Neo Load

NeoLoad is a performance testing tool specifically designed for web and mobile applications.

**scalable load generation**,

* means we can be able to increase or decrease the load **during the test execution as well**

**Advanced real time monitoring**

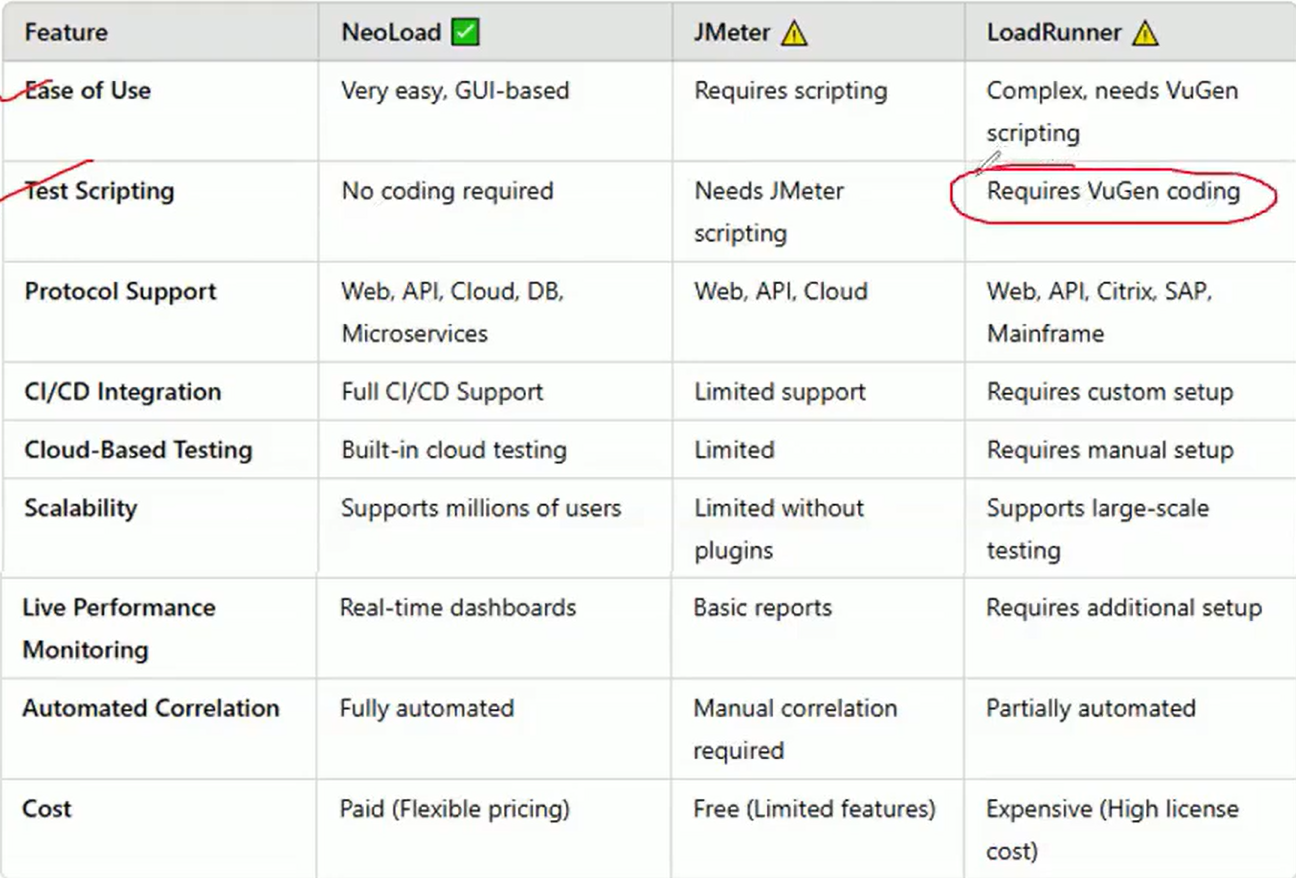
**Detailed Reporting-**

* **S**tandard report and Comparison report, Realtime Dashboard

**Cloud load generation**

* Refers to the process of simulating a large number of virtual users accessing an application or service from cloud-based infrastructure to test its performance under heavy load.
* These virtual users are generated from machines hosted in the cloud, allowing for the simulation of large-scale user activity.

