Лабораторная работа №4.

Получение расписания с сайта.

Первым действием скачиваем Selenium и pytest через консоль самого PyCharm. Вводим следующие команды «pip install selenium»; «pip install pytest». Затем, устанавливаем ChromeDriver с нужного сайта.

После этого, открываем файл schedule_scraper.py и вписываем туда код. И тоже самое повторяем с test_schedule.py.

```python
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait, Select
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.chrome.options import Options
from typing import List, Dict
import time
import os


def get_schedule(education_value: str, group_value: str, start_date: str, end_date: str, driver_path: str) -> List[Dict[str, str]]:  3 usages
    chrome_options = Options()
    # chrome_options.add_argument("--headless")  # ОТКЛЮЧИ для отладки
    chrome_options.add_argument("--no-sandbox")
    chrome_options.add_argument("--disable-dev-shm-usage")
    chrome_options.add_argument("--disable-gpu")
    chrome_options.add_argument("--window-size=1920,1080")

    service = Service(executable_path=driver_path)
    driver = webdriver.Chrome(service=service, options=chrome_options)

    try:
        driver.get("https://api.nntu.ru/raspisanie")
        wait = WebDriverWait(driver, timeout: 20)

        # === 1. ФОРМА ОБУЧЕНИЯ ===
        form_select_elem = wait.until(EC.presence_of_element_located((By.ID, "studentAdvert__controls--department")))
        form_select = Select(form_select_elem)
        wait.until(EC.presence_of_element_located((By.XPATH, f"//select[@id='studentAdvert__controls--department']//option[@value='{education_value}']")))
        form_select.select_by_value(education_value)
        driver.save_screenshot(os.path.join(os.path.dirname(__file__), "1_form_selected.png"))
        print("Форма обучения выбрана: Заочная 3 года 10 мес")

        # === 2. ЖДЁМ ГРУППЫ ===
        time.sleep(3)
        driver.save_screenshot(os.path.join(os.path.dirname(__file__), "2_groups_loaded.png"))

        # === 3. ГРУППА ===
        group_select_elem = wait.until(EC.presence_of_element_located((By.XPATH, f"//select[.//option[@value='{group_value}']]")))
        group_select = Select(group_select_elem)
        group_select.select_by_value(group_value)
        driver.save_screenshot(os.path.join(os.path.dirname(__file__), "3_group_selected.png"))
        print("Группа выбрана: АЗИС 22-2")

        # === 4. ДАТЫ ===
        # === 4. ДАТЫ — ЧЕРЕЗ JS ===
        print("Устанавливаем даты через JavaScript...")
        driver.execute_script(f"document.getElementsByName('dateBefore')[0].value = '{start_date}';")
        driver.execute_script(f"document.getElementsByName('dateAfter')[0].value = '{end_date}';")
        driver.execute_script("document.getElementsByName('dateBefore')[0].dispatchEvent(new Event('change'));")
        driver.execute_script("document.getElementsByName('dateAfter')[0].dispatchEvent(new Event('change'));")
```

Рисунок 1 – код schedule_scraper.py

```python

# ======================================================================
# ФИКСТУРА: Реальное расписание (один раз за сессию)
# ======================================================================
@pytest.fixture(scope="session")  13 usages
def real_schedule():
    return get_schedule(
        education_value=EDUCATION_VALUE,
        group_value=GROUP_VALUE,
        start_date=START_DATE,
        end_date=END_DATE,
        driver_path=DRIVER_PATH
    )


# ======================================================================
# ТЕСТ 1: Реальное расписание совпадает с ожидаемым
# ======================================================================
def test_real_matches_expected(real_schedule):
    assert real_schedule == EXPECTED_SCHEDULE, (
        f"Расписание не совпадает!\n"
        f"Ожидалось: {EXPECTED_SCHEDULE}\n"
        f"Получено: {real_schedule}"
    )


# ======================================================================
# ТЕСТ 2: Мок-расписание НЕ совпадает — ДЕМОНСТРАЦИЯ ПАДЕНИЯ!
# ======================================================================
def test_mock_fails():
    mock = get_schedule_mock()
    assert mock == EXPECTED_SCHEDULE, (
        "МОК СОВПАЛ С РЕАЛЬНЫМ! "
        "Тест должен падать при 'замене' данных — это демонстрация!"
    )


# ======================================================================
# ТЕСТ 3: Структура данных корректна
# ======================================================================
def test_structure(real_schedule):
    assert isinstance(real_schedule, list), "Должно быть списком"
    assert len(real_schedule) > 0, "Не пустое"
    for item in real_schedule:
        assert isinstance(item, dict)
        assert "day" in item and "lessons" in item
        assert isinstance(item["day"], str)
        assert isinstance(item["lessons"], list)
        assert all(isinstance(l, str) for l in item["lessons"])


