/thoughtworks

Design as a Team Guidebook

Christopher Taylor Edwards
and Valerie Roske

Introduction

We believe cross-functional collaboration accelerates delivery of better outcomes for our users. However, it's not always obvious what exactly that might look like. What practices are necessary to the team's success?

You should read this if:

- You are a designer trying to define what design agility looks like, from strategy to execution
- You are a product manager who needs to articulate the difference between your role and that of a designer
- You are a delivery team member (developer / quality analyst / project manager / etc) wondering, "What even is design? Why or how should I participate?"
- You are a team lead trying to figure out how to build the right thing and build the thing right
- You are looking for some specific examples of how to break down silos in product development and quickly deliver new capabilities for your users

Getting Started

Through this guide, we'll provide specific examples of what cross-functional collaboration can look like, while sharing some of the values and principles that motivated us.

This **isn't** an exhaustive list of "best practices"; instead, we hope this guide can serve as inspiration for you and your team, and broaden your view of what design is along the way.

Feel free to pick and choose the practices that work for you, remix them, or invent your own!

Starter recipes

- Not sure which practices to try first? Our starter recipes highlight how designing as a team can help you work through real challenges you might be facing right now.

À la carte team practices

- If none of the starter recipes resonate with your context, you can craft your own! We've put together more than 20 different practices in this guide, formatted to help you make choices that will suit the social and technical needs of your team.

Starter Recipes

How might we quickly demonstrate the value of cross-functional collaboration and bring the team along on the journey?

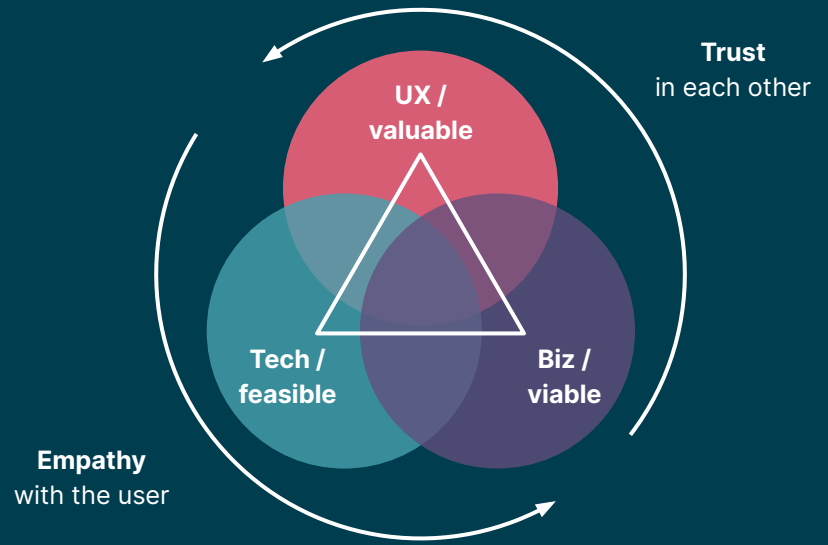
/thoughtworks

Minimum Viable Practices

We recognize that forming new group habits requires deliberate effort, so we always try to think of any new practice as a “Minimum Viable Practice.” A Minimum Viable Practice is the simplest thing we can do to get the learning we need, with the smallest possible group of participants required to derive that learning. We use real-time feedback and retrospectives in order to improve our practices and to ensure their purposes are still valid.

At the heart of each practice are the **Three Lenses of Innovation:**

- What are the needs we’re trying to meet? **(valuableness)**
- How do we deliver? **(feasibility)**
- Can we sustain our momentum? **(viability)**



The Three Lenses of Innovation: our goal is to create something that is valuable (desirable), feasible, and viable. We do this by espousing empathy and trust within our team and with the communities we serve.

Starter Recipe

Scenario 1: Large enterprise with little to no design capability

Perhaps you are “starting from zero” in your organization: intentional design is an afterthought, there are no formal or informal design communities, and only a few people on your team understand the value of organizing around different disciplines to solve problems.

How might we start what feels like an impossible journey?

Desired outcomes:

Defined team values; foundational ways of working; an initial, shared understanding of the user needs and the problems you want to solve

How do we measure success?

Everyone on the team can explain the business and user goals regardless of their role, and knows how their work contributes to these goals

Starter Recipe

Scenario 1: Large enterprise with little to no design capability

Your plan: Prioritize strategic investments over quick wins

Values, principles, practices, and champions (page 18)

Why: This is a great level-setting exercise that allows everyone to express what is important to them in a work environment. Having this conversation regularly allows you to intentionally shape the culture of your team and make better decisions based on shared values.

Definitions of ready and done (page 22)

Why: This is a quick way to set expectations on what is needed from everyone on the team for work to continue to flow and be completed.

Designing lean (page 28)

Why: The best way to test and learn quickly is through designing lightweight and lean prototypes.

Centering the user journey map (page 41)

Why: A user journey map is an artifact of systems thinking; it creates a shared language for how everyone's work connects to the user's goals.

Cross-functional problem exploration (page 34)

Why: Your new team needs to agree on the right thing to build. Cross-functional problem space exploration aligns the team toward a common vision upfront.

Starter Recipe

Scenario 2: Designers are left out of key product decisions

You are at a small, growing company or even a large enterprise where Design as a function is seen as subordinate to Product; as an implementation detail rather than as a strategic partner.

How might we make ourselves visible and get a seat at the table?

Desired outcomes:

Greater visibility into design and its impact on business; improved ways of working; improved prioritization of work

How do we measure success?

The team has a greater understanding of what design work is and can participate in it, regardless of role

Starter Recipe

Scenario 2: Designers are left out of key product decisions

Your plan: Prioritize quick wins over strategic investments

Design work on the story wall (page 27)

Why: This is a quick win that gets everyone aware of and talking about design work and its relationship to development work.

Technical analysis round robin (page 25)

Why: Effective communication of software requirements demands tight collaboration between Product, Design, and Tech. This analysis activity gets everyone talking to each other as equals, aligning on the purpose, value, and feasibility of user stories.

Leadership trifecta (page 24)

Why: Product, Tech, and Design should be equally involved in leadership in order to maximize innovation and ensure we are accountable for delivering great customer experiences.

Open design pin-ups (page 42)

Why: Design pin-ups create opportunities for everyone to talk about what designing as a team means, by facilitating an open space for critique, in-depth reviews of research findings, and general discussion. When used along with Design Work on the Story Wall, the team gains a deeper understanding of both the problem space and the solution space.

Paired solution exploration (page 37)

Why: This is an extremely flexible practice, and one that is key to breaking down silos. The fastest way to get feedback from someone else is to simply ask them and involve them in your process.

Roles and responsibilities (page 23)

Why: Whenever role expectations are unclear or needs are going unmet, it's important to be upfront with each other and recognize how to turn these misalignments into opportunities for improved collaboration.

Starter Recipe

Scenario 3: Daily work primarily revolves around engineering

Your team is composed solely of engineers, with no designers or product strategists embedded in daily work. The engineers have only a very basic understanding of the value of their work to the business and to the end-users. People outside the team are seen as a monolithic entity and never seem to understand how engineering constraints impact their goals.

How might we forge the relationships we need to fill in our gaps in knowledge and effectively communicate risks and issues?

Desired outcomes:

The team and the business have increased empathy with the end-users; business stakeholders are visible to the team; and a shared language to describe the product in a way that can be understood by everyone

How do we measure success?

Improved user engagement metrics; improvements to feedback culture and learning loops

Starter Recipe

Scenario 3: Daily work primarily revolves around engineering

Your plan: Prioritizes strategic investments over quick wins

Business stakeholder collaboration (page 21)

Why: Establishing mutual trust with stakeholders is critical; you need others to understand and be involved with the team's processes just as much as you need to understand their goals and motivations.

