**Best Practices Template[[1]](#footnote-1)**

This section will be aimed at developers and other technical practitioners within government and mirrors the structure of the policy to provide guidance.

**Contributing**

Comments in the contributed code can be added in the following way:

This code contributed by Jane Smith <jane.smith@government.xx>. © 2016 ENTITY\_NAME.

Refer to the contributing file of the project to have your name added to the project.

Even if the project is using the CC0 license, contributors’names should be tracked.

**Personal / Professional contributions**

Use only one account but separate your governmental / professional and personal email address. To switch, have two repository, one for professional contribution, one for personal

* In the professional repository

git config user.email <email@pro.gov>

* In the personal repository

git config user.email <email@private.xx>

Within the repository, you can check that your command was taken into account.

git config --get user.email

**Contributing to Existing Open Source Projects**

No specific technical guidance in addition to the policy. Follow the contribution rules of the existing project.

**Releasing and Managing Open Source Projects**

**Initial checks:**

A comprehensive resource to identify *registered* trademarks worldwide is [TMView](https://www.tmdn.org/tmview/welcome) (usually software is under class 9, but also 16, 38, 41 and 42 can be suggestive). Please note, in addition, that most of the time FOSS project do not register trademarks, therefore a standard search is unlikely to be exhaustive.

Also <http://fossmarks.org/> might be helpful.

**Choosing a Version Control System**

Projects should choose a version control system that is distributed and which is preferably interoperable with git. Projects may want to take a consider using one of the following project hosting systems based on their functionality and significant adoption by the open source community:

* [Github](https://github.com/)
* [Gitlab](https://gitlab.com/)
* [Bitbucket](https://bitbucket.org/)

**Issue Tracking**

Issue reporting and tracking tools should be used for the project wherever possible. Many project hosting sites offer this functionality built into them, or they can be provided by stand-alone projects such as:

* [Trac](https://trac.edgewall.org/)
* [Redmine](http://www.redmine.org/)
* [Taiga](https://taiga.io/)

**Testing**

Continuous integration systems are useful to reduce the time to find bugs, by enabling automated daily tests of the whole software base. This is especially useful for projects involving a large developer community. Many projects require passage of these test systems as a requirement before change can be accepted into their main repository. Examples of these systems are:

* [Travis CI](https://travis-ci.org/)
* [Gitlab CI](https://about.gitlab.com/gitlab-ci/)

**Versioning**

Having a consistent versioning policy is recommended. The Semantic Versioning Guidelines (<http://semver.org/>) can serve as a good example to follow when possible.

**Licenses:**

A quick, non authoritative, overview of licenses in "plain language" can be found on <https://tldrlegal.com/>

Following licenses could be used depending on use cases and how resilient to inclusion in proprietary or differently licensed software projects should be allowed:

* Non copyleft (non reciprocal): (default Apache 2.0)
* Weak copyleft (reciprocal): (MPL-2.0, LGPL-3.0, EPL-1.0)
* Strong copyleft (reciprocal): (default GPL v3.0)
* Network copyleft (remote reciprocal): (default AGPL v3.0)

**Non-software licenses**

For the part of the projects that consists of documents, wikis and non-code material, the recommendation at this time is to use the [Creative Commons Attribution 4.0 International License](http://creativecommons.org/licenses/by/4.0/) (CC BY 4.0).

**Enforcing DCO**

**Syncing git repositories**

**Recommended files in repositories.**

Ensure you adhere to the minimum content requirements for establishing README, CONTRIBUTING and LICENSE files.

* README: description of project
* CONTRIBUTING: contributing guidelines, how to get involved, and identification of license or agreement under which contributions are accepted
* LICENSE: full text of outbound license of project
* MAINTAINERS: list of project maintainers, usually have formal voting privileges
* ROADMAP: public roadmap
* CONDUCT: a non-binding code of conduct for individuals interacting with one another in the context of the repository. Principles: inclusion, respect... Some examples of can be found <https://www.djangoproject.com/conduct/> and <https://github.com/18F/code-of-conduct>. This should include instructions on requesting change in the conduct document
* GOVERNANCE: describes how the project is governed. See the Governance section of this document for more details of what should be contained here
* NFR: Non-functional requirements documenting stable technical decisions around the project

These files are typically plain text files, containing no or minimal (e.g. Markdown) markup. It is strongly discouraged to use "binary" document formats (e.g. PDF).

At the very least, the README should contain a short description of the project and contact information (name and email-address and/or website) of the public administration in charge of the project.

**Source file headers**

Their format may be defined in the license. If not, they should contain in this order:

1- Status of the software (initial or derived software)

2- Name of the authors

3- Name of the owner (employer for a civil servant)

4- Copyright Notice and dates

5- License name

6- Production period

7- Version, if tracked

Examples:

Original software <AUTHORS>, Copyright © 2013-2015 ENTITY\_NAME SPDX-License-Identifier: GPL-3.0 v1.0

or

Derived software <AUTHORS> Copyright © 2013-2015 ENTITY\_NAME SPDX-License-Identifier: GPL-2.0 Based on <Name of the pre-existing software> <AUTHORS> Copyright © 2001-2003 ENTITY\_NAME SPDX-License-Identifier: GPL-2.0

To specify the license of the file and project in an easily parsable format, please use the [SPDX License List short identifiers](https://spdx.org/licenses/)so that others can easily identify the license. For more information on this format, please see [SPDX specification, Appendix V: Using SPDX short identifiers in Source Files] (<https://spdx.org/spdx-specification-21-web-version#h.twlc0ztnng3b>).

**Validation and Quality Management**

Established documentation and coding guidelines must be followed. These guidelines are often language and/or platform specific. Developers are encouraged to follow coding best practices such as: R. Kannan, "Coding best practices." <http://www.scribd.com/doc/8526130/Coding-Best-Practices>.

Using logging tools (e.g. log4j for Java) is recommended to allow configurable levels of logs (e.g. TRACE, DEBUG, WARNING, ERROR …).

It is also recommended to perform some basic quality and security checks using freely available tools: many modern IDEs already offer tools for checking consistency of source code documentation, unused code, uninitialized variables etc.

In addition, some vendors of commercial audit tools also offer free (basic), security and quality audits for open source projects (see also <https://www.owasp.org/index.php/Source_Code_Analysis_Tools>).

Although example configuration files should be provided, make sure they do not contain sensitive information (i.e. valid usernames and passwords, private PKI keys or other credentials for accessing production servers).

All projects should participate in the CII Best Practices badging effort to promote best practice adoption in the project: <https://bestpractices.coreinfrastructure.org/> (see criteria for more detailed information: <https://github.com/linuxfoundation/cii-best-practices-badge/blob/master/doc/criteria.md>)

Projects should consider code scanning tools such as:

* <https://codeclimate.com/>
* <http://www.sonarqube.org/>
* <https://www.fossology.org/>
* <http://www.binaryanalysis.org/>

**Community management**

*Due to the nature of open development model, people outside ENTITY\_NAME may want to submit bug reports and/or enhancements. To encourage the participation by other public administrations or other external contributors be prepared to answer general support questions and make sure there is a clear communication about the minimum requirements to accept contributions.*

1. This best practices template originates from here: https://github.com/DISIC/foss-contrib-policy-template/blob/master/BestpracticesTemplate.md [↑](#footnote-ref-1)