TIVIX

HOW TO BUILD BETTER SOFTWARE WITH A PRODUCT DISCOVERY PROCESS

Contents

How to Build Better Software With a Product Discovery Process

Are you planning to build a digital product? Whether it's an internal app or a product that you're ultimately planning to bring to market for consumer use, it's essential to have a comprehensive picture of what you need in place before you start your buildout.

That means starting out with a strategic phase dedicated to discovery, in which all stakeholders work together to clarify the ultimate goals of the product, and align on a vision for building a product that meets all of those goals. Embarking on a development project without a detailed discovery phase is a fool's game—you'll end up with a product that tries so hard to be everything to everyone that it ends up pleasing no one.

When working with an outside development agency, the discovery phase is also crucial for helping you to gain a better sense of the total budget required to execute on your vision. In fact, 70% of development agencies have made this a non-negotiable part of their process. In this phase, you can discuss many of your ideal features, and get a better sense from the agency of how many of those features will fit into your target budget and what kind of roadmap makes sense to bring your minimum viable product (MVP) to market, with future releases from there.

At Tivix, we always begin our client engagements with an in-depth discovery stage. Here's a look at how the process works.

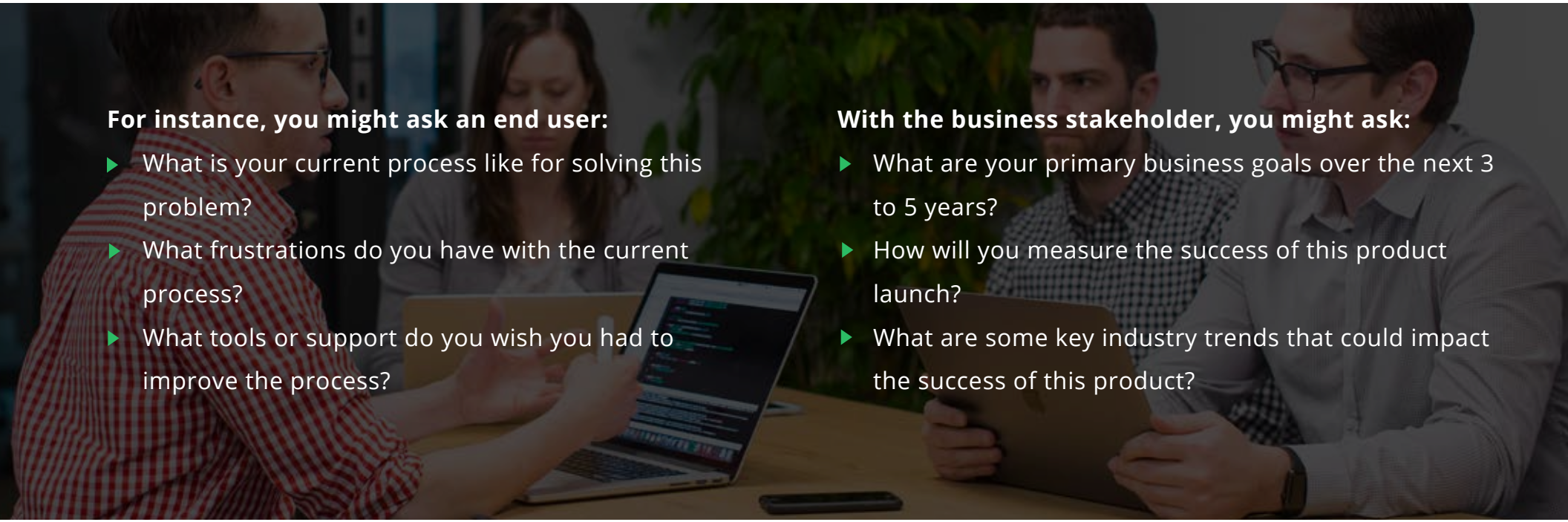
Commit to a Vision

Before you can move forward, you need to know what the goalposts are. That means it's important for all stakeholders to agree on a vision of what problem your solution is designed to solve, and how it will solve it in a way that's different or better than other available solutions.

