



SELENIUM & JMETER RAPPORT

Management de qualité

Dr. Driss Essabar



25 DECEMBRE 2024

EL ASSOULI NOUHAYLA
DIDI ALAOUI LATIFA
ROUITA CHAIMAE
TIZGHA NIHAL

Tests Unitaires

➤ Résultats des Tests JUnit pour le Projet :

The screenshot shows the Android Studio interface with the project navigation bar at the top. The main area displays the code for `PlanteServiceTest.java`. Below the code editor is a table showing the execution results for the `PlanteServiceTest` class:

Test Method	Time	Details
<code>testGetAllPlantes()</code>	1 sec 891 ms	
<code>testGenererRecommandations()</code>	8 ms	
<code>testUpdatePlante()</code>	9 ms	
<code>testDeletePlante()</code>	5 ms	
<code>testAddPlante()</code>	5 ms	
<code>testRechercheAvancee()</code>	4 ms	
<code>testDeletePlanteNotFound()</code>	8 ms	
<code>testGetDetails()</code>	6 ms	

At the bottom of the interface, the path `Backend > src > test > java > com > example > Plantes > Service > PlanteServiceTest` is visible along with the timestamp `19:14 CRLF UTF-8 4 spaces`.

The screenshot shows the Android Studio interface with the project navigation bar at the top. The main area displays the code for `CommentaireServiceTest.java`. Below the code editor is a table showing the execution results for the `CommentaireServiceTest` class:

Test Method	Time	Details
<code>testGetCommentairesByPlantId()</code>	1 sec 707 ms	
<code>testAjouterCommentaire()</code>	6 ms	

At the bottom of the interface, the path `Backend > src > test > java > com > example > Plantes > Service > CommentaireServiceTest` is visible along with the timestamp `10:55 CRLF UTF-8 4 spaces`.

The screenshot shows the Android Studio interface with the project navigation bar at the top. The main area displays a Java test class, `PlantRepositoryTest`, with code for testing plant repository methods. Below the code, the run tool window shows the execution results for `PlantRepositoryTest`. It lists three test methods: `testFindAllWithComments()`, `testFindByAllIgnoreCase()`, and `testFindByUsesContaining()`. All tests are marked with a green checkmark, indicating they passed. The total execution time for all tests is 816 ms.

```
public class AdminPlanteControllerTest {  
    public void testDeletePlante() throws Exception {  
        .andExpect(status().isNoContent());  
    }  
  
    @Test  
    public void testGetPlanteById() throws Exception {  
        Plante plante = Plante.builder()  
            .id(1L)  
            .name("Rose")  
    }  
}
```

The screenshot shows the Android Studio interface with the project navigation bar at the top. The main area displays a Java test class, `CommentaireRepositoryTest`, with code for testing comment repository methods. Below the code, the run tool window shows the execution results for `CommentaireRepositoryTest`. It lists two test methods: `testFindByPlantId()` and `testFindByPlantId_NoComments()`. Both tests are marked with a green checkmark, indicating they passed. The total execution time for all tests is 746 ms.

```
public class AdminPlanteControllerTest {  
    public void testDeletePlante() throws Exception {  
        .andExpect(status().isNoContent());  
    }  
  
    @Test  
    public void testGetPlanteById() throws Exception {  
        Plante plante = Plante.builder()  
            .id(1L)  
            .name("Rose")  
    }  
}
```

The screenshot shows the Android Studio interface with the project navigation bar at the top. The main area displays a Java test class, `RecommandationControllerTest`, with code for testing recommendation controller methods. Below the code, the run tool window shows the execution results for `RecommandationControllerTest`. It lists one test method: `testObtenirRecommandations()`. The test is marked with a green checkmark, indicating it passed. The total execution time for the test is 331 ms.

```
public class AdminPlanteControllerTest {  
    public void testDeletePlante() throws Exception {  
        .andExpect(status().isNoContent());  
    }  
  
    @Test  
    public void testGetPlanteById() throws Exception {  
        Plante plante = Plante.builder()  
            .id(1L)  
            .name("Rose")  
    }  
}
```

The screenshot shows the Android Studio interface with the project navigation bar at the top. Below it is a code editor with three tabs: PlanteServiceTest.java, AdminPlanteControllerTest.java, and pom.xml (CataloguePlantes). The code editor displays Java test code for the PlanteControllerTest class. The bottom half of the screen is a test results summary. It lists the PlanteControllerTest class under the 'Run' tab. The results show 6 green checkmarks indicating successful tests: testGetDetailsPlantNotFound(), testGetDetailsPlantExists(), test GetAllPlantes(), testGenererRecommandations(), and testRechercheAvancee(). The total execution time is 2 sec 959 ms.

```
public class AdminPlanteControllerTest {  
    public void testDeletePlante() throws Exception {  
        .andExpect(status().isNoContent());  
    }  
  
    @Test  
    public void testGetPlanteById() throws Exception {  
        Plante plante = Plante.builder()  
            .id(1L)  
            .name("Rose")  
    }  
}
```

This screenshot shows the same Android Studio environment as the first one, but the test results are for the FileUploadControllerTest class. The results table shows 3 green checkmarks: testUploadFile(), testgetFile(), and testGetFileNotFoundException(). The total execution time is 2 sec 467 ms.

```
public class AdminPlanteControllerTest {  
    public void testDeletePlante() throws Exception {  
        .andExpect(status().isNoContent());  
    }  
  
    @Test  
    public void testGetPlanteById() throws Exception {  
        Plante plante = Plante.builder()  
            .id(1L)  
            .name("Rose")  
    }  
}
```

This screenshot shows the test results for the CommentaireControllerTest class. The results table shows 2 green checkmarks: testGetCommentaires() and testAjouterCommentaire(). The total execution time is 3 sec 434 ms.

```
public class AdminPlanteControllerTest {  
    public void testDeletePlante() throws Exception {  
        .andExpect(status().isNoContent());  
    }  
  
    @Test  
    public void testGetPlanteById() throws Exception {  
        Plante plante = Plante.builder()  
            .id(1L)  
            .name("Rose")  
    }  
}
```

The screenshot shows the Android Studio interface with the project navigation bar at the top. The main area displays the code for `AdminPlanteControllerTest.java`. Below the code editor is a 'Run' section showing the execution results for `AdminPlanteControllerTest`. The results table includes columns for the test name, duration, and execution status (green checkmark). The bottom status bar indicates the current time as 10:55 and file encoding as CRLF.

```
public class AdminPlanteControllerTest {  
    public void testDeletePlante() throws Exception {  
        .andExpect(status().isNoContent());  
    }  
  
    @Test  
    public void testGetPlanteById() throws Exception {  
        Plante plante = Plante.builder()  
            .id(1L)  
            .name("Rose")  
    }  
}
```

Test	Time	Status
testGetAllPlantes()	6 sec 815 ms	✓
testGetPlanteById()	49 ms	✓
testUpdatePlante()	137 ms	✓
testDeletePlante()	20 ms	✓
testAddPlante()	28 ms	✓

The screenshot shows the Android Studio interface with the project navigation bar at the top. The main area displays the code for `PlanteServiceTest.java`. Below the code editor is a 'Run' section showing the execution results for `RecommandationServiceTest`. The results table includes columns for the test name, duration, and execution status (green checkmark). The bottom status bar indicates the current time as 19:14 and file encoding as CRLF.

```
import java.util.Optional;  
import static org.junit.jupiter.api.Assertions.*;  
import static org.mockito.Mockito.*;  
  
public class PlanteServiceTest {  
    @Mock 17 usages  
    private PlantRepository planteRepository; // Mocking the PlantRepository  
}
```

Test	Time	Status
testGenererRecommendations_WithMedicalHistory()	1sec 579 ms	✓
testGenererRecommendations_WithPreferences()	7 ms	✓
testGenererRecommendations_NoRecommendations()	5 ms	✓

➤ Scénarios de Test Automatisé avec Selenium IDE :

