

The Lean Launch

ConceptCodify

The Lean Launch

ConceptCodify

This book is for sale at <http://leanpub.com/leanlaunch>

This version was published on 2013-10-02



This is a [Leanpub](#) book. Leanpub empowers authors and publishers with the Lean Publishing process. [Lean Publishing](#) is the act of publishing an in-progress ebook using lightweight tools and many iterations to get reader feedback, pivot until you have the right book and build traction once you do.

©2013 ConceptCodify

Contents

The Lean Launch	1
Define, Phase 1/5	2
Design, Phase 2/5	6
Develop, Phase 3/5	10
Distribute, Phase 4/5	15
Iterate, Phase 5/5	22

The Lean Launch

ConceptCodify¹ was planned, designed, developed, and launched, all with under 100 hours of combined time. How was this possible? Many talk about [minimum viable product](#)²; we actually did it and then some.

We'll cover the process we used to get ConceptCodify to launch. There's essentially five major stages to our launch strategy: **define, design, develop, distribute, and iterate.**

Overall, there's about 24 total steps. That may sound like a ton, but most are fairly quick. Best of all, our plan keeps getting to launch user-focused and user-driven.

Phases

- [Phase 1, Define](#)³: *Thesis, Users, Goals, Objects, Relationships, Attributes, & Processes*
- [Phase 2, Design](#)⁴: *Requirements, Wireframes, Style guide, & Composition*

¹<http://conceptcodify.com>

²http://en.wikipedia.org/wiki/Minimum_viable_product

³<http://blog.conceptcodify.com/2013/05/12/the-lean-launch-define-phase-one-of-five/>

⁴<http://blog.conceptcodify.com/2013/05/18/the-lean-launch-design-phase-two-of-five/>

- [Phase 3, Develop](#)⁵: *Server, Models, Controllers, Views, Development, & Launch*
- [Phase 4, Distribute](#)⁶: *Branding, Account, Write, Business plan, Research, Legal, & Monetize*
- [Phase 5, Iterate](#)⁷: *Research, Design, Develop, Market, & Support*

Define, Phase 1/5

First, we'll be working on describing the process we used to get a [minimum viable product](#)⁸ release of [ConceptCodify](#)⁹ out the door in less than one hundred hours of total time. For the first phase, we focused on planning and research. This process focuses on a [top-down approach](#)¹⁰, and of course, is user centric throughout. We've started a few apps before and this process is the refinement of several launched MVPs.

Define, Step 1: *State your thesis.*

The very first thing you need to do once you have your thoughts together is come up with a thesis statement for

⁵<http://blog.conceptcodify.com/2013/05/20/the-lean-launch-develop-phase-three-of-five/>

⁶<http://blog.conceptcodify.com/2013/05/26/the-lean-launch-distribute-phase-four-of-five/>

⁷<http://blog.conceptcodify.com/2013/06/01/the-lean-launch-iterate-phase-five-of-five/>

⁸http://en.wikipedia.org/wiki/Minimum_viable_product

⁹<http://conceptcodify.com>

¹⁰http://en.wikipedia.org/wiki/Top-down_and_bottom-up_design

your product. The easiest way to do this is to define the problem you are solving first, and then state how you are going to solve it. Don't focus on getting all the features or details down yet, leave it to a sentence or two just to get started. But take the time to make this the most important sentence you'll ever write.

Example: *ConceptCodify is a card sorting application designed so everyone making websites and apps can conduct their own card sorts without expert research knowledge.*

Start a document at this point. I love [markdown](#)¹¹ for these types of documents, but any word processing environment will do.

Define, Step 2: *List your types of users.*

The next thing you need to do is list out the major types of users that will use your product. Some cases, there will only be one type of user, most of the time there is two. Sometimes there will be up to five types of users. More than that and you'll probably go crazy. Don't get caught up in the details yet, but come up with a good word for each type of user, such as *researcher* and *participant*, because you'll be using them a ton.

