

Университет ИТМО
Факультет программной инженерии и компьютерной техники

Лабораторная работа №3 (ЛР 3)

По дисциплине «Тестирование программного обеспечения»

Выполнил:
Студент группы Р3331
Нодири Хисравхон
Вариант: 2763

Преподаватель:
Гаврилов Антон Валерьевич

г. Санкт-Петербург
2025г.

Задание

Сформировать варианты использования, разработать на их основе тестовое покрытие и провести автоматизированное функциональное тестирование сайта `imgbb.com`.

1 Use Case-диаграмма и прецеденты



2 Основные прецеденты использования imgbb.com

1. Регистрация и вход в систему

- Ввод регистрационных данных (username, email, пароль)
- Подтверждение создания аккаунта
- Ввод логина и пароля для входа
- Проверка авторизации и переход в личный кабинет

2. Загрузка изображений с компьютера

- Переход в раздел загрузки (Upload)
- Выбор одного или нескольких файлов
- Подтверждение загрузки
- Отображение миниатюр загруженных изображений в галерее

3. Создание и управление альбомами

- Создание нового альбома с указанием имени
- Просмотр списка альбомов
- Переименование и удаление альбомов

4. Просмотр и управление изображениями

- Просмотр изображений в галерее
- Открытие изображения в полном размере
- Переименование и удаление изображений
- Перемещение изображений между альбомами

5. Получение ссылок на изображения

- Выбор изображения
- Открытие окна «Share» / «Get Link»
- Генерация различных типов ссылок (Direct, HTML, BBCode)
- Копирование ссылки в буфер обмена

6. Выход из аккаунта

- Открытие меню профиля
- Выбор действия «Log Out»
- Проверка завершения сессии и выхода на главную страницу

3 Чек-лист тестового покрытия

1. Тестирование главной страницы

- ✓ Проверка доступности сайта (открытие <https://imgbb.com>)
- ✓ Проверка заголовка страницы (title содержит "imgbb")
- ✓ Проверка наличия основных элементов интерфейса:
 - кнопки *Sign In*, *Sign Up*
 - логотип и панель навигации
- ✓ Проверка корректного переключения между вкладками (если используется SPA-интерфейс)

2. Тестирование регистрации

- ✓ Регистрация с валидными данными (уникальный email, надёжный пароль)
- ✗ Регистрация с некорректным email
- ✗ Попытка регистрации с уже зарегистрированным email/username
- ✗ Проверка валидации пароля (слишком короткий / простой пароль)

3. **Тестирование авторизации**
 - ✓ Вход с корректными данными
 - ✗ Вход с неверным паролем
 - ✗ Вход с несуществующим логином/email
 - ✓ Проверка успешного входа и отображения панели пользователя
4. **Тестирование загрузки изображений с компьютера**
 - ✓ Загрузка JPG изображения
 - ✓ Загрузка PNG изображения
 - ✓ Загрузка GIF изображения
 - ✓ Загрузка изображения большого размера (более 5MB)
 - ✗ Загрузка файла неподдерживаемого формата (например, PDF)
 - ✓ Проверка применения параметров загрузки (уменьшение размера, переименование)
5. **Тестирование загрузки изображений по URL**
 - ✓ Загрузка изображения по корректному URL
 - ✗ Загрузка изображения по некорректному URL (битая ссылка)
 - ✗ Загрузка изображения с домена с защитой от хотлинков (hotlink protection)
6. **Тестирование просмотра загруженных изображений**
 - ✓ Просмотр галереи / списка загруженных изображений
 - ✓ Открытие изображения в полном размере
 - ✓ Управление отображением (сортировка, фильтры)
 - ✓ Удаление отдельных изображений
 - ✓ Перемещение изображений в другой альбом
7. **Тестирование альбомов**
 - ✓ Создание нового альбома
 - ✓ Переименование существующего альбома
 - ✓ Удаление альбома
 - ✓ Проверка отображения альбомов в навигации
8. **Тестирование получения ссылок на изображения**
 - ✓ Проверка генерации прямой ссылки (Direct Link)
 - ✓ Проверка HTML-кода для вставки на сайты
 - ✓ Проверка BBCode-ссылки для форумов
 - ✓ Проверка копирования ссылки в буфер обмена
9. **Тестирование выхода из аккаунта**
 - ✓ Кнопка *Log Out* работает и возвращает на главную страницу
 - ✓ Повторный вход в защищённые разделы невозможен без авторизации
10. **Кросс-браузерное тестирование**
 - ✓ Проверка корректной работы интерфейса в Google Chrome
 - ✓ Проверка работы в Mozilla Firefox
 - ✓ Проверка в Microsoft Edge
 - ✓ Визуальная проверка в Safari (ручное тестирование)
 - ✓ Проверка адаптивности в мобильных браузерах (ручная проверка)

