Azure docs disaster recovery process

**Outcome:** The Content Team has a documented go-to process for efficiently and reliably recovering an accidentally deleted repository.

**Rationale:** Recently, the public PowerBI content repository was deleted by accident by its writer-administrator. While the repository content and repository itself were recoverable though GitHub’s standard recovery process, the “configuration” of the repository was lost and had to be manually reconstructed. Additionally, we’ve heard that the Azure SDK team accidentally deleted its public repository; this just goes to show that while the chances are small, it is possible that any repo the Content Team is responsible for could be deleted by accident. If this were to happen with the Azure-docs-pr repository, it would affect a large percentage of the Azure technical documentation set, potentially pausing publishing for days. This effort also will serve to put in writing lessons learned after the PowerBI repo deletion. This effort is focused only on recovery; issues such as clearer roles and responsibilities, management for repositories, JIT permissions for admins, and training of administrators are part of a larger and longer-term effort in the Content Operations space.

**Suggested steps:**

1. Document the configuration (everything that is not a file) of the repo: accounts and levels, apps, webhooks, etc.
2. Investigate/understand the repository template functionality in GitHub/OSPO(?) to see if we can save the Azure-docs-pr and azure-docs repos as templates and evaluate if there is a way to efficiently do this on a weekly or monthly basis. Recommend reaching out to GitHub support to explain our scenario and get their recommendations. A screenshot of a computer

   Description automatically generated
3. Create a repository from template - [Creating a repository from a template - GitHub Docs](https://docs.github.com/en/repositories/creating-and-managing-repositories/creating-a-repository-from-a-template)
4. Creating a template repository - [Creating a template repository - GitHub Docs](https://docs.github.com/en/repositories/creating-and-managing-repositories/creating-a-template-repository)
5. Research and understand the implications of the following information for recovery of our repos:   
      
   *A deleted repository can be restored within 90 days, unless the repository was part of a fork network that is not currently empty. A fork network consists of a parent repository, the repository's forks, and forks of the repository's forks. If your repository was part of a fork network, it cannot be restored unless every other repository in the network is deleted or has been detached from the network. For more information about forks, see "About forks." If you want to restore a repository that was part of a fork network that is not currently empty, you can contact us through the GitHub Support portal.* Source: [Restoring a deleted repository - GitHub Enterprise Cloud Docs](https://docs.github.com/en/enterprise-cloud@latest/repositories/creating-and-managing-repositories/restoring-a-deleted-repository#about-repository-restoration)
6. Determine the best way (documentation, template, other) to store the configuration for recovery and implementation.
7. Create a new repository for testing purposes and configure it exactly like the Azure private repository.
8. Have 2-3 people fork the repo so we can test recovery through GitHub support and confirm that forks are also restored.
9. Delete the repository.
10. Work with GitHub to restore the repository through their support site. Use their support form, do NOT use the chat control. Using their contact form means they will reply by email, so content they provide can be incorporated into the documentation if relevant.
11. Document all configuration loss in the restored repository and all challenges/difficulties/unexpected issues that occur in working with GitHub to restore the test repository.
12. Restore the configuration using the documented method and update the documentation as necessary.
13. If successful, add documentation to the Platform manual and develop best practices recommendations within that content.
14. Based on outcome, determine if a larger project is needed to create other repository backup documentation or if the experience lends itself to a tooling requirement.

Resources

* [Restoring a deleted repository - GitHub Enterprise Cloud Docs](https://docs.github.com/en/enterprise-cloud@latest/repositories/creating-and-managing-repositories/restoring-a-deleted-repository#about-repository-restoration)
* [Backing up a repository - GitHub Docs](https://docs.github.com/en/repositories/archiving-a-github-repository/backing-up-a-repository#title-h1)
* [MySQL infrastructure testing automation at GitHub - The GitHub Blog](https://github.blog/2017-07-06-mysql-testing-automation-at-github/)
* [Disaster Recovery #2465803 - GitHub Support](https://support.github.com/ticket/personal/0/2465803)