



**POLYTECHNIQUE  
MONTRÉAL**

UNIVERSITÉ  
D'INGÉNIERIE

# **LOG3430 – MÉTHODES DE TEST ET DE VALIDATION DU LOGICIEL**

## **TP1 : Couverture de test**

Automne 2022

Groupe 3 (4) :

Victor Kim – 1954607

Jérémy Perreault – 1903274

Soumis à :

Maxime Lamothe

Dmytro Humenium

Adam Halim

[27 septembre 2022]

**Question 1 : En générant un rapport de couverture de code à l'aide de l'IDE PyCharm, quelle est la couverture totale du code (files, lines, statement coverage, missing, excluded, coverage %) pour Requests v2.28.1 ? Vous devez aussi soumettre le rapport de couverture sous forme de fichier zip nommé Q1.zip.**

Comme il est possible de le constater à la figure 1, la couverture totale du code pour Requests v2.28.1 est de 88%.

Coverage report: 88%				
coverage.py v6.4.4, created at 2022-09-19 15:24 -0400				
Module	statements	missing	excluded	coverage
D:\log3430 TP1\requests\requests\__init__.py	68	24	0	65%
D:\log3430 TP1\requests\requests\__version__.py	10	0	0	100%
D:\log3430 TP1\requests\requests\_internal_utils.py	19	0	0	100%
D:\log3430 TP1\requests\requests\adapters.py	216	15	0	93%
D:\log3430 TP1\requests\requests\api.py	19	3	0	84%
D:\log3430 TP1\requests\requests\auth.py	173	21	0	88%
D:\log3430 TP1\requests\requests\certs.py	4	1	0	75%
D:\log3430 TP1\requests\requests\compat.py	30	2	0	93%
D:\log3430 TP1\requests\requests\cookies.py	239	53	0	78%
D:\log3430 TP1\requests\requests\exceptions.py	37	0	0	100%
D:\log3430 TP1\requests\requests\help.py	62	19	0	69%
D:\log3430 TP1\requests\requests\hooks.py	14	0	0	100%
D:\log3430 TP1\requests\requests\models.py	455	33	0	93%
D:\log3430 TP1\requests\requests\packages.py	17	0	0	100%
D:\log3430 TP1\requests\requests\sessions.py	268	12	0	96%
D:\log3430 TP1\requests\requests\status_codes.py	14	0	0	100%
D:\log3430 TP1\requests\requests\structures.py	39	0	0	100%
D:\log3430 TP1\requests\requests\utils.py	482	68	0	86%
D:\log3430 TP1\requests\tests\__init__.py	3	0	0	100%
D:\log3430 TP1\requests\tests\compat.py	12	2	0	83%
D:\log3430 TP1\requests\tests\conftest.py	37	3	0	92%
D:\log3430 TP1\requests\tests\test_help.py	12	0	0	100%
D:\log3430 TP1\requests\tests\test_hooks.py	9	0	0	100%
D:\log3430 TP1\requests\tests\test_lowlevel.py	227	0	0	100%
D:\log3430 TP1\requests\tests\test_packages.py	7	0	0	100%
D:\log3430 TP1\requests\tests\test_requests.py	1649	24	0	99%
D:\log3430 TP1\requests\tests\test_structures.py	43	0	0	100%

Figure 1. Rapport de couverture de Request v2.28.1 avec l'IDE PyCharm

**Question 2 : Utilisez l'IDE PyCharm pour activer la couverture des branches. En générant un nouveau rapport de couverture de code à l'aide de l'IDE PyCharm, quelle est la couverture totale du code (files, lines, statements, missing, excluded, branches, partial coverage %) pour Requests v2.28.1 ? Y a-t-il une différence avec le rapport précédent ? Expliquez pourquoi il y a/n'y a pas de différence Vous devez aussi soumettre le rapport de couverture sous forme de fichier zip nommé Q2.zip.**

Après avoir activé la couverture des branches avec l'IDE PyCharm, la couverture totale obtenue est de 85%. La figure 2 présente ce résultat :

