Generate a policy for API rate limiting.

**API Rate Limiting Policy**

**1. Purpose and Scope**

This policy establishes guidelines for implementing and managing rate limiting across all organizational APIs to ensure service availability, prevent abuse, and maintain optimal performance for legitimate users.

**Scope**: This policy applies to all public-facing APIs, internal APIs, and third-party API integrations managed by the organization.

**2. Rate Limiting Objectives**

* **Service Protection**: Prevent system overload and maintain service availability
* **Fair Usage**: Ensure equitable access to API resources among all users
* **Security**: Mitigate abuse, DDoS attacks, and unauthorized access attempts
* **Cost Management**: Control infrastructure costs associated with API usage
* **Performance Optimization**: Maintain consistent response times and system performance

**3. Rate Limiting Tiers and Limits**

**3.1 User Tiers**

**Free Tier**

* Rate Limit: 100 requests per hour, 1,000 requests per day
* Burst Limit: 10 requests per minute
* Applicable to: Public users, trial accounts

**Basic Tier**

* Rate Limit: 1,000 requests per hour, 10,000 requests per day
* Burst Limit: 50 requests per minute
* Applicable to: Paid individual accounts

**Premium Tier**

* Rate Limit: 10,000 requests per hour, 100,000 requests per day
* Burst Limit: 200 requests per minute
* Applicable to: Business accounts, verified partners

**Enterprise Tier**

* Rate Limit: Custom limits based on contract negotiations
* Burst Limit: Negotiated based on use case
* Applicable to: Large enterprise clients with SLAs

**3.2 Endpoint-Specific Limits**

**Read Operations** (GET requests)

* Standard rate limits apply as per user tier

**Write Operations** (POST, PUT, DELETE requests)

* 50% of standard rate limits
* Additional validation and monitoring

**Authentication Endpoints**

* 5 failed attempts per 15 minutes per IP address
* 20 successful authentications per hour per user

**File Upload Endpoints**

* 10 uploads per hour for Free tier
* Size limits: 10MB per file for Free/Basic, 100MB for Premium/Enterprise

**4. Implementation Standards**

**4.1 Rate Limiting Algorithms**

**Primary Algorithm**: Token Bucket

* Provides burst capacity while maintaining average rate control
* Tokens replenish at a steady rate based on tier limits

**Secondary Algorithm**: Sliding Window Log

* Used for precise tracking of authentication attempts
* Maintains detailed logs for security analysis

**4.2 Rate Limiting Keys**

Rate limits are applied based on:

* **Primary**: API key or authenticated user ID
* **Secondary**: IP address (for unauthenticated requests)
* **Tertiary**: Combination of user ID and endpoint category

**4.3 Headers and Response Codes**

**Required Response Headers**:

X-RateLimit-Limit: [requests per time window]

X-RateLimit-Remaining: [requests remaining in current window]

X-RateLimit-Reset: [UTC timestamp when limit resets]

X-RateLimit-Window: [time window in seconds]

**HTTP Status Codes**:

* 429 Too Many Requests: Rate limit exceeded
* 503 Service Unavailable: System overload protection triggered

**5. Monitoring and Alerting**

**5.1 Key Metrics**

* Request volume per tier and endpoint
* Rate limit hit ratios by user and tier
* Geographic distribution of rate-limited requests
* Average response times during high-traffic periods
* False positive rate limit triggers

**5.2 Alert Thresholds**

**Warning Level**:

* 70% of users in a tier hitting rate limits
* Sustained 80% rate limit utilization across any tier

**Critical Level**:

* 90% of premium/enterprise users hitting limits
* System-wide rate limiting activated
* Suspected coordinated abuse patterns

**5.3 Reporting Requirements**

* Weekly rate limiting effectiveness reports
* Monthly abuse pattern analysis
* Quarterly tier limit optimization reviews
* Annual policy effectiveness assessment

**6. Exception Handling**

**6.1 Temporary Limit Increases**

**Approval Process**:

* Business justification required
* Technical impact assessment
* Maximum 72-hour duration without executive approval
* Automatic reversion to standard limits

**Emergency Overrides**:

* CTO or designated technical lead approval required
* Must be documented within 24 hours
* Post-incident review mandatory

**6.2 Whitelist Management**

**Criteria for Whitelisting**:

* Critical business partners with SLAs
* Internal monitoring and health check systems
* Emergency services or public safety applications

**Review Process**:

* Quarterly review of all whitelisted entities
* Annual revalidation of business justification
* Immediate removal upon contract termination

**7. User Communication and Support**

**7.1 Documentation Requirements**

* Clear rate limit information in API documentation
* Best practices for efficient API usage
* Error handling guidance for rate limit responses
* Contact information for limit increase requests

**7.2 Notification Procedures**

**Proactive Notifications**:

* Users approaching 80% of their rate limit
* 24-hour advance notice for policy changes
* Scheduled maintenance affecting rate limiting

**Reactive Communications**:

* Immediate notification of emergency rate limiting
* Status page updates during incidents
* Post-incident communication within 2 hours

**8. Compliance and Audit**

**8.1 Data Privacy**

* Rate limiting logs contain minimal personal data
* Data retention limited to 90 days for operational logs
* Compliance with applicable data protection regulations

**8.2 Audit Requirements**

* Monthly technical audit of rate limiting effectiveness
* Semi-annual business impact assessment
* Annual third-party security assessment including rate limiting controls

**9. Policy Governance**

**9.1 Roles and Responsibilities**

**API Product Manager**: Policy oversight and business impact assessment **Engineering Lead**: Technical implementation and monitoring **Security Team**: Abuse pattern analysis and threat assessment **Support Team**: User communication and exception handling

**9.2 Review and Updates**

* Quarterly policy effectiveness review
* Annual comprehensive policy update
* Emergency updates as needed for security incidents
* Stakeholder approval required for major changes

**10. Implementation Timeline**

**Phase 1 (Immediate)**: Implement basic rate limiting for public APIs **Phase 2 (30 days)**: Deploy tier-based limits and monitoring **Phase 3 (60 days)**: Implement advanced algorithms and alerting **Phase 4 (90 days)**: Complete documentation and user communication

**Appendix A: Technical Implementation Guidelines**

**Rate Limiting Infrastructure**

* Distributed rate limiting using Redis cluster
* Graceful degradation when rate limiting service unavailable
* Circuit breaker pattern for rate limiting service failures

**Performance Considerations**

* Rate limiting checks must complete within 5ms
* Minimal impact on API response times
* Horizontal scaling capability for high-volume APIs

**Security Integration**

* Integration with WAF and DDoS protection
* Correlation with security incident response procedures
* Automated blocking of obvious abuse patterns

**Document Version**: 1.0  
**Effective Date**: [Date]  
**Next Review Date**: [Date + 1 year]  
**Policy Owner**: [API Product Manager]  
**Approved By**: [CTO/Engineering Director]