Empathetic testing

Developing with compassion and humility

Marc Gibbons

October 17, 2023


NumPy


pandas

```
Periods: 3 yearly payments
Interest rate: 10%
Principal: $10,000

Expected schedule:

payment      principal   interest     balance
$ 4,021.15   $ 3,021.15  $ 1,000.00   $ 6,978.85
$ 4,021.15   $ 3,323.26  $ 697.89     $ 3,655.59
$ 4,021.15   $ 3,655.59  $ 365.56     $ 0.00
```

ppmt
ipmt

./manage.py test


ImportError
NameError
AttributeError
https://commons.wikimedia.org/wiki/File:Operation_Upshot-Knothole_-_Badger_001.jpg

AssertionError: Exception not raised

```
def test_range_exceeded(self):
    """
    Given that the current iteration's month exceeds the
    end month, the StopIteration exception should be raised.

    This means that this iterator will yield when the iteration
    month equals the end_month.
    """
    with self.assertRaises(StopIteration):
        self.generator.current_month = Month(2017, 1)
        self.generator.end_month = Month(2016, 12)
        self.sut()
```

git blame

Calculator

```
GET /calculator/?rate=0.1&principal=10000&number_of_periods=3
```

```
HTTP 200 OK
Allow: GET, OPTIONS
Content-Type: application/json
Vary: Accept

[
    {
        "balance": 6978.85,
        "interest": 1000.0,
        "payment": 4021.15,
        "principal": 3021.15
    },
    {
        "balance": 3655.59,
        "interest": 697.89,
        "payment": 4021.15,
        "principal": 3323.26
    },
    {
        "balance": 0.0,
        "interest": 365.56,
        "payment": 4021.15,
        "principal": 3655.59
    }
]
```

```
tests/test_admin.py .                                                    [  9%]
tests/test_api_calculator.py .                                           [ 18%]
tests/test_api_loans.py .                                                [ 27%]
tests/test_calculator.py .......                                         [ 90%]
tests/test_models.py .                                                   [100%]

---------- coverage: platform darwin, python 3.6.15-final-0 ----------
Name                                 Stmts   Miss  Cover
------------------------------------------------------
empathetic_mortgage/__init__.py          0      0   100%
empathetic_mortgage/settings.py         17      0   100%
loans/__init__.py                        0      0   100%
loans/admin.py                          13      0   100%
loans/apps.py                            4      0   100%
loans/calculators.py                    22      0   100%
loans/models.py                          8      0   100%
loans/views.py                          27      0   100%
tests/test_admin.py                     15      0   100%
tests/test_api_calculator.py            10      0   100%
tests/test_api_loans.py                 21      0   100%
tests/test_calculator.py                32      0   100%
tests/test_models.py                     4      0   100%
------------------------------------------------------
TOTAL                                  173      0   100%


============================================== 11 passed in 0.31s ==============================================
```

3.11

modified: requirements.txt

```
@ requirements.txt:4 @
Django==3.2.21
Django==4.2.5
djangorestframework==3.14.0
numpy==1.19.5
pandas==1.1.5
pytest-cov==4.0.0
numpy==1.26.0
pandas==2.1.1
pytest-cov==4.1.0
```

```
---------- coverage: platform darwin, python 3.11.1-final-0 --------
--
Name                                        Stmts   Miss  Cover
---------------------------------------------------------------
empathetic_mortgage/__init__.py                 0      0   100%
empathetic_mortgage/settings.py                17      0   100%
empathetic_mortgage/urls.py                     3      0   100%
loans/__init__.py                               0      0   100%
loans/admin.py                                 14      0   100%
loans/apps.py                                   4      0   100%
loans/calculators.py                           22      0   100%
loans/migrations/0001_initial.py                5      0   100%
loans/migrations/__init__.py                    0      0   100%
loans/models.py                                 8      0   100%
loans/urls.py                                   3      0   100%
loans/views.py                                 29      0   100%
tests/test_admin.py                            16      0   100%
tests/test_api_calculator.py                   11      0   100%
tests/test_api_loans.py                        22      0   100%
tests/test_calculator.py                       33      0   100%
tests/test_models.py                            4      0   100%
---------------------------------------------------------------
TOTAL                                         191      0   100%

================== 11 passed, 4 warnings in 0.54s ==================
```

RuntimeError at /calculator/

In accordance with NEP 32, the function ipmt was removed from NumPy version 1.20. A replacement for this function is available in the numpy_financial library: https://pypi.org/project/numpy-financial