Differentiation,



**In NeoLoad, -- Load**

An on-premise load generator is referred to as a Local Load Generator or On-Premise Load Generator, while a cloud-based load generator is called a Cloud Load Generator**.**

**A diagram of a computer system

AI-generated content may be incorrect.**

**On Permise load generator,**

An on-premises load generator is a software or service that simulates user traffic to test the performance of an application, but unlike cloud-based generators, it's installed and run within the organization's own network (on-premise) rather than on external servers. It's particularly useful for testing applications that are not accessible from the public internet, such as those behind a firewall or within an internal network

A diagram of a server

AI-generated content may be incorrect.

**#Controller= Neo load controller**

**Use of On Permise load Generator,**

1. On-prem LGs generate virtual **users inside your own infrastructure**, making them ideal when testing internal applications, such as:

* Internal APIs
* ERP/CRM platforms
* Intranet web apps

This avoids firewall, VPN, or cloud routing issues that cloud LGs might face.

1. **Test Applications Behind Firewalls / VPNs**

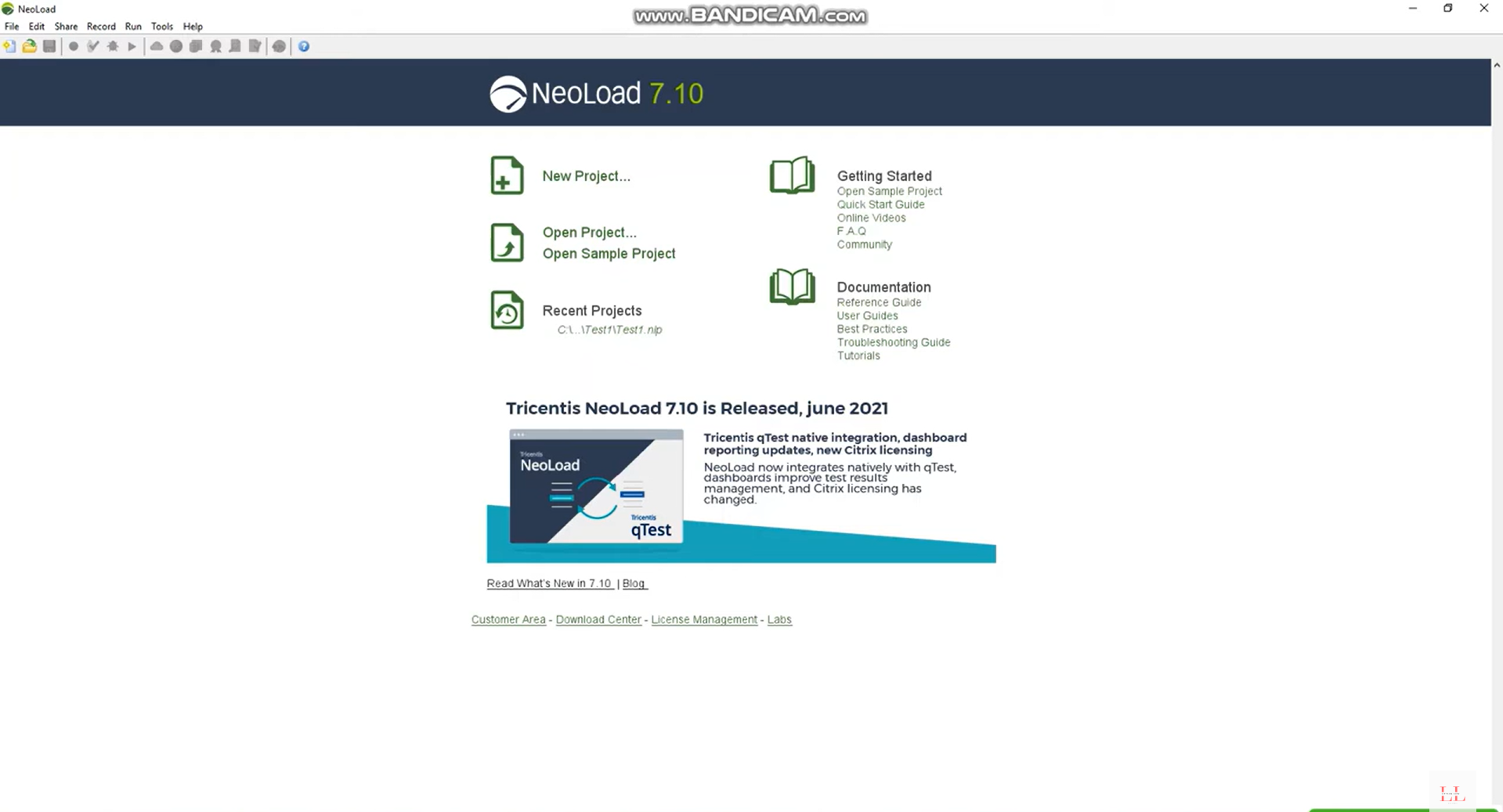
* If your application is not exposed to the internet, only on-prem LGs can access it.
* You control all aspects of network routing, security, and data privacy

