SENG406
Threat Modelling Report

Authors

Hanghang Song
Xiaoping Ma
Xiayi Yao
Yue Pan

[Group 5]

Due Date: 8 August 2025

1. Introduction

This report provides a comprehensive threat model analysis of ShareNote, a web-based collaborative note-taking application. The primary goal of this analysis is to identify potential security threats, evaluate associated risks, and propose applicable mitigation strategies to strengthen the system's security. The STRIDE framework is used to guide the threat identification and analysis process.

In this report, the key components of the ShareNote system - the actors, trust levels, assets, entry/exit points and dependencies of ShareNote, are identified. Based on the system's functional requirements, a data flow diagram (DFD) is constructed to visualize different data flows through the application and highlight the critical trust boundaries. Each element in the DFD is systematically analyzed for potential threats, and mitigation strategies are proposed where applicable.

1.1. System Overview

***ShareNote* allows users to:**

- Create and edit notes containing text, images, hyperlinks, and internal note references.
- Share notes via secret links (with read/edit permissions) or make them publicly readable.
- Comment on shared notes (text-only).
- Manage access controls (creator can delete comments, restrict editing).

1.2. Key Components

- Frontend: Web interface (HTTPS, accessible on desktop/mobile).
- Backend: On-premises server (firewall-protected, reverse proxy).
- Database: On-premises, not directly internet-accessible.

2. System Analysis

2.1. Actors

Actors are individuals or groups that interact with a system, either by exchanging data with it or exploiting its vulnerabilities for malicious purposes.

Table 1 below includes a list of potential system actors determined from 3 different aspects: *website users, system maintainer, attacker*.

Table 1: System Actors.

ID	Actor Name	Description
Website Users		
1	Guest (Anonymous User)	Anonymous users are individuals who access the *ShareNote* system without authenticating. They have not logged in and therefore possess limited access rights, typically restricted to publicly shared content or features that do not require user identification or authorization.
2	Logged-in User	Logged-in users are individuals who have successfully authenticated and accessed their accounts within the *ShareNote* system, granting them authorized access to platform features and resources based on their credentials and assigned permissions in accordance with established security protocols.
System Maintainers		
3	Database Server Administrator	Database server administrators oversee system management, handling user accounts, access controls, note content modification and deletion, backups, and security monitoring. Their responsibilities ensure appropriate permissions, data integrity, and protection of the *ShareNote* system against unauthorized access or potential security threats.
4	Internal Staff	Internal staff members are designated support personnel with restricted privileges. They possess read-only access to user account information and system logs. They also can utilize the management portal to create or delete user accounts and manage note list.

Attackers		
5	Potential Attackers	A potential attacker refers to any unauthorized individual or malicious organization that seeks to exploit vulnerabilities in ShareNote's system to gain access to sensitive data. These actors pose a significant security threat, as they may compromise the confidentiality, integrity, and availability of information through illegal intrusion or harmful actions

2.2. Dependencies

Dependencies are external components or systems that ShareNote relies on for proper functionality. These include third-party services and infrastructure elements that must be secured and maintained, as they can introduce potential attack surfaces.

Table 2 below lists the system dependencies that support the operations of ShareNote but are not part of its internal logic.

Table 2: System Dependencies.

ID	Name	Description
1	Web Server	Provides the front-end interface for *ShareNote*, delivering HTML, JavaScript, and CSS files to users over HTTPS. It manages user interactions and securely communicates with the backend services.
2	Reverse Proxy	Serves as a secure intermediary between clients and the back-end server, managing TLS termination to decrypt HTTPS traffic. It enforces security policies such as access control and rate limiting. Commonly implemented with firewall. This combination provides advanced safeguard communication and maintains application security and performance.
3	Database	Stores persistent data for *ShareNote*, including user information, notes, comments, and sharing permissions. Access to the database is strictly limited to the web application backend, ensuring controlled data flow and preventing direct external access. This design enhances data security, integrity, and enforces proper access control within the system.

4	Firewall	Implements traffic filtering and strict access control, permitting only TLS-encrypted connections to ensure secure data transmission. By combining a reverse proxy with a firewall, the system provides enhanced protection for backend infrastructure, mitigating unauthorized access, blocking malicious traffic, and enforcing security policies at the network and application layers.

2.2.1. Documented Assumptions

Given the limited functional specifications, the following assumptions were made:

- **Core functions:**
  - Separated databases for public shared notes, private notes, comments, and user account details.
  - Sharing methods may include Shareable links or user-to-user invitations.
- **Maintenance roles:**
  - Dedicated staff perform:
    - Database backups/restores.
    - User account management(create/deletion)
    - Content moderation

2.3. Trust Levels

Trust levels, also referred to as privilege levels, define degree of trust the system assigns to various components or actors within the architecture. These levels determine the extent of access and control granted to external entities, helping to establish security boundaries by distinguishing trusted and untrusted components or users.

Table 3 below outlines system trust levels in the system of *ShareNote*, categorized into three key actor types: non-logged in users, logged in users, and system administrators.

Table 3: System Trust Levels.

ID	Name	Description
Non-logged in Users		
1	Anonymous user (Guest)	▪ **Definition:** A user who accesses the ShareNote website without authentication. ▪ **Permissions:** Read-only access to public notes and comments.

2	User with Invalid login Credentials	▪ **Definition:** A user attempting to log in with incorrect or invalid authentication credentials. ▪ **Permissions:** Read-only access to public notes and comments (same as Anonymous User).
	Standard User Roles	
3	User with valid login credentials	▪ **Definition:** A user with valid login credentials. ▪ **Permissions:** o Full access to public notes o Ability to leave comments o Create and share new notes o Access to notes shared by other users
4	Note creator	▪ **Definition:** An authenticated user who creates content ▪ **Permissions:** o All standard user privileges (refer to trust level ID 3.) o Ability to share created notes with other users o Full management of created notes
5	Invited read-only users	▪ **Definition:** An authenticated user with view-only access ▪ **Permissions:** o All standard user privileges (refer to trust level ID 3.) o Ability to view (but not edit) shared private notes
6	Invited read/edit users	▪ **Definition:** An authenticated user who creates content ▪ **Permissions:** o All standard user privileges (refer to trust level ID 3.) o All read-only collaborator privileges o Ability to edit shared private notes
	Administrative Roles	
7	Database Server Admin	▪ **Definition:** system security and data management lead ▪ **Permissions:** o Full read/write database access o User account management o Data backup/recovery operations o Content removal authority

8	Database Read Admin	▪ **Definition:** User with read-only administrative access ▪ **Permissions:** o View all user information User account management o Access management portal (user account operations) o Database read access only
9	Database Read/Write Admin	▪ **Definition:** User with limited write access ▪ **Permissions:** o All read administrator privileges o Additional database write capabilities Database read access only

2.4. Entry Points

Entry points represent the interfaces through which data or commands enter the ShareNote system. These interfaces constitute potential attack surfaces where malicious actors might attempt to compromise system security or influence operations.

Table 4 shows the primary entry points of the *ShareNote* system. These entry points involve users with varying trust levels, from anonymous guests to high-privilege administrators.

Table 4: System Entry Points.

