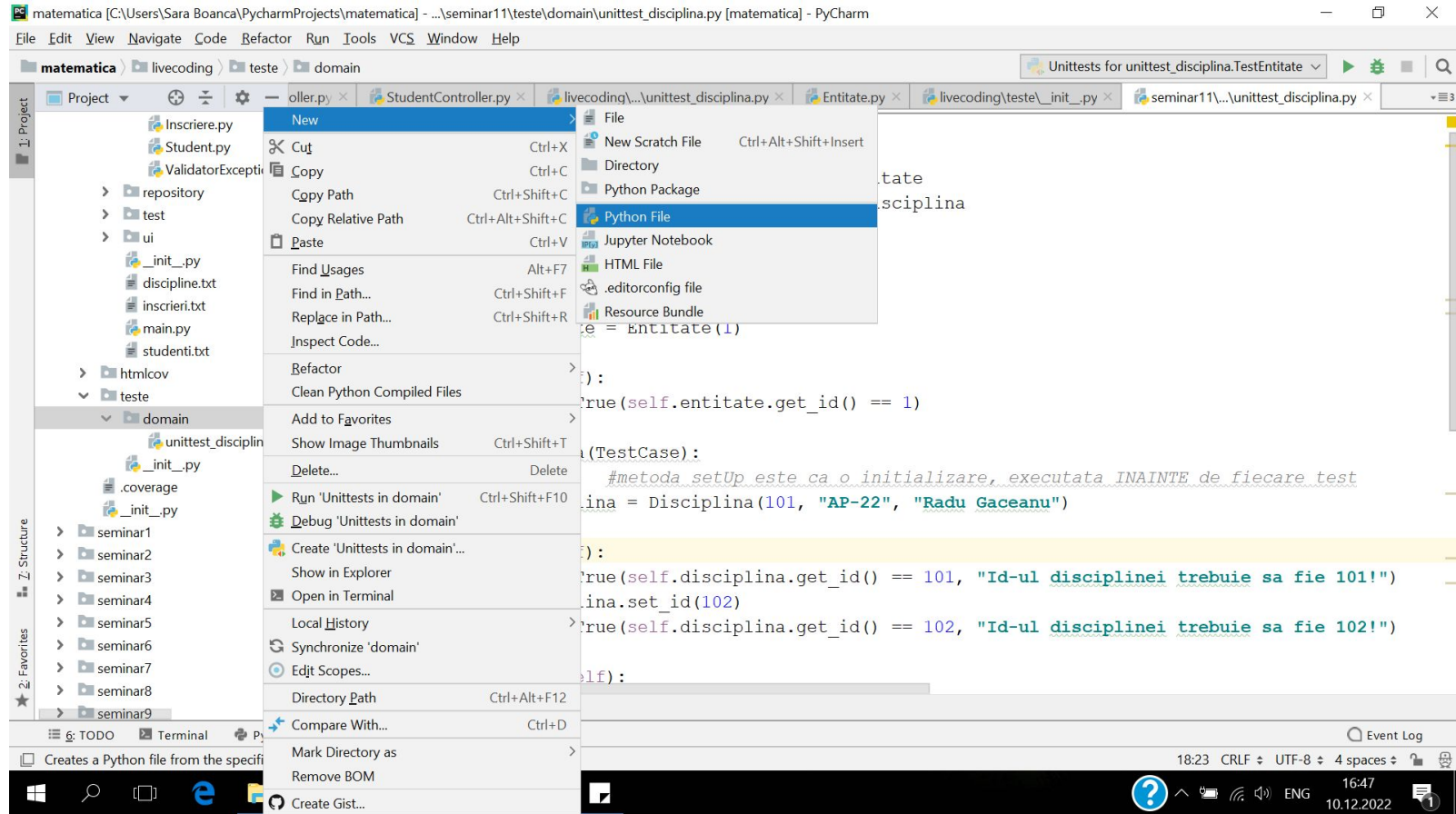
