МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»**
Кафедра дискретной математики и информационных технологий

ИССЛЕДОВАНИЕ ВОЗМОЖНОСТЕЙ ЯЗЫКА PYTHON ДЛЯ АВТОМАТИЗИРОВАННОГО ТЕСТИРОВАНИЯ

АВТОРЕФЕРАТ МАГИСТЕРСКОЙ РАБОТЫ

студента 2 курса 271 группы
направления 09.04.01 — Информатика и вычислительная техника
факультета КНиИТ
Астраханцева Андрея Сергеевича

Научный руководитель
доцент, к. ф.-м. н. _____ В. А. Поздняков

Заведующий кафедрой
доцент, к. ф.-м. н. _____ Л. Б. Тяпаев

Саратов 2022

ВВЕДЕНИЕ

В настоящее время программное обеспечение оказывает значительное влияние на все аспекты жизни нашего общества, и его функционирование имеет решающее значение. Как следствие, обеспечение правильности и качества программных систем и компонентов становится первостепенной задачей.

За последние годы жизненные циклы разработки программного обеспечения постоянно сокращаются, а новые версии продукта выходят все чаще и чаще. Чтобы гарантировать качество этих выпусков, в индустрии разработки программного обеспечения стал заметен сильный сдвиг в сторону автоматизированного тестирования на всех уровнях тестирования.

Автоматизированное тестирование позволяет сократить время и ресурсы ручного тестирования, а также снизить риск выпуска некачественного продукта на рынок.

Проблемой автоматизированного тестирования является обеспечение работы автоматических тестов. Для того, чтобы упростить данный процесс, предоставляется возможность использовать фреймворки автоматизированного тестирования. Такой подход дает возможность сфокусироваться на поставленной задаче, не задумываясь о разработке вспомогательных инструментов.

На текущий момент существует достаточное количество фреймворков, которые используются для автоматизированного тестирования, но они нацелены на узкий круг области использования.

Целью данной работы является исследование инструментов на языке Python для разработки универсального фреймворка автоматизированного тестирования.

В ходе работы были поставлены следующие задачи:

- обозначить проблемы автоматизированного тестирования;
- анализ предметной области и формулирование требований;
- анализ и выбор существующих инструментов на языке Python;
- проектирование проекта;
- разработка фреймворка для автоматизированного тестирования.

Объектом исследования данной работы является автоматизированное тестирование. Предметом исследования является фреймворк для автоматизированного тестирования, основанный на исследуемых инструментах, которые написаны на языке Python.

В первой главе рассматривается понятие, значение и проблема автоматизированного тестирования. Во второй главе продемонстрированы инструменты на языке Python, которые позволяют запускать тесты, а также те, при помощи которых предоставляется возможность взаимодействовать с разными объектами тестирования. В третьей главе проведен анализ тестируемого приложения и выдвинуты требования, на основе которых сформирован список требований для тестирования программного продукта. Четвертая глава посвящена проектированию, разработке и применению фреймворка автоматизированного тестирования.

Практическая значимость результата данной работы: разработанные тесты, при помощи реализованного фреймворка, в настоящий момент используются в процессе тестирования комплекса программных продуктов, которые разрабатываются в рамках одной компании. Его внедрение позволило ускорить время на тестирование и разработку, а также повысить качество разрабатываемых продуктов.

Магистерская работа состоит из введения, четырех глав, заключения и списка используемых источников. Общий объем работы — 61 страниц, из них 60 страниц — основное содержание, включая 31 рисунок, список использованных источников информации — 22 наименования.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

В первой главе рассказывается о автоматизированном тестировании. В ней дается понятие, перечисляются преимущества и недостатки автоматизированного тестирования, а также продемонстрированы уровни автоматизации. Автотестирование не может полностью исключить ручное тестирование из процесса разработки, но может значительно сократить его долю, тем самым помочь убрать рутину из процесса тестирования, ускорить и улучшить качество разрабатываемых программных продуктов. Также поднимается проблема автоматизированного тестирования, которая заключается в недостатке квалификации или специализированных знаний языка и среды разработки, а также использования инструментов, которые создают тесты при помощи записи и воспроизведения. Такие тесты очень тяжелы в поддержке. Данную проблему может решить фреймворк, потому что он основан на концепциях и практиках, направленные на переиспользование, уменьшение затрат на поддержку и повышение надежности использования тестов.

