

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

Кафедра МОЕВМ

**ОТЧЕТ
по «UI-тестирование» практике
Тема: Gmail.com**

Студент гр. 3388

Тимошук Е.
Потапов Р.
Кулач Д.

Руководитель

Шевелева А. М.

Санкт-Петербург

2025

ЗАДАНИЕ

НА «UI-тестирование» ПРАКТИКУ

Студенты: Тимошук Е. Потапов Р. Кулач Д.

Группа 3388

Тема практики: **Gmail.com**

Задание на практику:

Написать архитектуру для реализации 10 UI-тестов для системы Gmail

Сроки прохождения практики: 26.06.2025 – 05.07.2025

Дата сдачи отчета: 07.07.2025

Дата защиты отчета: 05.07.2025

Студент

Тимошук Е.
Потапов Р.
Кулач Д.

Руководитель

Шевелева А.М.

АННОТАЦИЯ

В данном отчете представлены результаты прохождения практики, посвященной UI-тестированию интерфейса Gmail. Были рассмотрены основные элементы пользовательского интерфейса, проведено функциональное и юзабилити-тестирование, а также проанализирована работа сервиса в разных браузерах и на различных устройствах.

В ходе тестирования использовались ручные методы проверки и инструменты разработчика (DevTools). Составлены тест-кейсы, выявлены возможные баги и предложены рекомендации по улучшению интерфейса.

Отчет содержит описание методологии тестирования, примеры тест-кейсов с результатами, список обнаруженных проблем и выводы по проделанной работе

SUMMARY

This report presents the results of the practice dedicated to UI testing of the Gmail interface. The main elements of the user interface were reviewed, functional and usability testing was conducted, and the service's performance in different browsers and on various devices was analyzed.

During the testing, manual verification methods and developer tools (DevTools) were used. Test cases were created, possible bugs were identified, and recommendations for improving the interface were proposed.

The report contains a description of the testing methodology, examples of test cases with results, a list of detected issues, and conclusions on the work performed.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ	5
ВЫПОЛНЕНИЕ РАБОТЫ	6
1. ЧЕК ЛИСТ	6
2. ОПИСАНИЕ КЛАССОВ И МЕТОДОВ	13
2.1 Классы и методы элементов страницы	13
2.2 Классы и методы страницы	15
2.3 Базовый класс для тестов	19
2.4 Классы и методы тестов	20
3. ТЕСТИРОВАНИЕ	23
3.1 Тестирование авторизации	23
3.2 Тестирование всех тестов	25
UML – диаграмма	29
ЗАКЛЮЧЕНИЕ	30
СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ	31

ВВЕДЕНИЕ

Цель работы: Изучение методов UI-тестирования на примере Gmail.

Задачи:

- Анализ интерфейса Gmail.
- Проверка юзабилити и функциональности.
- Составление тест-кейсов.

ВЫПОЛНЕНИЕ РАБОТЫ

1. ЧЕК ЛИСТ

№ теста	Тесты	Шаги теста	Ожидаемый результат
1	Установка фильтров для писем	Сначала авторизуйтесь в почтовой системе. Затем перейдите в раздел «Настройки» и выберите подраздел «Фильтры и правила». Нажмите на кнопку «Создать новый фильтр». В открывшемся окне задайте условия фильтрации: в поле «Отправитель» укажите адрес example@domain.com, а в качестве действия выберите «Переместить в папку «Важное»». Сохраните созданный фильтр.	Письмо от example@domain.com автоматически перемещается в папку «Важное».
2	Изменение имени и фамилии пользователя	Чтобы изменить данные профиля в системе, сначала авторизуйтесь в своём аккаунте. Затем перейдите в раздел «Настройки» и выберите вкладку «Профиль». В открывшемся окне найдите поля «Имя» и «Фамилия» и введите новые значения, например, "Иван Иванов". После внесения изменений нажмите кнопку «Сохранить»	Письмо исчезает из исходной папки и появляется в «Корзине». При восстановлении — возвращается на прежнее место.

3	Удаление письма	<p>Процесс удаления письма:</p> <p>Для удаления письма сначала авторизуйтесь в системе, затем перейдите во «Входящие», выберите нужное письмо и нажмите кнопку «Удалить». После этого письмо переместится в раздел «Корзина» — зайдите туда, чтобы убедиться, что оно действительно там находится.</p> <p>Восстановление письма:</p> <p>Если вы случайно удалили письмо, его можно вернуть. Для этого откройте «Корзину», выберите нужное письмо и нажмите «Восстановить». После этого проверьте исходную папку (например, «Входящие»), куда письмо должно вернуться.</p>	Сообщение удалено или успешно восстановлено
4	Добавление контакта	<p>Для добавления нового контакта сначала авторизуйтесь в своей почтовой системе. Затем перейдите в раздел «Контакты» и нажмите кнопку «Новый контакт». В открывшейся форме заполните необходимые поля:</p> <ul style="list-style-type: none"> - В поле «Имя» введите «Даниил» - В поле Email укажите адрес kulachdv@gmail.com 	В списке контактов появился «Даниил»

		После заполнения данных нажмите кнопку «Сохранить».	
5	Проверка восстановления сессии после разрыва соединения	<p>Для проверки устойчивости почтовой системы к разрывам соединения необходимо сначала авторизоваться в аккаунте. После успешного входа следует создать новое письмо, введя произвольный текст и прикрепив тестовый файл. На этом этапе важно искусственно прервать интернет-соединение – это можно сделать через настройки сети устройства или с помощью инструментов разработчика в браузере.</p> <p>После обрыва соединения нужно выждать 1-2 минуты, затем восстановить подключение к интернету. Система должна автоматически сохранить черновик письма со всем введенным содержимым и вложениями.</p>	Система сохраняет все введенные данные
6	Смена языка	Для смены языка в веб-интерфейсе Google необходимо сначала войти в свою учетную запись. После авторизации перейдите в раздел "Управление аккаунтом Google", где собраны все основные настройки профиля. В	Язык системы был изменен

