Kickstart Timeline

WEEK 1

Prepare

- Kickoff call

WEEK 2

Design

- Design and team governance

WEEK 3-4

Build

- Sketch review and design system structure

WEEK 5

Integrate

- DSM for developers

WEEK 6-7

Scale

- Versioning and scaling

WEEK 8

- The Connected Workflow

Kickstart

The goal of Kickstart is to help customers build a foundation for their design system. We will guide them through the setup of their design system and uploading 5-10 components. As we do this we'll be teaching them how to duplicate this building process for additional products and components. Moving forward they'll have all of the tools and information they need to build, grow, and manage their design system for long-term success.

What's included?

- Access to on-demand content
- Remote, live trainings and office hours
- Worksheets, templates, and checklists

The program content is a mix of self-serve and on-demand resources with live training calls delivered in a shared, group call. Customers are taken through the program in cohorts. These cohorts typically include 2-5 customers that all progress through their journey at the same pace.

Who should be involved?

- For the Kickoff call we recommend including everyone who has a seat in DSM.
- After the call, we will also have you define your design system makers at the beginning, so you may choose to have a smaller or larger group that represents your team.

How much time do I need to dedicate?

- We suggest scheduling at least 1.5 hours per week to work together as a team to complete the activities plus the live training calls.

InVision University

Kickstart
Journey Map

This journey map will give you an overview of what to expect through your program. We have a phased approach with five modules. The bold task are our calls, the italics are optional activities, depending on where you are in your design system journey.

InVision DSM.

Prepare

- [ ] **Kickoff call**
- [ ] *Conduct an Interface Inventory Audit*
- [ ] *Formalize your Design Purpose and Principles*
- [ ] DSM and design systems overview

Design

- [ ] Complete the Team Mapping worksheet
- [ ] Complete the Governance worksheet
- [ ] Complete the Design System worksheet
- [ ] Complete the Scope worksheet
- [ ] Choose your approach and make your component selection
- [ ] Prepare your Sketch file

Build

- [ ] Sketch review and design system structure
- [ ] Review and update Sketch file
- [ ] Complete the Naming Convention worksheet and sign off
- [ ] **Office hours**
- [ ] Complete the Design System Structure worksheet and sign off
- [ ] Complete the Documentation worksheet and sign off
- [ ] Build and review your design system in DSM and sign off

Integrate

- [ ] DSM for developers
- [ ] Create one test component with your development team
- [ ] Storybook + DSM setup
- [ ] Attach any additional dev repos/resources
- [ ] Review the live component integration and sign off
- [ ] **Office hours**

Scale

- [ ] Versioning and scaling
- [ ] Complete the Contributes Process Guide worksheet
- [ ] Complete the Version Strategy worksheet and sign off
- [ ] **The connected workflow call**
- [ ] Your design system and digital product design workflow
- [ ] **Release your version 0.1.0**

Project Plan

This Project Plan is a detailed and in-depth overview of your design system journey through theprogram. We have a phased approach with five modules. The bold task are our calls, the italics are optional activities.

Prepare

- [ ] **Kickoff call**

  Intro call to review the program, what is included, the timeline, and introduce you to all of your resources.

- [ ] *Conduct an Interface Inventory Audit*

  Based on your chosen product, complete an inventory of all the components and start to define which ones you want in your DS. Be sure to discard duplicates and unwanted variations. Use this as an opportunity to see what your development team has, in respect to those components. *If you already completed this, feel free to move on.

- [ ] *Formalize your Design Purpose and Principles*

  Set your northstar for your design system that matches your company's visions and goals

- [ ] DSM and design systems overview

  Now that you have your team, make sure everyone has a basic understanding of DSM. For those who don't, have them check out the overview video in the Resource hub.

Design

- [ ] Define your team

  Begin by identifying the people who are going to help you get this launched by completing the TEAM MAPPING worksheet.

- [ ] Decide on a governance model and sign off

  Complete the GOVERNANCE worksheet.

- [ ] Designing your design system

  Complete the DESIGN SYSTEM QUESTIONS worksheet. This will help us recommend resources for you and help you document your goals.

- [ ] Define your minimum viable product (MVP)

  Complete the SCOPE worksheet to identify your MVP. This will help you narrow down your focus.

- [ ] Choose your approach to selecting components

  Complete the CHOOSING COMPONENTS worksheet and choose which approach you will use to select your components.

- [ ] Prepare your Sketch file with your MVP components

  Consolidate the design elements and components you have selected into a single Sketch file. This should include your colors, text styles, layer styles, icons, components, templates, grids, and/or pages.

Build

- [ ] Sketch review and design system structure

  We'll go over best practices for Sketch hygiene, building, and structuring your design system.

- [ ] Sketch file amendments

  Take time to review your Sketch file to ensure you are following our recommended best practices.

- [ ] Sign off on your Sketch file

  Gather your team to review changes made to the Sketch file and have everyone confirm the changes.

- [ ] Discuss naming conventions with your designers and developers together

  We'll touch on best practices for naming conventions. Take some time to complete the NAMING CONVENTION worksheet. Review it with developers to align or adopt a certain naming convention.

- [ ] Sign off on naming conventions

  Gather your team to align and confirm your naming structure for your chosen atoms and components.

- [ ] **Office hours**

  Come to office hours to get your questions answered live.

- [ ] Decide on your design system structure

  Complete the DESIGN SYSTEM STRUCTURE worksheet.

- [ ] Design system structure amendments

  Review your design system structure with your developers and designers and make changes.

- [ ] Sign off on your library structure

  After making amendements, gather your team to sign off on the agreed design system structure.

- [ ] Define your documentation

  Speak to other teams about documentation needs and finalize your DOCUMENTATION worksheet.

- [ ] Documentation amendments

  Make the final tweaks to your documentation template before using it to start documenting your components.

- [ ] Sign off on documentation

  We recommend adding this to your DSM so it's accessible to anyone needing to create documentation.

- [ ] Build your design system structure in InVision DSM

  Based on your completed Design System Structure Worksheet and training, build out your libraries and folders.

