OnTask Documentation

Release 1.1.0

Abelardo Pardo

Feb 18, 2018

Contents:

1 What is OnTask **1**
1.1 Research 2
1.2 License 2

2 Installation process **3**
2.1 Requirements 3
2.2 Installing the required tools 3
2.2.1 Install and Configure Redis 4
2.2.2 Install and Configure PostgreSQL 4
2.2.3 Install Python 4
2.2.4 Download, install and configure OnTask 4
2.3 The Administration Pages 6
2.4 Production Deployment 7
2.5 Authentication 7
2.6 Scheduling tasks 8

3 Using the tool **9**
3.1 The Workflow 11
3.2 Workflow Details 13
3.2.1 The Columns 18
3.3 Data Operations 18
3.3.1 CSV Files 21
3.3.2 Merge Operation 21
3.4 The Table 25
3.5 The Actions 26
3.5.1 Actions In 26
3.5.2 Actions Out 30
3.6 The Scheduler 36
3.7 The Logs 36

4 OnTask Tutorial **41**
4.1 Create a new workflow 42
4.2 Open a workflow 42
4.3 Data Upload 44
4.4 Browsing the table 48
4.5 Workflow Operations 48
4.5.1 Attributes 49

    4.5.2 Share 49
    4.5.3 Export 49
    4.5.4 Clone 50
    4.5.5 Rename 50
    4.5.6 Flush Data 50
    4.5.7 Delete 50
  4.6 Actions 50
    4.6.1 Actions Out 51
    4.6.2 Actions In 51

5 Advanced Features **55**
  5.1 The API (Application Programming Interface) 55

CHAPTER 1

What is OnTask

Welcome to OnTask, the platform offering instructors and educational designers the capacity to use data to personalise the learners experience.

Learning is complex, highly situated, and requires interacting with peers, instructors, resources, platforms, etc. This complexity can be alleviated providing learners with the right support actions. But this process becomes increasingly complex when the number of learners grows. The more learners, the more difficult is for instructors to provide support and usual solutions usually include generic resources that are only relevant to a subset of the audience.

Testimonials

Comments extracted from focus interviews with learners.

"its not the email itself, but what it represents: the accountability for your learning"

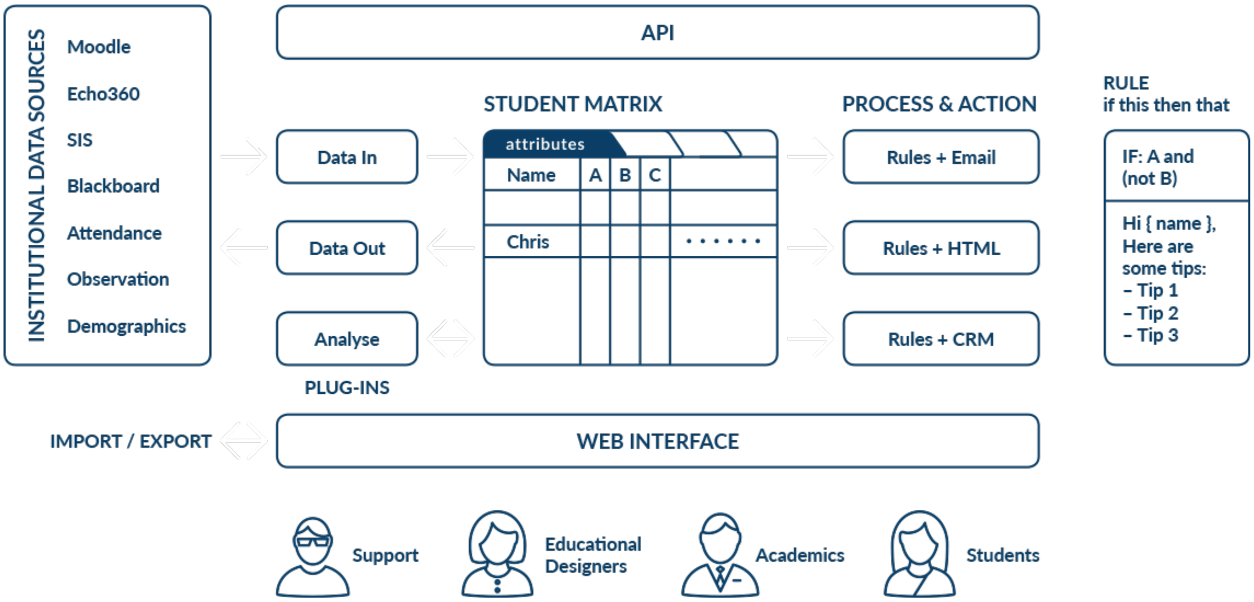
"I think it makes sense as a reflection of the whole semester, oh this is where I struggled with."

In parallel wih this increase in complexity, learning platforms now generate a wealth of data when activities are mediated by technology. This data can be collected and used to help instructors and designers provide a truly personalised experience. Why is this not hapenning in current platforms? Because the connection between this data and learner support actions is challenging to implement. This is the focus of OnTask: provide instructors and designers with a platform to connect data emerging from learning environments with highly personalised student support actions.

OnTask is a web application that allows instructors and designers to combine data previously captured and stored in a table with a set of rules to personalise the content of a web document. That document can then be sent as an email or made available to each learner. The following picture shows the high level structure of the platform.

The rest of the document is divided into three blocks. The *first block* covers the technical details to download, install and configure the tool. It requires technological expertise and access to the adequate computing facilities (a virtual machine, a server, or similar). The main audience of this part is system administrators and advanced users that want to use the tool within their institution or for their own use.

The *second part* explains how to use OnTask in the context of a learning experience, how to load data, manipulate the table and create the personalised actions. The audience for this part is teachers and designers that need to personalise the interactions they have with learners in a platform offering some data sources.


INSTITUTIONAL DATA SOURCES
Moodle
Echo360
SIS
Blackboard
Attendance
Observation
Demographics
API
Data In
Data Out
Analyse
PLUG-INS
STUDENT MATRIX
attributes
Name
A
B
C
Chris
PROCESS & ACTION
Rules + Email
Rules + HTML
Rules + CRM
RULE
if this then that
IF: A and (not B)
Hi { name }, Here are some tips:
– Tip 1
– Tip 2
– Tip 3
IMPORT / EXPORT
WEB INTERFACE
Support
Educational Designers
Academics
Students

The *third part* covers some more advanced functionalities such an application programming interface (API) that allows other platforms to upload data to OnTask.

1.1 Research

There are several platforms that implement similar functionality or follow a similar approach (see for example SRES. The common thread among them is the positive impact that personalised communication may have when supporting learners. There are a few scientific publications that document the ideas and processes that inspired the creation of OnTask:

- Liu, D. Y.-T., Taylor, C. E., Bridgeman, A. J., Bartimote-Aufflick, K., & Pardo, A. (2016). Empowering instructors through customizable collection and analyses of actionable information Workshop on Learning Analytics for Curriculum and Program Quality Improvement (pp. 3). Edinburgh, UK.
- Liu, D. Y. T., Bartimote-Aufflick, K., Pardo, A., & Bridgeman, A. J. (2017). Data-driven Personalization of Student Learning Support in Higher Education. In A. Peña-Ayala (Ed.), Learning analytics: Fundaments, applications, and trends: A view of the current state of the art: Springer. doi:10.1007/978-3-319-52977-6_5
- Pardo, A., Jovanović, J., Dawson, S., Gašević, D., & Mirriahi, N. (In press). Using Learning Analytics to Scale the Provision of Personalised Feedback. British Journal of Educational Technology. doi:10.1111/bjet.12592

1.2 License

The OnTask software is open source and available under the MIT License.

CHAPTER 2

Installation process

OnTask is a Web application that manages data about learners to provide them with personalised support. For this reason, it is recommended an installation that observes a set of tight security restrictions. Some of these restrictions lie within the scope of the tool, but some other are part of the environment in which the application is installed. We strongly recommend to install OnTask in a web server that uses TTL encryption (HTTPS) to serve all the pages. The application requires exchanging with your browser sensitive information about your session, so the information should be encrypted.

2.1 Requirements

OnTask has been developed as a Django application. Django is a high-level, python-based web framework that supports a rich set of functionalities typically required in applications like OnTask. But as with many other applications, OnTask requires a set of additional applications for its execution:

- Python 2.7.13 (or later)
- Django (version 1.11 or later)
- Additional django modules (included in the requirements/base.txt) file
- Redis (version 4.0 or later)
- PostgreSQL (version 9.5 or later)

Some of these requirements are (hopefully) properly handled by Python through its package index application pip.

2.2 Installing the required tools

The following installation steps assume that you are deploying OnTask in a production web server capable of serving pages using the HTTPS protocol.

2.2.1 Install and Configure Redis

1. Download and install redis.

   Follow the instructions to configure it to be used by Django.

2. Test that it executes properly

For OnTask you only need executing in the machine. If you use the default settings, there are not additional changes required in OnTask (the code is alrady using this application internally).

2.2.2 Install and Configure PostgreSQL

1. Download and install postgresql.
2. Create the role `ontask` with the command `createuser`. The role should be able to create new databases but not new roles and you should define a password for the user (use `createuser --interactive -W`).
3. Adjust the access configuration in postgresql (usually in file `pg_hba.conf`) to allow the newly created user to access databases locally.
4. Create a new database with name `ontask` with the `createdb` command.
5. Use the client application `psql` to verify that the user has access to the newly created database and can create and delete a new table and run regular queries. Try to connect to the database with the following command:

   ```
   psql -h 127.0.0.1 -U ontask -W ontask
   ```

   If the client does not connect to the database, review your configuration options.

2.2.3 Install Python

In the following sections we assume that you can open a command line interpreter and you can execute the python intepreter.

1. Install python
2. Verify that the python interpreter can run and has the right version (2.7) using the command line interpreter.
3. Install pip (the package may be called `python-pip`). This tool will be used by both python and django to install numerous libraries that are required to execute OnTask.

2.2.4 Download, install and configure OnTask

1. Download or clone_actions a copy of OnTask.
2. Using a command interpreter, go to the OnTask folder and locate a folder inside it with name `requirements`. Verify that the `requirements` folder contains the files `base.txt`, `production.txt` and `development.txt`. The first file contains a list of python modules that are required by OnTask. The second is a set of additional modules to run a *production* instance, and the third is a list if you intend to run a *development* instance.
3. If you plan to run a production instance of OnTask execute the command:

   ```
   pip install -r requirements/production.txt
   ```

   Alternatively, if you plan to run a development instance of OnTask then execute the commmand:

```
pip install -r requirements/development.txt
```

This command traverses a list of libraries and modules and installs them as part of the python libraries in the system. These modules include Django, Django Rest Framework, django braces, etc.

At this point you have the major modules in place. The next steps include the configuration of the Django environment to run OnTask.

If you plan to install a production instance of OnTask, using a plain text editor (nano, vim, emacs or similar) in a command line interpreter, open the file `manage.py` in the `src` folder of the project. Modify line 14 replacing the value `"ontask.settings.development"` by `"ontask.settings.production"`. Save and close the file.

Using the same plain text editor create a file with name `local.env` in the folder `src/ontask/settings` with the following content (note there is no space between variable names and the equal sign):

```
TIME_ZONE='[YOUR LOCAL PYTHON TIME ZONE]'
# syntax: DATABASE_URL=postgres://username:password@127.0.0.1:5432/database
DATABASE_URL=postgres://[PSQLUSERNAME]:[PSQLPWD]@127.0.0.1:5432/ontask
SECRET_KEY=
LTI_OAUTH_CREDENTIALS=key1=secret1,key2=secret2
```

1. Open a command interpreter and execute the following python command:

   ```
   python -c 'import tzlocal; print(tzlocal.get_localzone().zone)'
   ```

   Replace `[YOUR LOCAL PYTHON TIME ZONE]` in the `local.env` file by the description of your time zone produced by the previous command.

2. Modify the line starting with `DATABASE_URL=` and change the field `[PSQLUSERNAME]` with the name of the Postgresql user created in the previous step (the one that could access the ontask database and run queries). If you decided to use a different name for the database, adjust the last part of the line accordingly (replace *ontask* by the name of your database).

3. Open a command interpreter and execute the following python command:

   ```
   python -c 'import random; import string; print("".join([random.SystemRandom().
   ↪choice(string.digits + string.ascii_letters + string.punctuation) for i in␣
   ↪range(100)]))'
   ```

   Copy the long string produced as output and add it at the end of the last line of the file `local.env`. It should look something like (with different content after the equal sign):

   ```
   SECRET_KEY=4o93jf0572094jv...
   ```

4. Modify the line starting with `LTI_OAUTH_CREDENTIALS` and include a comma-sepparated list of pairs key=secret for LTI authentication. See the section *Authentication* for more details about this type of authentication.

5. Create a new folder with name `logs` in the OnTask top folder (next to the `requirements` folder). This folder **is different** from the folder with the same name in the `src` folder.

6. If at some point during the following steps you want to reset the content of the database, run the commands `dropdb` and `createdb`

7. Execute the following commands from the `src` folder to prepare the database initialization:

   ```
   python manage.py makemigrations profiles accounts workflow dataops
   python manage.py makemigrations table action logs scheduler
   ```

8. Execute the following command to create the database internal structure:

   ```
   python manage.py migrate
   ```

   A few messages should appear on the screen related to the initalizaton of the database.

9. Execute the following command to upload to the platform some initial data structures:

   ```
   python manage.py runscript -v1 --traceback initial_data
   ```

   The command should run without any error or exception.

10. Execute the command to create a superuser account in OnTask:

    ```
    python manage.py createsuperuser
    ```

    Remember the data that you enter in this step so that you use it when you enter OnTask with your browser.

11. Go to the `docs` folder to generate the documentation. Make sure this folder contains the sub-folders with name `_static` and `_templates`. Execute the command:

    ```
    make html
    ```

    The documentation is produced by the `sphinx-doc` application and generates the directory `_build`. The documentation for the platform is in the folder `_build/html`.

12. Copy the entire `html` folder (inside `_build`) over to the `src/static` folder (in Unix `cp -r _build/html ../src/static`).

13. From the `src` folder execute the following command to collect and install the static content:

    ```
    python manage.py collectstatic
    ```

14. If you are running a production instance, execute the following command to check the status of the platform:

    ```
    python manage.py check --deploy
    ```

    The command should print just one warning about the configuration variable X_FRAME_OPTIONS. If you are running a development instance, you will get various additional warning that are derived most of them from running the instance without HTTPS.

15. Execute the following command to start the OnTask server:

    ```
    python manage.py runserver
    ```

    If there are no errors, the message on the screen should say that your server is running and available in the URL 127.0.0.1:8000

16. If OnTask is going to be accessed through a web server like Apache or Nginx, stop the application and configure the web server accordingly.

2.3 The Administration Pages

As many applications developed using Django, OnTask takes full advantage of the administration pages offered by the framework. The account created with the command `createsuperuser` has complete access to those pages through a link in the upper right corner of the screen.

These pages offer access to several important operations:

- The elements of each of the models stored in the database (workflows, actions, conditions, columns, etc). Each model has its corresponding page allowing the creation, update and deletion of any object.
- The user information. This is a special model representing the users, their name, credentials, etc. The platform allows the creation of user accounts.
- The group information. The platform differentiates users based on groups. Each group has different functionalities.

Once the instance is running, visit these pages and configure the platform to your needs.

2.4 Production Deployment

Once OnTask is executing normally, you may configure a web server (nginx, apache or similar) to make it available to a community of users. The instructions to make such deployment are beyond the scope of this manual but they are available for users to consult.

2.5 Authentication

OnTask comes with three default authentication mechanisms (and are used in the following order): LTI, `REMOTE_USER` and basic authentication.

- IMS Learning Tools Interoperability (IMS-LTI). LTI is a standard developed by the IMS Global Leanring Consortium to integrate multiple tools within a learning environment. In LTI terms, OnTask is configured to behave as a *tool provider* and assumes a *tool consumer* such as a Learning Management System to invoke its functionality. Any URL in OnTask can be give nto the LTI consumer as the point of access.

  Ontask only provides two points of access for LTI requests coming from the consumer. One is the url with suffix `/lti_entry` and the second is the URL provided by the actions to serve the personalised content (accessible through the `Actions` menu.

  To allow LTI access you need:

  1. A tool consumer that can be configured to connect with OnTask. This type of configuration is beyond the scope of this manual.
  2. A set of pairs key,value in OnTask to be given to the tool consumers so that together with the URL, they are ready to send the requests. The key/value pairs are specified in the file `local.env` in the folder `src/ontask/settings` together with other local configuration variables. For example:

     ```
     LTI_OAUTH_CREDENTIALS=key1=secret1,key2=secret2
     ```

     If you change the values of this variable, you need to restart the server so that the new credentials are in effect.

  This authentication has only basic functionality and it is assumed to be used only for learners (not for instructors).

- `REMOTE_USER`. The second method uses the variable REMOTE_USER that is assumed to be defined by an external application. This method is ideal for environments in which users are already authenticated and are redirected to the OnTask pages (for example, using SAML). If OnTask receives a request from a non-existent user through this channel, it automatically and transparently creates a new user in the platform with the user name stored in the `REMOTE_USER` variable. OnTask relies on emails as the username differentiator, so if you plan to use this authentication method make sure the value of `REMOTE_USER` is the email.