Paired solution exploration (page 37)

Why: This is an extremely flexible practice, and one that is key to breaking down silos. The fastest way to get feedback from someone else is to simply ask them and involve them in your process.

Centering the user journey map (page 41)

Why: A user journey map is an artifact of systems thinking; it creates a shared language for how everyone's work connects to the user's goals.

Testing the observable UI (page 43)

Why: Adopting a testing strategy that centers the user instead of the engineer puts development effort into context and highlights its value.

Open user research sessions (page 31)

Why: One of the most effective and eye-opening ways to build empathy with end-users is to observe them using your product.

Practices for Designing as a Team

If a starter recipe doesn’t resonate with your context, feel free to try any of the following à la carte practices, remix them, or invent your own!

/thoughtworks

Understanding the Team

There are a few factors driving our team's ability to sustain new practices: **team maturity, values** at play, **effort** required, and model of **participation**. Understanding these factors helps us to self-regulate and ensure we can effectively manage whatever change we introduce.

Team maturity

We find Tuckman's stages of group development help us to examine:
How are we doing as a team now? What do we ultimately hope to achieve? How do we break down our goals into manageable, attainable milestones?

- **Forming:** still getting to know each other, defining rituals, routines, and ways of working
- **Storming:** experiencing some growing pains, churn, or conflicts
- **Norming:** team performance is relatively predictable and morale is healthy and stable
- **Performing:** able to work at a more strategic level instead of being purely focused on tactics

Values

We want to ensure that whenever we try something new that it attempts to address a real need for our team. Having a set of values helps us identify our gaps, and realize quickly when a practice is no longer working for us. Each practice in this guide is designed to address one or more of these values:

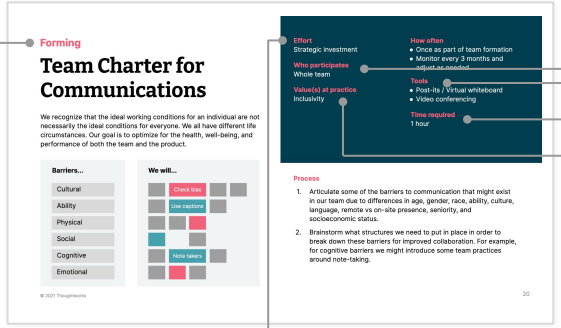
- **Shared purpose:** Creating alignment around value delivered for both the business and the user
- **Quality:** Delivering an experience we are confident in
- **Inclusivity:** Equal opportunity participation of team members, stakeholders, and users
- **Fast feedback:** Working closely together in order to learn from each other - is it valuable, viable, and feasible?
- **Accountability:** Do we live up to our values?
  Are we continuously improving?

Understanding the Practices

There are a few factors driving our team's ability to sustain new practices: **team maturity, values** at play, **effort** required, and model of **participation**. Understanding these factors helps us to self-regulate and ensure we can effectively manage whatever change we introduce.

Team maturity

We find Tuckman's stages of group development help us to examine: How are we doing as a team now? What do we ultimately hope to achieve? How do we break down our goals into manageable, attainable milestones?



Effort

A **strategic investment** requires more buy in from participants or more thought and consideration as to how to actually implement. A **quick win** is something teams can deploy very quickly and get immediate feedback.

A quick win isn't necessarily less valuable than a strategic investment, it's simply easier to just do. On the flip side, strategic investments aren't necessarily super resource-intensive. Even in time-restricted contexts where quick learning loops are essential, it's still important (and possible!) to make strategic investments in order to balance short-term and long-term needs of the team and the product.

Who participates

Some activities involve the **whole team**, while others are more focused and can be done in pairs or small groups: for example, a designer and developer.

You might be familiar with the idea of pair-programming. While we do a lot of that at ThoughtWorks, we've also become accustomed to the idea of "**pairing**" as inclusive of other roles and activities besides just programming. When we say "pairing", we mean at least two people collaborating in real time on the same activity, and in the context of designing as a team, those people generally perform different roles.

Tools

We list tools that could be useful for the activity

How often / time required

We put an approximate time to help you understand how long it takes to run the activity

Value(s) at practice

What needs are we trying to address for our team?
(Shared Purpose • Quality • Inclusivity • Accountability • Fast Feedback)

The Practices: Index Grid

Team maturity	Practice name	Effort	Values	Participation	Page
Forming	Values, Principles, Practices, and Champions	Strategic investment	Shared Purpose, Inclusivity, Accountability	Whole team	18
	Team Charter for Communications	Strategic investment	Inclusivity	Whole team	20
	Business Stakeholder Collaboration	Strategic investment	Fast feedback, Quality, Accountability	Pairing: Led by design and product, open to all	21
	Definitions of Ready and Done	Quick win	Accountability, Inclusivity	Whole team	22
Storming	Roles and Responsibilities	Quick win	Fast feedback, Accountability	Whole team	23
	The Leadership Trifecta	Quick win	Fast feedback, Accountability, Inclusivity	Pairing: design, product, tech	24
	Technical Analysis Round Robin	Quick win	Shared Purpose, Fast feedback, Accountability	Whole team	25
	Design Work on the Story Wall	Quick win	Shared purpose, Fast feedback, Accountability	Pairing: designers and product strategists	27
	Designing Lean	Quick win	Fast feedback, Accountability	Pairing: design-led with product and engineering	28
	Co-creating Design Principles	Quick win	Shared purpose, Quality, Accountability	Whole team: facilitated by designers	29

The Practices: Index Grid

Team maturity	Practice name	Effort	Values	Participation	Page
Norming	Open User Research Sessions	Strategic investment	Fast feedback, Quality, Accountability	Whole team: open to everyone	31
	Showcases	Quick win	Inclusivity, Accountability	Whole team: open to everyone	32
	Story Kick-Offs and Desk Checks	Quick win	Shared purpose, Fast feedback, Quality	Pairing: developers, quality analysts, product strategists, designers	33
	Cross-functional Problem Exploration	Strategic investment	Shared purpose, Inclusivity, Fast feedback	Whole team: design-led, with analysts, developers, users, and other stakeholders	34
	Paired Solution Exploration	Quick win	Inclusivity, Fast feedback	Pairing: any roles	37
	Reusable Interaction Patterns	Strategic investment	Shared purpose, Quality, Fast feedback	Pairing: across the team, in pairs and small groups	40
	Centering the User Journey Map	Strategic investment	Shared purpose	Pairing: design, product, tech	41
	Open Design Pin-ups	Quick win	Shared purpose, Fast feedback	Whole team: open invitation	42
	Testing the Observable UI	Strategic investment	Quality	Pairing: developers and quality analysts	43
	Design Systems and Reusable Components	Strategic investment	Shared purpose, Fast feedback, Quality	Whole team	45

The Practices: Index Grid

Team maturity	Practice name	Effort	Values	Participation	Page
Performing	Design and Tech Debt Audit	Quick win	Quality, Inclusivity	Pairing: designers, quality analysts, and developers	47
	Technical and Design Decision Log	Quick win	Quality, Accountability	Whole team	48
	Measuring Impact	Strategic investment	Shared purpose, Quality, Inclusivity, Fast feedback, Accountability	Whole team	49

Forming

Values, Principles, Practices, and Champions

Aligning on team values and sociotechnical needs to inform our ways of working

By examining our values and needs regularly, and questioning how our practices are helping us make progress on our goals, we believe we will be more effective at our jobs, deliver meaningful products for people, and become better humans.

Effort
Strategic investment

Who participates
Whole team

Value(s) at practice
- Shared Purpose
- Inclusivity
- Accountability

How often
- Revisit principles and values once every three months
- Revisit practices at recurring retrospectives

Tools
- Post-its / Virtual whiteboard
- Video conferencing

Time required
2-3 hours

Process

1. Individuals brainstorm the top 5 things they value in a work environment. For example: “collaboration”.
2. Everyone has the opportunity to share why those values are meaningful to them.
3. Everyone votes on the top 5 values that will be the foundation for the whole group.