- [ ] Upload and document your atoms

  Upload all your atoms first; colors, layer styles, text styles and icons. Add your documentation for your atomic elements.

- [ ] Upload and document your components

  Begin to add your components to DSM and document them. Make sure all the atoms that make up those components are already uploaded to DSM.

- [ ] Review your design system

  Review your design system with all your components.

Integrate

- [ ] DSM for developers

  Learn how your developers can use DSM and how to combine design and development with InVision DSM to create a single source of truth.

- [ ] Complete the Developer Checklist

  Optional: Review the questions in the DEVELOPER CHECKLIST to make sure you have covered your bases. Then follow the checklist as you set up your coded components.

- [ ] Build one component to test

  We recommend having your developers create one of the components in Storybook and ensure it looks and interacts as per the design.

- [ ] Storybook+DSM setup

  Create the link between Storybook and DSM following our online instructions.

- [ ] Link one component

  Link a coded component to the designed component in DSM. We recommend starting with one test component for now.

- [ ] Review and sync with designers

  Have your developers and designers sync on all the created components and ensure that they are coded as expected per designs.

- [ ] Sign off on built components and Storybook integration

  Sign off on the coded components you're ready to link to designed components.

- [ ] Build components

  Have the development team build out all the components for your design system release.

- [ ] Link all components

  Have the development team link all your Storybook components to DSM.

- [ ] **Office hours**

  Come to office hours to get your questions answered live.

Scale

- [ ] Versioning and scaling

  Review versioning tips and best practices and how it is done with InVision DSM. We'll also cover tips and best practices for scaling your design system.

- [ ] Decide on a contributors process and sign off

  Complete the CONTRIBUTORS PROCESS GUIDE worksheet.

- [ ] Add your governance model and process to your design system

  Add in documentation to your DSM design system in a section for your governance model and the process for adding or requesting in new components.

- [ ] Decide on your version strategy and communcation process

  Complete the VERSION STRATEGY AND COMMUNICATION PLAN worksheet.

- [ ] Sign off on your versioning strategy and communication process

  Communication is key. Gather your team to go over how to notify people of version releases and other communication around your design system.

- [ ] Add your version strategy to your design system documentation

  Add in documentation to your DSM in a section for your versioning strategy.

- [ ] Add your communication plan design system documentation

  Add in your communication plan to your DSM in a section for communication.

- [ ] Add your purpose and princples to your design system documentation

  If you haven't done so yet, add in your design purpose and principles to your DSM in a section.

- [ ] Add a "Get Started" and "Intro" section to your design system documentation

  Optional: add in a "Getting Started" section to introduce newbies to your design system and help get them oriented quickly.

- [ ] End-user training

  We recommend creating your own training video using your design system, and you can use our DSM TRAINING OUTLINE as a guide for what to include.

- [ ] **MVP beta release**

  Release v0.1.0 of your design system!

Scale

- [ ] Start internal communication

  Start sending internal communication about your design system, building excitement.

- [ ] Testing/UAT period

  Collect feedback during a testing period. Track where your design system users stumble or excel and how they use it.

- [ ] Review feedback from testing and make amendments

  Time for you to review feedback and make changes to your design system as needed.

- [ ] What's next

  Use the PRODUCT MAPPING WORKSHEET to plan out your next product(s) as you continue to build your design system.

- [ ] Begin building your full first version

  Follow this process again as you fully build out the first full version of your product design system.

- [ ] Release checklist

  Build, test, review, and update your beta release. Once you are ready to release V1 use the MVP RELEASE CHECKLIST to make sure you have completed all the recommended steps.

- [ ] **The connected workflow call**

  Learn how you can bring everything together with InVision DSM and Enterprise.

- [ ] Review and decide on your next product and MVP

  Redo and adapt the process as needed for each product.

- [ ] Onboard new members of your team

  We've included the ONBOARDING CHECKLIST to help new members of your team start using your design system.

Design System Questions

This design system worksheet will help you identify where you are in your design system journey, create goals, and identify any potential blockers. You and your team will use this document as a reference throughout the program.

Organization/Company Name:

Foundation

☐ YES	☐ NO	Have you defined your designs purpose and principles?
☐ YES	☐ NO	Do you have an existing centralized Sketch sticker sheet, pattern library, or style guide that you plan to use to build your design system?
☐ YES	☐ NO	Have you aligned with your development team regarding your design system's goals and priorities?
☐ YES	☐ NO	Do you have a process for vetting, making, and communicating changes to your system's elements (components, styles, usages guidelines, etc.)?

Overview

What tools do you currently use to share components/assets across the design org?

- ☐ Email
- ☐ Slack
- ☐ Website
- ☐ Shared Drive (Dropbox, Google Drive, etc.)
- ☐ Nothing
- ☐ Other:

What is your current handoff process from design to development?

Design System Success

How would you describe the state of our design system at the moment?

- [ ] We don't have one.
- [ ] We picked out our design elements.
- [ ] We consolidated design elements into a single Sketch file.
- [ ] We added elements to DSM.
- [ ] We added our documentation.
- [ ] We linked design with our code.
- [ ] We are scaling our system.
- [ ] Other:

Do you have any known obstacles between you and a successful design system?

Is there anything else you'd like to share with us about your goals?

Congratulations on taking the first steps towards a successful design system setup.

Define your makers

You need a team of heroes to get this thing launched. Document your team here. It should be cross-functional but not too big.

If you're a small team just looking to get this off the ground, we'd suggest a super small team of 1 UI and UX designer and 1 developer. This team will make the final decisions and manage the updates to your DS. But this doesn't mean other members of the team can't contribute to the design system.

A crucial part of creating a successful design system is understanding your end users and stakeholders. Defining these users and stakeholders will help your makers as they create and plan your documentation, structure, governance, and communications.

For more information, check out a wonderful excerpt on who should be involved from Chapter 2 in the Design Systems Handbook, Designing your design system by Jina Anne.