Define, Step 3: *Define the goal for each user type.*

Just as you defined the goal for the overall project, take this time to list the problem and solution for each type of user. Refine it down into one sentence each. Once you

¹¹<http://daringfireball.net/projects/markdown/>

have this, you'll likely want to similarly define some of the [expectations, resources, motivations, and attitudes](#)¹² for each user type. You may need to do a little user research at this phase to be able to articulate these aspects of your users. [Qualitative surveys](#)¹³ can help to illuminate these aspects quickly. You may even realize there are more types of users as well.

Define, Step 4: *Define the objects in the process.*

The next thing we need to define are the major conceptual objects the users will be interacting with as they try to accomplish their goals. For example, in ConceptCodify, the objects are *studies*, *collectors*, and *participant responses*. These are somewhat abstract objects. You'll likely revisit and refine this step as you work through the next few steps, and that's an important part of the planning process.

Define, Step 5: *Define the relationships between users and objects.*

Now that you have a list of users and objects, you'll want to draw out a map of their *major* interactions. Don't worry about every little detail, refine so that each user type has about 2-3 major interactions. I find it easiest to write each of the user types and objects in a circle, and draw lines between them.

¹²<http://blog.conceptcodify.com/2013/01/19/the-five-most-important-aspects-to-know-about-your-users/>

¹³<http://blog.conceptcodify.com/2013/02/10/user-research-techniques-surveys/>

You'll likely want to add each relationship in your document under each type of user and object.

Define, Step 6: *Define the attributes of each user and object.*

Now that we know what the major relationships are, we can now talk about what information we'll need for each type of user and object. For example, users will typically have a user name, email address, and password. *Researchers* in ConceptCodify also have notifications settings, organization name, and logo. Go through each type of user and object and determine all the attributes you'll need. Again, as you continue through your journey, you'll come back and revisit and refine.

Define, Step 7: *Define the major user processes.*

Now that we have a basic understanding of who, what, and why, we need to define how. Don't focus on visuals or layouts just yet. At this stage, revisit your user goals. Create a step-by-step process for how the user interacts with the objects and other users to achieve their goal. Describe what attributes of the objects and the users are impacted at each step. Continue to refine this step. You may need to revisit and refine earlier steps at this point. If you are also unclear as to your users' *expectations* or *resources*, I highly suggest conducting some user research to clarify your users' thoughts during these early planning stages. You may want to review our article on the [cognitive](#)

conversion process¹⁴ at this stage.

If you already have a name in mind, now's not a bad time to register your domain name¹⁵ and procure¹⁶ the user name on Facebook, Twitter, and any relevant social media sites.

Review

- The seven steps of the of *define phase* are Thesis, Users, Goals (Personas), Objects, Relationships, Attributes, and Processes.
- Don't worry about visuals or implementation yet. Start from the largest ideas, the big picture, and break it down from there.
- Conduct research where you are unsure. This is the time where research will have the most impact.
- You don't have to get it perfect. You'll refine your definition as you work on your project.

Design, Phase 2/5

Continuing on from our process in the *define phase*¹⁷, we now have a document that lists our thesis, users, their goals, objects, relationships, attributes, and processes. Now we're

¹⁴<http://blog.conceptcodify.com/2013/02/01/a-cognitive-model-of-conversion-that-will-vastly-improve-your-site-or-app/>

¹⁵<http://gandi.net>

¹⁶<http://namechk.com>

¹⁷<http://blog.conceptcodify.com/2013/05/12/the-lean-launch-define-phase-one-of-five/>

ready to start getting down to realizing our plan on paper. But don't jump to [Photoshop](#)¹⁸ just yet! As we've gone with a top-down approach, we'll continue that in the design phase as well.

Design, Step 1: *Create a series of screen requirements.*

Now it's time to translate our processes into a conceptual set of screens. What steps in the processes logically combine together? What states do each element appear (e.g. initial, focused, accepted, error)? You may need to conduct some user research, such as [card sorting](#)¹⁹ or participatory design, to correctly organize these elements into screens. Don't worry about layout or visuals just yet, first focus on listing out what needs to be on each screen, and how the screens connect to each other.