(continued on next page)

Forming

Values, Principles, Practices, and Champions

Process (continued from previous page)

4. Split into smaller groups and assign one value per group. For each value, identify 1-3 key principles that drive the behaviors you want to see on the team. An example for collaboration might be “Anyone can work on anything”.
5. Regroup to share the principles and discuss how they reflect the values. This discussion is important, so don’t rush it.
6. The same small groups identify 1-2 practices the team is currently doing that demonstrate those principles in action, and / or 1-2 potential new practices to try.
7. Regroup to discuss who will champion each value to keep the team accountable. What does it mean to be a champion? Will you rotate them? How are they selected?

Effort
Strategic investment

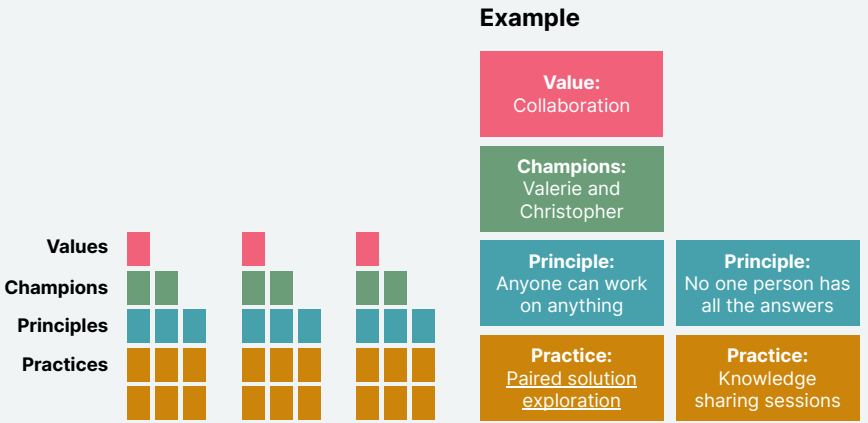
Who participates
Whole team

Value(s) at practice
- Shared Purpose
- Inclusivity
- Accountability

How often
- Revisit principles and values once every three months
- Revisit practices at recurring retrospectives

Tools
- Post-its / Virtual whiteboard
- Video conferencing

Time required
2-3 hours



Forming

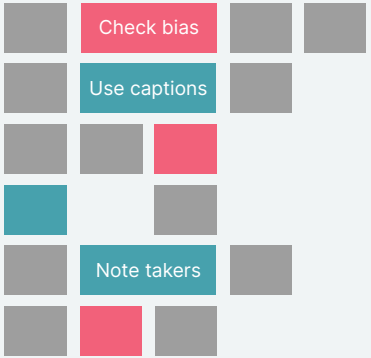
Team Charter for Communications

We recognize that the ideal working conditions for an individual are not necessarily the ideal conditions for everyone. We all have different life circumstances. Our goal is to optimize for the health, well-being, and performance of both the team and the product.

Barriers...

- Cultural
- Ability
- Physical
- Social
- Cognitive
- Emotional

We will...



Effort
Strategic investment

Who participates
Whole team

Value(s) at practice
Inclusivity

How often
- Once as part of team formation
- Monitor every 3 months and adjust as needed

Tools
- Post-its / Virtual whiteboard
- Video conferencing

Time required
1 hour

Process

1. Articulate some of the barriers to communication that might exist in our team due to differences in age, gender, race, ability, culture, language, remote vs on-site presence, seniority, and socioeconomic status.
2. Brainstorm what structures we need to put in place in order to break down these barriers for improved collaboration. For example, for cognitive barriers we might introduce some team practices around note-taking.

Forming

Business Stakeholder Collaboration

Helping those invested in the outcomes of the product understand and be involved with the team's processes

We value shared understanding and empathy with each other and the user. In order to assure that these values influence decision-making at the stakeholder level, we work to bring our stakeholders along with our continuous discovery journey.

Effort
Strategic investment

How often
Several times per week

Who participates
Pairing: Led by design and product, open to all

Value(s) at practice

- Fast feedback
- Quality
- Accountability

Tips

We put business stakeholder collaboration at the center of our work, in order to build relationships and mutual trust.

1. First, map and identify which stakeholder's trust we most need.
2. Ensure we include that stakeholder throughout our work, from the beginning, to co-create and collaboratively sketch features.
3. In addition to ceremonies like showcases and desk checks, have regular facetime with the stakeholder to update on progress, validate stories, and ask questions. Some of those practices are detailed in this playbook.

Forming

Definitions of Ready and Done

Establishing shared expectations and team ownership for delivering working software

A set of checklists or guidelines describing what it means for a user story to be ready to be picked up for development, as well as what it means for a user story to be complete. A “definition of ready” is less commonly used than a “definition of done”, but when done together, we gain additional insight into the work it takes to deliver new capabilities to our users. We may also create nuanced definitions for different types of work, such as spikes, design jams, and defects.

Effort
Quick win

Who participates
Whole team

Value(s) at practice
- Accountability
- Inclusivity

How often
- Once as team moves to storming
- Monitor and adjust as needed

Tools
- Story wall
- Story templates / checklists

Time required
1 hour

Sample definition of ready
- Mockups and interaction flows attached
- Acceptance criteria defined
- ...

Sample definition of done
- All tests are passing
- Deployed to production
- ...

Storming

Roles and Responsibilities

This exercise helps team members set expectations of needs upfront and map interaction patterns. It also can be a way to identify new pairing opportunities we wouldn't have thought about otherwise. It is an effective way to bring clarity to what everyone should be doing.

Effort
Quick win

Who participates
Whole team

Value(s) at practice
- Fast feedback
- Accountability

How often
- Once as team moves to storming
- Monitor and adjust as needed

Tools
- Post-its / Virtual whiteboard
- Video conferencing

Time required
1 hour

Process

1. Team members articulate what they offer through their role expertise, and what they need from others in order to be effective.
2. Each person takes turns sharing their needs. If someone else can satisfy that need, they raise their hand, and that pair is mapped on the board.

Storming

The Leadership Trifecta

We frequently observe design specialists are left out of leadership roles and team meetings, and that Design is frequently seen as subordinate to Product. We feel strongly that Product, Tech, and Design should be equally involved in leadership in order to maximize innovation and ensure we are accountable for delivering great customer experiences.

Design is accountable for: is the product or feature valuable for our customers?

Product is accountable for: is the product or feature viable for our business?

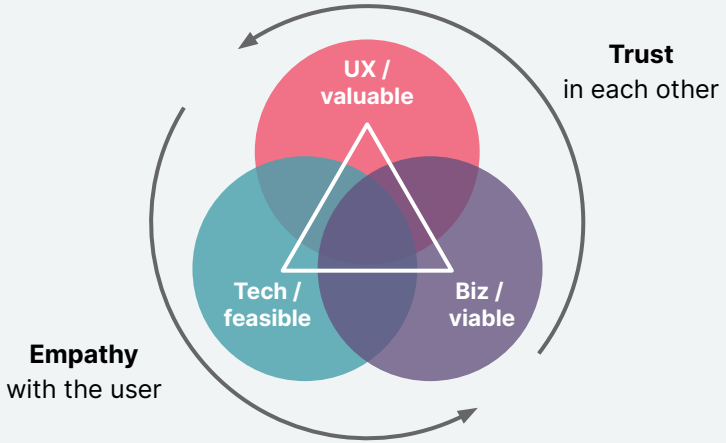
Tech is accountable for: is the product or feature feasible to implement and can we maintain it sustainably?

Effort
Quick win

How often
Continuously

Who participates
Pairing: design, product, tech

Value(s) at practice
- Fast feedback
- Accountability
- Inclusivity



Storming

Technical Analysis Round Robin

Building a shared understanding of upcoming work, both in terms of complexity and value

In agile teams it's common to have a "sprint planning" session in order to provide a high-level view into upcoming work, as not everyone may be aware of everything that is going on. However, this ceremony is frequently unengaging and painful, and often raises more questions than answers.

