**Этапы SDLC**  
В ходе разработки перед переходом от текущего этапа к следующему необходимо выполнить каждый его шаг, для чего их следует лучше понимать. В этом отношении первые три этапа стараются дать ответы на проверочные вопросы, а последние три оптимизированы для достижения фактических результатов.

**Анализ требований** отвечает на вопрос «Какие проблемы требуют решений?»

**Планирование** отвечает на вопрос «Что мы хотим сделать?»

**Проектирование и дизайн** отвечает на вопрос «Как мы добьемся наших целей?»

**Разработка ПО** регулирует процесс создания продукта.

**Тестирование** регулирует обеспечение качественной работы продукта.

**Развертывание** регулирует использование финального продукта.

**Этап #1: Анализ требований**  
На этом этапе SDLC вам необходимо получить обратную связь и поддержку от соответствующих внутренних и внешних заинтересованных сторон. Вспомните мой недавний пример с разработкой приложения по учету времени: вам нужно будет широко задуматься о том, кто станут вашими потенциальными пользователями. Некоторые идеи будут включать ваших клиентов, дизайнеров, вашего начальника или других технических специалистов команды. В целом вы хотите ответить на следующий вопрос: «Какие проблемы требуют решений?» Быть внимательным и делать заметки будет очень полезно на этом этапе.  
  
Когда полученные ответы вас удовлетворят, вы сможете перейти к следующей фазе.

**Этап #2: Планирование**  
На этом этапе вы ищете ответ на следующий вопрос: «Что вы хотите сделать?» Этот вопрос может вдохновить вас на понимание юнит-экономики вашего плана (затраты и выгоды), факторов снижения рисков и ожидаемых стоимостей. По аналогии с планированием отпуска, вам нужно будет разложить ваши вещи и подумать о том, что следует взять с собой.  
  
Хороший пример:  
  
Я много читал об истории Инстаграма, чей этап планирования занял невероятно много времени. Это совпало с бурным ростом социальных сетей, поэтому взаимодействие пользователей с продуктом во многом все еще было неизвестно. Разработчики знали, что сильный первичный опыт (съемка, редактирование и обмен фотографиями) обеспечит рост, успех и высокую конверсию, а корректное планирование упростит проектирование, поэтому планировали соответствующе и тратили на дизайн много времени. Они всегда смотрели на шаг вперед и думали о будущем социальных сетей и электронной коммерции.  
  
Планируйте то, что вы можете контролировать, и помните о вещах, планировать которые вы не сможете. Это поможет вам получить прочную основу для перехода к третьему этапу.

**Этап #3: Проектирование и дизайн**  
К этому этапу вы уже должны знать требования вашего продукта и в целом понимать чего вы вообще хотите, и прежде чем приступить к написанию кода, этого понимания должно быть достаточно для ответа на следующий вопрос: «Как мы добьемся наших целей?» Иначе говоря, вам необходимо понять, что именно вы оптимизируете и проектировать соответствующе.  
  
Допустим, вы хотите создать безопасное, высокопроизводительное, эффективное и выдерживающее нагрузки приложение. Какой из этих четырех принципов наиболее для вас наиболее важен? Почему? Согласны ли с этим заинтересованные стороны из первого этапа? Важно обеспечить одобрение всех участников.  
  
После фазы дизайна вы наконец-то сможете засесть за клавиатуры, и внесение изменений в отношении времени и потраченных ресурсов будет неуклонно расти, а также буду постепенно накапливаться всевозможные малые факторы. В этой фазе для принятия окончательных решений по вопросам дизайна я рекомендую учитывать несколько основных его элементов: операционное превосходство, безопасность, надежность, эффективность производительности, и оптимизация затрат.

**Этап #4: Разработка ПО**  
На этапе разработки вы стремитесь не столько отвечать на вопросы, сколько произвести результаты, или, говоря точнее, вам необходимо склоняться к действиям и создать прототип или систему, испытать которую смогут другие. На этом этапе ваша задача – заручиться доверием заинтересованных сторон через воплощение образа мышления разработчика. Для соответствия результата ожиданиям критично при начале разработки следовать первым трем этапам.  
  
Доставайте ваш компьютер, убедитесь, что окружение способствует рабочей атмосфере, хватайте ваш горячий кофе – и приступайте к делу.

