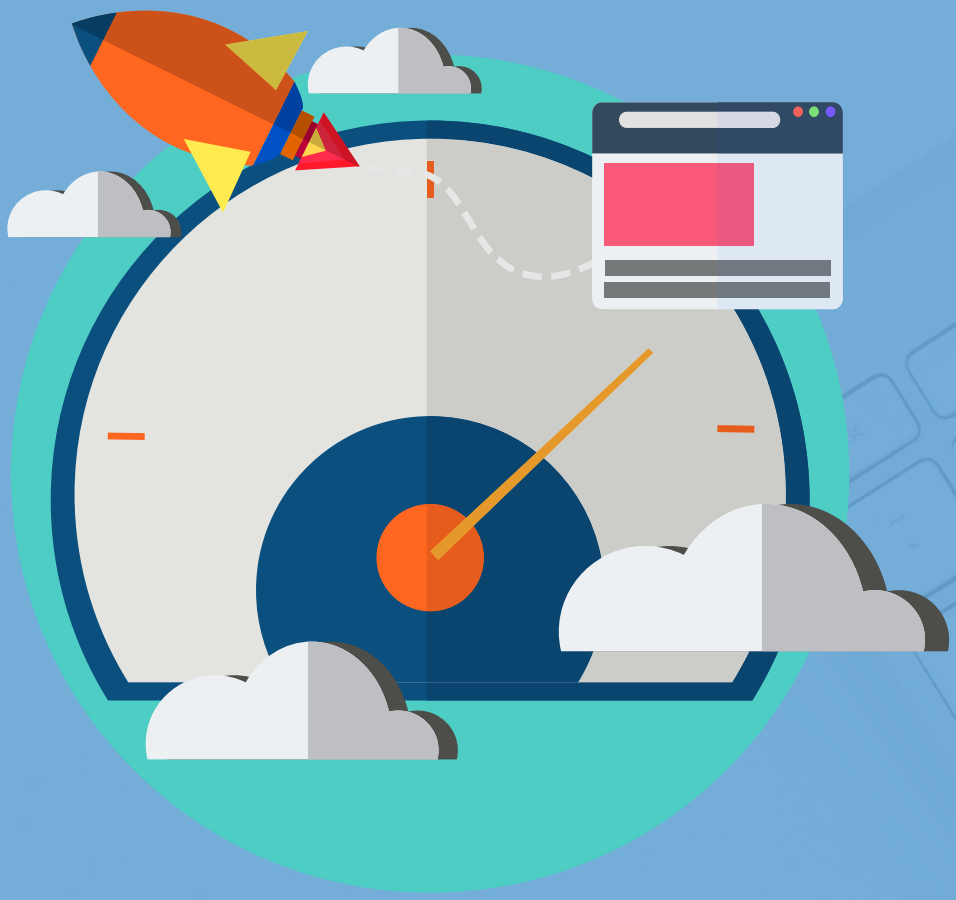