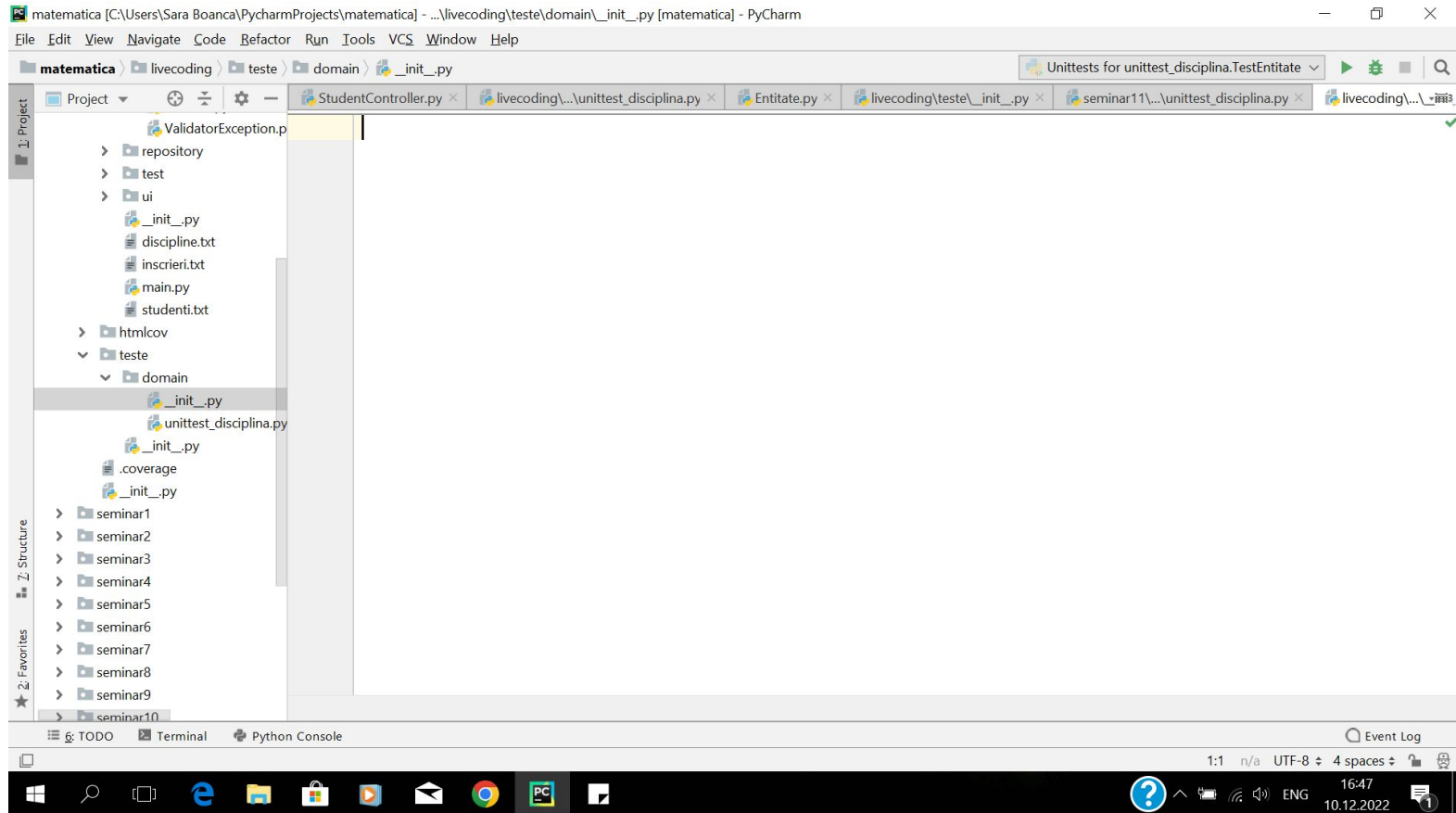


