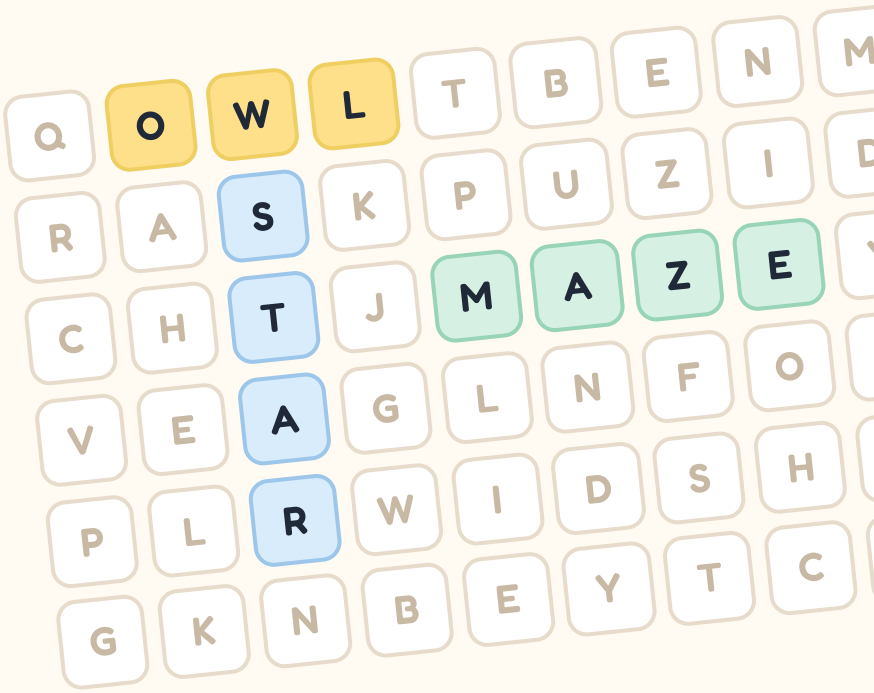


TECHNICAL CASE STUDY · NOVEMBER 2026

Puzzld

Kid-safe printable puzzles on any topic, built in two days with a spec, eight plans and a test gate

How a Turborepo of puzzle engines, a print-first PDF renderer and a layered Claude safety pipeline became a live product for teachers, daycares, Sunday schools and families.

puzzld.codefrontlabs.com | Next.js 16 · Firebase · Stripe | Claude Sonnet 5.5 + Opus 5.5

Codefront Labs

At a glance

Puzzles that have to be right the first time

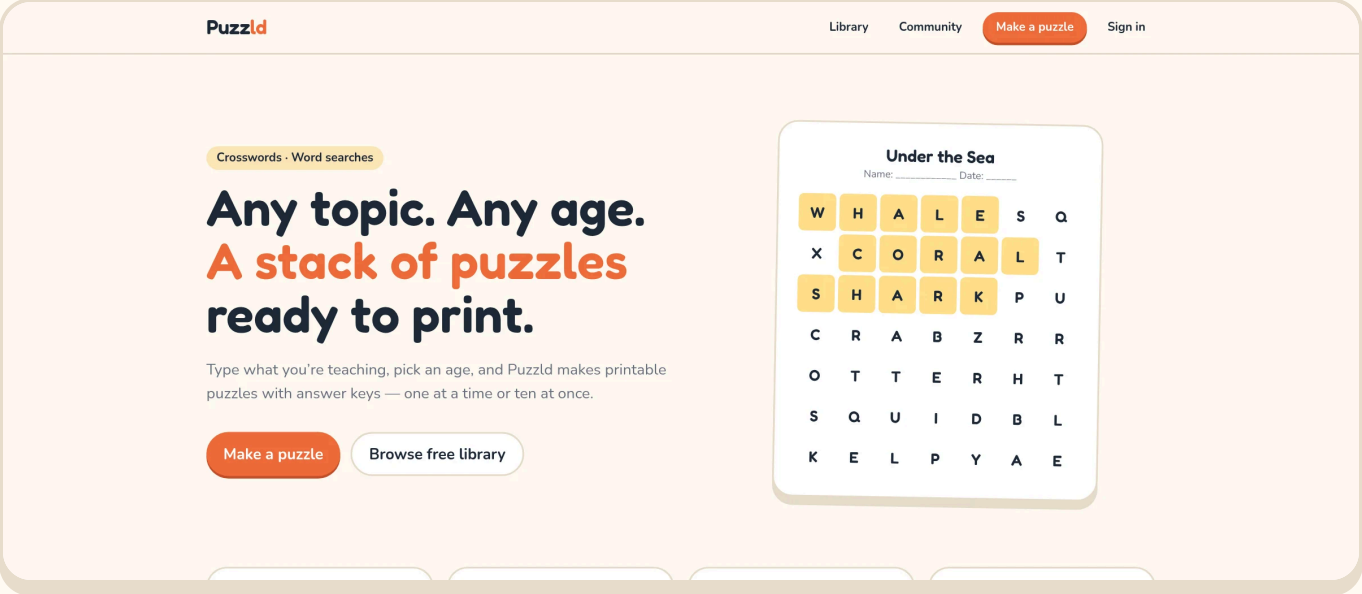
A teacher types this week's topic, picks an age from Pre-K to adult, and gets a stack of printable puzzles with answer keys. The puzzles go straight in front of children, so a wrong answer key, an unreadable word or an unsuitable topic isn't a small bug. It's the product failing.