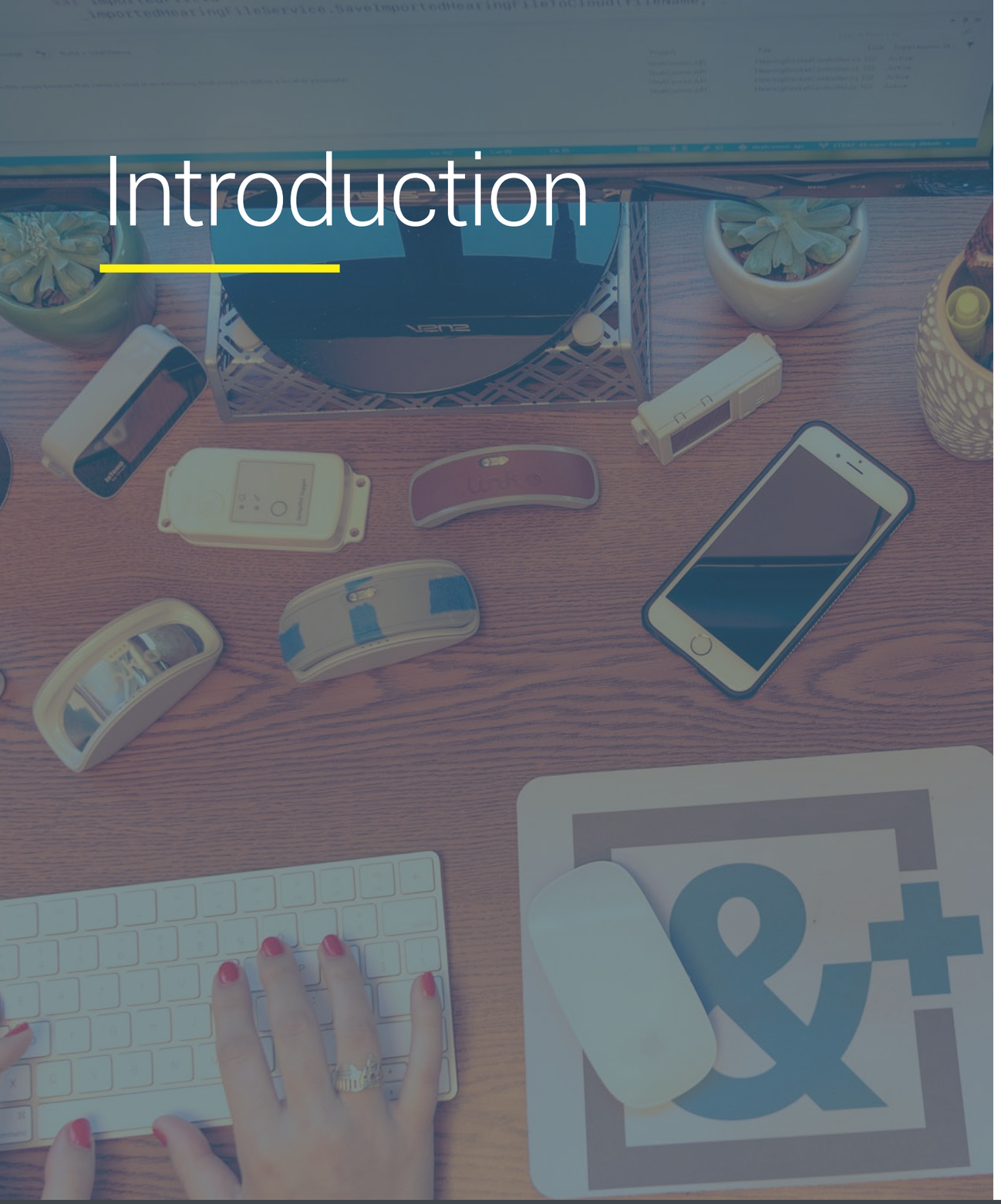
The 100 Day MVP



