# Отчёт по лабораторной работе №5
курс «Тестирование программного обеспечения Aquarius»

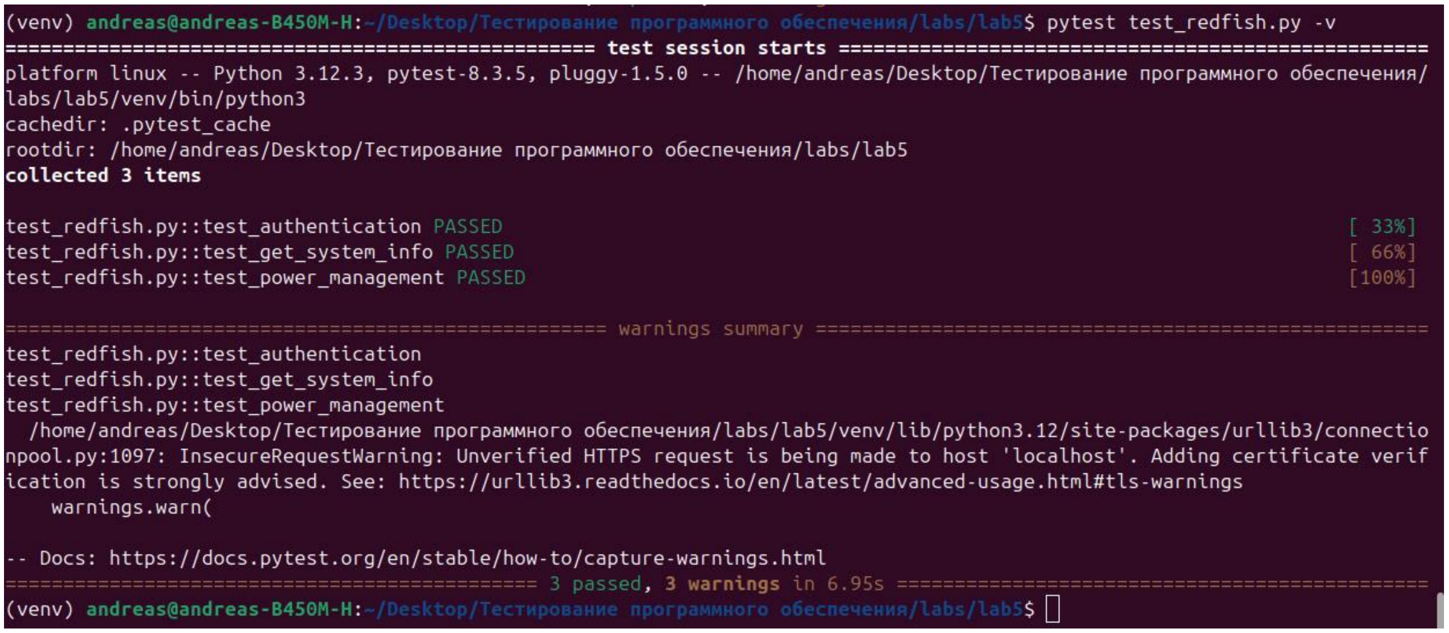
Выполнил студент группы ИП-211:
Вайберт Андреас Андреевич
Преподаватель курса:
Бочкарёв Борис Вячеславович

Новосибирск 2025

Цель работы

Освоить написание автоматизированных тестов для REST API на примере **Redfish API** в OpenBMC, используя **PyTest**. Научиться отправлять HTTP-запросы, проверять их корректность и анализировать ответы сервера.

Результат тестирования:



```
(venv) andreas@andreas-B450M-H:~/Desktop/Тестирование программного обеспечения/labs/lab5$ pytest test_redfish.py -v
================================================== test session starts ==================================================
platform linux -- Python 3.12.3, pytest-8.3.5, pluggy-1.5.0 -- /home/andreas/Desktop/Тестирование программного обеспечения/
labs/lab5/venv/bin/python3
cachedir: .pytest_cache
rootdir: /home/andreas/Desktop/Тестирование программного обеспечения/labs/lab5
collected 3 items

test_redfish.py::test_authentication PASSED                                                                        [ 33%]
test_redfish.py::test_get_system_info PASSED                                                                       [ 66%]
test_redfish.py::test_power_management PASSED                                                                      [100%]

=================================================== warnings summary ===================================================
test_redfish.py::test_authentication
test_redfish.py::test_get_system_info
test_redfish.py::test_power_management
  /home/andreas/Desktop/Тестирование программного обеспечения/labs/lab5/venv/lib/python3.12/site-packages/urllib3/connectio
npool.py:1097: InsecureRequestWarning: Unverified HTTPS request is being made to host 'localhost'. Adding certificate verif
ication is strongly advised. See: https://urllib3.readthedocs.io/en/latest/advanced-usage.html#tls-warnings
    warnings.warn(

-- Docs: https://docs.pytest.org/en/stable/how-to/capture-warnings.html
============================================= 3 passed, 3 warnings in 6.95s =============================================
(venv) andreas@andreas-B450M-H:~/Desktop/Тестирование программного обеспечения/labs/lab5$
```

