



S I V I Q U A N T L A B S

CHALLENGE 01 · FRONTEND + 3D

2D DIELINE → 3D BOX

Read a flat cutting layout. Fold it into a closed box. Show us the math move.

STACK	DELIVERABLE	ELIGIBILITY
Your choice + a 3D lib	~60-sec screen recording	Doers. Fast learners.

WHO WE ARE

We build deep-tech 3D systems and sovereign AI agents — engineered from first principles, not wrappers over someone else's API. We work with customers worldwide and ship things that are measurable, auditable, and fast. This challenge is a small slice of the kind of problem we solve every day.

THE CHALLENGE

Build a web application that takes a **2D box cutting (a dieline)** and turns it into a **3D box** — and folds it closed in front of you.

A dieline is the flat, unfolded cutting layout of a package. Your job: read that flat layout, understand which panels fold where, and assemble it into a closed 3D box with a **visible folding animation**.

WHAT IT MUST DO

- 01 **Upload** — let me upload a 2D dieline file from the browser.
- 02 **Parse** — read the panels and fold lines from that file.
- 03 **Build** — construct the 3D box geometry from those panels.
- 04 **Fold & close** — animate the flat layout folding into a closed box. The closing must be **clearly visible** — I want to watch it fold, not see "before" and "after".
- 05 **View** — let me orbit / rotate / zoom the finished box.

Use **any** frontend stack (React, Vue, Svelte, vanilla) and **any** 3D library (Three.js, Babylon.js, ...). We don't care what you pick — we care that you make it work and can explain why.

THE SAMPLE FILE

Bundled with this brief is a real dieline — [SAMPLE_DIELINE.PDF](#) [SAMPLE_DIELINE.PNG](#) — a straight tuck-end carton, the standard cosmetics / pharma box shape.



DIELINE PRIMER

- Dielines are **color-coded by line type**. Cut lines are the boundary; crease lines tell you where panels hinge.
- The flat layout is a connected set of rectangles: a central band of four wall panels (front, side, back, side) plus a glue flap, with top and bottom tuck & dust flaps hanging off the walls.
- "Folding it up" means rotating each panel about its shared crease line until the box is closed.

The bar: make it work end-to-end on this sample box. You may make reasonable assumptions about the box type to get a clean result. Handling arbitrary, unseen dielines is a **stretch goal**, not a requirement.

WHAT YOU SEND

One screen recording, roughly **60 seconds** total. Talk over it — we want to hear how you think.

- **First ~30 sec — Demo.** Upload the dieline, show the box build and the folding-closed animation, rotate it.
- **Next ~30 sec — Code walkthrough.** How it's organized, where the fold logic lives, what the hard part was.

WHO CAN APPLY

Anyone who's finished a degree. The degree isn't the point — we want a **doer**: someone who picks up a new library fast, builds the thing, and explains it. Never touched Three.js and learned it for this? That's exactly the signal.

WHAT WE GRADE

- **It works** on the sample dieline.
- You **understand your own code** — the fold logic, not "the library did it".
- You **learn fast**. Clean > over-engineered.
- **Clarity** in your 60 seconds.

STRETCH GOALS · OPTIONAL

- Parse cut vs. crease generically — handle a dieline you weren't given in advance.
- Realistic materials / lighting; map the dieline artwork onto the box faces.
- Let the user scrub the fold (slider: flat → closed).

SUBMIT

Email your ~60-sec video + a link to your code

SUBJECT: BUILD CHALLENGE — <YOUR NAME>

connect@siviquantlabs.com

Build something real, show us it works, tell us how you did it.