

ОТЧЕТ

о выполнении лабораторных работ 6

Выполнили: студенты группы. 1-ИСП11-19

Мейстер Ю. А.

Суковатов Н. Д.

Проверил: преподаватель Рогулин А.А.

2022

Оглавление

[Лабораторная работа №1 3](#_Toc106352713)

[Лабораторная работа №2 11](#_Toc106352714)

[Лабораторная работа №3 14](#_Toc106352715)

[Лабораторная работа №4. 20](#_Toc106352716)

[Лабораторная работа №6 27](#_Toc106352717)

[Лабораторная работа №7 33](#_Toc106352718)

## Лабораторная работа №1

**Виды тестирования. Планирование тестирования**

*Цель:* изучить классификацию видов тестирования, разработать проверки для различных видов тестирования, научиться планировать тестовые активности в зависимости особенностей поставляемой на тестирование функциональности

|  |  |  |
| --- | --- | --- |
| **Объект тестирования:** Клавиатура | | |
| **Вид тестирования** | **Краткое определение вида тестирования** | **Тестовые проверки** |
| Functional Testing | Тестирование, анализ спецификации компонентов или системы | Проверка отклика кнопок и подключения к системе |
| Safety Testing | Тестирование безопасности продукта | Проверка на изолированность проводов |
| Security Testing | Тестирование с целью оценивания защищённости | Проверка на ударопрочность |
| Compatibility Testing | Проверка совместимости | Проверка подключения к различным устройствам |
| GUI Testing | Тестирование пользовательского интерфейса | Проверка символов на клавиатуре на читаемость |
| Usability Testing | Тестирование удобство пользования | Проверка на наличие второго языка и соответствие QWERTY |
| Accessibility Testing | Тестирование доступности для пользователей | Проверка на наличие второго языка и соответствие QWERTY |
| Internationalization Testing | Тестирование интернационализации продукта | Проверка на наличие второго языка |
| Performance Testing | Тестирование производительности программного продукта | Проверка на скорость отклика |
| Stress Testing | Тестирование продукта на пределах система | Проверка за пределами ударопрочности и силы нажатия |
| Negative Testing | Тестирование системы на некорректность данных | - |
| Black Box Testing | Тестирование системы без знания о продукте | Проверка на интуитивное понимание |
| Automated Testing | Тестирование без участия человека | Отсутствует |
| Unit/Component Testing | Тестирование отдельных частей системы | Все ли отдельные модули работают корректно |
| Integration Testing | Тестирование взаимодействий между модулями системы | Работает ли корректно взаимодействие со всеми модулями системы |
|  | | |

Вывод: Объект тестирования клавиатура – должен полностью выполнять свои функции, отдельные компоненты системы должны работать исправно, а пользовательский интерфейс должен быть понятен для любого пользователя.

**Контрольные вопросы:**

1. Что такое тестирование?

Это метод проверки соответствия фактического программного продукта ожидаемым требованиям, который также необходим, чтобы убедиться, что продукт не содержит дефектов

2. Что такое качество программного обеспечения?

Это то насколько программное обеспечение удовлетворяет предъявляемым к нему требованиям, исходя из определённых задач

3. Что такое дефект?

Это несоответствие продукции требова­ниям

4. Назовите три условия обнаружения дефекта.

Необходимо знать фактический результат, знать ожидаемый результат, зафиксировать факт разницы между фактическим и ожидаемым результатом.

5. Какие существуют виды тестирования в зависимости от объекта тестирования? Дайте характеристику каждому.

Функциональные виды тестирования

Функциональные тесты базируются на функциях и особенностях, а также взаимодействии с другими системами, и могут быть представлены на всех уровнях тестирования. Нефункциональные виды тестирования

Нефункциональное тестирование описывает тесты, необходимые для определения характеристик программного обеспечения, которые могут быть измерены различными величинами.

Связанные с изменениями виды тестирования

После проведения необходимых изменений, таких как исправление бага/дефекта, программное обеспечение должно быть пере тестировано для подтверждения того факта, что проблема была действительно решена.

6. Какие существуют виды функционального тестирования? Дайте характеристику каждому.

Тестирование black box (черный ящик) — проведение функционального тестирования без доступа к коду системы,

Тестирование white box (белый ящик) — функциональное тестирование с доступом к коду системы.

7. Какие существуют виды нефункционального тестирования? Дайте характеристику

каждому.

Все виды тестирования производительности:

нагрузочное тестирование (Performance and Load Testing)

стрессовое тестирование (Stress Testing)

тестирование стабильности или надежности (Stability / Reliability Testing)

объемное тестирование (Volume Testing)

Тестирование установки (Installation testing)

Тестирование удобства пользования (Usability Testing)

Тестирование на отказ и восстановление (Failover and Recovery Testing)

Конфигурационное тестирование (Configuration Testing)

Тестирование безопасности (Security and Access Control Testing)

8. Какие существуют виды тестирования в зависимости от глубины покрытия? Дайте характеристику каждому.

