Python QA Engineer

Курс по активной прокачке навыков программирования на Python для QA-инженеров

Длительность курса: 192 академических часа

1 Введение в автоматизацию тестирования

1	**Введение в разработку и тестирование**	Изучить виды, цели, инструменты и инфраструктуру для автоматизированного тестирования. Домашние задания 1 Подготовка рабочего окружения Подготовка рабочего окружения для разработки автоматических тестов. 2 Пирамида автоматического тестирования (Опционально) Описать пирамиду автоматического тестирования своего текущего проекта. 3 Запуск тестов с помощью Pytest. Создание и запуск тестов с помощью Pytest.
2	**Введение в Pytest**	Познакомить студентов с фреймворком для запуска тестов Pytest.

3	**Тестирование API**	Научиться тестировать REST API-сервисы. Домашние задания 1 Параметризованное тестирование REST API. Тестирование REST API сервиса с помощью Python используя библиотеки pytest, requests, json.
4	**Data Driven Testing**	Изучить подходы Data Driven Testing. Научиться выбирать правильный формат представления данных.

2 Тестирование UI

1 Основы Selenium

Изучить минимальный набор знаний и навыков для работы с Selenium.

Домашние задания

1. Настройка окружения

   1. Установить OpenCart по инструкции
   2. Настроить selenium для запуска тестов

2. Первый тест

   1. Написать фикстуру для запуска трех разных браузеров (ie, firefox, chrome) в полноэкранном режиме с опцией headless. Выбор браузера должен осуществляться путем передачи аргумента командной строки pytest. По завершению работы тестов должно осуществляться закрытие браузера.
   2. Добавить опцию командной строки, которая указывает базовый URL opencart.
   3. Написать тест, который открывает основную страницу opencart (http://<ip_or_fqdn>/opencart/) и проверяет, что мы находимся именно на странице приложения opencart.

---

2 Поиск элементов

Научить искать элементы с помощью Selenium. Познакомиться с базовыми алгоритмами поиска элемента в массиве.

Домашние задания

1. Поиск элементов на странице.

   Найти элементы на странице используя различные виды локаторов.

---

3 Работа с элементами

Научиться работать с элементами. Изучение основ ООП в Python.

Домашние задания

1. Работа с элементами.

   Работа с текстом и другими атрибутами элемента веб страницы.

---

4 Действия с элементами

Научиться работать с объектом WebElement и изучить основы ООП в Python.

---

5 **Ожидание элементов**

Научиться работать с ожиданиями элементов. Изучить работую с исключениями в Python.

Домашние задания

1 Ожидание элементов.

Нажатие кнопок, заполнение и очистка текстовых полей.

---

6 **Шаблон проектирования PageObject**

Изучение паттерна PageObject.

Домашние задания

1 PageObject.

Пишем тесты в паттерне PageObject.

---

7 **Работа с окнами**

Научиться работать с окнами.

Домашние задания

1 Написать тест создания товара с добавлением картинок

Добавить 3 картинки к товару

2 Тест с перетаскиванием пункта меню

Выполнять ДЗ на демо по адресу http://demo23.opencart.pro/admin/ юзер demo пароль demo

3 Добавить файл для скачивания (в админ панели кнопка добавить - в выпадающем списке пункт "Загрузку"

/admin/index.php?route=catalog/download/add&token=oESV9zO9k5pKb3vpYTJlFt1OtrJqfGxY

---

8 **Протоколирование и отчетность**

Научиться логировать действия Selenium.

Домашние задания

1 Протоколирование

Настроить протоколирование проекта

---

9 **Удаленный запуск Selenium**

Научиться запускать Selenium удаленно.

Домашние задания

1 Постройте небольшой грид

Установите виртуальную машину, внутри которой работает Windows/Linux, и создайте грид, который состоит из диспетчера, работающего на вашей основной машине, и двух узлов -- один тоже на основной машине, а другой внутри виртуальной машины.

Настройте узлы так, чтобы в виртуальной машине был доступен браузер Firefox/Chrome, а на основной машине, наоборот, он был недоступен.

Попробуйте запустить какие-нибудь тесты удалённо на этом гриде, указывая разные браузеры, и убедитесь, что Firefox/Chrome, действительно запускается внутри виртуальной машины, а другие браузеры, наоборот, на вашей основной машине.

Можно использовать любую систему виртуализации, но если у вас нет предпочтений -- берите https://www.virtualbox.org/

Готовые образы Windows для разных систем виртуализации можно найти здесь: https://developer.microsoft.com/en-us/microsoft-edge/tools/vms/

2 Научитесь использовать облачный грид

Запустить несколько тестов в каком-нибудь облачном сервисе на выбор:

https://www.browserstack.com/
https://www.gridlastic.com/
https://saucelabs.com/
https://testingbot.com/

3 Отчётность и анализ логов

1 **Pytest-отчёты**

Научиться генерировать Pytest-отчёты.

Домашние задания

1 Кастомизированные отчеты pytest.

---

2 **Allure-отчёты**

Изучить Allure для генерации отчётов.

Домашние задания

1 Отчеты Allure

Поддержка отчетов Allure.

---

3 **Анализ логов веб-серверов**

Научить студентов разбираться в логах веб серверов.

