Platform Handbook

Deliverable Number	D2.2
Lead Beneficiary	2
Nature/Dissemination Level	PU (Public)
Working Group/Task	WP2, T2.2
Editor	Davide Galletti (GKNT)
List of Authors	Rui Ferreira(IT), Alfredo Matos(CMS), Andreia Palma (CMS), Roberto Galoppini (GKNT), Davide Galletti (GKNT)
Date	05/09/13

Executive Summary

The main goal of Work Package 2 is to provide and operate an online platform to host Open Source artefacts of EU projects. Based on the results of a public survey, the Allura Platform has been chosen as the starting point for the PROSE Platform. This document describes the PROSE platform, a heavily customised version of Allura in terms of layout, configuration and functionalities to respond better to PROSE's requirements. The present deliverable details all platform features, providing developers and administrators with an operational manual. It also includes the platform deployment details, features, and future roadmap, outlining the PROSE strategy concerning the platform. This work will be the starting point for the training material provided by Work Package 3, and for key dissemination activities carried out in Work Package 4.

The Platform is accessible at the following address: *http://opensourceprojects.eu/*

Table of Contents

List of Figures

Acronyms

Acronym	Meaning
API	Application Programming Interface
FLOSS	Free/Libre and Open Source Software
IaaS	Infrastructure-as-a-Service
ICT	Information and Communications Technology
PROSE	Promoting Open Source in European Projects
QA	Quality Assurance
SCM	Source Code Management
URL	Uniform Resource Locator
UI	User Interface
WP	Work Package

1. Introduction

One of the main strategies for PROSE to promote FLOSS within the ICT community is defined as providing a platform where projects can develop FLOSS as well as establish a community. In this process, the platform plays a decisive role, both as an internal tool for projects to organise their development, and as an external tool, enabling community engagement on the software being produced as part of the research results executed in projects.

In deliverable D2.1. [3] we introduced the selected platform, based on the requirements gathered through a public consultation process using a survey [4] targeting the ICT community. In response to the conclusions of the PROSE survey the Allura platform [1] was selected as the basis for the platform supporting FLOSS development. The main reasons behind this decision are the support for the most common tools for code management (GIT and SVN), support for private projects and sub-projects, and the possibility to run a standalone deployment, under the full control of the project (to enable the platform customisation and to support an integration plan with potential results stemming from other ICT projects).

Following up on the decision to deploy a self-hosted Allura platform, **opensourceprojects.eu** was created, the PROSE main platform, making it necessary to provide a thorough description of the platform that is currently reaching the first deployment milestone and public release. The main intent is to provide a comprehensive guide of the most important features on the platform. These include projects, code repositories, tickets and communication tools (mainly Wiki and discussion forums). As a self-hosted solution, it is important to provide information on the actual deployment model so that the platform's reliability is correctly assessed, focusing specially on availability and support. As part of the available deployment we also provide a short overview of the platform's API that can be provided to other ICT projects, opening the door to a tight relationship with projects that are developing code and community assessment tools. To complement the technical information regarding features and deployment, it is necessary to document how to actually use the platform, focusing particularly on the most common work-flows that will be available on the first release. They deal directly with creating projects, repositories and managing them through tickets and community interaction. This content will also serve as the initial input towards WP3 in order to facilitate the creation of training programs and documentation that can assist in educating the ICT community on the PROSE platform.

In this document we provide an overview of the platform's main features and the deployment considerations for the entire system (Sec. 2.), followed by a brief guide of how to use the platform (Sec. 3.). In Sec. 4., we provide a technical roadmap for the platform deployment and future iterations in order to allow a strategic outlining of potential collaborations, and also for platform dissemination purposes. The last section (Sec. 5.) provides the closing remarks on the overall platform deployment, as well as the main issues encountered throughout the process and the next steps towards completing the upcoming platform milestones.

2. PROSE Platform

The PROSE platform, registered under **opensourceproject.eu**, will host ICT Project software, and future open source projects in the EC spectrum. As already introduced in D2.1 it is based on Allura [1], the same platform used by SourceForge[1]. In this section we highlight the main features of the PROSE platform, which adds on top of the existing Allura code base to provide a structured User Interface (UI) targeting ICT projects and, more importantly provides access to private projects and the entire feature set highlighted in D2.1 through a standalone deployment as opposed to direct utilisation of a SourceForge neighbourhood.

2.1. Features

Allura provides a project management platform that allows users to create projects and associate various tools to them. The main distinguishing features of the platform are the high degree of flexibility (e.g. sub-projects inside projects) and the increased access control provided to project administrators (through user roles and access control lists). These features led to its selection in D2.1 [3].

To understand the operations within the platform it is first necessary to realise the natural structure it enforces. Content managed falls into one of three organisation structures: Neighbourhoods, Projects or Tools.

Neighbourhoods are primarily a management structure that refers to a large group of projects, to which the administrators of the platform attach a set of policies and where statistics are gathered. The PROSE platform holds a single neighbourhood for all its projects, so the policies for all hosted projects are identical.

Inside a neighbourhood, users can create projects and for each project they can define its associated tools, user groups and associate access policies. For example, a project can be private, which means that it is only accessible by its members. Nevertheless, complex configurations are possible, so the user can create policies for custom groups, or even create private sub-projects inside a public project.

To enable functionality within each project, its administrator enables various tools for the project. The following is a list of all the tools available to all new projects:

- Git source code repository
- Subversion source code repository
- Issue tracker
- Wiki
- Discussion Forums
- Blog

1 http://sourceforge.net

Each project can hold multiple tools or even multiple instances of the same tool. For example, one project can have multiple source code repositories and multiple issue trackers simultaneously. Some of the tools complement the access control features for each project by offering more fine grained access control policies (Sec. 3.2.7.).

From the PROSE perspective there are a number of features in the platform that are directly aligned with its objectives, in particular those that deal with gathering statistical data regarding the hosted projects:

1. The platform itself provides platform statistics as a part of its administration interface.
2. External tools can also be integrated with the platform to track its usage. For this purpose Google Analytics is being reused to track platform usage, similarly to what was done with the PROSE website.

The later feature also allows for individual projects to enable per project tracking, using Google Analytics, making this information also available to project administrators.

2.2. Platform deployment

For any platform there are several operational considerations that must be addressed upon its deployment. While the PROSE hosting platform is based on an open source solution (Allura), it is important to understand the characteristics of such a deployment in order to enable the means of continuity for the platform where other parties themselves may wish to host a similar deployment.

2.2.1. Hosting Environment

Hosting for the **opensourceprojects.eu** platform is provided by IT Aveiro within its data centre. The platform is hosted as a virtualised machine running the Allura platform on top of the OpenStack IaaS stack. The benefits of running it over a virtual machine are twofold: .

Internally the deployment of the platform is mainly built around four components: a database that holds the information stored in the platform (Wiki, Tickets, etc.); an LDAP user directory for user account information; an instance of the search and indexing server Solr[2]; and finally an instance of the Allura platform along with source code repository hosting.

Presently the operating system running on this instance is Ubuntu 12.04 Long Term Support, the database is MongoDB 2.0.4, Solr v1.4.1, and the directory server is OpenLDAP 2.4.28.

At the moment these components are collocated, but they could be deployed separately. If it becomes required to scale up the platform the first step to take would be to relocate these components into separate instances. This could be realised with standard scaling techniques available in both MongoDB and OpenLDAP.

The **opensourceprojects.eu** web platform is supported by periodic backups that encompass all aspects of the platform (databases, repository hosting). Backups of the platform are done

2 http://lucene.apache.org/solr/

in two duty cycles:

1. Every four hours the base virtualised environment is backed up. This form of live backup is faster but more expensive (storage) – these fast backups are kept for the purpose of fast recovery in case of failure.
2. Daily incremental backups for the platform itself (database and repositories). These backups are less expensive (storage) but the backup/recovery process takes longer. The main advantage concerns long term storage.

Additionally, and in order to reduce the risk of hardware failure, the storage for the platform is itself redundant.

Currently the storage space allocated to the platform is 200 Gigabytes split between the base operating system and the platform storage itself. However, the latter is based on a virtual storage volume, and additional space can be easily allocated as the platform storage needs become more demanding. The full provision for the entire virtualisation slice totals one Terabyte.

2.2.2. Relation between Allura the PROSE platform

The PROSE platform is an instance of the open source software Allura, and derives the majority of its features from the Allura code base. There are clear benefits in keeping up with the latest versions of Allura for the following reasons:

- To keep up with security updates and other software fixes
- To benefit on new features and improvements upon the current feature set
- To enable integration of existing and future tools, specifically aligned with the goals of PROSE (e.g. gathering of software metrics).

For this to be feasible it becomes necessary to identify the main differences between our deployment and the Allura default configuration. Throughout the construction of the PROSE deployment a number of shortcomings or challenges had to be addressed; that led to the introduction of various types of extensions and customisations. From an operational perspective it is relevant to detail the three most prominent differences between the PROSE deployment and the default Allura settings:

1. opensourceprojects.eu uses a custom theme for the platform, this introduces some significant interface and interaction differences that lead to minor usability changes
2. The default authentication provider in Allura is based on MongoDB, whereas in the PROSE instance the authentication provider being used is an LDAP provider
3. The default repository hosting provided by Allura for SVN, Git and Mercurial (Hg) uses the SSH protocol along with a custom filesystem access control handler. In our deployment Git and SVN are currently provided using the HTTP(S) protocol.

Allura allows instances to extend the look and feel, a capability that was explored during the

customisation of the platform theme. Throughout the deployment a custom frontend theme was defined for our deployment. This addressed some of the issues found in Allura, and more importantly presented an aesthetic that would closely relate to the goals of the platform.

Naturally such changes tend to alter the behaviour of the platform, but so far no significant usability deviations from the original Allura were introduced. Should any arise they will have to be properly addressed in the platform documentation.

Two authentication providers are available within Allura: one storing user credentials in the regular MongoDB database (used by default), and the other storing the credentials in an LDAP directory (and all ancillary data in the database). For our deployment, we choose the second authentication provider, mainly because it maximises the feature set of the platform:

- The LDAP authentication provides some additional features (e.g support for SSH keys), as a result we proactively enabled LDAP as the default provider, in case we want to enable such features at a later time.
- Using an LDAP directory service for user credentials allows us to later separate credential storage from the rest of the deployment.

Finally, our instance deploys Git and SVN using HTTP and doesn't provide SSH yet. For most other software forges HTTP is quickly becoming a common denominator since SSH might not be available (due to network firewall blocking), and might require some additional work in order to integrate the regular access to Git/SVN with the Allura access control mechanisms.

Whenever possible we intend to contribute any significant changes back to the Allura project. All the mentioned changes to the platform are currently hosted within the platform itself, in a dedicated project that also hosts documentation and support tools made available to its users[3].

2.2.3. Staging updates to the platform

Besides the resources committed to the live instance of the platform running at **opensourceprojects.eu**, there are some additional resources that deal with testing and staging upgrades to the platform.

Currently there is a single machine dedicated to platform staging and testing, but more can be added if it's needed. These machines run on the same virtualisation service as the production version of the platform, but they are significantly more constrained in terms of resources and they cannot leverage the backup features made available on the production environment.

The fact that both instances run over a common virtualisation stack enables us to simulate platform staging efforts, such as upgrading its components (e.g. Allura itself), or enabling a new tool or a feature for testing. Allura itself as well as some of its dependencies provide automated tests migration tools for this purpose.

This in turn enables a two step deployment process where changes are first committed and

3 http://opensourceprojects.eu/p/osp/

tested in one or more staging servers before they are actually deployed in the production version of the platform.

2.3. Configuration and maintenance

In this section we briefly describe the configuration of the whole platform. Tool specific configuration are adequately detailed, in order to understand the definitions for the initial deployment. We also briefly describe a project hosted in the platform used to keep track of maintenance of the platform itself, as means of continuously improving the platform and providing a direct channel to the community.

2.3.1. Platform configuration

The site administration page[4] gives access to some site-wide statistics and functionalities.

The main page shows simple statistics on projects and users.

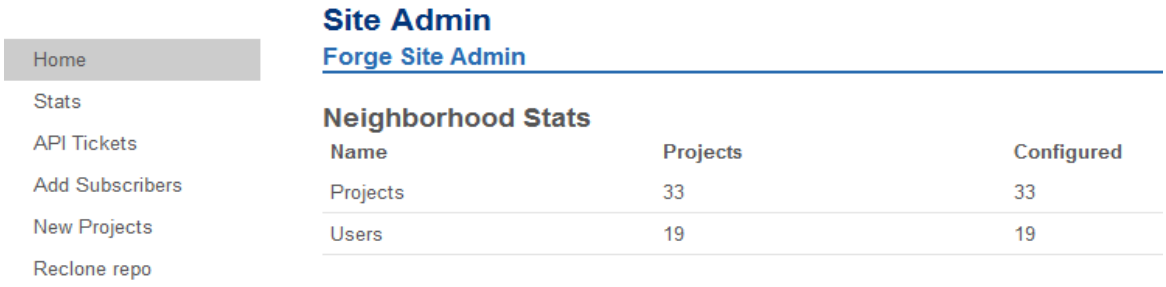


Figure 1: Site admin page with simple stats and links to sub-pages

The **API Tickets** page allows to generate special API key pairs that can be used to import data third party trackers. The full process is documented in detail on the “Third Party Data Migration” [2] document provided by the Allura project.

With the **Add Subscribers** page the administrator can subscribe a user to notification emails for an artefact (ticket, wiki page, etc.).

The **New Projects** is a monitoring page which lists all projects in order of registration starting from the most recent. It can be used to keep spam under control.

2.3.2. Maintenance project

Adapting the Allura platform to PROSE's needs has meant making quite a few customisations to it. The customisation has influenced both layout and functionalities. Furthermore there is an ongoing effort to keep **opensourceprojects.eu** platform aligned with Allura's updates.

4 http://opensourceprojects.eu/nf/admin/

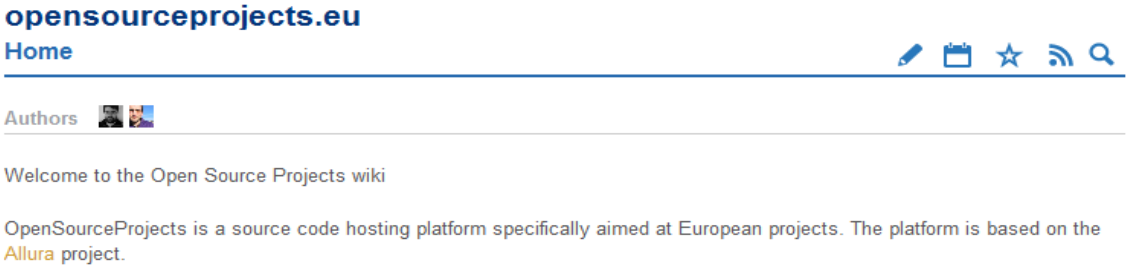


Figure 1: The maintenance project wiki page

The project has also an active ticket tracker which records both customisation activities and bug fixes.

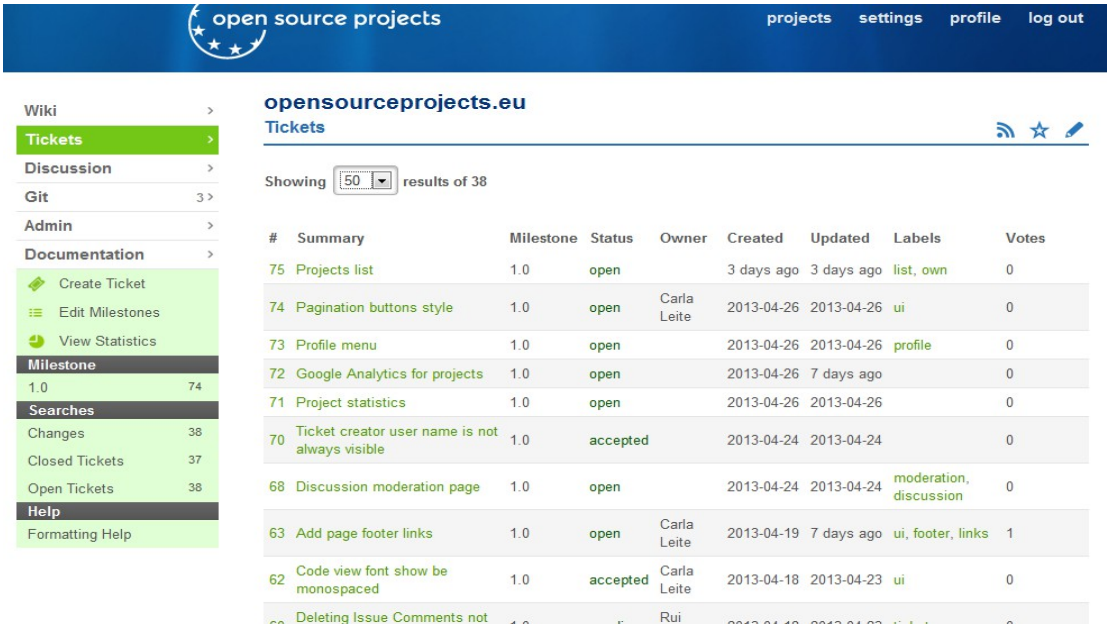


Figure 2: The Ticket tracker of the maintenance project

2.4. External API

Besides accessing the platform through its web interface, several functionalities are also accessible also via a REST API. This can be used to by external tools to query the platform to retrieve data, such as mobile applications or automated testing systems.

Without authentication, all API requests have the permissions of an anonymous visitor. To view or change anything that requires a login, you must authenticate to the API using OAuth.

To create an OAuth token you must login to the platform and visit the page

http://opensourceprojects.eu/auth/oauth/

where you can register one or more OAuth applications and see their consumer keys and consumer secrets. Registering an application you authorize it to act on your behalf. A Python example of this procedure can be found here:

https://sourceforge.net/p/forge/documentation/Allura%20API/ .

