МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра МО ЭВМ

ОТЧЕТ

по «UI-тестирование» практике

Тема: Тестирование системы The-internet (Heroku)

Студенты гр. 3341 Перевалов П. И.
Рябов М. А.
Лодыгин И. А.

Руководитель _____ Шевелёва А.М.

Санкт-Петербург

2025

ЗАДАНИЕ НА «UI-ТЕСТИРОВАНИЕ» ПРАКТИКУ

Студенты

Перевалов П. И.

Рябов М. А.

Лодыгин И. А.

Группа 3341

Тема практики: UI-тестирование

Задание на практику:

Технологии: Java, Selenide (Selenium), Junit, Maven, логирование

Чеклист: подробное описание созданных тестов в одном файле:

название системы, которую тестируете

название тестов

входные данные (по возможности)

описание теста – то, как это бы делал пользователь.

Тесты:

Нужно написать 10 тестов по одному определенному блоку / функционалу системы. .

Примечание: была выбрана система the-internet.herokuapp.com

Сроки прохождения практики: 25.06.2025 – 08.07.2025

Дата сдачи отчета: 07.07.2025

Дата защиты отчета: 07.07.2025

Студенты		Перевалов П. И. Рябов М. А. Лодыгин И. А.
Руководитель		Шевелёва А.М.

АННОТАЦИЯ

В ходе практики проведено тестирование ключевых интерактивных элементов демонстрационного сайта The Internet (Heroku). Основное внимание уделено проверке следующих функций: базовая HTTP-авторизация, обработка физических нажатий клавиш, динамическое добавление и удаление элементов, работа с битым контентом, интерактивные реакции на наведение курсора и действия мыши, выпадающие списки, а также операции с файлами (загрузка и скачивание). Дополнительно протестированы числовой ввод и контекстные действия.
Критериями оценки выступили: корректность отображения интерфейсов, точность реакции на действия пользователя, сохранение введенных данных, наличие уведомлений и сообщений, соответствующих сценарию, а также корректная обработка ошибок и исключений.
Тестирование проводилось на специально подготовленных страницах ресурса, каждая из которых предназначена для проверки отдельного типа функциональности.

SUMMARY

During the practice, key interactive elements of the demo site The Internet (Heroku) were tested. The main focus was on checking the following functions: basic HTTP authorization, processing physical keystrokes, dynamic addition and deletion of elements, working with broken content, interactive reactions to hovering and mouse actions, drop-down lists, as well as file operations (uploading and downloading). Numeric input and contextual actions were also tested.

The evaluation criteria were: correctness of interface display, accuracy of response to user actions, saving of entered data, presence of notifications and messages corresponding to the scenario, as well as correct handling of errors and exceptions.

Testing was carried out on specially prepared pages of the resource, each of which is designed to check a separate type of functionality..

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Цель:

Проведение комплексного тестирования интерактивных элементов сайта The Internet (Heroku), направленного на демонстрацию типовых пользовательских сценариев, обработки ввода и реакций интерфейса.

Задачи:

1. Тестирование авторизации:
   - Проверка корректного прохождения базовой HTTP-авторизации при вводе валидных данных.
   - Анализ обработки ошибок при вводе некорректных логина и пароля.
2. Тестирование работы с пользовательским вводом:
   - Проверка отображения результатов при физическом нажатии клавиш на клавиатуре.
   - Валидация числового поля: проверка фильтрации недопустимых символов в инпуте типа number.
3. Тестирование динамических элементов:
   - Добавление и удаление элементов через интерфейс.
   - Проверка отображения и удаления объектов при действиях пользователя.
4. Тестирование отображения медиа-контента:
   - Проверка наличия битых изображений и корректной реакции интерфейса.
5. Тестирование взаимодействия с элементами UI:
   - Наведение курсора на изображения пользователей и отображение дополнительной информации.
   - Проверка работы контекстного меню при вызове правой кнопкой мыши.
   - Тестирование выпадающего списка: выбор и отображение выбранного пункта.
6. Тестирование операций с файлами:
   - Скачивание файлов: проверка загрузки по ссылке.
   - Загрузка файлов: проверка процесса выбора и загрузки файла, а также отображения успешного результата.

1. ОПИСАНИЕ РЕАЛИЗУЕМЫХ ТЕСТОВ

Описание тестов

1. Авторизация

1.1. Прохождение базовой авторизации:

Тест осуществляет переход на страницу авторизации (Basic Auth), вводит корректные учетные данные (admin / admin) и проверяет отображение сообщения об успешной авторизации.

1.2. Провал базовой авторизации:

Тест осуществляет переход на страницу авторизации, вводит некорректные данные (например, неправильный пароль) и проверяет отображение сообщения об ошибке или отклонение доступа.

2. Ввод с клавиатуры

2.1. Нажатие клавиши на клавиатуре:

Тест переходит на страницу Key Presses, нажимает любую клавишу на клавиатуре и проверяет корректное отображение названия нажатой клавиши в поле You entered:.

3. Работа с элементами

3.1. Добавление элемента:

Тест переходит на страницу Add/Remove Elements, нажимает кнопку “Add Element” и проверяет отображение новой кнопки “Delete” в списке элементов.

3.2. Удаление элемента:

После добавления элемента, тест нажимает кнопку “Delete” и проверяет, что элемент исчезает со страницы.

4. Проверка изображений

4.1. Отображение битых изображений:

Тест переходит на страницу Broken Images и проверяет, что на странице присутствуют не загруженные (битые) изображения (например, с иконкой ошибки загрузки).

5. Наведение курсора

5.1. Отображение информации при наведении:

Тест переходит на страницу Hovers, наводит курсор на одну из иконок пользователей и проверяет отображение имени пользователя и ссылки на его профиль.