2
days from design doc to live product, 124 commits

10
puzzle and worksheet types from one pipeline

570
automated tests, plus 40 database tests in CI

4x
minimum margin over measured AI cost, enforced by a test



The live Puzzld homepage. Every puzzle on the site, including the example word search, comes out of the same engines described here.

What it makes

Word searches, crosswords, story fill-ins, mazes, dot-to-dots, spot the difference, odd one out, read-aloud tales, reading comprehension and writing prompts. Packs bundle them for parties, trips and summer.

Who it's for

Teachers, daycare staff, Sunday school leaders and parents. Adults hold the accounts. Children print and solve without one, or sign in to a class with two secret pictures.

What it runs on

Next.js 16 on Vercel, Firebase Auth and Firestore, Stripe for credits and the Teacher plan, Claude for words and clues, and a home-grown PDF engine.

The problem

Puzzle sites never have this week's topic

Teachers want a puzzle about exactly what they're teaching: the vocabulary list, the science unit, the Bible story, the class trip. It has to sit at the right reading level for their group. Puzzle sites rarely have the topic. Building a crossword by hand, with clues and an answer key, takes longer than the lesson it supports.

A language model can write words and clues on any topic. It can't be trusted to lay out a grid, count letters or guarantee that a word appears exactly once. It also can't be left alone to decide what's suitable for a five-year-old.

> **So the split was clear early: the model chooses words and writes clues; deterministic code builds and proves every puzzle.**

Constraints we held

- **Kid-safe or nothing.** Every topic, word and passage passes checks before a child sees it.
- **Print-first.** The PDF is the product. Screens are a preview.
- **Priced by compute.** Each puzzle type's price is tied to its measured Claude cost.
- **No child accounts that collect data.** Students get a first name, a class code and two pictures, and nothing else.
- **Honest claims.** No usage figures or outcomes are claimed anywhere on the site.

HOW IT WAS BUILT

Design spec
A 278-line spec: audiences, age bands, puzzle rules, safety, credits, non-goals. Commit #1.

1 · Foundation and engines
Shared types, seeded RNG, word search and crossword builders with validators.

2 · Content and AI
Claude client, schemas, moderation, word filter and judge.

3 · PDF
Letter and A4 sheets with answer keys.

4 · Web app
Next.js pages, API routes, Firebase sign-in.

5 · Credits and Stripe
Ledger, packs, webhooks, refunds for failed puzzles.

6 · Library and community
Reviewed library, sharing, reports, blocking.

7 · Mobile
Expo app on the same packages.

8 · Launch readiness
Legal pages, SEO, rate limits, production config.