All of the API endpoints are prefixed with */rest/* and the path to the project and tool. In order to access a wiki installed in the 'test' project, for instance, the endpoint would be */rest/p/test/wiki* . The following is list of API endpoints both for project information as well as for each of the tools in the platform.

Project

Endpoint: /rest/p/<*projectname*>/

- GET: - returns a list of tools installed on the project. name specifies the type of tool, label is the label specified by the project admin, and mount_point specifies the URL component of the tool. It can be used for constructing further API queries to specific tools, as follows.

Wiki

Endpoint: /rest/p/<projectname>/<wikiname>/

- GET */rest/p/<projectname>/<wikiname>/* - returns a list of page titles
- GET */rest/p/<projectname>/<wikiname>/<title>* - returns a JSON representation of page <title>
- POST */rest/p/<projectname>/<wikiname>/title* - creates or updates the titled page
    - parameter text: page text
    - parameter labels: comma-separated list of page labels

Tickets

Endpoint: /rest/p/<projectname>/<trackername>

- GET */rest/p/<projectname>/<trackername>/* - returns a list of ticket numbers
- GET */rest/p/<projectname>/<trackername>/number* - returns a JSON representation of a ticket
- POST */rest/p/<projectname>/<trackername>*/new - creates a new ticket
    - parameter ticket_form.summary - ticket title
    - parameter ticket_form.description - ticket description
    - parameter ticket_form.status - ticket status
    - parameter ticket_form.assigned_to - username of ticket assignee
    - parameter ticket_form.labels - comma-separated list of ticket labels
    - parameter ticket_form.attachment - (optional) attachment
    - parameter ticket_form.*custom field name* - custom field value
- POST */rest/p/<projectname>/<trackername>/number*/save - updates an existing ticket

- parameters are the same as *prefix*/new

Discussion

Endpoint: */rest/p/<projectname>/_discuss/*

- GET */rest/p/<projectname>*/_discuss/ - returns summary information about the tool discussion, including the threads in the discussion (there is one thread per ticket)
- GET */rest/p/<projectname>*/_discuss/thread/*thread id*/ - returns summary information about a thread, including the posts in a thread
- GET */rest/p/<projectname>*/_discuss/thread/*thread id*/*post slug*/ - returns detailed information about a post
- POST */rest/p/<projectname>*/_discuss/thread/*thread id*/new - create a post in the given thread
    - parameter text - the text of the post
- POST */rest/p/<projectname>*/_discuss/thread/*thread id*/*post slug*/reply - create a threaded reply to the given post
    - parameter text - the text of the reply

3. Working with the platform

The previously highlighted features help accomplish a platform that suits the development needs of ICT projects, and will assist in the open sourcing of several projects. However, as many of the projects that are intended to work with the platform may not be familiar with the actual process of using a forge in their development work-flow, or may not be familiar with this particular platform, it is necessary to provide a step by step guide for the end-user, in order to simplify the platform adoption process.

While the PROSE platform has many interesting features, we focus on the ones that provide the first impact on the platform. Specially focusing on the initial user and project setup. This will help to quickly get users engaged in the development process.

This means, that users should be able to sign up and update their profile, manage projects and sub-projects, as well as associated code, issues and communication tools.

We provide an overview of these key elements in the platform, namely focusing on five key issues, that are crucial for getting the most utility out of the platform:

- **User Management**: How to manage user related information.
- **Project Management**: How to configure projects, sub-projects and the necessary tools
- **Code Management**: How to place source code on the project, under version control.
- **Issue Management**: How to manage the issues (or tickets) associated with a project
- **Communication Tools**: How to use the available communication tools.

These tools are the most important to focus on, since they allow developers and project managers to setup an account and start collaboratively developing software, and incorporating the tools into their daily development process.

In each of the highlighted steps we first include the navigation flow, in order to situate the user, and the corresponding steps and options that are important to achieve the outlined goal. All of the information is based on the current snapshot of the platform, available at **http://opensourceprojects.eu**.

3.1. User management

User Management is the section of platform exclusive for users to manage their account settings and personal information. It allows users to change their account and profile settings and their personal data. This chapter of the deliverable aims to help users get comfortable with the User Management aspects of the platform.

3.1.1. User Settings

Login > Settings: Change the user's personal details and settings.

On this page, the user is able to set or change several personal details, like the gender, birth date, location and time zone. A part on this page is reserved for user contact details, where the user may define personal and social contacts, such as email address(es) and phone number or Skype, Facebook, Google+, LinkedIn and Twitter. The user is also allowed to set a website if the user holds one.

Concerning the account details, the user is only allowed to change the displayed name and password and set or change the primary email address associated to the account.

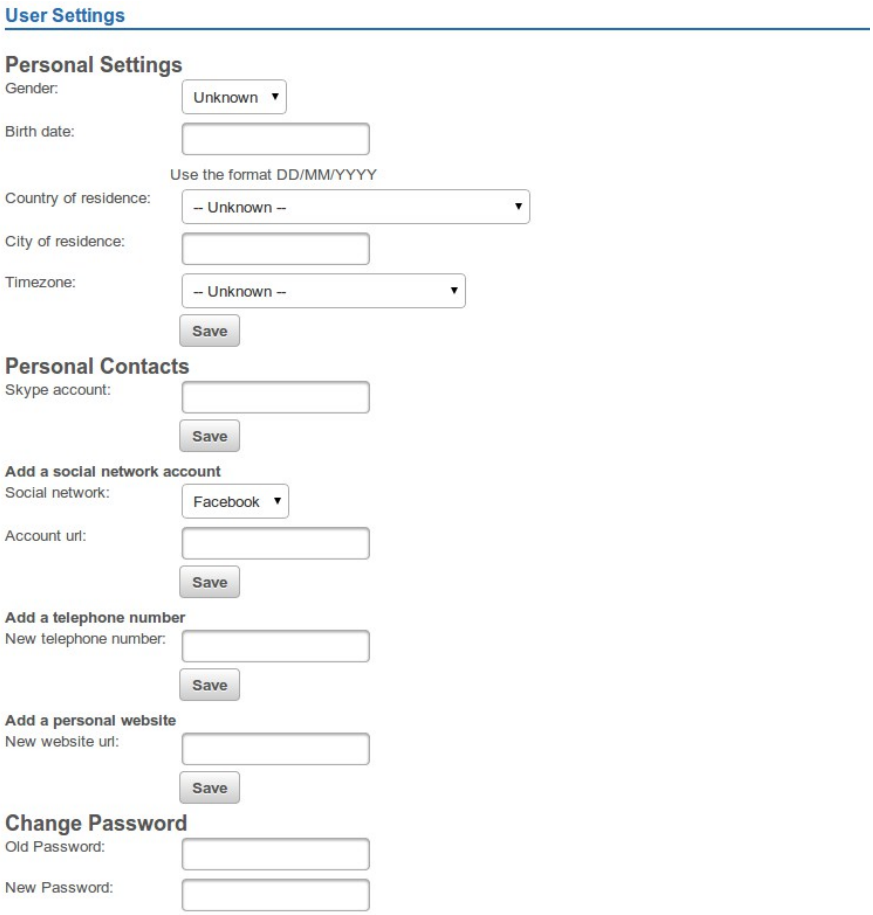
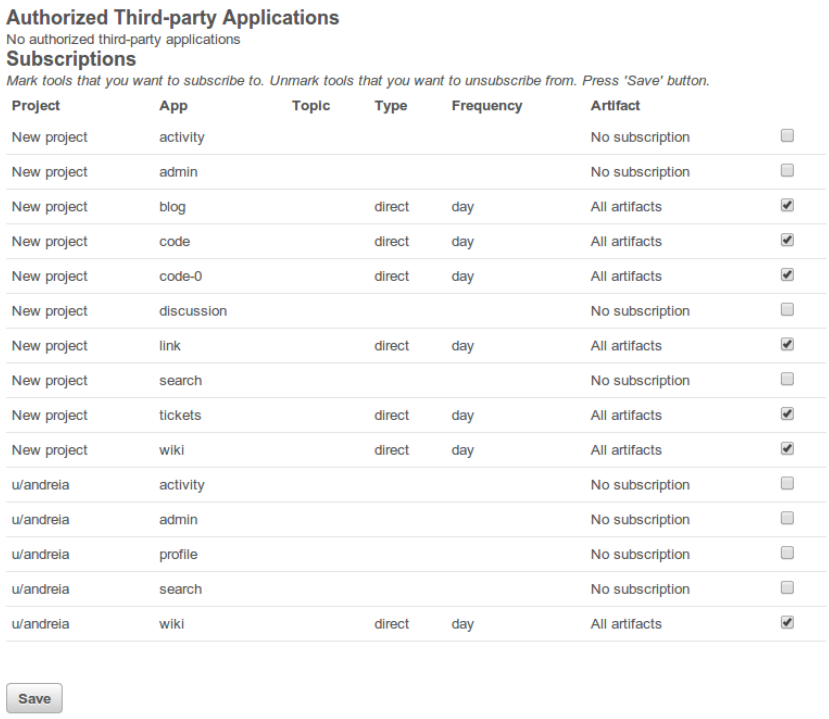


Figure 3: Part of User Settings form

Another feature of the user settings page is to offer the user a single-page management for the subscriptions.

Login > Settings > Subscriptions : Change the user's notification subscriptions.

Subscriptions are the items for which the user will receive an email notification whenever there is updated information on the platform. The user is automatically subscribed only to user's own wiki by default, but can follow other applications such as activity, admin, profile, search and others' wiki. We can see an example in Figure 4. In this screen the user can select each subscription item by pressing the corresponding check box and hitting the save button.



Authorized Third-party Applications

No authorized third-party applications

Subscriptions

Mark tools that you want to subscribe to. Unmark tools that you want to unsubscribe from. Press 'Save' button.

Project	App	Topic	Type	Frequency	Artifact	
New project	activity				No subscription	☐
New project	admin				No subscription	☐
New project	blog		direct	day	All artifacts	☑
New project	code		direct	day	All artifacts	☑
New project	code-0		direct	day	All artifacts	☑
New project	discussion				No subscription	☐
New project	link		direct	day	All artifacts	☑
New project	search				No subscription	☐
New project	tickets		direct	day	All artifacts	☑
New project	wiki		direct	day	All artifacts	☑
u/andreia	activity				No subscription	☐
u/andreia	admin				No subscription	☐
u/andreia	profile				No subscription	☐
u/andreia	search				No subscription	☐
u/andreia	wiki		direct	day	All artifacts	☑

Save

Figure 4: Subscriptions

3.1.2. User Profile

The user profile deals with all of the user related information, and explains how it is handled. The user profile is deployed as a project where the user is the administrator, and can be configured like any other project in the platform. We discuss the default tools available to the user profile which are: Profile, Wiki and Admin. These tools allow the user to manage their information and the visibility of that information.

3.1.2.1. Profile

On the Profile there is a list of the user's projects and wikis, user's personal data (such as genre or location) and user's email addresses. This information is reached by going to the

Profile (top right corner of the screen) option on the main navigation bar after Login.

Login > Profile: User details (e.g. location, email...)

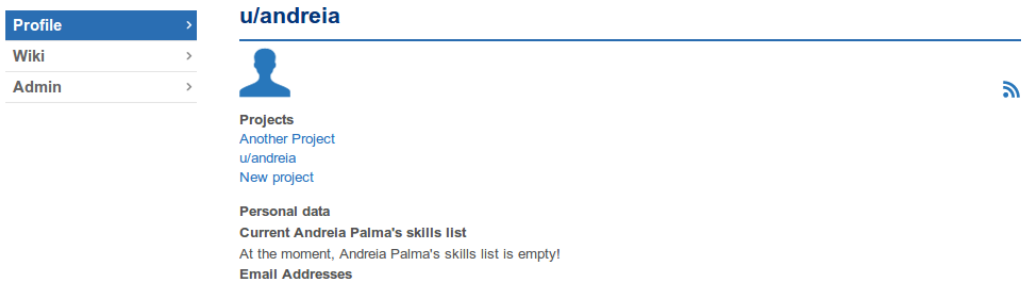


Figure 5: User Profile form

The information displayed in this page is the same personal information as defined in the user settings page (described in 3.1.1.), compounded with the user's current projects, as highlighted in Figure 5.

3.1.2.2. Wiki

There is another feature on the User Profile section of the platform, which allows the user to create one or more wikis with multiple pages.

Login > Profile > Wiki: User's wiki

On the wiki the user can store documentation about the user's profile or any information that can be shared, either public or private, depending on the defined permissions. The Wiki can also be used to store tutorials, or any other miscellaneous information created by the user or other information relevant to a specific set of people, or even private to the author.

This feature provides several subsections, like:

- **Create Page**: Where the user may create a new page for the selected wiki;
- **Wiki Home**: Which takes the user to the current wiki's home page;
- **Browse Pages**: Containing a list of all existing pages for the wiki;
- **Browse Labels**: Displays a list of all used labels within all the wiki pages;

- **Formatting Help**: A formatting tutorial to help the user getting familiar to the website's formatting options.

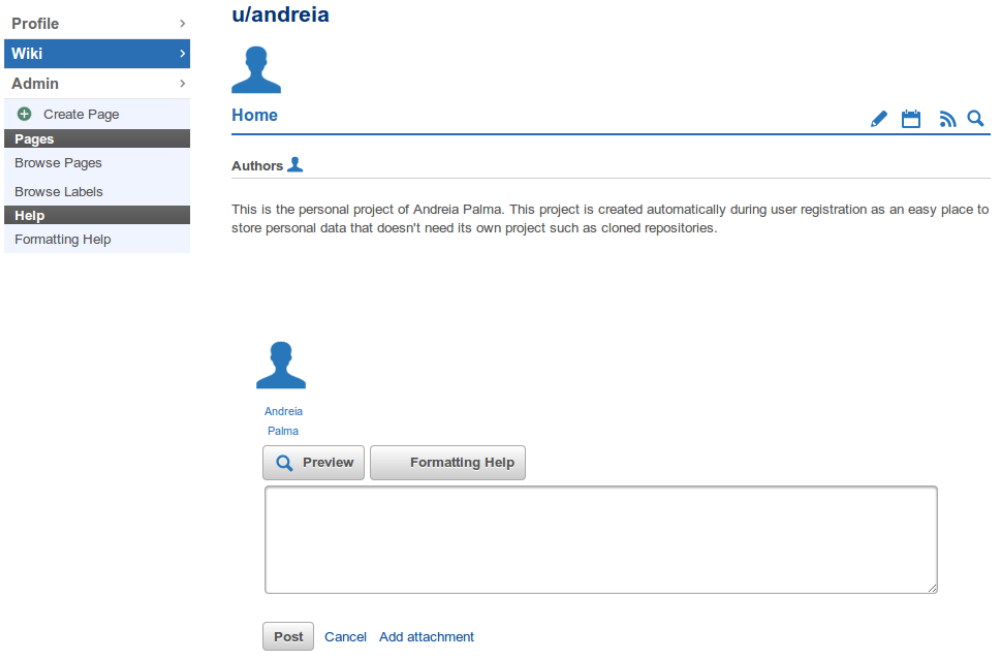


Figure 7: Profile Wiki form

After selecting a specific wiki page, the user may edit it by clicking on the edit button. This takes the user to a page editing form. Here the user may change page's title, content, labels and attachments.

This feature is also available for Projects and the information will be similar in nature.

3.1.2.3. Admin

The Admin tool is also available under the user's profile, as shown below.

Login > Profile > Admin: Manage user account

This platform's section provides access to a set of sub-features, such as user's metadata, tools, third-party users' permissions and audit trail, and is managed in a similar way to any available project. The interfaces are similar as seen on Figure 8.

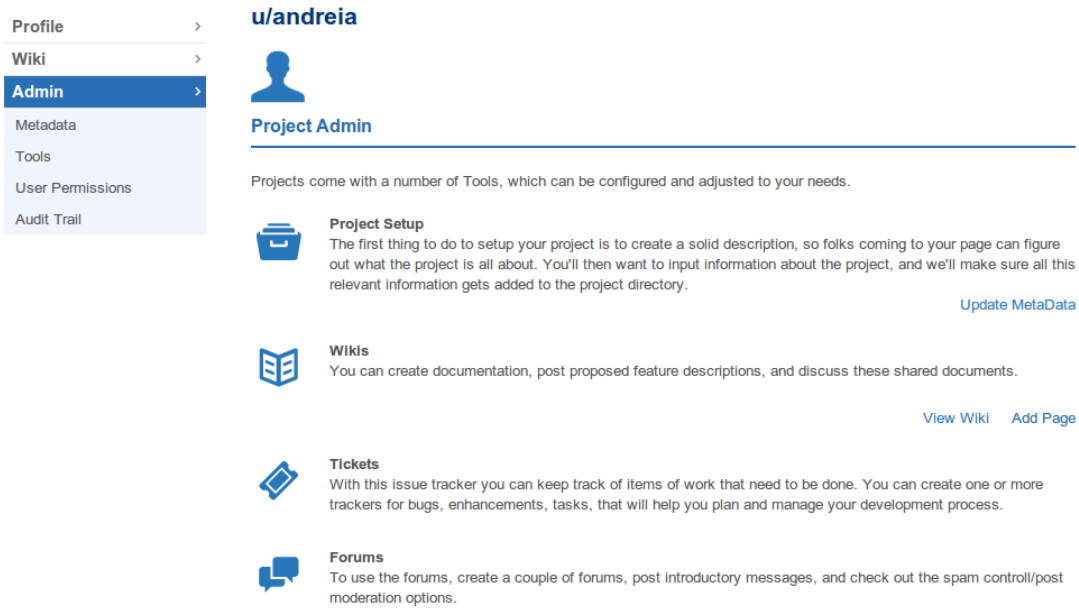


Figure 8: Profile Admin form

Out of the available options, it is important to highlight the main alternatives:

- **Metadata**: Allows the user to change User Profile page title and icon.
- **Tools**: For the user to add any available tool (external links, code repositories, blog, etc.) to their own account.
- **User Permissions**: Where the user may set new permissions of third-party users.
- **Audit Trail**: Displays a list of account management activity.

