It provides a sufficient but minimal solution to each of these data problems:

1. storing
2. sharing
3. creating
4. viewing
5. searching
6. automatic manipulation
7. spam messages

Storing

A user's data is stored in a local cache, in IndexedDB in their browser. There is also a storage server, so the data can be synchronised between machines and backed up.

Sharing

There is a message-passing server for sharing data between people. It stores messages till they are collected by the recipient.

The cost of the server is met by subscriptions. It is free to use the server to communicate with a subscriber, but only subscribers can communicate with non-subscribers.

Message formats

Messages are encrypted and decrypted on users' machines, and can't be read by the server. To the server, a message is either:

- a public signing key change message or
- an encrypted blob and its nonce

Each message also contains the public keys of the recipient and sender. A public key change message must contain a cryptographic signature.

The encrypted blob must be no more than 16KB long. Before encryption and encoding it is one of:

1. a new public encryption key
2. a request to be whitelisted, using a one-time code
3. a chunk of a program
4. a chunk of a document

A chunk of a program or document contains:

1. the cryptographic hash of the whole document or program
2. the offset: 0 for the first chunk, 1 for the second, and so on
3. whether the chunk is the final one in the document or program
4. the chunk body

A document contains:

1. a body and
2. the name of the program that can open it.

The body of a document is either a binary blob or some text with links to other documents in it. A link is the cryptographic hash of the document linked to.

A program contains:

1. the code
2. its name
3. a description
4. a version number

The version number is an integer, starting at 0. All versions must be able to read data created by previous versions.

Programs

The actions that a program can do are:

1. Display a document.

2. Access a local database.

3. Read and delete messages sent to it from other people.

4. Read user input. Text documents are displayed as editible text areas. A program can subscribe to be notified of any changes to the text area. A program can also prompt users for a file upload from the local file system.

5. Send messages to other people.

Spam

Each user has a whitelist of people they will accept messages from. Messages from anyone else are rejected unless they have a valid one-time code. Connections are started by somone sending a one-time code to someone else, by some existing method of communication, such as email. This code is used to authenticate the first message.

Software components

1. Message-passing server. Messages are accepted if they are to or from subscribers. It deletes messages when they have been read.
2. Javascript client. It has an inbox categorized by program, and a set of programs. The main view is a list of programs and a box to search for them. Clicking on a program launches it. There is a built-in programming language interpreter for running the programs - probably an editor and tooling all built in too.
3. Backup and synchronisation server.

Security

Each user has a pair of keys for encryption and signing which are generated from a password only known by the user. This means that data is only accessible in unencrypted form by the user, but has the dowside that if the user loses their password and their local data then they can't recover it.

Key pairs are changed by sending the new keys to everyone on the whitelist, signed by the old signing key.