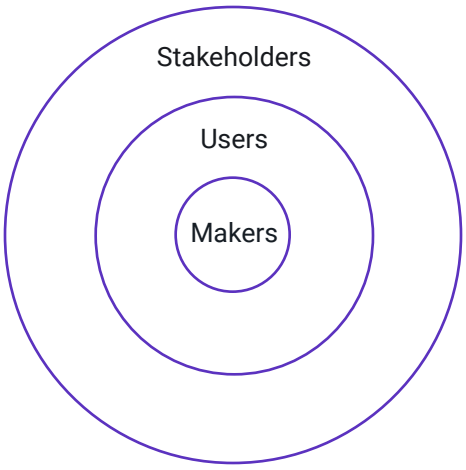
Ask These Questions

1. Does this person have the autonomy to make key decisions?
2. Does this person have the technical skills to work with the tooling?
3. Does this person have the bandwidth (regular standups)?
4. Does this person have a voice to communicate things that need to be addressed?
5. What is this person's proximity/reach to the rest of the organization?
6. What is this person's involvement in the creation of products?

Role	Name
UI Designer	
UX Designer	
Developer	
Product Owner	
Operations Manager	

Responsibilities	Release Versions	Write Documentation	Create Release Notes	Create approved Components
UI Designer				
UX Designer				
Developer				
Product Owner				
Operations Manager				

Stakeholder List



List out everyone who will have involvement in the system. You can do this in a document or use sticky notes in a workshop format. The idea here is to list each individual and their job title who will have involvement in the system, whether a user or other stakeholders. Don't worry about defining their involvement yet. Just focus on who those individuals are. If you don't have a specific person's name, you can simply list their role (i.e. Product Manager from group X).

Once you have your list of stakeholders, begin organizing them into the Stakeholder Roles.

1. Design System User–those who will be consuming from the system, but not directly contributing (designers and front-end devs)
2. Core Stakeholder–other key individuals who will participate in the design system project. This can often include QA engineers, backend devs, product managers, ops, marketing, and copywriting.
3. Other Stakeholders–informed roles who have no direct role in the system, but need to keep apprised of system updates.

Fill in the table below or use this raw data to create a visual map orchart in the tool of your choosing. For this list, focus on the users and stakeholders that would be within the product your components are from. During the SCALE phase we will review this again to build out your larger design system.

Users	Core Stakeholders	Other Stakeholders

Governance

Design system governance is critical. And what it really means is making sure the system keeps up with the needs of the product and as the design system improves. This is a cyclical process that needs to be clearly defined within your organization.

This can start with something as small as creating one or two checkpoints as components come into and out of the DS.

Begin to plan out how you want to govern your system. You will find that this changes over time, and that's okay. Review this as a team and determine your initial strategy for governance.

Governance Plan

Team Name	*Optional: Have some fun and give your governance team a name*
What is your design system governance?	*Short description of how your team views governance.*
Who's involved?	*List who is involved and note their responsibilities for governing the system.*
What's your team model?	*Solitary, Centralised, Federated Model, Hybrid, other*

Governance Plan

Cadence	*We recommend defining this and getting something on the calendar. This way the meetings don't get pushed to the sidelines for other projects.* *Who will meet?* *How often?* *For how long?* *How will you communicate decisions?*
Error Logging	*How do we want to receive bugs?* *How will we communicate the progress of these bugs and communicate back to users when we have queries?*

Other Considerations

Product team adoption	*How will you encourage your product team to adopt the design system and be aware of the updates?*
QA process	*How will you handle questions regarding the system? How will you encourage the team to send in bugs or questions?*
Support process	*How will the Design System makers get contacted for support issues? (You may tie this into the contributor's guide below)* *How will they handle those support requests (bugs, questions, etc.)? (You may tie this into the contributor's guide below)*
DSM component goals	*Optional: Feel free to set goals for your design system team. Some examples could include: 1) Receive at least 4 submissions for new components each week 2) Receive less than 3 bugs after a sprint releas*
Additional priorities	

Start to think about specific scenarios you may face as a team and think abou thow you would handle them.

Scenario 1	*When the DS's components don't exist or don't fulfill requirements?*
Scenario 2	*When is a new component just a Snowflake vs Part of the DS?*
Scenario 3	*When do we need to update or remove a component from our DS?*
Scenario 4	*What happens when we have legacy components?*
Scenario 5	*How do we manage the updating of legacy products?*
Scenario 6	*When we have an urgent need for a component that a) hasn't been coded yet b) the schedule release cycle has just passed?*

Scope

One of the biggest failures with launching a design system is not having a clear scope and date to work towards. As such, we would strongly advise defining a clear scope for your Minimum Viable Product (MVP) and setting a realistic (but not too far away) target.

To ensure you stick to this target, we'd also suggest committing this date to the business. Sounds scary, but if you set expectations right and meet them, you'll find people will buy-in and look forward to the next release.

First review what products you have to choose from.This is meant to be a high-level review to help you narrow down your scope. Determine your potential libraries, purpose and principles, and your overall success criteria.

Libraries

List all of the libraries you'll have. For example: Core components library. Web components, App components OR Brand A, Brand B, etc.	

Description/Introduction

Add in an introduction to your design system. Think about what and why you're doing it.	

Purpose and Principles

Add in the purpose and principles for your design system.

Success Criteria

How will you measure the success of your design system?

Your MVP

Now that you have reviewed your options, let's narrow down the list to your MVP, focusing on one product the program. You'll build this document throughout your program, so be sure to keep it handy!

Design System Name:

Pick a name for your design system! It'll help with adoption and buy-in. Plus it's cool.

Generate excitement for your design system by running a competition to name it!

Product Name: **Device:**

Choose the product you will be focusing on and which device. We would suggest focusing on just one product rather than trying to tackle too many.

If you have multiple products and devices that share common atoms and components, you may want to consider starting with a global library.

Release Target Date:

When? Think about when you want to release your MVP. This will largely depend on what is in your MVP. But as an idea, if you were only focusing on the atoms (colors, layer styles, text styles and icons) and 5-8 components (buttons, input fields) then 7-8 weeks from the date you start is a realistic target.

