1. Відкрийте веб-сайт Вашої улюбленої соцмережі.

Опишіть, які дії виконуються на кожному з рівнів тестування , якби у Вас була повна документація.

Instagram

1) Unit testing (проводиться розробником) на цьому рівні тестуються окремі модулі соцмережі. В інстаграмі це реєстрація, налаштування профілю, додавання фото (можливість додати фото с галереї або зробити фото в моменті і одразу додати), додавання коментарів, додавання сторіз, діректу, можливість додавати і відписатись від підписників. Кожен з цих модулів тестується окремо на рівні коду дуже часто за допомогою авто тестів. Розробники тестують: компоненти, юніти чи модулі, код та структура даних, класи, моделі баз даних. Типові баги: неправильна функціональність, проблеми передачі даних, неправильна логіка та помилки в коді.

2) Integration testing (проводиться розробниками) на цьому рівні об’єднують окремі модулі в єдину систему або групи модулів і перевіряють наскільки коректно працюють модулі в загальній системі. Після об’єднання в систему перевіряють чи працює кожен модуль соцмережі, як вони взаємодіють між собою, чи все працює гармонійно. Розробники тестують підсистеми, бази даних, інфраструктура, API, мікросервіси. Типові баги: неправильні дані, втрачені дані, неправильне кодування; неправильний формат даних під час передачі даних між модулями; помилки у "спілкуванні" між модулями.

3) System testing. На цьому рівні Інстаграм тестують вже тестувальники на отченні, яке будуть застосовувати користувачі. Так як Інстаграм це мобільний додаток його бажано тестувати на IOS та Android оточенні. Далі тестуються всі функції згідно тест-кейсів. Ми реєструємося в Інстаграмі, налаштовуємо профіль, додаємо фото, коментарі, перевіряємо чи працює діректу, подаємо підписників, видаляємо підписників. Якщо якась функція не працює пишемо баг репорти.

4) Acceptance testing. На цьому етапі перевіряємо відповідність системи вимогам та побажанням замовника. Звіряємо чи виглядає наш Інстаграм так, як хотів його бачити замовник.

2. У вас є програма-калькулятор (наприклад, стандартний калькулятор Windows) та 5 хвилин на проведення приймального тестування. Що б Ви протестували і чому?

Перш за все я б перевірила чи взагалі вона працює (не зламали щось при виправленні багів). Перевірка простих, основних функцій +, -, %.
Потім звірила те, ПЗ що в нас вийшло з вимогами замовника. (дизайн, кольори)

3. Вам необхідно провести системне тестування банківської програми (наприклад, Приват24/Mono) але у Вас відсутня будь-яка документація. Опишіть - 5 головних, на Ваш погляд, Use кейсів для тестування такої програми

UC 1: Переказ на картку в Приват24
Діюча особа: клієнт банку
Передумови: користувач знаходиться на головній сторінці сайту
Основний сценарій:

- користувач натискає кнопку “Переказ на картку”
- сайт відображає форму для заповнення
- користувач заповнює форму і натискає “Продовжити”
- відкривається сторінка Підтвердження
- користувач натискає “Сплатити”
- сайт показує сторінку Результат

Альтернативні сценарії:

- користувач натискає кнопку “Переказ на картку”
- сайт відображає форму для заповнення
- користувач заповнює форму і натискає “Продовжити”
- не натискається кнопка та інші сценарії буде баг

UC 2: Платежі
Діюча особа: клієнт банку
Передумови: користувач знаходиться на головній сторінці сайту
Основний сценарій:

- користувач натискає кнопку “Платежі”
- сайт відображає перелік платежів, які можна здійснити
- користувач натискає наприклад “Комунальні платежі”
- відкривається сторінка Комунальні платежі
- користувач натискає на адресу
- сайт показує список платежів

