CatIS Administration Guide

Version 24.2

Table of Contents

About this Guide

This guide is meant for users, who will define and manage the application forms, data structures, workflow process rules, presentation of information, etc.

Chapter 1: Users, accounts and user rights

After installing a new catalogue environment there are no users defined in the system. To add the first administrator user, navigate to the Log In page and click "Sign up now". On the self-registration page create a new user. This very first user will automatically be set as application administrator and depending on the system configuration (see separate Installation and Configuration Guide for details), the self-registration could be disabled from there on. After the first admin logs in, he can add new users in Administration->Users and designate them as application administrators if needed, or if the self-registration is not disabled, new users can register accounts themselves.

Application administrators

A user can be designated as application administrator by checking the checkbox in the user's profile (only existing administrators can change this setting).

The application administrators have access to all parts of the application and edit rights to all objects, therefore this level of access should only be granted to as few people as possible.

Only application administrators can modify forms, views and workflows, designate other users as application admins, or edit and delete user accounts.

Also, only application administrators can create new institutions and delete them.

Institution users

When a new institution has been created in the catalogue, it is then possible to register new user accounts for that institution's users. When the application admin registers a new institution (either via registration request or directly), the first user for that institution should also be registered (or he/she could already be self-registered) and assigned to this institution as institution's admin / responsible user. This is done on the "User Accounts" tab of the institution page. This user can later add other user accounts for the institution and possibly mark them as institution admins as well.

Users can be added to institution by creating a completely new user or by assigning an existing user as this institution's user. A user can simultaneously belong to more than one institution.

All users created from the institution "User Accounts" page are automatically related to that institution.

Owner roles

Each object in the catalogue can have one or more owners. Owners have edit rights for the object (in addition, application administrators have edit rights to all objects). Institution owners can be specified on the institution "User Accounts" page, clicking on the "Edit owner access" button. Select the institution users that should be able to edit institution data and click "Save changes".

All other objects (information systems, services and assets) should have owners assigned too. Owners can be selected from the institution users on the Owners tab on the object's page. Only users assigned as owners can edit object's data (as well as application administrators).

Chapter 2: Defining forms

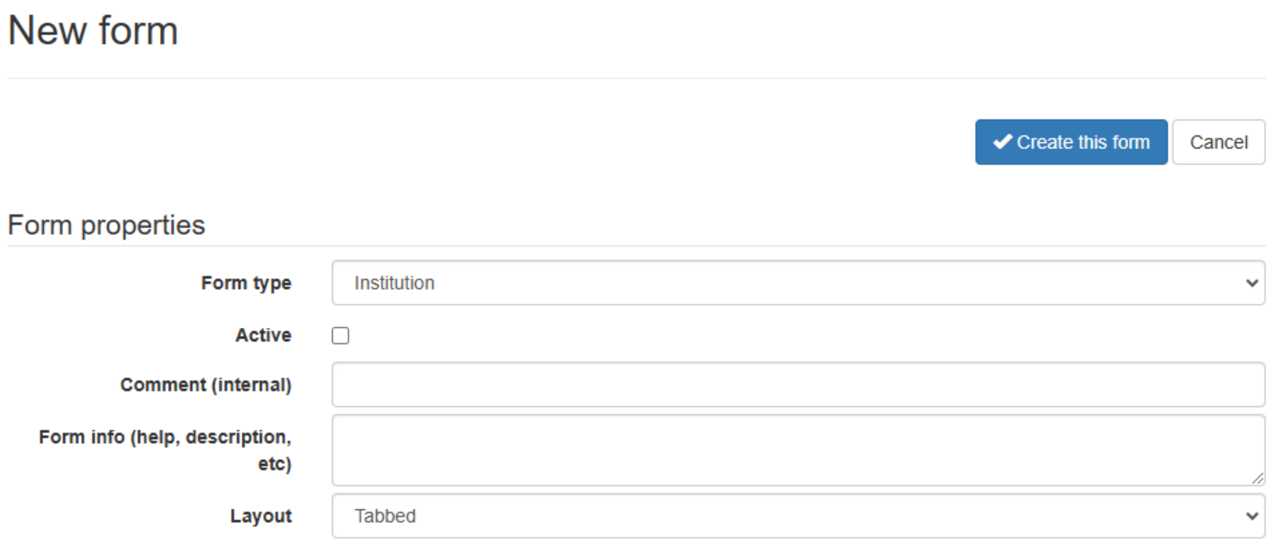
One of the first things to do in a new application installation is to define the data structures of the application objects (institutions, systems, assets, etc).

Form types

There are 8 main object types in the catalogue: Institutions, Systems, Services, Public Services, Assets, Registration Requests, Whole of Government and Enabler. For each object type the catalogue holds certain data about that object, but the exact data structure is configurable by the catalogue administrators.

What objet types are used can be configured in the application configuration file and object types that are not necessary can be disabled to that they will not show in the user interface.

After a new catalogue installation, application admins must define the data structure and input forms for the necessary object types. To define the forms, log in as an administrator and navigate to Administration->Forms. If there are no forms listed yet or if not all types are visible, create a new form definition by clicking "Add new form". Select a form type and enter some descriptive information about the form. This information will be displayed to the users when they edit the respective object data, so it should describe the main purpose of the object, could contain helpful hints, etc.



New form

Form properties

Form type	Institution
Active	□
Comment (internal)	
Form info (help, description, etc)	
Layout	Tabbed

You can create many forms of the same type, for example when preparing a new form while the old version is still in use or to try out different combinations, etc. You decide which of the forms from the same type is actually used by marking that form as active. When creating multiple forms of the same type, use the Comment field to describe the purpose or version of each form so you can tell them apart later.

Make sure that there is at least one form defined for each object type, 6 forms in total, add any that are missing. One (and only one) of the forms from each type should be set as active.

Form layout

It's possible to present for data in two differnet ways, controlled by the Layout selection

When layout is "Tabbed" (default) the sections are displayed one-at-a-time, user can select which section to view from the horizontal list of section headigs.

When layout is "Linear", all sections are displayed at once, user can scroll down to particular section or quickly jump to a section from the menu on the left of the form.

Default fields

Most of the data is freely configurable for each object type, however certain fields are always present on the final forms and should not be defined separately. These are "name" and "indentifier". Name is the name of the object, e.g. institution name, identifier is for example registration code or a short name, etc.

Form structure

All forms are defined in the same way and consist of sections, field groups and fields. At the top level the form contains one or more sections, each section contains one or more field groups, each field group contains one or more fields.

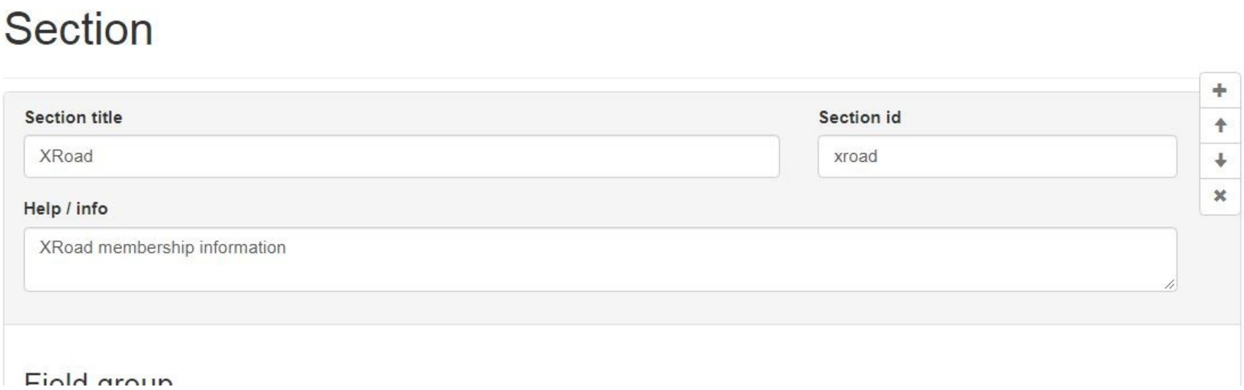
Help/Info texts for sections, fieldgroups and fields

When entering help / informative texts for sections, fieldgroups and fields, only the first paragraph of the help text is initially displayed on the resulting forms. If you enter multiple paragraphs of help text, the subsequent paragraphs are initially hidden and a "show more" link is displayed to the user. When the user clicks on the link, the rest of the text is then displayed.

Enter short descriptive text as a first paragraph, that should just remind the user of the purpose of the field and take no more that one line on the forms. Then on next paragraphs you can include longer explanations, examples, etc for first-time users.

Sections

Sections define the main sections of the resulting form. When creating a new form, by default one empty section definition is displayed. To add new sections, click on the "Add new section" button at the end of the page (hover mouse over the "+" and other buttons to view the button description) or if you want to insert a new section before the current one, the "Insert new section" at the beginning of each section definition. You can also rearrange sections by clicking "Move section up" or "Move section down". The buttons are on the right side of the form.



Section

Section title: XRoad

Section id: xroad

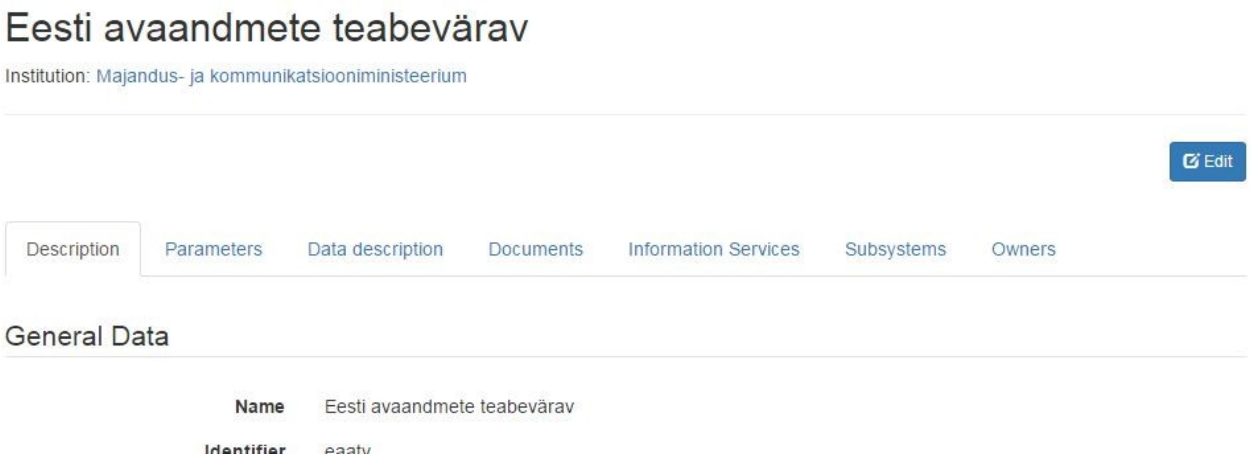
Help / info: XRoad membership information

`Section title:` the title of the section as displayed to the users.

`Section id`: internal identifier of the data section. It is best to use short but descriptive names, e.g. "contacts", "documents", etc. NB! Changing the identifiers later when there is already data in the catalogue will result in the old data no longed being displayed!

`Help / info`: any helpful information you want to be displayed to the users when they are filling out the form, e.g. purpose of the section or any special rules for inputting data, etc.

Later when the users of the catalogue view information about the object (e.g. Institution), the sections are displayed as tabs in the tabbed layout, for example the following picture shows sections "Description", "Parameters", "Data description", etc.



Eesti avaandmete teabevärav

Institution: Majandus- ja kommunikatsiooniministeerium

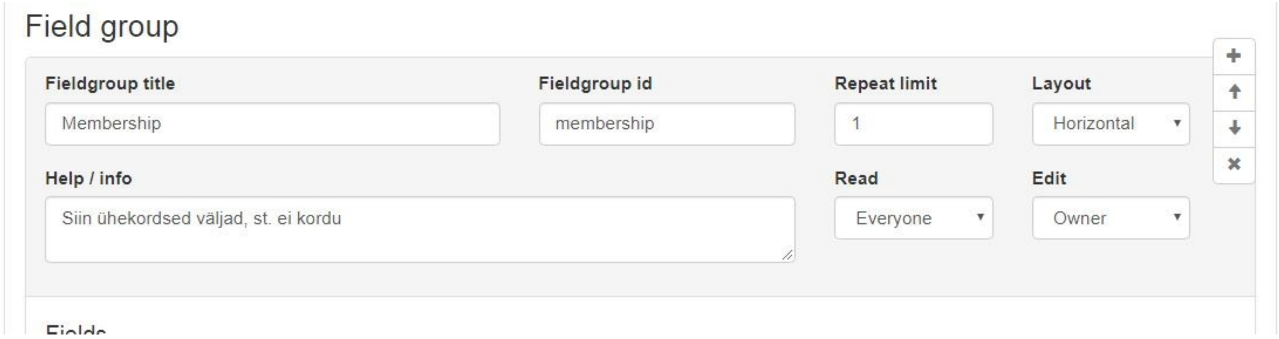
General Data

Name Eesti avaandmete teabevärav

Identifier eaatv

Field groups

Each section consists of one or more field groups. Field groups organize and group related data and can have different visual representations to present the data fields in most meaningful way, for example field group "Contacts" could group fields "First name", "Last name", "E-mail" and "Phone" and present them later in a table layout.