Components

Review the approaches in the "Choosing Components" worksheet. Choose your approach and your components to start creating your library.

Choose 5 Components and list them here:

1.

2.

3.

4.

5.

Component Breakdown

Either after the Sketch review or before (if you feel comfortable), take the 5ish components and break down their atoms. List the colors, text styles, layer styles, and icons that make up your larger components.

Atoms

Colors	
Text Styles	
Layer Styles	
Icons	

Molecules

Choosing Components

There are many ways you can approach your first minimum viable product (MVP) and decide which components to pick. Here are just a few suggestions.

- Developer-built components
- Components with highest design debt
- Atomic structure
- Specific page/screen

Review each of these approaches and decide as a team which direction is best for you.

Caveat; In all of these cases, you will define and set up your atoms (colors, text styles, layer styles and icons) first

Component Selection

Developer-built components	*Recommended approach* • Identify the components your developers have built in code and start with those.
Highest design debt	• Complete your interface inventory to identify components that are creating the highest design debt. After reducing the debt and saving your most valuable components, add them to your design system. This will create efficiency and consistency in your team's designs.
Atomic approach	• Choose your most common components (between 5-10) you use. Then break them down to the fundamental building blocks and work up from there. • *Reference Atomic Design by Brad Frost*
Specific page/screen	• Choose one page or screen and start with the components that make up that page or screen.

Your Approach

Which approach above works best for your team and your vision for your design system moving forward?	

Naming Convention

When considering naming conventions for components in a design system, it's important to prioritize consistency, clarity, and meaning.

Our recommendation is to use all lowercase letters, separate words with hyphens or forward slashes, and follow an "order of operations" type structure.

Avoid using numbers or symbols, words that describe visual characteristics, like colors, or the specific font family (e.g. blue or arial), and focus on the components role (e.g. brand or input). The goal is to make it easy to get a feel for what role it plays in a system, just by looking at the name.

Style and Component Specific Guides

Colors	
Structure	*use/variation*
Examples	*primary/default*
	tertiary/light

Text Styles	
Structure	*category/type/size/style/alignment*
Examples	*para/large/bold/right*
	h1/inverse/right
	link/small/bold/center

Layer Styles	
Structure	*category/use/variation or category/group/variant/characteristics*
Examples	*ui/input/default*
	fill/primary/hover
	shadow/high
	border/error/1px
	elevation/high

Components	
Structure	*category/type/element/variation [or] organism/molecule/atom/variation*
Examples	*btn/primary/hover*
	nav/header/mobile/dark
	cards/media-block/complex/web

Area	
Category	*mobile, inverse, ui*
Type	*header, hero, title, subtitle, paragraph, button, input, caption*
Size	*large, medium, small, default*
Alignment	*left, center, right*

Other Tips

We have the structure, but how do we actually name the categories?

While naming, try to focus on the element's role in the system, but not its form. Don't name your primary action button 'button/blue/default', just because it's blue. It's a primary button in your system, so name it something like button/primary/default. This sets you up for success down the road as your system continues to evolve. For example, if the color of a primary button changes, you won't need to rename every instance of that button in your system and files. Additionally, this allows you to get a feel of what role it plays in a system, just by looking at the element's name.

A case against indexing

Try to make names distinctive and clear; avoid using indexes. Names that make contextual sense are a key to avoiding confusion and mistakes.

e. g. dropdown/main, dropdown/secondary instead of dropdown1, dropdown2.

Following the Atomic methodology

If you'd like to strictly adhere to an atomic structure, you could add a letter label to each component, indicating which category it represents: atom, molecule, organism.

e.g. a/btn/primary/default, m/data/table/large, o/navigation/header/light

Your Naming

Now that you have reviewed the template, work with your designers and developers to align on your naming convention plan.

Area	Convention
Colors	
Layer Styles	
Text Styles	
Icons	
Components	

Take some time to review and decide on your Sketch naming conventions for building components. This will help your designers use a more consistent naming structure for layers and help your developers as they build out components.

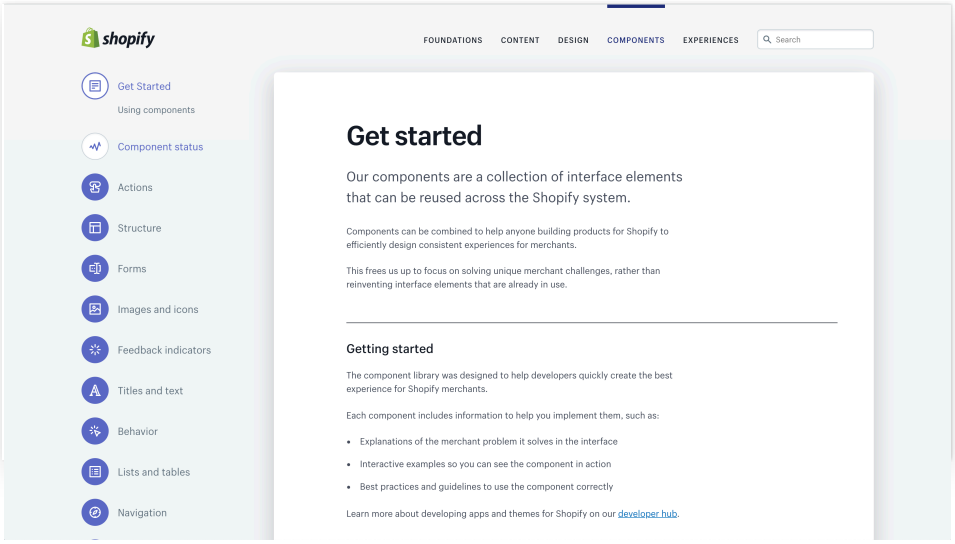
Area	Convention
Background fills	*Base*
Borders, drop-shadows	*Component state*

Option #1

Categorical Organization

For teams migrating to DSM from sticker sheets or shared libraries, this is probably the most familiar structure type. It is also the structure represented in this library.

