Руслан Раянов

Управление проектом разработки сайта или веб-приложения

От идеи до внедрения

www.web-automation.ru 2015

Введение

Начнем с признания. Мне еще очень много нужно сделать, чтобы вести проекты на действительно высоком уровне. Я не претендую на эксклюзивность моих идей. Просто излагаю свой опыт и мысли, и, надеюсь, что это будет для вас хорошим бумажным (вернее электронным) помощником при ведении проектов.

Сразу хочу сказать, что пишу эту книгу исходя из своих сугубо прагматических интересов. В первую очередь, книга направлена на людей, с которыми я работаю. Это наши разработчики и менеджеры проектов. Именно они делают результат, который мы имеем сейчас. Мне бы очень хотелось иметь общее видение и представление о ведении проектов, и, чтобы каждому члену команды были понятны наши общие принципы и основные методики ведения проекта.

Во-вторых, эта книга для наших клиентов. Т.е. для тех, кто является заказчиком проектов, которые мы ведем. Понимание того, как ведется разработка, значительно упрощает взаимодействие между исполнителем и заказчиком. Обычно недопонимание возникает из-за отсутствия полной картины по проекту, либо неучтенной сложности. В книге мы рассмотрим моменты, связанные с заказчиком и его роли в разработке сайта или веб-приложения.

В-третьих, эта книга для тех, кто просто хочет научиться управлять проектами или узнать что-то новое по этой теме. Причем это не обязательно должен быть программный проект. По своей сути, практически все проекты имеют очень сходную структуру. У всех есть начало, конец, ресурсы, цель, план и т.д. В книге мы рассмотрим все составляющие в контексте проекта на разработку сайта.

После прочтения книги у вас будет представление о том:

- что такое проект
- что такое принципы эффективности
- как вести проект
- как управлять ресурсами на проекте
- как планировать проект
- как решать проблемы
- как вести ежедневную текучку по проектам
- как делать отчеты по проектам
- как организовать взаимодействие с заказчиком и исполнителями

Сейчас есть довольно большое количество различных методик (agile, scrum и т.д.). Мы не будем рассматривать ведение проекта в контексте какой-либо методологии. Я просто

опишу наше видение, как надо вести проекты. Что-то может пересекаться с известными методологиями, что-то - нет. Поэтому при прочтении прошу не сравнивать с другими подходами ведения проекта.

Какие проекты мы будем рассматривать в книге?
В первую очередь, это те проекты, которые мы сами разрабатываем. Это сайты и веб-приложения со своей специфической бизнес-логикой, которые требуют разработки на заказ. Т.е. это, например, CRM, в которой автоматизируются сложные бизнес-процессы, либо интернет-магазин с нетипичной функциональностью, либо агрегатор-биржа по какой-то отрасли. В общем, все то, что не подпадает под стандартные CMS. Подобные проекты отличаются от потокового создания визиток в первую очередь тем, что здесь гораздо больше неизвестных переменных: сложность модулей, программисты и их навыки, изменчивые требования и т.д.

Другой момент, мы не рассматриваем большие проекты со стоимостью выше 1-2 млн. рублей. У меня нет опыта разработки подобных проектов и думаю там есть значительные отличия. В первую очередь в том, что надо учитывать значительно больше интересов разных сторон. Важный момент - при увеличении размера проекта сложность его ведения увеличивается непропорционально. Т.е., чем больше объектов задействовано в проекте, тем сложнее всем этим управлять, и зависимость сложности от количества нелинейная.

Возможно, те, кто далек от мира разработки сайтов и веб-приложений, сочтут, что это чтиво не для них. Специально для вас, вся наша жизнь - это продвижение от одного проекта к другому: поездка на отдых, уборка квартиры, свидание, устройство на новую работу, открытие своего бизнеса, разработка сайта, - все это проекты. Воспринимайте все, что вы делаете для достижения определенных целей как проекты. Принимая такой контекст, примените эту книгу для достижения своих целей. Мы не будем рассматривать в книге постановку целей (цель в наших проектах - сделать сайт). Вы сами можете их определить и прорабатывать их в соответствии с принципами, заложенными в этой книге. Рекомендую вам начинать с одного проекта - иначе вы рискуете забросить это дело в середине пути, а это самый худший исход проекта.

Итак, давайте начинать. В первой главе мы изучим принципы, на основании которых мы ведем проекты.

Глава 1. Принципы ведения проекта

Принципы - это основа. Это фундамент. Всегда отталкивайтесь от принципов, а не от правил, ограничений, прихотей заказчика и т.д. Принципы - это компас. В любую непогоду он выведет вас на нужный путь. Следуя четким принципам, вы снижаете риски принятия неверных решений. Принимая сложное решение, всегда спрашивайте себя, а как это соотносится с вашими принципами? В общем, вы меня поняли - в ситуации неопределенности действуйте согласно принципам, а не каким-то другим соображениям.

Очень важно, чтобы вся команда действовала по одинаковым принципам и ценностям. Если это не так, то начинаются трение и недопонимание.

Конечно, со временем принципы меняются. Что-то оказалось неэффективным, что-то мы узнаем новое и пробуем. Давайте разделим все принципы на основные и дополнительные. Основные имеют приоритет и раскрывают ваши основные ценности. У каждой команды они могут быть разные. Важно, чтобы внутри команды было согласие по ним.

Здесь я приведу наши принципы работы. В вашем случае они могут быть совсем другие. Вы сами должны определить для себя свои принципы.

Основные принципы ведения проектов

Лучше делать быстро, чем медленно.

Мы всегда стараемся оптимизировать по времени наши действия на проекте. Я бы условно разделил людей на “тугодумов” и “скорострелов”. Так вот мы - “самострелы”. Скорость выполнения для нас - приоритет. Конечно, не всегда это получается, и у нас бывают простои и задержки. Но мы работаем над этим, и скорость выполнения работы очень важна для нас.

Достаточно хорошо.

Не стремитесь все сделать идеально. Не разбазаривайте ограниченные ресурсы на то, что дает 0,2% от результата. Сделайте достаточно хорошо и переходите к следующей задаче. Не страдайте перфекционизмом и идеализмом. Если вы будете вечно допиливать проект - он может стать неактуальном при релизе. Сделайте так, чтобы заказчик был доволен, и этого достаточно. Конечно, бывают случаи, когда надо доводить до блеска (WOW эффект или компонент общего пользования), но в общем случае, опасайтесь идеализма.

Контроль.

Чтобы проект двигался, нужен постоянный контроль. Без него все начинает рассыпаться. Контроль, на мой взгляд, самое сложное место в проекте. И ему надо уделить внимание по максимуму. Создайте свою систему отчетности, сделайте ее простой. Понимание того, что происходит сейчас на проекте, дает вам возможность принимать правильные управленческие решения.

Итерации.

Всегда реализуйте проект итеративно. Сначала вы делаете общий набросок картины. Затем прорисовывайте крупные общие линии. Затем детализируете каждый нюанс на картинке. После каждой итерации у вас должно быть вполне работающее приложение с ограниченным функционалом. Другая аналогия - это телескоп. Вы смотрите на небо через телескоп. Сначала оно размытое, ничего не видно, но в целом пятно вы видите. Затем вы настраиваете четкость, и звезды понемногу начинают проясняться. Также мы доводим проект по итерациям до ясного состояния, когда ясно прорисована каждая деталь.

Это основные принципы, и они наиболее приоритетны на проекте.

Дополнительные принципы

Активное участие заказчика.

Если заказчик не принимает участие в проекте, то риски его провалить очень велики. Обязательно привлекайте заказчика к активному участию. Задавайте вопросы по проекту, давайте ему соответствующие задачи, обсуждайте свои решения на проекте. Очень плохая идея забыть заказчика на месяц - другой и затем представить готовое решение. В этом случае готовьтесь к масштабным доработкам и значительному выходу за бюджет.

Все течет, все меняется.

Не думайте, что единожды определив требования в техническом задании, они не будут меняться. Готовьтесь сразу к тому, что требования могут быть очень сильно изменены. Это нормальная ситуация. Просто сделайте перерасчет бюджета и реализуйте то, что нужно клиенту. Помните о параметрах проекта: объем, качество, сроки, бюджет. Вы можете изменять один параметр за счет других. Например, если вам нужно урезать бюджет проекта, вы можете сделать попроще (качество), либо отказаться от каких-либо функций (объем).

Правильная коммуникация.

Хорошая коммуникация очень упрощает взаимодействие на проекте и, следовательно, снижает его стоимость. Во-первых, старайтесь говорить с клиентом на одном языке. Все умеют пользоваться браузером, вот и говорите с ним о кнопочках, списках и т.д. В этом сильно помогает использование средств макетирования. Следующий момент - никаких

истерик и претензий. Это очень напрягает всех участников проекта. Таких людей начинают избегать, от чего они еще больше начинают истерить. Поэтому старайтесь быть максимально корректными по отношению к своим партнерам по проекту. И последний момент - будьте доступными для коммуникации в разумных пределах. Особенно это касается менеджера проекта. Его задача - обеспечить всех необходимыми ресурсами и координировать деятельность. Он должен быть максимально доступен для заказчика и исполнителей. При этом обратный момент - не садитесь на уши. Особенно это касается клиентов, которые садятся на уши менеджеру и менеджеру, который садится на уши исполнителям. Постарайтесь выработать свой ритм взаимодействия по проекту. Все моменты лучше обсуждать за один раз, а не созваниваться по 5 раз за 2 часа. Составьте список всего что нужно обсудить и обсудите это за 1 раз.

Принцип жонглера.