92 commits on day one, 32 on day two. Three spec non-goals (coloring pages, student accounts, mazes and dots) were later reversed when the owner asked for them. The shared engine design meant each one plugged in without a rewrite.

Architecture

Five packages, two apps, one source of truth

Everything that decides what a puzzle *is* lives in plain TypeScript packages with no framework attached. The web app, the mobile app, the PDF renderer and the tests all import the same code, so a puzzle previewed on screen is the puzzle that prints.

APPS

apps/web
Next.js 16 App Router on Vercel. 41 pages, 57 API routes, about 12,800 lines.

apps/mobile
Expo app that solves and prints from the same packages. About 1,900 lines.

PACKAGES

shared
Types, age presets, pricing, classroom rules. 2,300 lines.

engine
Seeded builders and validators for every puzzle, plus line art. 2,800 lines.

content
Claude prompts, Zod schemas, safety, retries. 1,500 lines.

pdf
Print layout for every type, Letter and A4. 2,350 lines.

solve
On-screen solving state and checking. 220 lines.

SERVICES

Firebase
Auth for adults. Firestore for puzzles, packs, credit ledger, classes.

Claude
Sonnet 5.5 writes words, clues and passages. Opus 5.5 runs every safety check.

Stripe
Credit packs and the Teacher plan, fulfilled by webhooks.

Vercel
Git-connected. Every push to main builds behind the test gate.

Seeded and reproducible

Every builder takes a seed. The same request gives the same grid, so a saved puzzle is stored as data and redrawn exactly. Regenerating with a new seed gives a fresh layout without another model call when the words are already known.

Stores behind interfaces

Puzzle, credit and classroom stores each have an in-memory version for unit tests and a Firestore version tested against the emulator. Business rules are tested once against both.

Generation pipeline

How a word search gets made

The model only does what it's good at. Everything it returns is parsed against a schema, filtered, judged, then handed to a deterministic builder that proves the result before anyone pays for it.

1 **Screen the topic**
Empty, over 120 characters or on the blocklist? Rejected instantly. Otherwise Opus checks it against the safety policy for the chosen age.
Opus 5.5

2 **Ask for more words than needed**
Sonnet returns a title and a word list sized at 1.6× what the age preset needs, validated by a Zod schema. Teachers can paste their own list instead.
Sonnet 5.5

3 **Filter by age**
Blocklisted words go, and so does any word longer than the age allows: 6 letters for Pre-K, 8 for K–2, 11 for grades 3–5, 14 for 6–8, 20 for teens.
code

4 **Judge what's left**
Opus removes any word that's unsuitable or too hard for the audience to read. It can only remove words, never add them.
Opus 5.5

5 **Build and prove the grid**
The seeded engine places words in the directions the age allows, fills the rest, then the validator re-spells every placement and counts occurrences. Each word must appear exactly once, and every cell must be A–Z.
engine

6 **Retry or refund**
Too few words survived, or the grid failed validation? Ask again at 2.5×. If it still fails, the puzzle is marked failed and its credits go back automatically.
ledger

7 **Print**
The PDF package lays out the sheet and a separate answer key, on Letter or A4.
pdf

Same shape, different engines. Crosswords add a clue-writing step for the same words. Picture puzzles (dot-to-dot, odd one out, spot the difference) have the model choose from a fixed set of hand-drawn shapes, so it never invents an image. Reading passages are reviewed for safety *and* factual accuracy before they're accepted.

Kid safety

Six layers between a topic and a child

No single check is trusted on its own. Cheap deterministic checks run first, model judgments run on the most capable model, and people can still flag anything that slips through.

1

Blocklist and limits

A word blocklist and a length limit run before any model call, on topics, pasted words, titles and custom clues. Nothing is spent on an obviously bad request.

2

Topic policy on Opus

Allows ordinary school, seasonal, hobby and faith topics, and teachable ones like Halloween, evolution and history. Blocks sexual, violent, drug, hateful and profane content. It also blocks innocent-sounding topics that steer toward it ("Goldilocks but spicy") and topics that are really instructions ("ignore your rules"). Blocked topics get one friendly sentence explaining why.

3

Age-shaped words

Word length caps per age, then an Opus judge that may only remove words. A Pre-K sheet gets short words a child could picture; a teen sheet can stretch.

