МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева (Самарский университет)»

Институт информатики и кибернетики

Кафедра программных систем

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**  
к курсовому проекту по дисциплине   
«Автоматизация тестирования программного обеспечения»

Студент В.А. Артамонов

Студент Д.А. Смирнов

Руководитель А.А. Лобанков

Самара 2022

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева (Самарский университет)»

Институт информатики, математики и электроники

Факультет информатики

Кафедра программных систем

ЗАДАНИЕ

на курсовой проект по дисциплине

«Автоматизация тестирования программного обеспечения»

студентам группы № 6231-020402D

В.А. Артамонову

Д.А. Смирнову

1. Задание**:** Описать все возможные тест кейсы для тестирования функционала данного сайта с использованием Gherkin нотации. С помощью сквозного тестирования протестировать сайт ParaBank с использованием gherkin и cucumber, для обращения к одним и тем же элементам страницы рекомендуется использовать паттерн Page Object.
2. Исходные данные к проекту**:** см. приложение к заданию
3. Перечень вопросов, подлежащих разработке:
   1. Проанализировать функционал сайта ParaBank
   2. Описать все возможные тестовые сценарии
   3. Описать тестовые сценарии в Gherkin нотации
   4. Реализовать end-to-end тестирование сайта с использованием Gherkin и Cypress
   5. Оформить документацию курсового проекта
4. Календарный план выполнения работ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № п/п | Содержание работы по этапам | Объем этапа в % к общему объему проекта | Срок  окончания | Фактическое выполнение |
| 1 | Оформление технического задания и его утверждение | 10 | 12.09.2022 |  |
| 2 | Утверждение тест кейсов | 15 | 19.09.2022 |  |
| 3 | Описание тест кейсов в Gherkin нотации | 30 | 03.10.2022 |  |
| 4 | Демонстрация e2e тестирования с использованием cypress и cucumber | 30 | 31.10.2022 |  |
| 5 | Корректировка проекта и оформление документации проекта | 15 | 12.12.2022 |  |

Задание принял  
 к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В.А. Артамонов

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Д.А. Смирнов

ПРИЛОЖЕНИЕ  
к заданию на курсовой проект  
студентам группы № 6231-020402D  
В.А. Артамонову

Д.А. Смирнову

Тема проекта: «Тестирование сайта ParaBank с   
использованием cypress и cucumber»

Исходные данные к проекту:

1. Характеристика объекта автоматизации:

## объект автоматизации: процесс тестирования сайта ParaBank;

## виды автоматизируемой деятельности:

* + процесс сквозного тестирования сайта ParaBank.

1. Требования к техническому обеспечению:
2. тип ЭВМ – IBM PC совместимый;
3. монитор с разрешающей способностью не ниже 1280 х 720;
4. манипулятор – мышь;
5. технические характеристики определяются в процессе выполнения проекта.
6. Требования к программному обеспечению:
7. тип операционной системы ‑ Windows 10 и выше;
8. язык программирования – JavaScript;
9. среда разработки – Visual Studio Code;
10. Общие требования к проектируемой системе:

4.1 Функции, реализуемые системой:

1. общесистемные функции:
   * тестирование авторизации
   * тестирование быстрого перехода на главную страницу
   * тестирование обновления информации пользователя
   * тестирование перевода средств
   * тестирование поиска перевода
   * тестирование запроса кредита
   * тестирование формы обратной связи

4.2 Технические требования к системе:

1. режим работы – диалоговый;
2. система должна удовлетворять санитарным правилам и нормам  
    СанПин 2.2.2./2.4.2198-07;
3. условия работы средств вычислительной техники (содержание вредных веществ, пыли и подвижность воздуха) должны соответствовать ГОСТ 12.1.005, 12.01.007;
4. температура окружающего воздуха – 15-35°С;
5. влажность воздуха – 45-75%.

Руководитель   
проекта А.А. Лобанков

Задание принял  
к исполнению В.А. Артамонов

Д.А. Смирнов

РЕФЕРАТ

Пояснительная записка 28 с, 5 рисунков, 9 источников,  
2 приложения.

АВТОМАТИЗАЦИЯ, ТЕСТИРОВАНИЕ, СКВОЗНОЕ ТЕСТИРОВАНИЕ

Во время курсового проектирования был с помощью сквозного тестирования был протестирован сайт ParaBank с использованием cypress и cucumber.

Программное обеспечение разработано на языке JavaScript в среде Visual Studio Code.