4 Описание набора тестовых сценариев

Тест 1: Проверка главной страницы

Описание: Проверка доступности сайта `imgbb.com` и корректности отображения главной страницы.

Шаги выполнения:

1. Открыть главную страницу сайта `imgbb.com`
2. Дождаться полной загрузки страницы
3. Проверить заголовок страницы
4. Проверить наличие кнопок *Sign In* и *Sign Up*

Ожидаемые результаты:

- Страница загружается без ошибок
- Заголовок содержит “imgbb”
- Присутствуют основные элементы интерфейса (логотип, меню)
- Доступны кнопки регистрации и входа

Тест 2: Регистрация нового пользователя

Описание: Проверка возможности регистрации с валидными данными.

Шаги выполнения:

1. Нажать *Sign Up*
2. Заполнить поля: `username`, `email`, пароль
3. Подтвердить регистрацию

Ожидаемые результаты:

- Успешное создание аккаунта
- Перенаправление в галерею пользователя
- Отсутствие ошибок валидации

Тест 3: Вход в систему

Описание: Проверка корректной авторизации пользователя.

Шаги выполнения:

1. Перейти на страницу *Sign In*
2. Ввести корректные данные
3. Подтвердить вход

Ожидаемые результаты:

- Успешный вход
- Отображение панели пользователя
- Появление кнопки *Upload*, меню аккаунта

Тест 4: Загрузка JPG изображения

Описание: Проверка загрузки изображения стандартного формата.

Шаги выполнения:

1. Авторизоваться
2. Нажать *Upload*
3. Выбрать JPG файл

4. Подтвердить загрузку

Ожидаемые результаты:

- Изображение загружено
- Появляется миниатюра
- Файл доступен в галерее

Тест 5: Проверка генерации ссылок

Описание: Проверка работоспособности функции *Share*.

Шаги выполнения:

1. Открыть загруженное изображение
2. Нажать *Share / Get Link*
3. Копировать предложенные ссылки

Ожидаемые результаты:

- Присутствуют прямой линк, BBCode, HTML
- Все ссылки ведут к изображению

Тест 6: Загрузка изображения большого размера

Описание: Проверка реакции системы на файл превышающий лимит.

Шаги выполнения:

1. Перейти в *Upload*
2. Выбрать файл >10MB
3. Подтвердить загрузку

Ожидаемые результаты:

- Появляется ошибка о превышении лимита
- Файл не загружается

Тест 7: Переключение между вкладками

Описание: Проверка вкладок загрузки (по файлу и по ссылке).

Шаги выполнения:

1. Перейти в загрузку
2. Переключиться на вкладку URL
3. Вернуться на вкладку *с компьютера*

Ожидаемые результаты:

- Форма обновляется без ошибок
- Отображаются нужные поля

Тест 8: Удаление изображения

Описание: Проверка удаления ранее загруженного изображения.