How to Design, Build, and Launch Your Next Project in 100 Days

AndPlus



Introduction

The history of software development is littered with methods that were slow, inefficient, and often resulted in buggy code, impenetrable documentation, and unusable user interfaces. The drive to “get it out the door” often conflicted with the goal of delivering software that actually makes users’ lives easier.

Software development, it turns out, is not particularly hard—middle-school kids do it all the time. *Good* software development is much more difficult to achieve.

So, after many false starts and peculiar fads, the practice of software engineering has finally arrived at practices that can overcome the problems of the past. The most popular of these include the “agile” methodology and its sidekick, “scrum.” In the right environment, these methods have proven invaluable in breaking down the complex task of developing software into manageable chunks. The result is high-quality software that meets the needs of its users.

It’s not easy to go from zero to agile; programming chops alone won’t cut it.

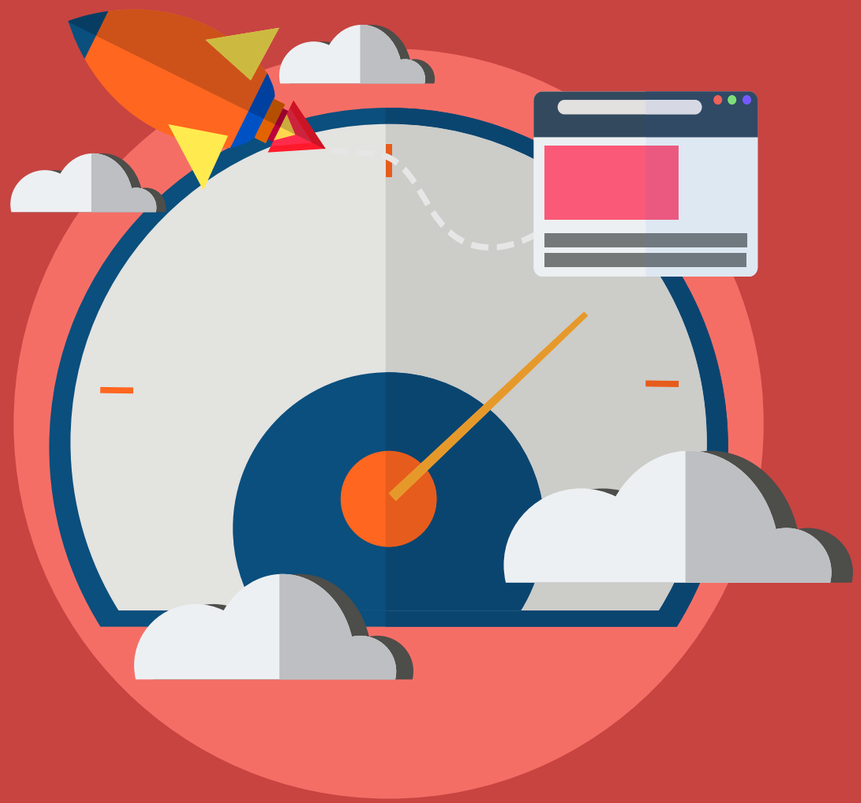
In addition to top-tier coding talent, your team needs focus, discipline, leadership, and teamwork—and above all, practice. Each project that a team works on should be better than the last.

At AndPlus, our long years of experience have taught us what works—and what doesn’t. The method we have settled on is based on the agile methodology. We call it the **100-Day MVP** method, in which a project is rigorously broken down into a 100-day cycle that results in the “minimum viable product” (MVP)—that is, a solid, basic “version 1.0” on which future enhancements can be built.

This e-book outlines the 100-Day MVP method and how we use it to deliver top-quality software.

Chapter 1

Days 1-10:
The Design Sprint



AndPlus

The Design Sprint:

Before You Start

Although the Design Sprint starts on Day 1, there is quite a bit of work that has to happen beforehand. Make sure you and your team understand the following before starting any kind of design work:

What problem are you solving? Make sure the customer's business problem or product vision is thoroughly understood. What should end users be able to do with the software that they can't do (easily, or at all) now?

Who are the users? What are the different types of users or target market for the software? What are their characteristics and demographics? Do they speak different languages, or conduct transactions in different currencies? In what environments, and on what devices will they be using the software? How do they currently perform the tasks with which the software will help?

What are the specific requirements? Make sure the high-level software requirements are thoroughly known and documented, and remember that you are only interested at this point in what the software needs to do—the how will be reflected in the design at the end of Day 10.

What's the revenue model? If the software is intended to generate revenue for your customer, how do they intend to do it; direct sales, in-app ads, free vs. premium versions? This choice has a strong influence on the overall solution design.

What is your team? Who are the coders, database experts, testers, scrum master, project sponsor? Who has vacation scheduled in the next 100 days? Who is covering for vacationers?

Armed with this information, you will be ready to start the Design Sprint.