Effort
Quick win

Who participates
Whole team

Value(s) at practice
- Shared purpose
- Fast feedback
- Accountability

How often
Once every 2 weeks

Tools
- Story wall
- Video conferencing (breakout rooms)

Time required
30 minutes to 1 hour

The technical analysis round robin is an alternative activity that helps developers gain much deeper insight into the work they'll be expected to do, and allows them the space to ask questions and provide feasibility reviews. This gives designers and product strategists fast feedback on whether user stories are ready for development.

(continued on next page)

Storming

Technical Analysis Round Robin

Effort
Quick win

How often
Once every 2 weeks

Who participates
Whole team

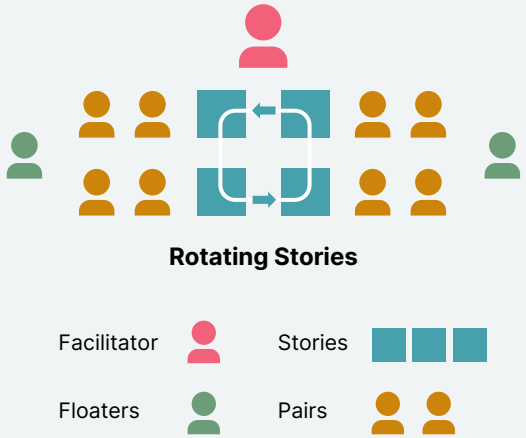
Tools
- Story wall
- Video conferencing (breakout rooms)

Value(s) at practice
- Shared purpose
- Fast feedback
- Accountability

Time required
30 minutes to 1 hour

Process (continued from previous page)

1. Have a list of user stories prepared in advance that you would like to review with the team. As a bonus, you can share how these stories fit into the overall journey.
2. The developers and quality analysts should split into pairs. Assign each pair a story to review for 10 minutes.
3. The pairs will read their story, talk through the implications or consequences with each other, and raise questions. Product strategists and designers float between pairs to discuss concerns that arise on the fly.
4. A determination is made if the story is ready to be played, based on the team's Definition of Ready.
5. If time allows, have each pair look at a second and third story, even if it has already been looked at previously. This helps the team build context around other upcoming work.



Storming

Design Work on the Story Wall

Story walls can become myopically focused on tracking engineering labor. To refocus standups toward visibly socializing work of the entire team, we include the work of design on our story wall.

We capture design processes as cards to bring visibility to design work. This allows the team to discuss progress and address blockers cross-functionally.

Tips

Size of story. The usual assumption is that user stories include design and product work. We choose to put in our story wall the bigger feature work.

Pairing help. Use story discussions in stand-up to foster paired solution exploration.

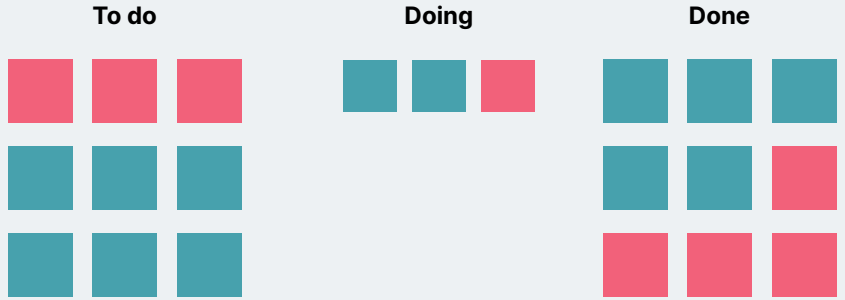
Effort
Quick win

How often
Continuously

Who participates
Pairing: designers and product strategists

Tools
- Story wall

Value(s) at practice
- Shared purpose
- Fast feedback
- Accountability



Storming

Designing Lean

Prioritizing quick feedback, minimizing wasteful processes, and emphasizing lighter deliverables in order to enable continuous delivery

We can apply lean methods to our focus on cross-functional design collaboration so that we learn and test quickly. We keep prototypes as light as needed for the learning we want - most often, using paper and whiteboards - and trust in pairing on the code to cover any information gaps. This allows us to step away from big up front design and be more responsive to change.

Effort
Quick win

Who participates
Pairing: design-led
with product and engineering

Value(s) at practice
- Fast feedback
- Accountability

How often
With every feature, especially complex interaction flows

Tools
- Paper, (virtual) whiteboards
- Light usage of design tools

Process

1. Instead of creating extensive high-fidelity flows, emphasize sketches and low-fidelity prototypes. What's the smallest, quickest thing we can build to get the learning we need?
2. Create static screens with design tools, and capture UX flows on paper or digital whiteboards.
3. Utilize designer-developer pairing to address questions.

Storming

Co-creating Design Principles

Defining design principles is fundamental to building easy-to-use, pleasurable designs

Building a consistent, usable experience is important, and so is defining our process for how we're going to achieve that. Our outputs and outcomes are informed by how we work.

Design principles are the guidance for design, usability, and implementation decisions that affect the user experience and how the team will get there. They are developed as a team, not handed down but coached by the designers. We ask ourselves: "What is of highest value for the user experience?" and "How will the team deliver it?"

(continued on next page)

Effort
Quick win

Who participates
Whole team: facilitated by designers

Value(s) at practice
- Shared purpose
- Quality
- Accountability

How often
Once, monitored and adjusted as needed

Tools
- Post-its / Virtual whiteboard
- Video conferencing

Time required
30 minutes

Storming

Co-creating Design Principles

Effort
Quick win

Who participates
Whole team: facilitated by designers

Value(s) at practice
- Shared purpose
- Quality
- Accountability

How often
Once, monitored and adjusted as needed

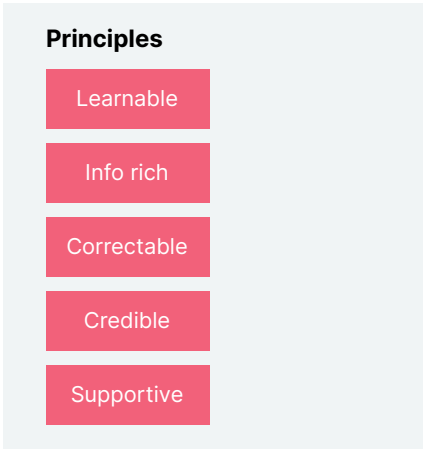
Tools
- Post-its / Virtual whiteboard
- Video conferencing

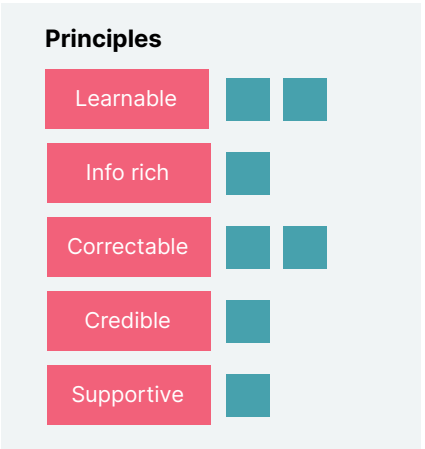
Time required
30 minutes

Process (continued from previous page)

1. The design roles seed a list with some general design principles - usable, accessible, learnable, etc. - that are explained for the team to review on a whiteboard.
2. In small groups, the entire team reviews the principles, paying attention to the meaning of the principles.
3. Each group adds examples to the principles in order to shape our understanding and ensure we have captured the concept correctly. For instance, alt text for accessibility, or plain language for supportive and learnable. The groups can make suggestions for new wording or whole new principles.
4. The designers clean up the board to make the resulting contributions easy to digest and understand. Final principles are available to the entire team to see.