Start by asking questions—lots of them. And not just to the people who are writing your checks. You want to know not only what the key business stakeholders are looking for, but also what the *end users* of the product are looking for. After all, the product will live or die based on their feedback, so it's important to get into their minds to make sure you're getting things right.

The Interview Process

Begin by asking everyone—business stakeholders and end users alike—a series of questions to get at the heart of what the problem they're facing is, and what ideal features a solution to fix it might include.



For instance, you might ask an end user:

- What is your current process like for solving this problem?
- What frustrations do you have with the current process?
- What tools or support do you wish you had to improve the process?

With the business stakeholder, you might ask:

- What are your primary business goals over the next 3 to 5 years?
- How will you measure the success of this product launch?
- What are some key industry trends that could impact the success of this product?

For more on the Q&A process and what to look for in your interviewees' answers, take a look at our recent blog post on product discovery questionnaires.

User Research

Interviewing end users is important, but you need more than their words to know how they actually feel about something. They might summarize their perspective on an issue during a Q&A, but you're missing the valuable context that comes from seeing them engage with an issue or challenge in a real-world situation.

In order to get a better sense of who your users really are and what concerns them, focus on in-the-field research. If you're building an app to help warehouse workers more efficiently accept and inspect deliveries, for instance, then spend a day or two with them in the field, watching how they currently navigate the process and talking with them about any challenges they face.

Group research settings can also be helpful: Often, a group of people who engage in a particular activity will be able to bounce insights off one another, and come to new realizations together—or even get into arguments that serve to illustrate opposing viewpoints about what a good solution would involve. In order to do this, it can sometimes be worth investing in setting up a focus group. A focus group can be helpful both in the initial research stage as well as when you have a prototype or early beta product to test out and solicit feedback on. You can gather insights around their existing pain points, their daily habits, and the usability (and lovability) of your proposed solutions.

Competitor Research

Alongside your user research process, you should also spend time on competitive research to understand what other solutions are currently available and how your product can differentiate. The goals may be different, depending on whether you're planning to monetize your product or to use it internally to fill a gap in your existing process.