The Design Sprint

Days 1 and 2: Map

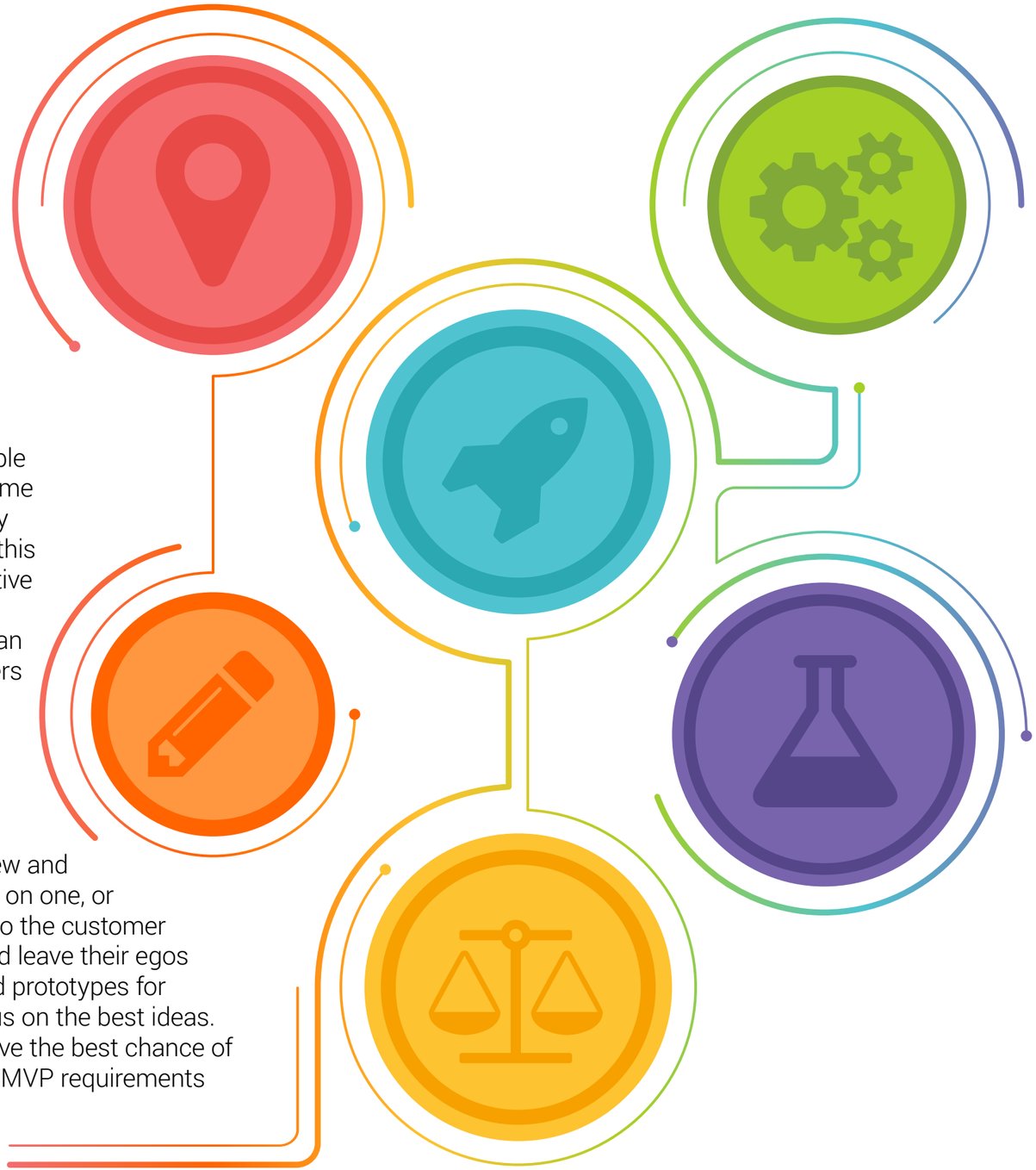
The task for the first two days is to use the information you have gathered and agree on the end goal (usually, the MVP of the product). With the goal in mind, you put together a map that defines how you will reach the goal by Day 100.