ID	Scénario de Test	Préconditions	Étapes	Résultat Attendu	Vérification
1	Vérifier que les boutons Home, Plants, Recommendations, et Admin sont affichés et fonctionnels.	Accéder à la page d'accueil.	1. Accéder à la page d'accueil. 2. Cliquer sur chaque bouton.	Navigation correcte vers les pages correspondantes	verifyElement
2	Vérifier que la page Plants affiche la liste des plantes avec images et boutons Read More.	La page d'accueil est chargée.	1. Cliquer sur le bouton Plants depuis la page d'accueil.	Chargement de la liste des plantes avec toutes les informations nécessaires.	verifyElement
3	Vérifier que la page Recommendations contient les champs pour les besoins, préférences, et antécédents médicaux.	La page d'accueil est chargée.	1. Cliquer sur le bouton Recommendations depuis la page d'accueil.	Affichage des champs et du bouton Obtenir Résultat.	verifyElement
4	Vérifier que la page Admin redirige vers un formulaire d'authentification	La page d'accueil est chargée.	1. Cliquer sur le bouton Admin depuis la page d'accueil.	Chargement de la page d'authentification .	verifyElement
5	Vérifier que la recherche fonctionne pour des mots-clés valides dans Plants.	La page Plants est chargée.	1. Entrer un mot-clé correspondant à une plante existante. 2. Cliquer sur le bouton Rechercher.	Affichage des plantes correspondantes	assertText
6	Vérifier que la recherche gère les mots-clés non valides dans Plants.	La page Plants est chargée.	1. Entrer un mot-clé inexistant dans la barre de recherche. 2. Cliquer sur Rechercher.	La plante ne s'affiche pas.	verifyElementNotPresent
7	Vérifier que la recherche retourne les plantes selon les besoins de santé dans Recommendations.	La page Recommendations est chargée.	1. Entrer des informations dans le champ Besoins de Santé. 2. Cliquer sur Obtenir Résultat.	Liste des plantes adaptées aux besoins.	verifyElement, assertText
8	Vérifier que la recherche gère les champs vides dans Recommendations.	La page Recommendations est chargée.	1. Laisser tous les champs vides. 2. Cliquer sur Obtenir Résultat.	Aucune plante n'est retournée.	assertElementPresent
9	Vérifier que le bouton Read More redirige vers la bonne page de détails des plantes.	La page Plants est chargée.	1. Cliquer sur le bouton Read More pour une plante spécifique.	Chargement de la page contenant les informations détaillées de la plante.	assertElementPresent
10	Vérifier que toutes	La page de détails	1. Comparer les	Toutes les	assertText,

	les informations des plantes sont correctes sur la page de détails.	d'une plante est chargée.	informations affichées (image, description, etc.) avec les données attendues.	informations sont exactes.	storeAttribute, assert, assertElementPresent
	Vérifier que l'authentification réussit avec des informations valides.				
11	Vérifier que l'authentification échoue avec des informations incorrectes.	La page Admin est chargée.	1. Entrer des identifiants valides. 2. Cliquer sur Connexion.	Accès au tableau de bord administrateur.	verifyElement, assertalert
12	Vérifier que le bouton d'ajout de plante redirige vers le formulaire d'ajout de plante.	La page Admin est chargée.	1. Entrer un nom d'utilisateur ou mot de passe invalide. 2. Cliquer sur Connexion.	Affichage d'un message "Invalidcredentials!".	assertAlert
13	Vérifier que la suppression d'une plante fonctionne correctement.	Page Admin chargée.	1. Cliquer sur le bouton Ajouter une plante.	Redirection vers la page de formulaire d'ajout de plante.	assertElementPresent
14	Vérifier que la modification d'une plante redirige vers le formulaire de modification.	Page Admin chargée.	1. Cliquer sur le bouton de suppression pour une plante spécifique. 2. Vérifier que la plante a été supprimée de la liste.	La plante est supprimée de la liste.	verifyElementNotPresent
15	Vérifier que l'utilisateur peut se déconnecter avec succès.	Page Admin chargée.	1. Cliquer sur le bouton Editer pour une plante spécifique.	Redirection vers la page de formulaire de modification de plante.	assertElementPresent
16		Page Admin chargée.	1. Cliquer sur le bouton Logout.	Redirection vers la page de connexion.	assertElementPresent

✓ Résultat du Scénario de Test (1) :

The screenshot shows the Selenium IDE interface with the project 'Test Plantes' open. The 'Tests' panel on the left lists various test cases: admin_page, details, details_plants, edit_plant*, home_page*, logout*, plant, plants, recom, recommandation, and recommandation_page. The 'Log' panel at the bottom contains the following log entries:

Step	Action	Result	Time
9.	open on /recommendations	OK	15:58:42
10.	verifyElementPresent on css=.admin-button	OK	15:58:42
11.	click on css=.admin-button	OK	15:58:43
12.	open on /admin-login	OK	15:58:43
'home_page*' completed successfully			

✓ Résultat du Scénario de Test (2) :

The screenshot shows the Selenium IDE interface with the project 'Test Plantes' open. The 'Tests' panel on the left lists various test cases: admin_page, details, details_plants, edit_plant*, home_page*, logout*, plant, plants*, platn_page*, recom, and recommandation. The 'Log' panel at the bottom contains the following log entries:

Step	Action	Result	Time
4.	verifyElementPresent on css=.plant-card:nth-child(1) > .plant-name	OK	16:58:49
5.	click on css=.plant-card:nth-child(1) > .plant-name	OK	16:58:49
6.	verifyElementPresent on css=.plant-card:nth-child(1) > button	OK	16:58:49
7.	click on css=.plant-card:nth-child(1) > button	OK	16:58:49
'platn_page*' completed successfully			

✓ Résultat du Scénario de Test (3) :

Selenium IDE - Test Plantes*

Project: Test Plantes*

Executing ✓ recommandation_page*

http://localhost:4200

Command	Target	Value
3 ✓ verify element present	css=.nav-button:nth-child(3)	
4 ✓ click	id=besoins-de-sante	
5 ✓ verify element present	id=preferences	
6 ✓ verify element present	id=antecedents-medicaux	
7 ✓ verify element present	css=ng-pristine > button	

Command: verify element present

Target: id=antecedents-medicaux

Value:

Description:

Runs: 1 Failures: 0

Log Reference

```

3. verifyElementPresent on css=.nav-button:nth-child(3) OK
4. Trying to find id=besoins-de-sante... OK
5. verifyElementPresent on id=preferences OK
6. verifyElementPresent on id=antecedents-medicaux OK
7. verifyElementPresent on css=ng-pristine > button OK
'recommandation_page' completed successfully
  
```

✓ Résultat du Scénario de Test (4) :

Selenium IDE - Test Plantes*

Project: Test Plantes*

Tests ✓ admin_page*

home_page

✓ recommandation_page*

http://localhost:4200

Command	Target	Value
3 ✓ verify element present	css=.admin-button	
4 ✓ click	id=username	
5 ✓ click	id=password	
6 ✓ click	css=.submit-btn	
7 ✓ assert alert	Invalid credentials!	

Command: verify element present

Target: css=.admin-button

Value:

Description:

Log Reference

```

3. verifyElementPresent on css=.admin-button OK
4. Trying to find id=username... OK
5. click on id=password OK
6. click on css=.submit-btn OK
7. assertAlert on Invalid credentials! OK
'admin_page' completed successfully
  
```

Selenium IDE - Test Plantes*

Project: Test Plantes*

Tests ✓ admin_page*

home_page

✓ recommandation_page*

http://localhost:4200

Command	Target	Value
1 ✓ open	/home	
2 ✓ set window size	660x654	
3 ✓ verify element present	css=.admin-button	
4 ✓ click	id=username	
5 ✓ click	id=password	

Command: verify element present

Target: css=.admin-button

Value:

Description:

Log Reference

```

3. verifyElementPresent on css=.admin-button OK
4. Trying to find id=username... OK
5. click on id=password OK
6. click on css=.submit-btn OK
7. assertAlert on Invalid credentials! OK
'admin_page' completed successfully
  
```

✓ Résultat du Scénario de Test (5) :

Selenium IDE - Test Plantes*

Project: Test Plantes*

Executing ✓ plants*

http://localhost:4200

Command	Target	Value
✓ open	/home	
✓ set window size	660x654	
✓ click	css=.nav-button:nth-child(2)	
✓ click	css=.search-input	
✓ assert text	css=.search-input	mint

Command: assert text
Target: css=.search-input
Value: mint
Description:

Runs: 1 Failures: 0

Log Reference

```

3. click on css=.nav-button:nth-child(2) OK
4. click on css=.search-input OK
5. assertText on css=.search-input OK
6. click on css=.search-button OK
7. click on css=.plant-card:nth-child(1)> img OK
'plants' completed successfully
  
```

✓ Résultat du Scénario de Test (6) :

Selenium IDE - Test Plantes*

Project: Test Plantes*

Tests ✓ admin_page*, home_page, ✓ plant*, ✓ plants*, ✓ recommandation_page*

http://localhost:4200

Command	Target	Value
✓ click	css=.nav-button:nth-child(2)	
✓ click	css=.search-input	
✓ type	css=.search-input	Agave
✓ click	css=.search-button	
✓ verify element not present	id=.plant-card	

Command
Target
Value
Description

Log Reference

```

3. click on css=.nav-button:nth-child(2) OK
4. click on css=.search-input OK
5. type on css=.search-input with value Agave OK
6. click on css=.search-button OK
7. verifyElementNotPresent on id=.plant-card OK
'plant' completed successfully
  
```

✓ Résultat du Scénario de Test (7) :

Selenium IDE - Test Plantes*

Project: Test Plantes*

Executing ✓ recommandation*

http://localhost:4200

Command	Target	Value
✓ click	id=antecedents-medicaux	
✓ type	id=antecedents-medicaux	Asthme
✓ click	css=.ng-touched > button	
✓ verify element present	css=.result-container	
✓ assert text	css=.plant-title	Plante Recommandée: Menthe

Command
Target
Value
Description

Runs: 1 Failures: 0

Log Reference

```

8. click on id=antecedents-medicaux OK
9. type on id=antecedents-medicaux with value Asthme OK
10. click on css=.ng-touched > button OK
11. verifyElementPresent on css=.result-container OK
12. assertText on css=.plant-title with value Plante Recommandée: Menthe OK
'recommandation' completed successfully
  
```

✓ Résultat du Scénario de Test (8) :

The screenshot shows the Selenium IDE interface with the following details:

- Project:** Test Plantes*
- Test Name:** ✓ admin_page*
- Commands Table:**

	Command	Target	Value
1	✓ open	/recommendation	
2	✓ set window size	660x654	
3	✓ click	css=.nav-button:nth-child(3)	
4	✓ assert element present	id=besoins-de-sante	
5	✓ assert element present	id=preferences	
6	✓ assert element present	id=antecedents-medicaux	
7	✓ click	css=.ng-pristine > button	
- Current Command Form:**
 - Command: click
 - Target: css=.ng-pristine > button
 - Value:
 - Description:
- Log:**
 - 5. assertElementPresent on id=preferences OK
 - 6. assertElementPresent on id=antecedents-medicaux OK
 - 7. click on css=.ng-pristine > button OK
 - 'recom' completed successfully

✓ Résultat du Scénario de Test (9) :

The screenshot shows the Selenium IDE interface with the following details:

- Project:** Test Plantes*
- Test Name:** ✓ details*
- Commands Table:**

	Command	Target	Value
3	✓ click	css=.nav-button:nth-child(2)	
4	✓ assert element present	css=.plant-card	
5	✓ click	css=.plant-card button	
6	✓ open	plant-details/1	
7	✓ assert element present	css=.app-container	
- Current Command Form:**
 - Command: assert element present
 - Target: css=.app-container
 - Value:
 - Description:
- Log:**
 - 3. click on css=.nav-button:nth-child(2) OK
 - 4. assertElementPresent on css=.plant-card OK
 - 5. click on css=.plant-card button OK
 - 6. open on plant-details/1 OK
 - 7. assertElementPresent on css=.app-container OK
 - 'details' completed successfully

✓ Résultat du Scénario de Test (10) :

The screenshot shows the Selenium IDE interface with the following details:

- Project:** Test Plantes*
- Test Name:** ✓ details_plants*
- Commands Table:**

	Command	Target	Value
17	✓ assert element present	css=.comment-input	
18	✓ type	css=.comment-input	safaa
19	✓ assert element present	css=.comment-textarea	
20	✓ type	css=.comment-textarea	good
21	✓ assert element present	css=.add-comment > button	
- Current Command Form:**
 - Command:
 - Target:
 - Value:
 - Description:
- Log:**
 - 17. assertElementPresent on css=.comment-input OK
 - 18. type on css=.comment-input with value safaa OK
 - 19. assertElementPresent on css=.comment-textarea OK
 - 20. type on css=.comment-textarea with value good OK
 - 21. assertElementPresent on css=.add-comment > button OK
 - 'details_plants' completed successfully

✓ Résultat du Scénario de Test (11) :

The screenshot shows the Selenium IDE interface with a project named "Test Plantes". A test case named "admin/login" is selected. The test steps are as follows:

	Command	Target	Value
4	✓ type	id=username	admin
5	✓ click	id=password	
6	✓ type	id=password	admin123
7	✓ verify element present	css=.submit-btn	
8	✓ assert alert	Login successful!	

The log shows the execution details:

```

4. type on id=username with value admin OK
5. click on id=password OK
6. type on id=password with value admin123 OK
7. verifyElementPresent on css=.submit-btn OK
8. alertDialog on Login successful! OK
'admin/login' completed successfully

```

✓ Résultat du Scénario de Test (12) :

The screenshot shows the Selenium IDE interface with a project named "Test Plantes". A test case named "invalid_login" is selected. The test steps are as follows:

	Command	Target	Value
4	✓ type	id=username	admin
5	✓ click	id=password	
6	✓ type	id=password	123
7	✓ click	css=.submit-btn	
8	✓ assert alert	Invalid credentials!	

The log shows the execution details:

```

4. type on id=username with value admin OK
5. click on id=password OK
6. type on id=password with value 123 OK
7. click on css=.submit-btn OK
8. alertDialog on Invalid credentials! OK
'invalid_login' completed successfully

```

✓ Résultat du Scénario de Test (13) :

The screenshot shows the Selenium IDE interface with a project named "Test Plantes". A test case named "admin_dashboard" is selected. The test steps are as follows:

	Command	Target	Value
1	✓ open	http://localhost:4200/admin	
2	✓ set window size	660x654	
3	✓ assert element present	css=.add-plant-btn	

The log shows the execution details:

```

'admin_dashboard' completed successfully
Running 'admin_dashboard'
1. open on http://localhost:4200/admin OK
2. setWindowSize on 660x654 OK
3. assertElementPresent on css=.add-plant-btn OK
'admin_dashboard' completed successfully

```

✓ Résultat du Scénario de Test (14) :

The screenshot shows the Selenium IDE interface with the following details:

- Project:** Test Plantes*
- Executing:** delete_plant*
- URL:** http://localhost:4200/admin
- Log:**
 - Running 'delete_plant'
 - 1. open on http://localhost:4200/admin OK
 - 2. setWindowSize on 660x654 OK
 - 3. click on css=tr:nth-child(1).fa-trash-alt OK
 - 4. verifyElementNotPresent on xpath=/tr[td[contains(text(),'Aloe Vera')]] OK
 - 'delete_plant' completed successfully

✓ Résultat du Scénario de Test (15) :

The screenshot shows the Selenium IDE interface with the following details:

- Project:** Test Plantes*
- Tests:** admin_page, details, details_plants, edit_plant*, home_page, plant, plants, recom, recommendation, recommendation_page
- Executing:** edit_plant*
- URL:** /admin
- Log:**
 - 1. open on /admin OK
 - 'edit_plant' completed successfully
 - Running 'edit_plant'
 - 1. open on /admin OK
 - 2. assertElementPresent on css=tr:nth-child(1) button:nth-child(1) OK
 - 'edit_plant' completed successfully

✓ Résultat du Scénario de Test (16) :

The screenshot shows the Selenium IDE interface with the following details:

- Project:** Test Plantes*
- Executing:** logout*
- URL:** http://localhost:4200/
- Log:**
 - Running 'logout'
 - 1. open on /admin OK
 - 2. click on css=.logout-btn OK
 - 3. verifyElementPresent on css=button.submit-btn OK
 - 'logout' completed successfully

➤ Tests d'Automatisation avec Selenium WebDriver et JUnit :

Test Cases with Methods Used :

Test Case	Scenario	Steps	Method Used
Logout	Redirect to login page	1. The user is on the admin page (/admin). 2. Click on the 'Logout' button.	driver.findElement(By.xpath("xpath")).click()
Admin Login	Successful login	1. The user is on the login page (/login). 2. Enter valid credentials. 3. Click on 'Login'.	driver.findElement(By.id("username")) driver.findElement(By.id("password")) driver.findElement(By.xpath("xpath")).click()
Add Plant	Add a valid plant	1. The user is on the plant addition page (/admin). 2. Fill in all the required fields. 3. Click on 'Add'.	driver.findElement(By.id("plant-name")).sendKeys("PlantName") driver.findElement(By.id("plant-description")).sendKeys("PlantDescription") driver.findElement(By.xpath("xpath")).click()
Update Plant	Update an existing plant	1. The user is on the admin page (/admin). 2. Select a plant to edit. 3. Modify the required fields. 4. Click on 'Save'.	driver.findElement(By.xpath("xpath")).click() driver.findElement(By.id("edit-name")).sendKeys("UpdatedPlantName") driver.findElement(By.id("edit-description")).clear() driver.findElement(By.xpath("xpath")).click()
Delete Plant	Delete an existing plant	1. The user is on the admin page (/admin). 2. Select a plant to delete. 3. Click on 'Delete'.	driver.findElement(By.xpath("xpath")).click()
Read More	View detailed plant information	1. The user is on the recommendations page (/recommendations). 2. Click on 'Read More' for a specific plant.	driver.findElement(By.xpath("xpath")).click() driver.findElement(By.xpath("xpath")).getText()
Plant Search	Search for an existing plant	1. The user is on the admin page (/admin) with a search bar visible. 2. Enter a valid keyword. 3. Click on 'Search'.	driver.findElement(By.id("search-bar")).sendKeys("PlantName") driver.findElement(By.xpath("xpath")).click() driver.findElement(By.xpath("xpath")).getText()
Recommendations	Recommandation pour "Skin Care"	1. Aller à `/recommendations`. 2. Remplir le champ "Besoins de Santé" avec "Skin Care". 3. Cliquer sur "Get Recommendations".	driver.get(URL) new WebDriverWait(driver, Duration.ofSeconds(20)) driver.findElement(By.cssSelector("button[type='submit']"))

The screenshot shows the Android Studio interface with the project structure on the left and the code editor on the right. The code editor displays the `LogoutTest.java` file, which contains a test method `testLogout()` for a web application. The test involves clicking a logout button, waiting for a redirection, and then asserting that the current URL is the login page. The run tab at the bottom shows the test passed with a duration of 5 seconds and 198 milliseconds.

```
public class LogoutTest {
    public void testLogout() {
        // Cliquer sur le bouton de déconnexion
        logoutButton.click();

        // Attendre que la redirection vers la page de connexion se produise
        wait.until(ExpectedConditions.urlToBe("http://localhost:4200/login"));

        // Vérifier que l'URL actuelle est bien la page de connexion
        assertEquals(expected: "http://localhost:4200/login", driver.getCurrentUrl(), message);

        // Afficher un message de succès
        System.out.println("Déconnexion réussie, redirection vers la page de connexion.");
    }
}
```