- Basic authentication. If the variable `REMOTE_USER` is not set in the internal environment of Django where the web requests are served, OnTask resorts to conventional authentication requiring email and password. These credentials are stored in the internal database managed by OnTask.

There are other possibilities to handle user authentication (LDAP, AD, etc.) but they require ad-hoc customizations in the tool and are not provided as out-of-the-box solutions.

2.6 Scheduling tasks

OnTask allows to program certain tasks to execute at some point in the future. This functionality is implemented using a combination of persistent storage (the database), and a time-based job scheduler external to the tool. Time-based job schedulers are present in most operating systems (`cron` in Unix/Linux, or `at` in Windows). OnTask provides the functionality to describe the task to execute and assing a date and time. Additionally, the tool has a script that checks the date/time, selects the appropriate task, and executes it. These instructions describe the configuration assuming an underlying Linux/Unix architecture using `crontab`.

Create a *crontab* file for the user running the server in your production environment with the following content:

```
MAILTO="[ADMIN EMAIL ADDRESS]"
PATH=/usr/local/sbin:/usr/local/sbin:/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin
SHELL=/bin/bash
ONTASK_PROJECT=[PATH TO ONTASK PROJECT ROOT]

0,15,30,45 * * * * python ${ONTASK_PROJECT}/src/manage.py runscript scheduler_script -
↪v3 --traceback --script-args="-d" > [CRONTAB LOG] 2>&1
```

Modify the previous script with the following information:

- In the first line change the email address by an address to receive the notifications produced by `crontab`. The server running the application needs to have the capacity to send emails.
- Make sure the variable `PATH` contains the path to execute `python`.
- Define the variable `SHELL` to point to the shell to use to execute `python`.
- Change the value of the variable `ONTASK_PROJECT` to point to the root of OnTask (the folder containing the source code).
- In the last line replace `[CRONTAB LOG]` with a file to capture the output that the screen would write to a regular terminal. We recommend this file to be in a temporary directory.

The script `scheduler_scrip` has been configured to print messages through a logger that writes the messages to the file `script.log` in the `logs` folder.

The remaining parameter to adjust is the frequency in which the script `scheduler_script` runs. In the example above, the script executes every 15 minutes (at minutes 0, 15, 30 and 45 minutes in the hour). Adjust this frequency to suit your needs. Once adjusted, go to the admin menu in OnTask, open the section with name *Core Configuration*, click in the preferences and adjust the value of the *Minute interval to program scheduled tasks* and match it (in minutes) to the interval reflected in the crontab.

CHAPTER 3

Using the tool

In a nutshell, the idea of OnTask is to help instructors and designers to use data available about a what is happening in a learning experience to design and deploy personalised learner support actions. This last term, *personalised support actions* is purposefully vague to include any action that is given to learners in different form depending on some personal conditions. The following figures illustrates an example of how OnTask can help instructors and learners.

Imagine a learning experience in which you want to provide personalised messages to the learners in three instances. In the first week, you want to send a welcome message and change slightly the text based on the type of background. The second week you want to send some comments and suggestions about the participation in the forum but the text will depend on the measures of engagement obtained from the platform. Finally, you want to send a third personalised message depending on the level of engagement with the videos in the course. The main idea of these messages is that you want to change some portions based on the information available in the table.

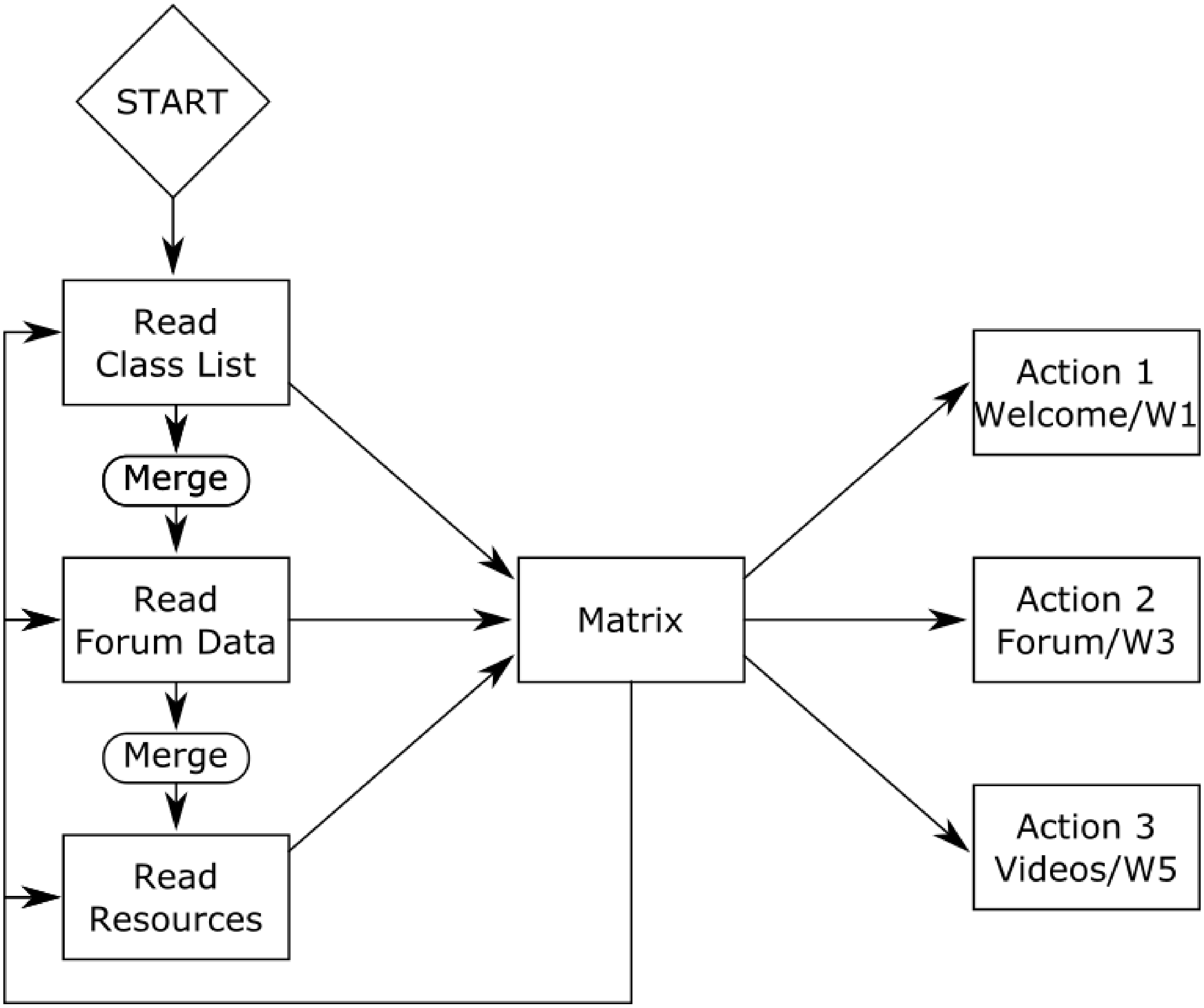
The main entity in the platform is called a **workflow** and represents a set of data capturing procedures, a table with current data, and a set of actions. The usual steps require first to populate the table with data extracted from the learning environment. In the figure we assume three data sources: the class list (basic information about the learners), data about participation in an online discussion forum, and data about resource usage.

These three sources are combined and stored in the second entity in OnTask: the table. Think of the table as a conventional excel sheet storing the information about the learners (one learner per row and a set of features about each learner as columns).

The third entity in OnTask is the *action* that is basically a text with elements that can be changed and adapted to each learner depending on the values of the features included in the table. This text can be included in an email, made available through a web page, or forwarded to another system for further processing.

A workflow in OnTask contains a single table and a set of actions. This container is conceived to manage the data and actions related to a learning experience. You may use the workflow shown in the documentation importing the `ELON3509 workflow`.

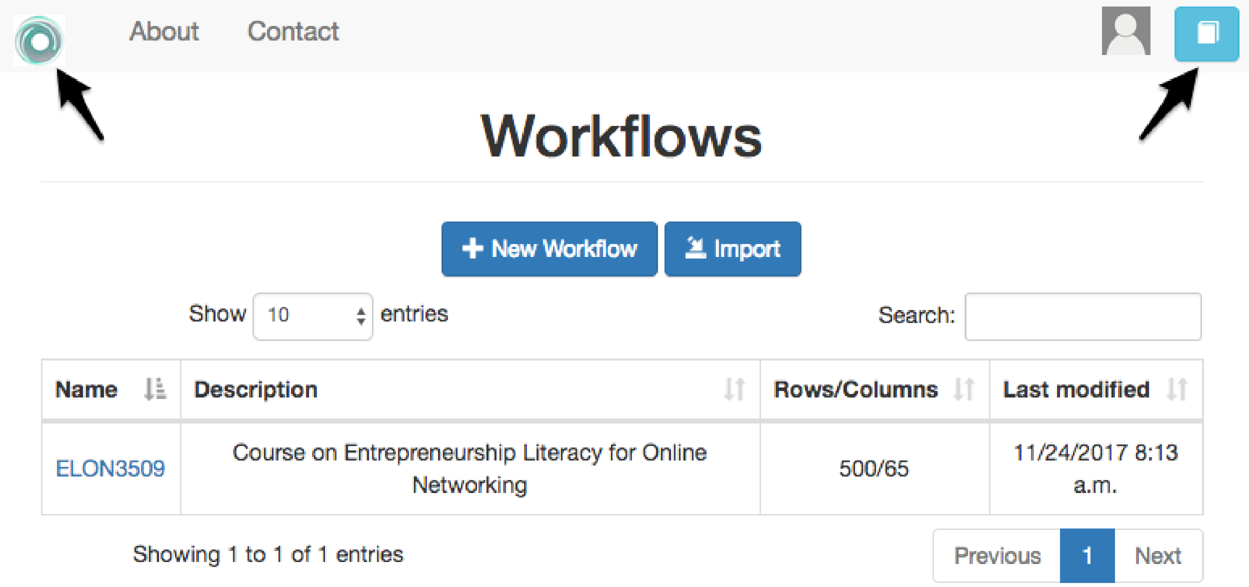
The following sections offer a more in-depth description of all these elements as well as examples with real scenarios.


START
Read Class List
Merge
Read Forum Data
Merge
Read Resources
Matrix
Action 1 Welcome/W1
Action 2 Forum/W3
Action 3 Videos/W5

3.1 The Workflow

> "But anyone who has experienced flow knows that the deep enjoyment it provides requires an equal degree of disciplined concentration." - Mihaly Csikszentmihalyi

The workflow in OnTask is simply a container that brings together the table, the operations to upload the data, and the actions to create the personalised content. The initial screen in OnTask shows the available workflows as shown in the following figure.



Name	Description	Rows/Columns	Last modified
ELON3509	Course on Entrepreneurship Literacy for Online Networking	500/65	11/24/2017 8:13 a.m.

Showing 1 to 1 of 1 entries

Clicking in the OnTask icon in the upper left corner brings you back to the list of workflows from any page in the application. Clicking in the icon in the upper right corner opens the documentation page.

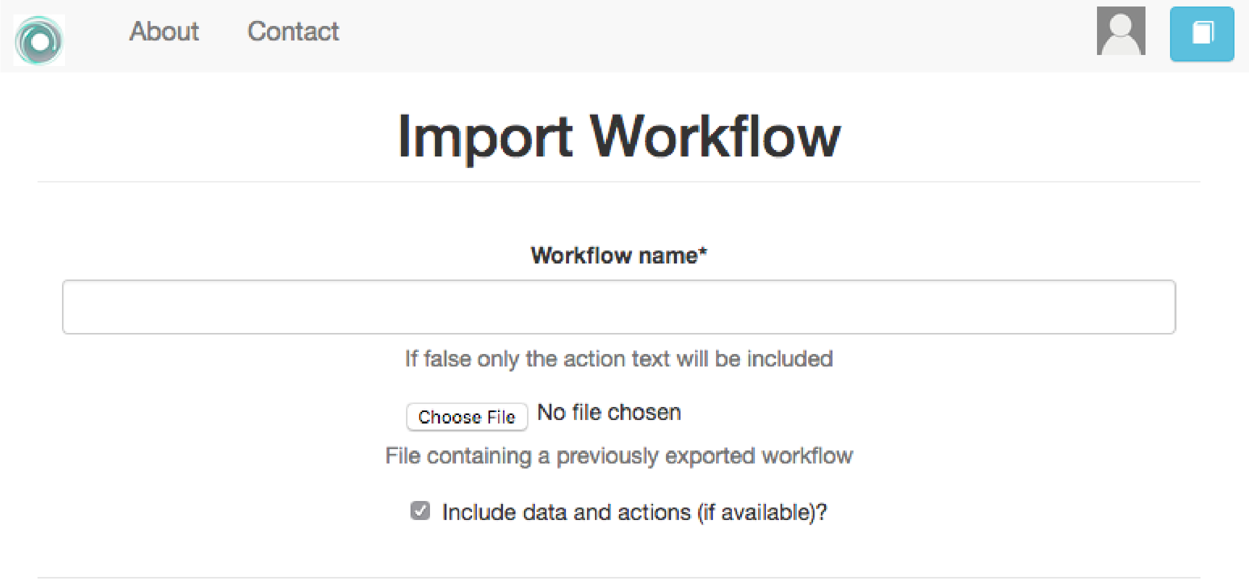
The screen includes three operations:

Going back

Remember that clicking in the icon in the upper left corner *closes* the workflow and you see the list of available workflows again.

1. Create a new workflow: Clicking in the *New Workflow* button will ask you for the name and description of the element.
2. Importing a file containing a previously exported workflow. Clicking in the *Import* button will ask you for the new for the new workflow, the file and will let you choose to import only the data, or the data and the actions as shown in the following figure
3. Open a workflow to work on it. This is perhaps the most common initial step. Once you open or *enter* a worklow, all the operations are applied to that context. The platform will remind you in which workflow you are working by inserting its name right under the top navigation bar as shown in the following figure.

Once you open a workflow, it is locked and no other user can manipulate it (see *sharing a workflow*). If access a workflow and another user is currently working with it, the platform will not allow you to see the data and will instead tell you who is holding the lock.


About
Contact
Choose File

Import Workflow

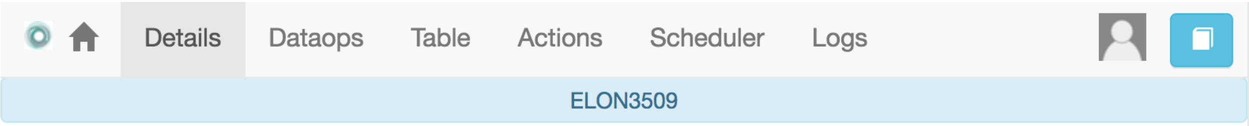
Workflow name*

If false only the action text will be included

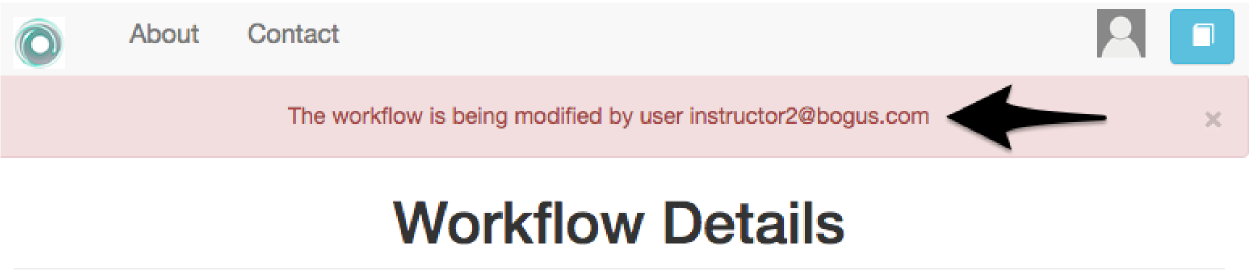
No file chosen

File containing a previously exported workflow

☑ Include data and actions (if available)?