Days 3 and 4: Sketch

These two days are for brainstorming possible solutions for the user interface. Although some of the design will be dictated, to an extent, by the target platform's standard conventions, this is your team's chance to come up with creative approaches that make the user experience more streamlined, intuitive, simple, or fun than conventional approaches. The team members will individually sketch their ideas on paper.

Days 5 and 6: Decide

At this point, the team gets together to review and evaluate the ideas, with the goal of deciding on one, or a small number of, design ideas to present to the customer later on in this sprint. Team members should leave their egos at the door. There's not enough time to build prototypes for every idea, so the team has reach consensus on the best ideas. Remember, the best ideas are those that have the best chance of meeting the goal of fulfilling the customer's MVP requirements within the 100-day project cycle.



Days 7 and 8: Prototype

Now you take those top designs and turn them into prototypes. Use a good prototyping tool that can give the customer a realistic model of the solution—and that you can use to make changes on the fly from customer feedback. Make sure you cover all of the important functionality that will distinguish this solution from others in the marketplace, or solve the customer's biggest business pains.

Days 9 and 10: Test

This is where the rubber meets the road: You will present your prototypes to the customer. Getting the customer involved at this point has the advantages of:

- Validating that the team thoroughly understands the customer's needs
- Getting buy-in from the customer on the UX design
- Level-setting expectations so that the customer understands exactly what they will (and will not) get on Day 100 At the end of Day 10, you should have the UX design nailed down and agreed to by both the team and the customer. Then it's time to design the underlying architecture that will bring the front-end design to life.

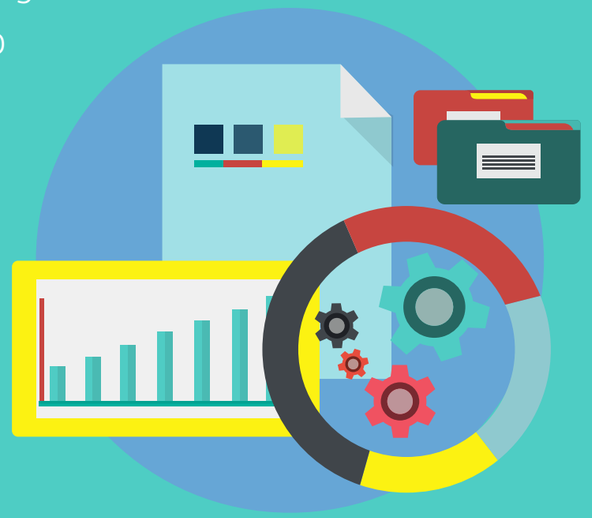
Chapter 2

Days 11-15: Technical Architecture

The technical architecture phase is where the team designs what goes on "under the hood" to support the UX design from the first phase. It's only five days, but it's an action-packed week in which many important decisions are made. It's just as critical to get this part right as it is to get the UX part right, because there will be precious little opportunity to fiddle with it later.

Here are the major components of this phase:

- Backlog Development
- Prioritization and Release Planning
- Development Environment Setup
- Ready, Set...



AndPlus

Backlog Development

Using an application called JIRA, the project is broken down into product backlog items, with features expressed as "User Stories"'. During this process, the team makes crucial decisions about the product architecture, that is, the major components of the system and how they will work with each other and with the hardware platform.

Some of the questions AndPlus will answer are:

- **What's the basic platform?** Web app, native mobile, traditional Windows/ Mac forms-based application?
- **What technologies are most appropriate?** Why? How does that affect the final product?
- **If the product is a mobile app, do you try parallel development in native Android and iOS, focus on one platform, or do you try a cross-platform approach?** Or do you go with a responsive web app instead?
- **What hardware interfaces are needed?** What APIs are needed to support them?
- **Does the app need to work with other apps?** How do they communicate with each other?
- **Do you need a database?** What kind? Where will it reside?
- **What frameworks are needed?**
- **What level of security is needed?** How will it be implemented?
- **Is the app standalone, or do you use a client-server model?** Thick client or thin?