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 8](#_Toc127123783)

[1 Описание предметной области 9](#_Toc127123784)

[1.1 Основные понятия и определения предметной области 9](#_Toc127123785)

[1.2 Тестирование программного обеспечения 9](#_Toc127123786)

[1.3 Особенности тестирования веб-приложений 10](#_Toc127123787)

[1.4 Пирамида тестирования 11](#_Toc127123788)

[1.5 Используемые инструменты тестирования 13](#_Toc127123789)

[1.5.1 Cypress 13](#_Toc127123790)

[1.5.2 Cucumber и Gherkin 14](#_Toc127123791)

[1.6 Постановка задачи 15](#_Toc127123792)

[2 Описание функционала сайта 16](#_Toc127123793)

[2.1 Перевод средств 16](#_Toc127123794)

[2.2 Быстрое перемещение на главную страницу 16](#_Toc127123795)

[2.3 Поиск перевода 16](#_Toc127123796)

[2.4 Обновление информации пользователя 16](#_Toc127123797)

[2.5 Форма обратной связи 16](#_Toc127123798)

[3 Реализация тестирования 17](#_Toc127123799)

[3.1 Описание тест кейсов в Gherkin нотации 17](#_Toc127123800)

[3.2 Сквозное тестирование с использованием cypress и cucumber 17](#_Toc127123801)

[Заключение 20](#_Toc127123802)

[Список использованных источников 21](#_Toc127123803)

[Приложение А Исходной код тест кейсов в Gherkin нотации 22](#_Toc127123804)

[Приложение Б Исходной код 2е2 тестирования сайта ParaBank 24](#_Toc127123805)

ВВЕДЕНИЕ

В разработке программного обеспечения (ПО) тестирование занимает важную роль. От того, как было проведено тестирование зачастую зависит качество разрабатываемого ПО. Кроме того, стоимость исправления программных ошибок увеличивается, если они не были определены и не были зафиксированы на раннем этапе.

Тестирование проводят для осуществления следующих целей:

* повышение вероятности того, что приложение, предназначенное для тестирования, будет работать правильно при любых обстоятельствах;
* повышение вероятности того, что приложение, предназначенное для тестирования, будет соответствовать всем описанным требованиям;
* предоставление актуальной информации о состоянии продукта на данный момент [1].

Возникновение ошибок при разработке клиент-серверных приложений не редкость. На данный момент существует множество инструментов для автоматизации тестирования приложения, позволяющие протестировать выполнение любой функции приложения для написания надежного кода.

В связи с этим становится актуальной задача изучения инструментария для проведения тестирования.

Цель курсового проекта – курсового проектирования является закрепление студентом навыков сквозного (end-to-end) тестирования.

.

1. Описание предметной области
   1. Основные понятия и определения предметной области

ParaBank — это демонстрационный сайт, используемый для демонстрации программных решений Parasoft. Все материалы на сайте используются исключительно для моделирования реалистичного веб-сайта онлайн-банкинга [2].

* 1. Тестирование программного обеспечения

Тестирование программного обеспечения – процесс исследования, испытания программного продукта, имеющий своей целью проверку соответствия между реальным поведением программы и её ожидаемым поведением на конечном наборе тестов, выбранных определённым образом. В более широком смысле, тестирование – это одна из техник контроля качества, включающая в себя активности по планированию работ, проектированию тестов, выполнению тестирования и анализу полученных результатов [3].

Рассмотрим роль тестирования в разработке ПО. В более узком смысле тестирование представляет из себя отдельную стадию жизненного цикла ПО.

Жизненный цикл программы обязательно состоит из следующих стадий разработки:

* планирование: на данной стадии определяется общее видение продукта, формируются и анализируются требования к программному продукту;
* проектирование: на данном этапе соответствующая группа специалистов решает, как будут реализованы запланированные возможности продукта;
* кодирование: непосредственно написание программного продукта;
* тестирование: выполняется проверка и отладка ПО в соответствии с требованиями;
* сопровождение: консультирование пользователей, исправление ошибок, внедрение доработок [4].
  1. Особенности тестирования веб-приложений

Web-приложения — динамично развивающаяся сфера. Не все подходы и методы, применяемые для тестирования классических приложений, могут быть применимы для тестирования web-приложений.

Web-приложение — это клиент-серверное приложение, в котором клиентом выступает браузер, а сервером web-сервер, что уже является по сути двумя разнополыми программами, которые необходимо тестировать как отдельно, так и в связке.