Details
Dataops
Table
Actions
Scheduler
Logs

ELON3509


About
Contact
×

The workflow is being modified by user instructor2@bogus.com

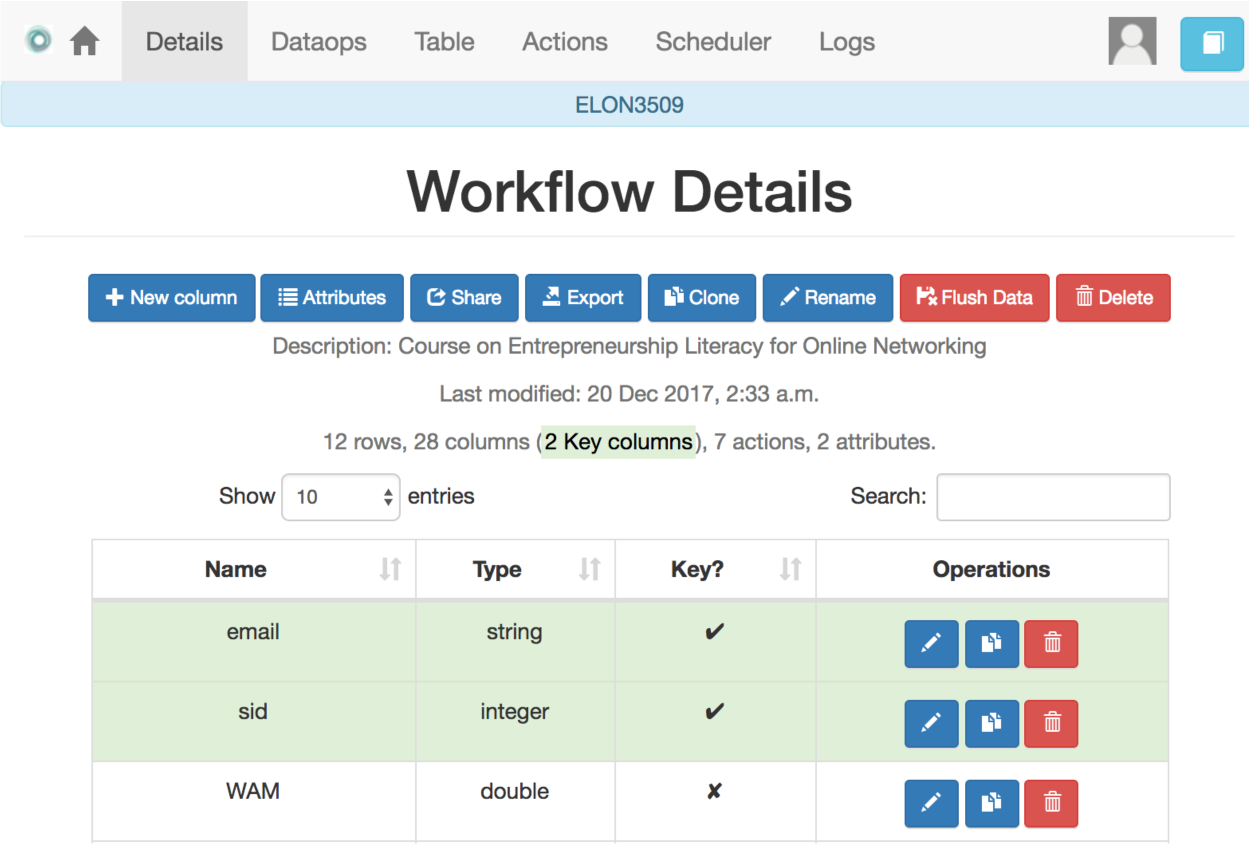
Workflow Details

The operations to manage a workflow all become visible once you select it by clicking on its name.

3.2 Workflow Details

> "The details are not the details. They make the design" – Charles Eames

After selecting a workflow to manage, the *details* page appears with a lot of information about operations, structure of the data, information about the columns, etc. The top of the screen contains the information shown in the following figure.



Description: Course on Entrepreneurship Literacy for Online Networking

Last modified: 20 Dec 2017, 2:33 a.m.

12 rows, 28 columns (2 Key columns), 7 actions, 2 attributes.

Name	Type	Key?	Operations
email	string	✔	
sid	integer	✔	
WAM	double	✘	

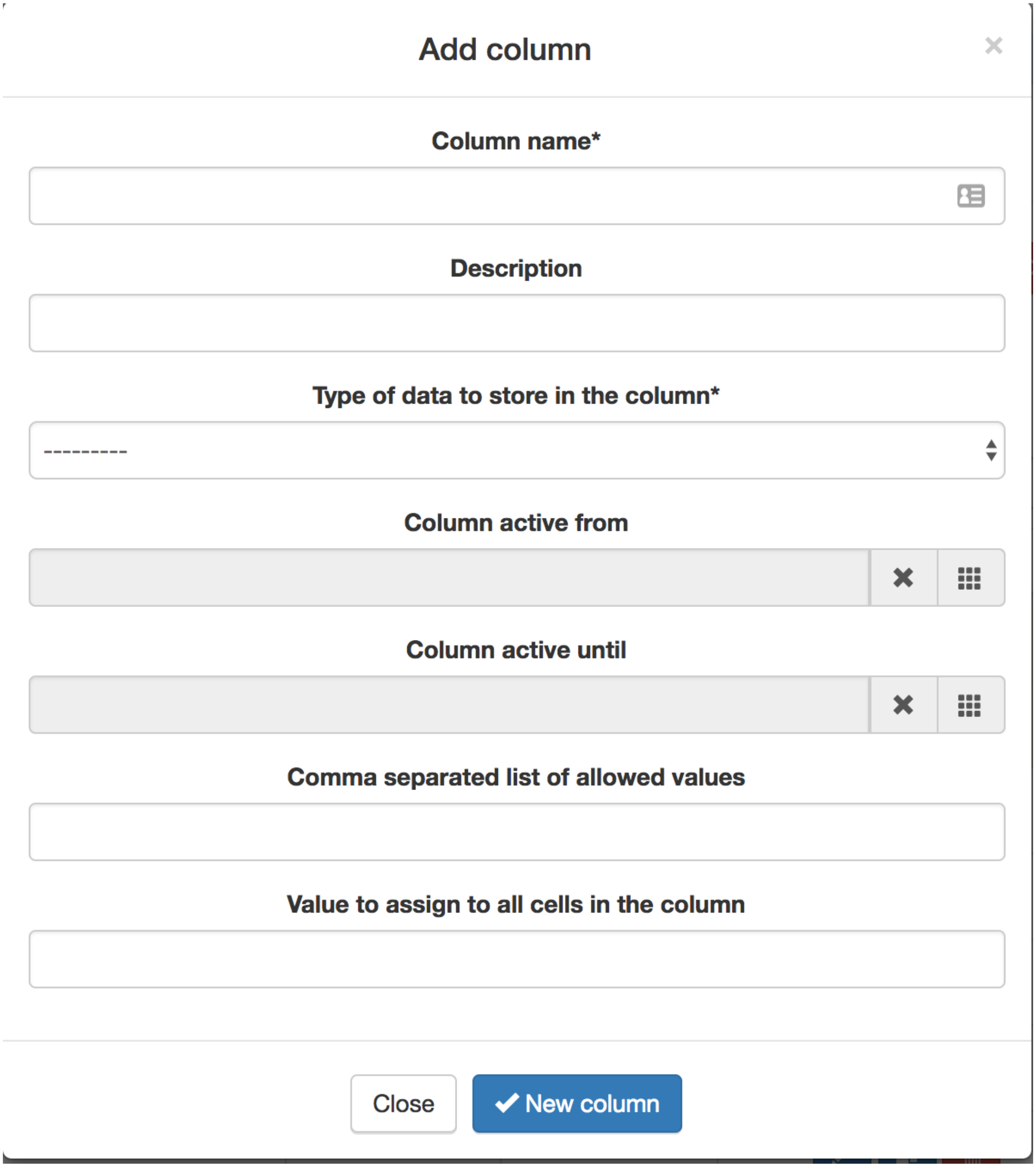
The top of the page now includes links to additional menus with operations to perform over the workflow (some of them will be available depending on your user profile). Under the top bar the application shows the name of the workflow selected for manipulation. Under the title *Workflow Details* there are buttons (available depending on your profile and ownership of the workflow) that offer the following operations:

Add a new column Opens a dialog to create a new column in the table. It requires the name, type of column, a comma separated list of allowed values (optional) and a value to assign to all cells in the column (optional).

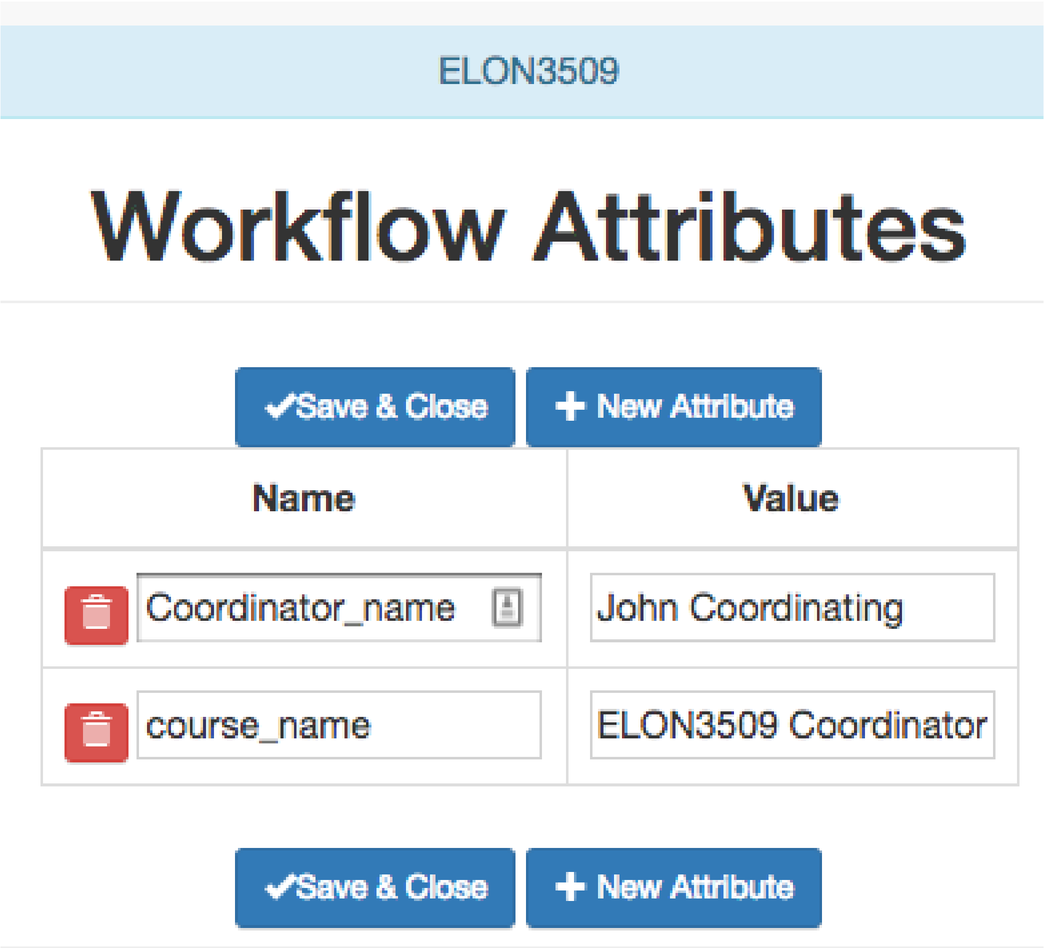
Attributes This is simply a dictionary of pairs `(name, value)` so that whena `name` appears in a personalised text, it is replaced by the `value`. The main use of these attributes is when a value has to appear in various locations and you may want to change all its occurrences. For example, the instructor name could be included as one of the attributes so that if it changes, modifying the attribute is the only required step.

Share A screen to make the workflow accessible to other users. You are supposed to know the user identification (there is no search functionality available).

Export This functionality allows you to take a snapshot (or a picture) of the content of the workflow and store it in a file for your records. You may select which actions are included in the exported file


Add column
×
Column name*
Description
Type of data to store in the column*

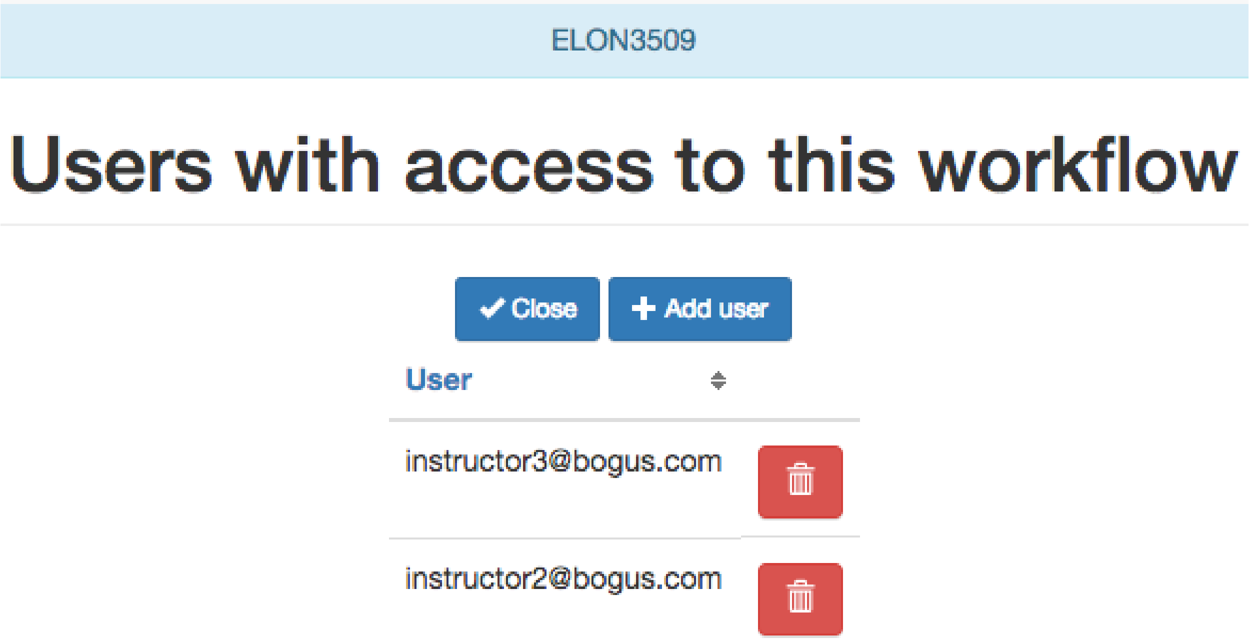
Column active from
Column active until
Comma separated list of allowed values
Value to assign to all cells in the column
Close
✓ New column


✓Save & Close
+ New Attribute
✓Save & Close
+ New Attribute

ELON3509

Workflow Attributes

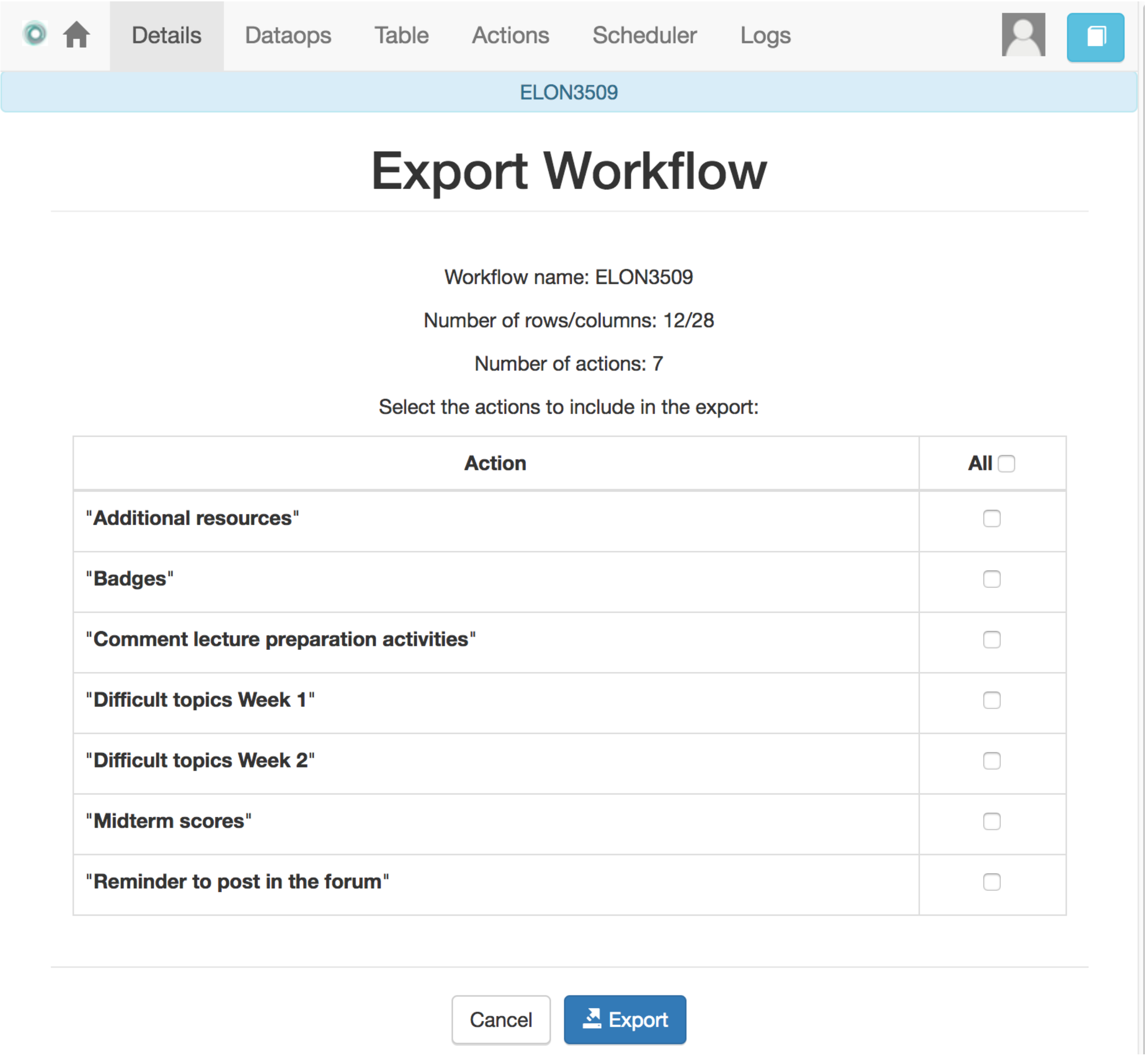
Name	Value
Coordinator_name	John Coordinating
course_name	ELON3509 Coordinator


