Healenium: L'IA qui Révolutionne la Gestion des Localisateurs Dynamiques avec Selenium

1. **Problématique **
2. **Qu'est-ce que HEALENIUM ?**
3. **Avantages de l'utilisation de HEALENIUM**
4. **Configuration et Installation**
5. **Conclusion **

Problématique

Tous les tests fonctionnels automatisés d'une application web finissent tôt ou tard par rencontrer la redoutable exception "NoSuchElementException" en raison des modifications de l'interface web et des localisateurs incorrects ou dynamiques : une source de frustration pour l'automaticien.

En conséquence, le test KO et le flux régulier du pipeline d'automatisation est interrompu. Cette problématique récurrente impacte considérablement la fiabilité des tests tout au long des sprints, nécessitant une solution robuste pour assurer la stabilité et la continuité des tests dans un environnement agile en constante évolution.

Les automaticiens doivent consacrer beaucoup d'efforts à la rédaction de scripts et à la réalisation des modifications nécessaires.

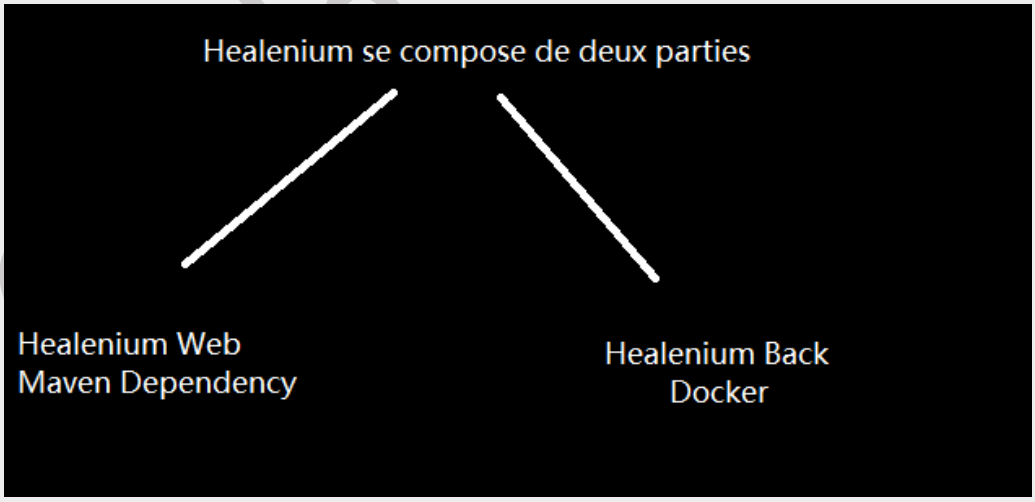
Et si ces changements pouvaient être détectés et effectués automatiquement pendant l'exécution ? Oui, c'est possible avec l'aide de Healenium.

Dans cet article, je vais présenter une vue d'ensemble d'une bibliothèque qui peut remédier à ce problème appelée Healenium, ses capacités et son objectif d'utilisation.

Qu'est-ce que Healenium ?

Healenium est une bibliothèque open source alimentée par l'intelligence artificielle qui améliore la stabilité des tests basés avec Selenium. Elle gère automatiquement les changements d'éléments web mis à jour et contribue à surmonter le problème de l'instabilité des tests automatisés de l'interface web en utilisant un mécanisme d'autoréparation.

Dans des situations pratiques, les applications web et mobiles subissent des mises à jour régulières à chaque sprint, entraînant parfois des modifications de localisateurs dans le DOM. Healenium utilise un algorithme de machine-learning (ML) pour analyser l'état actuel de la page, traiter l'exception NoSuchElement et corriger les tests défaillants en cours d'exécution. En fait il remplace le localisateur défaillant par une nouvelle valeur qui correspond de manière optimale (ayant le score le plus élevé), permettant ainsi une action réussie avec le nouvel élément. Après l'exécution du test, Healenium génère un rapport détaillé, incluant les localisateurs corrigés et des captures d'écran.



Avantages de l'utilisation de HEALENIUM

Healenium apporte une valeur significative à l'entreprise :

- Il réduit le temps de maintenance du code d'automatisation des tests, permettant à l'équipe de se concentrer sur l'augmentation de la couverture.

