STRETCHLEARN · WORKBOOK

Wireframing — Workbook

This workbook turns the course into a real wireframing project. You will pick one feature and carry it through the same steps a UX designer follows: choosing fidelity, ranking content by hierarchy, building screens on a grid in a real tool, annotating behavior, wiring a clickable lo-fi prototype, and running a critique. Work each section as you finish the matching course module, and use the templates to keep your fidelity decisions, hierarchy ranking, screen-state matrix, and annotation log in one place. Keep everything in grayscale: the whole point is to test structure before paint.

What Wireframing Is and the Fidelity Ladder

Choose the feature you will design, fix the fidelity for the decision you are making, and commit to the grayscale rule before you draw a screen.

Worksheet: Define Your Feature and Fidelity

Choose one feature you will wireframe throughout this workbook (keep it small, two to five screens). Fill in each field so every later decision has a clear reference point.

Feature name and one-sentence purpose

Primary user and the single most important task they complete

The 2 to 5 screens this feature needs (e.g. list, detail, form, confirmation)

Who will review these wireframes (yourself / a developer / a non-design stakeholder)

Starting fidelity for that audience (sketch / low / mid) and why

Canvas size you will design on (e.g. desktop 1440 px or mobile 375 px)

Exercise: Sketch Three Layouts in Ten Minutes

On paper or in Excalidraw, sketch three different rough layouts for your most important screen, two minutes each. The goal is quantity and disposability, not polish, so do not erase, just start a new sketch.

• For each of the three sketches, what is the single most prominent element, and is it the most important one for the user's task?

• Which layout best answers the three wireframe questions (what is on the screen, what matters most, how it connects)?

• What did you learn from a layout you rejected that you will carry into the chosen one?

Checklist: Foundations Readiness

[ ] I can state the three questions my wireframe must answer (content, hierarchy, flow)
[ ] I have chosen one small feature and listed its screens
[ ] I picked a deliberate starting fidelity matched to my reviewer
[ ] I am committing to grayscale, with at most one accent color meaning interactive
[ ] I sketched at least three throwaway layouts before settling on one
[ ] I can explain why fixing structure now is cheaper than fixing it after high-fidelity

Information Hierarchy and Layout Structure

Rank your content on purpose, group it with Gestalt principles, and put it on a real grid with 8-point spacing.

Worksheet: Rank the Content Before You Draw

For your main screen, list every content block and control, then assign a priority. The layout must later make the highest-priority item the most visually prominent.

List of every content block / control on the screen

Priority order, numbered 1 (most important) to last

The single primary action for this screen (the one dark/filled element)

Reading pattern this screen should use (F-pattern for dense, Z-pattern for sparse) and why

How you will signal the top item's importance without color (size / weight / contrast / whitespace / position)

Exercise: Fix the Grouping With Gestalt

Take a busy screen (your settings or form screen, or one from an app you use) and improve its structure using proximity, similarity, common region, and alignment, with no new elements and no color.

• Where is spacing currently even everywhere, leaving the eye unable to tell which items are grouped, and how will you tighten within groups and widen between them?

• Which related items should sit inside a shared container (common region) to read as one unit?

• Do all elements of the same type (every button, every row) look identical (similarity), and where do they not?

• What single left edge can you align the content to so the screen reads as orderly?

Worksheet: Set Up Your Grid and Spacing Scale

Define the structural skeleton for your screens. Fill in each field so spacing is decided, not eyeballed.

Grid: columns, margins, and gutters (e.g. desktop 12 columns / 24 px gutters; mobile 4 columns)

Spacing scale you will use (multiples of 8: 8, 16, 24, 32; 4 only when truly needed)

Spacing between a label and its value / field

Spacing between separate content groups or sections

How many columns each major block spans (e.g. card spans 4 of 12)

Checklist: Hierarchy and Layout Readiness

[ ] I wrote the content priority order before laying anything out
[ ] The most important element is the most visually prominent on the screen
[ ] There is exactly one clear primary action per screen
[ ] I matched the layout to a reading pattern (F for dense, Z for sparse)
[ ] Related items are grouped by proximity, with clear space between groups
[ ] Everything sits on a column grid with 8-point spacing, not eyeballed gaps

Building Wireframes in the Tools

Pick the right tool, build a small reusable component kit, and assemble each screen including the states beginners forget.

Worksheet: Choose Your Tool Stack

Decide which tool serves each stage of your project so you neither over-tool early sketches nor under-tool the build. Fill in each field.

