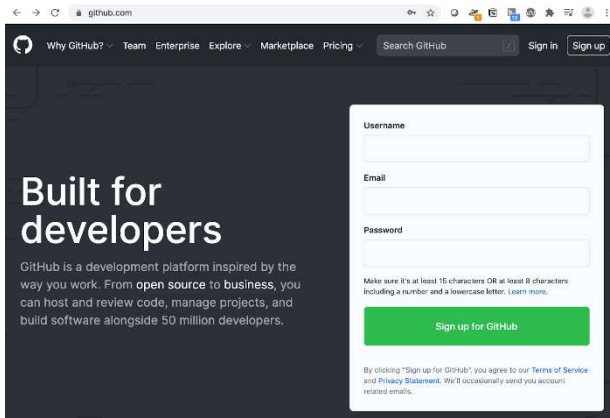


Github can be a little daunting in its terminology and its interface, but sooner or later, anyone working in software has to roll up their sleeves and use it. We made a simple illustrated step-by-step walkthrough. Clicking any of the screengrabs should take you to the page depicted, or at least nearby.



We will start at the very beginning: creating an account. On the home page you click on the green button to get a new account.

Here is the form where you pick a username. If you have an online handle you use other places on the web and want to stay “on-brand,” you can use the same username as you do elsewhere. “Git trails are forever,” though, so many privacy-oriented users and identity people prefer to be as opaque and un-correlated as possible.

My main Github account I use my handle “IdentityWoman” and Juan’s handle is “BumbleFudge”.



Welcome to GitHub

Woohoo! You've joined millions of developers who are doing their best work on GitHub. Tell us what you're interested in. We'll help you get there.

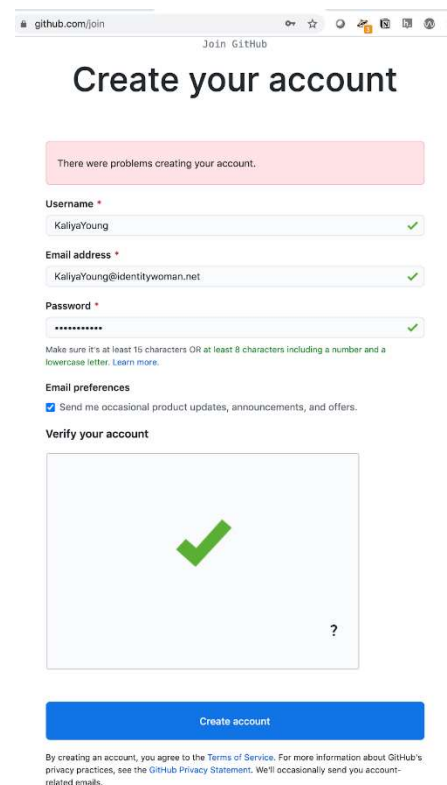
What kind of work do you do, mainly?

Software Engineer I write code	Student I go to school
Product Manager I write specs	UX & Design I draw interfaces
Data & Analytics I write queries	Marketing & Sales I look at charts
Teacher I educate people	Other I do my own thing

How much programming experience do you have?

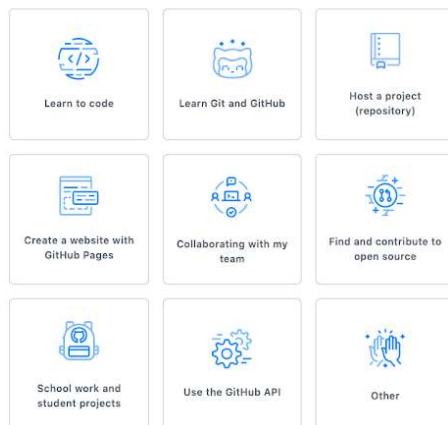
None I don't program at all	A little I'm new to programming
A moderate amount I'm somewhat experienced	A lot I'm very experienced

The first “customization” question tailors recommendations to you about what the user-experience/interface defaults will be. The second determines what kind of tutorials, pop-ups, and explanations will be shown to you when you edit code.