6. Контекстное меню

6.1. Вызов контекстного меню:

Тест переходит на страницу Context Menu, нажимает правой кнопкой мыши на выделенную область и проверяет появление всплывающего push-уведомления с текстом You selected a context menu.

7. Выпадающий список

7.1. Выбор значения из выпадающего списка:

Тест переходит на страницу Dropdown, раскрывает список значений и выбирает один из доступных вариантов (Option 1 или Option 2). Затем проверяет, что выбранное значение корректно отображается как активный пункт.

8. Скачивание файла

8.1. Скачивание файла:

Тест переходит на страницу File Download, нажимает на ссылку для скачивания файла (например, image.png) и проверяет, что файл успешно загружается на устройство пользователя.

9. Загрузка файла

9.1. Загрузка текстового файла:

Тест переходит на страницу File Upload, выбирает файл test.txt с устройства и нажимает кнопку Upload. Проверяет отображение имени загруженного файла и наличие заголовка “File Uploaded!”.

10. Числовой ввод

10.1. Проверка валидации числового поля:

Тест переходит на страницу Inputs, вводит в числовое поле значение 55f и проверяет, что в поле остаётся только корректная числовая часть 55, а недопустимые символы отфильтрованы.

2. ОПИСАНИЕ КЛАССОВ И МЕТОДОВ

2.1. Классы и методы элементов страниц

1. BaseElement
Базовый абстрактный класс, инкапсулирующий общую логику взаимодействия с веб-элементами через Selenium WebDriver.
Поля:
- **WAIT_SECONDS** — стандартное время ожидания (10 секунд).
- **driver** — экземпляр WebDriver.
- **locator** — XPath-локатор элемента.
Методы:
- **BaseElement(WebDriver driver, String xpathTemplate, String attributeValue)** Конструктор, формирует XPath-локатор по шаблону и значению атрибута.
- **boolean isDisplayed()** - проверяет, виден ли элемент, с использованием стандартного таймаута.
- **boolean isDisplayed(int seconds**) - проверяет видимость элемента с кастомным временем ожидания.
boolean isEnabled() - проверяет, доступен ли элемент для взаимодействия (включён ли он).
- **String getAttribute(String attributeName)** - возвращает значение атрибута указанного имени или null, если элемент не найден.

2. Класс Button
Поля:
- static final String ID_XPATH, TEXT_XPATH, CLASS_XPATH, TYPE_XPATH, ARIA_LABEL_XPATH — шаблоны XPath для поиска кнопки по id, тексту, классу, типу и aria-метке
- static final int WAIT_SECONDS — время ожидания кликабельности кнопки (10 секунд)
Наследует: BaseElement
Конструктор:
- Button(WebDriver driver, String xpathTpl, String param) — инициализирует кнопку с указанным шаблоном и параметром
Методы:
- void click() — ждёт, пока кнопка станет кликабельной, и выполняет по ней клик
- static Button byId(WebDriver driver, String id) — создаёт кнопку по её id
- static Button byText(WebDriver driver, String text) — создаёт кнопку по её видимому тексту
- static Button byClass(WebDriver driver, String className) — создаёт кнопку по CSS-классу
- static Button byType(WebDriver driver, String type) — создаёт кнопку по атрибуту type

- static Button byAriaLabel(WebDriver driver, String ariaLabel) — создаёт кнопку по aria-метке
- static Button byXPath(WebDriver driver, String xpath) — создаёт кнопку по любому XPath

3. Класс ContextArea
Поля:
- **static final String HOTSPOT_XPATH** — жёстко заданный XPath для контекстной области
Наследует: BaseElement
Конструктор:
- ContextArea(WebDriver driver) — инициализирует область контекста с предопределённым XPath

4. Класс DownloadLink
Наследует: BaseElement
Конструктор:
- DownloadLink(WebDriver driver, String fileName) — находит ссылку на скачивание по подстроке в href
Методы:
- void click() — выполняет клик по ссылке для загрузки файла

5. Класс: Dropdown
Поля:
- Select select — обёртка Selenium для работы с выпадающим списком
Наследует: BaseElement
Конструктор:
- **Dropdown(WebDriver driver)** — инициализирует объект Select на основе фиксированного XPath
Методы:
- void selectByVisibleText(String text) — выбирает опцию по видимому тексту
- **String getSelectedOptionText()** — возвращает текст текущей выбранной опции
- boolean isOptionDisplayed(String value) — проверяет, отображается ли опция с заданным value

6. Класс Image
Поля:
- static final String INDEXED_IMAGE_XPATH — шаблон XPath для поиска тега <img> по порядковому индексу
Наследует: BaseElement

Методы:
- **Image(WebDriver driver, int index)** — инициализирует поиск изображения по его номеру на странице (XPath-индексация с 1)
- **boolean isBroken()** — проверяет, сломано ли изображение (naturalWidth или naturalHeight равны 0); при ошибке также возвращает true

7. Класс: Input
Наследует: BaseElement
Методы:
- **Input(WebDriver driver, String xpathTemplate, String attributeValue)** — инициализирует элемент ввода по заданному шаблону XPath и значению атрибута
- **void type(String text)** — очищает поле и вводит указанный текст
- **String getValue()** — возвращает текущее значение атрибута value поля
- **void sendKeys(String filePath)** — отправляет путь к файлу в инпут (для <input type="file">)

8. Класс: Label
Поля:
- **static final String TEXT_XPATH** — шаблон XPath для поиска элемента <p> по частичному тексту
- **static final String EXACT_TEXT_XPATH** — шаблон XPath для поиска элемента <p> по точному тексту
Наследует: BaseElement
Конструктор:
- **Label(WebDriver driver, String xpathTemplate, String value)** — инициализирует поиск метки по заданному шаблону и значению
Методы:
- **String getText()** — возвращает текст элемента
- **static Label byExactText(WebDriver driver, String exactText)** — создаёт метку по точному совпадению текста
- **static Label byPartialText(WebDriver driver, String partialText)** — создаёт метку по частичному совпадению текста
- **static Label byXPath(WebDriver driver, String xpath)** — создаёт метку по произвольному XPath

