



HAUTE ÉCOLE  
D'INGÉNIERIE ET DE GESTION  
DU CANTON DE VAUD  
[www.heig-vd.ch](http://www.heig-vd.ch)

IL - TIC - HEIG-VD

APPLICATION MULTITIERS

---

# Rapport labo1 : Test et JPA

---

*Auteurs :*

M. Olivier DJEULEZECK

*Encadrants :*

M. Patrick LACHAIZE

Version du :  
30 novembre 2020

**Table des matières**

**Table des figures**

## 1. Introduction

Ce laboratoire est divisé en deux parties. l'une dédié aux tests de performance, aux tests d'acceptance et au test de gestion des transactions, et l'autre dédiée au mapping objet relationnel.

## 2. Test

### 2.1. Test de performance avec l'outil JMeter

pour la réalisation de cette partie nous avons défini trois scénario. chaque scénario comporte trois plan de test. chaque plan de test va nous permettre de un nombre d'utilisateur simultané qui pourront utilisé l'application. par ailleurs pour chaque sénario nous avons défini un nombre de page fixe pour la pagination des produits.

#### 2.1.1. Exemple de configuration d'un plan de test

— Thread groupe  
c'est ici qu'on va definir le nombre utilisateurs et la durée de la monté en charge.  
dan

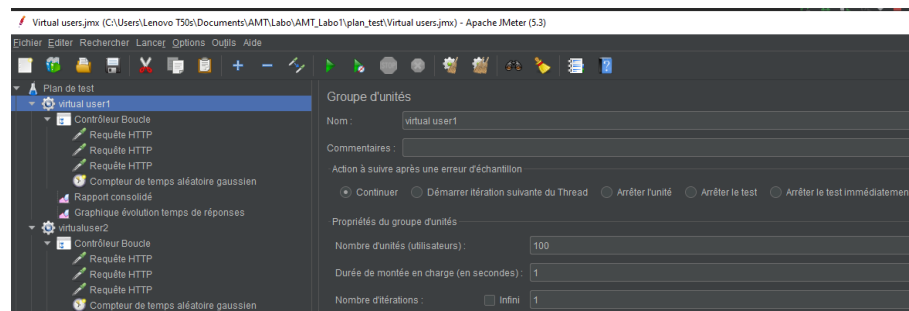


FIGURE 1 – thread group avec 100 utilisateur simultané

— controleur logique  
Se contrôleur logique permet de signifier que chaque utilisateur virtuelle va effectuer x fois le scénario suivant. dans notre cas nous avons pris  $x = 100$  ;

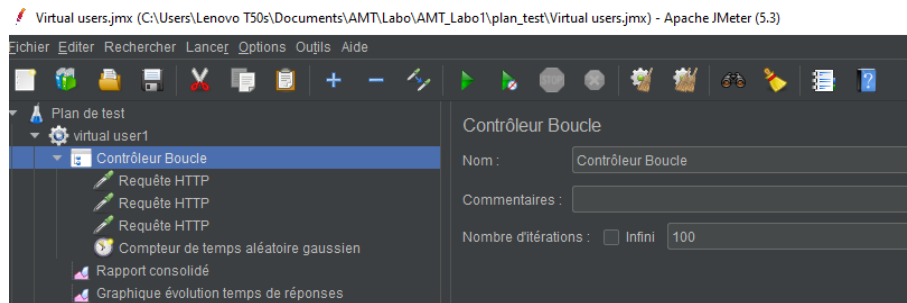


FIGURE 2 – Controleur logique

#### — Sampler

Permet de définir le type de requête échangé avec le serveur (HTTP, FTP, ..) et de les émettre.