 A screenshot of the GitHub 'Create your account' page. The browser address bar shows 'github.com/join'. The page has a dark header with the GitHub logo and navigation links. The main content area is white and features a large heading 'Create your account'. Below the heading, there's a pink error message 'There were problems creating your account.' followed by a form with fields for 'Username', 'Email address', and 'Password'. The 'Username' field contains 'KaliyaYoung' and the 'Email address' field contains 'KaliyaYoung@identitywoman.net'. The 'Password' field is masked with asterisks. Below the form, there's a checkbox for 'Email preferences' and a 'Verify your account' section with a large green checkmark. At the bottom, there's a blue 'Create account' button.

What do you plan to use GitHub for?

(Select up to 3)

**I am interested in:**

languages, frameworks, industries

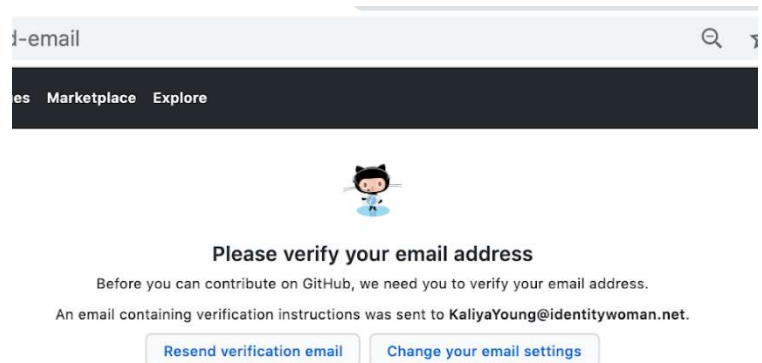
We'll connect you with communities and projects that fit your interests.

For example: windows anki indieweb

Complete setup

The third customization question covers the most common use cases for github, but you might notice “writing specifications” is not one of them—if that is your primary interest, you have to click “Other.” These options also determine default settings and recommendations.

In addition to authoring code and technical documentation, github is also an important place for reading those of others, and more importantly, **finding** them. For the sake of discovery and recommendations of projects you might find interesting to read about, it is worth picking search terms and topics carefully for the fourth personalization question!

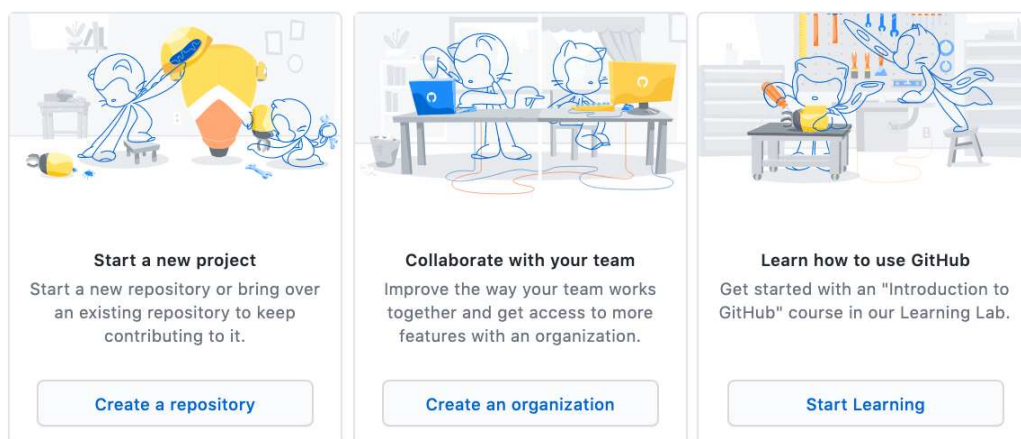


GitHub is also an “**identity provider**” and can be used for single-sign-on at other sites, as well as a target for

hackers, so we recommend not just verifying your email, but also taking advantage of additional security measures like multi-factor authentication and PGP keys, particularly if you are working on consequential projects... like DIF repositories!