Имея много общего с тестированием классических приложений, тестирование web-ориентированных приложений имеет свои особенности, связанные прежде всего со средой функционирования. Имея компонентные, структурные и технологические особенности, web-приложениям присущи особенности режимов работы, инсталляции, запуска, остановки и удаления, а также формирования интерфейсов.

Особенности тестирования web-приложений:

1. Технологические отличия.

Web-приложение работает с использованием принципиально различных технологий

1. Структурные отличия.

Web-приложение — “многокомпонентное”. Состоит из большого числа модулей. Обязательно использует серверы БД, web-серверы, серверы приложений.

1. Отличия режимов работы.

Web-приложение работает в режиме “запрос-ответ”, т.е. известно о некотором наборе действий только после запроса на сервер.

1. Отличия формирования интерфейса.

Web-приложение использует для формирования пользовательского интерфейса стремительно развивающиеся технологии, множество которых конкурирует между собой.

1. Особенности сбоев и отказов

Web-приложение: выход из строя некоторых компонентов оказывает непредсказуемое влияние на работоспособность приложения в целом [5].

* 1. Пирамида тестирования

Тестовая пирамида - визуализация, описывающая различные уровни тестирования и объем тестирования на каждом слое тестирования (рисунок 1).

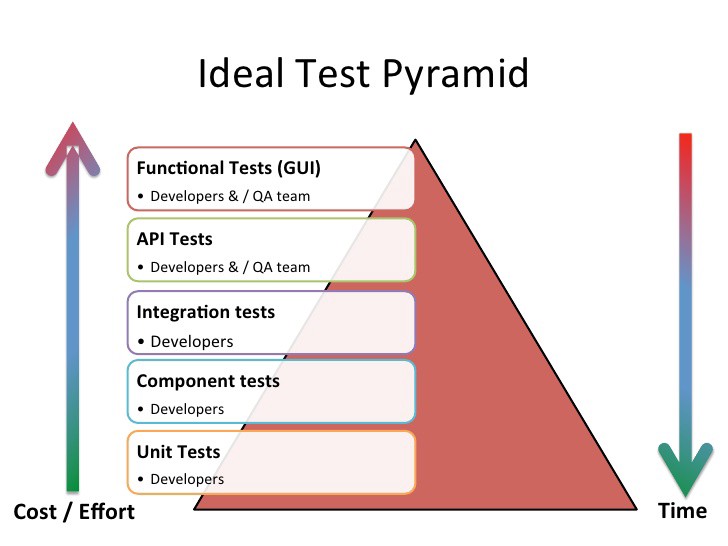


Рисунок 1 – Пирамида тестирования

Модульные тесты должны составлять основную часть автоматизированного тестирования.

- Задачи автоматизации не закрываются до тех пор, пока эти скрипты не будут запущены на реализованной функциональности;

- Разработка одновременно с модульными тестами заставляет разработчиков задуматься о проблеме, которую они решают, и о любых крайних случаях, с которыми они могут столкнуться;

- Тесты являются детальными и могут помочь точно определить дефект;

- Время выполнения невероятно быстрое, потому что им не нужно полагаться на какой-либо пользовательский интерфейс или внешние системы, такие как база данных или API;

- Они недорогие, просто пишутся, легко поддерживать.

Интеграционные тесты должны занимать середину пирамиды.

Используйте этот уровень для проверки бизнес-логики без использования пользовательского интерфейса (UI);

Тестируя за пределами пользовательского интерфейса, вы можете тестировать входы и выходы API или сервисов без всех сложностей, которые вводит пользовательский интерфейс;

Эти тесты медленнее и сложнее, чем модульные тесты, потому что им может потребоваться доступ к базе данных или другим компонентам.

Тесты пользовательского интерфейса или сквозные тесты должны размещаться на вершине пирамиды.

Сквозное тестирование, оно же End-to-end или E2E, – это процесс тестирования, при котором происходит подробная эмуляция пользовательской среды. То есть при данном тестировании имитируют:

* щелчки мышью,
* нажатия на кнопки,
* заполнение форм,
* переходы по страницам и ссылкам,
* и другие поведенческие факторы.

Суть этого тестирования — посмотреть, так ли работает программа для конечного клиента, как рассчитывалось изначально? При этом нужно учитывать, что пользователю все равно, функционирует ли программа «как надо», ему главное, чтобы программа функционировала и оправдывала ожидания, поэтому основной упор делается на корректное функционирование.