Coverage report: 85%

coverage.py v6.4.4, created at 2022-09-19 15:29 -0400

Module	statements	missing	excluded	branches	partial	coverage
D:\log3430 TP1\requests\requests\__init__.py	68	24	0	10	4	62%
D:\log3430 TP1\requests\requests\_version_.py	10	0	0	0	0	100%
D:\log3430 TP1\requests\requests\_internal_utils.py	19	0	0	2	0	100%
D:\log3430 TP1\requests\requests\adapters.py	216	15	0	82	8	92%
D:\log3430 TP1\requests\requests\api.py	19	3	0	0	0	84%
D:\log3430 TP1\requests\requests\auth.py	173	21	0	68	15	84%
D:\log3430 TP1\requests\requests\certs.py	4	1	0	2	1	67%
D:\log3430 TP1\requests\requests\compat.py	30	2	0	2	1	91%
D:\log3430 TP1\requests\requests\cookies.py	239	53	0	106	9	72%
D:\log3430 TP1\requests\requests\exceptions.py	37	0	0	52	0	100%
D:\log3430 TP1\requests\requests\help.py	62	19	0	18	5	60%
D:\log3430 TP1\requests\requests\hooks.py	14	0	0	10	0	100%
D:\log3430 TP1\requests\requests\models.py	455	33	0	198	18	92%
D:\log3430 TP1\requests\requests\packages.py	17	0	0	10	0	100%
D:\log3430 TP1\requests\requests\sessions.py	268	12	0	104	5	95%
D:\log3430 TP1\requests\requests\status_codes.py	14	0	0	10	0	100%
D:\log3430 TP1\requests\requests\structures.py	39	0	0	12	0	100%
D:\log3430 TP1\requests\requests\utils.py	482	68	0	214	12	86%
D:\log3430 TP1\requests\tests\__init__.py	3	0	0	0	0	100%
D:\log3430 TP1\requests\tests\compat.py	12	2	0	0	0	83%
D:\log3430 TP1\requests\tests\conftest.py	37	3	0	0	0	92%
D:\log3430 TP1\requests\tests\test_help.py	12	0	0	2	0	100%
D:\log3430 TP1\requests\tests\test_hooks.py	9	0	0	2	0	100%
D:\log3430 TP1\requests\tests\test_lowlevel.py	227	0	0	8	0	100%
D:\log3430 TP1\requests\tests\test_packages.py	7	0	0	0	0	100%
D:\log3430 TP1\requests\tests\test_requests.py	1649	24	0	108	6	98%
D:\log3430 TP1\requests\tests\test_structures.py	42	0	0	4	0	100%

Figure 2. Rapport de couverture de Request v2.28.1 avec l'IDE PyCharm et la couverture des branches activées

Il est possible de constater que la couverture a diminué de 3%. Cette réduction peut s'expliquer par le fait qu'en parcourant seulement les branches, moins d'instructions sont couvertes.

**Question 3 : Vous devez exclure un fichier de code source du processus de génération de rapport de test. Quel fichier avez-vous exclu (nom et répertoire du fichier) ? Comment l'avez-vous exclu du processus ? Comment les résultats changent-ils par rapport au dernier rapport (conserver l'option pour avoir la couverture des 2 branches) ? Vous devez soumettre le rapport généré sous forme de fichier zip nommé Q3.zip.**

Le fichier qui a été exclu du processus est "test\_structures.py". Celui-ci a été exclu en exécutant la commande suivante :

```
coverage run -m --0 pytest --ignore=tests\\test_structures.py
```

C'est l'option --ignore=test\\test\_structures.py qui permet d'ignorer les tests compris dans le fichier lors de la couverture.

Comme il est possible de l'observer à la figure 3, la couverture du fichier "structure.py" est maintenant de 94%. À la question 2, cette valeur était de 100%, ce qui représente une réduction de 6%. Or, ce changement n'affecte pas la couverture globale qui elle demeure à 88%.