Это скорее относится к менеджерам, которые ведут несколько проектов. Жонглер управляет несколькими шарами одновременно, но в конкретный момент времени в его руках лежит только один шарик. Так же и менеджер в конкретный момент времени должен управлять только одним проектом, не отвлекаясь на другие. Сделайте по максимуму на одном проекте: загрузите всех задачами, делегируйте задачи помощникам, сделайте все, что только можно на проекте, упритесь в какое-то обстоятельство (например, ждем ответа от клиента, или ждем выполнения задачи исполнителем), - и переходите на следующий проект. Работая по такой схеме, вы будете гораздо больше успевать.

Распределенность команды.

Мы изначально исходим из того, что все разработчики будут работать удаленно. Все процессы мы строим исходя из этого принципа. У нас есть офис, но там нет ни одного разработчика. Самое важное для продуктивной работы разработчика - чтобы его поменьше беспокоили. В этом плане удаленная работа как нельзя лучше для этого подходит. Конечно, далеко не каждый сможет работать в таком режиме (кому-то нужен надзиратель за спиной), но такие люди и не задерживаются у нас надолго. Весь контроль строится на основании метрик и задач. Если человек выдает нужный результат, то нет смысла его унижать дополнительным контролем и ограничивать его свободу.

Дублирование.

В веб-разработке есть такой принцип - дублировать код плохо. В ведении проекта все наоборот: дублирование - это хорошо. Вы должны подстраховываться и иметь запасные варианты в случае, если будут проблемы с основными действующими лицами на проекте. Не должно быть такого, что кто-то один на проекте знает то, что другие совсем не в курсе и не представляют как это работает. Обязательно распространяйте знания между разработчиками, чтобы была хотя бы частичная взаимозаменяемость.

Видение бизнеса клиента.

Вы должны видеть дальше, чем просто создать сайт. Сайт нужен для чего-то, а не просто сам по себе. Смотрите глубже. Предугадывайте потребности клиента. Учите программистов и помощника видеть дальше, чем просто разработку очередного модуля.

Прозрачность.

Будьте максимально прозрачными для участников проекта. Когда вы работаете (время для звонков и взаимодействия), какие соглашения используете на проекте, отчетность и т.д. Чем более предсказуемыми вы будете для партнеров, тем лучше.

Проблемы - это нормально.

Не бывает проектов без проблем. Проблемы - это совершенно нормальная вещь для проекта. Вы должны быть готовы решать проблемы быстро и с небольшими затратами. Никогда не избегайте проблем (а вместе с ним и клиента). Решайте их в самом начале, а не тогда, когда проблема достигает гигантских масшабов.

Сервис.

Сервис - это основа всего. Все, что мы делаем - мы делаем для клиента. Именно он платит деньги, а не работодатель. Наша задача - быстро и легко давать клиенту то, что он просит.

Вот, пожалуй, все принципы, на которые мы ориентируемся в своей работе. Пропустите их через себя и подумайте, какие вам подходят, а с какими вы не согласны. Составьте список своих принципов ведения проекта, и в дальнейшем следуйте им.

В следующей главе будем изучать, кто такой менеджер проекта, а также коснемся вопроса, какое участие в проекте принимает заказчик.

Глава 2. Менеджер проекта и заказчик.

Менеджер проекта - это главный человек на проекте. Именно от него во многом зависит успех. Именно менеджер проекта принимает ключевые решения. Именно менеджер объединяет людей с разными навыками и возможностями для достижения определенной бизнес-цели.

Давайте выделим основные функции, которые выполняет менеджер на проекте:

- Взаимодействие с заказчиком. Менеджер выясняет детали проекта, вносит корректировки в проект по пожеланиям заказчика, докладывает клиенту о ситуации на проекте. Менеджер - это вход для заказчика в группу разработки.
- Взаимодействие с исполнителями. Это постановка задач, пояснение их особенностей, проведение планерок и т.д. Важен постоянный контакт между исполнителями и менеджерами. Менеджер должен знать, как идет выполнение задач и общий моральный климат на проекте. Исполнители также должны чувствовать поддержку менеджера и его активное участие (особенно при удаленном способе работы). Менеджер должен давать адекватную обратную связь исполнителю по его работе и стилю выполнения задач (медленно/быстро, количество ошибок, дедлайны, форма отчета, полнота решения и т.д.).
- Бизнес-аналитика. Очень часто менеджер выполняет эту роль. Он изучает предметную область заказчика, прорабатывает отдельные моменты и выступает внутри команды от имени заказчика в решении вопросов по бизнес-логике. В больших проектах для проработки бизнес-логики выделяется отдельный человек, который занимается только этим (бизнес-аналитик).
- Контроль выполнения задач. Контроль может выполняться по 2 параметрам: по качеству и по срокам. Для первого параметра менеджер должен обладать некоторой технической квалификацией. Обычно эту работу делают тестировщики, но в небольших проектах эта задача возлагается на менеджера. Второй параметр - это чисто управленческий момент. Если задача не выполнена в срок, то надо разбираться почему, есть ли проблемы, когда будет сделана и т.д.
- Планирование проекта. Менеджер на основании технического задания составляет план итераций и планы по каждой итерации отдельно. Нет смысла прописывать весь план очень подробно. Все равно будут изменения и ваш план будет перекраиваться не раз. Гораздо лучше составить общий примерный план и четко спланировать очередную итерацию (на 1-2 недели).
- Выделение ресурсов. Обычно у менеджера есть набор ресурсов. Это могут быть денежные средства, время, люди, программные средства и т.д. Задача менеджера - грамотно и экономно ими распорядится для достижения цели проекта за минимальное количество шагов.
- Решение проблем. Если возникают какие-то проблемы на проекте, то обычно именно менеджер их разруливает. У него для этого есть все полномочия и права, а также ответственность. Исполнители также должны стараться самостоятельно

решать проблемы. Они знают принципы и могут на основании них решать большинство таких моментов. Проблемы на проекте могут быть очень разнообразными, но можно выделить несколько типов:
    - человеческий фактор. Обычно это конфликт между людьми. Изучайте переговоры - это позволит эффективно и взаимовыгодно решать любые вопросы.
    - проблемы с ПО. Что-то не так работает, сбои и т.д.
    - нехватка ресурсов. Это самый частый вид проблем. Не хватает времени, выход за рамки бюджета, не хватает квалификации для решения задач и т.д. Все это провоцирует напряженность на проекте, и задача менеджера - так эффективно перераспределить ресурсы, чтобы проект дальше продолжил свое шествие к намеченной цели.
- Метрики и отчеты. Метрики обычно считаются автоматически. Отчеты необходимо писать для заказчика, а также для внутреннего пользования (для руководства). Эти моменты обеспечивают прозрачность проекту и работу на нем.

Теперь давайте рассмотрим качества, которые присущи хорошему менеджеру:

- Во-первых, он должен понимать и принимать основные принципы разработки и ведения проекта. Менеджер должен быть оплотом этих принципов и борцом за них. Если этого нет - то проект будет сложно продвигаться.
- Быстрое переключение между задачами. В силу своей доступности, менеджеру часто приходится прерываться для выполнения запросов от исполнителей и клиентов. Поэтому он должен уметь быстро включаться в другую работу. Это не очень хорошо в плане концентрации, но такова особенность работы менеджера. Ему приходится вести себя как порхающая бабочка, которая перелетает с цветка на цветок легко, плавно и быстро.
- Навыки переговоров. Менеджер – это, в первую очередь, общение. Львиная доля работы менеджера - это общение с клиентами, исполнителями и другими заинтересованными лицами. Это может быть разговор по телефону, живое общение или переписка (email, чат или отчет). В любом случае, если вы хотите быть менеджером, вам надо прокачать навык переговоров и продаж, которые идут рука об руку.
- Системность. Системность позволяет добиваться результатов меньшими усилиями. Вы просто регулярно применяете определенные методики и практики. Это, в первую очередь, требует определенной дисциплины и мотивации. Людям всегда сложно дается внедрение новых паттернов поведения, особенно заставляющих их делать что-то дополнительное. Научитесь смотреть на все через призму систем и процессов, протекающих в них. Все в этом мире - это системы и подсистемы. Научитесь видеть взаимосвязи между звеньями системы. Очень часто для прорыва в системе нужно поменять лишь структуру, расположение звеньев, не трогая содержимое. Продумайте это на своем примере.
- Умение говорить на техническом и бизнес-языке. Вы должны понимать своих клиентов. Обычно они говорят на языке маржи, процентных ставок и т.д. Т.е. они говорят на языке своей предметной области. Разработчики обычно говорят в

терминах базы данных, процедур, классов, URL и т.д. Вы должны понимать их терминологию и уметь соотносить бизнес-язык заказчиков с техническим языком разработчиков. Не ставьте задачу исполнителю на бизнес-языке, т.к. возникает большой риск непонимания и последующего неверного решения. Будьте транслятором между этими 2 мирами, а не испорченным телефоном.

- Общее понимание различных предметных областей. Это значительно помогает вам в решении задач клиентов и общении с ними. Вы должны иметь представление, как работает склад, на чем зарабатывают банки, как работает интернет-магазин в полном цикле, как обрабатываются лиды (продажи) и др. Чем больше вы знаете подобных типовых решений, тем проще вам будет подобрать решение для текущего клиента.
- Исполнительность. Исполнительность означает, что если вы задумали сделать какое-то дело, то вы его просто ДЕЛАЕТЕ, а не бесконечно его затягиваете, ожидая идеального момента, придумываете сложности и т.д. Этот навык крайне важен для менеджера. Просто берем и делаем. Проблемы с исполнительностью часто бывают у интеллигентных или просто умных людей. Они слишком много думают и мало делают. Сделайте упор именно на Делание, а не Продумывание. В конечном счете именно действия будут определять ваш результат, а не мысли (хотя правильные мысли тоже хорошая штука). Мысли без действия - это как семена без земли. Положив их в коробку, вы никогда не получите урожая.
- Навык принятия решения и ответственность. Ответственность для менеджера должна быть привычным делом. Сила менеджера соотносится с тем, насколько масштабные проблемы он может взять на себя. Принимая любое решение, вы берете на себя его последствия. Никогда не будьте страусом - просто пряча от проблемы голову в песок. Ответственно принимайте решения, основанные на правильных принципах и здравом смысле, и в 95% вы будете правы (5% - бывают фатальные случаи, когда сложно предугадать развитие ситуации).