**On Permise load generator VS Cloud load generator**

If a company wants to test how well their website handles traffic, they might:

* Use **on-premise load generators**: Set up their own servers in their office or data center to simulate traffic.
* Versus **cloud load generators**: Rent servers from a cloud provider like AWS, Azure, or Google Cloud to do the same thing.

NeoLoad UI:



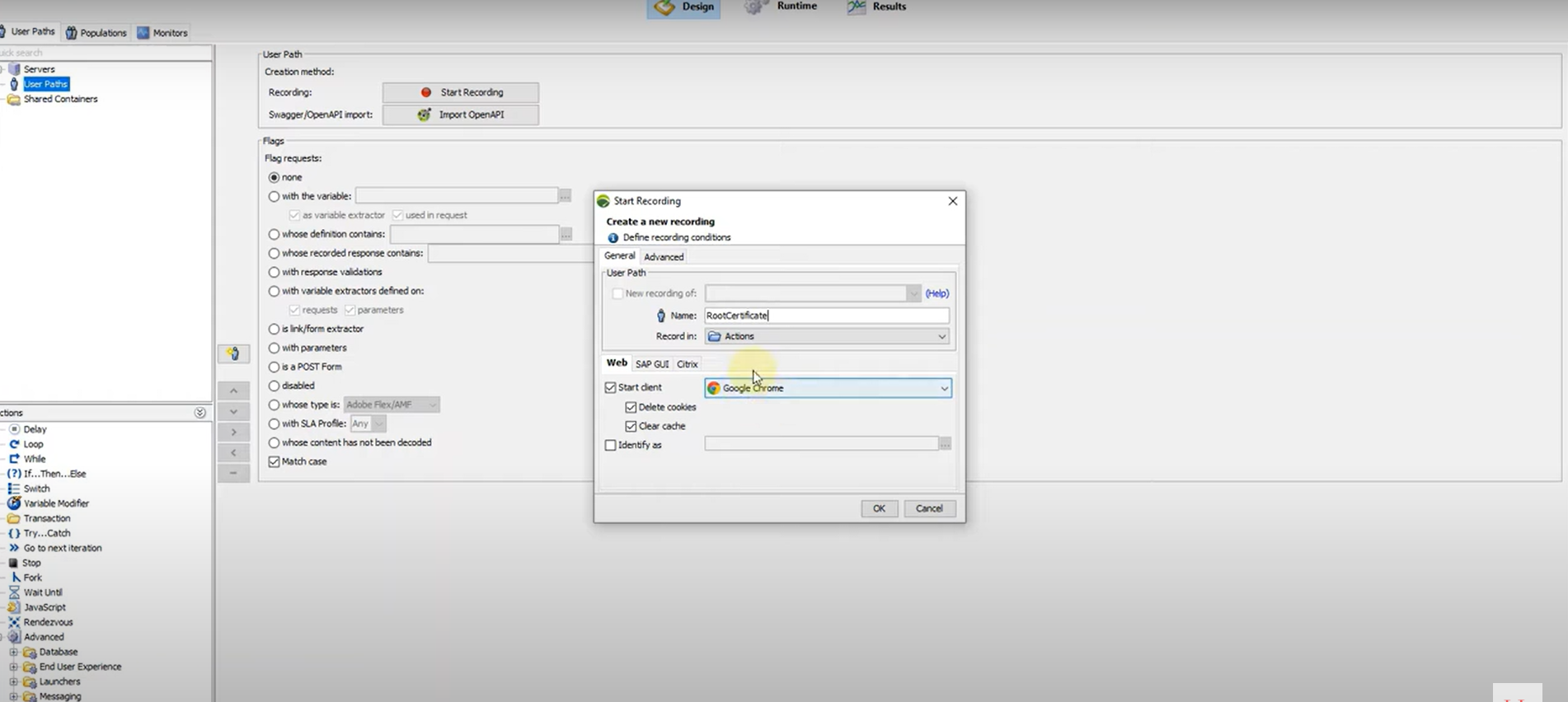
To Create a new script, and open a new script we can select the appropriate options on the above

A screenshot of a computer

AI-generated content may be incorrect.