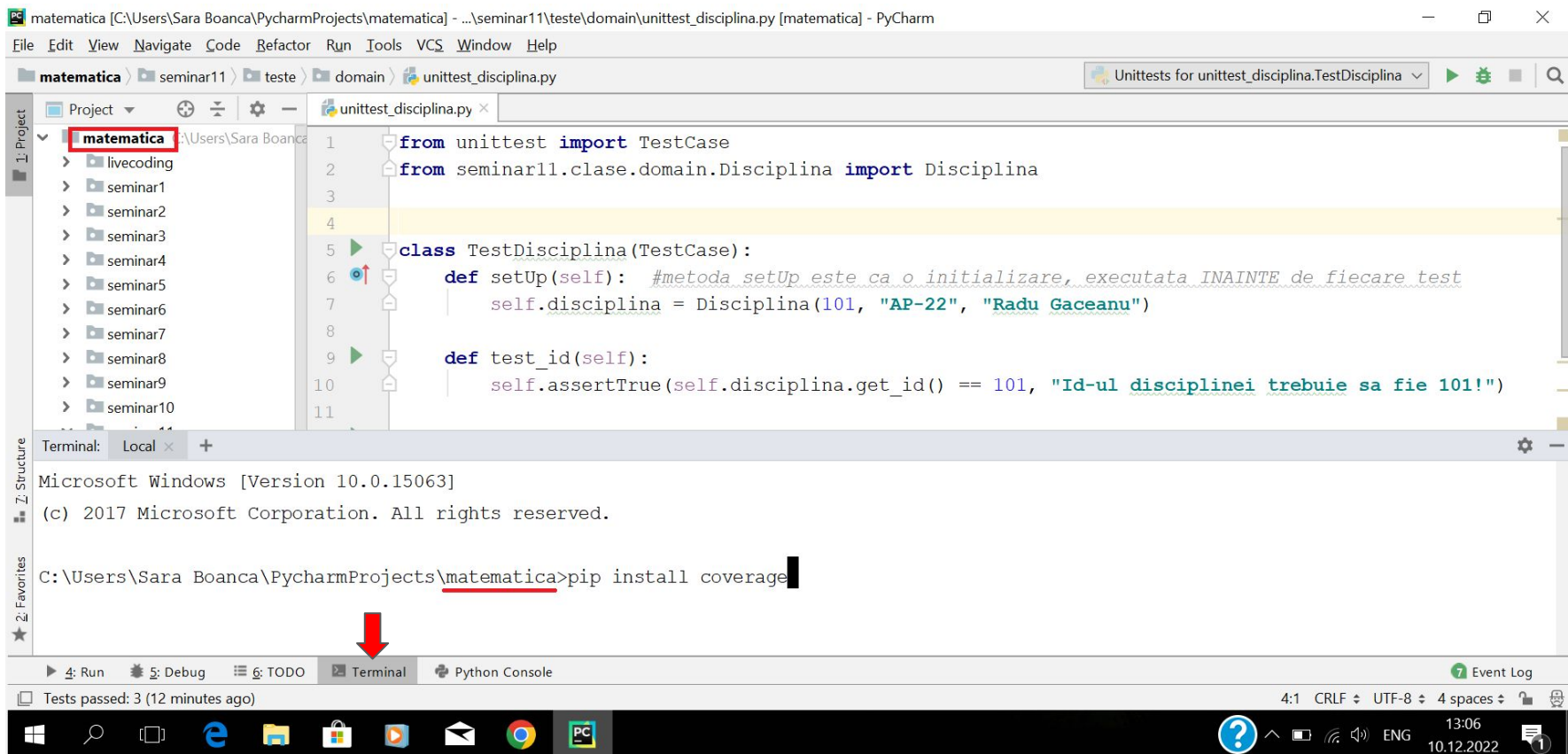
Înainte de toate, asigurați-vă că toate modulele (la mine ele se numesc: seminar11, clase, teste, domain, repository, ...) au fișierul `__init__.py` în interiorul lor. Acel fișier ne arată că `seminar11`, `clase`, `teste`, `domain`, ... sunt module Python. E important să faceți asta, ca să nu apară erori. Dacă există un modul unde nu aveți `__init__.py` (ex. `domain`), adăugați-l:



Denumiti fisierul `__init__` (cu doua `_` inainte si dupa `init`). E in regula daca ramane gol.
Observati ca acum apare adaugat in modulul domeniu.



În partea de jos a ecranului, dați click pe Terminal. Verificați să fiți în modulul principal. La mine acesta e matematica. Introduceți în terminal comanda: ***pip install coverage*** (și dați Enter). Asta va instala modulul predefinit Python care ne va ajuta să vedem ce procent din cod acoperă testele noastre.



Modulul coverage se va instala.

La fel cum ati rulat comanda pip install coverage, rulati acum comanda: ***pip install pytest*** (si dati Enter)

Dupa ce modulul pytest se instaleaza, rulati comanda: ***pip install pytest-cov*** (si dati Enter)

The screenshot shows the PyCharm IDE interface. The top toolbar includes buttons for Run, Debug, and Test. The main editor window displays the file `unittest_disciplina.py` with the following Python code:

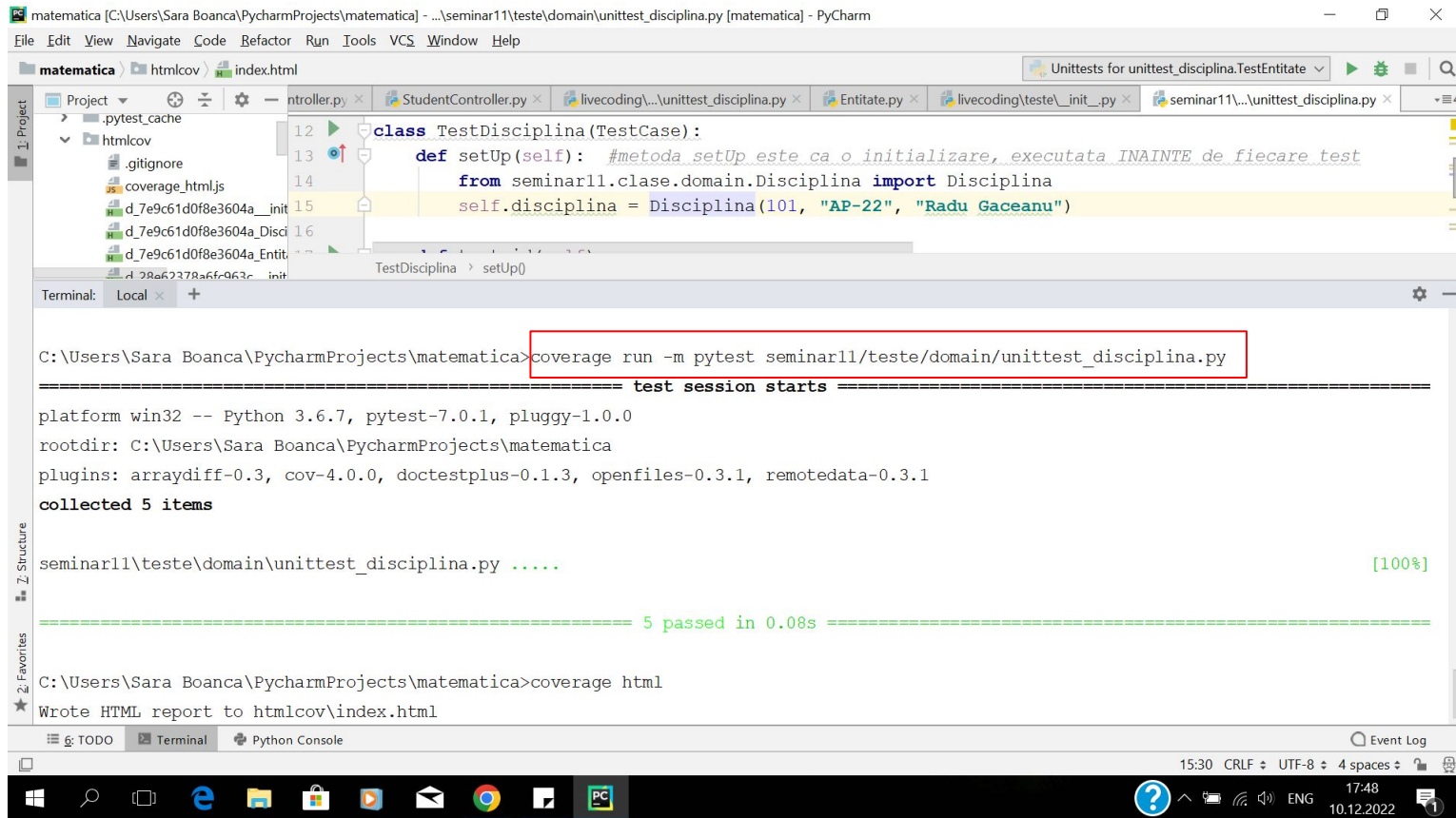
```
1 from unittest import TestCase
2 from seminar11.clase.domain.Disciplina import Disciplina
3
4
5 class TestDisciplina(TestCase):
6     def setUp(self): #metoda setUp este ca o initializare, executata INAINTE de fiecare test
7         self.disciplina = Disciplina(101, "AP-22", "Radu Gaceanu")
8
9     def test_id(self):
```

The bottom panel shows the Terminal output for the command `pip install coverage`:

```
C:\Users\Sara Boanca\PycharmProjects\matematica>pip install coverage
Collecting coverage
  Downloading coverage-6.2-cp36-cp36m-win_amd64.whl (183 kB)
    |████████████████████████████████████████| 183 kB 726 kB/s
Installing collected packages: coverage
Successfully installed coverage-6.2
WARNING: You are using pip version 21.0.1; however, version 21.3.1 is available.
You should consider upgrading via the 'c:\users\sara boanca\anaconda3\python.exe -m pip install --upgrade pip' command.
C:\Users\Sara Boanca\PycharmProjects\matematica>
```

The status bar at the bottom indicates "Tests passed: 3 (12 minutes ago)" and "4:1 CRLF UTF-8 4 spaces".

Scrieti in terminal comanda: **coverage run -m pytest seminar11/teste/domain/unittest_disciplina.py** (si dati Enter)
La mine unittest_disciplina.py este fisierul unde am scris unit testele si el este in modulul seminar11/teste/domain.
Ca sa va dati seama, puteti sa va uitati la ierarhia de clase din Seminar11 pe care l-am incarcat pe Teams.