Example Shopify Polaris



Categorical libraries are organized by component type, from simple components (buttons, forms) to more complex components (cards, navigation). This approach can make it easier for designers to find the components they are looking for while designing.

However, be mindful of how your development team has their web components organized if you work closely with engineers. They may take a different approach, so you'll need to discuss how closely the structure of your DSM library should match their web framework.

Get Started

1 Take the category names from your existing sticker sheet or from your interface audit. Create a folder for each category.

2 Add symbols/components to their relevant folders. Subfolders can be used to capture different states, atomic levels of complexity within a folder, or for subcategories.

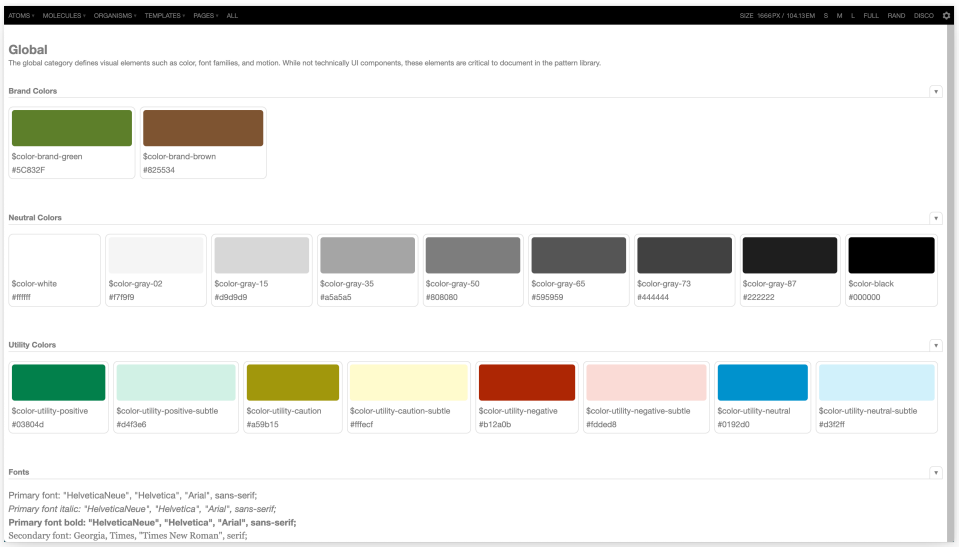
3 To reorganize subfolders, open the parent folder and select the ••• menu on the top right. Select Sort Manually. You should now be able to change the order of subfolders in the tree view.

Option #2

Atomic Organization

This approach is based off of Brad Frost's atomic framework, in which styles, patterns, and components are organized into categories of increasing complexity. Each additional layer builds upon the former—atoms are the most basic styles, molecules are made of combined atoms, organisms consist of multiple molecules, and so on. If you're unfamiliar with this concept, there are tons of great resources available, including Frost's own website.

Example Pattern Lab



This structure is often adopted by developers when organizing their front-end framework. If you work closely with engineers, you may want to adopt this structure as well. It also forces you to think critically about the dependencies and complexities of your components

It's worth noting that it does require a bit more work to organize and maintain, and may make it more difficult to search for components. Make sure your team is aligned on what defines an atom, molecule, etc.

Get Started

1. Create folders based on the atomic structure: atoms, molecules, organisms, templates, pages, etc.

2. Add each component to its relative folder in increasing complexity. For example, when a media card is added to organisms, the text block from the card is added to molecules, a single button is added to atoms, and so on.

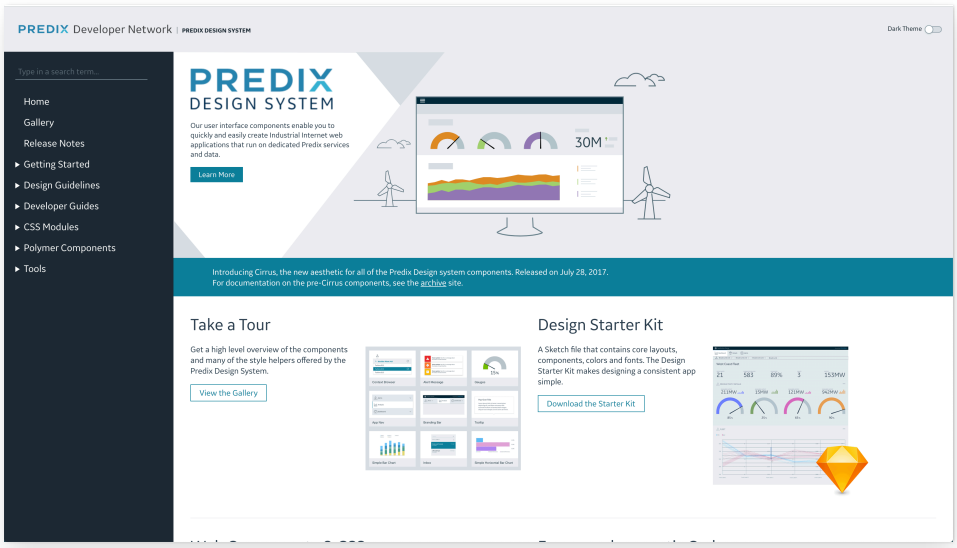
3. Determine which components you want to document fully and which will exist as support. In the web view, each component has a unique url, so you can cross-link documentation as well if it makes sense.

Option #3

Pseudo-Atomic Organization

Perhaps you want to adopt the atomic framework, but make some changes to fit your own needs. This may include increasing or decreasing the layers of complexity supported in your folder structure, or adjusting nomenclature based on your own shared language.

Example GE Predix



Get Started

1. Determine as a team how you want to modify the atomic framework.
2. Follow the steps from the atomic organization to structure your library.
3. For reference, read about how the GE Predix team generated and organized their library.

The basics of atomic design can be maintained, where styles, patterns, and components are nested into different levels of complex categorization, with small adjustments.

