Asignatura	Datos del alumno	Fecha
Entornos Integración y Entrega Continua	Apellidos: Ingles Martinez	18/05/2025
	Nombre: Alejandro	

Automatización de pruebas con Python (actividad individual)

Código fuente

- ### app/calc.py

Añado las funciones de cálculo de raíces cuadradas y logartimos en base 10.

calc.py

app > calc.py

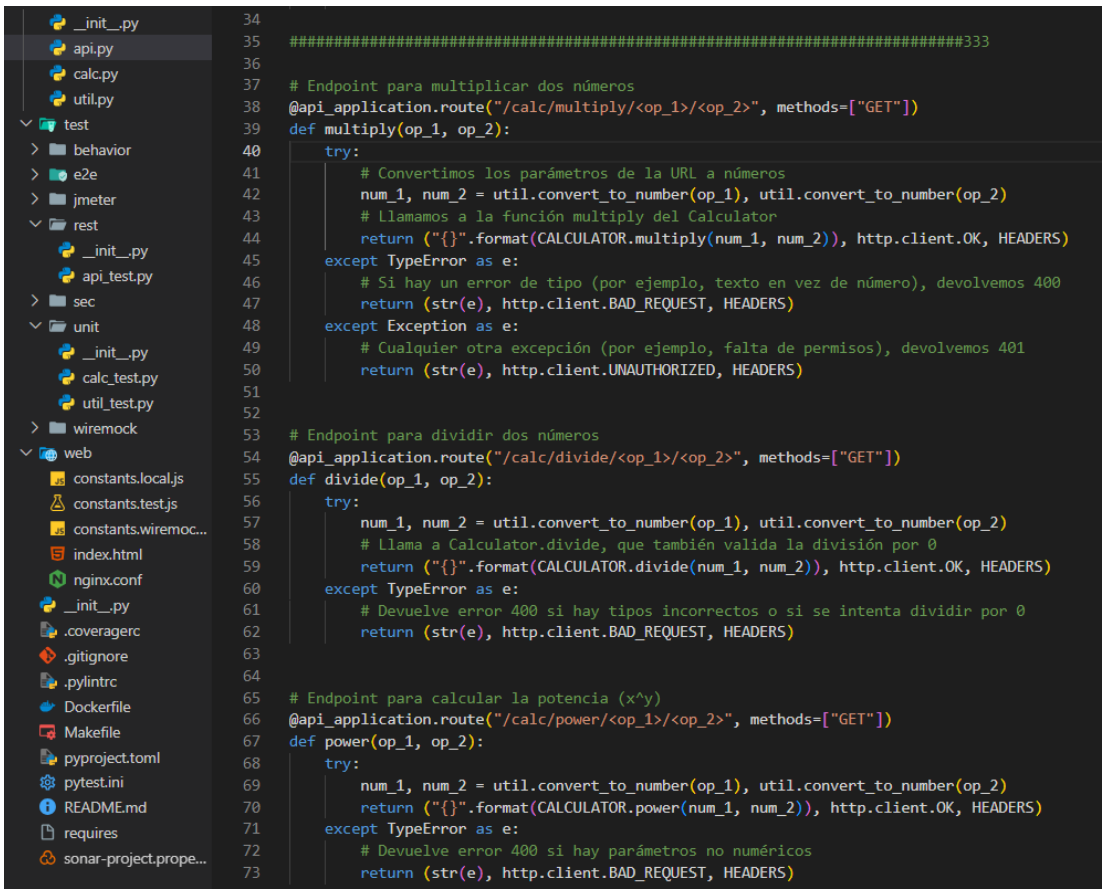
```
8   class Calculator:
35      def check_types(self, x, y):
36          if not isinstance(x, (int, float)) or not isinstance(y, (int, float)):
37              raise TypeError("Parameters must be numbers")
38
39  ##########################################
40      def sqrt(self, x):
41          self._check_single_number(x)
42          if x < 0:
43              raise ValueError("Cannot compute square root of a negative number")
44          return math.sqrt(x)
45
46      def log10(self, x):
47          self._check_single_number(x)
48          if x <= 0:
49              raise ValueError("Logarithm base 10 undefined for zero or negative numbers")
50          return math.log10(x)
51
```

- ### app/api.py

No tiene lógica matemática directamente. Solo recibe las peticiones, llama a los métodos de Calculator y devuelve la respuesta.

