Частное учреждение профессионального образования

«Высшая школа предпринимательства (колледж)»

(ЧУПО «ВШП»)

Кафедра информационных технологий

Внедрение и поддержка компьютерных систем

Принципы тестирования

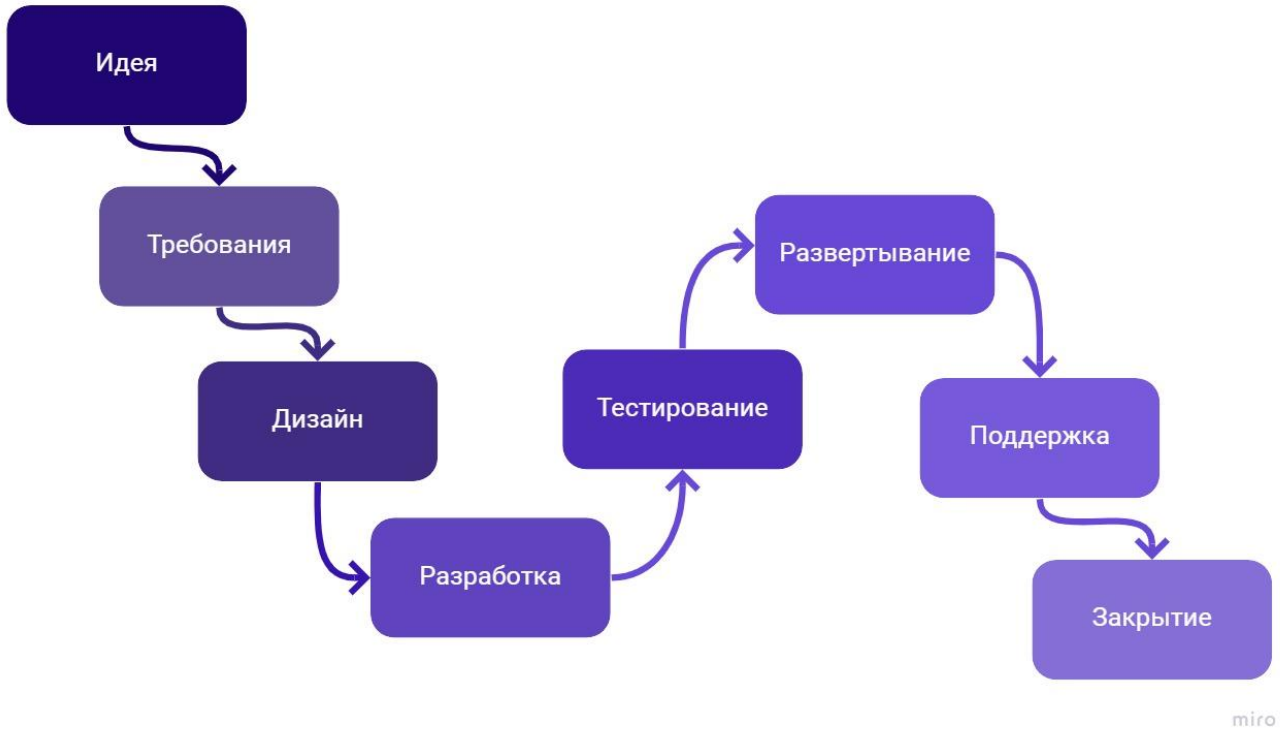
Принципы тестирования

Принцип 1 — Тестирование демонстрирует наличие дефектов
Принцип 2 — Исчерпывающее тестирование невозможно
Принцип 3 — Раннее тестирование
Принцип 4 — Скопление дефектов (принцип Парето)
Принцип 5 — Парадокс пестицида
Принцип 6 — Тестирование зависит от контекста
Принцип 7 — Заблуждение об отсутствии ошибок

Этапы жизненного цикла разработки

Этапы жизненного цикла разработки

Идея
Определение требований
Дизайн
  (архитектура) системы
Разработка
Тестирование
Развертывание
Поддержка
Закрытие



Каскадная модель

Каскадная модель

Каскадная модель – модель, в которой процесс разработки выглядит как поток, переходящий от одной стадии к другой в строгом порядке, без возможности пропуска стадии или возврата назад.

Обычно, порядок следующий:
Определение требований
Проектирование системы
Реализация
Тестирование
Запуск и поддержка

Итеративная (инкрементальная) модель

Итеративная (инкрементальная) модель

Итеративная модель – модель, в которой работы выполняются параллельно с непрерывным анализом полученных результатов и корректировкой последующих этапов работы.
Начальное планирование
Разработка, которая выполняется циклически:
Планирование
Написание требований
Архитектура / дизайн
Реализация
Тестирование
Оценка
Запуск и поддержка

Скрам (Scrum)

Канбан (Kanban)

Скрам (Scrum)

Скрам (Scrum) – это фреймворк, предназначенный для разработки, поставки и поддержки сложных продуктов.