ID	Name	Description	Trust Level
1	HTTPS port(port:443)	The *ShareNote* website is available via HTTPS.	(1) Anonymous user (Guest) (2) User with invalid login credentials (3) User with valid login credentials (4) Logged in users (Note creator) (5) Logged in users (Invited read-only users) (6) Logged in users (Invited read/edit users)

2	*ShareNote* Main page	Entry points for all Authorized users	(4) Logged in users (Note creator) (5) Logged in users (Invited read-only users) (6) Logged in users (Invited read/edit users)
3	Public sharing notes pages	All notes made public by its creator provides entry point for all users including guests	(1) Anonymous user (Guest) (2) User with invalid login credentials
4	Private sharing notes pages	Notes only shared with specific login users by its creator	(3) User with valid login credentials (4) Logged in users (Note creator) (5) Logged in users (Invited read-only users) (6) Logged in users (Invited read/edit users)
5	Login page	Guests, admins, internal staffs must log in to system to gain further access.	(1) Anonymous user (Guest) (2) User with invalid login credentials (3) User with valid login credentials (4) Logged in users (Note creator) (5) Logged in users (Invited read-only users) (6) Logged in users (Invited read/edit users) (7) Database Server Admin (8) Database Read Admin (9) Database Read/Write Admin
5.1	Login function	Accept user supplied valid credentials and verify them with those in the database.	(3) User with valid login credentials (4) Logged in users (Note creator) (5) Logged in users (Invited read-only users) (6) Logged in users (Invited read/edit users) (7) Database Server Admin (8) Database Read Admin (9) Database Read/Write Admin

6	Administrative Entry Points	SSH Administrative Access (Port 22 – Assumed). System administration and maintenance access.	(7) Database Server Admin (8) Database Read Admin (9) Database Read/Write Admin

2.5. Exit Points

Exit points are interfaces through which data leaves the ShareNote system, including both intended outputs (e.g., note exports) and unintended leaks (e.g., data exposure in HTTP responses). These points frequently correspond to entry points, with vulnerabilities potentially emerging from their interaction. When exit points handle sensitive information like user data or private notes, inadequate security controls may lead to unauthorized access.

Table 5 lists ShareNote's primary exit points, which require proper security measures to prevent unauthorized data disclosure, particularly for sensitive content like notes and user activities.

Table 5: System Exit Points.

ID	Name	Description	Trust Level
1	Notes Sharing with *ShareNote* links (etc. email sharing)	A note can be accessed by a private link provided by its creator. These links can be shared by email or any other way outside of the *ShareNote*.	(3) User with valid login credentials (4) Logged in users (Note creator) (5) Logged in users (Invited read-only users) (6) Logged in users (Invited read/edit users)
2	Administrative Exit Points	SSH Administrative exit (Port 22 – Assumed. System Provide feedback on users' CRUD requests	(7) Database Server Admin (8) Database Read Admin (9) Database Read/Write Admin

3	HTTP response	Some data can be accessed by sending HTTP response by the webserver, when the webserver receives HTTP GET or POST.	(1) Anonymous user (Guest) (2) User with invalid login credentials (3) User with valid login credentials (4) Logged in users (Note creator) (5) Logged in users (Invited read-only users) (6) Logged in users (Invited read/edit users) (7) Database Server Admin (8) Database Read Admin (9) Database Read/Write Admin
4	Note Download	Any notes that users have access to can be downloaded to users' local machine	(3) User with valid login credentials (4) Logged in users (Note creator) (6) Logged in users (Invited read/edit users)
5	Public shared note	Any notes that are shared to guests or users	(1) Anonymous user (Guest) (2) User with invalid login credentials (3) User with valid login credentials (4) Logged in users (Note creator) (5) Logged in users (Invited read-only users) (6) Logged in users (Invited read/edit users) (7) Database Server Admin (8) Database Read Admin (9) Database Read/Write Admin

6	Public-viewable comments	Any comments that can be viewed by other users	(1) Anonymous user (Guest) (2) User with invalid login credentials (3) User with valid login credentials (4) Logged in users (Note creator) (5) Logged in users (Invited read-only users) (6) Logged in users (Invited read/edit users) (7) Database Server Admin (8) Database Read Admin (9) Database Read/Write Admin

2.6 Assets

Assets are the components or data within the *ShareNote* system that hold value and importance, making them potential targets for attackers. Assets can be both physical assets (such as stored data) or abstract assets (such as note content or note access permission). These assets are the elements that need protection from misuse, loss, or unauthorized access, often directly tied to the confidentiality, integrity, and availability of the system.

Table 6 below lists the key assets identified in the ShareNote application, along with their descriptions and associated trust levels.

Table 6: System Assets.

ID	Asset Name	Description	Trust Level
1	User Credentials data	The login credentials that a user or an administrator will use to log into note create website.	(3) User with valid login credentials (4) Logged in users (Note creator) (5) Logged in users (Invited read-only users) (6) Logged in users (Invited read/edit users)
2	Administrator Credentials	The login credentials that an administrator will use to login to the internal website.	(7) Database Server Admin (8) Database Read Admin (9) Database Read/Write Admin

3	Personal data	The *ShareNote* application stores personal information relating to the users and administrators.	(3) User with valid login credentials (4) Logged in users (Note creator) (5) Logged in users (Invited read-only users) (6) Logged in users (Invited read/edit users) (7) Database Server Admin (8) Database Read Admin (9) Database Read/Write Admin
4	Note content	Text, images, hyperlinks, and links to other notes. Public notes visible to anyone, private notes restricted.	(4) Logged in users (Note creator)
5	Comments	Text comments associated with notes. Visibility matches note access.	(3) User with valid login credentials (4) Logged in users (Note creator)
6	Note Access Permissions	Information about who can read or edit which note. This information stores in SharedNote objects.	(4) Logged in users (Note creator) (7) Database Server Admin (9) Database Read/Write Admin
7	Sharing Links	Secret URLs used to share notes with read/edit permissions.	(4) Logged in users (Note creator) (5) Logged in users (Invited read-only users) (6) Logged in users (Invited read/edit users) (7) Database Server Admin (8) Database Read Admin (9) Database Read/Write Admin
8	Audit Logs / Metadata	Logs of actions performed by users (e.g., created note, shared note) — useful for accountability.	(7) Database Server Admin (8) Database Read Admin (9) Database Read/Write Admin