Añado 6 nuevos endpoints para la multiplicacion, la división, la potencia, la raíz cuadrada y el logaritmo en base 10.

Asignatura	Datos del alumno	Fecha
Entornos Integración y Entrega Continua	Apellidos: Ingles Martinez	18/05/2025
	Nombre: Alejandro	




```python
# Endpoint para raíz cuadrada
@api_application.route("/calc/sqrt/<op>", methods=["GET"])
def sqrt(op):
    try:
        num = util.convert_to_number(op)
        return ("{}".format(CALCULATOR.sqrt(num)), http.client.OK, HEADERS)
    except (TypeError, ValueError) as e:
        return (str(e), http.client.BAD_REQUEST, HEADERS)

# Endpoint para logaritmo en base 10
@api_application.route("/calc/log10/<op>", methods=["GET"])
def log10(op):
    try:
        num = util.convert_to_number(op)
        return ("{}".format(CALCULATOR.log10(num)), http.client.OK, HEADERS)
    except (TypeError, ValueError) as e:
        return (str(e), http.client.BAD_REQUEST, HEADERS)
```

Asignatura	Datos del alumno	Fecha
Entornos Integración y Entrega Continua	Apellidos: Ingles Martinez	18/05/2025
	Nombre: Alejandro	

- **Pruebas unitarias (test/unit/calc_test.py):**

Creo pruebas para **todas las funciones de la clase Calculator**. Casos de **error** también (tipos incorrectos, división por 0, permisos, etc.)

```
60  ##############################################################3
61      # Resta: x - y
62      def test_substract_method_returns_correct_result(self):
63          self.assertEqual(0, self.calc.substract(2, 2))     # 2 - 2 = 0
64          self.assertEqual(4, self.calc.substract(2, -2))    # 2 - (-2) = 4
65          self.assertEqual(-4, self.calc.substract(-2, 2))   # -2 - 2 = -4
66
67      # Resta: CASOS DE FALLOS por parámetros no numéricos
68      def test_substract_method_fails_with_nan_parameter(self):
69          self.assertRaises(TypeError, self.calc.substract, "2", 2)   # string - número
70          self.assertRaises(TypeError, self.calc.substract, 2, None)  # número - None
71
72      # Potencia: x ** y
73      def test_power_method_returns_correct_result(self):
74          self.assertEqual(8, self.calc.power(2, 3))        # 2^3 = 8
75          self.assertEqual(1, self.calc.power(10, 0))       # 10^0 = 1
76          self.assertEqual(0.25, self.calc.power(2, -2))    # 2^-2 = 1/4 = 0.25
77
78      # Potencia: CASOS DE FALLOS por parámetros no numéricos
79      def test_power_method_fails_with_nan_parameter(self):
80          self.assertRaises(TypeError, self.calc.power, "2", 3)   # string como base
81          self.assertRaises(TypeError, self.calc.power, 2, "3")   # string como exponente
82
83      # Raíz cuadrada: sqrt(x)
84      def test_sqrt_method_returns_correct_result(self):
85          self.assertEqual(2, self.calc.sqrt(4))            # √4 = 2
86          self.assertEqual(0, self.calc.sqrt(0))            # √0 = 0
87          self.assertAlmostEqual(1.4142, self.calc.sqrt(2), places=4)  # √2 ≈ 1.4142
88
89      # Raíz cuadrada: CASO FALLO si x < 0
90      def test_sqrt_method_fails_with_negative_number(self):
91          self.assertRaises(ValueError, self.calc.sqrt, -1)     # √(-1) → error
92
93      # Raíz cuadrada: CASO FALLO por tipo no numérico
94      def test_sqrt_method_fails_with_nan_parameter(self):
95          self.assertRaises(TypeError, self.calc.sqrt, "4")     # string
96          self.assertRaises(TypeError, self.calc.sqrt, None)    # None
97          self.assertRaises(TypeError, self.calc.sqrt, object())# objeto
```

```python
    # Logaritmo base 10: log10(x)
    def test_log10_method_returns_correct_result(self):
        self.assertEqual(1, self.calc.log10(10))              # log10(10) = 1
        self.assertEqual(0, self.calc.log10(1))               # log10(1) = 0
        self.assertAlmostEqual(2, self.calc.log10(100), places=4)  # log10(100) = 2

    # Logaritmo base 10: CASO FALLO si x <= 0
    def test_log10_method_fails_with_zero_or_negative_number(self):
        self.assertRaises(ValueError, self.calc.log10, 0)     # log10(0) → indefinido
        self.assertRaises(ValueError, self.calc.log10, -10)   # log10(-10) → error

    # Logaritmo base 10: CASO FALLO por tipo no numérico
    def test_log10_method_fails_with_nan_parameter(self):
        self.assertRaises(TypeError, self.calc.log10, "10")    # string
        self.assertRaises(TypeError, self.calc.log10, None)    # None
        self.assertRaises(TypeError, self.calc.log10, object())# objeto
```

