|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Назва тестової документації** | **№** | **Основні характеристики** | **Переваги** | **Недоліки** |
| Чек-ліст | 1 | **Структурованість**(Для розуміння, що потрібно робити)  **Компактність**(Не має містити зайвих деталей)  **Актуальність**(Повинен бути актуальним і оновлюватися при необхідності)  **Ієрархія**(Може бути структурованим, від загальних завдань до деталізації конкретної дії) | * Використання чеклістів сприяє структуруванню інформації у співробітника. * При правильному записі необхідних дій у співробітника з'являється однозначне розуміння завдань. Це сприяє підвищенню швидкості навчання нових співробітників. * Чеклісти допомагають уникнути невизначеності та помилок, що пов'язані з людським фактором. Збільшується покриття тестами програмного продукту. * Підвищується ступінь взаємозамінності співробітників. * Економія робочого часу. Написавши чекліст одного разу, його можна використовувати повторно, з огляду на актуальність інформації. | * Обмежений контекст(Чек-лист може не забезпечувати огляду або розуміння складних проблем або завдань) * Створення та підтримка(Розробка та підтримка чек-листів може бути часомістким процесом, особливо в ситуаціях, коли необхідно часто оновлювати документ) * Недостатня гнучкість(Деякі завдання або ситуації можуть вимагати більше креативного підходу або нестандартного контролю) * Загальність(Універсальні чек-листи можуть бути менш ефективними для вирішення специфічних завдань) |
| Тест-кейс | 2 | **Ідентифікатор**(Тест-кейс має унікальний номер або ідентифікатор для відстеження)  **Опис**(Короткий опис змісту тест-кейсу, його цілей і мети)  **Кроки**(Кожен крок має бути однозначним і детально описаним)  **Очікуваний результат**  **Фактичний результат**  **Статус**  **Пріоритет** | * Документація вимог(Допомагає перевірити, чи відповідає програма заданим специфікаціям і вимогам) * Ефективність і повторюваність(дозволяє ефективно і повторювано виконувати тестові сценарії, що дозволяє найти та виправити дефект на ранніх стадіях розробки) * Систематичне тестування(Це дозволяє покрити всі основні функції та можливі сценарії взаємодії з програмою) * Можливість автоматизації(Можуть бути автоматизовані, що збільшує продуктивність тестування та зменшує ризик помилок людського фактора) * Відстеження прогресу(можна відстежувати прогрес тестування і оцінювати, скільки тестових сценаріїв залишилося виконати) * Точне відтворення проблем(Тест кейс можна використовувати для точного відтворення проблеми після виправлення) | * Часові та ресурсні затрати(Створення та підтримка великої кількості тест-кейсів може займати багато часу та ресурсів) * Обмежена охопленість(Тест-кейси можуть не здатні показати всі можливі сценарії взаємодії користувача з програмою. Може не вистачити гнучкості для виявлення деяких проблем) * Залежність від людського фактора(Успішність тестування залежить від вміння тестувальника правильно виконати тест-кейс) * Застарілість(При зміні в програмному продукті тест-кейс може стати застарілим і вимагати оновлення або переробки) * Схильність до дублювання(У великих проектах може статися, що різні тест-кейси повторюють одні й ті ж дії або мають спільні етапи виконання) |
| Користувацький сценарій | 3 | **Попередні умови**(Опис передумов, які повинні бути виконані перед початком сценарію)  **Основна послідовність дій**(Кроки, які користувач виконує в процесі взаємодії з системою мають бути в чіткі послідовності)  **Неочікувані результати**(Опис можливих неочікуваних або небажаних результатів, які можуть виникнути)  **Результати**  **Залежності**(Посилання на інші сценарії або умови, які повинні бути виконані перед або після даного сценарію) | * Зосередженість на користувачеві(Користувацькі сценарії зосереджені на тому, як реальні користувачі будуть взаємодіяти з програмним продуктом. Це дозволяє розробникам та дизайнерам краще зрозуміти потреби та очікування своєї аудиторії) * Візуалізація(Сценарії можуть бути легко візуалізовані у вигляді тексту, діаграм або рисунків, що допомагає зрозуміти послідовність подій та дій користувача) * Виявлення потенційних проблем(Використання сценаріїв допомагає виявити потенційні проблеми або неочікувані ситуації, з якими користувачі можуть зіткнутися) * Тестування та валідація(Можуть бути використані як основа для тестування та валідації програмного продукту з позиції користувача) * Легке зрозуміння замовниками(Сценарії є ефективним способом зрозуміти, як буде працювати програмний продукт, для замовників та зацікавлених сторін) | * Застарілість(Якщо програмний продукт зазнає змін, користувацькі сценарії можуть застаріти та стати неактуальними, що вимагає регулярного оновлення) * Важкість створення(Створення користувацьких сценаріїв може бути часомістким процесом, особливо при невизначених проектах) * Помилки(У процесі створення сценаріїв можуть робитися припущення про поведінку користувачів, які не завжди відповідають реальності) * Витратність(При розробці великих проектів може вимагатися значна кількість часу та ресурсів на створення та підтримку користувацьких сценаріїв) * Нехватка реалізму(Сценарії можуть не завжди повністю відображати реальні умови та ситуації, з якими користувачі можуть зіткнутися) |