3. Threat Modelling

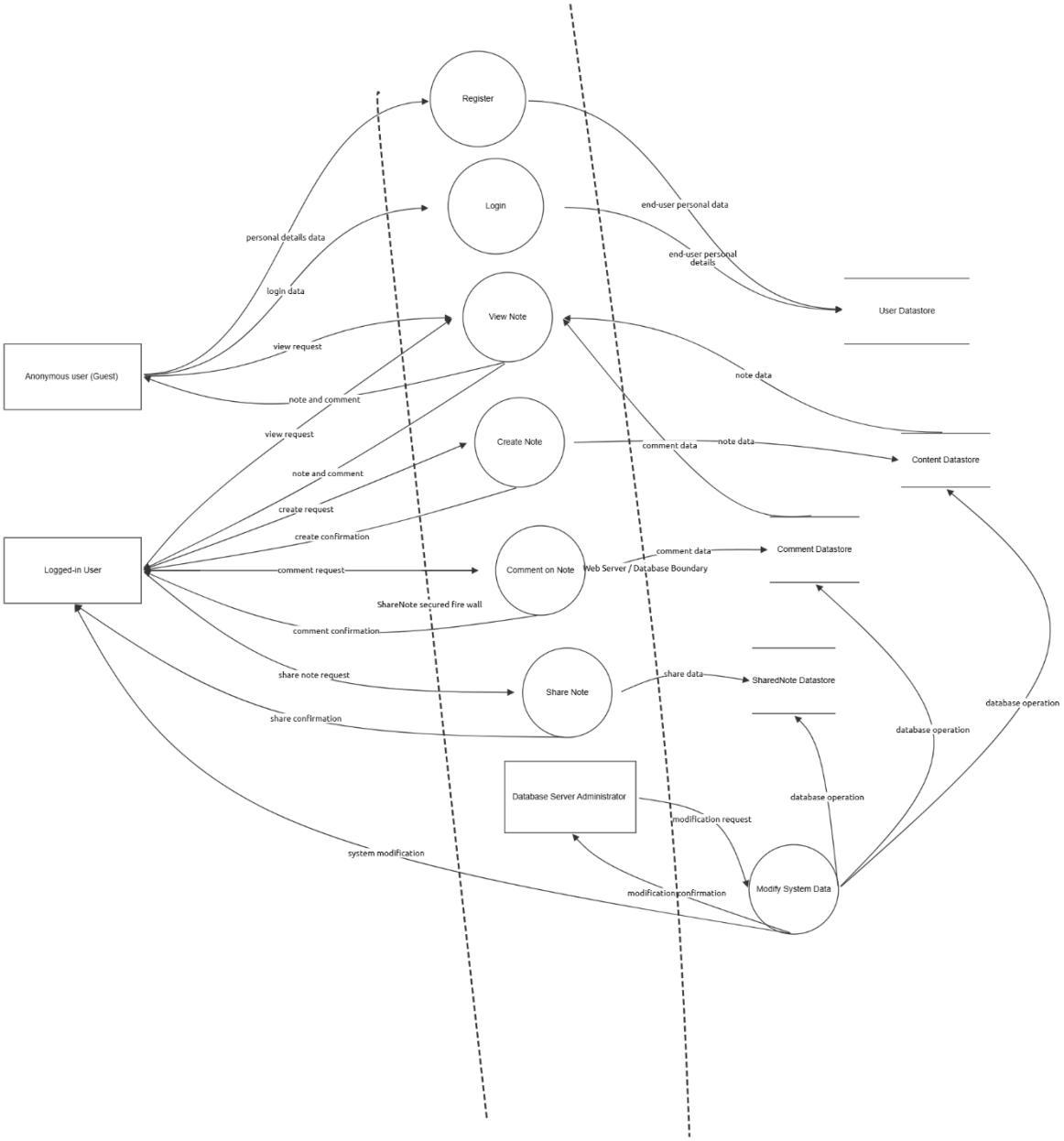


Figure 1: Data Flow Diagram.

STRIDE threats and their countermeasures are included in the Threat Modelling report.

Threat Modelling Output Report

The completed output report is appended at the end of the report.

4. Potential Function Improvements

The following critical improvements could enhance security posture of ShareNote addressing vulnerabilities in its current design. Each proposal includes a threat analysis evaluating risks before and after implementation.

➢ Enforce Stronger Authentication Mechanisms

Current Risk – weak passwords

Weak passwords or lack of multi-factor authentication (MFA) increase susceptibility to brute-force and credential-stuffing attacks.

Proposed change

- Mandate MFA (SMS/OTP or Authentication APP) for all users, especially note creators and collaborators [1].
- Implement password complexity requirements (minimum length, special characters.)

Threat Analysis

Threat Scenario	Before	After
Brute-force attacks	High risk because weak passwords	Reduced risk (MFA blocks unauthorized access)
Credential theft	Account takeover possible	MFA adds a secondary layer of defense
Password reuse exploits	Vulnerable	Mitigated via complexity rules

➢ Enhanced User Account Management System

Current Risks – User Account Lockout Mechanism for Recovery

- Unlimited recovery attempts with no progression delay or protection against brute-force of reset tokens.
- User enumeration through different messages or timing during recovery.
- Outbound recovery email or SMS can be abused for flooding.

Proposed Change

- Gate recovery requests with rate limits, backoff and lockouts, and use uniform responses to prevent enumeration.
- Increase assurance and token security by requiring step-up MFA for sensitive changes, issuing single-use short-TTL tokens, revoking other tokens and sessions, and notifying users of changes [3].
- Add abuse prevention by throttling email or SMS delivery, introducing bot challenges, and monitoring with alerting.

Threat Analysis

Threat Scenario	Before	After
Reset token brute-force	Unlimited attempts and no delay	Per-IP and per-account limits with backoff and short-TTL single-use tokens
Credential stuffing via recovery	High success at scale	Central limits, challenges, and anomaly monitoring reduce impact
Recovery channel flooding	Unbounded email or SMS volume	Throttled delivery with alerts and temporary lockouts

➢ Add Expiration and Revocation Options for Share Links

Current Risk – Uncontrolled Link Sharing

Once a private share link is generated, it can be forwarded indefinitely. Without expiration or revocation, there is no way to stop access if it's leaked or no longer needed.

Proposed Change

- Allow note creators to set expiry dates for share links (e.g., 7 days, 30 days).
- Add a link revocation feature so creators can manually disable a link at any time.
- Display active share links and their expiry status in the note's settings page.

Threat Analysis

Threat Scenario	Before	After
Link leakage	Anyone with the link can access the note indefinitely	Link becomes invalid after expiry or revocation
Unauthorized long-term access	Possibly if a recipient leaves an organization but still has the link	Access automatically ends at expiry date
Difficulty managing shared access	No visibility or control after link creation	Creators can view, disable, or regenerate links at will

➢ Implement Comment Moderation and Edit Restrictions

Current Risk – Malicious or Inappropriate Comments
Currently, all users with access to a note can comment without automated content filtering, and comments are permanent unless deleted by the creator. This can be abused by posting spam, offensive material, or malicious links.

Proposed Change

- Add content filtering (block scripts, detect spam keywords, strip harmful markup).
- Allow note owners to temporarily disable comments on their notes.
- Provide a report comment option for collaborators, triggering moderation review.

Threat Analysis

Threat Scenario	Before	After
Injection of malicious code (XSS)	Possible via HTML/JavaScript in comments	Sanitization removes dangerous code before saving
Spam or harassment in comments	Persistent until manually deleted by owner	Can be filtered automatically or flagged for review
Comment flooding	Multiple spam comments degrade usability	Owners can disable commenting to stop abuse

➢ Session timeout and Rotation

Current Risks – Session fixation and persistence [2].

The reliance of system on long-lived session tokens introduces two critical vulnerabilities:

- Session fixation Attacks: an attacker can force a user to authenticate with a known session ID. Once the user logs in, the attacker hijacks the authenticated session.
- Prolonged Exposure Window: tokens remain valid indefinitely unless explicitly revoked. If a token is stolen, attackers gain persistent access.

Proposed change

- Inactivity-Based Session Expiry: implementation of 15-minute idle timeout so that sessions automatically expire after inactivity.
- Session rotation for privilege changes: Generate a new session ID after operations related to security and invalidate old tokens.

