**Домашняя работа**

**Задание №1**

**Пример интеграционного тестирования**

Для того, что бы начать интеграционное тестирование приложения электронной почты нам необходимо декомпозировать её элементы.

Разделим их на «Страница входа», «Главная страница», кнопки “Отправить письмо”, “Удалить письмо”, разделы “Удаленные”, “Отправленные”, “Входящие”, “Черновики”.

1. **Проверка входа в систему**

а) Начнём со «Страницы входа», проверим, что страница входа работает корректно и позволяет пользователю войти в систему.

б) Далее проверяем, что после входа в систему пользователь перенаправляется на главную страницу приложения.

2. **Главная страница**

Проверяем верность верстки главной страницы, правильное расположение кнопок, цвет, соответствие требованиям заказчика.

1. **Проверка отправки письма**

а) Проверяем, что кнопка "Отправить письмо" работает корректно и позволяет пользователю отправить новое письмо.

б) Необходимо проверить, что после отправки письма оно появляется в разделе "Отправленные".

1. **Проверка раздела "Черновики"**

а) Проверяем, что раздел "Черновики" отображает все черновики писем, которые пользователь начал создавать, но не отправил.

б) Проверяем, что после сохранения черновика письма он появляется в разделе "Черновики".

1. **Проверка исправления ошибки с удаленными письмами**

а) Проверяем, что после исправления ошибки удаленные письма больше не попадают в раздел "Отправленные"

б) Также проверяем, что удаленные письма корректно отображаются в разделе

"Удаленные".

1. **Проверка взаимодействия между разделами**

а) Проверяем, что при переходе между разделами "Входящие", "Отправленные", "Удаленные" и "Черновики" приложение корректно отображает соответствующие письма.

б) Проверяем, что при удалении письма из раздела "Входящие" оно корректно перемещается в раздел "Удаленные".

1. **Проверка работы приложения в различных сценариях**

а) Проводим, что то наподобие нагрузочного тестирования. Проверяем, что приложение корректно работает при наличии большого количества писем в каждом разделе.

б) Проверяем, что приложение корректно работает при одновременном использовании несколькими пользователями.

**Пример дымового тестирования**

Примеры дымового тестирования приложения электронной почты включают проверку основных функциональных возможностей приложения, таких как:

**1. Проверка входа в систему:**

а) Проверяем, что страница входа работает корректно и позволяет пользователю войти в систему.

б) Проверяем, что после входа в систему пользователь перенаправляется на главную страницу приложения.

**2. Проверка отправки письма:**

а) Проверяем, что кнопка "Отправить письмо" работает корректно и позволяет пользователю отправить новое письмо.

б) Проверяем, что после отправки письма оно появляется в разделе "Отправленные".

**3. Проверка удаления письма:**

а) Проверяем, что кнопка "Удалить письмо" работает корректно и позволяет пользователю удалить выбранное письмо.

б) Проверяем, что после удаления письма оно появляется в разделе "Удаленные".

**4. Проверка "Черновики":**

а) Проверяем, что раздел "Черновики" отображает все черновики писем, которые пользователь начал создавать, но не отправил.

б) Проверяем, что после сохранения черновика письма он появляется в разделе "Черновики".

**5. Проверка исправления ошибки с удаленными письмами:**

а) Проверяем, что после исправления ошибки удаленные письма больше не попадают в раздел "Отправленные".

б) Проверяем, что удаленные письма корректно отображаются в разделе "Удаленные".

**6. Проверка взаимодействия между разделами:**

а) Проверяем, что при переходе между разделами "Входящие", "Отправленные", "Удаленные" и "Черновики" приложение корректно отображает соответствующие письма.

б) Проверяем, что при удалении письма из раздела "Входящие" оно корректно перемещается в раздел "Удаленные".

**Пример ре-теста приложения**

Ре-тест приложения - это процесс повторного тестирования функциональности приложения после внесения изменений или исправления ошибок. Он проводится для того, чтобы убедиться, что изменения не повлияли на работоспособность других частей приложения и не вызвали новых ошибок. Ре-тест может включать в себя повторное выполнение предыдущих тестов, а также проверку новых функциональных возможностей.

Фактически, мы можем использовать сценарий тестирования из примера выше.

**Пример нефункционального тестирования**

Нефункциональное тестирование – это такой тип тестирования, при котором проверяются нефункциональные аспекты программного приложения.

К числу основных можно отнести:

1. нагрузочное тестирование (данный вид тестирования позволяет убедиться, что приложение электронной почты способно обрабатывать большие нагрузки и работать стабильно при больших объемах данных).
2. проверка безопасности (тестирование защиты от несанкционированного доступа к аккаунту пользователя, проверка защиты от фишинга и спама).
3. проверка производительности (например, время загрузки приложения не должно превышать 3 секунд при одновременном доступе к нему до 1000 пользователей).
4. тест совместимости (программное обеспечение должно быть установлено на всех версиях Windows и Linux).
5. тест доступности (проверка доступности приложения для пользователей с ограниченными возможностями (например, для пользователей с нарушениями зрения или слуха).
6. тест надежности (проверка стабильности работы приложения при больших нагрузках, а также проверка возможности восстановления после сбоев).
7. проверка соответствия требованиям (например, законодательства).

