Security and privacy

How your devices, a huddle and the Munkki backend handle what you share.

Words you'll see in Munkki

Munkki
The app for sharing everyday moments and making plans together.

A munkki
A short personal message, such as "Thinking of you" or a note you write.

Album
A collection of memories about a trip, person or topic.

Huddle
A shared space in Munkki with its own members and encryption keys.

Memory
A post that can include text, photos, video, voice, files or a place.

Plan
Something you want to do, with or without a date. You can link it to a memory later.

What end-to-end encryption means here

Munkki encrypts huddle content before it leaves your device. The recipient's app opens it using the right huddle key. This covers memory text and attachments, the contents of albums and plans, comments and answers, and the message content of a munkki and its replies.

Your device	→	Munkki backend	→	Recipient's device
Encrypts the post using keys for the destination huddle.		Stores encrypted content and the information needed to sync it.		Fetches permitted content and opens it with its local key.

How a post moves between devices. The app connects to the backend over HTTPS.

> **What the backend can see**
>
> The backend can see which account belongs to a huddle, when a record was created, and where to deliver it. The text or media inside the encrypted payload is protected by its key.

This guide describes the app design reviewed in November 2026.

SHARING

Who gets to see what

A huddle has its own keys. The service also checks your account's permissions when you request content.

WHEN YOU…	HERE'S WHAT HAPPENS
Invite someone	An owner uses an invite ID, link or QR code to add an account. The app sends encrypted copies of the relevant huddle keys to registered devices.
Choose full history	The new member can get access to history within the inviter's allowed window, using older key versions that the inviting device has available.
Choose start fresh	The service applies a start date to the new member's access. The app shares the current key version. Earlier content can have been encrypted with that same version.
Remove a member	The app ends their membership and then attempts to change the key for future posts. A key change takes effect once new content uses the new key. People can keep anything they already downloaded.
Post to several huddles	Your device creates a separate copy of the memory and media for each huddle and encrypts it with that huddle's key.
Send a munkki or reply	The service restricts access to a conversation to its two participants. The message content uses the huddle key for encryption.

Features that control when something appears

View once. The server records that a person has opened the memory and can hide it from that viewer on later requests. Someone who already saved a copy can keep it.

Future letters and hidden answers. The app can wait until a reveal date to display a letter, or wait until you answer to show other answers. The content uses the normal huddle encryption. The timing is controlled by the app.

Albums and exports. Albums group memories inside a huddle. When you create a ZIP, save to your gallery, or share a rendered film or yearbook, the destination receives a readable copy.

KEYS AND RECOVERY

Keys for a new device

Signing in connects a device to your account. A device also needs the keys for the content you want to open.

Recover with your phrase

Encrypted backups
Your device downloads the sealed huddle key bundles.

→

Restore on your device
Enter your phrase so the app can derive a key locally.

→

Keys on your device
The app opens the bundles and saves the recovered keys.

The phrase stays on your device during this restore. New setups generate six words in a specific order.

A recovery phrase protects the backup. The server stores encrypted key bundles and can store an encrypted phrase verifier. Anyone with both your phrase and the backup can recover the bundled keys.

Ask another device

Approval. An eligible device can approve the request and encrypt huddle keys for the new device's public key. The app shows a short safety code you can compare.

Your own devices. An approval from a device on the same account can also send encrypted recovery words to the new device.

The algorithms

PURPOSE	WHAT MUNKKI USES
Encrypt content	ChaCha20-Poly1305 protects both confidentiality and integrity. Each media object uses a fresh file key. The huddle key encrypts that file key.
Share a key with a device	X25519 establishes a shared secret with the device. SHA-256 derives the wrapping key and ChaCha20-Poly1305 seals the huddle key. Routine invites use registered device public keys.
Protect a recovery backup	Argon2id turns the phrase into an encryption key. New backups use 64 MiB of memory, four iterations and a random salt. ChaCha20-Poly1305 seals the backup.

SERVER STORAGE

What the backend stores

Munkki uses Supabase for account sign-in, the database and file storage. Different kinds of data have different protections.

WHERE	WHAT'S STORED THERE
Account and membership	Login information, account and device IDs, registered public keys, huddle membership, roles and access dates. These are readable to the backend so it can authenticate and authorize requests.
Content records	Encrypted payloads for memories, huddle and member profiles, album details, plan content, comments and messages. Operational fields such as author ID, row time and key version sit alongside the payload.
Media storage	Encrypted file bytes, an encrypted file key and information needed for decryption. Object names, sizes and some metadata remain readable. Upload metadata can include a filename, media type and deletion date.
Key delivery and recovery	Device-specific encrypted copies of huddle keys, encrypted recovery bundles and account recovery verifier. The server also stores request status, device IDs, huddle IDs and backup format information.
Other features	Community polls, release notes and service records use ordinary database fields. Lookup requests, such as a location search or a GIF request, may go through Munkki's backend and then to the provider.

How notifications work

Munkki backend
Routes a regular huddle alert using the recipient's device token.

→

Push services
Firebase and Apple relay encrypted content and delivery data.

→

Your device
Can decrypt the preview and show it in the notification tray or on the lock screen.

The push path can include identifiers and generic alert text. Key access requests use a separate generic alert.

Deleting content. A deletion removes the service copy according to the feature's rules. Some media cleanup runs separately from row deletion. An exported file or a copy saved by a recipient has its own lifecycle.

DEVICE STORAGE

What stays on your device

Your device needs readable content to show it to you. That can leave copies in app storage, in system features, or wherever you choose to export them.

ON YOUR DEVICE	WHAT MUNKKI KEEPS OR CREATES
Secure key storage	The device's private identity key, huddle keys and the recovery secret, when stored, use secure storage. iOS uses Keychain. Android uses secure storage backed by Android Keystore.
Local database and sign-in	The local memory database keeps encrypted payloads alongside indexes such as memory IDs and dates. App preferences keep session information so the app can resume your sign-in.
Media cache and upload queue	Munkki can write readable media to the app's cache for faster access. Pending upload files and memory payloads are encrypted in the upload queue, alongside upload and progress information.
Widgets and notifications	Widget pictures and text can be saved locally for display. On iOS, the notification extension can access huddle keys after the first device unlock. Android can keep readable conversation previews for notifications.
Imports, playback and exports	Files sent into Munkki can be staged locally before you post. Playback and export can create temporary readable files. Gallery copies, exported ZIPs, and files sent to other apps follow the destination's rules.

Common questions

Can someone save what I share? A recipient who can see a post can keep a copy. This applies even when a feature hides it later or the original is deleted.

What happens when I sign out? Munkki's sign-out flow attempts to clear account keys, app caches, pending posts and notification data. Copies you saved in your gallery or sent to another app remain with that destination.

Why does the server need metadata? The service needs to know who belongs to a huddle, where a post belongs, and which devices should get an alert. That information supports delivery and access checks while the content itself stays encrypted on its way through the service.