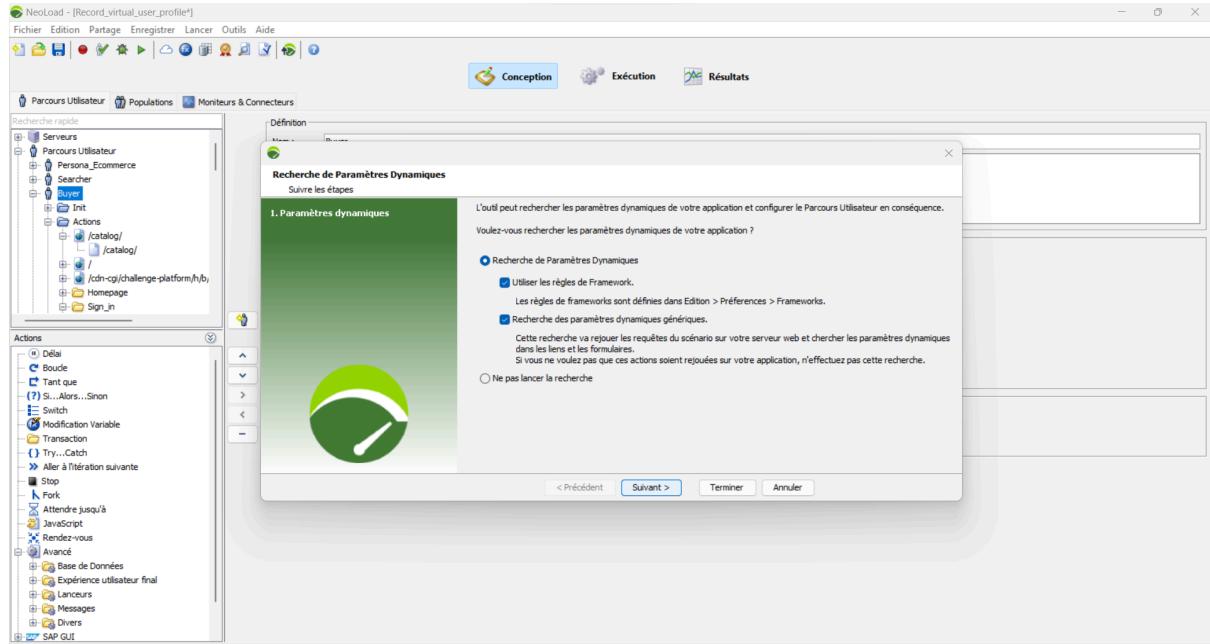
La corrélation est l'action d'identifier, d'extraire et d'utiliser des variables dynamiques provenant du serveur. Aussi, si nous avons des données telles que les **CSRF token** il est fort probable que lors du lancement du **check user path** qu'on ait des **erreurs** car les CSRF token ne sont joués qu'**une seule fois**. Le but ici est de pouvoir corriger ces erreurs et faire des tests ressemblant à des cas réels.

Les différents types de corrélation sont :

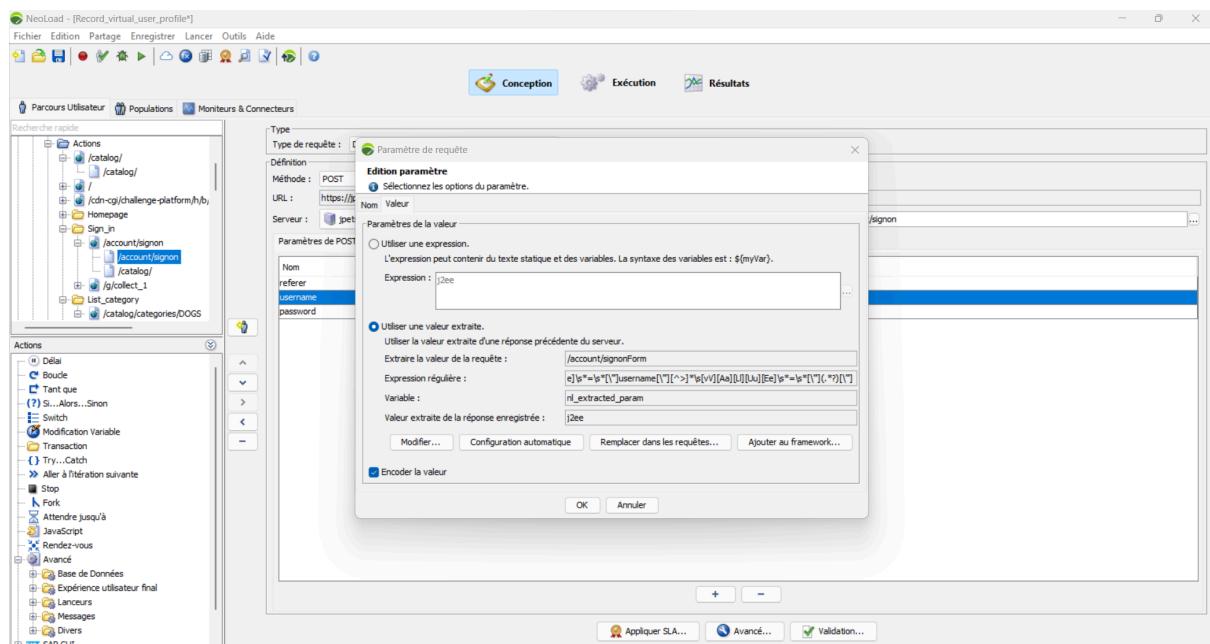
1. Autocorrelation
2. Automatic configuration

3. Manual correlation
4. Advanced correlation

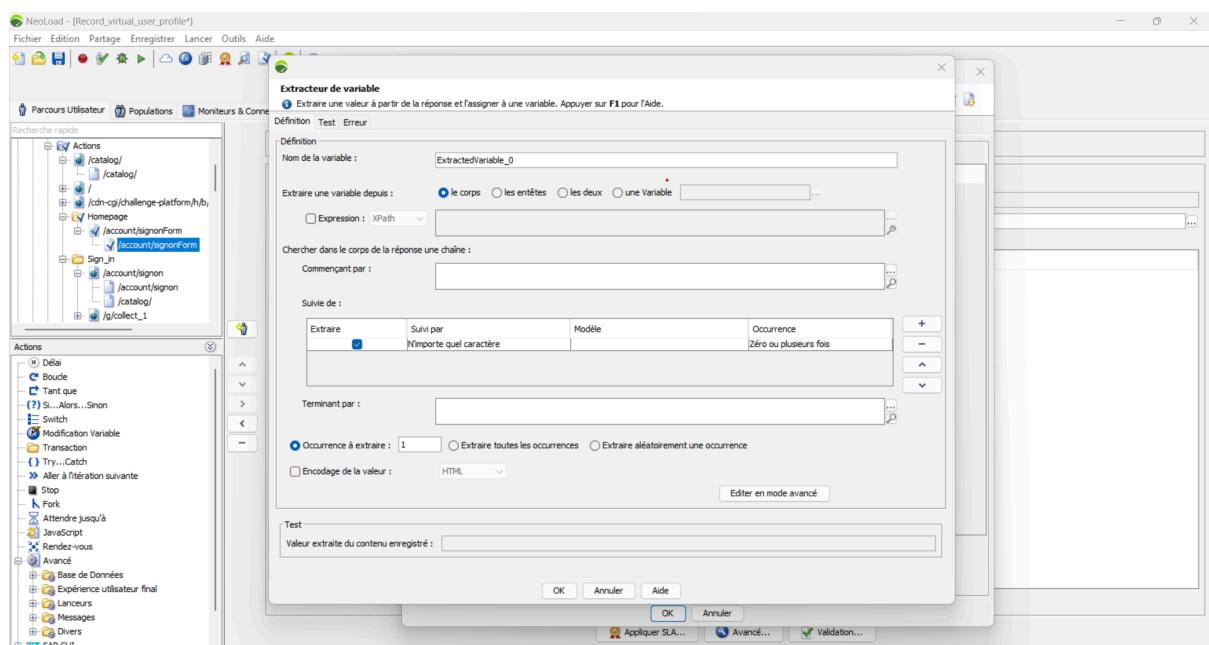
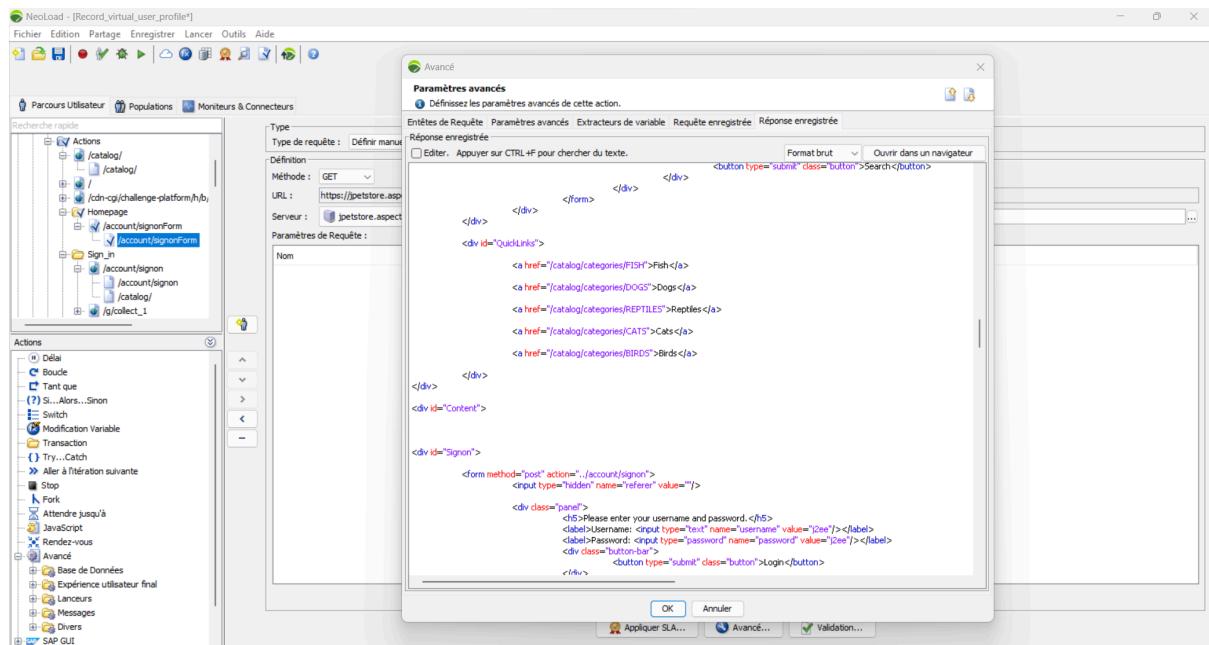
L'autocorrélation c'est lorsqu'on laisse neoload chercher lui même les variables dynamiques, les extraire et faire la corrélation. Cependant, neoload ne peut pas connaître l'ensemble des variables dynamiques d'un workflow.