Your Structure

After exploring options start planning how you will structure your library. We provided some examples in italics. Don't feel the need to fill up the provided space. Include what you think is most important, it is okay if you need more space.

Atoms

Level 1	Level 2
Colors	*Primary, Status, etc.*
Layer Styles	*Gradients, Borders, etc.*
Text Styles	*Header, Links, etc.*
Icons	*Core, Social, etc.*

Molecules and Organisms

Level 1	Level 2	Level 3
*Components**	*Button*	*Primary*
		Secondary
Inputs	*Dropdown*	
	Text	*Basic*
		Icon

Level 1	Level 2	Level 3

*You can put your molecules and organisms within the component folder or you can add additional folders, some examples are actions, inputs, and navigation.

Documentation

Level 1	Level 2	Level 3
Getting Started	*Designers*	
	Developers	
Design Council	*How we work*	
	Communicate with us	*Who and what*
		SLA's
How we version	*Version strategy*	
	Release notes template	
Purpose and principles		

Level 1	Level 2	Level 3

Documentation

In our experience, good documentation is:

- Consistent—You have a template or content pattern for every component.
- Clear—It's understood by all audiences and is not ambiguous.
- Useful—There's no boiler plating. Everything written is there for a purpose and so quality over quantity is important here.

When creating documentation you want to consider your audience, scope, and how you are going to socialize. Below is a breakdown of what you may want to consider when creating your documentation.

Audience

UI Designer	*Rules, accessibility*
UX Designer	*Interactions, animations*
Front-End Dev	*Spacing, margins*
Other Roles (Marketing, Sales, QA)	*How to use logos, imagery, fonts*

Scope and Types

The design system—think of this as the intro to your product, where your end-users will go to their questions answered about the design system.

What is it/What is its purpose?

- Goals and Objectives
- Business Case

What products/devices does it serve?

- List all the products and devices it will serve

Who manages it?

- Who is the design system team - names, roles

What do I need to know before I use it?

- Purpose and principles
- How do we prototype?
- Naming conventions
- Terminology
- When do we release versions?

How do I interact with it?

- Training guides
- When do I sync with new versions

Component Documention

Section: Overview	Example
Introduction into the atom, component. What's its purpose?	*Our colors are designed to clearly communicate actions, status, and direction within the interface. They serve to make things easier to understand. Follow these guidelines to understand how each color should and shouldn't be used.*

Section: Usage	Example
Add in examples of where this component can be used or should be used. This can include visual examples added in as an image or gif. *As part of the usage, think about adding subsections in, DO's, DON'Ts, corner cases or nuances.*	**When to use** *1. Primary calls to action, links, and attention-grabbing accents* *2. Only once per screen or card* *3. To draw attention to the primary action in view* **When not to use** *1. Headers, paragraph text, anything that could be confused as a link* *2. As the primary outline on buttons or cards* *3. Multiple times in a single view* *4. As backgrounds*

Section: Principles or Guidelines	Example
If practical, add in principles here that help users understand how to use, evaluate, and create	*Follow these guidelines to create clear, consistent interfaces:* **Colors for Communication** *Each color in our designs should have a clear purpose and meaning. If a color does not communicate, it should not be used.* **Design for Accessibility** *Color combinations should be created with color blindness and contrast in mind. We choose contrast over trendiness.* **Color should guide, not distract** *Use color to help clear calls to action stand out. Color is there to inform, not to distract.*

Component Documention

Accessibility	Example
Add in any accessibility rules here that helps your designer and developers create, use, and develop.	*1. Always use white text on top of Primary and Primary Dark.* *2. Always used white text set to semi-bold or heavier on top of Primary.* *3. Primary Light should never be used behind text.* *4. Always use black text on top of Primary Light.*

Theming Rules	Example
Add in notes on how this component can be themed	*1. Theming affects color* *2. We support two themes: dark and light*

Code	What to Include
Add in notes on how to access the component's code	*1. What do devs need to know when building components?* *2. Where does dev documentation live and how will you link that to DSM? i.e. Links tab, links inline components*

Process/Governance	What to Include
Add in notes on how to makes changes to this component (this does not need to be on a component-level, but may be different per element)	*Documentation around the process of your team:* *1. How do I submit new components?* *2. How do I use DSM?* *3. How does hand-off work?* *4. When do I sync with a new version of DSM?* *5. How do I submit bugs?*

Socialization	What to Include
Add in notes on how to use and share this component (this does not need to be on a component-level, but may be different per element)	*1. Success stories* *2. Stats/figures* *3. Communication plan*

Your Component Documentation Template

For your component documentation, our recommendation is to create a template that can be applied to each component ensuring consistency.

Section	Description
Overview	
Usage	
Principles or Guidelines	
Accessibility	
Theming Rules	
Behavior	

Developer Checklist

A true design system cannot exist without code. The purpose of these systems is not the workflow so much as it is the output.

Answer the below questions. Based on what we have already completed, you may have already answered these questions on another document. If so, feel free to expand on it or remind yourself where to reference your answer elsewhere.

Questions	
How do designers and developers decide shared naming conventions?	
What permission level does each developer need within DSM?	
What is dev involvement in component request(s) and approval(s)?	
How are tickets added to the backlog when a component or style requires updates?	
What is the timeline or SLA for coding a component?	

Questions:

If using Storybook, how do developer versions relate to DSM versions?	
What does the passoff/handoff look like from design to development?	