Threat Analysis

Threat Scenario	Before	After
Session Fixation	High risk because attackers pre-set a session ID, hijacks post-login	Reduced risk (Session tokens rotate on privilege changes, invalidating fixed IDs)
Token theft	It is critical since stolen tokens grant indefinite access	Tokens expire after 15 min inactivity, limiting exposure window

Reference

[1] “Authentication methods: choosing the right type”, National Cyber Security Centre, Last updated: Jun 12th, 2025. [online]. Access Date: 10/8/2025.
Available: https://www.ncsc.gov.uk/guidance/authentication-methods-choosing-the-right-type

[2] Šlekytė, I. “Session Fixation VS. Session Hijacking Attacks: Prevention and the Main Differences”, NordStellar, Last updated: Apr 22nd, 2025. [online]. Access Date: 10/8/2025.
Available: https://nordstellar.com/blog/session-fixation-vs-session-hijacking/

[3] “Token Authentication”, akamai techdocs, [online]. Access Date: 12/8/2025.
Available: https://techdocs.akamai.com/adaptive-media-delivery/docs/add-token-auth

Contribution

Author	Sections Worked On	Contribution (%)
Hanghang Song	1, 3	25
Xiaoping Ma	2, 4	25
Xiayi Yao	2, 4	25
Yue Pan	3, 4	25

Declaration of Usage of Generative AI Model

AI tool was used to enhance clarity and language refinement in this report. All content remains the responsibility of the authors. AI generated suggestions were reviewed and edited to ensure accuracy and alignment with the intended message.

Appendix – Threat Dragon Report

ShareNote Threat Model

Owner: Hanghang Song
Reviewer: Xiaoping Ma
Contributors: Yue Pan, Xiayi Yao
Date Generated: Wed Aug 13 2025

OWASP Threat Dragon

Executive Summary

High level system description

The system under study is a web-based collaborative note-taking app, called ShareNote. The app allows users to create notes, and share them so others can either read or edit notes, depending on permissions set by the creator of the note. Users can also comment on notes.

System Overview

ShareNote allows users to:

Create and edit notes containing text, images, hyperlinks, and internal note references.

Share notes via secret links (with read/edit permissions) or make them publicly readable.

Comment on shared notes (text-only).

Manage access controls (creator can delete comments, restrict editing).

Key Components

Frontend: Web interface (HTTPS, accessible on desktop/mobile).

Backend: On-premises server (firewall-protected, reverse proxy).

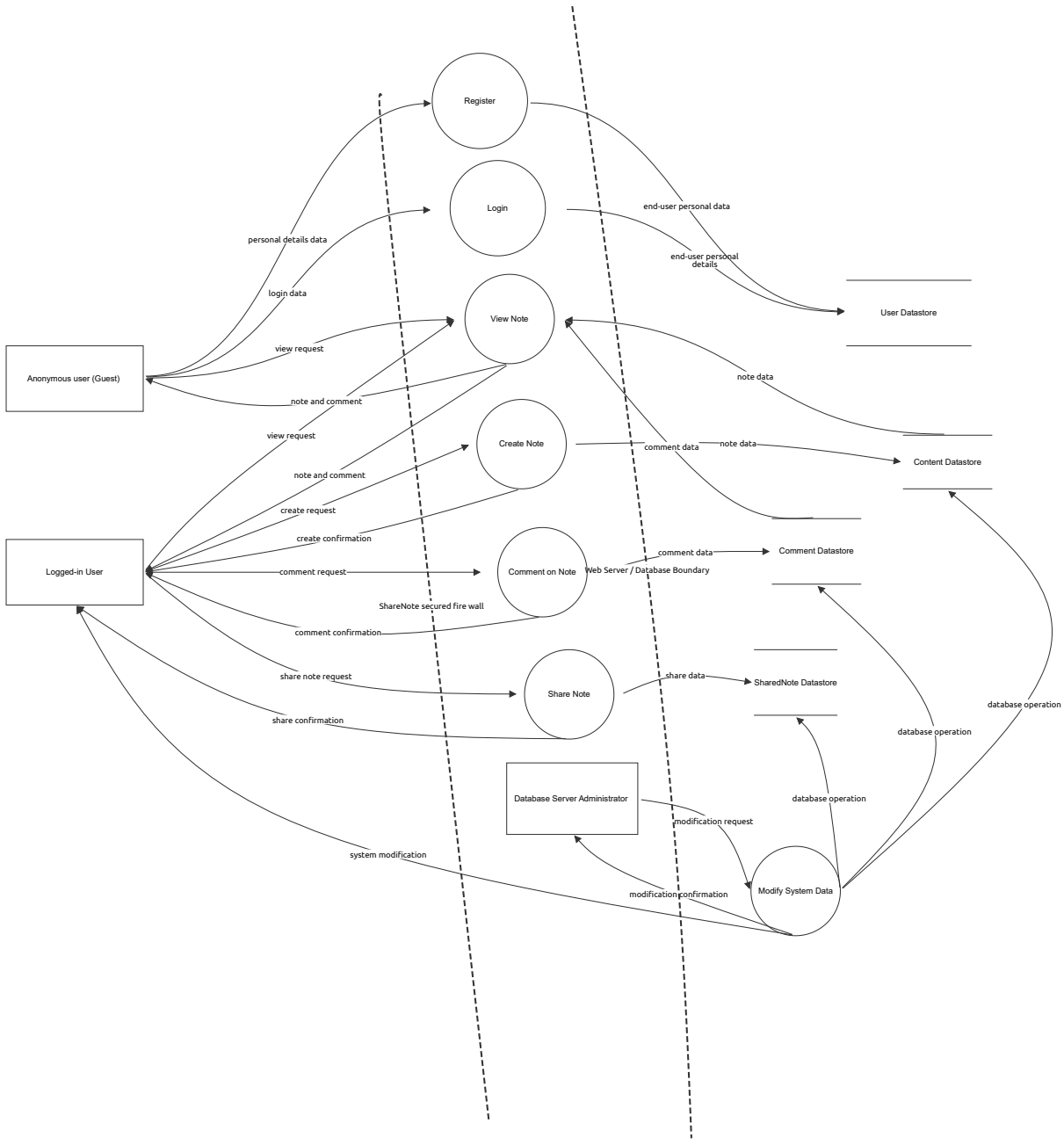
Database: On-premises, not directly internet-accessible.