		<p>открывшемся меню выберите вкладку "Личная информация", которая содержит персональные данные и предпочтения отображения.</p> <p>Среди различных параметров найдите блок "Общие настройки веб-интерфейса" - именно здесь находится опция выбора языка. В соответствующем поле можно установить предпочитаемый язык из доступного списка. После изменения этого параметра интерфейс Google автоматически обновится, и все меню, подсказки и системные сообщения будут отображаться на выбранном языке.</p>	
7	Отправка письма несуществующему адресату (проверка ошибки).	<p>После авторизации в почтовой системе необходимо создать новое сообщение, нажав кнопку "Написать письмо". В появившейся форме отправки следует ввести в поле "Кому" заведомо неполный адрес электронной почты в формате "user@" без указания доменной части. Попытавшись отправить такое письмо, вы сможете проверить, как система</p>	<p>Система дает возможность отправить письмо любому пользователю (даже несуществ.)</p>

		обрабатывает некорректно введенные адреса.	
8	Постановление задачи	<p>Для добавления новой задачи сначала необходимо авторизоваться в системе. После входа в аккаунт перейдите в раздел "Задачи", где отображаются все текущие и завершенные задания. Нажав кнопку "Добавить задачу", вы откроете форму создания, где нужно придумать и ввести краткое, но понятное название, отражающее суть предстоящей работы.</p> <p>После заполнения названия обязательно сохраните задачу соответствующей кнопкой - теперь она появится в вашем списке и будет доступна для редактирования, отметки о выполнении или добавления дополнительных деталей.</p>	Система создает задачу
9	Пометить письмо как прочитанное/непрочитанное	После авторизации в почтовой системе откройте папку "Входящие" и найдите среди списка сообщений непрочитанное письмо (обычно такие письма выделяются жирным шрифтом или специальным индикатором). Чтобы изменить его статус на	Статус письма меняется на «Прочитано»

		прочитанное, достаточно кликнуть по нему правой кнопкой мыши - это откроет контекстное меню, где нужно выбрать опцию "Пометить как прочитанное".	
10	Поиск, по ключевым словам, в письмах	<p>После авторизации в системе используйте глобальную поисковую строку для проверки работы фильтров. Сначала введите реально существующее слово «Здравствуйте» (которое должно встречаться в переписке) и выполните поиск, затем попробуйте найти заведомо вымышленное сочетание «Несуществующее Слово» — это поможет проверить реакцию системы на отсутствующие в базе данные.</p> <p>Далее измените параметры поиска, активировав фильтр «Тема», и повторите те же запросы — теперь система будет искать указанные слова исключительно в заголовках писем. После этого переключитесь на фильтр «Слова в письме», который проверяет только основной текст сообщений, и снова</p>	<p>«Здравствуйте» -> показываются только письма, где слово найдено в выбранной области; результатов> 0.</p> <p>«Несуществующее Слово» -> «Нет результатов», список пуст.</p>

		<p>выполните поиск по обоим вариантам запросов.</p>	
--	--	---	--

2. ОПИСАНИЕ КЛАССОВ И МЕТОДОВ

2.1 Классы и методы элементов страницы

Класс ``BaseComponent`` (Утилита для работы с локаторами)

Методы этого класса служат фабриками для создания объектов типа ``By``, которые используются в Selenium для поиска элементов на веб-страницах. Каждый метод возвращает конкретную стратегию локации элемента:

1. ``byClass()``

Создает локатор, находящий элементы по их CSS-классу. Принимает строку с именем класса (без точки) и возвращает объект ``By.className``. Ищет элементы, у которых атрибут ``class`` содержит указанное значение.

2. ``byName()``

Генерирует локатор для поиска элементов по значению атрибута ``name``. Возвращает объект ``By.name``. Особенно полезен для работы с формами (поля ввода, кнопки).

3. ``byCss()``

Создает локатор на основе CSS-селектора. Возвращает объект ``By.cssSelector``. Поддерживает сложные селекторы (комбинаторы, псевдо классы), что позволяет точно находить элементы по их положению в DOM.

4. ``byXPath()``

Формирует локатор с помощью XPath-выражения. Возвращает объект ``By.xpath``. XPath обеспечивает максимальную гибкость поиска, включая навигацию по иерархии элементов и фильтрацию по любым атрибутам.

5. ``byText()``

Специальный метод для поиска элементов по точному текстовому содержимому. Внутри использует XPath-шаблон ``//*[text()='...']``. Находит только элементы, чей текст полностью совпадает с переданным значением.

6. ``byLinkText()``

Оптимизированный метод для поиска гиперссылок (`<a>` тегов) по их тексту. Возвращает объект `By.linkText`. Работает только с текстом внутри тега ссылки.

Класс `BaseElement` (Обёртка для веб-элементов)

Инкапсулирует стандартный `WebElement` Selenium, добавляя автоматическую обработку ожиданий и логирование:

1. Конструктор

Принимает два параметра:

- `WebElement` (базовый элемент Selenium)
- `WebDriverWait` (объект для явных ожиданий)

Инициализирует внутреннее состояние обёртки.

