Лабораторная работа №3

Тестирование программного обеспечения: разработка тестов

Цель: разработать рабочую тестовую документацию для тестирования web-приложения.

План занятия:

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

Теоретические сведения

Рабочая тестовая документация значительно улучшает качество последующего тестирования за счет анализа и детального планирования тестов. После завершения тестирования наличие тестовой документации позволяет оценить, насколько успешно были проведены все этапы тестирования, а для заказчика является подтверждением реального объема работ.

Рабочую тестовую документацию тестировщик может разрабатывать исключительно на основе спецификации еще до поставки программного обеспечения. В этом случае после поставки на тестирование версии программного продукта специалист по тестированию может сразу приступить к поиску дефектов.

Существуют следующие виды рабочей тестовой документации (таблица 4.1):

1. Check List.
2. Acceptance Sheet.
3. Test Survey.
4. Test Cases.

Основные факторы выбора тестовой документации – сложность бизнес-логики проекта, сроки проекта, размер команды и объем проекта.

На одном проекте могут комбинироваться несколько типов тестовой документации. Например, для всего проекта составлен Acceptance Sheet, но для наиболее сложных частей составлены Test Cases. Если какие-либо модули программного продукта будут подвергаться автоматизированному тестированию, то для таких модулей в обязательном порядке составляются Test Cases.

Таблица 4.1 – Виды рабочей тестовой документации и их характеристика

Тип документации	Что описывают	Когда используют
Check List	Вспомогательный тип документации, содержащий список основных проверок	Для типовой функциональности
Acceptance Sheet	Перечень всех модулей и функций приложения, подлежащих проверке	Небольшие (до 3 месяцев), простые по бизнес-логике проекты
Test Survey	Перечень всех модулей и функций приложения, а также конкретные проверки для них. Может содержать ожидаемый результат	Средние или большие проекты с понятной бизнес-логикой
Test Cases	Перечень всех модулей и функций приложения, а также конкретные проверки для них. Каждая проверка содержит набор входных значений, предусловий, шагов выполнения и ожидаемых результатов. Всегда приводятся ожидаемые результаты для каждого шага проверки	Большие и долгосрочные проекты, проекты со сложной бизнес-логикой, проекты с большой командой

При составлении рабочей тестовой документации необходимо указать номер тестируемой сборки, тип выполняемой тестовой активности, период времени тестирования, ФИО тестировщика, тестовое окружение (операционная система, браузер и др.).

Рабочая тестовая документация представляет собой перечень всех проверок для модулей/подмодулей приложения. В качестве одного модуля, как правило, выступает рабочее окно приложения, в качестве подмодулей – логически завершенные блоки этого окна.

Для каждого модуля в обязательном порядке выполняется тестирование GUI, а также общие функциональные проверки (General) для любого типа приложения (навигация, скроллинг, табуляция и др.). Далее в рамках модуля формулируются проверки соответствия функционала приложения требованиям, заявленным в спецификации. Степень детализации каждой из таких функциональных проверок зави-

ствующих элементов GUI. Для Test Cases каждую из позитивных и негативных проверок описывают в виде последовательности шагов с указанием ожидаемого результата. Для Test Survey напротив каждой проверки указывается глубина тестирования: Smoke, MAT, AT. Для Acceptance Sheet в качестве глубины тестирования всегда указывается AT.

Примеры фрагментов рабочей тестовой документации

Таблица 4.2 – Примеры Check List, Acceptance Sheet, Test Survey

Вид рабочей тестовой документации	Пример	Глубина тестирования
1	2	3
Check List	Протестировать форму авторизации	–
Acceptance Sheet	Форма авторизации: 1. GUI. 2. General. 3. Поле «Эл. адрес». 4. Поле «Пароль». 5. Кнопка «Войти». 6. Чекбокс «Не выходить из системы». 7. Ссылка «Забыли пароль»	 AT AT AT AT AT AT AT
Test Survey	Форма авторизации: 1. GUI. 2. General. 3. Валидный эл. адрес + валидный пароль. 4. Валидный эл. адрес с пробелами в начале и в конце + валидный пароль с пробелами в начале и в конце. 5. Валидный эл. адрес с пробелами в середине + валидный пароль с пробелами в середине. 6. Валидный эл. адрес минимально допустимой длины + валидный пароль минимально допустимой длины. 7. Валидный эл. адрес максимально допустимой длины + валидный пароль максимально допустимой длины. 8. Пустое поле эл. адреса + пустое поле пароля.	 AT AT Smoke MAT MAT MAT MAT AT