- Comme l'utilisation de Healenium s'effectue en temps d'exécution, les tests automatisés de bout en bout seront stables et les changements d'interface utilisateur ne les affecteront plus.
- Le pipeline d'intégration continue ne sera en échec qu'en cas de problèmes liés au produit.

Configuration et Installation

Veuillez suivre les instructions de configuration.

Prérequis : Vous devrez disposer d'un projet Maven d'automatisation avec Selenium prêt à mettre en œuvre Healenium.

Je vous recommande d'utiliser l'application web suivante **https://elenastepuro.github.io/test_env/index.html** .

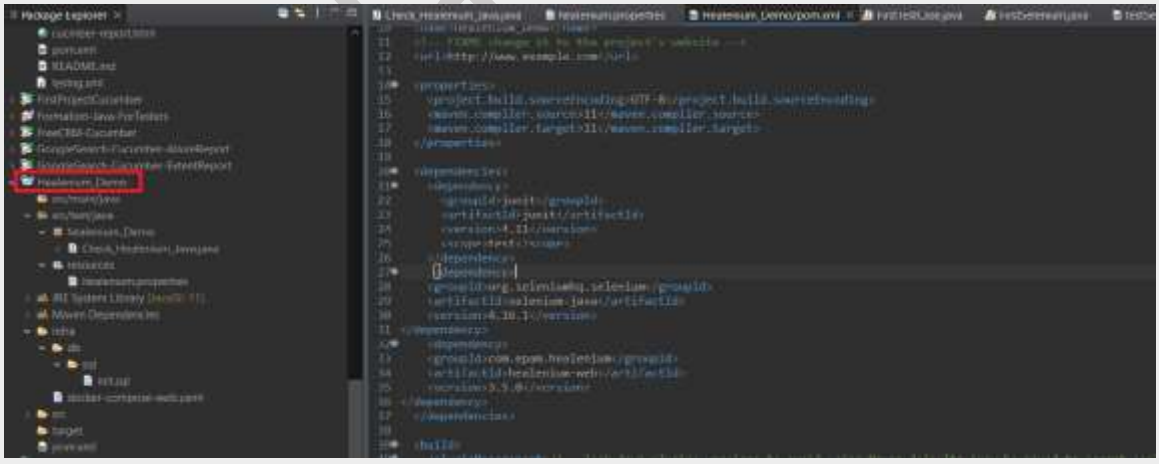
Le site web possède des éléments dynamiques et ses localisateurs peuvent être modifiés en cliquant sur le bouton "Change Locators" sur le site. Cela vous aidera à comprendre clairement la logique de tests, le fonctionnement de Healenium et sa valeur ajoutée.

Pour chacune des étapes, j'ai joint l'image correspondante pour votre référence.

1. Ajoutez la dépendance Healenium dans le fichier pom.xml de votre projet maven en copiant et collant les lignes ci-dessous dans la balise de dépendances :