Request Method:	GET
Request URL:	http://localhost:8000/calculator/?rate=0.1&principal=10000&number_of_periods=3
Django Version:	4.2.5
Exception Type:	RuntimeError
Exception Value:	`In accordance with NEP 32, the function ipmt was removed from NumPy version 1.20.  A replacement for this function is available in the numpy_financial library: https://pypi.org/project/numpy-financial`
Exception Location:	/Users/marcgibbons/.pyenv/versions/empathetic-testing/lib/python3.11/site-packages/numpy/__init__.py, line 303, in _expired
Raised during:	loans.views.calculator
Python Executable:	/Users/marcgibbons/.pyenv/versions/empathetic-testing/bin/python
Python Version:	3.11.1

Python Path:	`['/Users/marcgibbons/projects/empathetic-testing',`
	`'/Users/marcgibbons/.pyenv/versions/empathetic-testing/lib/python3.11/site-packages/_pdbpp_path_hack',`
	`'/Users/marcgibbons/.pyenv/versions/3.11.1/lib/python311.zip',`
	`'/Users/marcgibbons/.pyenv/versions/3.11.1/lib/python3.11',`
	`'/Users/marcgibbons/.pyenv/versions/3.11.1/lib/python3.11/lib-dynload',`
	`'/Users/marcgibbons/.pyenv/versions/empathetic-testing/lib/python3.11/site-packages']`

Server time:	Sun, 08 Oct 2023 01:15:02 +0000

Traceback

Switch to copy-and-paste view

/Users/marcgibbons/.pyenv/versions/empathetic-testing/lib/python3.11/site-packages/django/core/handlers/exception.py, line 55, in inner

```
55.                 response = get_response(request)
                               ^^^^^^^^^^^^^^^^^^^^^
```

▶ Local vars

/Users/marcgibbons/.pyenv/versions/empathetic-testing/lib/python3.11/site-packages/django/core/handlers/base.py, line 197, in _get_response

```
197.                 response = wrapped_callback(request, *callback_args, **callback_kwargs)
                               ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

▶ Local vars

/Users/marcgibbons/.pyenv/versions/empathetic-testing/lib/python3.11/site-packages/django/views/decorators/csrf.py, line 56, in wrapper_view

```
modified: requirements.txt

@ requirements.txt:3 @
Django==4.2.5
djangorestframework==3.14.0
numpy-financial==1.0.0
numpy==1.26.0
pandas==2.1.1
pytest-cov==4.1.0
```

```
modified: loans/calculators.py
```

```
@ loans/calculators.py:2 @
import numpy as np
import numpy_financial as npf


def get_amortization_schedule(principal, rate, number_of_periods):
@ loans/calculators.py:34 @ def get_per(number_of_periods):


def get_interest_paid(rate, per, nper, pv):
    return -np.round(np.ipmt(rate=rate, per=per, nper=nper, pv=pv), 2)
    return -np.round(npf.ipmt(rate=rate, per=per, nper=nper, pv=pv), 2)


def get_principal_paid(rate, per, nper, pv):
    return -np.round(np.ppmt(rate=rate, per=per, nper=nper, pv=pv), 2)
    return -np.round(npf.ppmt(rate=rate, per=per, nper=nper, pv=pv), 2)
```

Calculator

```
GET /calculator/?rate=0.1&principal=10000&number_of_periods=3
```

```
HTTP 200 OK
Allow: GET, OPTIONS
Content-Type: application/json
Vary: Accept

[
    {
        "balance": 6978.85,
        "interest": 1000.0,
        "payment": 4021.15,
        "principal": 3021.15
    },
    {
        "balance": 3655.59,
        "interest": 697.89,
        "payment": 4021.15,
        "principal": 3323.26
    },
    {
        "balance": 0.0,
        "interest": 365.56,
        "payment": 4021.15,
        "principal": 3655.59
    }
]
```

```
tests/test_api_calculator.py .                                           [  9%]
tests/test_admin.py .                                                    [ 18%]
tests/test_api_loans.py .                                                [ 27%]
tests/test_calculator.py .FF....                                         [ 90%]
tests/test_models.py .                                                   [100%]