What do you want to do first?

Every developer needs to configure their environment, so let's get your GitHub experience optimized for you.

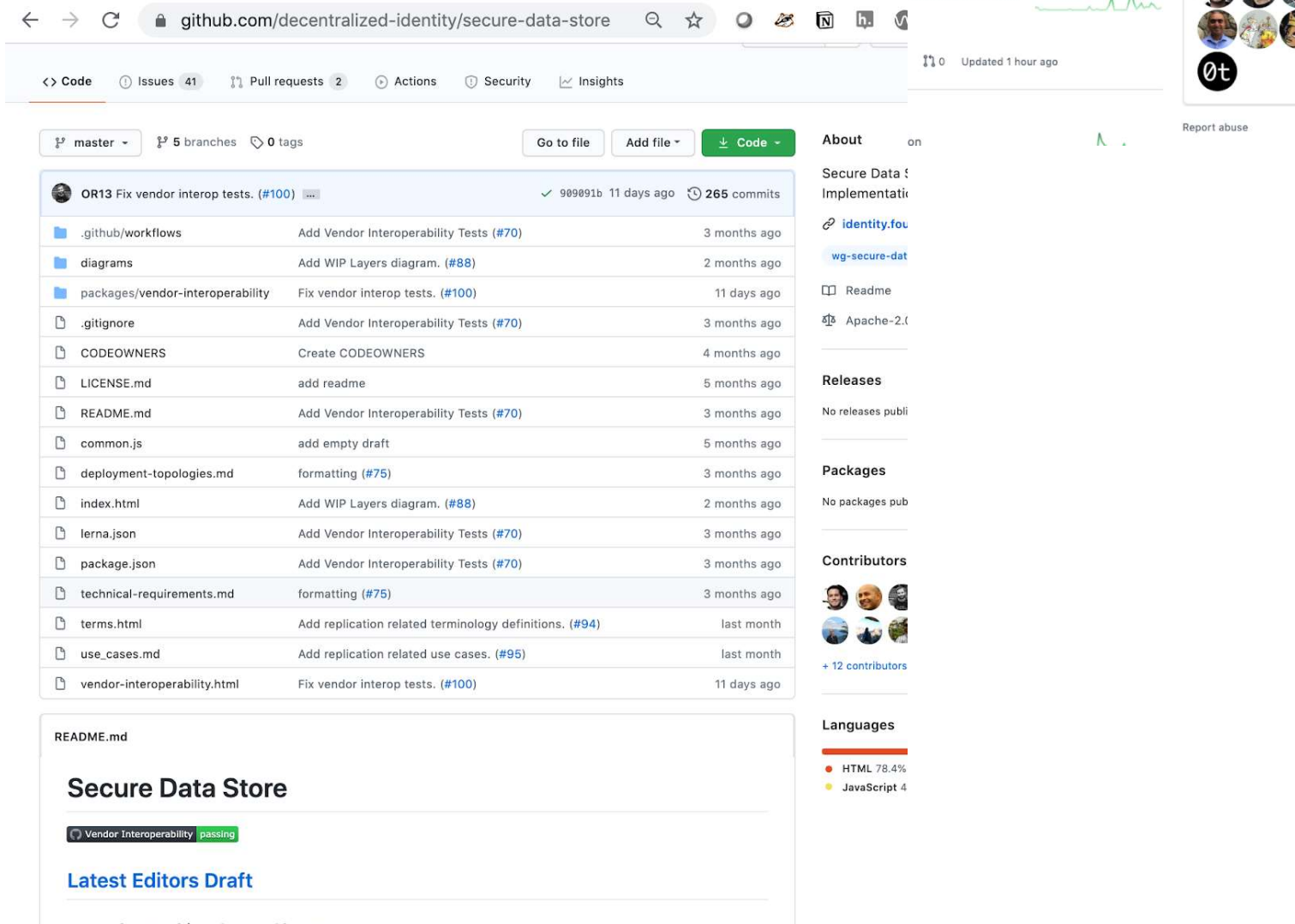
[Skip this for now >](#)

Once you have established your account’s security features, you will be offered some tutorials. All of these are recommended! Feel free to do those now or come back to them later. The rest of this guide will walk you through some specific DIF examples for context, if you prefer examples from DIF’s own backyard.