Функциональное тестирование (functional testing)

Системное тестирование (system testing)

Тестирование проиводительности (performance testing)

Регрессионное тестирование (regression testing)

Модульное тестирование (unit testing)

Тестирование безопасности (security testing  
9. Какие существуют тестовые активности? Дайте характеристику каждому.

Модульное тестирование: тестирование функциональности каждой отдельной операции.

Функциональное тестирование: тестирование функциональности более широких сценариев, связанных с созданием тестового примера, валидацией, регрессией

10. Какие существуют виды тестирования в зависимости от знания кода?

Тестирование «белого ящика» – с доступом к коду.

Тестирование «черного ящика» – без доступа к коду продукта.

Тестирование «серого ящика» – на основе ограниченного знания внутренней структуры ПО

11. Какие существуют виды тестирования в зависимости от степени автоматизации? Дайте характеристику каждому.

Функциональное тестирование включают в себя:

защищенность

соответствие стандартам

способность к взаимодействию

функциональная пригодность

точность

Нефункциональное тестирование включает в себя:

Удобство (В основном производиться оценка удобства для пользователей)

Масштабируемость (проверяется как вертикальная так и горизонтальная масштабируемость тестируемого приложения)

Производительность (Способность работы приложения при различных нагрузках)

Безопасность (Защита пользовательских данных, защита данных приложения, стойкость на взлом)

Портируемость (Совместимость и переносимость приложения для и под различные окружения, платформы и т.д.)

Надежность (Поведение системы при различных непредвиденных ситуациях, способность обработки нестандартных действий пользователя)

12. Какие существуют виды тестирования в зависимости от изолированности  
компонентов? Дайте характеристику каждому.

Тестирование компонентов, Интеграционное тестирование, Системное тестирование

13. Какие существуют виды тестирования в зависимости от подготовленности? Дайте характеристику каждому.

Модульное / юнит-тестирование – проверка корректной работы отдельных единиц ПО, модулей. Этот вид тестирования могут выполнять сами разработчики.

Интеграционное тестирование – проверка взаимодействия между несколькими единицами ПО.

Системное – проверка работы приложения целиком.

Приёмочное – оценка соответствия заявленным требованиям к программному продукту.

14. Какие существуют виды тестирования в зависимости от места и времени  
проведения? Дайте характеристику каждому.

Альфа-тестирование

Дымовое тестирование (англ. smoke testing)

Тестирование новой функции (new feature testing)

Подтверждающее тестирование

Регрессионное тестирование

Приемочное тестирование

Бета-тестирование

15. Какие этапы составляют процесс тестирования?

Выбор тестовых методик.

Инструктирование испытуемых.

Контроль за выполнением заданий.

Интерпретация результатов и подведение итогов тестирования.

16. Какая композиция тестов выполняется для первой поставки программного  
продукта?

Для первого билда рекомендуется проводить Smoke+AT готовой функциональности: поверхностное тестирование (Smoke Test) выполняется для определения пригодности сборки для дальнейшего тестирования; полное тестирование системы или ее части как на корректных, так и на некорректных данных/сценариях (Acceptance Test, AT) позволяет обнаружить дефекты и внести запись о них в багтрэкинговую систему.

17. Какая композиция тестов выполняется для последующих поставок программного продукта?

Для последующих поставок программного обеспечения композиции тестов могут быть следующими. Testing, NFT), проверка старых функциональностей, т. е. регрессионное тестирование (Regression Test).

## Лабораторная работа №2

Цель: выявить и описать пользовательские требования в виде вариантов использования (Use Cases)

|  |  |
| --- | --- |
| **Категории варианта использования** | **Описание** |
| Идентификатор  и название | UC-1 Запрос заказа яиц |
| Основное  действующее  лицо | Сотрудник склада |
| Описание | Сотрудник, указывает в запросе  Количество упаковок яиц, которое необходимо.  Система выполняет запрос |
| Триггер | Сотрудник указывает, что хочет заказать яйца |
| Предварительные условия | PRE-1. Личность пользователя аутентифицирована.  PRE-2. Пользователь имеет право размещать заказы  PRE-3. База данных по запросам доступна |
| Выходные условия | POST-1. Запрос сохраняется в Computer Tracking System.  POST-2. Запрос отправлен на склад поставщика |
| Нормальное  направление  развития  варианта  использования | 1.0 Запросить яйца со склада  1.0.1 Сотрудник указывает требуемое кол-во яиц  1.0.2 Система перечисляет упаковки имеющиеся на складе.  1.0.3 Сотрудник может просмотреть историю любой упаковки.  1.0.4 Сотрудник выбирает определенную упаковку или просит отправить запрос поставщику (см. п. 1.1).  1.0.5 Сотрудник вводит остальную информацию, чтобы завершить запрос.  1.0.6 Система сохраняет запрос и отправляет его на склад |
| Исключения | 1.2 Яиц нет на складе  1.2.1 Система отображает сообщение «У поставщиков нет такого яиц».  1.2.2 Система предлагает выйти из программы или заказать другой товар.  1.2.3.1 Сотрудник запрашивает другой товар(см. п. 1.2.3.2).  1.2.4.1 Система заново начинает нормальное направление варианта  использования.  1.2.3.2 Сотрудник решает выйти из системы.  1.2.4.2 Система завершает вариант использования |