- **Pruebas de API (test/rest/api_test.py):**

Necesito probar los endpoints:

- /calc/add/<x>/<y>
- /calc/substract/<x>/<y>
- /calc/multiply/<x>/<y>
- /calc/divide/<x>/<y>
- /calc/power/<x>/<y>
- /calc/sqrt/<op>
- /calc/log10/<op>
- Casos de éxito (200 OK)
- Casos de error (400 Bad Request, 401 Unauthorized...)

```python
    # Prueba API: resta 10 - 3 = 7
    def test_api_substract(self):
        url = f"{BASE_URL}/calc/substract/10/3"
        response = urlopen(url, timeout=DEFAULT_TIMEOUT)
        self.assertEqual(response.status, http.client.OK)

    # Prueba API: multiplicación 2 * 4 = 8 (requiere permisos)
    def test_api_multiply(self):
        url = f"{BASE_URL}/calc/multiply/2/4"
        response = urlopen(url, timeout=DEFAULT_TIMEOUT)
        self.assertEqual(response.status, http.client.OK)

    # Prueba API: división 8 / 2 = 4
    def test_api_divide(self):
        url = f"{BASE_URL}/calc/divide/8/2"
        response = urlopen(url, timeout=DEFAULT_TIMEOUT)
        self.assertEqual(response.status, http.client.OK)

    # Prueba API: potencia 2^3 = 8
    def test_api_power(self):
        url = f"{BASE_URL}/calc/power/2/3"
        response = urlopen(url, timeout=DEFAULT_TIMEOUT)
        self.assertEqual(response.status, http.client.OK)

    # Prueba FALLO API: división por 0 debe dar error 400
    def test_api_divide_by_zero_returns_400(self):
        try:
            urlopen(f"{BASE_URL}/calc/divide/1/0", timeout=DEFAULT_TIMEOUT)
            self.fail("Expected exception due to division by zero")  # Debe fallar
        except Exception as e:
            self.assertIn("HTTP Error 400", str(e))  # Confirma que devuelve 400

    # Prueba FALLO API: suma con operando inválido ("foo") debe dar error 400
    def test_api_invalid_operand_returns_400(self):
        try:
            urlopen(f"{BASE_URL}/calc/add/1/foo", timeout=DEFAULT_TIMEOUT)
            self.fail("Expected exception due to invalid operand")  # "foo" no es número
        except Exception as e:
            self.assertIn("HTTP Error 400", str(e))
```

Asignatura	Datos del alumno	Fecha
Entornos Integración y Entrega Continua	Apellidos: Ingles Martinez	18/05/2025
	Nombre: Alejandro	

```python
    # Prueba API: raíz cuadrada de 9 = 3
    def test_api_sqrt(self):
        url = f"{BASE_URL}/calc/sqrt/9"
        response = urlopen(url, timeout=DEFAULT_TIMEOUT)
        self.assertEqual(response.status, http.client.OK)

    # Prueba FALLO API: raíz cuadrada de número negativo (-1) → error 400
    def test_api_sqrt_negative_returns_400(self):
        try:
            urlopen(f"{BASE_URL}/calc/sqrt/-1", timeout=DEFAULT_TIMEOUT)
            self.fail("Expected exception due to sqrt of negative number")
        except Exception as e:
            self.assertIn("HTTP Error 400", str(e))

    # Prueba API: logaritmo base 10 de 1000 = 3
    def test_api_log10(self):
        url = f"{BASE_URL}/calc/log10/1000"
        response = urlopen(url, timeout=DEFAULT_TIMEOUT)
        self.assertEqual(response.status, http.client.OK)

    # Prueba FALLO API: logaritmo base 10 de 0 → error 400
    def test_api_log10_zero_returns_400(self):
        try:
            urlopen(f"{BASE_URL}/calc/log10/0", timeout=DEFAULT_TIMEOUT)
            self.fail("Expected exception due to log10(0)")
        except Exception as e:
            self.assertIn("HTTP Error 400", str(e))

    # Prueba FALLO API: logaritmo base 10 de -5 → error 400
    def test_api_log10_negative_returns_400(self):
        try:
            urlopen(f"{BASE_URL}/calc/log10/-5", timeout=DEFAULT_TIMEOUT)
            self.fail("Expected exception due to log10 of negative number")
        except Exception as e:
            self.assertIn("HTTP Error 400", str(e))