Idea / sketch tool (paper, Excalidraw, FigJam)

Build tool for low/mid-fidelity (Figma / Balsamiq / Whimsical) and why

Whether you need deliberate roughness for a stakeholder (and if so, where Balsamiq fits)

Tool you will use for the clickable prototype

One feature of the build tool you will deliberately NOT use yet (to avoid over-polishing)

Exercise: Build Your Starter Component Kit

In your build tool, create reusable grayscale versions of the core elements (or duplicate a free Figma Community wireframe kit) so every screen stays consistent. Make each one a real component/symbol, not a one-off.

• Did you build a filled primary button and an outlined secondary so action hierarchy is built in?

• Do your image placeholder (crossed rectangle) and avatar placeholder (circle) read instantly as stand-ins?

• Which elements vary randomly across your screens right now, and how will a shared component fix that?

Worksheet: Map Every State of Each Screen

For each data screen in your feature, plan the four core states, not just the happy path. Fill in each field per screen (repeat for each screen).

Screen name

Default state: what normal content looks like

Loading state: skeleton placeholder vs spinner

Empty state: the message and the first action offered when there is no data

Error state: the plain-language message and how the user retries

Checklist: Build Readiness

[ ] I chose a tool per stage and kept the earliest sketches tool-light
[ ] I have a reusable grayscale component kit (button, field, nav, card, row, placeholders)
[ ] Every primary button looks identical across screens; every list row is consistent
[ ] I identified the common pattern each screen is built from (list, detail, form, etc.)
[ ] Every data screen has a default, loading, empty, and error state designed
[ ] My worked feature (e.g. search) exists as separate frames for each state, not one frame

Annotation, Lo-Fi Prototyping, and Critique

Annotate behavior and logic, wire a clickable lo-fi prototype, test it, and run a focused critique before handoff.

Exercise: Annotate One Screen for a Stranger

Take your busiest screen and annotate it so someone who has never heard you describe the project could build it. Use numbered callouts in the margin and keep notes short and specific.

• For each interactive element, what does it do and where does it navigate?

• What states or conditions change the screen (logged-in only, error appears when input is invalid), and what content rules apply (max 280 characters, shows 10 most recent)?

• Where are you tempted to annotate something obvious (labeling a button 'button'), and how will you cut that noise?

Worksheet: Plan and Build the Clickable Prototype

Wire your screens into a low-fidelity clickable prototype that walks the real task. Fill in each field before and after building.

The core task a user must complete end to end (e.g. find and buy a product)

The main path: list the screens in order, and which element links to the next

Key alternate paths to wire (e.g. empty results, error, back)

Transition style (instant or simple dissolve) and where back behavior matters

What felt awkward when you clicked through it yourself (missing step, dead end, one screen too many)

Exercise: Test the Prototype on Real People

Give two or three people a goal and watch silently as they attempt it on your grayscale prototype. Do not guide them; note where they hesitate, mis-tap, or get stuck.

• What goal did you give each tester, and where exactly did they hesitate or tap the wrong thing?

• Did anyone fail to find the primary action, and on which screen?

• Because it is grayscale, did feedback stay on flow and structure rather than colors? What is the top flow fix?

Checklist: Critique and Handoff Readiness

[ ] Annotations explain behavior, states, content rules, navigation, and logic for every screen
[ ] Annotations use a consistent convention (margin notes, numbered callouts) distinct from the UI
[ ] A clickable prototype covers the core task and key alternate paths
[ ] I tested the prototype on at least two people and captured where they got stuck
[ ] I ran a critique that framed the problem and asked specific questions, and I captured feedback without defending
[ ] I triaged feedback into real problems vs preferences and fixed the recurring structural issues
[ ] Every screen state is present, all links work, and frames are named and organized for handoff

Your Action Plan

1. Pick one small feature (2 to 5 screens), write its purpose and primary task, and choose a starting fidelity for your reviewer
2. Sketch three throwaway layouts of the main screen on paper or Excalidraw, then choose one direction
3. Write the content priority order for each screen and name the single primary action
4. Set up a column grid (12 desktop / 4 mobile) and an 8-point spacing scale in your build tool
5. Build a reusable grayscale component kit (or duplicate a free Figma wireframe kit) so screens stay consistent
6. Lay out each screen on the grid, grouping content with proximity and ranking by visual weight, all in grayscale
7. Design the loading, empty, and error state for every data screen as separate frames, not just the happy path
8. Annotate every screen for behavior, states, content rules, navigation, and logic using numbered margin callouts
9. Wire a clickable low-fidelity prototype of the core task plus key alternate paths and walk it yourself
10. Test the prototype on two or three people, run a focused critique, triage the feedback, fix the structural problems, and hand off clean named frames