

Full audio explanation from employer:

And it's going to mean more development time because basically his concept is, you should be able to press a button and then put in the email address of someone that you want to have a personal repository for and then press send. And that's it. And what Foldy should do is automatically generate a folder for that person that you're sending it to. And then on the user interface side, on our side, the user side, you should be able to easily sort through all the files and folders by email address. And you should be able to see every single file that a certain email address has ever sent you. And it's more complicated. I can get into it. You and I should have a call soon to really discuss products and plan this out because it's got a lot of potential. It's got a lot of potential. I don't know how much development work it's going to be, but that email sorting and the email repository thing is a key feature. That's something really important to him because from his perspective, for example, he says when he does his taxes, he has like 30 people that he has to send and receive files from and to. And he has to make sure all of them can get access to it and all of them can send files to and to each other. And when he wants to find a file, he has to open up seven different email addresses and search on the email box. And it's so hard to find it. But if on Foldy, like let's say he did his taxes using Foldy, then the folder for his taxes would already include access to like the 30 different people that he sends files to all the time. They send files back to him and he can just search any other email addresses and see all the files they've already sent. Or he can just open that tax folder and see all the files there. And then he can put a file in there and he can make sure everyone else gets it. And when they're emailing each other, all they got to do is send each other that link, that Foldy link. And so yeah, that's the way he's describing it. So I think it makes a lot of sense. The reason that we're going to win is because setting up Foldy files and folders or setting up a Foldy files and folders or setting up a Foldy account is like really quick and easy. Foldy account is like really quick and easy. And setting up the folders is also really easy. But the way it is now, I mean, basically it's different from what I originally imagined. His concepts and what he wants to see is a little bit different, but I think it's much better. So yeah, we want to add that feature, a new feature where you just click a button, you can put in the emails, it'll send them a link right away from foldly.com. And that link will lead directly to this folder specifically for those people and you. And your personal space should be categorizing files by not just folders, but also emails that have been sent to you. Sorry, not just the folders that you've created, but also sort them by the files, which are marked by the emails that have sent you those files. It's complicated. Let's do a call. Let's do a call soon. Sorry.



Foldy – Link/Folder/Email Model

Discussion Summary and Brainstorming



Summarized Audio Explanation

The concept requires more development time because essentially the workflow is:

- A user should be able to press a button, enter the email address of someone they want to have a personal repository for, and press send.
- Foldy should automatically generate a folder for that person.
- On the user interface side, users should be able to:
 - Sort through all files and folders by email address.
 - See every file that a certain email address has ever sent.

This means:

- Email sorting and personal repositories become **key features**.
- For example, when doing taxes with 30+ people, instead of digging through multiple inboxes, Foldy would provide:
 - A dedicated folder for taxes.
 - Built-in access to all 30 people.
 - The ability to search files by email.
 - A shared folder that consolidates everything.
- Everyone can use the Foldy link to share files seamlessly.


Core strength: setting up Foldy accounts and folders is quick and easy compared to existing workflows.

? Q&A / Discussion

Link–Folder Relationship

- **Consideration:** Links must always own folders. No link can exist without a folder.
- If a folder couples to a link, check parent folder: if parent already has that link, duplication should not be possible.

Question: Should links simply be represented as folders in the UI?

- **Option A:** Represent links = folders. Risk of confusion with personal-only folders.
- **Option B:** Show all folders, but visually mark linked ones with a badge (.

 **Recommendation:** Yes, represent as folders, but differentiate linked ones with icons.

Storage Model

- **Consideration:** Files shared via links are stored in Supabase.
- Proposal: Two storages → one for personal files, one for shared files.

Question: Should we use one Supabase storage or two?

- **Option A (One storage):**
 - Easier to implement.
 - Needs metadata tagging for separation.
 - Risk of messiness at scale.
- **Option B (Two storages):**
 - Cleaner separation of personal vs. shared.

- Easier permissions and policies.
- Slightly more infra complexity.

✅ **Recommendation:** Use two storages for clarity and scalability.

✉ **Emails and Access Control**

- **Consideration:**
 - Users can add/remove emails from access list.
 - Links can be public or private.
 - Public links accept uploads from anyone.

Question: What happens when switching between public/private?

- **Option A (Lenient):** Public uploads auto-append emails to access list. Switching to private keeps them.
- **Option B (Strict):** Public uploads don't alter the list. Only manual additions are allowed.

⚖ **Trade-off:**

- **Lenient = dynamic list growth** (continuity).
 - **Strict = user-controlled only** (cleaner, stricter).
-

🗑 **Folder/Link Deletion**

Question: What happens when a folder tied to a link is deleted?

- **Option A:** Cascade delete (remove folder, link, and all tied emails).
- **Option B:** Archive state (inactive but recoverable).

- **Option C:** Detach link but keep folder.

✓ **Recommendation:** Cascade delete for simplicity.

Filtering by Email

Question: How to filter files shared by many emails?

- **Option A:** Tag every file with link_id + uploader_email.
- **Option B:** Group uploads by email.
- **Option C:** Use tags only for permissions.

✓ **Recommendation:** Store uploader_email per file for granular filtering.

Removing Emails from Access List

Question: What happens to files uploaded by an email after removal?

- **Option A (Strict):** Delete files uploaded by that email.
- **Option B (Lenient):** Keep files, block future uploads.
- **Option C:** Let owner choose per removal event.

✓ **Recommendation:** Use lenient model (keep files, block new uploads).

Edge Case: Public Links & Repeated Uploaders

Scenario: Public link + same email uploads multiple times.

- If email is already on access list → uploads proceed.
- If not on list → two approaches:

1. Auto-append once

- First upload adds email to access list.
- If link later turns private, that email still has access.

2. No auto-append

- Public uploads are allowed but don't add emails.
- Switching to private means that uploader loses access unless added manually.

Trade-off:

- **Auto-append once** = smoother continuity.
- **No auto-append** = strict, cleaner lists, more control.

👉 **Decision point:** Should access lists be **dynamic** (grow on public uploads) or **strict** (user-controlled only)?

✅ Final Summary of Recommendations

1. Represent links as folders, but mark shared ones with icons.
2. Use two storages (personal + shared).
3. Cascade delete folders/links with tied emails.
4. Tag files with uploader_email for filtering.
5. Use lenient model for removed emails (keep files, block uploads).
6. Clarify policy for public links: auto-append vs strict control (open design decision).