API Documentation

# Overview

This API processes text input by checking for offensive content, performing sentiment analysis, and summarizing the text using OpenAI's GPT-4 model. It also provides a history of processed texts and a health check for the API.

# Base URL

http://<your-server-ip>:5000

# Endpoints

## 1. POST /process

This endpoint processes the input text by:  
- Checking for offensive content.  
- Analyzing sentiment.  
- Categorizing the text based on sentiment.  
- Summarizing the input text using GPT-4.

### Request

Method: POST  
Endpoint: /process  
Rate Limit: 3 requests per minute  
Request Payload:  
{  
 "text": "Your input text here"  
}

### Response

Success (200):  
{  
 "processed\_text": "Summarized text output here",  
 "category": "Positive Content",  
 "sentiment": "POSITIVE"  
}  
  
Error Responses:  
- 400 - INVALID\_INPUT: Text input cannot be empty.  
 {"error\_code": "INVALID\_INPUT", "error\_message": "Text input cannot be empty."}  
- 400 - INPUT\_TOO\_LONG: Text exceeds token length limit.  
 {"error\_code": "INPUT\_TOO\_LONG", "error\_message": "Text input exceeds maximum length of 2048 tokens."}  
- 400 - OFFENSIVE\_CONTENT: Text contains offensive content.  
 {"error\_code": "OFFENSIVE\_CONTENT", "error\_message": "Text input contains offensive content."}  
- 429 - RATE\_LIMIT\_EXCEEDED: Exceeded rate limit.  
 {"error\_code": "RATE\_LIMIT\_EXCEEDED", "error\_message": "Rate limit exceeded: <message>"}  
- 401 - AUTHENTICATION\_ERROR: Invalid OpenAI authentication.  
 {"error\_code": "AUTHENTICATION\_ERROR", "error\_message": "Authentication error: <message>"}  
- 500 - OPENAI\_API\_ERROR: OpenAI API error.  
 {"error\_code": "OPENAI\_API\_ERROR", "error\_message": "OpenAI API error: <message>",  
 "openai\_error\_message": "<specific error message>", "openai\_error\_type": "<error type>"}  
- 500 - UNEXPECTED\_ERROR: Unexpected error.  
 {"error\_code": "UNEXPECTED\_ERROR", "error\_message": "An unexpected error occurred: <message>"}

## 2. GET /history

This endpoint returns the history of processed texts.

### Request

Method: GET  
Endpoint: /history

### Response

Success (200):  
{  
 "history": [  
 {  
 "input\_text": "Your input text here",  
 "processed\_text": "Summarized text output here",  
 "category": "Positive Content",  
 "sentiment": "POSITIVE"  
 },  
 ...  
 ]  
}

## 3. GET /health

This endpoint checks the health of the API by making a test request to the OpenAI API.

### Request

Method: GET  
Endpoint: /health

### Response

Success (200):  
{  
 "status": "OK",  
 "openai\_status": "reachable"  
}  
  
Error (500):  
{  
 "status": "degraded",  
 "error\_message": "<error details>"  
}

# Rate Limiting

3 requests per minute for /process endpoint.  
200 requests per day for /process endpoint.

# Example Use Case

## 1. Process Text Request Example

Request:  
curl -X POST http://<your-server-ip>:5000/process -H "Content-Type: application/json" -d '{"text": "I love programming!"}'  
  
Response:  
{  
 "processed\_text": "Programming is fun and rewarding.",  
 "category": "Positive Content",  
 "sentiment": "POSITIVE"  
}

## 2. Get History Request Example

Request:  
curl http://<your-server-ip>:5000/history  
  
Response:  
{  
 "history": [  
 {  
 "input\_text": "I love programming!",  
 "processed\_text": "Programming is fun and rewarding.",  
 "category": "Positive Content",  
 "sentiment": "POSITIVE"  
 }  
 ]  
}