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).

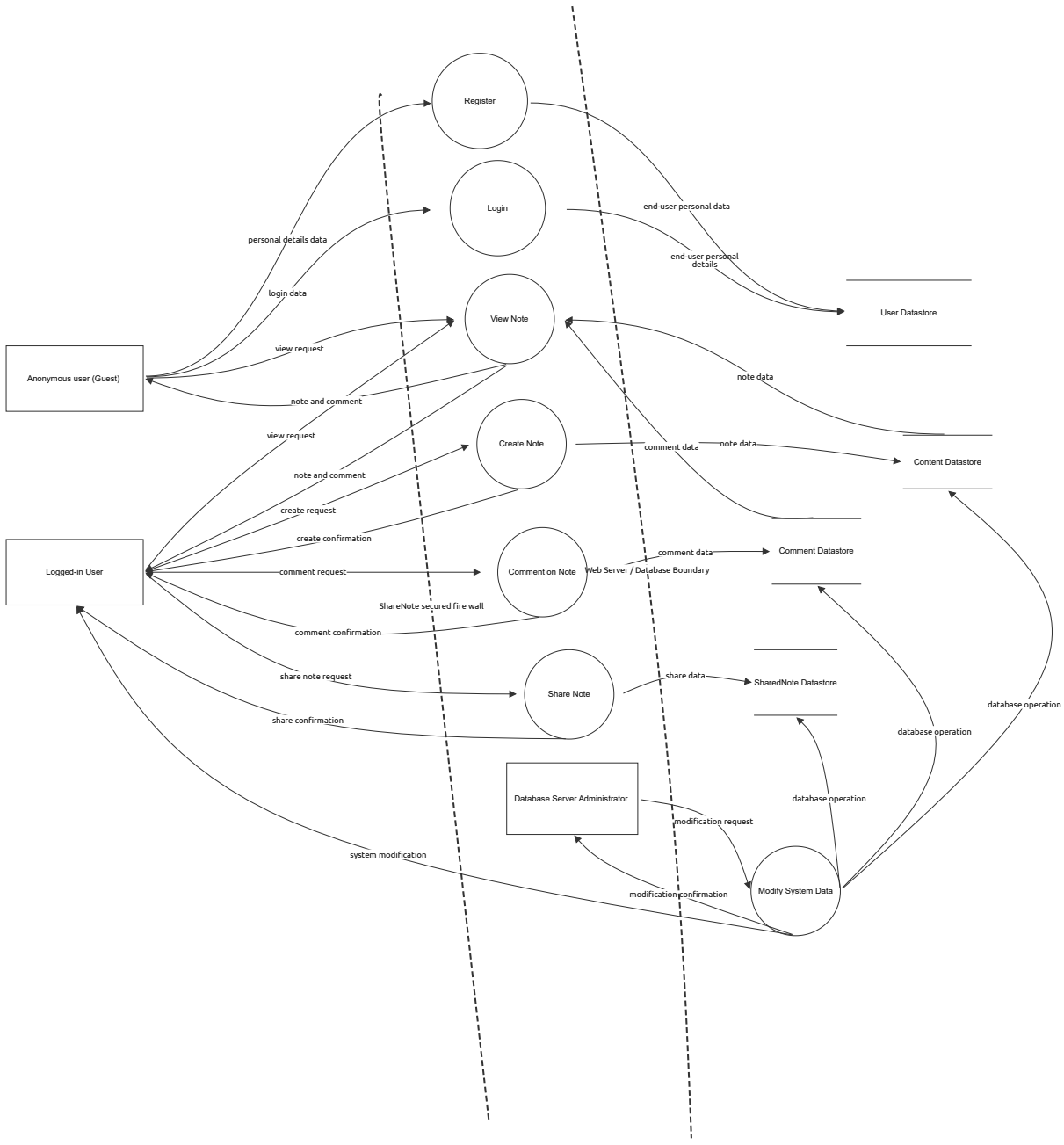
Database: On-premises, not directly internet-accessible.

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)