================================================ FAILURES =================================================
_____________________________________ test_get_interest_paid ______________________________________

    def test_get_interest_paid():
        kwargs = {"rate": 0.1, "per": [1, 2, 3], "nper": 3, "pv": 10_000}
        with patch("numpy.ipmt", return_value=-0.333) as mock_ipmt:
            interest_paid = calculators.get_interest_paid(**kwargs)

>       assert interest_paid == 0.33
E       ValueError: The truth value of an array with more than one element is ambiguous. Use a.any() or a.a
ll()

tests/test_calculator.py:16: ValueError
_____________________________________ test_get_principal_paid ____________________________________
```

Implementation bias

```python
# tests/test_api_calculator.py
from unittest.mock import patch

from django.test import RequestFactory
from loans.views import calculator


@patch("loans.views.get_amortization_schedule")
def test_calculator_api(mock):
    rf = RequestFactory()
    data = {"principal": 10_000, "rate": 0.1, "number_of_periods": 3}
    request = rf.get("/schedule/", data)
    response = calculator(request)

    mock.assert_called_once_with(**data)
    assert response.status_code == 200
    assert response.data == mock.return_value

```

forecloses refactoring

wrong url

?

A better API test

An empathetic test

empathy

relate to others
feel the feelings

compassion

rooted in empathy
do something about it

humility

things change
your code is **discardable**

Calculator

```
GET /calculator/?rate=0.1&principal=10000&number_of_periods=3
```

```
HTTP 200 OK
Allow: GET, OPTIONS
Content-Type: application/json
Vary: Accept

[
    {
        "balance": 6978.85,
        "interest": 1000.0,
        "payment": 4021.15,
        "principal": 3021.15
    },
    {
        "balance": 3655.59,
        "interest": 697.89,
        "payment": 4021.15,
        "principal": 3323.26
    },
    {
        "balance": 0.0,
        "interest": 365.56,
        "payment": 4021.15,
        "principal": 3655.59
    }
]
```

```python
# tests/test_api_calculator.py
from rest_framework.test import APIClient


def test_api_10_000_loan_with_3_payments_over_3_years_at_10_percent():
    data = {"principal": 10_000, "rate": 0.1, "number_of_periods": 3}
    client = APIClient()
    response = client.get("/calculator/", data)
    assert response.status_code == 200

    result = response.json()
    expected = [
        {"balance": 6_978.85, "interest": 1_000, "payment": 4_021.15, "principal": 3_021.15},
        {"balance": 3_655.59, "interest": 697.89, "payment": 4_021.15, "principal": 3_323.26},
        {"balance": 0, "interest": 365.56, "payment": 4_021.15, "principal": 3_655.59},
    ]
    assert result == expected
```

Real request

Real assertion

```
========================================= test session starts ==========================================
platform darwin -- Python 3.11.1, pytest-7.4.2, pluggy-1.3.0
django: settings: empathetic_mortgage.settings (from ini)
rootdir: /Users/marcgibbons/projects/empathetic-testing
configfile: pyproject.toml
plugins: cov-4.1.0, django-4.5.2
collected 1 item

tests/test_api_calculator.py .                                                              [100%]

---------- coverage: platform darwin, python 3.11.1-final-0 ----------
Name                    Stmts   Miss  Cover
-----------------------------------------
loans/calculators.py       23      1    96%
-----------------------------------------
TOTAL                      23      1    96%


========================================== 1 passed in 0.24s ===========================================
```

Coverage for **loans/calculators.py**: 96%

23 statements | 22 run | 1 missing | 0 excluded

« prev ^ index » next coverage.py

```
1  import numpy as np
2  import numpy_financial as npf
3
4
5  def get_amortization_schedule(principal, rate, number_of_periods):
6      if not number_of_periods:
7          return []
8
9      pv = principal
10     nper = number_of_periods
11
12     per = get_per(number_of_periods)
13     interest_paid = get_interest_paid(rate, per, nper, pv)
14     principal_paid = get_principal_paid(rate, per, nper, pv)
15     payment = get_payment(interest_paid, principal_paid)
16     balance = get_balance(principal, principal_paid)
```

```
35
36 def test_api_with_0_periods():
37     data = {"principal": 10_000, "rate": 0.1, "number_of_periods": 0}
38
39     client = APIClient()
40     response = client.get("/calculator/", data)
41     assert response.status_code == 200
42     assert response.json() == []
43
```

