Лабораторная работа № 8
Разработка плана и проведение тестирования пользовательского интерфейса
Поиск и документирование дефектов

Цель: разработать план и провести тестировать пользовательского интерфейса приложения и описать найденные дефекты.

План занятия:

1. Изучить теоретические сведения.
2. Выполнить практическое задание по лабораторной работе.
3. Оформить отчёт и ответить на контрольные вопросы.

Теоретические сведения

Для любого приложения выполняется тестирование графического интерфейса пользователя

Перечень основных GUI проверок для всего приложения

Название проверки	Описание проверки
1. Правописание	Лексические, грамматические и пунктуационные ошибки
2. Расположение и выравнивание	Выравнивание по левому или правому краю (в зависимости от требований приложения), отступы, идентичность расстояний между названием и полем. Корректное расположение текста, длинный текст не выходит за границы поля при вводе.
3. Длинные названия	Длинные названия корректно обрезаются с помощью многоточия в конце, при наведении возникают хинты с полнотекстовым вариантом.
4. Соответствие названий форм / элементов GUI их назначению	Проверка названий форм / элементов GUI с точки зрения их смысловой нагрузки.
5. Унификация (стиля, цвета, шрифта, названий)	Единообразие цвета, шрифта, размеров (высоты/ширины), выравнивания полей, названий полей, категорий меню и др. в рамках всего приложения.
6. Эффект «нажатия»	Изменение вида ссылок, кнопок, позиций меню и др. при наведении курсора. Изменение вида курсора при наведении на ссылки, кнопки, позиции меню и др.
7. Хинты	Проверка всплывающих подсказок с точки зрения правописания, выравнивания, соответствия назначению и др.

8. Сообщения об успешном / неуспешном завершении действия, о подтверждении действия	Проверка верхней панели (логотипа и названия) формы с сообщением. Если присутствует кнопка «Отмена», то в правом верхнем углу формы с сообщением присутствует «крестик» для альтернативной возможности закрыть форму. Сообщения о подтверждении удаления по умолчанию активированы на кнопку «нет».
9. Изменение размеров окна, изменение масштаба страницы	Появление скроллинга при уменьшении размера окна. Сохранение взаимного расположения элементов при уменьшении окна, изменении масштаба). Перераспределение элементов с сохранением пропорций при изменении масштаба страницы.

Дефекты, обнаруженные тестировщиком, должны быть корректно и понятно описаны, чтобы разработчик смог воспроизвести данный дефект и устранить его. Описание каждого дефекта сохраняется в специализированной – багтрэкинговой – системе (например, JIRA, Bugzilla, Mantis, Redmine и др.) или в предварительно созданном в программной среде Microsoft Excel файле (пример приведен в таблице 1).

Таблица 1 – Пример описания дефекта

№	Название дефекта	Важность	Алгоритм воспроизведения	Фактический результат	Ожидаемый результат	Приложение	Примечание
39	Администраторская часть: Файлы: Выбор файла: Ссылка "Скачать файл": Администратор не имеет возможности скачать загруженные студентом файлы.	Critical	Шаги по воспроизведению: 1. Входим на web сайт. 2.Проходим авторизацию: admin/admin. 3. Выбираем в меню категорию "Файлы". 4.Выбираем подкатегорию меню "Выбор файла". 5.Выбираем в поле "Обзор файла" любой доступный файл. 6.Нажимаем на ссылку "Скачать файл".	Переход к странице "HTTP Status 404".	Происходит скачивание выбранного файла.	39.png	REQ-26

Описание дефекта включает следующие обязательные поля:

1. Headline – название дефекта.

2. Severity – степень критичности (важность дефекта).
3. Description – алгоритм воспроизведения.
4. Result – фактический результат.
5. Expected result – ожидаемый результат.
6. Attachment – прикреплённые файлы (приложение).

В багтрэкинговых системах для каждого дефекта автоматически генерируется его уникальный номер, в случае использования Microsoft Excel номер дефекту необходимо присваивать вручную.

Требование спецификации, которое нарушает обнаруженный дефект, можно дополнительно вынести в примечание.