Summary

Total Threats	93
Total Mitigated	93
Total Open	0
Open / Critical Severity	0
Open / High Severity	0
Open / Medium Severity	0
Open / Low Severity	0

ShareNote STRIDE diagram

ShareNote STRIDE diagram



ShareNote STRIDE diagram

Logged-in User (Actor)

Description: Logged-in users are registered users who have successfully authenticated into the ShareNote system. They can view notes they have access to, create new notes, and share their notes with other users either publicly or through private share links. Additionally, they can comment on notes they have permission to access.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
23	User identity spoofing. Credential too weak or stolen	Spoofing	Medium	Mitigated		Credentials of user can be stolen or guessed. Attacker may impersonate users to gain access to notes	Enforce strong authentication, session management.
24	Repudiation of Performed Actions	Repudiation	High	Mitigated		A logged-in user might deny having performed an action, such as editing a note or sharing it, making it difficult to hold them accountable.	- Maintain tamper-proof audit logs - Include timestamps and user IDs in logs - Digitally sign critical transactions or records - Restrict access to log modification
37	Impersonation of Another User	Repudiation	High	Mitigated		A user may attempt to masquerade as another user by guessing session tokens, forging cookies, or manipulating request headers.	- Enforce strong session management - Validate authentication tokens - Reject requests without valid credentials

create request (Data Flow)

Description: The request data sent by a user to the system when creating a note.
create request data contains:
* userId: String
* title: String
* content: Content

Number	Title	Type	Severity	Status	Score	Description	Mitigations
27	Bypass permission checks	Tampering	Medium	Mitigated		Access or edit notes without authorization	Enforce strict backend authorization, validate every action against role
115	Sensitive data exposure	Information disclosure	Medium	Mitigated		Sensitive information may be transmitted without sufficient protection or accidentally logged. This could allow unauthorized users or attackers to read or misuse the data, especially if intercepted during transit or leaked through error messages.	- Encrypt all communications via TLS - Avoid including sensitive information in error responses

modification request (Data Flow)

Description: The request data sent by an admin to the system when modifying system data.
modification request data contains:
* admin credentials
* operation data

Number	Title	Type	Severity	Status	Score	Description	Mitigations
87	Insecure connection	Tampering	Critical	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.

database operation (Data Flow)

Description: database operation data contains database operation commands

Number	Title	Type	Severity	Status	Score	Description	Mitigations
89	Insecure connection	Tampering	Critical	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.

system modification (Data Flow)

Description: The modification data sent by system to a user when a related modification is completed by an admin.
system modification data contains:
* the data returned by the system and displayed on the user interface, can be note data, comment data, SharedNote data

Number	Title	Type	Severity	Status	Score	Description	Mitigations
81	Insecure connection	Tampering	Medium	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.
121	Sensitive data exposure	Information disclosure	High	Mitigated		Sensitive information may be transmitted without sufficient protection or accidentally logged. This could allow unauthorized users or attackers to read or misuse the data, especially if intercepted during transit or leaked through error messages.	- Encrypt all communications via TLS - Avoid including sensitive information in error responses

note data (Data Flow)

Description: The data sent by the system to the content database when the system is trying to store an note's content data.
note data contains:
* note: Note (note data that stored)

Number	Title	Type	Severity	Status	Score	Description	Mitigations
86	Insecure connection	Tampering	High	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.

comment data (Data Flow)

Description: The data sent by the system to the content database when the system is trying to store data of a comment.
comment data contains:
* comment: Comment (comment data that stored)

Number	Title	Type	Severity	Status	Score	Description	Mitigations
84	Insecure connection	Tampering	High	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.

share data (Data Flow)

Description: The data sent by the system to the content database when the system is trying to store sharing data of a note.

share data contains:
sharedNoteId: String
note: Note
public: Boolean
sharedWith == []: List<User>
accessLevel: Enum {Read, Write}

Number	Title	Type	Severity	Status	Score	Description	Mitigations
85	Insecure connection	Tampering	High	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.

share confirmation (Data Flow)

Description: The response data sent by the system to a user when note sharing is completed.
share confirmation data
- either contains (set private or public):
* message (Indicates success or failure of the operation)
* sharedNoteId: String
* public:Boolean
- or:
* message (Indicates success or failure of the operation)
* shareLink: URL (a shared secret link)
* sharedNoteId: String
* public: Boolean
* accessLevel:Enum {Read,Write}

Number	Title	Type	Severity	Status	Score	Description	Mitigations
80	Insecure connection	Tampering	Medium	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.
120	Sensitive data exposure	Information disclosure	Medium	Mitigated		Sensitive information may be transmitted without sufficient protection or accidentally logged. This could allow unauthorized users or attackers to read or misuse the data, especially if intercepted during transit or leaked through error messages.	- Encrypt all communications via TLS - Avoid including sensitive information in error responses

comment request (Data Flow)

Description: The request data sent by a user to the system when commenting on a note
comment request data contains:
* userId: String
* email: String
* username: String
* commentData: Comment

Number	Title	Type	Severity	Status	Score	Description	Mitigations
77	Insecure connection	Tampering	Medium	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.
117	Sensitive data exposure	Information disclosure	Medium	Mitigated		Sensitive information may be transmitted without sufficient protection or accidentally logged. This could allow unauthorized users or attackers to read or misuse the data, especially if intercepted during transit or leaked through error messages.	- Encrypt all communications via TLS - Avoid including sensitive information in error responses

share note request (Data Flow)

Description: The request data sent by a user to the system when sharing a note.
share note request
- either contains (set private or public):
* ownerUserId: String
* public: Boolean
- or:
* ownerUserId: String
* noteId: String
* linkAccessLevel: Enum{Read, Write}

Number	Title	Type	Severity	Status	Score	Description	Mitigations
79	Insecure connection	Tampering	Medium	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.
119	Sensitive data exposure	Information disclosure	High	Mitigated		Sensitive information may be transmitted without sufficient protection or accidentally logged. This could allow unauthorized users or attackers to read or misuse the data, especially if intercepted during transit or leaked through error messages.	- Encrypt all communications via TLS - Avoid including sensitive information in error responses

database operation (Data Flow)

Description: database operation data contains database operation commands

Number	Title	Type	Severity	Status	Score	Description	Mitigations
91	Insecure connection	Tampering	Critical	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.

database operation (Data Flow)

Description: database operation data contains database operation commands

Number	Title	Type	Severity	Status	Score	Description	Mitigations
90	Insecure connection	Tampering	Critical	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.

comment confirmation (Data Flow)

Description: The response data sent by the system to a user when note commenting is completed.
comment confirmation data contains:
* message (Indicates success or failure of the operation)
- if success:
* commentId: String
* content: Text
* createdAt: DateTime
* author: User

Number	Title	Type	Severity	Status	Score	Description	Mitigations
78	Insecure connection	Tampering	Medium	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
118	Sensitive data exposure	Information disclosure	Medium	Mitigated		Sensitive information may be transmitted without sufficient protection or accidentally logged. This could allow unauthorized users or attackers to read or misuse the data, especially if intercepted during transit or leaked through error messages.	- Encrypt all communications via TLS - Avoid including sensitive information in error responses

create confirmation (Data Flow)

Description: The response data sent by the system to a user when note creation is completed.
create confirmation data contains:
* message (Indicates success or failure of the operation)
- if success:
* redirectNote: Note (object to which the user will be redirected)

Number	Title	Type	Severity	Status	Score	Description	Mitigations
76	Insecure connection	Tampering	Medium	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.
116	Sensitive data exposure	Information disclosure	Medium	Mitigated		Sensitive information may be transmitted without sufficient protection or accidentally logged. This could allow unauthorized users or attackers to read or misuse the data, especially if intercepted during transit or leaked through error messages.	- Encrypt all communications via TLS - Avoid including sensitive information in error responses

comment data (Data Flow)

Description: The data sent by the content database to the system when the system is trying to retrieve a note's comment data.
comment data contains:
* comments: List<Comment> (comment data that requested)

Number	Title	Type	Severity	Status	Score	Description	Mitigations
83	Insecure connection	Tampering	Medium	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.