Field group

Fieldgroup title: Membership

Fieldgroup id: membership

Repeat limit: 1

Layout: Horizontal

Help / info: Siin ühekordsed väljad, st. ei kordu

Read: Everyone

Edit: Owner

Fields

`Fieldgroup title`: Visible name for the field group, for example "Contact Persons".

`Fieldgroup id`: the internal identifier of the field group. It is best to use short descriptive names, for example "contacts". Like section id, changing this later can result in data loss.

Help / info: any helpful information you want to be displayed to the users when they are filling out the form

Repeat limit: field group content can be repeated when filling out the forms, for example users can add more than one contact person for an institution. The "Repeat limit" setting controls this as follows: if the setting is 0, there is no limit, fields are repeatable as many times as needed. If the setting is 1, the fields are not repeatable (can only appear once). Any other number represents the number of times the grup of fields can be repeated, for example you can specify that only up to five contact persons can be entered by setting this to 5.

Layout: How the group of fields should be displayed to the users. The options are "Vertical", "Horizontal" and "Inline". The vertical layout has field labels on top of the fields and is best for long fields. Horizontal layout has field labels in front of the fields and is better for compact forms.

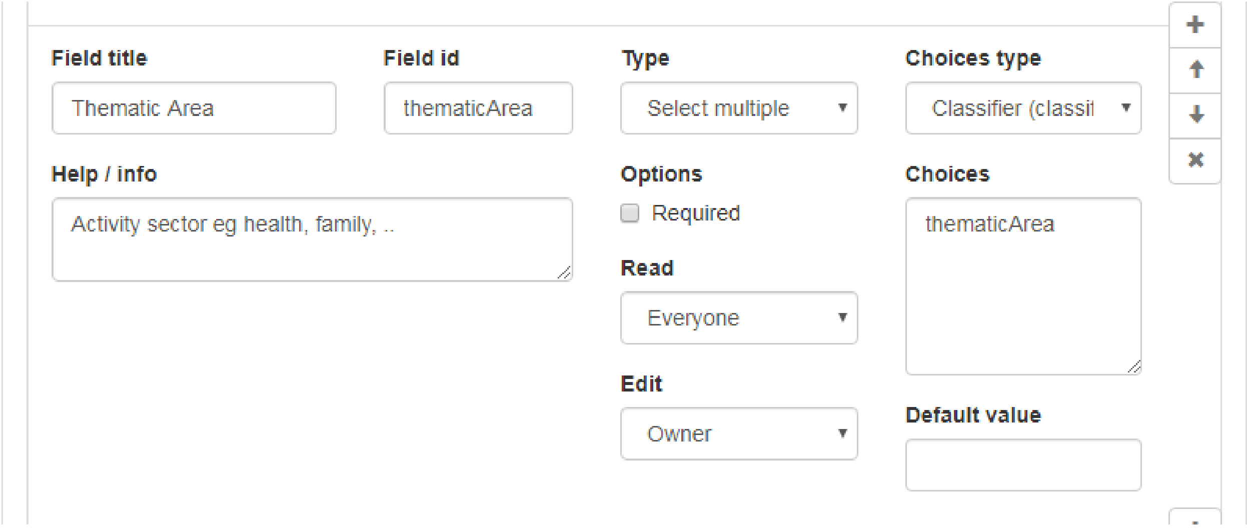
Both horizontal and vertical layouts repeat the group of fields as a whole with all labels and fields. In contrast, "Inline" layout displays the data as a table, with labels as column headers and each group of fields as a row in the table.

Read: controls who can see this field group. The options are Everyone (no restrictions), Authenticated (not publicly visible, only visible to logged in users), Same Organization (the field group is only visible to users from the same organization that the object belongs to), Admin Only (only CatIS administrators can see this field group). Define the visibility according to the sensitivity of the data, for example contact information and email addresses should not be publicly visible on the Internet, etc. CatIS administrators can always see all field groups irrespective of this setting.

Edit: controls who can edit data in this fieldgroup. Options are Everyone (everyone, even anonymous users, can edit data), Owner (only users marked as this object's owner can edit data) and Admin Only (only CatIS administrators can edit). The Everyone option is most useful on new registration request forms, where the user is not yet authenticated. For other forms the default should be Owner, but some data could be restricted to CatIS admins only. CatIS administrators can always edit any field group irrespective of this setting.

Fields

Field groups contain fields. Each field is defined by several attributes, fields can be added, inserted, moved around just like sections and field groups.



Field title: Thematic Area

Field id: thematicArea

Type: Select multiple

Choices type: Classifier (classif

Help / info: Activity sector eg health, family, ..

Options: □ Required

Read: Everyone

Edit: Owner

Choices: thematicArea

Default value:

Field title: Visible name for the field, for example "First Name".

Field Id: the internal identifier if the field. It is best to use short descriptive names, for example "firstname". Like other id's, changing this later can result in missing data. The field id's don't need to be unique within the whole form, but they must be unique within the same field group.

Help / info: any helpful information you want to be displayed to the users when they are filling out the form

Type: field type. The available types for the fields are:

Text - simple one-line text input

Number - shorter input field, validates numerical format, has spinner controls

Date - shorter input field, validates date format. Has a popup calendar to choose a date from.

Email - validates email format, in view mode presents as emails links

URL - in view mode presents input as URL link

Textarea - multiline text box for longer text. Textarea fields do support Markdown (https://en.wikipedia.org/wiki/Markdown) formatting if you need to use it, but they don't use a WYSIWYG editor.

Markdown - larger multiline text box for formatted text. Markdown fields use a WYSIWYG editor when editing form data.

Boolean - one checkbox with true/false value

Checkbox - one or more checkboxes with textual values

Select one - input list for selecting a single value

Select multiple input list for selecting one or more values

File - file upload

Asset - input list to select an existing asset from the catalogue

Service - input list to select an existing service from the catalogue

Public service - input list to select an existing public service from the catalogue

System - input list to select an existing information system from the catalogue

Institution - input list to select an existing institution from the catalogue

Choices type: for field types that have choices (checkbox, selects), select if the possible values come from a classifier or are entered directly on the form definition.

Choices: if the choices type field is "Enter values", enter possible choices here, each entry on a new line. If the choices type field is "Classifier", enter the classifier lookup key here. See next chapter about defining classifiers.

Default value: if the field should have a default value on new objects, enter the value here. If the field is a select field, the value should match one of the possible choices for this field.

Options:

- Required: if the field is required to be filled out (cannot be left empty)

Read: define who can see this field. See the description of the fieldgroup above for possible values. Use if you only want to protect some fields from the field group. Note that if the read access on the containing field group is more restrictive, it already hides all the fields that belong to the field group and setting this per single field has then no effect.

Edit: define who can modify this field. See the description of the fieldgroup above for possible values. Use if you only want to protect some fields from the field group. Note that if the edit access on the containing field group is more restrictive, it already disables editing of all the fields that belong to the field group and setting this per single field has then no effect.

Computed default values

For the "Default value" attribute in field definition, in addition to static values it is also possible to use values that are computed when the new object is created with that form. To use computed values, enter one of the following tags as the "Default value" attribute:

`{{date}}` – computes to current date

`{{datetime}}` – computes to current date and time

`{{day}}` – day of month

`{{month}}` – month of year

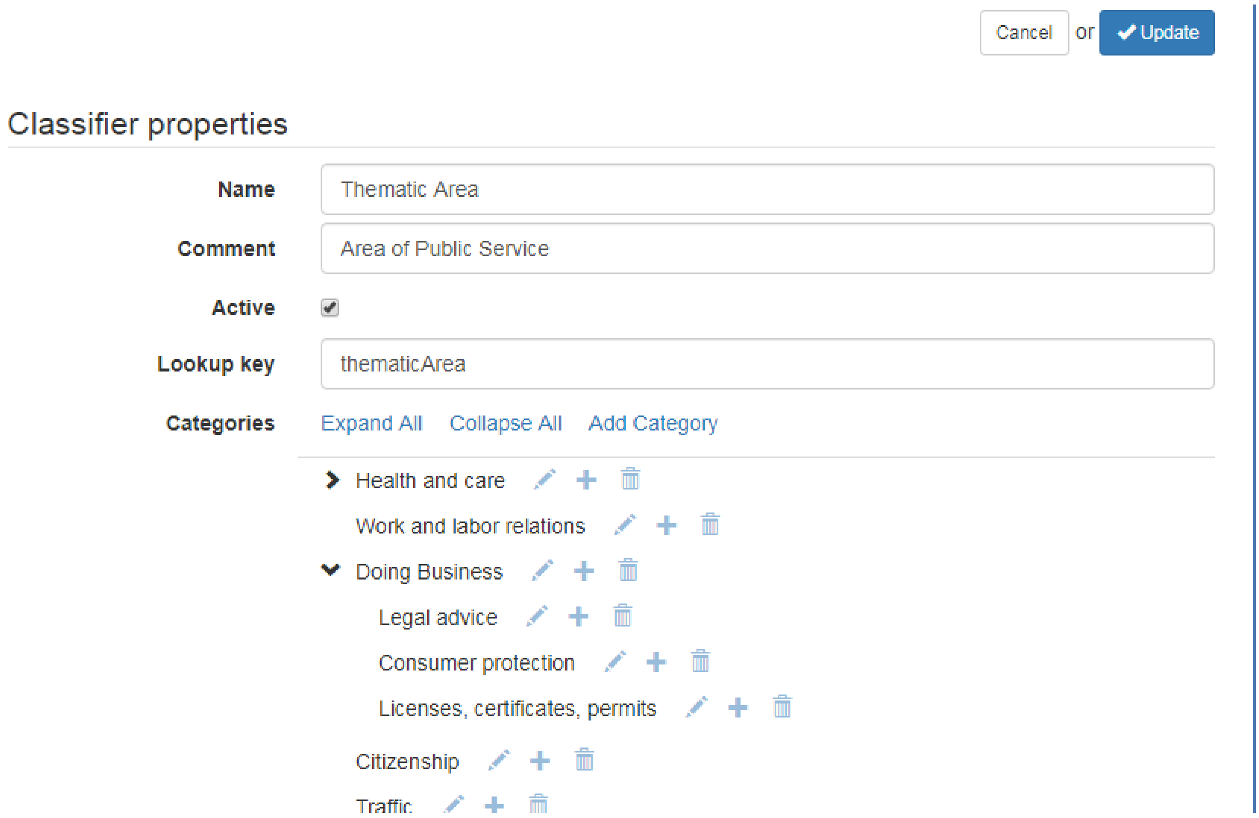
`{{year}}` – current year

`{{username}}` – current user name

Chapter 3. Defining classifiers

Classifiers are used to provide controlled lists of values for selection fields. Instead of entering all the possible values for form field option lists, it is preferred to define the more commonly used categories as classifiers and link the form fields to such classifiers. In this way similar fields on many forms can use the same set of values and when the list of values changes, there is no need to modify all the forms that may use these lists.

To define classifiers, navigate to Administration –> Classifiers, where you can add new classifiers and modify existing ones.



Classifier properties

Name: Thematic Area

Comment: Area of Public Service

Active: ☑

Lookup key: thematicArea

Categories:

- Health and care
- Work and labor relations
- Doing Business
  - Legal advice
  - Consumer protection
  - Licenses, certificates, permits
- Citizenship
- Traffic

Classifier name is a descriptive name of the classifier. This name is not visible to the users, only for admins to find the right classifier.

Comment is an internal comment for this classifier, e.g. when you are working with many classifiers with similar names or with another version of the same classifer, you can use the comment field to distinguish between them.

Active should be checked, if this classifier is to be usable on the forms.

Lookup key is a short name for this classifier that is used to link form field to the classifier (see the previous chapter). It should be unique, short and descriptive, without spaces, dots or any fancy characters.

Categories section shows all the defined values for this classifier. Classifiers can be hierarchical, with categories, sub-categories, sub-sub-categories and so on. To expand or collapse the category tree, you can click on Expand All or Collapse All, to add a new value to the end of the list, click Add Category.

To edit existing value, click on the pencil icon next to the category name, to add another category immediately after a category, click the + icon, to delete a category, click the trashcan icon.