**Задание №2**

**Вопрос №1**

Для начала нам нужно дать понятие регрессионного тестирования. Регрессионное тестирование - это процесс проверки того, что изменения в коде не привели к появлению новых ошибок или не нарушили работу уже существующих функциональных возможностей.

В нашем случае добавлена новая функциональность – появился раздел «Спам».

При добавление новой функциональности могут возникнуть риски появления ошибок, нарушится работа уже существующих разделов («Входящие», «Отправленные», «Удаленные» и другие).

Могут появиться баги и в новом разделе – «Спам». Например, ошибки связанные с не корректной обработкой спам-сообщений.

Также, изменения в коде могут повлиять на производительность приложения электронной почты, может увеличиться время загрузки страницы, снизиться скорости работы приложения.

**Вопрос №2**

При рассмотрении данного случая, может показаться, что функциональность не изменилась, поменялось только название кнопки. Однако, даже при изменении названия раздела, могут возникнуть риски, которые необходимо проверить с помощью регрессионного тестирования.

Например, изменения в названии раздела могут привести к путанице у пользователей, что может повлиять на их взаимодействие с приложением.

По логике работы электронной почты, когда пользователь удаляет сообщение из раздела «Удаленные», оно перемещается в раздел «Корзина».

Если, механизм удаления сообщений использует название раздела для определения, куда переместить сообщение после удаления, то переименование раздела может привести к неправильной работе этого механизма.

Таким образом, при переименовании раздела «Удаленные» на раздел «Корзина» может нарушиться механизм удаления сообщений.

**Вопрос №3**

В случае если на «Странице входа» устранен ранее обнаруженный дефект, нам также необходимо провести регрессионное тестирование, поскольку, изменения в коде могли привести к возникновению новых ошибок, устранение дефекта может привести к нарушению работы других частей приложения.

Поэтому, для проверки корректности работы необходимо проведение регрессионного тестирования.

**Задание №3**

В первую очередь на новом билде(релизе) приложения желательно провести интеграционное тестирование. Интеграционное тестирование - это фаза тестирования программного обеспечения, при которой отдельные программные модули объединяются и тестируются в группе.

Необходимо убедиться, что все модули программного обеспечения работают корректно на новом билде, исправлены ошибки, которые могут влиять на работу других частей приложения.

Поэтому, в первую очередь на новом билде приложения необходимо провести интеграционное тестирование, чтобы убедиться в его общей работоспособности и отсутствии критических ошибок. После этого можно провести функциональное тестирование для более детальной проверки новых функциональных возможностей.

**Задание №4**

В результате добавления раздела “Черновик” перестала корректно работать кнопка “Удалить письмо”. Какой вид тестирования позволит обнаружить нам данный дефект?

Для выявления данного дефекта нам поможет регрессионное тестирование, которое проводится после изменений в коде, исправления ошибок. Оно помогает нам выяснить, исправлен ли дефект, не появились ли новые ошибки, корректно ли работают модули приложения, не нарушена функциональность приложения.

По условиям задачи: «В новом релизе был добавлен раздел “Черновики” и исправлена ошибка, при которой удаленное письмо попадало в раздел “Отправленные”. В результате добавления раздела “Черновик” перестала корректно работать кнопка “Удалить письмо”».

**Пример проведения регрессионного теста для нашего случая:**

**1. Проверка входа в систему:**

а) Проверяем, что страница входа работает корректно и позволяет пользователю войти в систему.

б) Проверяем, что после входа в систему пользователь перенаправляется на главную страницу приложения.

**2. Проверка отправки письма:**

а) Проверяем, что кнопка "Отправить письмо" работает корректно и позволяет пользователю отправить новое письмо.

б) Проверяем, что после отправки письма оно появляется в разделе "Отправленные".

**3. Проверка удаления письма:**

а) Проверяем, что кнопка "Удалить письмо" **работает корректно** и позволяет пользователю удалить выбранное письмо.

б) Проверяем, что после удаления письма оно появляется в разделе "Удаленные".

**4. Проверка "Черновики":**

а) Проверяем, что раздел "Черновики" отображает все черновики писем, которые пользователь начал создавать, но не отправил.

б) Проверяем, что после сохранения черновика письма он появляется в разделе "Черновики".

**5. Проверка исправления ошибки с удаленными письмами:**

а) Проверяем, что после исправления ошибки удаленные письма больше не попадают в раздел "Отправленные".

б) Проверяем, что удаленные письма корректно отображаются в разделе "Удаленные".

**6. Проверка исправления ошибки с кнопкой «Удалить письмо»**

Проверяем, что кнопка "Удалить письмо" работает корректно, письмо удаляется и перемещается в раздел "Удаленные".

**7. Проверка взаимодействия между разделами:**

а) Проверяем, что при переходе между разделами "Входящие", "Отправленные", "Удаленные" и "Черновики" приложение корректно отображает соответствующие письма.

б) Проверяем, что при удалении письма из раздела "Входящие" оно корректно перемещается в раздел "Удаленные".