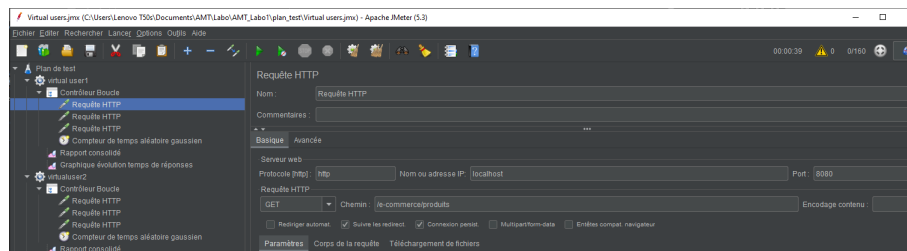


FIGURE 3 – Controleur logique

Dans notre cas on va signaler à JMeter que l'application écoute sur localhost sur le port 8080 et elle écoute sur url /e-commerce/produits. l'idée pour nous est de tester d'image que peut supporter l'application et la pagination.

#### — Listener

Elle permet de récupérer et d'afficher les résultats des tests (fichier, graphe)



FIGURE 4 – listener pour afficher des resultat agregé

Avec ce listener on peut observer le temps de réponses min, max et moyen mais également taux d'erreur.

#### — listener tree view

Elle permet d’observer chaque requête et chaque réponse. et permet également de tracer l’évolution des tests.

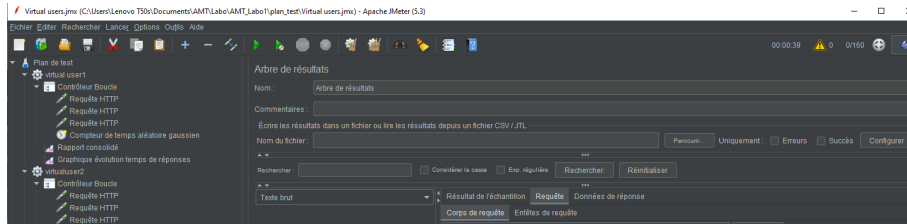


FIGURE 5 – listener pour tracer des requêtes

Pour résumer le plan de test on aura au départ 100 utilisateurs virtuels en parallèle chaque utilisateur virtuel va faire 100 fois la boucle et à chaque itération va faire une requête vers le compteur puis on aura des résultats.

Afin de mesurer l’impact du nombre d’utilisateurs virtuels nous allons effectuer trois plans de test avec respectivement 10, 50 et 100 utilisateurs.

## 2.2. Analyse sur l’impact du nombre d’utilisateurs virtuels

— Résultats avec 10 utilisateurs

Rapport consolidé

Nom :

Rapport consolidé

Commentaires :

Écrire les résultats dans un fichier ou lire les résultats depuis un fichier CSV / JTL

Nom du fichier :

Persoudre...

Uniquement: ☐ Erreurs ☐ Succès ☐ Configurer

Libellé	# Echantillons	Moyenne	Min	Max	Ecart type	% Erreur	Débit	Ko/sec reçus	Ko/sec émis	Moy. octets
Requête HTTP	3000	146	3	773	181.05	0.00%	54.8/sec	427.41	7.43	7980
TOTAL	3000	146	3	773	181.05	0.00%	54.8/sec	427.41	7.43	7980

FIGURE 6 – résultat plan de test pour 10 users

— résultats avec 50 utilisateurs

Rapport consolidé

Nom :

Rapport consolidé

Commentaires :

Écrire les résultats dans un fichier ou lire les résultats depuis un fichier CSV / JTL

Nom du fichier :

Parcourir...

Uniquement : ☐ Erreurs ☐ Succès

Libellé	# Echantillons	Moyenne	Min	Max	Ecart type	% Erreur	Débit	Ko/sec reçus	Ko/sec émis	Moy. octets
Requête HTTP	15000	146	3	805	181.51	0.00%	273.8/sec	2133.88	37.08	7980.4
TOTAL	15000	146	3	805	181.51	0.00%	273.8/sec	2133.88	37.08	7980.4

FIGURE 7 – résultat plan de test pour 50 users

— résultats avec 100 utilisateurs

Rapport consolidé

Nom :

Rapport consolidé

Commentaires :

Écrire les résultats dans un fichier ou lire les résultats depuis un fichier CSV / JTL

Nom du fichier :

Parcourir...

