Writing Assignment 2

Writing Online Content

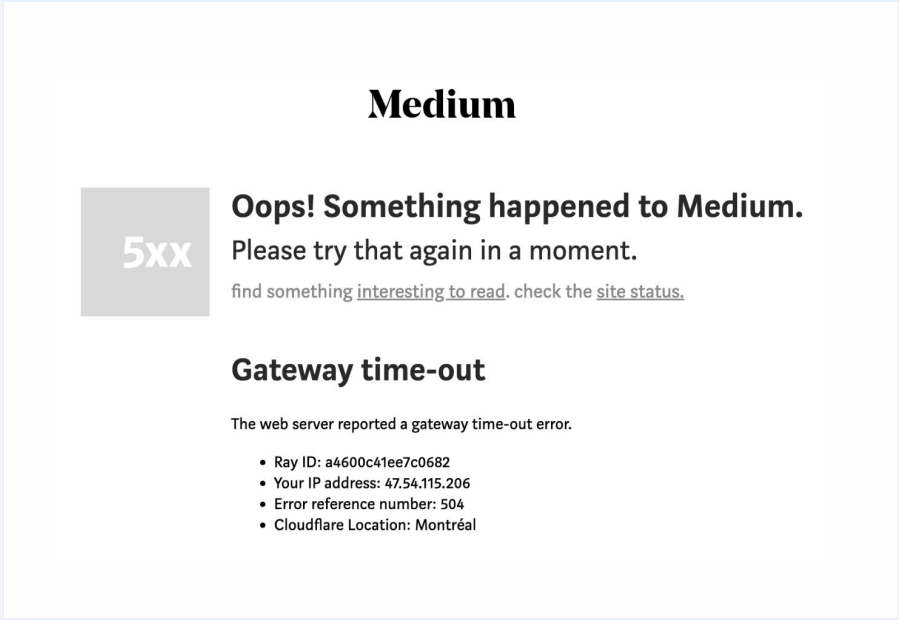


Image source: Medium

OISE

Oct 14, 2026

Transforming content into an engaging self-study digital lesson

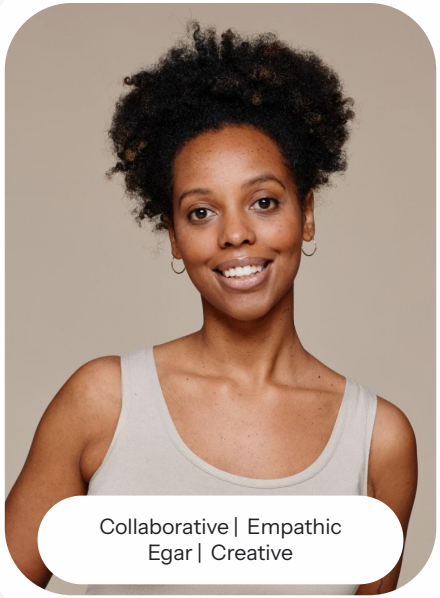
Content type:
1. Build your own custom solution

Project at a glance

- **Approach:** Video, presentation, and exercise templates
- **Project title:** Designing Error State Interfaces
- **Audience:** Designers
- **Problem statement:** Design a better way for designers and engineers to better anticipate and collaborate on error states so that error states are well designed for end users in time for engineering hand off.
- **Goal:** Confidently plan for, collaborate on, and design solutions for thoughtful error state experiences that help users easily identify, diagnose and recover from expected and non-standard error states in user interfaces.
- **WIIFM:**
    - Improving error state experiences for end users
    - Avoiding engineering timeline delays
    - Better anticipating engineering needs related to error state designs
    - Centralizing error states for unified experiences

Persona

“The best thing about design is how it supports people.”



Catherine

UX Product Designer

Loves digital media & peer guidance

Need statement:
As a designer, I need a better way to design error states so that our users can easily identify, diagnose, and recover from them.

Goals:
Design error states before engineering begins implementing the designs so that she is not pulled back into a project while the engineering team is working on our late night pull request.

Background:
New hire completing onboarding before she is placed within her dedicated product team

Education:
Degree in UX Design

Motivations:

- Designing thoughtful experiences for users and developers
- Managing time efficiently
- Reducing support costs for the business

Obstacles:

- Working ahead of engineering involvement in design projects
- Short timelines for design delivery

Learning series

Designing Error State Interfaces



Image source: Medium

Overview

Designers will confidently plan for, collaborate on, and design error state experiences that help users easily identify, diagnose and recover from expected and non-standard error states in user interfaces.