The screenshot shows the PyCharm IDE interface. The top toolbar includes icons for File, Edit, View, Navigate, Code, Refactor, Run, Tools, VCS, Window, and Help. The main editor displays the file `unittest_disciplina.py` with the following Python code:

```
12 class TestDisciplina(TestCase):
13     def setUp(self): #metoda setUp este ca o initializare, executata INAINTE de fiecare test
14         from seminar11.clase.domain.Disciplina import Disciplina
15         self.disciplina = Disciplina(101, "AP-22", "Radu Gaceanu")
16
```

The left sidebar shows the Project structure with folders like `.pytest_cache` and `htmlcov`. The bottom panel shows the Terminal output:

```
C:\Users\Sara Boanca\PycharmProjects\matematica>coverage run -m pytest seminar11/teste/domain/unittest_disciplina.py
===== test session starts =====
platform win32 -- Python 3.6.7, pytest-7.0.1, pluggy-1.0.0
rootdir: C:\Users\Sara Boanca\PycharmProjects\matematica
plugins: arraydiff-0.3, cov-4.0.0, doctestplus-0.1.3, openfiles-0.3.1, remotedata-0.3.1
collected 5 items

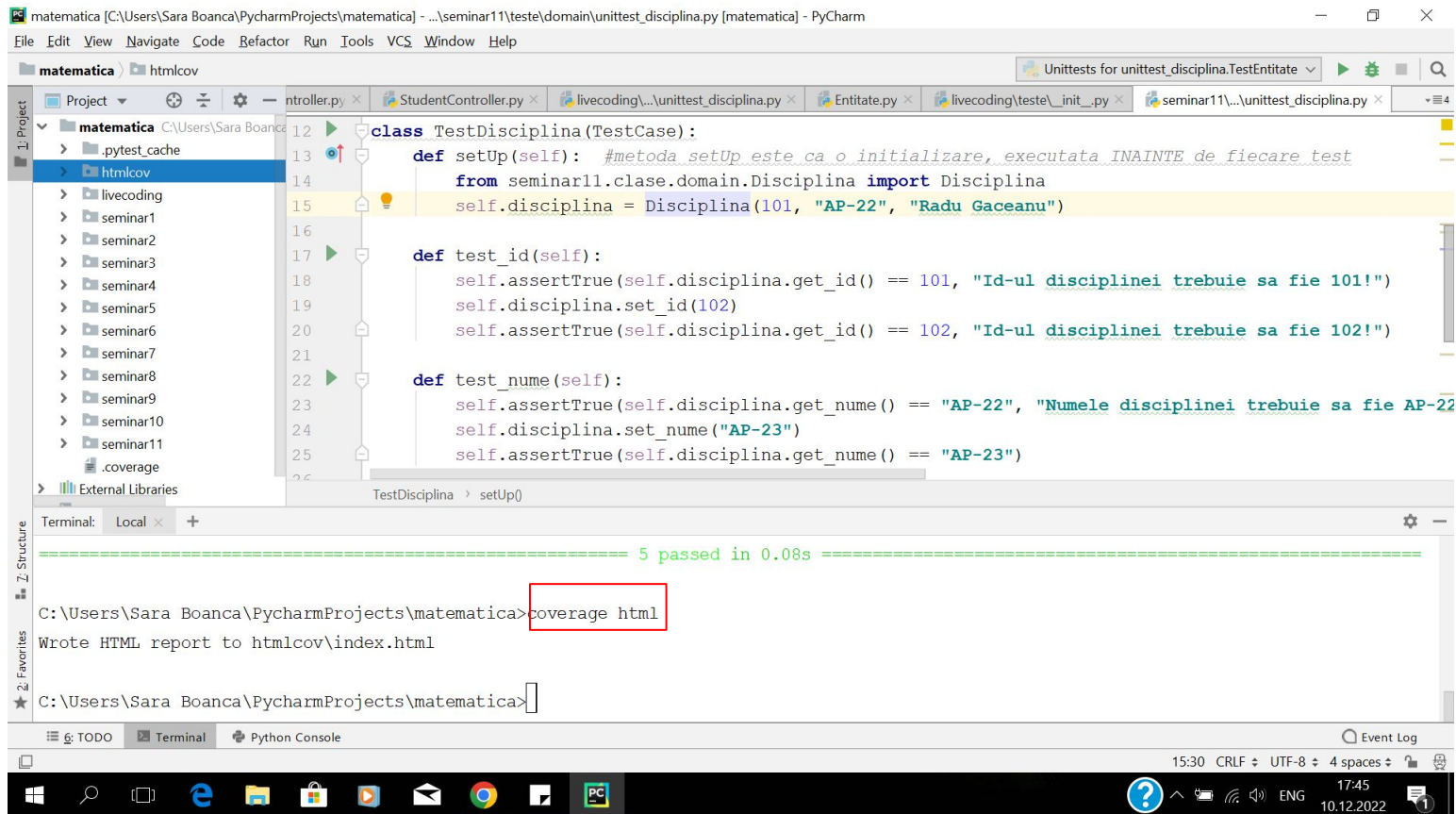
seminar11\teste\domain\unittest_disciplina.py ..... [100%]

===== 5 passed in 0.08s =====

C:\Users\Sara Boanca\PycharmProjects\matematica>coverage html
Wrote HTML report to htmlcov\index.html
```

The terminal output is highlighted with a red box around the command and a green box around the success message. The status bar at the bottom shows the time as 15:30, CRLF, UTF-8, 4 spaces, and the date 10.12.2022.