3.2. Project management

Users can manage and monitor all of their projects from the Project Management panel. This section highlights the features available to users from the perspective of what they can do within the Project Management features available on the platform. Some of these management functionalities include the ability to create projects, manage tools, permissions, information (metadata) and categorization, and optionally Wiki pages, blog posts and screenshots to each project as a way of interacting with the project's community.

3.2.1. Creating a project

The most important part of the management tools is the actual project creation process.

Login > Projects > + Add a Project: Creates new project

To create a project it is necessary that a user first logins to his account. Once logged in, the user should do the following:

- Navigate to the projects page (Login > Projects), and select “+ Add a Project”
- Type in the new Project Name and "URL Name" (3-15 characters long, using only letters, numbers, and dashes). This name will determine the project's base URL[5].
- Select the services that will be initially enabled, as presented in Figure 9. You can add or delete services at any time, as mentioned below.

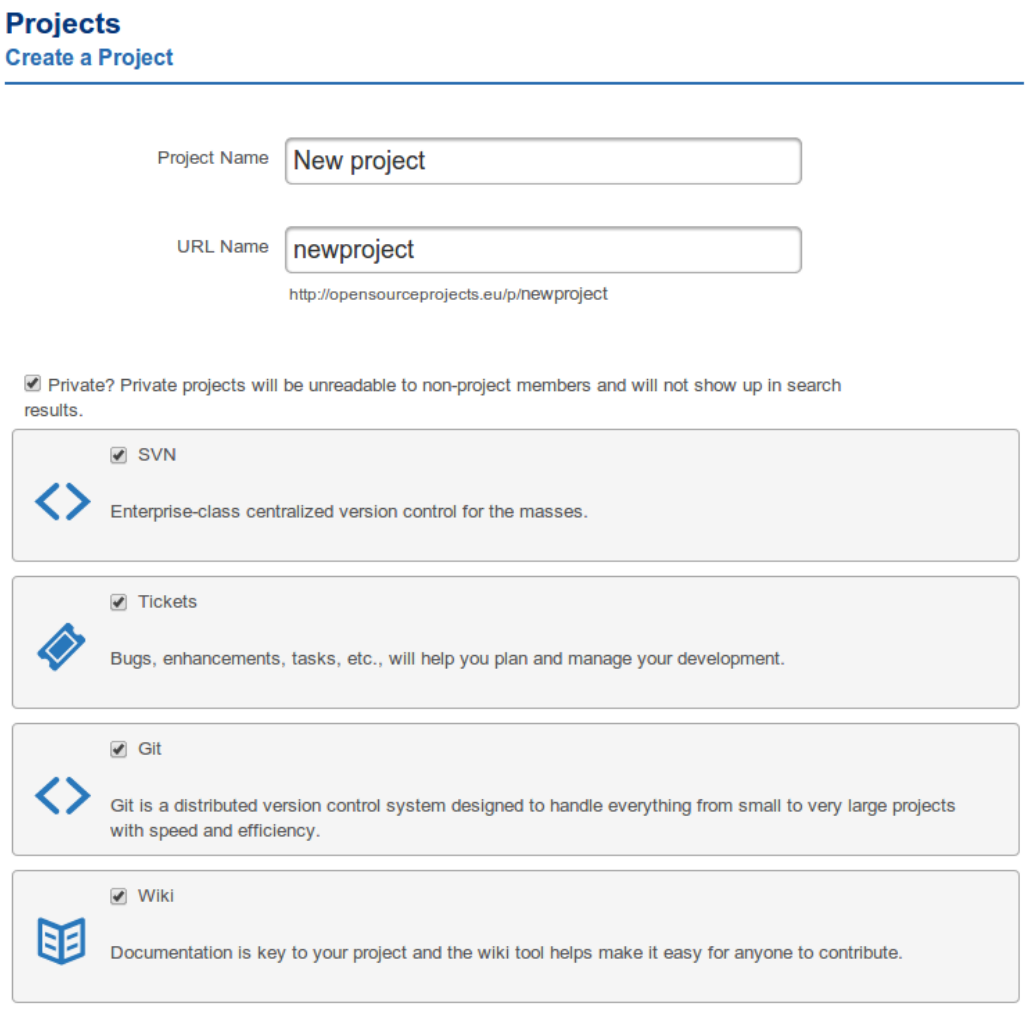


Figure 9: Creating new project

5 Take care when selecting your URL Name. Unlike other settings, this is not easily changed once you've registered your project.

To make a private project the user can select the private option as shown in Figure 9. This can also be changed at a later time, as discussed in the Permission definitions, explained in 3.2.7.

3.2.2. Metadata (Icons, Description, Screenshots)

A project's metadata is the information that characterises the project, and the available references to the project. This configuration is available as part of the project's administration pages.

> **Login > Projects >** *ProjectName* **> Admin > Metadata:** Manage information about the project

When you create a new project, you automatically are given administrator rights for that particular project. After creating a new project, the user will be brought to the project's "Metadata" page, which may be accessed at any time. Figure 9 shows an example of the configurations available to the end user. The more important settings are the following:

- **Name**: This is the publicly viewable name of the project, and will appear on project listings. It should be the full name of your project.
- **Unixname**: This is the project name as it appears in the project URL. This value cannot be changed after project creation. .
- **Home Page**: This is a URL to your project's Homepage if you have a website for your project.
- **Short Summary**: This is a text-only project summary which will be included on search results and project listings. Typically this is one or two sentences describing the project, and is text-only.
- **Full Description**: This is a longer text-only description. Use this to describe your project in greater depth than just the short summary. This is shown in search results in the expanded detail view.
- **Twitter Handle:** If you have a Twitter account for your project, you can define that here.
- **Facebook page:** If you have a Facebook page for your project, you can define that here.
- **Analytics Tracking ID:** If enabled for the neighbourhood, you may enter a Google Analytics tracking ID here to track your project pages.
- **Support Page**: Use this to indicate the best way for your users to get help with your software.
- **Icon**: This is where you can upload the icon for your project. It will be shown on the project home page and in search results. If no icon is uploaded, the default project icon

will be used.

- **Project Status**: If you later stop development, move hosting to a different location, or want to delete the project, you can use this setting to indicate that. “Deleting” a project doesn't actually remove the contents of the project, it will make it inaccessible to anyone except for project administrators, who may also “undelete” a project.

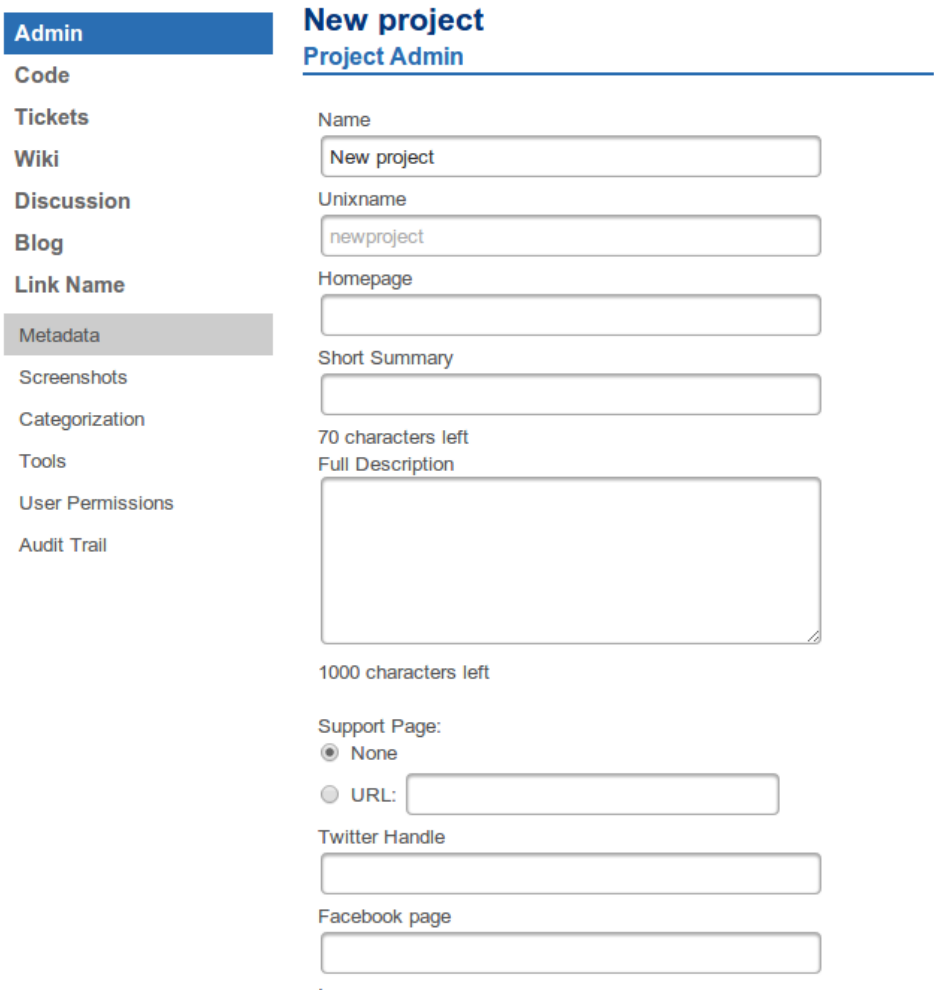


Figure 10: Project's metadata form

3.2.3. Screenshots

This form allows the users to add screenshots and respective captions to concerning project. This option can be accessed through the following flow:

Login > Projects > *ProjectName* **> Admin > Screenshots:** Project's screenshots

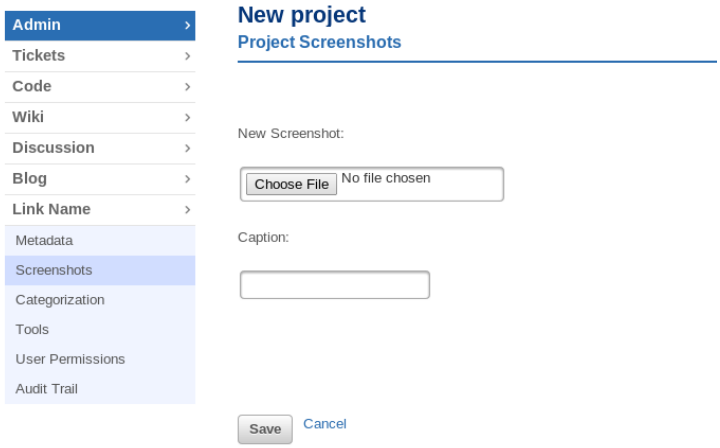


Figure 11: Screenshots form

At this location, shown in Figure 11, the user can add several screenshots, and corresponding caption, that illustrate the project. This can be for example, operational screenshots, user interface, or any type of image desired by the project administrator.

3.2.4. Categorization

The categorization of the project influences how the platform indexes the platform, concerning the type of project and all of the associated information. This option is more relevant in public projects, as it influences the way the community perceives the project.

Login > Projects > *ProjectName* **> Admin > Categorization:** Project's categorization

At this page, shown in Figure 12, the user can set the project's labels and add topic categories, license categories, database environment categories, development status categories, intended audience categories, operating system categories, programming

language categories, translations categories and user interface categories.

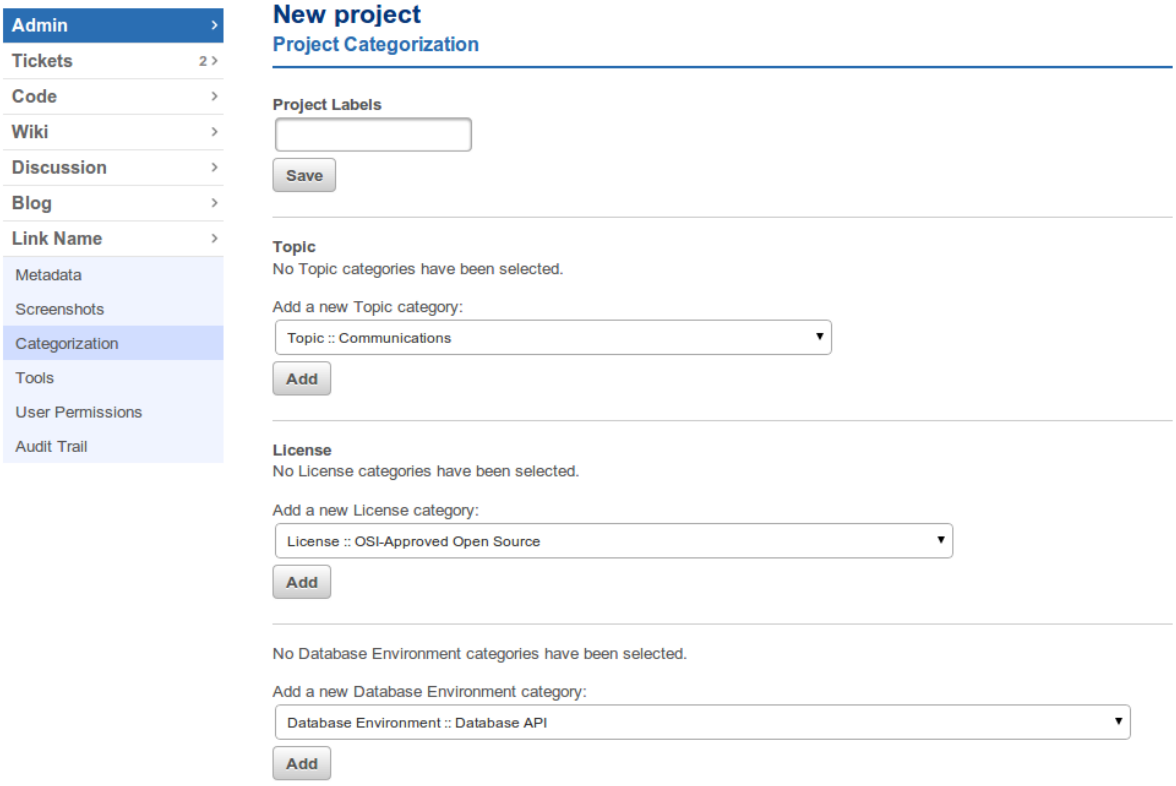


Figure 12: Part of the Categorization form

The project categorization will be the subject of future updates and reflections as PROSE strives to provide an easy path towards open-sourcing projects. As mentioned, categorization will greatly influence how the project is perceived and used, and should be well document in order to assist ICT projects towards creating vibrant FLOSS communities.

3.2.5. Tools

Each project is composed of several tools. This provides a very flexible approach allowing projects to turn on or off the the features that are more relevant to each effort. This is aligned with the responses obtained in D2.1, where the basic tool set was well defined, but more advanced features showed mixed results concerning interest.

Login > Projects > *ProjectName* **> Admin > Tools:** Select desired tools for the project

To enable/disable project tools it is necessary to have administrator rights. On the admin screen of your project, click on the Tools link.

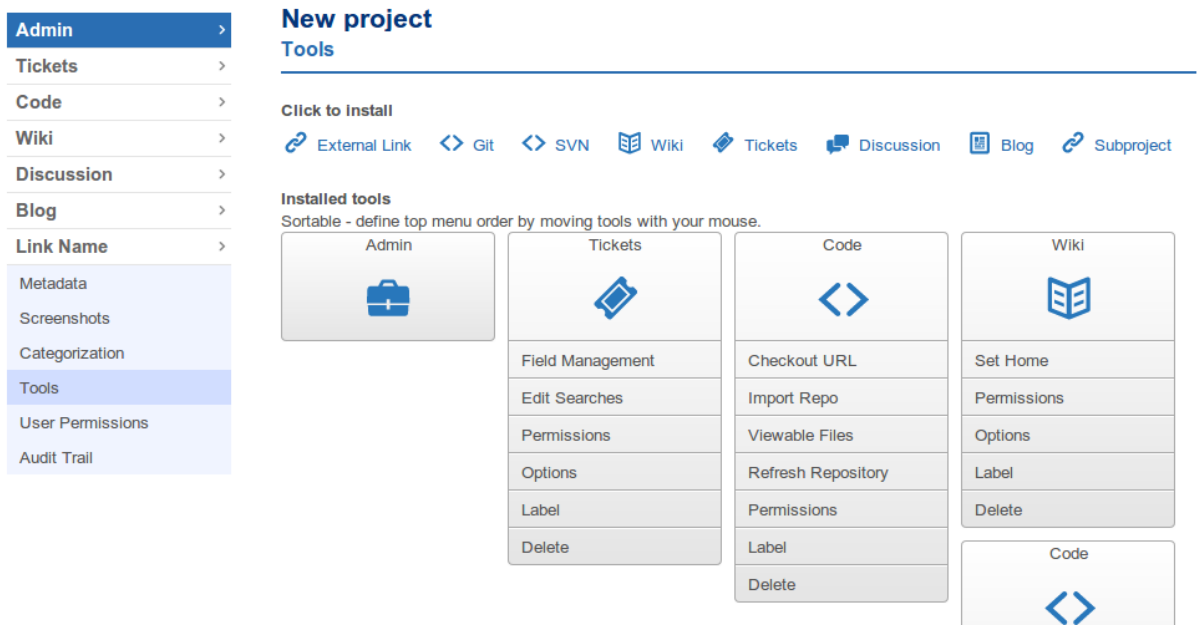


Figure 13: Tools section of the Admin screen of a project

In the top of the platform's section, there is a list of available tools, as shown in Figure 13. Clicking the desired tool in the “Click to Install” section, will prompt for a "Label" to describe the tool and "Mount point" for the tool in the project:

The **Label** is what shows in the project's navigation bar, this can be re-configured later if necessary.

The **Mount Point** is used to determine the URL of the tool. This cannot be changed after the tool is created. For example, if my project's URL name is uberproject and I create a new Tickets tool with the mount point bugs, that new Tickets instance will be installed to http://<domain>/p/uberproject/bugs/

If you decide you no longer need one of the tools available, simply click Delete and the tool will be permanently removed from your project.

