Improving the Apache Ozone Website

Motivation

Apache Ozone's website and documentation have not been refreshed in a while and need significant improvements. Improving the website will further improve our ability to gain new users and contributors. This has become increasingly evident with the addition of Github Discussions, which are filled with setup and installation questions, which also consume developer time helping to troubleshoot. I believe the limitations with Ozone's documentation and website can be attributed to the following things:

1. **Three different website components trying to function as a cohesive website**
   Ozone documentation is tracked in the main Ozone repository. Developer documentation is tracked in Confluence where anyone with permissions can update it without a review process. All other pieces of the website are tracked in the ozone-site repository, which also attempts to link the other two sources together.
2. **The layout does not scale with new content**
   The layout of the website and docs does not provide robust subsections and visual distinctions to add more content as it is needed. This increases friction when adding new content and pushes developers to favor other options. These include adding it to less visible places like Confluence, or not adding content all together. As new content is added, it incrementally decreases the navigability of the current site.
3. **Unpolished appearance**
   This is a bit more subjective, but I think most people would agree that the fit and finish of the Ozone website could use some improvement. This is the result of multiple smaller issues like inconsistent formatting across different pages, poor choice of color scheme, and our practice of using images with quotes instead of substantive release notes.

I believe these issues are so fundamental to Ozone's website that it is easier to start with a new setup than refresh the current one. This document lays out a structured plan to re-write the Ozone website and its content. While some text content may be reused from existing sources, everything added to the new site should go through a fresh review process.

A Multi-Phased Approach to Rewriting the Website

In order to create a quality website, we must treat it the same as any new Ozone feature. This document attempts to separate the rewrite of the website into phases that easily map to a section of the website and can each be tracked under parent jiras. Each phase consists of small or medium sized tasks that correspond to a smaller piece of the site and may be done in one or more jiras/pull requests. Upon completion of the phase, the exit criteria should be evaluated by both those involved and uninvolved in the changes to ensure the section is complete. Phases should be completed in order to prevent knowledge gaps being left in the site as development progresses. Phases are roughly ordered from least technical to most technical, so there should not be any developers

blocked on earlier phases completing before they can contribute. All phases before Phase 6: Publishing will take place on a feature branch. Merging this branch will publish the site once it reaches feature parity with the current site.

Phase 1: Outline

Tasks

1. Decide where to host the Ozone website source.
    - Should we continue the split model between the ozone and ozone-site repos, or combine everything to one of the repos?
2. Choose a framework to build the new website.
    - Simple hello-world type sites may be created using each of the frameworks to aid in decision making.
    - Criteria for choosing a framework include:
        i. **Maintainability**: How easy is it to understand and update?
        ii. **Features**: Does it provide built in features like indexing/search and versioning with minimal boilerplate?
        iii. **Appearance**: How easy is it to customize and create a nice looking site?
3. Choose a theme for the site.
4. Tools or dependencies to build the site are added to the repo.

Exit Criteria

- A framework to build the site is chosen, and a core team has a basic understanding of how to use it.
- A location to put the new website's source code is chosen.
- Code for a simple hello-world type page is checked in.
- The demo site can be easily built and displayed locally by developers.

Phase 2: Homepage

Tasks

1. Define sections with dropdowns and quick links visible at the homepage header.
2. Banner graphic
3. Quick link buttons in the banner or near the top.
4. A few quick points on the "What" and "Why" of Ozone to view while scrolling down the page.
5. Gifs/graphics to accompany each of the above points.
6. Define footer with links that will be present at the bottom of every page.
    - The content the links point to may not be defined yet.

Exit Criteria

- Home page looks complete. There should be no giveaway that the site is incomplete until a link to a different page is clicked on.
- Graphics and color scheme are readable and cohesive.
- Home page is informative to new users and provides reasons to try out the system.