```

Asignatura	Datos del alumno	Fecha
Entornos Integración y Entrega Continua	Apellidos: Ingles Martinez	18/05/2025
	Nombre: Alejandro	

Proceso de despliegue y ejecución

Compilo y creo la imagen de docker calculator-app

```
aleingmar@DESKTOP-2IE63G4:~/MASTER/CICD/Act2-pruebasPython$ ls
Dockerfile  Makefile  README.md  __init__.py  app  pyproject.toml  pytest.ini  requires  sonar-project.properties  test  web
aleingmar@DESKTOP-2IE63G4:~/MASTER/CICD/Act2-pruebasPython$ make build
docker build -t calculator-app .
failed to fetch metadata: fork/exec /usr/local/lib/docker/cli-plugins/docker-buildx: no such file or directory

DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  127.5kB
Step 1/5 : FROM python:3.6-slim
3.6-slim: Pulling from library/python
a2abf6c4d29d: Pull complete
625294dad115: Pull complete
838e3a5a04bf: Pull complete
e93b4e59b689: Pull complete
c4401b8c7f9e: Pull complete
Digest: sha256:2cfebc27956e6a55f78606864d91fe527696f9e32a724e6f9702b5f9602d0474
Status: Downloaded newer image for python:3.6-slim
 ---> c1e40b69532f
```

```
aleingmar@DESKTOP-2IE63G4:~/MASTER/CICD/Act2-pruebasPython$ docker images
REPOSITORY                          TAG       IMAGE ID       CREATED              SIZE
calculator-app                      latest    c9a63a218b9a   About a minute ago   152MB
act2-dockercompose-capa1-nginx      latest    6371a94c7349   8 days ago           192MB
act2-dockercompose-capa2-express    latest    a330141c78c7   8 days ago           1.12GB
act2-dockercompose-capa3-mongo      latest    c980c1f073df   2 weeks ago          817MB
python                              3.6-slim  c1e40b69532f   3 years ago          119MB
```

Ejecuto las pruebas unitarias

Esto ejecutará pytest sobre los test marcados con @pytest.mark.unit y generará:

- Un informe XML (results/unit_result.xml)
- Un informe HTML (results/unit_result.html)
- Un informe de cobertura en results/coverage/

ERROR: Debido a que: Se me había olvidado programar la función check_single_number, se me había olvidado importar el modulo math y en una prueba de división entre 0, devolvía un TypeError en vez de un ValueError, que es lo que tiene más sentido y es lo que esperaba la prueba recibir.