You can install multiple tools of the same type. They will be collapsed into a single menu item to save space in the navigation bar. The bottom of Admin > Tools will let you change the threshold for the number of the same tool before collapsing same types.

3.2.6. Creating a sub-project

Sub projects are a common part of large projects that require maintaining several tools and source code under a common group, such as an ICT project.

Login > Projects > *ProjectName* **> Admin > Tools > Subproject:** Connects another project to main project

The user is given two choices to create a sub-project: associate a sibling project or create a new project as sub-project of the main one.

To achieve either of available options, it is necessary to navigate to the main project's tools page (***Login > Projects > ProjectName > Admin > Tools***) and click to add a new ***Subproject tool***. A form will pop up for the user to fill in the sub-project's information, such as its Label (to be listed on sidebar) and its Mount Point (URL name from Create a Project form).

The platform then checks if the mount-point already exists and 1) if so, associates that corresponding project to the main one; or 2) creates a new project with specified title in Label and URL name (which won't also be listed in Project's list). This is saved by pressing the save button as shown in Figure 13.

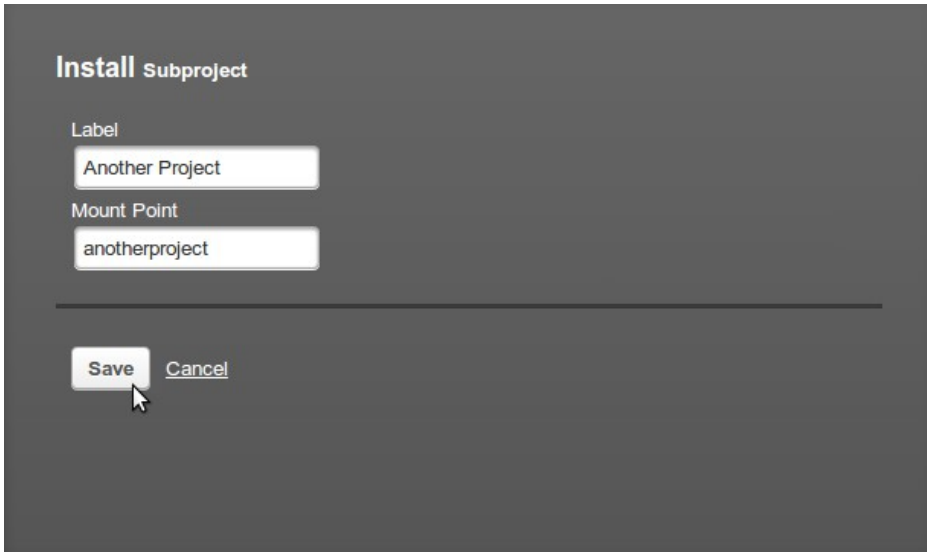


Figure 14: Install Subproject

After the pop-up window is dismissed, the new Subproject tool will be listed along with the other installed tools, as shown in the following figure.

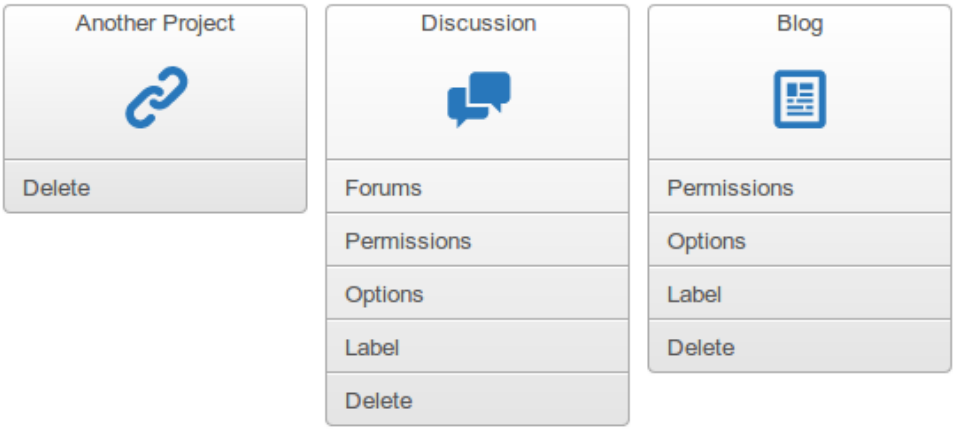


Figure 15: New Tool

3.2.7. User permissions

Permissions are an important part of the platform as they determine the capabilities of each user on the platform. More importantly they allow defining the visibility and scope of projects. This is particularly interesting for defining the visibility of projects, since it allows creating private projects to either specific users or groups of users.

3.2.7.1. Project wide User groups and permission settings

It is possible to manage individual permissions regarding projects, either by managing single users or by adding groups of users, from the User Permissions administration screen.

Login > Projects > *ProjectName* **> Admin > User Permissions:** Manage user permissions to corresponding project

To add a user to a group, select "+ Add" under the appropriate group and enter their *username*. Note that this is the unique *username*, not a display name.

To remove a user, select the "-" (minus) next to their *username* and confirm removal, as identified in Figure 15.

Default Groups

- **Admin**: Project administrators have full control over the projects, and can add/remove other members
- **Developer**: This group also contains all Admins, most tools will allow Developers to perform most non-admin functions by default, though these can be customized using

Individual Tool Permissions.

- **Members**: This group contains all developers, members generally have fewer permissions by default, but like Developers, this can be customized via Individual Tool Permissions.
- ***Authenticated**: Any user signed into *OpenSourceProjects.eu*.
- ***Anonymous**: Everyone, regardless of whether they are logged in or out.

Keep in mind that the "Member" group contains the "Developer" group, and the "Developer" group also contains the "Admin" group. In other words, an Admin is also, by inheritance, in the Developer and Member groups.

Likewise, "*anonymous" (all users, both logged in and logged out) is a supergroup of "*authenticated" (i.e., logged in users only).

Custom Groups

Use the "+ Add a new group" link at the bottom of the page to add a new group. This can be used to define specific teams (eg., "documentation", or "support") that have their own customised permissions set under the Individual Tool permissions.

Permissions

Permissions are grouped into three important categories:

- **Read**: Defines who can view content, note however that we do not support removing *anonymous read permissions from the project level
- **Admin**: Defines groups with administrative access (ie., full control) of a project
- **Create/Update**: These settings are currently not useful without Admin privileges and their usage is under review[6].

These permissions allow a very complete and flexible control over the entire platform, and more importantly allow project managers to control who is able to see the project, download software, and even contribute to the source code. Permissions are simpler to understand visually, as highlighted in Figure 16, as groups are easily perceivable and complicated features such as permissions inheritance are automatically handled and propagated through the user interface, as discussed next.

6 http://sourceforge.net/p/allura/tickets/6084/

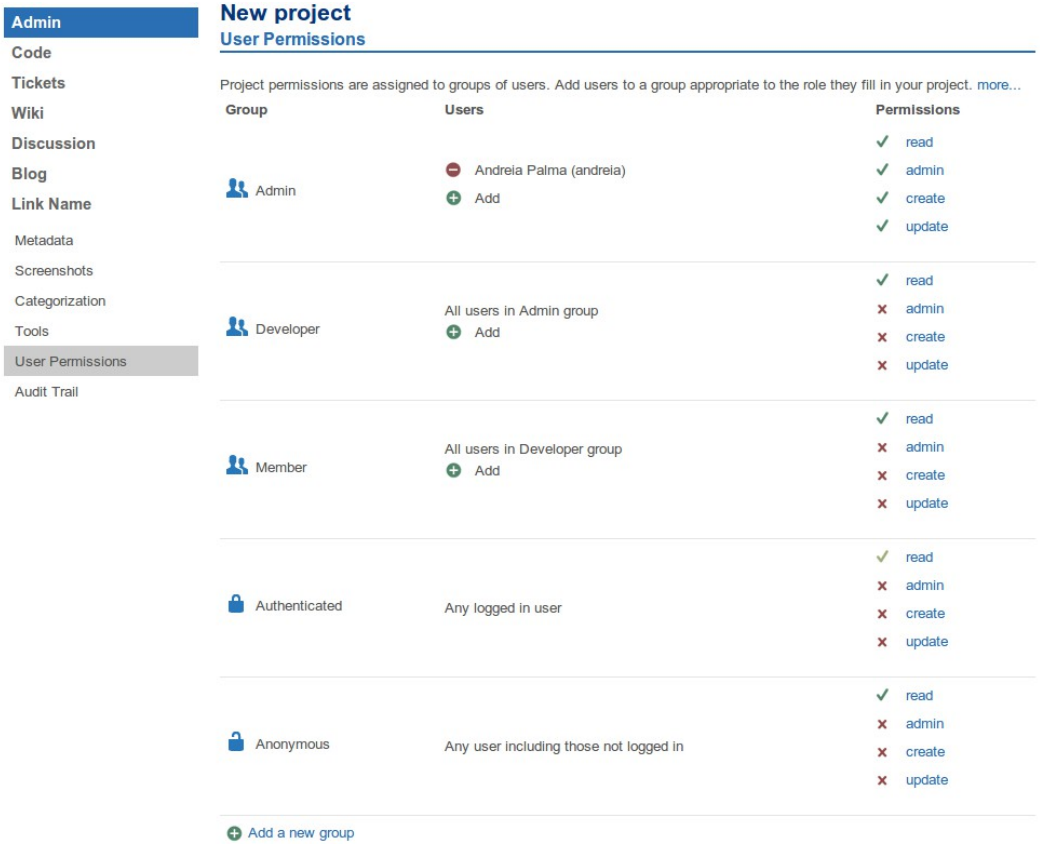


Figure 16: User permissions page

3.2.7.2. Individual tool permissions

As mentioned before it is possible to give permissions to different users for performing different tasks. This is possible because Allura supports permissions on a per-tool basis, allowing users to customize each tool's permissions, according to the project needs.

Login > Projects > *ProjectName* **> Admin > Tools > Permissions:** Manage permissions for corresponding tool within the specific too

Permissions for individual tools can be set via the Tools page, by clicking on the "Permissions" link underneath each installed tool, as demonstrated in Figure 16. These permissions are per tool, not per tool type. So if you want to edit permissions for each tickets instance, you will need to change the permissions for each one.

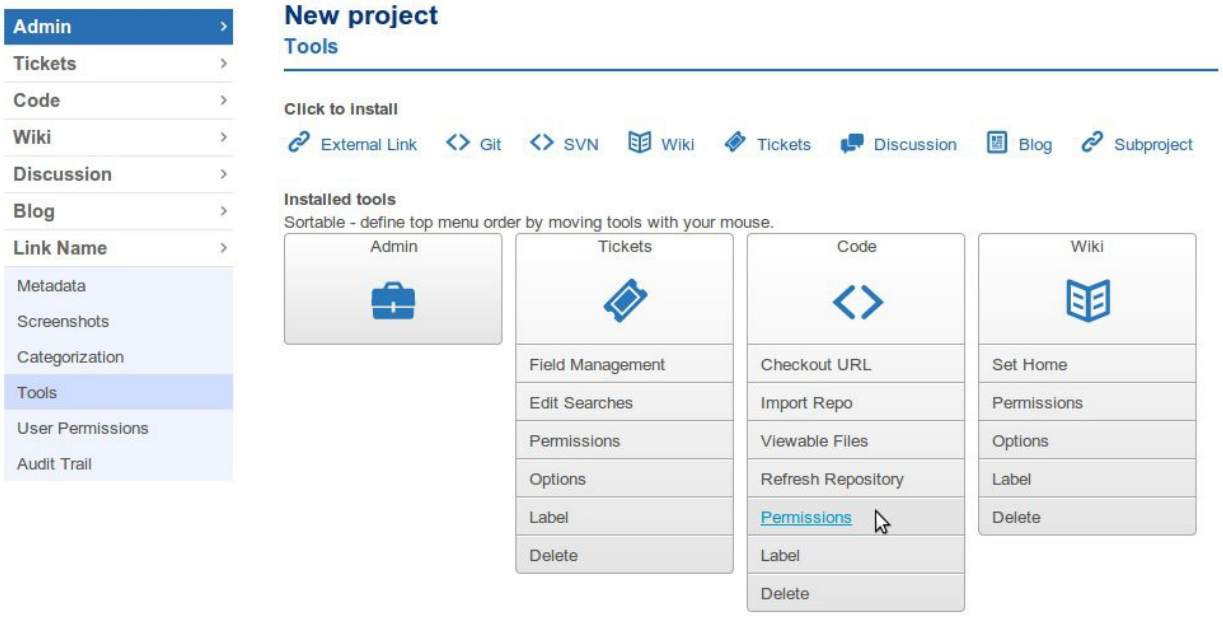


Figure 17: Tools permissions

Common tool permissions

A number of the permissions are used by a number of different tools.

- **ADMIN**: Controls permissions for administrator settings
- **READ**: Controls who can view the contents of the tool.
- **POST/UNMODERATED_POST**: Controls who can use the discussion features (eg., commenting on tickets, wiki pages, or merge requests). If a user is in *POST*, but not the *UNMODERATED_POST*, their comments will first enter a moderation queue before it's publicly viewable.
- **MODERATE**: Controls who can approve posts that have been moderated, as well as who can edit and delete comments.

Tool specific permissions

Several tools have permissions specific to the tool. However, we highlight the important ones in terms of the project's lifetime, concerning the actual code management and wiki:**Git**

- **SCMs[7] (GIT, SVN)**
  - **WRITE:** Controls what users can write to the repository (i.e., commit/push) (a more thorough example is shown in Figure 17.)

7 For SCMs, comment permissions (eg., POST, listed above) are used for comments on merge requests.

- **Wikis**
  - **CREATE**: Controls who can create new wiki pages
  - **EDIT**: Controls who can edit existing pages
  - **DELETE**: Controls who can delete existing pages
- **Tickets**
  - **CONFIGURE**: Controls who can edit milestones
  - **CREATE**: Controls who can log new tickets (note that further comments are controlled by the *POST* permission)
  - **UPDATE**: Controls who can update the description and metadata on existing tickets (ie., Summary, original description, etc.)
  - **DELETE**: Controls who can delete tickets
  - **SAVE_SEARCHES**: Controls who can save custom searches
- **Discussion**
  - **CONFIGURE**: Controls who can add forums

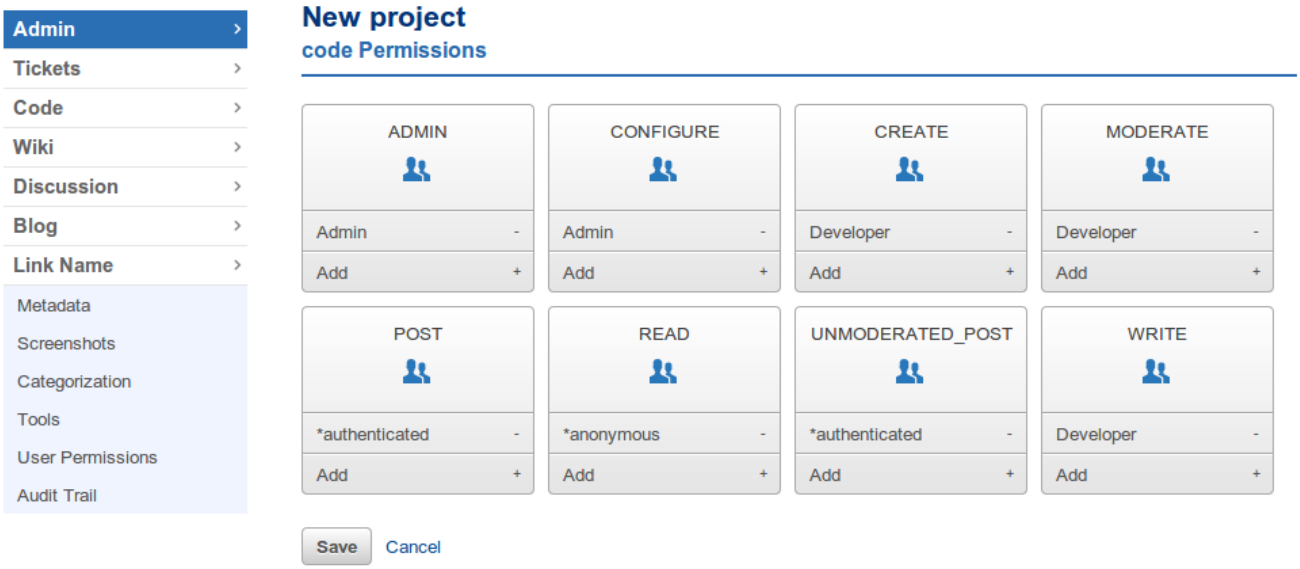


Figure 18: A sample permissions page for the code tool

3.2.8. Creating a private project

Creating a private project can be achieved by selecting the appropriate option on the project creation page.

Login > Projects > + Add a Project > Select “private”: Creates a new private project

A private project denies access to non-members of the project, such as anonymous and authenticated users.