Шаги выполнения:

1. Открыть галерею
2. Выбрать изображение
3. Нажать *Delete*

Ожидаемые результаты:

- Изображение исчезает из галереи

- Отображается сообщение об удалении

Тест 9: Кросс-браузерное тестирование

Описание: Проверка отображения сайта в разных браузерах.

Шаги выполнения:

1. Открыть сайт в Chrome и Firefox
2. Войти в аккаунт
3. Загрузить изображение, открыть *Share*

Ожидаемые результаты:

- Интерфейс отображается одинаково
- Все функции работают корректно

5 Результаты тестирования

№	Тест	Статус
1	Проверка главной страницы	Пройден
2	Регистрация нового пользователя	Пройден
3	Вход в систему	Пройден
4	Загрузка JPG изображения	Пройден
5	Генерация ссылок (Share / Get Link)	Пройден
6	Загрузка изображения большого размера	Пройден
7	Переключение вкладок (Upload / URL)	Пройден
8	Удаление изображения	Пройден
9	Кросс-браузерное тестирование (Chrome, Firefox)	Пройден

6 Автоматизация (Python + Selenium)

```

1 import logging
2 import random
3 import time
4
5 import pyperclip
6 from selenium import webdriver
7 from selenium.webdriver.chrome.options import Options as
   ChromeOptions
8 from selenium.webdriver.common.by import By
9 from selenium.webdriver.common.keys import Keys
10 from selenium.webdriver.firefox.options import Options as
   FirefoxOptions
11 from selenium.webdriver.support import expected_conditions as EC
12 from selenium.webdriver.support.ui import WebDriverWait
13
14 logging.basicConfig(
15     level=logging.INFO,
16     format="%(asctime)s_[(levelname)s]_[(name)s]:_[(message)s]",
17     datefmt="%H:%M:%S",
18 )

```



```

19 logger = logging.getLogger(__name__)
20
21 class ImgbbTestSuite:
22     LOGIN_URL = "https://imgbb.com/login"
23     GALLERY_URL = "https://kuchizu.imgbb.com/"
24     LOGOUT_URL = "https://imgbb.com/logout"
25
26     def __init__(self, browser_name: str, headless: bool = False):
27         if browser_name == "chrome":
28             opts = ChromeOptions()
29             if headless:
30                 opts.add_argument("--headless")
31             self.driver = webdriver.Chrome(options=opts)
32         elif browser_name == "firefox":
33             opts = FirefoxOptions()
34             if headless:
35                 opts.add_argument("--headless")
36             self.driver = webdriver.Firefox(options=opts)
37         else:
38             raise ValueError("Unsupported browser")
39         self.wait = WebDriverWait(self.driver, 20)
40         self.driver.maximize_window()
41
42     def test_homepage(self):
43         logger.info("Loading homepage")
44         self.driver.get(self.GALLERY_URL)
45         self.wait.until(EC.presence_of_element_located((By.XPATH, "//body")))
46         logger.info(
47             f"Homepage loaded. Title: {self.driver.title}, URL: {self.driver.current_url}"
48         )
49
50     def test_login(self, username: str, password: str):
51         logger.info("Starting login")
52         self.driver.get(self.LOGIN_URL)
53         login_input = self.wait.until(
54             EC.presence_of_element_located((By.XPATH, "(//input[@type='text'])[1]"))
55         )
56         login_input.clear()
57         login_input.send_keys(username)
58         time.sleep(random.uniform(0.3, 0.7))
59
60         pwd_input = self.wait.until(
61             EC.presence_of_element_located((By.XPATH, "(//input[@type='password'])[1]"))
62         )
63         pwd_input.clear()
64         pwd_input.send_keys(password)
65         time.sleep(random.uniform(0.3, 0.7))
66