Design, Step 2: *Draw up wireframes for each screen.*

Now that you have your screen requirements, its time to figure out how to lay them out. There's a million [wire-framing tools](#)²⁰ out there; I'm still an advocate for pen and paper wireframing. Don't think about graphical elements yet, just focus on laying out each screen and making them work together.

Pay particular attention here to keeping similar functions on the same part of the page throughout. Also emphasize

¹⁸<http://www.photoshop.com/>

¹⁹<http://blog.conceptcodify.com/2013/04/16/user-research-techniques-card-sorting/>

²⁰<http://www.balsamiq.com/>

the [hierarchy](#)²¹ of elements; which elements are the most important?

Normally, it's a good idea to come up with several wireframes for each screen, and to look at them collectively to find patterns that work well together. You may wish to conduct [usability testing](#)²² on your wireframes, or surveying to decide user priorities.

Design, Step 3: *Design a style guide that defines the look and function of elements.*

Now that we know what elements are going to appear throughout our project, and we have an idea of how things will be represented, we can come up with a [style guide](#)²³. You'll want to develop a consistent look and feel that covers most of the element you need. If you aren't familiar with style guides for application projects, you may wish to look at the [Twitter Bootstrap](#)²⁴ or [Github CSS Style Guide](#)²⁵ as well-developed examples.

Your style guide should end up in the medium of the project, be it HTML and CSS or XCode ready so that it is put into practice. Pay particular attention to typography, grids, and metrics, as these will define most of your other elements. I highly recommend conducting user research

²¹http://en.wikipedia.org/wiki/Maslow%27s_hierarchy_of_needs

²²<http://blog.conceptcodify.com/2013/03/07/user-research-techniques-usability-testing/>

²³http://en.wikipedia.org/wiki/Style_guide

²⁴<http://twitter.github.io/bootstrap/>

²⁵<https://github.com/styleguide/css>

in this stage as well to find out how your users identify themselves, such as intellectual, funny, detail-oriented, opinionated, etc., and let the visuals reflect your users.

Design, Step 4: *Compose your screens.*

In the last stage of the design phase, you want to put the wireframes and the style guide together to make compositions of each screen. You can develop these compositions in any medium you choose, but I'd recommend in the end having them ready to be implemented in code as with the style guide. In some cases, once you have the style guide and wireframes, that may be enough information by itself to generate your pages: YMMV²⁶. In more complex cases, a composition will be necessary.

If you have more than one person on your team, it's easy to distribute the project across multiple people and in an iterative fashion. You don't have to do the steps in precise, [waterfall](http://www.internetslang.com/YMMV-meaning-definition.asp)²⁷, order. You'll need to start the planning phase first, but once you define the user goals you can start design and development simultaneously and iteratively.

Review

- You'll want to do most of the *define* phase before entering the *design* phase.

²⁶<http://www.internetslang.com/YMMV-meaning-definition.asp>

²⁷http://en.wikipedia.org/wiki/Waterfall_model

- The four steps of the design phase are *screen requirements*, *wireframes*, *style guide*, and *composition*.
- Use a top-down approach. It's easier to make good decisions when you focus on one problem at a time.
- Conduct user research as you design; research data will make your decision-making much easier and quicker.

Develop, Phase 3/5

Now that's we've discussed the [define](#)²⁸ and [design](#)²⁹ phases of building an application, we're ready to get technical. The develop phase can largely run in parallel with the design phase. As always, we'll take a top down approach and we'll use the planning we started in the *define* and *design* phases to help organize us in the *development phase*.