Asignatura	Datos del alumno	Fecha
Entornos Integración y Entrega Continua	Apellidos: Ingles Martinez	18/05/2025
	Nombre: Alejandro	

```
aleingmar@DESKTOP-2IE63G4:~/MASTER/CICD/Act2-pruebasPython$ make test-unit
docker run --rm --volume `pwd`:/opt/calc --env PYTHONPATH=/opt/calc -w /opt/calc calculator-app:la
results/coverage.xml --cov-report=html:results/coverage --junit-xml=results/unit_result.xml -m uni
============================= test session starts ==============================
platform linux -- Python 3.6.15, pytest-5.4.3, py-1.11.0, pluggy-0.13.1
rootdir: /opt/calc, inifile: pytest.ini
plugins: cov-2.10.0
collected 31 items / 13 deselected / 18 selected

test/unit/calc_test.py ..F..FFF...FFF..                                  [ 88%]
test/unit/util_test.py ..                                                [100%]

=================================== FAILURES ===================================
_________ TestCalculate.test_divide_method_fails_with_division_by_zero _________

self = <unit.calc_test.TestCalculate testMethod=test_divide_method_fails_with_division_by_zero>

    def test_divide_method_fails_with_division_by_zero(self):
>       self.assertRaises(ValueError, self.calc.divide, 2, 0)       # división por cero

test/unit/calc_test.py:47:
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

    def divide(self, x, y):
        self.check_types(x, y)
        if y == 0:
```

```
=========================== short test summary info ============================
FAILED test/unit/calc_test.py::TestCalculate::test_divide_method_fails_with_division_by_zero
FAILED test/unit/calc_test.py::TestCalculate::test_log10_method_fails_with_nan_parameter
FAILED test/unit/calc_test.py::TestCalculate::test_log10_method_fails_with_zero_or_negative_number
FAILED test/unit/calc_test.py::TestCalculate::test_log10_method_returns_correct_result
FAILED test/unit/calc_test.py::TestCalculate::test_sqrt_method_fails_with_nan_parameter
FAILED test/unit/calc_test.py::TestCalculate::test_sqrt_method_fails_with_negative_number
FAILED test/unit/calc_test.py::TestCalculate::test_sqrt_method_returns_correct_result
================= 7 failed, 11 passed, 13 deselected in 0.43s ==================
```

Programo la función

```
46      def log10(self, x):
47          self._check_single_number(x)
48          if x <= 0:
49              raise ValueError("Logarithm base 10 undefined for zero or negative numbers")
50          return math.log10(x)
51
52      # Verifica que un único parámetro sea numérico
53      def _check_single_number(self, x):
54          if not isinstance(x, (int, float)):
55              raise TypeError("Parameter must be a number")
```

Importo el módulo

```
1   import app
2   import math
3
```

Cambio el tipo de fallo de TypeError a Value error como espero en la prueba para que tenga más sentido

Asignatura	Datos del alumno	Fecha
Entornos Integración y Entrega Continua	Apellidos: Ingles Martinez	18/05/2025
	Nombre: Alejandro	

```python
def divide(self, x, y):
    self.check_types(x, y)
    if y == 0:
        raise ValueError("Division by zero is not possible")

    return x / y
```

```python
# División: CASOS DE FALLOS por división entre cero
def test_divide_method_fails_with_division_by_zero(self):
    self.assertRaises(ValueError, self.calc.divide, 2, 0)      # división por cero
    self.assertRaises(ValueError, self.calc.divide, 2, -0)     # -0 es 0 → división por cero
    self.assertRaises(ValueError, self.calc.divide, 0, 0)      # 0 / 0 → indefinido
    self.assertRaises(TypeError, self.calc.divide, "0", 0)     # string + número → error de tipo
```

ARREGLADO

```
aleingmar@DESKTOP-2IE63G4:~/MASTER/CICD/Act2-pruebasPython$ make test-unit
docker run --rm --volume `pwd`:/opt/calc --env PYTHONPATH=/opt/calc -w /opt/calc calculator-app:latest pytest --cov --cov-report=xml:
results/coverage.xml --cov-report=html:results/coverage --junit-xml=results/unit_result.xml -m unit || true
============================= test session starts ==============================
platform linux -- Python 3.6.15, pytest-5.4.3, py-1.11.0, pluggy-0.13.1
rootdir: /opt/calc, inifile: pytest.ini
plugins: cov-2.10.0
collected 31 items / 13 deselected / 18 selected

test/unit/calc_test.py ..............                                    [ 88%]
test/unit/util_test.py ..                                                [100%]

------------ generated xml file: /opt/calc/results/unit_result.xml -------------

---------- coverage: platform linux, python 3.6.15-final-0 -----------
Coverage HTML written to dir results/coverage
Coverage XML written to file results/coverage.xml

====================== 18 passed, 13 deselected in 0.29s =======================
docker run --rm --volume `pwd`:/opt/calc --env PYTHONPATH=/opt/calc -w /opt/calc calculator-app:latest junit2html results/unit_result
.xml results/unit_result.html
```

```
aleingmar@DESKTOP-2IE63G4:~/MASTER/CICD/Act2-pruebasPython$ ls results/
coverage  coverage.xml  unit_result.html  unit_result.xml
```

Asignatura	Datos del alumno	Fecha
Entornos Integración y Entrega Continua	Apellidos: Ingles Martinez	18/05/2025
	Nombre: Alejandro	

Ejecuto las pruebas de API

- Lanza el servidor Flask en un contenedor
- Ejecuta los tests marcados con @pytest.mark.api
- Genera el informe XML y lo convierte a HTML

ERRORES: Debido a que: Por el cambio del tipo de error que se devuelve al dividir entre 0.

