ONS GitHub Usage Policy
GitHub Acceptable Use Policy

Implemented on (date)	20/05/2024
Approved by (name & role)	Head of Software Practices (Fahad Anwar) In consultation with TAG (Technical Advisory Group).
Last review on (date)	30/09/2024
Reviewed by	TAG (Technical Advisory Group)
Next review due on (date)	16/05/2025
Policy owner (name)	Head of Software Practices In consultation with Technical Advisory Group (TAG) representing Software Engineers, Cloud Division, TISS and Security Division.
Policy owner (division)	Digital Services and Technology (DST)
Main point of contact (name)	Fahad Anwar Software Engineering Head of Practice
Status	Approved

Policy Review Record

This Review Record is to be completed on each time a review is conducted. Its purpose is to maintain a record of reviews, recording who conducted the review (policy owner), the date of the review and the outcome of the review (policy fit for purpose, amendment required, policy no longer required, etc).

This Policy is to be reviewed annually.

Review No	Review Conducted By	Review Date	Review Outcome
01	TAG (Phase-2)	17/09/2024	Amendment to the GitHub Usage Policy

Amendment Details

Date	Amendment Summary	Amended by	Version
05/06/2024	Corrected typo in clause 4.3	Fahad Anwar	1.0
30/09/2024	• Updates to Section 2 • Update to Section 5 & 6 For detail reasoning please see TAG approved amendments document	Fahad Anwar	1.1

RASCI (For detail please – RASCI Information document)

Responsible	G6 Program Managers through Technical Leads, G7 and SEO's
Accountable	Head of Software Practices
Supportive	Head of Cloud Functions (Amazon, GCP, Azure) SAIM SIRA Software Engineering Community of Practice (SE-CoP)
Consulted	Technical Advisory Group (TAG) representing Software Engineers, Cloud Division, TISS and Security Division.
Informed	Senior Leadership Team Software Engineering Community

	SAIM SIRA Design Authority Chair

GitHub Acceptable Use Policy

Policy Statement

This GitHub Usage Policy outlines the best practices and guidelines for using GitHub within ONS (referred to as "the Organisation"). The use of GitHub for version control, collaboration, and project management (for public repositories) is encouraged and governed by the principles and rules set forth in this policy.

Scope

Digital Services and Technology and any external parties delivering services into DST (unless by agreement).

Responsibility

The Head of Software Engineering Practice assumes the overall responsibility for the policies on using GitHub at ONS.

Acceptable Use Policy

2. Account and Profile

2.1 Account Creation: GitHub users representing the Organisation can do so with either their personal GitHub account or one created specifically for Organisation work. An Organisation email address (e.g. @ons.gov.uk / @ext.ons.gov.uk) is required to be associated with any GitHub account undertaking Organisation work. Multiple email addresses can be associated to a single GitHub account, so in the case of personal accounts, the Organisation email address can be added as a secondary address alongside the primary personal email. This is required for accurate onboarding and offboarding.
2.2 Email privacy: GitHub email privacy must be enabled to prevent Organisation email addresses from being divulged when performing Git operations. This document explains how to do this. Note there are separate configuration locations for web-based operations and command line / Integrated Development Environment operations.
2.3 Profile Information: GitHub users must be mindful of the information they share in public GitHub profiles. It is possible to have very little information populated in the public profile and this is recommended for dedicated Organisation accounts.
2.4 Publicly available information: GitHub users should familiarise themselves with the Social Media Policy. With the ability for much information to be in the public domain on GitHub, this policy is largely appropriate.

3. Managing movers/leavers/joiners

3.1 New Joiner: For a new joiner to the GitHub ONSdigital organisation, the Technical Lead or nominated Sr. team members must raise a ServiceNow ticket. [1]

[1] **Note for Team Leads and requester:** This request is to send an invitation to join ONSdigital organisation. Users must create their own GitHub account before submitting this request to receive an invitation to join the ONSDigital GitHub organisation.

3.2 Leaver: If an individual leaves ONS then the Technical Lead or nominated Sr. team members must also remove them from the ONSdigital organisation.
3.3 Audit: GitHub audits joiners and leavers via the audit log.