Домашние задания

1 Анализ логов веб-сервера

В качестве источника логов взять логи opencart.
1. Написать скрипт анализа access.log для apache или nginx

Требования к реализации
1. Должна быть возможность указать директорию где искать логи или конкретный файл
2. Должна быть возможность выбрать все файлы логов отпарсить или только заданный
3. В случае если файл не может быть обработан, то скрипт должен завершится с ошибкой
4. Для access.log должна собираться следующая информация:
- общее количество выполненных запросов
- количество запросов по типу: GET - 20, POST - 10 и т.п.
- топ 10 IP адресов, с которых были сделаны запросы
- топ 10 самых долгих запросов, должно быть видно метод, url, ip, время запроса
- топ 10 запросов, которые завершились клиентской ошибкой, должно быть видно метод, url, статус код, ip адрес
- топ 10 запросов, которые завершились ошибкой со стороны сервера, должно быть видно метод, url, статус код, ip адрес
5. Собранная статистика должна быть сохранена в json файл
6. Должен быть README файл, который описывает как работает скрипт

---

4 **Траблшутинг в Linux. Файловая подсистема и работа процессов**

Научить студентов диагностировать проблемы в работе файловой подсистемы и работы процессов.

---

5 **Траблшутинг в Linux. Диагностика сетевых неисправностей**

Студенты научатся диагностировать проблемы на уровне сети.

---

6 **Траблшутинг в Linux. Архитектура ОС Linux. Дебаг.**

Научить студентов использованию инструментам дебага. Изучить архитектуру Linux.

4 Работа с бэкендом

1 **Работа с СУБД**

Научить студентов работать из Python с СУБД.

Домашние задания

1. Работа с СУБД.

   Тестирование работы СУБД на примере работы с PostgreSQL с помощью библиотеки psycopg2.

---

2 **Работа с сетью I. Протоколы прикладного уровня**

Научить студентов работать с сетевыми протоколами прикладного уровня.

Домашние задания

1. Работа с сетью. Протоколы прикладкого уровня.*

   Пишем код, который будет осуществлять подключения по SSH и работать с FTP.
2. Тесты бэкенд с использованием SSH клиента

   Добавить тесты, которые используют SSH клиент для реализации следующих сценариев: перезагрузка сервера opencart с последующей проверкой, что opencart доступен, рестарт основных сервисов для opencart с последующей проверкой, что сервис доступен.

---

3 **Работа с сетью II. Протоколы низкого уровня**

Научить работать с сетевыми протоколами низкого уровня, углубить знания в области работы сетей и веб-приложений.

Домашние задания

1. Работа с сетью. Протоколы низкого уровня.

   Пишем собственный HTTP клиент с использованием библиотеки socket.
2. Парсинг html страницы средствами python.*

   Необходимо расширить предыдущее домашнее задание по парсингу заголовков HTTP функциональностью парсинга тела ответа.

4	**Работа с ОС Linux с помощью Python**	Студенты научатся работать с операционной системой Linux средствами Python.

Домашние задания

1. Работа с ОС Linux с помощью Python.

   Тесты, которые работают с сетевой, файловой и системой управления процессами Linux.

5 Практики DevOps и безопасность

1 **Виртуализация. Контейнеры**

Изучить основные виды виртуализации.

Домашние задания

1. Работа с Docker контейнерами.

   Создаем свой контейнер, в который помещаем код тестового репозитория

---

2 **Виртуализация. Виртуальные машины**

Изучить основы виртуализации и узнать как реализованы докер контейнеры изнутри.

---

3 **Непрерывная интеграция**

Изучение основ непрерывной интеграции в контексте тестирования.

Домашние задания

1. Настройка стилистического анализа в Jenkins

   Настраиваем Jenkins на запуск стилистических анализаторов на проверку кода тестового фреймворка по каждому коммиту.
2. Установка Jenkins

   Установить Jenkins на виртуальную машину Ubuntu или в докер контейнер.
3. Запуск тестов с помощью Jenkins.

   Создать в дженкинсе пайплайн, который:
   1. Разворачивает докер контейнер с последним кодом тестов
   2. Запускает тесты
   3. Сохраняет отчет о тестах в артифакт джобы

---

4 **Подготовка тестового окружения**

Изучение основ сборки пакетов.
Изучить основные практики подготовки тестового окружения.

Домашние задания

1. Сборка wheel тестового фреймворка

   Добавить setup.py для тестового фреймворка.

5	**Основы безопасности веб-приложений**	Изучение основ безопасности веб-приложений для проведения тестирования безопасности.

6 Другие виды тестирования

1	**Основы Behaviour Drivet Testing. Введение в Robot Framework**	Изучение основ BDT и знакомство с Robot Framework. Домашние задания 1 BDT. Robot Framework. Пишем тесты на Robot Framework.
2	**Расширенное использование Robot Framework**	Закрепление навыков работы с Robot Framework.
3	**Модульное тестирование.**	Научить писать студентов модульные тесты. Домашние задания 1 Модульное тестирование. Mock Objects. Пишем модульные тесты в стиле xunit используя MockObjects.
4	**Использование Mock**	Изучение подхода Mock в тестировании.