What to expect

- Improve error state experiences for users
- Anticipate engineering needs to avoid delayed engineering timelines
- Build for unified experiences

“Error messages should be expressed in plain language (no error codes), precisely indicate the problem, and constructively suggest a solution.”
–Jakob Nielsen

Agenda

01

Intro to error states

- Why good error states matter
- What is and what isn't an error state
- Anatomy of an error state
- Footprint of an error state

02

How to guide

- Planning for error states
- Collaborating to identify error states
- Putting it all together

03

Join the community

- Share and follow us on social media

Intro to error states

Why good error states matter

45% of users

lose track of or completely forget their last task when encountering an error state —IDxF

25% of users

will not return to the task or application when encountering an error —IDxF

~48 hour

support turnaround times for most IT support requests

~2 to 4 hours

spent investigating complex error state support requests

$75 to $200 per hour

for billable IT support time

Poorly designed errors = massive cost + usability impacts

Intro to error states

Intro to error states

What is an error state

Standard http error states are subcategories of HTTP response status codes that indicate whether a request has been successfully completed.

HTTP error response status codes include:

- 400 to 499 - client side responses
- 500 to 599 - server side responses

Note: Other standardized errors can come from third-party tools, operating systems, APIs and more.

Non-standard response codes are established by the creator of the software solution and do not follow a standard protocol like expected HTTP response codes do.

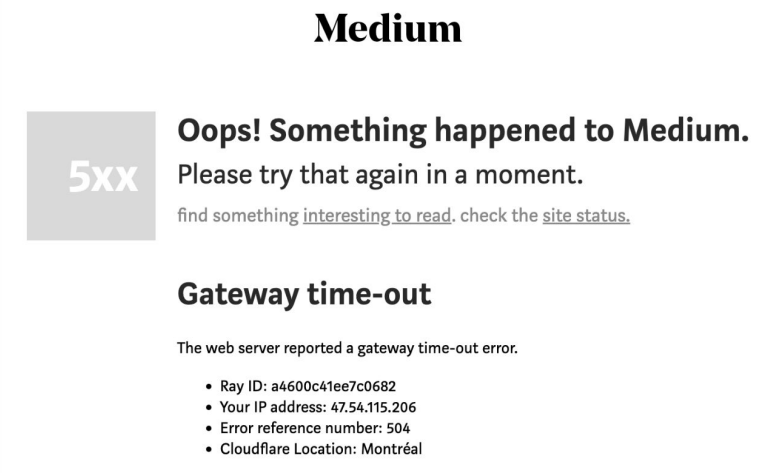


Image source: Medium

Intro to error states

What **isn’t** an error state

Empty states, **loading states**, and **first-time use states** are not the same thing as error states.

If a user is running a search that returns **zero results**, or a **software requires setup** for content to be populated these types of states should be considered separately and should be coded into the components separately as well.

For engineers, **hijacking a component’s loading state to display an error state is considered bad practice** because this will lead to technical debt in your component library.

Advocate for supplementing your design system components with appropriately designated error states.