```
========================================= test session starts ==========================================
platform darwin -- Python 3.11.1, pytest-7.4.2, pluggy-1.3.0
django: settings: empathetic_mortgage.settings (from ini)
rootdir: /Users/marcgibbons/projects/empathetic-testing
configfile: pyproject.toml
plugins: cov-4.1.0, django-4.5.2
collected 2 items

tests/test_api_calculator.py ..                                                          [100%]

---------- coverage: platform darwin, python 3.11.1-final-0 ----------
Name                    Stmts   Miss  Cover
-----------------------------------------
loans/calculators.py       23      0   100%
-----------------------------------------
TOTAL                      23      0   100%


========================================== 2 passed in 0.24s ===========================================
```

Start typing to filter...

AUTHENTICATION AND AUTHORIZATION

Groups + Add

Users + Add

LOANS

Loans + Add

Change loan

HISTORY

$10,000.00 – 3 years at 10.000%

Principal: 10000.0

Opening loan principal

Rate: 0.1

Rate per period

Number of periods: 3

Number of payments (years)

Amortization schedule:

BALANCE	INTEREST	PRINCIPAL	PAYMENT
6978.85	1000.00	3021.15	4021.15
3655.59	697.89	3323.26	4021.15
0.00	365.56	3655.59	4021.15

SAVE | Save and add another | Save and continue editing | Delete

admin.py • test_admin.py •

```python
from unittest.mock import patch

from django.contrib import admin

from loans.admin import LoanAdmin
from loans.models import Loan


@patch("loans.calculators.get_amortization_schedule")
@patch("pandas.DataFrame.from_dict")
def test_admin_change_view_renders_schedule(dataframe_mock, schedule_mock):
    expected = "<div>schedule</div>"
    to_html = dataframe_mock.return_value.to_html
    to_html.return_value = expected

    loan_admin = LoanAdmin(Loan, admin.site)
    loan = Loan()

    result = loan_admin.amortization_schedule(loan)
    assert result == expected

    schedule_mock.assert_called_once_with(
        number_of_periods=loan.number_of_periods,
        principal=loan.principal,
        rate=loan.rate,
    )
    dataframe_mock.assert_called_once_with(schedule_mock.return_value)
```

Registered to admin.site?

Are admin.urls included?

Values?

"private" method?

?

```python
# tests/test_admin.py
import pytest
from django.contrib.auth import get_user_model
from django.test import Client

from loans.models import Loan

pytestmark = pytest.mark.django_db


def test_admin_10_000_loan_with_3_payments_over_3_years_at_10_percent():
    user = get_user_model().objects.create(is_staff=True, is_superuser=True)
    client = Client()
    client.force_login(user)

    loan = Loan.objects.create(rate=0.1, number_of_periods=3, principal=10_000)
    expected = [
        {"balance": 6_978.85, "interest": 1_000, "payment": 4_021.15, "principal": 3_021.15},
        {"balance": 3_655.59, "interest": 697.89, "payment": 4_021.15, "principal": 3_323.26},
        {"balance": 0, "interest": 365.56, "payment": 4_021.15, "principal": 3_655.59}

    response = client.get(f"/admin/loans/loan/{loan.pk}/change/")
    assert response.status_code == 200
```

Superuser

Real object

Real request

```
<table border="1" class="dataframe schedule">
  <thead>
    <tr style="text-align: right;">
      <th>balance</th>
      <th>interest</th>
      <th>principal</th>
      <th>payment</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>6978.85</td>
      <td>1000.00</td>
      <td>3021.15</td>
      <td>4021.15</td>
    </tr>
    <tr>
      <td>3655.59</td>
      <td>697.89</td>
      <td>3323.26</td>
      <td>4021.15</td>
    </tr>
    <tr>
      <td>0.00</td>
      <td>365.56</td>
      <td>3655.59</td>
      <td>4021.15</td>
    </tr>
  </tbody>
</table>
```

pip install pyquery

```
...
from pyquery import PyQuery as pq
...
```

```
23
24      response = client.get(f"/admin/loans/loan/{loan.pk}/change/")
25      assert response.status_code == 200
26
27      dom = pq(response.content)
28      table = dom.find("table.schedule")
29      # >>> table.text()
30      #
31      #   'balance\ninterest\npayment\nprincipal\n'
32      #   '6978.85\n1000.00\n4021.15\n3021.15\n'
33      #   ...
```