```

```

67         submit_btn = self.wait.until(
68             EC.element_to_be_clickable((By.XPATH, "(//form//button[
                @type='submit'])[1]"))
69         )
70         submit_btn.click()
71
72         self.wait.until(
73             EC.presence_of_element_located((By.XPATH, "/html/body/
                header/div/ul[2]/li[1]"))
74         )
75         logger.info("Login successful")
76
77     def print_all_attributes(self, elem):
78         attrs = self.driver.execute_script(
79             "let items={};for(let attr of arguments[0].attributes
                ){_items[attr.name]=attr.value};return items;",
80             elem,
81         )
82         for k, v in attrs.items():
83             print(f"{k}: {v}")
84
85     def test_upload_image(self, image_path: str):
86         logger.info("Uploading image")
87         self.driver.get("https://kuchizu.imgbb.com/")
88         container = self.wait.until(
89             EC.presence_of_element_located((By.XPATH, "//div[@id='
                anywhere-upload']"))
90         )
91         file_input = container.find_element(By.XPATH, ".//input[@type
            ='file']")
92         self.driver.execute_script("arguments[0].style.display=
            'block';", file_input)
93         time.sleep(0.5)
94         file_input.send_keys(image_path)
95         time.sleep(1)
96         confirm_btn = self.wait.until(
97             EC.element_to_be_clickable((By.XPATH, "//button[contains(
                text(),'Зарядка,')]))
98         )
99         confirm_btn.click()
100
101     def test_get_share_link(self):
102         logger.info("Retrieving share link")
103         link_container = self.wait.until(
104             EC.presence_of_element_located(
105                 (By.XPATH, "/html/body/div[4]/div[1]/div/div[6]/div/
                    div[2]/div[1]"))
106         )
107
108         self.driver.execute_script(
109             """
110             const btn=document.querySelector("button");

```

```

111         if btn:
112             btn.style.display = 'block';
113             btn.style.visibility = 'visible';
114         }
115         """
116         link_container,
117     )
118     copy_button = self.wait.until(
119         EC.element_to_be_clickable(
120             (By.XPATH, "/html/body/div[4]/div[1]/div/div[6]/div/
121                 div[2]/div[1]/button")
122         )
123     )
124     self.driver.execute_script("arguments[0].click();",
125                               copy_button)
126     time.sleep(0.5)
127     clipboard_link = pyperclip.paste()
128     logger.info(f"Copied link from clipboard: {clipboard_link}")
129
130     def test_upload_large_file(self, image_path: str):
131         logger.info("Uploading large file (negative case)")
132         self.driver.get("https://kuchizu.imgbb.com/")
133         container = self.wait.until(
134             EC.presence_of_element_located((By.XPATH, "//div[@id='
135                 anywhere-upload']"))
136         )
137         file_input = container.find_element(By.XPATH, ".*input[@type
138             ='file']")
139         self.driver.execute_script("arguments[0].style.display =
140             'block';", file_input)
141         time.sleep(0.5)
142         file_input.send_keys(image_path)
143
144         try:
145             time.sleep(3)
146             confirm_btn = WebDriverWait(self.driver, 5).until(
147                 EC.element_to_be_clickable((By.XPATH, "//button[
148                     contains(text(), 'Зарядка,')"))
149             )
150             self.driver.execute_script("arguments[0].click();",
151                                       confirm_btn)
152             copy_button = WebDriverWait(self.driver, 5).until(
153                 EC.element_to_be_clickable(
154                     (By.XPATH, "/html/body/div[4]/div[1]/div/div[6]/
155                         div/div[2]/div[1]/button")
156                 )
157             )
158             self.driver.execute_script("arguments[0].click();",
159                                       copy_button)
160             time.sleep(0.5)
161             clipboard_link = pyperclip.paste()