Продолжение таблицы 4.2

1	2	3
	9. Валидный эл. адрес + невалидный пароль (спец-символы).	АТ
	10. Валидный эл. адрес + невалидный пароль (русские буквы).	АТ
	11. Валидный эл. адрес + невалидный пароль (длина превышает максимально допустимую).	АТ
	12. Валидный эл. адрес + невалидный пароль (длина меньше минимально допустимой).	АТ
	13. Невалидный эл. адрес (спецсимволы кроме ’_’, ’-’, ’@’, ’. ’) + валидный пароль.	АТ
	14. Невалидный эл. адрес (русские буквы) + валидный пароль.	АТ
	15. Невалидный эл. адрес (использование символа ’@’ дважды) + валидный пароль.	АТ
	16. Невалидный эл. адрес (отсутствие символа ’.’) + валидный пароль.	АТ
	17. Невалидный эл. адрес (длина превышает максимально допустимую) + валидный пароль.	АТ
	18. Невалидный эл. адрес (длина меньше минимально допустимой) + валидный пароль.	АТ
	19. Запомнить данные: выйти из системы и зайти обратно.	МАТ
	20. Ссылка «Забыли пароль»	МАТ

Тест-кейс (таблица 4.3) – набор входных данных, условий выполнения и ожидаемых результатов, разработанный с целью проверки того или иного свойства или поведения программного средства.

Под тест-кейсом также понимают соответствующий документ, представляющий формальную запись тест-кейса.

Высокоуровневый тест-кейс – тест-кейс без конкретных входных данных и ожидаемых результатов. Как правило, ограничивается общими идеями и операциями, схож по своей сути с подробно описанными пунктами чек-листа.

Низкоуровневый тест-кейс – тест-кейс с конкретными входными данными и ожидаемыми результатами. Представляет собой полностью готовый к выполнению тест-кейс и является наиболее классическим видом тест-кейсов. Начинающих тестировщиков чаще всего учат писать именно такие тесты, т. к. прописать все данные

подробно намного проще, чем понять, какой информацией можно пренебречь, при этом не снизив ценность тест-кейса.

В зависимости от внутренних шаблонов компании и инструмента управления тест-кейсами внешний вид их записи может немного отличаться (могут быть добавлены или убраны отдельные поля), но концепция остается неизменной.

Общая структура тест-кейса включает (см. таблицу 4.3):

- идентификатор;
- связанное с тест-кейсом требование;
- модуль и подмодуль приложения;
- заглавие тест-кейса;
- исходные данные, приготовления к тест-кейсу;
- шаги тест-кейса;
- ожидаемые результаты по каждому шагу тест-кейса.

Таблица 4.3 – Пример test Case

Номер тест-кейса	Приоритет	Требование	Модуль	Подмодуль	Описание тест-кейса	Ожидаемые результаты
18	A	REQ3.7	Главная страница	Форма авторизации	Авторизация с помощью валидного эл. адреса и валидного пароля. 1. Ввести в поле «Эл. адрес» abc@mail.ru. 2. Ввести в поле «Пароль» qwerty. Нажать кнопку «Войти»	1. В поле «Эл. адрес» отображается abc@mail.ru. 2. В поле «Пароль» отображается qwerty. Осуществляется переход на домашнюю страницу пользователя abc@mail.ru, qwerty

Идентификатор представляет собой уникальное значение, позволяющее однозначно отличить один тест-кейс от другого и используемое во всевозможных ссылках. В общем случае идентификатор тест-кейса может представлять собой просто уникальный номер, но (если позволяет инструментальное средство управления

тест-кейсами) может включать префиксы, суффиксы и иные осмысленные компоненты, позволяющие быстро определить цель тест-кейса и часть приложения (или требований), к которой он относится.