test_redfish.py:

```python
import pytest

import requests
import logging

logging.basicConfig(level=logging.INFO)
logger = logging.getLogger(__name__)

BMC_URL = "https://localhost:2443"
AUTH_URL = f"{BMC_URL}/redfish/v1/SessionService/Sessions"
SYSTEM_URL = f"{BMC_URL}/redfish/v1/Systems/system"
RESET_URL = f"{BMC_URL}/redfish/v1/Systems/system/Actions/ComputerSystem.Reset"
CREDENTIALS = {"UserName": "root", "Password": "0penBmc"}

@pytest.fixture(scope="session")
def session_token():
    logger.info("Creating session for tests")
    try:
        response = requests.post(
            AUTH_URL,
            json=CREDENTIALS,
            headers={"Content-Type": "application/json"},
            timeout=5,
            verify=False
        )
        response.raise_for_status()
        token = response.headers.get("X-Auth-Token")
        if not token:
            logger.error("Response body: %s", response.text)
            pytest.fail("No X-Auth-Token found in response headers")
        yield token
    except requests.RequestException as e:
        logger.error(f"Failed to create session: {e}")
        logger.error(f"Response body: {response.text if 'response' in locals() else 'No
response'}")
        pytest.fail(f"Session creation failed: {e}")

def test_authentication():
    logger.info("Running authentication test")
    try:
        response = requests.post(
            AUTH_URL,
            json=CREDENTIALS,
            headers={"Content-Type": "application/json"},
            timeout=5,
            verify=False
        )
        assert response.status_code in (200, 201), f"Expected status 200 or 201, got
{response.status_code}. Response: {response.text}"
        assert "X-Auth-Token" in response.headers, "Session token missing in response
headers"
        logger.info("Authentication test passed")
    except requests.RequestException as e:
        logger.error(f"Authentication request failed: {e}")
        logger.error(f"Response body: {response.text if 'response' in locals() else 'No
response'}")
        pytest.fail(f"Authentication request failed: {e}")

def test_get_system_info(session_token):
    logger.info("Running system information test")
    headers = {"X-Auth-Token": session_token}
    try:
        response = requests.get(SYSTEM_URL, headers=headers, timeout=5, verify=False)
        assert response.status_code == 200, f"Expected status 200, got
{response.status_code}. Response: {response.text}"
        data = response.json()
        assert "Status" in data, "Status field missing in response"
        assert "PowerState" in data, "PowerState field missing in response"
        logger.info("System information test passed")
    except requests.RequestException as e:
        logger.error(f"System info request failed: {e}")
        logger.error(f"Response body: {response.text if 'response' in locals() else 'No
response'}")
        pytest.fail(f"System info request failed: {e}")
```

```
def test_power_management(session_token):
    logger.info("Running power management test")
    headers = {"X-Auth-Token": session_token}
    try:
        initial_response = requests.get(SYSTEM_URL, headers=headers, timeout=5, verify=False)
        assert initial_response.status_code == 200, f"Expected status 200, got
{initial_response.status_code}. Response: {initial_response.text}"

        reset_response = requests.post(
            RESET_URL,
            json={"ResetType": "On"},
            headers=headers,
            timeout=5,
            verify=False
        )
        assert reset_response.status_code in (200, 202, 204), f"Expected status 200/202/204,
got {reset_response.status_code}. Response: {reset_response.text}"

        import time
        time.sleep(5)

        updated_response = requests.get(SYSTEM_URL, headers=headers, timeout=5, verify=False)
        assert updated_response.status_code == 200, f"Expected status 200, got
{updated_response.status_code}. Response: {updated_response.text}"
        power_state = updated_response.json().get("PowerState")
        assert power_state in ("On", "PoweringOn"), f"PowerState should be On or PoweringOn
after reset. Got: {power_state}. Response: {updated_response.json()}"
        logger.info("Power management test passed")
    except requests.RequestException as e:
        logger.error(f"Power management request failed: {e}")
        logger.error(f"Response body: {response.text if 'response' in locals() else 'No
response'}")
        pytest.fail(f"Power management request failed: {e}")
```