2. `click()`

Выполняет безопасный клик по элементу:

- Автоматически дожидается видимости элемента и его активности (кликабельности) через `WebDriverWait`
- Фиксирует действие в системном логе (информация о целевом элементе)
- Вызывает нативный метод `click()` у базового элемента

Гарантирует, что элемент готов к взаимодействию перед выполнением действия.

3. `enterText()`

Обеспечивает надёжный ввод текста:

- Ожидает видимость элемента в DOM
- Логирует операцию с указанием вводимого текста
- Автоматически очищает поле перед вводом
- Передаёт текст базовому элементу через метод `sendKeys()`

Защищает от распространённых ошибок (ввод в скрытые/недоступные поля).

Принцип работы в связке

1. Создание локатора

Класс страницы вызывает методы ``BaseComponent`` для получения локаторов:

```
`By emailLocator = BaseComponent.byName("email")`
```

2. Поиск элемента

Через драйвер находится сырой элемент:

```
`WebElement rawElement = driver.findElement(emailLocator)`
```

3. Создание обёртки

Сырой элемент оборачивается в ``BaseElement``:

```
`BaseElement emailField = new BaseElement(rawElement, wait)`
```

4. Взаимодействие

Тесты используют безопасные методы обёртки:

```
`emailField.enterText("test@example.com")`
```

2.2 Классы и методы страницы

1. Базовый класс ``BasePage``

Назначение:

Абстрактный класс, содержащий общую логику для всех страниц приложения.

Ключевые методы:

- ``pauseMillis()``

Выполняет асинхронную паузу через `JavaScriptExecutor`. Используется для искусственных задержек.

- ``pauseFiveSeconds()``

Специализированная пауза на 5 секунд (вызов ``pauseMillis(5000)``).

- ``waitForPageLoad()``

Ожидает полной загрузки страницы через проверку состояния ``document.readyState``.

2. Класс ``ComposePage`` (Диалог создания письма)

Назначение:

Управление диалогом "Написать письмо" в Gmail.

Ключевые методы:

- ``enterRecipient()``

Вводит адрес получателя в поле "Кому".

- ``enterSubject()``

Заполняет поле "Тема" письма.

- ``enterBody()``

Вводит текст в тело письма.

- ``clickSend()``

Кликает кнопку "Отправить".

3. Класс ``ContactsPage`` (Управление контактами)

Назначение:

Работа с разделом "Контакты" внутри Gmail.

Ключевые методы:

- ``clickNewContact()``

Открывает форму создания нового контакта.

- ``enterName()``

Заполняет поле "Имя" контакта.

- ``enterEmail()``

Вводит email контакта.

- ``waitForSaveAndReturn()``

Ожидает сохранения и возвращается в основной контекст страницы.

4. Класс ``FiltersPage`` (Управление фильтрами)

Назначение:

Работа с разделом "Фильтры и заблокированные адреса".

Ключевые методы:

- ``clickCreateNewFilter()``

Иницирует создание нового фильтра.

- ``enterFromAddress()``

Заполняет поле "От" для фильтрации.

- ``clickNeverSpamOption()``

Активирует опцию "Никогда не отправлять в спам".

5. Класс `InboxPage` (Входящие письма)

Назначение:

Основная рабочая область Gmail (список писем).

Ключевые методы:

- `getUnreadCount()`

Возвращает количество непрочитанных писем.

- `selectFirstUnread()`

Выбирает первое непрочитанное письмо.

- `clickMarkAsRead()`

Помечает выделенные письма как прочитанные.

- `openContactsPage()`

Переходит в раздел "Контакты" (возвращает `ContactsPage`).

6. Класс `LoginPage` (Авторизация)

Назначение:

Обработка процесса входа в аккаунт Gmail.

Ключевые методы:

- `enterEmail()`

Вводит email в поле идентификатора.

- `clickEmailNext()`

Отключает WebDriver-флаг и переходит к вводу пароля.

- `enterPassword()`

Заполняет поле пароля.

7. Класс `MyAccountPage` (Управление аккаунтом)

Назначение:

Работа с персональными данными в Google My Account.

Ключевые методы:

- `openNameEdit()`

Открывает форму редактирования имени (с обработкой сложного UI).

- `updateName()`

Обновляет имя и фамилию пользователя.

8. Класс `OfflinePage` (Эмуляция сети)

Назначение:

Управление сетевыми условиями через Chrome DevTools Protocol (CDP).

Ключевые методы:

- `emulateOffline()`

Переводит браузер в режим offline.

- `emulateOnline()`

Восстанавливает онлайн-режим.

- `getOfflineErrorText()`

Возвращает текст системного сообщения об ошибке.

9. Класс `ProfileMenuPage` (Меню профиля)

Назначение:

Взаимодействие с выпадающим меню профиля.

Ключевой метод:

- `goToMyAccount()`

Переключается в iframe меню, открывает "Мой аккаунт" в новом окне.

10. Класс `SettingsPage` (Настройки Gmail)

Назначение:

Управление настройками почты.

Ключевые методы:

- `clickAllSettings()`

Открывает полный интерфейс настроек.

- `selectLanguage()`

Изменяет язык интерфейса через выпадающий список.

11. Класс `TasksPage` (Управление задачами)

Назначение:

Работа с виджетом задач внутри Gmail.

Ключевые методы:

- `clickAddTaskButton()`

Открывает форму создания задачи.

- ``enterTaskTitle()``

Заполняет заголовок задачи.

2.3 Базовый класс для тестов

Базовый класс ``BaseTest``, который служит основой для всех тестовых классов в проекте автоматизации. Он содержит общую логику, необходимую для запуска и завершения браузера, настройки параметров `WebDriver`, открытия начальной страницы приложения перед каждым тестом, а также предоставляет универсальный метод авторизации в системе.