```
35
36      # >>> tr = table.find("tr")
37      # >>> tr.text_content()
38      #   '\n      balance\n      interest\n      payment\n      principal\n    '
39
40      data = [
41          row.text_content().strip().replace(" ", "").split("\n")
42          for row in table.find("tr")
43      ]
44
45      # [
46      #   ['balance', 'interest', 'payment', 'principal'],
47      #   ['6978.85', '1000.00', '4021.15', '3021.15'],
48      #   ['3655.59', '697.89', '4021.15', '3323.26'],
49      #   ['0.00', '365.56', '4021.15', '3655.59']
50      # ]
51
```

```python
import pandas as pd

...
```

```python
    # [
    #   ['balance', 'interest', 'payment', 'principal'],
    #   ['6978.85', '1000.00', '4021.15', '3021.15'],
    #   ['3655.59', '697.89', '4021.15', '3323.26'],
    #   ['0.00', '365.56', '4021.15', '3655.59']
    # ]

    df = pd.DataFrame(data[1:], columns=data[0])
    #    balance interest  payment principal
    # 0  6978.85  1000.00  4021.15   3021.15
    # 1  3655.59   697.89  4021.15   3323.26
    # 2     0.00   365.56  4021.15   3655.59

    df = df.astype(float)  # Values parsed as strings, cast as floats.
    result = df.to_dict("records")

```

```
    result = df.to_dict("records")

    expected = [
        {
            "balance": 6_978.85,
            "interest": 1_000,
            "payment": 4_021.15,
            "principal": 3_021.15,
        },
        {
            "balance": 3_655.59,
            "interest": 697.89,
            "payment": 4_021.15,
            "principal": 3_323.26,
        },
        {
            "balance": 0,
            "interest": 365.56,
            "payment": 4_021.15,
            "principal": 3_655.59,
        },
    ]
    assert result == expected
```

```
========================================= test session starts ==========================================
platform darwin -- Python 3.11.1, pytest-7.4.2, pluggy-1.3.0
django: settings: empathetic_mortgage.settings (from ini)
rootdir: /Users/marcgibbons/projects/empathetic-testing
configfile: pyproject.toml
plugins: cov-4.1.0, django-4.5.2
collected 1 item

tests/test_admin.py .                                                                    [100%]

---------- coverage: platform darwin, python 3.11.1-final-0 ----------
Name                    Stmts   Miss  Cover
-----------------------------------------
loans/calculators.py       23      1    96%
-----------------------------------------
TOTAL                      23      1    96%


========================================== 1 passed in 0.69s ===========================================
```

HTTP request

to a third-party

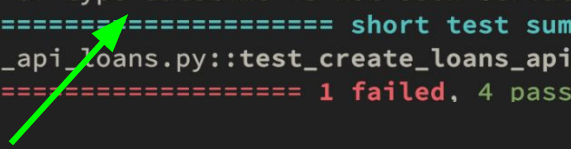
```python
# tests/test_api_loans.py

@patch("loans.views.requests")
def test_create_loan_makes_http_request(requests_mock):
    client = APIClient()
    data = {"principal": 10_000, "rate": 0.1, "number_of_periods": 3}
    response = client.post("/loans/", data)
    assert response.status_code == 201

    result = response.json()
    loan = Loan.objects.get(pk=result["id"])

    schedule = [
        {"balance": 6_978.85, "interest": 1_000, "payment": 4_021.15, "principal": 3_021.15},
        {"balance": 3_655.59, "interest": 697.89, "payment": 4_021.15, "principal": 3_323.26},
        {"balance": 0, "interest": 365.56, "payment": 4_021.15, "principal": 3_655.59},
    ]
    requests_mock.post.assert_called_once_with(
        "https://notifications.test",
        json={
            "loan_id": loan.pk,
            "created_datetime": loan.created_datetime,
            "schedule": schedule,
        },
    )
```

pip install responses

```python
# tests/test_api_loans.py
import json

import pytest
import responses
from rest_framework.test import APIClient

from loans.models import Loan

pytestmark = pytest.mark.django_db


@responses.activate
def test_create_loans_api():
    responses.post("https://notifications.test")  # Register HTTP call

    client = APIClient()
    data = {"principal": 10_000, "rate": 0.1, "number_of_periods": 3}
    response = client.post("/loans/", data)
    ...

```

```
33
34      ...
35
36      assert responses.calls, "Expected HTTP call"
37      request = responses.calls[0].request  # Grab first (and only) request
38      assert request.url == "https://notifications.test/"
39      assert request.method == "POST"
40      sent_data = json.loads(request.body)
41
42      assert sent_data == {
43          "loan_id": loan.pk,
44          "created_datetime": loan.created_datetime,
45          "schedule": schedule,
46      }
```