Coverage report: 88% filter...

coverage.py v6.4.4, created at 2022-09-22 12:52 -0400

Module	statements	missing	excluded	branches	partial	coverage
requests\__init__.py	68	24	0	10	4	62%
requests\__version__.py	10	0	0	0	0	100%
requests\_internal_utils.py	19	0	0	2	0	100%
requests\adapters.py	216	15	0	82	8	92%
requests\api.py	19	3	0	0	0	84%
requests\auth.py	173	21	0	68	15	84%
requests\certs.py	4	1	0	2	1	67%
requests\compat.py	30	2	0	2	1	91%
requests\cookies.py	239	53	0	106	9	72%
requests\exceptions.py	37	0	0	52	0	100%
requests\help.py	62	19	0	18	5	60%
requests\hooks.py	14	0	0	10	0	100%
requests\models.py	455	32	0	198	18	92%
requests\packages.py	17	0	0	10	0	100%
requests\sessions.py	268	12	0	104	5	95%
requests\status_codes.py	14	0	0	10	0	100%
requests\structures.py	39	3	0	12	0	94%
requests\utils.py	482	58	0	214	13	88%
<b>Total</b>	<b>2166</b>	<b>243</b>	<b>0</b>	<b>900</b>	<b>79</b>	<b>88%</b>

coverage.py v6.4.4, created at 2022-09-22 12:52 -0400

Figure 3. Rapport de couverture de Request v2.28.1 avec le fichier "test\_structures.py" exclu de l'exécution

**Question 4 : Veuillez expliquer, étape par étape, comment utiliser coverage.py pour calculer la couverture de test de Requests sans IDE (utiliser l'option pour branch coverage). Vous devez soumettre le rapport généré par coverage.py sous forme de fichier zip nommé Q4.zip.**

Pour calculer la couverture de test à l'aide de coverage.py, il ne suffit d'exécuter la commande suivante dans un terminal :

```
coverage run -m --branch pytest
```

L'option `--branch`, permet d'activer la couverture de branche. La figure 4 démontre que le résultat obtenu est similaire à celui de la question 2.

Coverage report: 85% filter...

coverage.py v6.4.4, created at 2022-09-19 15:56 -0400

Module	statements	missing	excluded	branches	partial	coverage
D:\log3430 TP1\requests\requests\__init__.py	68	24	0	10	4	62%
D:\log3430 TP1\requests\requests\_version_.py	10	0	0	0	0	100%
D:\log3430 TP1\requests\requests\_internal_utils.py	19	0	0	2	0	100%
D:\log3430 TP1\requests\requests\adapters.py	216	15	0	82	8	92%
D:\log3430 TP1\requests\requests\api.py	19	3	0	0	0	84%
D:\log3430 TP1\requests\requests\auth.py	173	21	0	68	15	84%
D:\log3430 TP1\requests\requests\certs.py	4	1	0	2	1	67%
D:\log3430 TP1\requests\requests\compat.py	30	2	0	2	1	91%
D:\log3430 TP1\requests\requests\cookies.py	239	53	0	106	9	72%
D:\log3430 TP1\requests\requests\exceptions.py	37	0	0	52	0	100%
D:\log3430 TP1\requests\requests\help.py	62	19	0	18	5	66%
D:\log3430 TP1\requests\requests\hooks.py	14	0	0	10	0	100%
D:\log3430 TP1\requests\requests\models.py	455	33	0	198	18	92%
D:\log3430 TP1\requests\requests\packages.py	17	0	0	10	0	100%
D:\log3430 TP1\requests\requests\sessions.py	268	12	0	104	5	95%
D:\log3430 TP1\requests\requests\status_codes.py	14	0	0	10	0	100%
D:\log3430 TP1\requests\requests\structures.py	39	0	0	12	0	100%
D:\log3430 TP1\requests\requests\utils.py	482	68	0	214	12	86%
D:\log3430 TP1\requests\tests\__init__.py	3	0	0	0	0	100%
D:\log3430 TP1\requests\tests\compat.py	12	2	0	0	0	83%
D:\log3430 TP1\requests\tests\conftest.py	37	3	0	0	0	92%
D:\log3430 TP1\requests\tests\test_help.py	12	0	0	2	0	100%
D:\log3430 TP1\requests\tests\test_hooks.py	9	0	0	2	0	100%
D:\log3430 TP1\requests\tests\test_lowlevel.py	227	0	0	8	0	100%
D:\log3430 TP1\requests\tests\test_packages.py	7	0	0	0	0	100%
D:\log3430 TP1\requests\tests\test_requests.py	1649	24	0	108	6	98%