Вторая глава посвящена исследованию инструментов языка Python, которые могут использоваться в автотестировании. Одним из главных инструментов автоматизированного тестирования — это инструмент сбора и запуска тестов. Тут продемонстрированы два популярных инструмента, такие как Unittest и Pytest, в которых приведены примеры использования и описаны все плюсы и минусы. На основе этих заключений, в качестве сбора и запуска тестов был выбран Pytest. Также были продемонстрированы инструменты для взаимодействия с браузером Selenium, API на основе инструмента request и базой данных Psycopg2.

В третьей главе рассматривается анализ предметной области и формулирование требований, где описывается и исследуется программное обеспечение «ПК ЦУМИ», которое будет в дальнейшем покрыто автоматизированными тестами. Был произведен анализ существующих программных реализаций, которые используются для автотестирования web-приложений. На основе вышеприведенных исследований, были сформулированы требования и задачи к разработанному фреймворку для автоматизированного тестирования web-приложений.

Четвертая глава посвящена разработке фреймворка для автоматизированного тестирования web-приложения. В данной главе сформирована логи-

ческая модель фреймворка, которая продемонстрирована на рисунке 1. Далее были описаны и разработаны пакеты инструментов, которые будут позволять взаимодействовать с пользовательским интерфейсом, с удаленным сервером при помощи протоколов SSH и SFTP, с реляционной базой данных PostgreSQL, с API сервера. Также разработаны пакеты, в которых хранятся конфигурационные файлы и классы исключений и определены все повторяющиеся основные методы для взаимодействия с тестируемым продуктом. Все пакеты будут храниться в локальном удаленном репозитории Nexus Repository Manager. Далее были определены характеристики входной и результирующей информации. Входной информацией является сценарий тестирования (тест-кейс) и тестовые данные, необходимые для их выполнения. Сценарии тестирования в разрабатываемом фреймворке будут соответствовать методологии BDD. В соответствии с данной методологией, сценарий тестирования описывается в следующем формате: дается (Given) начальное значение, если (When) выполняется условие, то (Then) получаем определенный результат. Данная схема позволяет довольно легко переходить от описанного в текстовом виде к сценарию, описанному для автоматического выполнения фреймворком. Результативная информация, поступающая после завершения работы разрабатываемого фреймворка, представляет собой сведения о прогонах тестов, представленные в разных формах для разных категорий пользователей. Результаты будут представляться в следующих формах: лог файл, в котором записываются действия в шагах, описанные в сценариях тестирования; визуальный отчет. Данный отчет служит для выведения информации более широкому кругу пользователей. Он отображается в формате интерактивной web-страницы, которая содержит результаты прогонов конкретных тестов, тестовые наборы, а также общую статистику. На основе всех требований, была спроектирована и разработана архитектура фреймворка для автоматизированного тестирования web-приложений, изображенная на рисунке 2.