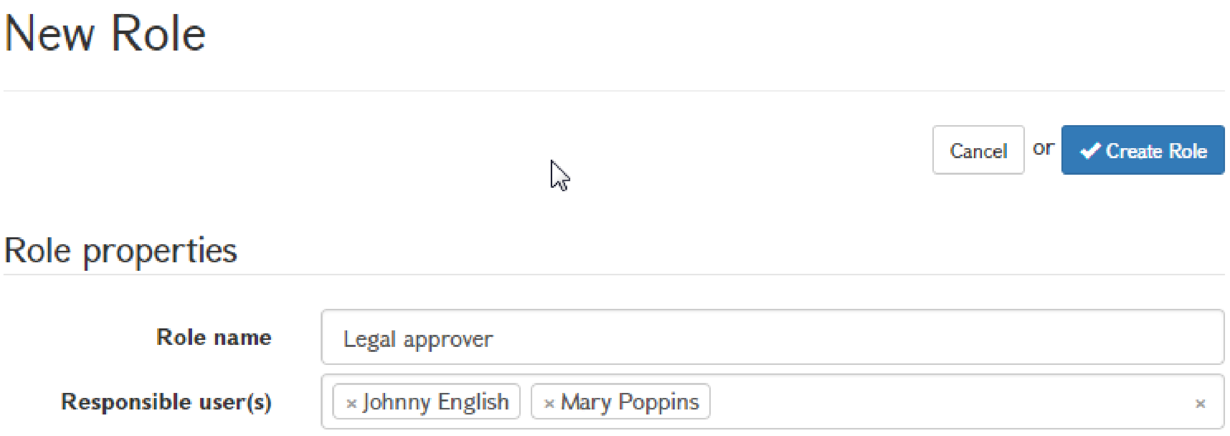
Categories can be rearranged simply by dragging them with the mouse. Drag a category over another category to make it a sub-category. If you create hierarchical classifier (with sub-categories), the fields that use this classifier will also present it as a hierarchical selection.

When done, click Update.

Chapter 4. User roles

Roles are used in workflows to connect a workflow role to a specific person or persons. For example, you could define a role "Legal reviewer", assign it to a certain person and use that role in multiple workflows. When the person leaves or another person takes the responsibility, you can just change the associated person in that role and all workflows continue working, without the need to update all the workflow definitions where that person might have been included.

To define roles, navigate to Administration –> User Roles and click on Add new role. Fill in the role name and select one or more responsible users.


New Role

Role properties

Role name: Legal approver

Responsible user(s): × Johnny English × Mary Poppins ×

A role can have more than one responsible users and a person can belong to any number of roles.

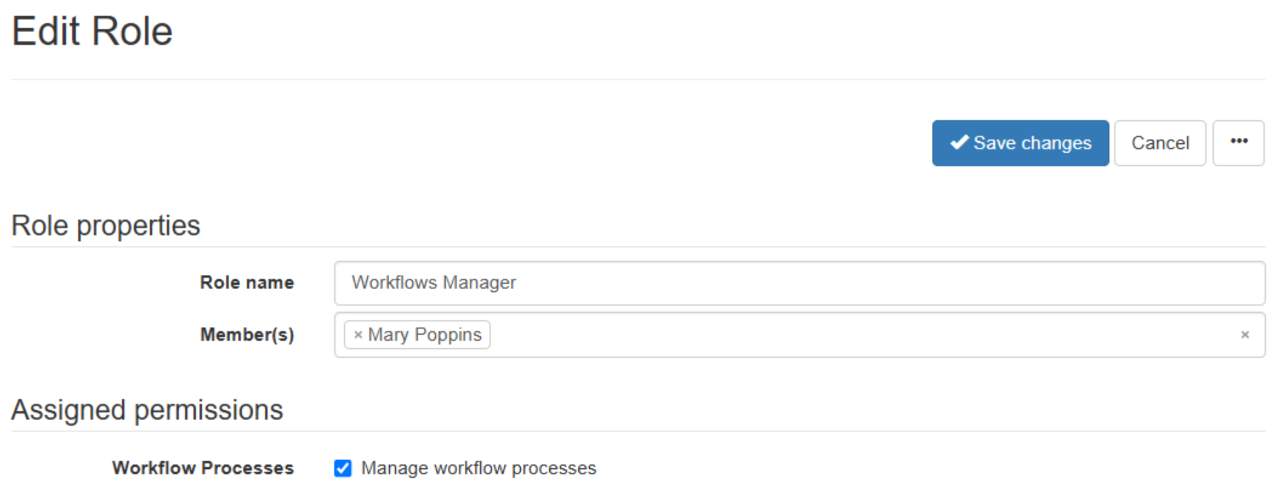
When the role is later used in the workflow, any of the listed users can perform that role, so that when one person is unavailable, another can step in.

Role name can be translated to different languages, see Chapter 8 Languages and Translations.

Chapter 5. Role permissions

Special permissions can be assigned to roles, depending on the requirements. These permissions allow some users that are assigned to the role to perform some operations in the system, even if they are not given access by "normal" means. For example, the permission "Manage workflow processes" allows to perform workflow decisions in all workflows on behalf of the regular user, if the normally assigned user is for some reason not able to perform the action themselves.

New permissions will be added to the system as required.



Edit Role

Role properties

Role name: Workflows Manager

Member(s): × Mary Poppins ×

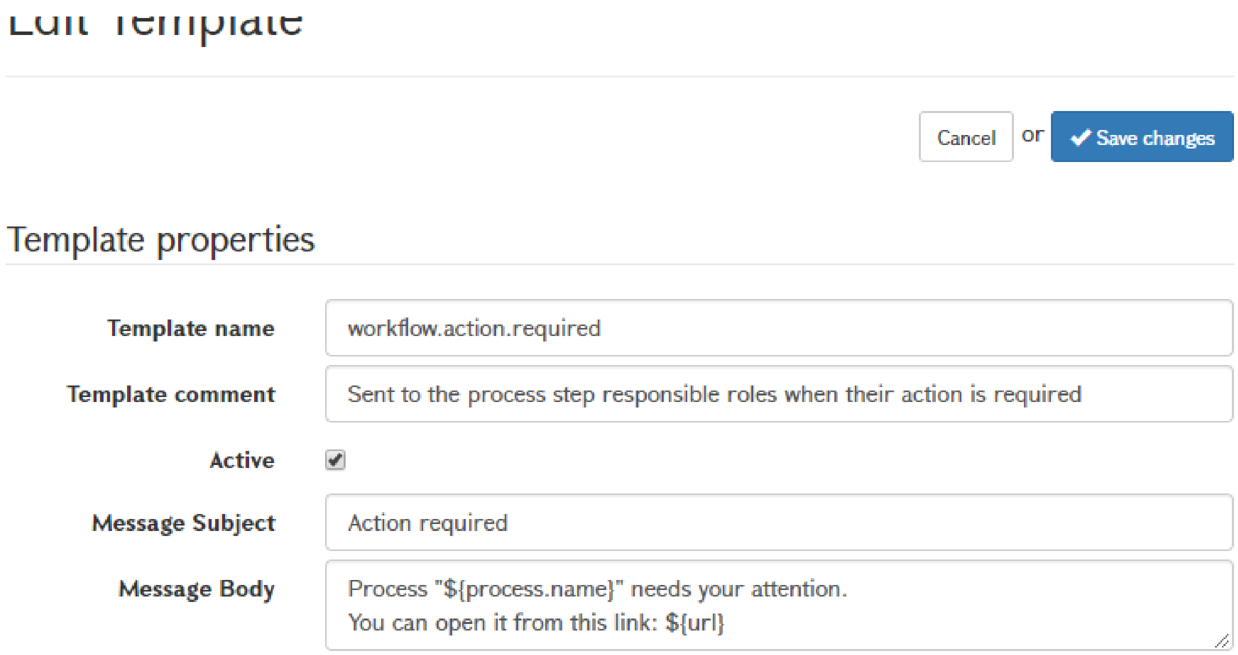
Assigned permissions

Workflow Processes: ☑ Manage workflow processes

Chapter 6. Email templates

During the workflow process, the application notifies the responsible users of the required actions by sending e-mail messages. The templates of those messages should be defined before the messages can be sent (when there is no template for some messages, no error is generated and the message is just ignored).

To define the message templates, navigate to Administration -> Email templates



Edit template

Template properties

Template name	workflow.action.required
Template comment	Sent to the process step responsible roles when their action is required
Active	☑
Message Subject	Action required
Message Body	Process "${process.name}" needs your attention. You can open it from this link: ${url}

Template name is used as an identification key within the application and is not visible to users. It should be unique and concise.

Template comment is for the admins to keep track of what template is meant for what use, etc.

Active should be checked, if this template is to be used in the workflows. You deactivate a template (and therefore that message) without actually deleting it or you can keep several templates with the same key in the application, for example when working on different template versions, but at most one of the templates with the same key (template name) should be marked active at the same time.

Message Subject is the subject line of the message and Message Body is the message itself.

Email template Subject and Body texts can be translated to different languages, see Chapter 8 Languages and Translations. The workflow notifications are automatically sent according to the user's preferred language (set in the user profile), if the language version exists, othervise the default language template is used.

Variable substitution

You can use certain process variables in the email message Subject and Body texts. The format of the variables is ${<variable name>}, where <variable name> can be a process property, a related object property (a related object is the "parent" object of the workflow process instance, e.g. and institution or an asset) or a literal name "url".

The properties of the workflow process are prefixed with the word "process" and the properties of the related object with the word "object".

The properties of the process are:

process.name – name of the process, from workflow definition

process.info – process description from the workflow definition

process.createComment – comment or additional information provided by the person who started the process

The properties of the object are:

object.name – the name of the obejct, e.g. institution name

object.<section>.<fieldgroup>.<field> - fields as defined on the corresponding object form. NB! Care must be taken when using the fields in such way. If the field does not exist (e.g. the form is changed sometime later), the template must be changed also.

The URL to the obejct can be included in the message using the variable ${url}. The configuration property "play.http.urlprefix" (see the Installation and Configuration Guide) must point to the proper scheme, server and port for the links to work.

Chapter 7. Defining workflows

To support various business processes the application incorporates role-based workflow functionality. Workflow processes can be started automatically (e.g. somebody creates a new institution registration request, a review process could be started behind the scenes to review and accept/decline the application) or manually by selecting a process from a list and starting it (e.g. submit an application to connect an information system to the information exchange infrastucture).

Filters are used in the workflow definition to restrict specific workflows to certain objects and their statuses, e.g. “Connect an IS to Xroad” process is only available on an IS form when the status of the IS is “Production”, etc.

Roles are used to support workflow processes, where steps in the process are assigned to roles instead of individual users. Roles are then mapped to users in a many-to-many relationship, i.e. each role could be fulfilled by more than one user and each user can have more than one role.

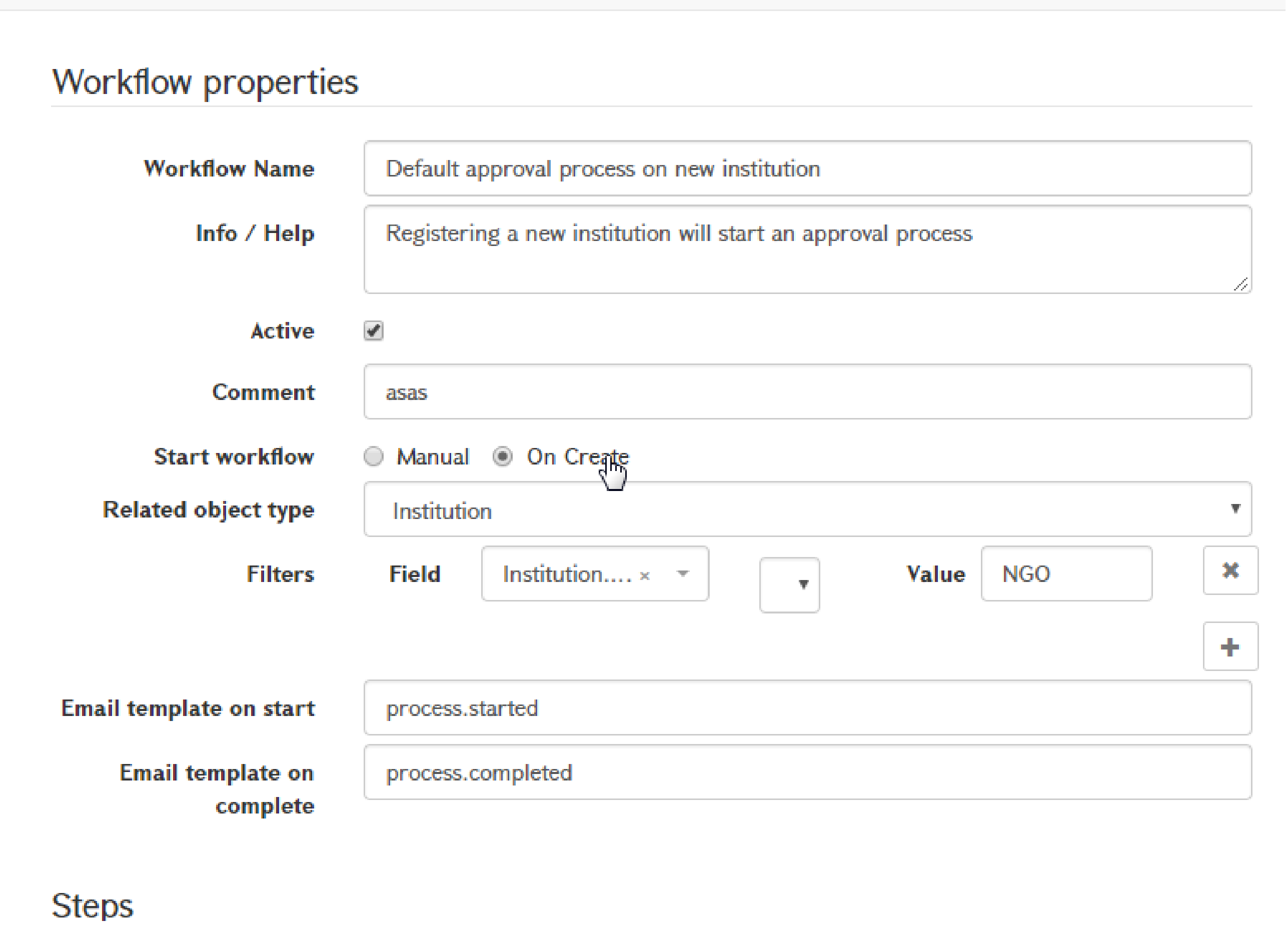
Workflow processes are composed of steps or states, where each step represents some action that has to be taken or a decision to be made, e.g. review or approval. Each step is assigned to some role who is responsible for that action. The system will automatically resolve roles to assigned users, send them email notifications and give them appropriate access to perform that step.