The screenshot shows the Android Studio interface with the project structure on the left and the code editor on the right. The code editor displays the `DeletePlantes.java` file, which is part of the `CataloguePlantesApplicationTests` package. The file contains imports for Selenium and JUnit Jupiter, and a test class `DeletePlantes`. The run tab at the bottom shows the test passed with a duration of 5 seconds and 49 milliseconds, indicating that a plant was successfully deleted.

```
package com.example.Plantes.webdriver;

import org.junit.jupiter.api.Test;
import org.openqa.selenium.Alert;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;
import static org.junit.jupiter.api.Assertions.assertFalse;
import static java.time.Duration.*;

public class DeletePlantes {
```

Screenshot of Android Studio showing a successful test run for the `UpdatePlant` method. The terminal output shows:

```
Process finished with exit code 0
```

Screenshot of Android Studio showing a successful test run for the `testAddPlant` method. The terminal output shows:

```
Process finished with exit code 0
```

Screenshot of Android Studio showing a successful test run for the `testAdminLogin` method. The terminal output shows:

```
Process finished with exit code 0
```

The screenshot shows the Android Studio interface with the project structure on the left and three tabs at the top: CataloguePlantesApplication.java, PlantSearchTest.java, and ReadMoreTest.java. The ReadMoreTest.java tab is active, displaying Java code for a test class. Below the tabs, the run history shows a successful run of the ReadMoreTest with a duration of 5 sec 8 ms. The test result details a plant named Aloe Vera, providing its description, precautions, articles, region, uses, properties, videos section, and comments section.

```
public class ReadMoreTest {
    public void testReadMoreButton() {
        System.out.println("Properties should be present");
        WebElement properties = plantDetails.findElement(By.xpath("//span[contains(@class, 'text')]/p"));
        assertNotNull(properties, "Properties should be present");
        System.out.println("Properties: " + properties.getText());
        WebElement videos = plantDetails.findElement(By.xpath("//div[@class='video-section']"));
    }
}
```

Plant Name: Aloe Vera
Plant details are visible.
Description: Succulent plant known for its soothing properties.
Precautions: Avoid usage if allergic to aloe or if pregnant.
Articles: <https://pmc.ncbi.nlm.nih.gov/articles/PMC2763764/>
Region: Africa, Asia, South America
Uses: Skin burns, Moisturizer, Hair care
Properties: Soothing, Anti-inflammatory, Healing
Videos section found
Comments section found

The screenshot shows the Android Studio interface with the project structure on the left and three tabs at the top: CataloguePlantesApplication.java, PlantSearchTest.java, and AdminLogin.java. The PlantSearchTest.java tab is active, displaying Java code for a test class. Below the tabs, the run history shows a successful run of the PlantSearchTest with a duration of 4 sec 42 ms. The test result details a search for the plant Aloe Vera, showing its name and image source.

```
import java.time.Duration;
import static org.junit.jupiter.api.Assertions.*;
import static org.openqa.selenium.By.*;

public class PlantSearchTest {
    private WebDriver driver; 6 usages

    // Initialisation du WebDriver avant chaque test
    @BeforeEach
    public void setUp() {
        // Assurez-vous que le chemin vers le driver Chrome est correct
        System.setProperty("webdriver.chrome.driver", "C:\\chromedriver.exe");
    }
}
```

Plant Name: Aloe Vera
Image source: http://localhost:8080/admin/files/Aloe_vera.jpg

The screenshot shows the Android Studio interface with the project structure on the left and four tabs at the top: UpdatePlantes.java, DeletePlantes.java, LogoutTest.java, and RecommendationsTest.java. The RecommendationsTest.java tab is active, displaying Java code for a test class. Below the tabs, the run history shows a successful run of the RecommendationsTest with a duration of 3 sec 663 ms. The test result details a health need for the plant Calendula.

```
package com.example.Plantes.webdriver;

import org.junit.jupiter.api.Test;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.support.ui.ExpectedConditions;
import org.openqa.selenium.support.ui.WebDriverWait;

import java.time.Duration;

import static org.junit.jupiter.api.Assertions.assertTrue;

public class RecommendationsTest {
```

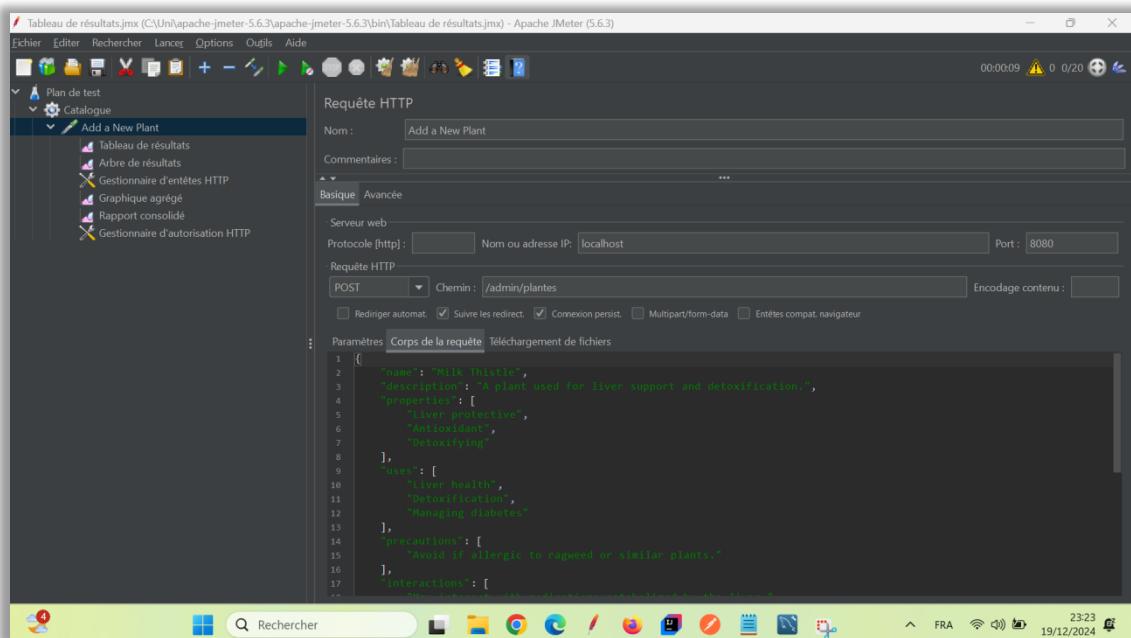
Nom de la plante : Calendula

Tests de Performance (JMeter)

➤ Test Cases with Methods Used

Test Case	Request Type	URL
1. Test Plant Recommendations API	POST	/plantes/recommandations
2. Test Retrieve All Plants API	GET	/plantes
3. Test Add a New Plant	POST	/admin/plantes
4. Test Get Comments for Plant by ID API	GET	/plantes/{id}/commentaires
5. Test File Upload	POST	/admin/files/upload

❖ Add New Plant :



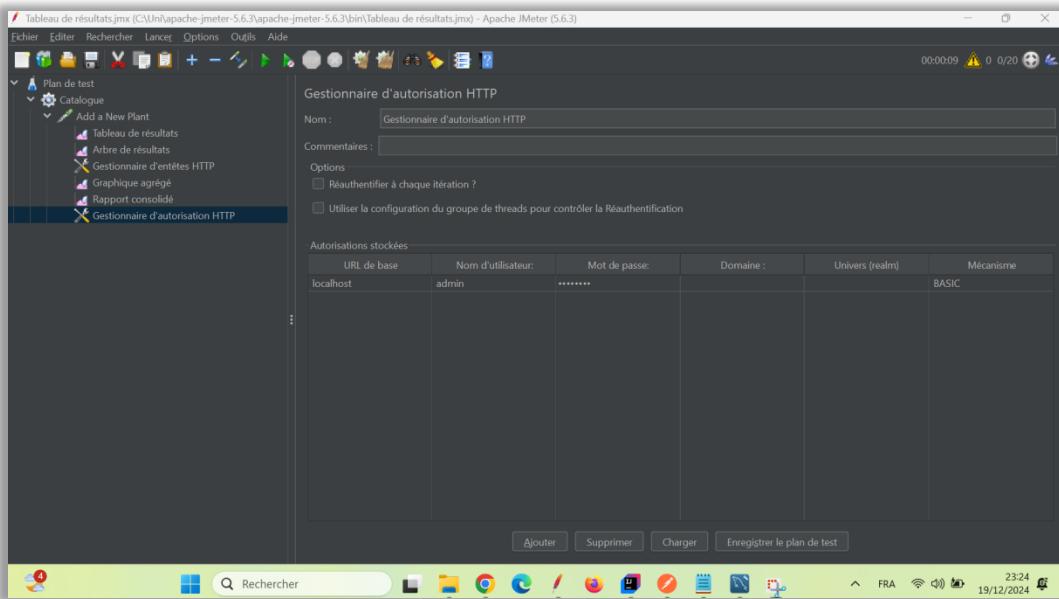
➤ Vue Résultats en Tableau :

The screenshot shows the Apache JMeter interface with the title "Tableau de résultats.jmx". The left sidebar has a tree view with "Plan de test" expanded, showing "Catalogue" and "Add a New Plant" which is selected. Under "Add a New Plant", there are several items like "Tableau de résultats", "Arbre de résultats", etc. The main panel displays a "Tableau de résultats" table with 20 rows of data. The columns are: ID, Heure début, Nom d'unité, Libellé, Temps (ms), Statut, Octets, Octets envoyés, Latence, Établ. Conn.(ms). The data shows various requests to "Catalogue" with response times ranging from 19 to 517 ms and success rates indicated by green checkmarks.