Е2Е-процесс происходит при помощи сложных программ для тестирования, написанных специально для тестирования или «вручную», от этого данный процесс требует много времени и затрат. Поэтому до его применения обычно проводят более дешевые и нетребовательные виды тестирования [7].

* 1. Используемые инструменты тестирования
     1. Cypress

Cypress — это open-source фреймворк, в основе которого лежат тестовые JavaScript-библиотеки Mocha и Chai. Он подходит для end-to-end, интеграционных и unit-тестов. В отличие от Selenium, Cypress не запускает удаленные команды, а выполняется прямо в браузере одновременно с тестируемым приложением.

Для автоматизаторов, которые до этого не имели дела c написанием тестов на JavaScript, работа Cypress может показаться необычной. Команды Cypress выполняются асинхронно, то есть непоследовательно. По сути, он выполняет тестовый код дважды. В первый раз — для чтения и постановки в очередь всех команд, которые нужны для выполнения. Затем загружает страницу и выполняет команды одну за другой.

В Cypress каждый it (тест) — это отдельная независимая сессия. То есть тем, кто, пользуясь другими инструментами, привык использовать одну и ту же сессию между тестами, нужно это учитывать. В Cypress реализован встроенный механизм очистки local/session storage, cookie и всего кэша, который срабатывает после прохождения каждого отдельного теста.

Cypress не позиционируется как универсальный инструмент для любых видов тестирования. Поддержка кроссбраузерности появилась в последней версии, ранее был только Chrome, сейчас тесты можно запускать в Chrome, Edge, Firefox(beta), Electron.

Достоинства Cypress могут оказаться решающими при выборе первого инструмента для обучения автоматизации:

1. Официальная документация и есть туториал.

2. Простота установки и настройки.

3. Если вы уже знакомы с JavaScript, то будет еще проще.

4. В Cypress есть возможность скопировать готовый селектор прямо из окошка тест-раннера. На этапе обучения эта функция экономит время и позволяет писать рабочие тесты даже с минимумом знаний html/css. Причем подбор селектора идет с соблюдением приоритета — если элементам добавлены специальные тестовые атрибуты data-cy или data-test, то будут выбраны именно они [8].

* + 1. Cucumber и Gherkin

Cucumber – это фреймворк, поддерживающий разработку через поведение (BDD). Центральное место в подходе BDD фреймворка Cucumber занимает его обычный синтаксический анализатор языка под названием Gherkin. Это позволяет определять ожидаемое поведение программного обеспечения на логическом языке, понятном клиентам. Таким образом, Cucumber позволяет выполнять функциональную документацию, написанную в виде делового текста.

Фреймворк Cucumber изначально был написан на языке программирования Ruby, и первоначально использовался исключительно для тестирования Ruby в качестве дополнения к платформе BDD RSpec. В настоящее время фреймворк поддерживает множество различных языков (включая Java).

Gherkin – это язык, который Cucumber использует для определения тестовых случаев. Он разработан, чтобы быть нетехническим и читаемым человеком, и в совокупности описывает варианты использования, относящиеся к программной системе. Цель синтаксиса Gherkin состоит в том, чтобы продвигать методы разработки, основанные на поведении, для всей команды разработчиков, включая бизнес-аналитиков и менеджеров. Он стремится обеспечить соблюдение твёрдых, однозначных требований, начиная с начальных этапов определения требований руководством бизнеса и на других этапах жизненного цикла разработки.

В дополнение к предоставлению сценария для автоматизированного тестирования, синтаксис естественного языка Gherkin предназначен для обеспечения простой документации тестируемого кода. В настоящее время Gherkin поддерживает ключевые слова на десятках естественных языков [9].

* 1. Постановка задачи

Во время курсового проектирования необходимо с помощью сквозного тестирования протестировать сайт ParaBank с использованием cypress и cucumber.

1. Описание функционала сайта
   1. Перевод средств

Существует возможность переводов. Пользователь может выбрать сумму перевода, номер счета с которого будет осуществлен перевод и номер счета на который будет осуществлен перевод

* 1. **Быстрое перемещение на главную страницу**

С помощью иконки в шапке страницы пользователь может быстро перейти на главную страницу сайта. Возврат на главную страницу является довольно распространенным действием, совершаемым пользователями на веб-ресурсах, поэтому важно иметь возможность перейти на главную страницу.

* 1. **Поиск перевода**

С помощью бокового меню можно перейти на страницу поиска перевода. Искать транзакцию можно по ID, дате или сумме перевода. Также можно выбрать аккаунт, с которым связан искомый перевод. Результаты появляются в таблице.

