Non-Functional Requirements Specification

1. **Security (Paramount & International Standard)**
   1.1. **Data Encryption:** Encrypt all sensitive data at rest (database, file storage for multi-modal content) and in transit (all network communication, API calls) using strong, internationally recognized cryptographic standards.
   1.2. **Access Control:** Implement granular, attribute-based or role-based access control (ABAC/RBAC) for all platform features and data, extending to multi-modal content, with distinct levels for Public Viewers, Trustees, Administrators, etc.
   1.3. **Input Validation & Sanitization:** Rigorously sanitize and validate all user inputs (from UI, APIs, and multi-modal submissions) to prevent injection attacks (SQL, XSS, command injection, path traversal) and malicious file uploads.
   1.4. **API Security:** Secure all internal and external APIs (e.g., OAuth 2.0, robust API key management, stringent rate limiting, DDoS protection).
   1.5. **Vulnerability Management:** Implement continuous security monitoring, regular independent security audits, penetration testing, and vulnerability scanning by certified ethical hackers.
   1.6. **Compliance:** Strict adherence to international data privacy regulations (e.g., GDPR, CCPA, UN privacy principles), legal archiving standards, and relevant international human rights frameworks.
   1.7. **Authentication & Authorization:** Implement multi-factor authentication (MFA) as a mandatory requirement for platform users (especially Trustees and Admins), robust password hashing, and granular authorization checks across all services and data access points. Public scoring requires some form of authentication (e.g., email verification, standard login process) but not a more advanced authentications as in the full Trustee-level vetting. These security mechanisms should insure that sufficient effort has been taken to prevents bots, hack attempts, and disruptive or malicious programs from specifically manipulating public scoring , or hashtag designation processes, or derailing the platform and harming it in geenral.
   1.8. **Anonymity Guarantees:** Implement technical and procedural safeguards to ensure the anonymity of user submissions, preventing the linking of submitted content back to the submitter, unless legally compelled and under strict protocol.
   1.9. **Zero Trust Architecture:** Adopt a Zero Trust security model, verifying every access request regardless of origin.
2. **Performance**
   2.1. **Ingestion Throughput:** Capable of processing a very high volume of social media posts (identified via hashtag replies) and user-submitted multi-modal content daily (specify target numbers if possible, e.g., 5-10 million items/day, including large multimedia files).
   2.2. **Initial Analysis Latency:** Initial assessment and flagging of "potential hate-speech" should occur within minutes of ingestion.
   2.3. **Thorough Analysis Latency:** Comprehensive, "thorough analysis" for *confirmed* hate speech should be initiated immediately after trustee approval and completed within a defined timeframe (e.g., within X hours).
   2.4. **UI Responsiveness:** Web UI must be highly responsive, with complex multi-lingual query results and page loads within acceptable limits (< 2 seconds for typical operations, < 5 seconds for complex aggregations). Trustee voting interface must be extremely responsive.
   2.5. **Database Query Speed:** Aggregation and multi-lingual search queries should execute

efficiently, even with petabytes of diverse datasets.

2.6. **Scalability:**

2.6.1. **Horizontal Scalability:** The system architecture must support massive horizontal scaling of data ingestion, multi-modal analysis, database storage, and web services to handle exponentially increasing data volumes, user loads, and geographic distribution.

2.6.2. **Elasticity:** Ability to automatically scale resources up or down based on demand.

3. **Reliability & Resiliency (Global Operations)**

3.1. **High Availability:** Critical services (database, web server, multi-modal analysis engine, storage, trustee voting platform) must be designed for continuous high availability (e.g., 99.99% uptime), minimizing downtime even during maintenance.

3.2. **Fault Tolerance:**

3.2.1. **Loose Coupling:** Design all services using a microservices architecture, ensuring extreme loose coupling so the failure of one service has minimal to no impact on others. This includes independent deployment, scaling, and failure isolation.

3.2.2. **Graceful Degradation:** The system should degrade gracefully if non-critical services fail, maintaining core functionality.

3.2.3. **Retry Mechanisms:** Implement robust, exponential backoff retry mechanisms for API calls, external service integrations, and internal service communications.

3.2.4. **Circuit Breakers:** Implement circuit breakers to prevent cascading failures.