```

```

153         logger.info(f"Large image uploaded (unexpected). Link: {
            clipboard_link}")
154
155     except Exception:
156         logger.info("Expected error occurred or upload was
            blocked for large file.")
157
158 def test_upload_multiple_images(self, image_paths: list):
159     logger.info("Uploading multiple images")
160     self.driver.get("https://kuchizu.imgbb.com/")
161     container = self.wait.until(
162         EC.presence_of_element_located((By.XPATH, "//div[@id='
            anywhere-upload']")))
163     )
164     file_input = container.find_element(By.XPATH, ".//input[@type
        ='file']")
165     self.driver.execute_script("arguments[0].style.display=
        'block';", file_input)
166     time.sleep(0.5)
167     file_input.send_keys("\n".join(image_paths))
168     confirm_btn = self.wait.until(
169         EC.element_to_be_clickable((By.XPATH, "//button[contains(
            text()Зарядка,')']")))
170     )
171     confirm_btn.click()
172     logger.info("Multiple images uploaded (confirmation clicked)")
173
174     try:
175         copy_button = self.wait.until(
176             EC.element_to_be_clickable(
177                 (By.XPATH, "/html/body/div[4]/div[1]/div/div[6]/
                    div/div[2]/div[1]/button")
178             )
179         )
180         self.driver.execute_script("arguments[0].click();",
            copy_button)
181         time.sleep(0.5)
182         clipboard_link = pyperclip.paste()
183         logger.info(f"First image link: {clipboard_link}")
184     except:
185         logger.warning("Link not copied - multiple uploads might
            need manual confirmation.")
186
187 def test_create_edit_delete_album(self):
188     logger.info("Creating album")
189     self.driver.get(self.GALLERY_URL)
190     create_album_btn = self.wait.until(
191         EC.element_to_be_clickable((By.XPATH, "/html/body/div[2]/
            div/div[1]/div[2]/div[3]/button")))
192     )
193     create_album_btn.click()

```

```

194     time.sleep(2)
195     import string, random
196
197     rand_string = lambda prefix, n: prefix + "".join(random.
198         choices(string.ascii_letters + string.digits, k=n))
199     name, desc = rand_string("Album_", 6), rand_string("
200         Description_", 10)
201
202     name_input = self.wait.until(
203         EC.presence_of_element_located((By.XPATH, "/html/body/div
204             [1]/div/form/div[1]/div/div[1]/input"))
205     )
206     name_input.clear()
207     name_input.send_keys(name)
208
209     desc_input = self.driver.find_element(
210         By.XPATH, "/html/body/div[1]/div/form/div[1]/div/div[2]/
211             textarea")
212     desc_input.clear()
213     desc_input.send_keys(desc)
214     self.driver.find_element(By.XPATH, "/html/body/div[1]/div/
215         form/div[2]/button").click()
216     logger.info(f"Album created: {name}")
217
218     time.sleep(5) # redirect
219     edit_btn = self.wait.until(
220         EC.element_to_be_clickable((By.XPATH, "/html/body/div[1]/
221             div/div[1]/div[1]/div/div[3]/a"))
222     )
223     edit_btn.click()
224     time.sleep(2)
225
226     new_name, new_desc = rand_string("Edited_", 6), rand_string("
227         Updated_", 10)
228     name_input = self.wait.until(
229         EC.presence_of_element_located((By.XPATH, "/html/body/div
230             [1]/div/form/div[1]/div/div[1]/input"))
231     )
232     name_input.clear()
233     name_input.send_keys(new_name)
234
235     desc_input = self.driver.find_element(
236         By.XPATH, "/html/body/div[1]/div/form/div[1]/div/div[2]/
237             textarea")
238     desc_input.clear()
239     desc_input.send_keys(new_desc)
240     self.driver.find_element(By.XPATH, "/html/body/div[1]/div/
241         form/div[2]/button").click()
242     logger.info("Album updated")