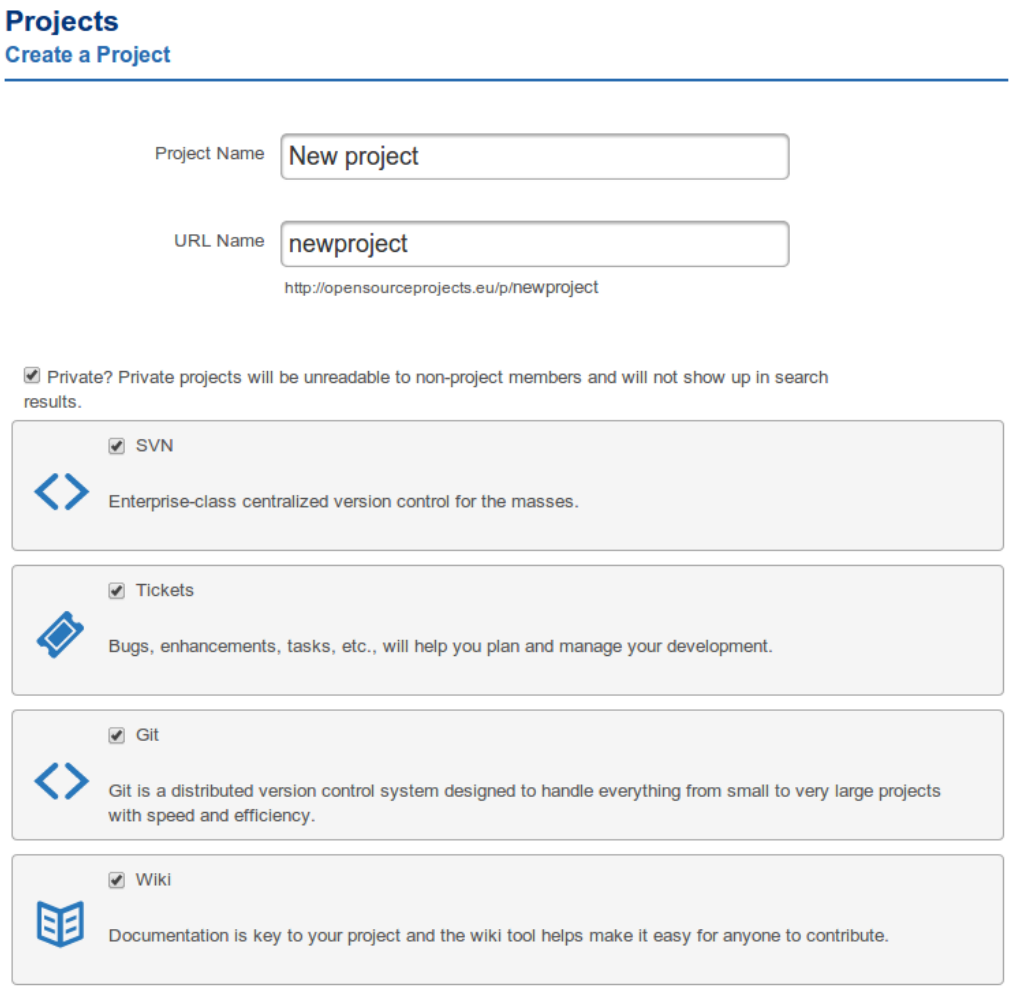


Figure 19: Create private project

In order to create a private project Add a Project page and select “Private” just below the project name and URL form, as identified in Figure 19.

Making a project private (or public) can also be done by changing its permissions. To transform a public project into a private one is a matter of navigating to the projects permissions and removing read access to anonymous and authenticated users. It may be also necessary to adjust the permissions for the installed tools accordingly. To achieve this, it is necessary to follow the instruction in Section 3.2.7..

3.3. Code management

An important part of any forge platform is the actual source code management. As of this deliverable, there are two supported technologies for source code management on the platform: Git and SVN. They are supported by adding the corresponding tool (as described in Section 3.2.5.), and can support multiple tools for the same project. Therefore it is possible to have more than one GIT repository for a project, or even concurrent GIT and SVN repositories[8].

3.3.1. Git

GIT is enabled as a project tool, as already mentioned. Once added a new Code entry will appear on the project.

Login > Projects > *ProjectName* **> Code:** Shows the project's configured source code repository.

After creating a new project, one can choose the needed tools for that project. For Source Code Management (SCM), when both Git and SVN are chosen, GIT is defined as the default when selecting the Code entry.

Git is a SCM, a tool for software developers which supports collaborative development of software within a team, and the tracking of changes to software source code over time.

Git is used by developers, and advanced users who need the very latest changes to the software (before releases occur). Software users generally do not need Git; typically they will download official file releases made available by the project instead.

Developers should familiarize themselves with Git by reading the Git Documentation[9] or traversing through a Git tutorial.

8 Internally the platform does not maintain any synchronization between repositories of the same project and they are handled completely independent of each other, therefore adding a GIT and an SVN repository will corresponding to two entirely different source code trees.

9 http://git-scm.com/doc

3.3.1.1. Using the GIT repository

The standard way to modify the contents of your repository is using a Git client as detailed in the Git User's Manual. If the code repository is empty, the code browser will display some instructions to help make the first commit and push, such as:

```
cd myproject
git init
# add all your files.  Use can use specific filenames or directories instead of '.'
git add .
git commit -a -m 'Initial commit'
git remote add origin http://user@opensourceprojects.eu/git/p/newproject/code-0
git push origin master
git branch --set-upstream master origin/master  # so 'git pull' will work later
```

If using Git with an existing repository:

```
cd myproject
git remote add origin http://andreia@opensourceprojects.eu/git/p/newproject/code-0
git push origin master
git branch --set-upstream master origin/master  # so 'git pull' will work later
```

We strongly recommend that when modifying a repository, other committers be notified of the direct edit window and that you make your own backups prior to editing the content so you can restore it readily yourself in the case of an accident.

3.3.1.2. Management

Login > Projects > *ProjectName* **> Tools > Code**

To manage the Git tool, go to Tools page and choose an option within the available from Git tool's block. It allows the administrators to set visible files, refresh the repository, set user permissions, set new label or delete tool as visible in figure below.

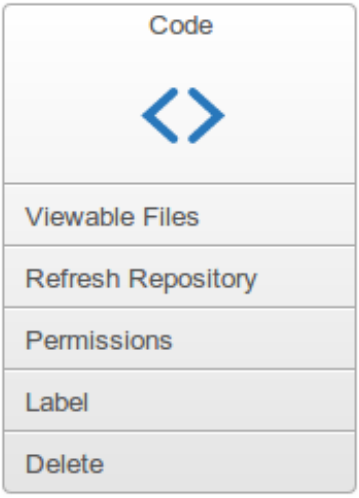


Figure 20: Git tool's block

3.3.2. SVN

Depending on the current tool configuration when adding an SVN repository it may be available on the Code tab or at the selected name for the tool.

Login > Projects > *ProjectName* **> Code**

Apache Subversion is a software version and revision control system distributed under an open source license. It allows users to maintain current and historical versions their project files.

To use an existing project it is only necessary to checkout the source code (command line example[10]) :

```
svn checkout --username=user http://opensourceprojects.eu/svn/p/project/code/trunk project-code
```

To import an already existing source code.

```
cd existing-helloworld-code
svn import http://opensourceprojects.eu/svn/p/project/code/ -m "Initial commit"
```

10 For other tools it is only necessary to use the project svn url in the appropriate locations and indicate by the tools' instructions.

3.3.2.1. Management

Login > Projects > *ProjectName* **> Tools > Code**

To manage the SVN tool, go to Tools page and choose an option within the available from SVN tool's block. It allows to checkout URL, import repository, set visible files, refresh the repository, set user permissions, set new label or delete tool as depicted in Figure 21.

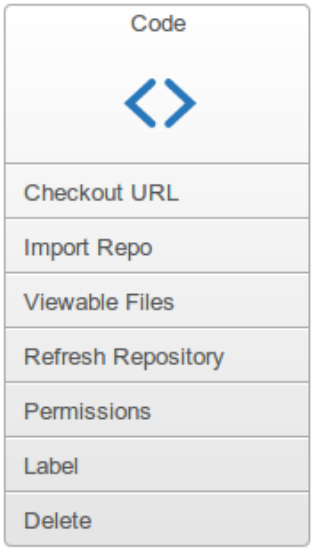


Figure 21: SVN tool's block

3.4. Issue management - Tickets

Issue management is available on the platform as one of the more important tools, as it directly supports the development process and allows the users and developers to interact with the project maintainers and main developers. In this section we provide an overview of how tickets are configured in a specific project and how end-users can take advantage of them.

Tickets are specific items that correspond to problems identified in the software, missing features requested by end users, or general communication items between developers and users. In many tools they are also know as “issues” or “bugs”, and hence the name of issue or bug trackers.

The ticket tracker tool is designed to allow for flexibility in how you track items of work that need to be completed. This includes bugs, feature requests from end users, or any other task you want to keep track of. You can even install multiple ticket trackers for different purposes. For example, a large project could have individual trackers for the design team, the

documentation team, the developers, and another for support. Each of these trackers can have their own unique list of tickets, as well as custom fields, custom milestones, and their own "mailing list." Tickets can even be moved between tickets instances, even if they're on different projects.

3.4.1. Configuring tickets

Login > Projects > *ProjectName* **> Admin > Tools > Tickets:** Allows users to report project's issues and suggestions

You may install a ticket tracker via Tools page and by clicking Tickets to get to a form similar to Section 3.2.5.. Once those instructions are followed, simply add the Ticket tool with the appropriate label and mount point, as seen in Figure 22.

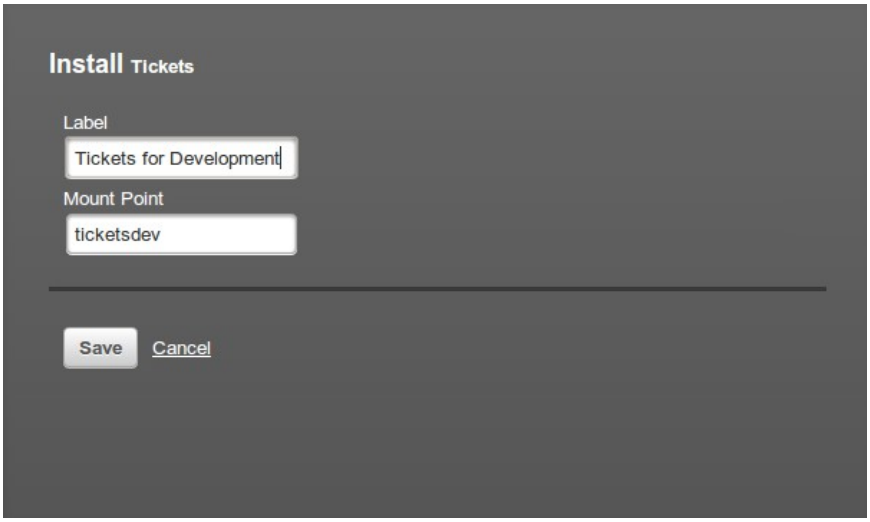


Figure 22: New ticket tool configuration.

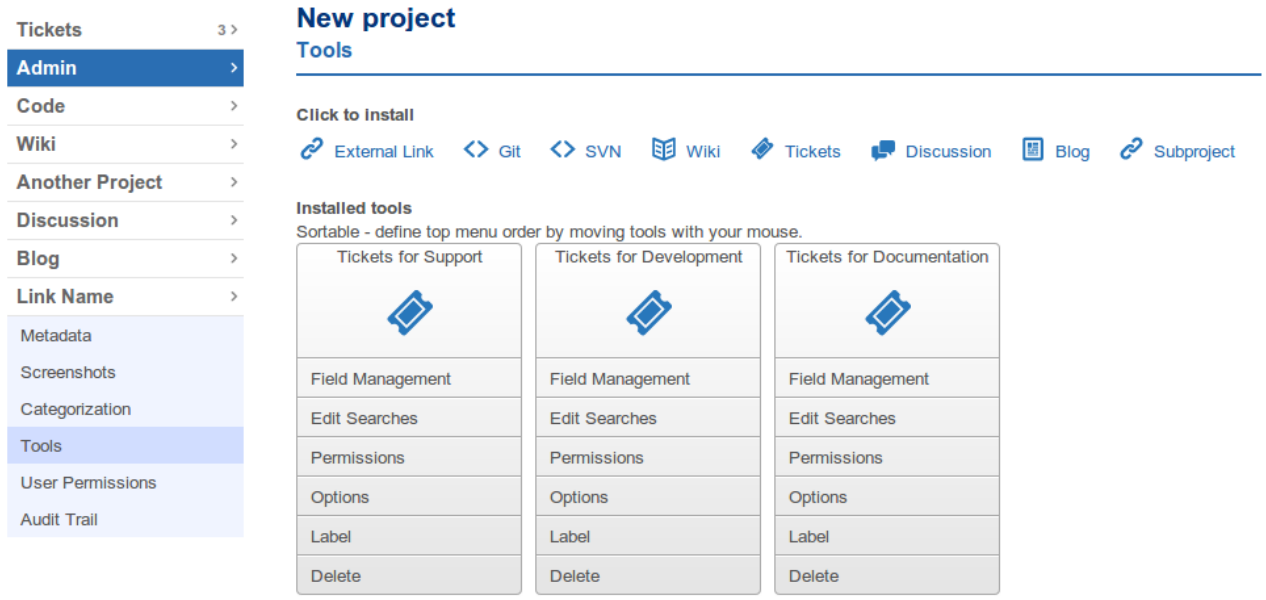


Figure 23: A project with three instances of the Tickets tool

After creating an instance of the Tickets tool (several may be created, depending on the projects needs[11]), the project tools page may look like Figure 23.

Once installed, there are a number of configuration options available for each tickets instance. In addition to the common tool options of Permissions, Label and Delete, there are also the following options specific to the Tickets tool:

3.4.1.1. Field Management

Field management allows administrators to configure how the fields in the ticket tracker appear:

Login > Projects > *ProjectName* **> Admin > Tools >** *TicketName* **> Field Management**

- **Open Statuses**: A space delimited list of status options for Open tickets. All open tickets are shown on the default ticket list
- **Closed Statuses**: A space delimited list of status options for Closed tickets. Closed tickets are not shown on the default ticket list, but are visible in various searches.
- **Custom Fields**: See the Custom Fields section for full details
- **Default Fields**: Check the boxes of the default fields that you wish to display on list

11 Projects can create multiple instances depending on the required needs. An example would be a public ticket instance for interacting with community members, and a private ticket instance for developers only to track internal or cutting-edge development.

views like the default view, searches, milestones, etc. Custom fields

If you are an admin on a project you can also add and delete custom fields to your tracker. Custom fields can be used to track everything from a list of components, to the relative size of the ticket, the QA person responsible for the ticket, or (hopefully) anything you can imagine.

You can edit the Custom Fields for the tracker under **Admin** > **Tools** for your project. Where your "tickets" instance is mentioned, select **Field Management**

There's a button there to add a new field, which results in Figure 24. Custom fields have the following options:

- **Label**: What the custom field is called
- **Show in list view**: whether or not this field should display in searches or other ticket listings.
- **Type:** Different type of information for the new field as shown below.
  - **Text**: Allows free form text
  - **Number**: is displayed as a free form text field, but will only allow numbers to be submitted
  - **Boolean**: a checkbox that can be checked or unchecked (i.e. true/false)
  - **Select**: Offers users a dropdown of preselected options. The options are space delimited, and a * prefix will designate a default selection (otherwise the first option will be the default)
  - **Milestone**: a milestone field is a special field which can be used to categorize tickets with a (optional) due date. Open milestones will automatically be added to the search bar on the side of the ticket views.
  - **User**: a drop down that lists project members

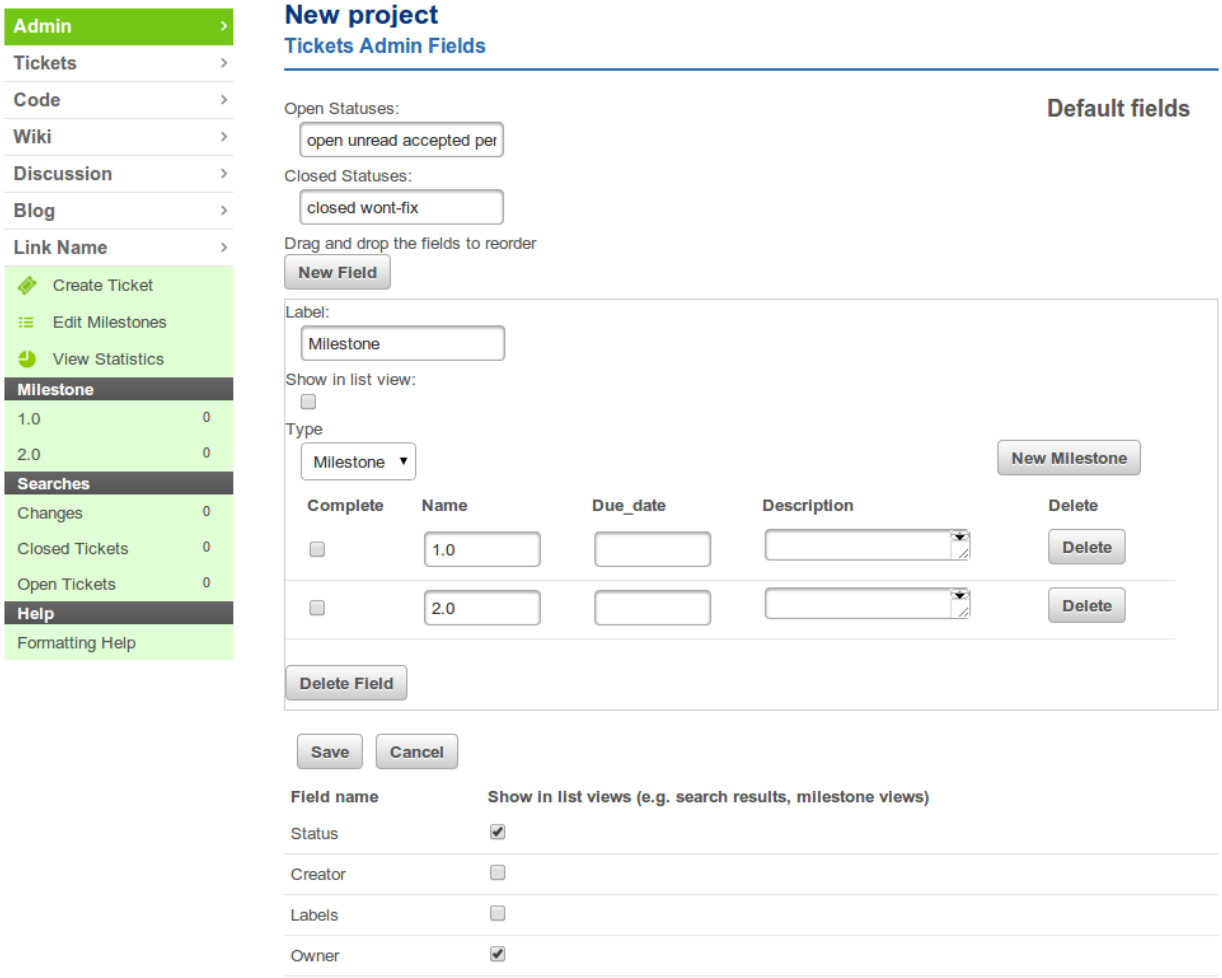


Figure 24: Field management page, adding a custom field

3.4.1.2. Configurable Searches

Login > Projects > *ProjectName* **> Admin > Tools >** *TicketName* **> Edit Searches**

If you have a search that you want all users of the tracker to be able to share, you can save that search with a name and it will be placed in the left navigation bar of the tracker for everybody to use. Commonly saved searches are "all open tickets," "all unassigned tickets," etc. You can use these saved searches to watch the flow of tickets through the system and make it easier for your development team to focus on groups of active tickets.