**Контрольные вопросы:**

1. Что такое требование?

отдельная документально подтвержденная физическая или функциональная потребность, которую стремится удовлетворить конкретный дизайн, продукт или процесс.

2. Какое значение имеют требования в проекте по разработке программного

обеспечения?

Реальное понимание целей и проблем заказчика, четкое формулирование задач и решений является основой успешного взаимодействия с заказчиком. Полнота и согласованность требований является условием построения простой и гибкой архитектуры, продуктивного и экономичного процесса разработки программного решения.

3. Какие существуют этапы работы над требованиями?

Обучение, Управление требованиями, Управление проектом, Сбор информации, Анализ, Спецификации, Проверка.

4. Кто выполняет работу с требованиями?

Клиенты

5. Какие существуют уровни требований?

На верхнем уровне представлены так называемые бизнес-требования

Следующий уровень - уровень требований пользователей

Третий уровень - функциональный

6. Что такое вариант использования?

В разработке программного обеспечения и системном проектировании это описание поведения системы, когда она взаимодействует с кем-то (или чем-то) из внешней среды

7. Для чего нужен вариант использования?

Это сценарная техника описания взаимодействия. С помощью Use Case может быть описано и пользовательское требование, и требование к взаимодействию систем, и описание взаимодействия людей и компаний в реальной жизни.

8. Какие элементы входят в состав описания варианта использования?

Последовательность, о которой идет речь в определении, - это в действительности определенный поток событий в системе, или экземпляр. Может существовать множество потоков событий, и многие могут быть очень простыми. Для того модель вариантов использования была понятной, сходные потоки событий необходимо сгруппировать в один вариант использования. Идентификация и описание варианта использования в реальности означает идентификацию и описание группы связанных потоков событий.

9. Что такое основной сценарий варианта использования?

Сценарий – это последовательность шагов, описывающая процесс решения задачи, которой посвящен вариант использования. Шаги удобно последовательно нумеровать

10. Что такое альтернативный сценарий варианта использования?

Сценарии использования могут содержать вторичные пути или альтернативные сценарии, которые являются вариантами основного. Каждое проверенное правило может привести к альтернативному пути и когда правил много количество альтернативных путей возрастает приводя к очень сложным документам.

11. Что описывают в исключениях варианта использования?\

Какие-либо исключения в разработке ПО и системном проектирование

12. В чем отличие альтернативного сценария от исключения в описании вари-

анта использования?

В отличие от альтернативного сценария спецификация последовательностей действий может осуществлять система, подсистема или класс, взаимодействуя с внешними факторами

Вывод: В этой работе мы научились выявлять и описывать пользовательские требования в виде вариантов использования (Use Cases)

## Лабораторная работа №3

**Тестирование требований**

Цель: изучить критерии качества требований, выполнить тестирование требований к программному обеспечению.

Практическое задание:

1. Требования: Решение математических уравнений, построение графиков

ПО – Photomath

1. Возможности:

* Решение уравнений;
* Построение графиков;
* Решение систем уравнений;
* Решение неравенств;
* Решение выражений

Критериям соответствует полностью, имеет дополнительные функции

1. Дефекты и расхождения отсутствуют

**Контрольные вопросы:**

1. Какие существуют разновидности рабочей тестовой документации?

Существуют следующие виды рабочей тестовой документации:

1. Check List.

2. Acceptance Sheet.

3. Test Survey.

4. Test Cases.

2. Check List: что описывают и когда используют?

Check List описывают вспомогательный тип документации, содержащий список основных проверок. Используют для типовой функциональности.

3. Acceptance Sheet: что описывают и когда используют?

Acceptance Sheet описывают перечень всех модулей и функций приложения, подлежащих проверке. Используют небольшие, простые по бизнес-логике проекты.

4. Test Survey: что описывают и когда используют?

Test Survey описывают перечень всех модулей и функций приложения, подлежащих проверке. Используют Средние или большие проекты с понятной бизнес-логикой.