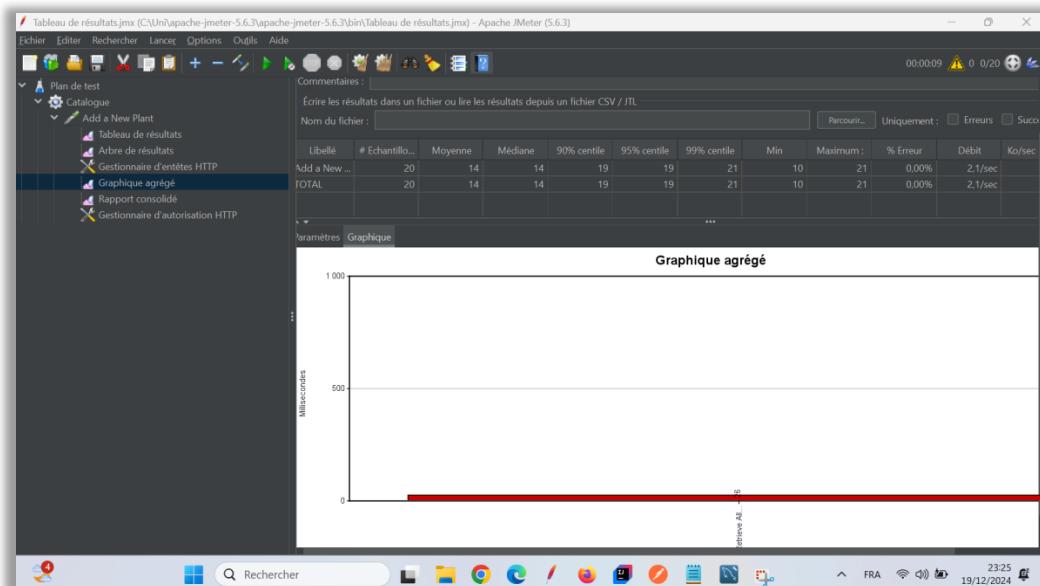
➤ Arbre de Résultats :

The screenshot shows the Apache JMeter interface with the title "Tableau de résultats.jmx". The left sidebar has a tree view with "Plan de test" expanded, showing "Catalogue" and "Add a New Plant" which is selected. Under "Add a New Plant", there are several items like "Tableau de résultats", "Arbre de résultats", etc. The main panel displays an "Arbre de résultats" tree view. The root node is "Add a New Plant" with many child nodes also labeled "Add a New Plant". To the right of the tree, there is a "Résultat de l'échantillon" panel showing a JSON response for one of the samples. The response contains details about a "Milk Thistle" plant, including its properties, interactions, and links to images and videos.

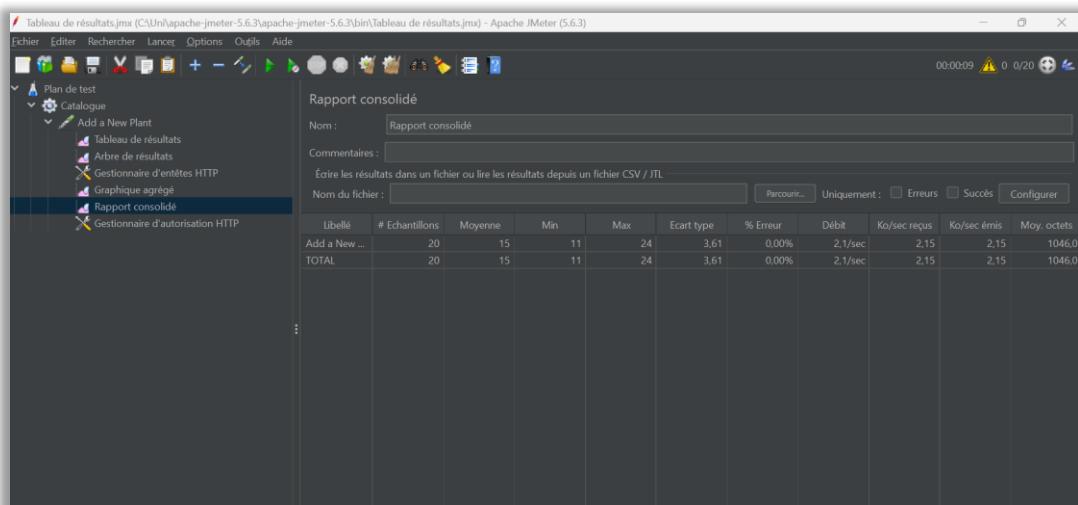
➤ Gestion de l'Autorisation HTTP



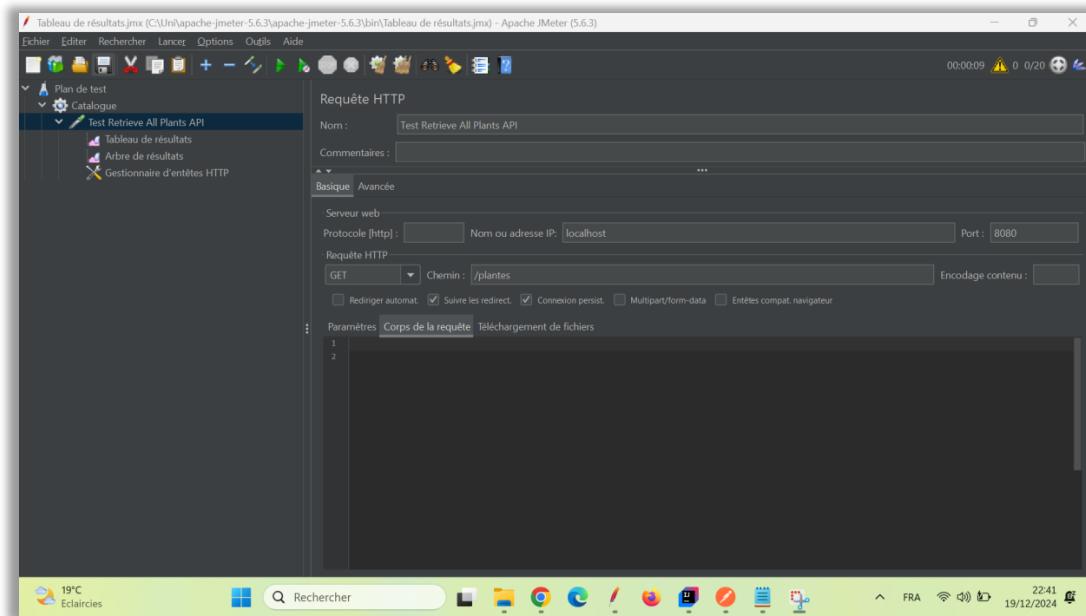
➤ Graphique des Résultats



➤ le Rapport Consolidé



❖ Test Retrieve All Plants API :



➤ Vue Résultats en Table :

The screenshot shows the Apache JMeter interface with the following details:

- Plan de test:** Catalogue > Test Retrieve All Plants API
- Tableau de résultats:** Tableau de résultats
- Résumé des échantillons:** 20 échantillons, Durée totale: 00:00:09, Taille moyenne: 135, Taille type: 300.
- Trajet:** Catalogue 1-1 à Catalogue 1-20.
- Tableau des résultats:** Détails des 20 échantillons avec colonnes: on #, Heure début, Nom d'unité, Libellé, Temps (ms), Statut, Octets, Octets envoyés, Latence, Établ. Conn.(ms).

on #	Heure début	Nom d'unité	Libellé	Temps (ms)	Statut	Octets	Octets envoyés	Latence	Établ. Conn.(ms)
1	22:40:01.971	Catalogue 1-1	Test Retrieve A...	26	OK	5505	176	26	1
2	22:40:02.485	Catalogue 1-2	Test Retrieve A...	32	OK	5505	176	32	1
3	22:40:02.982	Catalogue 1-3	Test Retrieve A...	26	OK	5505	176	26	1
4	22:40:03.469	Catalogue 1-4	Test Retrieve A...	29	OK	5505	176	29	1
5	22:40:03.969	Catalogue 1-5	Test Retrieve A...	28	OK	5505	176	27	1
6	22:40:04.469	Catalogue 1-6	Test Retrieve A...	1235	OK	5505	176	1235	1
7	22:40:04.970	Catalogue 1-7	Test Retrieve A...	756	OK	5505	176	756	1
8	22:40:05.471	Catalogue 1-8	Test Retrieve A...	255	OK	5505	176	255	1
9	22:40:05.971	Catalogue 1-9	Test Retrieve A...	34	OK	5505	176	34	1
10	22:40:06.470	Catalogue 1-10	Test Retrieve A...	24	OK	5505	176	24	1
11	22:40:06.970	Catalogue 1-11	Test Retrieve A...	33	OK	5505	176	33	1
12	22:40:07.470	Catalogue 1-12	Test Retrieve A...	25	OK	5505	176	24	1
13	22:40:07.970	Catalogue 1-13	Test Retrieve A...	23	OK	5505	176	23	1
14	22:40:08.469	Catalogue 1-14	Test Retrieve A...	29	OK	5505	176	29	1
15	22:40:08.970	Catalogue 1-15	Test Retrieve A...	24	OK	5505	176	24	1
16	22:40:09.469	Catalogue 1-16	Test Retrieve A...	33	OK	5505	176	33	1
17	22:40:09.969	Catalogue 1-17	Test Retrieve A...	18	OK	5505	176	18	1
18	22:40:10.469	Catalogue 1-18	Test Retrieve A...	34	OK	5505	176	34	1
19	22:40:10.969	Catalogue 1-19	Test Retrieve A...	19	OK	5505	176	19	1
20	22:40:11.470	Catalogue 1-20	Test Retrieve A...	32	OK	5505	176	32	1