✓ Close
+ Add user

ELON3509

Users with access to this workflow

User
instructor3@bogus.com
instructor2@bogus.com



Details
Dataops
Table
Actions
Scheduler
Logs
ELON3509

Export Workflow

Workflow name: ELON3509

Number of rows/columns: 12/28

Number of actions: 7

Select the actions to include in the export:

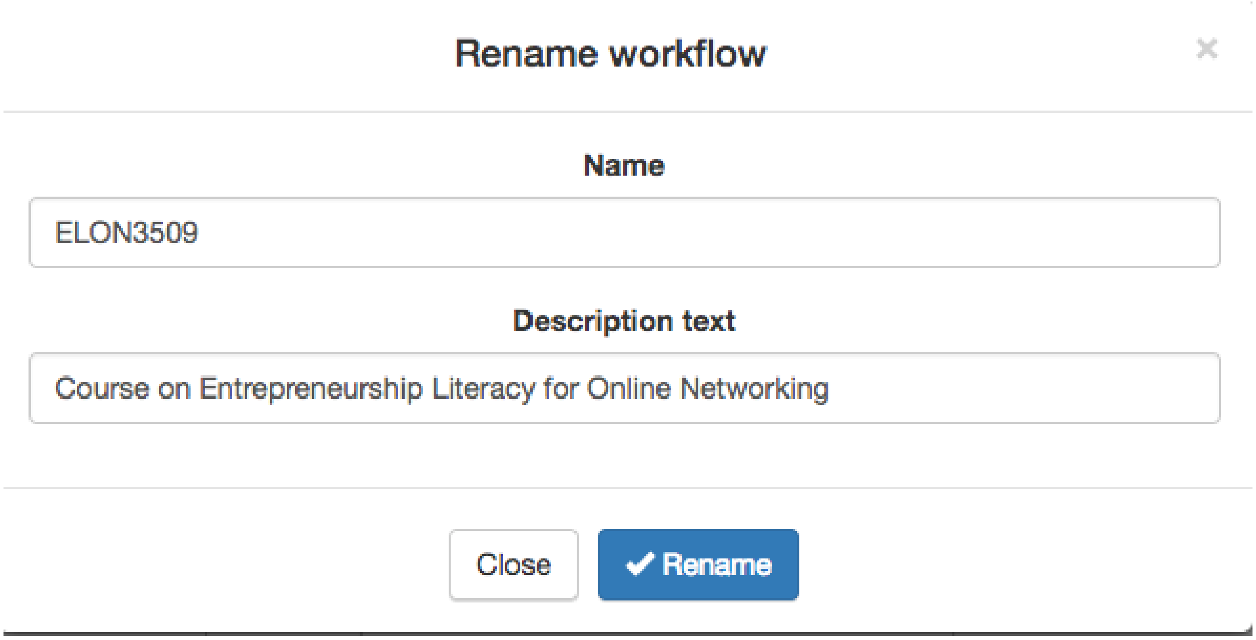
Action	All ☐
"Additional resources"	☐
"Badges"	☐
"Comment lecture preparation activities"	☐
"Difficult topics Week 1"	☐
"Difficult topics Week 2"	☐
"Midterm scores"	☐
"Reminder to post in the forum"	☐

Cancel
Export

The menu offers you the possibility of exporting only the data, or the data **and** the *action* in the workflow.

Clone This function creates a new workflow in the platform by duplicating the data, actions and conditions of the current workflow.

Rename This functionality allows to change either the name or the description of the workflow.



Rename workflow

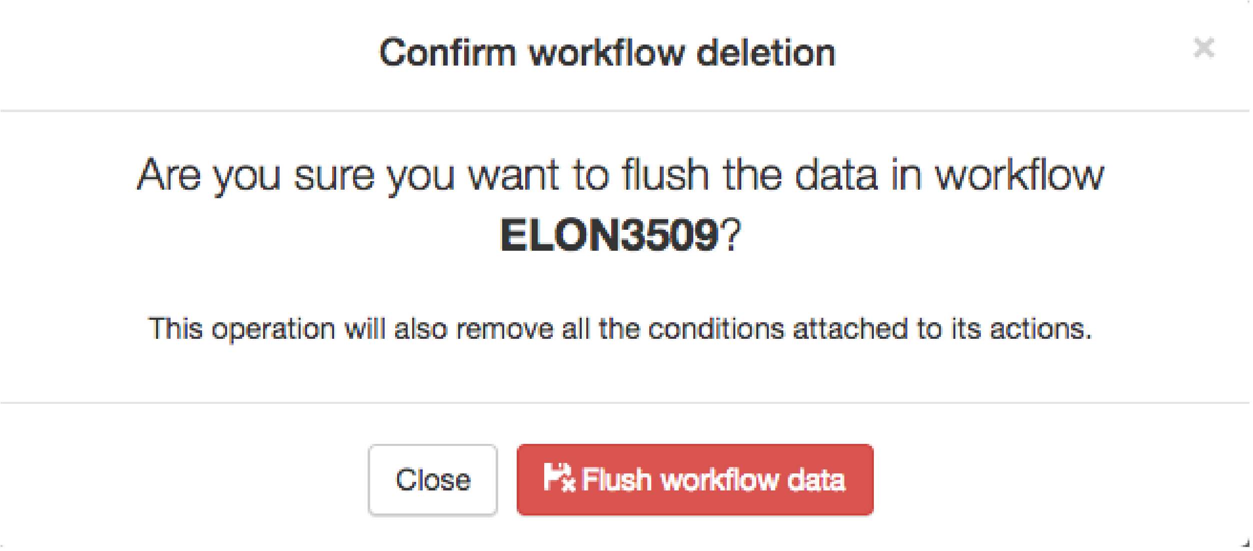
Name

ELON3509

Description text

Course on Entrepreneurship Literacy for Online Networking

Flush data This operation deletes all the data attached to the workflow, but preserves the workflow structure (that is, the name and the description only).



Confirm workflow deletion

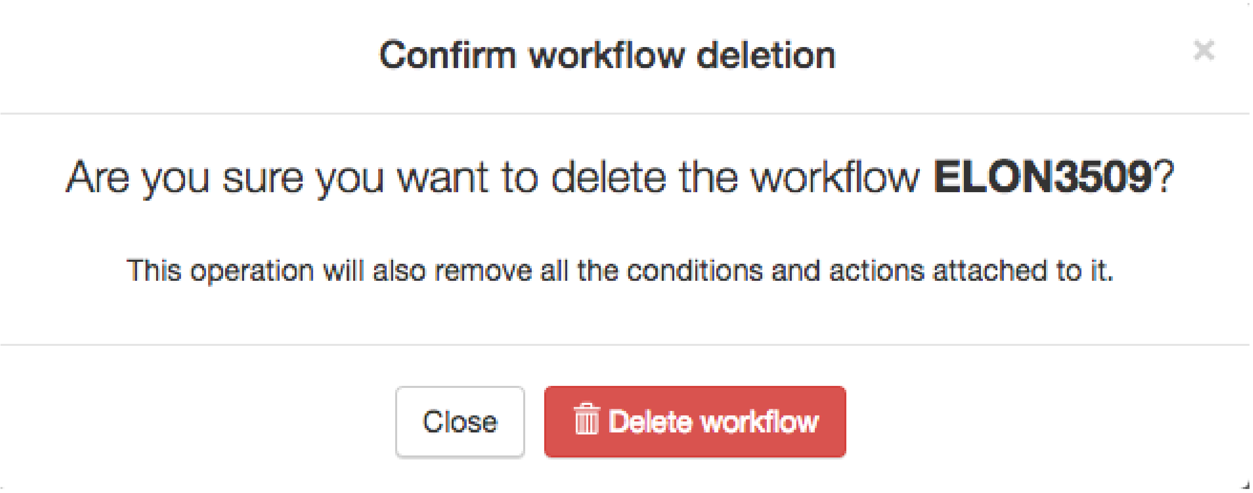
Are you sure you want to flush the data in workflow **ELON3509**?

This operation will also remove all the conditions attached to its actions.

Given the destructive nature of this operation the platform requires you to confirm this step.

Delete Operation similar to the previous one, but now the whole workflow is deleted and therefore un-selected. If executed, the platform will go back to the list of workflows as this one is no longer available for operations.

As in the previous case, the platform asks for confirmation before carrying out the delete operation.


Confirm workflow deletion

Are you sure you want to delete the workflow **ELON3509**?

This operation will also remove all the conditions and actions attached to it.

Under the buttons to carry out these workflow operations the platform shows a summary of the information contained in the workflow.

3.2.1 The Columns

The data in a workflow is stored in a structure called *a table* that is made of rows and colums (similar to a spreadsheet). The information about the columns is included in a table in the workflow details page.

Each column as a name (has to start with a character and can only have _, letters and numbers), a type (one of integer, string, double, boolean or datatime), a field stating if the values of that column are uniques for the rows, and two operations. When a column is marked as *Unique*, it means that all the values it contains are different and unique for each row. Think of a column containing a passport number. Such number is different for every person. There could be several columns with this property. The application detects automatically this property in a column. You may edit and change this properly as long as the values are the adequate ones (they satisfy the uniqueness property if you try mark a column as unique. The operations avaiable over columns are:

Edit It allows you to change the name, type, unique and values allowed in the column. If you are changing the column type, the application will check if the existing values are valid. If not, the change will not be allowed. Similarly, if the *Unique* property is selected, the application checks the values to make sure this property is satisfied.

The column may also have a *validity window* defined by two date/times. This validity is used when executing *action in* tasks.

Clone Clones the column in the workflow chaning its name adding the prefix *Copy of* to the name.

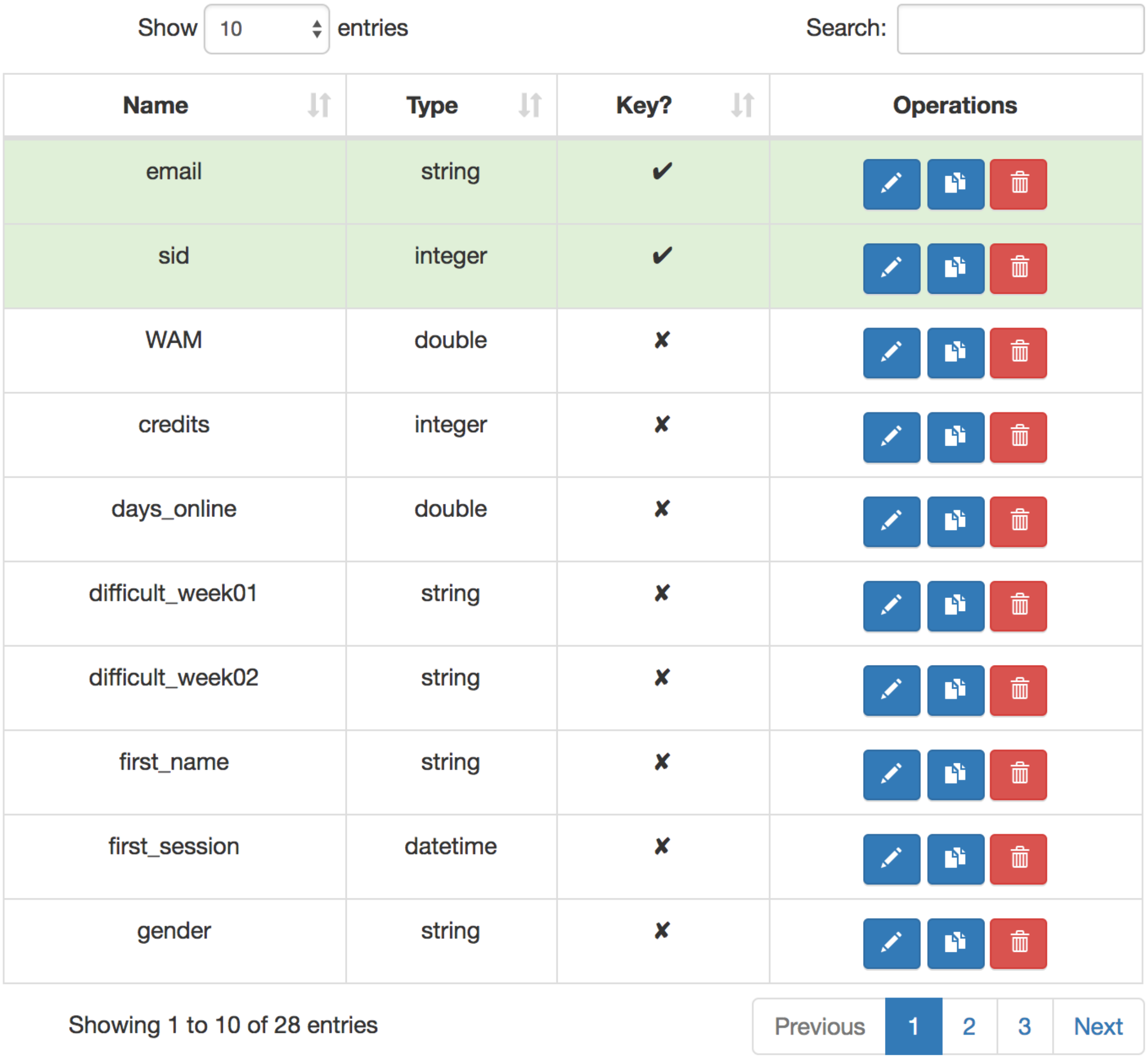
Delete Deletes the column from the workflow. If there are conditions inside *actions out* that use this column, those conditions will be removed from the action.

3.3 Data Operations

> “May be stories are are just data without a soul” – Brené Brown

This section describes one of the most advanced functionality of the platorm, how to upload data into the table. It may be the case that this task is already done, or it is done automatically before you use the workflow. If this is the case, you may skip this section.

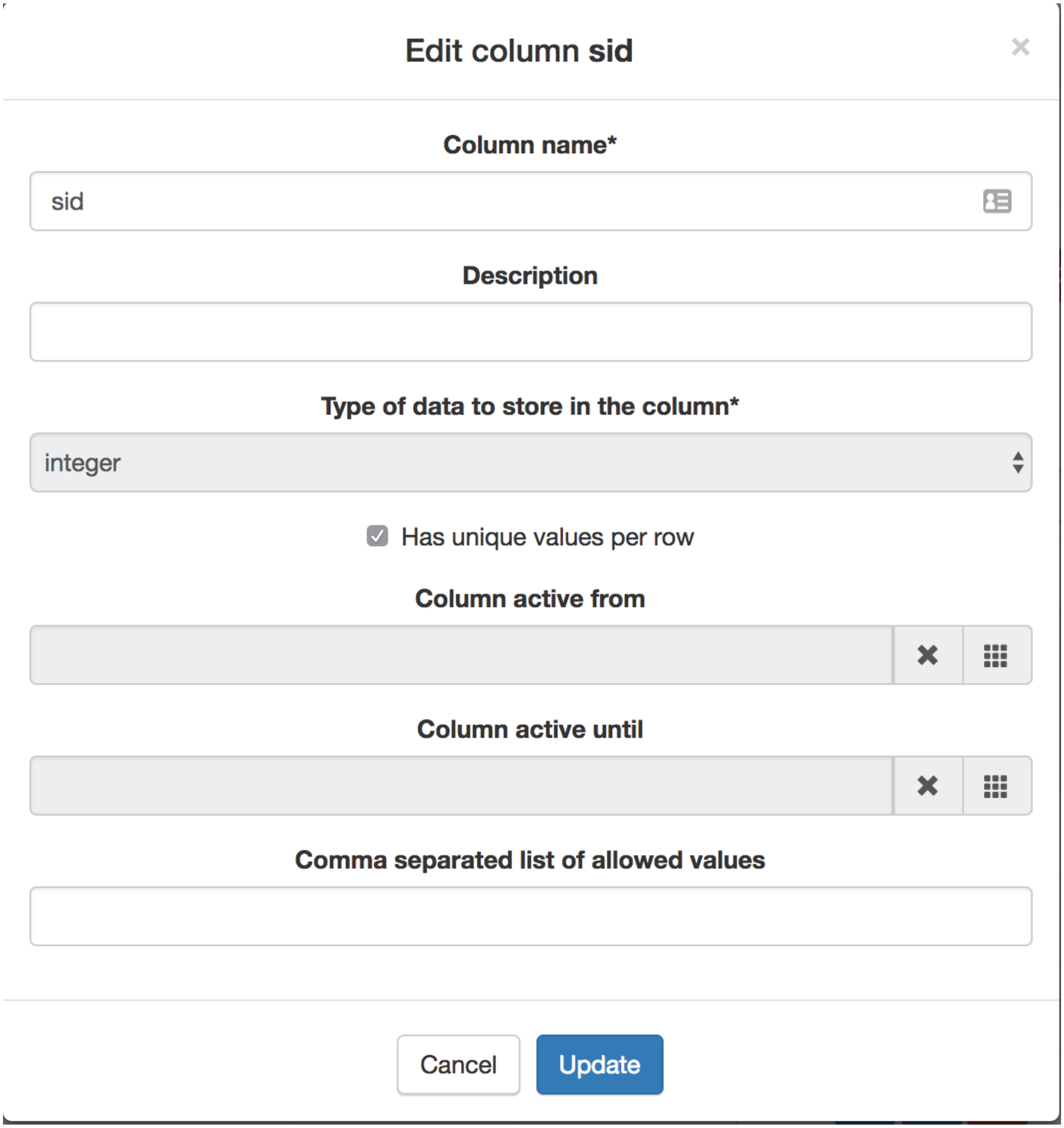
The data operations page offers various options to upload data to the table. It follows a brief description of each one of them with a discussion of the *merge* operation



Show 10 entries Search:

Name	Type	Key?	Operations
email	string	✔	✎ ⎘ 🗑
sid	integer	✔	✎ ⎘ 🗑
WAM	double	✘	✎ ⎘ 🗑
credits	integer	✘	✎ ⎘ 🗑
days_online	double	✘	✎ ⎘ 🗑
difficult_week01	string	✘	✎ ⎘ 🗑
difficult_week02	string	✘	✎ ⎘ 🗑
first_name	string	✘	✎ ⎘ 🗑
first_session	datetime	✘	✎ ⎘ 🗑
gender	string	✘	✎ ⎘ 🗑

Showing 1 to 10 of 28 entries



Edit column sid

Column name*: sid

Description:

Type of data to store in the column*: integer

☑ Has unique values per row

Column active from:

Column active until:

Comma separated list of allowed values:

3.3.1 CSV Files

CSV or "comma separated value" files are plain text files in which the first line contains a comma-separated list of column names, and every subsequent line contains the values of these columns for each row. It is a popular format to exchange data that can be represented as a table, and it is for this reason that OnTask allows to upload data in this format.