Uniquement : ☐ Erreurs ☐ Succès

Configurer

Libellé	# Echantillons	Moyenne	Min	Max	Ecart type	% Erreur	Débit	Ko/sec reçus	Ko/sec émis	Moy octets
Requête HTTP sur /...	10000	123	2	880	177.99	0.00%	153.7/sec	1087.64	20.71	7247.0
Requête HTTP	20000	140	0	41328	1721.86	50.00%	306.9/sec	1083.63	40.91	3615.5
TOTAL	30000	134	0	41328	1409.67	33.33%	460.2/sec	2169.07	61.58	4826.0

FIGURE 8 – resultat plan de test pour 100 user

aux regards des différents conclusion on peut dire que nombre d'utilisateur virtuel peut avoir un impact sur le bon fonctionnement de l'application. on voit bien qu'à 100 utilisateurs le temps d'attente est plus grand et le taux d'erreur est différent de zéro.

pour la suite on va tester l'impact de la pagination. pour se tester nous allons définir trois plans de test avec un nombre d'utilisateurs virtuel fixe(20). par contre nous allons faire varier le nombre de pages

## 2.3. analyse sur l'impact de la variation des pages

- résultat avec 30 pages
- résultats avec 100 utilisateurs

Rapport consolidé

Nom :

Rapport consolidé

Commentaires :

Écrire les résultats dans un fichier ou lire les résultats depuis un fichier CSV / JTL

Nom du fichier :

Parcourir...

Uniquement : ☐ Erreurs ☐ Succès

Configurer

Libellé	# Echantillons	Moyenne	Min	Max	Ecart type	% Erreur	Débit	Ko/sec reçus	Ko/sec émis	Moy octets
Requête HTTP	6000	75	2	893	76.02	0.00%	184.3/sec	1595.28	24.96	8863
TOTAL	6000	75	2	893	76.02	0.00%	184.3/sec	1595.28	24.96	8863

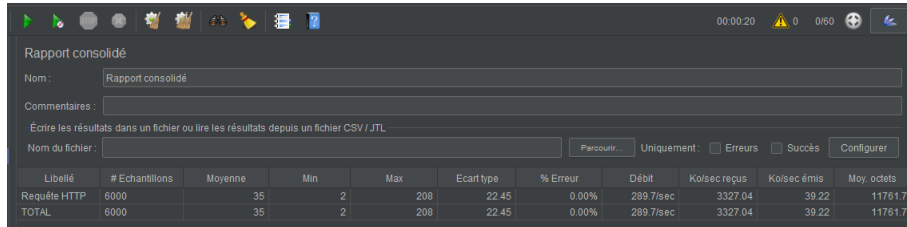
FIGURE 9 – resultat pagination avec 30 pages chaque page comporte 1 produit

- analyse avec 2 pages
- résultats avec 100 utilisateurs

FIGURE 10 – resultat avec pagination de 2 pages chaque page comportant 15 produits

- resultat d'analyse avec 1 page

— résultats avec 100 utilisateurs



Libellé	# Echantillons	Moyenne	Min	Max	Ecart type	% Erreur	Débit	Ko/sec reçus	Ko/sec émis	Moy. octets
Requête HTTP	6000	35	2	208	22.45	0.00%	289.7/sec	3327.04	39.22	11761.7
TOTAL	6000	35	2	208	22.45	0.00%	289.7/sec	3327.04	39.22	11761.7

FIGURE 11 – resultat pagination avec 1 page accueillant 30 produits

on peut observer au regard de ces différents résultats on constate que le vari en fonction du nombre de page. l'idée serai de mèttre un grand nombre de produit su une même page.

En conclusion on constat que plusieurs facteurs peuvent modifier la bon déroulement d'une application . il serai important de toujours faire un test de performance pour améliorer le débit et détecter les accès concurrents qui n'ont pas pu être détecter pendant le développement.

## 2.4. Test d'acceptation avec l'outil Selenium

Selenium est un outil qui permet d'effectuer des test d'acceptance automatisé d'un projet. Selenium n'est pas seulement un outil simple mais une suite d'outil chacun répondant à un besoin particulier.