5. От чего зависит степень детализации каждой функциональной проверки?

Степень детализации каждой из таких функциональных проверок зависит от выбранного типа тестовой документации (Acceptance Sheet, Test Survey, Test Cases).

6. Какая глубина тестирования указывается для проверок в Acceptance Sheet?

Глубина тестирования: AT

7. Какая глубина тестирования указывается для проверок в Test Survey?

Глубина тестирования: AT, Smoke, MAT.

8. Что такое Test Case?

Test Case – набор входных данных, условий выполнения и ожидаемых результатов, разработанный с целью проверки того или иного свойства или поведения программного средства

9. Какова структура описания Test Case?

Общая структура Test Case:

- идентификатор;

- связанное с Test Case требование;

- модуль и подмодуль приложения;

- заглавие Test Case;

- исходные данные, приготовления к Test Case;

- шаги Test Case;

- ожидаемые результаты по каждому шагу Test Case.

10. Что содержит Идентификатор в описании Test Case?

Идентификатор в описании Test Case содержит префиксы, суффиксы и иные осмысленные компоненты, позволяющие быстро определить цель Test Case и часть приложения

11. Что приводится в поле Приоритет описания Test Case?

В поле Приоритета в описания Test Case уровни A, B, C, D, E.

12. Что приводится в поле Требование описания Test Case?

В поле Требование в описания Test Case пункты, например, REQ3.7.

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

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.

14. Что приводится в поле Заглавие описания Test Case?

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.

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

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.

16. Что приводится в поле Шаги описания Test Case?

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 (напри мер, «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 про- писанных 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… о…».

17. Что приводится в поле Ожидаемые результаты описания Test Case?

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.

18. Для чего нужны Test Cases?

Большие и долгосрочные проекты, проекты со сложной бизнес- логикой, проекты с большой командой.

19. Какие проверки выполняют при тестировании GUI?

1) Правописание

2) Расположение и выравнивание

3) Длинные названия

4) Соответствие названий форм/элементов GUI их назначению

5) Унификация (стиля, цвета, шрифта, названий)

6) Эффект «нажатия»

7) Хинты

8) Сообщения об успешном/неуспешном завершении действия, о подтверждении действия

9) Изменение размеров окна, изменение масштаба страницы

20. Перемещение с помощью клавиатуры должно осуществ-

ляться сверху вниз слева направо. Недоступные поля д Хлебные крошки» – элемент навигации, являющийся признаком удобства пользования приложением в целом

и перемещением по его структуре должны пропускаться

## Лабораторная работа №4.

Тестирование программного обеспечения: разработка тестов

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

Практическое задание.

1) Web-приложение: Telegram бот – Dosug-bot

2) Вид рабочей тестовой документации: Acceptance Sheet

3) Модули: Интерфейс, ввод данных, вывод данных

Подмодули: Кнопки, команды для ввода, контекстные кнопки ответа

6) Предусмотреть проверки GUI для каждого модуля

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)

7) Правописание - лексические, грамматические и пунктуационные ошибки

Соответствующее изменения текста при использовании на разных платформах.

8) Функционирование ссылки (должен осуществиться переход на соответствующую страницу)

Срабатывание ссылки только при нажатии на саму ссылку или картинку

Контрольные вопросы.

1. Какие существуют разновидности рабочей тестовой документации?

Существуют следующие виды рабочей тестовой документации:

1. Check List.

2. Acceptance Sheet.

3. Test Survey.

4. Test Cases.

2. heck List: что описывают и когда используют?

Check List описывают вспомогательный тип документации, содержащий список основных проверок. Используют для типовой функциональности.

3. Acceptance Sheet: что описывают и когда используют?

Acceptance Sheet описывают перечень всех модулей и функций приложения, подлежащих проверке. Используют небольшие, простые по бизнес-логике проекты.

4. Test Survey: что описывают и когда используют?

Test Survey описывают перечень всех модулей и функций приложения, подлежащих проверке. Используют Средние или большие проекты с понятной бизнес-логикой.

5. От чего зависит степень детализации каждой функциональной проверки?

Степень детализации каждой из таких функциональных проверок зависит от выбранного типа тестовой документации (Acceptance Sheet, Test Survey, Test Cases).

6. Какая глубина тестирования указывается для проверок в Acceptance Sheet?

Глубина тестирования: AT

7. Какая глубина тестирования указывается для проверок в Test Survey?

Глубина тестирования: AT, Smoke, MAT.

8. Что такое Test Case?

Test Case – набор входных данных, условий выполнения и ожидаемых результатов, разработанный с целью проверки того или иного свойства или поведения программного средства

9. Какова структура описания Test Case?