Figure 4. Rapport de couverture de Request v2.28.1 avec coverage.py

Pour obtenir le rapport de couverture, la commande suivante a été exécutée :

```
coverage html
```

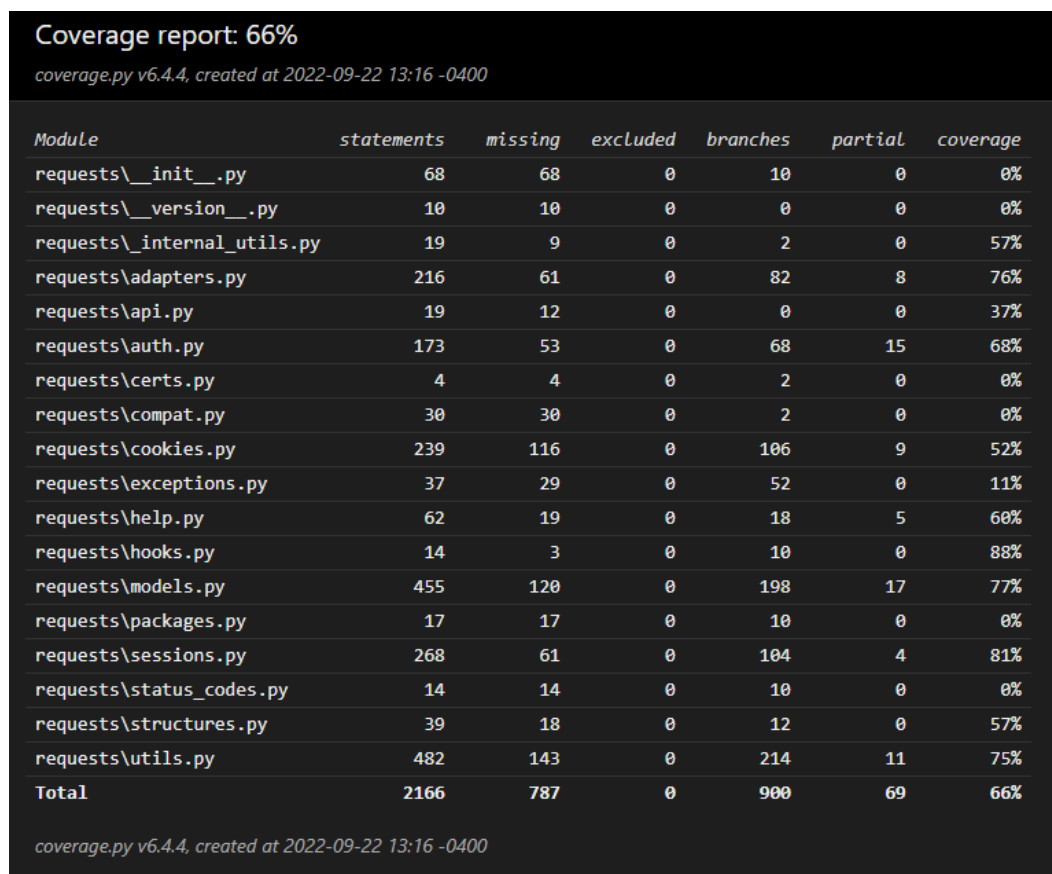
**Question 5 : Veuillez expliquer, étape par étape, comment utiliser pytest pour calculer la couverture de test de Requests sans IDE (utiliser l'option pour branch coverage). Vous devez soumettre le rapport généré sous forme de fichier zip nommé Q5.zip.**

Pour calculer la couverture de test avec pytest, il faut simplement exécuter la commande suivante dans un terminal :

```
pytest --cov-branch --cov-report html:cov_html --cov=requests tests/
```

L'option `--cov-branch` permet d'activer la couverture des branches, `--cov-report html:cov_html` permet de créer un rapport de couverture et de spécifier le nom du fichier de sortie en plus de son type, puis `--cov=requests tests/` permet de spécifier le dossier source où récupérer les tests.