Steps can include time limits, the system will send email reminders to the repsonsible users when the action is not performed in the specified time.

Steps can be optional, depending on some attribute of the object the process is related to, e.g. if the IS contains sensitive data, additional approvals may be needed in such cases, but skipped if the IS does not contain any sensitive data.

Currently executing and past process results are visible on the respective form so that the decisions made and who made them and when can be verified at a later date.

To define workflow processes, go to Administration -> Workflow Definitions, click on "Add new Workflow Definition". A new workflow definition form opens:



Workflow properties

Workflow Name	Default approval process on new institution
Info / Help	Registering a new institution will start an approval process
Active	☑
Comment	asas
Start workflow	○ Manual ● On Create
Related object type	Institution
Filters	Field: Institution.... × Value: NGO
Email template on start	process.started
Email template on complete	process.completed

Steps

General properties of the workflow process

`Workflow name` – the name of the workflow process. This name is visible to the users, it should be short and descriptive.

`Info / Help` – description of the process or informational text for the users. It should clarify the purpose of the process.

`Active` – if this process can be selected/started

`Comment` – internal comment about the process. This comment is not visible to the users and is meant for the application admins to keep track of the processes.

`Start Workflow` – the workflow process can be started either manually selecting it from a list or automatically, when a new object is created.

`Related object type` – the type of the object, that this workflow applies to. The workflow is only visible (or can be automatically started) in the context of this object type, e.g. Institution, Asset, etc.

`Filters` – to restrict visibility of the workflows, filters can be added to the workflow definition. The workflow definition is only visible in the context of those concrete objects (that must be of correct type first), that match the filter(s). If more than one filter is added, the object must match all conditions.

- `field` – the field from the form definition, e.g. institution type

- `condition` – could be either "equals", "does not equal" or "matches regular expression"

- value – the value to test against, e.g. "NGO"

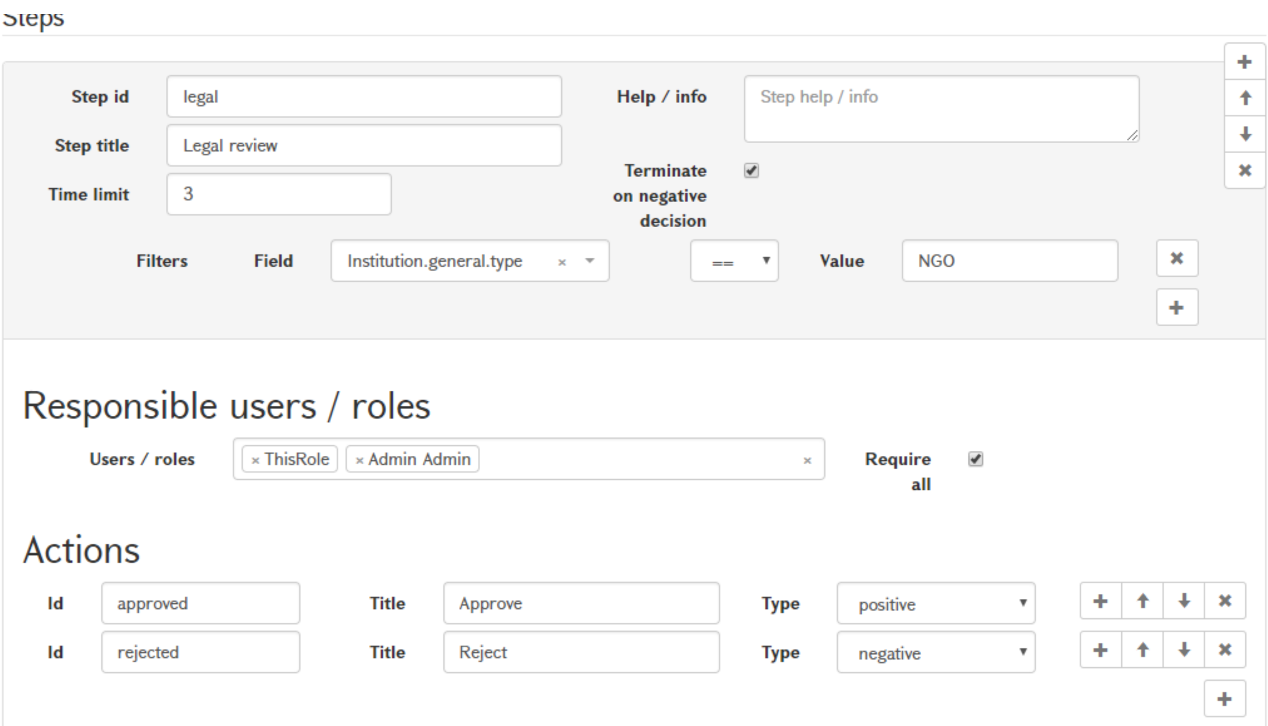
Email template on start – email template to use for messages that are sent when a new process is started (currently not used)

Email template on complete – email template to use for messages that are sent when the process completes. The messages are sent to the requester, i.e. the person who started the process. If this field is empty, the default template key "default.process.completed" is used. If the template with this key (or the default key) is not found, the message is not sent (no error is generated).

Email template on step – email template to use for messages sent to step responsible users. If this field is empty, the default template key "default.action.required" is used. If the template with this key (or the default key) is not found, the message is not sent (no error is generated).

Steps of the process

The workflow process consists of one or more steps. A step is one concrete task or decision performed by one or more roles or persons. Steps can be added, inserted, reordered and deleted using the buttons to the right of the step properties block.



Steps

Step id: legal

Step title: Legal review

Time limit: 3

Help / info: Step help / info

Terminate on negative decision: ☑

Filters — Field: Institution.general.type == Value: NGO

Responsible users / roles

Users / roles: ThisRole, Admin Admin

Require all: ☑

Actions

Id	Title	Type
approved	Approve	positive
rejected	Reject	negative

Each step has the following properties (some of them, like filters, are optional):

Step id – a unique (within the workflow definition) identifier

Step title – a user visible title or name of the step

Help / Info – a clarification of the step's purpose, to help responsible persons understand what they are required to do.

Time limit – time limit in days. If this many days has passed, but the step is not completed, the system will send out email notifications to the responsible persons. NB! This notification looks for an email template with the key "`default.action.reminder`", this is not currently configurable.

`Terminate on negative decision` – when checked, a negative decision in this step will complete the whole process. If unchecked, the decision is recorded, but the process will continue.

`Filters` – not required for simpler workflows, but in a more complext process some steps can be optional, depending on some field value of the related object. If the related object does not pass the filter(s), the step is skipped. A step can have one or more filters, if multiple filters are specified, all of them must match for the step to be included in the process.

- `field` – a field from the object form definition

- `condition` – a test condition, either "equals", "does not equal" or a regular expression for more complex conditions

- `value` – a value that the field should have (or not have) to pass the filter

`Responsible users/roles` – one or more roles or specific users who are responsible for performing the step actions (making a decision or doing something). A selection from all defined roles and users. If a role is selected, the persons responsible for that role are responsible for that step. If no suitable role is defined, specific persons could be selected also.

When the step is started, an email message is sent to the responsible persons automaticalli. This notification looks for the email template with the key specified in the workflow definition properties block "`Email template on next step`". If this field is empty, a default key "`default.action.required`" is used. This key is not currently configurable individually for each action.

If there are users assigned to a role that has the permission "Manage workflow processes", these users can perform the step on behalf of the responsible users/roles in all workflow processes.

`Require all` – if this is checked, all roles/persons selected in this field must perform their action before the step is completed. If unchecked, only one of them is sufficient for deciding the outcome of the step. NB. Each role is always performed by only one of the role responsibles, regardless of this setting.

`Actions` – this defines the possible actions that the responsible persons can take in this step. A step can have one or more actions (two is the most common). These actions are then presented to the responsbile persons when the process is executing.

- `id` – a unique (within the step) system id of the action

- `title` – a user visible title of the action. This is disaplyed in the possible actions list to the users

- `type` – type of the action, either positive, negative or neutral. For example in an approval process, a positive action marks approval, negative action marks rejection. In some processes, e.g. a FYI type of process, an action can be neutral, marking that the user has seen thedocument or has done what was requested, but he/she does not decide the the result of the process.

Actions can be added, inserted, deleted, reordered using the buttons to the right of the action.

Chapter 8: Registration forms and new registration requests

If enabled, external people can request that an institution be registered in CatIS by using the "Register new Institution" button in the Institutions view. When a new registration request is received in the system, an email notification is sent to the address specified with the `registration.sendto` parameter in `application.conf` configuration file (usually CatIS administrators). Then, CatIS administrator can find the new registration requests in the Administration-Registration Requests view, from where it is possible to review, modify, accept or deny the request. When accepting a request, a "Create new institution" form is opened automatically with predefined data from the request. An email notification about the decision is also sent out to the address given with the registration request.

Before submitting registration requests will be possible, there must be a form defined for these requests. It is best to define this registration form after the institution form has stabilized and use the same section/fieldgroup/field names as on the Institution form. That way, when CatIS admin approves the registration request and starts to create a new Institution in the system, all provided data can be carried over from the registration request.

There are 3 predefined fields on the registration forms, that cannot be removed and should not be duplicated: Institution name, registration code and contact email. Institution name and registration code are carried over to the Institution form when the request is accepted and the contact email address is used to send notifications about the request status or decisions made.

Also please note that the fieldgroups and fields that should be filled before submitting the request, should specify access right as "editable by everyone" in the form definition.

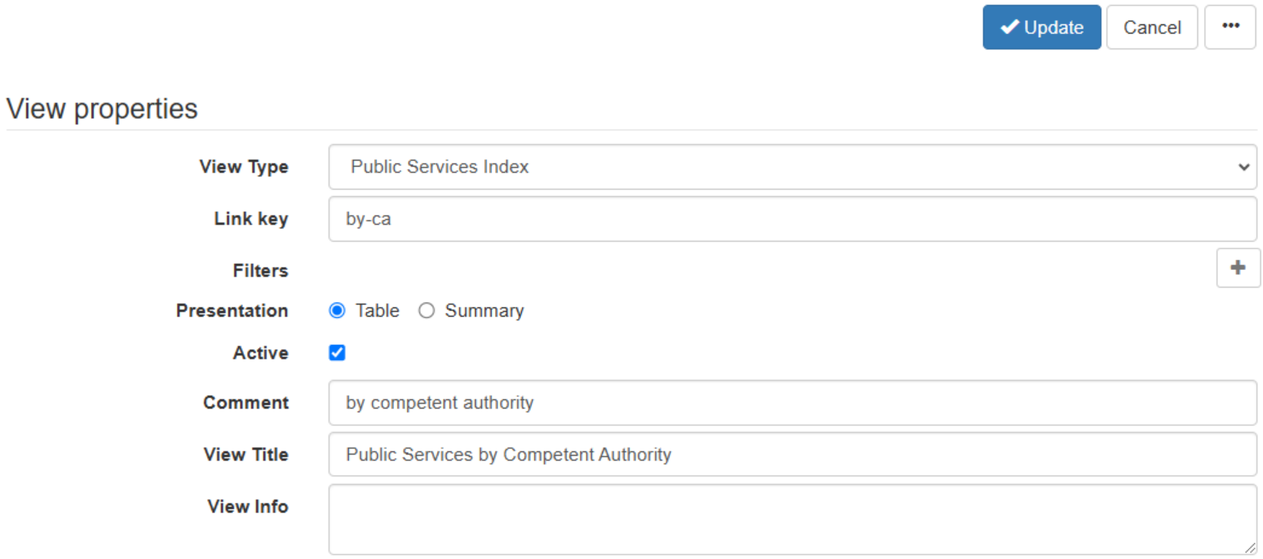
Creating new registration requests can be enabled for all users (including anonymous users), for only registered and authenticated users, or disabled. Two parameters in the application configuration file control what is enabled:

`authenticated.registration.enabled`: if this is set to "true", authenticated users can create new registration requests

`anonymous.registration.enabled`: if this is set to "true", anonymous users can create new registration requests

Chapter 9: Defining views

Views, or lists of objects (e.g. index view), can optionally be modified to change default columns and show most relevant data for each type of object. To define or modify views, log in as an administrator and navigate to Administration->Views. Create a new view definition by clicking on "Add new view", select a view type that you want to define, set the view attributes and choose the columns to display.