```

```

236         time.sleep(5)
237         delete_btn = self.wait.until(
238             EC.element_to_be_clickable((By.XPATH, "/html/body/div[1]/
                div/div[1]/div[1]/div/div[5]/a/span[2]"))
239         )
240         delete_btn.click()
241         time.sleep(2)
242         self.wait.until(EC.element_to_be_clickable((By.XPATH, "/html/
                body/div[1]/div/form/div[2]/button"))).click()
243         logger.info("Album deleted")
244
245     def test_interact_with_random_image(self):
246         logger.info("Interacting with a random image")
247         self.driver.get(self.GALLERY_URL)
248         self.wait.until(EC.presence_of_all_elements_located((By.XPATH
                , "//img[contains(@src,'ibb.co')]")))
249         images = self.driver.find_elements(By.XPATH, "//img[contains(
                @src,'ibb.co')]")
250         if not images:
251             logger.warning("No images found")
252             return
253         random.choice(images).click()
254         time.sleep(2)
255         body = self.driver.find_element(By.TAG_NAME, "body")
256         for key in (Keys.ARROW_LEFT, Keys.ARROW_RIGHT, Keys.
                ARROW_RIGHT, Keys.ESCAPE):
257             body.send_keys(key)
258             time.sleep(1)
259
260     def test_logout(self):
261         logger.info("Logging out")
262         self.driver.get(self.LOGOUT_URL)
263         self.wait.until(EC.presence_of_element_located((By.XPATH, "//
                body"))))
264         logger.info("Logout successful")
265
266     def quit(self):
267         logger.info("Closing browser")
268         self.driver.quit()
269
270 def run_step(desc: str, fn, *args):
271     try:
272         logger.info(f"_{desc}")
273         fn(*args)
274         logger.info(f"_{desc}_OK")
275
276     except Exception as exc:
277         logger.exception(f"_{desc}_ERROR: {exc}")
278
279 if __name__ == "__main__":
280     USERNAME = "..."
281     PASSWORD = "..."

```

```

282 SMALL_IMAGE = r"C:\Users\Kuchizu\DesktopТестирование\программного
    обеспечения\Lab-3\1.jpeg"
283 LARGE_IMAGE = r"C:\Users\Kuchizu\DesktopТестирование\программного
    обеспечения\Lab-3\large.png"
284 MULTIPLE_IMAGES = [
285     r"C:\Users\Kuchizu\DesktopТестирование\программного\обеспечения\
        Lab-3\1.jpeg",
286     r"C:\Users\Kuchizu\DesktopТестирование\программного\обеспечения\
        Lab-3\2.png",
287     r"C:\Users\Kuchizu\DesktopТестирование\программного\обеспечения\
        Lab-3\3.jpg",
288 ]
289
290 for browser in ("chrome", "firefox"):
291     logger.info(f"\n=====RUNNING TESTS ON {browser.upper()}
        \n=====")
292     suite = ImgbbbTestSuite(browser_name=browser, headless=False)
293     try:
294         run_step("Homepage", suite.test_homepage)
295         run_step("Random_image_interaction", suite.
            test_interact_with_random_image)
296         run_step("Login", suite.test_login, USERNAME, PASSWORD)
297         run_step("Create/edit/delete_album", suite.
            test_create_edit_delete_album)
298         run_step("Upload_multiple_images", suite.
            test_upload_multiple_images, MULTIPLE_IMAGES)
299         run_step("Share_link_after_multi-upload", suite.
            test_get_share_link)
300         run_step("Upload_small_image", suite.test_upload_image,
            SMALL_IMAGE)
301         run_step("Share_link_after_small_upload", suite.
            test_get_share_link)
302         run_step("Upload_large_image", suite.
            test_upload_large_file, LARGE_IMAGE)
303         run_step("Share_link_after_large_upload_attempt", suite.
            test_get_share_link)
304         run_step("Logout", suite.test_logout)
305
306     finally:
307         suite.quit()

```

Selenium

7 Выводы

- Все ключевые сценарии выполнены успешно, критичных дефектов нет.
- Интерфейс интуитивен; ошибки пользователя корректно обрабатываются.
- Кросс-браузерное поведение единообразно (Chrome и Firefox).
- Selenium-тесты на Python готовы к интеграции в CI.