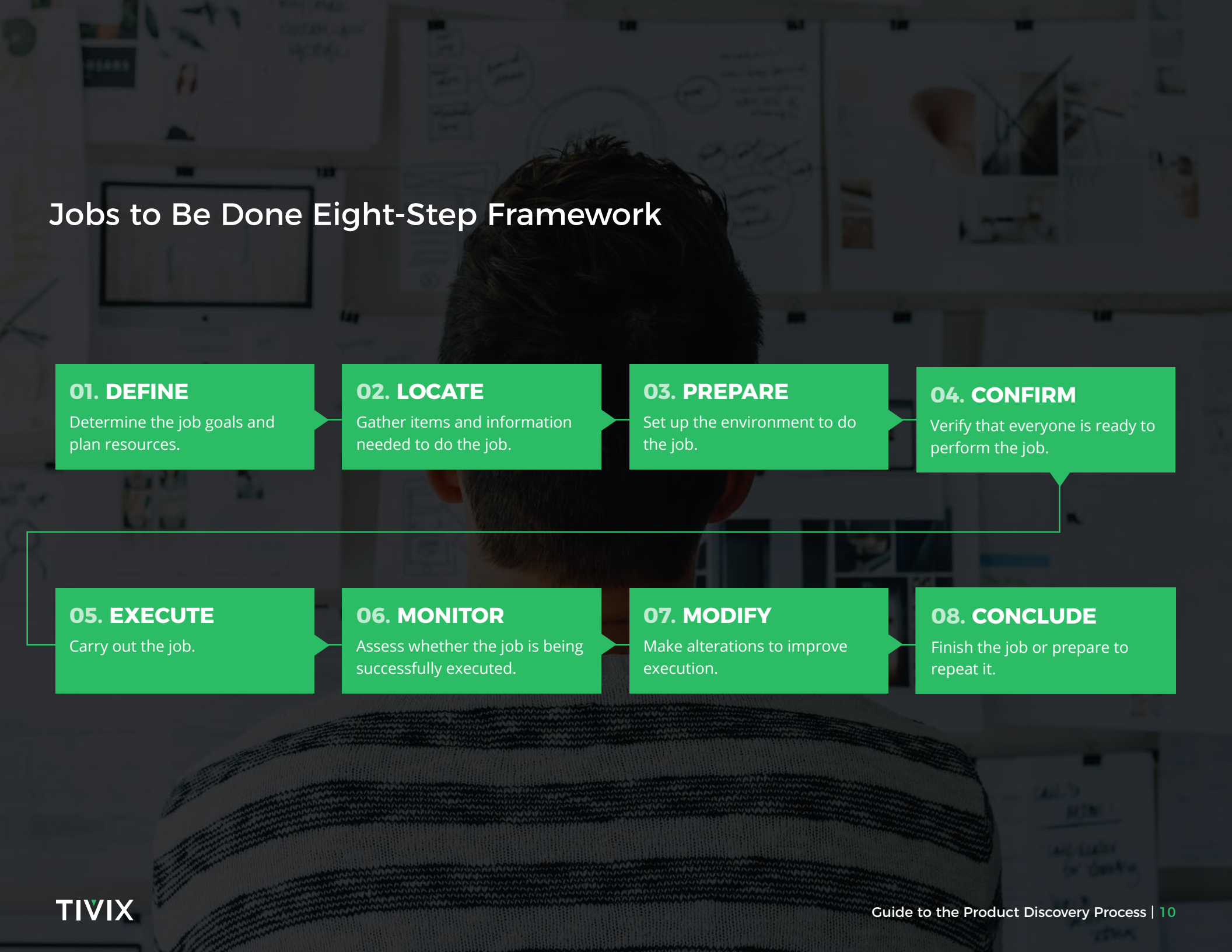
If you're planning a commercial launch, you'll want to ensure your product provides a better user experience, better features, and/or a better price point than those options already on the market. If you're planning to use the product internally, you'll want to evaluate existing off-the-shelf options to see if there's a viable solution out there already with the features you need—and if so, does the math work out better to license that solution, or to build your own version?

Jobs to Be Done Framework

Typically, we use the Jobs to Be Done framework to identify areas to innovate, and find under-served areas of the market or underestimated product features. This framework, defined by Strategyn Founder Tony Ulwick, identifies a series of eight steps that are critical in any job.

In this context, we use the term "jobs" based on the understanding that, as Clayton Christensen puts it in Competing Against Luck, "We all have jobs we need to do that arise in our day-to-day lives, and when we do, we hire products or services to get these jobs done." The Jobs to Be Done framework, then, provides a structure for determining what problems a customer or user will face in the course of accomplishing an overarching goal, and maps out the steps for solving those problems.

The Jobs to Be Done framework is used to capture all the customer's needs, and to systematically identify opportunities for growth. Working within this framework, development teams are able to build better user stories and personas, and gain a genuine understanding of the goals for the job on the customer's side. Collaborating in a work session with both the customer stakeholders and the development team, you can document the "jobs" in detail, and the steps involved in undertaking them, to build out key priorities to help your team meet goals. In order to ensure that you're asking the right questions and addressing all relevant areas, consider using the Jobs to Be Done Canvas that Strategyn offers as a free template.