```
          ....................................................
  File "/Users/marcgibbons/.pyenv/versions/3.11.1/envs/empathetic-testing/lib/python3.11/site-packages/requests/sess
ions.py", line 486, in prepare_request
    p.prepare(
  File "/Users/marcgibbons/.pyenv/versions/3.11.1/envs/empathetic-testing/lib/python3.11/site-packages/requests/mode
ls.py", line 371, in prepare
    self.prepare_body(data, files, json)
  File "/Users/marcgibbons/.pyenv/versions/3.11.1/envs/empathetic-testing/lib/python3.11/site-packages/requests/mode
ls.py", line 511, in prepare_body
    body = complexjson.dumps(json, allow_nan=False)
           ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/Users/marcgibbons/.pyenv/versions/3.11.1/lib/python3.11/json/__init__.py", line 238, in dumps
    **kw).encode(obj)
          ^^^^^^^^^^
  File "/Users/marcgibbons/.pyenv/versions/3.11.1/lib/python3.11/json/encoder.py", line 200, in encode
    chunks = self.iterencode(o, _one_shot=True)
             ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "/Users/marcgibbons/.pyenv/versions/3.11.1/lib/python3.11/json/encoder.py", line 258, in iterencode
    return _iterencode(o, 0)
           ^^^^^^^^^^^^^^^^^
  File "/Users/marcgibbons/.pyenv/versions/3.11.1/lib/python3.11/json/encoder.py", line 180, in default
    raise TypeError(f'Object of type {o.__class__.__name__} '
TypeError: Object of type datetime is not JSON serializable
=========================================== short test summary info ============================================
FAILED tests/test_api_loans.py::test_create_loans_api - TypeError: Object of type datetime is not JSON serializable
========================================== 1 failed, 4 passed in 0.72s ==========================================
```



Serialize datetime as ISO string

---

modified: loans/views.py

```
@ loans/views.py:53 @ def loans(request):
        json={
            "loan_id": loan.pk,
            "schedule": schedule.to_dict("records"),
            "created_datetime": loan.created_datetime,
            "created_datetime": loan.created_datetime.isoformat(),
        },
    )
```

---

modified: tests/test_api_loans.py

```
@ tests/test_api_loans.py:54 @ def test_create_loans_api():

    assert sent_data == {
        "loan_id": loan.pk,
        "created_datetime": loan.created_datetime,
        "created_datetime": loan.created_datetime.isoformat(),
        "schedule": schedule,
    }
```

assert True

```python
from django.db import models


class Loan(models.Model):
    """
    A model representing a loan.

    In this simple implementation, loans are paid annually.
    """

    number_of_periods = models.PositiveSmallIntegerField(
        help_text="Number of payments (years)",
    )
    principal = models.FloatField(help_text="Opening loan principal")
    rate = models.FloatField(help_text="Rate per period")
    created_datetime = models.DateTimeField(auto_now_add=True)

```

Hard to test

pip install freezegun

modified: tests/test_api_loans.py

```
@ tests/test_api_loans.py:6 @ import json

import pytest
import responses
from freezegun import freeze_time
from rest_framework.test import APIClient


@freeze_time("2023-10-17T16:00Z")
@responses.activate
def test_create_loans_api():
    responses.post("https://notifications.test")  # Register HTTP call
@ tests/test_api_loans.py:22 @ def test_create_loans_api():
    client = APIClient()
    data = {"principal": 10_000, "rate": 0.1, "number_of_periods": 3}
    response = client.post("/loans/", data)

    assert response.status_code == 201

    result = response.json()
@ tests/test_api_loans.py:57 @ def test_create_loans_api():

    assert sent_data == {
        "loan_id": loan.pk,
        "created_datetime": loan.created_datetime.isoformat(),
        "created_datetime": "2023-10-17T16:00:00+00:00",
        "schedule": schedule,
    }
```

Implementation agnostic

Refactor

Before & after

calculators.py

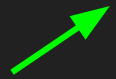
```python
import numpy as np
import numpy_financial as npf


def get_amortization_schedule(principal, rate, number_of_periods):
    if not number_of_periods:
        return []

