```
from unittest.mock import Mock, patch

from swarm_models.qwen import QwenVLMultiModal



def test_post_init():
    with patch(
        "swarms.models.qwen.AutoTokenizer.from_pretrained"
    ) as mock_tokenizer, patch(
        "swarms.models.qwen.AutoModelForCausalLM.from_pretrained"
    ) as mock_model:
        mock_tokenizer.return_value = Mock()
        mock_model.return_value = Mock()

        model = QwenVLMultiModal()
        mock_tokenizer.assert_called_once_with(
            model.model_name, trust_remote_code=True
        )
        mock_model.assert_called_once_with(
            model.model_name,
            device_map=model.device,
            trust_remote_code=True,
        )



def test_run():
```

```
    with patch(
        "swarms.models.qwen.AutoTokenizer.from_list_format"
    ) as mock_format, patch(
        "swarms.models.qwen.AutoTokenizer.__call__"
    ) as mock_call, patch(
        "swarms.models.qwen.AutoModelForCausalLM.generate"
    ) as mock_generate, patch(
        "swarms.models.qwen.AutoTokenizer.decode"
    ) as mock_decode:
        mock_format.return_value = Mock()
        mock_call.return_value = Mock()
        mock_generate.return_value = Mock()
        mock_decode.return_value = "response"

        model = QwenVLMultiModal()
        response = model.run(
            "Hello, how are you?", "https://example.com/image.jpg"
        )

        assert response == "response"


def test_chat():
    with patch(
        "swarms.models.qwen.AutoModelForCausalLM.chat"
    ) as mock_chat:
```

```
mock_chat.return_value = ("response", ["history"])


model = QwenVLMultiModal()
response, history = model.chat(
    "Hello, how are you?", "https://example.com/image.jpg"
)


assert response == "response"
assert history == ["history"]
```