Дополнительно в описании дефекта может быть указана Priority – степень срочности исправления дефекта разработчиком.

Рассмотрим подробно каждую категорию описания дефекта.

Headline (название дефекта). Цель составления заголовка дефекта – предоставить краткую и в тоже время понятную информацию о том, где, что и в результате чего произошло. Характеристиками качественного заголовка являются краткость, информативность, точная идентификация проблемы.

Заголовок дефекта должен отвечать на три вопроса:

1. Где? В каком месте интерфейса пользователя находится проблема.

В данной части заголовка следует также дополнительно указать особенности теста, если это поможет разобраться в проблеме (версия операционной системы, браузера, сторонних приложений, которые имеют отношение к тестируемому программному средству).

2. Что? Что происходит или не происходит согласно спецификации или представлению о нормальной работе программного продукта. При этом необходимо указывать на наличие проблемы, а не на ее содержание (его указывают в описании). Если содержание проблемы варьируется, все известные варианты указываются в описании.

3. Когда (при каких условиях)? В какой момент работы программы или по наступлению какого события проблема проявляется.

Пример: в приложении есть диалог «Преобразовать данные» с кнопкой «Преобразовать». При нажатии этой кнопки появляется сообщение об ошибке «Ошибка 315». Заголовок дефекта по описанной методике составляется так:

Где?: Диалог «Преобразовать данные».

Что?: Показывается сообщение об ошибке.

Когда?: При нажатии кнопки «Преобразовать».

Итоговый заголовок будет иметь следующий вид: Диалог "Преобразовать данные" показывает сообщение об ошибке при нажатии кнопки "Преобразовать". Уберём лишние слова, добавим код ошибки для удобства поиска: Диалог «Преобразовать данные»: сообщение об ошибке 315 при нажатии кнопки «Преобразовать».

Если сформулировать заголовок по формуле Где? Что? Когда (при каких условиях)? трудно, то можно воспользоваться следующим алгоритмом действий:

1. Пропустите заголовок дефекта.
2. Напишите описание дефекта, фактический и ожидаемый результаты.
3. Выделите ключевые моменты, руководствуясь формулой Где? Что? Когда (при каких условиях)?
4. Сложите эти ключевые моменты вместе.
5. То, что получится в итоге, и будет составлять заголовок дефекта.

Severity (степень критичности). Степень критичности (серьезности, важности) показывает степень ущерба, который наносится проекту существованием дефекта. В общем случае выделяют следующие градации критичности дефектов (таблица 2): Critical (критический), Major (значительный), Average (средней значимости), Minor (незначительный), Enhancement (предложение по улучшению). Description (алгоритм воспроизведения). Цель составления алгоритма воспроизведения дефекта – последовательно описать шаги для повторения дефекта. Description должен быть оформлен в виде списка перечисления действий:

1. *Шаг #1.*
2. *Шаг #2.*
3. *...*
4. *Шаг #n.*

В случае, если для воспроизведения дефекта требуется ряд начальных условий (например, должен быть создан определенный набор документов, пользователь или группа пользователей с особыми правами), то эти предусловия должны быть вынесены в начало описания:

Предусловия.

1. *Шаг #1.*
2. *Шаг #2.*
3. *...*
4. *Шаг #n.*

Таблица 2 – Степени критичности дефектов

Severity	Описание	Примеры
Critical (критический)	Функциональная ошибка, которая блокирует работу части функционала или всего приложения. Функциональная ошибка, которая нарушает ключевую (с точки зрения конечного пользователя или бизнеса заказчика) функциональность приложения.	Заблокирована вкладка категория меню). Неправильно подсчитывается итоговая сумма при вводе скидочного промокода. Раскрывается конфиденциальная информация.