The functionality assumes that you have such file available in your computer and provides a form to upload it to the platform. Upon uploading, OnTask does a preliminary processing of the data and shows a table with the columns detected in the file and a set of options.

For each column detected in the file, the table includes if it has been detected to be unique, its automatically detected type, a box to select, the name, and an alternative name (to allow column renaming). This step is to allow you to select those columns that are relevant and discard the rest. The platform requires you to choose **at least** one column with unique values.

Once you selected these values, a new workflow is created with the data from the CSV file.

3.3.2 Merge Operation

Merge a.k.a "Join"

Merging is actualy quite common in databases and is known as a *join* operation. There are several variants of join operations depending how the differences between the key columns are handled. These same variants exist when combining columns in data frames (or a table).

A merge operation is needed when you want to *merge* a set of columns with an **already existing table**. This operation is very common in data science contexts. One of the problems is to specify how the values in the columns are *matched* with respect to the ones already existing in the table. In other words, each new column has a set of values, but they need to be ordered in the right way so that the information is matched appropriately for every row. The solution for this problem is to include in both the existing table and the new data being merge a **unique or key column**. These two columns are used to compare the values, identify the matching row, and make sure the right rows are merged.

When uploading a CSV file in a workflow that already contains data, the platform automatically detects it and executes a *merge* operation. The first step is very similar to a regular update and requires you to select the columns that will be considered for the merge.

However, the difference with this step is that the columns selected will be *merged* with the existing ones using a given unique column. If no such column is selected the application will not proceed with the merge. As in the case of an initial CSV upload, you main change the names of the columns.

The next step is the most delicate one in a merge. Ir requires you to identify the unique columns in both the existing data table and the one being uploaded, the criteria to merge the rows, and how to deal when column names collide. We discuss each of these parameters in more detail.

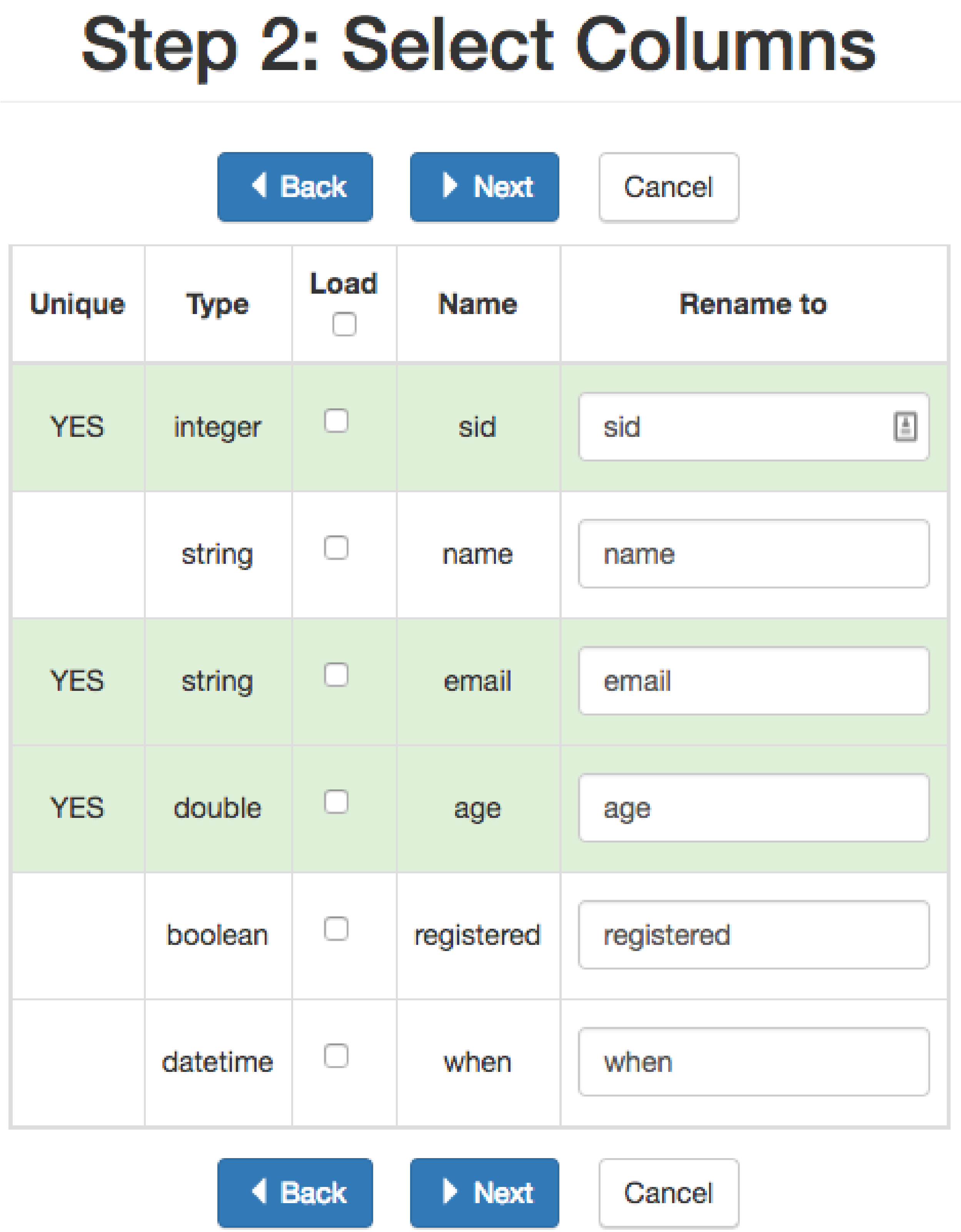
You have to select the pair of unique columns from those in the already existing data and those in the new data about to be merge. These colums are the only choices in the form.

The criteria to merge the rows offers four options:

Inner It will store only the rows for which values in both unique columns are present. Or in other words, any row for which there is no value in either of the key colums **will be dropped**.

Outer The rows that have only one value in one of the key columns will be considered. You have to be carefull with this option because it may produce columns that are no longer unique as a result.

Left Only the rows with a value in the existing table will be considered, the rest will be dropeed.

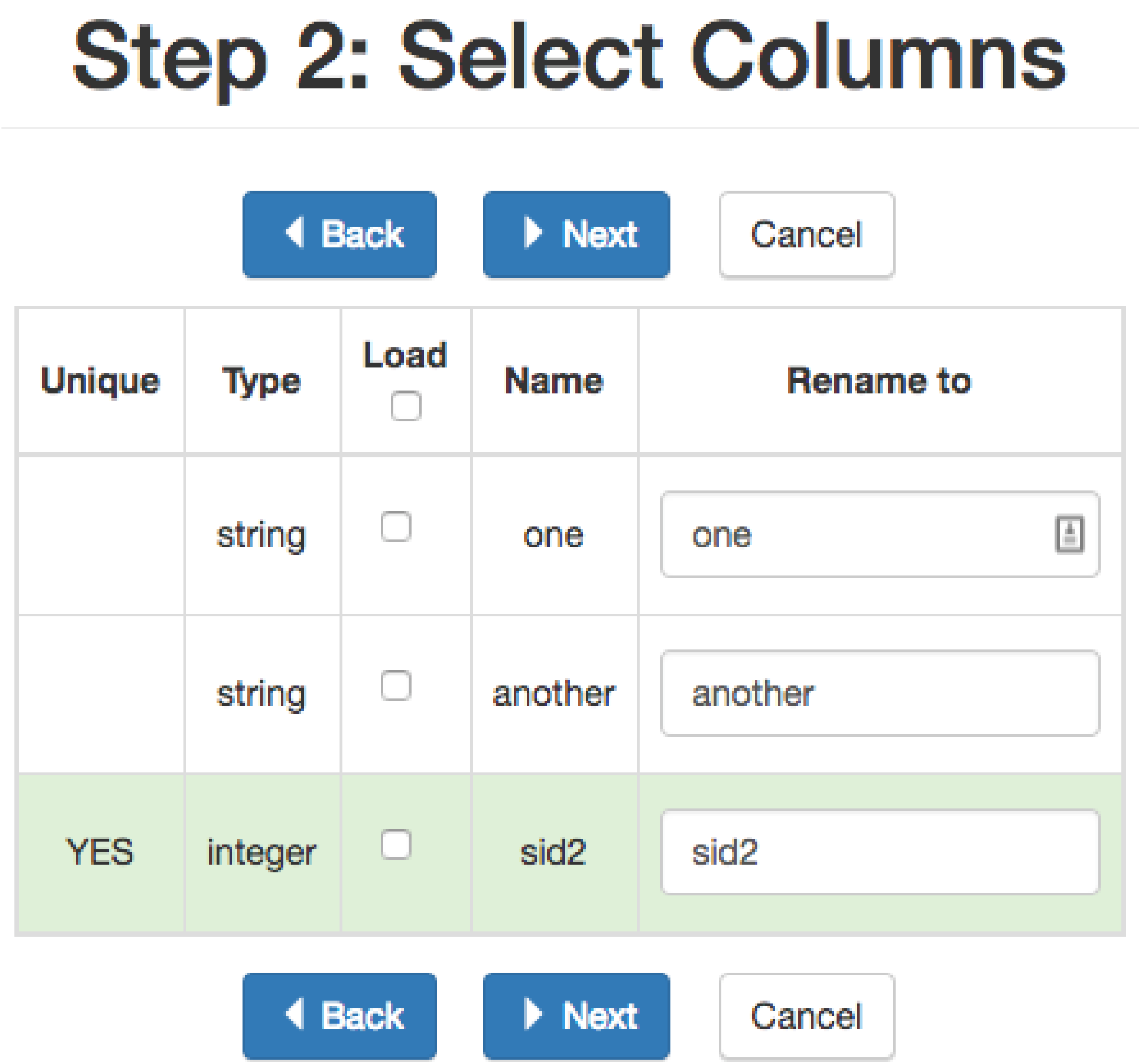


Step 2: Select Columns

Back
Next
Cancel

Unique	Type	Load ☐	Name	Rename to
YES	integer	☐	sid	sid
	string	☐	name	name
YES	string	☐	email	email
YES	double	☐	age	age
	boolean	☐	registered	registered
	datetime	☐	when	when

Back
Next
Cancel

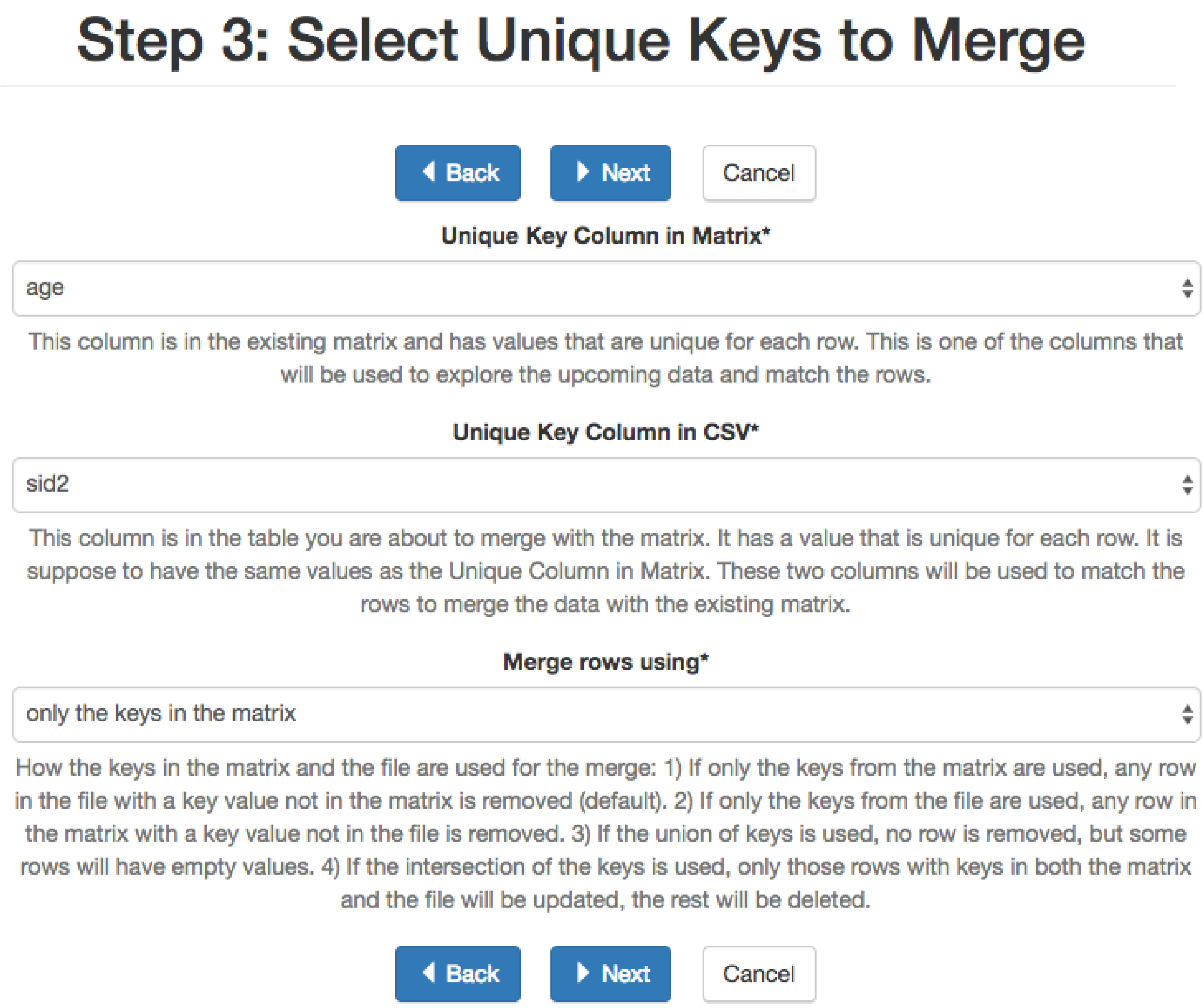


Step 2: Select Columns

Back
Next
Cancel

Unique	Type	Load ☐	Name	Rename to
	string	☐	one	one
	string	☐	another	another
YES	integer	☐	sid2	sid2

Back
Next
Cancel



Step 3: Select Unique Keys to Merge

Unique Key Column in Matrix*

age

This column is in the existing matrix and has values that are unique for each row. This is one of the columns that will be used to explore the upcoming data and match the rows.

Unique Key Column in CSV*

sid2

This column is in the table you are about to merge with the matrix. It has a value that is unique for each row. It is suppose to have the same values as the Unique Column in Matrix. These two columns will be used to match the rows to merge the data with the existing matrix.