Jobs to Be Done Eight-Step Framework

01. DEFINE
Determine the job goals and plan resources.

02. LOCATE
Gather items and information needed to do the job.

03. PREPARE
Set up the environment to do the job.

04. CONFIRM
Verify that everyone is ready to perform the job.

05. EXECUTE
Carry out the job.

06. MONITOR
Assess whether the job is being successfully executed.

07. MODIFY
Make alterations to improve execution.

08. CONCLUDE
Finish the job or prepare to repeat it.

Explore Ideas for Quick Validation

Once you've identified the key goals of the solution, it's time to move into ideation—followed by rapid iteration to validate your ideas.

First, hold a kickoff workshop to quickly generate ideas and move them quickly into prototyping stage. Your workshop stage may be as short as 2 to 3 days, or expand to more than a week. The goal here is to build out a concept for a product that solves the problems that you've identified, and then to rapidly move it into prototype stage for real-world feedback.

Prototypes can come in many formats. You may start with a “low fidelity” prototype, which uses rough sketches or wireframes to showcase the user flow through the product, but doesn’t showcase the design vision behind it. Or, you might build out a “high fidelity” prototype, using graphic animation tools. In this case, the prototype would look very similar to the app that you’re developing, but without any coding—the progression from one stage to the next might be illustrated through a recorded user flow.

Depending on the level of sophistication of your prototype, you’ll be requesting different types of feedback from your users. With low-fidelity prototypes, you’ll want to focus on whether the workflow makes sense, while high-fidelity prototypes can focus more on smaller design-related details, and how intuitive the product is for the user. As you continue testing prototypes, you’ll want to ensure that you offer opportunities to provide feedback on all potential use cases (or “user stories”) in which an individual may use the product. It often takes five or six rounds of prototyping to get to a point where you’re ready to begin product development, so take your time on this stage to ensure that you’ve clearly mapped out your needs. Learn more about the prototyping process in our new ebook.

Map Out the Process to Turn Your Prototype into a Product

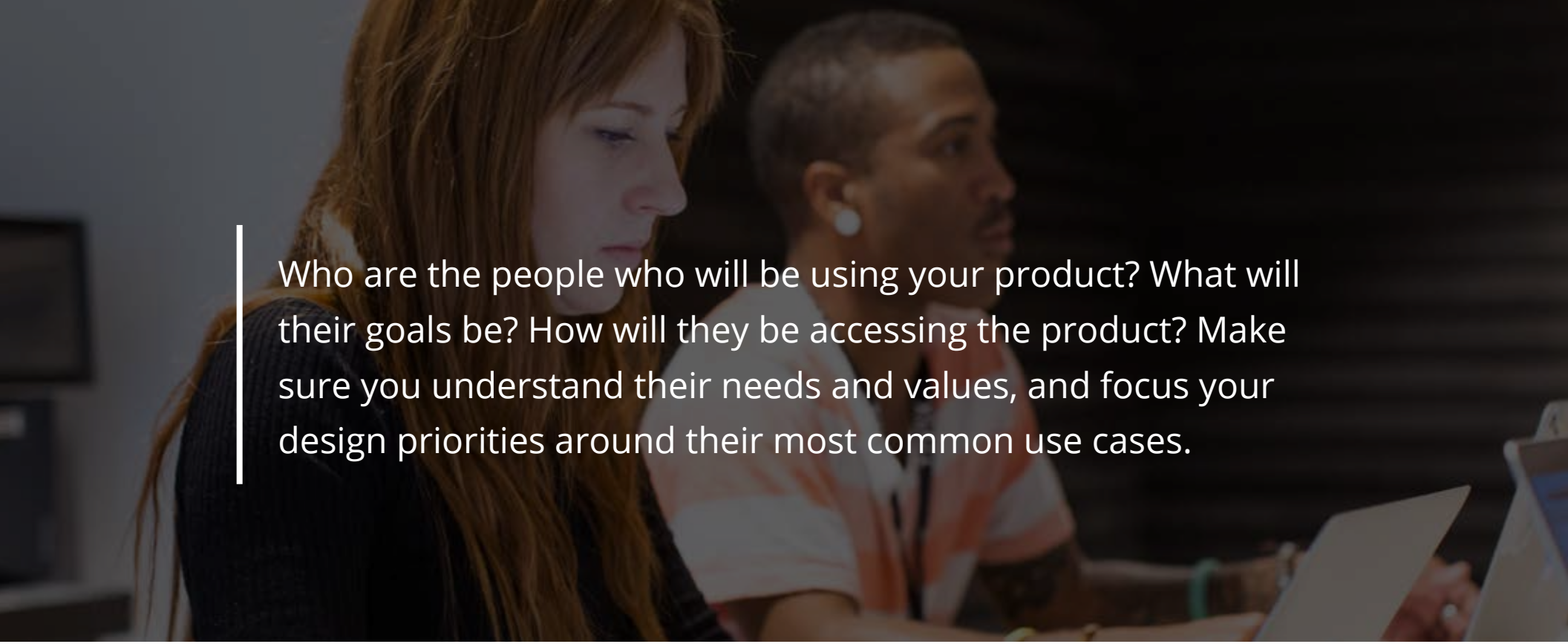
Once you've worked out the kinks in your prototyping process, it's time to begin the process of turning your concept into a fully-coded product. Creating a roadmap for this process can be complex, but is necessary to ensure that you've identified all of the project goals and aligned them to the business' priorities.