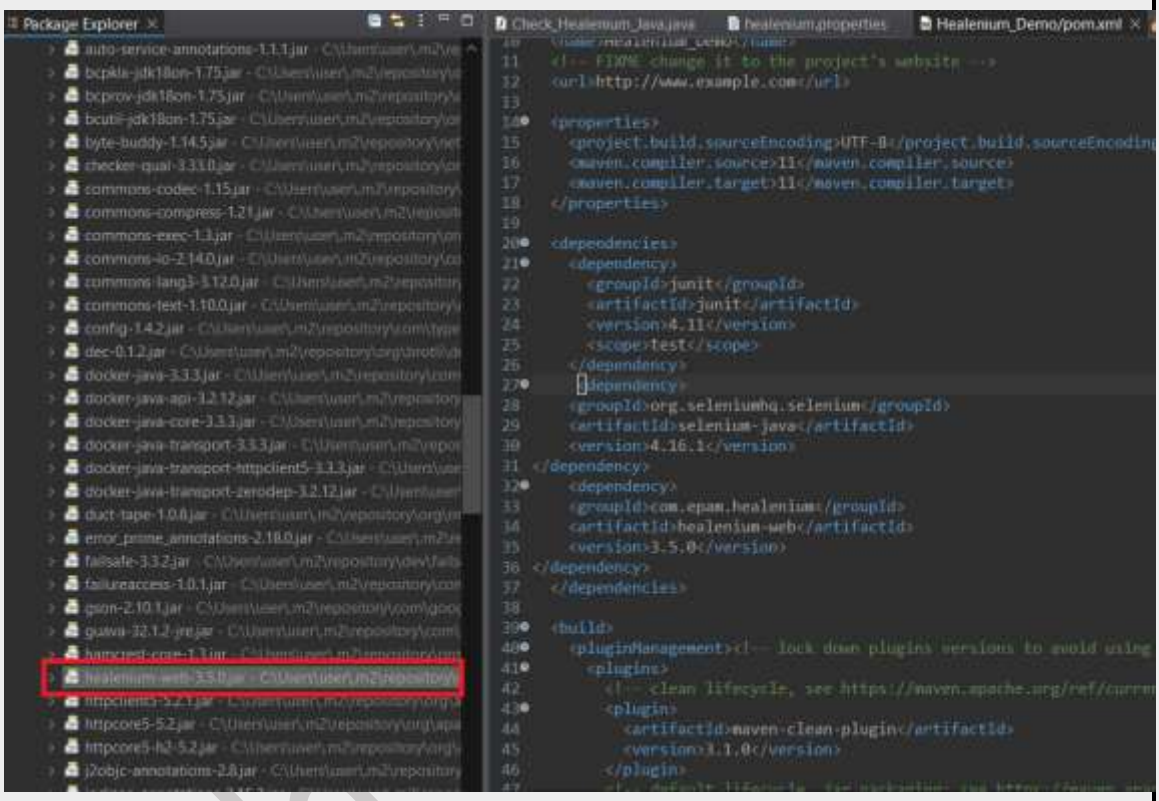
```
<dependency>
    <groupId>com.epam.healenium</groupId>
    <artifactId>healenium-web</artifactId>
    <version>3.5.0</version>
</dependency>
```

Assurez-vous de coller ces lignes à l'intérieur de la section `<dependencies>` de votre fichier pom.xml. Cette dépendance permettra à votre projet d'utiliser la version spécifiée de Healenium pour la stabilisation des tests Selenium.

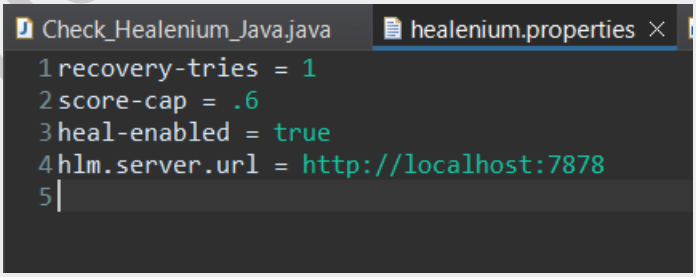


2. Assurez-vous de vérifier le répertoire Maven dependencies de votre projet pour confirmer que la bibliothèque healenium/epam a été

correctement ajoutée après avoir enregistré les modifications dans le fichier pom.xml. Cela confirmera que la dépendance a été intégrée avec succès dans votre projet.



3. Créez un fichier 'healenium.properties' dans le package resources sous src/test/java de votre projet et copiez les lignes ci-dessous :



recovery-tries : Le nombre de tentatives que l'algorithme effectuera pour découvrir un localisateur correspondant.

score-cap : Le score minimum de correspondance requis pour que le localisateur détecté soit accepté (50 % est représenté par le nombre 0,5).

heal-enabled : Un interrupteur bascule qui active ou désactive la fonction de guérison. Les valeurs acceptées sont True (Vrai) et False (Faux).