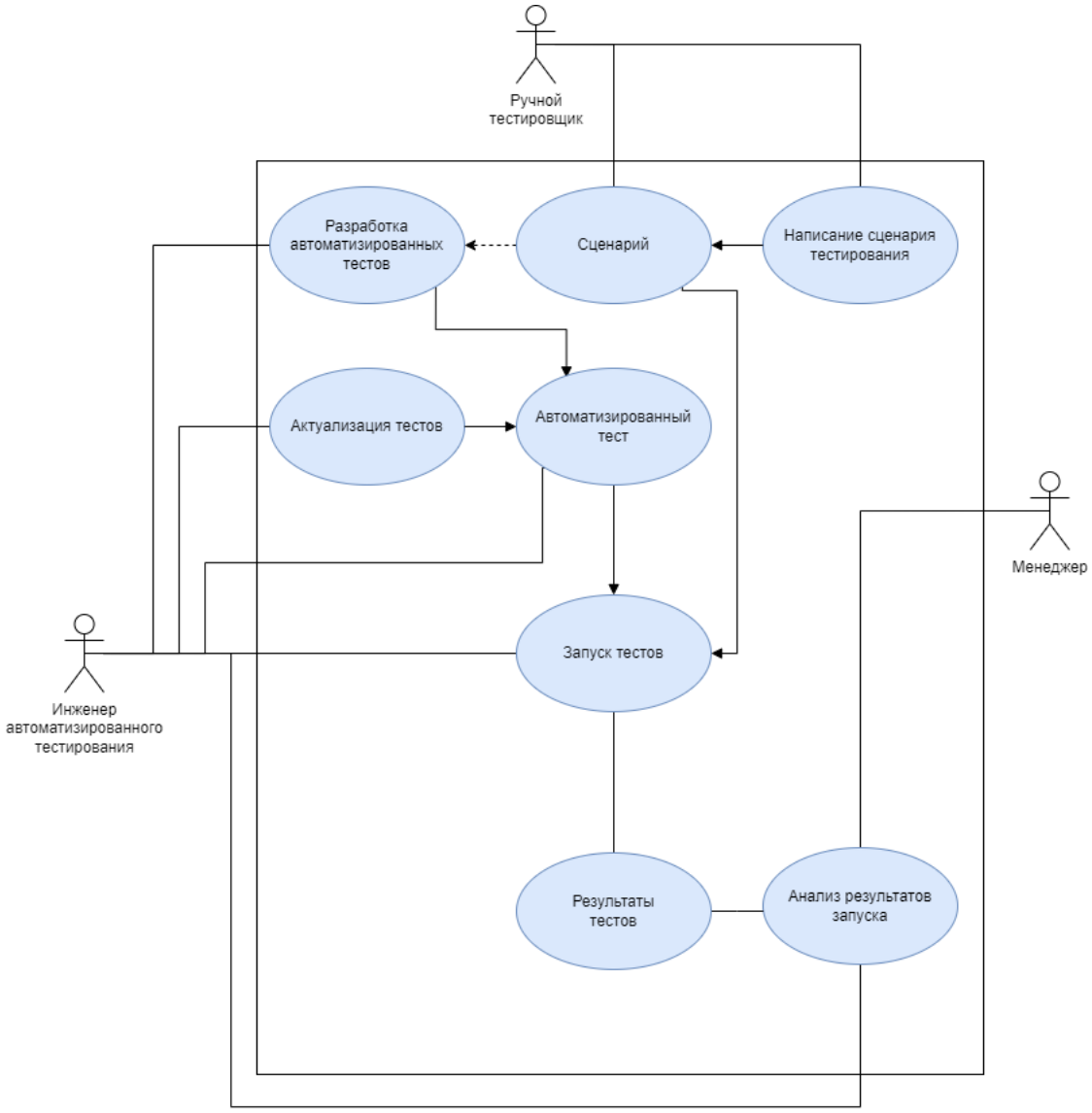

Рисунок 1 – Диаграмма прецедентов использования фреймворка

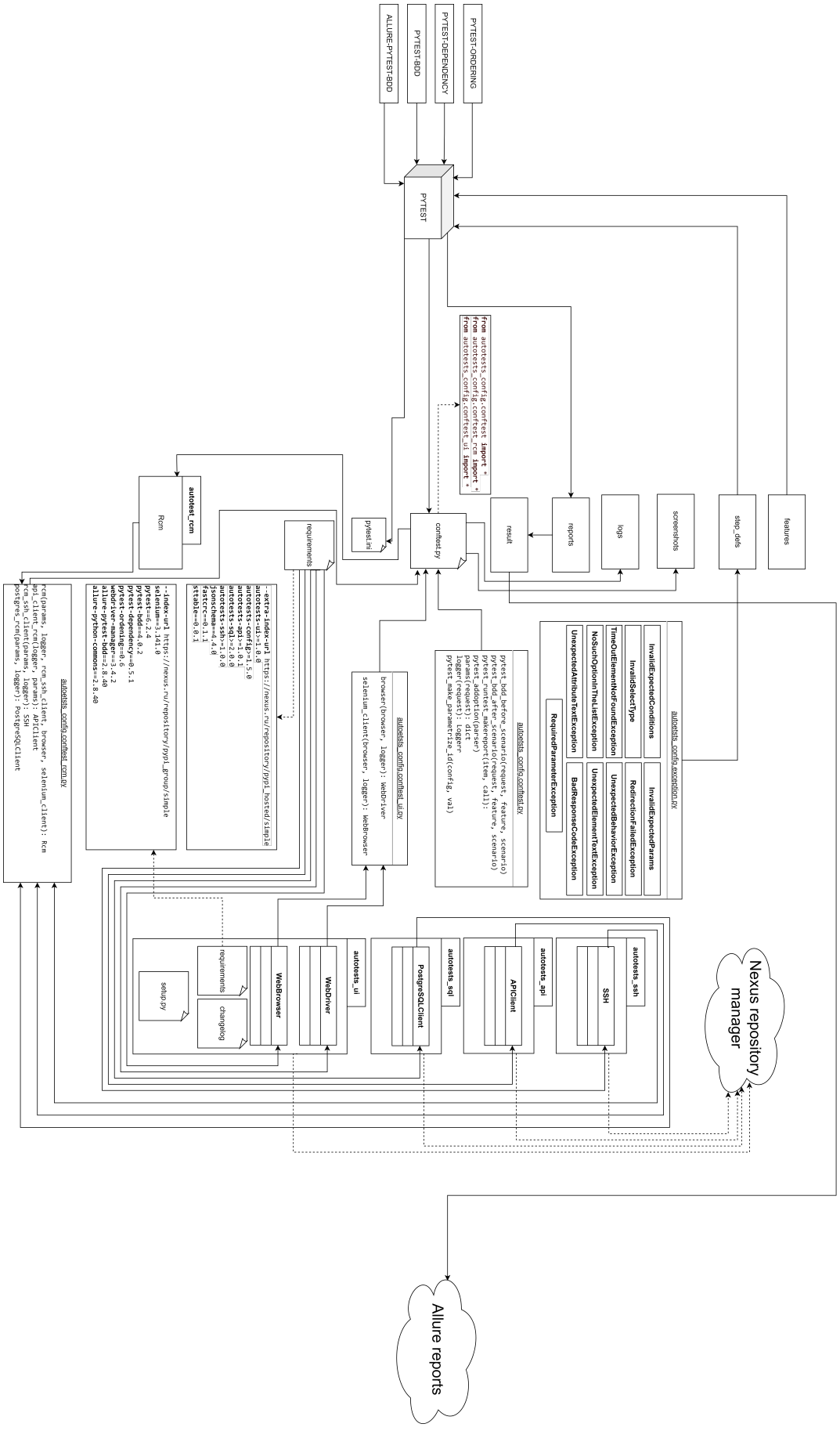

Рисунок 2 – Полная схема реализации фреймворка для автоматизированного тестирования web-приложений

ЗАКЛЮЧЕНИЕ

Фреймворк для автоматизированного тестирования может сыграть важную роль в эффективности написания тестов и проверки программного обеспечения. На текущий момент внедрение автоматизированного тестирования применяется во многих сферах, где разрабатывается программное обеспечение.

Разработанный фреймворк для автоматизированного тестирования web-приложений имеет возможность:

1. тестировать web-интерфейс, симулируя действия пользователя;
2. тестировать API запросы;
3. тестировать базы данных;
4. работать с удаленной машиной, при помощи протоколов SSH/SFTP;
5. логировать каждое действие, описанные в шаге сценария;
6. собирать отчет в формате html по пройденным тестам, при помощи фреймворка Allure.

Особенность данной разработки является возможность одновременного обращения к разным объектам тестирования, что позволяет тестировать несколько областей web-приложения.