- користувач вибирає платежі і натискає “Продовжити”
- Відкривається сторінка Підтверження
- користувач натискає “Сплатити”
- відкривається сторінка “Платіж успішний”

Альтернативні сценарії:

- користувач натискає кнопку “Платежі”
- сайт відображає перелік платежів, які можна здійснити
- користувач натискає наприклад “Комунальні платежі”
- не натискається кнопка та інші сценарії буде баг

UC 3: Добро в Приват24
Діюча особа: клієнт банку
Передумови: користувач знаходиться на головній сторінці сайту
Основний сценарій:

- користувач натискає кнопку “Добро”
- сайт відображає форму Благодійність
- користувач вибирає фонд для допомоги і натискає на нього
- відкривається сторінка Благодійність
- користувач пише суму і натискає “Продовжити”
- сайт показує сторінку Підтверження
- користувач натискає “Пожертвувати”
- відкривається сторінка Результат “Платіж успішно здійснено”

Альтернативні сценарії:

- користувач натискає кнопку “Добро”
- сайт відображає форму Благодійність
- користувач вибирає фонд для допомоги і натискає на нього
- відкривається сторінка Благодійність
- користувач пише суму і натискає “Продовжити”
- не натискається кнопка та інші сценарії буде баг

UC 4: Перевірка транзакцій по карті
Діюча особа: клієнт банку
Передумови: користувач знаходиться на головній сторінці сайту
Основний сценарій:

- користувач натискає на карту
- сайт відображає повну інформацію про карту та виписки
- користувач вибирає один з переказів
- відкривається сторінка Перекази
- користувач може переглянути Квітанцію, натискає на “Квітанція”
- сайт показує квітанцію по платежу
- користувач натискає на пусте поле

- вікно Квітанція закривається

Альтернативні сценарії:

- користувач натискає на карту
- сайт відображає повну інформацію про карту та виписки
- користувач вибирає один з переказів
- відкривається сторінка Перекази
- користувач може переглянути Квітанцію, натискає на “Квітанція”
- не натискається кнопка та інші сценарії буде баг

UC5 : Перевірка історії транзакцій
Діюча особа: клієнт банку
Передумови: користувач знаходиться на головній сторінці сайту
Основний сценарій:

- користувач натискає кнопку “Історія”
- сайт відображає список транзакцій в зворотньому порядку
- користувач вибирає любу транзакцію
- відкривається сторінка з інформацією по платежу
- користувач натискає “Х” інформація закривається

Альтернативні сценарії:

- користувач натискає кнопку “Історія”
- сайт не відображає список транзакцій
- не натискається кнопка та інші сценарії буде баг

4. Опишіть усі підходи інтеграційного тестування у разі тестування соціальної мережі Facebook

Знизу вгору (Bottom-up) - всі низькорівневі модулі, процедури або функції збираються докупи і потім тестуються. Після чого збирається наступний рівень модулів щодо інтеграційного тестування. Цей підхід вважається корисним, якщо всі або практично всі модулі, що розробляються, готові. Також цей підхід допомагає визначити за результатами тестування рівень готовності програми.

перевірка роботи “реакцій” до коментаря новин головної стрічки
перевірка роботи “відповіді” на коментар новин головної стрічки
тестування “реакції” на новину головної стрічки

Зверху донизу (Top-down integration) - спочатку тестуються всі високорівневі модулі, і поступово додаються низькорівневі. Усі модулі нижчого рівня симулюються заглушками з аналогічною функціональністю, потім замінюються реальними активними компонентами.

Основні модулі Історії, Reels, Massenger. Почати перевіряти з цих модулів потім менш важливі.
Великий вибух ("Big Bang" Integration) - всі або практично всі розроблені модулі збираються разом у вигляді закінченої системи або її основної частини, а потім проводиться інтеграційне тестування. Такий підхід добре підходить для збереження часу. Проте якщо тест кейси та його результати записані неправильно, тоді сам процес інтеграції сильно ускладниться.