Класс использует Selenium `WebDriver` (в данном случае — `ChromeDriver`) и применяет подход объектной модели страницы (`Page Object Model`), что делает тесты более структурированными, читаемыми и поддерживаемыми.

Метод ``setUp()``

Этот метод аннотирован как ``@BeforeEach``, то есть он выполняется перед каждым тестовым методом. В нём происходит:

- Инициализация драйвера браузера Google Chrome с настройками, которые позволяют скрыть факт автоматизации от веб-сайта (это помогает избежать блокировок или дополнительных проверок безопасности).

- Отключение некоторых функций, связанных с автоматизацией (``enable-automation``, ``useAutomationExtension``), чтобы сделать поведение браузера похожим на пользовательское.

- Установка максимального времени ожидания для поиска элементов на странице — 30 секунд.

- Открытие базовой страницы входа в Gmail.

Метод ``tearDown()``

Аннотированный как ``@AfterEach``, этот метод выполняется после каждого тестового сценария и отвечает за корректное закрытие браузера. Это важно для предотвращения утечек памяти и обеспечения изоляции между тестами.

Метод ``auth (String email, String password)``

Этот метод реализует логику входа пользователя в систему. Он принимает логин и пароль, взаимодействует с элементами страницы через класс

`LoginPage`, вводит данные и кликает по кнопкам "Далее". После успешного входа возвращает экземпляр класса `InboxPage`, представляющий собой главную страницу почтового ящика. Таким образом, тесты могут сразу продолжить выполнение действий уже на странице Inbox, без повторного написания логики авторизации.

Классы `LoginPage` и `InboxPage`

Эти классы являются частью паттерна Page Object. Они инкапсулируют действия и элементы конкретных страниц приложения:

- `LoginPage` — содержит методы для работы с полями ввода email и пароля, а также кнопками перехода.

- `InboxPage` — представляет собой страницу входящей почты и предоставляет методы для дальнейших действий с письмами.

2.4 Классы и методы тестов

1. Базовые классы (Page Objects)

- `BaseTest`

Базовый класс для всех тестов. Содержит общую логику:

- Инициализация драйвера
- Метод авторизации `auth()`
- Общие ожидания и утилиты

- `InboxPage`

Работа с почтовым ящиком:

- `clickSettingsIcon()`: Открывает настройки
- `openContactsPage()`: Переход в контакты
- `selectFirstUnread()`: Выбор первого непрочитанного письма
- `clickDelete()`: Удаление письма
- `getUnreadCount()`: Получение количества непрочитанных писем

- `AdvancedSearchPage`

Расширенный поиск:

- `openSearch()`: Открытие панели поиска

- `typeFirstQuery()`, `typeSecondQuery()`: Ввод запросов
- `clearFirst()`: Очистка поля
- `submitSearch()`: Запуск поиска

- `SettingsPage`

Управление настройками:

- `clickAllSettings()`: Открытие полных настроек
- `selectLanguage()`: Выбор языка
- `clickSaveChanges()`: Сохранение настроек

- `ContactsPage`

Работа с контактами:

- `clickNewContact()`: Создание контакта
- `enterName()`, `enterEmail()`: Заполнение данных
- `saveContact()`: Сохранение контакта

- `FiltersPage`

Управление фильтрами:

- `clickCreateNewFilter()`: Создание фильтра
- `enterFromAddress()`: Ввод адреса отправителя
- `clickNeverSpamOption()`: Активация опции "Не спам"
- `clickFinalCreateFilter()`: Финализация фильтра

- `ComposePage`

Создание писем:

- `enterRecipient()`, `enterSubject()`, `enterBody()`: Заполнение полей
- `clickSend()`: Отправка письма

- `OfflinePage`

Тестирование offline-режима:

- `enableNetwork()`, `emulateOffline()`, `emulateOnline()`: Управление

сетевым статусом

- `getOfflineErrorText()`: Получение сообщения об ошибке

2. Тестовые классы

- ``AdvancedSearchTest``
 - ``shouldPerformAllAdvancedSearchScenarios()``: Проверяет 4 сценария поиска с разными параметрами.
- ``ChangeLanguageTest``
 - ``shouldChangeInterfaceLanguage()``: Меняет язык интерфейса на английский.
- ``ContactsTest``
 - ``shouldAddNewContact()``: Создает новый контакт.
- ``DeleteMessageTest``
 - ``shouldDeleteFirstUnreadMessage()``: Удаляет первое непрочитанное письмо.
- ``FiltersTest``
 - ``shouldCreateNeverSpamFilter()``: Создает фильтр против спама.
- ``MessageSenderTest``
 - ``shouldSendEmailSuccessfully()``: Отправляет тестовое письмо.
- ``MessageTesterTest``
 - ``shouldMarkFirstUnreadAsRead()``: Помечает письмо как прочитанное.
- ``OfflineLoginTest``
 - ``shouldShowOfflineErrorWhenNetworkDisconnected ()``: проверяет поведение при отключении интернета.

3. ТЕСТИРОВАНИЕ

3.1 Тестирование авторизации

Page Object для авторизации в Gmail. Управляет элементами страницы входа.

Основные методы:

1. `enterEmail()` → Вводит логин
2. `clickEmailNext()` →
 - Переходит к паролю
3. `enterPassword()` → Вводит пароль
4. `clickPasswordNext()` →
 - Завершает авторизацию
 - Ожидает загрузки почтового ящика

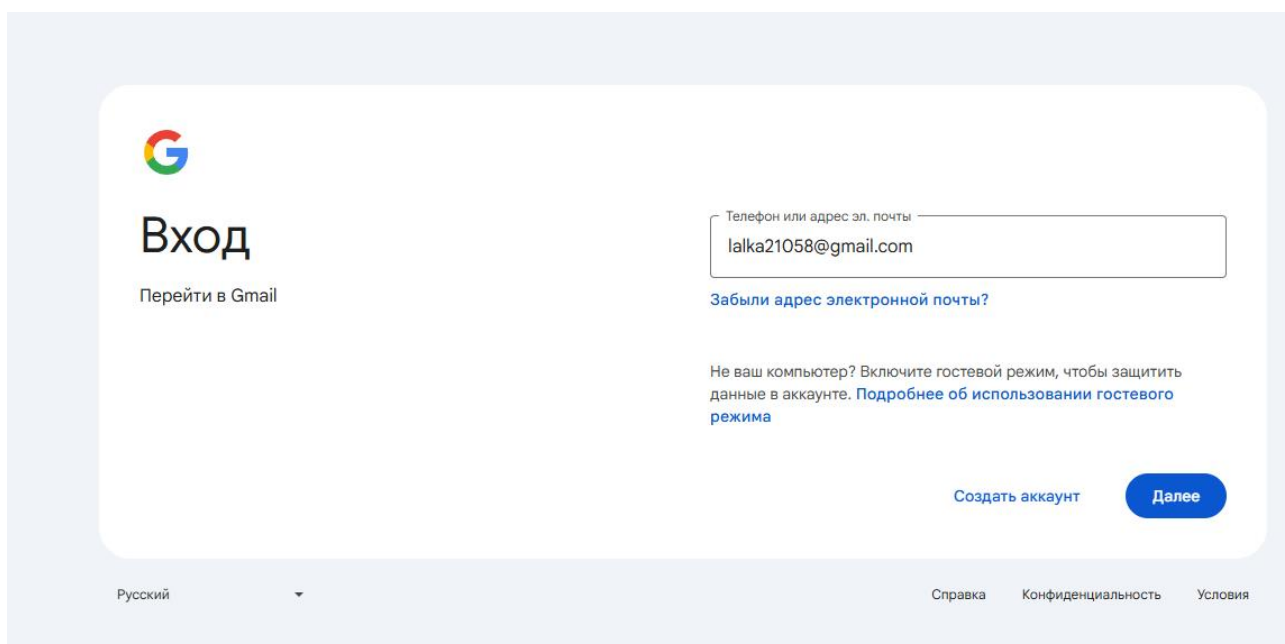


Рисунок 1 - Ввод логина

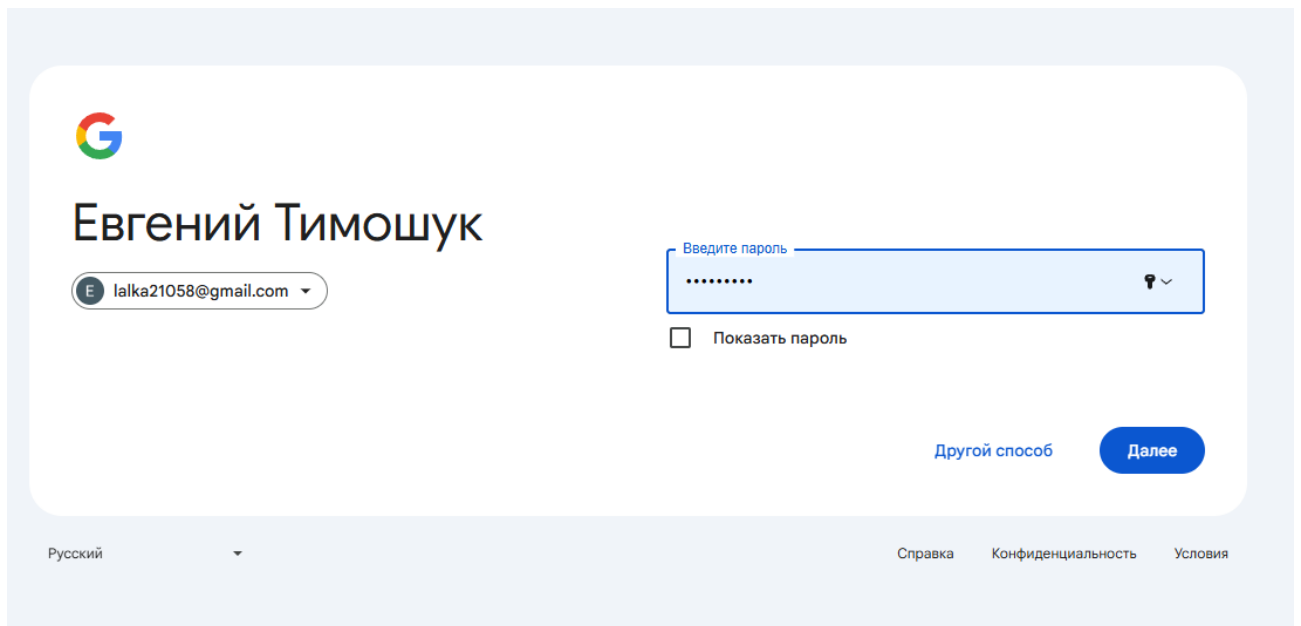


Рисунок 2 - Ввод пароля

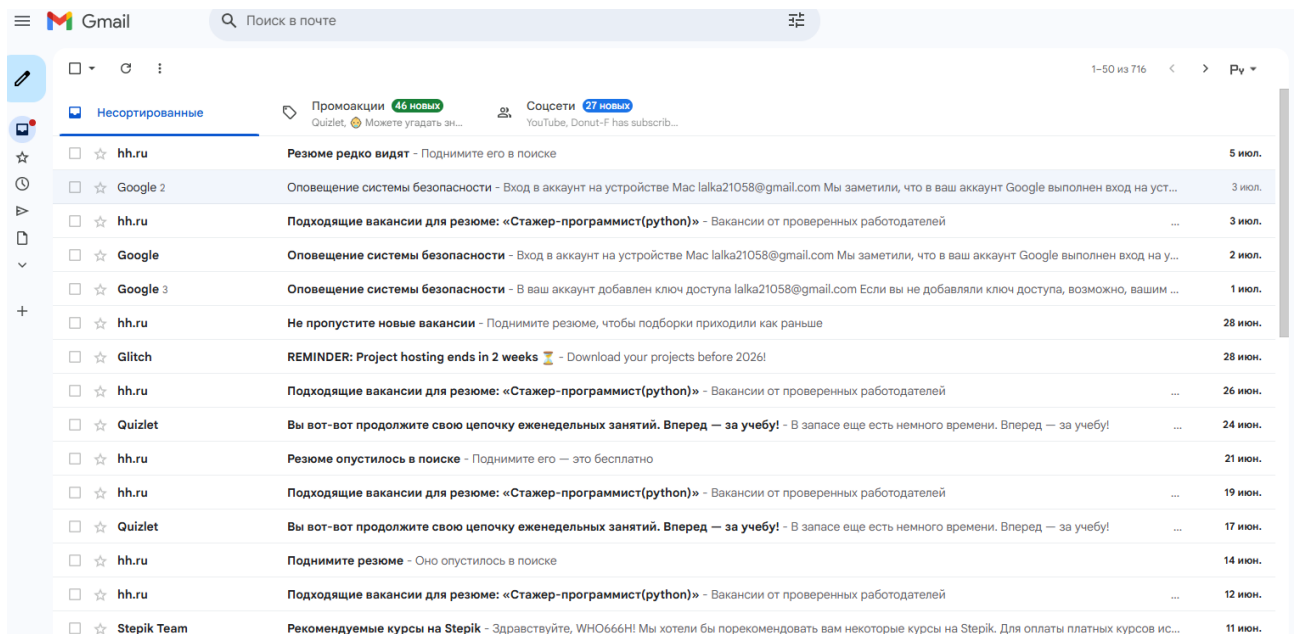


Рисунок 2 - главная страница

3.2 Тестирование всех тестов

Настройки

Общие Ярлыки Папка "Входящие" Аккаунты и импорт **Фильтры и заблокированные адреса**

Следующие фильтры применяются ко всем входящим письмам:

- ☐ Соответствует: **from:(pelmeshka@gmail.com)**
Действия: Никогда не отправлять в спам

Выбрать: **Все**, Ни одного

Экспорт

Удалить

Рисунок 4 - Создание фильтра



← **Имя**

Имя будет изменено во всех сервисах, где используется аккаунт Google. Указанное ранее имя может по-прежнему быть доступно для поиска или показываться в старых сообщениях. [Подробнее...](#)

Имя

Фамилия

Кто может видеть ваше имя

 Эту информацию смогут увидеть любые пользователи, которые будут общаться с вами или просматривать созданный вами контент в сервисах Google. [Подробнее...](#) 

Отмена **Сохранить**

Рисунок 3 - Форма по заполнение ФИО

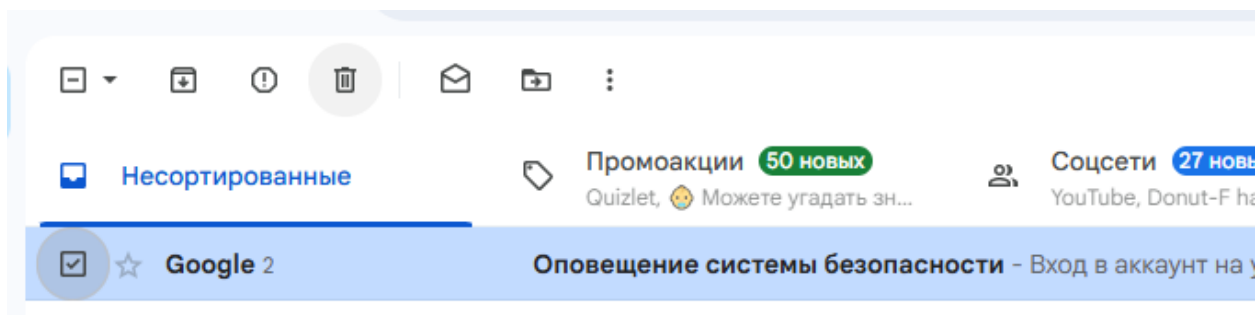


Рисунок 4 - Удаление письма

+ Новый контакт

Контакты (7)








-  fffff
-  Алик Братан
-  Даниил
kulachdv@gmail.com
-  Даниил
kulachdv@gmail.com
-  Даниил
kulachdv@gmail.com
-  Дома
-  Любимая

Рисунок 5 - Добавление Контакта

← Language

Your preferred language for Google services and other languages that you might understand. [Learn more](#)

Changes to your preferred language are reflected on the web. Google might use your language info to show you more relevant content on apps and services. To change the preferred language for mobile apps, go to the language settings on your device.

