Coding & Development / API Development

Create a client-side SDK (Python, JavaScript, or TypeScript) from an OpenAPI spec or described endpoints.
Difficulty: Advanced
Model: GPT-4 / Claude / Gemini
Use Case: SDK Generation, API Clients, Developer Experience
Updated: May 2026
Why This Prompt Exists
Most API clients are written manually — repetitive boilerplate for every endpoint.

You get:

- inconsistent error handling across endpoints
- no type safety (TypeScript could catch errors)
- duplicate authentication logic
- no retry logic for transient failures
- hard-to-maintain client code

But an SDK is not optional for public APIs.

It is the developer experience.

- Base client: authentication, request method, error handling
- Endpoint methods: named methods for each API call
- Type definitions: request/response types (TypeScript)
- Error classes: specific errors for different status codes
- Retry logic: exponential backoff for transient failures

Without an SDK, developers struggle to integrate your API.

This framework forces AI to generate production-ready SDKs.

The Prompt

```
Assume the role of a developer experience engineer who builds API
client SDKs.

Your task is to generate an API client SDK.

Generate:

1. BASE CLIENT CLASS
   - Constructor with base URL, auth token
   - Request method (handles headers, errors)

2. AUTHENTICATION
   - API key, Bearer token, or other method

3. ENDPOINT METHODS
   - One method per endpoint
   - Parameter validation
   - Type hints/TypeScript types

4. ERROR HANDLING
   - Custom error classes
   - Status code mapping

5. RETRY LOGIC (optional)
   - Exponential backoff for 5xx errors
```

6. USAGE EXAMPLE
   - How to instantiate and use the SDK

INPUTS:

Target Language:
[PYTHON / JAVASCRIPT / TYPESCRIPT]

OpenAPI Spec (if available) or Endpoint Descriptions:
[PASTE OR DESCRIBE ENDPOINTS]

Authentication Type:
[API KEY / BEARER TOKEN / NONE]

Base URL:
[INSERT]

Retry on Failure:
[YES / NO]

RULES:
- Use type hints (Python) or TypeScript types for type safety
- Handle common errors (network, timeout, invalid response)
- Don't expose internal details in error messages
- Make authentication configurable
- Add docstrings/JSDoc for all public methods
- Use async/await for network requests
- Provide usage examples in documentation

How To Use It

- Provide an OpenAPI spec if available — it generates better SDKs.
- If no OpenAPI spec, describe endpoints clearly (method, path, params).
- Use TypeScript for type safety (JavaScript developers can use the generated types).
- Include retry logic for production SDKs (5xx errors are often transient).
- Test the generated SDK against a real API endpoint.

Example Input

**Target Language:** TYPESCRIPT

**OpenAPI Spec:** TaskFlow API (from previous prompt) — endpoints: GET /users, POST /users, GET /users/{id}, PUT /users/{id}, DELETE /users/{id}

**Authentication Type:** BEARER TOKEN (JWT)

**Base URL:** https://api.taskflow.com/v1

**Retry on Failure:** YES (3 retries with exponential backoff)

Why It Works
Most API clients are written manually.

This framework improves outcomes by forcing:

- base client (DRY principle)
- authentication handling (security)
- endpoint methods (usability)
- type safety (TypeScript)
- retry logic (resilience)

Great SDKs don’t just work — they make developers want to use your API.

## Build Better AI Systems

Subscribe for advanced prompt engineering, AI coding tools, API development frameworks, and practical strategies for developers and engineers.

Carefully engineered prompts for people doing real work.

### Share this:

- Share on Facebook (Opens in new window) Facebook
- Share on X (Opens in new window) X

See also The Authentication Middleware Builder