Unlike the *define* and *design* phases, development does not necessarily need to go in order. As development also tends to be 80 to 90% of the work, if you can find a way to break development into smaller pieces and set flexible milestones, do so. You may also wish to identify the pieces that actually need to be there for launch, and save the rest for afterwards. If you have a team, I highly suggest using one of the major engineering management paradigms, such as

²⁸<http://blog.conceptcodify.com/2013/05/12/the-lean-launch-define-phase-one-of-five/>

²⁹<http://blog.conceptcodify.com/2013/05/18/the-lean-launch-design-phase-two-of-five/>

[Scrum](#)³⁰, [Kanban](#)³¹, or XP, as you develop towards launch. We won't cover those methodologies, but instead we will focus on how the work in the define and design phases ties into development.

For this phase, we assume your project will fall into the [Model-View-Controller paradigm](#)³², as is typical of web and mobile applications. Some projects may not fall into this category; in that case, you'll need to adjust your expectations and find alternative ways to build your planning into your development work.

Develop, Step 1: *Pick your technological stack.*

There's a few decisions you'll need to make before you fire up your code editor. Do you need a hosting service? Domain name? Email management? What languages will you use? What libraries and systems will you need? These decisions will depend both on the needs of your project and the skill set of you and your team. You don't have to make every decision up front, but you'll definitely want to know how you'll be storing information, how you'll be routing users to that information, and what you'll use to allow users to take actions. You'll want these major components settled before you start digging in.

Once you have your technical stack decided, I'd recommend getting a splash page up and let people enter their email addresses so you can notify them at launch. You can

³⁰http://en.wikipedia.org/wiki/Scrum_%28development%28

³¹<http://en.wikipedia.org/wiki/Kanban>

³²<http://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller>

also ask these same people to help with your research and testing.

Develop, Step 2: *Use your attributes and relationships from the define phase to make models.*

In code, models are representations of users and objects that allow various attributes to be stored about them. It's often best to start at the model layer. Use the documentation from the *define* phase about attributes and relationships to get started. The documentation won't map out 100% to your [Object-Relational Mapper](#)³³ (or similar system), but the documentation will ensure that how the application is planned and designed stays in sync with the way the application is developed.

Develop, Step 3: *Use your processes from the define phase to pseudocode controllers.*

Now that you have your basic models set up, it's a good time to use the *processes* from the *define* phase to start pseudo-coding controllers. Controllers define how an application functions. Most developers also like having the screen requirements and wireframes from the design phase before starting here. I would recommend thinking of the [RESTful](#)³⁴ actions while figuring out how to organize the functionality: show, list, new/create, edit/update, and delete. This pattern can explain the majority of actions that a user can take on themselves, another user, and various objects in the system. I would mostly write the ideas down in

³³http://en.wikipedia.org/wiki/Object-relational_mapping

³⁴http://en.wikipedia.org/wiki/Representational_state_transfer

comments first, and come back later to turn them into real working code. You'll likely want to establishing your routing ([URLs](#)³⁵) as well before moving on to the next step.

Develop, Step 4: *Use all four steps from the design phase to implement your views.*

It likely makes sense at this point to migrate the design work into the code base. *Views* is the technical term for how the functionality is presented to the user. You'll likely want to get the style guide into the system first, either in [CSS](#)³⁶, [XCode](#)³⁷, or whatever tooling you are using to develop your project. Then, use placeholder text and images to get the views laid out within your templating system. Don't focus on getting every detail on the first go; as with everything else in this process, use a top-down approach. Resolve the big issues first and then drill down into specifics later on.

Develop, Step 5: *Process-by-process, develop out your project.*

Divide your application by *processes*. Try to accomplish getting one process to work at a time. You'll need to translate pseudocode into real code, adjusting your models and views to work with the system as you go along. The order in which you develop will likely be obvious, but when in doubt, use a top-down approach again.

As you finish each process, I highly recommend setting up a variety of testing on each process so you don't get too far

³⁵http://en.wikipedia.org/wiki/Uniform_resource_locator

³⁶http://en.wikipedia.org/wiki/Cascading_Style_Sheets

³⁷<http://developer.apple.com/tools/xcode/>

ahead without regular testing. The types of testing you'll want will likely include usability, accessibility, multivariate/split, unit, functional, quality assurance, and sometimes performance and security. If you built it yourself, have someone else test it for you as well; don't rely on only one person for testing. Use your documents from the define and design stages to verify your work.