Comanda anterioara a rulat testele cu coverage. Scrieti apoi in terminal: **coverage html** (si dati Enter)
Vedeti ca in proiectul vostru a aparut un modul numit htmlcov. Dati click pe el si deschideti-l.



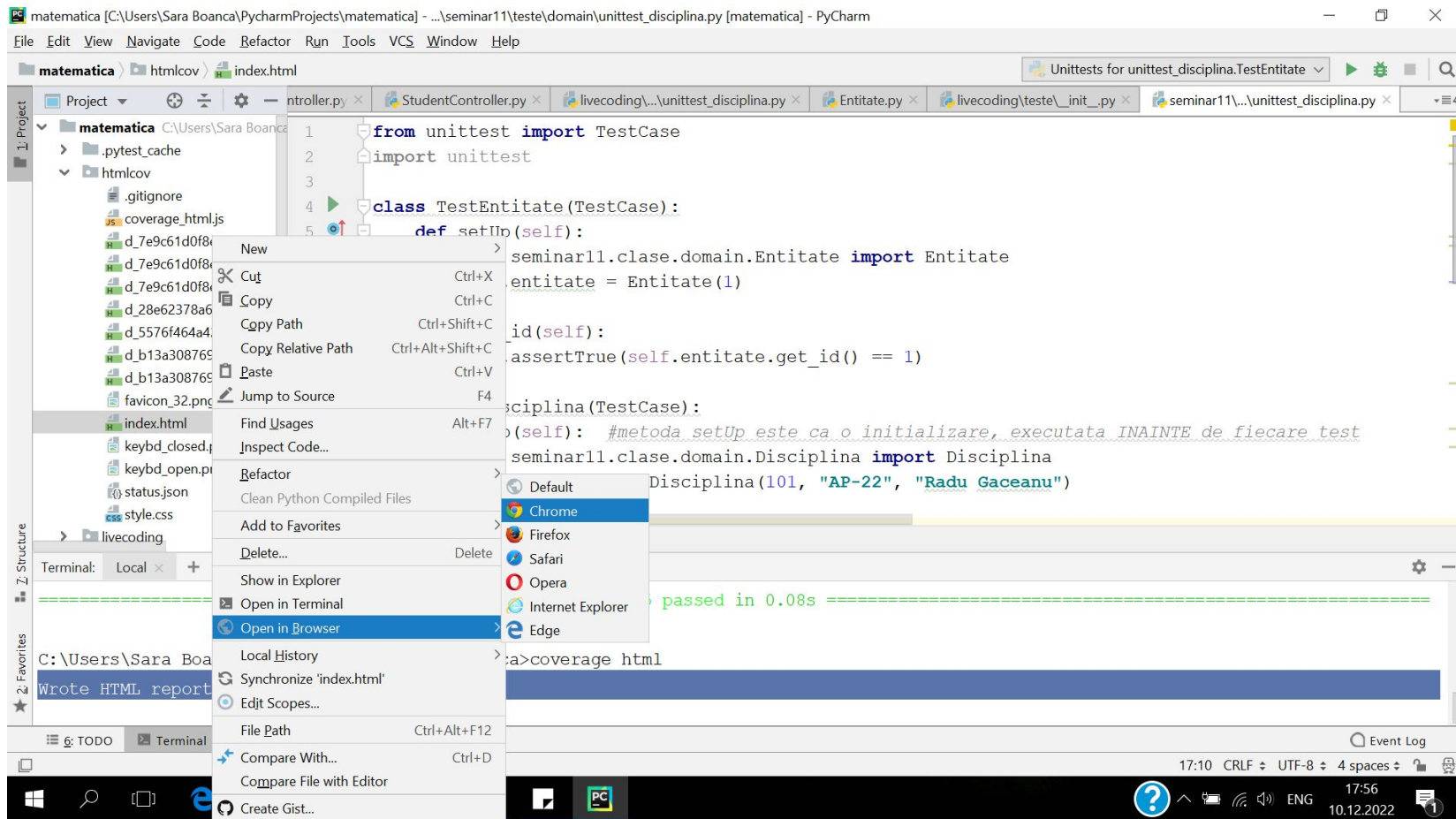
In htmlcov gasiti fisierul index.html. Acesta ne va ajuta sa vedem ce procent din cod acopera testele rulate.