Note: as with all ways of working, there's no reason the team could not revisit any of these principles as work progresses.





Norming

Open User Research Sessions

The essence of a requirement is found in building empathy with the audience that requires it

As there is value in distributing empathy for the user widely through the team, there is value in cross-disciplinary participation in user research.

By watching and questioning users as they interact with our prototypes and working software, the team is able to understand the pains and gains of the user's experience and develop empathy in the first person.

Effort
Strategic investment

Who participates
Whole team: open to everyone

Value(s) at practice
- Fast feedback
- Quality
- Accountability

How often
Qualitative research sessions once or twice a week with extended sessions - workplace tours, discovery workshops - at other times

Tools
- Script, agenda
- Note-taking tools
- Video conferencing

Time required
1-2 hours

Tips

1. Research sessions should be announced and scheduled openly in order to encourage participation.
2. Identify specific representation across roles to be present live, to ensure learning is distributed. Create and share recordings with the entire team.
3. After each session, debrief with the group to share and document insights.

Norming

Showcases

Frequently demonstrating working software and celebrating the team's achievements

Two key components to building trust, especially with business stakeholders, are credibility and reliability. We can demonstrate these qualities by regularly showcasing working software and highlighting how everyone's efforts make a quality user experience possible.

Effort
Quick win

Who participates
Whole team: open to everyone

Value(s) at practice
- Inclusivity
- Accountability

How often
Once every 2-4 weeks

Tools
- Script, agenda
- Video conferencing

Time required
30 minutes to 1 hour

Tips

1. Showcase working software on live environments as much as possible. "It works on my machine" doesn't quite convey the same level of credibility.
2. At the same time, be comfortable with showing some work in progress - it's okay to demo prototypes and mockups, too! The purpose of the showcase is less about having a perfectly polished demo, and more about making the team's contributions visible to each other and to other stakeholders.
3. Try to keep showcases brief and light-hearted; having a facilitator and an agenda can help to ensure the most salient points are communicated to the audience.

Norming

Story Kick-Offs and Desk Checks

Validating assumptions and aligning on desired outcomes of user stories

When developers begin to work on a user story, having a brief "kick-off" with representatives from product, design, and engineering helps to ensure everyone is on the same page about the expected outcomes. (This is sometimes referred to as "the three amigos".)

Similarly, when developers complete their work, a "desk check" is a reunion of the same people to validate that the acceptance criteria have been met.

Effort
Quick win

Who participates
Pairing: developers, quality analysts, product strategists, designers

Value(s) at practice
- Shared purpose
- Fast feedback
- Quality

How often
With every user story

Tools
- Story wall
- Video conferencing

Time required
10-20 minutes

Tip

Kick-offs and desk checks are intended to be short. The purpose is to confirm understanding and ask clarifying questions. If they go on for longer than 20 minutes that tends to be a sign that either the requirements need refinement or the scope of the story is too large.

Norming

Cross-functional Problem Exploration

The Double Diamond model describes how we can systematically **diverge** to investigate the problem space and **converge** on key insights.

The discovery phase is about moving from unknown unknowns to known unknowns. We encourage broad participation: we bring together users, stakeholders, product strategists, designers, quality analysts, and developers to wireframe, design, work through flows, and paper prototype.

In this way, we can synthesize these different points of view into a defined problem statement or a well-informed hypothesis on how the design will support the user's expectations.

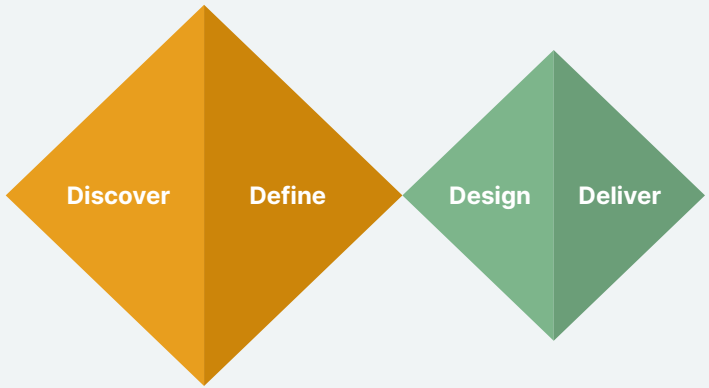
On the following pages, we illustrate an example process known as Design Jams.

Effort
Strategic investment

Who participates
Whole team: design-led, with analysts, developers, users, and other stakeholders

Value(s) at practice
- Shared purpose
- Inclusivity
- Fast feedback

How often
As needed for continuous discovery



Norming

Cross-functional Problem Exploration

Example: Design Jams

Jams borrow from the improvisation of musicians. They are exploratory and cross functional within a time-boxed framework

Design jams are similar to tech spikes in that are time-bound, focused investigation to unlock some capability for future work. Similarly, design jams work through complex design problems that fit outside the scope of a single feature.

Jams, however, borrow from the improvisation of musicians. They are exploratory and cross functional within a time-boxed framework.

Effort
Quick win

Who participates
Pairing: designers lead, other teammates participate as needed

Value(s) at practice
- Fast feedback
- Quality

How often
As needed to reduce design uncertainty

Tools
- Post-its
- Video-conferencing

Time required
Time box should be no longer than 1 week

Design jams are our primary means of design development. The five part structure participatory design sessions, rapid lightweight prototyping, and technical research. The result of a jam is a testable hypothesis that is valuable, feasible, and viable at that point in time. We can test our hypotheses in production through multivariate testing, analytics, usability studies, and / or task-based user testing.

(continued on next page)

Norming

Example: Design Jams

Process (continued from previous page)

A design jam is five parts, usually completed over five days, or half a sprint.

1. Designers and product strategists work with the client to establish requirements, often consulting with engineers.
2. The designers lead various exercises with any combination of team members to explore design solutions. This might include lightweight prototyping, co-sketching, and discussion and voting.
3. When an approach is agreed upon, the designers revise the concepts into higher fidelity mockups. Meanwhile, product and engineering discuss further feasibility and production concerns.
4. Further refinement with the client and discussion within the team takes place.
5. Artefacts are added as needed to places where design is shared, such as user stories or online collaborative design tools.

Effort
Quick win

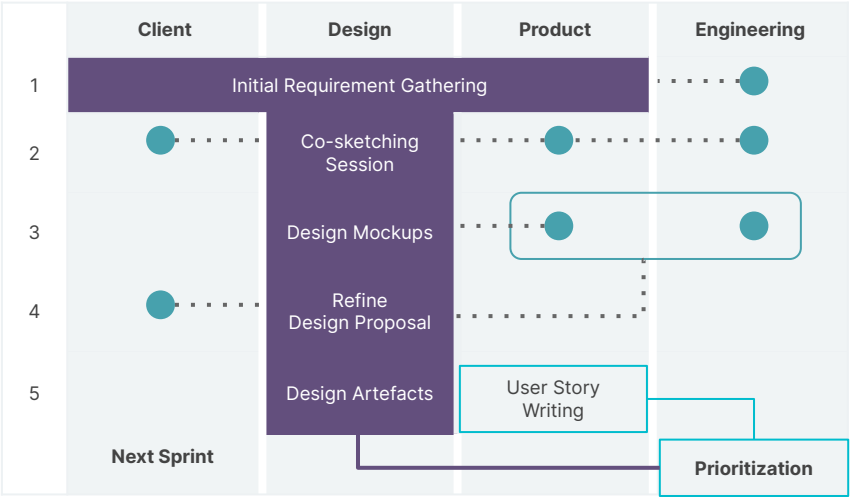
Who participates
Pairing: designers lead, other teammates participate as needed

Value(s) at practice
- Fast feedback
- Quality

How often
As needed to reduce design uncertainty

Tools
- Post-its
- Video-conferencing

Time required
Time box should be no longer than 1 week



Norming

Paired Solution Exploration

Paired solution exploration (the second half of the Double Diamond model) enables us to systematically **diverge** to analyze alternatives and **converge** on solutions that satisfy the user, the business, and the system.