Typing **decentralized-identity** (with a Z!) into the search bar or going to github.com/decentralized-identity/ will take you to the DIF organization. At time of press, 93 active repositories live here.

I'm the chair of Secure Data Store working group so I figured we could go through a tour of that repository. Below, you can see the home page. Across the top you have tabs showing you different cross-sections of activity:

- Code, i.e., current contents
- Issues, which are threaded discussions
- Pull requests, which represent concrete proposals to update the code/content
- Actions, which are automations and scripts that act on the code
- Security
- Analytical insights about activity



The screenshot shows the GitHub repository page for 'decentralized-identity'. The top navigation bar includes 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. The repository name 'Decentralized Identity Foundation' is displayed with a verified badge and the URL 'https://identity.foundation'. Below this, there are statistics for 'Repositories' (93), 'Packages', 'People' (11), and 'Projects' (1). A search bar and filters for 'Type' and 'Language' are present.

Three repositories are listed:

- did-resolver**: Universal did-resolver for javascript environments. It has 13 forks, 55 stars, and was updated 1 minute ago.
- web-did-resolver**: DID resolver for HTTPS domains. It has 2 forks, 21 stars, and was updated 1 hour ago.
- did-jwt-vc**: Create and verify W3C Verifiable Credentials and Presentations in JWT. It has 0 forks and was updated 1 hour ago.