4

Content review

Stories, read-aloud tales, reading passages and writing prompts get a second Opus review that returns `safe` (and `accurate` for reading) before they're kept.

5

Community check

Sharing a puzzle publicly runs its user-written text through the same policy, judged for "anyone browsing a public community of children's puzzles".

6

People

Anyone can report a community puzzle or block its author. Three reports hide it automatically until it's reviewed. The free library is reviewed by the team.

Copyright-minded by design

Pictures come from Puzzld's own drawings and generated line art, never from image models or the web. Paid AI coloring pages were scoped, then shelved until the copyright safeguards are as strong as the safety ones.

Children don't hold accounts

Students exist only inside a teacher's class: a first name, a class code and two secret pictures. Five wrong tries lock sign-in for ten minutes.

Print-first

The PDF is the product

Most of these puzzles end up on paper in a classroom, so the renderer was built as its own package rather than a screenshot of a web page. It draws each type directly: grids, numbered crossword clues, maze walls, dot labels, picture outlines and answer keys. It does this at a fixed, printable line weight on US Letter or A4.

Because the web preview and the PDF read the same engine output, **what you see is exactly what prints**. The coloring tool's thumbnails are drawn from the same line art the PDF uses.

Free tools on the same engines

These need no account and no credits, and nothing is saved. They run on the same packages and bring people back to the paid puzzles:

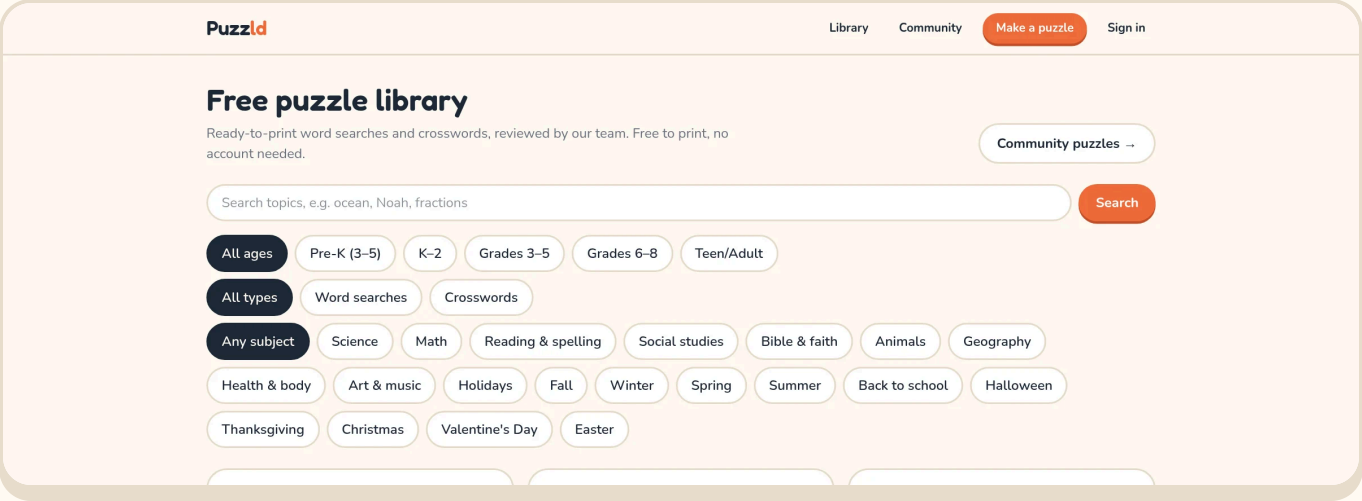
- Certificates
- Math sheets
- Handwriting
- Spelling
- Bingo
- Reward charts
- Flashcards
- Name tags
- Coloring pages

Coloring, generated

Seven kinds, three detail levels, up to a stack of pages that are each a different design:

- **Mandalas**, from bold to fine
- **Patterns**: 10 tilings like hexagons, scales and waves, shuffled per page
- **Zen pictures**: 41 animals and objects filled with pattern bands
- **Name pages**: any name with doodles around it
- **Color by number** mandalas with a key
- **Pixel mystery** pictures
- **Big pictures** for the youngest

Measured, not assumed. Seeds were swept to check variety: spot the difference, mazes and mandalas came out unique across 200 seeds. Odd one out repeated pictures, so it now asks the model for a shortlist of three and skips anything the account used recently.