Основными элементами фреймворка являются:
Небольшие Скрам-команды (< 10 человек)
Четкие роли каждого участника команды (Product-owner, Scrum-master, Development Team)
События (собрания / митинги)
Артефакты (беклог продукта, беклог спринта)
Правила

Особенности Скрам:
Наличие предварительного плана, который будет изменяться по мере реализации проекта
Фиксированные по длительности итерации разработки (1-4 недели)
Фиксированные команды разработки
Постоянная работа с заинтересованными сторонами (включая клиента)

Канбан (Kanban)

В переводе с японского **Канбан** = «рекламный щит, вывеска».

Канбан — это не методология. Это не замена **Scrum**. Это не метод разработки.

Канбан – это инструмент.

Testing

Обеспечение качества (QA — Quality Assurance)

Контроль качества (QC — Quality Control)

Верификация и валидация

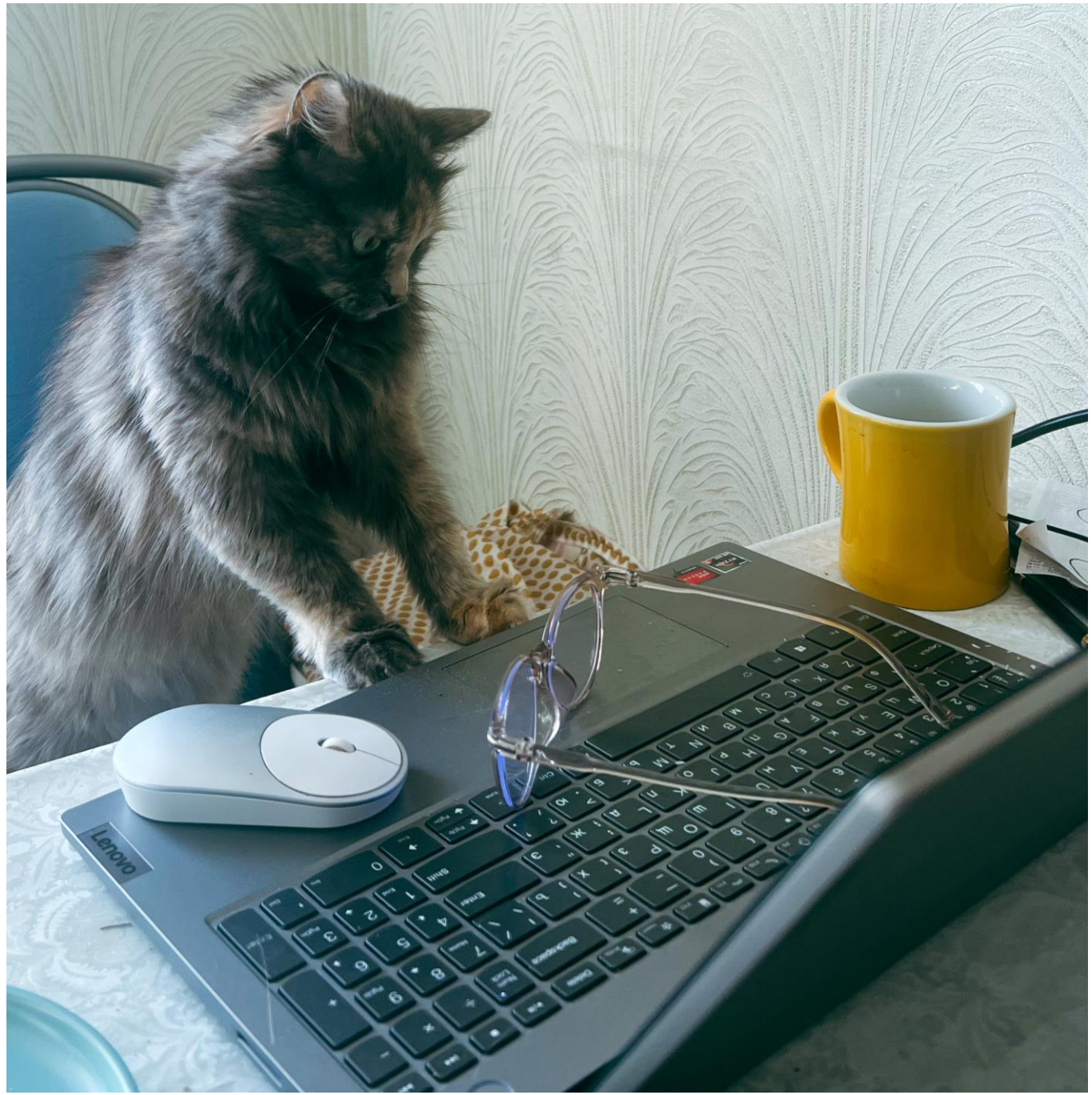
Верификация

Это статическая практика проверки документов, дизайна, архитектуры, кода и так далее.
Процесс **верификации** проходит без запуска кода.
Верификация всегда отвечает на вопрос **"делаем ли мы продукт правильно?"**.
Эта проверка связана с тем, что мы убеждаемся в том, что