The Product Owner (you!) is heavily involved in this process. If there are any additional clarifications or decisions needed from the customer, it's the Product Owner's job to get those answers so they can be incorporated into the design.

Prioritization and Release Planning

Hand-in-hand with backlog development is prioritization and release planning. When the backlog items are determined and broken down into manageable tasks, they are prioritized and organized into the development sprints that will occupy most of the remaining project lifecycle. Again, the Product Owner is a key player in this process.

Important considerations here include:

- MoSCoW (Must have, Should have, Could have, Won't have):
  Which pieces are absolutely essential, and which are more or less optional?
- Chickens and eggs
  What pieces have to be built before others will work? What pieces are mutually dependent and need to be built at the same time?
- Sprint-sizing
  This is a bit like bagging groceries. Given the constraints of dependency and priority, how many tasks can you fit into a development sprint without overloading it? Are there any tasks that are too big for a single sprint? Can they reasonably be subdivided into smaller tasks?
- Building in contingencies
  One of the concepts of agile software development is that of "discovered work," which is additional tasks that were not foreseen in the planning stage. Does the release schedule leave any wiggle room for this eventuality? It's best not to pack your sprints so tightly that any discovered work jeopardizes the schedule.

Development Environment Setup

This is where the team prepares all the tools they will need to develop the product.

This includes:

- Setting up the project in the application lifecycle management tool
- Setting up a test environment for executing test cases
- Preparing test devices—it's important to test not just on emulators but on actual devices, especially if you are going with cross-platform development or depend heavily on hardware interfaces
- Setting up automated builds and deployments
- Setting up a staging environment so that the team and customer can always access the latest release
- Preparing databases—even if you don't know the exact database structure yet, you can set up the database space, connection strings, and general permissions
- Laying down ground rules for the team, if anything is out of the ordinary
- Agreeing on a Definition of Done for the project

Ready, Set...

At the end of Day 15, the team should be thoroughly prepared to do what they do best:

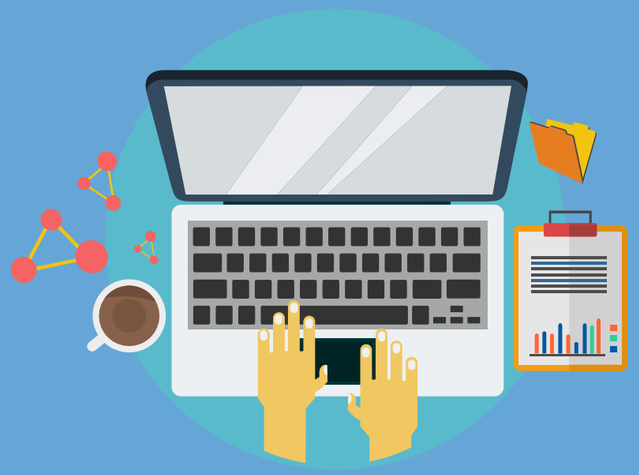
Coding! *So enjoy the weekend, because the first sprint starts Monday morning.*

Chapter 3

Days 16-95: Coding! (And Testing!) (Yay!)

Things start to get real here. This is the fun part, where the team brings the design to life and starts to churn out actual results that the customers can get their hands on so they can provide feedback.

The success of this phase depends not only in the team's coding and testing skills, but also on the team's experience working together as a team: Each member knows the others' style, skills, and limitations. This can make the difference between good, solid software and a half-completed, buggy mess. It's the part of the process on which AndPlus has built its reputation, and it's the reason customers come to us for their custom software needs.