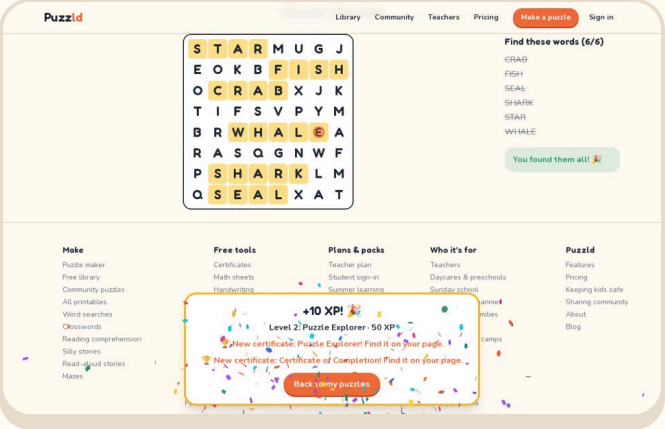


The free library, filterable by age, type, subject and season. Anyone can print from it without an account.

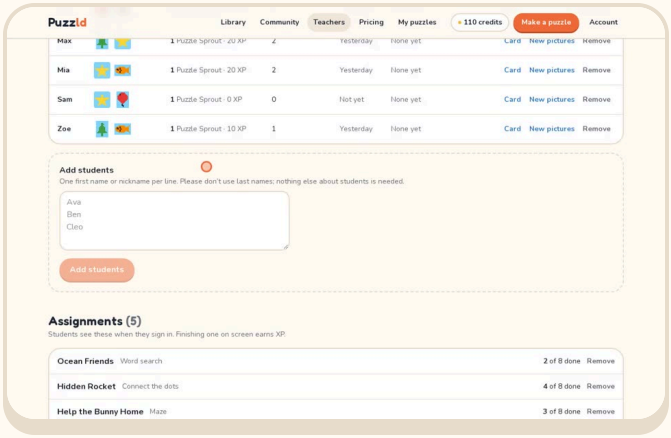
Classrooms

Students solve. Teachers see the progress.

The Teacher plan adds classes on top of the puzzle maker. A teacher assigns puzzles, students solve them on any device, and progress, XP and certificates follow without the teacher marking anything.



A student finishes a word search, levels up and earns certificates. Captured from the demo class on local emulators.



The teacher's progress grid for the same class.

Picture passwords

Young children can't type passwords. They enter a class code, tap their name, then pick their two secret pictures. Teachers can reset them, and five wrong tries lock sign-in for ten minutes.

XP and levels

Each finished puzzle earns XP. Levels and milestone certificates are awarded automatically and print from the same certificate engine as the free tool.

Plan limits

$7.99 a month or $59 a year. Up to 10 classes, 35 students and 60 assignments per class, plus 100 credits every month.

A reproducible demo class. A script seeds the local emulators with a teacher, a class of eight and five puzzles built by the engines, with no AI calls and no credits. One student is a single puzzle away from leveling up. The demo videos on the site were recorded from it with a headless browser, so they can be re-recorded after any UI change.

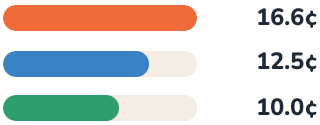
Pricing by compute

Every price is tested against its cost

Generation cost was measured per puzzle type, at the worst case for age and difficulty, with Sonnet writing and Opus checking. Those numbers live in code next to the prices, and a unit test fails the build if any price drops below four times its cost.