➤ Arbre de Résultats :

The screenshot shows the Apache JMeter interface with the title "Tableau de résultats.jmx". The left sidebar shows a tree structure under "Plan de test" with "Catalogue" expanded, containing "Test Retrieve All Plants API", "Tableau de résultats", "Arbre de résultats", and "Gestionnaire d'enêtes HTTP". The main panel is titled "Arbre de résultats" and displays the response body of a "Test Retrieve All Plants API" request. The response body is a JSON array of plants, including Aloe Vera and Ashwagandha, with their properties, interactions, and precautions.

➤ Graphique des Résultats :

The screenshot shows the Apache JMeter interface with the title "Tableau de résultats.jmx". The left sidebar shows a tree structure under "Plan de test" with "Catalogue" expanded, containing "Test Retrieve All Plants API", "Tableau de résultats", "Arbre de résultats", and "Gestionnaire d'enêtes HTTP". The main panel is titled "Graphique agrégé" and displays an aggregate report. The summary table shows metrics like Average, Median, and Percentiles for the "Test Retrieve All Plants API" test. Below the table is a chart titled "Graphique agrégé" with the Y-axis labeled "MILLISECONDS" ranging from 0 to 1,000. The chart shows a single data series represented by a red bar at the 0 mark.

➤ le Rapport Consolidé

The screenshot shows the Apache JMeter interface with the 'Tableau de résultats.jmx' file open. On the left, the 'Plan de test' tree view shows a 'Catalogue' node expanded, containing a 'Test Retrieve All Plants API' node which further expands to show 'Tableau de résultats', 'Arbre de résultats', 'Gestionnaire d'enêtes HTTP', 'Graphique agrégé', and 'Rapport consolidé'. The main panel displays the 'Rapport consolidé' configuration window. It includes fields for 'Nom:' (Nom : Rapport consolidé), 'Commentaires:', and 'Nom du fichier:' (Nom du fichier :). Below these are filtering options ('Uniquement : Erreurs, Succès') and a 'Configurer' button. A large table follows, showing aggregated results for the 'Test Retrieve...' sample. The table has columns for 'Libellé', '# Echantillons', 'Moyenne', 'Min', 'Max', 'Ecart type', '% Erreur', 'Débit', 'Ko/sec reçus', 'Ko/sec émis', and 'Moy. octets'. The data shows 20 samples with a mean of 25 ms, minimum of 17 ms, maximum of 36 ms, and an average error of 0.00%. The total throughput is 2.1/sec with 11.28 Ko/sec received and 0.36 Ko/sec sent, over a duration of 550.0 ms.

❖ HTTP Request Sampler for File Upload :

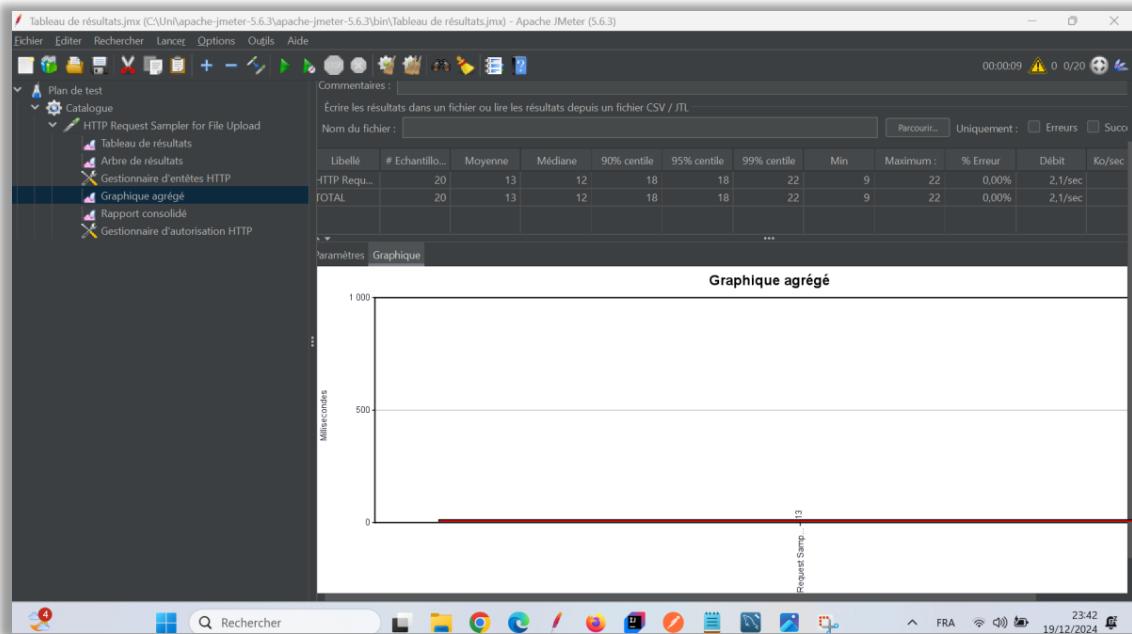
The screenshot shows the Apache JMeter interface with the 'Tableau de résultats.jmx' file open. The 'Plan de test' tree view shows a 'Catalogue' node expanded, containing a 'HTTP Request Sampler for File Upload' node which further expands to show 'Tableau de résultats', 'Arbre de résultats', 'Gestionnaire d'enêtes HTTP', 'Graphique agrégé', 'Rapport consolidé', and 'Gestionnaire d'autorisation HTTP'. The main panel displays the configuration for the 'HTTP Request Sampler for File Upload'. The 'Nom:' field is set to 'HTTP Request Sampler for File Upload'. Under the 'Basique' tab, the 'Protocole [http:]' is set to 'http', 'Nom ou adresse IP:' is 'localhost', and 'Port:' is '8080'. In the 'Requête HTTP' section, the 'Méthode' is 'POST' and the 'Chemin:' is '/admin/files/upload'. The 'Paramètres' tab shows a single parameter named 'file' with a value of 'C:\Uni\Medicinal_Plants_Catalog_App-main (1)\Medi...'. The 'Corps de la requête' tab is selected, showing the file path. At the bottom, there are buttons for 'Détail', 'Ajouter', 'Parcourir...', 'Ajouter depuis Presse-papier', 'Supprimer', 'Monter', and 'Descendre'.

➤ Vue Résultats en Tableau :