Общая структура Test Case:

- идентификатор;

- связанное с Test Case требование;

- модуль и подмодуль приложения;

- заглавие Test Case;

- исходные данные, приготовления к Test Case;

- шаги Test Case;

- ожидаемые результаты по каждому шагу Test Case.

10. Что содержит Идентификатор в описании Test Case?

Идентификатор в описании Test Case содержит префиксы, суффиксы и иные осмысленные компоненты, позволяющие быстро определить цель Test Case и часть приложения

11. Что приводится в поле Приоритет описания Test Case?

В поле Приоритета в описания Test Case уровни A, B, C, D, E.

12. Что приводится в поле Требование описания Test Case?

В поле Требование в описания Test Case пункты, например, REQ3.7.

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

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.

14. Что приводится в поле Заглавие описания Test Case?

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.

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

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.

16. Что приводится в поле Шаги описания Test Case?

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 (напри мер, «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 про- писанных 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… о…».

17. Что приводится в поле Ожидаемые результаты описания Test Case?

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.

18. Для чего нужны Test Cases?

Большие и долгосрочные проекты, проекты со сложной бизнес- логикой, проекты с большой командой.

19. Какие проверки выполняют при тестировании GUI?

1) Правописание

2) Расположение и выравнивание

3) Длинные названия

4) Соответствие названий форм/элементов GUI их назначению

5) Унификация (стиля, цвета, шрифта, названий)

6) Эффект «нажатия»

7) Хинты

8) Сообщения об успешном/неуспешном завершении действия, о подтверждении действия

## Лабораторная работа №6

**Тестирование программного обеспечения: разработка тестов**

Web приложение

1. Telegram бот – Dosug-bot представляет собой бота, который сообщает места досуга в Москве по тэгам или случайное
2. (Я не помню)
3. На некоторых устройствах текст на кнопках срезается
4. 1.1– номер сборки
5. Дефектов во время работы не обнаружено
6. Провести тестирование на большем числе различных устройств

**Контрольные вопросы.**

1. Какова структура итогового отчета о результатах тестирования?

Итоговый отчет можно разделить на части с соответствующей информацией:

1) Общая информация.

2) Сведения о том, кто и когда тестировал программный продукт.

3) Тестовое окружение.

4) Общая оценка качества приложения.

5) Обоснование выставленного качества.

6) Графическое представление результатов тестирования.

7) Детализированный анализ качества по модулям.

8) Список самых критичных дефектов.

9) Рекомендации.

2. Что содержится в разделе «Общая информация»?

Общая информация включает:

название проекта;

номер сборки;

модули, которые подверглись тестированию (в случае, если тестировался не весь проект);

виды тестов по глубине покрытия (Smoke Test, Minimal Acceptance Test, Acceptance Test), тестовые активности (New Feature Test, Regression Testing, Defect Validation);

количество обнаруженных дефектов;

вид рабочей тестовой документации (Acceptance Sheet, Test Survey, Test Cases).

3. Что содержится в разделе «Тестовое окружение»?

Тестовое окружение содержит ссылку на проект, браузер, операционную систему и другую информацию, конкретизирующую особенности конфигурации.

4. Как выставляется общая оценка качества приложения?

Общая оценка качества приложения выставляется на основании анализа результатов работы с приложением, количества внесенных дефектов, важности дефектов. Обязательно учитывается этап разработки проекта – то, что не критично в начале работы, становится важным при выпуске программного продукта. Уровни качества: высокое (High), среднее (Medium), низкое (Low).

5. Как обосновать выставленную оценку качества?

Обоснование выставленного качества является наиболее важной частью отчета, т. к. здесь отражается общее состояние сборки, а именно:

качество сборки на текущий момент;

факторы, повлиявшие на выставление именно такого качества сборки: указание функционала, который заблокирован для проверки, перечисление наиболее критичных дефектов и объяснение их важности для пользователя или бизнеса заказчика;

анализ качества проверенного функционала: улучшилось оно или ухудшилось по сравнению с предыдущей версией;

если качество сборки ухудшилось, то обязательно должны быть указаны регрессионные места;

наиболее нестабильные части функционала с указанием причин, по которым они таковыми являются.

6. Что такое метрика в тестировании?

Помимо вышеуказанных общих характеристик при выставлении и обосновании оценки качества программного продукта активно используются числовые характеристики качества – метрики.

7. Приведите примеры прямых метрик.

Типичные примеры прямых метрик – количество разработанных тест-кейсов, количество найденных дефектов и т. д.

8. Приведите примеры расчетных метрик.

Большинство общепринятых расчетных метрик могут быть собраны автоматически с использованием инструментальных средств управления проектами:

процентное отношение (не)выполненных тест-кейсов ко всем имеющимся;

процентный показатель успешного прохождения тест-кейсов;

процентный показатель заблокированных тест-кейсов;

плотность распределения дефектов;

эффективность устранения дефектов;

распределение дефектов по важности и срочности;

метрики покрытия.