    pv = principal
    nper = number_of_periods

    per = get_per(number_of_periods)
    interest_paid = get_interest_paid(rate, per, nper, pv)
    principal_paid = get_principal_paid(rate, per, nper, pv)
    payment = get_payment(interest_paid, principal_paid)
    balance = get_balance(principal, principal_paid)

    return [
        {
            "balance": balance[i],
            "interest": interest_paid[i],
            "payment": payment[i],
            "principal": principal_paid[i],
        }
        for i in range(number_of_periods)
    ]


def get_per(number_of_periods):
    return np.arange(number_of_periods) + 1


def get_interest_paid(rate, per, nper, pv):
    return -np.round(npf.ipmt(rate=rate, per=per, nper=nper, pv=pv), 2)


def get_principal_paid(rate, per, nper, pv):
    return -np.round(npf.ppmt(rate=rate, per=per, nper=nper, pv=pv), 2)


def get_payment(interest_paid, principal_paid):
    return np.round(interest_paid + principal_paid, 2)


def get_balance(principal, principal_paid):
    return principal - principal_paid.cumsum()
```

calculators.py

```
import numpy as np
import numpy_financial as npf
import pandas as pd


def get_amortization_schedule(principal, rate, number_of_periods):
    if not number_of_periods:
        return pd.DataFrame()

    npf_kwargs = {
        "nper": number_of_periods,
        "per": np.arange(number_of_periods) + 1,
        "pv": principal,
        "rate": rate,
    }
    principal_paid = -npf.ppmt(**npf_kwargs)
    interest_paid = -npf.ipmt(**npf_kwargs)

    df = pd.DataFrame(
        {
            "balance": principal - principal_paid.cumsum(),
            "interest": interest_paid,
            "principal": principal_paid,
            "payment": interest_paid + principal_paid,
        }
    )
    return df.round(2)
```

→ pd.DataFrame

```
loans/admin.py       |  4 +---
loans/calculators.py | 57 ++++++++++++++++++++----------------------------
loans/views.py       |  5 +++--
3 files changed, 23 insertions(+), 43 deletions(-)
```



Application code changed.
Tests did not.

```
========================================= test session starts ==========================================
platform darwin -- Python 3.11.1, pytest-7.4.2, pluggy-1.3.0
django: settings: empathetic_mortgage.settings (from ini)
rootdir: /Users/marcgibbons/projects/empathetic-testing
configfile: pyproject.toml
plugins: cov-4.1.0, django-4.5.2
collected 4 items

tests/test_api_loans.py .                                                              [ 25%]
tests/test_admin.py .                                                                  [ 50%]
tests/test_api_calculator.py ..                                                        [100%]

---------- coverage: platform darwin, python 3.11.1-final-0 ----------
Name                    Stmts   Miss  Cover
-----------------------------------------
loans/calculators.py       11      0   100%
-----------------------------------------
TOTAL                      11      0   100%


========================================== 4 passed in 0.78s ===========================================
```


if name in _builtins and isinstance(target, ModuleType):
self.create = True
if not self.create and original is DEFAULT:
> raise AttributeError(
"%s does not have the attribute %r" % (target, name)
)
E AttributeError: <module 'loans.calculators' from '/Users/marcgibbons/projects/empathetic-testing/loans/calculators.py'> does not have the attribute 'get_payment'
../../.pyenv/versions/3.11.1/lib/python3.11/unittest/mock.py:1410: AttributeError
short test summary info
FAILED tests/test_calculator.py::test_get_per - AttributeError: module 'loans.calculators' has no attribute 'get_per'
FAILED tests/test_calculator.py::test_get_interest_paid - AttributeError: module 'loans.calculators' has no attribute 'get_interest_paid'
FAILED tests/test_calculator.py::test_get_principal_paid - AttributeError: module 'loans.calculators' has no attribute 'get_principal_paid'
FAILED tests/test_calculator.py::test_get_balance - AttributeError: module 'loans.calculators' has no attribute 'get_balance'
FAILED tests/test_calculator.py::test_get_payment - AttributeError: module 'loans.calculators' has no attribute 'get_payment'
FAILED tests/test_calculator.py::test_get_loan_amortization_schedule - AttributeError: <module 'loans.calculators' from '/Users/marcgibbons/projects/empathetic-tes...
6 failed, 6 passed in 0.45s
Implementation bias

```
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        deleted:    tests/test_calculator.py
```

Rethinking the unit