Major (значительный)	Функциональная ошибка, которая нарушает нормальную работу приложения, но не блокирует работу части функционала в целом.	Невозможно загрузить видео-файлы на персональной страничке. Не работает система e-mail нотификации. Не работает интеграция с социальными сетями. Необходимо перезапускать приложение при выполнении типичных сценариев работы
Average (средней значимости)	Не очень важная функциональная ошибка. Критичные дефекты GUI.	Не работает сортировка. «Уехал» текст за пределы окна.
Minor (незначительный)	Редко встречающиеся незначительные функциональные дефекты. 90 % дефектов GUI.	Введены необязательные для заполнения поля, которых нет в спецификации. Грамматические, пунктуационные ошибки.
Enhancement (предложение по улучшению)	Функциональные предложения, советы по улучшению дизайна (оформления), навигации и др. Такой дефект является необязательным для исправления.	Добавить кнопку «Наверх» на длинных формах. Увеличить размер шрифта.

При составлении Description необходимо следовать приведенным ниже рекомендациям.

Description – это четкий алгоритм, в котором приветствуются короткие, понятные фразы и нумерация.

Нельзя использовать личные предложения формата «Я думаю, что так будет лучше», «Я зашел на страницу…» и т.д.

Можно использовать специальные символы «+», «=», «<>» которые помогут сделать подобие навигации: File > Open, DOC + XLS. Однако не рекомендуется писать шаги в строку через символы перехода: это затрудняет восприятие дефекта. Следует указывать специфичные условия воспроизведения дефекта, если таковые имеются. Например, под каким пользователем вы работаете (если это важно).

Описание предложений по улучшению должно быть максимально полным и аргументированным.

Result (фактический результат). Цель написания Result – четко описать полученный результат.

Expected result (ожидаемый результат). Цель написания Expected result – привести аргументы разработчикам, как именно должно работать приложение.

В Expected result должно быть четкое обоснование, почему именно так должно работать приложение. Лучше всего, если в нем приведена ссылка на конкретный пункт спецификации.

Если в Expected result приводится ссылка на спецификацию, сам тестировщик дополнительно цитирует текст спецификации, чтобы сократить время разработчика на анализ документа и однозначно указать на способ решения проблемы.

Если функция работает, но некорректно, то в Expected Result обязательно должно быть описание того, как она должна работать корректно.

Если сделана ошибка в надписи или интерфейсе проекта, необходимо грамотно и полностью указать, как она должна быть исправлена – написать надпись без ошибки или описать требуемые изменения интерфейса.

Expected Result всегда следует заполнять. Не стоит полагаться на очевидность представлений о правильном поведении приложения.

Текст Expected result рекомендуется писать законченными полными безличными предложениями.

Attachment (приложения). Attachment – это прикрепленный к дефекту файл, дополняющий описание: скриншот, файлы, необходимые для воспроизведения дефекта, логи программы, видео ошибки и т.д. Attachment является вспомогательным средством передачи информации о проблеме. Для всех GUI дефектов attachment обязателен.

Если к дефекту прикрепляется файл, об этом обязательно должно быть указано в описании дефекта («See the file/screenshot/log/video attached»). Прикрепленный файл не должен быть слишком большим по размеру (особенно это касается видео: до 10 мегабайт), а также должен относиться именно к описанной ошибке (например, из лога приложения стоит скопировать в прикрепляемый файл только данные об ошибке). Формат файла скриншота – PNG. Имя файла скриншота рекомендуется делать числовым, нейтральным – 1.png, 25.png и т.п.

Скриншот должен содержать следующие элементы: сама ошибка, выделение места ошибки, стрелка к прямоугольнику, описание ошибки: «Наблюдаемый результат» и/или «Ожидаемый результат». Текст на скриншоте также необходимо выделить: обвести в прямоугольник и набрать шрифтом, заметно отличающимся от шрифта программы. Качественно подготовленный скриншот должен давать возможность понять смысл дефекта без необходимости читать его описание (рисунок 1).