На этапе проектирования была выбрана методология BDD. Для составления отчета по пройденным тестам выбран инструмент Allure. Для связки автоматизированных тестов с тест кейсами, описанные в сценариях Gherkin, был выбран инструмент Pytest, к которому подключены следующие плагины: pytest-ordering, pytest-dependency, pytest-bdd, allure-pytest-bdd. Данные плагины отвечают за последовательность запусков тестов, зависимости между тестам, связку шага сценария с функцией и создание репорт отчета для Allure, соответственно.

Для работы с браузером был реализован пакет autotests_ui, который основам на таких инструментах, как: selenium, webdriver-manager. В данной библиотеке реализованы основные методы, которые повторяют действия пользователя. Для обращения по API к серверу создан пакет autotests_api. Разработанный инструмент позволяет взаимодействовать с заголовком запроса, а также отправлять и получать запросы от сервера, указывая ожидаемый код статус. Для отправки и получения ответа запроса к реляционной базе данных PostgreSQL реализован пакет autotests_sql. Для управления удаленной ОС по SSH/SFTP

реализован пакет autotests_ssh. Для хранения конфигурационных файлов реализован autotests_config. В нем хранятся основные хуки и фикстуры, в которых определяются экземпляры классов реализованных инструментов, для взаимодействия с объектами тестирования, непосредственно на запущенном сервере, а также реализованные исключения для увеличения понятности ошибки.