Мы рассмотрели вкратце, кто такой менеджер, чем он занимается и каким он должен быть. Теперь давайте перейдем к заказчику.

Каковы основные обязанности заказчика на проекте? Да, именно обязанности. Ни в коем случае не должно быть такой ситуации, когда заказчик дал деньги и растворился на пару месяцев, считая что проект - это ответственность только исполнителя. В этом случае риски проекта очень высоки. Если у заказчика твердое желание успешно завершить проект, то необходимо с его стороны выполнять следующие обязанности:

- Своевременно и однозначно отвечать на вопросы исполнителя. Здесь очень важно, чтобы заказчик и исполнитель говорили на одном языке. Проще всего - с помощью макетов (т.е. визуальный язык). Также учитывайте скорость ответа. Связанные с этим моментом простои довольно легко устранить на организационном уровне. И последний момент - прорабатывать вопросы лучше не по одному, а целой пачкой - так будет проще и оперативнее. Требуйте от исполнителя не один одиночный вопрос, а сразу пачку и обсуждайте голосом эти

вопросы (письменно - гораздо хуже, т.к. довольно сложно понять - правильно ли вас понял исполнитель).

- Своевременно готовить материалы для сайта - текст, графику и т.д. Если вы хотите, чтобы проект завершился до дедлайна, то в вашей силе этому поспособствовать - оперативно реагируйте на запросы по контенту. Очень часто из-за этого возникают простои. Либо контент меняется по несколько раз, т.к. заказчик недостаточно основательно продумал его с самого начала. Конечно, правки всегда могут быть, но их не должно быть очень много - это тормозит команду разработки (конечно, в идеальном случае это вообще не должно пересекаться - разработка и наполнение контентом, но в реальных условиях это довольно частая практика. Еще момент в пользу такого подхода - заказчик неявно начинает тестировать сайт на реальных данных).
- Своевременно давать обратную связь по проекту и направлять в нужное русло. По моему мнению, это самая важная функция заказчика. Его задача - не дать проекту уйти с нужного курса. Заказчик должен постоянно отслеживать текущий результат проекта и принимать меры, если получает неподходящий результат. Менеджер проекта и команда разработки не всегда могут полностью оценить бизнес-потребность в том или ином элементе. Именно заказчик лучше всех знает, как это будет применяться в его бизнесе. Поэтому только он может внести важные коррективы, которые позволят максимально эффективно использовать сайт в его бизнесе.
- Тестировать с точки зрения бизнес-процессов заказчика. Этот пункт несколько перекликается с предыдущим. Тестировщики от разработчиков обычно просто тестируют функциональность на предмет соответствия решения условиям задачи. Однако они не смогут выявить организационные ошибки и неучтенные бизнес-потребности. Это можно сделать только задействовав ресурсы заказчика. Правильно ли идут информационные потоки, все ли данные учтены, будет ли удобно сотрудникам заказчика работать в системе, каковы показатели эффективности каждой роли и т.д.

Помимо необходимых действий, есть еще пожелания к заказчикам. Это несколько рекомендаций, которые увеличивают шансы проекта на успех.

1. Следить за проектом. Мы об этом уже говорили и еще раз повторим. Нельзя пускать проект на самотек, какая бы хорошая команда разработки у вас ни была.
2. Выполнять пожелания и запросы по проекту от исполнителя. Это ускорит процесс выполнения проекта, а также позволит избежать возможных ошибок.
3. Согласовать взаимодействие с исполнителем. Что это значит? Это значит, что исполнитель должен четко знать ваши приоритеты и цели, связанные с проектом. Это упрощает принятие решений и взаимодействие. Также желательно определить приоритеты по параметрам проекта (бюджет, сроки, объем, качество). При необходимости, это позволит менеджеру проекта правильно перераспределять ресурсы.

4. В случае обнаружения ошибки указывайте точно местоположение ошибки (URL, под каким пользователем, браузер) и что конкретно не так (описание ошибки, ссылка на скрин). В некоторых более сложных случаях надо также указывать, как должно быть в итоге + ручные расчеты, чтобы исполнитель их мог проверить под отладчиком (т.е. вручную пройти программу и посмотреть внутренние вычисления).
5. Сразу определять свои ожидания по срокам и бюджетам. Если какой-то параметр для вас очень критичен и имеет строгий дедлайн - то дайте эту информацию менеджеру проекта. Не должно быть такого, когда вы посередине проекта начинаете вдруг требовать его форсированного завершения. В этом случае это крайне негативно будет влиять на качество.
6. Избегать задержек, связанных с финансами и контентом. Заранее готовьте материалы и откладывайте деньги под бюджет проекта. Это позволит избежать непроизводительных задержек с вашей стороны. Задержки в любом случае будут в проекте, и тут редко кому удается их избежать. Поэтому задача заказчика - хотя бы в относительно простых моментах их не допускать.

В этой главе мы рассмотрели, чем на проекте занимаются менеджер и заказчик. Далее мы переходим к первому этапу ведения проекта - проработке идеи проекта и ее тестированию.

Глава 3. Идея проекта. Как проработать и проверить идею.

Сложно переоценить этот этап. Именно здесь вы можете сэкономить большую кучу денег. Перед началом проекта вы обязаны проверить - действительно ли проект даст тот результат, на который вы нацелились? У вас должна быть некоторая степень уверенности, что выполнив проект, вы получите тот или иной бизнес-результат.

Имея в голове эту цель, мы займемся вопросом проработки идеи проекта, а также рассмотрим вопрос “Как протестировать свою идею на рынке”.

Первое, что вам надо сделать - это понять, а нужен ли кому-то проект кроме вас. Т.е. важно в самом начале определить аудиторию, которая будет потреблять результаты вашего проекта. Для этой цели очень удобно использовать сервис Wordstat - wordstat.yandex.ru. Определите целевые запросы, которые будет набирать ваша целевая аудитория в поисковиках и посмотрите, какая у них частотность в Wordstat - там самым вы поймете, есть ли у вас рынок или нет, и насколько он большой.

Если у вас рынок есть, то поздравляю, двигаемся дальше.

Изучаем конкурентов - кто они, какая у них посещаемость, сколько их в нише, какая у них товарная линейка и т.д. Составьте список своих конкурентов и проанализируйте их. Чем лучше вы будете знать конкурентов, тем больше нюансов вы сможете учесть в своем проекте.

Также изучите по конкурентам такой важный момент как бизнес-модель. На чем именно делают деньги ваши конкуренты? Это не всегда просто выяснить. Не всегда самый доходный товар или услуга на виду. Возможно вы видите только очень привлекательный и дешевый front end продукт, а основные деньги делаются на другом товаре, который предлагается только теплым клиентам. Подумайте об этом: что в вашем случае может быть front end продуктом (максимально простой вход для клиента), а что back end продуктом (на чем делаются основные деньги).

Примечание. Здесь мы не будем подробно останавливаться как проводить исследования целевой аудитории, конкурентов и их продукции. Сейчас наша цель - это управление проектом в целом. Поэтому, мы лишь поверхностно коснемся этих, без сомнения, важных вопросов.

Пока все, что мы сделали – это, в основном, аналитика, т.е. без реального взаимодействия с рынком. А именно только оно может достоверно показать, имеет ли ваш проект шанс на успех. Для этого вам необходимо организовать тест вашей идеи. Наверно самый лучший вариант - это сформулировать свое очень заманчивое для

клиента предложение, сформировать лендинг (т.е. простой одностраничный сайт) и запустить на него рекламу.

В результате вы получите какое-то количество кликов (пусть будет к примеру 200 или 500). Из них к вам обратилось 5 человек. В итоге вы получаете конверсию - 5 / 200 * 100% = 2,5%. Если ваша конверсия составит более 1%, то, значит, ваша идея стоит того, чтобы попробовать. Если нет – то, возможно, ваше предложение не настолько заманчивое, и следует над ним поработать.

Конечно, не во всех областях так просто выделить предложение и сделать лендинг под него. Например если вы задумали сделать агрегатор-биржу или социальную сеть, то здесь так просто не получится проверить с помощью лендинга. Альтернативой может служить опрос целевой аудитории в социальной сети - будут ли они использовать такой сервис и т.д. Вообще общение с целевой аудиторией очень помогает - оно раскрывает такие моменты, о которых вы даже и не могли подумать.

При создании предложения делайте его чуть лучше, чем в среднем по рынку (имеется в виду по цене). В идеале, это должно быть предложение, от которого просто глупо отказываться.

Этап тестирования очень часто пропускается (мы и сами его пропускали в некоторых своих проектах, и обжигались на этом). Сделайте хотя бы простой опрос из 100 человек (желательно как можно ближе к целевой аудитории проекта). Это не займет много времени, но зато может вам сэкономить очень приличную сумму денег и время.