La figure 5 représente le rapport de couverture résultant de cette opération :



Module	statements	missing	excluded	branches	partial	coverage
requests\__init__.py	68	68	0	10	0	0%
requests\__version__.py	10	10	0	0	0	0%
requests\_internal_utils.py	19	9	0	2	0	57%
requests\adapters.py	216	61	0	82	8	76%
requests\api.py	19	12	0	0	0	37%
requests\auth.py	173	53	0	68	15	68%
requests\certs.py	4	4	0	2	0	0%
requests\compat.py	30	30	0	2	0	0%
requests\cookies.py	239	116	0	106	9	52%
requests\exceptions.py	37	29	0	52	0	11%
requests\help.py	62	19	0	18	5	60%
requests\hooks.py	14	3	0	10	0	88%
requests\models.py	455	120	0	198	17	77%
requests\packages.py	17	17	0	10	0	0%
requests\sessions.py	268	61	0	104	4	81%
requests\status_codes.py	14	14	0	10	0	0%
requests\structures.py	39	18	0	12	0	57%
requests\utils.py	482	143	0	214	11	75%
<b>Total</b>	<b>2166</b>	<b>787</b>	<b>0</b>	<b>900</b>	<b>69</b>	<b>66%</b>

Figure 5. Rapport de couverture de Request v2.28.1 avec pytest

**Question 6 : Vous devez créer deux nouveaux tests pour augmenter la couverture de test de Requests. Quels tests avez-vous écrits ? En utilisant l'une des deux méthodes de couverture de test des questions précédentes (PyCharm ou Coverage.py ou pytest), montrez quels critères de couverture de test vos tests ont augmentés. Vous devez soumettre un zip «Q6.zip» du repo Requests complet dans lequel vous avez ajouté vos tests.**

Les deux tests qui ont été créés pour augmenter la couverture de test de Requests sont présentés à la figure 6.1.:

```
class TestDictToSequence:
    new
    @pytest.mark.parametrize(
        "value, expected",
        (
            ({'nom': 'Kim', 'prenom': 'Victor'}, {'nom': 'Kim', 'prenom': 'Victor'}.items()),
            ('nom', 'nom')
        )
    )
    def test_dict_to_sequence(self, value, expected):
        assert dict_to_sequence(value) == expected
```

Figure 6.1. Tests créés

Ceux-ci résident dans le fichier tests\_utils.py et ont pour but de tester la méthode dict\_to\_sequence du fichier utils.py. La figure 6.2. présente la méthode testée :

```
119 def dict_to_sequence(d):
120     """Returns an internal sequence dictionary update."""
121
122     if hasattr(d, "items"):
123         d = d.items()
124
125     return d
126
```

Figure 6.2. Méthode testée

Le premier test a pour but de tester la branche où la condition `hasattr` est vraie. Le deuxième test vérifie la sortie de la méthode si la condition n'est pas vraie.

En observant les rapports de couverture présentés aux figures 6.3. et 6.4., il est possible de constater une diminution du nombre d'instructions manquantes :