Conflicts will arise as you develop your application. Use your planning and user research data to resolve conflicts. Building anything is always more challenging than your original estimate; be flexible, communicative, and compassionate. Revisit and update earlier documents as needed. Conduct more research if you need to. Don't be afraid of pushing some things out to after launch; your goal is to build just enough to start getting people to use your project. Any more than that will likely be too difficult; you can and will add more later. And as always, keep a sustainable pace.

Develop, Step 6: *Launch and party!*

Once you've developed out all the processes you need to start getting people to use your project, and you've testing out those processes, make a final sweep. And then, *launch it!*

As always, take the time to celebrate your successes, big and small. Also allow yourself time to breathe, reflect, and prepare mentally for the next phase. All of this work, and soon reality sinks in... you have to get the word out, and start establishing yourself and your organization.

Review

- Let your planning and research guide your technological organization and decision-making.
- Keep the top-down approach going. Focus on the big picture first.
- Stay flexible, keep calm, work sustainably, and be compassionate to yourself and others. Building things is much more difficult than it initially may seem.
- Take everything in one step at a time. Break big things into smaller things as you need to.
- Active, ongoing testing is critical to development success.
- Celebrate your successes, big and small.

Distribute, Phase 4/5

I have to admit that *distribute* is perhaps not the ideal word for this phase. At any rate, now that we've made our plans, created the vision, and built it out, now it's time to deal with organizational issues and marketing. Most of this phase can be run concurrently with the previous three phases. I'll continue the 'steps' paradigm here but what matters more in the distribute phase is the prerequisites. I'll mention those at each step.

Distribute, Step 1: *Establish a branding guide for marketing.*

Prerequisite: Style guide (Design, Step 3).

Once you have a style guide, you can then set a branding guide that will help you in your marketing and outreach. You'll want to define a few different things: colors, logos (in different sizes), fonts, iconography, writing style, copy snippets, and screenshots. This doesn't have to be a huge document, just enough for the next step.

Distribute, Step 2: *Setup all organizational accounts.*

Prerequisite: Branding guide.

Now you'll need to go through and setup all the various kinds of account you'll need. This is a tedious process, but it is necessary. Here are some of the various types of accounts you may need:

- [Domain registrar](#)³⁸
- [Hosting](#)³⁹
- [Version control](#)⁴⁰
- [Analytics](#)⁴¹
- [Advertising](#)⁴²
- [Documentation](#)⁴³
- [Internal email](#)⁴⁴

³⁸<http://gandi.net>

³⁹<http://rackspace.com>

⁴⁰<http://github.com>

⁴¹<http://google.com/analytics>

⁴²<http://google.com/adwords>

⁴³<http://apps.google.com>

⁴⁴<http://gmail.com>

- Blog⁴⁵
- Support⁴⁶
- Project management⁴⁷
- Issue management⁴⁸
- Press kit⁴⁹
- Site monitoring⁵⁰
- Social media (Facebook⁵¹, Twitter⁵², LinkedIn⁵³, Youtube⁵⁴, Flickr⁵⁵...)
- Social automation⁵⁶
- Payment management⁵⁷
- Accounting⁵⁸
- Email marketing⁵⁹
- Surveying⁶⁰
- User research accounts⁶¹

⁴⁵<http://wordpress.com>

⁴⁶<http://uservoice.com>

⁴⁷<https://www.wunderlist.com/>

⁴⁸<https://github.com/features/projects/issues>

⁴⁹<http://totemapp.com>

⁵⁰<http://pingdom.com>

⁵¹<http://facebook.com>

⁵²<http://twitter.com>

⁵³<http://linkedin.com>

⁵⁴<http://youtube.com>

⁵⁵<http://flickr.com>

⁵⁶<http://hootsuite.com>

⁵⁷<http://stripe.com>

⁵⁸<https://www.waveapps.com/>

⁵⁹<http://mailchimp.com>

⁶⁰<http://surveymonkey.com>

⁶¹<http://uxmastery.com/resources/tools/>

- [Cloud storage](#)⁶²
- [Bank account](#)⁶³
- [PO Box](#)⁶⁴
- [Landing page system](#)⁶⁵
- [SEO tool](#)⁶⁶

