

Full-Stack Developer Challenge

🎯 Context

We want to offer a simple **feedback widget** that can be embedded into any website or app.

Your task is to build both the **frontend SDK/widget** and a **minimal backend API** to collect feedback.

This exercise is designed to test your ability to:

- Design and implement a small **end-to-end solution** (frontend + backend).
 - Make thoughtful choices about architecture, frameworks, and trade-offs.
 - Document clearly so others can use your work easily.
-



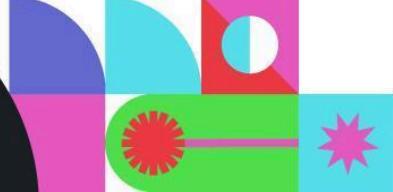
What to Build

1) Frontend SDK / Widget

- An initialization method that sets configuration (e.g., projectId, API key).
 - A way to open a feedback modal or form (rating 1–5, optional comment).
 - A unique user identifier that persists locally and is reused.
 - A method to submit feedback to your backend API.
 - Error handling (log or display meaningful messages).
 - The SDK should be usable both by importing into a project and by embedding via a `<script>` tag.
-

2) Backend API

- Endpoint to **receive feedback submissions**:
 - Accepts projectId, userId, rating, optional comment, and timestamp.
 - Validates input and rejects invalid payloads.
 - Requires some form of authentication (API key or similar).
- A simple **health check endpoint**.
- Store feedback somewhere (database, file, in-memory — your choice).



3) Developer Experience

- Clear **README** with setup instructions, examples, and how to test end-to-end.
 - Include **sample integration** (e.g., a minimal HTML page that uses your SDK).
 - Add some **tests** for both SDK and API (at least a few basic ones).
 - Provide a short **ADR (Architecture Decision Record)** explaining the technologies and approaches you chose, and why.
-

📦 Deliverables

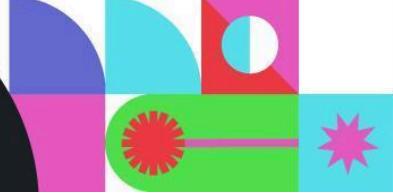
- A repository containing:
 - `sdk/` (your widget code)
 - `server/` (your backend API)
 - README with:
 - How to install/run locally in a few steps.
 - Example usage with a snippet.
 - How to test the API (curl examples, Postman, etc.).
 - ADR (1–2 paragraphs).
-

✓ Acceptance Criteria

- SDK can be initialized, open a modal, and submit feedback.
 - User identifier is persisted and reused.
 - API validates inputs and enforces basic authentication.
 - End-to-end flow works: submitting feedback from widget reaches backend.
 - Documentation is clear and easy to follow.
-

⭐ Nice to Have (Optional)

- Retry or queueing when offline.
- Configurable backend URL for different environments.
- Debug mode to log requests.
- Collect small extra context (e.g., app version, user agent).



Evaluation

We will look at:

- **Correctness:** does it work end-to-end?
- **Code Quality:** is the code readable, maintainable, and tested?
- **DX (Developer Experience):** easy setup, good documentation.
- **Security & Reliability:** reasonable handling of keys, validation, errors.
- **Reasoning:** clarity in your ADR about *why* you made certain choices.

👉 Feel free to use any technologies you like. What matters most is how you **design, justify, and deliver** your solution.