Приоритет показывает важность тест-кейса. Он может быть выражен буквами (A, B, C, D, E), цифрами (1, 2, 3, 4, 5), словами («крайне высокий», «высокий», «средний», «низкий», «крайне низкий») или иным удобным способом. Количество градаций также не фиксировано, но чаще всего лежит в диапазоне от трех до пяти. Приоритет тест-кейса может коррелировать с важностью требования, с которым связан тест-кейс; потенциальной важностью дефекта, на поиск которого направлен тест-кейс. Основная задача этого атрибута – упрощение распределения внимания и усилий команды, а также упрощение планирования и принятия решения о том, чем можно пожертвовать в некоей форс-мажорной ситуации, не позволяющей выполнить все запланированные тест-кейсы.

Связанное с тест-кейсом требование показывает то основное требование, проверке выполнения которого посвящен тест-кейс. Наличие этого поля улучшает такое свойство тест-кейса, как прослеживаемость. При этом следует отметить, что некоторые тест-кейсы могут разрабатываться вне прямой привязки к требованиям. Хоть такой вариант и не считается хорошим, он достаточно распространен.

Модуль и подмодуль приложения указывают на части приложения, к которым относится тест-кейс, и позволяют лучше понять его цель. Идея деления приложения на модули и подмодули проистекает из того, что в сложных системах практически невозможно охватить взглядом весь проект целиком. Тогда приложение логически разделяется на компоненты (модули), а те, в свою очередь, на более мелкие компоненты (подмодули). Для таких небольших частей приложения разработать тест-кейсы становится намного проще. Как правило, иерархия модулей и подмодулей создается как единый набор для всей проектной команды, чтобы исключить путаницу из-за того, что разные люди будут использовать разные подходы к такому разделению или даже просто разные названия одних и тех же частей приложения. Модули и подмодули можно выделять на основе графического интерфейса пользователя (крупные области и элементы внутри них), на основе решаемых приложением задач и подзадач и т. д. Главное, чтобы эта логика была одинаковым образом применена ко всему приложению.

Заглавие тест-кейса призвано упростить быстрое понимание основной идеи тест-кейса без обращения к его остальным атрибутам. Именно это поле является наиболее информативным при просмотре списка тест-кейсов. Заглавие тест-кейса должно быть информативным, уникальным.

Исходные данные, необходимые для выполнения тест-кейса, позволяют описать все то, что должно быть подготовлено до начала выполнения тест-кейса, например, состояние базы данных, состояние файловой системы и ее объектов, состояние серверов и сетевой инфраструктуры.

Шаги тест-кейса описывают последовательность действий, которые необходимо реализовать в процессе выполнения тест-кейса. Общие рекомендации по написанию шагов таковы:

– не пишите лишних начальных шагов (запуск приложения, очевидные операции с интерфейсом и т. п.);

– даже если в тест-кейсе всего один шаг, нумеруйте его;

– если вы пишете на русском языке, используйте безличную форму (например, «открыть», «ввести», «добавить» вместо «откройте», «введите», «добавьте»);

– соотносите степень детализации шагов и их параметров с целью тест-кейса, его сложностью, уровнем и т. д.; в зависимости от этих и многих других факторов степень детализации может варьироваться от общих идей до предельно четко прописанных значений и указаний;

– ссылайтесь на предыдущие шаги и их диапазоны для сокращения объема текста (например, «повторить шаги 3–5 со значением…»);

– пишите шаги последовательно, без условных конструкций вида «если… то…».

