УДК 004.422

СПЕЦИФИКА РАЗРАБОТКИ ФРЕЙМВОРКА ДЛЯ АВТОМАТИЗАЦИИ ТЕСТИРОВАНИЯ

Сирко А.С.

Белорусский государственный университет информатики и радиоэлектроники, г. Минск, Республика Беларусь

Научный руководитель: Балтрукович П. И. – к. т. н, доцент, доцент кафедры ИПиЭ

Аннотация. Статья посвящена специфике создания фреймворка для автоматизации тестирования. В ней рассматриваются необходимый функционал и основные задачи по созданию современного инструмента автоматизации.

Ключевые слова: автоматизация тестирования, фреймворк, разработка фреймворка

Введение. На данный момент в сфере тестирования большинство операций осуществляется вручную специалистами по тестированию программного обеспечения (ПО). Процесс функционального тестирования затрагивает анализ требований, составление тестовой документации, осуществление собственно тестирования, обнаружение и описание дефектов.

Такой жизненный цикл тестирования занимает много времени у исполнителей и требует сильной концентрации внимания. Для решения этой проблемы на современных проектах вводится автоматизация тестирования. Роль специалиста по автоматизации тестирования относительно новая и инновационная в индустрии информационных технологий.

Автоматизация тестирования играет ключевую роль в обеспечении качества программных продуктов. Она позволяет проводить тесты быстрее, эффективнее и точнее, чем это возможно при ручном тестировании. Это особенно важно в условиях современных методологий разработки, таких как Agile (гибкая методика разработки) и DevOps (методология разработки и эксплуатации), где скорость и гибкость являются критическими факторами успеха [1].

Однако, несмотря на все преимущества, автоматизация тестирования – это сложная задача, требующая глубоких знаний и опыта. Она включает в себя выбор правильных инструментов, разработку эффективных стратегий и управление сложными процессами.

Разработка эргономичного фреймворка для автоматизации своих тестовых сценариев позволяет специалисту по тестированию ПО не тратить рабочее время на повторное выполнение однотипных задач, доверяя этот процесс современным технологиям. В связи с описанными проблемами есть необходимость разработки нового фреймворка для автоматизации тестирования, который покроет нужды тестировщиков и избавит их от рутинных задач.

Основная часть. Для создания эффективного фреймворка требуется предоставить набор классов, методов и интерфейсов, разработанных с учетом потребностей специалистов по тестированию.

С его помощью специалисты должны иметь возможность автоматизировать свои тестовые сценарии, соблюдая TDD (Test-driven development) и BDD (Behavior-driven development) подходы [2]. Также он должен позволить производить автоматический запуск тестов для регрессионного тестирования, не тратя рабочее время на повторное выполнение однотипных задач.

Основные задачи фреймворка для автоматизации базовых тестовых сценариев включают в себя следующие:

– поиск и хранение веб-элементов в интерфейсе веб-приложений и сервисов;

– симуляцию взаимодействия с элементами интерфейса;
– обеспечение доступа к состоянию элементов и метаданным страницы;
– автоматизированный запуск и анализ тестов.

Данный функционал предоставлен в любом существующем фреймворке для автоматизации. Но существует необходимость разработки такого инструмента, который смог бы покрыть более обширные действия тестировщиков [3].

Это, прежде всего связано с тем, что в современном мире обеспечение качества программных продуктов включает в себя не только функциональное тестирование интерфейса, а также и тестовые сценарии, включающие в себя:

– взаимодействие приложения со сторонними сервисами и микросервисами;
– корректную работу API (Application programming interface);
– взаимодействие с базой данных приложения;
– обработку скачивания и загрузки файлов;
– использование инструментов разработчика, встроенных в браузер.

Построение архитектуры для работы с API – это то, чего не хватает существующим фреймворкам и для добавления этого функционала нужна современная разработка. Визуализация взаимодействия фреймворка (клиента) и API сервера предоставлена на рисунке 1:

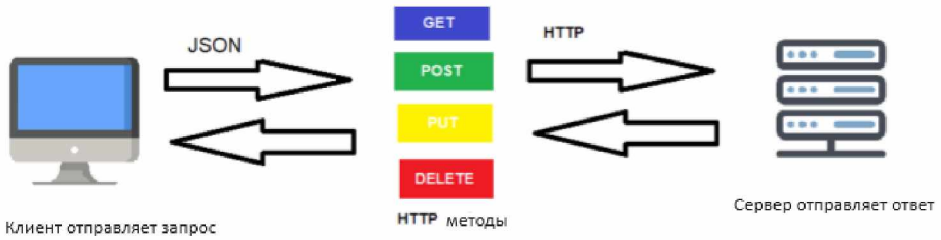

Рисунок 1. – Взаимодействие фреймворка и REST API клиента через HTTP запросы

Опыт показывает, что для такой разработки требуется три основных этапа.