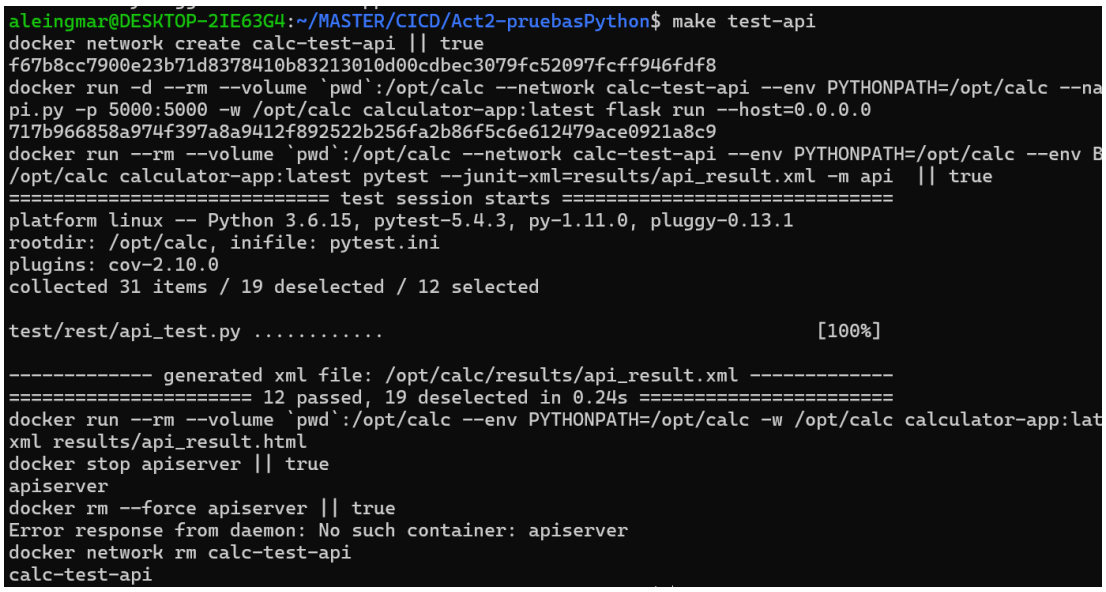
```
    def http_error_default(self, req, fp, code, msg, hdrs):
>       raise HTTPError(req.full_url, code, msg, hdrs, fp)
E       urllib.error.HTTPError: HTTP Error 500: INTERNAL SERVER ERROR

/usr/local/lib/python3.6/urllib/request.py:650: HTTPError

During handling of the above exception, another exception occurred:

self = <rest.api_test.TestApi testMethod=test_api_divide_by_zero_returns_400>

    def test_api_divide_by_zero_returns_400(self):
        try:
            urlopen(f"{BASE_URL}/calc/divide/1/0", timeout=DEFAULT_TIMEOUT)
            self.fail("Expected exception due to division by zero")  # Debe fallar
        except Exception as e:
>           self.assertIn("HTTP Error 400", str(e))  # Confirma que devuelve 400
E           AssertionError: 'HTTP Error 400' not found in 'HTTP Error 500: INTERNAL SERVER ERROR'

test/rest/api_test.py:55: AssertionError
------------ generated xml file: /opt/calc/results/api_result.xml -------------
========================== short test summary info ==========================
FAILED test/rest/api_test.py::TestApi::test_api_divide_by_zero_returns_400 - ...
================ 1 failed, 11 passed, 19 deselected in 0.51s =================
docker run --rm --volume `pwd`:/opt/calc --env PYTHONPATH=/opt/calc -w /opt/calc calculator-app:latest junit2html
xml results/api_result.html
docker stop apiserver || true
apiserver
docker rm --force apiserver || true
Error response from daemon: No such container: apiserver
docker network rm calc-test-api
calc-test-api
```

