Фаза 1: Анализ требований и планирование (Неделя 1)

Действия:

1) Необходимо тщательно провести анализ требований на предмет полноты, непротиворечисовсти и осуществимости. Здесь возможны два варианта развития событий: Если документация отсутствует — провести анализ на основе функционала(составить чек-лист, например). Если же документация есть, то ознакомиться с ней.
2) Далее необходимо согласовать с разработчиками и аналитиками понимание требований, чтобы у всех было одно представление и не было разногласий в понимании.
3) Затем выявляем основные функции продукта:

- Просмотр каталога товаров (поиск товаров, фильтрация товаров, сортировка товаров);
- Добавление товаров в корзину (добавление товара, удаление товара);
- Оформление заказа (заполнение формы);
- Выбор способа доставки (заполнение формы);
- Оплата через сторонний сервис (интеграция API: SberPay Api, Яндекс Доставка API, например).

4) Определяем ключевые риски:

- Уменьшение объема продаж (коммерческий тип риска);
- Уменьшение количества постоянных клиентов (коммерческий тип риска);
- Несоответствие продукции описанию (операционный тип риска);
- Нестабильность платежного сервиса (потеря транзакций с оплатой, например: операционный тип риска);
- Несоответствие статуса оплаты (операционный тип риска);
- Проблемы с отображением в разных браузерах;
- Ошибки при оформлении заказа (потеря данных);
- Безопасность данных пользователей (несанкционированный доступ к данным, утечка данных);
- Потеря связи с клиентом(производственный тип риска) .

5) Составим стратегию тестирования (подходы, уровни тестирования, инструменты).

Оценим трудоемкость:

- 30% - Функциональное тестирование;
- 25% - Тестирование производительности:
    - 5% Нагрузочное тестирование (тестирование при заданной нагрузке);
    - 10% Стресс тестирование (что будет при незапланированной нагрузке, важна планируемая деградация);
    - 10% - тестирование производительности (как быстро работают функции);
- 10% Конфигурационное тестирование (проверка совместимости на различном железе/ОС/Браузерах)
- 10% Юзабилити тестирование (для анализа удобства использования сайта)
- 10% Тестирование локализации (вероятно наш сайт будет работать не только на одном языке)
- 15% Тестирование безопасности.

6) Создаем возможные тестовые кейсы и анализируем их трудоемкость.

Результат: Документ с результатами анализа требований, выявленными рисками и планом их минимизации, документ со стратегией тестирования, чек-лист ключевых сценариев, тест-кейсы, матрица трассируемости требований.

Фаза 2: Планирование тестирования (Неделя 2-3)

Действия:

1) Разработка подробного плана тестирования, который будет включать в себя:

- Определение стратегии тестирования (пирамида тестирования).
- Планирование видов тестирования (функциональное, нефункциональное, юзабилити, безопасности).
- Выбор уровней тестирования (модульное, интеграционное, системное, приемочное).
- Определение тестовой среды.
- Планирование ресурсов (время, персонал, инструменты).
- Составление графика тестирования.
- Оценка трудоёмкости.

- **Ресурсы:** Инструменты для управления проектами (Jira, Trello), шаблоны тестовой документации.
- **Результат:** Документ с планом тестирования, оценкой трудоемкости, графиком тестирования.

Фаза 3. Тестирование альфа-версии (Недели 3–4)

Цель: Проверить базовую функциональность, формирование баг-репортов.

Действия:

• Модульное тестирование — проверка отдельных компонентов(модулей) на корректность работы:

- Просмотр каталога (поиск в каталоге, сортировка каталога);
- Работа корзины (добавление товара, удаление товара, обновление суммы итоговой в корзине);
- Оформление заказа (валидация полей, переход к оплате).

• Интеграционное тестирование — проверка соответствия функциональности заявленным требованиям. Использование методов black-box тестирования:

- Проверка API (например, между фронтендом и бэкендом корзины) с использованием Postman или Insomnia;
- Тестирование платежного шлюза;
- Кросс-браузерность (Chrome, Firefox, Safari) через BrowserStack;
- Адаптивность (мобильные устройства);
- Тестирование локализации;
- **Регрессия:** Быстрая проверка критичных багов после фиксов.

