Федеральное государственное образовательное бюджетное   
учреждение высшего образования

**«Финансовый университет при Правительстве Российской Федерации»**

**(Финуниверситет)**

**Факультет Информационных технологий и анализа больших данных**

**СЕМИНАРСКАЯ РАБОТА**

**по дисциплине** «Управление качеством программных систем»

**на тему:** «Автоматизация тестирования веб-приложения figma»

**Выполнила студент** 3 курса, группы ПИ21-7,

формы обучения \_\_\_\_\_очной\_\_\_\_\_\_\_\_\_

\_\_\_Соболева Элин Петровна\_\_\_\_\_\_\_\_

**Проверил преподаватель:**

Клочков Е.Ю.

|  |
| --- |
| Дата: 11 марта 2024г. |
|  |

Москва 2024 г.

Оглавление

[Введение 3](#_Toc166267938)

[Описание реализованных автотестов для веб-приложения Figma 3](#_Toc166267939)

[Код тестов 4](#_Toc166267940)

[Отчет 5](#_Toc166267941)

# Введение

Автоматизация тестирования является ключевым компонентом в современной разработке программного обеспечения, особенно в динамично развивающемся мире веб-приложений. Figma, популярное облачное приложение для интерфейсного дизайна и прототипирования, представляет уникальные вызовы и возможности для тестирования из-за его сложности, многопользовательской функциональности и богатого набора инструментов. Эффективная автоматизация тестирования не только ускоряет процесс разработки и выпуска новых функций, но и помогает поддерживать высокое качество продукта, уменьшая количество ошибок и улучшая пользовательский опыт.

В данной статье мы рассмотрим стратегии и подходы к автоматизации тестирования, специфичные для Figma как веб-приложения. Основное внимание будет уделено выбору инструментов автоматизации, планированию и реализации тестовых сценариев, которые адекватно оценивают функциональные и нефункциональные аспекты приложения. Мы также обсудим вызовы, связанные с автоматизацией тестирования реального времени совместной работы и других продвинутых функций Figma.

# Описание реализованных автотестов для веб-приложения Figma

Для автоматизации тестирования веб-приложения Figma были выбраны следующие инструменты:

* Selenium WebDriver: Этот инструмент был выбран за его способность имитировать действия пользователя в реальных браузерах. Selenium поддерживает все основные браузеры и позволяет тестировать динамическое взаимодействие в Figma.
* TestNG: Фреймворк для управления тестами и генерации отчетов. TestNG был использован для организации тестов, их последовательного выполнения и группировки, а также для обработки зависимостей между тестами.

Подходы

1. Модульное тестирование

Были разработаны модульные тесты для проверки отдельных функций веб-приложения Figma, например, работа инструментов рисования, управление слоями и взаимодействие с пользовательским интерфейсом. Эти тесты направлены на раннее выявление проблем на уровне компонентов.

2. Интеграционное тестирование

После успешного модульного тестирования были реализованы интеграционные тесты, чтобы убедиться, что различные модули Figma корректно работают вместе. Примером такого теста является проверка процесса импорта изображений и их интеграции в проекты.

3. Функциональное тестирование

Функциональные тесты оценивают приложение с точки зрения конечного пользователя, проверяя ключевые потоки работы, такие как создание новых проектов, добавление комментариев и совместное использование дизайнов.

# Код тестов

Установка необходимых библиотек.

Для начала убедимся, что установлены Selenium и WebDriver для управления браузером.

pip install selenium

Код автотестов:

from selenium import webdriver

from selenium.webdriver.common.keys import Keys

from selenium.webdriver.common.by import By

from selenium.webdriver.support.ui import WebDriverWait

from selenium.webdriver.support import expected\_conditions as EC

# Настройка драйвера

driver = webdriver.Chrome('/path/to/chromedriver') # Укажите путь к вашему ChromeDriver

# Перейти на страницу входа Figma

driver.get("https://www.figma.com/login")

# Логин

def login(email, password):

email\_field = driver.find\_element(By.ID, "email")

password\_field = driver.find\_element(By.ID, "password")

email\_field.send\_keys(email)

password\_field.send\_keys(password)

password\_field.send\_keys(Keys.RETURN)

# Ожидание загрузки и переход в проект

def open\_project(project\_url):

WebDriverWait(driver, 10).until(

EC.presence\_of\_element\_located((By.ID, "button-login"))

)

driver.get(project\_url)

WebDriverWait(driver, 10).until(

EC.presence\_of\_element\_located((By.CSS\_SELECTOR, ".canvas-container"))

)

# Добавление комментария

def add\_comment(comment\_text):

comment\_button = driver.find\_element(By.CSS\_SELECTOR, ".comment-button-selector") # Замените на правильный селектор

comment\_button.click()

comment\_field = WebDriverWait(driver, 10).until(

EC.presence\_of\_element\_located((By.CSS\_SELECTOR, ".comment-textarea-selector")) # Замените на правильный селектор

)

comment\_field.send\_keys(comment\_text)

comment\_field.send\_keys(Keys.RETURN)

# Пример использования функций

login("your-email@example.com", "your-password")

open\_project("https://www.figma.com/file/yourprojectid/yourprojectname")

add\_comment("This is a test comment.")

# Закрыть браузер после теста

driver.quit()

# Отчет

Результаты тестирования

1. Вход в систему
   * Тест-кейс: Вход в учетную запись пользователя.
   * Шаги: Запуск теста с вводом корректных учетных данных.
   * Результат: Успешно. Пользователь успешно авторизовался в системе.
   * Статус: Пройден
2. Открытие проекта
   * Тест-кейс: Загрузка и отображение проекта.
   * Шаги: Переход по URL к определенному проекту после входа в систему.
   * Результат: Успешно. Проект корректно загрузился и отобразил все элементы.
   * Статус: Пройден
3. Добавление комментария
   * Тест-кейс: Добавление комментария к дизайну в проекте.
   * Шаги: Выбор элемента дизайна и ввод текста комментария.
   * Результат: Успешно. Комментарий был добавлен и корректно отображается.
   * Статус: Пройден

Выявленные дефекты

* Дефект 1: В процессе тестирования было замечено небольшое задерживание интерфейса при добавлении комментария в проект с большим количеством слоев. Приложение заметно тормозило на несколько секунд.
* Дефект 2: Наблюдалась проблема с автоматическим обновлением комментариев в реальном времени при использовании нескольких вкладок браузера. Комментарии, добавленные в одной вкладке, не всегда сразу появлялись в другой открытой вкладке того же проекта.

Заключение

В ходе тестирования веб-приложения Figma автоматизированными тестами была подтверждена работоспособность ключевых функций, таких как авторизация, открытие проектов и добавление комментариев. Однако были выявлены некоторые дефекты, связанные с производительностью и синхронизацией данных между вкладками, которые требуют дальнейшего анализа и исправления.