As you can see, there are a ton of possible accounts you could setup. I recommend documenting all the accounts you setup for your project. I also recommend keeping it free or as close to free as you can; you'd run up the bills pretty quickly on accounts if you aren't careful. Use the branding guide as you setup the accounts, and try to get links back to your project as you setup accounts.

Distribute, Step 3: *Establish an editorial calendar.*

Prerequisite: Accounts.

Once you have your accounts setup, create a schedule for yourself with automatic reminders. You'll want to post a blog post almost every week and engage on social media sites regularly. Online marketing is about consistent presence; not flashy one time campaign.

I do mean engage, not spam. The secret of online marketing is to always give the user something in return for their time and attention. Just getting your name out there and a little

⁶²<http://dropbox.com>

⁶³<http://www.asmarterchoice.org/>

⁶⁴<https://www.usps.com/>

⁶⁵<http://unbounce.com>

⁶⁶<http://seomoz.com>

link in your signature is often all that is required. Focus on providing content, not advertising.

If you are new to online marketing, I recommend reading “[The Noob Guide to Online Marketing](#)”⁶⁷ by Oli Gardner from Unbounce. [SEOMoz](#)⁶⁸ and [Smashing Magazine](#)⁶⁹ have several excellent introductory articles on everything from content marketing, social marketing, email marketing, landing page marketing, search engine marketing, press releases, lead generation, and analytics. It’s overwhelming at first; I recommend sticking with content marketing (blogging, social engagement) until you start getting traffic.

Distribute, Step 4: *Write up a brief business plan.*

Prerequisite: User goals (Define, Step 3).

A short business plan can help focus your attention and provide a means to explain your project quickly to lawyers, accountants, and the like. I recommend, in about one to three sentences each, covering the following:

- Problem
- Your solution
- Business model
- Who is the market

⁶⁷<http://www.seomoz.org/blog/the-noob-guide-to-online-marketing-with-giant-infographic-11928>

⁶⁸<http://www.seomoz.org/blog>

⁶⁹<http://www.smashingmagazine.com/>

- Your competitive advantages
- Your marketing strategy
- Management team
- Financial plan

You'll refine your plan as you go.

Distribute, Step 5: *Setup a research calendar.*

Prerequisite: Launch (Develop, Step 6).

Once you've launched you'll want to setup a regular schedule to conduct user research. Set aside time each week to conduct a survey, card sort, usability test, or other types of studies as needed. You'll want to spend some time reviewing your analytics without getting paranoid. For each study, you'll want to plan it, conduct it, and analyze the results. User research is a cycle.

Distribute, Step 6: *Get legal.*

Prerequisite: Business plan.

Almost all projects will need to speak to an attorney at some point. Some recommend doing so early, some later. You may need to incorporate your project or organization if you are planning on charging your users directly. Business and nonprofit lawyers are fairly easy to find and not as expensive as you might initially think.

If your project is internal or a new feature, you'll have to check if you need to speak to an attorney before launching it.

Distribute, Step 7: Monetize.

Prerequisite: Legal, Schedules, Iteration.

Money is a fundamental resource for most projects. Many projects start with direct monetization as a primary goal. Others add revenue-streams on later. Some organization focus on donations or grants. Only you can decide how you will have resources to continue to market, support, and develop your project. You may need to speak with an accountant at this time.

If you reach monetization, be sure to celebrate!

Review

- Expenses can get out of hand quickly. Try to do everything free or as cheap as possible, at least initially. You can always move to a better system later.
- Continue the top-down approach from the other phases. Start with the most important priorities first, and drill down to the details later.
- Marketing and maintaining your project is all about consistent. Setup schedules and reminders for yourself so you don't fall behind.
- Take the necessary precautions, but don't be afraid to take some risks too.