Preferred language

English
Zimbabwe

Other languages

Русский (Russian)
Added for you

Save

+ Add another language

Рисунок 6 - Смена языка

Новое сообщение

asdadadadadadd@gmail.com

Тема

Рисунок 7 - Отправка письма

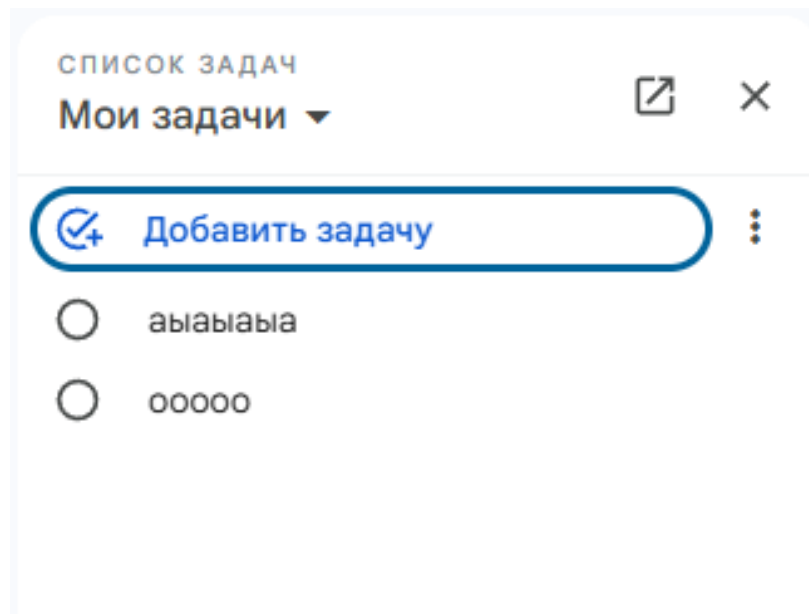


Рисунок 8 - Добавление задачи

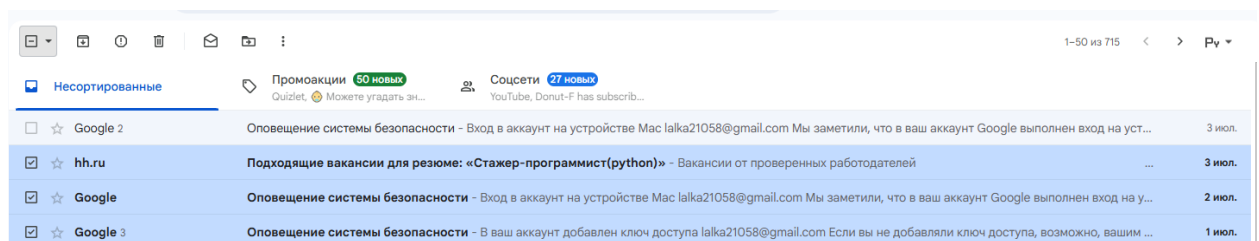


Рисунок 9 - Письма с отметкой

<input type="checkbox"/>	☆	NVIDIA Accounts	Входящие	Аутентификация по адресу эл. почты - Здравствуй! Нажмите ссылку внизу, чтобы подтвердить этот адрес и перейти в NVIDIA GeForce Experienc...	2 мар.
<input type="checkbox"/>	☆	Андрей Павлов	Входящие	Яндекс.Диск: доступ к папке «3388» - Здравствуй! Андрей Павлов (rawlov.andrusha2014@yandex.ru) предоставляет вам полный доступ к папке...	11.09.2023
<input type="checkbox"/>	☆	Stepik Team	Входящие	Рекомендуемые курсы на Stepik - Здравствуй! WHO666H! Мы хотели бы порекомендовать вам некоторые курсы на Stepik. Для оплаты платных...	11 июн.
<input type="checkbox"/>	☆	QR-Code.io	Входящие	Добро пожаловать в QR-Code.io - Для получения дополнительной помощи посетите раздел часто задаваемых вопросов или свяжитесь с нами н...	1 февр.
<input type="checkbox"/>	☆	NVIDIA Accounts	Входящие	Подтверждение адреса эл. почты в NVIDIA - Здравствуй! Вы указали этот адрес электронной почты для входа в учетную запись NVIDIA. Нажмит...	16.08.2023
<input type="checkbox"/>	☆	Группа 3388 каф. МО.	Входящие	Приглашение в группу "Группа 3388 каф. МОЭВМ (ФКТИ)" - Здравствуй! lalka21058@gmail.com! onextopplay@gmail.com приглашает Вас присоед...	20.09.2023
<input type="checkbox"/>	☆	Mathway	Входящие	Добро пожаловать в Mathway - Здравствуй! Добро пожаловать в Mathway, ваш личный помощник при выполнении домашних заданий по мат...	08.07.2023
<input type="checkbox"/>	☆	QR-Code.io	Входящие	[Напоминание] Не забудьте скачать QR-код - Здравствуй! Ваши друзья в QR-Code.io. QR-Code.io. support@qr-code.io © 2025, QR-Code.io. Вс...	4 февр.
<input type="checkbox"/>	☆	QR-Code.io	Входящие	RE: [Требуется действие] Ваш деактивированный QR-код мог быть отсканирован! - Здравствуй! Ваши друзья в QR-Code.io. QR-Code.io. su...	6 мар.
<input type="checkbox"/>	☆	QR-Code.io	Входящие	[Требуется действие] Ваш деактивированный QR-код мог быть отсканирован! - Здравствуй! Ваши друзья в QR-Code.io. QR-Code.io. suppor...	3 мар.
<input type="checkbox"/>	☆	hh.ru	Входящие	Спасибо за ваше резюме! - Здравствуй! Евгений! Ваш отклик на вакансию Сотрудник склада Озон (Шушары) успешно зарегистрирован и на...	30.06.2024
<input type="checkbox"/>	☆	ASOS	Входящие	Спасибо за ваш заказ! - Здравствуй! Женя! Мы получили твой заказ. ASOS - Открой мир моды онлайн. Заказ размещен. Здравствуй! Женя! ...	02.01.2022
<input type="checkbox"/>	☆	Команда Gmail	Входящие	В Gmail всё по полочкам - Здравствуй! Женя! Пользоваться Gmail удобно. Видео о почте. Категории входящих сообщений. Gmail группирует в...	24.05.2015