Во-первых, необходимо определить конечные точки API, с которыми будет происходить взаимодействие. Это включает в себя определение базового URL (Uniform resource locator), методов HTTP (GET, POST, PUT, DELETE и т.д.), параметров запроса и ожидаемых ответов. Эта информация обычно предоставляется разработчиками API и должна быть четко задокументирована [4].

Во-вторых, в рамках фреймворка для автоматизации тестирования, необходимо создать функции или методы, которые будут выполнять запросы к API и обрабатывать ответы. Эти функции должны быть способны обрабатывать различные сценарии, включая успешные запросы, ошибки и исключения. Они также должны быть способны проверять, что ответы от API соответствуют ожидаемым результатам.

В-третьих, эти функции затем могут быть интегрированы в тестовые сценарии, которые будут автоматически выполняться в рамках процесса автоматизации тестирования. Это позволяет обеспечить непрерывное тестирование API на протяжении всего жизненного цикла разработки программного обеспечения, обеспечивая высокое качество и надежность системы.

Для реализации на самой актуальной сейчас платформе .NET 8 с использованием языка C# предложено использовать библиотеку RestSharp [5]. А для работы с форматом JSON в запросах используется базовая библиотека System.Text.Json [6].

Не менее важным дополнением, которое отсутствует в существующих фреймворках является реализация работы с базами данных.

Разработка логики базы данных во фреймворке необходима для:

– создания тестовых пользователей;
– испытания SQL-инъекций;

– очистки тестовых данных после окончания автотеста;

– сравнения ожидаемого результата из базы данных с действительными данными в приложении.

Для разработки этого функционала предлагается использование модели MODEL-CONTROLLER-VIEW (модель-контроллер-вид) (Рисунок 2).

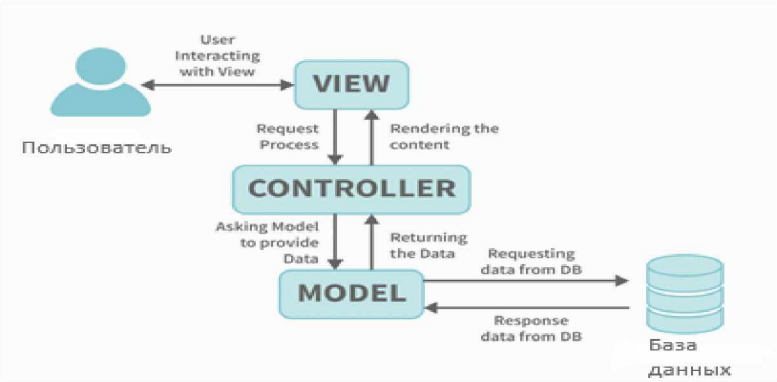

Рисунок 2 – Вид взаимодействия с базой данных через MODEL-CONTROLLER-VIEW

Для реализации этой модели предложено использовать ORM (Object-relational mapping) библиотеку Entity Framewok для .NET.

Обработку скачивания и загрузки файлов во фреймворке планируется обеспечить через классы пространства имен System.IO, такие как FileStream, Path, Directory. Для оптимальной нагрузки рекомендуется использовать отдельный поток для чтения файлов.

Заключение. В статье была рассмотрена необходимость и специфика создания фреймворка для автоматизации тестирования, функционал, отсутствующий в существующих инструментах автоматизации. Также предложен подход к разработке необходимого функционала для упрощения работы специалиста по тестированию ПО.

Список литературы

1. What is Automation Testing: Benefits, Strategy, Tools[Электронный ресурс] – Режим доступа: https://www.browserstack.com/guide/automation-testing-tutorial – Дата доступа 04.02.2024

2. TDD vs BDD vs ATDD : Key Differences[Электронный ресурс] – Режим доступа:https://www.browserstack.com/guide/tdd-vs-bdd-vs-atdd – Дата доступа 04.02.2024

3. 35 Best Test Automation Frameworks for 2024[Электронный ресурс] – Режим доступа: https://www.lambdatest.com/blog/best-test-automation-frameworks/

4. HTTP request methods[Электронный ресурс] – Режим доступа: https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods – Дата доступа 04.02.2024

5. RestSharp documentation [Электронный ресурс] – Режим доступа: https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods - Дата доступа 04.02.2024

6. How to work with JSON in .NET [Электронный ресурс] – Режим доступа: https://learn.microsoft.com/en-us/dotnet/standard/serialization/system-text-json/how-to- Дата доступа 04.02.2024

UDC 004.422

SPECIFICS OF DEVELOPING A FRAMEWORK FOR TEST AUTOMATION

Sirko A. S.

Belarusian State University of Informatics and Radioelectronics, Minsk, Republic of Belarus

Baltrukovich P.I.– Cand. of Sci., associate professor, associate professor of the department of IP&E

Annotation. The article is devoted to the specifics of creating a testing automation framework. It discusses the necessary functionality and main tasks for creating a modern automation tool.

Keywords: test automation, framework, framework development.