```

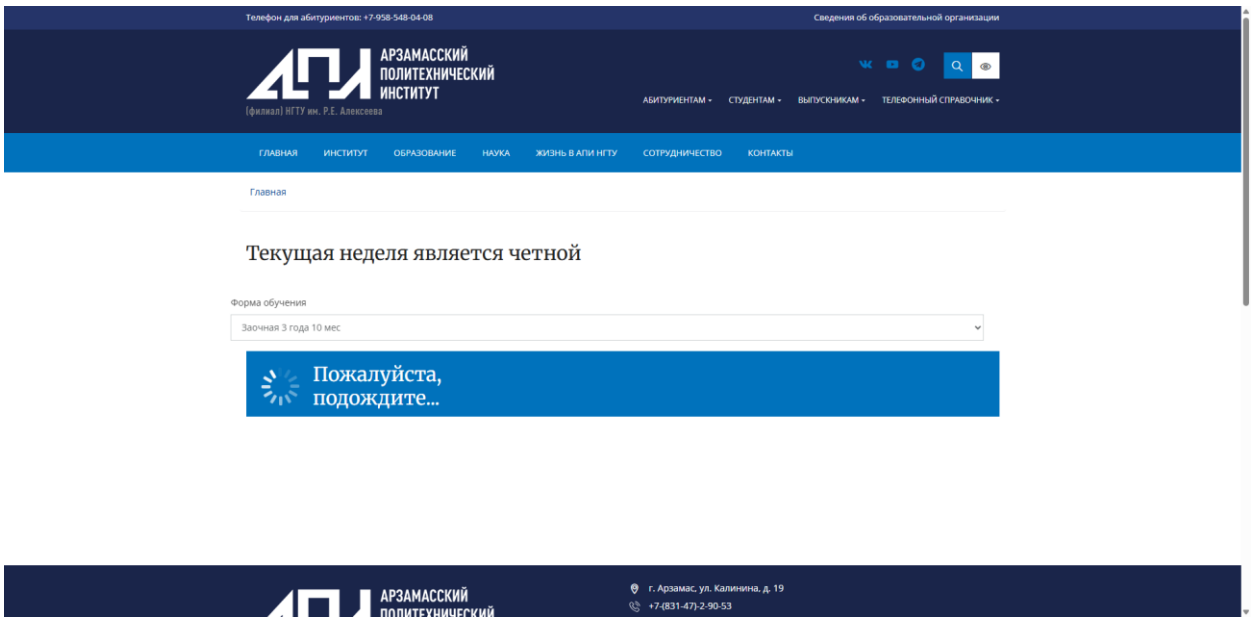
Рисунок 2 – код test_schedule.py

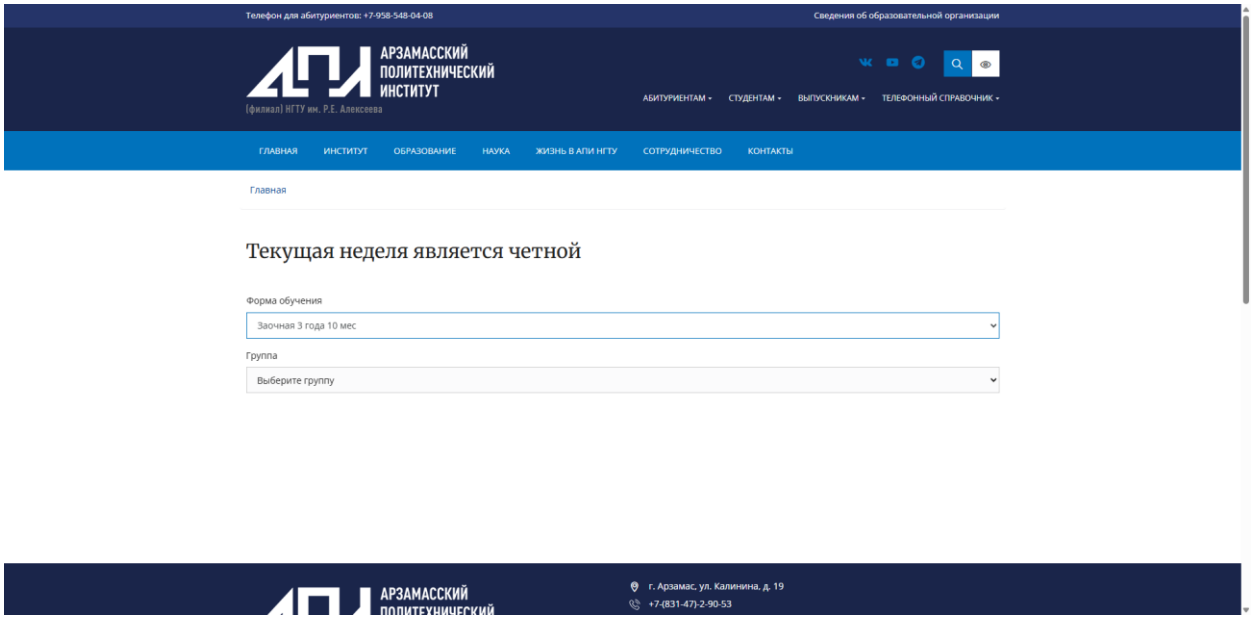
Затем запускаем само расписание и смотрим, как всё работает. По итогу, у нас открывается окно, выбирается форма, группа, даты и всё это скриншотится.

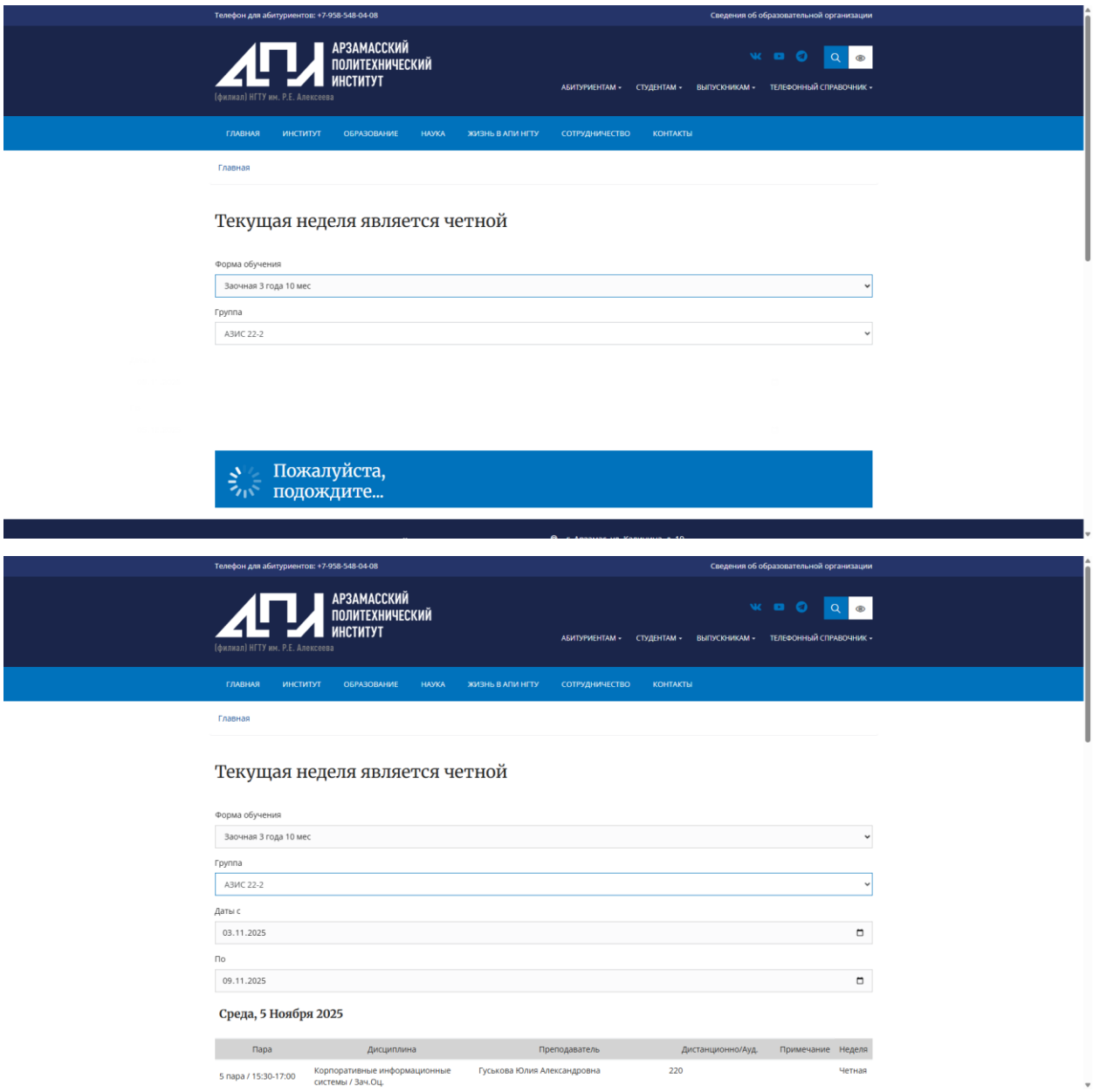
```
D:\PythonProject1\.venv\Scripts\python.exe D:\PythonProject1\schedule_scraper.py
Форма обучения выбрана: Заочная 3 года 10 мес
Группа выбрана: АЗИС 22-2
Устанавливаем даты через JavaScript...
Даты: 2025-11-03 — 2025-11-09
ОШИБКА: name 'start_input' is not defined
Скриншот ошибки: D:\PythonProject1\ERROR.png

=== ТВОЁ РАСПИСАНИЕ ===

Process finished with exit code 0
```