система хорошо спроектирована и безошибочно.
Верификация всегда происходит *ДО* **валидации**.

Валидация

Процесс оценки конечного продукта, когда необходимо проверить: соответствует ли программное обеспечение ожиданиям и требованиям клиента.
Это динамический процесс, в отличии от **верификации**. Этот процесс всегда включает в себя запуск кода программы и отвечает на вопрос **"делаем ли мы правильный продукт?"**.
Процесс **валидации** всегда происходит *ПОСЛЕ* **верификации**, поэтому на этапе **валидаци**и можно уловить ошибки, которые процесс **верификации** не позволил нам найти.


Lenovo

Дефекты и отчеты о дефектах

Дефект (bug) — отклонение фактического результата от ожидаемого.

Отчёт о дефекте (bug report) — документ, который содержит отчет о любом недостатке в компоненте или системе, который потенциально может привести компонент или систему к невозможности выполнить требуемую функцию.

Атрибуты отчета о дефекте

Уникальный идентификатор (ID) — присваивается автоматически системой при создании баг-репорта.

Тема (краткое описание, Summary) — кратко сформулированный смысл дефекта, отвечающий на вопросы: Что? Где? Когда(при каких условиях)?

Шаги для воспроизведения (Steps To Reproduce) — описание четкой последовательности действий, которая привела к выявлению дефекта. В шагах воспроизведения должен быть описан каждый шаг, вплоть до конкретных вводимых значений, если они играют роль в воспроизведении дефекта.

Атрибуты отчета о дефекте(1)

Фактический результат (Actual result) — описывается поведение системы на момент обнаружения дефекта в ней. чаще всего, содержит краткое описание некорректного поведения(может совпадать с темой отчета о дефекте).

Ожидаемый результат (Expected result) — описание того, как именно должна работать система в соответствии с документацией.

Атрибуты отчета о дефекте(2)

Серьёзность дефекта (важность, Severity) — характеризует влияние дефекта на работоспособность приложения.

Приоритет дефекта (срочность, Priority) — указывает на очерёдность выполнения задачи или устранения дефекта.

Окружение (Environment) – окружение, на котором воспроизвелся баг.

Severity vs Priority

Серьёзность (severity) показывает степень ущерба, который наносится проекту существованием дефекта. Severity выставляется тестировщиком.

Срочность (priority) показывает, как быстро дефект должен быть устранён. Priority выставляется менеджером, тимлидом или заказчиком

Градация Серьезности дефекта (Severity)

Блокирующий (S1 – Blocker)
тестирование значительной части функциональности вообще недоступно. Блокирующая ошибка, приводящая приложение в нерабочее состояние, в результате которого дальнейшая работа с тестируемой системой или ее ключевыми функциями становится невозможна.

Критический (S2 – Critical)
критическая ошибка, неправильно работающая ключевая бизнес-логика, дыра в системе безопасности, проблема, приведшая к временному падению сервера или приводящая в нерабочее состояние некоторую часть системы, то есть не работает важная часть одной какой-либо функции либо не работает значительная часть, но имеется workaround (обходной путь/другие входные точки), позволяющий продолжить тестирование.

Градация Серьезности дефекта (Severity) (1)

Значительный (S3 – Major)
не работает важная часть одной какой-либо функции/бизнес-логики, но при выполнении специфических условий.

Незначительный (S4 – Minor)
часто ошибки GUI, которые не влияют на функциональность, но портят юзабилити или внешний вид. Также незначительные функциональные дефекты, либо которые воспроизводятся на определенном устройстве.

Тривиальный (S5 – Trivial)
почти всегда дефекты на GUI — опечатки в тексте, несоответствие шрифта и оттенка и т.п.

Градация Приоритета дефекта (Priority):

P1 Высокий (High)
Критическая для проекта ошибка. Должна быть исправлена как можно быстрее.

P2 Средний (Medium)
Не критичная для проекта ошибка, однако требует обязательного решения.

P3 Низкий (Low)
Наличие данной ошибки не является критичным и не требует срочного решения. Может быть исправлена, когда у команды появится время на ее устранение.