Automatic configuration : C'est lorsqu'on laisse neoload dynamiser la variable via la configuration automatique. Ici on passe sur chaque variable dynamique et on applique la configuration automatique.



Manual Corrélation : Il s'agit d'identifier chaque variable dynamique, chercher la variable dans le code source (onglet réponse enregistrée), extraire cette variable dynamique (variable extractor) et enfin la remplacer dans les paramètres des requêtes.



Partie 14 : Randomization

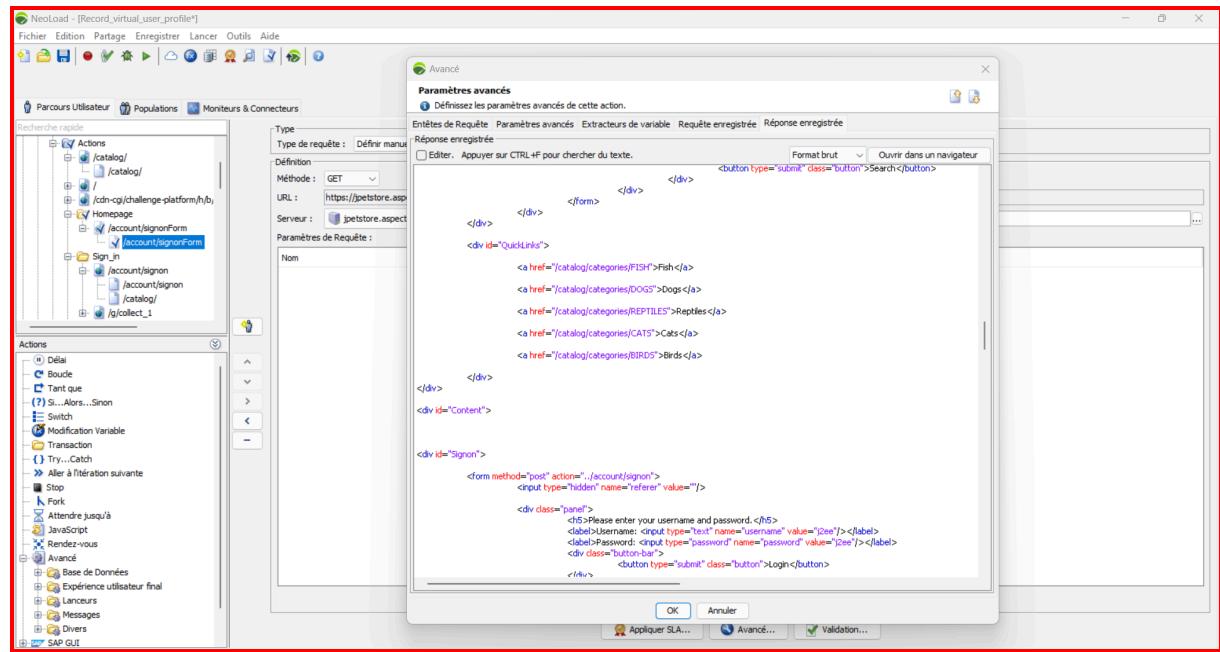
La randomization est l'action de choisir des variables dynamiques tels que des noms de catégories ou bien des ID de produits et les lancer dynamiquement via l'extraction de variable pour se rapprocher beaucoup plus des test réels.

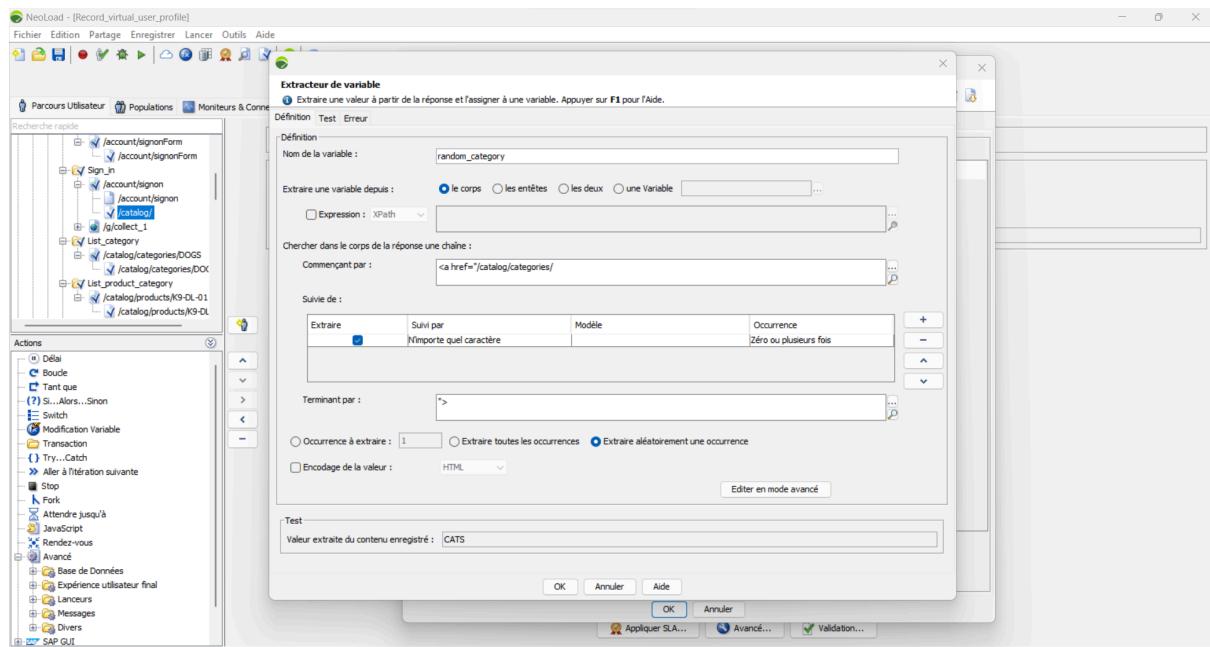
Les différentes étapes de la randomization : identifier les valeurs dynamiques, appliquer la corrélation sur ces valeurs (via extractor variable) et faire le remplacement dans toutes les transactions signalées dans le flag request.

Exemple : Lors d'un recording, on ajoute un produit Tapila appartenant à la catégorie fish. Pour simuler d'autres produits appartenant à Tapila alors on doit implémenter la randomization.

Important : Dans un site e-commerce, pour simuler des achats de divers produits provenant de diverses catégories, alors on doit faire une corrélation des catégories, des sous-catégories (s'il y'en a) et des produits.

Question : Pourquoi lors de la randomization des variables, lorsqu'on fait un flag request pour déterminer toutes les transactions ayant cette variable là, on commence toujours par la transaction précédant la transaction sur lequel on a cliqué ?



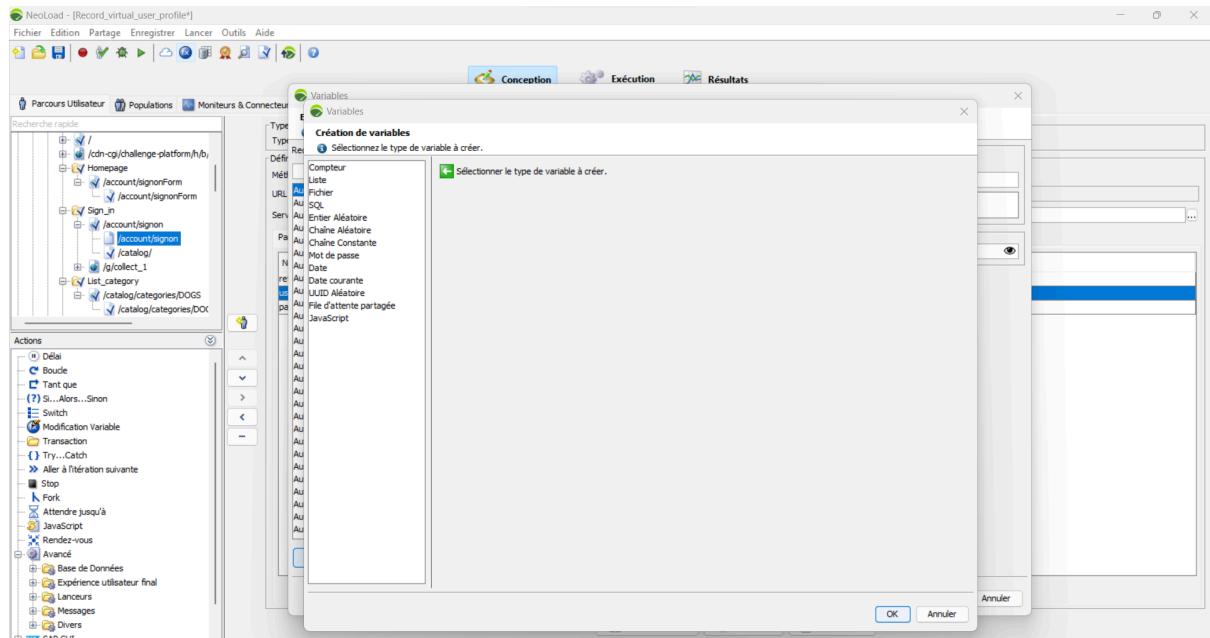


Partie 15 : Parametrization (variable)

Les variables sont des données envoyées par le client au serveur. Les variables sont appelés paramètres dans Neoload.

Ex : Lors d'un login, on envoie le login et le mot de passe au serveur pour une vérification

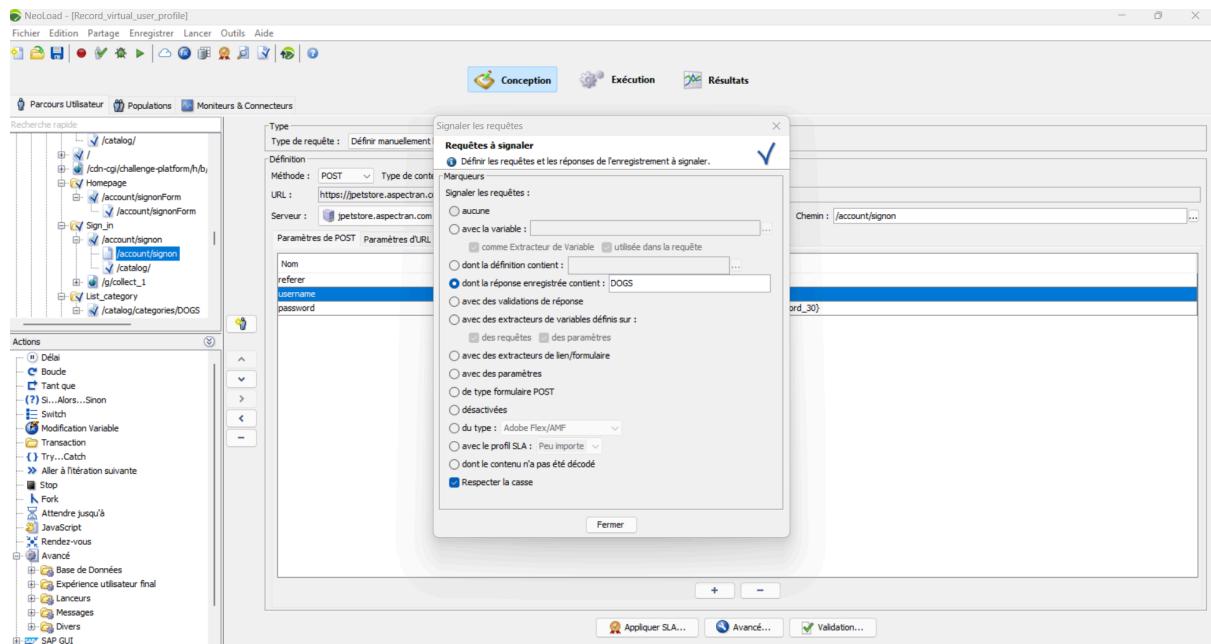
Nous avons différents types de variables (voir image) : Liste (data in table format), File (txt, csv), SQL, Integer, String, etc. **On peut aussi générer une valeur via le code javascript.**



Question : Dans quel cas utiliser les variables SQL (parametrization) ?

Partie 16 : Flag request

Les flags sont utilisés pour chercher des mots clés, des valeurs, des données dans les requêtes ou dans les réponses sous certains critères.



Partie 17 : Assertion

Les assertions permettent de valider une réponse HTTP provenant du serveur sous certains critères.

Il ya 2 types d'assertions :

- Assertion locale : elle est basée sur le contenu de la réponse HTTP, sa taille et sa durée
- Assertion globale :

Assertion basée sur la durée : Elle permet de fixer un délai en ms pour le chargement de telle ou telle page et de voir si cette page parvient à être complètement chargée en fonction du délai fixé.

Valider un Parcours Utilisateur

Vérifier la validité du Parcours Utilisateur

Selectionner le Parcours Utilisateur à valider

Parcours Utilisateur : Buyer

Lancer la validation

Signaler les requêtes...

Avancé...

Le Parcours Utilisateur **Buyer** contient 1 erreur(s).

Parcours Utilisateur :

- Init
- Actions
- /catalog/
- /
- /cdn-cgi/challenge-platform/h/b/cv
- Homepage
- Sign_in
- /account/signon
- /g/collect_1
- List_category
- /catalog/categories/DOGS
- .../catalog/categories/DOGS
- List_product_category
- /catalog/products/K9-DL-01
- List_product_category_1
- /cart/addItemToCart
- Cart
- /order/newOrderForm
- Checkout
- /order/newOrder
- /order/submitOrder
- Confirm_checkout
- /account/signoff
- End

Date Action Code de réponse Assertions Différence

00:00:03.275	/account/signon	302		
00:00:03.540	/catalog/	200		2 %
00:00:03.547	/g/collect_1			
00:00:03.597	/images/banner_reptiles.gif	304		
00:00:03.602	List_category			
00:00:03.602	/catalog/categories/DOGS	200	0/1 assertion réussie	0 %
00:00:03.894	/catalog/categories/DOGS	200	0/1 assertion réussie	0 %
nn:nn:03.954	List_product_category			

Détails Variables Serveurs Rendu HTML Rendu HTML enregistré Rendu HTML comparé

Page : /catalog/categories/DOGS

Requête /catalog/categories/DOGS Référent :

Date : 00:00:03.894 Code de réponse : 200 Durée de réponse : 0,29 s Taille de réponse : 5 Ko

Détails

Requête Réponse Assertions

Le temps de réponse 290 est plus long que 20.

Fermer

Neoload - [Record_virtual_user_profile*]

Fichier Edition Partage Enregistrer Lancer

Parcours Utilisateur Populations Moniteur

Valider un Parcours Utilisateur

Selectionner le Parcours Utilisateur à valider

Parcours Utilisateur : Buyer

Lancer la validation

Signaler les requêtes...

Avancé...

Le Parcours Utilisateur **Buyer** contient 1 erreur(s).

Parcours Utilisateur :

- Init
- Actions
- /catalog/
- /cdn-cgi/challenge-platform/h/b/cv/
- Homepage
- Sign_in
- /account/signon
- /g/collect_1
- List_category
- /catalog/categories/DOGS
- .../catalog/categories/DOGS
- List_product_category
- /catalog/products/K9-DL-01
- List_product_category_1
- /cart/addItemToCart
- Cart
- /order/newOrderForm
- Checkout
- /order/newOrder
- /order/submitOrder
- Confirm_checkout
- /account/signoff
- End

Date Action Code de réponse Assertions Différence

00:00:05.194	/images/banner_reptiles.gif	304		
00:00:05.199	List_category			
00:00:05.199	/catalog/categories/DOGS	200	0/1 assertion réussie	0 %
00:00:05.985	/catalog/categories/DOGS	200	0/1 assertion réussie	0 %
00:00:06.052	List_product_category			
00:00:06.052	/catalog/products/K9-DL-01	200	0/1 assertion réussie	0 %
00:00:06.819	/catalog/products/K9-DL-01	200	0/1 assertion réussie	0 %
nn:nn:06.819	List_product_category			

Détails Variables Serveurs Rendu HTML Rendu HTML enregistré Rendu HTML comparé

Page : /catalog/categories/DOGS

Requête /catalog/categories/DOGS Référent :

Date : 00:00:05.985 Code de réponse : 200 Durée de réponse : 0,786 s Taille de réponse : 5 Ko

