Sound Advice

A moderated music help site for serious musicians. Secure by design.

Security in Sound Advice is built in layers, on the principle of defence in depth. If one protection below is bypassed the others still stand, and our most sensitive data is never in reach to begin with.

Even if our database is stolen, your sensitive data stays protected.

Passwords, email addresses, and security codes/tokens are each protected separately before they are stored. An attacker who stole the database would not be able to use these values directly.

What you give us	What we store	What an attacker sees
jane@example.com	Encrypted email + lookup fingerprint	**9f3a2b8c1d4e5f6a…**
MyPassword123!	Scrambled hash + per-password salt + a server-only secret key	**$2b$12$Rx9k…**
Security code: 472819	One-way fingerprint signed with a server-only secret key	**a8f3d2c1…**

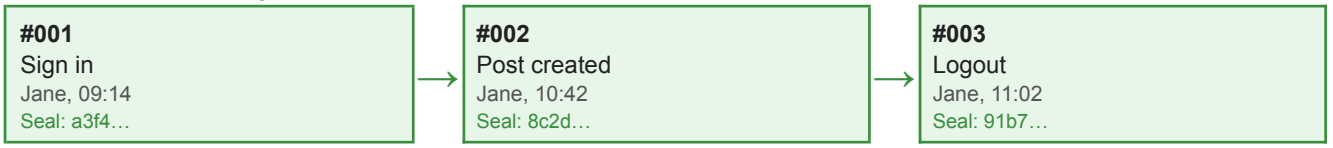
An attacker would need to break several separate protections, and the keys that protect them live externally in the server's configuration, never in the database itself.

A tamper-evident record of key events.

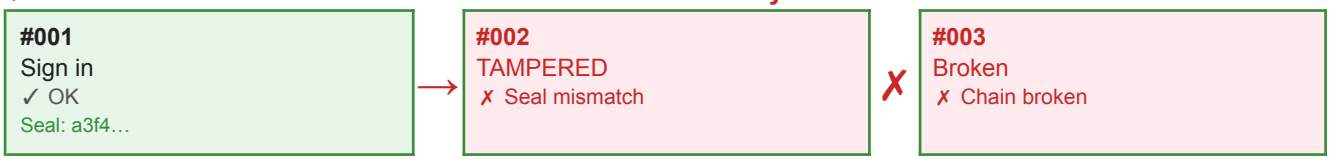
Every sign-in, post, and account change is logged in a chain, each record sealed to the one before it. Change one record and the chain breaks.

Our admin dashboard runs this check across every record:

✓ Chain intact: every record sealed and linked



✗ Someone alters record #002: the chain breaks immediately



Each record's seal is calculated from the previous record's seal. Changing one record breaks every record that follows - and the broken chain is quickly detectable.

We check what's inside each file, not what it's called.

Renaming a virus to "cat.png" won't get past our system. We read the bytes inside every uploaded file and reject anything that isn't a real image before it's written to disk.

cat.png ✓ ACCEPTED	evil.exe.png ✗ REJECTED
First bytes inside the file: **89 50 4E 47 0D 0A 1A 0A** **✓ Matches the PNG signature.** This is a real image. Upload accepted.	**First bytes inside the file:** **4D 5A 90 00 03 00 00 00** **✗ Does not match PNG signature.** This is a Windows program in disguise. Rejected.

So a harmful program renamed to look like a picture never makes it onto the server.

We share the minimum, and we check who you are at every step.

- Every action checks that you actually own what you're accessing. Requests for something you're not allowed to see return 'not found' rather than 'forbidden', so you can't even confirm it exists.
- Our database queries name the exact columns they need, no wildcards, no extras. Things like password hashes or another user's email address can't appear in a public API response.
- The database account the application uses holds only the permissions it needs, and it can't grant itself more - the principle of least privilege.

What you see when viewing a profile

```
{
  username: "jane"
  joined:   "Jan 2024"
  bio:      "I like jazz"
}
```

What's actually in our database

```
{
  id:               17,
  username:         "jane",
  email_hash:       "9f3a2b…",
  email_encrypted:  "7d28…",
  password:         "$2b$12$Rx…",
  is_admin:         false,
  bio:              "I like jazz",
…
}
```

Passwords built to NIST standards.

Guideline 1 - Set a minimum password length Every password must be at least 8 characters.	**Guideline 2 - Do not set password expiry dates** Passwords are never set to expire in our system.
Guideline 3 - Remove complexity requirements There are no rules forcing a mix of uppercase, digits or symbols.	**Guideline 4 - Screen all new passwords** Each password is checked against the SecLists top 100,000 most common and breached passwords.

+ Passwords with a user's username or email are also rejected, as NIST warns against context-specific words.

Tested end-to-end with OWASP ZAP across every endpoint of the site.

Using a widely used open-source web security testing tool, we scanned all 31 endpoints. The scan found **0 high-severity, 0 medium-severity and 0 confirmed vulnerabilities.**