Pour cette partie du laboratoire nous allons utiliser la partie selenium webdriver.

### 2.4.1. Installation de selenium

dans un environnement windows il faut télécharger l'executable chromedriver et le déposer dans un sous dossier du projet.

— test d'acceptance pour le login

Pendant la réalisation on à rencontrer un souci de configuration. la partie à été developé mais la compilation ne fonctionnet pas.

— implémentation pour la gestion du login

```

public class LoginPage extends AbstractEcommercePage {

    By tfUserName = By.id("userName");
    By tfPasswordLocator = By.id("inputPassword");
    By bSignInLocator = By.id("bSignIn");

    public LoginPage(WebDriver driver) {
        super(driver);

        // Check that we're on the right page.
        if (!"Login Page".equals(driver.getTitle())) {
            throw new IllegalStateException("This is not the correct page");
        }
    }

    public LoginPage typeUserName(String username) {
        driver.findElement(tfUserName).sendKeys(username);
        return this;
    }

    public LoginPage typePassword(String password) {
        driver.findElement(tfPasswordLocator).sendKeys(password);
        return this;
    }

    public Page submitForm(Class<? extends Page> expectedPageClass) {
        driver.findElement(bSignInLocator).click();
        Page targetPage = null;
        try {
            targetPage = expectedPageClass.getConstructor(WebDriver.class).newInstance(driver);
        } catch (Exception ex) {
            Logger.getLogger(LoginPage.class.getName()).log(Level.SEVERE, msg: null, ex);
            throw new RuntimeException("Exception when using reflection: " + ex.getMessage());
        }
        return targetPage;
    }

    public LoginPage submitFormExpectingFailure() {
        driver.findElement(bSignInLocator).click();
        return this; //new LoginPage(driver);
    }
}

```

FIGURE 12 – gestion de test pour le login

— implémentation pour la validation des test effectuer



```

public class AmtEcommercePerformanceTest {

    private String baseUrl = "http://localhost:8080/e-commerce/checkout";
    private WebDriver driver;

    @Before
    public void openBrowser() {
        //driver = new FirefoxDriver();
        System.setProperty("webdriver.chrome.driver", "/chromedriver/chromedriver.exe");
        driver = new ChromeDriver();
    }

    @Test
    @ProbeTest(tags = "WebUI")
    public void itShouldNotBePossibleToSignInWithAnInvalidEmail() {
        driver.get(baseUrl);
        LoginPage loginPage = new LoginPage(driver);
        loginPage.typeUserName("this is not a valid user name");
        loginPage.typePassword("any password");
        loginPage.submitFormExpectingFailure();
    }

    @Test
    @ProbeTest(tags = "WebUI")
    public void successfulSignInShouldBringUserToHomePage() {
        driver.get(baseUrl);
        LoginPage loginPage = new LoginPage(driver);
        loginPage.typeUserName("olivier");
        loginPage.typePassword("any password");
        HomePage homePage = (HomePage)loginPage.submitForm(HomePage.class);
    }

    @Test
    @ProbeTest(tags = "WebUI")
    public void aUserTryingToGetToAboutPageShouldBeRedirectedThereAfterSignIn() {
        driver.get(baseUrl + "/e-commerce/index");
        LoginPage loginPage = new LoginPage(driver);
        loginPage.typeUserName("olivier");
        loginPage.typePassword("any password");
    }
}

```

FIGURE 13 – gestion de test pour le login

pour cette partie nous manquer un peut de temps pour resoudre les erreurs  
 lier à la configuration

## Références

<https://github.com/SoftEng-HEIGVD/Teaching-HEIGVD-AMT-Transactions-with-EJB.git>  
<https://docs.oracle.com/javase/5/tutorial/doc/bncij.html>

<https://www.youtube.com/playlist?list=PLfKkysTy70QZ619qRZfFMov0fT4o9oLGJ>  
[https://github.com/SoftEng-HEIGVD/Teaching-HEIGVD-AMT-2019-](https://github.com/SoftEng-HEIGVD/Teaching-HEIGVD-AMT-2019-Main.git)  
Main.git