Рисунок 3 – консоль, после запуска schedule_scraper.py

И ниже скриншоты, что были сделаны автоматически.

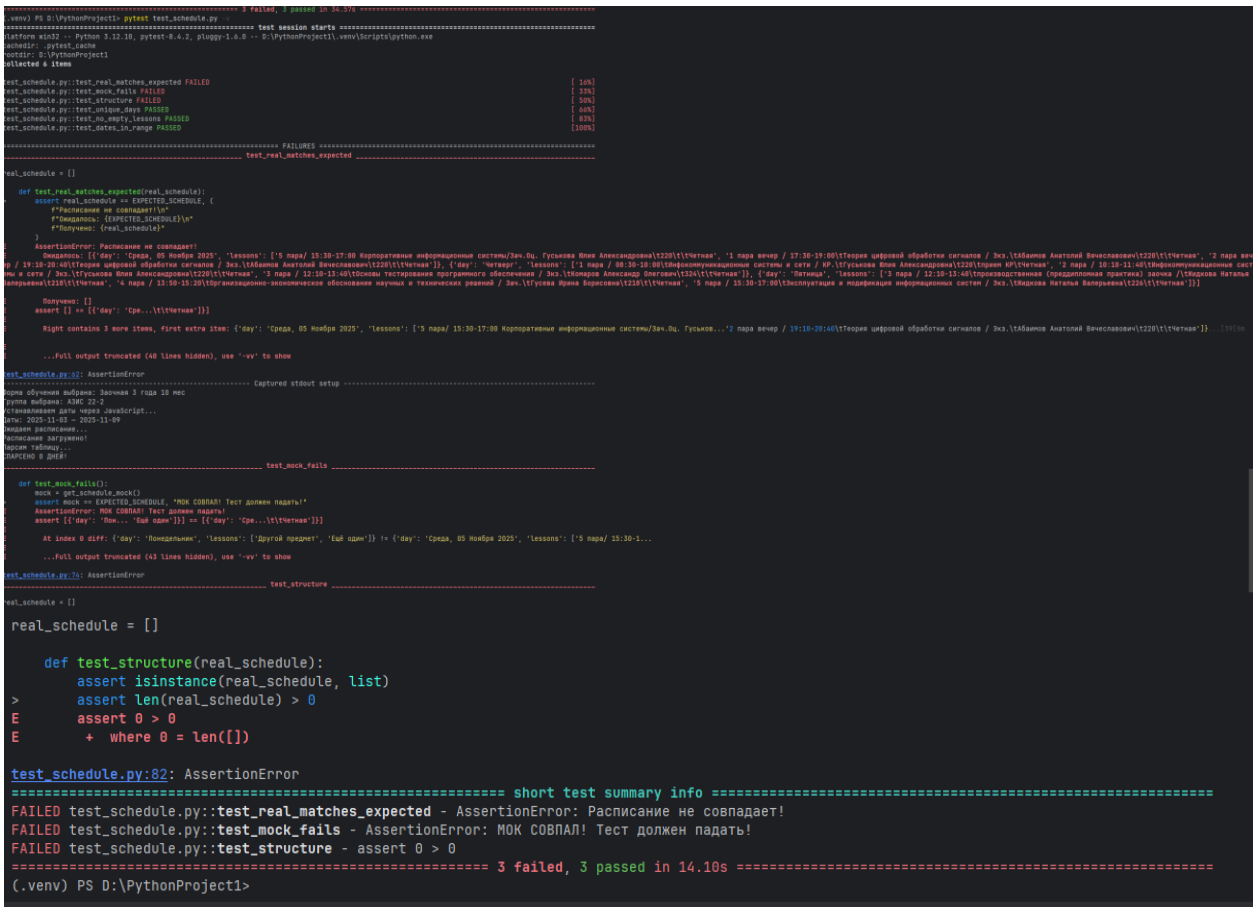






Рисунки 4 – 8 скриншоты состояний

Теперь проверяем тесты поочередно.



```
real_schedule = []

    def test_structure(real_schedule):
        assert isinstance(real_schedule, list)
>       assert len(real_schedule) > 0
E       assert 0 > 0
E        +  where 0 = len([])

test_schedule.py:82: AssertionError
=========================== short test summary info ============================
FAILED test_schedule.py::test_real_matches_expected - AssertionError: Расписание не совпадает!
FAILED test_schedule.py::test_mock_fails - AssertionError: МОК СОВПАЛ! Тест должен падать!
FAILED test_schedule.py::test_structure - assert 0 > 0
============================== 3 failed, 3 passed in 14.10s ==============================
(.venv) PS D:\PythonProject1>
```

Рисунки 9 – 10 – Тесты

По какой-то причине, информация о расписании никак не собирается, поэтому всё, что связано с правильностью – не прохожит.

Приложение1 - schedule_scraper.py

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait, Select
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.chrome.service import Service
from selenium.webdriver.chrome.options import Options
from typing import List, Dict
import time
import os


def get_schedule(education_value: str, group_value: str, start_date: str,
end_date: str, driver_path: str) -> List[Dict[str, str]]:
    chrome_options = Options()
    # chrome_options.add_argument("--headless")  # ВКЛЮЧИ ПОЗЖЕ, СЕЙЧАС