С тестами закончили, теперь давайте сформулируем цель проекта. Что мы в итоге хотим от него получить. При этом цель, конечно, лучше ставить в вещественных параметрах и в цифрах. Напишите, какой конкретно результат у вас будет по завершению проекта. Это могут быть:

- деньги
- трафик
- функционал
- аудитория
- какие-либо единицы в проекте (видео, тексты и т.д.).

Определите точную метрику и дату, к которой вы хотите ее получить. Ставьте значение на границе своих возможностей. Слишком много - это будет демотивировать и подрывать весь проект, мало - тоже плохо (будет все затягиваться и интерес к проекту быстро пропадет).

Следующий момент - оцените, от чего вам придется отказаться. В мире очень много возможностей и успеха добивается тот, кто умеет отказаться от хорошего в пользу великого. Ваши ресурсы не безграничны, поэтому определитесь сразу, готовы ли вы пожертвовать всем остальным ради этого проекта.

Напишите список альтернативных проектов и осознанно откажитесь от каждого в пользу текущего проекта. Конечно, если у вас большая команда, то вы можете делегировать мелкие проекты. Здесь мы говорим о проектах, которые могут занимать практически все ваши текущие ресурсы (время, люди, деньги, энергия).

Также стоит рассмотреть вопрос о запуске сначала какого-то более мелкого проекта, пилотного проекта. Мне нравится идея прототипных решений - вы не делаете сразу большое полноценное решение - сначала вы делаете пилотный быстрый проект для получения обратной связи. Т.е. этот проект гораздо легче основного проекта и на нем вы понимаете, будет ли это работать или нет. В каком-то смысле, проведение теста и является пилотным проектом, но степень “пилотности” может быть разной.

К примеру, при разработке доски объявлений, вы можете сделать сначала самую простую доску (возможно на бесплатном движке, либо сделать свой движок с ограниченным количеством функций). Поработать с ним, понять, что в итоге вам нужно и сделать уже свой полноценный движок со всеми наворотами.

Как мы уже упоминали, сразу соотносите свои ресурсы с проектом. Сможете ли вы его потянуть в том виде, который вы определили. Нет ли разрыва между возможностями и потребностями? Самое важное - будьте честными с собой. На берегу решите все моменты с ресурсами для проекта, чтобы потом не закрывать проект на середине пути. Два самых важных параметра - это деньги и время. Хватит ли вам денежных средств для поднятия проекта до того момента, пока он не станет окупать сам себя? Будет ли у вас время на полноценное участие в проекте? Без проработки этих 2 параметров риски проекта существенно увеличиваются.

На этой немного суровой ноте мы заканчиваем главу про проработку идеи проекта и переходим к предпроектной деятельности. На этом этапе вы уже твердо решили, что проект будем делать и теперь мы переходим к тактическим вопросам подготовки проекта к запуску.

Глава 4. Предпроектная деятельность

Что мы должны сделать до начала проекта?
Во-первых, это общая оценка проекта.
Во-вторых, определить основные параметры проекта.
И, в-третьих, формализовать все моменты по проекту в виде концепции проекта (это письменный документ, а не что-то, что у вас в голове).

Начнем с оценки. Самый важный момент, надо различать оценку, обязательство и фактический показатель. Оценка - это прогноз по метрике. Обязательство - это обещание сделать в рамках определенных метрик. И фактическая метрика - это то, что по факту получилось.

Очень часто просят назвать очень точную оценку по краткому содержанию проекта. Тем самым заказчик уже закладывает гигантские риски в проект. Невозможно дать точную оценку на начальном этапе. Она в принципе не может быть точной. Есть такое понятие как “Конус неопределенности”. На начальном этапе оценка очень широкая, затем по мере движения проекта вилка оценки уменьшается (т.к. уменьшается неопределенность в проекте). Таким образом, не пытайтесь очень точно оценить проект на самом раннем этапе. Лучше сосредоточьтесь на следующем - сделайте так, чтобы в 90% случаев фактическая метрика вошла в ваша начальный диапазон оценки.

Какие бывают виды оценки?

1. Быстрая оценка. Делается по одному взгляду на проект. Мы ее очень часто используем для первичной оценки проекта. Как она делается:
    a. Делаем градацию по проектам. Например, 3 градации, у каждой своя вилка и критерии попадания под нее.
    b. При оценке проекта соотносим ее с одной из градаций.
    c. Получаем оценку (оценка этой градации, которая сделана заранее).

Плюс этого подхода в скорости и малых затратах. Минус - это очень приблизительный способ. Но для первого ознакомления с проектом этого бывает вполне достаточно. Еще момент - этот способ практически не требует технической квалификации от оценщика. Его задача - просто корректно соотнести проект с нужной градацией.

2. Модульная оценка. Вы разбиваете конкретный проект на крупные блоки, модули и отдельно оцениваете каждый модуль. Обычно это делает уже технический специалист, либо менеджер. Здесь, как минимум, нужно детально ознакомиться с концепцией или ТЗ проекта. Модулями могут быть крупные функциональные блоки, которые являются более-менее типовыми. Для таких блоков можно сделать типовые оценки. Например, сделать форум - это N часов (или X рублей). Это немного упрощает оценку.

Эта оценка существенно точнее предыдущего способа и, если у вас одиночный проект, то именно ее лучше всего использовать на начальном этапе. Вы также можете попросить дать оценку у других людей, не всего проекта, а отдельного модуля. Это также позволит вам получить более объективное представление о каждом модуле.

3. Детальная оценка. Вы разбиваете весь проект на мелкие кусочки - функции и компоненты и детально оцениваете каждый из них. Будет очень хорошо если это будут делать разработчики, и их будет несколько. Это сделает оценку существенно точнее. Параллельно вы можете проработать проект - выявить недостающие нюансы.

Для этой оценки очень важно, чтобы ТЗ было уже написано и оно было достаточно детальным. Если этого не будет, то разработчики завалят вас кучей вопросов по деталям проекта.

Оценку лучше вести в часах и определить нормированную стоимость этого часа. У каждого компонента (или страницы) будет 2 оценки - минимальная и максимальная. Чем точнее написано задание, тем ближе они будут друг к другу.

В итоге, суммируя эти оценки, вы получите довольно точное представление о бюджете и сроках проекта.

Очень важно при этой оценке учесть все дополнительные работы (о которых могут забыть разработчики), а именно:

- подготовка проекта
- совещания
- тестирование
- первичная поисковая оптимизация
- вынесение настроек в админку
- дизайн для админки
- уведомления
- типовой функционал, не учтенный в ТЗ (смена пароля, восстановление пароля, выход и т.д.)
- создание системы бекапов
- настройка мониторинга доступности
- регистрация в поисковых системах
- обработка новых запросов клиента
- ведение проекта
- составление отчетности по проекту.

Еще важный момент, учитывайте, что в любом случае будут неучтенные доработки. Их размер зависит, в основном, от клиента - именно он в конечном счете определяет какие функции надо изменить. Здесь важно помнить, что чем позже вы внедряете изменения,

тем дороже они будут стоить. Поэтому желательно сразу правильно определять состав функциональности в проекте.

При оценке важно учитывать такой показатель как уровень абстракции. Выберите приемлемый для себя уровень абстракции и именно на нем проводите оценку. Если у вас нет ресурсов для уточненной оценки, оценивайте на уровне модулей. В любом случае начинайте двигаться о верхнего уровня к нижнему, т.е. сначала в целом поймите на уровне подсистем что это будет, а затем разбирайте каждую подсистему на модули, а модули на компоненты. На любом из уровней абстракции вы можете остановиться, посчитав его достаточным для целей оценки.

Еще момент, выберите уровень технических решений и качества. Здесь имеет смысл брать ориентир на другие сайты, т.е. этот блок сделать примерно как на site.ru. Конечно, это очень субъективно, но позволит команде разработке сделать свое представление о выбранном уровне качества.

После того как мы сделали оценку по срокам и бюджету, хорошо бы собрать всю информацию воедино в так называемую концепцию проекта.

Что в нее входит:

1. Основные параметры проекта:
    a. Цель проекта и результат. Что в итоге вы должны получить? Зачем вам нужен этот проект? Входными данными для какого следующего проекта будет результат текущего проекта? Четко определите критерии завершения проекта.
    b. Сроки. Сколько в целом месяцев/недель займет проект?
    c. Бюджет. Сколько вы запланировали заложить средств на реализацию проекта?
    d. Объем. Что нужно реализовать в рамках проекта? Что мы оставим за бортом? Определите границы проекта.
    e. Команда. Кто будет реализовывать этот проект? Кто за что будет отвечать на проекте?
    f. Ресурсы. Какие дополнительные ресурсы вам потребуются для реализации проекта? Это могут быть знания, технические средства, место для работы и т.д.
    g. Приоритеты внутри проекта. Что в проекте важнее всего из параметров? Сроки, бюджет, качество, объем? Возьмем, например, самолет. В нем важны, в первую очередь, качество и объем. Объем нельзя урезать, да и качество должно быть максимальным. Теперь возьмем социальную сеть. В ней могут быть важны бюджет (т.к. это деньги инвестора) и качество. Объем здесь не так важен – его можно развивать по ходу запуска проекта. Понимание приоритетов даст вам в будущем возможность варьировать параметры проекта. Например, вы не успеваете сделать в срок, а сроки являются приоритетом. Что делать? Либо уменьшайте объем, либо

увеличивайте бюджет, либо внедряйте более простые и быстрые решения. При этом, разумеется, вы исходите из прописанных приоритетов проекта.

   h. Риски. Каковы риски проекта? Выпишите список рисков, связанных с проектом. Вы можете их категоризовать. Например, организационные, технические, внешние, по параметрам проекта. Далее для каждого риска вы проставляете степень критичности и степень вероятности (от одного до трех). Далее в порядке приоритетности прорабатываете каждый риск – пишите меры, которые уменьшают вероятность и критичность риска. Тем самым вы получите карту рисков проекта и план по их обработке.
   i. Ориентировочный план. Вы примерно должны определить, как вы достигнете цели проекта. Здесь не нужна детализация – просто определите этапы достижения цели. Это позволит команде разработки в целом понимать в какую сторону надо двигаться. Для каждого этапа наметьте ориентировочные сроки. Здесь следует понимать, что это не обязательство или план к исполнению. Это просто желательный план развития проекта, который в процессе может изменяться.