```python
@pytest.mark.parametrize(
    "get_schedule",
    [
        get_schedule_from_calculator_api,
        get_schedule_from_loan_admin,
        get_schedule_sent_on_loan_create,
    ],
)
def test_10_000_loan_with_3_payments_over_3_years_at_10_percent(get_schedule):
    schedule = get_schedule(principal=10_000, rate=0.1, number_of_periods=3)
    assert schedule == [
        {"balance": 6_978.85, "interest": 1_000, "payment": 4_021.15, "principal": 3_021.15},
        {"balance": 3_655.59, "interest": 697.89, "payment": 4_021.15, "principal": 3_323.26},
        {"balance": 0, "interest": 365.56, "payment": 4_021.15, "principal": 3_655.59},
    ]
```

Public ways to get the schedule

Focus.
Given an input, expect a result.

```python
# tests/test_calculators.py_
import pytest
from django.test import Client
from rest_framework.test import APIClient


def get_schedule_from_calculator_api(**kwargs):
    client = APIClient()
    response = client.get("/calculator/", data=kwargs)
    assert response.status_code == 200
    return response.json()

```

```python
def get_schedule_from_loan_admin(**kwargs):
    user = get_user_model().objects.create(is_staff=True, is_superuser=True)
    client = Client()
    client.force_login(user)
    loan = Loan.objects.create(**kwargs)

    response = client.get(f"/admin/loans/loan/{loan.pk}/change/")
    assert response.status_code == 200

    dom = pq(response.content)
    table = dom.find("table.schedule")
    data = [
        row.text_content().strip().replace(" ", "").split("\n")
        for row in table.find("tr")
    ]
    df = pd.DataFrame(data[1:], columns=data[0])
    df = df.astype(float)  # Values parsed as strings, cast as floats.
    return df.to_dict("records")

```

```python
@responses.activate
def get_schedule_sent_on_loan_create(**kwargs):
    responses.post("https://notifications.test")  # Register HTTP call

    client = APIClient()
    response = client.post("/loans/", data=kwargs)
    assert response.status_code == 201

    assert responses.calls, "Expected HTTP call"
    request = responses.calls[0].request  # Grab first (and only) request
    sent_data = json.loads(request.body)
    return sent_data["schedule"]
```

```
================================= test session starts ==================================
platform darwin -- Python 3.11.1, pytest-7.4.2, pluggy-1.3.0
django: settings: empathetic_mortgage.settings (from ini)
rootdir: /Users/marcgibbons/projects/empathetic-testing
configfile: pyproject.toml
plugins: cov-4.1.0, django-4.5.2
collected 3 items

tests/test_calculators.py ...                                                  [100%]

---------- coverage: platform darwin, python 3.11.1-final-0 ----------
Name                    Stmts   Miss  Cover
-----------------------------------------
loans/calculators.py       11      1    91%
-----------------------------------------
TOTAL                      11      1    91%


================================== 3 passed in 0.65s ===================================
```

```python
@pytest.mark.parametrize(
    "get_schedule",
    [
        get_schedule_from_calculator_api,
        get_schedule_from_loan_admin,
        get_schedule_sent_on_loan_create,
    ],
)
def test_no_periods(get_schedule):
    schedule = get_schedule(principal=0, rate=0, number_of_periods=0)
    assert schedule == []
```

```
================================= test session starts ==================================
platform darwin -- Python 3.11.1, pytest-7.4.2, pluggy-1.3.0
django: settings: empathetic_mortgage.settings (from ini)
rootdir: /Users/marcgibbons/projects/empathetic-testing
configfile: pyproject.toml
plugins: cov-4.1.0, django-4.5.2
collected 6 items

tests/test_calculators.py ......                                          [100%]

---------- coverage: platform darwin, python 3.11.1-final-0 ----------
Name                     Stmts   Miss  Cover
--------------------------------------------
loans/calculators.py        11      0   100%
--------------------------------------------
TOTAL                       11      0   100%


================================== 6 passed in 0.75s ===================================
```

Why test this way?

Why focus on refactorability?

flexibility
speed
productivity
profit?

act of kindness

create an **enjoyable** work experience

You *can* make
a difference

Focus on outcomes & uses

rather than defend implementations

Seek alternatives to patch / mock

Try writing
tests first

The takeaway

Does ***this*** test enable, or prevent future **changes**?

git blame

Complete test coverage

marcgibbons committed on Dec 19, 2016

https://github.com/marcgibbons/empathetic-testing

marc@marcgibbons.com

@marcgibbons@mastodon.social