# Laravel Practical Test: Twitter Feed API

## Objective

The goal of this test is to evaluate your proficiency in Laravel, Object-Oriented Programming (OOP) principles, API development, and working with third-party APIs. You are required to create a Laravel-based API that interacts with a Twitter-like data, retrieves tweets, and processes them to return specific information.

---

## Requirements

You need to implement the following API endpoints in a Laravel application:

1. **GET /api/tweets**
   - Returns the first 20 tweets from a specific public Twitter user's timeline.
2. **GET /api/tweets/most-liked**
   - Returns the tweet with the highest number of likes from the user's timeline.
3. **GET /api/tweets/most-commented**
   - Returns the tweet with the highest number of replies (comments) from the user's timeline.

---

## Constraints

1. **OOP Design**
   - You must adhere to OOP principles.
   - Use proper separation of concerns: Controllers should delegate logic to Services or Repositories.
   - Utilize Dependency Injection (DI) where necessary.
2. **Error Handling**
   - Handle API errors gracefully (e.g., network issues, rate limits, invalid responses).
   - Return meaningful error responses with appropriate HTTP status codes.
3. **Response Format**
   - Return the data in a well-structured JSON format.
   - Example response for GET /api/tweets:
   - {
   -  "status": "success",
   -  "data": [
   -  {
   -  "id": "123456789",
   -  "text": "This is a sample tweet",
   -  "likes": 100,

- "comments": 50,
- "created_at": "2025-01-06T12:00:00Z"
- },
- ...
- ]
- }

4. **Testing**
    - Write unit tests or feature tests for the implemented endpoints.
5. **Documentation**
    - Provide brief documentation (a README file) on how to set up and run your application.

---

## Setup Instructions

**Twitter API**
Use the following data file to fetch tweets**:** mock_tweets.json

---

## Submission

Your submission should include:

1. A link to a Git repository (GitHub, GitLab, etc.) containing:
    - The Laravel project code.
    - A README file with setup instructions.
2. Ensure that your repository is accessible (public or shared appropriately).

---

## Bonus Points

You can earn extra points by implementing:

1. **Caching**
    - Cache the response of GET /api/tweets for a specified duration (e.g., 5 minutes) to reduce the number of external API calls.
2. **Pagination**
    - Implement pagination for the list of tweets in GET /api/tweets.
    - ate API docs.

---

## Evaluation Criteria

Your submission will be evaluated based on the following criteria:

1. **Code Quality**
    - Adherence to OOP principles and Laravel best practices.
    - Clean, readable, and maintainable code.
2. **Functionality**
    - Correct implementation of the required endpoints.
3. **Error Handling**
    - Robust error handling and meaningful error messages.
4. **Testing**
    - Presence and quality of unit/feature tests.
5. **Documentation**
    - Clarity and completeness of the provided documentation.

---

## Timeline

You have **2 days** from the date of receiving this task to complete and submit your solution.