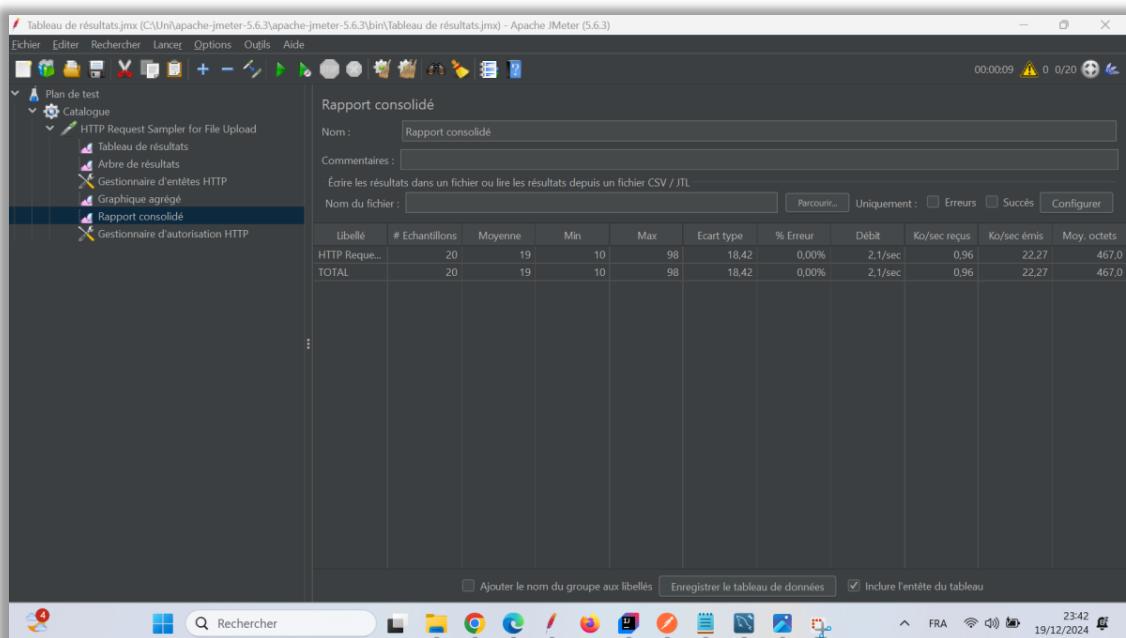
on #	Heure début	Nom d'unité	Libellé	Temps (ms)	Statut	Octets	Octets envoyés	Latence	Etabl. Conn(ms)
1	23:39:52.469	Catalogue 1-1	HTTP Request ...	417	200	467	10842	416	1
2	23:39:52.966	Catalogue 1-2	HTTP Request ...	21	200	467	10857	21	1
3	23:39:53.467	Catalogue 1-3	HTTP Request ...	19	200	467	10842	19	1
4	23:39:53.966	Catalogue 1-4	HTTP Request ...	18	200	467	10842	18	1
5	23:39:54.467	Catalogue 1-5	HTTP Request ...	16	200	467	10854	16	1
6	23:39:54.966	Catalogue 1-6	HTTP Request ...	12	200	467	10848	12	1
7	23:39:55.465	Catalogue 1-7	HTTP Request ...	16	200	467	10851	16	1
8	23:39:55.964	Catalogue 1-8	HTTP Request ...	13	200	467	10869	13	1
9	23:39:56.465	Catalogue 1-9	HTTP Request ...	12	200	467	10860	11	1
10	23:39:56.965	Catalogue 1-10	HTTP Request ...	16	200	467	10866	16	1
11	23:39:57.465	Catalogue 1-11	HTTP Request ...	12	200	467	10866	12	1
12	23:39:57.964	Catalogue 1-12	HTTP Request ...	16	200	467	10845	16	1
13	23:39:58.465	Catalogue 1-13	HTTP Request ...	11	200	467	10860	11	1
14	23:39:58.968	Catalogue 1-14	HTTP Request ...	21	200	467	10863	21	1
15	23:39:59.464	Catalogue 1-15	HTTP Request ...	17	200	467	10845	17	2
16	23:39:59.965	Catalogue 1-16	HTTP Request ...	12	200	467	10839	12	1
17	23:40:00.463	Catalogue 1-17	HTTP Request ...	12	200	467	10842	12	1
18	23:40:00.967	Catalogue 1-18	HTTP Request ...	14	200	467	10866	14	1
19	23:40:01.466	Catalogue 1-19	HTTP Request ...	11	200	467	10839	11	1
20	23:40:01.967	Catalogue 1-20	HTTP Request ...	17	200	467	10860	17	1

➤ Arbre de Résultats :

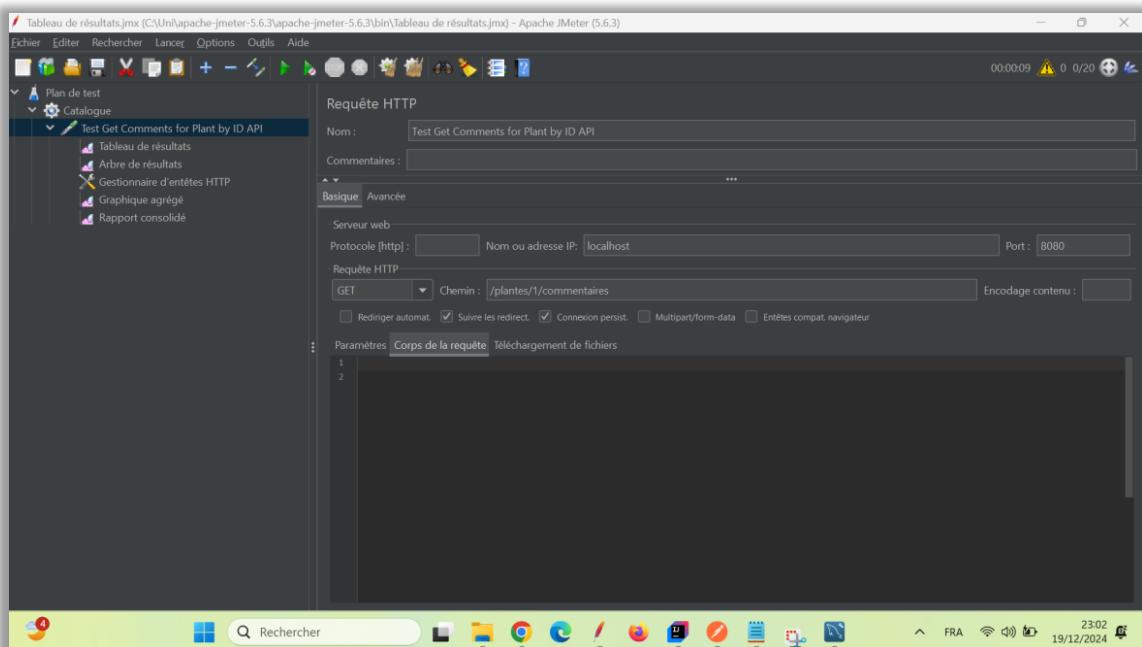
➤ Graphique des Résultats :



➤ le Rapport Conso



❖ Test Get Comments for Plant by ID API :



➤ Vue Résultats en Table :

on #	Heure début	Nom d'unité	Libellé	Temps (ms)	Statut	Octets	Octets envoyés	Latence	Établ. Conn(ms)
1	23/02/09.348	Catalogue 1-1	Test Get Com...	157	200	501	191	156	1
2	23/02/09.861	Catalogue 1-2	Test Get Com...	12	200	501	191	12	1
3	23/02/10.360	Catalogue 1-3	Test Get Com...	5	200	501	191	5	1
4	23/02/10.847	Catalogue 1-4	Test Get Com...	12	200	501	191	11	1
5	23/02/11.347	Catalogue 1-5	Test Get Com...	6	200	501	191	6	1
6	23/02/11.847	Catalogue 1-6	Test Get Com...	13	200	501	191	13	1
7	23/02/12.347	Catalogue 1-7	Test Get Com...	8	200	501	191	7	1
8	23/02/12.847	Catalogue 1-8	Test Get Com...	9	200	501	191	9	1
9	23/02/13.347	Catalogue 1-9	Test Get Com...	10	200	501	191	10	1
10	23/02/13.847	Catalogue 1-10	Test Get Com...	7	200	501	191	7	1
11	23/02/14.348	Catalogue 1-11	Test Get Com...	10	200	501	191	9	1
12	23/02/14.848	Catalogue 1-12	Test Get Com...	10	200	501	191	10	1
13	23/02/15.347	Catalogue 1-13	Test Get Com...	7	200	501	191	7	1
14	23/02/15.847	Catalogue 1-14	Test Get Com...	8	200	501	191	8	1
15	23/02/16.348	Catalogue 1-15	Test Get Com...	7	200	501	191	7	1
16	23/02/16.847	Catalogue 1-16	Test Get Com...	13	200	501	191	12	1
17	23/02/17.347	Catalogue 1-17	Test Get Com...	7	200	501	191	7	1
18	23/02/17.847	Catalogue 1-18	Test Get Com...	13	200	501	191	13	1
19	23/02/18.349	Catalogue 1-19	Test Get Com...	9	200	501	191	9	1
20	23/02/18.848	Catalogue 1-20	Test Get Com...	7	200	501	191	7	1

➤ Arbre de Résultats :

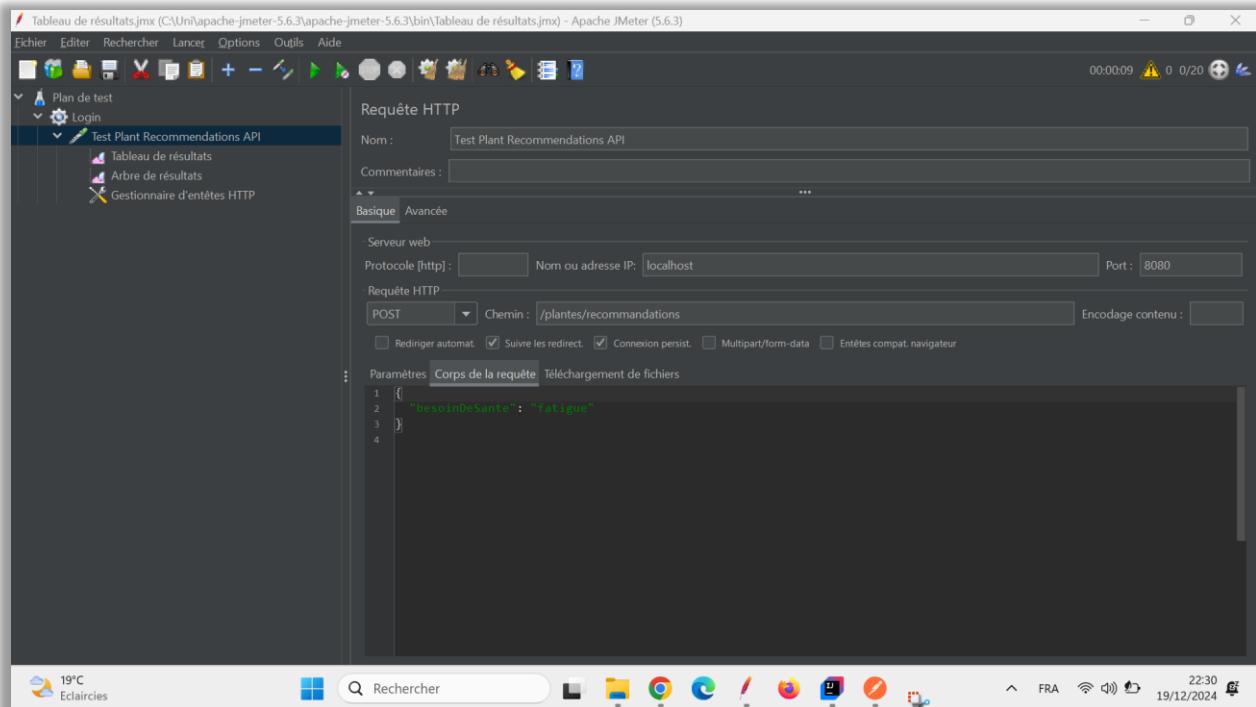
The screenshot shows the Apache JMeter interface with the "Arbre de résultats" (Results Tree) selected in the left sidebar under the "Catalogue" section. The tree view displays multiple entries for "Test Get Comments for Plant by ID API", all marked with a green checkmark, indicating success. The right panel shows the raw response text for one of the requests, which includes JSON data: [{"id":56,"nom":"Khadija","contenu":"It healed my wounds perfectly"}]. The status bar at the bottom right shows the date and time as 19/12/2024 23:03.