2.2. Классы и методы страниц

1. BasePage
Абстрактный класс, представляющий общую функциональность для всех страниц UI. Используется как основа для создания Page Object'ов.
Поля:
- **WAIT_SECONDS** — стандартное ожидание в секундах.
- **BASE_ELEMENT_XPATH** — XPath-шаблон для поиска базового элемента страницы.
- **driver** — WebDriver, связанный со страницей.
- **baseLocator** — локатор для идентификации загруженной страницы.
Методы:
- **BasePage(WebDriver driver, String uniqueElementIdentifier)**
Конструктор, задаёт локатор по уникальному атрибуту страницы.
- boolean isLoaded()
Проверяет, загружена ли страница — ищет основной элемент по локатору.
- **<T extends BasePage> T refresh(Class<T> pageClass)**
Обновляет страницу и возвращает новый экземпляр текущего Page Object'a.
- **<T extends BasePage> T createPage(Class<T> pageClass)**
Создаёт экземпляр страницы заданного класса через рефлексию. Предполагается наличие конструктора (WebDriver).

2. Класс AddRemovePage
Поля:
- **private final Button addElementButton** — кнопка добавления элементов
Наследует: BasePage
Конструктор:
- **AddRemovePage(WebDriver driver)** — инициализирует страницу и кнопку добавления
Методы:
- **void clickAddElement()** — кликает кнопку добавления элемента
- **List<WebElement> getDeleteButtons**() — возвращает список кнопок удаления
- **int getDeleteButtonsCount()** — возвращает количество кнопок удаления
- **void deleteAllElements()** — удаляет все элементы через клик по кнопкам
- **boolean hasDeleteButtons()** — проверяет наличие кнопок удаления

3 Класс AuthPage
Поля:
- **private final Label successMessage** — метка успешной авторизации
- **private static final String UNIQUE_TEXT** — уникальный текст страницы
Наследует: BasePage
Конструктор:

- **AuthPage(WebDriver driver)** — инициализирует страницу и метку сообщения
Методы:
- **String getMessage()** — возвращает текст сообщения

4.Класс BrokenImagesPage
Поля:
- **private final By imagesLocator** — локатор для поиска изображений
Наследует: BasePage
Конструктор:
- **BrokenImagesPage(WebDriver driver)** — инициализирует страницу
Методы:
- **List<Image> getAllImages()** — возвращает все изображения как объекты Image
- **boolean anyImageBroken()** — проверяет наличие сломанных изображений

5. Класс ContextMenuPage
Поля:
- **private final ContextArea contextArea** — область для контекстного меню
Наследует: BasePage
Конструктор:
- **ContextMenuPage(WebDriver driver)** — инициализирует страницу и контекстную область
Методы:
- **void rightClickOnHotspot()** — выполняет правый клик в целевой области
- **String getAlertText()** — возвращает текст алерта
- **void closeAlert()** — закрывает алерт

6. Класс DownloadPage
Поля: отсутствуют
Наследует: BasePage
Конструктор:
- **DownloadPage(WebDriver driver)** — инициализирует страницу
Методы:
- **DownloadLink getDownloadLink(String fileName)** — возвращает ссылку для скачивания по имени файла

7. Класс DropdownPage
Поля:
- **private final Dropdown dropdown** — выпадающий список
Наследует: BasePage
Конструктор:
- **DropdownPage(WebDriver driver)** — инициализирует страницу и выпадающий список
Методы:
- **void selectOption(String visibleText)** — выбирает опцию по видимому тексту
- **String getSelectedOption()** — возвращает текст выбранной опции

- **boolean isOptionDisplayed(String value**) — проверяет отображение опции

8. Класс HoversPage
Поля:
- **private final WebElement avatarIcon** — иконка аватара
- **private static final String UNIQUE_IDENTIFIER** — уникальный идентификатор страницы
Наследует: BasePage
Конструктор:
- **HoversPage(WebDriver driver)** — инициализирует страницу и иконку аватара
Методы:
- **void hoverOnAvatar()** — наводит курсор на иконку аватара
- **String getUserNameText()** — возвращает текст имени пользователя после наведения

9. Класс InputsPage
Поля:
- **public final Input numberInput** — поле для ввода чисел
Наследует: BasePage
Конструктор:
- **InputsPage(WebDriver driver)** — инициализирует страницу и поле ввода
Методы:
- **void open()** — открывает страницу с полем ввода

10. Класс KeyPressPage
Поля:
- **private final Actions actions** — объект для действий с клавиатурой
- **private final By resultText** — локатор текста результата
Наследует: BasePage
Конструктор:
- **KeyPressPage(WebDriver driver**) — инициализирует страницу и объект действий
Методы:
- **void pressKey(CharSequence key)** — имитирует нажатие клавиши
- **String getDisplayedText()** — возвращает текст результата

11. Класс UploadPage
Поля: отсутствуют
Наследует: BasePage
Конструктор:
- **UploadPage(WebDriver driver)** — инициализирует страницу загрузки
Методы:
- **Input fileInput()** — возвращает поле выбора файла
- **Button uploadButton()** — возвращает кнопку загрузки
- **Label successMessage()** — возвращает метку успешной загрузки

- **Label uploadedFileName()** — возвращает метку с именем загруженного файла

2.3. Классы и методы тестов

1.Класс BaseTest
Базовый тестовый класс, содержащий конфигурацию окружения для UI-тестов на Selenium WebDriver.
Поля:
- **driver** — экземпляр WebDriver.
BASE_URL — базовый адрес тестируемого приложения (https://the-internet.herokuapp.com/).
Методы:
- **@BeforeEach void setUp()**
Настраивает окружение перед каждым тестом: создает ChromeDriver с нужными опциями, задаёт таймауты, открывает стартовую страницу.
- **@AfterEach void tearDown()**
Завершает работу WebDriver после теста, если он был создан.