To edit searches, it is necessary to go through the following steps:

Login > Projects > *ProjectName* **> Admin > Tools >** *TicketName* **> Edit Searches > Edit > Save.**

These steps will enable defining custom searches and edit them, which may facilitate the way users and developers interact with the ticketing system. Managing these options is a straightforward process, as seen in Figure 25, where the interface allows a very visual management of the searches.

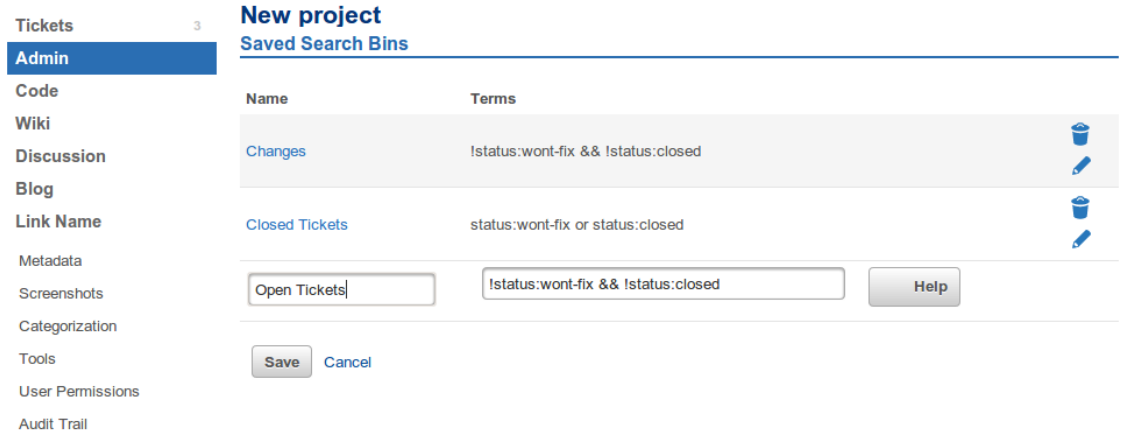


Figure 25: Editing a search on Tickets

3.4.1.3. Ticket Options

The default behaviour of the Tickets tool can be modified through the tool options in the Admin section, reachable through the following steps:

Login > Projects > *ProjectName* **> Admin > Tools >** *TicketName* **> Options**

At this location it is possible to:

- **Enable voting on tickets**: This will enable the voting feature which will let users vote tickets up or down. Votes will also be indexed in search, so you can search and sort by votes.
- **Email ticket notifications to**: Most commonly used for sending ticket notifications to mailing lists.
- **Help text**: This help text will be shown to users on new ticket pages and ticket listings respectively. This section may be markdown formatted.

3.4.2. Using tickets

Tickets are a very powerful tool that enable interaction between users and developers. Previously we have highlighted how a project administrator can setup the ticket system and customize it to fit the project's needs. However, it is also necessary for the end-user to interact with the system.

3.4.2.1. Creating and editing tickets

With the appropriate permissions, you can create a new ticket using the **Create Ticket** link in the left sidebar, highlighted in Figure 26, reachable through the following steps:

Login > Projects > *ProjectName* **> Tickets > Create Ticket**

At the new ticket form, fill in a title, any appropriate fields, and an issue description. To edit the main ticket description or fields on an existing ticket (again, assuming appropriate permissions), select the **EDIT** button in the upper right of a ticket. This view is similar to the new ticket view, except it doesn't display any new ticket help text, and there's space at the bottom to provide a comment for the edit.

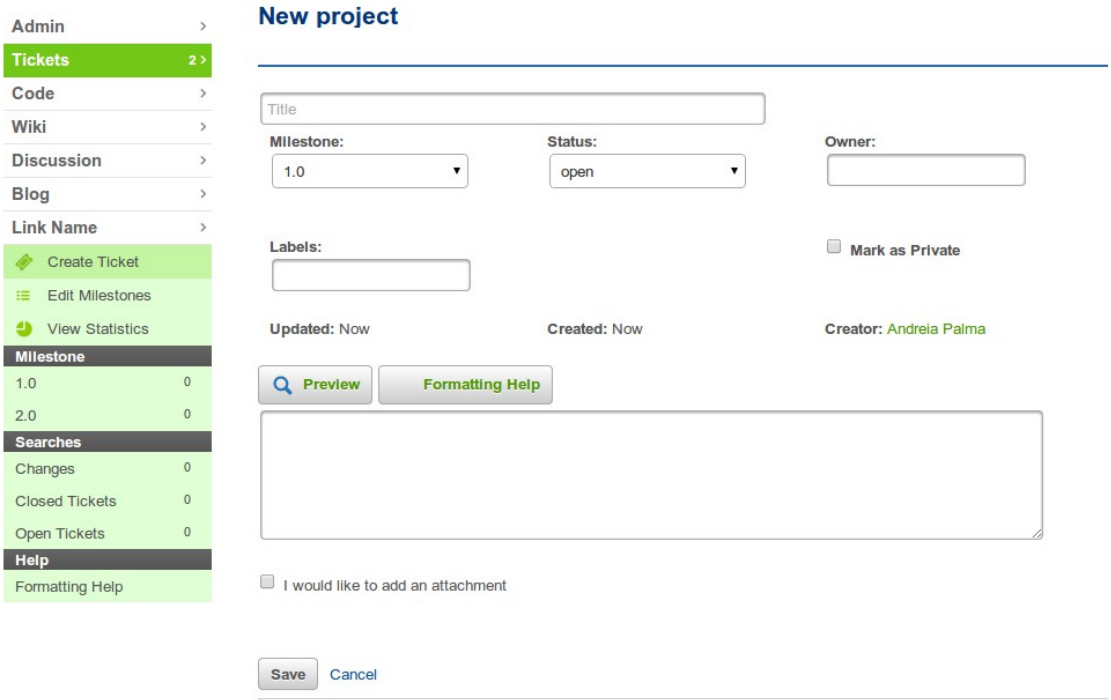


Figure 26: Creating a ticket

If further discussion about a ticket is needed, use the comment field at the bottom of the screen in Figure 26.

It is important to understand that Tickets may also be created and edited using the Allura API[12], detailed in Section 2.4.. This will allow for integrating the ticket system with external tools when necessary.

3.4.2.2. Moving and Deleting Tickets

To move a ticket, select the **Move** option in the upper right of the ticket view. This will then display all the valid ticket instances you have the proper permissions to move the ticket to. Any field values that can't be converted will be noted in the move message.

To delete a ticket, select the **Trash Can** icon in the upper right of the ticket view. This ticket will now be "deleted", however it may still be "undeleted" if you select the **+** icon which replaces the **Trash Can** icon. If a deleted ticket matches a list view, there will be a "Show deleted tickets" link which will allow you to see them.

3.4.2.3. Ticket Lists

The default view shows all open tickets, most recent first. Open milestones are listed on the left sidebar and will each generate a list view for each milestone.

3.4.2.4. Searching tickets

You can do full text searches for tickets at any time, but sometimes you want a bit more. You want to be able to filter results by milestone, or by who's working on the ticket, or by status, or some combination of the above.

Searches use the Solr Lucene query syntax[13]. On any search page you will see a "Help" button which will show the different available search fields and some search examples.

3.4.2.5. Bulk Edit

To edit a group of tickets all at once, start with a list view, then click the **Bulk Edit** button at the upper right of that page (i.e., the pencil icon). Check the boxes on the tickets you want to edit, and enter the new values for the tickets in the form provided.

3.4.2.6. Integrated Comments and E-mail Subscriptions

One of the fundamental features of the tracker is that every ticket is its own mini-mailing list. Ultimately this idea comes from another bug tracker ***roundup***[14] which called the idea *nosy lists*. The idea is that everybody who edits or comments on a ticket is added to a mailing list

12 https://sourceforge.net/p/forge/documentation/Allura%20API/#tracker
13 http://www.solrtutorial.com/solr-query-syntax.html
14 http://roundup.sourceforge.net/

(along with anybody who subscribes to that ticket). That mailing list is integrated with the in-page comments for that ticket, and makes it easy to keep up with the tickets in various trackers that interest you.

If you have your e-mail address set up in your user settings, you will automatically be subscribed to any ticket you edit.

You can also click the subscribe link (e-mail icon) on any ticket page to subscribe/unsubscribe from any ticket's mailing list.

If you would like to subscribe to every change/every comment on the whole tracker there is another subscribe link right at the top of the tracker page.

3.4.2.7. Integrated Markdown-Style Formatting Support

In all of the forge tools you can use the markdown syntax to create rich formatting of your text or comments on tickets. For more information on using markdown syntax see SourceForge Markdown Syntax Guide[15].

3.5. Communication Tools

Communication is key for the engagement between users and projects. The platform allows users to interact through two methods of communication, via the Wiki, and the forum. There is an implicit third one-way channel which is via the blog. This section helps user to learn how to configure and use each of these tools.

3.5.1. Wiki

The wiki is a collaborative document editor which is easily edited and can be used for various documentation needs, both internal and external. By default any project developer can edit the project wiki, even though permissions can be regulated (as for all tools in the platform). Like most modern wikis, all edits are non-destructive and a copy of every version is stored.

3.5.1.1. Configuring the Wiki

The wiki can be installed through the standard tool process highlighted in Section 3.2.5., the step summary is the following:

Login > Projects > *ProjectName* **> Admin > Tools > Wiki**

Once installed, there are a number of configuration options available for each wiki. In addition to the common tool options of permissions, Label and Delete, there are also these options

15 http://sourceforge.net/p/forge/documentation/markdown_syntax/

specific to the Wiki tool:

Set Home

Login > Projects > *ProjectName* **> Admin > Tools > Wiki > Set Home**

Set Home defines which page the wiki will redirect to when the base URL is used. By default, this is set to "Home".

For example, if the "**New Project**"[16] wiki has "Home" set as its Home page for its wiki, the main wiki URL *http://opensourceprojects.eu/p/newproject/wiki/* will actually redirect to *http://opensourceprojects.eu/p/newproject/wiki/Home/*, allowing the user to set a customized home page for the wiki.

Options

Login > Projects > *ProjectName* **> Admin > Tools > Wiki > Options**

The **Options** page in admin provides you with the ability to add or remove three different pieces of the wiki functionality.

- **Show discussion**: The comments section on each wiki page
- **Show left bar**: Links to page creation, browse pages/labels, and formatting help. All these pages will still be available if this is disabled, but you may need to use direct links.
- **Show metadata**: The author, label, and attachment display near the top of each page

If you want to maximize the screen real estate of the wiki, or treat it more as a CMS, you may want to remove all three of these options.

3.5.1.2. Using the wiki

Wiki content itself, like the rest of Allura tools, uses Markdown formatting[17], along with some additional handling for Artifact Links and some custom macros. An example of a default Wiki page is shown in Figure 27, which presents a page view and the available shortcuts, explained below.

16 New Project is the name of the project used in the example.
17 http://opensourceprojects.eu/u/wiki/markdown_syntax/

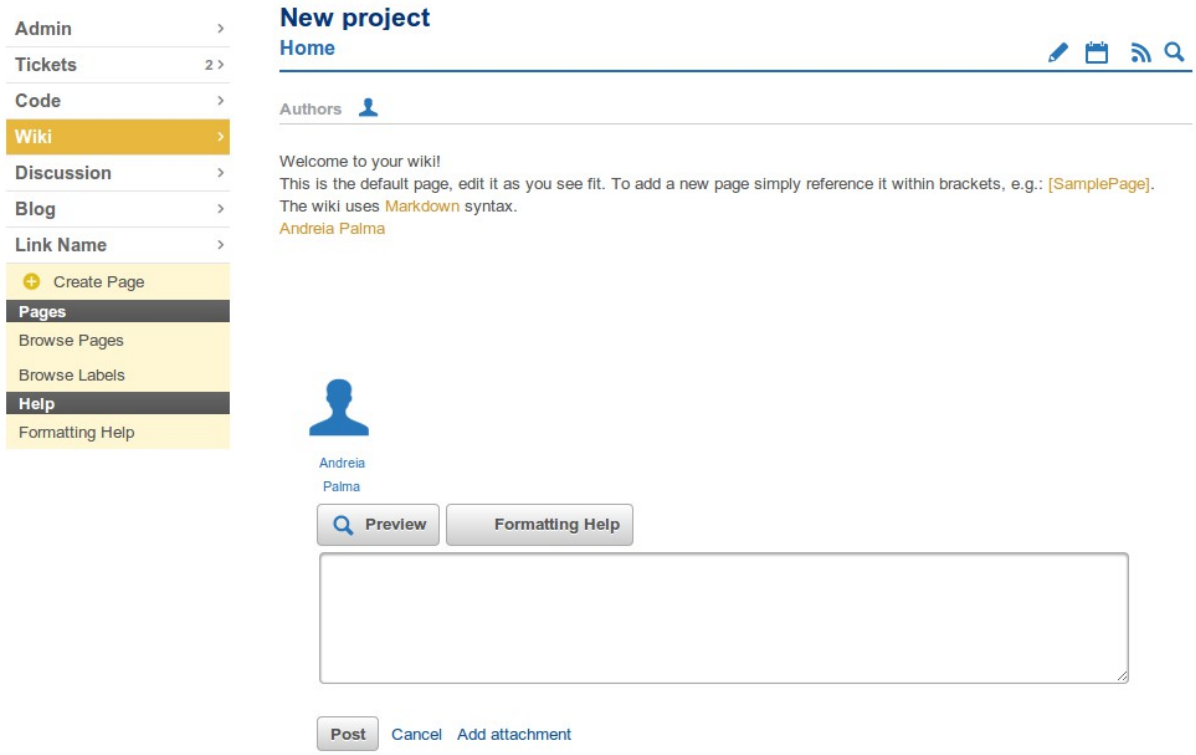


Figure 27: A wiki page with Edit, History, Subscribe, Feed and Search

Creating and Editing Pages

Login > Projects > *ProjectName* **> Wiki > Create Page**

Creating a page can be done a few different ways, assuming you have the appropriate permissions:

- From the left sidebar, select **Create Page** and enter the desired name.
- On an existing page, create a reference to a new page by putting the new page name in brackets (e.g. [New page]), then click on that link.
- Enter the URL where you want the page to reside, if it does not already exist you'll be presented with an edit link.

To edit an existing page, simply use the **Edit** link in the upper right of a page (the pencil icon).

Wiki pages may also be created and edited using the Allura API[18]

18 https://sourceforge.net/p/forge/documentation/Allura%20API/#wiki

Attachments

Login > Projects > *ProjectName* **>** *WikiName* **> Edit > Add attachment**

To add an attachment to a page, first save the page content, as you may lose unsaved changes otherwise. Select **Add attachment** button at the bottom of the edit view, as shown in Figure 28, then follow the steps presented:

1. Select a file to upload
2. Select **Attach file**

Repeat as necessary for multiple files.

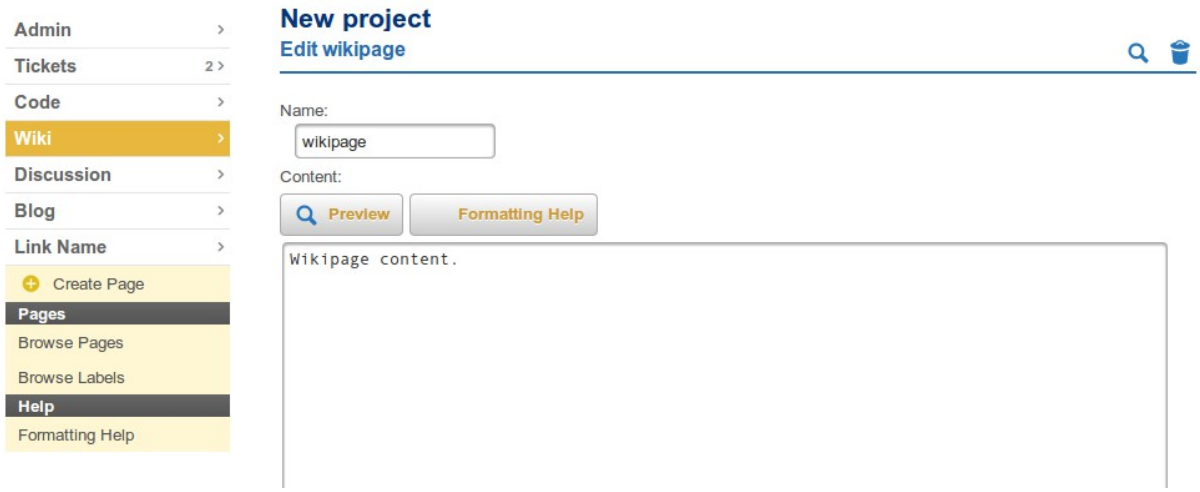


Figure 28: Adding attachments to the Wiki

Moving pages

Login > Projects > *ProjectName* **>** *WikiName* **> Edit > Name**

To move a page, first enter the edit view presented in Figure 29, then change the name of the page. Moving a page in this way will preserve the wiki page history.