Ожидаемые результаты по каждому шагу тест-кейса описывают реакцию приложения на действия, описанные в поле «шаги тест-кейса». Номер шага соответствует номеру результата. По написанию ожидаемых результатов можно порекомендовать следующее:

– описывайте поведение системы так, чтобы исключить субъективное толкование (например, вместо недопустимого «приложение работает верно» следует писать «появляется окно с надписью…»);

– пишите ожидаемый результат по всем шагам без исключения, даже если результат некоего шага будет совершенно тривиальным и очевидным;

– пишите кратко, но не в ущерб информативности;

– избегайте условных конструкций вида «если… то…».

Наличие тест-кейсов позволяет:

– структурировать и систематизировать подход к тестированию;

– вычислять метрики тестового покрытия и принимать меры по его увеличению;

– отслеживать соответствие текущей ситуации плану (сколько примерно понадобится тест-кейсов, сколько уже есть, сколько выполнено и т. д.);

– уточнить взаимопонимание между заказчиком, разработчиками и тестировщиками (тест-кейсы зачастую намного более наглядно показывают поведение приложения, чем это отражено в требованиях);

– хранить информацию для длительного использования и обмена опытом между сотрудниками и командами;

– проводить регрессионное тестирование и повторное тестирование;

– повышать качество требований.

Далее рассмотрим подробно базовые проверки графического интерфейса пользователя и функциональности web-, desktop- и mobile-приложений.

Для любого приложения выполняется тестирование графического интерфейса пользователя (таблица 4.4).

Таблица 4.4 – Перечень основных GUI-проверок для всего приложения

Название проверки	Описание проверки
1	2
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

Продолжение таблицы 4.4

1	2
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

Общие проверки функциональности для всех типов приложений, а также общие проверки для web-приложений приведены в таблицах 4.5, 4.6.

Таблица 4.5 – Перечень общих проверок функциональности для любого типа приложения

Название проверки	Описание проверки
1. Табуляция	Перемещение с помощью клавиатуры должно осуществляться сверху вниз слева направо. Недоступные поля должны пропускаться
2. «Хлебные крошки»	«Хлебные крошки» – элемент навигации, являющийся признаком удобства пользования приложением в целом и перемещением по его структуре
3. Скроллинг	Отсутствие скроллинга в случае, если текст вмещается на странице без прокрутки. Соответствующее изменения текста при использовании скроллинга. Возможность изменения положения скроллинга при помощи мыши, кнопок Page up/down, Home/End
4. Взаимосвязь компонентов	Поведение одного компонента при изменении/удалении другого (например, при удалении категории товара не должны удаляться все товары в этой категории)
5. Фокус на кнопке для исполнения действий	Ввод данных → нажатие Enter → действие осуществилось

Таблица 4.6 – Перечень общих проверок функциональности для web-приложений

Название проверки	Описание проверки
1. Подготовка к тестированию	Перед тестированием каждой новой сборки необходимо осуществить очистку кэша и cookies. Для этого можно воспользоваться приложением CCleaner
2. 404 Error	Переход по некорректному адресу должен вести на страницу с Error 404, а не на страницу Page cannot be found, например. Страница Error 404 должна быть реализована в общем дизайне тестируемого приложения
3. Логотип	Логотип должен быть ссылкой на главную страницу
4. Email нотификации	Проверка работоспособности отправки email-нотификаций (как администратору, так и пользователю), если только отсутствие писем не является спецификой проекта
5. Отображение flash-элементов при отключенном или неустановленном в браузере flash-плеере	Пользователю должно быть предложено скачать и установить последнюю версию flash-плеера; на месте flash-объекта должно отображаться альтернативное изображение
6. Проверка работоспособности приложения при отключенном JavaScript	Основная функциональность и навигация должна работать

В таблицах 4.7–4.16 приведены основные проверки для элементов пользовательского интерфейса: поле для ввода данных, поле для загрузки файлов, поле для ввода даты, поле со списком/выпадающий список, кнопка, радиобаттон, чекбокс, меню, таблица, календарь, ссылка, сообщения, поп-ап (всплывающие окна).