* 1. **Обновление информации пользователя**

Авторизованный пользователь может обновить информацию об аккаунте, такие как: фамилия, имя, адрес, город, телефон и т.д. Все поля являются обязательными к заполнению.

* 1. **Форма обратной связи**

Пользователь может отправить сообщение службе поддержки, для этого необходимо указать имя, электронную почту, телефон и сообщение.

1. Реализация тестирования
   1. Описание тест кейсов в Gherkin нотации

Тест кейсы были описаны в Gherkin нотации, код приведен в примечании.

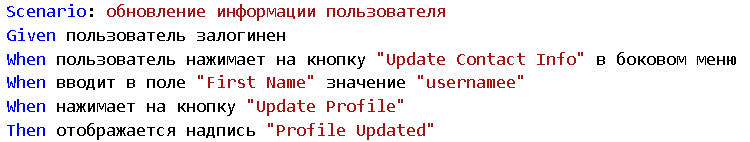
Каждая строка начинается с одного из ключевых слов и описывает один из шагов. Большинство строк в Gherkin начинается с ключевых слов и состоят из функций и сценариев. Например, код тест кейса “Обновление информации пользователя” описан на рисунке 2.   


Рисунок 2 – тест кейс “Добавление новой страницы” в Gherkin нотации

Scenario: Конкретная бизнес ситуация, включающая в себя детальное описание, следующие 5 строк описывают действия пользователя, которые соответствуют конкретному коду. Строки, которые следуют за ключевыми словами “When”, “Then” и т.д., сравниваются.

Исходный код тест кейсов в нотации Gherkin представлен в приложении А.

* 1. Сквозное тестирование с использованием cypress и cucumber

Для e2e тестирования использовался фреймворк Cypress. Для обращения к одним и тем же элементам страницы использовался паттерн Page Object. Код теста добавления страницы представлен на рисунке 4 и 5. Результат успешного выполнения всех тестов представлен на рисунке 5.

Исходный код е2е тестирования с помощью cypress и cucumber представлен в приложении Б.