The delivery phase is about moving from known unknowns to known knowns, and encouraging experimentation along the way. What could we learn if quality analysts and designers paired? Developers and customer support staff?

For example, when designers and developers pair on code, we can respond to emerging constraints on the fly. When developers and designers pair on sketches, we can apply our respective expertise to work through various interactions.

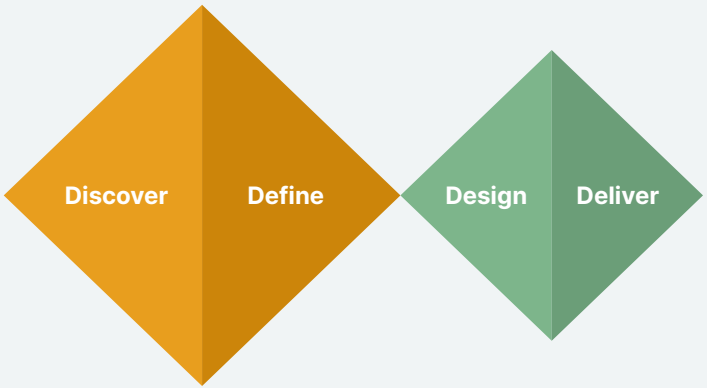
On the following pages we illustrate an example process known as Technical Spikes.

Effort
Quick win

Who participates
Pairing: any roles

Value(s) at practice
- Inclusivity
- Fast feedback

How often
On the fly, as often as the opportunity or need arises



Norming

Paired Solution Exploration

Example: Technical Spikes

Methodical deep dives into technical solutions, in order to reduce unknowns and determine what's fit for purpose

One example of paired solution exploration is a technical spike. The term "spike" comes from Extreme Programming: it's an investigation that involves technical analysis and rapid prototyping of various alternatives in order to reduce technical uncertainty or risk.

Effort
Quick win

Who participates
Pairing: developers lead, other teammates participate as needed

Value(s) at practice
- Fast feedback
- Quality

How often
As needed to reduce technical uncertainty

Time required
Time box should generally not be longer than 2 weeks

We split out spikes from user stories so that we surface complexities and unknowns with our stakeholders. Product requirements help to set explicit boundaries on our investigations, to ensure we're actively removing blockers and unlocking capabilities required for future work.

After implementing our solutions, we can test our hypotheses in production through a variety of means: multivariate testing, analytics, open research sessions, task-based user testing, contextual inquiries, rapid prototyping, etc.

(continued on next page)

Norming

Paired Solution Exploration

Technical Spikes

Effort
Quick win

Who participates
Pairing: developers lead, other teammates participate as needed

Value(s) at practice
- Fast feedback
- Quality

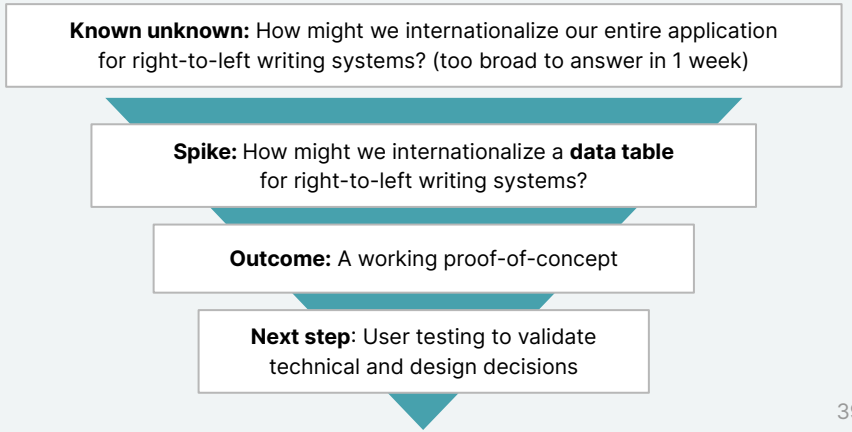
How often
As needed to reduce technical uncertainty

Time required
Time box should generally not be longer than 2 weeks

Process (continued from previous page)

1. Define the focus for the investigation: what is the **one** question that needs to be answered? You might also consider amending your team's Definitions of Ready and Done to include what you expect from spikes.
2. Define a time box for the spike, and what should happen if satisfactory progress is not made within that time. The time box should be defined in days, not weeks; if the time box is too long, that's generally a sign that too many questions are being asked at once, and that it can be broken down into multiple spikes. Time boxes should be treated as checkpoints, not deadlines.
3. Define the desired outcomes and outputs. For example, a proof-of-concept, a show and tell session with the team, or a technology recommendation.
4. Record discoveries and recommendations in a Technical Decision Log to capture the context around the investigation, and determine next steps for testing.

Design and Deliver: Narrowing the Solution Space



Norming

Reusable Interaction Patterns

Facilitating consistent user flows and interactions, while multiple features are in development simultaneously

Reusable interaction patterns (often referred to as “design patterns”) are developed, prototyped, and evaluated cross-functionally in groups and pairs. We work through these interaction patterns continuously and throughout feature shaping and implementation. Shared ownership enables team members to troubleshoot their own work, because they can think through the flow and how it applies in different contexts.

These patterns are the glue that hold reusable components together. Whenever a conflict or need for a new pattern is discovered, we discuss whether we need to immediately propagate the change universally or if we will prioritize it against our existing UX debt.

Effort
Strategic investment

Who participates
Pairing: across the team, in pairs and small groups

Value(s) at practice
- Shared purpose
- Quality
- Fast feedback

How often
As needed but often as part of feature discovery

Norming

Centering the User Journey Map

Highlighting the capabilities that our product supports

Effort
Strategic investment

Who participates
Pairing: design, product, tech

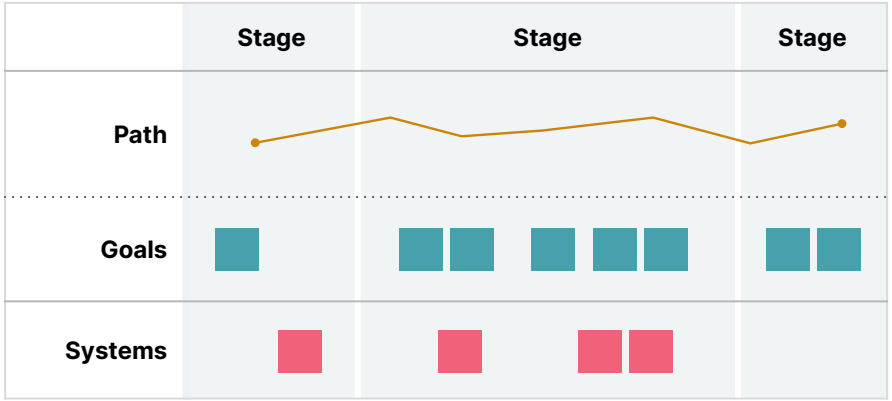
Value(s) at practice
Shared purpose

How often
Once to establish a core journey, updated continuously as needed

Tools
- Virtual whiteboard
- Alternate: spreadsheet

The user experience journey map includes the user's processes, their goals or jobs to be done (JTBD), as well as underlying systems. It reveals to us gaps in our understanding and places for hypothesis driven design and development.

We consciously integrate the journey map into our work. We drive home the value of the journey to entire team and our stakeholders by updating progress during showcases and referring to it during sprint planning.



Tips

Focus on user goals, not just experience with the product. The difference between a flow and journey is context for actions. Goals drive user action.

Represent the entire team's work. The journey map includes user actions and system actions. This helps us understand the interconnected relationships driving toward the customer experience.

Norming

Open Design Pin-ups

Establishing socially-visible progress of design work

We are always looking for opportunities to make the practice of design visible to the heads-down work of developers and product analysts.