Таблица 4.7 – Перечень основных проверок для поля ввода данных

Functional Test	GUI Test
1. Обязательность ввода. 2. Обработка только пробелов. 3. Использование пробелов в тексте: 3.1. Пробелы в начале и в конце строки должны отсекаться при сохранении. 3.2. Пробелы внутри текста отсекаться не должны. 4. Минимально/максимально допустимое количество символов. 5. Формат данных (исходя из его логического назначения и требований приложения). 6. Формат числовых данных (если допускаются): негативные, дробные с точкой и запятой. 7. Использование специальных символов (введенные символы должны отобразиться в том же виде, в котором они были введены, если только ввод специальных символов не запрещен требованиями приложения). 8. Возможность редактирования введенных значений. 9. Корректное распределение текста по строкам (переход на новую строку автоматически). 10. Уникальные данные (например, уникальность логина, email). 11. Автоматическая постановка курсора в первое поле для ввода при открытии формы. 12. Ввод тегов и скриптов (введенные теги и скрипты должны отобразиться в том же виде, в котором они были введены).	1. Название поля (правописание, соответствие названия тематике модуля/страницы). 2. Выравнивание названий полей (выравнивание по левому или правому краю в зависимости от требований приложения, отступы, идентичность расстояний между названием и полем). 3. Корректное расположение текста, длинный текст не выходит за границы поля при вводе. 4. Унификация дизайна по отношению ко всему приложению (цвет, шрифт, размер (высота/ширина), выравнивание полей). 5. Расположение вводимого текста внутри поля (унификация, выравнивание)

Таблица 4.8 – Перечень основных проверок для поля загрузки файлов

Functional Test	GUI Test
1	2
1. Обязательность выбора файла. 2. Форматы: корректные/некорректные. 3. Корректный формат, но отсутствует/модифицировано расширение.	1. Унификация дизайна по отношению ко всему приложению (цвет, шрифт, высота/ширина).

Продолжение таблицы 4.8

1	2
4. Ограничения на размер (включая загрузку файлов нулевого размера, большого размера).Загрузка исполняемых файлов (EXE, PHP, JSP др.). 5. Загрузка переименованного EXE-файла. 6. Путь к файлу меньше 259 символов. 7. Путь к файлу равен 260 символов. 8. Путь к файлу больше 260 символов. 9. Корректный путь введен с клавиатуры. 10. Имитировать сбой загрузки (например, с использованием flash-накопителя). 11. Одновременная загрузка нескольких файлов	2. Выравнивание названий загруженных файлов.

Таблица 4.9 – Перечень основных проверок для радиобаттона

Functional Test	GUI Test
1. Функциональность: включение/выключение. 2. Не может быть меньше двух радиокнопок. 3. По умолчанию одна радиокнопка должна быть включена. 4. Не может быть включено более одной радиокнопки. 5. При переходе на следующую страницу и возвращении назад выбранная радиокнопка не должна сбрасываться. 6. Активация путем нажатия как на символ, так и на текст	1. Унификация дизайна по отношению ко всему приложению. 2. Выравнивание расположения радиобаттона с соответствующим названием. 3. Выравнивание расположений радиобаттонов. 4. Изменение радиобаттона при наведении курсора. 5. Изменение курсора при наведении на радиобаттон

Таблица 4.10 – Перечень основных проверок для чекбоксов

Functional Test	GUI Test
1	2
1. Функциональность: включение/выключение. 2. Наличие дополнительного чекбокса, выставляющего/снимающего все чекбоксы при наличии больше 10 чекбоксов.	1. Унификация дизайна по отношению ко всему приложению. 2. Выравнивание расположения чекбокса с соответствующим названием.