**Этап #5: Тестирование**  
Сотрудники в футболках с надписями вида «Разрабатывать круто, тестировать не очень» были для меня привычным зрелищем, но вы должны понимать, что не получится создать финальную версию продукта, пока вы на нем собаку не съедите. По завершению этого этапа вы должны будете в состоянии обеспечить рабочее состояние продукта. Отслеживайте ошибки и неточности, выслушивайте чужие точки зрения, и глубоко погружайтесь в вопрос с целью поиска тормозящих выход финального продукта ошибок. Вам просто необходимо обеспечить прочную основу.

**Этап #6: Развертывание**  
Возьмите ваш продукт и пользуйтесь им. Предложите заинтересованным сторонам из первого этапа пользоваться вашим продуктом в естественных условиях, начните отслеживать вовлеченность в продажи. Снова и снова прислушивайтесь к пользователям, ведь благодаря обратной связи через опросы и рекомендации вы сможете вернуться к первой фазе и начать собирать новые требования. И не забудьте отпраздновать релиз.

**Этапы STLC**

STLC имеет несколько взаимосвязанных фаз и в целом очень похож на SDLC. Эти фазы являются последовательными и называются:

* **Анализ требований** (Requirement Analysis): один из важнейших этапов, потому что именно на нем можно почти бесплатно исправить недостатки проекта. Этап анализа требований также определяет потенциальную потребность в автоматизированном тестировании и позволяет производить экономические расчеты затрат на рабочую силу на основе оценки проекта. На этом же этапе обсуждаются и документируются критерии начала и окончания тестирования.
  + Entry Criteria: BRS (Business Requirement Specification)
  + Deliverables: список всех проверяемых требований, технико-экономическое обоснование автоматизации (если применимо);
* **Планирование тестирования** (Test Planning): на этом этапе формируется план тестирования, т.е. мы определяем действия и ресурсы, которые помогут достичь целей тестирования (участники и их роли, инструменты, окружение). Во время планирования мы также пытаемся определить метрики, метод сбора и отслеживания этих метрик. План составляют исходя из требований, тестовой стратегии и анализа рисков.
  + Entry Criteria: Requirements Documents;
  + Deliverables: Test Strategy, Test Plan, and Test Effort estimation document.
* **Разработка тест-кейсов** (Test Case Development): подразумевает использование ручного и автоматизированного тестирования для достижения полного охвата функциональности программного обеспечения, при этом процесс основан на заранее установленных требованиях. Чаще всего тест-кейсы для автоматического тестирования пишутся отдельно, так как кейсы для ручного тестирования описаны в виде шпаргалок (cheat sheets).
  + Entry Criteria: Requirements Documents (Updated version);
  + Deliverables: Test cases, Test Scripts (if automation), Test data.
* **Настройка тестовой среды** (Test Environment Setup): в плане тестирования четко указано, какую тестовую среду следует использовать. На этом этапе STLC настраиваются операционные системы и виртуальные машины, развертываются инструменты тестирования, такие как Selenium, Katalon Studio, а также тестовая среда и базы данных проекта. Мы также обращаемся с запросами к DevOps и администраторам, если требуется поддержка.
  + Entry Criteria: Test Plan, Smoke Test cases, Test Data;
  + Deliverables: Test Environment. Smoke Test Results.
* **Выполнение тестов** (Test Execution): тесты выполняются на основе готовой тестовой документации и правильно настроенной тестовой среды. Все результаты тестирования регистрируются в Системе управления тестированием. Отрицательно пройденные тесты, в которых фактический результат отличается от ожидаемого, регистрируются как ошибки и передаются команде разработчиков на доработку с последующей перепроверкой после

исправления.

* + Entry Criteria: Test Plan document, Test cases, Test data, Test Environment;
  + Deliverables: Test case execution report, Defect report, RTM.
* **Завершение цикла испытаний** (Test Cycle Closure): окончательная генерация отчетов о тестировании для клиента. Они должны включать затраченное время, процент обнаруженных ошибок и положительных результатов тестирования, общее количество обнаруженных и исправленных ошибок. Что касается отдела тестирования, то это момент для анализа его работы, подведения итогов, анализа его продуктивности и возможности внести предложения по улучшению качества тестирования.
  + Entry Criteria: Test Case Execution report (убедитесь, что нет открытых high severity defects), Defect report;
  + Deliverables: Test Closure report, Test metrics.

**Разница STLC и SDLC**

STLC и SDLC тесно связаны друг с другом, но они одновременно преследуют разные задачи с одной и той же целью, а именно:

* сбор требований в желаемой форме и разработка заявленной функциональности (SDLC);
* анализ требований, помощь клиенту и команде разработчиков и подтверждение качества реализованной функциональности (STLC).

Общая цель - удовлетворение клиента и получение максимально возможного балла на этапах верификации и валидации.