note data (Data Flow)

Description: The data sent by the content database to the system when the system is trying to retrieve a note's content data.
note data contains:
* note: Note (note data that requested)

Number	Title	Type	Severity	Status	Score	Description	Mitigations
82	Insecure connection	Tampering	High	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.

view request (Data Flow)

Description: The request data sent by a user to the system when request to view a note.
view request data contains:
* noteId: String

Number	Title	Type	Severity	Status	Score	Description	Mitigations
74	Insecure connection	Tampering	Medium	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.
113	Sensitive data exposure	Information disclosure	High	Mitigated		Sensitive information may be transmitted without sufficient protection or accidentally logged. This could allow unauthorized users or attackers to read or misuse the data, especially if intercepted during transit or leaked through error messages.	- Encrypt all communications via TLS - Avoid including sensitive information in error responses

view request (Data Flow)

Description: The request data sent by a user to the system when request to view a note.
view request data contains:
* noteId: String

Number	Title	Type	Severity	Status	Score	Description	Mitigations
72	Insecure connection	Tampering	Medium	Mitigated		for example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.
111	Sensitive data exposure	Information disclosure	High	Mitigated		Sensitive information may be transmitted without sufficient protection or accidentally logged. This could allow unauthorized users or attackers to read or misuse the data, especially if intercepted during transit or leaked through error messages.	- Encrypt all communications via TLS - Avoid including sensitive information in error responses

note and comment (Data Flow)

Description: The response note data sent by the system to a user when viewing a note.
note and comment data contains:
* noteId: String
* title: String
* content: Content,
* createdAt: DateTime
* owner: User
* comments: List<Comment>

Number	Title	Type	Severity	Status	Score	Description	Mitigations
75	Insecure connection	Tampering	Medium	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.
114	Sensitive data exposure	Information disclosure	High	Mitigated		Sensitive information may be transmitted without sufficient protection or accidentally logged. This could allow unauthorized users or attackers to read or misuse the data, especially if intercepted during transit or leaked through error messages.	- Encrypt all communications via TLS - Avoid including sensitive information in error responses

note and comment (Data Flow)

Description: The response note data sent by the system to a user when viewing a note.
note and comment data contains:
* noteId: String
* title: String
* content: Content,
* createdAt: DateTime
* owner: User
* comments: List<Comment>

Number	Title	Type	Severity	Status	Score	Description	Mitigations
73	Insecure connection	Tampering	Medium	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.
112	Sensitive data exposure	Information disclosure	High	Mitigated		Sensitive information may be transmitted without sufficient protection or accidentally logged. This could allow unauthorized users or attackers to read or misuse the data, especially if intercepted during transit or leaked through error messages.	- Encrypt all communications via TLS - Avoid including sensitive information in error responses

modification confirmation (Data Flow)

Description: The data sent by the system to an admin when the system notify the confirmation of modification to the admin.
modification confirmation data contains:
* message (Indicates success or failure of the operation)
* the data returned by the system and displayed on the interface, can be note data, comment data, SharedNote data

Number	Title	Type	Severity	Status	Score	Description	Mitigations
88	Insecure connection	Tampering	Critical	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.

login data (Data Flow)

Description: The request data sent by a user to the system when logging to the system.

login data contains:
* Username / Email – The unique identifier for the user’s account.
* Password – The secret key chosen by the user for account access.
Session Token / Cookie – Created after successful login to keep the user authenticated without re-entering credentials.
* Multi-Factor Authentication (MFA) Code (if implemented) – A one-time passcode sent to email/SMS or generated by an authenticator app.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
98	Insecure connection	Tampering	High	Mitigated		for example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.
110	Sensitive data exposure	Information disclosure	Critical	Mitigated		Sensitive information may be transmitted without sufficient protection or accidentally logged. This could allow unauthorized users or attackers to read or misuse the data, especially if intercepted during transit or leaked through error messages.	- Encrypt all communications via TLS - Avoid including sensitive information in error responses

end-user personal details (Data Flow)

Description: The data sent by the user database to the system when the system is trying to verify an end-user's personal data.
Identity Information
Full name
Username / display name
Profile picture
Contact Information
Email address
Phone number (if required for MFA or notifications)

Number	Title	Type	Severity	Status	Score	Description	Mitigations
100	Insecure connection	Tampering	Critical	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.

personal details data (Data Flow)

Description: The request data sent by a user to the system when registering.
Personal details data contains:
 * full name
 * address
 * mobile phone number
 * email address
 * password

Number	Title	Type	Severity	Status	Score	Description	Mitigations
104	Personal data can be stolen	Information disclosure	High	Mitigated		An adversary in the middle can steal the personal data of the user.	Use Transport Layer Security (TLS) for the communication.
109	Insecure connection	Tampering	Critical	Mitigated		for example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.

end-user personal data (Data Flow)

Description: The data sent by the system to the user database when the system is trying to store an end-user's personal data.
end-user personal data contains:
* Identity Information, this includes:
Full name
Username / display name
Profile picture
Contact Information
Email address
Phone number (if required for MFA or notifications)

Number	Title	Type	Severity	Status	Score	Description	Mitigations
105	Insecure connection	Information disclosure	Critical	Mitigated		For example, data can be tampered by an adversary in the middle where the address for the delivery can be modified.	Use Transport Layer Security for the communication.

Create Note (Process)

Description: Action for a user to create a note for themselves in the ShareNote system.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
4	HTTPS downgrade attack	Tampering	Low	Mitigated		Attempt to access site via HTTP	- Enforce HTTPS with HSTS headers - Redirect HTTP request to HTTPS
44	Excessive Note Creating Requests	Denial of service	Medium	Mitigated		A user sends a large number of create requests, overloading the server or database.	- Enable DDOS protection on reverse proxy server - Enable CAPTCHA - Apply rate limiting per user/IP - Cache non-sensitive content - Monitor and block abnormal request patterns
52	Impersonation of Another User to Create Note	Spoofing	Medium	Mitigated		An attacker attempts to create a note under another user’s account by forging authentication credentials or session tokens.	- Validate session tokens for each request - Enforce server-side checks to confirm note ownership - Use Multi-Factor Authentication (MFA) for sensitive actions

Number	Title	Type	Severity	Status	Score	Description	Mitigations
56	Injection of Malicious Content	Tampering	Critical	Mitigated		A user inserts harmful scripts or malicious code into the note content to exploit other users when the note is viewed.	- Implement server-side input validation and sanitization - Use context-aware output encoding - Apply Content Security Policy (CSP) to prevent script execution
57	Repudiation of Note Creation	Repudiation	Medium	Mitigated		A user denies creating a specific note, making it hard to hold them accountable.	- Maintain tamper-proof creation logs - Include user ID, timestamp, and note metadata in logs - Store logs in an append-only or immutable format
58	Unauthorized Access to Sensitive Data	Information disclosure	Critical	Mitigated		The note creation process inadvertently exposes sensitive information through debug messages, error responses, or unsecured storage.	- Avoid including sensitive data in client responses - Use secure error handling without revealing internal details - Encrypt sensitive fields in storage
61	Escalating Privileges through Note Creation	Elevation of privilege	Critical	Mitigated		A user exploits the note creation function to gain unauthorized privileges, such as setting themselves as the owner of another user's note.	- Enforce strict server-side ownership assignment - Validate all role and permission changes - Implement role-based access control (RBAC) at the database level

Content Datastore (Store)

Description: This database contains all note content data.
every entry contains:
* text: String
* images: List<Image>
* links: List<Link>
* referencedNotes:List<Note>

Number	Title	Type	Severity	Status	Score	Description	Mitigations
92	Datastore unwanted access	Information disclosure	Critical	Mitigated		The datastore must be access-protected to only authorised personnel.	- Datastore credentials are securely stored on password vaults with limited set of users having access (apply least privilege principle). - Datastore credentials are injected into connecting software from secured variables or files (apply least privilege principle).