Продолжение таблицы 4.10

1	2
3. При переходе на следующую страницу и возвращении назад выбранный чекбокс не должен сбрасываться. 4. Активация путем нажатия как на символ, так и на текст	3. Изменение чекбокса при наведении курсора. 4. Изменение курсора при наведении на чекбокс

Таблица 4.11 – Перечень основных проверок для поля со списком

Functional Test	GUI Test
1. Сортировка по алфавиту или по смыслу. 2. В случае, если значения выходят за границы списка и нет возможности увеличения размера списка, то необходимо отображение хинтов (всплывающих подсказок). 3. Выбор пункта списка по нажатии соответствующей первой буквы на клавиатуре. 4. Возможность введения значений вручную (если это позволяет приложение). 5. Возможность выбора значения из списка как с помощью мыши, так и с клавиатуры	1. Правописание. 2. Подсветка при выборе каждого из значений, при выборе нескольких значений одновременно. 3. Унификация дизайна (цвет, шрифт, размер (высота/ширина), цвет подсветки, выравнивание)

Таблица 4.12 – Перечень основных проверок для меню

Functional Test	GUI Test
1. Осуществление соответствующего перехода при выборе пункта меню. 2. Визуальное различие в момент работы на определенной вкладке (подсветка, подчеркивание)	1. Подсветка категории меню при наведении курсора. 2. Изменение курсора при наведении на категорию меню. 3. Если в данный момент выполняется работа в выбранной вкладке, то в меню она отличается визуально и является неактивной. 4. Совпадение названий категорий меню в случае, если меню дублируется в нескольких местах

Таблица 4.13 – Перечень основных проверок для ссылки

Functional Test	GUI Test
1. Функционирование ссылки (должен осуществиться переход на соответствующую страницу). 2. Переход по загруженной ссылке должен осуществляться в новой вкладке или во всплывающем окне. 3. Форматы ссылок и префиксов. 4. Срабатывание ссылки только при нажатии на саму ссылку, а не на пустую область возле нее	1. Унификация стилей (в соответствии с дизайном сайта). 2. Расположение ссылок (в соответствии с дизайном сайта). Например, расположение всех ссылок слева или справа от элементов. 3. Названия (унификация, идентичность названий ссылок одинакового назначения, спеллинг, соответствие с открытым модулем или страницей, вместимость названия ссылки в отведенном блоке). 4. Изменение вида курсора при наведении на ссылку. 5. Изменение вида ссылки при наведении курсора (подчеркивание)

Таблица 4.14 – Перечень основных проверок для таблицы

Functional Test	GUI Test
1. При появлении нескольких страниц есть кнопки Вперед, Назад, На первую, На последнюю страницу (пагинация). 2. Проверка сортировок, в том числе сортировки по умолчанию. 3. Обновление значений таблицы после добавления, изменения, удаления данных. 4. Единичное/множественное выделение нескольких значений	1. Унификация дизайна для всего приложения (цвет, шрифт, размер (высота/ширина), выравнивание). 2. Название (соответствие с текущим модулем, спеллинг). 3. Выравнивание значков сортировки в названии колонок. 4. Выравнивание названий колонок, значений внутри таблицы. 5. Корректное отображение длинных названий (соответствующие переходы на новые строки, сокращение названий (появление многоточия либо сокращение по слову). 6. Корректное отображение данных после использования сортировки (размеры колонок и столбцов фиксированы, текст не разбивает структуру таблицы)

Таблица 4.15 – Перечень основных проверок для календаря

Functional Test	GUI Test
1. Ввод даты с помощью календаря. 2. Ввод даты вручную: разные форматы, номер месяца: > 12, день: > 31 (+ для февраля). 3. Логика работы поля (например, подсчет возраста после ввода даты рождения; невозможность ввести дату рождения свыше текущего дня)	1. Унификация дизайна для всего приложения (цвет, шрифт, размер (высота/ширина), выравнивание). 2. Отображение календаря рядом с полем. 3. Корректное выравнивание всех элементов и ссылок в календаре