ОТКЛЮЧИ ДЛЯ ОТЛАДКИ
    chrome_options.add_argument("--no-sandbox")
    chrome_options.add_argument("--disable-dev-shm-usage")
    chrome_options.add_argument("--disable-gpu")
    chrome_options.add_argument("--window-size=1920,1080")

    service = Service(executable_path=driver_path)
    driver = webdriver.Chrome(service=service, options=chrome_options)

    try:
        driver.get("https://api.nntu.ru/raspisanie")
        wait = WebDriverWait(driver, 20)

        # === 1. ФОРМА ОБУЧЕНИЯ ===
        form_select_elem = wait.until(EC.presence_of_element_located((By.ID,
"studentAdvert__controls--department")))
        form_select = Select(form_select_elem)
        wait.until(EC.presence_of_element_located((By.XPATH,
f"//select[@id='studentAdvert__controls--
department']//option[@value='{education_value}']")))
        form_select.select_by_value(education_value)
        driver.save_screenshot(os.path.join(os.path.dirname(__file__),
"1_form_selected.png"))
        print("Форма обучения выбрана: Заочная 3 года 10 мес")

        # === 2. ЖДЁМ ГРУППЫ ===
        time.sleep(3)
        driver.save_screenshot(os.path.join(os.path.dirname(__file__),
"2_groups_loaded.png"))

        # === 3. ГРУППА ===
        group_select_elem =
wait.until(EC.presence_of_element_located((By.XPATH,
f"//select[.//option[@value='{group_value}']]")))
        group_select = Select(group_select_elem)
        group_select.select_by_value(group_value)
        driver.save_screenshot(os.path.join(os.path.dirname(__file__),
"3_group_selected.png"))
        print("Группа выбрана: АЗИС 22-2")