Tenía que incluir en el código del endpoint ambos códigos, TypeError y ValueError (el nuevo que se devuelve al dividir /0)

```
53  # Endpoint para dividir dos números
54  @api_application.route("/calc/divide/<op_1>/<op_2>", methods=["GET"])
55  def divide(op_1, op_2):
56      try:
57          num_1, num_2 = util.convert_to_number(op_1), util.convert_to_number(op_2)
58          # Llama a Calculator.divide, que también valida la división por 0
59          return ("{}".format(CALCULATOR.divide(num_1, num_2)), http.client.OK, HEADERS)
60      except (TypeError, ValueError) as e:
61          # Devuelve error 400 si hay tipos incorrectos o si se intenta dividir por 0
62          return (str(e), http.client.BAD_REQUEST, HEADERS)
```

ARREGLADO

Asignatura	Datos del alumno	Fecha
Entornos Integración y Entrega Continua	Apellidos: Ingles Martinez	18/05/2025
	Nombre: Alejandro	

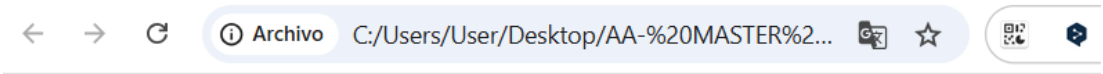


Resultados

Una vez ejecutados los tests, se generan los informes en la carpeta results/:

- results/unit_result.html: resumen visual de las pruebas unitarias
- results/api_result.html: resumen visual de las pruebas de API

Asignatura	Datos del alumno	Fecha
Entornos Integración y Entrega Continua	Apellidos: Ingles Martinez	18/05/2025
	Nombre: Alejandro	



Test Report : unit_result.xml

test.unit.calc_test.TestCalculate
- test_add_method_fails_with_nan_parameter
- test_add_method_returns_correct_result
- test_divide_method_fails_with_division_by_zero
- test_divide_method_fails_with_nan_parameter
- test_divide_method_returns_correct_result
- test_log10_method_fails_with_nan_parameter
- test_log10_method_fails_with_zero_or_negative_number
- test_log10_method_returns_correct_result
- test_multiply_method_returns_correct_result
- test_power_method_fails_with_nan_parameter
- test_power_method_returns_correct_result
- test_sqrt_method_fails_with_nan_parameter
- test_sqrt_method_fails_with_negative_number
- test_sqrt_method_returns_correct_result
- test_substract_method_fails_with_nan_parameter
- test_substract_method_returns_correct_result

test.unit.util_test.TestUtil
- test_convert_to_number_correct_param
- test_convert_to_number_invalid_type

Test Suite: pytest

Results

Duration	0.295 sec
Tests	18
Failures	0

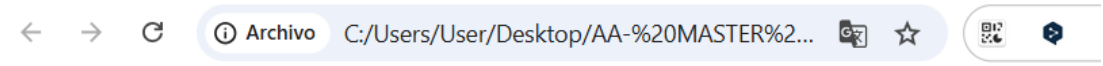
Tests

test.unit.calc_test.TestCalculate

Test case:	**test_add_method_fails_with_nan_parameter**
Outcome:	Passed
Duration:	0.001 sec

Test case:	**test_add_method_returns_correct_result**
Outcome:	Passed
Duration:	0.001 sec

Asignatura	Datos del alumno	Fecha
Entornos Integración y Entrega Continua	Apellidos: Ingles Martinez	18/05/2025
	Nombre: Alejandro	



Test Report : api_result.xml

test.rest.api_test.TestApi

- test_api_add
- test_api_divide
- test_api_divide_by_zero_returns_400
- test_api_invalid_operand_returns_400
- test_api_log10
- test_api_log10_negative_returns_400
- test_api_log10_zero_returns_400
- test_api_multiply
- test_api_power
- test_api_sqrt
- test_api_sqrt_negative_returns_400
- test_api_substract

Test Suite: pytest

Results

Duration	0.245 sec
Tests	12
Failures	0

Tests

test.rest.api_test.TestApi

Test case:	**test_api_add**
Outcome:	Passed
Duration:	0.006 sec

Test case:	**test_api_divide**
Outcome:	Passed
Duration:	0.005 sec