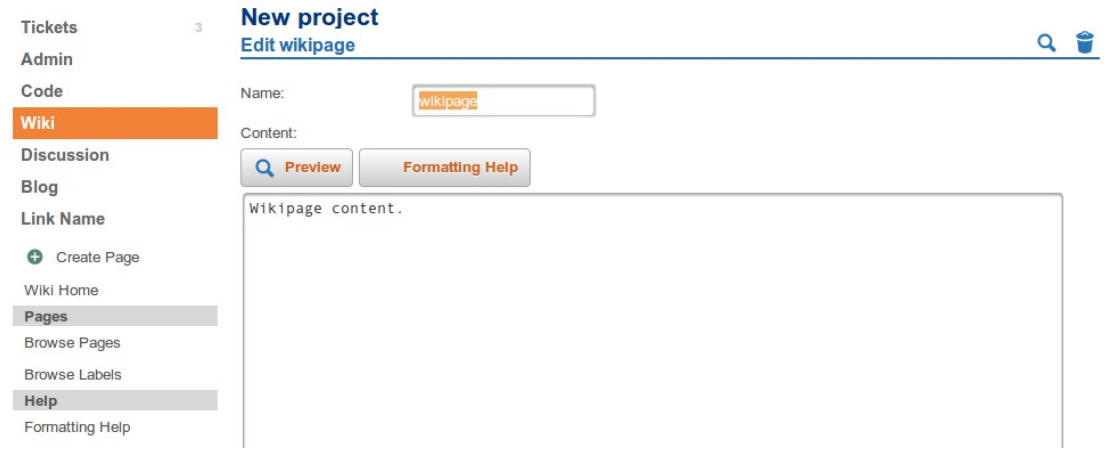


Figure 29: Editing a wiki page

Deleting pages

Login > Projects > *ProjectName* **>** *WikiName* **> Edit > Trash**

To delete a page, enter the edit view, as shown above, then select the **Trash** icon in the upper right. Deleting a page will remove it from view, however the page may still be undeleted if necessary. To do so, select the **+** icon when viewing the deleted page. You will then enter the edit view and you may rename the page name as appropriate.

Deleted pages can be found under the **Browse Pages** view by selecting the **Show deleted pages** link at the bottom of the list.

Viewing Page History and Reverting

Login > Projects > *ProjectName* **>** *WikiName* **> History**

To view the history of a page, select the **Newspaper** icon. This action will lead to the interface presented in Figure 30. From that page, it is possible to view specific revisions by selecting the **Magnifying glass** icon. The other icon will revert the page to that revision.

You can also compare revisions. Use the radio buttons to select the ones to compare, then select **Compare revisions.**

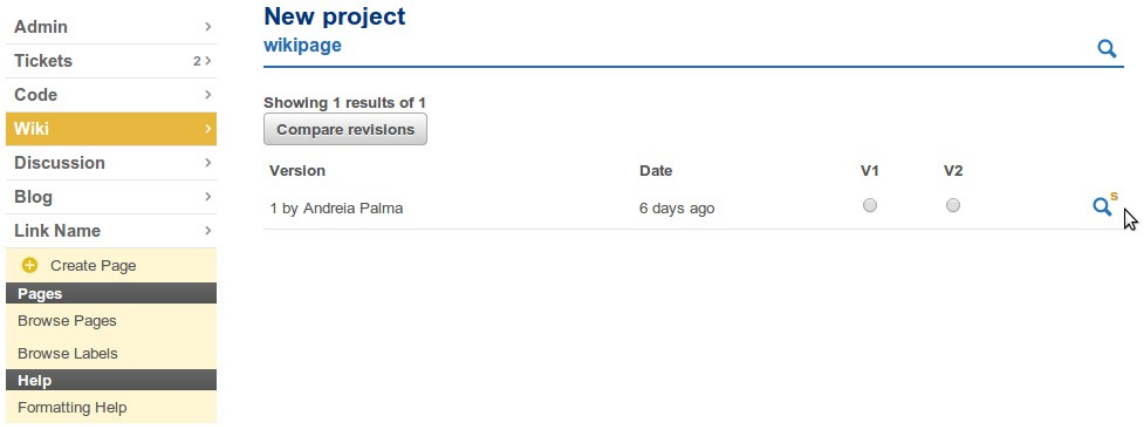


Figure 30: History page

3.5.2. Forum

Discussion forums are a way to collaborate with other developers on the project, or with end users. A project administrator can create as many different forums as needed by clicking on "Add Forum" on the left sidebar of the discussion page, provided that the discussion module is active (Section 3.5.2.1.). The steps are summarized as follows:

Login > Projects > *ProjectName* **> Discussion > Add Forum**

Example forums for end users include "Troubleshooting" or "Project Announcements", but it is up to the administrator to create and name as many forums as required by the project work flow, creating a very flexible environment.

3.5.2.1. Configuring Discussion

You may install or remove a Discussion via Admin > Tools, as already highlighted in the standards tool process in Section 3.2.5.. The step summary is the following:

Login > Projects > *ProjectName* **> Admin > Tools > Discussion**

Once installed, for each Discussion instance, there are a number of configuration options available. In addition to the common tool options of permissions, Label and Delete, there are also these options specific to the Discussion tool:

Forums

Login > Projects > *ProjectName* **> Discussion > Forum > Add another forum**

At the bottom of this page, shown in Figure 31, there's an **Add Forum** link. The following fields are requested upon forum creation:

- **Name**: The long display name for the forum
- **Short Name**: The short name that forms the URL this forum uses
- **Parent Forum**: Select a parent if the forum is to be a subforum of another one
- **Description**: A short description for this forum section, to help users determine the correct forum to use
- **Monitoring Email**: Enter an email address that notifications should be sent to. Often used with a developer mailing list.
- **Icon**: Select an Icon for the discussion
- **Developer Only**: This option can be used to set a forum to be viewable by developers only, even if most forums are publicly available
- **Anonymous Post**: You can use this option to allow anonymous posts to a specific forum even if they aren't generally allowed for the discussion tool.

These values may be edited after forum creation as well.

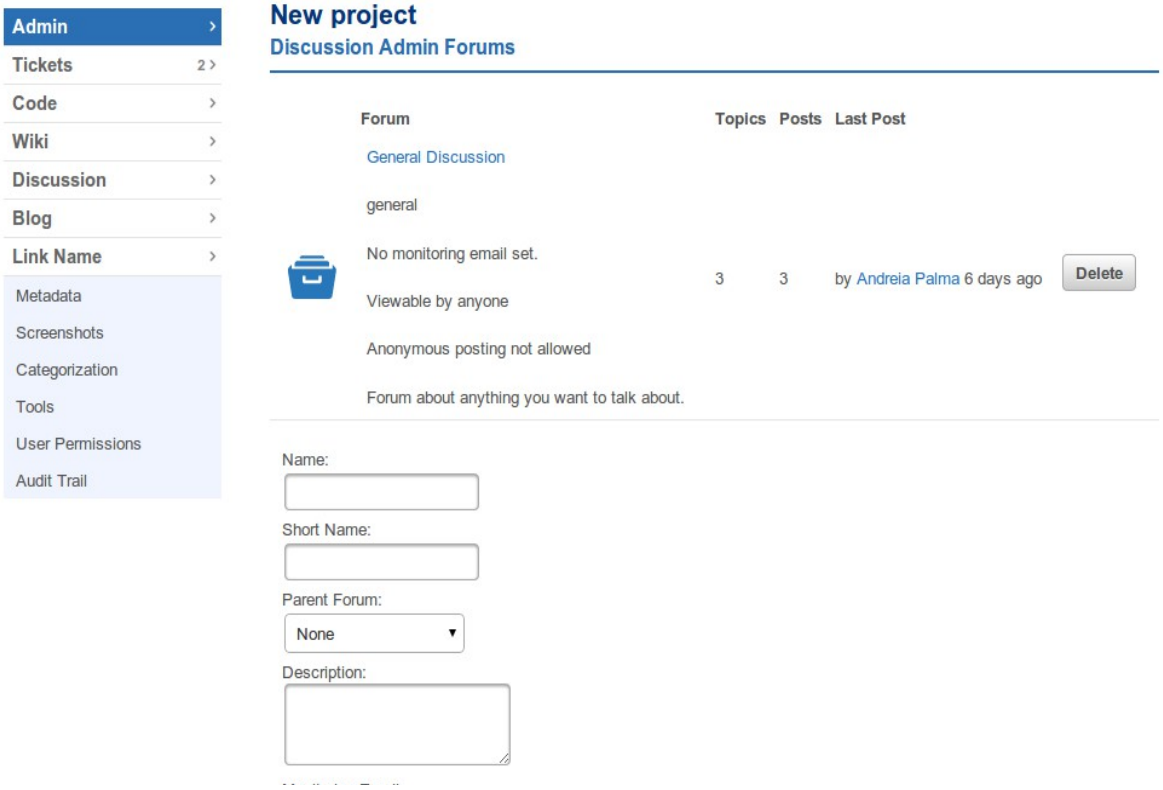


Figure 31: Add another forum page

Options

Login > Projects > *ProjectName* **> Discussion > Forums > Options**

The **Posting Policy** determines how moderated posts are handled according to the view presented in Figure 32:

- **Approve Once Moderated**: a post is approved after moderation
- **Approve All**: a post is approved immediately

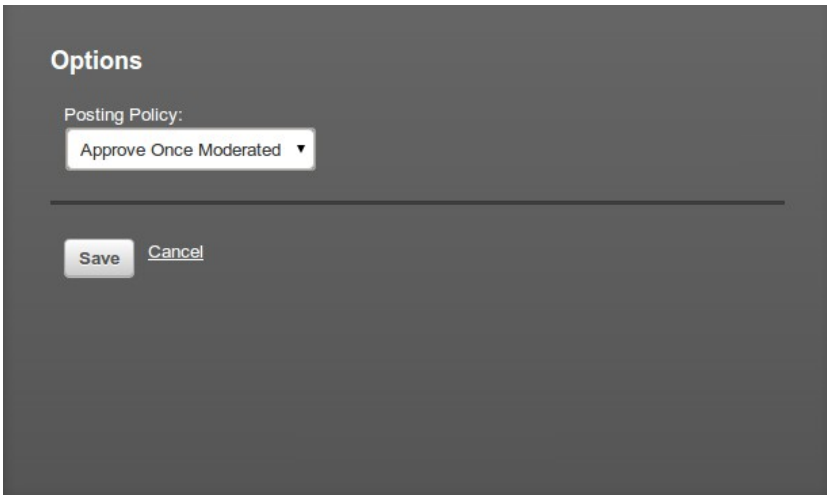


Figure 32: Forum options

3.5.2.2. Using Forums

As discussing for the ticket tracker, beyond the standard configuration of the tool on the admin interface, it is necessary to get familiarized with the forum process from the user's perspective.

Creating, posting and replying to threads

Login > Projects > *ProjectName* **> Discussion > Create Topic**

To create a new thread, use the "+ **Create Topic"** link on the left sidebar of the interface shown in Figure 33. Enter an appropriate **Subject**, **Forum**, and then enter a **Description** and post, an example of this form can be seen in the following figure.

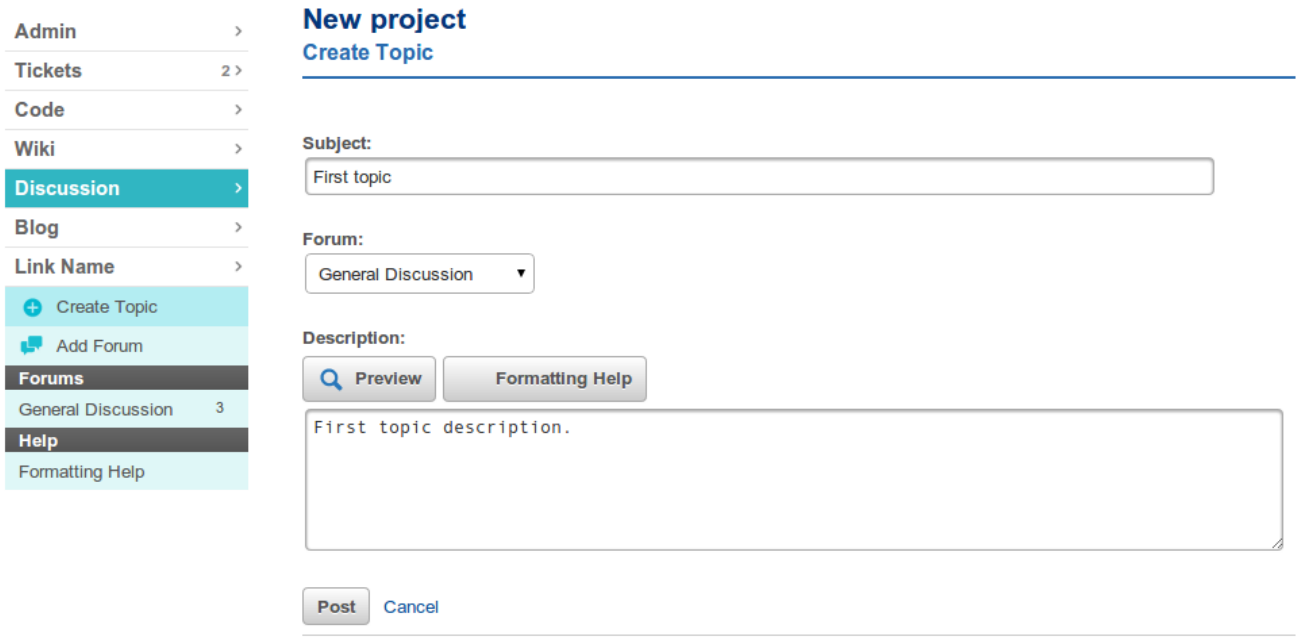


Figure 33: Creating new topic

Login > Projects > *ProjectName* **> Discussion > Create Topic > Reply/Post**

To reply to an existing thread, when viewing the thread either select "reply" next to a comment to reply to that specific comment, or use the dialogue box at the bottom of the discussion to add to the end. Both of these options are exemplified in Figure 34.

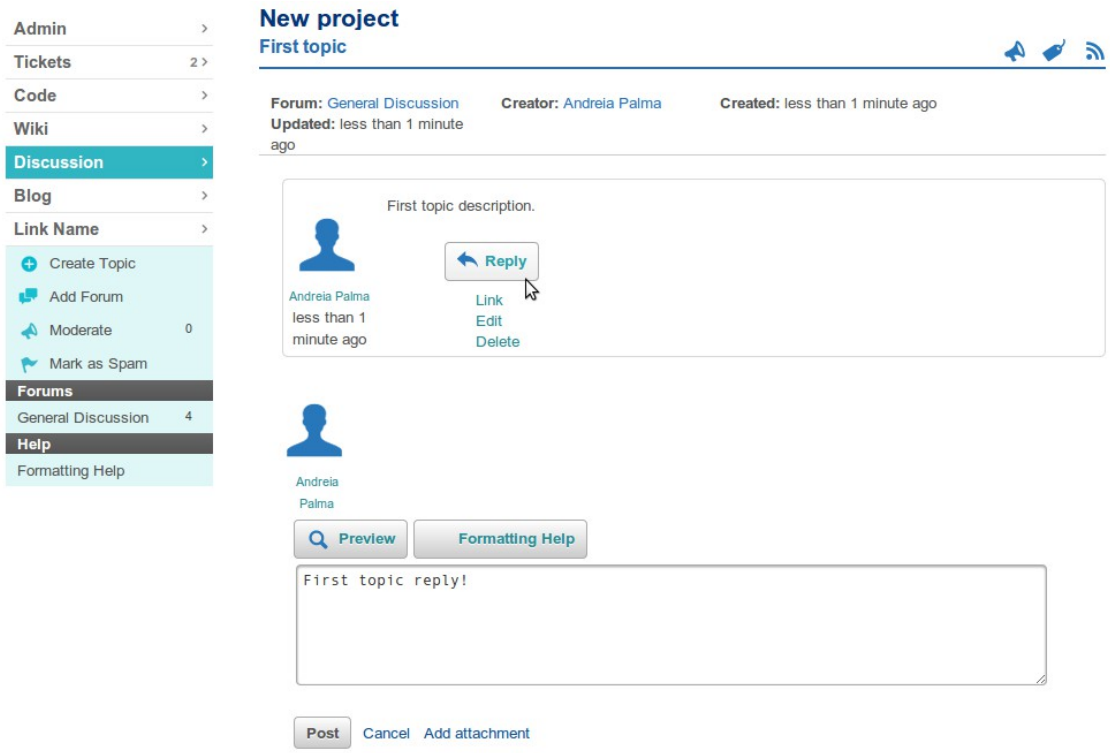


Figure 34: Topic reply

Moderation menu

With appropriate permissions, forum moderators have the following options available using the **Pencil** icon in the upper right corner (Figure 34).

- **Moving Threads:** To move a thread, simply select a new forum from the moderate menu and save.
- **Sticky/Announcement:** Check the appropriate values to set a thread as an Announcement or a Sticky thread.
  - **Announcement** gets a dedicated section at the top of the forum
  - **Sticky** stays at the top of the particular forum, and is marked **Sticky**, but it is still grouped with the other threads.

A thread may be both an **Announcement** and **Sticky.**

3.5.3. Blog

The blog tool can be used for providing news about your projects. Examples include release notes and change logs, announcing events related to your project, etc. Nowadays blogs are very important communication streams, as they provide a way of keeping up-to-date with project specific news and provide a direct communication channel to the target audience.

3.5.3.1. Configuring the Blog

As usual, the blog may be installed by adding the corresponding tool in the administration area via Admin > Tools shown in Section 3.2.5.. The step summary is the following:

Login > Projects > *ProjectName* **> Admin > Tools**

Once installed, there are a number of configuration options available for each Blog. In addition to the common tool options of permissions, Label and Delete, there are also options specific to the Blog tool.