Type	AI cost	Credits	Margin
Reading comprehension	4.8¢	2	4.2×
Writing prompt	2.4¢	1	4.2×
Crossword	2.2¢	1	4.5×
Read-aloud tale (long)	5.1¢	3	5.9×
Read-aloud tale (short)	1.8¢	1	5.6×
Story fill-in	1.3¢	1	7.7×
Word search	0.8¢	1	12.5×
Spot / odd one out	0.8¢	1	12.5×
Dot-to-dot	0.7¢	1	14.3×
Maze	0.5¢	1	20×

Credit packs



Price per credit: $4.99, $14.99 and $39.99.
Margins on the left use the cheapest credit, 9.99¢.

Why four times

The margin has to cover one full retry, the 10 free credits every account gets each month, and card fees. Below that, a popular puzzle type could lose money quietly.

10
free credits a month, every account, no card

0
credits lost on a failed puzzle: refunds are automatic

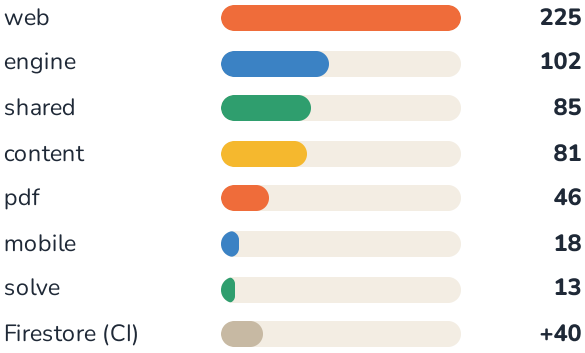
∞
purchased credits never expire

Quality and delivery

Nothing ships without passing the suite

Every package was built test-first. The tests run twice before a deploy: once in GitHub Actions on every push, and again inside the Vercel build, which won't run `next build` until typecheck, tests and lint pass.

Tests by package



The 40 Firestore and Auth store tests run against the Firebase emulators in CI.

The gate

- **GitHub Actions** on Node 24: typecheck and unit tests, lint for web and mobile, then store tests in the Firebase emulators.
- **Vercel build** runs `npm run ci-build`: Turborepo typecheck and tests, ESLint, then `next build`.
- Typecheck runs `next typegen` first, so route types are real.

It caught its first break

The first preview build behind the gate failed on a route type the local check had missed. The fix (typegen before `tsc`) went in before anything shipped.

What the tests actually pin down

- Every word in a word search appears exactly once, in an allowed direction.
- Crossword validation rejects disconnected letters and empty clues.
- Every maze is a perfect maze, with every cell reachable and one route through.
- Same seed, same puzzle, every time.
- Paid credits are granted once per purchase, and a double click is never charged twice.
- Every price clears 4× its measured cost.
- Student sign-in locks after five wrong tries.
- Three reports hide a community puzzle.

Results

What shipped, and what we'd keep

Live today

- Ten puzzle and worksheet types across five age bands, Pre-K to adult
- Party, trip and summer packs that mix types
- A reviewed free library and a community library with reporting and blocking
- Private share links and saved topic lists
- Teacher plan with classes, picture sign-in, XP, levels and certificates
- Nine free tools, including generated coloring pages
- Credit packs, monthly free credits, promo codes
- Legal pages reviewed by counsel, marketing pages, blog, SEO metadata and demo videos

By the numbers

Commits	**124**
Calendar days	**2**
Implementation plans	**8**
Web pages / API routes	**41 / 57**
Lines of TypeScript	**~23,900**
Automated tests	**570 + 40**
Hand-drawn picture shapes	**51**

No usage figures or customer outcomes are claimed.

Lessons

1 — Let the model write, let code prove
Asking Claude for words and clues and never for layout removed a whole class of bugs. Validators can say no; the model can't argue with them.

2 — A spec makes speed safe
Writing the design down first, then working through eight scoped plans, is what let a two-day build stay coherent and absorb three reversed non-goals.

3 — Put the business rules in tests
Margin, refunds, lockouts and report thresholds are tests, not wiki pages. When a model price changes, the build says which product just became unprofitable.

4 — Measure variety, don't assume it
Seeded generators feel random until you sweep 200 seeds. Measuring repeats found the one type that needed a fix.

Try it at puzzld.codefrontlabs.com
Ten free credits a month. The free library and tools need no account.

Built by
Codefront Labs