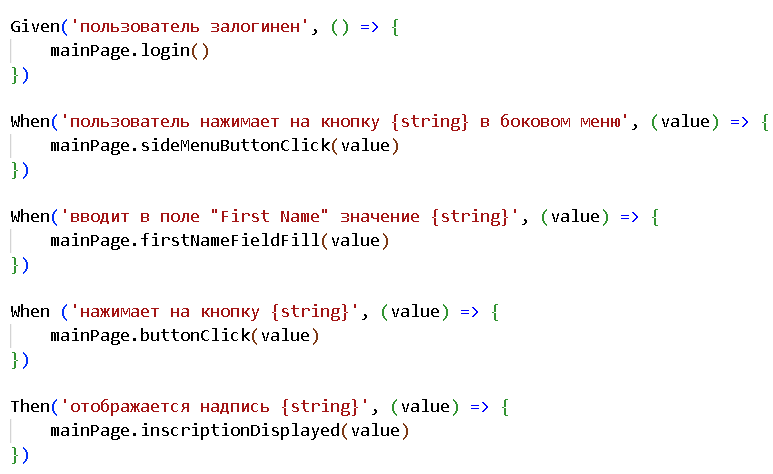


Рисунок 3 – Пример теста с использованием Cypress



Рисунок 4 – Пример использования паттерна Page Object

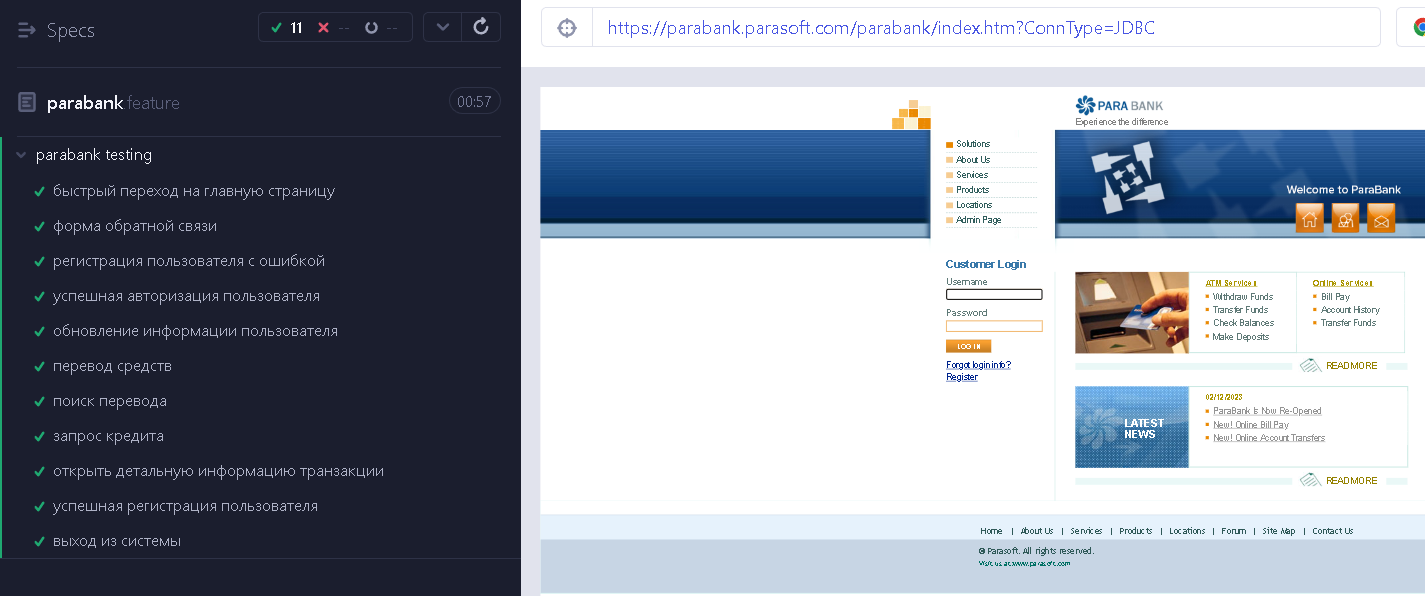


Рисунок 5 – результат успешного выполнения всех тестов

**Заключение**

**В ходе выполнения курсового проектирования были изучены основные принципы сквозного тестирования программного обеспечения, были закреплены полученные навыки тестирования.**

**В первой главе приведены основные определения предметной области, рассмотрены особенности тестирования веб-приложений, а также основные уровни тестирования.**

**Во второй главе приведены описание функционала тестируемого сайта** ParaBank**.**

**В третьей главе продемонстрированы описание тест кейсов в Gherkin нотации и результаты тестирования.**

**Список использованных источников**

1. Тестирование. Фундаментальная теория [Электронный ресурс] // habr – URL: https://habr.com/ru/post/279535/ (дата обращения: 04.12.2022).
2. Parasoft Demo Website [Электронный ресурс]// URL: https://parabank.parasoft.com/parabank/about.htm (дата обращения: 04.12.2022)
3. Тестирование программного обеспечения [Электронный ресурс]. – URL: https://ru.wikipedia.org/wiki/Тестирование\_программного  
   \_обеспечения (дата обращения 04.12.2022).
4. Канер, Сэм. Тестирование программного обеспечения. Фундаментальные концепции менеджмента бизнес-приложений [Текст]: пер. с англ./ Сэм Канер, Джек Фолк, Енг Кек Нгуен. – К.: Издательство «ДиаСофт», 2001. – 544 с.
5. Особенности тестирования web-приложений [Электронный ресурс] // QA evolution URL: https://qaevolution.ru/osobennosti-testirovaniya-web-prilozhenij/ (дата обращения 04.12.2022).
6. Пирамида тестов (тестирования) [Электронный ресурс] // <URL:https://www.qastart.by/mainterms/64-piramida-testov-testirovaniya> (дата обращения 14.12.2022).
7. End-to-end или E2E-процесс: что это? Сквозное тестирование [Электронный ресурс]. URL: https://codernet.ru/articles/drugoe/end-to-end\_ili\_e2e-proczess\_chto\_eto\_skvoznoe\_testirovanie/ (дата обращения: 04.11.2021)
8. Автоматизация тестирования: почему стоит попробовать Cypress [Электронный ресурс] URL: <https://blog.noveogroup.ru/2020/05/avtomatizatsiya-testirovaniya-cypress/> (дата обращения: 04.12.2022).
9. Cucumber (software) – Wikipedia [Электронный ресурс]. URL: <https://en.wikipedia.org/wiki/Cucumber_(software)> (дата обращения: 04.12.2022).