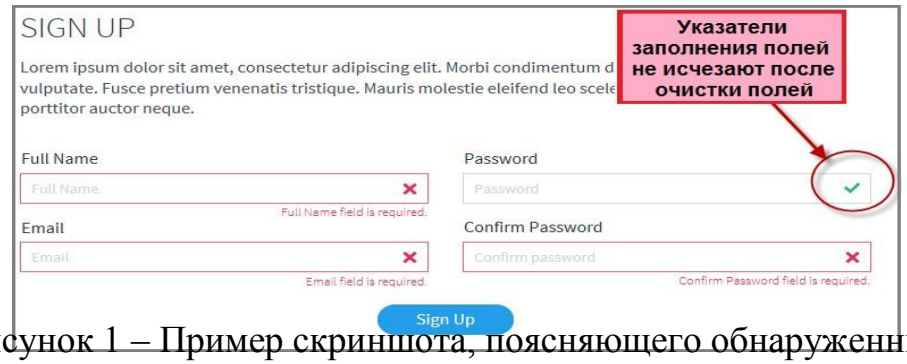

Рисунок 1 – Пример скриншота, поясняющего обнаруженный дефект

Делать снимок всего окна программы необязательно: на снимке должна быть видна ошибка и место, в котором она находится. Если для понимания дефекта необходим контекст, то помимо собственно ошибки фиксируют необходимую

информацию (браузер, в котором открыто web приложение, всё окно программы в фоне с названием диалогового окна, где эта ошибка появилась). Если необходимо привести снимки нескольких страниц проекта, связанных между собой, лучше сделать это на одном скриншоте, совместив изображения по горизонтали и при необходимости отметив стрелками переходы значений, полей и т.п.

Рекомендации по описанию дефектов.

Часто сообщение об ошибке превращается в сокращённую запись только основных действий, необходимых для воспроизведения ошибки, опуская все несущественные. Но, будучи незнакомым с внутренней структурой приложения, тестировщик не может знать, какие из выполненных им действий наиболее существенны для диагностирования данной ошибки. Пренебрегая действиями, которые кажутся незначительным, повышается риск потери важной информации. Лучший способ избежать этой проблемы состоит в том, чтобы просто перечислить все действия, которые необходимы для воспроизведения ошибочного поведения, начиная с открытия нужной формы в проекте.

Если есть подозрение на повторение дефекта в нескольких модулях проекта, этот факт нужно исследовать еще до внесения дефекта и при его описании указать все места, где дефект воспроизводится.

Дефект не должен содержать фразу: «Это не работает», дефект должен показать, что и при каких условиях не работает.

Чтобы внести дефект, его следует воспроизвести минимум два раза, причем начиная с самых нейтральных условий воспроизведения, и только после гарантированного повторения описать последовательность действий.

Нельзя не описывать дефекты только потому, что их не получается воспроизвести. Факт невозможности выяснить причину дефекта в таком случае обязательно должен быть указан в описании дефекта.

Дефекты целесообразно группировать, однако делать это необходимо в соответствии с приведенными ниже правилами.

GUI дефекты могут группироваться в один по признаку формы, на которой они находятся, т.е. если одна форма содержит несколько GUI дефектов с одинаковым уровнем Severity, то их можно объединить.

Функциональные дефекты группируются в том случае, если речь идет об однотипных дефектах, которые воспроизводятся в различных модулях, страницах или полях (например, динамическое обновление не работает в модулях 1, 2 и 4 или отсутствует валидация на спецсимволы на всех полях страницы).

Группировка функциональных дефектов по признаку формы, на которой они найдены, не применяется.

Недопустимо объединять в один дефекты разного типа, например, функциональные и GUI. В таком случае пишутся несколько дефектов на каждый тип.

Рекомендации по хорошему описанию дефектов:

1. Шаги воспроизведения, фактический и ожидаемый результаты должны быть подробно описаны.

2. Дефект должен быть понятно описан (с использованием общеупотребимой

лексики, точных названий программных средств).

3. Необходимо давать ссылку на соответствующее требование, к нарушению которого приводит фактический результат работы программного средства.

4. Если существует какая-либо информация, которая поможет быстрее обнаружить или исправить дефект, необходимо сообщить эту информацию.

5. Окружение (ОС, браузер, настройки и т.п.), под которым возникла ошибка, должно быть четко указано.

6. Создавать дефект и описывать его необходимо сразу же, как только он был обнаружен. Откладывание «на потом» приводит к риску забыть о дефекте или и каких-либо деталях его воспроизведения. Несвоевременное создание дефекта не позволяет проектной команде реагировать на ее обнаружение в реальном времени.