9. Для чего используется графическое представление результатов тестирования в итоговом отчете?

Графическое представление результатов тестирования способствует более полному и быстрому пониманию текстовой информации.

10. Что содержится в разделе «Детализированный анализ качества»?

В данной части отчета описывается более подробная информация о проверенных частях функционала, устанавливается качество каждой проверенной части функционала (модуля) в отдельности, дается аргументация выставленного уровня качества. Как правило, данный раздел отчета представляется в табличной форме. В зависимости от вида проводимых тестовых активностей эта часть отчета будет отличаться.

При оценке качества функционала на уровне Smoke теста оно может быть либо приемлемым (Acceptable), либо неприемлемым (Unacceptable). Если все наиболее важные функции работают корректно, то качество всего функционала на уровне Smoke может быть оценено, как приемлемое.

Если это релизная или предрелизная сборка, то для выставления приемлемого качества на уровне Smoke не должно быть найдено функциональных дефектов.

В части о детализированной информации качества сборки следует более подробно описать проблемы, которые были найдены во время теста.

При оценке качества функционала на уровне Defect Validation указываются результаты валидации дефектов, а именно:

общее количество всех дефектов, поступивших на проверку;

количество неисправленных дефектов и их процент от общего количества;

список дефектов, которые не были проверены и причины, по которым этого не было сделано;

наглядная таблица с неисправленными дефектами.

По вышеуказанным результатам выставляется качество теста. Если процент неисправленных дефектов меньше 10 %, то качество приемлемое (Acceptable), если больше 10 %, то качество неприемлемое (Unacceptable).

При оценке качества функционала на уровне New Feature Test (полный тест нового функционала) качество отдельно проверенного функционала может быть высокое (High), среднее (Medium), низкое (Low).

Важно отдельно указывать информацию о качестве каждого модуля нового функционала с аргументацией выставленной оценки.

При оценке качества функционала на уровне Regression Testing нужно анализировать динамику изменения качества проверенной функциональности в сравнении с более ранними версиями сборки. Для этого приводится сравнительная характеристика каждой из частей функционала в сравнении с предыдущими версиями сборки, даются ясные пояснения о выставлении соответствующего качества каждой функции в отдельности. Также, как и у предыдущего вида тестов, качество может быть высокое (High), среднее (Medium), низкое (Low).

Список самых критичных дефектов содержит 3–5 ссылок на наиболее критичные дефекты с указанием их названия и уровня критичности.

11. Что содержится в разделе «Рекомендации»?

Рекомендации включают краткую информацию о всех проблемах приложения с пояснениями, насколько оставшиеся проблемы являются критичными для конечного пользователя. Обязательно указывают функционал и дефекты, скорейшее исправление которых является наиболее приоритетным. Кроме того, если сборка является релизной или предрелизной, то любое ухудшение качества является критичным и важно это обозначить.

## Лабораторная работа №7

1)Программа: Telegram

2) Экспертный подход:

1. Информативность – Система полностью оповещает о совершённых действиях если пользователь не настроит обратное.
2. Схожесть с реальным миром – 13 официальных переводов, около 50 любительских, к тому же можно создать свой. Большинство объектов оснащено иконками.
3. Свобода действий - Приложение позволяет удалять и редактировать сообщения
4. Единообразие - положительно
5. Предотвращение ошибок – Есть исчезающий текст в полях, в остальном всё элементарно ясно.
6. На виду – есть поиск по сообщениям, список документов, ссылок и фотографий из переписки
7. Гибкость – действия быстрые и чаще всего в одном “клике” доступности.
8. Дизайн – Положительно, можно изменять и модифицировать
9. Понимание проблем – ошибки помечаются красным восклицательным знаком и всплывающим сообщением
10. Справочные материалы – имеются

3)Дефекты не выявлены

4)Пользовательский подход:

5)Цель заказчика - Создать мессенджер, монетизируемый с помощью рекламы, который будет удобен пользователю и где пользователь будет защищён

Цель пользователя – Получить мессенджер с надёжной приватизацией

6)Целевая аудитория:

Пол – не важен

Возраст – 18 ~ 70

Образование - не важно

Доход – не важен

Деятельность – не важна

Семейное положение – не важно

Нет других особых характеристик

7)Карта эмпатии

Что думает?

- Для него важна приватность и быстродействие

Что видит?

- Современная, информационная среда

Что слышит?