2. Краткое описание. Очень желательно иметь лаконичное описание, которое позволит сразу схватить суть проекта и показать его особенность. Это нужно для взаимодействия с внешним миром (дизайнеры, программисты, партнеры и т.д.). Хорошее описание сэкономит вам кучу времени при взаимодействии с возможными исполнителями на проект – вам не будут писать те, кто заведомо не подходит.
3. Критерии отбора исполнителей. Лучше их заранее проработать, чтобы опять же сэкономить время. Что могут включать эти критерии:
   a. Общие требования, например, технологии, географическое расположение, образование, студия/фрилансер, под ключ/специализация. Указывайте требования так, чтобы сразу по максимуму отсеять кандидатов. Подумайте с какими кандидатами вы точно не хотели бы работать. При этом не указывайте аморфные «мастер своего дела» и прочее – это совершенно ни о чем не говорит кандидатам.
   b. Мини тест на адекватность. Если кандидат хочет с вами работать, то пусть сделает простую последовательность действий, которую вы для него подготовили. Например, отправить на почту письмо с темой «XXX» с вложением КП в формате PDF. Если исполнитель вам пишет не в вашем формате, то скорее всего он будет так же себя вести и на проекте. Исполнительность – это первое, что нужно проверять для кандидатов.

Итак, мы написали концепцию. Исходя из этой концепции, мы будем инициировать проект. В следующей главе мы изучим моменты, связанные непосредственно с началом проекта.

Глава 5. Начинаем проект

Для начала давайте определимся, из чего мы исходим в начале проекта:

- У нас есть концепция со всем параметрами.
- У нас есть техническое задание. Мы не рассматривали его написание в этой книге. По сути, это техническое описание, что надо сделать на проекте. Обычно оно состоит из макетов и системных требований.
- Состав команды разработки определен. Опять же мы лишь частично затронули момент с поиском исполнителей на проект. Предполагаем, что команда разработки у нас уже есть.
- Вы – менеджер этого проекта. Т.е. у вас есть все полномочия по принятию решений на этом проекте, а также выделены ресурсы на проект, которыми вы можете распоряжаться.

Инициацию проекта можно условно разделить на 2 пункта:

- Подготовка и выделение ресурсов
- Планирование проекта

В плане подготовки инструментов и выделения ресурсов вам необходимо рассмотреть следующие моменты:

- Вы должны обеспечить подготовку тестового домена, сервера, где будет работать тестовое приложение, базы данных, инструментов разработки (SVN, среды разработки и др.). Одним словом, вы должны подготовить тестовую среду.
- Дайте права и полномочия членам проекта. Это доступы к системе планирования, доступ к тестовому приложению (база, код, иногда сервер). Дайте контакты участников проекта.
- Необходимо провести обучение участников проекта предметной области. Особенно это важно для таких проектов, где предметная область очень далека от простого разработчика, например, специфические процессы продаж, сложное производство и т.д. Также на этом этапе важно выработать общую терминологию и пояснить ее участникам проекта.
- Определение границ проекта для участников. Разработчики должны понимать область проекта, иначе есть риски, что проект пойдет не в том направлении. Обсудите параметры проекта с разработчиками, чтобы они понимали требуемый уровень качества и приоритеты клиента.
- Наконец, проведите общее совещание по проекту. Это обобщает все вышесказанное + дает возможность познакомиться членам проекта.

Теперь переходим к плану проекта. Как надо планировать? Систем планирования очень много, я просто расскажу свое видение этого вопроса.

Как мы говорили выше, изначально лучше планировать общими этапами с очень примерными сроками. Очередной крупный этап детализируется на итерации (1-2 недели).

Скрупулезно планировать вы должны только текущую итерацию. Определите совместно с заказчиком, что вы должны сделать на этой итерации. По сути это будут приоритеты заказчика на итерацию.

Сформулируйте, исходя из приоритетов, задачи для исполнителей. При этом следует понимать, что сам приоритет – это по факту не задача. Задача требует указания контекста и особенностей. Также в задаче вы можете дать рекомендации по решению и описать особенности, если это требуется. Для каждой задачи необходимо определить следующие параметры:

- Исполнитель
- Дедлайн
- Оценочное время
- Чек-лист. Задание проще всего формулировать в виде чек листа. Так будет проще закрывать задачу исполнителю, а вам – проверять. Чек-лист – это антипод монолитного куска текста в котором размышления автора перемешиваются с заданием и вопросами к исполнителю. Структурируйте свои задачи в форме простых и понятных действий, которые надо сделать исполнителю для закрытия задачи.

Таким образом, у вас на итерацию будет пакет задач для исполнителей, который они должны сделать к определенному сроку. Если итерация составляет 1 неделю, то лучше, чтобы задачи были сделаны к четвергу, чтобы было время внести корректировки в рамках этой итерации. По окончании итерации необходимо написать отчет, в котором мы отражаем насколько хорошо мы закрыли приоритеты клиента на эту итерацию.

Еще очень важный момент по планированию – планируйте так, чтобы в конце итерации всегда получать рабочий результат. Что это значит? Представьте, что вы рисуете картину, например, портрет. Сначала вы делаете общий набросок, на котором определяете контуры лица. Затем вы наносите на холст крупные детали (глаза, рот, нос). После этого вы переходите к более мелким деталям (уголочки глаз, губ, морщинки). И, наконец, вы переходите к теням и т.д. На каждом этапе вы получаете полный рисунок и постепенно его детализируете.

Альтернативой такому подходу может быть сначала полная прорисовка глаз, затем нос и т.д. Но это не будет гармоничным рисунком.

Также и в вашем случае. Вы делаете скелет приложения, создаете пустые страницы, затем делаете общие компоненты без стилизации и т.д. В итоге на каждом этапе у вас будет рабочее приложение.

Теперь давайте разбираться с вопросом «Как разбивать на этапы и итерации?».

Условно веб-приложение мы можем разбить на подсистемы/модули. Каждая подсистема может быть разбита на компоненты/страницы.

Первое, что вам нужно сделать, это определить роли в системе и подсистемы, которые в ней будут. Роли – это менеджер, администратор, продавец, оператор и т.д. Подсистемы – это обработка заявки, подготовка КП, форум, система сообщений.

Совместно с заказчиком вы определяете приоритеты по подсистемам и личным кабинетам ролей. В результате вы создаете примерный план этапов проекта с учетом приоритетов бизнес-целей заказчика. При этом этап – это не обязательно только реализация одной подсистемы. Он может включать реализацию разных подсистем или их частей.

Далее вы берете текущий этап и разбиваете его на страницы. В большинстве случаев каждая страница – это одна задача. Конечно, есть очень тяжелые страницы, которые разбиваются на множество задач, но вы можете сами определять порядок композиции задач. В итоге вы получаете список задач на итерацию исходя из этапа и приоритетов заказчика внутри этапа.

Незаметно мы подобрались к концу главы. Сейчас у вас есть примерный план по этапам проекта и детализированный план на текущую итерацию в форме конкретных задач для исполнителей с понятными и простыми чек-листами. Следующий момент – самый сложный в проекте. Именно он определяет успех проекта и скорость его движения к долгожданному результату. Об этом мы будем говорить в следующей главе.

Глава 6. Текучка по проекту

Итерация (неделя) началась. Наша задача, как менеджера проекта, это качественное выполнение всех задач итерации в срок.

Если говорить в общем, то на входе итерации мы имеем следующее:

- Задачи исполнителей
- Приоритеты клиента

На выходе мы должны получить:

- Сделанные задачи
- Отчет для клиента
- Обратная связь клиента по итерации

С чего начинается любая задача – передача задачи исполнителю. В начале недели обсудите задачи с исполнителями голосом, получите от них обратную связь по задачам. Есть ли у них вопросы? Предложения? Проблемы с реализацией? Верно ли оценено время? В итоге вы должны разрешить все вопросы и сделать так, чтобы исполнитель поставил статус задач в «Принято в работу».

Через 1-2 дня вы должны проверить, как идут дела, желательно посмотреть промежуточный результат. Нужна ли помощь по задаче? Если задач несколько, то исполнитель должен понимать приоритеты по задачам.

Если исполнитель выполнил раньше времени все задачи (а такое очень редко бывает), то либо давайте что-то со следующей итерации, либо передавайте ему задачи от других исполнителей на проекте.

В идеале надо каждый день следить за ходом задач на проекте и давать обратную связь клиенту по ходу работы и исполнителям по сделанному функционалу. Это позволит вам держать руку на пульсе проекта.

Важный момент – сразу договоритесь с заказчиком по минимуму влезать в середине итерации. Только в критических ситуациях. Заказчик в пятницу получает отчет по итерации и начинает тестировать приложение. В этот же день можно начинать формулировать приоритеты и задачи на следующую итерацию. Также желательно иметь от заказчика обратную связь в виде оценки, которая характеризует степень его удовлетворенности. Если вы получили низкую оценку более 1 раза – это знак, что надо что-то менять на проекте.

Как проверять задачи?