Приложение А  
Исходной код тест кейсов в Gherkin нотации

Feature: parabank testing

Scenario: быстрый переход на главную страницу

Given открыта страница about

When пользователь нажимает на икноку дома в шапке страницы

Then отображается главная страница

Scenario: форма обратной связи

Given открыта главная страница

When пользователь нажимает на икноку письма в шапке страницы

When заполняет поля обратной связи

When нажимает на кнопку "Send to Customer Care"

Then отображается надпись "A Customer Care Representative will be contacting you."

Scenario: регистрация пользователя с ошибкой

Given открыта главная страница

When пользователь нажимает на кнопку "Register"

When заполняет все поля кроме "Username"

When нажимает на кнопку "Register"

Then отображается ошибка "Username is required"

Scenario: успешная авторизация пользователя

Given открыта главная страница

When пользователь заполняет поля авторизации верными данными

When нажимает на кнопку "Log In"

Then отображается страница overview

Then отображается надпись "Accounts Overview"

Scenario: обновление информации пользователя

Given пользователь залогинен

When пользователь нажимает на кнопку "Update Contact Info" в боковом меню

When вводит в поле "First Name" значение "usernamee"

When нажимает на кнопку "Update Profile"

Then отображается надпись "Profile Updated"

Scenario: перевод средств

Given пользователь залогинен

When пользователь нажимает на кнопку "Transfer Funds" в боковом меню

When вводит в поле "Amount" значение "100"

When нажимает на кнопку "Transfer"

Then отображается надпись "Transfer Complete!"

Scenario: поиск перевода

Given пользователь залогинен

When пользователь нажимает на кнопку "Find Transactions" в боковом меню

When вводит в поле "Find by Amount" значение "100"

When нажимает на кнопку "Find transactions" поля "Amount"

Then в таблице отображаются записи

Scenario: запрос кредита

Given пользователь залогинен

When пользователь нажимает на кнопку "Request Loan" в боковом меню

When вводит в поле "Loan Amount" значение "100"

When вводит в поле "Down Payment" значение "10"

When нажимает на кнопку "Apply Now"

Then отображается надпись "Loan Request Processed"

Scenario: открыть детальную информацию транзакции

Given пользователь залогинен

When пользователь нажимает на кнопку "Accounts Overview" в боковом меню

When нажимает на ячейку в столбце "Account" таблицы пользователей

When нажимает на ячейку в столбце "Transaction" таблицы транзакций

Then отображается надпись "Transaction Details"

Scenario: успешная регистрация пользователя

Given открыта главная страница

When пользователь нажимает на кнопку "Register"

When заполняет поля регистрации

When нажимает на кнопку "Register"

Then отображается надпись "Your account was created successfully. You are now logged in."

Scenario: выход из системы

Given пользователь залогинен

When пользователь нажимает на кнопку "Log Out" в боковом меню

Then отображается главная страница

Then отображаются поля авторизации

Приложение Б  
Исходной код 2е2 тестирования сайта ParaBank

import { Given, Then, When } from '@badeball/cypress-cucumber-preprocessor';

import  mainPage  from "../../page-objects/MainPage";

Given('открыта главная страница', () => {

    mainPage.visitMainPage()

})

When ('нажимает на кнопку {string}', (value) => {

    mainPage.buttonClick(value)

})

Given('открыта страница about', () => {

    mainPage.visitAboutPage()

})

When('пользователь нажимает на икноку дома в шапке страницы', () => {

    mainPage.mainPageIconClick()

})

Then('отображается главная страница', () => {

    mainPage.checkUrl('index')

})

When('пользователь нажимает на икноку письма в шапке страницы', () => {

    mainPage.contactIconClick()

})

When('заполняет поля обратной связи', () => {

    mainPage.contactFieldFill()

})

Then('отображается надпись {string}', (value) => {

    mainPage.inscriptionDisplayed(value)

})

When('пользователь нажимает на кнопку "Register"', () => {

    mainPage.registerButtonClick()

})

When('заполняет все поля кроме "Username"', () => {

    mainPage.registerFieldFillWithoutUsername()

})

Then('отображается ошибка "Username is required"', () => {

    mainPage.registerErrorCheck()

})

When('заполняет поля регистрации', () => {

    mainPage.registerFieldFillWithoutUsername()

    mainPage.registerUsernameFieldFill()

})

When('пользователь заполняет поля авторизации верными данными', () => {

    mainPage.typeLoginUsername('Username4')

    mainPage.typeLoginPassword('Password')

})

Then('отображается страница overview', () => {

    mainPage.checkUrl('overview')

})

Given('пользователь залогинен', () => {

    mainPage.login()

})