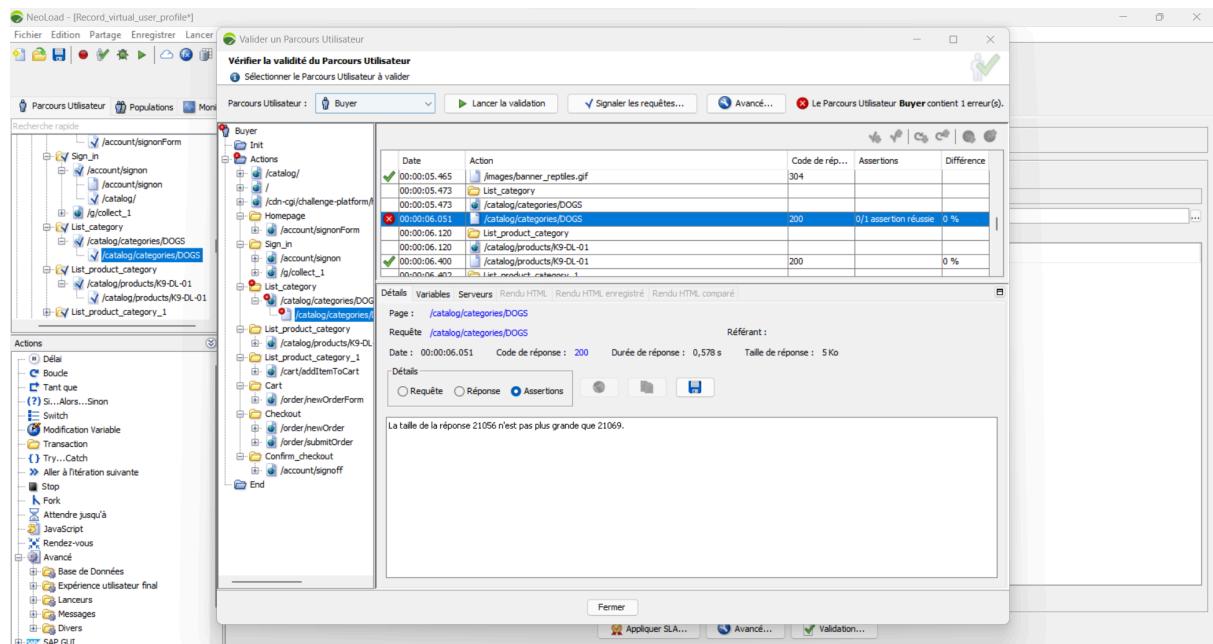
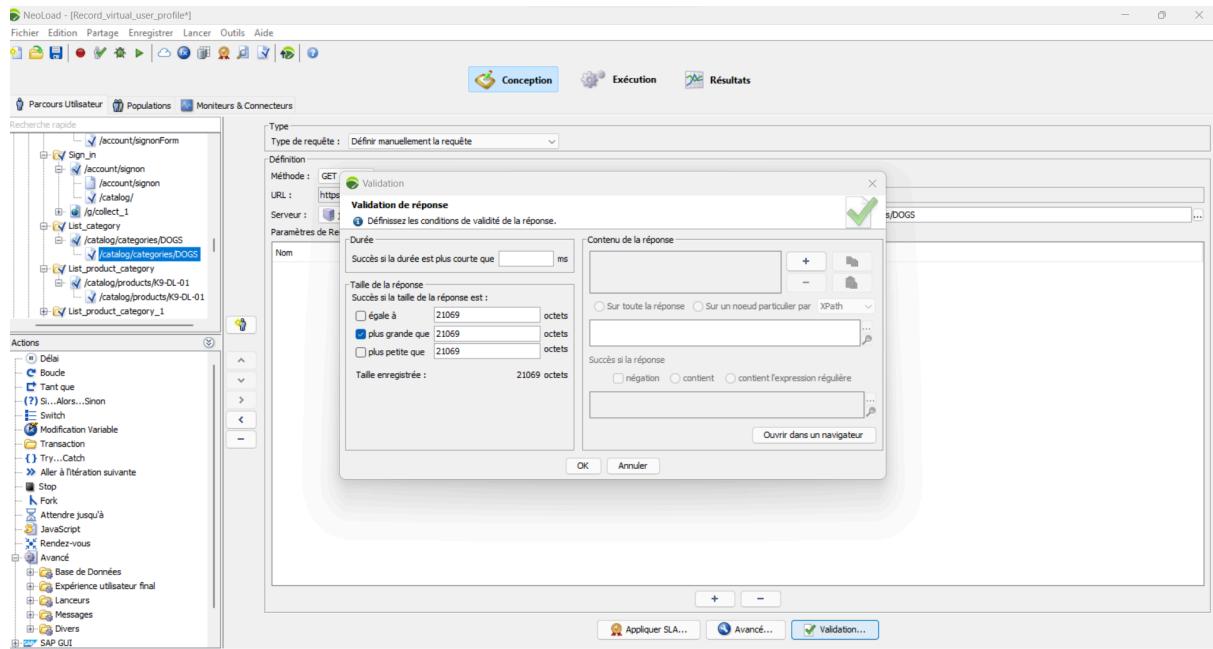
Détails

Requête Réponse Assertions

Le temps de réponse 786 est plus long que 200.

Fermer

Assertion basée sur la taille : L'assertion basée sur la taille permet de faire une comparaison entre la taille de la réponse HTTP reçue et la taille passée en paramètre. Elle est utilisée pour ajouter une assertion dans le chargement des fichiers du serveur vers le client.



Assertion basée sur le contenu de la réponse : Elle permet de vérifier si parmi les réponse HTTP provenant du serveur nous avons tel ou tel texte ou donnée ou nombre, etc

The screenshot displays the NeoLoad application interface, specifically the 'Validation' configuration and the 'Validation Results' window.

Validation Configuration (Left Panel):

- Type de requête :** Définir manuellement la requête
- Méthode :** GET
- URL :** https://www.chezvous.com/Validation
- Validation** (Selected): Définissez les conditions de validité de la réponse.
- Paramètres de Requête :** Nom: var_search
- Contenu de la réponse :**
 - Durée :** Succès si la durée est plus courte que 21069 ms
 - Taille de la réponse :** Succès si la taille de la réponse est égale à 21069 octets
 - Succès si la réponse :** Contient l'expression régulière <title>(.*)</title>

Validation Results (Right Panel):

Vérifier la validité du Parcours Utilisateur

Parcours Utilisateur : Buyer

Validation Summary: Le Parcours Utilisateur Buyer est valide.

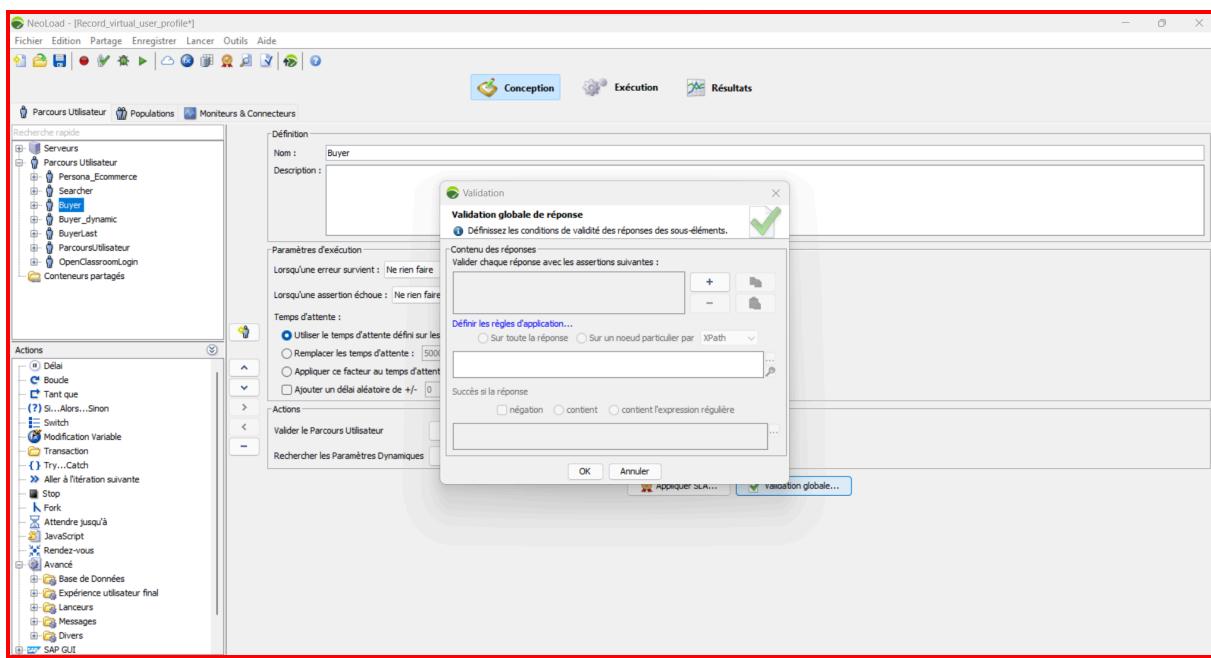
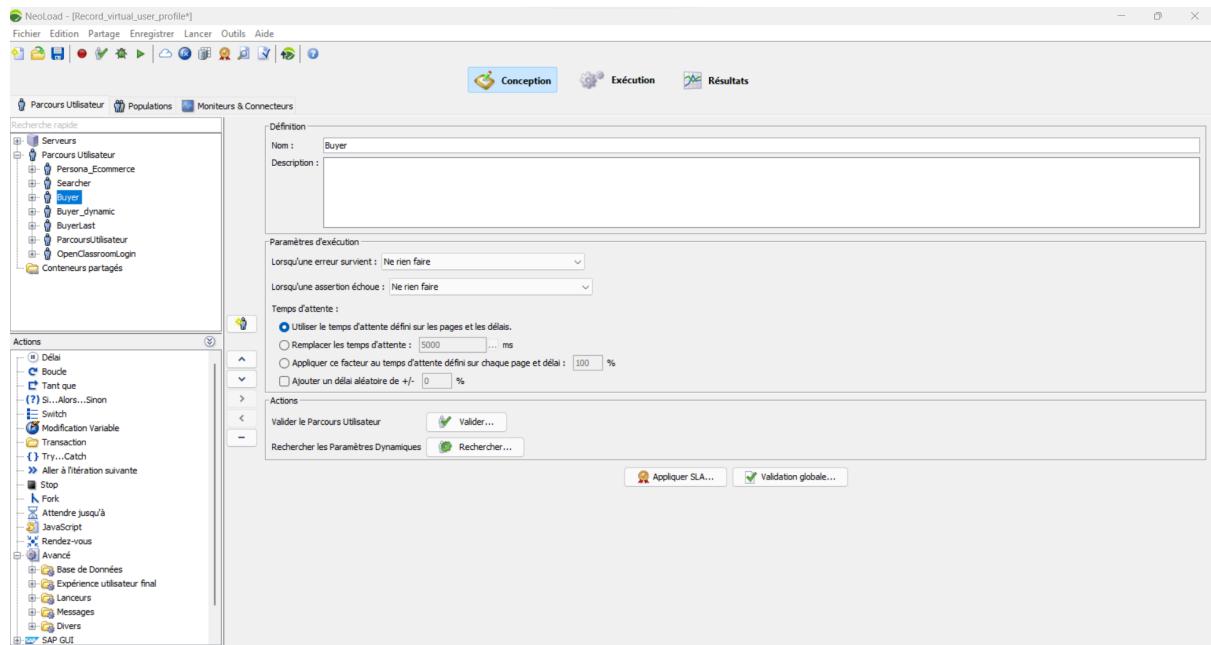
Validation Details:

Date	Action	Code de réponse	Assertions	Déférence
00:00:03.295	/g/collect_1	304		
00:00:04.110	/images/banner_reptiles.gif			
00:00:04.115	List_category			
00:00:04.115	/catalog/categories/DOGS			
00:00:04.392	/catalog/categories/DOGS	200	1/1 assertion réussie 0 %	
00:00:04.651	List_product_category			
00:00:04.651	/catalog/products/K9-DL-01			

Details View:

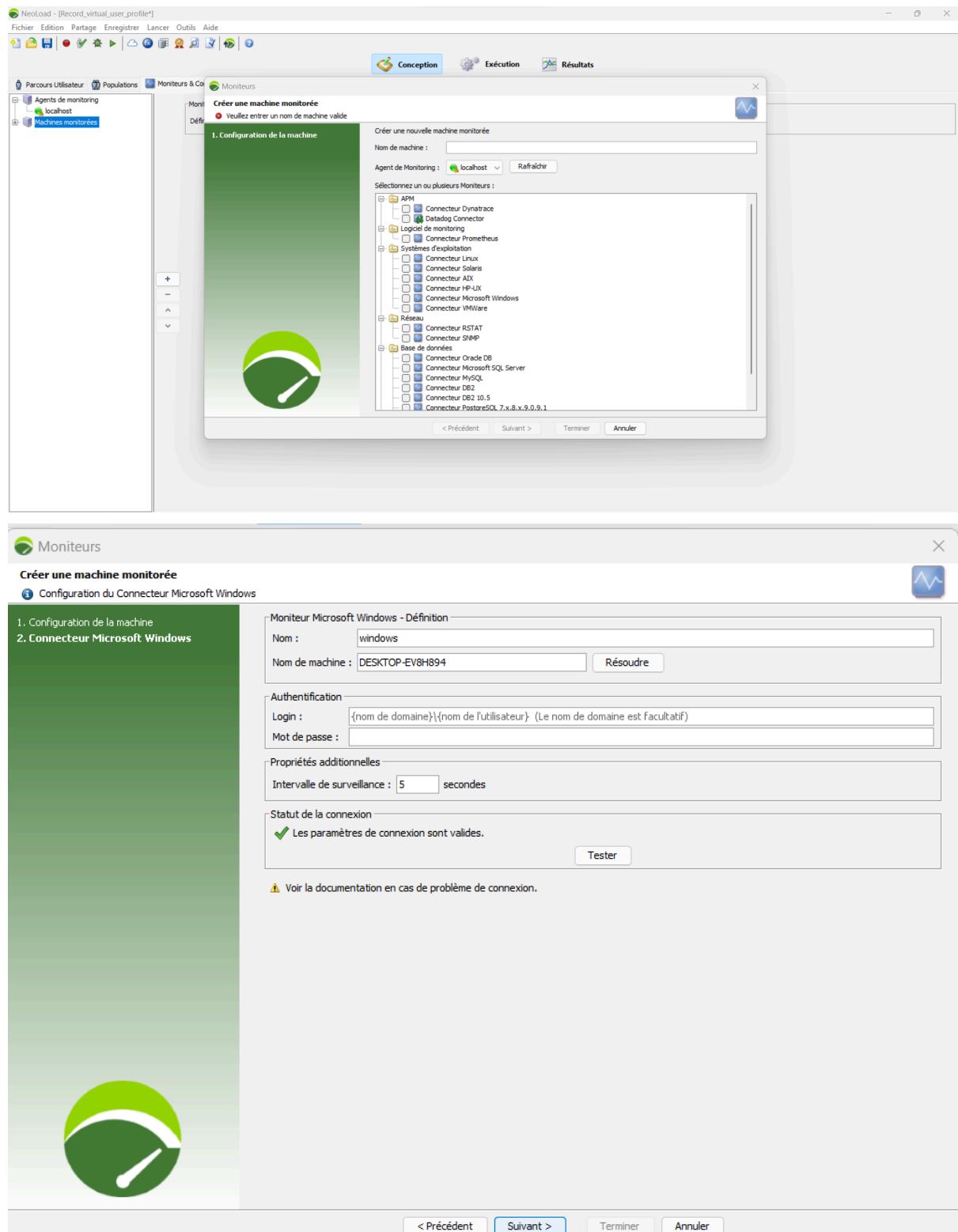
- Page : /catalog/categories/DOGS
- Requête : /catalog/categories/DOGS
- Référent :
- Détails : Requête, Réponse, Assertions

Assertions globales : on double click sur le nom du parcours utilisateur ou workflow. Cette assertion va concerner l'ensemble des transactions ou containers.



Partie 18 : Monitor machine

Les moniteurs dans Neoload nous permettent de visualiser le comportement des serveurs, des bases de données, de la RAM etc durant l'exécution des tests.



1. Configuration de la machine

Créer une nouvelle machine monitrice

Agent de Monitoring : localhost

Sélectionnez un ou plusieurs Moniteurs :

- APM
- Logiciel de monitoring
- Systèmes d'exploitation
- Réseau
- Base de données

2. Connecteur Microsoft Windows

Moniteur Microsoft Windows - Définition

Nom : windows

Nom de machine : DESKTOP-EV8H894

Authentification

Login : {nom de domaine}\{nom de l'utilisateur} (Le nom de domaine est facultatif)

Mot de passe :

Propriétés additionnelles

Intervalle de surveillance : 5 secondes

Statut de la connexion

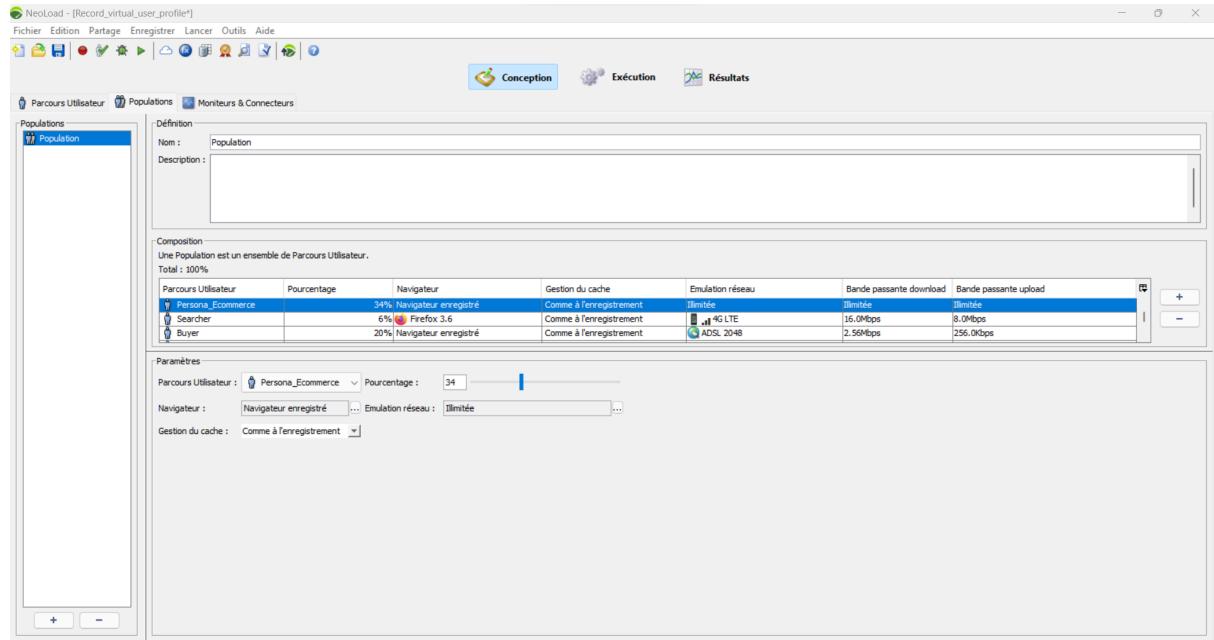
✓ Les paramètres de connexion sont valides.

Voir la documentation en cas de problème de connexion.

< Précédent Suivant > Terminer Annuler

Partie 19 : Population

Une population est un ensemble d'utilisateurs virtuels ayant des parcours utilisateurs communs ou différents.