-

8)Сегментация:

Аудитория занимающаяся бизнесом

Аудитория занимающаяся новостями и развлекательным контентом

Аудитория рядовая

Разработчики ботов

9)

|  |  |
| --- | --- |
| Разработчики ботов | |
| Создание новых ботов для своих или чужих нужд | |
|  | Описание:  Увлечение/род занятий – программирование на любом языке |
| - | |
| Создание ботов | |
| Авторизация бота, работа со справочными материалами, реализация продукта. | |
| - | |

10)

|  |  |
| --- | --- |
| Шаг1 | Авторизация бота |
| Вопросы | Как авторизовать/создать бота |
| Пожелания | Доступный и понятный справочный материал |
| Эмоции | Зависит от профессионализма: От отрицательных до положительных |
| Комментарии |  |

|  |  |
| --- | --- |
| Шаг2 | Программирование |
| Вопросы | Какой язык доступен? Есть ли библиотеки? |
| Пожелания | Доступный и понятный справочный материал, наличие библиотек, как можно большее число доступных языков программирования |
| Эмоции | Зависит от профессионализма: От отрицательных до положительных |
| Комментарии |  |

11)Отобрал

12)Пронаблюдал

13) вопрос 10

Пользователь из аудитории, занимающейся бизнесом был доволен – 4/5

Пользователь из аудитории, занимающейся новостями и развлекательным контентом был полностью удовлетворён – 5/5

Пользователь из рядовой аудитории был полностью удовлетворён – 5/5

Пользователь, занимающийся разработкой был доволен, но из-за недостаточно хорошего справочного материала его действия были затруднены, большое разнообразие доступных языков программирования обрадовало пользователя – 4/5

Вопросы:

1. Что такое юзабилити?

Юзабилити *–* степень, с которой продукт может быть использован опреде- ленными пользователями при определенном контексте для достижения опреде- ленных целей с должной эффективностью, результативностью и удовлетворенно- стью. Юзабилити отражает степень удобства использования программного про- дукта конечными пользователями. Так как взаимодействие пользователя и про- граммного обеспечения осуществляется посредством пользовательских интерфей- сов, то понятие юзабилити прежде всего относится к процессу разработки пользо- вательских интерфейсов.

1. Какие существуют подходы к юзабилити-тестированию?

Проверка соответствия принципам обеспечения удобства пользования и корректного визуального представления в контексте функциональных требований посредством экспертной оценки (экспертный подход).

Изучение опыта взаимодействия пользователя с приложением через имита- цию поведения пользователей (пользовательский подход).

Для юзабилити-тестирования одного программного обеспечения могут при- меняться оба подхода (методика двойной проверки).

Далее рассмотрим вышеуказанные техники юзабилити-тестирования более подробно.

3.Охарактеризуйте экспертный подход юзабилити-тестирования.

При экспертном подходе в качестве пользователей выступают два и более экспертов (оптимальное количество для больших проектов 5–6 человек). Экспер- ты проходят основные сценарии поведения пользователей и анализируют их с точки зрения:

* стандартов юзабилити для конкретного типа программного продукта (например, Android Material Design для мобильных приложений на платформе An- droid);
* общих принципов юзабилити (эвристики Якоба Нильсена);
* здравого смысла и опыта.

По результатам прохождения пользовательских сценариев составляется отчет о дефектах.

Преимущества экспертного подхода:

* быстрый в применении;
* эксперты гарантировано понимают общие задачи программного продукта. Недостаток данного подхода – субъективизм (эксперты не являются реаль-

ными пользователями).

4.Перечислите и дайте характеристику принципам юзабилити Якоба Нильсена.

70

1. Информированность о состоянии системы. Пользователь всегда должен ориентироваться и четко понимать, что происходит в системе. Взаимодействие между пользователем и системой должно быть как можно более логичным и быстрым. Для этого целесообразно реализовать обратную связь в виде сообщений подтверждения успешности выполнения действий, запросов на подтверждение удаления, сообщений об ошибках и др.
2. Схожесть системы с реальным миром. Система должна общаться с пользо- вателем на понятном ему языке. Использование инфографики, слов, фраз и поня- тий, знакомых пользователю в реальном мире, намного предпочтительнее, чем ис- пользование специализированных терминов.
3. Свобода действий. Необходимо предоставить пользователям возможность отмены действий, а также возврата к ранее осуществленным действиям.
4. Единообразие и стандарты. Не следует вводить в заблуждение пользовате- ля, описывая одни и те же вещи разными словами и терминами.
5. Предотвращение ошибок. Важно свести к минимуму количество условий, в которых могут быть допущены ошибки. Например, можно давать пользователям подсказки, поясняющие, какую информацию надо вводить в текстовые поля.
6. На виду, а не в памяти. Не следует вынуждать пользователя запоминать большое количество объектов, действий и опций. Вся необходимая информация должна быть размещена в пределах доступности для пользователя.

7.Гибкость и эффективность. Не следует нагружать пользователей лишней информацией – предоставьте им возможность совершать часто повторяющиеся действия как можно быстрее и проще.