Coverage report: 87%

coverage.py v6.4.4, created at 2022-09-22 15:12 -0400

Module	statements	missing	excluded	branches	partial	coverage
C:\Users\perre\Documents\GitHub\requests\requests\__init__.py	68	24	0	10	4	62%
C:\Users\perre\Documents\GitHub\requests\requests\__version__.py	10	0	0	0	0	100%
C:\Users\perre\Documents\GitHub\requests\requests\_internal_utils.py	19	0	0	2	0	100%
C:\Users\perre\Documents\GitHub\requests\requests\adapters.py	216	15	0	82	8	92%
C:\Users\perre\Documents\GitHub\requests\requests\api.py	19	3	0	0	0	84%
C:\Users\perre\Documents\GitHub\requests\requests\auth.py	173	21	0	68	15	84%
C:\Users\perre\Documents\GitHub\requests\requests\certs.py	4	1	0	2	1	67%
C:\Users\perre\Documents\GitHub\requests\requests\compat.py	30	2	0	2	1	91%
C:\Users\perre\Documents\GitHub\requests\requests\cookies.py	239	53	0	106	9	72%
C:\Users\perre\Documents\GitHub\requests\requests\exceptions.py	37	0	0	52	0	100%
C:\Users\perre\Documents\GitHub\requests\requests\help.py	62	19	0	18	5	60%
C:\Users\perre\Documents\GitHub\requests\requests\hooks.py	14	0	0	10	0	100%
C:\Users\perre\Documents\GitHub\requests\requests\models.py	455	33	0	198	18	92%
C:\Users\perre\Documents\GitHub\requests\requests\packages.py	17	0	0	10	0	100%
C:\Users\perre\Documents\GitHub\requests\requests\sessions.py	268	12	0	104	5	95%
C:\Users\perre\Documents\GitHub\requests\requests\status_codes.py	14	0	0	10	0	100%
C:\Users\perre\Documents\GitHub\requests\requests\structures.py	39	0	0	12	0	100%
C:\Users\perre\Documents\GitHub\requests\requests\utils.py	482	68	0	214	12	86%
<b>Total</b>	<b>2166</b>	<b>251</b>	<b>0</b>	<b>900</b>	<b>78</b>	<b>87%</b>

coverage.py v6.4.4, created at 2022-09-22 15:12 -0400

Figure 6.3. Rapport de couverture de Request v2.28.1 avant l'ajout des tests.

Coverage report: 87%

coverage.py v6.4.4, created at 2022-09-22 15:06 -0400

Module	statements	missing	excluded	branches	partial	coverage
C:\Users\perre\Documents\GitHub\requests\requests\__init__.py	68	24	0	10	4	62%
C:\Users\perre\Documents\GitHub\requests\requests\__version__.py	10	0	0	0	0	100%
C:\Users\perre\Documents\GitHub\requests\requests\_internal_utils.py	19	0	0	2	0	100%
C:\Users\perre\Documents\GitHub\requests\requests\adapters.py	216	15	0	82	8	92%
C:\Users\perre\Documents\GitHub\requests\requests\api.py	19	3	0	0	0	84%
C:\Users\perre\Documents\GitHub\requests\requests\auth.py	173	21	0	68	15	84%
C:\Users\perre\Documents\GitHub\requests\requests\certs.py	4	1	0	2	1	67%
C:\Users\perre\Documents\GitHub\requests\requests\compat.py	30	2	0	2	1	91%
C:\Users\perre\Documents\GitHub\requests\requests\cookies.py	239	53	0	106	9	72%
C:\Users\perre\Documents\GitHub\requests\requests\exceptions.py	37	0	0	52	0	100%
C:\Users\perre\Documents\GitHub\requests\requests\help.py	62	19	0	18	5	60%
C:\Users\perre\Documents\GitHub\requests\requests\hooks.py	14	0	0	10	0	100%
C:\Users\perre\Documents\GitHub\requests\requests\models.py	455	33	0	198	18	92%
C:\Users\perre\Documents\GitHub\requests\requests\packages.py	17	0	0	10	0	100%
C:\Users\perre\Documents\GitHub\requests\requests\sessions.py	268	12	0	104	5	95%
C:\Users\perre\Documents\GitHub\requests\requests\status_codes.py	14	0	0	10	0	100%
C:\Users\perre\Documents\GitHub\requests\requests\structures.py	39	0	0	12	0	100%
C:\Users\perre\Documents\GitHub\requests\requests\utils.py	482	65	0	214	12	86%
<b>Total</b>	<b>2166</b>	<b>248</b>	<b>0</b>	<b>900</b>	<b>78</b>	<b>87%</b>

coverage.py v6.4.4, created at 2022-09-22 15:06 -0400

Figure 6.4. Rapport de couverture de Request v2.28.1 après l'ajout des tests

En effet, on peut voir la valeur située à la colonne “missing” passer de 68 à 65. Trois nouvelles instructions sont donc belles et bien couvertes. Toutefois, la couverture du fichier reste inchangée.