Image source: Google

Intro to error states

Anatomy of an error state

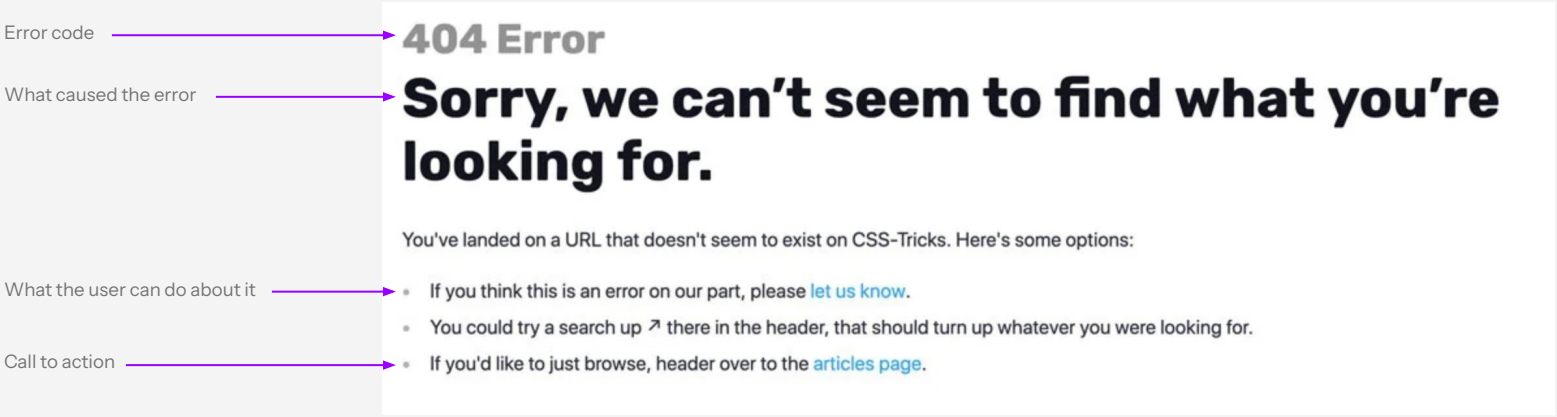


Image source: CSS Tricks

Decide which parts of the error state anatomy to include or exclude depending on the location and size of your error state.

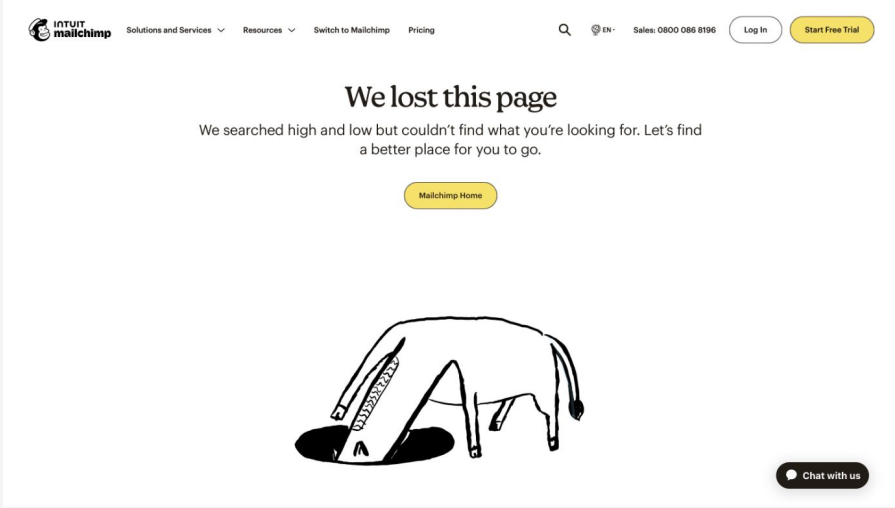
Intro to error states

Footprint of an error state

Error states can be

- full page,
- partial page,
- full component,
- partial component,
- or micro.

Make your error states lean by creating rules to fit your components and design system. Omit parts of the anatomy of your error state to fit its intended use and location.



Above image: Full page error state from Mailchimp

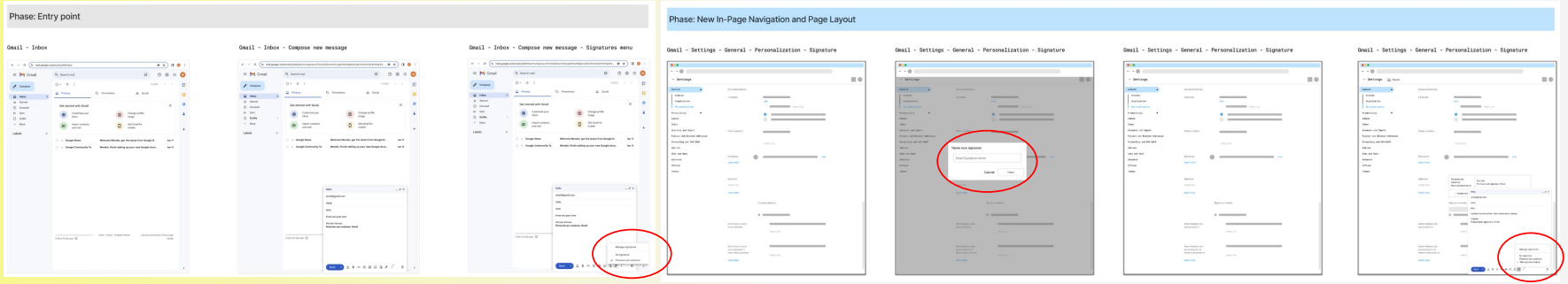


Above image: Partial component error state from Google

How to guide

How to guide

Planning for error states



Above image: Google setting page redesign by Jenna Lawrence

When designing your happy paths, take note of when a request is submitted on the client side, servers side, or via API etc. These moments are where you will need to plan for error states.