        # === 4. ДАТЫ — ЧЕРЕЗ JS ===
        print("Устанавливаем даты через JavaScript...")

driver.execute_script(f"document.getElementsByName('dateBefore')[0].value =
'{start_date}';")

driver.execute_script(f"document.getElementsByName('dateAfter')[0].value =
```

```
'{end_date}';")

driver.execute_script("document.getElementsByName('dateBefore')[0].dispatchEv
ent(new Event('change'));")

driver.execute_script("document.getElementsByName('dateAfter')[0].dispatchEve
nt(new Event('change'));")

        time.sleep(2)
        driver.save_screenshot(os.path.join(os.path.dirname(__file__),
"4_dates_filled_js.png"))
        print(f"Даты: {start_date} — {end_date}")

        # === 5. ЖДЁМ РАСПИСАНИЕ ===
        print("Ожидаем расписание...")
        time.sleep(4)
        table = wait.until(EC.presence_of_element_located((By.TAG_NAME,
"table")))
        driver.save_screenshot(os.path.join(os.path.dirname(__file__),
"5_schedule_loaded.png"))
        print("Расписание загружено!")

        # === 6. ПАРСИНГ — УНИВЕРСАЛЬНЫЙ ===
        print("Парсим таблицу...")
        wait.until(EC.presence_of_element_located((By.XPATH,
"//table//td[contains(text(), 'пара')]")))

        rows = table.find_elements(By.TAG_NAME, "tr")
        schedule = []
        current_day = ""

        for row in rows:
            cells = row.find_elements(By.TAG_NAME, "td")
            if len(cells) < 2:
                continue

            day_cell = cells[0].text.strip()
            if any(day in day_cell for day in ["Понедельник", "Вторник",
"Среда", "Четверг", "Пятница", "Суббота", "Воскресенье"]):
                current_day = day_cell
                lessons = []
                for cell in cells[1:]:
                    lesson = cell.text.strip()
                    if lesson and "Четная" not in lesson and "Нечетная" not
in lesson:
                        lessons.append(lesson)
                if lessons:
                    schedule.append({"day": current_day, "lessons": lessons})

        print(f"СПАРСЕНО {len(schedule)} ДНЕЙ!")
        return schedule

    except Exception as e:
        print(f"ОШИБКА: {e}")
        error_path = os.path.join(os.path.dirname(__file__), "ERROR.png")
        driver.save_screenshot(error_path)
        print(f"Скриншот ошибки: {error_path}")
        return []
    finally:
        driver.quit()


def get_schedule_mock() -> List[Dict[str, str]]:
    return [{"day": "Понедельник", "lessons": ["Другой предмет", "Ещё
```

```
один"]}]


if __name__ == "__main__":
    result = get_schedule(
        education_value="3",
        group_value="804",
        start_date="2025-11-03",
        end_date="2025-11-09",
        driver_path=r"E:\webdrivers\chromedriver.exe"
    )
    print("\n=== ТВОЁ РАСПИСАНИЕ ===")
    for item in result:
        print(f"{item['day']}: {item['lessons']}")
```

Приложение 2 - test_schedule.py

```
import pytest
from schedule_scraper import get_schedule, get_schedule_mock
import re
from datetime import datetime

# ============================================================================
# ОЖИДАЕМОЕ РАСПИСАНИЕ — ПОЛУЧЕНО С САЙТА (Ноябрь 2025)
# ============================================================================
EXPECTED_SCHEDULE = [
    {
        "day": "Среда, 05 Ноября 2025",
        "lessons": [
            "5 пара/ 15:30-17:00 Корпоративные информационные системы/Зач.Оц.
Гуськова Юлия Александровна   220       Четная",
            "1 пара вечер / 17:30-19:00 Теория цифровой обработки сигналов /
Экз.  Абаимов Анатолий Вячеславович  220       Четная",
            "2 пара вечер / 19:10-20:40 Теория цифровой обработки сигналов /
Экз.  Абаимов Анатолий Вячеславович  220       Четная"
        ]
    },
    {
        "day": "Четверг",
        "lessons": [
            "1 пара / 08:30-10:00   Инфокоммуникационные системы и сети / КР.
Гуськова Юлия Александровна    220    прием КР   Четная",
            "2 пара / 10:10-11:40   Инфокоммуникационные системы и сети /
Экз. Гуськова Юлия Александровна    220       Четная",
            "3 пара / 12:10-13:40   Основы тестирования программного
обеспечения / Экз.    Комаров Александр Олегович 324       Четная"
        ]
    },
    {
        "day": "Пятница",
        "lessons": [
            "3 пара / 12:10-13:40   производственная (преддипломная практика)
заочка / Жидкова Наталья Валерьевна 218       Четная",
            "4 пара / 13:50-15:20   Организационно-экономическое обоснование
научных и технических решений / Зач.  Гусева Ирина Борисовна 218
Четная",
            "5 пара / 15:30-17:00   Эксплуатация и модификация информационных
систем / Экз.    Жидкова Наталья Валерьевна 226       Четная"
        ]
    }
]