**Controller window**

**Recorder Dialogbox:**

****

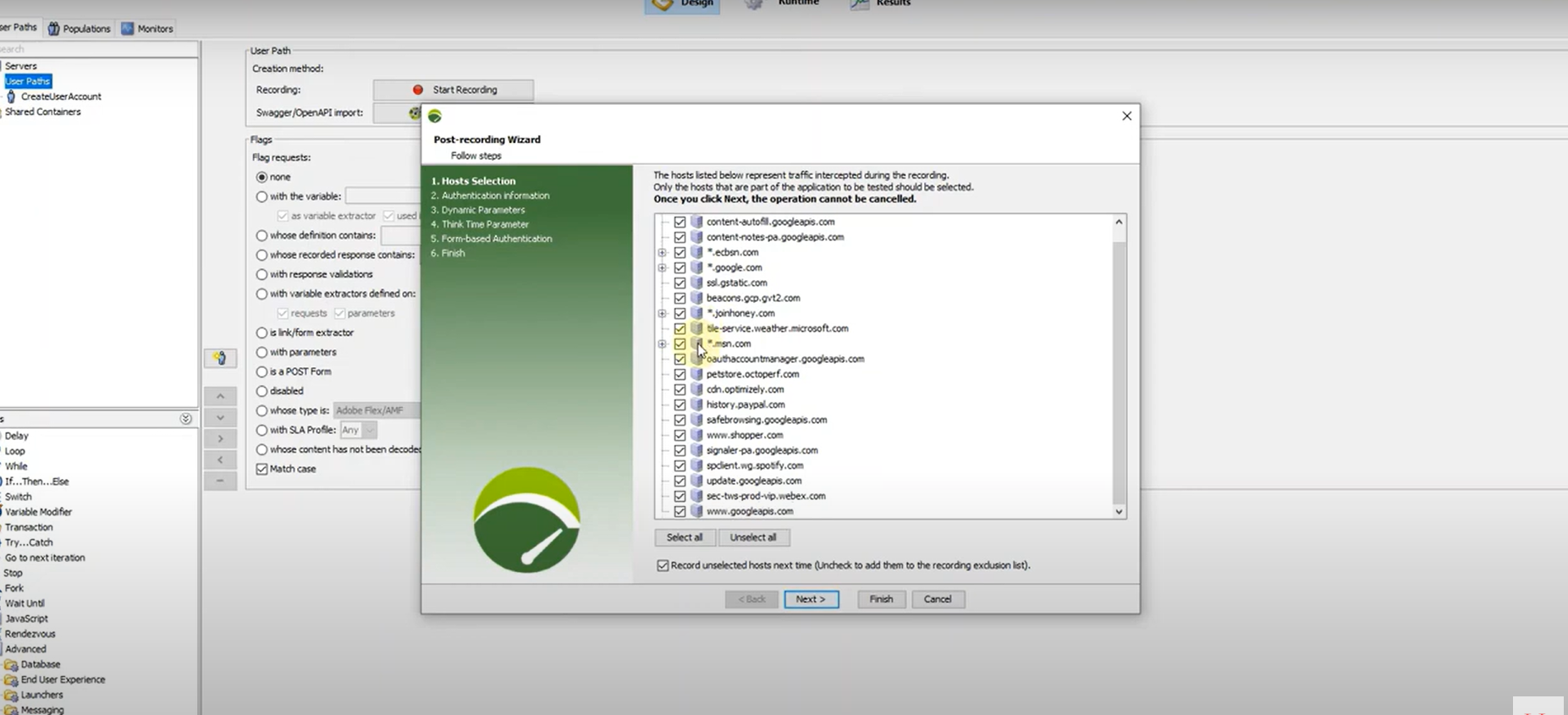
* Default Record IN Option selected in : Actions Folder but totally 2 folders are there INIT and ACTION Folder

**Recording:**

We can record the scenarios by giving the transaction names in the record dialog box.