AndPlus

The Development Sprints

This phase is typically divided into 8 two-week agile sprints. The basic sprint cycle follows this pattern:

- The team reviews the backlog items for the sprint to ensure everyone understands the purpose of each.
- The Development Team self-organizes and begins to work on Product Backlog Items.
- As items are completed, Developers will pick up the next highest-priority backlog in the Sprint.
- In the daily Scrum meeting, team members briefly discuss what they completed yesterday, what they will work on today, and what obstacles they are encountering. The Scrum Master and the Product Owner help overcome the obstacles.
- As developers complete their tasks and have parts of the software that are ready for testing, the QA team start testing the functionality and report any bugs. Designers will also review the User Experience.
- Finally, a Technical Architect will review the code to ensure that it is maintainable, well-written, and documented.
- At the end of each week, a "Grooming" meeting is held with the team as a reality check to clarify upcoming tasks and keep the development moving and on track.

At the End of Each Sprint

According to the principles of agile development, at the end of each sprint, a bug-free, “releasable” product is built. “Releasable” doesn’t mean “finished,” it merely means that it is in a state where the code can be compiled and the executable software runs without error. This is tricky with software that is not “complete” in the sense that it has all the required functionality. Meeting this principle requires careful planning of each sprint and close attention to the progress that is being made.

Using the built product, a Sprint Review is held, with the customer if possible, to demonstrate the progress so far, receive feedback, and re-prioritize the remaining backlog items if needed.

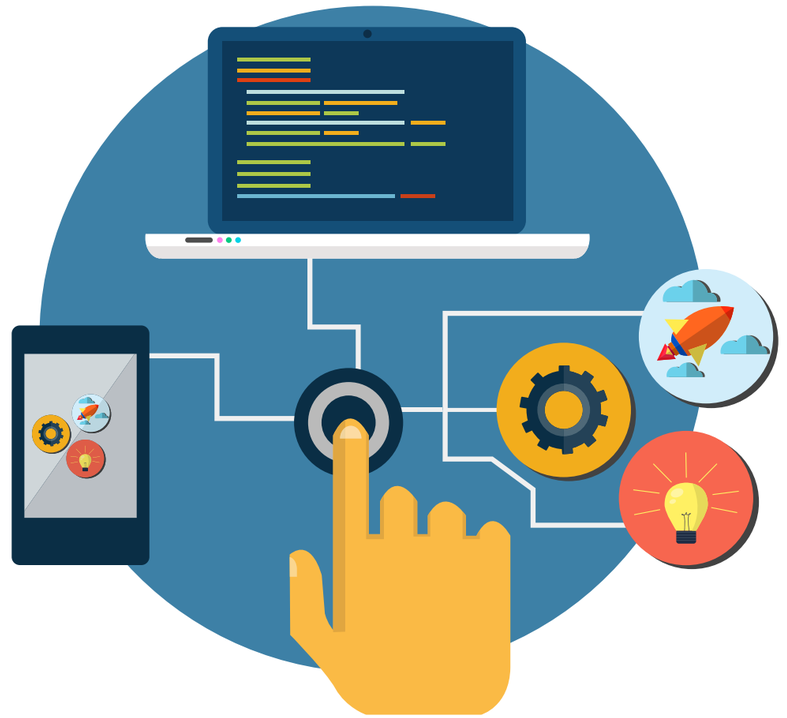
Re-prioritizing the backlog is crucial to the overall success of the project. The backlog may need to be re-prioritized for a number of reasons, including:

- Discovered work that spawns additional tasks or extends the effort required for existing tasks
- Bugs that need to be fixed
- Tasks that are completed more quickly than expected, allowing lower-priority tasks to be addressed
- Changed priorities from the customer

At the End of This Phase

By the end of the last sprint, the team should have a solid MVP release candidate that incorporates all of the “must-have” items and as many of the lower-priority items as possible. The product will have been thoroughly tested and should have no bugs.

All that remains to do is a few mop-up activities in the final five days of the project.