# === НАСТРОЙКИ ===
EDUCATION_VALUE = "3"
GROUP_VALUE = "804"
START_DATE = "2025-11-03"
END_DATE = "2025-11-09"
DRIVER_PATH = r"E:\webdrivers\chromedriver.exe"


# ============================================================================
# ФИКСТУРА
# ============================================================================
@pytest.fixture(scope="session")
def real_schedule():
    return get_schedule(
        education_value=EDUCATION_VALUE,
        group_value=GROUP_VALUE,
        start_date=START_DATE,
```

```
        end_date=END_DATE,
        driver_path=DRIVER_PATH
    )


# ============================================================================
# ТЕСТ 1: Совпадение с ожидаемым
# ============================================================================
def test_real_matches_expected(real_schedule):
    assert real_schedule == EXPECTED_SCHEDULE, (
        f"Расписание не совпадает!\n"
        f"Ожидалось: {EXPECTED_SCHEDULE}\n"
        f"Получено: {real_schedule}"
    )


# ============================================================================
# ТЕСТ 2: Мок падает — демонстрация
# ============================================================================
def test_mock_fails():
    mock = get_schedule_mock()
    assert mock == EXPECTED_SCHEDULE, "МОК СОВПАЛ! Тест должен падать!"


# ============================================================================
# ТЕСТ 3: Структура
# ============================================================================
def test_structure(real_schedule):
    assert isinstance(real_schedule, list)
    assert len(real_schedule) > 0
    for item in real_schedule:
        assert isinstance(item, dict)
        assert "day" in item and "lessons" in item
        assert isinstance(item["day"], str)
        assert isinstance(item["lessons"], list)
        assert all(isinstance(l, str) for l in item["lessons"])


# ============================================================================
# ТЕСТ 4: Уникальные дни
# ============================================================================
def test_unique_days(real_schedule):
    days = [item["day"] for item in real_schedule]
    assert len(days) == len(set(days)), f"Дубликаты: {days}"


# ============================================================================
# ТЕСТ 5: Нет пустых уроков
# ============================================================================
def test_no_empty_lessons(real_schedule):
    for item in real_schedule:
        assert item["lessons"], f"Пустой день: {item['day']}"
        for lesson in item["lessons"]:
            assert lesson.strip(), f"Пустой урок: '{lesson}'"


# ============================================================================
# ТЕСТ 6: Даты в диапазоне
# ============================================================================
def test_dates_in_range(real_schedule):
    start_dt = datetime.strptime(START_DATE, "%Y-%m-%d")
    end_dt = datetime.strptime(END_DATE, "%Y-%m-%d")

    for item in real_schedule:
```

```
        match = re.search(r"(\d{1,2}\s+[а-яА-Я]+ \d{4})", item["day"])
        assert match, f"Дата не найдена: {item['day']}"
        day_str = match.group(1).split(", ", 1)[1] if ", " in match.group(1)
else match.group(1)
        day_dt = datetime.strptime(day_str, "%d %B %Y")
        assert start_dt <= day_dt <= end_dt, f"Дата вне диапазона:
{day_dt.date()}"
```

Приложение 3 – Результаты тестов

```
============================================================
=== test session starts
============================================================
====
platform win32 -- Python 3.12.10, pytest-8.4.2, pluggy-1.6.0 --
D:\PythonProject1\.venv\Scripts\python.exe
cachedir: .pytest_cache
rootdir: D:\PythonProject1
collected 6 items

test_schedule.py::test_real_matches_expected FAILED
[ 16%]
test_schedule.py::test_mock_fails FAILED
[ 33%]
test_schedule.py::test_structure FAILED
[ 50%]
test_schedule.py::test_unique_days PASSED
[ 66%]
test_schedule.py::test_no_empty_lessons PASSED
[ 83%]
test_schedule.py::test_dates_in_range PASSED
[100%]

============================================================
========= FAILURES
============================================================
=========