Once you recorded all the transactions click on stop button it will open **post recording wizard**

**Post recording wizard**



1. **Hosts Selections-**

* After recording a scenario, NeoLoad identifies all the **hosts (servers)** that were contacted during the session.
* You **select which hosts** are relevant to your test.
* This helps **filter out third-party services**

**Tip:** Keep only the hosts that are under your control or are critical to the user journey

1. **Authentication Information**

* If your application uses **authentication mechanisms** (like Basic Auth, OAuth, or NTLM), NeoLoad allows you to:
* **Enter credentials** for each host that requires authentication.
* **Configure login behavior** so that virtual users can authenticate during test execution

**Tip:** Use **variables** for usernames and passwords to simulate multiple users.

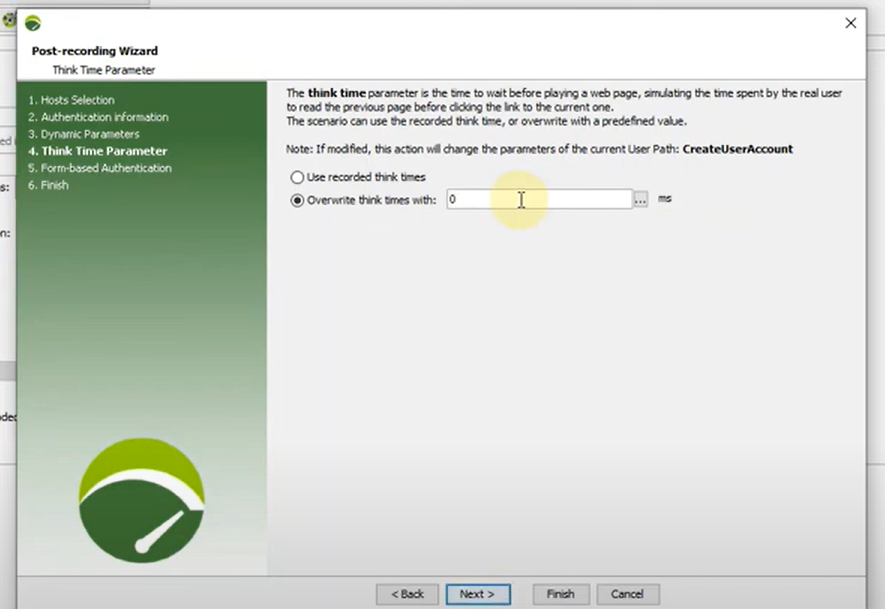
1. **Dynamic Parameters**

* Modern web applications often use **dynamic values** (like session IDs, tokens, timestamps) that change with each request. In this step:
* NeoLoad detects and highlights **dynamic parameters**.
* You can **correlate** these values using **extractors** (like regular expressions or JSONPath).
* Then, **replace hardcoded values** in subsequent requests with the extracted ones.

**Tip:** Use the **"Flag as dynamic parameter"** feature to automate correlation.

1. **Think time Parameters**

* **delay between user actions**, making the test more realistic.:
* Recorded think time (as captured during recording)
* Random think time (within a range)
* Fixed think time



**Tip:** Use realistic think times for **load testing** and disable them for **stress testing**.

1. **Form based Authentication**

* If your application uses form-based login (common in web apps), NeoLoad allows you to:
* Identify the login request and form fields (like username and password).
* Parameterize these fields using variables.
* Ensure the login is successful by validating the response (e.g., checking for a welcome message or redirect).
* We can also select **Use the account from the CSV**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Tip: Use validation rules to confirm successful login during test execution.**

1. Finish

* Once all the above steps are completed:
* NeoLoad finalizes the **recorded user path**.

**Recorded Script:**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Types of Variables to Parameterize,**

**1. User Path Variables**

**These are variables defined within a specific user path and are used to simulate dynamic user data.**

* **Constant**: Fixed value throughout the test.
* **File**: Values are read from a CSV or text file (e.g., usernames, passwords).
* **Random**: Generates random values (numbers, strings, etc.).
* **Counter**: Increments a value with each iteration or user.
* **Date**: Generates date/time values in specific formats.
* **List**: Predefined list of values to be used sequentially or randomly.
* **SQL**: Fetches values from a database query.
* **JavaScript**: Custom logic using JavaScript expressions.
* **Extractors**: Values extracted from server responses (e.g., session IDs, tokens).

**2. Built-in Variables**

NeoLoad provides several **predefined variables** for convenience:

* NL-CURRENT-USER: Current virtual user ID.
* NL-CURRENT-ITERATION: Current iteration number.
* NL-CURRENT-TIME: Current timestamp.
* NL-SCENARIO: Name of the running scenario.

1. **Project Variables**

These are **global variables** accessible across all user paths in a project.

* Useful for **shared configuration values**, like environment URLs or credentials.
* Can be overridden at runtime using **command-line parameters** or **runtime files**

**4. Secret Variables**

Used to store **sensitive data** like passwords or API keys.

* Values are **encrypted** and not visible in the UI or logs.
* Helps maintain **security and compliance** during testing.