Development Setup Checklist

- [ ] Get developer team buy-in on design system
- [ ] Align on naming convention for colors
- [ ] Align on naming convention for text styles
- [ ] Review and align on design system structure
- [ ] Add developer documentation (how devs consume assets that are hosted, downloaded or integrated)
- [ ] Agree on versioning strategy for your design system and any coded components
- [ ] Link code and component (through any integration/embed/link)
- [ ] Test initial coded component
- [ ] Add code for all components
- [ ] Add additional developer resources (e.g. comparing screenshots of components to previous versions)
- [ ] Do visual test for UI bugs (e.g. comparing screenshots of components to previous versions)
- [ ] Do unit test for UI functionality
- [ ] Ship regular updates
- [ ] *Optional: Look into integrations with open source release management*
- [ ] *Optional: Look into continuous integration tools for ongoing maintenance*

Contributors Process Guide

Now that you have planned out your process, we recommend documenting a contributors guide and sharing that in your design system. This way your users and stakeholders will understand the process for contributing content. Use the Contributors Example, it's its own stand alone worksheet

Step 1	*What should the contributor do prior to submitting/suggesting the component/design element?*
Step 2	*When submitted is there anything the contributor should include. We recommend:* • Name/role/team of person requesting the change • Net new component/documentation or change to existing? • Name of the component (and URL if already in DSM) • Description of the change/addition • If new, why doesn't a current component fit the need? • Is this reusable or for a single-use case? • Does it pass accessibility rules? • (Optional) Visual of the image or an attached Sketch file
Step 3	*How will the Design system makers contact the contributor and let them know the decision?*

Step 4	*Will the contributor be a part of the process to get the component/design element into the system? If so, how? (for example documentation)*
Step 5	*What is the process for getting the design element coded?*
Step 6	*Is there going to be any additional QA for the design or coded, outside of what the makers did when reviewing? If so, what?*
Step 7	*Will the contributor be a part of the sharing/versioning/communication process to share the update? If so, how?*

Contributors Example

When having an open contribution policy for your DSM, it's best to put rules in place that standardize what and how to submit to the Design System. Below is an example of what that contribution process could look like for components in your DSM. This may vary based on your team and governance.

In this scenario, we are using a Slack channel within the company, called #dsm, to communicate changes. Depending on your company's platforms, you could use email, team meetings or another messaging service in place of Slack.

1. Define the need

Before submitting a component, make sure to clearly define the problem it is solving.

- Define the purpose of the component
- Message the team in the #dsm Slack channel to confirm there isn't an existing solution

2. Create a proposal

The best components are the most reusable. All components are open to use by any team, so they must be easy to understand and flexible for different scenarios.

Proposals should include:

- Google Doc format (update based on your company's tools)
- Who is requesting it (include any designers and developers involved)
- Component name
  - If new: recommended naming. *Be sure to name based on purpose rather than description (ie. button primary not button-blue)
  - If current: name of the existing component
- Description of the change or addition
- Any research done to support the change
- Visual example of the change
  - Screenshots are sufficient
  - Link to the corresponding Sketch file
- Any states or variations associated
- Where in the DSM folder structure you believe this component belongs
  - Does it live within an existing folder or is a new folder required?
- Corresponding Documentation
  - If new, what documentation and best practices need to be included with this component in the DSM?

- Why it's needed and why none of the existing components fill the need
  - What are the advantages of this component over others?
  - Does a similar component already exist?
  - Have you tested this component for use cases across products to ensure it is something that can be used globally by the team?
- Does this component fit within our accessibility standards (WCAG AA, color contrast ratio)

3. Refine

Ideas often need refinement. You'll work with the DSM team to refine your component to ensure it's the best it can be.

- If denied, the DSM team will provide feedback explaining why
- If accepted, the DSM team may follow up via Slack or schedule meeting time to review, ask questions, make recommendations
  - Add iterations to the proposal doc based on feedback

4. Upload

Once you've finished working with the DSM team and they have approved the proposed component, it's ready for DSM.

- Push the component to the shared draft of DSM
- Add the corresponding documentation to the component in DSM
  - This can be done via the Sketch plugin or from the web monument view

5. Share

Now that the component has been added to DSM, it's ready to be circulated with the larger team.

Your DSM team will:

- Release a new version of DSM that includes the component and corresponding release notes
- Sync with developers to notify of any breaking changes and to make sure corresponding code and developer documentation is added

You're Done!
Take time to celebrate. Thanks for contributing.

Version Strategy and Communication Plan

Versioning your design system will help you communicate changes to the larger team(s).

Start by creating documentation to support the team members who will be releasing new versions of the system so that they can continue to provide consistent releases over time.

Then communicate this out to the larger product team. When a designer or developer sees that a new version number is available, they have a baseline of understanding of the impact of change before they even open up the release notes.

Begin to plan out your versioning strategy. We have included questions to get you thinking and some examples to give you an idea of what other customers have done.

Version Strategy

What is versioning?	*What does versioning mean to you?*
Who versions DSM?	*Who will update design components?* *Who will release notes?* *Who will publish the version?* *Who will update coded components?* *Is there anyone else that should be part of versioning? If so who and what should they do?*

When do we version?	*Do you have a set cadence, will it depend on what has been submitted to the system for updates, or something else?*
How do we version?	
What versioning format will you use? Info Semantic Versioning 1.0.0 The first number is the Major—a breaking change (e.g. a rebrand, new feature set) The second number is the Minor—a non-breaking, but noteworthy change (e.g. new components, updated group of styles) The third number is the Patch—a small request or bug fix (e.g. fix to a buggy component edit of a text style)	
If Semantic, how will your team define your Major, Minor, and Patches?	*Major:* *Minor:* *Patch:*

Version History

INFO

To access your library's version history, and keep track of the changes, click "view on web" to open the library's web view. In the bottom left corner, click on the version name to view the full version history.

With version control, you can revert or compare every design update to a previous version. Hover over any component's side-by-side visual comparison and click "more details" to see the granular difference.

Version Alignment

How will you keep everything aligned and determine what goes into what version?

Example

We will include links to JIRA tickets, InVision boards, and additional points of logged comments/decisions. This will ensure full transparency around why revisions were made.

When designs are finalized but not built yet, they remain in the "Pattern" section.

Engineering Alignment

How will you engage your engineering/development partners? Will you wait to publish the version until they have built the component or publish the designed component then have them build the component?

Now that we know how our makers will handle versioning of the system, we need to plan out how we will communicate this with the great team.