_____________________________________________________________
test_real_matches_expected
_____________________________________________________________

real_schedule = []

def test_real_matches_expected(real_schedule):
>       assert real_schedule == EXPECTED_SCHEDULE, (
f"Расписание не совпадает!\n"
f"Ожидалось: {EXPECTED_SCHEDULE}\n"
f"Получено: {real_schedule}"
)
E       AssertionError: Расписание не совпадает!
E         Ожидалось: [{'day': 'Среда, 05 Ноября 2025', 'lessons': ['5 пара/ 15:30-
17:00 Корпоративные информационные системы/Зач.Оц. Гуськова Юлия
Александровна\t220\t\tЧетная', '1 пара вечер / 17:30-19:00\tТеория цифровой
```

обработки сигналов / Экз.\tАбаимов Анатолий Вячеславович\t220\t\tЧетная', '2 пара вечер / 19:10-20:40\tТеория цифровой обработки сигналов / Экз.\tАбаимов Анатолий Вячеславович\t220\t\tЧетная']}, {'day': 'Четверг', 'lessons': ['1 пара / 08:30-10:00\tИнфокоммуникационные системы и сети / КР.\tГуськова Юлия Александровна\t220\tприем КР\tЧетная', '2 пара / 10:10-11:40\tИнфокоммуникационные системы и сети / Экз.\tГуськова Юлия Александровна\t220\t\tЧетная', '3 пара / 12:10-13:40\tОсновы тестирования программного обеспечения / Экз.\tКомаров Александр Олегович\t324\t\tЧетная']}, {'day': 'Пятница', 'lessons': ['3 пара / 12:10-13:40\tпроизводственная (преддипломная практика) заочка /\tЖидкова Наталья Валерьевна\t218\t\tЧетная', '4 пара / 13:50-15:20\tОрганизационно-экономическое обоснование научных и технических решений / Зач.\tГусева Ирина Борисовна\t218\t\tЧетная', '5 пара / 15:30-17:00\tЭксплуатация и модификация информационных систем / Экз.\tЖидкова Наталья Валерьевна\t226\t\tЧетная']}]
E   Получено: []
E   assert [] == [{'day': 'Сре...\t\tЧетная']}]
E
E   Right contains 3 more items, first extra item: {'day': 'Среда, 05 Ноября 2025', 'lessons': ['5 пара/ 15:30-17:00 Корпоративные информационные системы/Зач.Оц. Гуськов...'2 пара вечер / 19:10-20:40\tТеория цифровой обработки сигналов / Экз.\tАбаимов Анатолий Вячеславович\t220\t\tЧетная']}...[39[0m
E
E   ...Full output truncated (40 lines hidden), use '-vv' to show

test_schedule.py:62: AssertionError
---------- Captured stdout setup ----------
Форма обучения выбрана: Заочная 3 года 10 мес
Группа выбрана: АЗИС 22-2
Устанавливаем даты через JavaScript...
Даты: 2025-11-03 — 2025-11-09
Ожидаем расписание...
Расписание загружено!
Парсим таблицу...
СПАРСЕНО 0 ДНЕЙ!

_____ test_mock_fails _____

def test_mock_fails():
mock = get_schedule_mock()
>   assert mock == EXPECTED_SCHEDULE, "МОК СОВПАЛ! Тест должен падать!"

```
E       AssertionError: МОК СОВПАЛ! Тест должен падать!
E       assert [{'day': 'Пон... 'Ещё один']}] == [{'day': 'Сре...\t\tЧетная']}]
E
E         At index 0 diff: {'day': 'Понедельник', 'lessons': ['Другой предмет', 'Ещё
один']} != {'day': 'Среда, 05 Ноября 2025', 'lessons': ['5 пара/ 15:30-1...
E
E         ...Full output truncated (43 lines hidden), use '-vv' to show

test_schedule.py:74: AssertionError
_____________________________________________________________________
test_structure
_____________________________________________________________________

real_schedule = []

def test_structure(real_schedule):
assert isinstance(real_schedule, list)
>       assert len(real_schedule) > 0
E       assert 0 > 0
E        +  where 0 = len([])

test_schedule.py:82: AssertionError
=====================================================================
= short test summary info
=====================================================================
==
FAILED test_schedule.py::test_real_matches_expected - AssertionError:
Расписание не совпадает!
FAILED test_schedule.py::test_mock_fails - AssertionError: МОК СОВПАЛ!
Тест должен падать!
FAILED test_schedule.py::test_structure - assert 0 > 0
=====================================================================
3 failed, 3 passed in 14.10s
=====================================================================
```