Comment Datastore (Store)

Description: This database contains all comment data.
every entry contains:
* commentId: String
* content: Text
* createdAt: DateTime
* author:User

Number	Title	Type	Severity	Status	Score	Description	Mitigations
93	Datastore unwanted access	Information disclosure	Critical	Mitigated		The datastore must be access-protected to only authorised personnel.	- Datastore credentials are securely stored on password vaults with limited set of users having access (apply least privilege principle). - Datastore credentials are injected into connecting software from secured variables or files (apply least privilege principle).

Comment on Note (Process)

Description: Action for a user to comment on a specific note they can access in the ShareNote system.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
8	Note or comment tampering	Tampering	High	Mitigated		Unauthorized editing or deletion via compromised session or link without authorization.	- Use secure tokens, validate permissions on every request. - Implement role-based permissions for comment management
9	Denial of creating/editing notes or comments	Repudiation	Medium	Mitigated		A user denies having created or removed a comment, making it difficult to trace actions.	- Maintain tamper-proof comment logs - Include user ID, timestamp, and note ID in records - Store logs in an append-only format
11	Flooding note creation/comments	Denial of service	Medium	Mitigated		A malicious user sends a large number of comment requests, overloading the server or database and overwhelm storage.	- Limit number of comment submissions per user under the same note - Enable DDOS protection on reverse proxy server - Enable CAPTCHA - Apply rate limiting per user/IP - Cache non-sensitive content - Monitor and block abnormal request patterns
49	HTTPS downgrade attack	Tampering	Low	Mitigated		Attempt to access site via HTTP	- Enforce HTTPS with HSTS headers - Redirect HTTP request to HTTPS
53	Impersonation of Another User to Comment on Note	Spoofing	Medium	Mitigated		An attacker attempts to comment on a note under another user’s account by forging authentication credentials or session tokens.	- Validate session tokens for each request - Enforce server-side checks to confirm note ownership - Use Multi-Factor Authentication (MFA) for sensitive actions
59	Unauthorized Access to Sensitive Data	Information disclosure	Critical	Mitigated		The commenting process inadvertently exposes sensitive information through debug messages, error responses, or unsecured storage.	- Avoid including sensitive data in client responses - Use secure error handling without revealing internal details - Encrypt sensitive fields in storage
62	Leaking Sensitive Information Through Comments	Information disclosure	High	Mitigated		Comments may inadvertently contain sensitive content (e.g., private note details, credentials) that becomes visible to others.	- Educate users about safe content in comments - Allow note owners to delete inappropriate comments - Restrict comment visibility based on note access
63	Injection of Malicious Content	Tampering	High	Mitigated		A user inserts harmful scripts or malicious code into the comment to exploit other users when the note is viewed.	- Implement server-side input validation and sanitization - Use context-aware output encoding - Apply Content Security Policy (CSP) to prevent script execution

Share Note (Process)

Description: Action for a note creator to share a note by making it public in read-only mode or by generating a secret link granting read or edit access to specific users.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
13	Shared link misuse	Spoofing	Medium	Mitigated		Link-based access lacks identity verification.	Add optional login requirement for link-based access.
14	Replay attacks with links	Tampering	Medium	Mitigated		Editing via previously shared links.	Implement link expiry, one-time use options.
16	Leaking Secret Share Links	Information disclosure	Critical	Mitigated		Private share links may be exposed via referrer headers, browser history, or intercepted if not protected. Anyone with the link may access sensitive content.	- Generate high-entropy, unguessable tokens - Allow owners to revoke share links and set expiration times
17	Public notes indexed by search engines	Information disclosure	Medium	Mitigated		Public notes may unintentionally expose data.	Use robots.txt, add warnings before making notes public.
25	Excessive Viewing Requests	Denial of service	Medium	Mitigated		A user sends a large number of share note requests, overloading the server or database.	- Enable DDOS protection on reverse proxy server - Enable CAPTCHA - Apply rate limiting per user/IP - Cache non-sensitive content - Monitor and block abnormal request patterns
51	HTTPS downgrade attack	Tampering	Low	Mitigated		Attempt to access site via HTTP	- Enforce HTTPS with HSTS headers - Redirect HTTP request to HTTPS
55	Impersonation of Another User to Share Note	Spoofing	High	Mitigated		An attacker attempts to share a note under another user’s account by forging authentication credentials or session tokens.	- Validate session tokens for each request - Enforce server-side checks to confirm note ownership - Use Multi-Factor Authentication (MFA) for sensitive actions
60	Unauthorized Access to Sensitive Data	Information disclosure	Critical	Mitigated		The sharing note process inadvertently exposes sensitive information through debug messages, error responses, or unsecured storage.	- Avoid including sensitive data in client responses - Use secure error handling without revealing internal details - Encrypt sensitive fields in storage
64	Altering Share Permissions	Tampering	Critical	Mitigated		A malicious user modifies parameters (e.g., accessLevel, public flag) in the share request to grant unauthorized access.	- Validate all input server-side - Enforce role-based access control (RBAC) at the API level - Reject unauthorized changes to note permissions
65	Denying a Share Action	Repudiation	Medium	Mitigated		A user claims they never shared a note, making it difficult to prove the action took place.	- Maintain tamper-proof share activity logs - Include user ID, note ID, access level, and timestamp - Protect logs from modification
66	Escalating Access Through Sharing	Elevation of privilege	High	Mitigated		A user with limited permissions shares a note in a way that grants themselves or others higher privileges (e.g., read → edit).	- Validate permission changes on the server - Restrict share operations to authorized roles - Require explicit confirmation for privilege-increasing shares

SharedNote Datastore (Store)

Description: This database contains all note sharing information data.
every entry contains:
* sharedNoteId: String
* note: Note
* public: Boolean
* sharedWith: List<User>
accessLevel:Enum {Read, Write}

Number	Title	Type	Severity	Status	Score	Description	Mitigations
94	Datastore unwanted access	Information disclosure	Critical	Mitigated		The datastore must be access-protected to only authorised personnel.	- Datastore credentials are securely stored on password vaults with limited set of users having access (apply least privilege principle). - Datastore credentials are injected into connecting software from secured variables or files (apply least privilege principle).