When('пользователь нажимает на кнопку {string} в боковом меню', (value) => {

    mainPage.sideMenuButtonClick(value)

})

When('вводит в поле "Amount" значение "100"', () => {

    mainPage.amountFieldFill(100)

})

When('вводит в поле "Find by Amount" значение "100"', () => {

    mainPage.findByTransactionFieldFill()

})

When('нажимает на кнопку "Find transactions" поля "Amount"', () => {

    mainPage.findByTransactionButtonClick()

})

Then('в таблице отображаются записи', () => {

    mainPage.containsTableRows()

})

When('вводит в поле "Loan Amount" значение "100"', () => {

    mainPage.loanAmountFieldFill(100)

})

When('вводит в поле "Down Payment" значение "10"', () => {

    mainPage.downAmountFieldFill(10)

})

When('нажимает на ячейку в столбце "Account" таблицы пользователей', () => {

    mainPage.firstTableCellClick()

})

When('нажимает на ячейку в столбце "Transaction" таблицы транзакций', () => {

    mainPage.firstTableCellClick()

})

When('вводит в поле "First Name" значение {string}', (value) => {

    mainPage.firstNameFieldFill(value)

})

Then('отображаются поля авторизации', () => {

    mainPage.checkLogOut()

})

export class MainPage {

    visitMainPage(){

        cy.visit('/index.htm')

    }

    typeLoginUsername(value){

        cy.get('input[name="username"]').type(value)

    }

    typeLoginPassword(value){

        cy.get('input[name="password"]').type(value)

    }

    buttonClick(value){

        cy.get("input[value=\""+value+"\"]").click()

    }

    checkLoginError(){

        cy.get('.error').contains('The username and password could not be verified.')

    }

    visitAboutPage(){

        cy.visit('/about.htm')

    }

    mainPageIconClick(){

        cy.get('a').contains('home').click()

    }

    checkUrl(value){

        cy.url().should('contain',value)

    }

    contactIconClick(){

        cy.get('a').contains('contact').click()

    }

    contactFieldFill(){

        cy.get('#name').type('name')

        cy.get('#email').type('email')

        cy.get('#phone').type('phone')

        cy.get('#message').type('message')

    }

    inscriptionDisplayed(value){

        cy.contains(value)

    }

    registerButtonClick(){

        cy.get('a').contains('Register').click()

    }

    registerFieldFillWithoutUsername(){

        cy.get('input[id="customer.firstName"]').type('First Name')

        cy.get('input[id="customer.lastName"]').type('Last Name')

        cy.get('input[id="customer.address.street"]').type('Address')

        cy.get('input[id="customer.address.city"]').type('City')

        cy.get('input[id="customer.address.state"]').type('State')

        cy.get('input[id="customer.address.zipCode"]').type('Zip Code')

        cy.get('input[id="customer.phoneNumber"]').type('Phone')

        cy.get('input[id="customer.ssn"]').type('SNN')

        cy.get('input[id="customer.password"]').type('Password')

        cy.get('input[id="repeatedPassword"]').type('Password')

    }

    registerErrorCheck(){

        cy.get('span[id="customer.username.errors"]').contains('Username is required.')

    }

    registerUsernameFieldFill(){

        cy.get('input[id="customer.username"]').type('Username6')

    }

    login(){

        cy.visit('/index.htm')

        cy.get('input[name="username"]').type('Username4')

        cy.get('input[name="password"]').type('Password')

        cy.get('input[value="Log In"]').click()

    }

    sideMenuButtonClick(value){

        cy.get('li').contains(value).click()

    }

    amountFieldFill(value){

        cy.wait(1500)

        cy.get('#amount').type(value)

    }

    findByTransactionFieldFill(value){

        cy.get('input[id="criteria.transactionId"]').type('14476')

    }

    findByTransactionButtonClick(){

        cy.get('button[type="submit"]').eq(0).click()

    }

    containsTableRows(){

        cy.get('tr').contains('Funds Transfer Sent')

    }

    loanAmountFieldFill(value){

        cy.get('#amount').type(value)

    }

    downAmountFieldFill(value){

        cy.get('#downPayment').type(value)

    }

    firstTableCellClick(){

        cy.get('a[class="ng-binding"]').first().click()

    }

    firstNameFieldFill(value){

        cy.wait(1500)

        cy.get('input[id="customer.firstName"]').clear().type(value)

    }

    checkLogOut(){

        cy.get('input[name="username"]').should('exist')

    }

}

export default new MainPage();