View properties

View Type	Public Services Index
Link key	by-ca
Filters	
Presentation	● Table ○ Summary
Active	☑
Comment	by competent authority
View Title	Public Services by Competent Authority
View Info	

A view definition has the following properties:

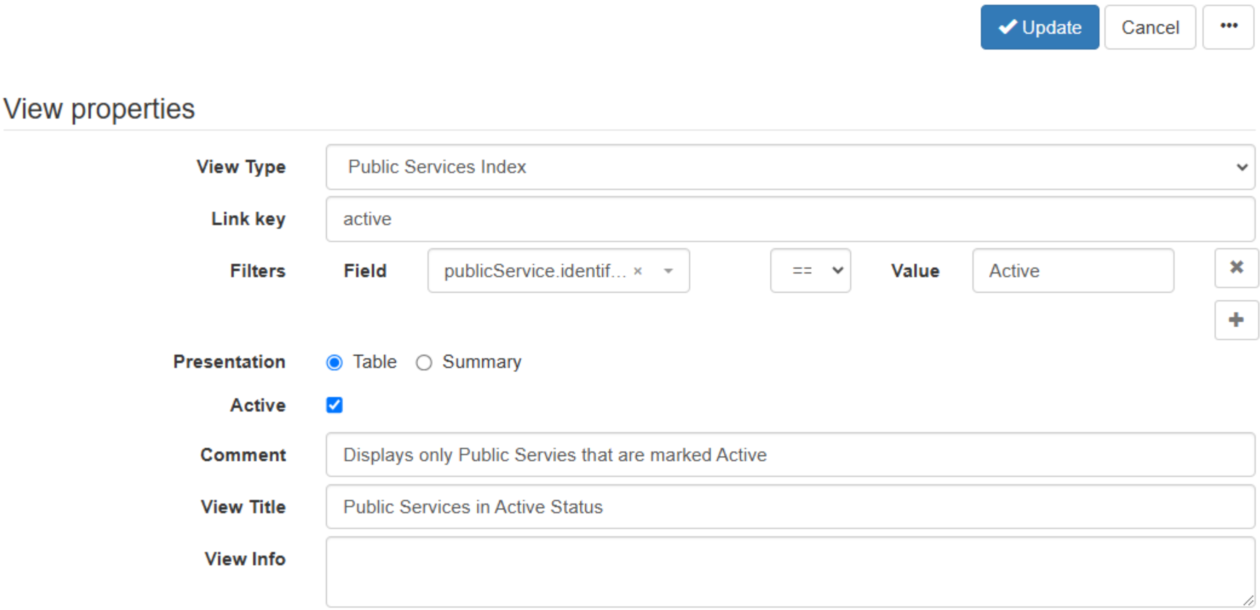
`View Type`: where in the application this view is located, e.g. Institutions Index is the main Institutions view, System – Subsystems is the subsystems list on the Subsystems tab on System form, etc.

`Link key`: if you need to define multiple active views of the same type, e.g. multiple public service index views, you must provide a "link key" for all but the default view. In that case the multiple views will appear as a dropdown selection in the main navigation bar and the link key is necessary to distinguish between the different views, or all the links would open one and the same view.

View Filters

Field-level filters can be added to the view to restrict view to only those objects that match the filter. For example, adding a filter by a status field with value "Active" to a public services view, the view will only display public services that are marked as Active.

Multiple filters can be added to a view definition, in this case all filters must match for the object to be displayed in the view.



View properties

View Type Public Services Index

Link key active

Filters **Field** publicService.identif... × == **Value** Active

Presentation ● Table ○ Summary

Active ☑

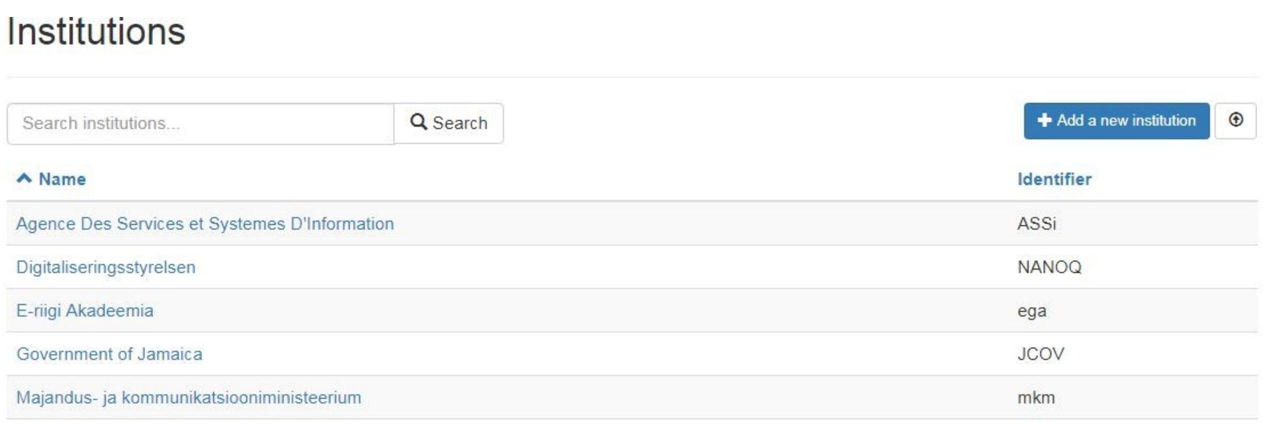
Comment Displays only Public Servies that are marked Active

View Title Public Services in Active Status

View Info

View Presentation

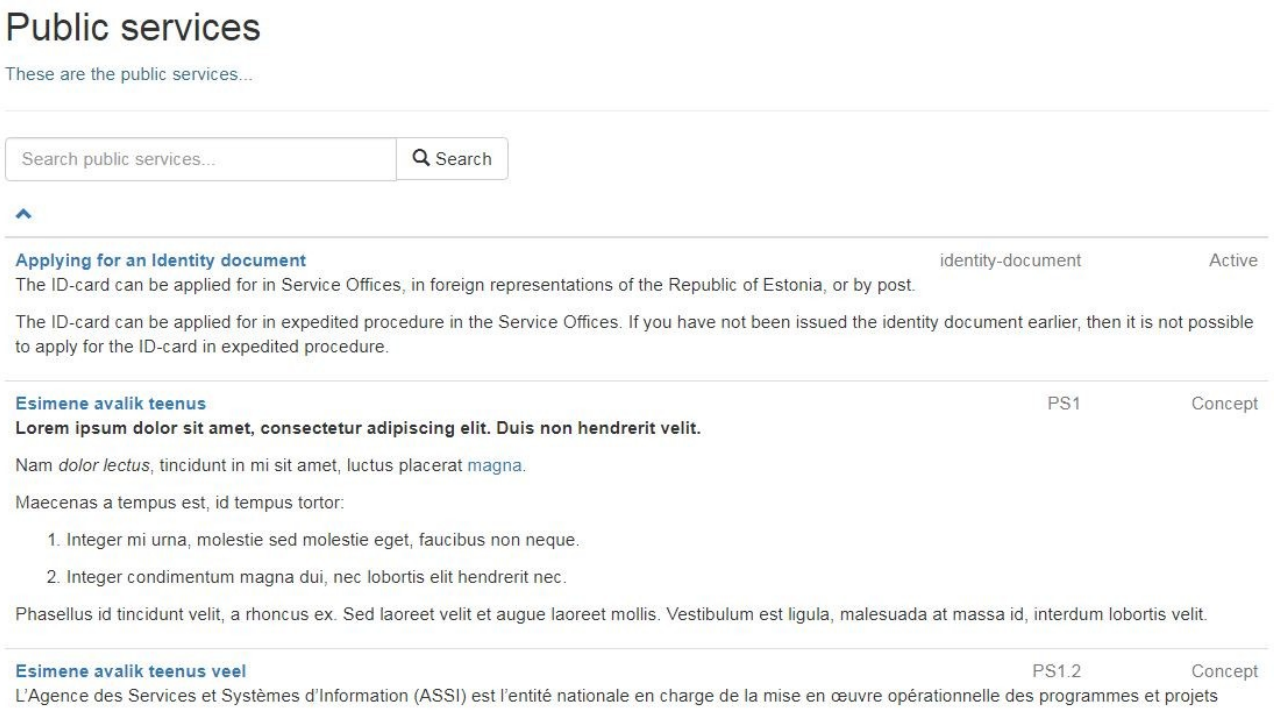
A view can be displayed in 2 different ways. One is a traditional table layout with columns side-by-side:



Institutions

^ Name	Identifier
Agence Des Services et Systemes D'Information	ASSi
Digitaliseringsstyrelsen	NANOQ
E-riigi Akadeemia	ega
Government of Jamaica	JCOV
Majandus- ja kommunikatsiooniministeerium	mkm

Another option is a summary view, that allows to display an overview or summary of each entry all on one page, in addition to main title and some secondary data.



Public services

These are the public services...

Applying for an Identity document identity-document Active

The ID-card can be applied for in Service Offices, in foreign representations of the Republic of Estonia, or by post.

The ID-card can be applied for in expedited procedure in the Service Offices. If you have not been issued the identity document earlier, then it is not possible to apply for the ID-card in expedited procedure.

Esimene avalik teenus PS1 Concept

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Duis non hendrerit velit.

Nam *dolor lectus*, tincidunt in mi sit amet, luctus placerat magna.

Maecenas a tempus est, id tempus tortor:

1. Integer mi urna, molestie sed molestie eget, faucibus non neque.
2. Integer condimentum magna dui, nec lobortis elit hendrerit nec.

Phasellus id tincidunt velit, a rhoncus ex. Sed laoreet velit et augue laoreet mollis. Vestibulum est ligula, malesuada at massa id, interdum lobortis velit.

Esimene avalik teenus veel PS1.2 Concept

L'Agence des Services et Systèmes d'Information (ASSI) est l'entité nationale en charge de la mise en œuvre opérationnelle des programmes et projets

`Active`: if checked, the view is in use by the application. To disable a view (e.g. work in progress), uncheck.

`Comment`: an internal comment for the application admins

`View Title`: the title of the view as displayed on the view page header.

`View Info`: An informative text that is displayed below the view title, should describe the view and what data is displayed, etc.

View columns

A view consists of columns. For each column you need to select source field and optionally enter the column titles. Table views usually benefit from having column titles, summary views often don't need them.

For views that are presented as a table, the columns are simply displayed side by side, from left to right, in the order they appear in the view definition.

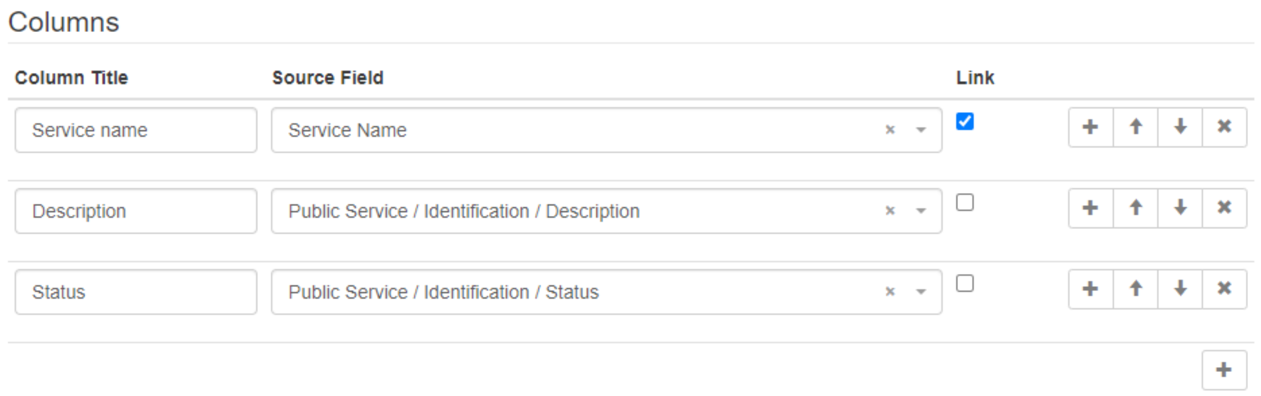
For summary views, the columns are displayed as follows:

- first column is displayed as a title, with link to the object page

- second column is displayed as summary. It makes sense to use a textarea field for this purpose, that has a few paragraphs of text. You can also use Markdown fields, they are displayed with formatting also in summary views.