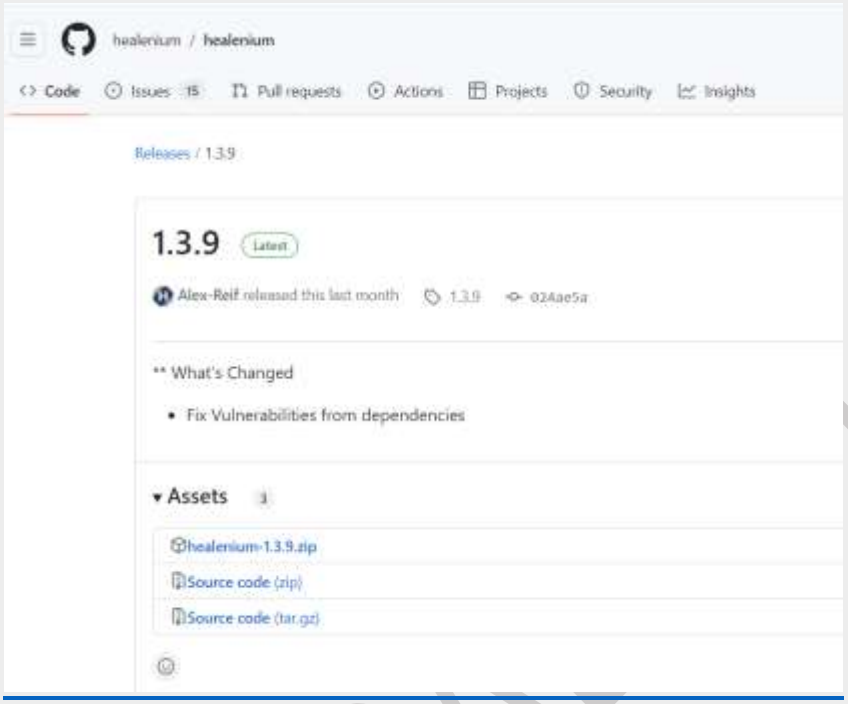
Hlm.server.url : L'URL du serveur de conteneurs Docker établi lors de la configuration du backend et le port du serveur

4. Téléchargez et installez Docker sur votre système. Vous pouvez installer Docker en utilisant le lien suivant :

https://docs.docker.com/desktop/install/windows-install/

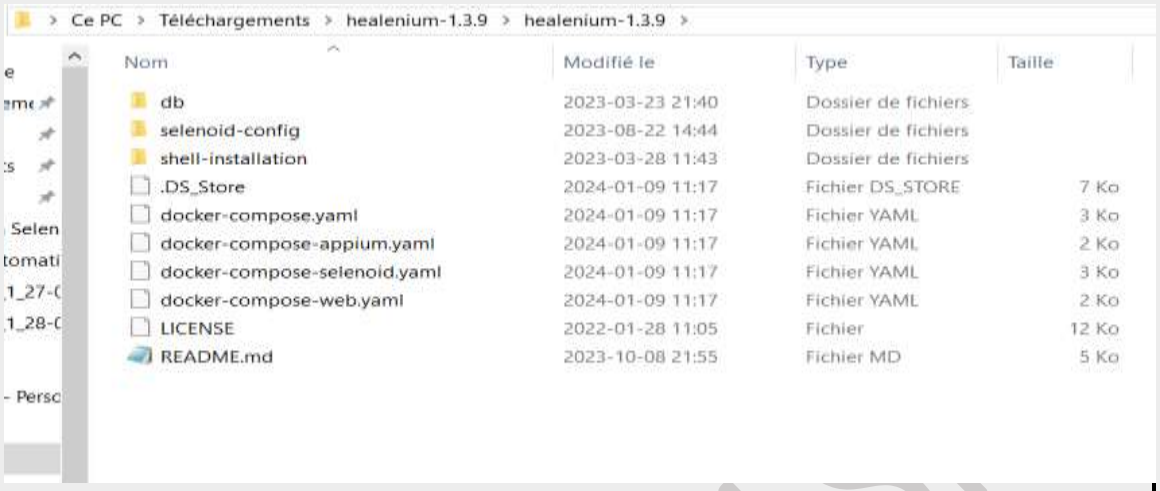
5. Téléchargez healenium.zip en utilisant l'URL ci-dessous :

https://github.com/healenium/healenium/releases/download/1.3.9/healenium-1.3.9.zip

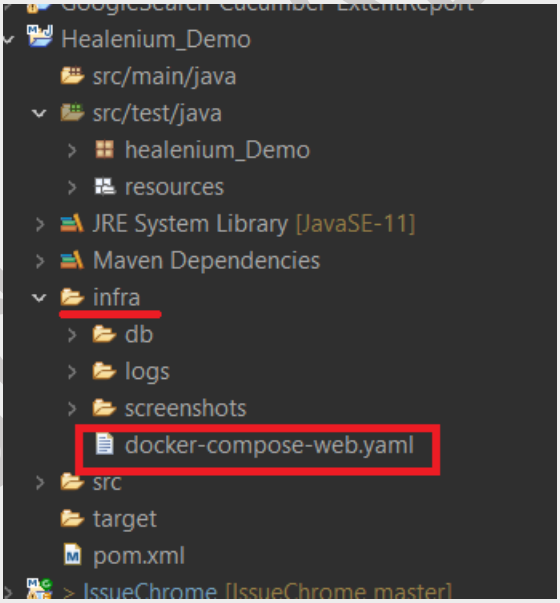


Veuillez noter que les versions de Healenium peuvent changer. Vous pouvez utiliser cette URL ou vérifier également la dernière version dans le dépôt GitHub de healenium.io. L'URL correspondante est **https://github.com/healenium.**