Partie 20 : Runtime

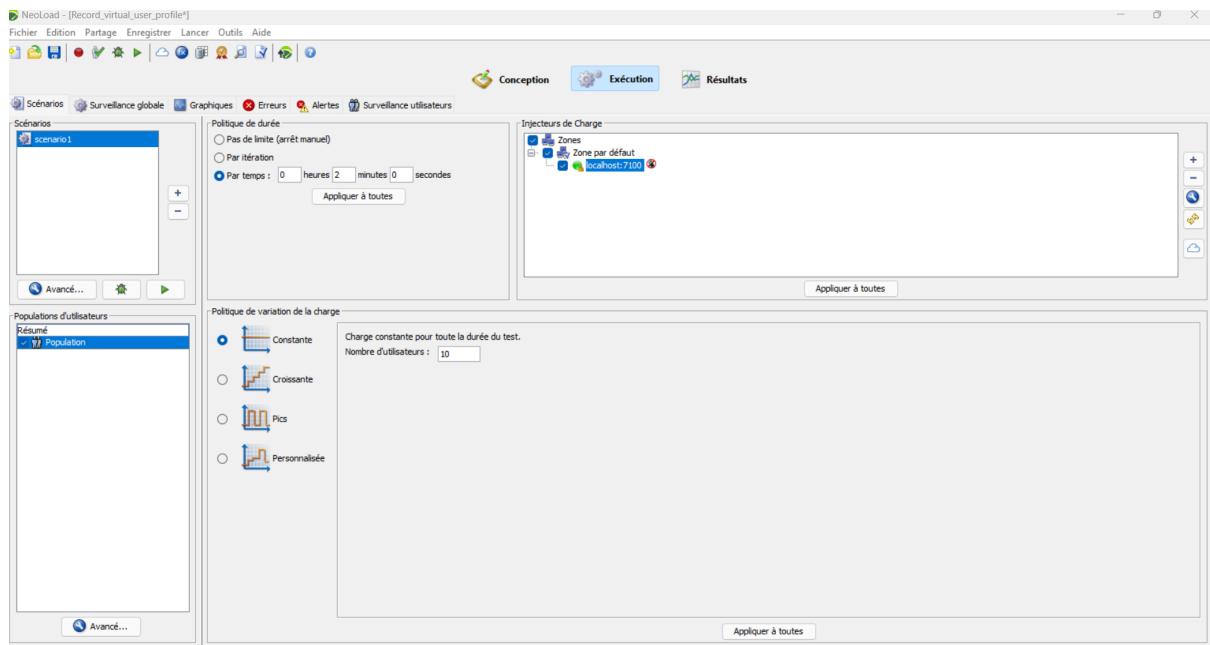
Runtime permet de mettre en place des scénarios et d'exécuter les tests.

Un workflow est l'ensemble des transactions ou users stories. C'est le parcours utilisateur.

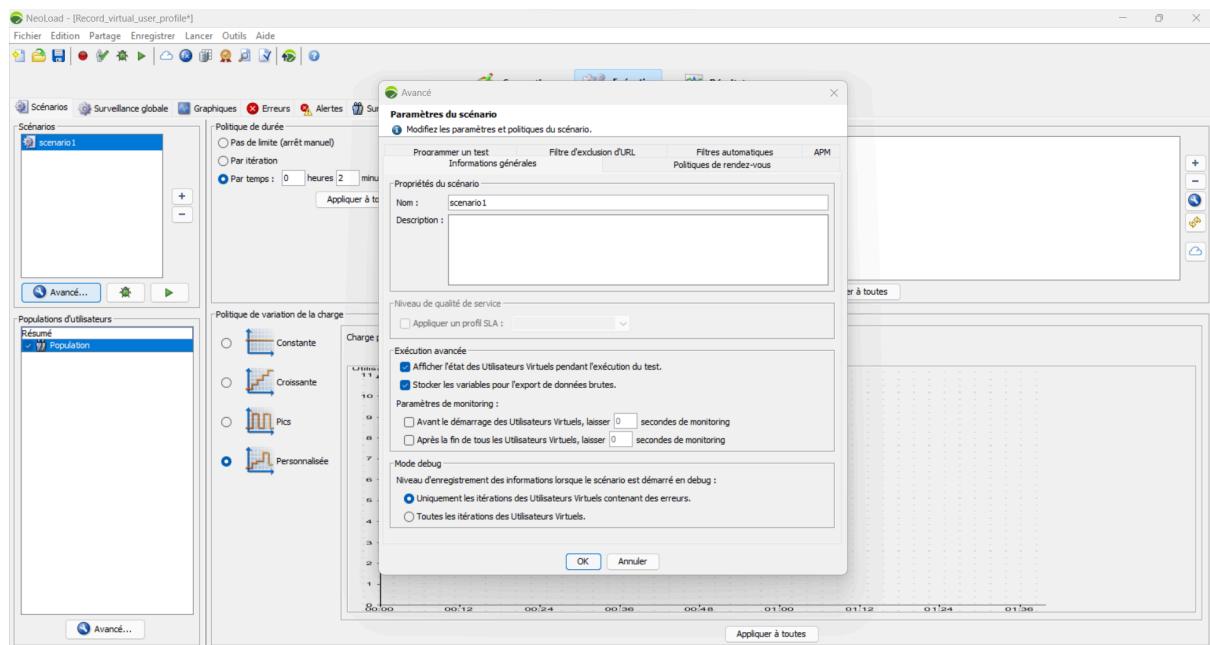
Un scénario permet de définir le type de test de performance que l'on doit exécuter et nous renseigne sur comment le faire avec ses parties **politique de durée, injecteur de charge, politique de variation de la charge et population** (configuré dans la partie population)

Il ya différents types de test de performance :

- Load test
- Smoke test
- Stress test
- Spike test
- Scalability test
- Endurance test



Dans la partie avancé nous avons :



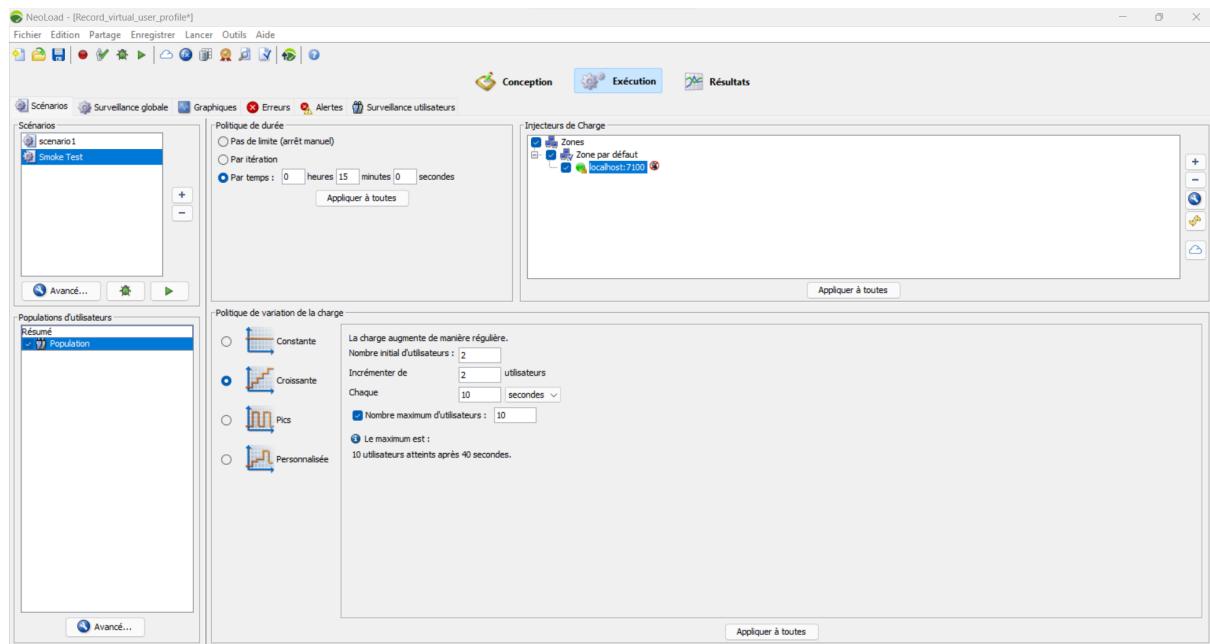
Load variation policy ou politique de variation de la charge : il permet de configurer le comportement des utilisateurs durant l'exécution des tests. On peut appliquer une charge constante, une charge croissante, une charge décroissante des utilisateurs.

Duration policy ou politique de durée : il permet de chronométriser la durée du test

Load generator ou injecteur de charge : il permet de partager la charge durant les tests.