- the rest, starting from third columns, are displayed as secondary information at the right side of the row

The selection available for source field depends on the view type (what objects are displayed in this particular view) and are fetched from the respective form definition. Naming of the source fields is based on the form definition and is composed of section name, fieldgroup name and field name. Therefore views should be defined after forms are ready and each time a form is changed later, the views should be verified and if necessary changed to reflect new fields.



Columns

Column Title	Source Field	Link	
Service name	Service Name	☑	+ ↑ ↓ ✖
Description	Public Service / Identification / Description	□	+ ↑ ↓ ✖
Status	Public Service / Identification / Status	□	+ ↑ ↓ ✖

Column Title – enter the text to show for the column title

Source Field – what field value is displayed in that column

Link – checking the Link checkbox next to the column will make text in that column link to the object, clicking on the link opens the object page. Usually, views should have at least one column marked as Link, but several columns can be links or none at all.

Add/delete/rearrange more columns using the buttons on the right side, but be conservative, don't make the view too crowded.

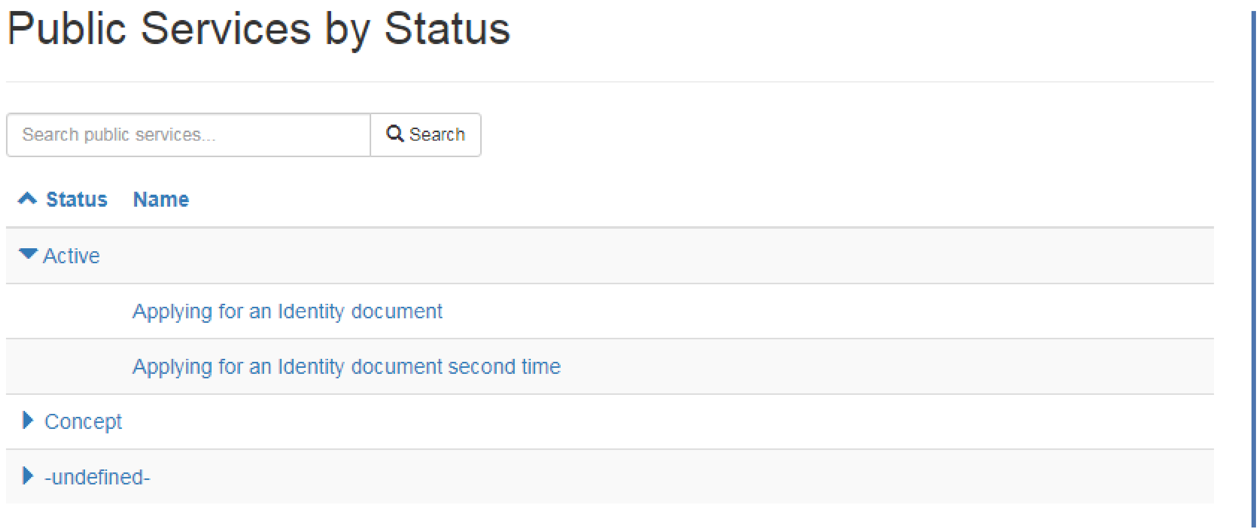
If you want the new definition to take effect, you also need to tick the "Active" checkbox. Then Save the view definition and verify it shows what you wanted.

Similar to forms, you can have multiple view definitions for the same view type, but only one of each type should normally be made active. Alternatively, you can have multiple active views of the same type, but you must define unique keys in the "Link key" attribute. If you don't define the link keys, it causes no error in the application, but it is not predictable, which one gets used.

If there is no (active) view definition for particular type, the system defaults (name, id) are used.

Category columns

Categorised views are views that present the results grouped by some field value, e.g. Status:



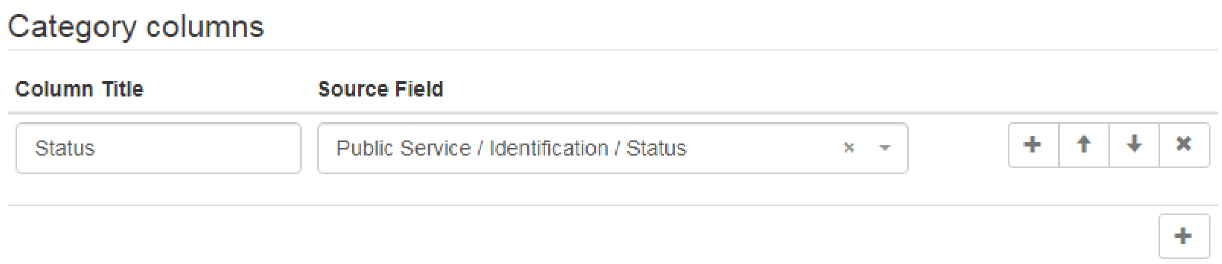
Public Services by Status

Search public services... | Search

Status	Name
Active	
	Applying for an Identity document
	Applying for an Identity document second time
Concept	
-undefined-	

The categories can be expanded/collapsed by clicking on the category name.

You can add a category column to a view definition to get such grouped results:



Category columns

Column Title	Source Field
Status	Public Service / Identification / Status

The following restrictions apply when defining category columns:

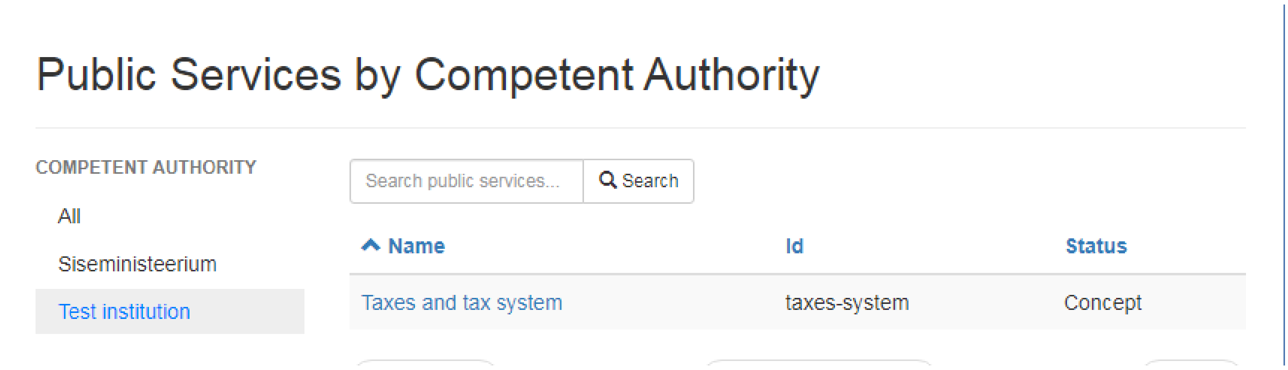
- only one category column can be added. The view definition allows you to add more, but only the first is currently used in the view.

- you can group by a field that is a selection type field, that has possible values defined in the form definition, either by use of a classifier or directly entered values. Alternatively you can group by a reference-type field, e.g. Institution field.

If you define a category column that uses hierachical classifier, the grouping in the view will also be hierachical.

Side navigation

An alternative to categorised views is to define a side navigation by some field value, like the one on the left side of this example:



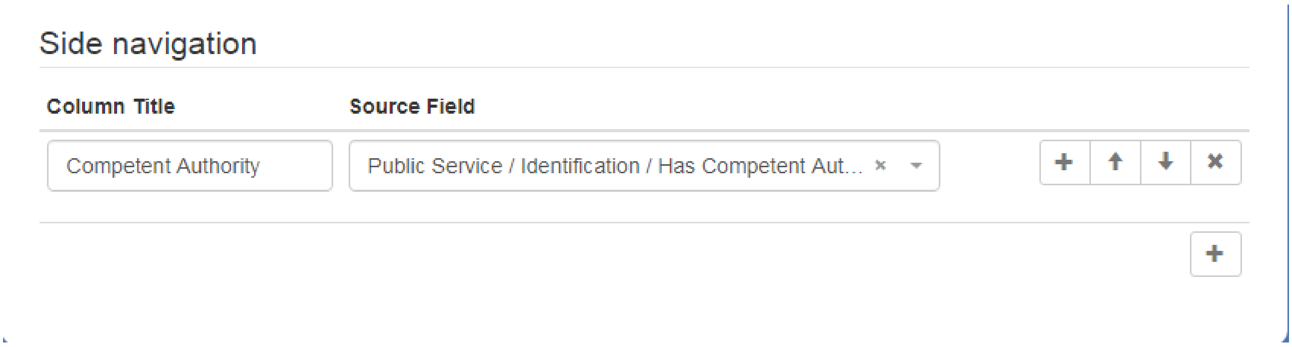
Public Services by Competent Authority

COMPETENT AUTHORITY

- All
- Siseministeerium
- Test institution

^ Name	Id	Status
Taxes and tax system	taxes-system	Concept

To define a side navigation, add a "column" to the Side navigation section of the view definition:



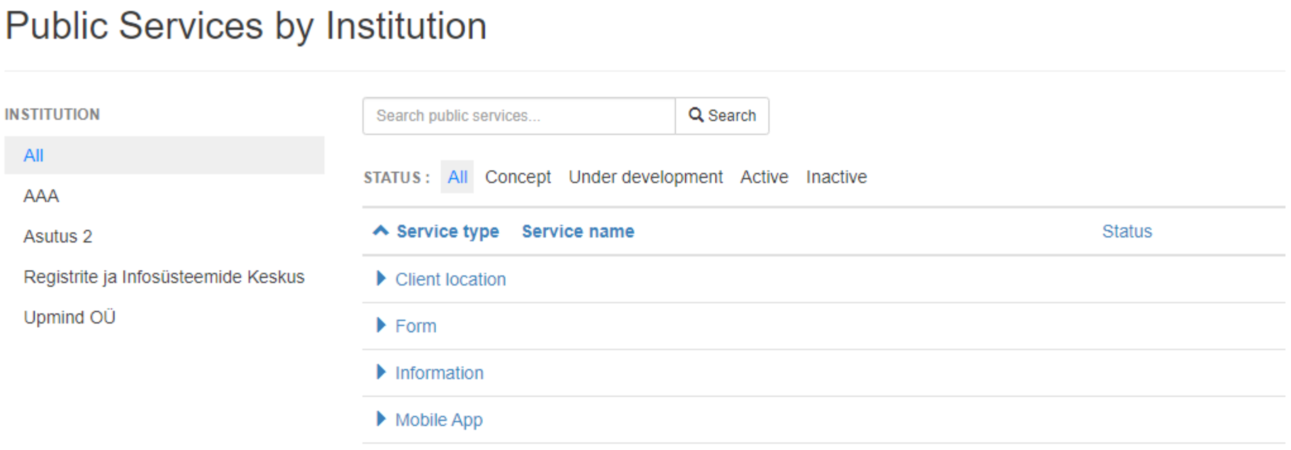
Side navigation

Column Title	Source Field
Competent Authority	Public Service / Identification / Has Competent Aut... ×

The same restrictions (only one select- or reference-type field) apply to the side navigation as to the category column. Additinally, if the column is linked to a classifier, the side navigation currently supports only single-level classifiers and not hierarchical classifiers.

Top Filter

A secondary filter can be added to the views, that appears on the top of the view. The following picture shows a filter by status field added to the public services view. Clicking on the status values filters the view to show only public services with selected status.



