**Лабораторная работа 3**

**Классификация видов тестирования**

Тестирование можно классифицировать по очень большому количеству признаков, к тому же в различных источниках классификации могут различаться. Эталонной классификации не существует. Но, несмотря на это классификация позволяет упорядочить знания и значительно ускоряет процессы планирования тестирования и разработки тест-кейсов.

У разных авторов можно найти различные схемы классификации.

**Ваша задача:**

1. Найти схему классификации различных видов тестирования с описанием (можно по ссылкам в лекции, можно воспользоваться книгой С.Куликова «Тестирование программного обеспечения», размещенной на дискстейшене или найти свои источники).

2. По итогам знакомства с видами тестирования необходимо заполнить таблицу, описать в ней не менее 20 различных видов тестирования и привести свои уникальные примеры тестов на основе своего курсового проекта (или проекта, протестированного во второй лабораторной работе).

|  |  |  |
| --- | --- | --- |
| **Вид тестирования** | **Краткое определение вида тестирования** | **Тестовые проверки** |
| 1. Тестирование пользовательского интерфейса | Тестирование интерфейса ПО на соответствие требованиям (размер, шрифт, цвет и т. д.) | · Проверить, что кнопка «Добавить в корзину» соответствует макету (размер, цвет, шрифт).  · Проверить, что текст на кнопках не обрезается при изменении размера окна.  · Проверить, что все элементы формы заказа выровнены и не выходят за границы окна.  · Проверить, что при нажатии на иконку корзины открывается список товаров.  · Проверить, что кнопка «Оформить заказ» становится активной только при наличии товаров в корзине.  · Проверить, что в тёмной теме интерфейса текст остаётся читаемым.  · Проверить, что иконки блюд загружаются корректно и не растягиваются. |
| 1. Тестирование удобства использования | Установление степени удобства использования, понятности и привлекательности для пользователей разрабатываемого продукта | · Проверить, может ли новый пользователь без инструкции оформить заказ за 3 минуты.  · Провести A/B тестирование двух вариантов кнопки «Оформить заказ» и выяснить, какая понятнее.  · Проверить, можно ли добавить блюдо в корзину одним кликом.  · Проверить, легко ли отменить заказ, если пользователь передумал. |
| 1. Тестирование безопасности | Тестирование защищенности системы, анализ рисков, связанных с обеспечением целостного подхода к защите приложения, атак хакеров, вирусов, несанкционированного доступа к конфиденциальным данным | · Проверить, что пароли пользователей хранятся в зашифрованном виде в БД.  · Попытаться войти в аккаунт с SQL-инъекцией.  · Проверить, что после выхода из аккаунта доступ к корзине блокируется. |
| 1. Нагрузочное тестирование | Тестирование работы системы под разными уровнями нагрузки до предельного значения, которое она должна выдерживать и которые содержатся в требованиях | · Проверить, выдерживает ли сервер 5000 запросов в минуту на получение списка блюд.  · Проверить, что при высокой нагрузке оформление заказа занимает не более 5 секунд.  · Проверить, как приложение ведет себя при 1000 активных сессиях в базе данных.  · Проверить, что при резком увеличении нагрузки (скачок с 10 до 500 запросов/сек) сервер не падает.  · Проверить, как ведет себя приложение при загрузке 1000 изображений блюд одновременно.  · Проверить, как приложение ведет себя при 10 000 параллельных регистрациях новых пользователей. |
| 1. Объемное   тестирование | Тестирование, которое проводится для получения оценки производительности при увеличении объемов данных в базе данных приложения | · Проверить, работает ли поиск по меню при 1 миллионе записей в базе данных.  · Проверить, увеличивается ли время отклика сервера при добавлении 5 млн пользователей.  · Проверить, насколько быстро загружается статистика продаж за последние 5 лет.  · Проверить, не падает ли сервер при массовом удалении 1 млн старых заказов.  · Проверить, как ведёт себя приложение при загрузке 500 МБ изображений блюд. |
| 1. Ручное тестирование | Выполняется тестировщиком вручную, без использования инструментов автоматизации | · Проверить вручную, что при оформлении заказа клиент получает корректное уведомление.  · Проверить, что при удалении товара из корзины его стоимость корректно пересчитывается.  · Проверить, что при выходе из аккаунта пользователь не может оформить заказ.  · Проверить, как приложение ведёт себя при обрыве интернет-соединения. |
| 1. Автоматизированое тестирование | Проверки выполняются с использованием программных средств для выполнения тестовых сценариев | · Запустить нагрузочный автотест, отправляющий 1000 запросов к серверу за 1 минуту.  · Автоматизировать проверку правильности расчета итоговой суммы заказа.  · Запустить автотест, проверяющий, что при отмене заказа его статус меняется на «Отменен». |
| 1. Позитивное тестирование | Тестирование, при котором ПО или его элементы реагируют согласно требованиям на совершенные действия при использовании корректных тестовых данных | · Оформить заказ, введя корректные данные (имя, адрес, номер телефона) – заказ должен пройти успешно.  · Добавить блюдо в корзину – оно должно отобразиться со своей ценой.  · Успешно зарегистрировать нового пользователя с уникальным email. |