2. Класс AddRemoveTest
Наследует: BaseTest
Методы:
- **@Test void testAddAndRemoveElements()** — тестирует функционал добавления/удаления элементов:
Добавляет 2 элемента
Проверяет количество кнопок удаления
Удаляет все элементы
Проверяет отсутствие кнопок

3. Класс AuthTest
Наследует: BaseTest
Методы:
- **@Test void testBasicAuth()** — проверяет авторизацию и сообщение об успехе

4. Класс BrokenImageTest
Наследует: BaseTest
Методы:
- **@Test void testBrokenImagesPresent()** — проверяет наличие сломанных изображений

5. Класс ContextMenuTest
Наследует: BaseTest
Методы:
- **@Test void testContextMenuAlertAppearsCorrectly()** — проверяет текст алерта

при контекстном меню

6. Класс DownloadTest
Поля:
- **private WebDriver driver** — экземпляр драйвера
- **private static final String BASE_URL** — базовый URL
- **private static final String DOWNLOAD_DIR** — директория загрузок
- **private static final String FILE_NAME** — имя файла
Методы:
- **@BeforeEach void setUp()** — настраивает окружение перед тестом
- **@Test void testFileDownload()** — проверяет скачивание файла
- **@AfterEach void tearDown()** — закрывает драйвер после теста

7. Класс: DropdownTest
Поля:
- **private WebDriver driver** — экземпляр драйвера
- **private static final String BASE_URL** — базовый URL
- **private static final String DOWNLOAD_DIR** — директория загрузок
- **private static final String FILE_NAME** — имя файла
Методы:
- **@BeforeEach void setUp()** — настраивает окружение перед тестом
- **@Test void testFileDownload()** — проверяет скачивание файла
- **@AfterEach void tearDown()** — закрывает драйвер после теста

8. Класс: HoversTest
Наследует: BaseTest
Методы:
- **@Test void testHoverShowsUserName()** — проверяет отображение имени при наведении

9. Класс: InputsTest
Наследует: BaseTest
Методы:
- **@Test void testNumberInputIgnoresNonNumericCharacters()** — проверяет игнорирование нечисловых символов в поле ввода

10. Класс: KeyPressTest
Наследует: BaseTest
Методы:
- **@Test void testKeyPresses()** — проверяет отображение нажатых клавиш (буквы и спец. клавиши)

11. Класс: UploadTest
Наследует: BaseTest
Методы:
- **@Test void testFileUpload()** — проверяет загрузку файла через UI:
    1. Создаёт временный файл
    2. Загружает файл через форму
    3. Проверяет сообщение об успехе и имя файла
    4. Удаляет временный файл

3. ТЕСТИРОВАНИЕ

1. Демонстрация работы теста 9

Тест 9 осуществляет проверку загрузки файла на сайт с устройства пользователя. Производятся следующие действия:

1. Переходим на сайт (рисунок 1)
2. Находим заголовок File Upload и кликаем на него (рисунок 2)
3. Нажимаем на кнопку выберите файл (рисунок 3)
4. Выбираем файл и нажимаем на кнопку upload (рисунок 4)

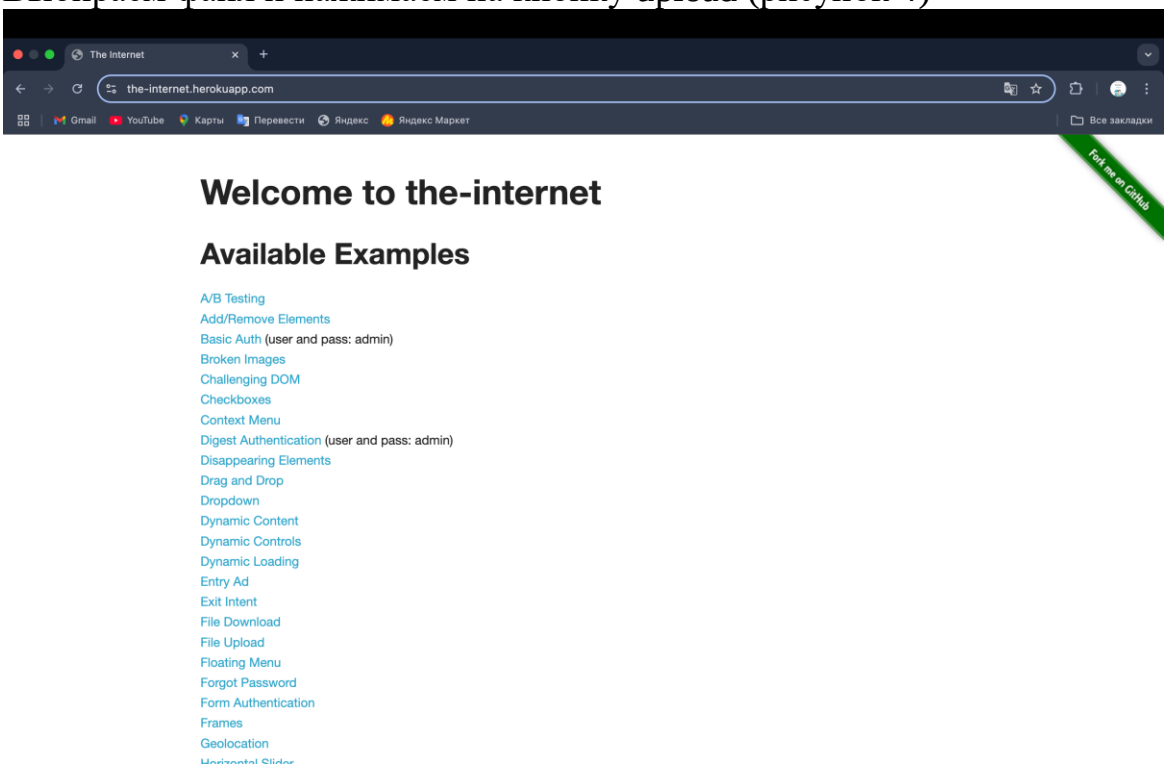

Рисунок 1 - Главная страница .

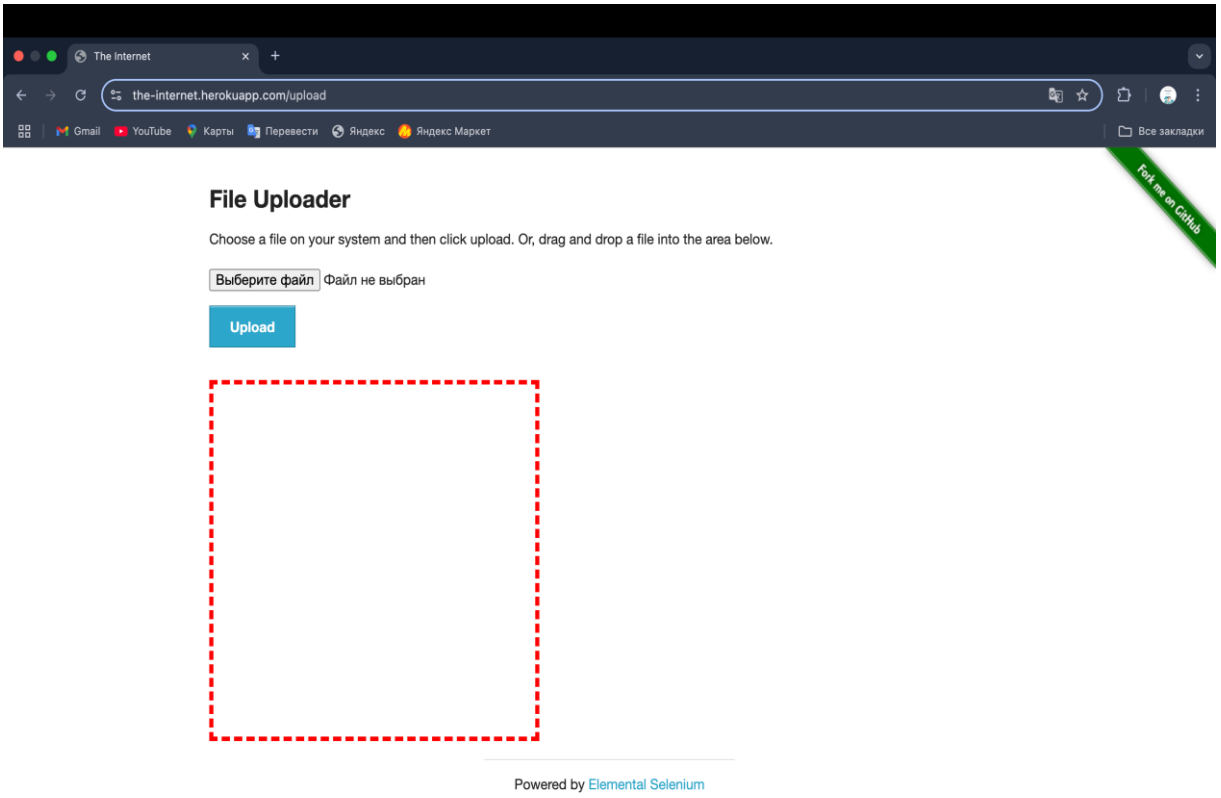

Рисунок 2 – Страница с загрузкой файла.