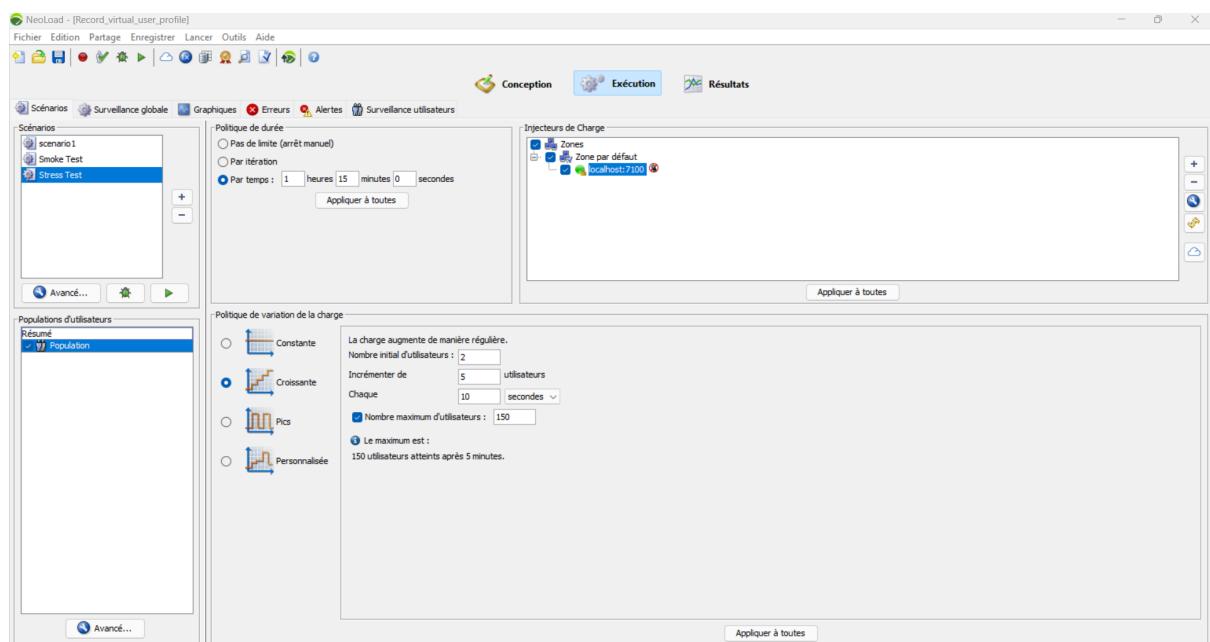
Partie 21 : Smoke Test

Il permet de vérifier si le scripting ne présente aucun problème lors des tests. Ici on ne fait pas focus sur la charge, on vérifie juste que l'application fonctionne même avec un seul utilisateur). C'est un test qui dure généralement 15 minutes avec 10 utilisateurs virtuels.



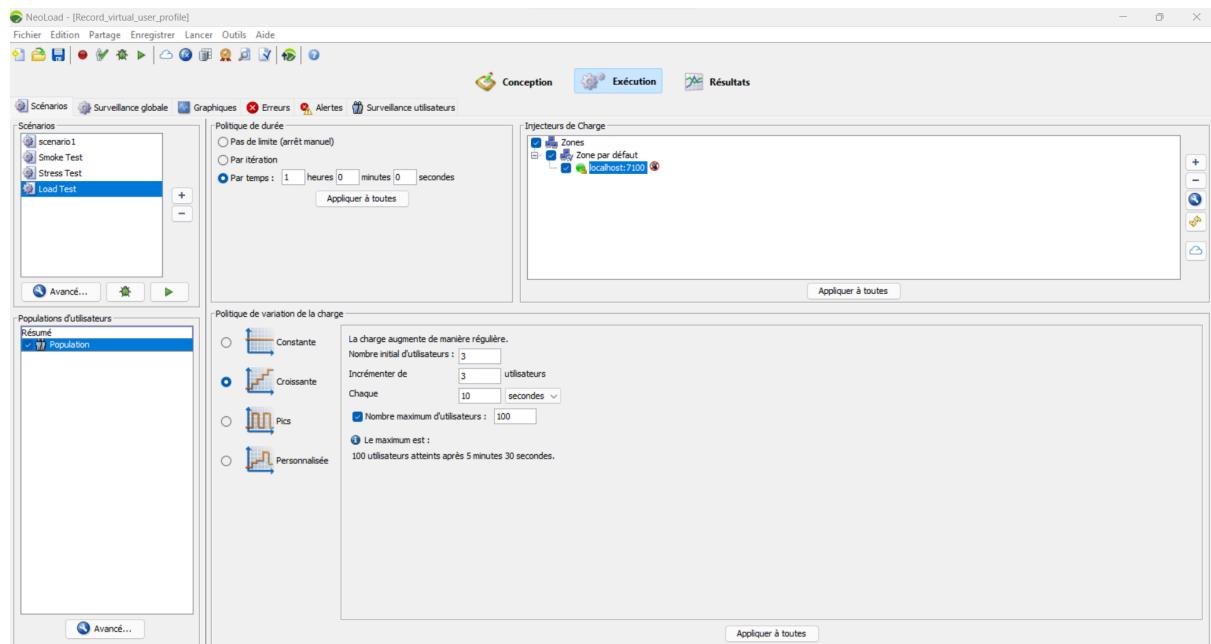
Partie 22 : Stress Test

Le test de stress s'effectue en envoyant une charge au système au-dessus de la normale pour voir quelles parties du système vont résister ou faiblir face à la montée en puissance des requêtes.



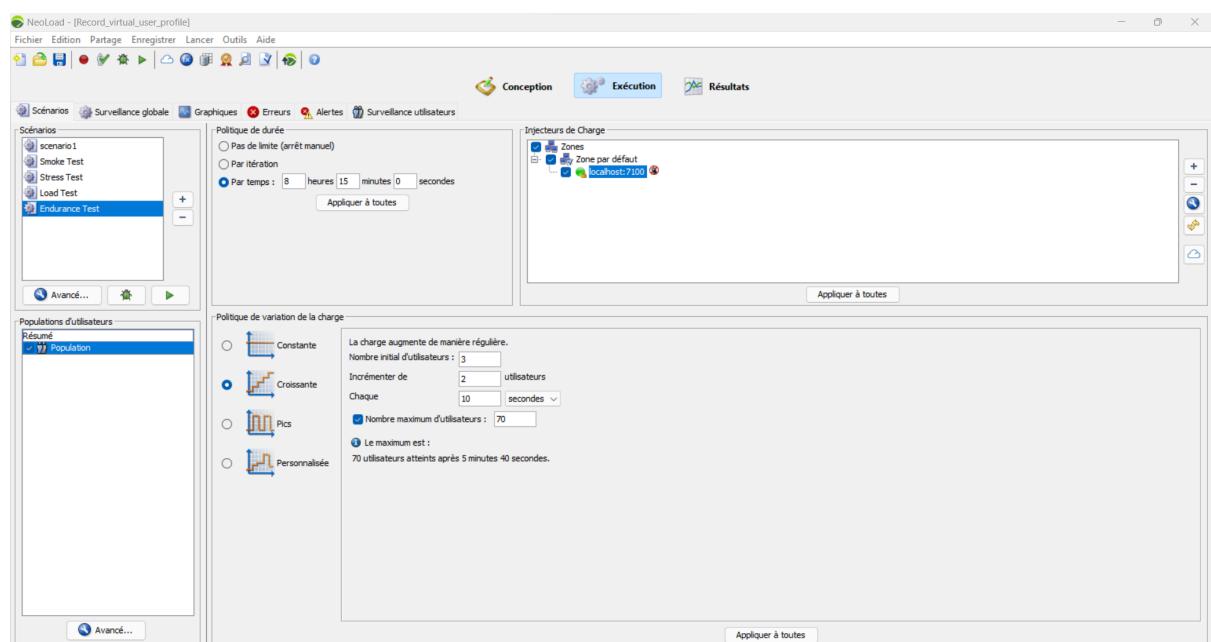
Partie 23 : Load Test

Un test de charge est l'action d'envoyer une charge normale à un système et de mesurer les réponses. Ici on met l'accent sur les temps de réponse dans des conditions de requête normale.



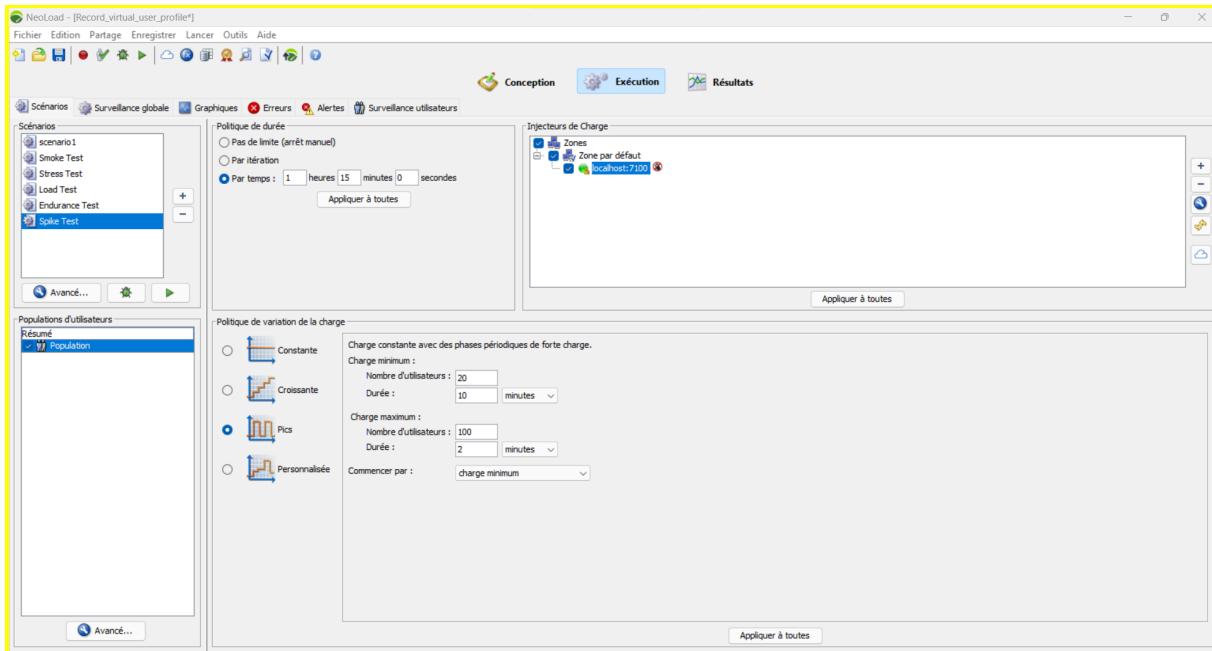
Partie 24 : Endurance Test

Le test d'endurance s'effectue en envoyant une charge constante au système pendant plusieurs heures pour vérifier si le système est capable de tenir le rythme des requêtes pendant longtemps.