| 1. Негативное тестирование | Вид тестирования, направленный на проверку того, что система или приложение ведут себя должным образом в негативных ситуациях, то есть, когда они получают недопустимые или неожиданные входные данные | · Ввести в поле номера телефона буквы – система должна отклонить ввод.  · Ввести пустой email при регистрации – система должна показать ошибку.  · Ввести минусовую сумму – система должна запретить ввод.  · Закрыть приложение во время оформления заказа – не должно быть ошибок при перезапуске. |
| 1. Тестирование белого ящика | Полный доступ тестировщика к коду системы | · Проверить, что метод расчета стоимости заказа правильно суммирует цены товаров.  · Проверить, что база данных обновляется корректно после отмены заказа.  · Написать тесты для функции проверки доступности блюда перед оформлением заказа.  · Проверить, что в коде нет жестко закодированных паролей.  · Проанализировать логи работы сервера на предмет неожиданных ошибок. |
| 1. Тестирование серого ящика | Частичный доступ к коду проекта, комбинация методов белого и черного ящиков | · Проверить, какие SQL-запросы выполняются при оформлении заказа.  · Проверить, какие запросы отправляются на сервер при нажатии кнопки «Заказать».  · Внести изменения в тестовую базу данных и проверить, как они отобразятся в UI.  · Оценить производительность выполнения SQL-запросов к базе с 1 млн записей.  · Изменить данные пользователя в базе и проверить, как они отобразятся в интерфейсе. |
| 1. Тестирование чёрного ящика | Не предполагает доступа к системе, т.е. основывается на работе исключительно с внешним интерфейсом тестируемой системы | · Проверить, что при успешном входе пользователь попадает на главную страницу.  · Проверить, что при вводе некорректного логина появляется сообщение об ошибке.  · Проверить, что фильтр меню по категории отображает только нужные блюда.  · Проверить, что поиск по названию блюда выдает корректные результаты.  · Проверить, что при удалении товара из корзины его стоимость пересчитывается. |
| 1. Сценарное тестирование | Тестирование по заранее подготовленной тестовой документации | **Сценарий 1** (Оформление заказа):   1. Войти в аккаунт. 2. Добавить несколько блюд в корзину. 3. Оформить заказ. 4. Убедиться, что заказ появился в истории заказов.   **Сценарий 2** (Регистрация нового пользователя):   1. Открыть форму регистрации. 2. Ввести имя, email, телефон и пароль. 3. Подтвердить регистрацию. 4. Войти в систему с новыми учетными данными. |
| 1. Исследовательское тестирование | Тестовую документацию составляют по ходу проверки сервиса или приложения, а не заранее. | · Открыть разные страницы приложения и искать несоответствия в дизайне.  · Ввести случайные данные в форму заказа и проверить реакцию системы.  · Менять настройки приложения и проверять, не ломается ли логика работы.  · Прокликать все возможные кнопки, искать неожиданные ошибки.  · Попробовать оформить заказ с минимальными и максимальными возможными данными. |
| 1. Статическое   тестирование | Предполагает, что программный код во время тестирования не будет выполняться | · Проверка пользовательской документации на соответствие требованиям.  · Анализ кода на соответствие стандартам кодирования.  · Проверка структуры базы данных перед ее использованием.  · Анализ дизайна интерфейса на соответствие гайдам |
| 1. Динамическое   тестирование | Подразумевает запуск кода для проведения функциональных и нефункциональных проверок ПО | · Запуск приложения и проверка, как оно ведет себя при разных входных данных.  · Измерение времени загрузки главной страницы.  · Тестирование взаимодействия с базой данных в реальном времени. |
| 1. Модульное   тестирование | Проверяется отдельная часть приложения | · Написание теста для метода расчета стоимости заказа.  · Проверка работы функции валидации email-адресов.  · Тестирование метода получения списка блюд из базы данных.  · Проверка работы функции добавления товара в корзину. |
| 1. Интеграционное   тестирование | Направленный на проверку корректного взаимодействия нескольких компонентов системы между собой | · Проверка связи между интерфейсом и базой данных (добавление заказа).  · Проверка работы модуля аутентификации с базой данных пользователей. |
| 1. Сквозное   тестирование | Проверяется система целиком | · Полный тест процесса оформления заказа:  1. Вход в систему.  2. Добавление товаров в корзину.  3. Оформление заказа.  4. Получение подтверждения.  5. Тестирование обновления информации о доставке в реальном времени. |
| 1. Операционное   тестирование | Процесс проверки системы на удовлетворение всех потребностей пользователя и соответствия бизнес‑требованиям | · Проверка работы приложения на разных версиях Windows.  · Тестирование приложения на устройствах с разными параметрами (слабый/мощный ПК).  · Проверка работы при медленном интернете.  · Проверка поведения приложения при нехватке оперативной памяти.  · Анализ загрузки CPU при выполнении ресурсоемких операций. |

1. В отчете разместить схему классификации, которую вы использовали для выполнения данного задания с отмеченными на ней видами тестирования, которые вы внесли в таблицу, например, так:

