MANIPAL INSTITUTE OF TECHNOLOGY
MANIPAL
(A constituent unit of MAHE, Manipal)

Synopsis

CryptTalk – Secure Message Delivery Platform

Team Members:
Shubham Sethia - 230911236
Siddhartha Saha- 230911150
Harsh Mangesh Dagare - 230911158
Pushkar Ojha - 230911156

1) Project Overview:

PrivyText is a secure messaging and file sharing platform designed to safeguard user privacy through advanced encryption, self-destructing content, and intelligent monitoring. The system integrates a robust Python backend with modern web and mobile frontends, offering a secure yet user-friendly experience suitable for both personal and enterprise communication needs.

2) Problem Statement:

Many existing communication platforms compromise either on security or user experience. Some provide strong encryption but lack intuitive interfaces, while others prioritize ease of use at the cost of privacy. PrivyText aims to bridge this gap by delivering enterprise-grade encryption, configurable content expiry, and intelligent monitoring within an accessible, cross-platform environment.

3) Objectives:

- Implement end-to-end encrypted real-time messaging and secure file sharing.
- Enable self-destructing content to minimize sensitive data exposure.
- Ensure cross-platform availability on web, desktop, and mobile devices.
- Integrate intelligent monitoring to detect suspicious activities without violating user privacy.
- Offer modular advanced features for enhanced security and media handling.

(4) Features and Team Allocation:

Team A – Frontend-Focused with Backend Integration:

- Develop user interfaces with encryption/decryption support (AES-256-GCM, RSA-4096, perfect forward secrecy, zero-knowledge principles).
- Implement self-destructing content UI with time/view-based expiry and encrypted file deletion triggers.
- Build real-time messaging with WebSocket-powered instant chats, encrypted group conversations, voice messages with expiry, and encrypted search.

• Deliver responsive and accessible applications across web, native mobile, desktop, and browser extensions, including offline queuing.

Team B – Backend-Focused with Frontend Collaboration

• Manage core security backend components: key management, encryption processing, and secure data storage.
• Implement optional advanced authentication: multi-factor authentication, biometrics, device/session management, and secure key backup.
• Develop content management features: message drafts, presence/status tracking, encrypted contact profiles, markdown/emoji support, and secure file preview with scanning.
• Build file security system supporting chunked encryption, tamper detection, overwrite deletion, and role-based sharing.
• Enable intelligent monitoring: keyword detection, suspicious activity alerts, admin dashboards, privacy-preserving logs, and real-time notifications.
• Integrate optional advanced capabilities: AI-powered security analysis, privacy tools , plausible deniability, and advanced media features like encrypted voice notes and transcription.
• Provide customization through themes, notifications, drag-and-drop file sharing, and cross-device synchronization.

(5) Workflow:

• User Initiates Communication: User sends a message or file from web/mobile/desktop app. Content is encrypted on the client side using AES-256-GCM, keys exchanged using RSA-4096.
• Server Handling: Server receives encrypted payload; stores temporarily or delivers instantly via WebSocket. Expiry timers or view-count triggers stored alongside metadata (but not content).
• Self-Destruct Mechanism: Once timer/view count expires, server deletes encrypted content and notifies all clients to erase local copies.
• Intelligent Monitoring: AI module scans metadata patterns (not message content) for suspicious behavior. Alerts admins while maintaining zero-knowledge privacy.
• File Handling: Large files split into chunks, each encrypted, stored, and verified for integrity before delivery.
• Cross-Platform Sync: Changes in one client (e.g., message deletion) instantly sync across all logged-in devices.

(6) Software Requirements:

• Frontend: React.js with WebCrypto API for encryption and Socket.IO for real-time messaging.
• Backend: Python using FastAPI or Django REST Framework with PyCryptodome and MongoDB database.
• Containerization: Docker for easy deployment and scalability.

(7) Expected Outcomes:

• A secure, scalable, and fully functional messaging platform balancing usability and privacy.
• Effective collaboration between frontend and backend teams, ensuring seamless integration of encryption and system features.
• Modular architecture allowing optional advanced features to be enabled as needed by users or enterprises.
• A user-friendly experience that encourages adoption without compromising security principles.