The screenshot shows the PyCharm IDE interface. The top toolbar includes buttons for running tests, generating coverage, and searching. The main editor displays the file `seminar11\...\unittest_disciplina.py` with the following code:

```
1 from unittest import TestCase
2 import unittest
3
4 class TestEntitate(TestCase):
5     def setUp(self):
6         from seminar11.clase.domain.Entitate import Entitate
7         self.entitate = Entitate(1)
8
9     def test_id(self):
10        self.assertTrue(self.entitate.get_id() == 1)
11
12 class TestDisciplina(TestCase):
13     def setUp(self): #metoda setUp este ca o initializare, executata INAINTE de fiecare test
14         from seminar11.clase.domain.Disciplina import Disciplina
15         self.disciplina = Disciplina(101, "AP-22", "Radu Gaceanu")
16
```

The left sidebar shows the project structure with `index.html` highlighted under the `htmlcov` directory. The bottom terminal window shows the command `C:\Users\Sara Boanca\PycharmProjects\matematica>coverage html` and its output: `Wrote HTML report to htmlcov\index.html`. The status bar at the bottom indicates the current encoding is UTF-8 with 4 spaces.

Dati click dreapta pe fisierul index.html si deschideti-l in browser-ul vostru de internet. Eu l-am deschis in Chrome.



Veti fi dusi automat intr-o pagina unde vedeti procentajul de cod acoperit de unit testele executate pentru fiecare fisier. Observati ca pentru fisierul Entitate.py (din modulul clase/domain) are acoperire doar 89%. Puteti sa dati click pe acest fisier ca sa vedeti exact care linii din cod sunt acoperite de teste si care nu.

← → ↺ ⓘ http://localhost:63342/matematica/livecoding/htmlcov/index.html?_ijt=ai6nkutqeojo03j78deunko42o 🔍 ↗ ☆ 📷 ⚙️ ☰ 🖱️ 👤 ⋮

Coverage report: 98%

filter...



Module	statements	missing	excluded	coverage
__init__.py	0	0	0	100%
clase__init__.py	0	0	0	100%
clase\domain\Disciplina.py	16	0	0	100%
clase\domain\Entitate.py	9	1	0	89%
clase\domain__init__.py	0	0	0	100%
teste__init__.py	0	0	0	100%
teste\domain__init__.py	0	0	0	100%
teste\domain\unittest_disciplina.py	28	0	0	100%
Total	53	1	0	98%

coverage.py v6.2, created at 2022-12-10 15:47 +0200

Cu verde vedem liniile de cod acoperite de teste, iar cu rosu le vedem pe cele pe care testele inca nu le acopera. Asta inseamna ca metoda `__str__()` din clasa `Entitate` nu este testata de unit testele create. Sa ne intoarcem in cod si sa adaugam teste si pentru ea.