4. Repository Management

4.1 Repository Creation: By default, all new repositories must be created within the ONSdigital GitHub organisation or one of ONS GitHub recognise organisations.
4.2 Repository name: Repository names should be clear, descriptive, and adhere to naming conventions of (use hyphens or underscores for readability. Avoid special characters, spaces, or uppercase letters in repository names). Repository names should be clear, descriptive, and adhere to naming conventions of (use hyphens or underscores for readability. Avoid special characters, spaces, or uppercase letters in repository names).
4.3 Private/Internal Repository Information: Repositories should be created as public by default unless there is specific need for them to be internal or private. If there is a requirement that the repository should be Private OR Internal, then the repository must include a PIRR.md (Private/Internal Repository Reasoning Record) file.
4.4 CODEOWNERS file: All repositories must include a CODEOWNERS file that contains the GitHub Team name(s) or the GitHub ID(s) of the team(s)/individual(s) responsible for that repository. The CODEOWNERS file must be located in either the root of the repository, a ".github" directory, or a "/docs" directory.
4.5 README File: All repositories should include a README file that provides an overview of the project, installation instructions, and usage guidelines. Markdown formatting is encouraged for clarity.
4.6 Licensing: A license file (e.g., LICENSE) must be included in each public repository to specify the terms of use for the project. Choose a license that aligns with the project's goals.
4.7 .gitignore: Utilise a .gitignore file to specify which files or directories should be ignored by Git to prevent unnecessary files from being committed to the repository.
4.8 Documentation: Maintain comprehensive and up-to-date documentation for projects, including usage instructions, installation guides, and troubleshooting tips.
4.9 Archiving Repositories: If an application is no longer used in production and there is no update on the repository for the last 01 year, the Technical Lead or nominated Sr. team member should archive the repository, Update the README file accordingly and resolve any open security alerts before arching the repository.

5. Collaboration and Communication

5.1 Team Names: Use clear and descriptive names for teams and organisations to help members understand their purpose.
5.2 Team Roles: Every team must have one or more members with the "maintainer" role assigned. These individuals are responsible for the repositories owned by the team and for maintaining the team membership in GitHub.
5.3 Collaboration through teams: Only teams should be granted access to repositories. This makes it clear why the access is needed and helps to ensure people have correct permissions within team moves. If there is an explicit need for an individual to be granted access this should be agreed with Technical Lead or nominated Sr. team member.
5.3.1 Main Teams structure
5.3.1.1 Member role: By default, all new members will be given non-administrative role in the Github organisation; this role has permissions such as creating repositories and project boards.
5.3.1.2 GitHub Organisational Owners Team: Complete administrative access to the GitHub organisation; limited to few members only but not less than two people in the team.
5.3.1.3 GitHub Team Leads/Sr. Software Engineers Team: "Members with limited Admin Permission", following permission attached to the team/role:

- Create Repositories, manage repository settings, including adding collaborators and setting their permissions, Create and Manage GitHub Teams
- Access to view code, pull requests, and manage issues.
- Ability to push to the repository and merge pull requests.
- Manage project without full access to sensitive or destructive actions.
- Manage issues and pull requests without write access.
- Approval/Management of GitHub App.

5.3.1.4 Security Management Team/Role: Permissions to manage security alerts and settings across the organization, as well as read permissions for all repositories in the organisation.

5.4 Code of Conduct: ONS encourages respectful and inclusive interactions on GitHub. Adopt and adhere to a code of conduct to ensure a welcoming and harassment-free environment for contributors.

5.5 Collaboration Workflow: Collaborate effectively by using pull requests for code reviews and discussions. Use issues and discussions to track and discuss project changes and enhancements.

5.6 Unsolicited pull requests: If we get an unsolicited pull request from outside of the organisation it must not be merged. It should be raised to the Technical Lead or nominated Sr. team member.

5.7 Branching Strategy: Implement a clear and effective branching strategy (e.g., feature branches, release branches) to organise and manage code changes.

5.8 Branch Protection rules: Branch Protection rules must be used to enforce security policies by preventing accidental branch deletions, enforcing code reviews, and requiring successful automated checks before pull requests can be merged etc.

5.9 Signed Commits: Use of signed commits must be used, so other people can verify that your work comes from a trusted source. If a commit or tag has a GPG or S/MIME signature that is cryptographically verifiable, GitHub marks the commit or tag as verified.

6. Security and Compliance

6.1 Privacy Settings: Review and adjust your privacy settings on GitHub to ensure that your contributions and personal data are appropriately protected.