By opening up in-progress reviews to the entire team, we underscore the work that goes into design decisions. This is also a quick way for the team to give feedback or raise red flags on our upcoming work, so the development team can be proactive. These updates also help foster shared ownership of design work.

Even with our practice of Design Work on the Story Wall, walking the wall during stand-up still tends to be a very tech-focused conversation. These design updates give space to talk in more detail about design work in progress, recent research findings, open space for critique and general discussion.

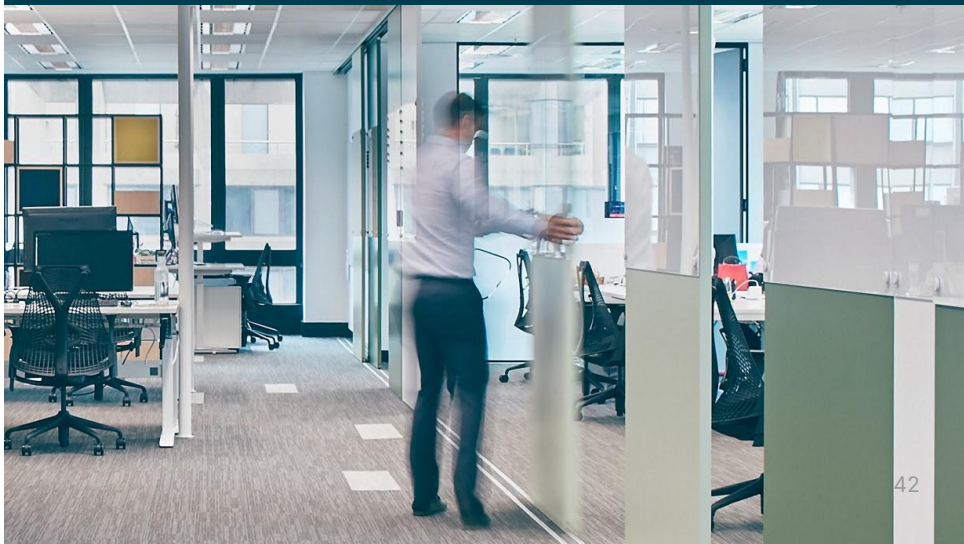
Effort
Quick win

How often
Weekly

Who participates
Whole team: open invitation

Tools
- Video conferencing
- Presentation program

Value(s) at practice
- Shared purpose
- Fast feedback



Norming

Testing the Observable UI

Tests should answer **why** the components work the way they do, not just **how** they work

Our automated testing strategy emphasizes the behaviors of the observable UI. This means we emulate how a user would interact with our application and we assert on what matters to them, instead of asserting on internal details.

(continued on next page)

Effort
Strategic investment

Who participates
Pairing: developers and quality analysts

Value(s) at practice
Quality

How often
With every user story

Tools
- Javascript testing tools
- Visual regression testing tools
- Accessibility testing tools
- Browser-based testing tools

Not this: Internal POV

```
const modalIsOpen = modal.getState('open');

expect(modalIsOpen).toBe(true);
```

But this: External POV

```
const modal = find('#modal');

expect(modal).toBePresent();
```

Norming

Testing the Observable UI

Instead of this:

System POV

```
test('clicking button opens the modal', () => {
    const button = find('#button');
    button.click();

    const modal = find('#modal');
    expect(modal).toBePresent();
});
```

Do this:

User POV

```
test('using quick shop displays product
information', () => {
    const quickShopButton =
        find('#product-123-quick-shop-button');

    quickShopButton.click();

    const quickShopModal =
        find('#quick-shop-modal');

    expect(quickShopModal)
        .toContain('product 123 info');
});
```


Effort
Strategic investment

Who participates
Pairing: developers and quality analysts

Value(s) at practice
Quality

How often
With every user story

Tools
- Javascript testing tools
- Visual regression testing tools
- Accessibility testing tools
- Browser-based testing tools

(continued from previous page)

It might be strange to think of an automated testing strategy as serving the goals of “design as a team”. Through this lens, our tests speak the language of our domain, and thereby document and demonstrate the real-world interactions we support.

We can apply this principle to almost every kind of test we write for the UI: Javascript component tests, visual regression tests, accessibility tests, and browser-based journey tests. This shouldn’t be interpreted as writing only integration-level tests instead of unit tests. We are instead redefining what a unit really is in the context of our domain and our architecture.

Norming

Design Systems and Reusable Components

Creating a shared language, building consistent experiences across products, and enabling effective reuse of assets

Design systems are a great way to share design and technical assets widely with many product teams. The first step to building a design system is to think in reusable components: they are the foundation for our product, rather than an afterthought or an outcome of refactoring code.

Imagining components as interchangeable and movable building blocks enables us to deliver faster and prevent technical and design debt. Most importantly, these components provide us with a taxonomy to describe our product suite that can be understood by everyone.

(continued on next page)

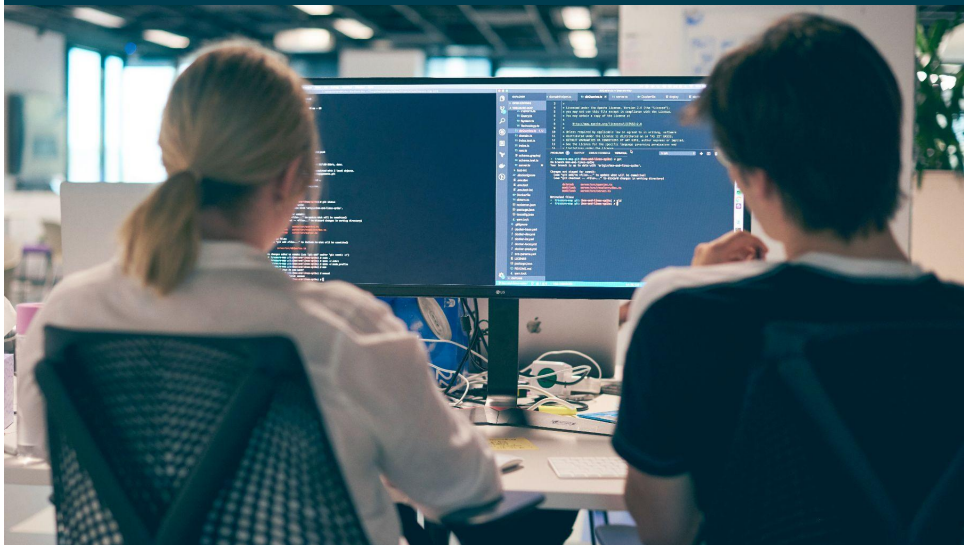
Effort
Strategic investment

Who participates
Whole team

Value(s) at practice
- Shared purpose
- Fast feedback
- Quality

How often
Evaluate with every user story if new interaction patterns or concepts are needed, or if we can reuse an existing pattern

Tools
- Component explorer
- Javascript
- HTML
- CSS
- Web-based design tools



Norming

Design Systems and Reusable Components

(continued from previous page)

A design system is an internal-facing product with other teams as consumers. They require organizational infrastructure, a clear governance model, and community outreach. While not every organization has a fully-fledged design system or is ready to invest in building one, that doesn't mean teams can't start thinking of their products in terms of reusable components to accelerate their delivery.

Process

1. Print out a page from your application, either as it is live today or a mockup.
2. First draw boxes around the smallest meaningful units that exist. For example, a text label by itself doesn't have meaning on its own, but when next to an input field, the two elements together create meaning as a form field.
3. Continue drawing boxes around the next-smallest units successively, until every single element has been enclosed by a single box, representing the entire page.