Coverage for **clase\domain\Entitate.py**: 89%



9 statements

8 run

1 missing

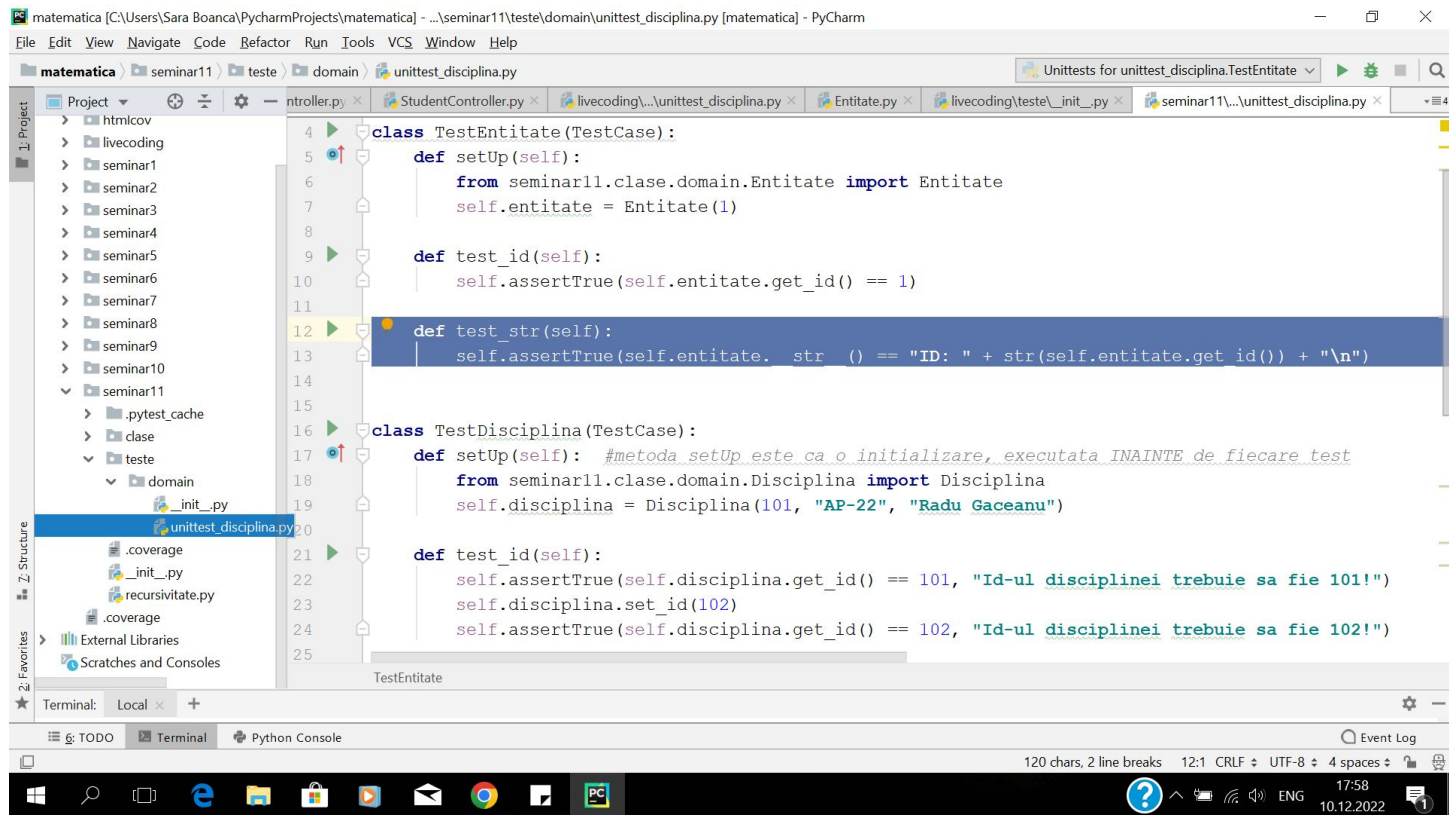
0 excluded

```
1 class Entitate:
2
3     def __init__(self, id):
4         self.__id = id
5
6     def get_id(self):
7         return self.__id
8
9     def set_id(self, id_nou):
10        self.__id = id_nou
11
12    def __str__(self):
13        return "ID: " + str(self.get_id()) + "\n"
```

Adaugam in seminar11/teste/domain/unittest_disciplina.py codul marcat.

Acum este testata si metoda `__str__()` din clasa Entitate.

Daca rulam din nou testele (exemplu Slide-ul 5) si in browser dam Refresh la pagina ce ne arata procentele de acoperire a codului, va aparea procentaj de 100% acoperire si pentru clasa Entitate.



```
1  class TestEntitate(TestCase):
2      def setUp(self):
3          from seminar11.clase.domain.Entitate import Entitate
4          self.entitate = Entitate(1)
5
6      def test_id(self):
7          self.assertTrue(self.entitate.get_id() == 1)
8
9      def test_str(self):
10         self.assertTrue(self.entitate.__str__() == "ID: " + str(self.entitate.get_id()) + "\n")
11
12  class TestDisciplina(TestCase):
13      def setUp(self): #metoda setUp este ca o initializare, executata INAINTE de fiecare test
14          from seminar11.clase.domain.Disciplina import Disciplina
15          self.disciplina = Disciplina(101, "AP-22", "Radu Gaceanu")
16
17      def test_id(self):
18          self.assertTrue(self.disciplina.get_id() == 101, "Id-ul disciplinei trebuie sa fie 101!")
19          self.disciplina.set_id(102)
20          self.assertTrue(self.disciplina.get_id() == 102, "Id-ul disciplinei trebuie sa fie 102!")
```

Acum codul din clasa Entitate este acoperit in proportie de 100%.

Va trebui sa faceti asta pentru toate clasele si metodele din proiectul vostru (mai putin cele din UI).
Succes!

← → ↺ ⓘ http://localhost:63342/matematica/livecoding/htmlcov/index.html?_ijt=ai6nkutqeojo03j78deunko42o 🔍 ↗ ☆ 📷 ⚙️ 📄 👤 ⋮

Coverage report: 100%



Module	statements	missing	excluded	coverage
__init__.py	0	0	0	100%
clase__init__.py	0	0	0	100%
clase\domain\Disciplina.py	16	0	0	100%
<u>clase\domain\Entitate.py</u>	9	0	0	100%
clase\domain__init__.py	0	0	0	100%
teste__init__.py	0	0	0	100%
teste\domain__init__.py	0	0	0	100%
teste\domain\unittest_disciplina.py	30	0	0	100%
Total	55	0	0	100%

coverage.py v6.2, created at 2022-12-10 16:04 +0200