```
from unittest.mock import Mock, patch

import pytest

from swarm_models.vilt import Image, Vilt, requests


# Fixture for Vilt instance
@pytest.fixture
def vilt_instance():
    return Vilt()


# 1. Test Initialization
def test_vilt_initialization(vilt_instance):
    assert isinstance(vilt_instance, Vilt)
    assert vilt_instance.processor is not None
    assert vilt_instance.model is not None


# 2. Test Model Predictions
@patch.object(requests, "get")
@patch.object(Image, "open")
def test_vilt_prediction(
    mock_image_open, mock_requests_get, vilt_instance
):
```

```
    mock_image = Mock()
    mock_image_open.return_value = mock_image
    mock_requests_get.return_value.raw = Mock()

    # It's a mock response, so no real answer expected
    with pytest.raises(
        Exception
    ):  # Ensure exception is more specific
        vilt_instance(
            "What is this image",

"https://images.unsplash.com/photo-1582538885592-e70a5d7ab3d3?ixlib=rb-4.0.3&ixid=M3wxMjA3
fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8fA%3D%3D&auto=format&fit=crop&w=1770&q=80",
        )


# 3. Test Exception Handling for network
@patch.object(
    requests,
    "get",
    side_effect=requests.RequestException("Network error"),
)
def test_vilt_network_exception(vilt_instance):
    with pytest.raises(requests.RequestException):
        vilt_instance(
            "What is this image",
```

```
"https://images.unsplash.com/photo-1582538885592-e70a5d7ab3d3?ixlib=rb-4.0.3&ixid=M3wxMjA3
fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8fA%3D%3D&auto=format&fit=crop&w=1770&q=80",
        )



# Parameterized test cases for different inputs
@pytest.mark.parametrize(
    "text,image_url",
    [
        ("What is this?", "http://example.com/image1.jpg"),
        ("Who is in the image?", "http://example.com/image2.jpg"),
        (
            "Where was this picture taken?",
            "http://example.com/image3.jpg",
        ),
        # ... Add more scenarios
    ],
)
def test_vilt_various_inputs(text, image_url, vilt_instance):
    with pytest.raises(
        Exception
    ):  # Again, ensure exception is more specific
        vilt_instance(text, image_url)
```

```
# Test with invalid or empty text
@pytest.mark.parametrize(
    "text,image_url",
    [
        (
            "",
"https://images.unsplash.com/photo-1582538885592-e70a5d7ab3d3?ixlib=rb-4.0.3&ixid=M3wxMjA3
fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8fA%3D%3D&auto=format&fit=crop&w=1770&q=80",
        ),
        (
            None,
"https://images.unsplash.com/photo-1582538885592-e70a5d7ab3d3?ixlib=rb-4.0.3&ixid=M3wxMjA3
fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8fA%3D%3D&auto=format&fit=crop&w=1770&q=80",
        ),
        (
            " ",
"https://images.unsplash.com/photo-1582538885592-e70a5d7ab3d3?ixlib=rb-4.0.3&ixid=M3wxMjA3
fDB8MHxwaG90by1wYWdlfHx8fGVufDB8fHx8fA%3D%3D&auto=format&fit=crop&w=1770&q=80",
        ),
        # ... Add more scenarios
    ],
)
def test_vilt_invalid_text(text, image_url, vilt_instance):
```

```
    with pytest.raises(ValueError):
        vilt_instance(text, image_url)



# Test with invalid or empty image_url
@pytest.mark.parametrize(
    "text,image_url",
    [
        ("What is this?", ""),
        ("Who is in the image?", None),
        ("Where was this picture taken?", " "),
    ],
)
def test_vilt_invalid_image_url(text, image_url, vilt_instance):
    with pytest.raises(ValueError):
        vilt_instance(text, image_url)
```