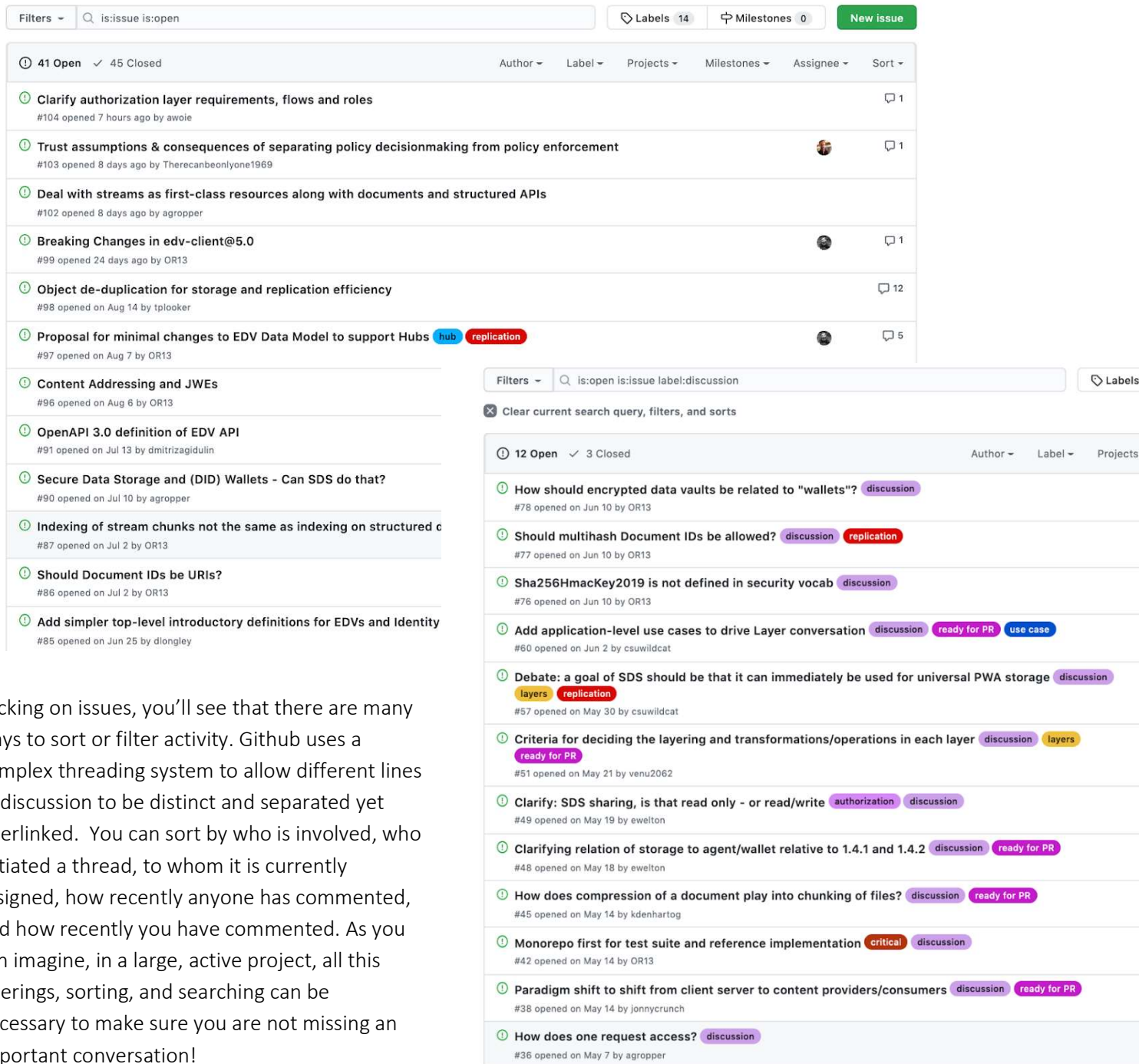
On the right side, there are sections for 'Top languages' (TypeScript, HTML, C#), 'Most used topic' (wg-id, decentral, universal-resolver, universal-registr), and 'People' (a list of contributors).

The main content area shows the 'Secure Data Store' repository. It has 41 issues, 2 pull requests, and 0 tags. The repository is in the 'master' branch with 5 branches and 0 tags. A table of recent commits is shown:

Commit	Description	Time
OR13	Fix vendor interop tests. (#100)	11 days ago
909091b	Add Vendor Interoperability Tests (#70)	3 months ago
	Add WIP Layers diagram. (#88)	2 months ago
	Fix vendor interop tests. (#100)	11 days ago
	Add Vendor Interoperability Tests (#70)	3 months ago
	Create CODEOWNERS	4 months ago
	add readme	5 months ago
	Add Vendor Interoperability Tests (#70)	3 months ago
	add empty draft	5 months ago
	formatting (#75)	3 months ago
	Add WIP Layers diagram. (#88)	2 months ago
	Add Vendor Interoperability Tests (#70)	3 months ago
	Add Vendor Interoperability Tests (#70)	3 months ago
	formatting (#75)	3 months ago
	Add replication related terminology definitions. (#94)	last month
	Add replication related use cases. (#95)	last month
	Fix vendor interop tests. (#100)	11 days ago

The README.md file is visible, showing the title 'Secure Data Store' and a status 'Vendor Interoperability passing'. Below the title, it says 'Latest Editors Draft'.

On the right side of the repository page, there are sections for 'About' (Secure Data Store Implementation, identity.foundation, wg-secure-dat), 'Releases' (No releases published), 'Packages' (No packages published), 'Contributors' (12 contributors), and 'Languages' (HTML 78.4%, JavaScript 4%).



The screenshot displays two views of the GitHub Issues interface. The left view shows a list of 41 open issues, sorted by 'New issue' (top). The right view shows a filtered list of 12 open issues, sorted by 'New issue' (top). Both views include a search bar, filters, and a 'New issue' button.