Database Server Administrator (Actor)

Description: Database server administrator are privileged users with control over the ShareNote system. They can manage user accounts, assign permissions, delete content, perform backups, and directly modify the database through administrative processes.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
38	Impersonation of an Administrator Account	Spoofing	Critical	Mitigated		An attacker or malicious insider may attempt to gain administrator privileges by stealing admin credentials, forging admin session tokens, or exploiting authentication flaws.	- Require Multi-Factor Authentication (MFA) for admin logins - Restrict admin access to trusted IP ranges or VPN - Use strong, unique passwords with enforced rotation - Monitor and alert on unusual login activity
39	Repudiation of Administrative Actions	Repudiation	Critical	Mitigated		An administrator could deny having performed a system change (e.g., deleting a note, modifying the database), making accountability difficult.	- Maintain secure, tamper-proof audit logs - Digitally sign and timestamp all administrative actions - Store logs in an append-only system - Regularly review logs with separation of duties

Modify System Data (Process)

Description: Action for an administrator to directly update, insert, or delete records in the database, including user accounts, notes, comments, and system configurations.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
29	Inadequate or Tampered Audit Logging	Repudiation	Medium	Mitigated		If administrative actions in the Modify System Data process are not logged, or if logs can be altered or deleted, malicious changes to system configuration, user data, or permissions may go unnoticed. This can allow a rogue admin or attacker to deny making changes and hinder incident investigations.	- Implement tamper-proof, append-only audit logs. - Record user ID, timestamp, action details, and affected records for every admin change. - Regularly back up logs and monitor them for suspicious activity.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
67	Impersonating an Administrator	Spoofing	Critical	Mitigated		An attacker gains access to admin credentials or session tokens and uses the modify system data process to alter critical records.	- Enforce Multi-Factor Authentication (MFA) for admin accounts - Restrict admin access to trusted IPs or VPNs - Monitor and alert on unusual admin activity
68	Repudiation of Administrative Actions	Repudiation	Critical	Mitigated		An administrator could deny having performed a system change (e.g., deleting a note, modifying the database), making accountability difficult.	- Maintain secure, tamper-proof audit logs - Digitally sign and timestamp all administrative actions - Store logs in an append-only system - Regularly review logs with separation of duties
69	Exposing Sensitive Data During Modification	Information disclosure	Critical	Mitigated		Admin changes may reveal sensitive data in logs, error messages, or insecure channels.	- Mask sensitive fields in logs - Use secure channels for all admin operations (HTTPS/SSH) - Restrict access
70	Privilege Abuse	Elevation of privilege	Critical	Mitigated		Admin (Database Server Administrator and Internal Staff) uses modify system data process to grant themselves or another account higher privileges than intended.	- Separate admin roles (e.g., DB admin vs. app admin) - Require change approvals for privilege alterations - Periodically review account privileges
71	Resource Exhaustion Through Bulk Changes	Denial of service	Medium	Mitigated		Admin runs heavy modification scripts or queries that overwhelm the database or lock key tables.	- Limit the size and frequency of bulk operations - Schedule large changes during maintenance windows - Implement query execution timeouts

Anonymous user (Guest) (Actor)

Description: Guests are users who have not logged into the ShareNote system. They can view public notes but cannot create, edit, comment on, or share notes.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
34	Impersonation of Another User	Spoofing	High	Mitigated		A guest may attempt to masquerade as another user by guessing session tokens, forging cookies, or manipulating request headers.	- Enforce strong session management - Validate authentication tokens - Reject requests without valid credentials
35	False Action Denial	Repudiation	TBD	Mitigated		Unauthenticated guests can perform actions (e.g., access public content) and later deny having done so, making tracking difficult.	- Use IP-based logging - Implement request fingerprinting - Append anonymous session IDs to activity logs

View Note (Process)

Description: Action for a user to view a specific note they have permission to access in the ShareNote system

Number	Title	Type	Severity	Status	Score	Description	Mitigations
40	Unauthorized Access to Notes	Spoofing	High	Mitigated		An attacker pretends to be an authorized user (e.g., by using stolen session tokens) to view notes they shouldn't have access to.	- Validate session tokens for every request - Enforce strict server-side access control - Implement Multi-Factor Authentication (MFA) for sensitive accounts
41	Manipulating Request Parameters	Tampering	Critical	Mitigated		A user alters parameters (e.g., noteId) in the request to view another user's note without permission.	- Perform server-side authorization checks - Validate all input against expected formats
42	Exposure of Sensitive Information	Information disclosure	Critical	Mitigated		Viewing a note may expose sensitive content to unauthorized parties if access control is flawed or the note is cached in public areas.	- Enforce strict access checks before returning content - Prevent caching of sensitive pages
43	Excessive Viewing Requests	Denial of service	Medium	Mitigated		A user sends a large number of view requests, overloading the server or database.	- Enable DDOS protection on reverse proxy server - Enable CAPTCHA - Apply rate limiting per user/IP - Cache non-sensitive content - Monitor and block abnormal request patterns
46	Bypassing Read-Only Restrictions	Elevation of privilege	High	Mitigated		A user with read-only access exploits a flaw to modify the note while viewing it.	- Enforce permissions at the API and database level - Use role-based access control - Validate all requests against allowed actions
47	HTTPS downgrade attack	Tampering	Low	Mitigated		Attempt to access site via HTTP	- Enforce HTTPS with HSTS headers - Redirect HTTP request to HTTPS

Login (Process)

Description: To access the ShareNote system, a user must perform the login process, which involves entering their registered credentials, such as username and password, into the system's authentication interface. This action verifies the user's identity and grants them access to the features and resources available within the platform.

Number	Title	Type	Severity	Status	Score	Description	Mitigations
95	Unauthorized Access to Account	Spoofing	Critical	Mitigated		An attacker pretends to be an authorized user (e.g., by using stolen session tokens) to login in the account they shouldn't have access to.	- Validate session tokens for every request - Enforce strict server-side access control - Implement Multi-Factor Authentication (MFA) for sensitive accounts
96	Manipulating Request Parameters	Tampering	Critical	Mitigated		A user alters parameters (e.g., noteId) in the request to view another user's note without permission.	- Perform server-side authorization checks - Validate all input against expected formats
97	Exposure of Sensitive Information	Information disclosure	Critical	Mitigated		Viewing a note may expose sensitive content to unauthorized parties if access control is flawed or the note is cached in public areas.	- Enforce strict access checks before returning content - Prevent caching of sensitive pages

User Datastore (Store)

Description: This database contains all user data.
every entry contains:
* userId: String
* username: String

* email: String

Number	Title	Type	Severity	Status	Score	Description	Mitigations
99	Datastore unwanted access	Information disclosure	Critical	Mitigated		The datastore must be access-protected to only authorised personnel.	- Datastore credentials are securely stored on password vaults with limited set of users having access (apply least privilege principle). - Datastore credentials are injected into connecting software from secured variables or files (apply least privilege principle).

Register (Process)

Description: Action to register a user account to the ShareNote app

Number	Title	Type	Severity	Status	Score	Description	Mitigations
101	Adversary in the middle attack	Information disclosure	High	Mitigated		A man-in-the-middle could steal the credentials as the user registers.	* Use secured connection protocol. * Ensure IP is consistent for the whole process.
102	User uses an email they don't own	Repudiation	Medium	Mitigated		The user pretends they have access to an email address they don't own.	* Ensure the email is not already in use. * Send a confirmation email with unique token to confirm email ownership.
103	Flooding of requests from same IP	Denial of service	Medium	Mitigated		Many register request per second coming from the same IP.	* Enable a CAPTCHA-like feature * Disallow multiple requests from same origin IP (IP blocking after multiple calls)