- **Юзабилити-тестирование:** оценка удобства использования интерфейса. Провести юзабилити-тестирование с привлечением нескольких пользователей.

Ресурсы: Альфа-версия приложения, браузеры разных типов, инструменты для отслеживания багов (Jira, Bugzilla), протоколирование сессий юзабилити-тестирования.

Результат: Отчет о тестировании альфа-версии с описанием найденных дефектов и их приоритетов. Проверена базовая функциональность сайта и собраны баг-репорты.

Фаза 4. Углубленное тестирование (Недели 4–8)

Цель: Тестирование бета-версии.

Действия:

- **Системное тестирование:** проверка всей системы в целом.
- **Функциональное тестирование:** более глубокая проверка функциональности, включая сценарии граничных значений, тестирование на стресс и производительность.
- **Нефункциональное тестирование:** тестирование производительности (нагрузка, время отклика), безопасности (авторизация, защита от SQL-инъекций, проверка на уязвимости), совместимости (разные браузеры, устройства).
- **Приемочное тестирование:** тестирование системы пользователями из целевой аудитории.
- **Применение техник:** классы эквивалентности, граничные значения (например, цена товара = 0).
- **Автоматизация:**
  - Автотесты на API (например, запросы к корзине);
  - Фаззинговое тестирование, если применимо (Случайная генерация данных для поиска).
- **Тестирование производительности:**
  - Проверка реакции сервера например 100+ одновременных заказов, важно гарантировать запас перед реальным сценарием, но и не перестараться;
  - Проверка скорости работы функций (проверка и грубо говоря установление эталона);
  - Проверка стабильности работы программы в течении продолжительного времени (утечки памяти, накопление ошибок округления и т.п.)
- **Тестирование безопасности:**
  - Проверка на SQL-инъекции в форме заказа, например, в любых местах, которые отправляются на сервер.
  - Уязвимости сессий .
- **Ресурсы:** Бета-версия приложения, инструменты для тестирования производительности (JMeter, LoadRunner), инструменты для тестирования безопасности (OWASP ZAP), платформа для проведения UAT.

- **Результат:** Отчет о тестировании бета-версии с описанием найденных дефектов и их приоритетов, полный набор тест-кейсов, отчет о производительности.

Фаза 5. Релизная подготовка (Недели 9–12)

Цель: Обеспечить готовность к релизу.

Действия:

- **Регрессионное тестирование:** Выполнение всех ключевых сценариев.
- **Запланированное тестирование:** Каждый день частичное тестирование основного функционала.Затем в ночное время, либо же на выходных, запуск полного тестирования.
- **Тестирование документации:**
  - Соответствие README;
  - Соответствие пользовательским подсказкам;
  - План тестирования
  - Тест-кейсы
  - Отчеты о тестировании (альфа, бета, пост-релизный)
  - Баг-репорты
  - Протоколы тестирования
- **Управление рисками:**

  В ходе анализа требований и планирования тестирования будут выявлены и оценены риски. Для каждого риска будет разработан план по минимизации. Риски будут отслеживаться на протяжении всего процесса разработки.
- **Используемые технологии:**
  - Jira/Trello для управления задачами и баг-трекинга
  - Браузеры (Chrome, Firefox, Edge)
  - Инструменты для тестирования производительности (JMeter, LoadRunner)
  - Инструменты для тестирования безопасности (OWASP ZAP)

Результат: Подготовленное к релизу приложение.

Фаза 6. Поддержка (Месяц после релиза)

Цель: Обеспечить быстрое реагирование на инциденты и их скорейшее устранение.

Действия:

- Приоритизация багов от пользователей;

- Анализ причин сбоев (например, падение платежного API, проблем с заказами);
- Обновление тест-кейсов под реальные сценарии;

Метрики:

- Количество найденных дефектов
- Плотность дефектов
- Критичность дефектов
- Время на исправление дефектов
- Уровень покрытия тестированием
- Уровень удовлетворенности пользователей