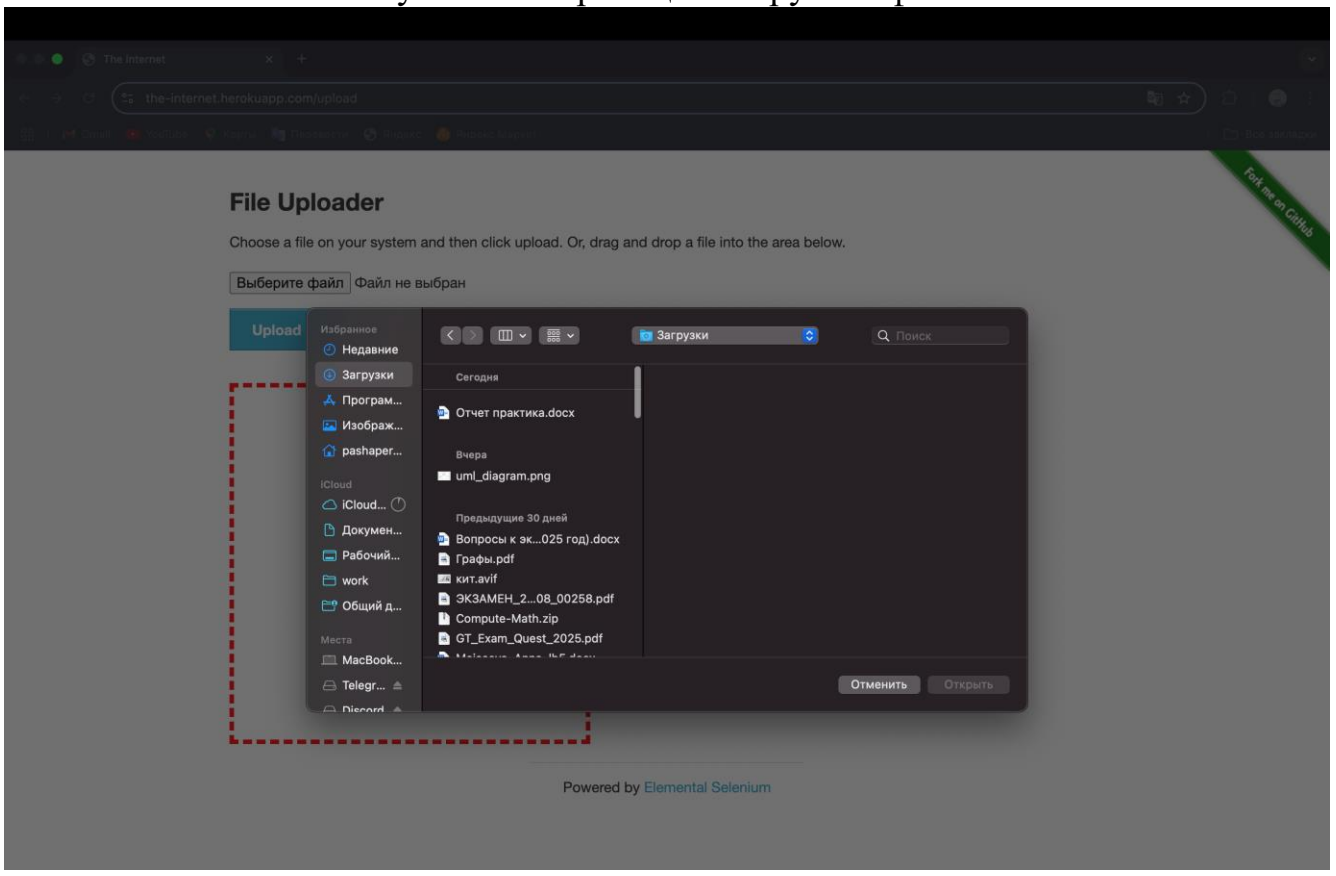

Рисунок 3 – Выбор файла.

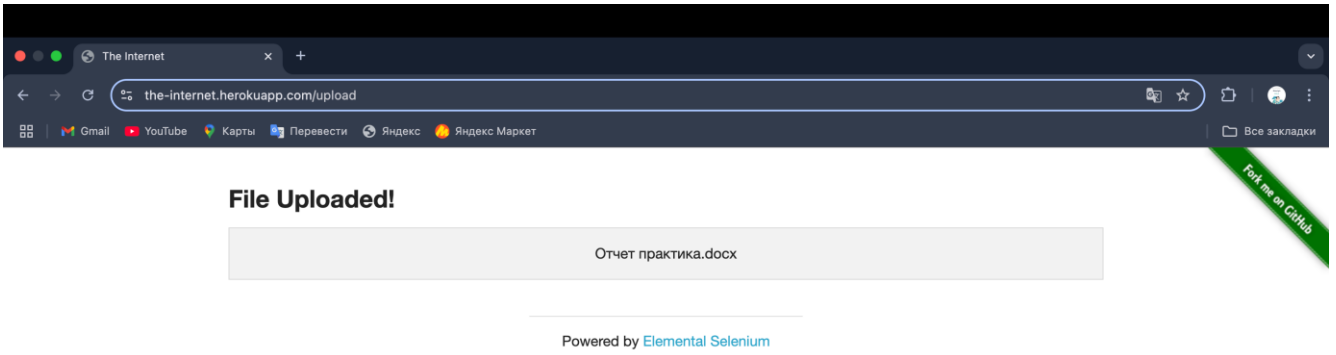

Рисунок 4 - Главная страница после загрузки выбранного вами файл.

ЗАКЛЮЧЕНИЕ

В результате работы было проведено комплексное тестирование демонстрационного сайта The Internet (Heroku). Были проверены ключевые пользовательские сценарии взаимодействия с элементами интерфейса, формами ввода и навигацией.

Протестированы такие, как: авторизация, нажатие клавиш с клавиатуры, добавление и удаление динамических элементов, работа с выпадающим списком, вызов контекстного меню, отображение информации при наведении курсора, загрузка и скачивание файлов, ввод числовых значений и проверка битых изображений.

Для тестирования реализованы классы страниц, элементов страниц и классы тестов. Класс элементов защищён от прямого обращения из тестов — взаимодействие с элементами осуществляется исключительно через методы классов страниц (Page Object Model).

Тестирование включало как автоматические (JUnit, Selenium), так и ручные проверки. Были протестированы как стандартные действия, так и поведение системы при некорректных сценариях.

Сайт использовался как платформа для отработки и отладки базовых сценариев пользовательского взаимодействия и проверки обработки событий браузером.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Automated Testing – XPath в веб-тестировании: написание и отладка. URL: https://automated-testing.info/t/xpath-v-web-testirovanii-napisanie-i-otladka/23808 (дата обращения: 30.06.2025)
2. Автотесты с нуля Selenide Java. URL: https://www.youtube.com/watch?v=IGaAZWmqZEE&t=2204s (дата обращения: 30.06.2025)
3. Поиск элементов на странице Xpath и CSS с нуля. URL: https://www.youtube.com/watch?v=zGYdbHACmLI&t=464s (дата обращения: 30.06.2025)
4. Xpath — залог стабильных UI-автотестов на Web и Mobile Web. URL: https://habr.com/ru/companies/vk/articles/806661/ (дата обращения: 30.06.2025)
5. Introduction to Selenide, Baeldung. URL: https://www.baeldung.com/selenid (дата обращения: 30.06.2025)

ПРИЛОЖЕНИЕ А

UML-ДИАГРАММА КЛАССОВ

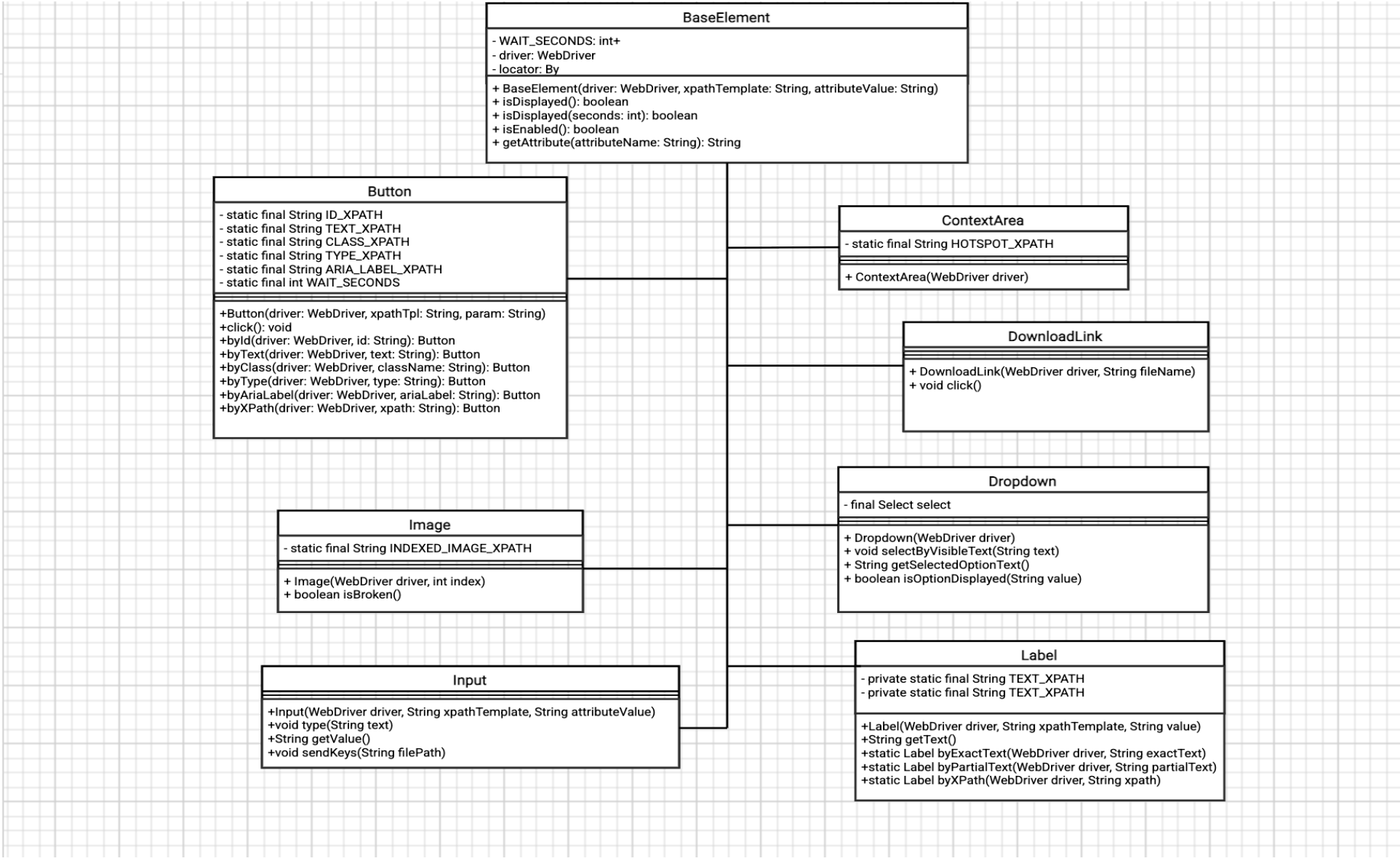

Рисунок 1 – UML-диаграмма наследников BaseElement

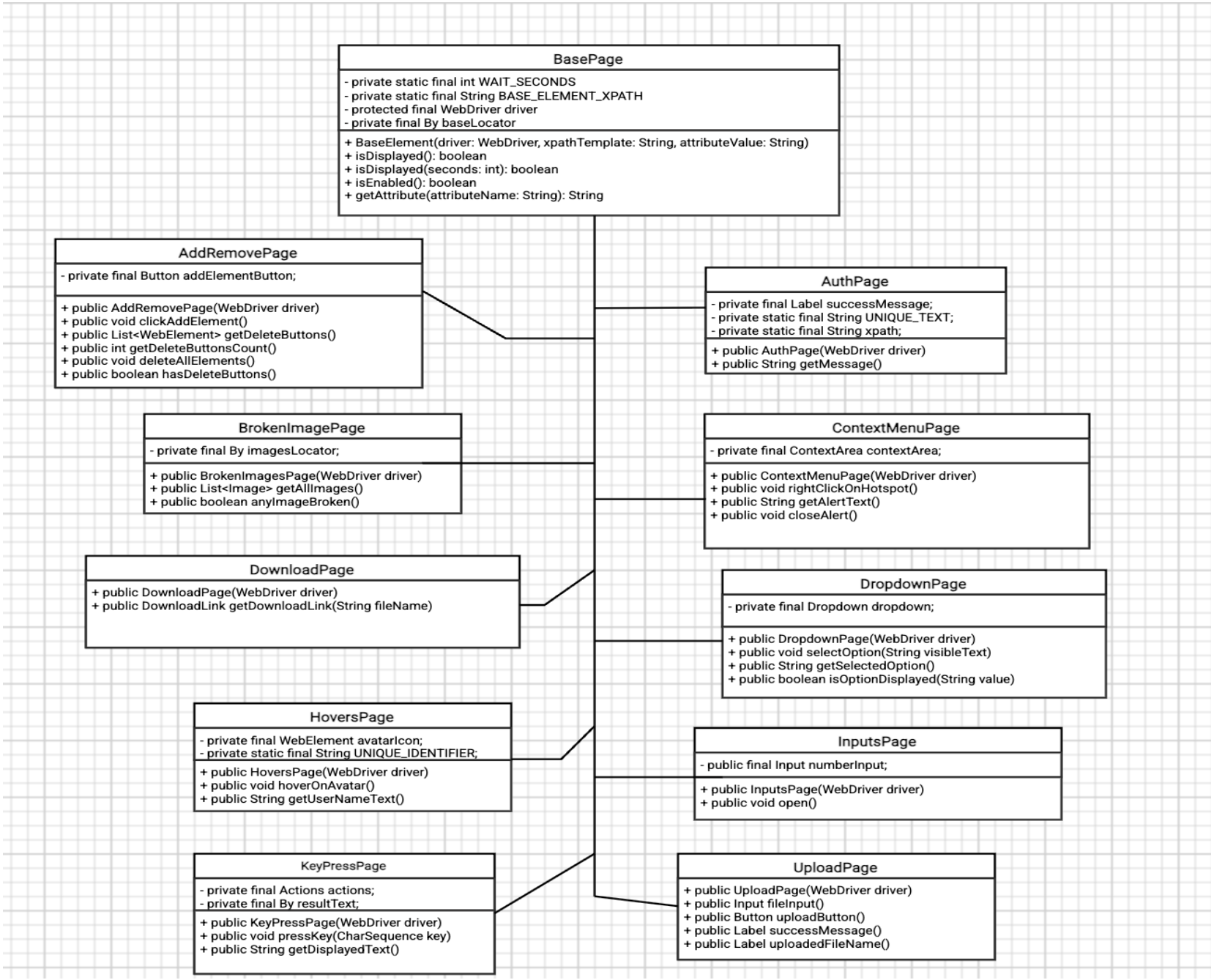