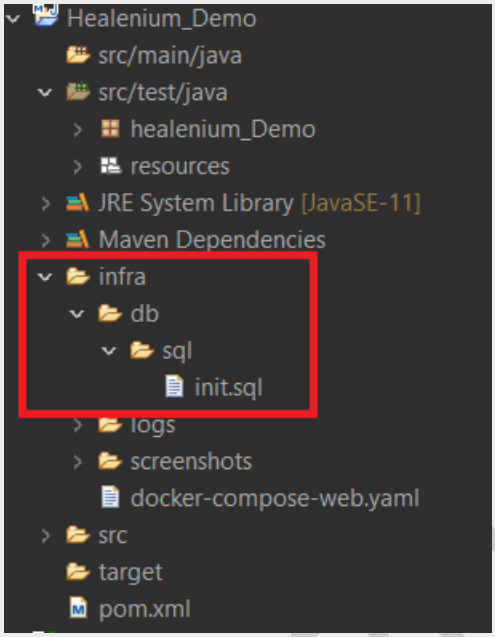
6. Dezippez le contenu du fichier zip. Après l'extraction, voici à quoi ressemblera la structure du projet :



7. Dans le dossier racine de votre projet, créez un nouveau dossier et nommez-le 'infra'. À partir du contenu extrait du fichier zip, copiez le fichier 'docker-compose-web.yaml' et collez-le à l'intérieur du dossier 'infra'.



8. À l'intérieur du dossier 'infra', créez un nouveau dossier 'db'. À l'intérieur de 'db', créez un nouveau dossier 'sql'. Copiez et collez le fichier 'init.sql' à partir du contenu extrait du dossier db dans le dossier 'sql' du projet Java.