Public Services by Institution

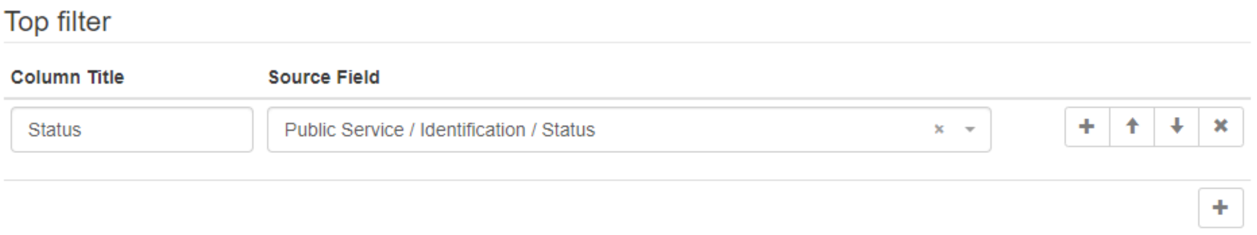
INSTITUTION

- All
- AAA
- Asutus 2
- Registrite ja Infosüsteemide Keskus
- Upmind OÜ

STATUS : All Concept Under development Active Inactive

Service type	Service name	Status
Client location		
Form		
Information		
Mobile App		

To define a top filter, add a "column" to the Top Filter section of the view definition:



Top filter

Column Title	Source Field
Status	Public Service / Identification / Status

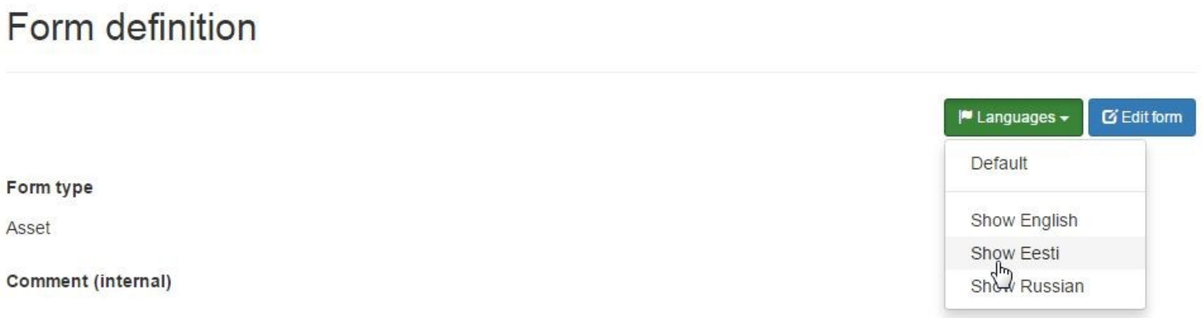
The same restrictions (only one select- or reference-type field, single level classifiers) apply to the top filter as to the side navigation. Additionally, it is recommended to use the top filter only with fields that have only a few possible values, e.g. status field or similar. Otherwise the page will get too cluttered with the values.

Chapter 10: Languages and translations

Translating forms, processes, etc

If the application should be used in multiple languages, the forms and many other objects defined in the system should be translated as well. The following description is base on a form definition, but the procedure is the same for view, workflows, email templates, etc.

Translations can be added after the form sections/fieldgroups/fields are (mostly) defined and the form definition is saved. To see the different language versions of the form, select another language from the Languages drop-down:



Form definition

Form type

Asset

Comment (internal)

If the language texts need to be changed, clikc on the Edit form button and replace the labels, help texts/etc with translated text, then save the form definition.

When editing language version of the form, only translatable labels, help, etc can be modified, the sections, fieldgroups or fields cannot be added or modified. To modify the form structure, use the Languages - Default option to switch to actual form definition and then click Edit form.

The list of languages that are used in the application are defined in the application configuration file. If application is used in single language only, the multilanguage feature can be disabled. For details, see the Installation and Configuration Guide.

Chapter 11. Connecting to XRoad infrastructure

If the organization managing CatIS is also connected to the XRoad infrastructure, it is possible to define the XRoad connection settings in the application configuration. Then, if a service is added to the catalogue and is specified as XRoad service, CatIS can construct the link to the service WSDL and automatically download the WSDL from the XRoad security server on demand. For this to work, a few settings must be correctly defined:

The institution form must have a field that holds the XRoad memberclass value for the institution (e.g. ORG, COM, NGO, etc). The canonical name of the field is then specified in the application configuration key `xroad.memberclass.fieldname`

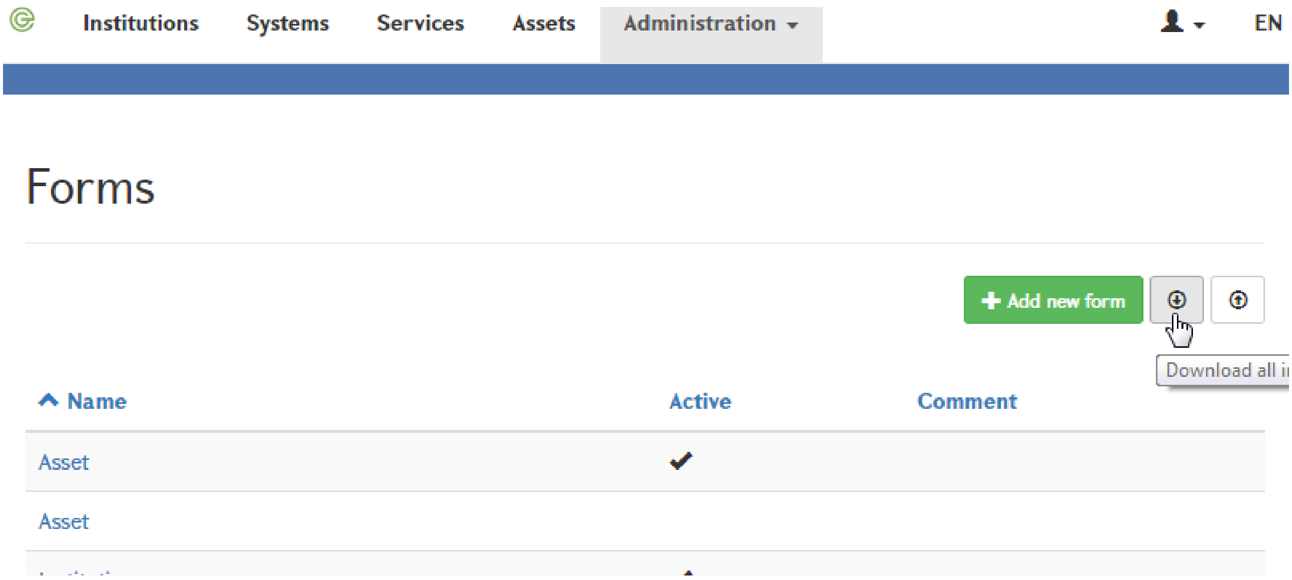
The service form must have a field, that marks a service as XRoad service. The name of the field and the value to match are then specified in the application configuration keys `xroad.service.fieldname` and `xroad.service.fieldvalue`

When the value of the field matches the value in the configuration file, CatIS generates the WSDL download link. If the service is not an XRoad service or the catalogue is not connected to XRoad, an ordinary file upload field can be used for providing access to the WSDL file.

See the Installation and Configuration Guide for the exact details about the configuration file and how to change the configuration values.

Chapter 12. Bulk downloading and uploading of application configuration objects

To support easier first configuration of the application or migrating from one instance to another, the configurable elements of the application (forms, views, roles, workflow definitions) can be downloaded in bulk from one instance of the application and then uploaded to another. For this, the corresponding views in the adminitrative section of the application have buttons on the right:



Name	Active	Comment
Asset	✓	
Asset		

Clicking on the "Download all in this view" will download all currently configured objects (e.g. forms, as shown in the above picture) in one JSON file. This file can then be uploaded to another instance of the application.

It is also possible to download just a single (form, view, etc) definition as a JSON file and upload to another instance, the link to download a single definition is on the definition form.

Chapter 13. Application logs

In addition to web server logs (if configured on the front-end web server), user actions in the application are logged in more human-readable form and can be accessed by application admins navigating to Administration -> Access Log. The view displays when, who, what object and what action was taken, with a linkt to the affected object. The access log can be searched and sorted like other views.

The amount of information can get quite large, if everything is logged and kept indefinitely. Therefore, it is possible to limit the amount of logged information by configuring the settings in the application configuration file (see Installation and Configuration Guide for the exact details). The following can be configured:

- if reads are logged – if you are not interested in who has read what page and only want to keep information about modifications, logging reads can be disabled, so the remaining logs are easier to work with

- if anonymous reads are logged – even if you decide to log, what information was accessed by authenticated users, for public-facing application, logging all public accesses ay still be too much. This is disabled by default.

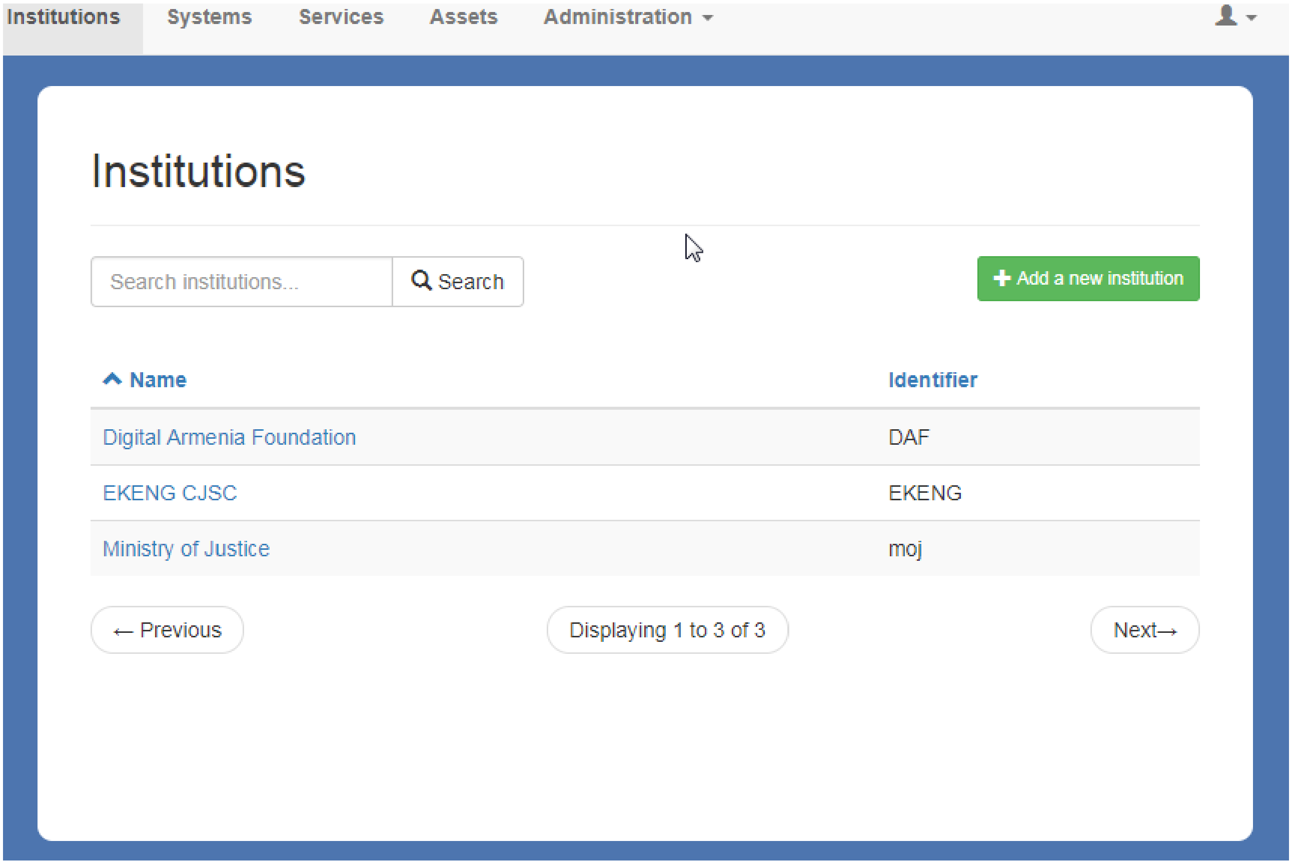
- how long the logs are kept – to prevent the log database from growing uncontrollably, the old logs are automatically deleted after a specified time, e.g. 30 days.

Chapter 14. JSON and XML API

The following chapters are meant for developers and other people, who plan or implement integrations with CatIS application. Users acting as administrative users of the application may find this information useful, however it is not required to support the day-to-day operation of the application.

For integrating with other external systems, the application provides a REST-style API that can be used to query the data in a few different formats. All the URI-s are the same, normally for browser requests the application responds with HTML content. If the API client specifies another format in the HTTP Accept: header, the application can respond with JSON or XML data respectively.