Рисунок 2 – UML-диаграмма наследников BasePage

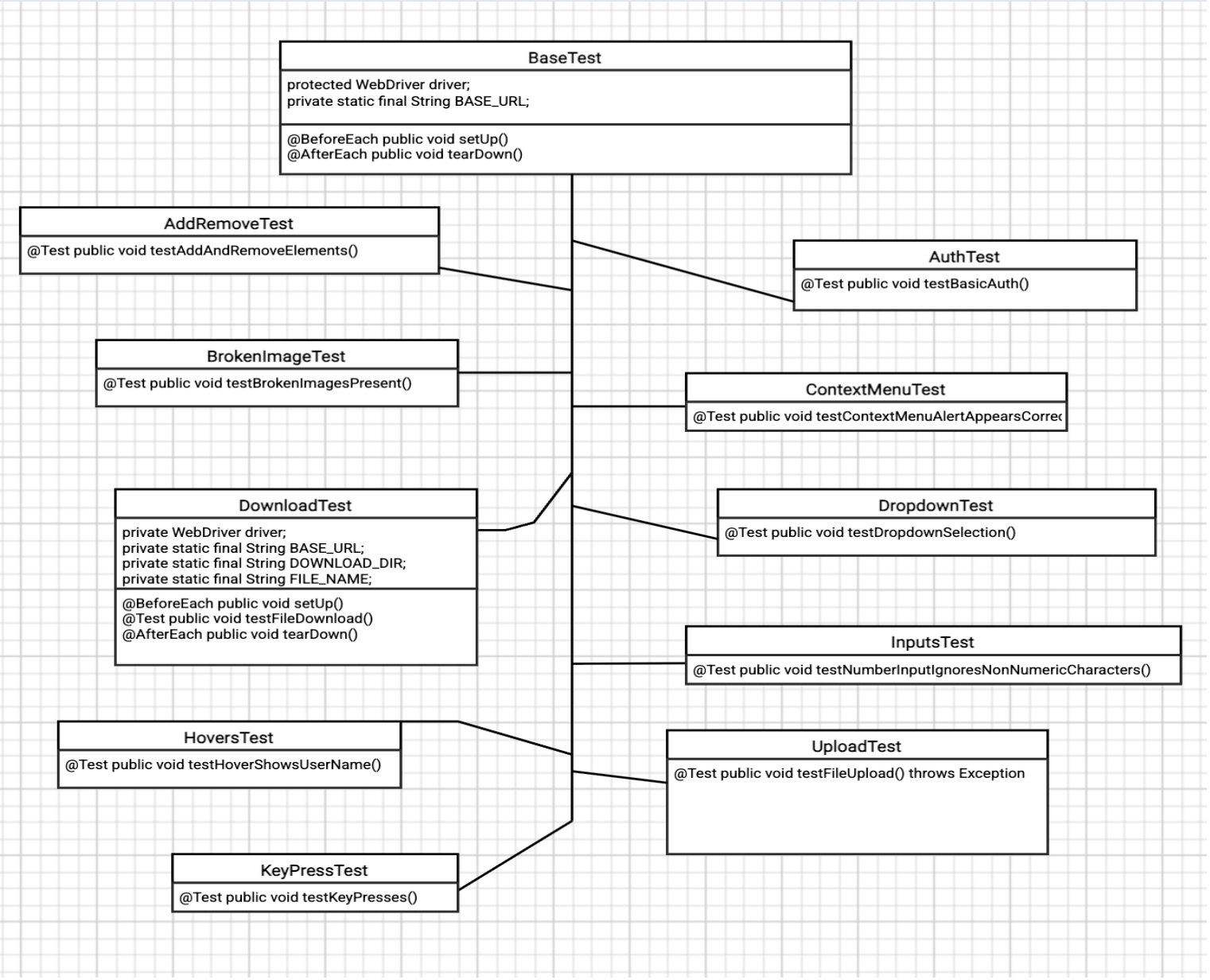

Рисунок 3– UML-диаграмма наследников BaseTest