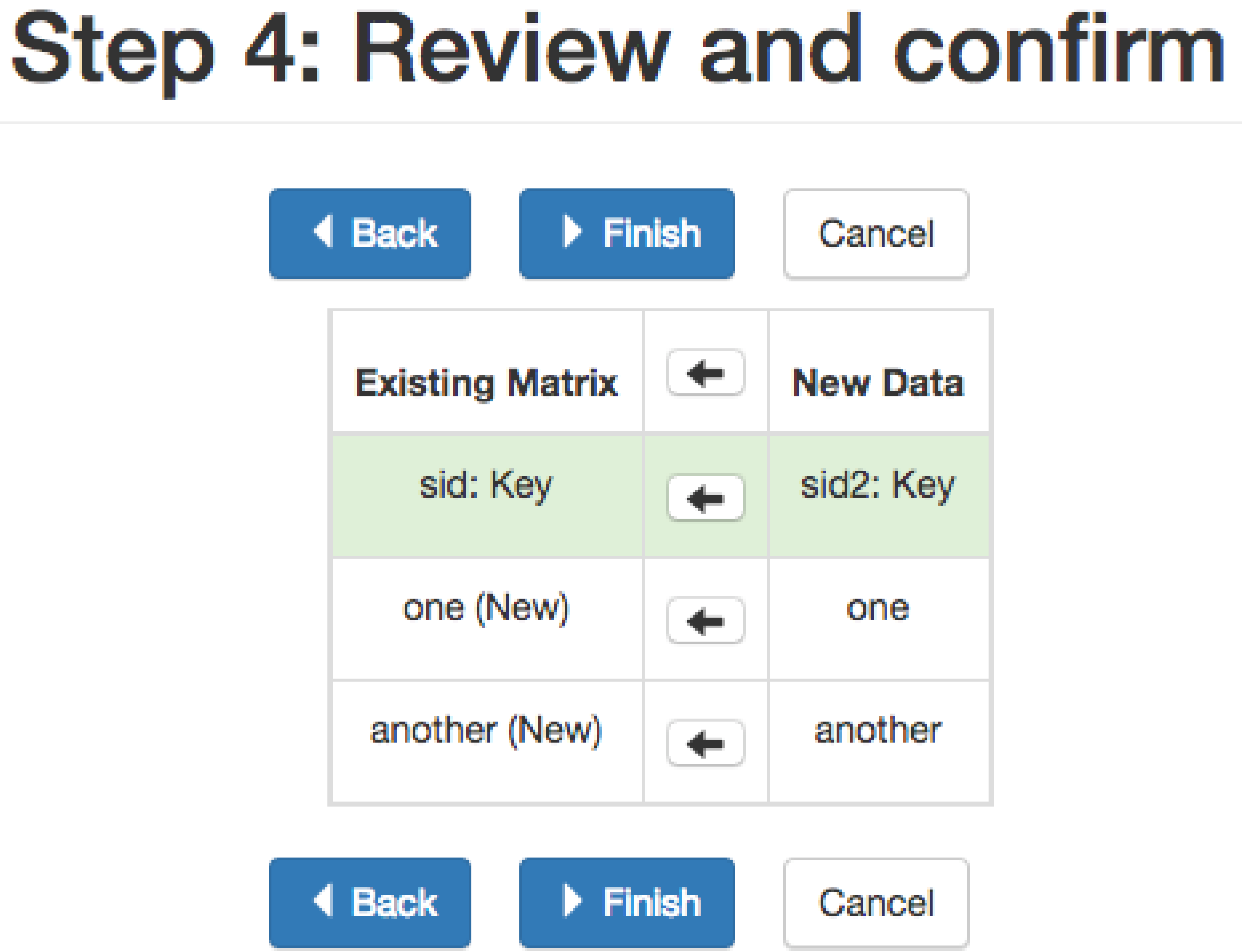
Merge rows using*

only the keys in the matrix

How the keys in the matrix and the file are used for the merge: 1) If only the keys from the matrix are used, any row in the file with a key value not in the matrix is removed (default). 2) If only the keys from the file are used, any row in the matrix with a key value not in the file is removed. 3) If the union of keys is used, no row is removed, but some rows will have empty values. 4) If the intersection of the keys is used, only those rows with keys in both the matrix and the file will be updated, the rest will be deleted.

Right Only the rows with a value in the table being uploaded will be considered, the rest will be dropeed.

You have to take extra care when performing this operation as it may destroy part of the existing data. In the extreme case, if you try to merge a table with a key column with no values in common with the existing key and you select the *inner* method, you may end up with an empty table. After selecting these parameters the platform will show you what it will happen with the various columns involved.



Step 4: Review and confirm

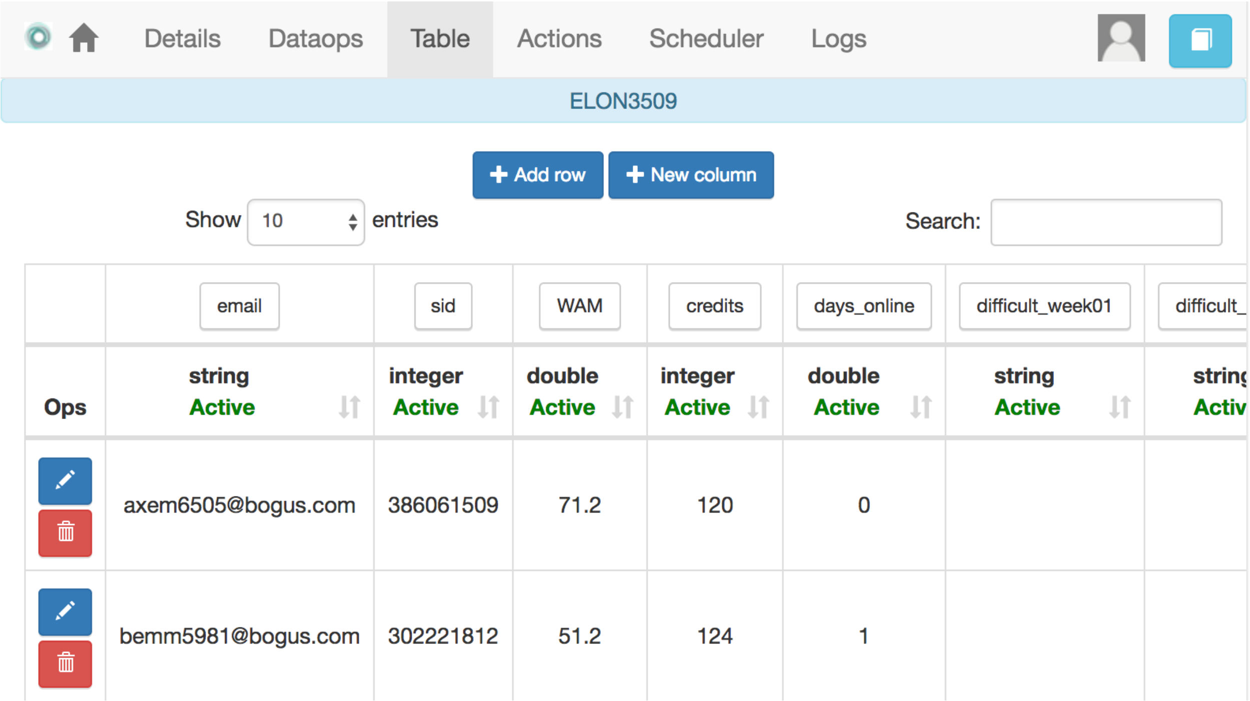
Existing Matrix	←	New Data
sid: Key	←	sid2: Key
one (New)	←	one
another (New)	←	another

3.4 The Table

> "You're here because you know something. What you know you can't explain, but you feel it" – Morpheus, The Matrix

This functionality of the application is basically to provide some basic visualisation of the values stored in the workflow. Due to the potential large size of this table in either number of rows or columns, the functionality is reduded simply to browse values, search, create/update/delete a row, or add a new column.

As in the case of *the columns*, the table is shown broken into pages (you may choose the number of entries per page in the upper left side of the table) and allows to search for values. The icons in the left side of each row allow you to edit any of the values or delete a row entirely. If the *Add row* operation is selected a form with one field per column is shown. The values entered in this form will be checked to verify that the unique key property of the columns is preserved.



	email	sid	WAM	credits	days_online	difficult_week01	difficult_
Ops	string Active	integer Active	double Active	integer Active	double Active	string Active	strin Activ
	axem6505@bogus.com	386061509	71.2	120	0		
	bemm5981@bogus.com	302221812	51.2	124	1		

3.5 The Actions

> "In order to carry a positive action we must develop here a positive vision" – Dalai Lama

This is the most important functionality of the platform. Actions are exchange of information with the learners. There are two types: actions in, and actions out. A workflow contains an arbitrary number of these actions. The *action* page shows a table with the actions in the selected workflow.

Similarly to the case of the workflow, each action is identified by a name (unique) and an optional description. There are several operations supported for each action (right most column in the table shown in the previous figure).

3.5.1 Actions In

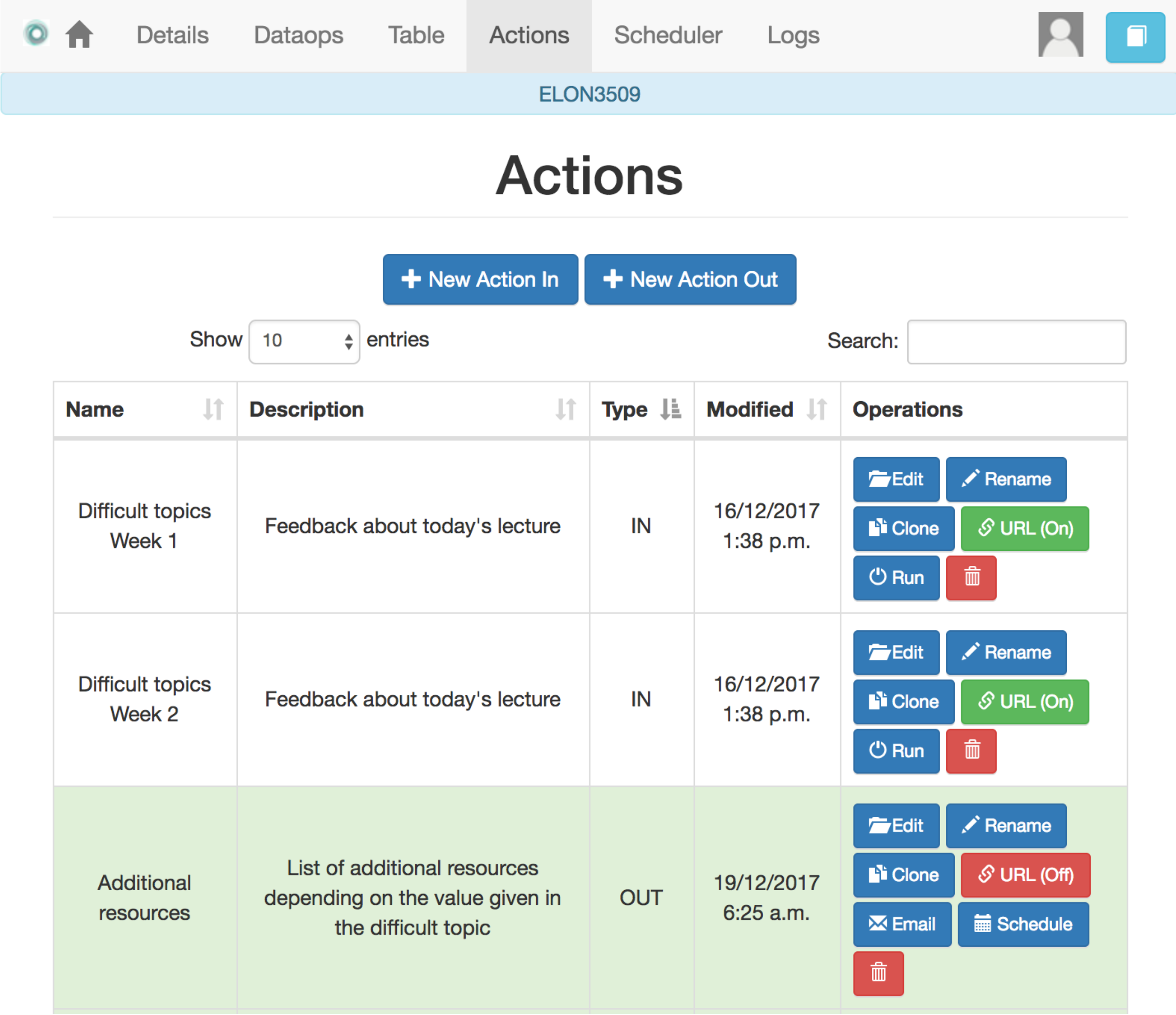
These actions allow you either instructors or students to introduce information in the table stored in a workflow. When providing information, you typicially are interested in a subset of the rows (some of the learners) and a subset of the columns (some of the factors). For example, you would like to check if a group of students is attending one face-to-face session, or if a group of students is engaging in a project discussion.

These actions are edited using the screen shown in the following figure:

The page has two elements. The first one is a filter to restrict the rows in the table considered for data entry. You create a condition, and those rows that satisfy it are then prepared for data entry. The second part of the screen is to select those columns that you want to include in the data entry form. You must select at leaset a *Unique* column (shown with green background) and a non-unique column.

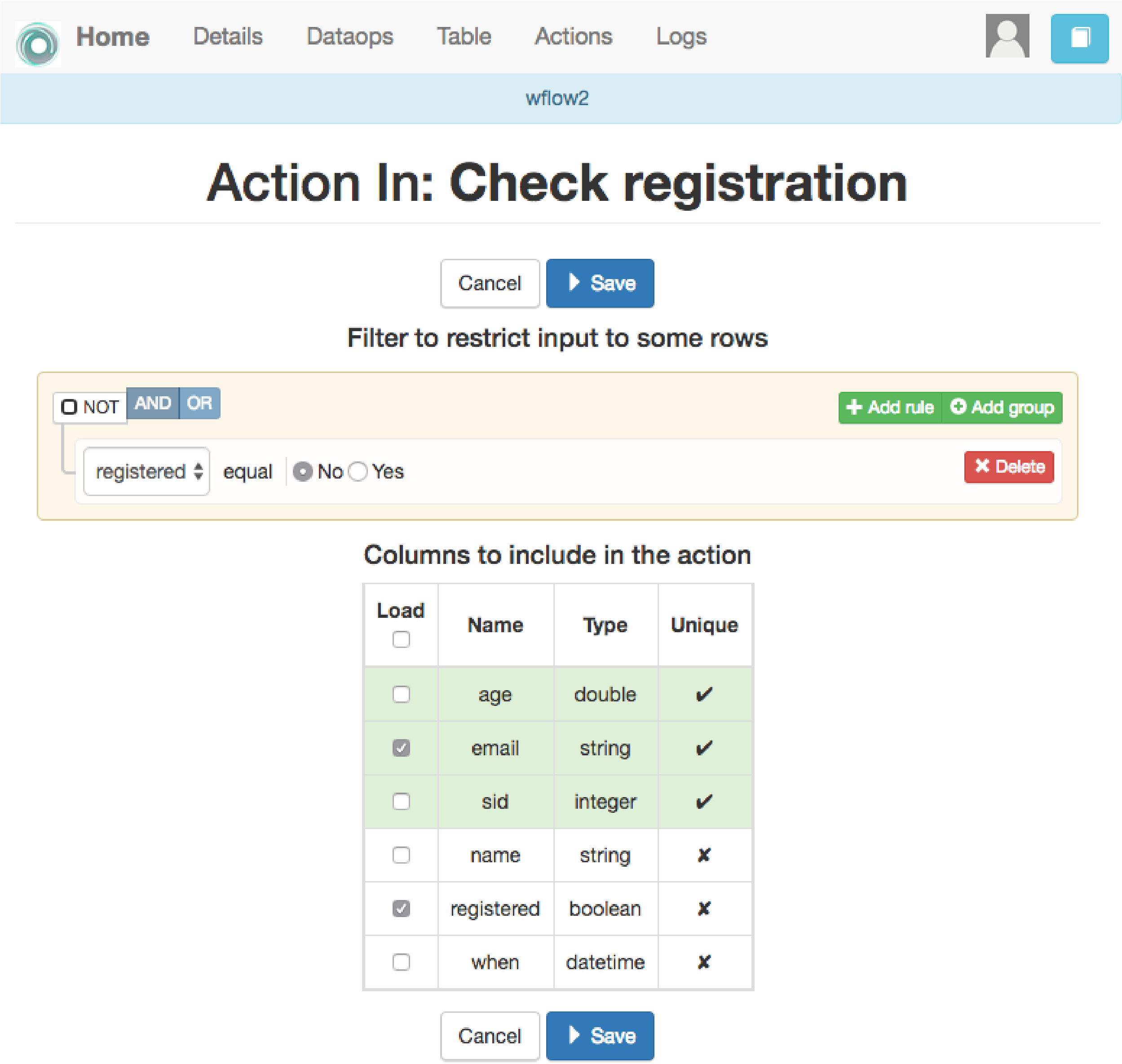
Once an *Action In* has been selected, there are two operations available represented by the buttons with labels *Run* and *URL*. The *Run* operation is intended for the instructors to enter the data for a set of learners. After clicking the link the platform shows a table with the data about the learners. The table has a search box in the upper left corner to quickly find a person as ilustrated in the next figure.

An instructor may click in the link available in the right column and it is offered the possibility of modifying the information in the pre-selected columns for that learner.