Build User Stories and Personas

Clearly delineate the “User Stories” that are central to the product. User Stories describe a product feature told from the perspective of a person who wants to use that feature. They typically follow this template:
As a < type of user >, I want < some goal > so that < some reason >.

For example, a customer of a pizza shop might write:
I want to be able to duplicate my past orders without inputting information so that I can save time and ensure accuracy when ordering.

This provides insight into a product feature that the product owner (i.e., the pizza chain) will seek to include in its application.



> Who are the people who will be using your product? What will their goals be? How will they be accessing the product? Make sure you understand their needs and values, and focus your design priorities around their most common use cases.

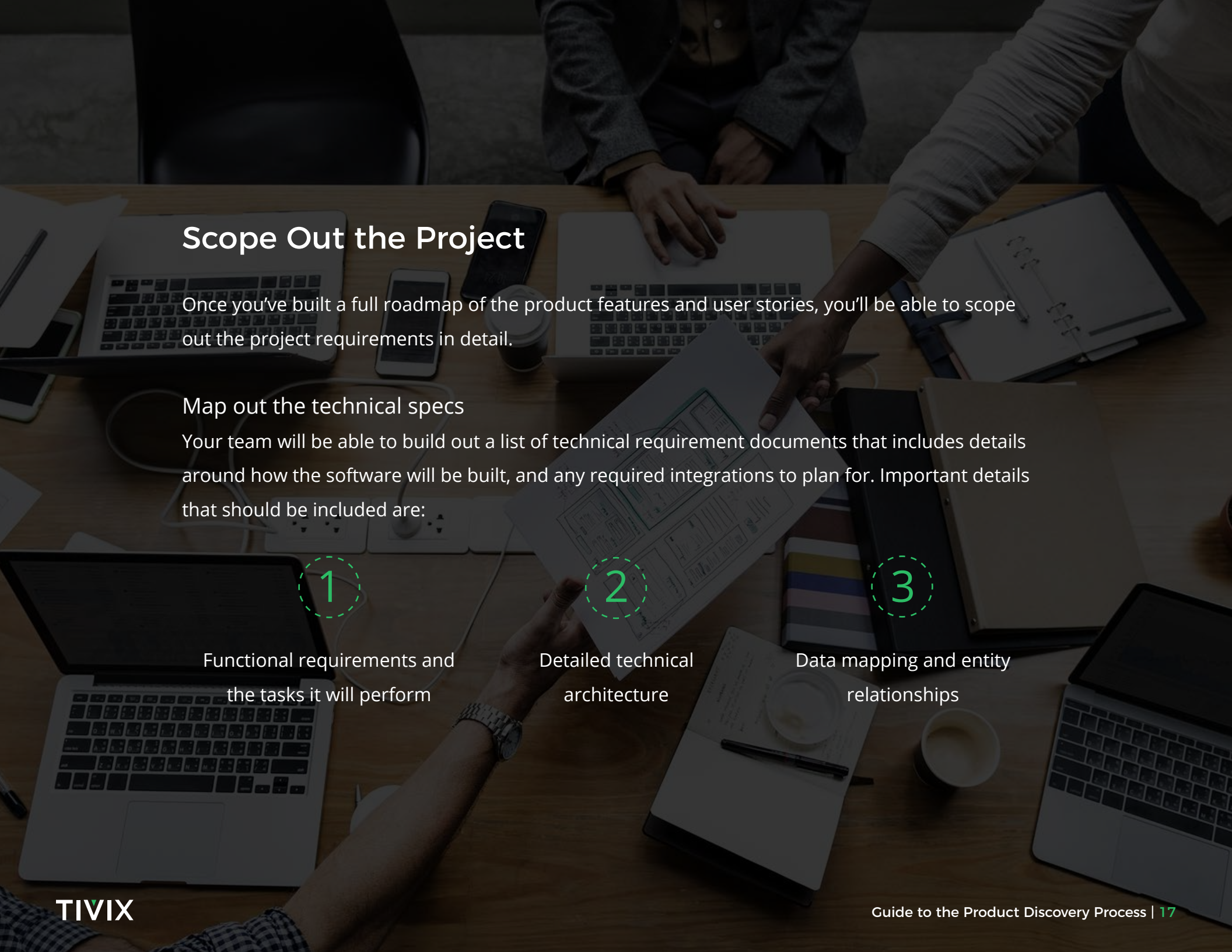
When building an app for a bank, for instance, most people will want to check their account balances and make remote deposits, so ensure that the functionality provides easy access to take these steps directly from the main home screen. More complex initiatives, such an applying for a mortgage, will likely require a more personal touch, so you'll want to provide tools to connect customers with loan officers to continue the conversation, rather than creating complex workflows and links to paperwork within your app.

As you build your product map, create user stories for each type of defined user, focusing on the types of features that will be critical to their use cases of the product. You can later prioritize the roll-out of these features based on the business' key goals and which use cases are the most popular or the most valuable to your bottom line.

Cover All the Edge Cases

While your team's main focus may be on the most frequent use cases or types of users, it is important that you don't ignore the less common situations—particularly when it comes to a "worst-case scenario" outlook. Ensure that if your app relies on an integration with another technology platform, that you provide opportunities for manual inputs if the integration fails. If you offer search functionality, make sure that you offer multiple ways to search, such as filtering out key terms that your user doesn't want to see.

But don't let your need to support edge cases complicate your design—following Google Search's model, you can spotlight the most common use case on your home screen, while enabling those who need variations to navigate to those options as needed (as Google does by offering clickable "advanced search" settings on a secondary page).



Scope Out the Project

Once you've built a full roadmap of the product features and user stories, you'll be able to scope out the project requirements in detail.