**Question 7 : Veuillez discuter s’il y a des lignes qui sont couvertes par un outil mais pas par l’autre et votre processus d’enquête sur la raison. Le processus de vérification devrait être automatisé. Comparer 2 outils serait suffisant. On recommande coverage.py et pytest –cov. Il faut considérer la couverture avec des branches.**

Oui, il y a des lignes qui sont couvertes par un outil mais pas par l’autre. Pour vérifier le tout, un script a été rédigé.

Pour faire en sorte que le script puisse fonctionner, deux commandes ont été exécutées en premier lieu :

```
coverage run -m --branch pytest
coverage json
```

Celles-ci ont pour but de générer un fichier json contenant les lignes couvertes par coverage.py.

Ensuite, la commande suivante a été effectuée pour générer un fichier XML contenant les lignes couvertes par pytest -cov :

```
pytest --cov --cov-branch --cov-report=xml
```

Après, il faut s’assurer de déplacer les fichiers coverage.json et coverage.xml dans le même dossier que le script scriptq7.py, puis dans le fichier du script scriptq7.py, dans la section Edit Configurations, il faut cliquer sur le “+” ,choisir “Python” et mettre le chemin du fichier test.py dans le script path. Finalement, un clique droit doit être effectué sur le script et l’option “Run Test” doit être choisie. Un fichier sous le nom de “linedifferences.json” va apparaître contenant les lignes uniquement couvertent par l’outil coverage.py, et non par pytest.

La figure 7 présente un aperçu du JSON obtenu :

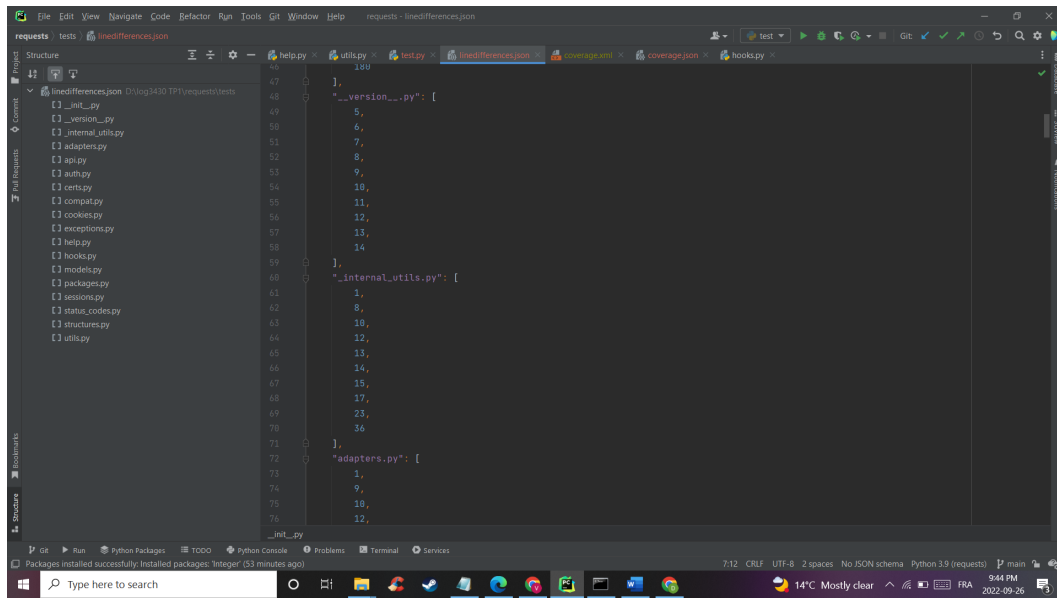


Figure 7. Aperçu du fichier linedifferences.json