|  |
| --- |
| GitHub Governance for Product Engineering Team Organization |
|  |
| Version 1.1.0 - 11 / 14 / 2023 |

Contents

[Introduction 3](#_Toc149203434)

[Repository Management 3](#_Toc149203435)

[Roles and Responsibilities 5](#_Toc149203436)

[Security and Data Protection 5](#_Toc149203437)

[Best Practices 6](#_Toc149203438)

[Runners Execution 6](#_Toc149203439)

[Conclusion 7](#_Toc149203440)

# Introduction

This document outlines the policies and procedures that govern the use of GitHub by DA Technology Services - Product Engineering Team. Its purpose is to foster effective collaboration, ensure compliance with security policies, and efficiently manage resources on GitHub.

**Purpose**

The purpose of this governance document is to:

* Establish guidelines for source code collaboration and version control.
* Define roles and responsibilities related to GitHub usage.
* Implement security measures and data protection protocols.
* Promote best practices in software development.

# Repository Management

**Repository Creation**

Members of the Product Engineering Team are encouraged to create new repositories following a structured naming convention in kebab-case. This convention should clearly represent the project name, the main technology used, and the functional context of the repository.

Structure:

[project-name]-[main-technology]-[functional-context]

Functional Contexts Include:

* frontend: For repositories focused on frontend development.
* backend: For repositories dealing with backend services.
* api: For repositories containing API-related code.
* lib: For libraries or shared code.
* data: For repositories handling data processing or storage.
* util: For utility or tool-related code.
* test: For repositories primarily used for testing.
* config: For configuration-related code.

Examples:

* travel-app-node-backend
* ecommerce-platform-react-frontend
* payment-gateway-python-api
* shared-utilities-ts-lib

Members are strongly encouraged to create new repositories using approved boilerplates whenever feasible.

Approved boilerplates are pre-defined project templates or structures that have undergone scrutiny and received the endorsement of a dedicated committee for the sake of ensuring consistency, adherence to best practices, and compliance with governance standards.

Utilizing approved boilerplates serves to guarantee uniformity, minimize redundant efforts, and encourage adherence to established best practices.

When an approved boilerplate is unavailable for a particular project, members should request a review from the Backend or Frontend Committees to create a new repository from scratch.

Repositories must incorporate comprehensive documentation that clearly outlines their purpose, structure, and contribution guidelines.

**Repository Access**

Access to repositories should follow the "need-to-know" principle. Members should only have access to repositories relevant to their roles and responsibilities.

Only Repository Owner and administrators could delete repositories.

Permission management will be done through the GitHub organization and will be periodically reviewed for consistency.

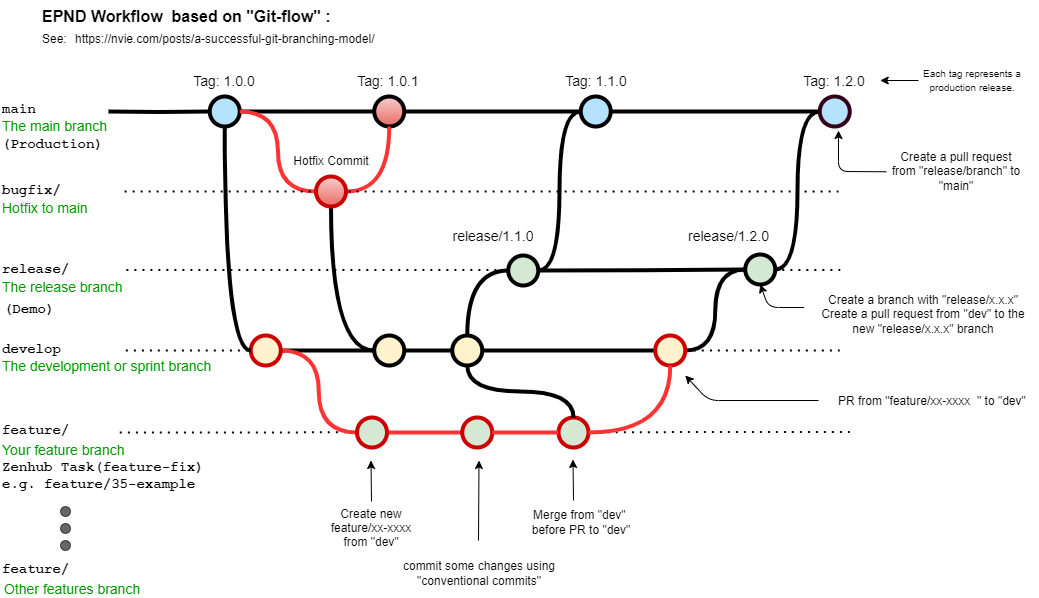
**Token Management for Repository Access**

Creation of access tokens is permitted only with fine-grained scopes selected. Broad or full access scopes are discouraged to enhance security.

