API Technical Requirements

1. Overview

API Name: Name of the API

Purpose: Briefly describe what the API does and its primary use case

Version: Version of the API (e.g., v1.0)

Contact Information: Name and contact details for the API owner or development team

2. Architecture

API Type: REST, GraphQL, SOAP, etc.

Protocol: HTTP, HTTPS, gRPC, etc.

Format: JSON, XML, etc.

Authentication: Type of authentication required (e.g., OAuth2, API Key, JWT)

3. Endpoints

Each endpoint should include:

Endpoint URL: The URI path (e.g., /users, /products/{id})

HTTP Method: GET, POST, PUT, DELETE, etc.

Description: What the endpoint does

Request Parameters:

Path Parameters: (e.g., /users/{id})

Query Parameters: (e.g., ?sort=asc)

Header Parameters: Authorization, Content-Type, etc.

Body Parameters: JSON schema, example payload

Responses:

Success Response: HTTP status codes (e.g., 200, 201), response structure, and examples

Error Responses: Possible error codes (e.g., 400, 404, 500), error message structure, and examples

4. Data Models

Entities: Describe each data entity the API interacts with (e.g., User, Product)

Schema: Attributes of each entity, including data type, format, constraints (e.g., username: string, required)

5. Rate Limits & Throttling

Limits: Requests per second or minute

Quota: Daily or monthly usage limits

Back-off Strategy: Recommended approach for handling rate limits (e.g., exponential back-off)

6. Security

Authentication: Details on token handling, expiry, and renewal

Authorization: Describe any role-based or permission-based access control

Data Encryption: SSL/TLS requirements, encryption standards

Data Privacy: Handling of sensitive data (e.g., PII, GDPR compliance)

7. Error Handling & Logging

Error Codes: List of standardized error codes with descriptions

Error Structure: JSON schema for error responses (e.g., {“error”: “Invalid request”, “code”: 400})

Logging: Required logging (e.g., request logs, error logs), logging levels, and storage location

8. Performance & Scalability

Latency Requirements: Expected response time per request

Concurrent Connections: Maximum connections the API can handle simultaneously

Caching: Use of caching (e.g., Redis), caching strategy (TTL, caching headers)

Load Balancing: Load balancing techniques (round-robin, least connections)

9. Monitoring & Alerts

Monitoring: Tools and metrics (e.g., API latency, error rates)

Alerting: Thresholds and notifications for critical issues

10. Documentation & Support

Swagger/OpenAPI: Availability of Swagger, OpenAPI, or other API documentation

Client Libraries: Languages or SDKs available for client integration

Support: Details on support channels (e.g., email, Slack, forums)