Первое, что надо сделать – это чтобы при выполнении исполнители неявно сами тестировали свой код. Мы это делаем следующим образом: при выполнении задачи исполнитель должен указать скрин по выполненной задаче на сервере.

При проверке задачи учитывайте следующие моменты:

- Тестируйте на реальных данных, а не на неправдоподобных (вроде строки из 100 символов без пробелов).
- В идеале созванивайтесь с исполнителем и совместно посмотрите задачи. Это ускорит процесс понимания и доработок (некоторые из них можно будет сделать по ходу просмотра задач). Дополнительный эффект – разработчику будет стыдно при большом количестве ошибок (все-таки личное общение), и это будет его мотивировать тщательнее проверять свои решения.
- При выявлении ошибки всегда указывайте конкретику: URL, скрин, в чем именно ошибка, как должно быть.
- Ведите метрики по ошибкам и выходам за дедлайн. По сути, 2 главные метрики программиста – это количество ошибок и выход за дедлайн. Используйте такую систему ведения плана, которая позволит вам отслеживать эти параметры. Мотивируйте исполнителей делать минимум ошибок и в пределах дедлайна.
- Организационно можно разбить проверку на 2 составляющие: качественную и организационную. Качество проверяет тестировщик. Организационная составляющая – это общий контроль задач проекта и сроков. Фактически эти задачи можно дать разным людям (тестировщик и менеджер проекта). Однако в большинстве случаев объединение этих ролей обосновано по соображениям рамок бюджета.

В рамках итерации вы должны дать исполнителям и заказчику обратную связь. Для исполнителей – это тестирование и рекомендации по выполнению задач. Для заказчика – это недельный отчет. Отчет по итерации может готовить либо менеджер проекта, либо старший разработчик.

Что должен включать такой отчет?

В первую очередь, это насколько команде удалось закрыть приоритеты клиента.
Во-вторых, это примерный план на следующую итерацию, т.е. что мы планируем делать дальше.
И последнее, это проблемы, особенности, предложения, возникшие в ходе итерации.
Ваша задача – это дать как можно более полную информацию по состоянию проекта для заказчика. Также стимулируйте заказчика тестировать разработанные модули, указывайте в отчете ссылки на разработанный функционал, просите обратной связи по нему.

Проблемы и риски

Не бывает ни одного проекта без нештатных ситуаций и проблем. Они обязательно будут и ваша задача – научиться решать их быстро и эффективно.
Важно выработать общий набор принципов, исходя из которых вы будете принимать решение. О них мы говорили ранее.

Имейте правильные договоренности с клиентом. Чем прозрачнее ваше взаимодействие с ним, тем больше доверия и тем больше возможностей для маневра.

При решении сложных вопросов всегда действуйте в формате Win-Win. Учитывайте интересы всех сторон. Если вы будете постоянно прожимать одну из сторон, в итоге это может очень негативно сказаться на проекте. Делайте диалог и взаимодействие максимально прозрачными и открытыми.

При возникновении проблемы задайте себе очень важный вопрос: «Это проблема систематическая?», т.е. имеет ли она повторяющуюся природу? Если да, то ищите глобальные причины проблемы и рубите корень, а не срывайте листья. Старайтесь подобные конфликты и проблемы вскрывать на ранней стадии, избегая проблем в текущий момент. Позже эти проблемы превращаются в драконов и монстров, которые потребуют сверхусилий для их разрешения.

Теперь поговорим о рисках. Как мы раньше говорили у риска есть критичность и вероятность. Постоянно пересматривайте риски проекта и реализуйте мероприятия по их уменьшению. Если вы не можете уменьшить риск, то вам придется его принять, т.е. согласиться с его возможными последствиями – это тоже своего рода обработка риска – осознанное принятие риска.

Давайте кратко рассмотрим основные типовые риски и меры для их нейтрализации.

- Нехватка бюджета. Меры: договоренности с клиентом, более точная оценка, уменьшение объема, качества и других параметров проекта.
- Уход ключевого исполнителя. Меры: обучение и документация, работа исполнителей в паре, мотивация исполнителей, договоренности с исполнителем (неустойка).
- Срыв сроков. Меры: уменьшение объема, подключение дополнительных исполнителей, упрощение проекта по качеству.
- Большой процент брака. Мотивация исполнителей, замена исполнителей, изменение работы тестировщиков.
- Изменение требований по проекту. Меры: гибкая архитектура, договоренности с клиентом, увеличение бюджета и сроков.
- Проект стал неактуальным. Меры: договоренности с клиентом, предоплата.
- Кража исходного кода. Меры: подписание соглашений, ограничение доступа к коду, психологические уловки.

- Разрыв отношений клиента и подрядчика. Меры: соглашения с клиентом, документация, доступ заказчика к исходному коду.
- Резкое пропадание подрядчика. Меры: документация, доступ заказчика к исходному коду, договор, постоплата по этапам.

Конечно, рисков может быть гораздо больше. Ваша задача – выявить их и проработать. Проработка рисков – довольно редко встречающаяся практика в небольших проектах. Но хотя бы поверхностное изучение своих рисков позволит вам избежать проблем в будущем, либо минимизировать ущерб от них.

Показатели и KPI

KPI позволяет вам управлять процессами на уровне отдельных показателей. Вы определяете главные метрики проекта и ориентируетесь на них. Если у вас есть просто план действий, то в целом для вас самая важная метрика, процент закрытых задач на итерации. Это самая основная метрика. Но могут быть еще и много дополнительных, например:

- Расход бюджета
- Отклонение оценки задач от фактических затрат времени
- Процент ошибок по исполнителям
- Процент выхода за дедлайны по исполнителям (или по итерациям)

Просто подумайте, что вы хотите улучшить в своем управлении проектов. На основании этого определите метрики, которые характеризуют это проблему. Просто начните отслеживать этот параметр. Затем подумайте, что вы можете внедрить в свою систему, чтобы эти показатели изменить. Т.е. выдвинете гипотезы, которые улучшат показатели.

Очень часто это будет касаться повседневных действий и поведения на проекте. Сделайте доступными эти метрики тем лицам, от которых они зависят. Превратите это в игру, где метрики будут играть роль табло со счетом. Завяжите это табло на мотивацию исполнителей, и показатели понемногу начнут улучшаться.

На проектах у вас должен быть стандартный список показателей, который характеризует качество проекта. На конкретный промежуток времени вы можете добавлять дополнительные параметры для получения требуемого уровня по определенным критериям качества (например, время реагирования исполнителей на задачи).

Организация взаимодействия по проекту

От эффективной коммуникации на проекте зависит очень многое. Если здесь будет минимум трений – то проект можно будет завершить гораздо быстрее, нежели в случае, когда любое общение по проекту приносит только раздражение и разочарование.

Здесь мы рассмотрим 3 взаимодействия:

- Клиент-менеджер
- Менеджер-исполнитель
- Клиент-исполнитель

Клиент-менеджер.
Самое важное – нельзя построить всю коммуникацию только на переписке. У нас было пару проектов, построенных по такому принципу взаимодействия. Проекты были успешно внедрены, но это значительно усложняет взаимопонимание и определение пути развития проекта. Отсутствие личного контакта уменьшает связанность и моральную ответственность.

Другая крайность – это долгие коллы и совещания. Все коллы надо делать не более 40 минут. После 40 минут восприятие информации падает, да и большинство вопросов можно обсудить за куда меньший срок.

Бывает так, что клиент очень ревниво относится к вниманию со стороны менеджера и постоянно тратит его время на непроизводительное общение. В этом случае он должен понимать, что он действует во вред проекту и по сути сжигает в никуда бюджет. Найдите правильные точки соприкосновения с клиентом и определите договоренности по этому вопросу.

Следующий важный момент – это приоритеты и цели заказчика. Задача менеджера – это просто реализовывать цели заказчика в рамках проекта. Очень сложно их реализовать, когда они либо не указаны, либо слишком размыты. Всегда спрашивайте клиента о его целях и приоритетах. В будущем это может даже защитить вас – вы всегда сможете сказать, что действовали исходя из целей клиента (если, конечно, вы действительно действовали исходя из его приоритетов). Трудные решения по проекту всегда принимайте на основе следующих моментов:

- своих принципов ведения проекта
- целей и приоритетов клиента
- параметров проекта (бюджет, срок, качество, объем)
- целей и интересов своей компании (т.е. подрядчика)

Следующий вопрос: «Как реагировать на хотелки клиента?».
Они в любом случае будут и надо быть к этому готовым. Любое изменение в проекте – это всегда риск выйти за его рамки. Очень важно, чтобы это понимали все участники проекта, особенно заказчик. Особенно критичен этот вопрос по отношению к срокам и бюджету проекта. Клиент, не думая, добавляет несущественный функционал и проект распухает как на дрожжах. Ваша задача, как менеджера, – принять в обработку пожелания клиента и уведомить его об изменениях проекта, либо проговорить договоренности по поводу таких доработок и изменений.

Также, помните о целях проекта и приоритетах. Ваша задача – это довести проект до логического результата и ваша обязанность – это убирать всевозможные препятствия на

пути к этому результату. Как ни парадоксально, но именно клиент может стать одной из главных проблем на пути достижения его же целей. Вы должны умело лавировать между сиюминутными запросами клиента, целями и приоритетами всего проекта.

Обсудите с клиентом вопрос доступности, и договоренности по времени и способу связи. Лучше прояснить этот вопрос заранее, т.к. иначе ожидания клиента по вашей доступности могут отличаться от вашего стиля ведения проекта. По возможности, вы должны быть доступны в удобное для клиента время. Согласуйте точные моменты, когда вы будете связываться с клиентом по проекту. Обсудите с клиентом, куда ему надо складывать свои запросы, когда вы в оффлайн, т.е. недоступны.