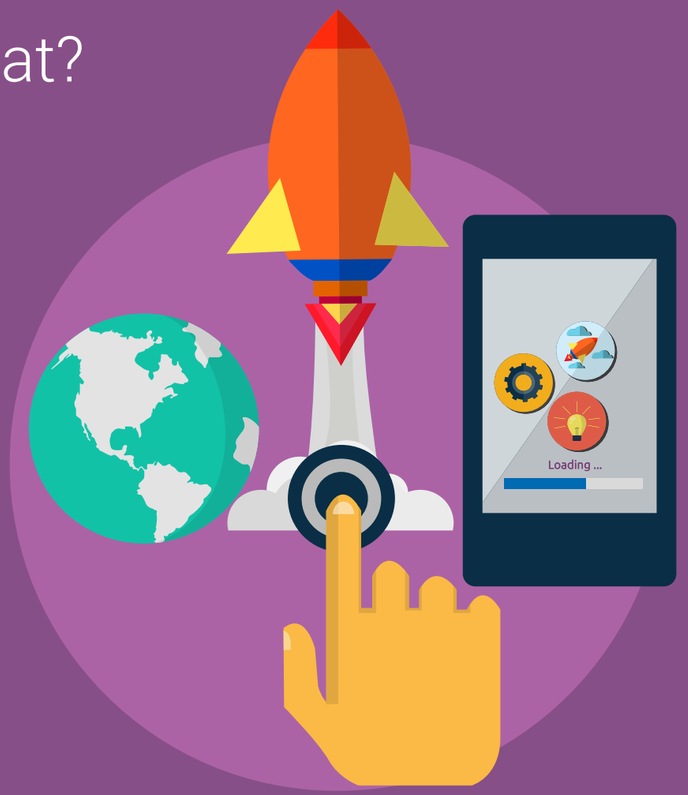


Chapter 4

Days 95-100: Final Touches and Launch

It's the last week of the project! By this time, all of the major development is done, the product has been thoroughly tested on all target platforms, any needed documentation is complete, and there are no "show-stopper" bugs.

Now what?



AndPlus

Tying Up the Loose Ends

This is the time for tying up the loose ends, putting the final polish on the product, and getting it ready for launch. Activities this week include:

- Checking the visual appearance of the user interface, ensuring all the graphical elements are aligned properly, all the words are spelled and punctuated correctly (in all applicable languages, if the product supports localization), the look and feel are consistent, and nothing looks out of place
- Identifying and fixing any last-minute problems that are small-scale and easily solved
- Getting final sign-off from the customer
- For mobile apps, starting the submission process for making the product available in the app stores
- For web apps, getting ready to promote the product to the production environment
- For traditional desktop apps, preparing and testing the installation package

After Day 100: Post-Launch Monitoring

After launch, the development team's work is pretty much finished for now, but they should be available to help with any major problems that crop up immediately after launch.

Things to be on the lookout for post-launch include:

- Bugs that were overlooked in testing
- Problems downloading or installing the software on various platforms
- User suggestions and requests for new features and capabilities
- General feedback

For any major problems, a "hotfix" version can be prepared and deployed. Minor problems and suggestions for improvement are incorporated into the backlog for the next regular release.

Conclusion

After the successful launch of the MVP, it's time to review what went well and what could have gone better. Perhaps not everything went perfectly, but the lessons learned are an invaluable part of the continuing evolution of the development team. Then, it's on to the next project. If the team has done its job well, the next project could be Version 2 of the software, which builds more value-added features and functionality on the MVP foundation.

The 100-Day MVP method is not something that a new team can implement and perfect overnight. It takes trust among the team members; a willingness to learn from each-other's successes and mistakes; and practice, practice, practice. At AndPlus, we take pride in knowing our teams are well-versed in the method and consistently delight our clients with top-quality software solutions. It's what sets us apart from the pack.

Visit us online at AndPlus.com to learn more.