Details
Dataops
Table
Actions
Scheduler
Logs
ELON3509
New Action In
New Action Out
Show 10 entries
Search:

Actions

Name	Description	Type	Modified	Operations
Difficult topics Week 1	Feedback about today's lecture	IN	16/12/2017 1:38 p.m.	Edit Rename Clone URL (On) Run
Difficult topics Week 2	Feedback about today's lecture	IN	16/12/2017 1:38 p.m.	Edit Rename Clone URL (On) Run
Additional resources	List of additional resources depending on the value given in the difficult topic	OUT	19/12/2017 6:25 a.m.	Edit Rename Clone URL (Off) Email Schedule


Home
Details
Dataops
Table
Actions
Logs
wflow2
Cancel
Save
☐ NOT
AND
OR
Add rule
Add group
Delete
Cancel
Save

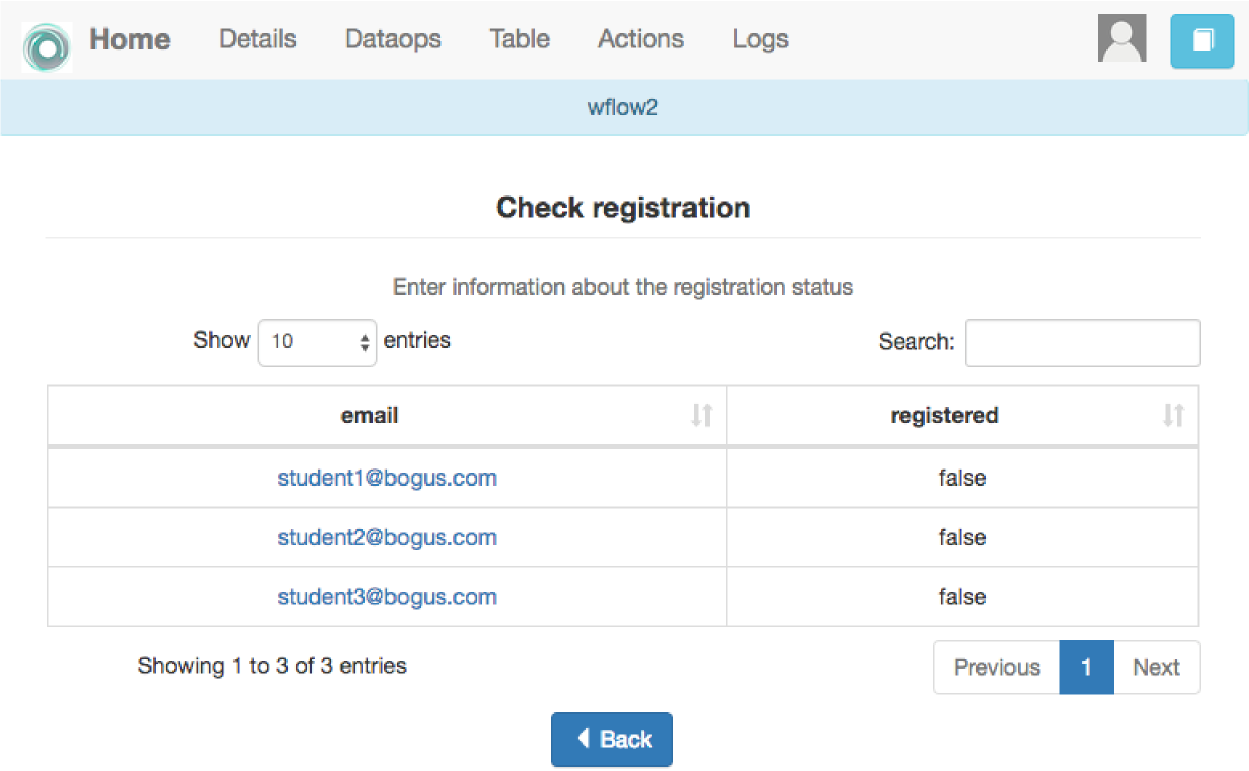
Action In: Check registration

Filter to restrict input to some rows

registered equal ● No ○ Yes

Columns to include in the action

Load ☐	Name	Type	Unique
☐	age	double	✔
☑	email	string	✔
☐	sid	integer	✔
☐	name	string	✘
☑	registered	boolean	✘
☐	when	datetime	✘


Home
Details
Dataops
Table
Actions
Logs
wflow2

Check registration

Enter information about the registration status

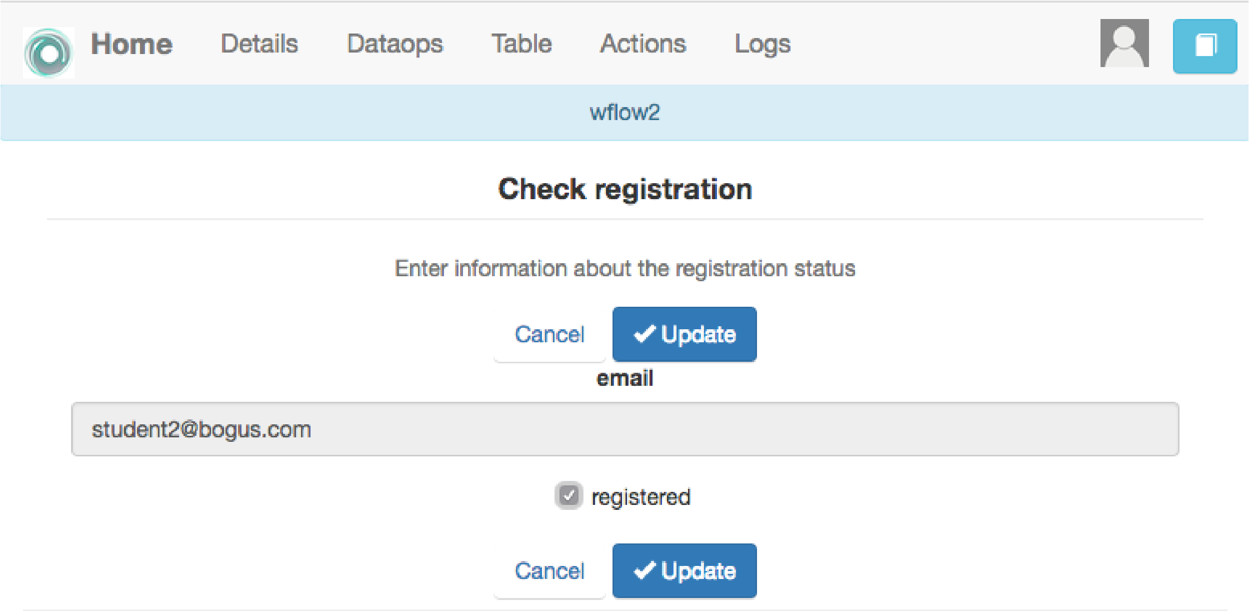
Show 10 entries

Search:

email	registered
student1@bogus.com	false
student2@bogus.com	false
student3@bogus.com	false

Showing 1 to 3 of 3 entries

Previous
1
Next
Back


Home
Details
Dataops
Table
Actions
Logs
wflow2

Check registration

Enter information about the registration status

Cancel
Update

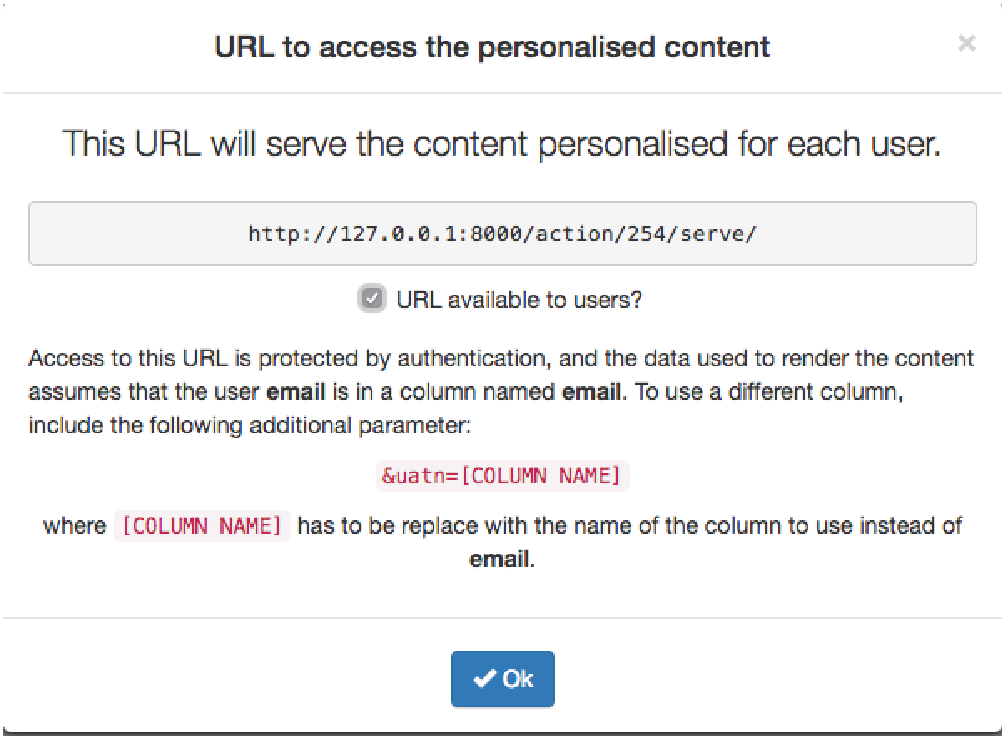
email: student2@bogus.com

☑ registered

Cancel
Update

After entering the information the platform refreshes the list of students for which the data entry is still allowed.

The second operation available for *Actions In* is to make available the URL to learners so that they individually enter the information themselves. If you go back to the table showing all the actions and click in the icon with label *URL* you are given the choice to enable/disable a specific URL for the students to access the data entry screen.



URL to access the personalised content

This URL will serve the content personalised for each user.

http://127.0.0.1:8000/action/254/serve/

☑ URL available to users?

Access to this URL is protected by authentication, and the data used to render the content assumes that the user **email** is in a column named **email**. To use a different column, include the following additional parameter:

`&uatn=[COLUMN NAME]`

where `[COLUMN NAME]` has to be replace with the name of the column to use instead of **email**.

You then may send or make available this URL and, after authentication, students will be able to enter the information requested and the values are automatically stored in the right row and column in the table.

These actions offer an ideal procedure to collect information about any aspect of a course in a way that is centralized and available for further processing. The power of these actions is complemented when combined with *Actions Out*, in which personalized content is made available to the learners.

3.5.2 Actions Out

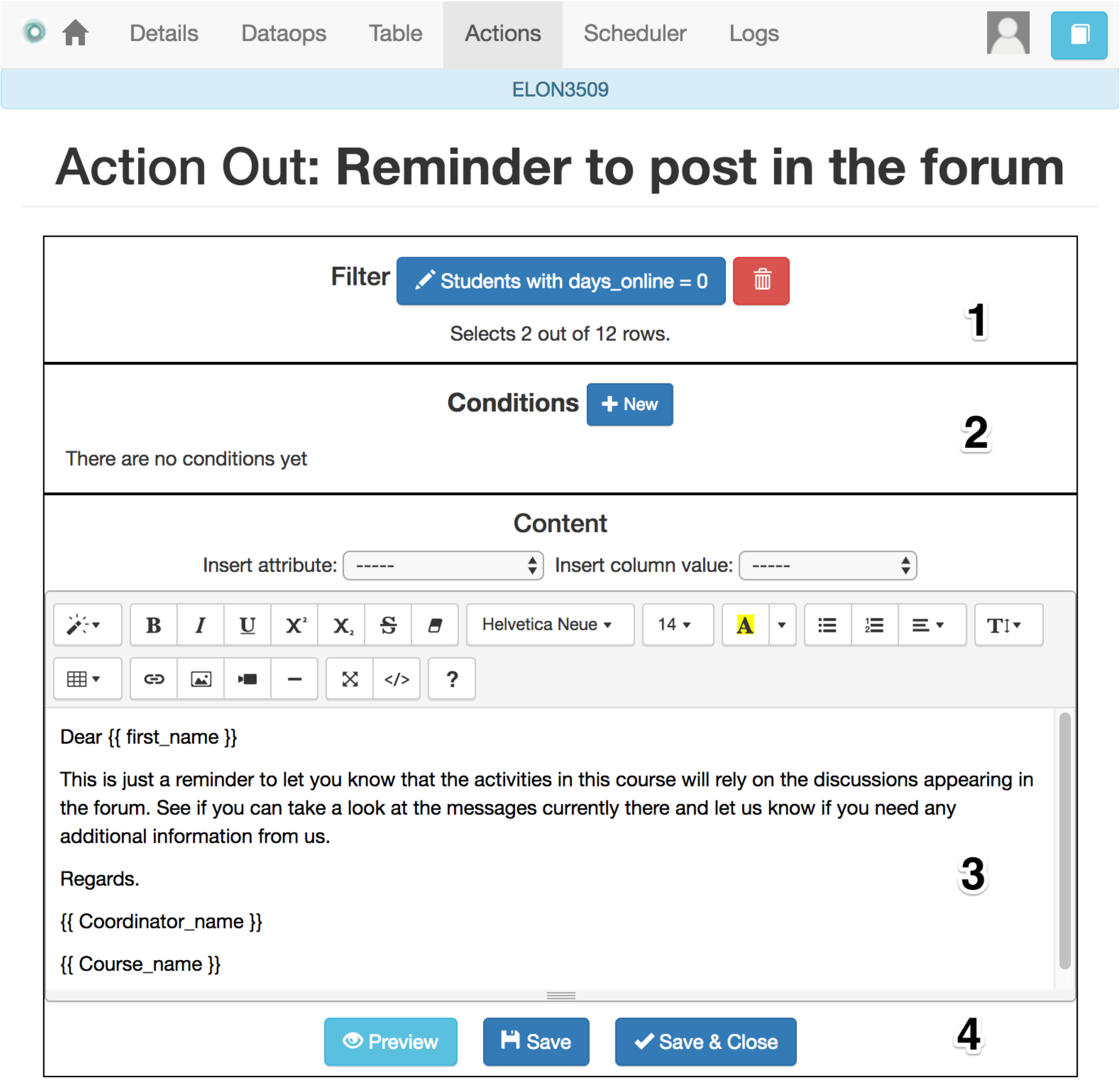
These actions allow to create a resource (in a format close to HTML) and mark certain elements with conditions that will control its appearance in the final view. Think of these *actions out* as a resource (item, message, tip, comment) you would give learners during a experience. You may have several of these items prepared to be used at different points during the expeience. The action is manipulated with the screen shown in the following figure

Before describing in detail the structure of this screen let's digress for a second and explore the concept of *condition*. A condition in OnTask is a Boolean expression, or if you prefer, an expression that when evaluated will return either **True** or **False**. These expressions are commonly used in other applications such as spreadsheets or programming languages. The following image shows an example of this condition.

The Boolean expression is contained under the title **Formula**. The expression can be alternatively read as:

```
Days_online_2 = 0
```

The first element of the expression is the variable `Days_online_2`. The second element is the equal sign, and the third component is the constant zero. The variable `Days_online_2` may be replaced by any value in a procedure we call *evaluation*. So, if the expression is evaluated replacing the variable by the value 3, it results in $3 = 0$ which is


Details
Dataops
Table
Actions
Scheduler
Logs
ELON3509

Action Out: Reminder to post in the forum

Filter: Students with days_online = 0

Selects 2 out of 12 rows.

Conditions + New

There are no conditions yet

Content

Insert attribute: -----
Insert column value: -----

Dear {{ first_name }}

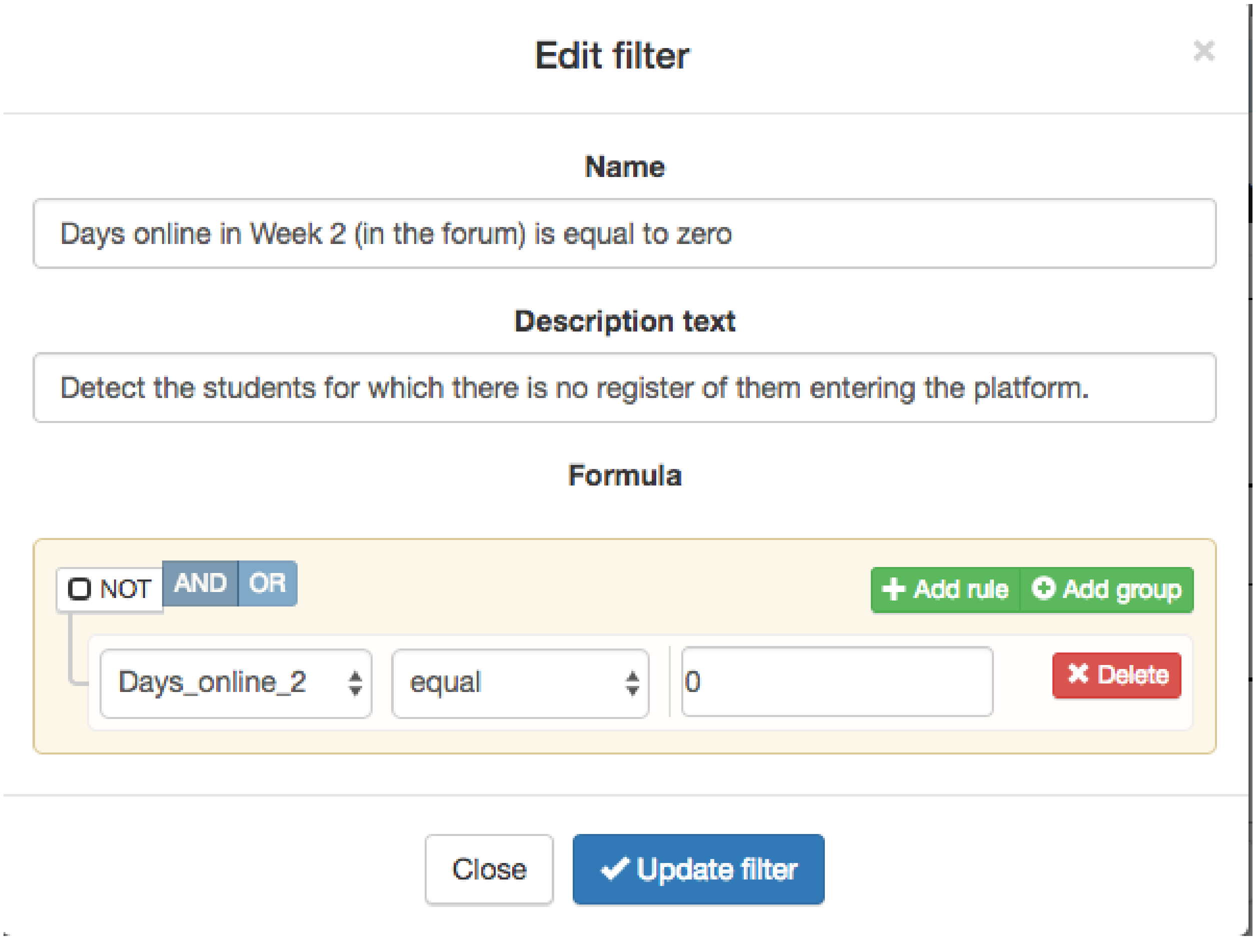
This is just a reminder to let you know that the activities in this course will rely on the discussions appearing in the forum. See if you can take a look at the messages currently there and let us know if you need any additional information from us.