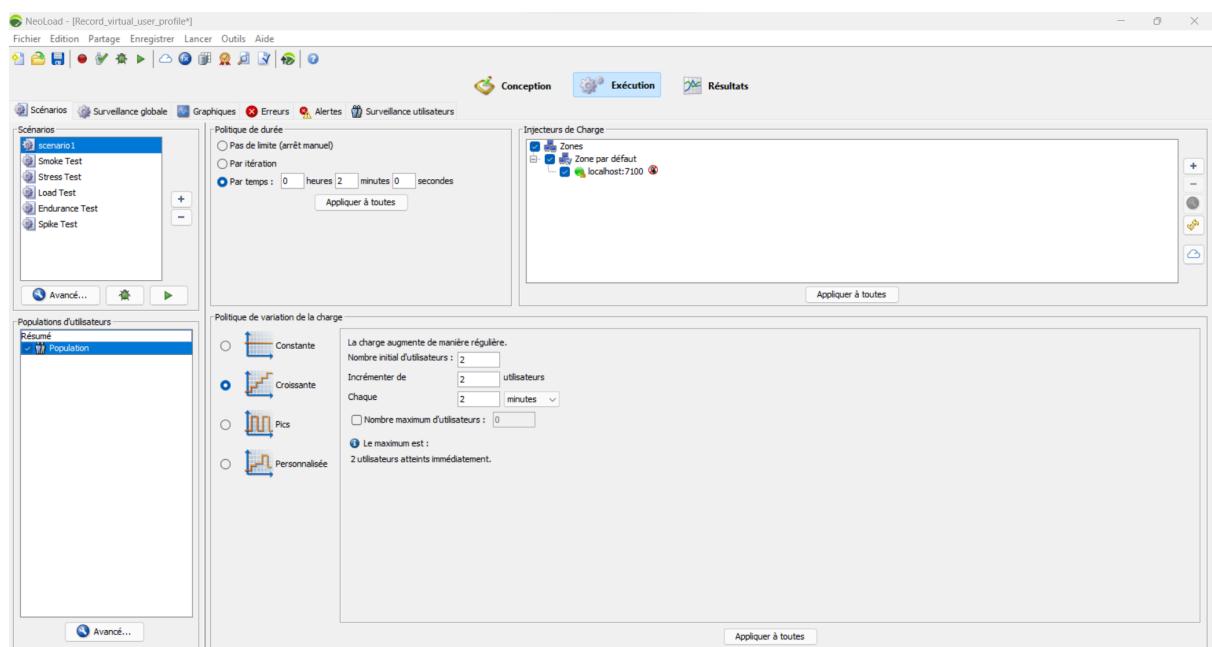
Partie 24 : Spike Test

Il s'effectue en envoyant des pics de requêtes, d'utilisateurs virtuels par moment dans le système. La charge peut être croissante comme décroissante.



Partie 25 : Exécution des Test

Pour exécuter un type de test de performance (test de stress, d'endurance, de charge, smoke test), il faudrait configurer la partie scénario qui se trouve dans la partie Runtime ou Exécution.

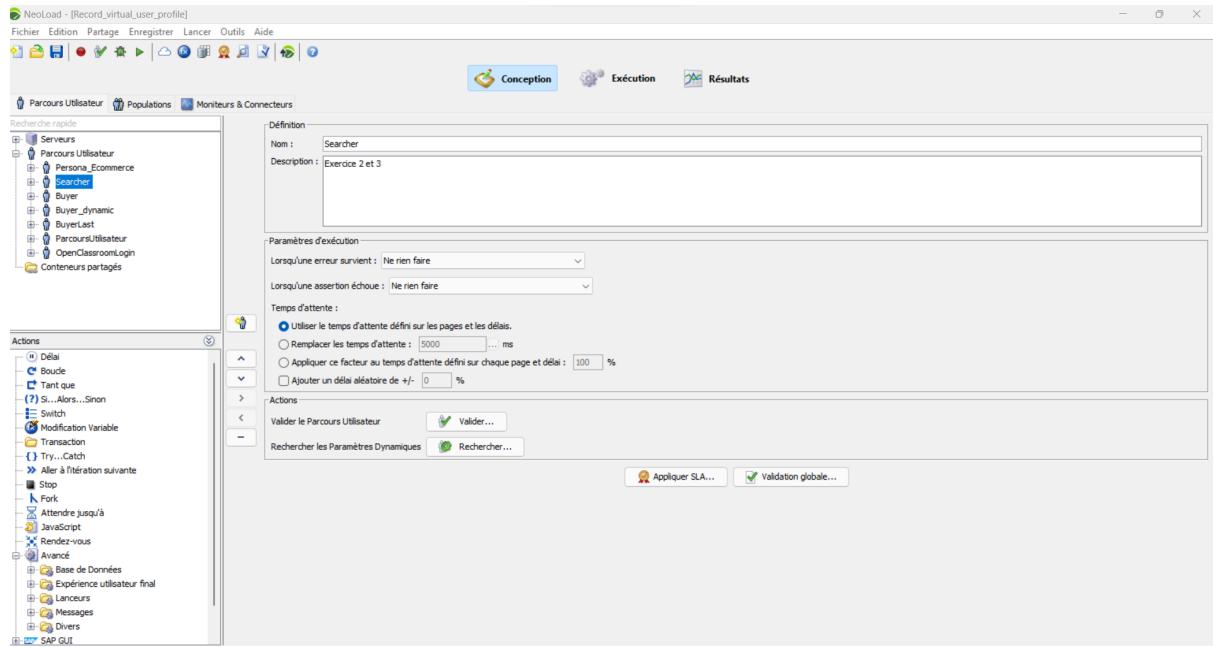


Ecran scénario

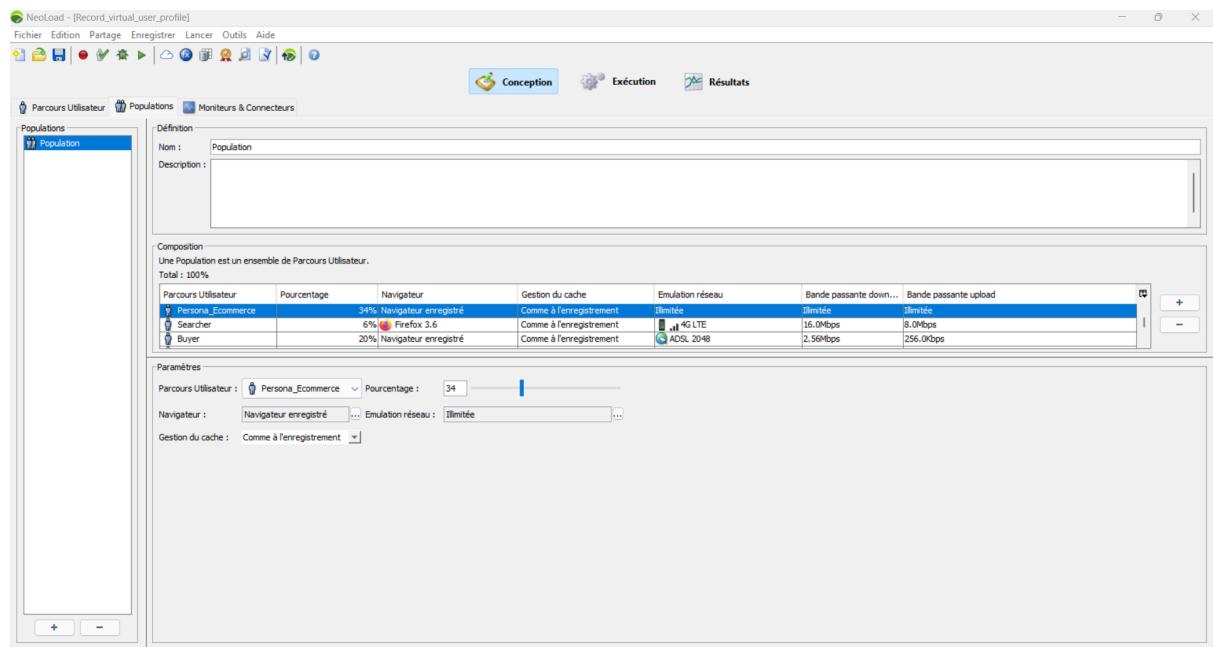
Pour disposer de tous éléments dans la partie scénario, il faudrait au préalable :

Accéder dans la partie Conception qui est divisé en 3 parties :

- **Parcours utilisateurs** : qui permet de faire le recording ou scripting d'une application web en se basant sur un parcours utilisateur.



- **Population** : elle permet de configurer tous les types d'utilisateurs virtuels utilisant l'application web. Aussi, des choix concernant les navigateurs et les réseaux (4g, H+, 3g) qu'ils utilisent ainsi que le nombre de chaque type d'utilisateurs virtuels sont configurables.



- **Moniteurs et connecteurs** : elle permet d'ajouter des équipements (serveur, machine, OS), des bases de données, des RAM etc sur laquelle vont s'exécuter nos types de test.