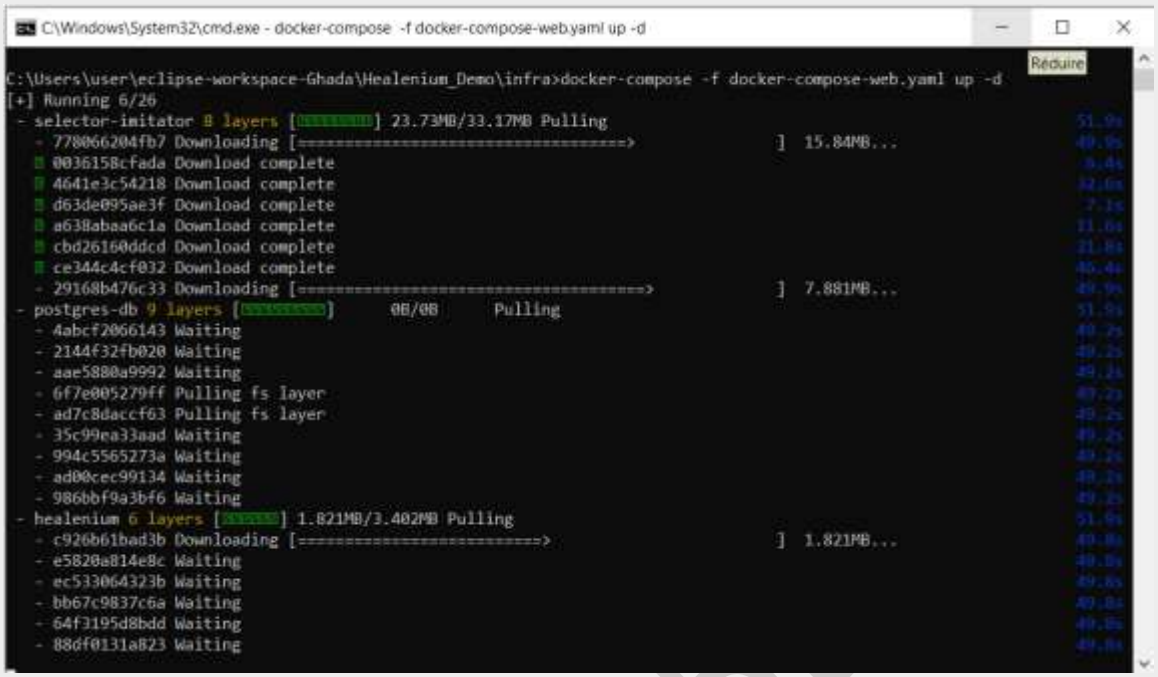


9. Initialiser une instance du pilote SelfHealingDriver

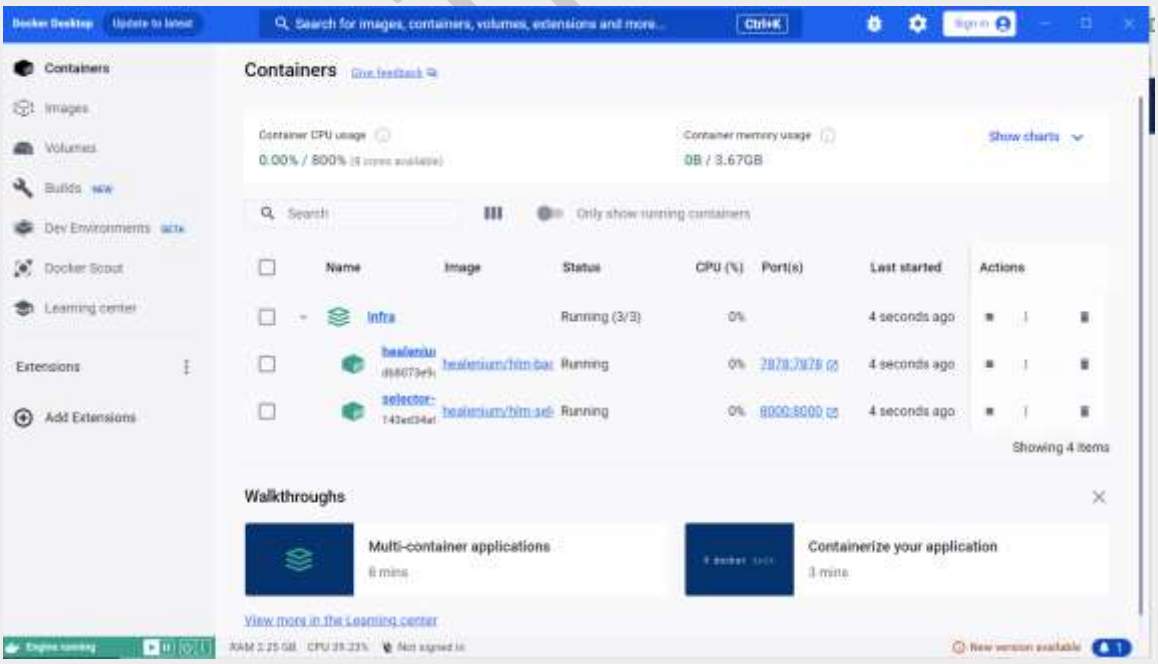
```
//declare delegate
  WebDriver delegate = new ChromeDriver();
  //create Self-healing driver
  SelfHealingDriver driver = SelfHealingDriver.create(delegate);
```

10. Lancez les services Docker à l'aide de la commande suivante dans le terminal. Veuillez noter que vous devez être à l'intérieur du dossier 'infra' pour exécuter la commande :

docker-compose -f docker-compose-web.yaml up –d



Une fois que vous avez exécuté cette commande, vous pourrez voir le conteneur 'infra' en cours d'exécution dans Docker, comme illustré dans l'image ci-dessous.