Iterate, Phase 5/5

After you've [planned](#)⁷⁰, [designed](#)⁷¹, [developed](#)⁷², and [marketed](#)⁷³ your project, what's next? Iteration. We'll cover a basic strategy for iterating to develop your project.

The most important thing to know about this phase is that it exists. Often when we make plans, we see all or nothing. But the truth is, it is much easier to build a small, usable product at first and then slowly build it up to your vision through iteration. Iteration will repeat itself until the foreseeable end of the project. Break up big things into smaller iterations; you don't want to be working on the same features for several months.

The other important note here is that iteration largely follows the strategies of getting to launch and monetization. In fact, iteration is just another walkthrough of the phases all over again for the most part.

Iterate, Step 1: *Research the problem or opportunity.*

Before you dive into building again, take a step back to research the issue or feature. [Define](#)⁷⁴ how the new

⁷⁰<http://blog.conceptcodify.com/2013/05/12/the-lean-launch-define-phase-one-of-five/>

⁷¹<http://blog.conceptcodify.com/2013/05/18/the-lean-launch-design-phase-two-of-five/>

⁷²<http://blog.conceptcodify.com/2013/05/20/the-lean-launch-develop-phase-three-of-five/>

⁷³<http://blog.conceptcodify.com/2013/05/26/the-lean-launch-distribute-phase-four-of-five/>

⁷⁴<http://blog.conceptcodify.com/2013/05/12/the-lean-launch-define-phase-one-of-five/>

component will fit into your users, their goals, the objects you already have. How does it impact the relationships and attributes between types of users and objects in the system? What processes will be the most effective? Conduct user research to start working on a plan.

Iterate, Step 2: *Design the components needed.*

Just like in the [design phase](#)⁷⁵, except at this point you now have a style guide and existing wireframes. Update the screen requirements and wireframes first to reflect the new components. Then, as before, you may wish to do separate compositions outside of the technical system, or you may be ready to start developing immediately.

Iterate, Step 3: *Develop.*

Just like before in the [development phase](#)⁷⁶, start with a technical assessment. Will you need new libraries or APIs to complete the feature? Then, break it up process-by-process, and then develop and test as you go.

Iterate, Step 4: *Market.*

And here, we relive the [distribute phase](#)⁷⁷ again. After all, once you've launched the new feature or solution, the best way to get users to use it is to tell them. Use social media, blogging, and let others know about it. Some features or

⁷⁵<http://blog.conceptcodify.com/2013/05/18/the-lean-launch-design-phase-two-of-five/>

⁷⁶<http://blog.conceptcodify.com/2013/05/20/the-lean-launch-develop-phase-three-of-five/>

⁷⁷<http://blog.conceptcodify.com/2013/05/26/the-lean-launch-distribute-phase-four-of-five/>

solutions may require talking to an accountant or lawyer, just like in the distribute phase; you'll likely want to do this before deploying them.

Iterate, Step 5: *Support*.

There's one more piece of the puzzle we haven't mentioned quite yet in this series, and that's support. There will inevitably be questions, technical problems, and feedback your users will want to give as they use your product and as you iterate it. Be sure to set aside resources to manage responding to these inquiries.

Review

- Iteration is key to long-term success. Don't commit to multi-month projects! Break things down into smaller parts
- Iteration requires the same steps as the first build. While you have some things already established, you need to account for your earlier work as you build.
- Celebrate successes, and let your users know about them!

We've mentioned a few times as well that the [waterfall method](http://en.wikipedia.org/wiki/Waterfall_model)⁷⁸ of trying to do everything in discrete steps is often

⁷⁸http://en.wikipedia.org/wiki/Waterfall_model

problematic. As a means of ending our journey, here is a flow chart that links up the prerequisites over our process.

[A flow chart of the Lean Launch process⁷⁹](#)

⁷⁹<http://conceptcodify.files.wordpress.com/2013/06/leanlaunch.pdf>