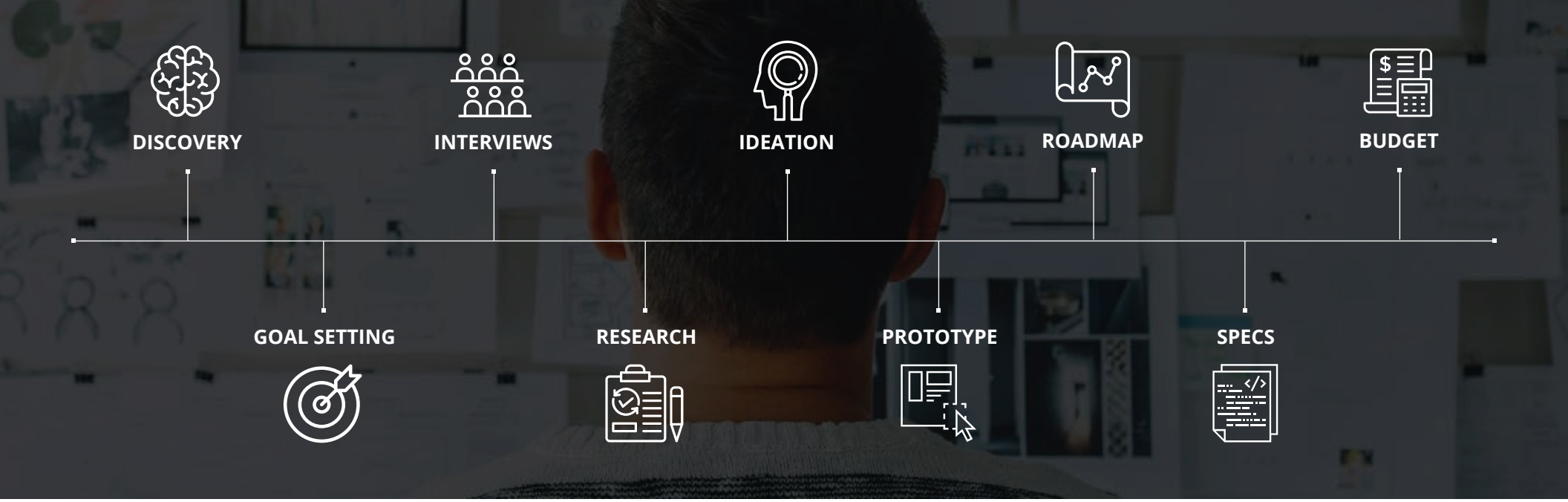
Map out the technical specs

Your team will be able to build out a list of technical requirement documents that includes details around how the software will be built, and any required integrations to plan for. Important details that should be included are:

1. Functional requirements and the tasks it will perform
2. Detailed technical architecture
3. Data mapping and entity relationships

Set Up Budget Estimates

When following an Agile methodology, your team will likely work based on a series of sprints (typically two weeks in length). With the complete product map in hand, your engineering team should now be able to break down the list of features into a sprint series, determining how many features they will be able to build in each sprint. This is still an estimate, as there are occasions where a feature doesn't work as planned and rework must be completed, but by mapping the features to a series of sprints, the business stakeholders should be able to get a good sense of the total budget overall.



Create a Timeline for Product Development

Working with the stakeholders, your team can now prioritize features for development based on your overall budget and your priorities. In most cases, it makes sense to launch with your minimum viable product (MVP), and roll out additional features in later releases. Create your product timeline with an initial target launch date for the core product offering, and additional staged release dates that incorporate new features according to your priorities for increasing functionality.

While many companies are eager to bring a new product to market and beat their competitors to the punch, it's critical to spend time diving into the product discovery process—without taking meaningful time to research your users, ask important questions, and scope out an accurate prioritization of features, you may end up with a product that doesn't perform well or meet the true needs of your users. By spending time on a comprehensive discovery and product roll-out, your business can build a product that's destined for a successful staged release, with plenty of opportunities to solicit feedback and improve the product along the way.

TIVIX

HOW TO BUILD BETTER SOFTWARE WITH A PRODUCT DISCOVERY PROCESS

Tivix is a digital product development firm headquartered in San Francisco, with additional offices in New York City, Europe (London & Wrocław), and Portland. Our focus is the agile development of web, cloud, and mobile applications - and helping organizations create and sustain digital innovation.

Need help with Product Discovery?

At Tivix we help businesses large and small build better software using design thinking principles. To learn how we can help you find innovative solutions to complex software **problems** **click here to get in touch.**

---

connect@tivix.com | **tivix.com**

Tivix, Inc. 2845 California Street, San Francisco, CA 94115