Проработав эти вопросы совместно с заказчиком, менеджер обеспечивает проекту надежную основу для взаимодействия с заказчиком в течение всего проекта. Также не забудьте проработать дополнительные моменты, которые могут быть специфичны для вашей ситуации. Например, если клиент находится в вашем городе, то хорошо устраивать личные встречи для демонстрации результата. Если клиент – иностранец, то хорошо бы заиметь переводчика, который будет работать в паре с менеджером.

Теперь мы переходим к следующему типу взаимодействия – Менеджер-исполнитель, т.е. внутреннему взаимодействию по проекту.

Менеджер-исполнитель

От качества этого взаимодействия будет зависеть напрямую процент выполненных задач на итерации, а также количество ошибок. Вы должны построить максимально доверительные и открытые отношения с исполнителями на проекте.

Первое. Больше общайтесь голосом по скайпу. Меньше пишите на почту. Проверяйте задачи совместно с исполнителями. Поясняйте, как делать задачи, отвечайте голосом на их вопросы.

Очень важный момент. Следуйте правилу «обсуждение голосом по скайпу/телефону, задачи только через систему». Не путайте исполнителя смешиванием задач и обсуждений. От этого кипит голова. Ставьте четко задачи без намеков на отступления и обсуждение открытых вопросов.

Как мы уже говорили, при проверке задач старайтесь давать исполнителю больше конкретики по ошибке и проверять совместно голосом с демонстрацией экрана по скайпу. Это значительно упрощает понимание ошибок и нюансов задачи.

Научитесь говорить на языке специалистов. Изучите все термины вашей области. Задавайте вопросы разработчикам. Постоянно осваивайте новые сегменты в веб-разработке и предметной области вашего проекта. Конечно, техническим специалистам-менеджерам в некотором смысле проще, но это не такое большое преимущество и оно

довольно легко достигается. Если вы менеджер без технической подготовки, то не расстраивайтесь. У вас есть очень важное преимущество – вы не влезаете в детали и не навязываете разработчикам свои решения, т.е. не делаете за них работу. Для менеджера это очень важно, особенно если он ведет несколько проектов. Это очень распространенная болезнь менеджеров - бывших программистов. Они до сих пор решают низкоуровневые вопросы, что конечно же негативно сказывается на их прямых обязанностях (обеспечение ресурсами, контроль, взаимодействие с заказчиком, планирование). Если вы технарь, то помните об этом нюансе и работайте над собой.

Следующий момент. Это дисциплина. Именно от менеджера зависит дисциплина на проекте. Не допускайте халатного отношения к условиям задачи, к дедлайнам и общему уровню исполнения. Наказывайте за это: плохими оценками, штрафами и др. В крайнем случае, снимайте исполнителя с проекта. Это лучше чем, тянуть резину и тормозить проект. Для такой возможности у вас, конечно, должен быть некоторый запас в исполнителях. Имея его, вы можете пробовать различные схемы, как тренер в хорошем заграничном футбольном клубе. Ведь по сути, менеджер проекта – это и есть аналог тренера команды. Он отвечает за уровень исполнителей, за их мотивацию, за дисциплину и, в конечном счете, - за результат. При этом игру выигрывает не тренер, а команда. Рассматривайте команду как продолжение своих идей. Именно команда делает результат, а задача менеджера – сделать для этого идеальные условия.

Клиент-исполнитель

Желательно свести к минимуму прямое общение между исполнителем и клиентом. Иначе очень велик риск, что клиент сядет на уши исполнителю и будет постоянно проталкивать новые требования вне ТЗ. Договоритесь с клиентом, что он будет ставить задачи напрямую исполнителям только в случае критической необходимости, когда срочно надо поправить ошибки на проекте, при этом, уведомив вас, как менеджера проекта.

Еще один негативный момент частого взаимодействия «клиент-исполнитель» – это постоянные прерывания в работе. Это тоже нехороший момент с точки зрения выполнения задач на итерации. Да, клиент будет доволен, что ему дают напрямую общаться со всеми на проекте, но важнее, чтобы проект двигался к логическому завершению, а не просто удовлетворялись сиюминутные пожелания клиента.

Если клиент более-менее технически подкован, то он может получить точную техническую информацию от исполнителя на проекте. Такое взаимодействие имеет место, но только по согласованию с менеджером. В идеале менеджер должен быть в курсе о любых движениях на проекте. Ему не обязательно везде участвовать, но проинформирован он быть должен.

Очень сложный момент с отчетами от исполнителя для клиента. Дело в том, что исполнитель обычно пишет на своем техническом языке и на своем уровне абстракции. Весьма вероятно, что клиент просто не поймет исполнителя. Вы должны учитывать эту

особенность и давать писать отчеты исполнителю, если он в состоянии написать отчет на языке клиента – простыми словами, поменьше букв, меньше технических деталей, общее положение по задачам и проекту в целом.

Последний момент, который мы обсудим в этой главе – это совмещение в менеджере функций менеджера, аналитика и тестера.

Функции менеджера:

- Планировать
- Контролировать сроки и исполнение задач
- Решать вопросы и управлять ресурсами
- Взаимодействовать с клиентом

Функции тестера:

- Проверять качество результата

Функции аналитика:

- Изучать предметную область клиента
- Переводить задачи с языка клиента на язык разработки
- Проектировать решения, отвечающие потребностям клиента.

Для небольших проектов эти роли можно совмещать. Особенно в случаях когда бизнес-логика на проекте не является очень сложной. Почему лучше обойтись одной ролью? Просто тогда усложняется взаимодействие между ними. Кто-то должен координировать это взаимодействие и система получается еще более сложной.

Например, у клиента возникла потребность что-то доделать. Он обращается к менеджеру и начинает объяснять, но менеджер не понимает его, т.к. бизнес-логика – не его компетенция. Либо клиент может обращаться к аналитику, но тогда менеджеру сложнее контролировать влияние клиента на проект.

На мой взгляд, в большинстве случаев можно обойтись одним человеком для совмещения этих ролей.

Пожалуй, это была самая важная глава. В ней мы рассмотрели, как вести ежедневно проект, как планировать итерации и какие взаимодействия бывают на проекте. Уделяя достаточно времени проекту по указанной выше схеме и при адекватной оценке ресурсов, вы уже вероятно доведете проект до логического результата. Однако вам необходимо изучить 2 специальных этапа, без которых не обходится ни один проект. В следующей главе мы как раз поговорим об одном из них.

Глава 7. Внедряем в эксплуатацию

Очень часто в проектах бывает так, что ввод в эксплуатацию происходит неявно. Т.е. делали, делали, а потом раз – и сайт уже работает в боевом режиме с реальными посетителями. У этого подхода есть ряд минусов:

- Посетители будут видеть ошибки и недоработки
- Есть риск вместе с тестовыми данными почистить реальные данные
- Могут быть ошибки, которые будут критичными для SEO и рекламы.

Учитывая эти моменты, мы распишем свое видение как надо внедрять проекты в эксплуатацию.

Первое, чем надо заняться, - это общее тестирование. Вы должны проверить совместную работу всех модулей, попробовать основные workflow ваших пользователей, т.е. сценарии, по которым будут действовать ваши посетители.

Также вы должны проверить корректность работы сайта в разных браузерах и на разных платформах. Конечно, проверять следует те браузеры и платформы, которыми пользуется ваша целевая аудитория.

Тестирование старайтесь вести на реальных данных и очень желательно провести тест на больших объемах данных. Со временем база данных может разрастись, и приложение, которое очень шустро работало в начале, может начать дико тормозить на больших объемах данных. Поэтому забейте программно в базу данных множество информации и посмотрите, как реагирует на это система.

При тестировании полезно выполнять параллельно ручные вычисления. Только так вы сможете понять – ошибается программа или нет.

И последний момент по тестированию – это тестирование бизнес-логики с представителями от клиента. Именно они будут использовать ваш разработанный продукт. Именно они знают, как он должен работать. Именно они знают все тонкости бизнес-процессов и могут указать на неявные ошибки в бизнес-логике приложения. Если вы закончили тестирование полностью, то дайте потестировать сайт третьей стороне или специализированной конторе. Уверен, что будут найдены еще недочеты, которые вы сможете исправить до запуска.

Следующий крупный блок – это перенос приложения на боевой сервер с предварительной настройкой этого сервера. Здесь в первую очередь следует побеспокоиться о безопасности данных приложения:

- Смените все пароли в системе (SQL Server, SVN, панель управления).

- Настройте правильно брандмауэр (ограничение по IP к SQL Server, закрыть все лишние порты и т.д.).
- Сделайте настройки на SQL сервер (закрыть доступ к sa, можно вообще внешний доступ отключить в SQL Server).
- Правильно установите настройки приложения на production среду (убрать режим отладки, отключить трассировку, настроить локальное подключение к базе, изменить ключи шифрования, поменять настройки/пароли на почте).
- Обеспечьте доступ только тем лицам, которые будут назначены на сопровождение проекта на этапе эксплуатации.
- Проработайте в системе создание резервных копий базы данных и файлов приложения.
- Проработайте план восстановления приложения после краха базы, краха сервера или другого форс-мажора. Проработайте всевозможные риски и соответствующие меры/инструкции.

Также необходимо настроить дополнительные сервисы, а именно:

- Мониторинг доступности приложения (например через UptimeRobot).
- Периодическое создание бекапов (либо внутри сервера, либо через сервис хостера, который предоставляет сервер).
- Мониторинг параметров сервера. Есть специальные инструменты (SNMP), которые позволяют это делать. Для этих задач вам потребуется системный администратор.
- Проверка работы почты, отправки СМС. Здесь вам необходимо создать регламент периодической проверки отправок. Иначе есть риск не получать важнейшие сообщения по системе.

