SecureNotes: Privacy-Focused Encrypted Notes App

Table of Contents

Executive Summary

SecureNotes is a privacy-focused, offline-first note-taking application built with React, CryptoJS, and IndexedDB. It provides robust AES-256 client-side encryption, protecting user data from unauthorized access and ensuring notes are secure both at rest and in transit. Designed for usability and security, it offers note CRUD operations, search, pinning, archiving, and optional cloud sync—all while never storing keys or plain text notes on any server.

Table of Contents

1. Introduction
2. Key Features
3. Technology Stack
4. Architecture Overview
5. Core Modules
6. Security Best Practices
7. Usage Guide
8. Development & Deployment

9. Future Improvements
10. Acknowledgements

1. Introduction

SecureNotes was developed to address the need for secure personal note-taking in environments where privacy is paramount. By implementing zero-knowledge encryption and an offline-first paradigm, this app stands apart from typical cloud-based notes apps in protecting user confidentiality.

2. Key Features

- **AES-256-GCM Encryption:** Notes are encrypted client-side before being stored.
- **PBKDF2 Key Derivation:** Master password is transformed into a cryptographic key; salt and IV are handled per industry standards.
- **Offline-First Architecture:** Full CRUD access to notes with IndexedDB, even without connectivity.
- **Auto-Lock Session:** All decryption keys reside only in memory and are cleared on inactivity.
- **Search, Pin, Archive:** Advanced note management and organization, with searchable titles, pinning for priority access, and archiving for decluttering.
- **Optional Cloud Sync:** Encrypted blobs supported for remote backups (e.g., via Firebase).
- **Export/Import:** Download encrypted JSON backups and restore on any device with the password.
- **Responsive UI:** Three-panel adaptive layout for desktop/tablet/mobile.

3. Technology Stack

- **Frontend:** React (Hooks, Context, Router)
- **Encryption:** CryptoJS (AES-256-GCM, PBKDF2-SHA256)
- **Persistent Storage:** IndexedDB (via native APIs)
- **Optional Cloud Sync:** Firebase integration
- **PWA:** Service Worker, Manifest
- **Testing:** Jest (Unit, Integration)

4. Architecture Overview

Main Layers

- **Presentation Layer:** Modular React components for authentication, notes list, editor, settings
- **Business Logic Layer:** Encryption service, storage manager, search, sync
- **Data Layer:** IndexedDB (notes & settings objects)

Security Model

- **Master password** (never stored) → Salted PBKDF2 key → AES-256 note encryption
- **Session keys** live in browser memory only, auto-cleared on lock/logout

5. Core Modules

5.1 Authentication

- Password setup and login, strength meter
- PBKDF2 key derivation with unique salt
- Session auto-lock

5.2 Notes CRUD & Encryption

- Create: UUID, encrypt with AES-256-GCM, save to IndexedDB
- Read: Decrypt in-memory, display in rich text editor
- Update: Re-encrypt; auto-save on change
- Delete/Archive: Soft delete first, then permanent removal

5.3 Search, Pin, Archive

- Search: Real-time, case-insensitive on decrypted notes
- Pinning: Priority sorting
- Archiving: Filter by status, restore with unarchive

5.4 Export, Import & Sync

- Export: Save encrypted JSON backup
- Import: Restore on any device using master password
- Sync (optional): Send/receive encrypted blobs to cloud

6. Security Best Practices

- **Never store master password or key in plain text.**
- **PBKDF2 iterations:** Minimum 100,000 (configurable)
- **Unique salt/IV per note:** For strong cryptographic isolation
- **Encryption/Decryption only in memory:** No decrypted state hits disk
- **Auto-lock on inactivity:** Further reduces risk of unauthorized access
- **Browser compatibility checks:** Warn if critical APIs unavailable

7. Usage Guide

First-Time User

1. Open app → Welcome screen
2. Set master password (strength feedback)
3. Write and save notes; all are encrypted immediately

Returning User

1. Enter master password to unlock notes
2. Search, edit, organize with full offline capability

Data Management

- Pin notes, archive finished items
- Export backups periodically; store in a secure location
- Auto-lock after 15 minutes; session cleared if browser/tab closed

Cloud Sync

(Optional) Authenticate and upload encrypted blobs for backup and restoration, ensuring vendors never access note contents.

8. Development & Deployment

- **React/Hooks:** SPA, code split for performance
- **IndexedDB Storage:** Two object stores (notes, settings)
- **Service Worker:** Asset caching and offline access
- **Testing:** Core cryptography and CRUD flows fully unit/integration tested
- **Deployment:** Static hosting (Netlify, Vercel, GitHub Pages);
- **Environment Variables:** PBKDF2 iterations, auto-lock timeout are configurable

9. Future Improvements

- Rich text/Markdown preview
- Multi-user collaboration (end-to-end encrypted)
- Attachments and file uploads
- Mobile/desktop wrappers (Electron, Cordova)
- AI-powered search and tagging (local models)
- Extensions, API integration

10. Acknowledgements

Special thanks to the open-source community for cryptography, offline storage, and PWAs—especially CryptoJS, React contributors, Firebase, and IndexedDB documentation teams.

Project created: October 2025

Author: Civil Engineering Student (Bihar), Web Dev Learner

Contact: [Add your contact or repository link]

End of Report