Все выше реализованные инструменты хранятся на удаленном локальном репозитории Nexus Repository Manager. Данное решение позволяет хранить реализованные решения локально, а также поддерживать версионирование пакетов.

В результате разработанный фреймворк автоматизированного тестирования web-приложений позволяет ускорить время написания автоматизированных тестов.

В настоящее время фреймворк для автоматизированного тестирования web-приложений используется сотрудниками компании ООО «СЦ ТТМ», а также все написанные тесты включены в непрерывную интеграцию. Внедрение автоматизации помогло сократить время на тестирование, а также улучшить качество продукта.

В качестве дальнейшего развития, предлагается разработка программного обеспечения для автоматизированного тестирования на базе разработанного фреймворка, что позволит запускать интеграционные и приемочные тесты, а также хранить артефакты пройденных тестов.

Основные источники информации:

1 Full pytest documentation — pytest documentation [Электронный ресурс]. — URL: https://docs.pytest.org/en/7.0.x/contents.html (Дата обращения 20.05.2021). Загл. с экр. Яз. англ.

2 *Дастин, Э.* Автоматизированное тестирование программного обеспечения / Э. Дастин, Д. Рэшка, Д. Пол. — Москва: Лори, 2003. — 592 с.

3 Selenium with Python — Selenium Python Bindings 2 documentation [Электронный ресурс]. — URL: https://selenium-python.readthedocs.io/ (Дата обращения 20.05.2021). Загл. с экр. Яз. англ.

4 Requests: HTTP for Humans™ — Requests 2.27.1 documentation [Электронный ресурс]. — URL: https://docs.python-requests.org/en/latest/ (Дата обращения 20.05.2021). Загл. с экр. Яз. англ.

5 unittest — Unit testing framework — Python 3.10.2 documentation [Электронный ресурс]. — URL: https://docs.python.org/3/library/unittest.html (Дата обращения 20.05.2021). Загл. с экр. Яз. англ.

6 *Okken, B.* Python Testing with pytest: Simple, Rapid, Effective, and Scalable / B. Okken. — Pragmatic Bookshelf; 1st edition, 2017. — 216 pp.

7 *Molina, A.* Crafting Test-Driven Software with Python. Write test suites that scale with your applications' needs and complexity using Python and PyTest / A. Molina. — Packt Publishing, 2021. — 338 pp.

8 *Westerveld, D.* API Testing and Development with Postman. A practical guide to creating, testing, and managing APIs for automated software testing / D. Westerveld. — Packt Publishing, 2021. — 340 pp.

9 *Raghavendra, S.* Python Testing with Selenium. Learn to Implement Different Testing Techniques Using the Selenium WebDriver / S. Raghavendra. — 2020. — 196 pp.

10 *Pajankar, A.* Python Unit Test Automation. Automate, Organize, and Execute Unit Tests in Python / A. Pajankar. — 2021. — 232 pp.

11 *Gundecha, U.* Learning Selenium Testing Tools with Python / U. Gundecha. — 2014. — 216 pp.

12 *Lawrence, R.* Behavior-Driven Development with Cucumber: Better Collaboration for Better Softwaren / R. Lawrence, P. Rayner. — 2019. — 208 pp.

13 *Smart, J. F.* BDD in Action: Behavior-driven development for the whole software lifecycle / J. F. Smart. — Manning, 2014. — 384 pp.

14 SSH agents — Paramiko documentation [Электронный ресурс]. — URL: https://docs.paramiko.org/en/stable/api/agent.html (Дата обращения 20.04.2022). Загл. с экр. Яз. англ.

15 Basic module usage — Psycopg 2.9.3 documentation [Электронный ресурс]. — URL: https://www.psycopg.org/docs/usage.html (Дата обращения 20.04.2022). Загл. с экр. Яз. англ.

16 Allure Framework [Электронный ресурс]. — URL: https://docs.qameta.io/allure/#_pytest (Дата обращения 20.04.2022). Загл. с экр. Яз. рус.