External Feeds

External feeds may be added through this setting. These external feeds may be updated every three hours and add posts to the blog accordingly. Multiple feeds may be added, which can serve to aggregate related news about the project. Unchecking a feed will remove it from the list.

3.5.3.2. Using the Blog

Like all Allura tools, the blog uses Markdown syntax. Each blog post will also have a discussion section for users to comment on the post.

Posting and editing

Login > Projects > *ProjectName* **> Blog > New Post**

To create a new post, select "New Post" from the left sidebar according to the steps above. This will result in the screen shown in Figure 35. Enter a title and description for the post and **Save** to publish the post. If you save a draft for later editing before making it public, change

the **State** to **Draft** before saving. Select the **Edit** link to edit a post or saved draft. You may also delete a post from the **Edit** view.

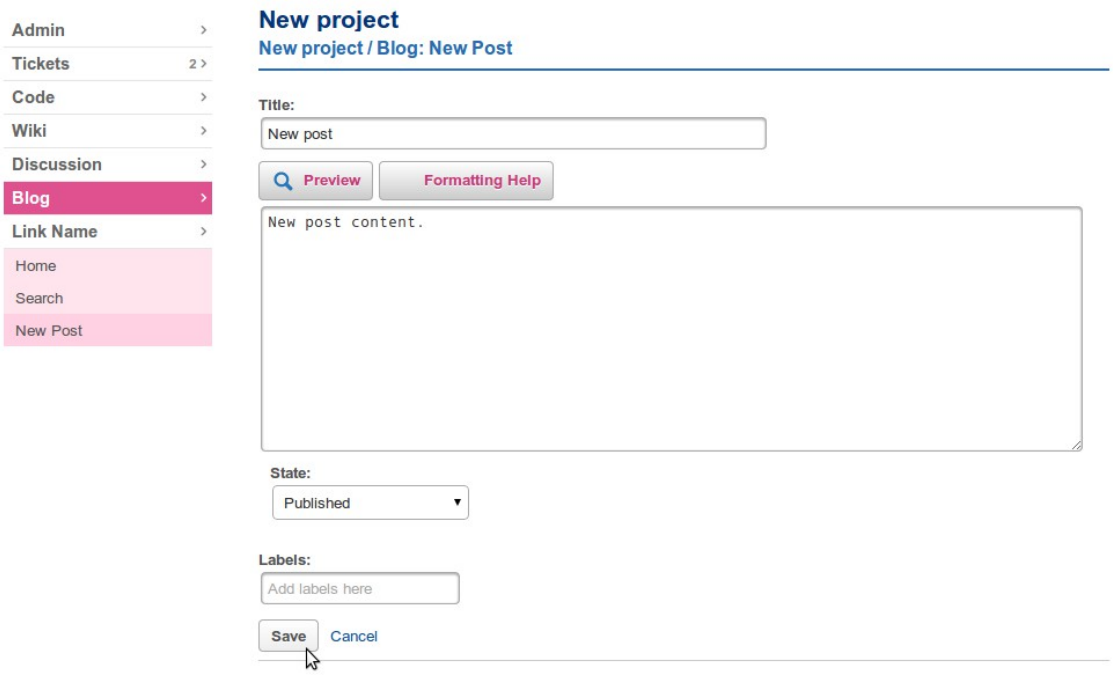


Figure 35: New post on blog

Note: The URL will be based on the original post date and title, and will *not* be updated if you change the contents of the post.

3.6. Statistics

The platform already provides some support for statistics collection and display. Even though limited, there are a few available options to projects. The more important issue now is to deal with in platform ticket statistics, and with external collection through Google analytics support. As the project grows in terms of in-platform statistics collection, more content will be added to this section so that project administrators are aware of the tools available to them.

3.6.1. Ticket statistics

Ticket statics are collected automatically by the platform, and presented to the users with permissions to view them.

Login > Projects > *ProjectName* **> Tickets > View Statistics**

In order to check collected statistics for tickets, it is necessary to follow the summary instructions: Once on the tickets page, select menu option for View Statistics, and the user will be directed to a page similar to the one presented in Figure 36.

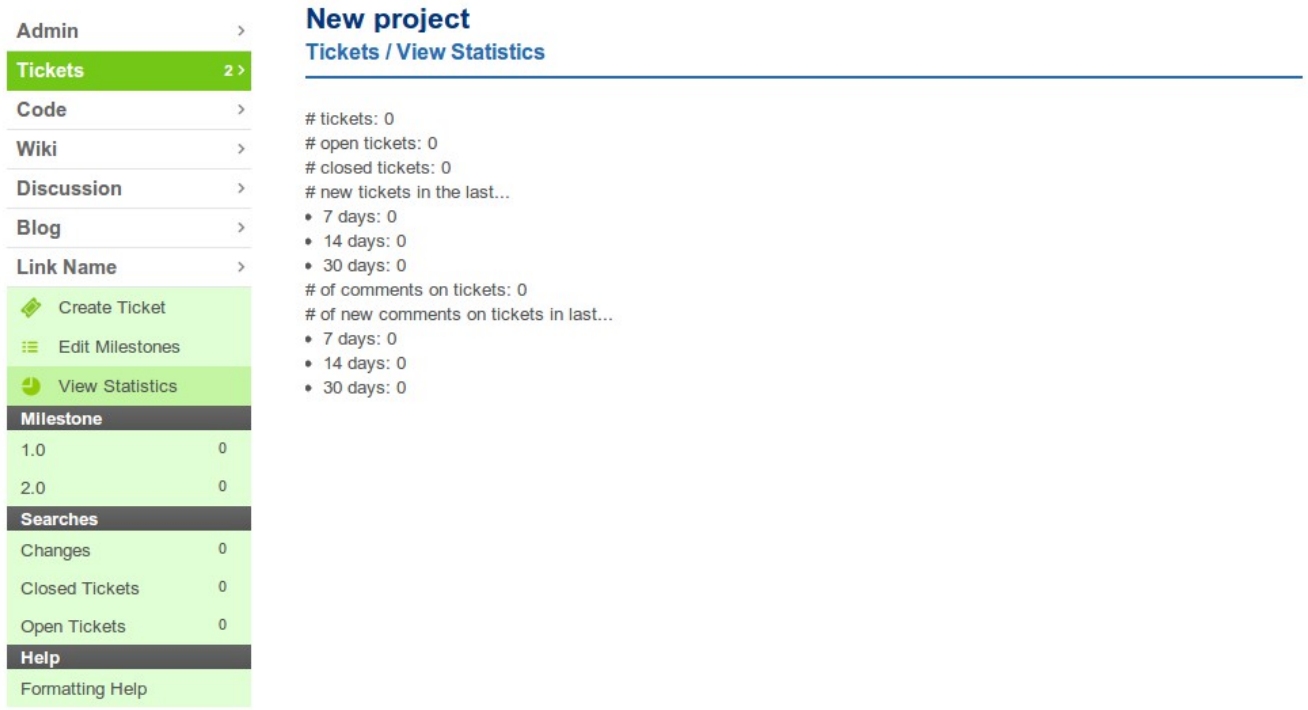


Figure 36: Statistics of a project

This already shows the most basic statistics to start analysing a project's activity and healthiness, regarding open and closed tickets, with a temporal factor.

3.6.2. Google Analytics

Google Analytics is a Google's service to track detailed statistics. To be able to use Google Analytics, it is necessary to fill in the tracking ID in project's metadata form. The metadata form can be reached through the following steps:

Login > Projects > *ProjectName* **> Admin > Metadata > Analytics Tracking ID**

Once on the metadata form, partially shown Figure 37 where several links can be inserted to completement the platform, it is only necessary to provide the Tracking ID (e.g. UA-123456-1) on the corresponding field form, which is provided by the Google Analytics service (this can be acquired by registering at the service webpage[19]).

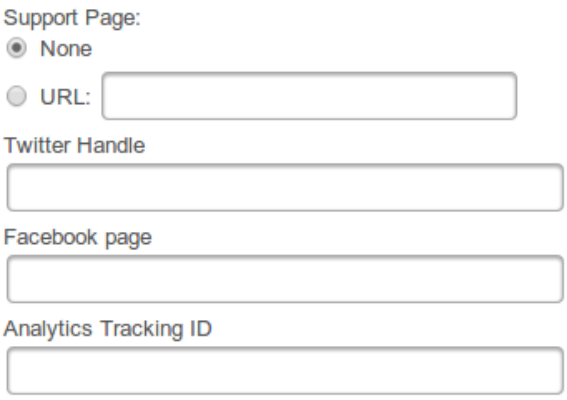


Figure 37: Google Analytics Tracking ID

4. Platform Roadmap (or Releases and Milestones)

As highlighted in the previous sections, the platform is defined by several features that build a cohesive experience around the work-flow of creating and maintaining FLOSS projects. This is aligned with the PROSE goals of promoting open source within the ICT ecosystem.

Strategy wise, the first goal is to actually engage projects on the platform for day-to-day use. To achieve this, the platform must facilitate the project's development cycle, making the software development support tools the first critical target for a phased release. In this scenario, the initial interaction of ICT project might even be only for private, internal project development. With a working set of tools it is possible to start disseminating the platform and to bring projects aboard, hosting their projects and supporting their development efforts.

19 http://www.google.com/analytics/

When projects are actively using the platform, we can start working towards outlining a strategy that results in open sourcing their software, addressing legal and business aspects in the process, and opening the effort to the external community. This also includes facilitating project governance as means towards building a thriving community around FLOSS achievements in the ICT.

The deployment milestones are aligned with the above-mentioned strategy. The initial focus is placed on releasing a functional version of the platform for development to start the dissemination process as soon as possible and to engage with projects as early as possible (M1.0 and M2.0). On a later stage the focus will be on actually open sourcing the software. From the platform perspective the focus will be integrating with 3rd party software either through the external API, or through tight integration with quality assessment tools coming from other ICT projects, or external tools (M3.0 and M4.0). This process is defined by a staged deployment process, focusing on the 4 important milestones:

- **M1.0: First release of the PROSE platform (July/2013)**
- **M2.0: Second Tier feature set (September/2013)**
- **M3.0: Functional External API (December/2013)**
- **M4.0: Beyond PROSE / Integration with 3rd parties (September/2014)**

Reaching the milestones in the time-frame highlighted in Figure 38 will provide an adequate path towards reaching ICT projects and enable the adoption of FLOSS. Each of the milestones, described in detail in the following sections, will be marked by public announcements, followed by the necessary dissemination process.

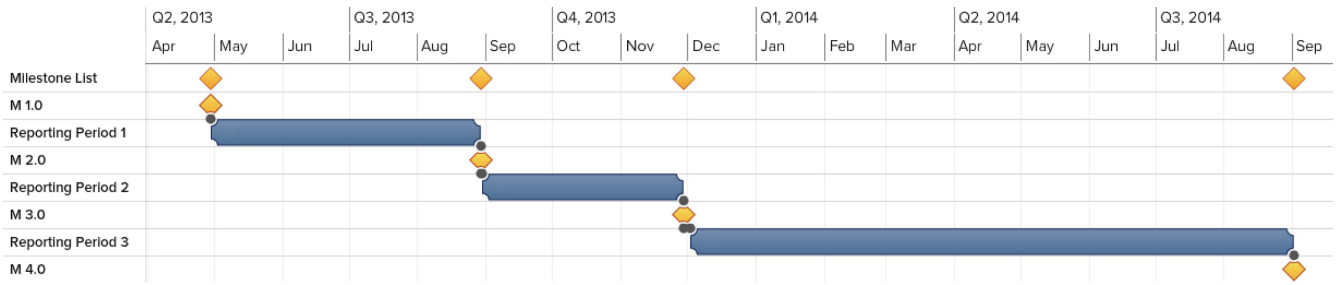


Figure 38: Milestone release schedule.

4.1. M1.0: First release of the PROSE platform

This initial milestone (M1.0) marks the first release of the platform to the public, enabling its core features for project hosting. The goal of this release is to make the primary features identified in D2.1 (platform survey) available to the ICT community.

- Source code hosting using Git and Subversion

- Hosting for both private and public projects within the platform
- Support for the creation on nested projects (sub-projects)
- Functional User interface

The release is also being adequately monitored in the platform's issue tracker[20] under project “opensourceprojects.eu”.

Target Date: May 2013. The release target for M1.0 is roughly aligned the delivery of this document, marking the initial availability of the information that must be provided to ICT projects, as well as with one of the first PROSE dissemination events at the Future Internet Assembly in Dublin[21].

4.2. M2.0: Second Tier feature set

This milestone increments the previous milestone with features that while not being critically important to the initial launch of the platform, were either identified as being significant or were still under revision at the time of the initial deployment.

- Support for other access mechanisms for source code repositories (e.g. SSH)
- SourceForge neighbourhood integration
- UI refinement
- Bitergia statistics integration (depending on upstream Allura project)

Target Date: September 2013. This release date is aligned with the final reporting period for year 1 and closes the platform's main feature set.

4.3. M3.0: Functional External API

As outlined before, it is important for the platform to provide meaningful information regarding the main tools. This will enable external applications to collected statistics and analyse the information on the platform, which can be useful for other ICT projects and for the EC. This milestone places the focus in making the platform statistics and information available to external parties through the use of a stable API.

The main features include (depending on availability):

- Issue information
- Collaboration information logs (Mailing list, Wiki or blog)
- Code-related information (e.g. commits, source code)
- User and project activity streams

20 http://opensourceprojects.eu/p/osp/tickets/
21 http://www.fi-dublin.eu/

This information will be based on the publicly available (FLOSS) projects, and privately for projects that opt-in and provide the necessary information.

Target Date: December 2013. This Milestone appears early on year 2 to allow time for projects and external entities to build tools that can take advantage of the PROSE collected information set.

4.4. M4.0: Beyond PROSE

This is the milestone where we activate features from 3rd parties, considering the possible contributions that can exist from other projects, and the existing tools for quality assessment (and statistics generation). These contributions may originate in the upstream Allura platform development project and integrated into the PROSE deployment, or alternatively from the collaboration with other ICT projects.

- Integration of quality assessment tools (from Allura or ICT projects)
- Integration of statistics generation tools (from Allura or ICT projects)
- Integration of 3rd party reporting tools for generating consistent project feedback.

Target Date: September 2014. This milestone release is slated to match the final reporting period, since it will likely require longer iteration as it includes external collaboration with other projects.

5. Conclusions

One of the main goals for the PROSE project is to provide a FLOSS project platform that can be adopted within the ICT community. This implied a public consultation in D2.1 and led to the several key decisions concerning the type of platform that would be deployed by the project, as well as the type of deployment that should be executed to best fit the requirements at hand. The final project decision defined running a private Allura instance – the software platform backing SourceForge website – to fit the two-fold goal of taking advantage of a well-known platform that is made available as FLOSS. It also has the benefit of a flexible platform that can include cutting-edge features developed at the Apache Software Foundation, as well as private repositories, which are among the top requirements identified by the project.

In this deliverable we presented the deployment of the main PROSE self-hosted Allura platform at ***opensourceprojects.eu***. Presently the platform deployment and source code is maintained as a FLOSS project in the platform itself, already taking advantage of the offered features and providing a very pragmatic use-case about the overall platform's goal. All of the deployment issues are being handled directly through internal tickets opened by the project participants to help refine the platforms functionality, and more importantly the user-interface, which is crucial for the success of the platform.

The highlighted platform issues deal mostly with configuration problems relating to the deployment (e.g. databases, permissions, email servers, auxiliary platforms), and specially with the UI. In fact, one of the main hurdles for adopting a self-hosted Allura instance relates with the fact that even though the back end source code is flexible, the frontend (UI) in most cases was very tied to its SourceForge roots, or non-existent in many cases. This important step consumed more than the expected resources, but is now in the processed of stabilising as the platform takes shape as identified by the many practical examples illustrated by figures throughout Section 3.

The aforementioned issues were defined as the first milestone to achieve on the platform deployment, taking care of the initial usability of the platform so that projects can start taking advantage of the platform's features for software development. This approach is aligned with the PROSE overall strategy which aims to first get projects on board, making sure that the tools and features available are sufficient to get them engaged with the platform, and then outlining several possibilities towards releasing the developed software as FLOSS and providing the tools and information to do it. This second stage, along with the refinement of the platform, composed the following milestones that we aim to provide as part of the project's roadmap. In the meantime, as a few changes may be considered as the platform evolves, deployment and integration experiments will be carried out on a staging instance in order to ensure that projects are not disrupted in the process.

It is important to mention that in parallel to the self-host solution we are setting up a mirror at SourceForge. It will act as a Neighbourhood that will mimic the project's projects, which will be used to increment projects' visibility and to serve downloads taking advantage of SourceForge mirror network. This will contribute directly to the success of PROSE and the

hosted projects, who will be able to take advantage of a world-wide distribution platform.

As demonstrated by the the presented features, the project strived to match the needs and requirements identified in D2.1 (E.g. private projects, that were initially out of scope), which created an overhead in terms of accommodating all of the necessary features. However, the effort is important to eliminate potential barriers that can be detrimental in the adoption of the platform, which in the long run would reduce the possibilities of generating FLOSS.

As the platform nears its public release state, the information contained in this deliverable will be critical towards the training material for WP3 and for the dissemination information provided by WP4. The next steps will be to start generating the required documentation for both training and dissemination. There is also a need to start focusing on providing paths towards open-sourcing the software hosted on the platform by giving access to the information that will assist projects with the process, an input that is currently being pursued in WP3, especially in tasks 3.2 and 3.3.

References

[1] Apache Allura, URL: http://incubator.apache.org/allura/
[2] “Third Party Data Migration”, Allura Project. URL: http://allura.sourceforge.net/migration.html
[3] PROSE Deliverable D2.1 Project Hosting Service, 2013
[4] “ICT PROSE Survey of Forge Platform Requirements”. URL: http://www.ict-prose.eu/surveys/index.php/survey/index/sid/474958