➤ le Rapport Consolidé :

The screenshot shows the Apache JMeter interface with the "Rapport consolidé" (Consolidated Report) selected in the left sidebar under the "Catalogue" section. The report table provides a summary of the test results:

Libellé	# Echantillons	Moyenne	Min	Max	Ecart type	% Erreur	Débit	Ko/sec reçus	Ko/sec émis	Moy. octets
Test Get Co...	20	9	4	17	3,58	0,00%	2,1/sec	1,03	0,39	501,0
TOTAL	20	9	4	17	3,58	0,00%	2,1/sec	1,03	0,39	501,0

❖ Test Plant Recommendations API :



➤ Vue Résultats en Table :

n°	Heure début	Nom d'unité	Libellé	Temps (ms)	Statut	Octets	Octets envoyés	Latence	Établ. Conn.(ms)
1	22:29:47.199	Login 1-2	Test Plant Reco...	304	✓	433	228	304	1
2	22:29:46.700	Login 1-1	Test Plant Reco...	806	✓	433	228	806	2
3	22:29:47.699	Login 1-3	Test Plant Reco...	9	✓	433	228	9	1
4	22:29:48.198	Login 1-4	Test Plant Reco...	16	✓	433	228	16	1
5	22:29:48.698	Login 1-5	Test Plant Reco...	107	✓	433	228	106	0
6	22:29:49.199	Login 1-6	Test Plant Reco...	89	✓	433	228	89	1
7	22:29:49.699	Login 1-7	Test Plant Reco...	25	✓	433	228	24	2
8	22:29:50.199	Login 1-8	Test Plant Reco...	22	✓	433	228	22	2
9	22:29:50.699	Login 1-9	Test Plant Reco...	17	✓	433	228	17	1
10	22:29:51.198	Login 1-10	Test Plant Reco...	14	✓	433	228	13	3
11	22:29:51.699	Login 1-11	Test Plant Reco...	12	✓	433	228	12	1
12	22:29:52.198	Login 1-12	Test Plant Reco...	11	✓	433	228	10	1
13	22:29:52.698	Login 1-13	Test Plant Reco...	20	✓	433	228	19	1
14	22:29:53.198	Login 1-14	Test Plant Reco...	14	✓	433	228	13	1
15	22:29:53.699	Login 1-15	Test Plant Reco...	10	✓	433	228	10	1
16	22:29:54.201	Login 1-16	Test Plant Reco...	11	✓	433	228	11	1
17	22:29:54.698	Login 1-17	Test Plant Reco...	21	✓	433	228	21	1
18	22:29:55.199	Login 1-18	Test Plant Reco...	9	✓	433	228	9	1
19	22:29:55.698	Login 1-19	Test Plant Reco...	8	✓	433	228	8	1
20	22:29:56.197	Login 1-20	Test Plant Reco...	9	✓	433	228	8	1

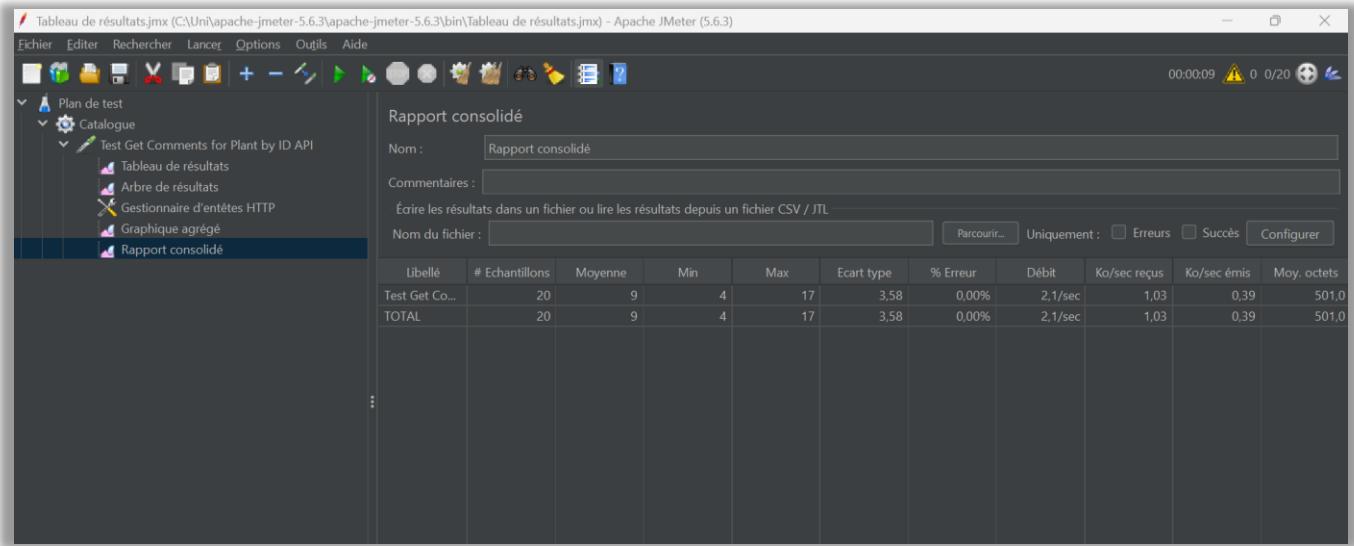
➤ Arbre de Résultats :

The screenshot shows the Apache JMeter interface with the 'Arbre de résultats' (Results Tree) selected in the left sidebar under the 'Test Plant Recommendations API' test plan. The main panel displays a tree structure with 20 entries, each representing a sample from the 'Test Plant Recommendations API'. The results are shown in a raw text format. At the top, there are various configuration options like 'Nom : Arbre de résultats', 'Commentaires :', and search filters. The system tray at the bottom indicates it's 19°C, 22:32, and the date is 19/12/2024.

➤ Graphique des Résultats :

The screenshot shows the Apache JMeter interface with the 'Graphique agrégé' (Aggregated Graph) selected in the left sidebar under the 'Test Plant Recommendations API' test plan. The main panel displays an aggregated graph titled 'Graphique agrégé' showing response times for 20 samples. The Y-axis represents time in milliseconds (ms), ranging from 0 to 1000. The X-axis represents the sample index. A single horizontal line represents the average response time for all samples. The system tray at the bottom indicates it's 19°C, 22:36, and the date is 19/12/2024.

➤ le Rapport Consolidé :



The screenshot shows the Apache JMeter interface with the 'Tableau de résultats.jmx' file open. The left sidebar displays a tree structure of test elements under 'Plan de test' and 'Catalogue'. The 'Rapport consolidé' node is selected. The main panel shows the 'Rapport consolidé' configuration dialog. The 'Nom:' field is set to 'Rapport consolidé'. Below it, there's a note: 'Écrire les résultats dans un fichier ou lire les résultats depuis un fichier CSV / JTL.' A 'Nom du fichier:' input field contains the path 'C:\Uni\apache-jmeter-5.6.3\apache-jmeter-5.6.3\bin\Tableau de résultats.jmx'. To the right are buttons for 'Parcourir...', 'Uniquement :', 'Erreurs', 'Succès', and 'Configurer'. A preview table below shows performance metrics for a single test element and a total row.

Libellé	# Echantillons	Moyenne	Min	Max	Ecart type	% Erreur	Débit	Ko/sec reçus	Ko/sec émis	Moy. octets
Test Get Co...	20	9	4	17	3,58	0,00%	2,1/sec	1,03	0,39	501,0
TOTAL	20	9	4	17	3,58	0,00%	2,1/sec	1,03	0,39	501,0