All access tokens must be approved by the administration team. This requires the administration to carefully evaluate and grant only the permissions necessary for each specific token use-case.

**Branching and Workflow**

Repositories should follow a branching workflow, such as GitFlow or GitHub Flow, to maintain proper version control and facilitate collaboration.



Pull Requests must be used for code review and approval before merging changes into the dev and main branch.

# Users, Roles and Responsibilities

**New Account Naming Convention**

New GitHub accounts created by members of the Product Engineering Team should follow the naming convention: [FirstInitial][LastName]-EVS.

Example: **JDoe-EVS**

**Repository Owner**

Maintenance: Responsible for overall repository maintenance, including managing pull requests and addressing repository-related issues.

Code Validation: Ensure that developers consistently update and sync the code generated in the cloud with the repository. Encourage regular uploads to ensure that the repository remains up-to-date and reflective of ongoing work.

**Collaborators**

Team members who can make contributions to the repository but are not responsible for its overarching management.

**Organization Administrators - Owners**

Organization administrators can manage permissions and resources within the GitHub organization.

The following individuals are designated as Organization Administrators with the responsibility for repository management:

* Richard Saavedra
* Amandeep Singh
* Bruno Rodriguez

# Security and Data Protection

Repository Privacy: All repositories, especially those containing sensitive or confidential information, must be set to 'private' to ensure limited access.

Authentication: Implement Two-Factor Authentication (2FA) for all GitHub accounts.

Data Storage:

* Utilize GitHub's built-in secrets management feature for securely storing and managing sensitive data such as API keys or simple tokens.
* For complex keys, like SSL keys or certain JWT tokens, which might exceed GitHub secrets' handling capability, encode them using base64 before storing.

Code Storage: All essential keys, credentials, and variables should be saved as secrets or environment variables in GitHub repositories. Directly hardcoding or committing such data into repositories is prohibited.

Access Management: Regularly review and update access permissions to secrets. Ensure that only authorized personnel have access, and promptly revoke access for those departing the organization.

# Best Practices

Best practices in software development, such as proper documentation, unit testing, and code review, should be followed.

Developers should follow the recommendations presented in TS [Co-HUB](https://product-engineering-team.github.io/) according to the technology used.

Team members are encouraged to use, Azure DevOps(Boards), GitHub's project management and issue tracking tools.

**Review and Updates**

This governance document will be reviewed and updated periodically to ensure relevance and effectiveness. Updates will be communicated to all members of the Product Engineering Team.

# Runners Execution

Any runners or scripts executed from repositories to external servers must comply with the guidelines and policies outlined in this governance document.

Prior to executing runners or scripts, team members should ensure that they have reviewed and follow the governance document's guidelines related to repository management, security, and best practices.

# Conclusion

Effective GitHub governance is essential for maintaining a collaborative and secure development environment within our Product Engineering Team. All team members should familiarize themselves with these policies and adhere to best practices when using GitHub. Should there be any questions or need for clarification regarding the guidelines, team members are encouraged to reach out for assistance. It's crucial that everyone understands and upholds these standards to ensure our collective success.

Signature: Richard Saavedra   Date: 17 – Oct – 2023

Reviewed by:

Ignacio Henríquez

# CHANGE LOG

|  |  |  |
| --- | --- | --- |
| **Date** | **Description** | **Author** |
| November 14, 2023 | Added new account naming convention rules | Richard Saavedra |
| November 14, 2023 | Enforced institutional account collaboration only | Richard Saavedra |
| November 14, 2023 | Updated token management and approval process | Richard Saavedra |
| November 14, 2023 | Change Organization Administrators | Richard Saavedra |