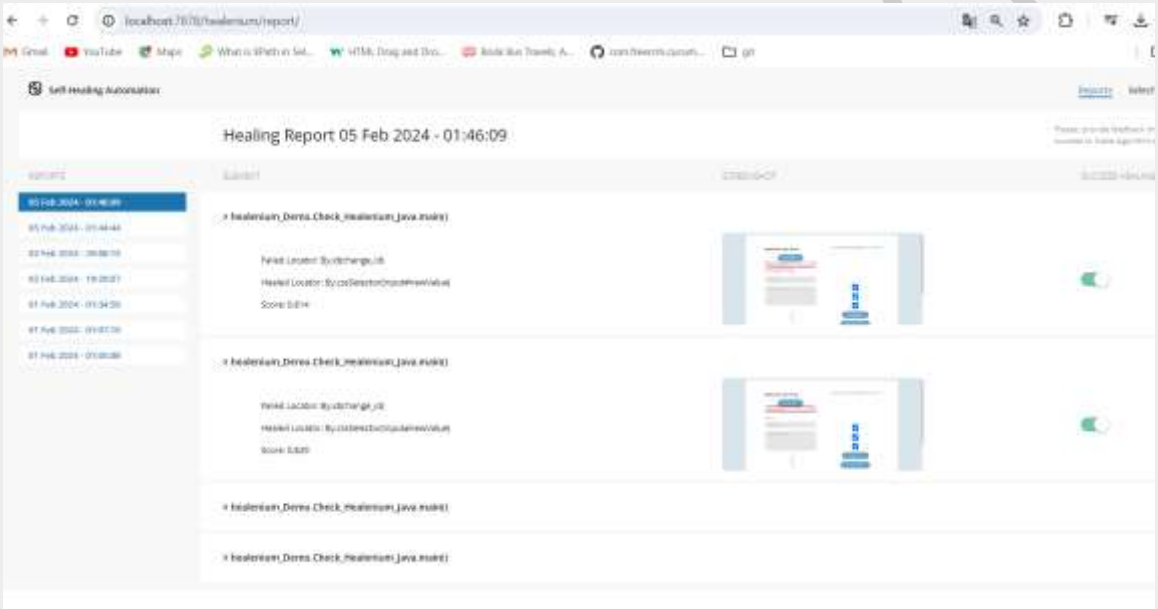


11. Vérifiez l'URL **http://localhost:7878/healenium/** pour vérifier si le backend de Healenium est en cours d'exécution.

12.Le rapport de Healenium peut être consulté à l'aide de l'URL suivante

http://localhost:7878/healenium/report

Ce rapport vous fournira un résumé des éléments qui n'ont pas été trouvés, des localisateurs par lesquels ils ont été remplacés par Healenium, de la capture d'écran, et des commentaires indiquant si cela a été réussi ou non. Il contiendra également une liste de tous les localisateurs utilisés dans le script.



Lorsque vous exécutez le script pour la première fois, Healenium stocke tous les localisateurs dans le backend et l'utilise comme référence pour les exécutions suivantes. Lorsque l'interface utilisateur change, le script d'automatisation peut échouer à moins que le script d'automatisation ne soit mis à jour manuellement. Healenium comprend automatiquement que l'élément n'est pas trouvé. Cela déclenche l'algorithme de machine learning, il transmet l'état actuel de la page, obtient les chemins des localisateurs précédemment réussis à partir du stockage, les compare et génère une liste de localisateurs récupérés.

Pour plus d'informations, vous pouvez consulter le site Web suivant :

https://healenium.io/

Conclusion

En conclusion, Healenium représente une avancée majeure dans l'écosystème des tests d'automatisation, particulièrement dans le contexte des projets Selenium. Healenium surmonte les défis de maintenance des scripts de test, libérant du temps pour étendre stratégiquement la couverture des tests.

L'intégration de l'intelligence artificielle avec Selenium est un aspect notable de Healenium. Grâce à ses algorithmes de machine learning, il peut dynamiquement ajuster les localisateurs lors de l'exécution, anticipant et s'adaptant aux changements de l'interface utilisateur. Cette fusion de Selenium avec l'intelligence artificielle marque une nouvelle ère dans l'automatisation des tests, offrant une stabilité accrue et une résilience face aux évolutions fréquentes des applications.

Ceci garanti la fiabilité des tests automatisés de bout en bout, transformant les défis liés à l'évolution de l'interface utilisateur en opportunités d'amélioration continue. Healenium, en simplifiant la maintenance des scripts, améliore la qualité des tests et contribue significativement à la réussite des initiatives d'automatisation. En embrassant Healenium, les équipes peuvent naviguer avec assurance au sein des dynamiques incessantes des interfaces utilisateur, redéfinissant ainsi les normes de l'automatisation des tests.

Lien GitHub : https://github.com/GhadaTrabelsi/Healenium_Demo