3.3. **Error Handling & Logging:** Comprehensive, centralized, multi-lingual logging of all system events, errors, warnings, and security alerts for debugging, auditing, and forensic analysis. Implement structured logging.

3.4. **Backup & Disaster Recovery (DR):** Automated, geographically redundant backups of all system components (application code, configurations, all databases, multi-modal content storage) with stringent RTO (Recovery Time Objective) and RPO (Recovery Point Objective) targets suitable for critical international operations (e.g., RPO < 1 hour, RTO < 4 hours). Regular DR drills are mandatory.

3.5. **Monitoring & Alerting:** Proactive, 24/7 monitoring of system health, performance metrics, and security events across all deployed regions, with automated, actionable alerting for critical issues to relevant international support teams.

4. **Maintainability & Extensibility**

4.1. **Modular Architecture:** Mandate a well-defined microservices or event-driven architecture to facilitate independent development, deployment, scaling, and technology upgrades.

4.2. **Code Quality & Documentation:** Adhere to exceptionally high coding standards, comprehensive multi-lingual documentation (code comments, API docs, architectural diagrams), and rigorous unit/integration/end-to-end testing.

4.3. **Configuration Management:** Externalize all configuration parameters (including language packs, AI model versions, API keys, dynamic voting thresholds, trustee roles) for dynamic modification without code changes or redeployments.

4.4. **API-First Design:** All internal and external service interactions must be defined via clear, well-documented, versioned APIs.

4.5. **Technology Stack:** Leverage battle-tested, open-source, and enterprise-grade technologies that are widely supported and scalable for global deployments.

4.6. **Upgradability:** Design components for seamless, zero-downtime upgrades to newer versions, models, or alternative technologies, including A/B testing for new AI models.

5. **Usability (Global & Inclusive)**
   5.1. **Intuitive Workflow:** User workflows should be highly logical, efficient, and straightforward for users with diverse technical backgrounds (especially Trustees), minimizing training requirements.
   5.2. **Accessibility:** Strict adherence to WCAG 2.1 AA (or higher) standards for accessibility, supporting users with disabilities across all supported languages.
   5.3. **Clear Feedback:** Provide clear, timely, and multi-lingual feedback to users on their actions, system status, and analysis results.
   5.4. **Customization:** Allow authenticated users (Trustees, Analysts) to personalize dashboards, reports, aggregation views, and language preferences.
   5.5. **Translation Management:** Implement a robust system for managing and deploying UI translations and linguistic model updates.
6. **Legal, Ethical & Operational Considerations (United Nations / International Authority Level)**
   6.1. **International Data Privacy Law:** Absolute compliance with the strictest international data privacy laws and organizational privacy policies for all data, including anonymization of user submissions and robust data retention/deletion policies.
   6.2. **Algorithmic Bias Mitigation:** Implement rigorous processes and continuous monitoring to actively identify and mitigate algorithmic bias in hate speech detection models and entity extraction, ensuring fairness and non-discrimination across all languages, cultures, and demographics. Regular audits of AI model performance are essential.
   6.3. **Transparency & Explainability:** Provide transparency in how hate speech is detected, classified, and how entities are extracted. Where feasible, offer explanations or justifications for AI classifications (Explainable AI - XAI). The trustee voting process adds a crucial human oversight layer to ensure ethical and accurate classification.
   6.4. **Reporting & Evidential Compliance:** Ensure all generated reports, aggregated findings, and stored evidence (including original multi-modal content and audit trails of trustee votes) meet international legal and evidential standards, suitable for use in court or by international tribunals. Implement tamper-proof logging of evidence chain-of-custody.
   6.5. **Jurisdictional Complexity:** Design the platform to handle the complexities of varying hate speech definitions and legal frameworks across different international jurisdictions.
   6.6. **UN/International Body Integration:** Consider potential integration points with existing UN or international authority systems (e.g., reporting pipelines, secure communication channels, data sharing protocols for high-priority cases like "Top 10" listings).
   6.7. **Human Rights Framework:** All aspects of the platform must align with and uphold international human rights principles, including freedom of expression balanced with protection from hate speech.
   6.8. **Data Sovereignty:** Consider requirements for data sovereignty and potential deployment in multiple data centers in different countries to comply with legal mandates.