Designers build CRUD experiences (create, read, update, delete, run, duplicate etc.) These happy path flows are referenced differently by our engineering partners. They use words like GET, PUT, POST, DELETE, and RUN.

Partner with your engineering team to identify errors, especially when designing for non-standard error states. Anticipate that at some point, you will encounter completely unique scenarios.

Putting it all together

Collaborating to identify error states

Flow name:
Flow description:

Error code	Title	What caused this error	What the user can do about it	Call to action	How to reproduce the error

Above image: Example conversation starter template by Jenna Lawrence

Pro tip:
Mozilla's documentation on http response codes is a helpful reference for identifying your errors. Focus on errors listed in the 400 and 500 sections.

Work with engineering partners to identify error states in your happy paths. Your engineering team will identify error state use cases that are completely unique to your tech stack.

Unhelpful and reused language in error states are often mistakenly used at the last minute to satisfy a late night pull request. Our goal is to give engineering everything they need so that they can accurately estimate and allocate their engineering effort.

Use the Conversation Starter template in partnership with your engineers to asynchronously co-create a list of error states. This list will be your guide in delivering impactful and effective error states.

Putting it all together

Using the components

Leverage appropriate error state components for your use case based on your design system's guidelines.

If your design system doesn't have guidelines, you can begin establishing rules by creating mock ups and socializing them with other designers to build consistency across your teams.

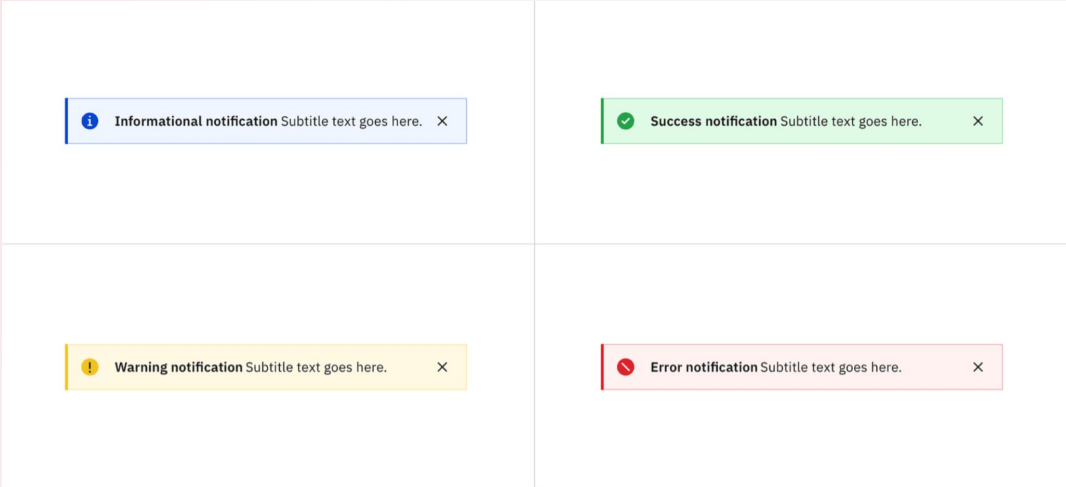


Image source: IBM's Carbon Design System notifications component

Putting it all together

Writing the copy

Using the conversation starter, create a finalized list of your errors with copy that follows your organization's editorial guidelines.

If you need help you can prompt your favourite AI tool to get started.

Here is a prompt to get you started.

Use your polished error states and visual design templates to finalize your designs with engineering for their input and hand-off your work.

"I am designing error states for a feature where we are designing a better way for USERNAME to do ABC so that they can XYZ.

Using the attached error state matrix, please help me by writing some polished error states that follow best practices for error state messaging."

We value your input



Thoughts?

Questions?

Feedback?

Join our community to share and follow us on social media to stay informed on the latest trends on designing error states.

@socialmediahandle
errorstatescommunityurl.com

Thank you



Questions? Contact
artistjenna@gmail.com