Left View (41 Open issues):

- Search: `is:issue is:open`
- Filters: 14 Labels, 0 Milestones
- Issues:
 - Clarify authorization layer requirements, flows and roles (#104 opened 7 hours ago by aweio)
 - Trust assumptions & consequences of separating policy decisionmaking from policy enforcement (#103 opened 8 days ago by Therecanbeonlyone1969)
 - Deal with streams as first-class resources along with documents and structured APIs (#102 opened 8 days ago by agroppler)
 - Breaking Changes in edv-client@5.0 (#99 opened 24 days ago by OR13)
 - Object de-duplication for storage and replication efficiency (#98 opened on Aug 14 by tplooker)
 - Proposal for minimal changes to EDV Data Model to support Hubs (#97 opened on Aug 7 by OR13)
 - Content Addressing and JWES (#96 opened on Aug 6 by OR13)
 - OpenAPI 3.0 definition of EDV API (#91 opened on Jul 13 by dmitrizagidulin)
 - Secure Data Storage and (DID) Wallets - Can SDS do that? (#90 opened on Jul 10 by agroppler)
 - Indexing of stream chunks not the same as indexing on structured c (#87 opened on Jul 2 by OR13)
 - Should Document IDs be URIs? (#86 opened on Jul 2 by OR13)
 - Add simpler top-level introductory definitions for EDVs and Identity (#85 opened on Jun 25 by dlingley)

Right View (12 Open issues):

- Search: `is:open is:issue label:discussion`
- Filters: Clear current search query, filters, and sorts
- Issues:
 - How should encrypted data vaults be related to "wallets"? (#78 opened on Jun 10 by OR13)
 - Should multihash Document IDs be allowed? (#77 opened on Jun 10 by OR13)
 - Sha256HmacKey2019 is not defined in security vocab (#76 opened on Jun 10 by OR13)
 - Add application-level use cases to drive Layer conversation (#60 opened on Jun 2 by csuwildcat)
 - Debate: a goal of SDS should be that it can immediately be used for universal PWA storage (#57 opened on May 30 by csuwildcat)
 - Criteria for deciding the layering and transformations/operations in each layer (#51 opened on May 21 by venu2062)
 - Clarify: SDS sharing, is that read only - or read/write (#49 opened on May 19 by ewelton)
 - Clarifying relation of storage to agent/wallet relative to 1.4.1 and 1.4.2 (#48 opened on May 18 by ewelton)
 - How does compression of a document play into chunking of files? (#45 opened on May 14 by kdenhartog)
 - Monorepo first for test suite and reference implementation (#42 opened on May 14 by OR13)
 - Paradigm shift to shift from client server to content providers/consumers (#38 opened on May 14 by jonnycrunch)
 - How does one request access? (#36 opened on May 7 by agroppler)

Clicking on issues, you'll see that there are many ways to sort or filter activity. Github uses a complex threading system to allow different lines of discussion to be distinct and separated yet interlinked. You can sort by who is involved, who initiated a thread, to whom it is currently assigned, how recently anyone has commented, and how recently you have commented. As you can imagine, in a large, active project, all this filterings, sorting, and searching can be necessary to make sure you are not missing an important conversation!

Issues can also be categorized dynamically over time. Clicking on a label takes you to a "survey" of all issues so labeled. The "labels" are used differently in every group, but the general idea is that some topics will span many seemingly unrelated threads, and show up in places you might not expect from the original subject line and participants, yet you want those threads included in a topic-wide check or survey. Many labels are topical (replication, data model, testing, compliance, etc), while others are procedural and common to all kinds of repositories. These include "discussion" (as in, we invite more opinions from everyone), "bikeshed" (not for the faint of opinion or time commitment), and "ready for PR". Since issues and pull requests are cross-linked automatically whenever they refer to each other, issues are often left open until a PR commitment makes them moot or a matter of historical record, at which point they can be closed.