Communication Plan

Outside of DSM, how will the new version be announced?	
Who is sending out this communication?	
Who is included in this communication?	*Certain teams, leaders, individuals, or everyone that is part of a business unit or department?* **Note: for now think of the product with the components you are working on currently, we will circle back to communication in scaling as it may change as your design system grows**
What's announced?	
Are there resources provided? If so, what? Example Links to DSM/version history, links to new components, links to code updated, etc.	*Are there any resources included in the communication?*

Version Control Template

This document is formatted in markdown for your DSM commit message. Simply copy and paste the structure for ideal formatting. Delete any title sections that aren't being used for a particular release.

```
# What's New
## Documentation
### Item 1
- Details of Item 1
- More details of item 1

## Styles
### Item 1
- Details of Item 1
- More details of item 1

## Components
### Item 1
- Details of Item 1
- More details of item 1

## Icons
### Item 1
- Details of Item 1
- More details of item 1

# Updates
## Updated Documentation
### Item 1
- Details of Item 1
- More details of item 1

## Updated Styles
### Item 1
- Details of Item 1
- More details of item 1

## Updated Components
### Item 1
- Details of Item 1
- More details of item 1

## Updated Icons
### Item 1
- Details of Item 1
- More details of item 1

# Bug Fixes
## Fix 1
- Details of Fix 1

# Deprecated
## Item 1
- Details of Item 1

# Assets
[Link Name](Link URL)

# Notices
- Sketch 53 required to correctly use added symbols
```

DSM Training Outline

As you prepare to release your library, you will want your users to know how to use it. We recommend creating a training that will teach your users how to navigate your design system, how they should incorporate your system into their workflow, and how to use InVision DSM.

Below we have provided an outline for recommended points to cover.

Designer focused

1. Accessing the Web Portal
   - Viewing documentation
   - Reading release notes
   - Viewing change list
2. Using the Sketch Plug-in
   - Accessing DSM
   - Navigating between libraries
   - Designing with DSM
   - Pulling updated

Developer focused

1. Accessing the Web Portal
   - Viewing documentation
   - Reading release notes
   - Viewing change list
2. Using the Developer API
   - Pulling design tokens
   - Downloading the Icons
   - Viewing live components/link/embed code
3. DSM and Inspect
   - What DSM information will show in Inspect

Stakeholder focused

1. Accessing the Web Portal
   - Viewing documentation
   - Reading release notes
   - Viewing change list

Product Mapping

As you complete the program, it is time to think big again. You have been working with one product and just a small amount of components. Now we want you to define your final product. Plan out your full design system, what products you will want to build in libraries, how you want to separate them (if at all) by theming or platform?

As you work through building each of these libraries, go back and use those templates, worksheets, and checklists.

1. List all products currently supported in the left column, along with any future products that are on the horizon.
2. Mark whether each product has its own branding or theming, separate from what you would consider the “global” design theme.
3. Acknowledge which platforms these products are designed for. For native mobile apps, please note if it covers iOS, Android, etc.
4. The review and answer the below questions.

Products	Branding/Theming	Platform
Example Product	*Dark Theme*	*iOS*

Additional points to consider	
Do you need additional executive sponsors for your additional libraries? If so, who?	

What other teams/makers should you get involved?	
What do you want to keep consistent across your libraries versus what would you expect to change?	
What is your plan to collaborate across teams for the design system?	

MVP Release Checklist

When releasing your first full version of your design system you want to make sure you have completed everything. Throughout the program, you have worked in a few different worksheets. Follow the below checklist to make sure you have completed everything prior to releasing your first version.

MVP Release Checklist

- [ ] We defined our design system users
- [ ] We defined our design system stakeholders
- [ ] We have written and uploaded our purpose and principles to the design system
- [ ] We aligned on naming convention between design, dev, and other teams
- [ ] We have added in all our colors, text styles, layer styles, icons, and components
- [ ] We have added in any templates, grids, design elements for the product
- [ ] We structured our design system in a way that is easy to navigate
- [ ] We have added in documentation for all design elements
- [ ] We have linked code to our components
- [ ] We have added developer related documentation
- [ ] We have determined how we will handle request from contributors
- [ ] We have documented the process for contributors to add request
- [ ] We agreed on a governance model
- [ ] We have a version strategy in place
- [ ] We have at least one executive sponsor
- [ ] We have begun to communicate and build interest for the system
- [ ] We have taught the team how to use the design system and DSM
- [ ] We have made sure the users know where to go for questions and assistance
- [ ] We have beta tested the system

New Design Maker Checklist

To help set you and your team up for success, here is a checklist for getting new makers comfortable with InVision and your design system. We included additional spaces for you to customize your onboarding checklist.

Designer Maker checklist

- [ ] Assign the appropriate role and permissions in Enterprise and DSM
- [ ] Share any projects/design systems with them
- [ ] Make sure they download Craft for Sketch — **Download Craft**
- [ ] Attend our DSM training session
- [ ] Attend an Enterprise training session
- [ ] Introduce them to your design system
- [ ] Attend/complete your design system onboarding

New Designer Checklist

To help set you and your team up for success, here is a checklist for getting new designers comfortable with InVision and your design system. We included additional spaces for you to customize your onboarding checklist.

Designer checklist

- [ ] Assign the appropriate role and permissions in Enterprise and DSM
- [ ] Share any projects/design system with them
- [ ] Make sure they download Craft for Sketch **Download Craft**
- [ ] Attend an Enterprise training session
- [ ] Introduce them to your design system
- [ ] Attend/complete your design system onboarding
- [ ] Complete/watch your design system training
- [ ] Make sure they understand who to go to with questions or support for the design system

New Developer or Engineer Checklist

To help set you and your team up for success, here is a checklist for getting new developers and engineers comfortable with InVision and your design system. We included additional spaces for you to customize your onboarding checklist.

Developer or Engineer checklist

- [ ] Assign the appropriate role and permissions in Enterprise and DSM
- [ ] Share any projects/design systems with them
- [ ] Read the Developer and Engineer guide
- [ ] Introduce them to your design system
- [ ] Attend/complete your design system onboarding
- [ ] Complete/watch your design system training
- [ ] Make sure they understand who to go to with questions or support for the design system