# Threat Model (STRIDE Analysis)

Project: Password Strength Checker & Manager

Date: October 2023

## 1. System Decomposition

The system consists of a Web Client (Browser), a Flask Web Server, and an SQLite Database. External interactions include the "Have I Been Pwned" (HIBP) API.

## 2. STRIDE Analysis Table

| **Threat Category** | **Description** | **Potential Impact** | **Mitigation Strategy** | **Status** |
| --- | --- | --- | --- | --- |
| **Spoofing** | An attacker impersonates a valid user using stolen credentials. | Unauthorized access to the password vault. | **Implemented:** Strong Session Management via Flask-Login. **Implemented:** TOTP 2FA requires physical device access. | ✅ Mitigated |
| **Tampering** | An attacker modifies the database files directly or via injection. | Privilege escalation (User -> Admin) or data corruption. | **Implemented:** SQLAlchemy ORM prevents SQL Injection. **Planned:** Database file encryption for production. | ⚠️ Partially Mitigated |
| **Repudiation** | A user denies performing an action (e.g., deleting a password). | Inability to trace malicious internal activity. | **Implemented:** created\_at timestamps on all records. **Planned:** Comprehensive audit logging table for all actions. | ⚠️ Partially Mitigated |
| **Information Disclosure** | Leakage of password hashes or API usage. | User passwords compromised; Privacy violation. | **Implemented:** Bcrypt hashing with salt. **Implemented:** k-Anonymity for HIBP API (only 5 chars sent). | ✅ Mitigated |
| **Denial of Service** | Flooding the checker endpoint to exhaust API limits or CPU. | Service unavailability for legitimate users. | **Implemented:** Timeout handling on API requests. **Planned:** Rate Limiting (Flask-Limiter) on public routes. | ⚠️ Partially Mitigated |
| **Elevation of Privilege** | A regular user accesses /admin or viewing other users' data. | Full system compromise. | **Implemented:** RBAC checks (current\_user.role) on all sensitive routes. | ✅ Mitigated |

## 3. Data Flow Diagram (Context)

* **User** -> [HTTPS] -> **Flask App**
* **Flask App** -> [SQLAlchemy] -> **SQLite DB**
* **Flask App** -> [HTTPS/GET] -> **HIBP API** (Anonymized Hash)

## 4. Risk Assessment Summary

The highest remaining risks involve **Denial of Service** (lack of rate limiting) and **Tampering** (local database file access). These are acceptable for a prototype but require remediation for production.