6.2 Security: Regularly review and update dependencies to address security vulnerabilities. Utilise automated security tools and services to identify and remediate vulnerabilities.

- All public repositories must have secret scanning enabled.
- All public repositories must have push protection enabled.
- All public repositories must have dependabot enabled.
- It is strongly advised that all development team should have dependabot Auto-Pull request feature enabled on all their repositories.
- Signed commits must be used for all commits of public repositories.
- Branch protection must be enabled on all repositories.

6.2.1 Secret Scanning Alerts SLO (Service Level Objectives): 05 working days to resolve any secret scanning alerts from the time of creation.

6.2.2 Dependabot alerts SLO (Service Level Objectives): following are SLO for dependabot alerts:

- **05** working days for CRITICAL category dependabot alert from the time of creation.
- **15** working days for HIGH category dependabot alert from the time of creation.
- **60** working days for MEDIUM/MODERATE category dependabot alert from the time of creation.
- **90** working days for LOW category dependabot alert from the time of creation.

These SLO will be effective from the date of this policy publication date.

6.2.3 Implementation Note: Note all above security feature might not be enabled on day one of publishing this policy, however the aim is to have all above security features enabled within 12 months of publication of this policy.

6.2.4 *Private / Internal repositories:* Providing GitHub Advance Security features are available all above security features should be enabled on Private and Internal repositories as well.
6.3 Compliance: Ensure that all GitHub activities and repositories comply with applicable laws, regulations, and organisational policies. Be aware of any data protection or export control regulations.

7. Continuous Integration and Build

7.1 CI/CD Pipelines: Where appropriate use GitHub Actions to implement continues integration and build pipelines to automate testing, building processes.

8. Reporting Issues

8.1 Reporting Security Concerns: Report security vulnerabilities or suspicious activities immediately to the appropriate contact within the Organisation or follow the established incident reporting procedures. [2]

9. Compliance and Enforcement

9.1 Policy Compliance: Violations of this GitHub Usage Policy may result in consequences, including warnings, access revocation, or other disciplinary actions, as determined by the Organisation.
9.2 Policy Enforcement Teams: Technical Lead or Sr. member of respective software development team is responsible for monitoring and enforcing policy compliance.
9.3 Periodic Audits: Conduct periodic audits [12 month] of repositories to ensure compliance. Use automation to assist in auditing and reporting. Form an Audit Team from Software Community of Practice where members can be rotated.

10. Exception process

It is acknowledged that situations arise in which the Policy Detail as above may not be able to be met. Where this is the case a documented policy exception must be sought from the Policy Owner (specified in the table on the title page).

11. Retrospective Implementation: A separate document/plan must be created to address how this policy should be implemented retrospectively.

12. Source Control System Training (use Pluralsight channels where possible)

12.1 Mandatory Training

12.1.1 For all software engineers

- Get started using GitHub to manage Git repositories and collaborate with others.
- Creating and managing repositories

12.1.2 For Technical Leads managing GitHub repositories/projects.

- Get started using GitHub to manage Git repositories and collaborate with others.
- Creating and managing repositories
- Build security into your GitHub workflow.

12.2 Pluralsight Training Channels (including mandatory training & more)

- **All users:** Pluralsight channel (GitHub: Still Level = Working): Working with GitHub Source Code Systems.
- **Repository owners:** Pluralsight channel (GitHub: Still Level = Practitioner): Working with GitHub Source Code Systems.

[2] Excluding the dependabot alerts

13. Definitions

13.1 Glossary: A glossary of key terms used in this policy is available for reference.

- PIRR = Private/Internal Repository Reasoning Record).
- TAG = Technical Advisory Group.
- SDP = Software Developers Portal.
- SLO = Service Level Objectives.

14. Useful links Definitions

- Base GitHub Template Repository = https://github.com/ONSdigital/ons-template

15. ONS recognised GitHub Organisations

- https://github.com/best-practice-and-impact
- https://github.com/datasciencecampus
- https://github.com/IDPdigital
- https://github.com/integrateddataservice
- https://github.com/ONS-centre-for-crime-and-justice
- https://github.com/ONS-Health-Index-and-Projections
- https://github.com/ONS-SST
- https://github.com/ONSBigData
- https://github.com/ONSdigital
- https://github.com/ONSgeo
- https://github.com/ONSvisual
- https://github.com/open-sdg

Note: If you believe there is a recognised GitHub organisation not listed above, please contact policy owner.