```
import logging
from unittest.mock import Mock, patch

import pytest
import requests

from swarm_models.together import TogetherLLM



@pytest.fixture
def mock_api_key(monkeypatch):
    monkeypatch.setenv("TOGETHER_API_KEY", "mocked-api-key")



def test_init_defaults():
    model = TogetherLLM()
    assert model.together_api_key == "mocked-api-key"
    assert model.logging_enabled is False
    assert model.model_name == "mistralai/Mixtral-8x7B-Instruct-v0.1"
    assert model.max_workers == 10
    assert model.max_tokens == 300
    assert model.api_endpoint == "https://api.together.xyz"
    assert model.beautify is False
    assert model.streaming_enabled is False
    assert model.meta_prompt is False
    assert model.system_prompt is None
```

```
def test_init_custom_params(mock_api_key):
    model = TogetherLLM(
        together_api_key="custom-api-key",
        logging_enabled=True,
        model_name="custom-model",
        max_workers=5,
        max_tokens=500,
        api_endpoint="https://custom-api.together.xyz",
        beautify=True,
        streaming_enabled=True,
        meta_prompt="meta-prompt",
        system_prompt="system-prompt",
    )
    assert model.together_api_key == "custom-api-key"
    assert model.logging_enabled is True
    assert model.model_name == "custom-model"
    assert model.max_workers == 5
    assert model.max_tokens == 500
    assert model.api_endpoint == "https://custom-api.together.xyz"
    assert model.beautify is True
    assert model.streaming_enabled is True
    assert model.meta_prompt == "meta-prompt"
    assert model.system_prompt == "system-prompt"
```

```
@patch("swarms.models.together_model.requests.post")
def test_run_success(mock_post, mock_api_key):
    mock_response = Mock()
    mock_response.json.return_value = {
        "choices": [{"message": {"content": "Generated response"}}]
    }
    mock_post.return_value = mock_response

    model = TogetherLLM()
    task = "What is the color of the object?"
    response = model.run(task)

    assert response == "Generated response"


@patch("swarms.models.together_model.requests.post")
def test_run_failure(mock_post, mock_api_key):
    mock_post.side_effect = requests.exceptions.RequestException(
        "Request failed"
    )

    model = TogetherLLM()
    task = "What is the color of the object?"
    response = model.run(task)
```

```
    assert response is None


def test_run_with_logging_enabled(caplog, mock_api_key):
    model = TogetherLLM(logging_enabled=True)
    task = "What is the color of the object?"

    with caplog.at_level(logging.DEBUG):
        model.run(task)

    assert "Sending request to" in caplog.text


@pytest.mark.parametrize(
    "invalid_input", [None, 123, ["list", "of", "items"]]
)
def test_invalid_task_input(invalid_input, mock_api_key):
    model = TogetherLLM()
    response = model.run(invalid_input)

    assert response is None


@patch("swarms.models.together_model.requests.post")
def test_run_streaming_enabled(mock_post, mock_api_key):
    mock_response = Mock()
```

```
    mock_response.json.return_value = {
        "choices": [{"message": {"content": "Generated response"}}]
    }
    mock_post.return_value = mock_response

    model = TogetherLLM(streaming_enabled=True)
    task = "What is the color of the object?"
    response = model.run(task)

    assert response == "Generated response"


@patch("swarms.models.together_model.requests.post")
def test_run_empty_choices(mock_post, mock_api_key):
    mock_response = Mock()
    mock_response.json.return_value = {"choices": []}
    mock_post.return_value = mock_response

    model = TogetherLLM()
    task = "What is the color of the object?"
    response = model.run(task)

    assert response is None


@patch("swarms.models.together_model.requests.post")
```

```
def test_run_with_exception(mock_post, mock_api_key):
    mock_post.side_effect = Exception("Test exception")

    model = TogetherLLM()
    task = "What is the color of the object?"
    response = model.run(task)

    assert response is None


def test_init_logging_disabled(monkeypatch):
    monkeypatch.setenv("TOGETHER_API_KEY", "mocked-api-key")
    model = TogetherLLM()
    assert model.logging_enabled is False
    assert not model.system_prompt
```