Devpost Submission Draft — Baseline Tooling Starter

Project name: Baseline Tooling Starter

Short description (one-liner): Integrate Baseline web feature data into developer tooling (ESLint rule, VS Code hint, and CI gate) to help teams adopt modern web features safely.

What it does (summary):

- Uses Baseline (web-features package / Web Platform Dashboard) to determine whether a web platform feature is broadly supported.
- Provides examples and adapters: ESLint rule to warn when unsupported features are used; VS Code hover hints showing baseline status; CI script that fails builds when using unstable features.
- Demo includes a simple CLI and offline sample data to show the flow.

Why this is innovative:

- Combines value-level Baseline checks with editor and CI integrations to provide a single, reproducible signal developers can act on. Unlike tools that only check feature presence, our approach warns on specific property/value pairs and uses curated exceptions and an offline Web Platform Dashboard cache to stay deterministic in CI.

Why this is useful:

- Integrates with workflows developers already use (ESLint, VS Code, GitHub Actions). It prevents accidental use of unstable values early (editor/linter) and enforces policy in CI without relying on runtime network calls. The plugin is lightweight and pluggable so teams can adopt incrementally.

Technologies used:

- Node.js, npm
- web-features (Baseline data)
- ESLint (adapter example — suggested)
- VS Code extension API (adapter idea)

Repository URL: (add your public repo URL here)

Hosted URL: (optional — add live demo link if hosted)

License: MIT (permissive)

How to run (short):

1. Clone the repo.
2. npm install
3. npm run demo
4. Record a >3-minute demo showing the ESLint rule, VS Code hints, and CI check in action.

Demo script outline (>3 minutes):

1. Intro: explain problem (1:00)
2. Show CLI demo with baseline data (0:30)
3. Show ESLint integration highlighting an unsupported feature (0:45)
4. Show VS Code hover hint (0:30)
5. Show CI gate failing on unstable feature usage (0:30)
6. Wrap up and next steps (0:25)

Submission checklist (Devpost questions):

- Full project description: this file + README
- Submitted code repo URL: (add link)
- License: MIT included
- Demo video: upload and link
- Answered all questions: yes

Evidence & links:

- Deterministic value-level checks: `eslint-plugin-baseline/lib/rules/use-baseline.js` implements exceptions → computeBaseline (optional) → web-features → webstatus-cache → heuristics fallback.
- Offline CI compatibility: `.github/workflows/lint.yml` prefetches `data/webstatus-cache.json` to avoid network IO during checks.
- Reproducible demo & tests: `src/demo.js`, `test/run.js`, and `test/lint-rule-test.js` demonstrate the end-to-end flow locally.
- License & submission materials: `LICENSE` (MIT), `README.md`, `SUBMISSION.md`.