**Performance Testing Report for CoinGecko API**

**1. Executive Summary**

This report outlines the performance testing results for the following CoinGecko API endpoints:

1. **Simple Price Endpoint**: GET /simple/price?ids=bitcoin&vs\_currencies=usd
2. **Coin List Endpoint**: GET /coins/list
3. **Coin Markets Endpoint**: GET /coins/markets?vs\_currency=usd&ids=bitcoin

The tests were conducted using **K6** to simulate **load** and **stress** scenarios. The results highlight the API’s behavior under expected and extreme conditions, including response times, error rates, and system stability.

**2. Testing Scenarios**

**2.1 Load Testing**

* **Objective**: Simulate expected user load to measure system performance.
* **Configuration**:
  + **Virtual Users (VUs)**: 50
  + **Duration**: 2 minutes (ramp-up, steady state, ramp-down)
* **Endpoints Tested**:
  + Simple Price
  + Coin List
  + Coin Markets

**2.2 Stress Testing**

* **Objective**: Gradually increase the load to identify the API’s breaking point.
* **Configuration**:
  + **Virtual Users (VUs)**: Up to 150
  + **Duration**: 3 minutes (ramp-up, steady state, ramp-down)
* **Endpoints Tested**:
  + Simple Price
  + Coin List
  + Coin Markets

**3. Key Metrics**

**3.1 Load Testing Results**

| **Endpoint** | **Throughput (RPS)** | **Avg Response Time** | **Error Rate** | **Checks Passed** |
| --- | --- | --- | --- | --- |
| Simple Price | 36.06 RPS | 39.8 ms | 99.77% | 50.11% |
| Coin List | 25.93 RPS | 355.58 ms | 99.58% | 0.41% |
| Coin Markets | 28.76 RPS | 303.64 ms | 99.68% | 0.31% |

**3.2 Stress Testing Results**

| **Endpoint** | **Throughput (RPS)** | **Avg Response Time** | **Error Rate** | **Checks Passed** |
| --- | --- | --- | --- | --- |
| Simple Price | 79.98 RPS | 38.97 ms | 99.88% | 50.01% |
| Coin List | 80.11 RPS | 35.62 ms | 99.90% | 0.09% |
| Coin Markets | 79.39 RPS | 43.72 ms | 99.89% | 0.10% |

**4. Detailed Findings**

**4.1 Simple Price Endpoint**

* **Load Testing**:
  + The API handled **36.06 requests per second (RPS)** with an average response time of **39.8 ms**.
  + **99.77% of requests failed**, indicating the API struggled to handle the load.
  + The **response time check (< 500ms)** passed for **50.11% of requests**.
* **Stress Testing**:
  + The API handled **79.98 RPS** with an average response time of **38.97 ms**.
  + **99.88% of requests failed**, showing the API’s inability to handle high load.
  + The **response time check (< 500ms)** passed for **50.01% of requests**.

**4.2 Coin List Endpoint**

* **Load Testing**:
  + The API handled **25.93 RPS** with an average response time of **355.58 ms**.
  + **99.58% of requests failed**, indicating poor performance under load.
  + The **response is an array** and **contains Bitcoin** checks passed for **0.41% of requests**.
* **Stress Testing**:
  + The API handled **80.11 RPS** with an average response time of **35.62 ms**.
  + **99.90% of requests failed**, showing the API’s inability to handle high load.
  + The **response is an array** and **contains Bitcoin** checks passed for **0.09% of requests**.

**4.3 Coin Markets Endpoint**

* **Load Testing**:
  + The API handled **28.76 RPS** with an average response time of **303.64 ms**.
  + **99.68% of requests failed**, indicating poor performance under load.
  + The **response is an array** and **contains Bitcoin market data** checks passed for **0.31% of requests**.
* **Stress Testing**:
  + The API handled **79.39 RPS** with an average response time of **43.72 ms**.
  + **99.89% of requests failed**, showing the API’s inability to handle high load.
  + The **response is an array** and **contains Bitcoin market data** checks passed for **0.10% of requests**.

**5. Observations**

1. **High Error Rates**:
   * All endpoints experienced **>99% error rates** under both load and stress testing.
   * This indicates the API is unable to handle even moderate levels of traffic.
2. **Response Times**:
   * Response times were generally low (**<500 ms**) for successful requests.
   * However, the high error rate suggests the API is failing to process most requests.
3. **System Stability**:
   * The API struggled to maintain stability under load, with frequent connection timeouts and forced closures.
4. **Rate Limiting**:
   * The CoinGecko API has a **rate limit of 10-50 requests per minute** for the free tier. Exceeding this limit likely contributed to the high error rates.

**6. Recommendations**

1. **Optimize API Performance**:
   * Investigate and address the root cause of the high error rates (e.g., server capacity, database bottlenecks).
2. **Implement Rate Limiting**:
   * Ensure the API adheres to CoinGecko’s rate limits to avoid being blocked or throttled.
3. **Scale Infrastructure**:
   * Increase server capacity or use a load balancer to handle higher traffic volumes.
4. **Improve Error Handling**:
   * Provide meaningful error messages and retry mechanisms for failed requests.
5. **Monitor API Health**:
   * Use tools like **Grafana** or **Prometheus** to monitor API performance and detect issues in real-time.

**7. Conclusion**

The CoinGecko API demonstrated significant performance issues under both load and stress testing. The high error rates and frequent connection failures indicate the API is not currently capable of handling moderate to high traffic volumes. Immediate action is required to optimize performance, scale infrastructure, and ensure compliance with rate limits.

**8. Next Steps**

1. Share this report with the development team for further analysis.
2. Implement the recommended improvements to enhance API performance.
3. Conduct follow-up testing to validate the effectiveness of the changes.