For example, making a browser request to `http://server.example.com/catis/institutions` could result in a page like this:



Institutions

Name	Identifier
Digital Armenia Foundation	DAF
EKENG CJSC	EKENG
Ministry of Justice	moj

Displaying 1 to 3 of 3

If the request header includes `Accept: application/json`, the same set of data would look like this:

```
{
    "page": 0,
    "offset": 0,
    "total": 3,
    "orderBy": 1,
    "filter": "",
    "items": [
        {
            "id": "IN00001",
            "name": "Digital Armenia Foundation",
            "href": "/am/catis/institutions/IN00001"
        },
        {
            "id": "IN00002",
            "name": "EKENG CJSC",
            "href": "/am/catis/institutions/IN00002"
        },
        {
            "id": "IN00003",
            "name": "Ministry of Justice ",
            "href": "/am/catis/institutions/IN00003"
        }
    ]
}
```

If the request header has Accept: application/xml, the result would be:

```
<?xml version="1.0" encoding="UTF-8"?>
<items page="0" offset="0" total="3" orderBy="1" filter="">
    <institution id="IN00001" name="Digital Armenia Foundation"
href="/am/catis/institutions/IN00001"/>
    <institution id="IN00002" name="EKENG CJSC"
href="/am/catis/institutions/IN00002"/>
    <institution id="IN00003" name="Ministry of Justice "
href="/am/catis/institutions/IN00003"/>
</items>
```

To fetch the data about some concete entity, the API client simply needs to follow the `href` attribute and make another request(s). The XML schema of the metadata is provided in Appendix 1. It must be noted that the schema does not specify the actual data itself (since the concrete dataset is provided via application configuration, i.e. what data is included with each object is specified in the forms definition process), but rather the format of presenting the metadata. An example of an institution metadata in XML format is provided in Appendix 2 and in JSON format in Appendix 3.

The JSON API has another method for retrieving list of objects together with the objects data in one single query, regardless of the Accept: header. This is intended for integration purposes with external systems, e.g. citizen portal or similar, that can get the data in a simpler way without needing to traverse the links separately. The URL for the query is with suffix /json, e.g. to get institutions' data in one query, make a HTTP request to the URL:

```
http://server.example.com/catis/institutions/json
```

Public Services have an additional experimental XML/RDF API according to the CPSV-AP v2.2 schema. To retrieve public services descriptions in CPSV-AP format, the application recognizes an `Accept: application/rdf+xml` header. An example of a public service metadata in CPSV-AP v2.2 format is provided in Appendix 4.

Request filtering

All HTTP requests regardless of the Accept: header can use additional query parameters for field-level filtering of the data. The parameters should follow the respective object's data definition -- section, fieldgroup and field names. Multiple query parameters can be used, the filters combine with an implicit AND.

Assuming for example that the institution form was defined such that there is a section with id: main that has a fieldgroup with id: identification that has fields with id: type and id: role, the request to retrieve only institutions with the institution type set to "Central Government" and role set to "Service Provider" would be:

```
http://server.example.com/catis/institutions?
main.identification.type=Central
%20Government&main.identification.role=Service%20Provider
```

NB! If the query parameter does not match the form definition, the parameter is simply ignored, so to get correct results it is important to match form definition precisely.

Appendix 1. XML Schema for metadata

```
<?xml version="1.0" encoding="utf-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" xmlns="http://ega.ee/catis"
targetNamespace="http://ega.ee/catis">
  <xs:element name="institution" type="institutionType"/>
  <xs:element name="system" type="systemType"/>
  <xs:element name="service" type="serviceType"/>
  <xs:element name="publicservice" type="publicServiceType"/>
  <xs:element name="asset" type="assetType"/>
  <xs:complexType name="institutionType">
    <xs:complexContent>
      <xs:extension base="compositeType"/>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="systemType">
    <xs:complexContent>
      <xs:extension base="compositeType">
        <xs:sequence>
          <xs:element name="_stdsys" type="xs:boolean"/>
          <xs:element name="_references">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="institution" type="refType"/>
                <xs:element name="standardSolution" type="refType"
minOccurs="0"/>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="serviceType">
    <xs:complexContent>
      <xs:extension base="compositeType">
        <xs:sequence>
          <xs:element name="_references">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="institution" type="refType"/>
                <xs:element name="parentinfosystem" type="refType"/>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="publicServiceType">
```

```
    <xs:complexContent>
      <xs:extension base="compositeType">
        <xs:sequence>
          <xs:element name="_references">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="institution" type="refType"/>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="assetType">
    <xs:complexContent>
      <xs:extension base="compositeType">
        <xs:sequence>
          <xs:element name="_references">
            <xs:complexType>
              <xs:sequence>
                <xs:element name="institution" type="refType"/>
              </xs:sequence>
            </xs:complexType>
          </xs:element>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
  <xs:complexType name="compositeType">
    <xs:sequence>
      <xs:element name="_ident" type="xs:string"/>
      <xs:element name="_owners">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="owner" type="refType" minOccurs="0"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="_history" type="historyType"/>
      <xs:element name="sections">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="section" type="sectionType"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string" use="required"/>
    <xs:attribute name="_id" type="xs:string"/>
  </xs:complexType>
  <xs:complexType name="refType">
    <xs:sequence>
      <xs:element name="id" type="xs:string"/>
      <xs:element name="name" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="historyType">
    <xs:sequence>
      <xs:element name="createdBy" type="xs:string"/>
      <xs:element name="createdAt" type="xs:dateTime"/>
      <xs:element name="modifiedBy" type="xs:string"/>
      <xs:element name="modifiedAt" type="xs:dateTime"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="sectionType">
    <xs:sequence>
      <xs:element name="fieldGroups">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="fieldGroup" type="fieldGroupType"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string" use="required"/>
  </xs:complexType>
  <xs:complexType name="fieldGroupType">
    <xs:sequence>
      <xs:element name="fieldGroupContents">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="fieldGroupContent" maxOccurs="unbounded">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="field" type="fieldType"
maxOccurs="unbounded"/>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string" use="required"/>
  </xs:complexType>
  <xs:complexType name="fieldType">
    <xs:choice>
      <xs:element name="text" type="xs:string"/>
      <xs:element name="textlist">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="text" type="xs:string" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="number" type="xs:integer"/>
      <xs:element name="numberlist">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="number" type="xs:integer" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="date" type="xs:dateTime"/>
      <xs:element name="datelist">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="dateTime" type="xs:dateTime"
maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="boolean" type="xs:boolean"/>
    </xs:choice>
    <xs:attribute name="name" type="xs:string" use="required"/>
  </xs:complexType>
</xs:schema>
```

Appendix 2. Example of metadata in XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<institution xmlns="http://ega.ee/catis" name="Digital Armenia Foundation"
_id="IN00001">
  <_ident>DAF</_ident>
  <_owners>
  </_owners>
  <_history>
    <createdBy>Admin Admin</createdBy>
    <createdAt>2017-11-07T12:53:51.820+02:00</createdAt>
    <modifiedBy>Admin Admin</modifiedBy>
    <modifiedAt>2017-11-07T12:58:58.200+02:00</modifiedAt>
  </_history>
  <sections>
    <section name="Institution">
      <fieldGroups>
        <fieldGroup name="general">
          <fieldGroupContents>
            <fieldGroupContent>
              <field name="englsh-name">
                <text></text>
              </field>
              <field name="regcode">
                <text></text>
              </field>
              <field name="URL">
                <text>http://www.gov.am/en/</text>
              </field>
              <field name="description">
                <text></text>
              </field>
              <field name="type">
                <textlist>
                  <text>Central Government</text>
                </textlist>
              </field>
              <field name="role">
                <textlist>
                  <text>Solution Developer</text>
                </textlist>
              </field>
              <field name="status">
                <textlist>
                  <text>Application</text>
                </textlist>
              </field>
              <field name="X-Road">
                <textlist>

                </textlist>
              </field>
            </fieldGroupContent>
          </fieldGroupContents>
        </fieldGroup>
        <fieldGroup name="contacts">
          <fieldGroupContents>
            <fieldGroupContent>
              <field name="name">
                <text>Eduard Nersisyan</text>
              </field>
              <field name="email">
                <text>eduard.nersisyan@gov.am</text>
              </field>
              <field name="phone">
                <text>+374 10 515710; +374 55 413088</text>
              </field>
              <field name="function">
                <text>x</text>
              </field>
            </fieldGroupContent>
          </fieldGroupContents>
        </fieldGroup>
        <fieldGroup name="events">
          <fieldGroupContents>
            <fieldGroupContent>
              <field name="type">
                <text></text>
              </field>
              <field name="date">
                <date></date>
              </field>
              <field name="decision">
                <text></text>
              </field>
            </fieldGroupContent>
          </fieldGroupContents>
        </fieldGroup>
      </fieldGroups>
    </section>
  </sections>
</institution>
```

Appendix 3. Example of metadata in JSON

```json
{
  "_id": "IN00001",
  "_owners": [],
  "_history": {
    "createdBy": "Admin Admin",
    "createdAt": "2017-11-07T12:53:51.820+02:00",
    "modifiedBy": "Admin Admin",
    "modifiedAt": "2017-11-07T12:58:58.200+02:00"
  },
  "_name": "Digital Armenia Foundation",
  "_ident": "DAF",
  "Institution": [
    {
      "general": [
        {
          "englsh-name": "",
          "regcode": "",
          "URL": "http://www.gov.am/en/",
          "description": "",
          "type": ["Central Government"],
          "role": ["Solution Developer"],
          "status": ["Application"],
          "X-Road": []
        }
      ]
    },
    {
      "contacts": [
        {
          "name": "Eduard Nersisyan",
          "email": "eduard.nersisyan@gov.am",
          "phone": "+374 10 515710; +374 55 413088",
          "function": "x"
        }
      ]
    },
    {
      "events": [
        {
          "type": "",
          "date": "",
          "decision": ""
        }
      ]
    }
  ]
}
```

Appendix 4. Example of Public Service metadata in CPSV-AP v2 XML/RDF format

```
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
xmlns:dcterms="http://purl.org/dc/terms/"
xmlns:dcat="http://www.w3.org/ns/dcat#" xmlns:cv="http://data.europa.eu/m8g/"
xmlns:adms="http://www.w3.org/ns/adms#" xmlns:schema="https://schema.org/">
  <rdf:Description rdf:nodeID="PS00012">
    <dcterms:identifier>PS00012</dcterms:identifier>
    <rdf:type rdf:resource="http://purl.org/vocab/cpsv#PublicService"/>
    <dcterms:title xml:lang="en">Applying for an Identity
document</dcterms:title>
    <dcterms:description>The ID-card can be applied for in Service Offices, in
foreign representations of the Republic of Estonia, or by post.

The ID-card can be applied for in expedited procedure in the Service Offices. If
you have not been issued the identity document earlier, then it is not possible
to apply for the ID-card in expedited procedure.</dcterms:description>
    <dcterms:type
rdf:resource="https://unstats.un.org/unsd/cr/registry/regcs.asp?
Co=Information"/>
    <dcterms:type
rdf:resource="https://unstats.un.org/unsd/cr/registry/regcs.asp?Co=Form"/>
    <dcterms:type
rdf:resource="https://unstats.un.org/unsd/cr/registry/regcs.asp?Co=Service
Bureau"/>
    <adms:status rdf:resource="http://purl.org/adms/status/Active"/>
    <cv:hasCompetentAuthority rdf:resource="ca_0"/>
    <cv:isGroupedBy rdf:nodeID="le_0_0"/>
    <cv:isGroupedBy rdf:nodeID="le_0_1"/>
    <cv:isGroupedBy rdf:nodeID="le_0_2"/>
    <cv:hasChannel rdf:resource="ch_0"/>
    <cv:hasChannel rdf:resource="ch_1"/>
  </rdf:Description>
  <rdf:Description rdf:about="ca_0">
    <rdf:type rdf:resource="http://data.europa.eu/m8g/PublicOrganisation"/>
    <dcterms:identifier>70000562</dcterms:identifier>
    <dcterms:title>Ministry of the Interior</dcterms:title>
  </rdf:Description>
  <rdf:Description rdf:nodeID="le_0_0">
    <rdf:type rdf:resource="http://data.europa.eu/m8g/LifeEvent"/>
    <dcterms:type rdf:resource="http://data.europa.eu/m8g/LifeEvent/Changing
relationship status"/>
  </rdf:Description>
  <rdf:Description rdf:nodeID="le_0_1">
    <rdf:type rdf:resource="http://data.europa.eu/m8g/LifeEvent"/>
    <dcterms:type rdf:resource="http://data.europa.eu/m8g/LifeEvent/Travelling
abroad"/>
  </rdf:Description>
  <rdf:Description rdf:nodeID="le_0_2">
    <rdf:type rdf:resource="http://data.europa.eu/m8g/LifeEvent"/>
    <dcterms:type rdf:resource="http://data.europa.eu/m8g/LifeEvent/Moving
to/from the country"/>
  </rdf:Description>
  <rdf:Description rdf:nodeID="ch_0">
    <rdf:type rdf:resource="http://data.europa.eu/m8g/Channel"/>
    <dcterms:type rdf:resource="http://data.europa.m8g/Web"/>
    <schema:openingHours></schema:openingHours>
  </rdf:Description>
  <rdf:Description rdf:nodeID="ch_1">
    <rdf:type rdf:resource="http://data.europa.eu/m8g/Channel"/>
    <dcterms:type rdf:resource="http://data.europa.m8g/Service Bureau"/>
    <schema:openingHours></schema:openingHours>
  </rdf:Description>
</rdf:RDF>
```