Таблица 4.16 – Перечень основных проверок для сообщений

Functional Test	GUI Test
1. Пользователь должен быть информирован о действиях, происходящих в системе посредством сообщений об успешном завершении операции. 2. На необратимые действия, такие как удаление, должны быть подтверждающие сообщения. 3. Введенные в форму данные не должны сбрасываться после появления сообщения	1. Правописание сообщений. 2. Соответствие сообщений смыслу выполняемого действия. 3. Соответствие названий полей в сообщениях названиям полей, форм, таблиц, кнопок и т. д. 4. Унификация стилей (цвет, размер) для всего приложения. 5. Если присутствует кнопка «Отмена», то в правом верхнем углу формы с сообщением присутствует «крестик» для альтернативной возможности закрыть форму. 6. Сообщения о подтверждении удаления по умолчанию активированы на кнопку «нет». 7. Соответствие цвета типу сообщения (красный для сообщений об ошибках, зеленый для сообщений об успешном завершении операции). 8. Невалидное значение не должно отображаться в сообщении об ошибке (неправильно: «Email 2309234@@mail.ru не соответствует допустимому формату»). 9. Согласование числительного и связанного с ним существительного (например, «1 день», «2 дня»)

Практическое задание:

1. Получить у преподавателя спецификацию с требованиями к web-приложению.
2. В зависимости от сложности бизнес-логики web-приложения выбрать наиболее подходящий вид рабочей тестовой документации (Acceptance Sheet, Test Survey, Test Cases).
3. Анализируемое web-приложение разбить на модули и подмодули.
4. Разработать рабочую тестовую документацию для всех модулей и подмодулей web-приложения.
5. Указать номер тестируемой сборки, название приложения, тип выполняемой тестовой активности, период времени тестирования, ФИО тестировщика, тестовое окружение (операционная система, браузер).
6. Предусмотреть проверки GUI для каждого модуля.
7. Предусмотреть общие функциональные проверки (General) для каждого модуля.
8. В рамках каждого модуля предусмотреть функциональные проверки. Степень детализации каждой из функциональных проверок должна соответствовать выбранному на этапе 1 типу тестовой документации.
9. Для каждой проверки указать глубину тестового покрытия (Smoke, MAT, AT) с учетом выбранного на этапе 1 типа тестовой документации.
10. Оформить отчет и защитить лабораторную работу.

Содержание отчета:

1. Цель работы.
2. Рабочая тестовая документация.
3. Выводы по работе.

Контрольные вопросы:

1. Какие существуют разновидности рабочей тестовой документации?
2. Check List: что описывают и когда используют?
3. Acceptance Sheet: что описывают и когда используют?
4. Test Survey: что описывают и когда используют?
5. От чего зависит степень детализации каждой функциональной проверки?
6. Какая глубина тестирования указывается для проверок в Acceptance Sheet?
7. К
8. Ч
9. Какова структура описания Test Case?
10. Что содержит Идентификатор в описании Test Case?
11. Что приводится в поле Приоритет описания Test Case?

т
а
к
о
е

12. Что приводится в поле Требование описания Test Case?
13. Что приводится в поле Модуль и подмодуль приложения описания Test

14. Что приводится в поле Заглавие описания Test Case?
15. Что приводится в поле Исходные данные, приготовления описания Test

16. Что приводится в поле Шаги описания Test Case?
17. Что приводится в поле Ожидаемые результаты описания Test Case?
18. Для чего нужны Test Cases?
19. Какие проверки выполняют при тестировании GUI?
20. Какие общие функциональные проверки выполняют для всего приложения?
21. Перечислите базовые проверки для поля ввода данных.
22. Перечислите базовые проверки для поля загрузки файлов.
23. Перечислите базовые проверки для ввода даты.
24. Перечислите базовые проверки для поля со списком.
25. Перечислите базовые проверки для радиобаттона.
26. Перечислите базовые проверки для чекбокса.
27. Перечислите базовые проверки для меню.
28. Перечислите базовые проверки для таблиц.
29. Перечислите базовые проверки для ссылок.
30. Перечислите базовые проверки для сообщений.