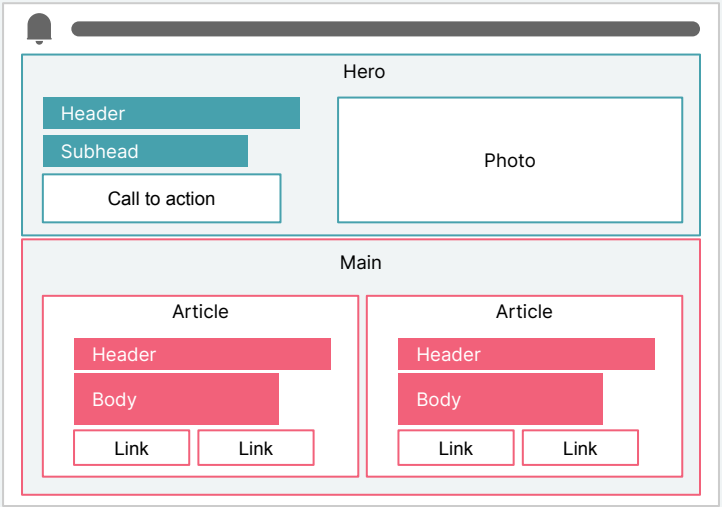
Effort
Strategic investment

Who participates
Whole team

Value(s) at practice
- Shared purpose
- Fast feedback
- Quality

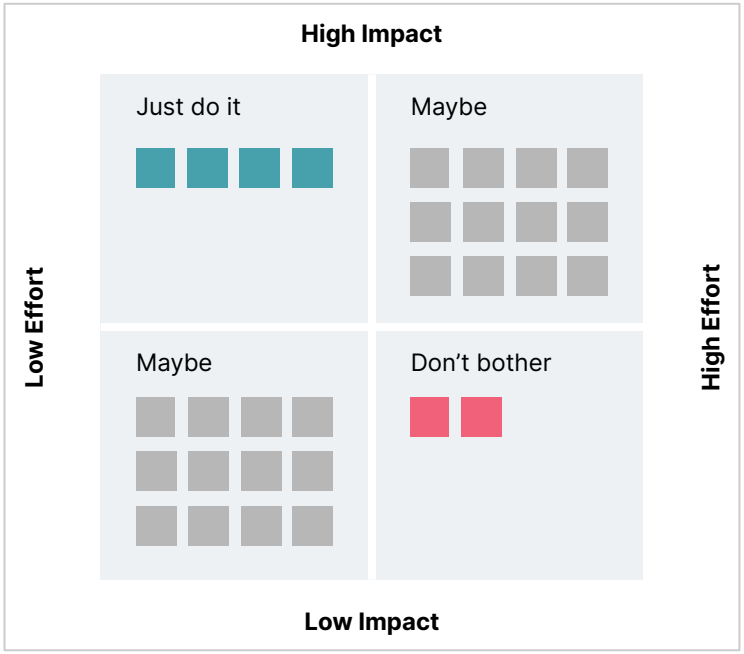
How often
Evaluate with every user story if new interaction patterns or concepts are needed, or if we can reuse an existing pattern

Tools
- Component explorer
- Javascript
- HTML
- CSS
- Web-based design tools



Performing

Design and Tech Debt Audit



Effort
Quick win

Who participates
Pairing: designers, quality analysts, and developers

Value(s) at practice
- Quality
- Inclusivity

How often
- Capture items as observed
- Review items once every 1-2 weeks

Tools
- Post-its / Virtual whiteboard
- Video conferencing

Time required
30 minutes

Technical debt represents a trade-off between quality or consistency and speed; too much can eventually hinder our ability to make changes with confidence. Similarly, the accumulation of design debt can erode users’ confidence in the system, or worse, in themselves, for not being able to figure out how to move past issues.

At best, these issues are slightly irritating quirks, and at worst, they actively prevent us from completing our objectives. We regularly perform a heuristic evaluation of our code and of our product and prioritize both types of debt via an effort-vs-impact analysis. In this way we ensure we are maximizing our ability to continuously improve our solution.

Performing

Technical and Design Decision Log

A technical decision log is a bit broader in scope than an architectural decision record (ADR). In addition to architectural decisions, it includes any critical decisions about engineering practices or conventions and important discoveries made during spikes and design jams. Just like with ADRs, we keep the decision log in source control and use a very lightweight template to capture just enough context for the decision. This practice can be extended to include interaction and visual design decisions, or even key product decisions.

Too often important context only lives inside someone's mind, or is buried deep in slide decks or emails. Decision logs are meant to be succinct and accessible.

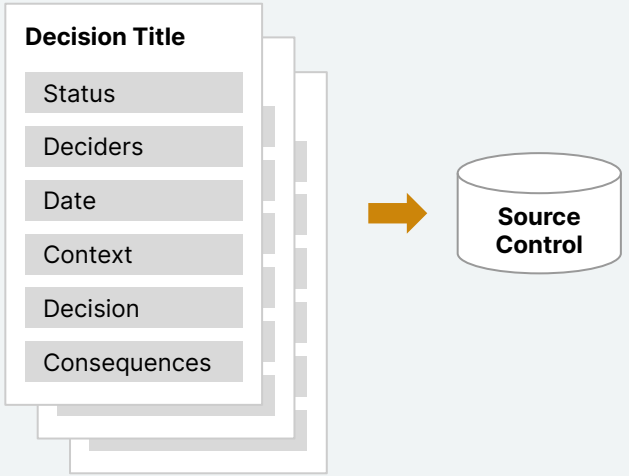
Effort
Quick win

Who participates
Whole team

Value(s) at practice
- Quality
- Accountability

How often
Decisions are recorded as they are made

Tools
- Ideally: Source control
- Alternative: spreadsheets or wikis



Performing

Measuring Impact

Continuously learning what is valuable, feasible, and viable, and reducing friction between product, tech, and design

How do we measure impact? How do we know that our work is in fact contributing to successful outcomes for our users?

It's a worthwhile exercise to continuously evaluate our practices against our values to decide when to add, modify, or even abandon those practices. Focusing on practices first helps us to gain fluency in the methodologies of cross-functional teams. Connecting these practices back to principles and values is what helps us understand why they're valuable, and enables us to replicate success from one team to the next.

This allows us to think about what agile software development looks like beyond the realm of just the developers, and what it ultimately means for our users and our business.

(continued on next page)

Effort
Strategic investment

Who participates
Whole team

Value(s) at practice
- Shared purpose
- Quality
- Inclusivity
- Fast feedback
- Accountability

How often
Establish a cadence within existing, relevant habits

Performing

Measuring Impact

(continued from previous page)

We can measure the effectiveness of specific practices through real-time retrospectives, but to evaluate our technical and social practices more holistically, there are a variety of techniques that can be excellent starting points:

Social practices

How are we fostering **empathy** with each other and our users?

- Contextual inquiries
- Hierarchy of needs: BICEPS

Are we maintaining psychological **safety and trust** on the team?

- Safety checks and retrospectives
- Healthy feedback culture
- Trust equation
- Crucial Conversations
- Radical Candor

Effort
Strategic investment

Who participates
Whole team

Value(s) at practice

- Shared purpose
- Quality
- Inclusivity
- Fast feedback
- Accountability

How often
Establish a cadence within existing, relevant habits

Technical practices

What are the needs we're trying to meet? (Is our solution **valuable**?)

- HEART framework
- AARRR framework
- Qualitative user research

How do we deliver? (Is our solution **feasible**?)

- Architectural Fitness Functions
- Service Level Objectives
- RAIDs

Can we sustain our momentum and evolve? (Is our solution **viable**?)

- EDGE / Lean Value Tree
- OKRs
- 4 Key Metrics

Thank you.

For more insight into our thinking behind this guidebook, read our article on Design as a Team. We developed these activities for a distributed team. Our Remote Work playbook is a complementary resource that you may also find useful for building cross-functional collaboration.

contact-us@thoughtworks.com | thoughtworks.com
thoughtworks.com/what-we-do/customer-experience-product-design

/thoughtworks