Phase 3: Non-technical content

Tasks

1. Roadmap page
2. Download page
3. Release notes pages
    - Port notes from old releases to this page as well.
4. Community related pages including:
    - Contacts
    - Meetings
    - Contributing code or documentation updates
    - Filing Issues
5. Links to talks and meetups
6. Companies using Ozone

Exit Criteria

- All non-technical content is easily reachable from the homepage and appears complete.
- New users or developers have no doubts about how to get in contact with maintainers.
- Potential users may be convinced to evaluate the system based on current users, linked talks, community health, and linked question/answer forums.

Phase 4: Quick start for new/potential users and developers

Tasks

1. High level architecture explanation.
    - Excludes component internals.
2. Quick start guide to run Ozone on various platforms from provided templates.
    - Bare metal
    - docker compose
    - K8s
3. Config basics
4. CLI basics
5. Hardware requirements
6. Links to reference architectures

Exit Criteria

- New users or developers can quickly try out Ozone to evaluate it.
- No HDFS background assumed or required.
- No issues requiring assistance are encountered for common case POC setups if the guides are followed.
    - A good test would be to have people with varying levels of experience set up installations using these docs.

Phase 5: Main User Guide

Tasks

1. Feature specific documentation, grouped by component
2. Logging configurations

3. Security
4. ACLs
5. Performance tuning guide
6. Tuning fault detection and recovery mechanisms like heartbeats (DN/SCM and Ratis) and disk checks
7. Decommissioning/maintenance/disk replacement procedures.
8. Upgrades, downgrades, and compatibility guarantees.
9. High level observability with Recon and `ozone admin`.
10. Recon REST API (in swagger or other standard format).
11. Integration with other components:
    - S3 clients/apps
    - Ranger
    - Prometheus
    - Grafana
    - Compute engines via ofs and s3a.

Exit Criteria

- Users should be able to maintain their Ozone cluster and resolve minor issues they encounter.
- New developers should have a high level understanding of the system they are working on.

Phase 6: Publishing

Tasks

1. Framework for handling site versioning with each Ozone release
2. Handle any technical issues with auto publishing changes.
3. (Optional) Integrate documentation sections into the Ozone build so they are reachable from Ozone webUIs even if air gapped.
    a. This is currently not an issue because the docs section is within the main ozone repo. If we change this, we will need to handle this case.
4. Update release guide to handle processes for updating the website.
5. Merge the new website feature branch to enable publishing.

Exit Criteria

- New Ozone website is feature complete with the previous Ozone homepage and docs
- Easy process for release managers to publish new versions of the site for releases.
- Updates from pull requests are automatically published at a defined frequency.

Phase 7: Developer Content

Tasks

1. Setting up dev environment.
2. Github Actions guide.
3. Maven invocations and flags specific to Ozone.
4. Guide for release managers.
5. Developing on arm machines like Apple Silicon Macs.

6. Java client API (built from javadoc).
7. Common Intellij configurations for Ozone development.
8. Deprecate/archive confluence wiki to prevent split-brain.

Exit Criteria

- New developers can write, build, and test their first change all the way to PR ready state without assistance.
- All material ported from Confluence has been re-reviewed for clarity and accuracy (not a blind copy/paste).
- Outdated material from Confluence is updated in the new site, or removed if irrelevant.
- Advanced users can build custom apps on top of the Java client API.

Phase 8: System Internals

Tasks

1. Best effort documentation of most config keys.
2. Which network protocols are used for each piece of communication (grpc, hadoop rpc, http, etc).
3. Disk layout of each component
4. Containers, replicas and their states
5. Pipelines and their states
6. Deep dive into read, write, and delete paths
7. High level OM Metadata layout and considerations when structuring data.
8. OM read and write request flow
9. How Recon syncs data
10. How Ranger syncs data
11. Background threads/tasks run within each component
12. Advanced debugging and repair tools

Exit Criteria

- Readers should understand all Ozone specific "jargon" after reading these pages.
- Developers have a better understanding of component functions to guide them when working through their code.
- Readers understand disk layout of each component when they `cd` to storage directories on their system.

After completing these phases, we should have a robust website that will enable new users and developers to work with Ozone with minimal friction. Further documentation updates and improvements should be easy for anyone to contribute with the more structured site layout. Leveraging the same pull request flow as existing Ozone changes should ensure the quality of the site matches that of the Ozone project.