Рисунок 10 - Письма с пометкой "Здравствуйте"

UML – диаграмма

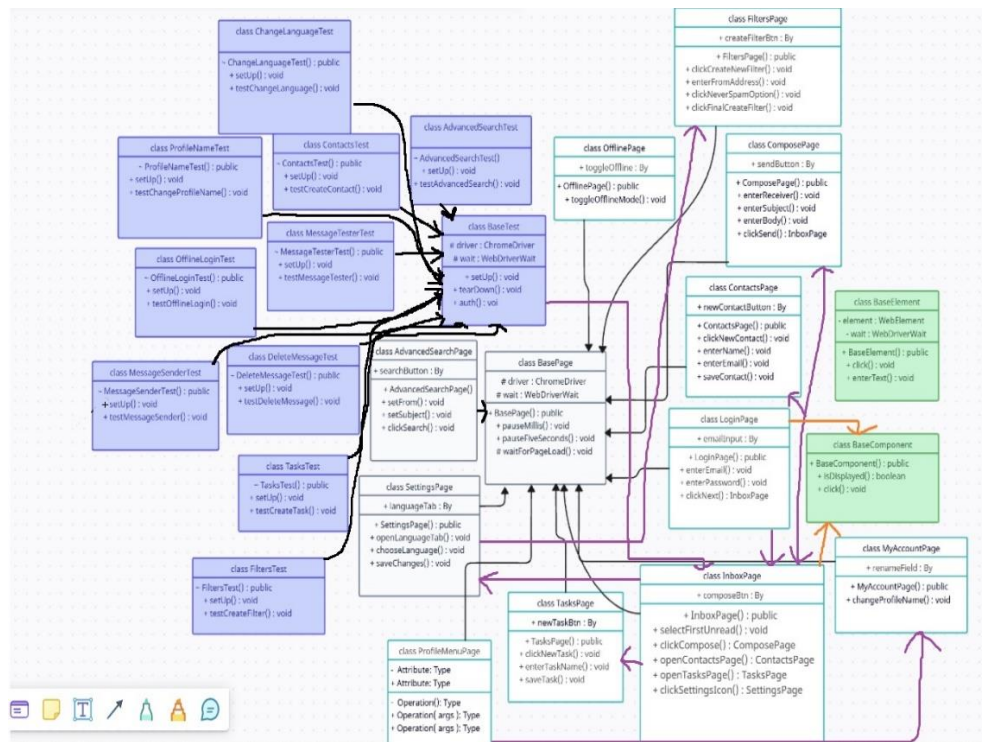


Рисунок 1 - UML

Цветовая кодировка и типы связей:

- синие блоки – тест-классы (JUnit/тестовые сценарии);
- белые блоки – Page Object-классы (страницы);
- зелёные блоки – базовые/утилитные классы;
- чёрные стрелки – наследование (extends);
- красные стрелки – ассоциации: методы одного класса возвращают объект другого;
- жёлтые стрелки – зависимости (static util-вызовы, использование констант).

ЗАКЛЮЧЕНИЕ

Над проектом автоматизации тестирования Gmail мы работали втроем, организовав процесс через GitHub с четким разделением обязанностей. Я отвечал за модуль авторизации и функционал работы с письмами, включая удаление сообщений и пометку прочитанных. Второй участник взял на себя реализацию тестов для настроек — фильтров, смены языка интерфейса и расширенного поиска. Третий участник разрабатывал тесты для работы с контактами, задачами, offline-режимом и отправкой писем.

Для координации мы создали общий репозиторий с единой структурой проекта, где сразу установили ключевые соглашения: именование методов в camelCase, обязательное использование Page Object Model и вынесение стандартных локаторов в BaseComponent. Каждый день мы проводили короткие 15-минутные созвоны для синхронизации прогресса, а для управления кодом использовали Git Flow с обязательным ревью перед мерджем в основную ветку.

В процессе столкнулись с типичными проблемами командной разработки. Конфликты версий решали через систему feature-веток и pull-request'ов. Чтобы предотвратить расхождения в стиле кодирования, внедрили Checkstyle с общим конфигурационным файлом. При обнаружении дублирующихся методов сразу выносили их в BasePage, а проблемы с зависимостями фиксировали через жесткое закрепление версий в pom.xml.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Apache Maven. Maven Guides [Электронный ресурс] // Apache Software Foundation. – URL: <https://maven.apache.org/guides/index.html> (дата обращения: 05.07.2025).
2. Google. Gmail [Электронный ресурс] // Google. – URL: <https://mail.google.com/> (дата обращения: 05.07.2025).
3. Oracle. Java Documentation [Электронный ресурс] // Oracle. – URL: <https://docs.oracle.com/en/java/> (дата обращения: 05.07.2025).
4. Potapov, R. Practika [Электронный ресурс] // GitHub. – URL: <https://github.com/rodionpotapov/Practika/tree/main> (дата обращения: 05.07.2025).
5. Selenide. Official Documentation [Электронный ресурс] // Selenide. – URL: <https://selenide.org/documentation.html> (дата обращения: 05.07.2025).