8.Эстетичный и минималистичный дизайн. Тексты не должны содержать бесполезной или устаревшей информации. Каждое лишнее слово делает восприя- тие все более трудным и лишает посетителя возможности достичь цели.

9.Понимание проблем и их решение. Сообщения об ошибках должны быть выражены на понятном пользователю языке, как можно более точно описывать проблему и предоставлять возможные варианты ее решения.

10.Справочные материалы и документация. Даже если система может ис- пользоваться без документации, в процессе работы с ней все же может потребо- ваться справочная информация. Подобные документы должны составляться таким образом, чтобы в них легко было найти необходимое.

5.Охарактеризуйте пользовательский подход юзабилити-тестирования.

Пользователям (3–5 человек из каждого сегмента целевой аудитории), согла- сившимся участвовать в тестировании, предлагают пройти наиболее распростра- ненные и наиболее проблемные сценарии. Эксперт протоколирует действия поль- зователя, фиксирует все в видеоформате, чтобы отследить реакцию (эмоции) пользователя, но никак не влияет на действия пользователя.

Преимущества пользовательского подхода:

* объективные результаты (участвуют реальные пользователи);
* процесс легко измерим.

Возможные измерения при юзабилити-тестировании:

* время выполнения задачи;
* успешность выполнения задачи;
* эффект первого впечатления (например, сколько раз улыбнулся). Недостатки пользовательского подхода:
* длительный по времени;
* дорогой (если пользователей привлекают на платной основе);
* большое внимание следует уделить подбору пользователей.

Для реализации пользовательского подхода юзабилити-тестирования необхо- димо провести предварительную работу, которая включает следующие этапы:

1. Определение цели пользователя, цели бизнеса.
2. Исследование целевой аудитории: составление ее общего портрета, сегмен- тация на группы, описание персонажей как ярких представителей каждой группы.
3. Выявление контекста – ситуаций, при которых пользователь обращается к программному продукту.
4. Составление пользовательских сценариев. Рассмотрим особенности перечисленных этапов.

6.Что такое целевая аудитория?

группа пользователей, на которую ориентировано со- держание программного продукта. При исследовании целевой аудитории на пер- вом этапе необходимо составить ее общий портрет

7.Для чего необходимо изучать целевую аудиторию?

1. Социально-демографические характеристики: пол, возраст, образование, уровень дохода, сфера деятельности, семейное положение.
2. Психографические характеристики: стиль жизни, особенности личности, черты характера, жизненная позиция, система ценностей. Это более ценная ин- формация для проектирования, чем первая группа критериев. Например, если из- вестно, что целевая аудитория больше всего ценит время, можно спроектировать простой интерфейс и дать возможность получать не весь контент, а самое ценное для конкретной целевой группы, или предоставить инструменты персонализации каждого человека.
3. Поведенческие характеристики: повод для регистрации, искомые выгоды, частота посещаемости конкурентов, степень готовности к переходу на другой продукт, отношение к проекту (если он не новый) и т. д. Поведенческие характе- ристики целевой аудитории помогают понять привычки, мотивацию, круг интере- сов, проблемы, надежды и ожидания пользователя (как именно человек делает вы- бор, что влияет на решение купить продукт или отказаться от покупки, какой па- раметр является главным: качество товара, известная марка, стоимость, мнение друзей). Собирать эти данные очень сложно. Эта информация может быть у заказ- чика или конкурента, если проектируется новая версия уже существующего про- екта, либо ее нужно будет собирать через опросы целевой аудитории или состав- ление карт эмпатии.
4. Географические характеристики: страна, город, район. Если стоит задача по проектированию национальных продуктов или продуктов с геолокацией, то важность этих характеристик резко возрастает.

8.Какие существуют способы сбора информации о целевой аудитории?

1. Статистика запросов поисковых систем: позволяет оценить величину целе вой аудитории по числу поисковых запросов.
2. Анализ данных из социальных сетей и других публичных источников.
3. Опросы аудитории (анкетирование пользователей, использование опросной формы или регистрации на уже функционирующем сайте).
4. Данные счетчика посещений: для уже функционирующих web-ресурсов позволяют изучить все действия пользователей и конкретизировать распределение аудитории по регионам, времени и др.

5.Составление карты эмпатии

9.Что такое эмпатия? Что такое карта эмпатии?

Это психологический термин, который отражает способность пони- мать чувства и настроения других людей, умение поставить себя на место другого.

10.Из каких характеристик состоит профиль персонажа?

фотографию, имя, возраст, пол, образование, профессию, семейное положение, личностные характеристики

11.Какой персонаж называют ключевым, а какой второстепенным?

для них будет происходить проектирование информационной системы

12.Что такое сценарий?

это ситуация взаимодействия персонажа с продуктом