7. После описания дефекта необходимо еще раз перечитать его: убедится, что все необходимые поля заполнены, и все написано верно.

Помимо собствено описания дефектов результаты тестирования вносят в рабочую тестовую документацию (Acceptance Sheet, Test Survey, Test Cases). Для этого напротив выполненной проверки указывают степень критичности обнаруженного дефекта, его номер и заголовок. Если по результатам конкретной проверки выявлено несколько дефектов, то перечисляют номера всех дефектов, а в качестве степени критичности и заголовка указывают наиболее серьезный дефект (рисунок 2).

Function	Result	Issues	Comments
Форма регистрации			
Поле "Имя"	Critical	15,16	Русскоязычное приложение не поддерживает ввод русских букв.
Поле "Фамилия"	Critical	15,16	Русскоязычное приложение не поддерживает ввод русских букв.
Поле "Эл. адрес или номер моб.телефона"	Major	21	Отсутствует проверка уникальности вводимых данных.
Поле "Повторно введите ваш эл. адрес или номер моб.телефона"	Average	24	Допускается оставить поле, обязательное для заполнения, пустым.
Поле "Новый пароль"	OK		

Рисунок 2 – Фрагмент Acceptance Sheet с внесенными после тестирования дефектами

Задание:

Используя тест-кейсы, провести функциональное тестирование приложения File Searcher.

Результаты тестирования оформить по образцу:
Project
Author

ID	Summary	Component	Severity	Priority	Discription	Steps to Reproduse	Result	Expected Result	Reproducible	Symptom	Attachment

Отчет должен содержать:

1. наименование и цель лабораторной работы;
2. описание дефектов по выданным ранее тест-кейсам.

Практическое задание:

1. Выбрать реальный web – сайт (например сайты учебных заведений Гродненской области), выделить в нем модули.
2. Разработать 20 и более тестовых проверок для выбранного объекта реального

мира с указанием тестируемого модуля и глубины тестового покрытия (Smoke, MAT, AT).

3. Сформулировать по два возможных дефекта на каждый уровень Severity (Critical, Major, Average, Minor, Enhancement) для выбранного объекта реального мира.

4. Описать по одному дефекту на каждый уровень Severity (Critical, Major, Average, Minor, Enhancement) для выбранного объекта реального мира.

5. Протестировать web приложение в соответствии с составленной ранее тестовой документацией.

6. Описать все найденные дефекты в отчете о дефектах в среде Microsoft Excel.

7. В отчете о дефектах указать номер тестируемой сборки, название приложения, период времени тестирования, ФИО тестировщика, тестовое окружение (операционная система, браузер).

8. Для каждого дефекта указать его порядковый номер, заголовок, важность, алгоритм воспроизведения, фактический результат, ожидаемый результат, приложение, примечание.

9. Для каждого дефекта обязательно сделать скриншоты.

10. В рабочую тестовую документацию внести результаты тестирования с указанием напротив соответствующей проверки степени критичности обнаруженного дефекта, его номера и заголовка.

11. Оформить отчет и защитить лабораторную работу.

Содержание отчета:

1. Цель работы.

2. Отчет о результатах тестирования выбранного объекта реального мира с перечислением тестовых проверок, сформулированных дефектов на каждый уровень Severity, описания дефектов.

3. Отчет о найденных дефектах web приложения.

4. Рабочая тестовая документация с внесенными дефектами web приложения.

5. Выводы по работе.

Контрольные вопросы:

1. Что такое дефект?
2. Какие характеристики необходимо указать при описании дефекта?
3. Что такое Headline/Summary в описании дефекта?
4. На какие три вопроса должен отвечать Headline/Summary?
5. Что такое Severity в описании дефекта?
6. Какие существуют степени Severity? Приведите примеры.
7. Что такое Description в описании дефекта?
8. Что такое Expected result в описании дефекта?
9. Зачем нужен Attachment при описании дефекта?
10. Какие существуют рекомендации по описанию дефектов?