Следующий шаг, который вы должны сделать – это перевести все сервисы в боевой режим. Это может быть, к примеру, прием платежей на сайте, отправка СМС и др. Проверьте, что они корректно работают в боевом режиме. Вы немного потратите средств на это – это лучше, чем спустя месяц узнать, что у вас, оказывается, не работает регистрация через социальные сети или прием платежей.

Далее мы должны подключить корректно скрипты аналитики и провести первичную поисковую оптимизацию проекта, если это необходимо. Это включает в себя следующие мероприятия:

- Проверить sitemap.xml
- Проверить robots.txt
- Зарегистрировать сайт в Google и Yandex
- Проверить, что везде корректно установлены title и description в соответствии с набором ключевых слов (семантическим ядром).
- Проверить работу редиректов 301, если они используются.
- Проверить сайт на наличие битых ссылок.
- Провести анализ сайта в различных SEO сервисах, например, **cy-pr**.com.
- Проверить ваш сайт в сервисе валидации HTML и CSS http://validator.w3.org/.

Мы не рассматриваем каждый пункт подробно. Пусть это будет просто чек-листом для вас перед запуском. Более подробную информацию вы сможете узнать у SEO-мастера, который занимается вашим сайтом.

Также не забывайте о документации. Сразу определите набор документации. Для оператора системы, для администратора, для программиста, который будет поддерживать проект.

Документацию можно сделать в формате видеоинструкций или кратких текстовых инструкций. Не нужно делать увесистые тома. Помните, что главное качество хорошей документации – это скорость внедрения человека в систему. Если она слишком сложна, то пользователь инстинктивно будет пробовать работать в системе наугад. Хороший вариант для документации – это презентации или комиксы. Расписывайте типовые процессы в комиксах. Это увлекательно и доступно для всех.

Следующий момент – рекомендую проверить быстродействие вашего веб-приложения в Google Page Speed. Это очень удобный сервис, который позволяет даже новичкам значительно оптимизировать загрузку вашего сайта.

Как лучше вводить пользователей? Правильный ответ: постепенно.
Если у вас система для внешнего пользования, например, интернет-магазин, то сначала дайте небольшую рекламу и посмотрите на активность пользователей через аналитику. Тем самым вы на небольшом объеме сможете понять – есть ли ошибки на сайте.

Если у вас система для внутреннего пользования, например, CRM, то сначала дублируйте процессы в новой системе и в старой (например, это может быть Excel) + переводите не всех пользователей, а только 1-2 человек. Таким образом, вы сможете в миниатюре начать работу системы, постепенно подключая новых пользователей и со временем отключив старый метод работы.

Очень большая ошибка сразу одним махом вводить всю систему – делая это, вы закладываете риски получить крайне негативные результаты:

- Будут потеряны деньги, например, на массовую рекламу.
- Будет утрачена лояльность вследствие критических ошибок.
- Возникнет напряженность в отношениях заказчика и разработчика. Заказчик в этом случае будет винить в неудаче разработчика. Частично он будет прав, но лишь частично: на стадии эксплуатации риск возникновения ошибок очень велик, и заказчик должен заранее обработать этот риск. Постепенное внедрение в эксплуатацию - и есть мера по борьбе с этим риском. Это уменьшает критичность этого риска.

Мы обсудили первый из специальных этапов, которые завершают проект.
Рассматривайте ввод в эксплуатацию именно как отдельный этап со своей

последовательностью действий, иначе могут возникнуть проблемы, о которых мы говорили в начале главы.

Переходим к последней главе – к завершению проекта. Честно признаюсь, мы очень часто упускаем этот момент в виду быстрого перехода на следующие проекты. Правильно завершить проект – это значит повысить качество будущих проектов. Об этом мы и будем говорить в следующей главе.

Глава 8. Завершение проекта

Самое важное, что вам нужно сделать на этапе завершения проекта – это ретроспектива. По сути, ретроспектива – это взгляд в прошлое. Обычно, - это небольшое совещание между участниками проекта, на котором обсуждаются следующие вопросы:

- Что было сделано хорошо?
- Что можно улучшить?
- Анализ метрик проекта.
- С какими проблемами мы столкнулись и как решили?
- Какие практики надо сделать постоянными?
- Получение обратной связи для каждого члена команды в плане участия в проекте.

Результаты подобного совещания лучше документировать для последующей оптимизации ведения проектов. Ретроспектива вам помогает расти как команде и повышает общую культуру ведения проекта и понимание этого вопроса всей команды, а не только менеджера.

Какие метрики можно документировать:

- Оценочные часы / Фактические часы. Насколько мы отклоняемся от оценки?
- % брака по задачам.
- Вклад в проект исполнителей (количество часов/задач).
- % выхода за сроки.
- Соблюдение параметров проекта (сроки, объем, бюджет).

Следующий момент – это освобождение ресурсов, связанных с проектом. В первую очередь, - это разработчики. Обычно они перекидываются на другой проект, поэтому, чем быстрее вы это сделаете, тем лучше будет для других проектов. Также вы освобождаете тестовую площадку – сайт, базу, пул в IIS, чтобы не расходовать ресурсы тестового сервера. При этом обязательно уведомите всех участников проекта об этом, чтобы не было недопонимания. Например, заказчик будет и дальше взаимодействовать с разработчиком, который уже перешел на другой проект.

Очень важной частью завершающего этапа является определение режима поддержки проекта после ввода в эксплуатацию. Очень мало проектов, которые один раз разработали, и они так и работает в неизменном виде. Уже по ходу проекта возникают новые идеи, как можно улучшить проект. В период эксплуатации системы реальными пользователями таких идей будет гораздо больше. Поэтому необходимо выделить некоторые ресурсы на развитие проекта. В первую очередь надо определиться с программистами, которые будут ответственны за сопровождение проекта. Также надо иметь договоренности с клиентом о режиме, в котором будет проходить поддержка. Очень часто бывает так, что проект закончили, программист переходит на новый проект, и

не хватает временных ресурсов на доработку существующего проекта. Необходимо сразу предвидеть, какой объем доработок будет у проекта в период эксплуатации.

Для этого составляется ориентировочный план развития проекта, в котором прописываются модули и подсистемы, которые будут реализованы по мере развития проекта. Сюда может войти все, что не вошло в проект по каким-то соображениям (обычно это либо бюджет, либо сроки).

Вашей обязанностью перед клиентом, как менеджера проекта, является предложение вариантов развития проекта. Вы предлагаете клиенту все разнообразие функционала и возможностей, которые можно в будущем внедрить в проект. Что ставить, а что нет – это уже дело клиента. Поэтому у вас в запасе должен быть такой список, в котором перечислены эти самые модули. Это может быть что угодно, например, чат, видеотрансляции, аудиокомментарии к задачам, дополнительный контроль целостности базы, обработка сообщений с почты и т.д.

План развития проекта очень желательно составить документально и согласовать с клиентом. Тем самым у вас будут договоренности на будущее по развитию проекта.

Немаловажная часть для вашей компании – маркетинговая. Очень желательно взять отзыв клиента, когда он максимально доволен вашей работой в эмоциональном плане. Вы можете составить отзыв в форме интервью, где вопросы закрывают какие-то возможные возражения будущих клиентов. Помимо отзыва хорошо бы получить разрешение на написание кейса по разработке. Кейсы очень полезны в плане проработки новых клиентов.

Также немаловажной частью является получение обратной связи от клиента в плане проработки качества. Составьте небольшую анкету для клиента и задайте следующие вопросы:

- Вы довольны результатом проекта?
- Почему выбрали именно нас?
- Что больше всего понравилось на проекте?
- Что больше всего не понравилось на проекте?
- Что можете сказать по взаимодействию с менеджером и разработчиками?

Последнее, что вам нужно сделать на проекте – это отпраздновать коллективную победу! Проект – это долгий, непростой процесс, в котором может участвовать множество людей. Обязательно надо ставить хорошую психологическую точку в проекте. Там самым вы можете эмоционально закрыть проект и двигаться дальше – к новым, более сложным проектам!

Заключение

Написанная книга – это тоже проект. Мы подошли к концу этого проекта. В результате у вас есть система взглядов на то, как вести проект.

Не пытайтесь внедрить все сразу. Внедряйте постепенно. Адаптируйте каждую практику по ведению проекта последовательно. Для начала используйте только указанную структуру ведения проекта, такие как концепция, итерации, внедрение. После прочной адаптации вы можете уже добавлять другие моменты, например, метрики разработчиков, оценки по задачам и т.д.

Еще важный момент, о котором хотелось бы упомянуть – это здравый смысл. Отталкивайтесь от здравого смысла, а не от модных методик. Адаптируйте то, что дает конкретный результат. Пробуйте новые практики и проверяйте их на деле, оставляя только то, что действительно работает и улучшает ваш результат.
Если у вас еще нет какого-то своего проекта – то придумайте! Проектом может быть что угодно. Наша книга заточена под веб-проекты, но никто не мешает использовать полученные знания для любых проектов.

Последнее, если вам понравилась книга, напишите отзыв на страничке http://web-automation.ru/book/project-man/. Ну, а если очень понравилась, то сделайте, пожалуйста, пост в социальной сети со ссылкой на книгу. Книга – бесплатная, и мы не продвигаем ее через какие-то рекламные площадки, поэтому ваш пост очень поможет доставить ее тем, кому она действительно нужна.

Удачного проекта!