Regards.

{{ Coordinator_name }}

{{ Course_name }}

1
2
3
4
Helvetica Neue
14
Preview
Save
Save & Close



Edit filter

Name

Days online in Week 2 (in the forum) is equal to zero

Description text

Detect the students for which there is no register of them entering the platform.

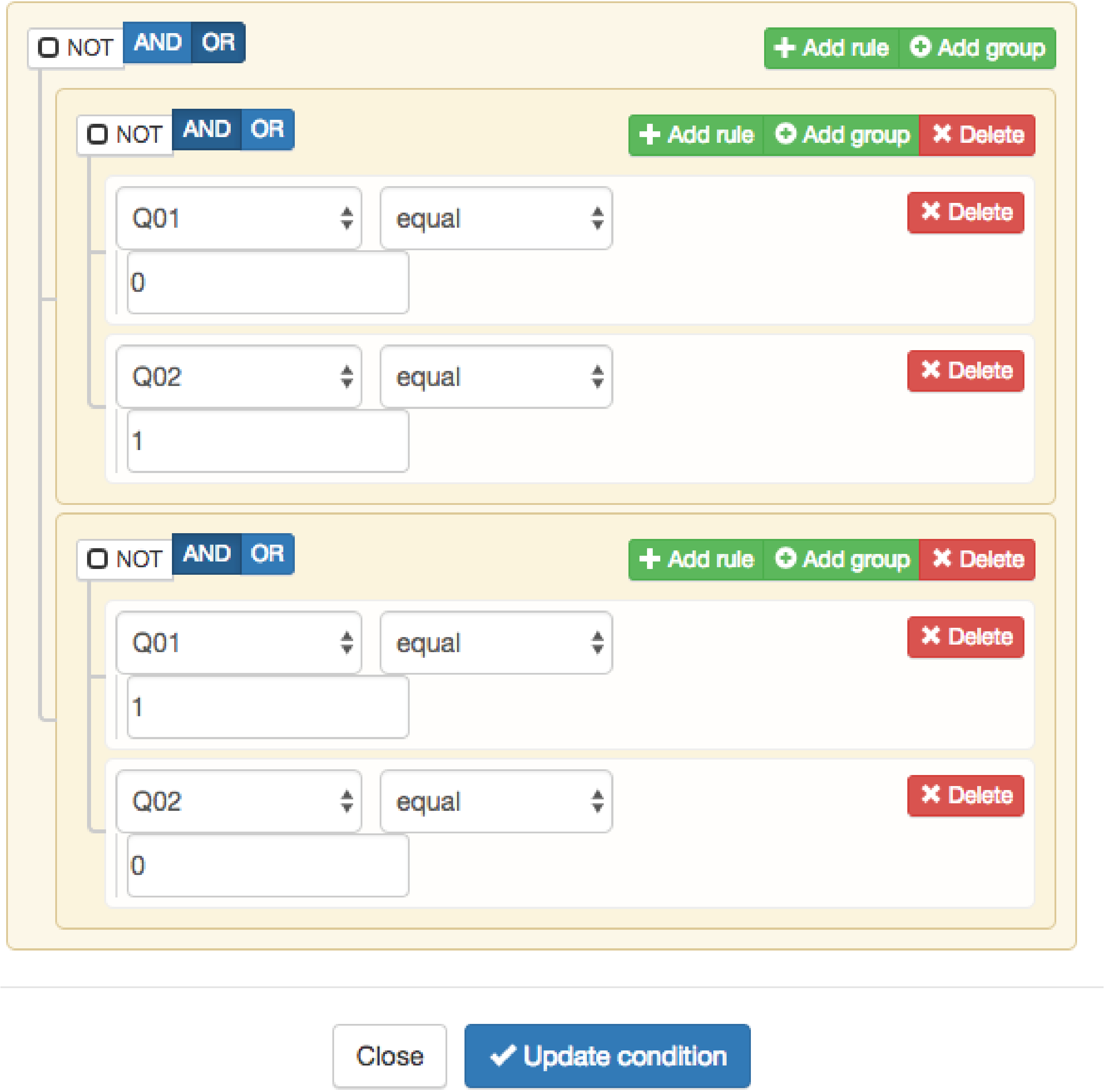
Formula

Days_online_2 equal 0

×
☐ NOT
AND
OR
+ Add rule
Add group
✖ Delete
Close
✔ Update filter

false. Alternatively, if we evaluate the expression replacing `Days_online_2` with the value 0, then the expression becomes $0 = 0$, which is trivally true. With this structure, any expression then is evaluated by replacing the variables by values and deciding if the resulting expression is true or false.

These conditions can have nested sub-expressions and get complex fairly quickly.



Q01 equal 0
Q02 equal 1

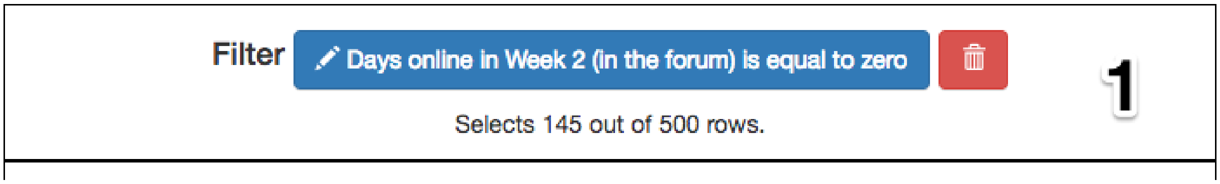
Q01 equal 1
Q02 equal 0

However, the underlying mechanism to evaluate them remains the same: replace variables with values and decide the result (true or false). OnTask relies on these expressions to personalise the content of the actions.

Let's now go back to the screen to edit an action. The area has four components

The filter The top area contains a *filter*. This element is an expression used to decide which table rows will be selected and used with this condition.

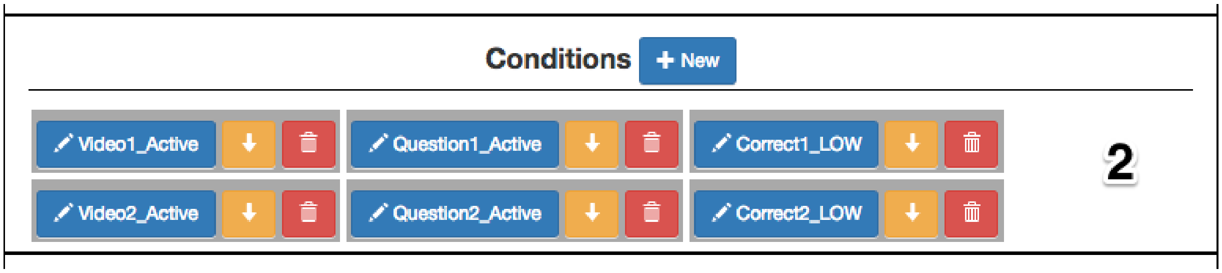
The line below the button to edit the expression states how many table rows satisfy the filter condition (and


Filter
Days online in Week 2 (in the forum) is equal to zero
Selects 145 out of 500 rows.
1

therefore are selected). In practice, this is as if you dropped from the table some of the rows (it is just that they are ignored, not dropped.

The conditions

This is the area immediately below the filter. Each condition allows you to edit its expression (first block with the pencil), use it in the text blow (block with the arrow), or delete it (trash can icon) as shown in the figure below



The HTML text

This is the area to create the personalised document. It is a conventional HTML editor offering the usual functionalities (inserting text in various forms, headings, lists, links, images, etc.) Right above the editor window you have two choice menus that you can use to insert either a *workflow attribute* or a column name that will be replaced by the corresponding value for each row.

The preview/save buttons

The *Save* button saves the content of the text editor, the *Save & Close* saves the content of the text editor and returns to the list of actions, and the *Preview* button shows how the text is rendered for every row in the table.

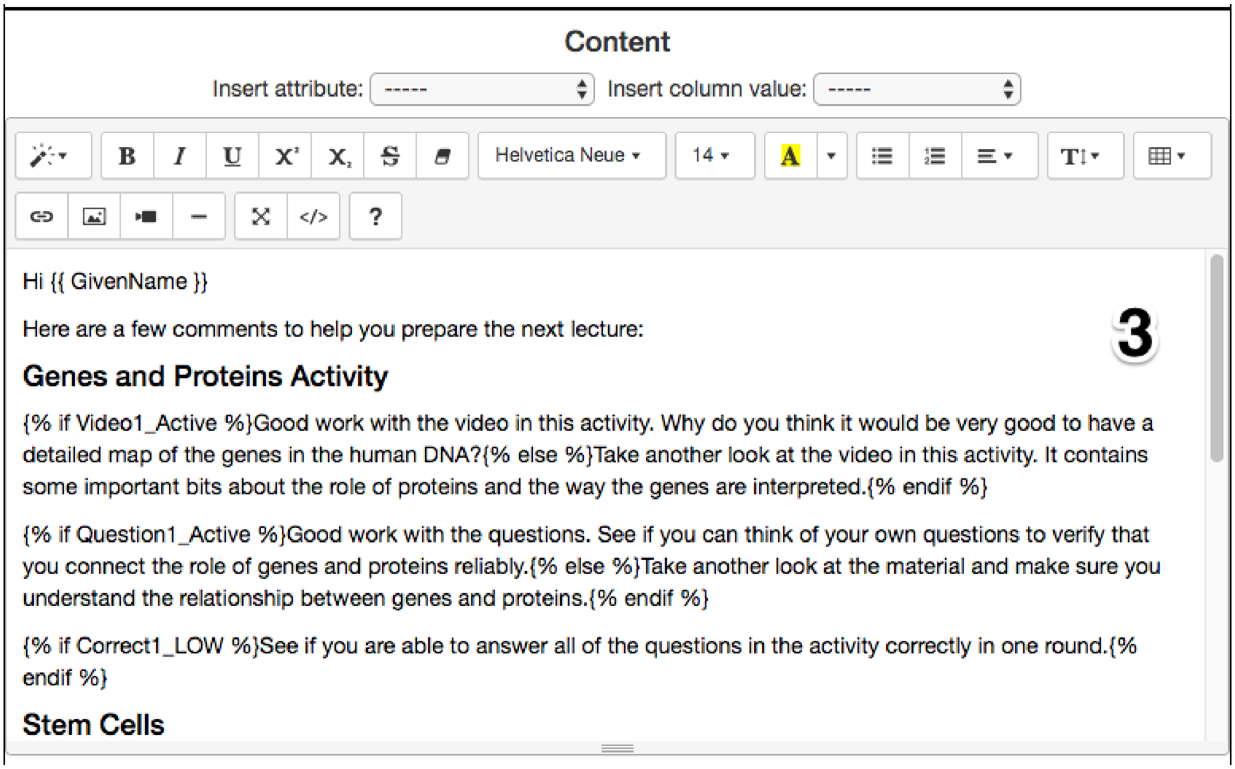
Using column values, attributes and conditions in an Action Out

The purpose of this page is to allow you to create a text in the editor that may include three types of elements that are personalised for each row: an attribute name, a column name or a condition.

To insert an attribute name simply place the cursor in the text editor in the location where you want the value of that attribute to appear. Then, select the attribute from the area above the editor and you will see how the name of the attribute surrounded by double curly braces appears in the text (for example `{{ course_name }}`. Only *the attributes* you previously created in the details page are available.

To insert a column name, you follow the same steps but this time you select one of the elements next to the text *Insert column value*. Place the cursor in the location in which you want that value to appear, select the column name from the pull-down menu, and the name appears in the text surrounded by double curly braces (for example `Hi {{ GivenName }}`.

These two elements will be included in the text with the corresponding values (the same for all rows in the case of the attribute, and the value of the corresponding row in the case of the column name. Inserting a condition is different.



Hi {{ GivenName }}

Here are a few comments to help you prepare the next lecture:

Genes and Proteins Activity

{% if Video1_Active %}Good work with the video in this activity. Why do you think it would be very good to have a detailed map of the genes in the human DNA?{% else %}Take another look at the video in this activity. It contains some important bits about the role of proteins and the way the genes are interpreted.{% endif %}

{% if Question1_Active %}Good work with the questions. See if you can think of your own questions to verify that you connect the role of genes and proteins reliably.{% else %}Take another look at the material and make sure you understand the relationship between genes and proteins.{% endif %}

{% if Correct1_LOW %}See if you are able to answer all of the questions in the activity correctly in one round.{% endif %}

Stem Cells

Highlight the text in the editor and then click in the arrow of one of the conditions. The text will be surrounded by two marks. For example if the condition name is `Video_active`, the text in the editor will appear as:

> {% if Video_active %}Good work with this week's video{% endif %}

This format states that the message *Good work with this week's video* should appear only if the condition `Video_active` is true. If not, the text should be ignored. The following figure illustrates this process.

Previewing the content of an Action Out

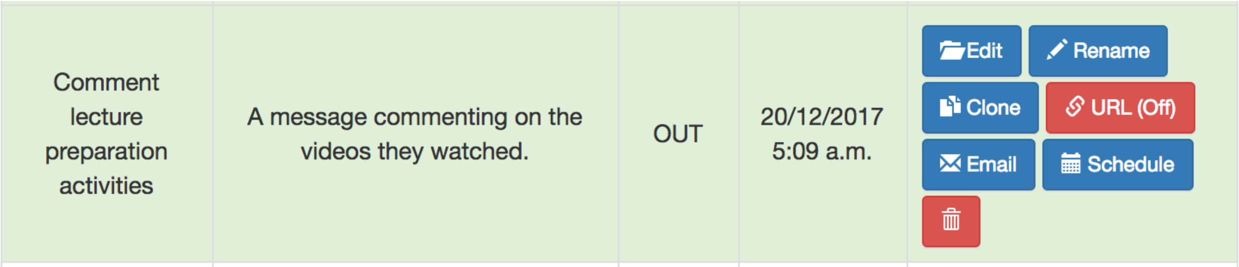
Once a text is created, you need to verify that all the elements are properly visualised for each of the rows. This is the purpose of the `Preview` button at the bottom of the page.

Sending personalised emails

You now have created an action and verified its content using the *Preview* button. Go back to the *Actions* screen (showing the table with the actions you created in the workflow). The right-most column shows a button that reads *Send Email*.

This functionality process the text in the action for each learner and sends the resulting text as an email. If you click in that button the platform asks you for additional information:

The subject A line to be included as subject of all the emails.



Comment lecture preparation activities	A message commenting on the videos they watched.	OUT	20/12/2017 5:09 a.m.

The column with the email address OnTask needs to know where to send the email. It assumes that you have a column containing that information for each learner and it needs you to select that column.

Send summary message If you select this option OnTask will send you an email with the summary of this operation (number of rows in the table that were selected by the filter, number of emails sent, date/time of the operation, etc.

Snapshot of the workflow If you select this option, after the emails are sent, the platform returns you a file that contains a snapshot (picture) of the workflow. It basically freezes the content of the workflow and places it in a file given to you. You may take this file and *import back the workflow*. In this new workflow you can check the values and messages at the time the operation was executed.

Making personalised content available to learners

Sending a personalised email is just one of various possible actions to do with a personalised text. Another one is to make the content available through a URL that can then be given to the learners. OnTask offers this possibility through the button labeled `URL` followed by either the word `(Off)` or `(On)`.

If you select this option, the platform will show you the URL providing access, the choice of making it available, and the possibility of using an alternative column containing the email address.

You may enable/disable this URL at any time. If a learner tries to access this URL and it is disabled, the platform informs the user that the information is not available.

3.6 The Scheduler

> "I have no regular schedule. I get up whenever I can." – Jimmy Wales

The *actions out* that are used to send emails can be *scheduled* to be executed at some point in the future. The *Schedule* operation in these actions opens a dialog like the one shown in the following figure:

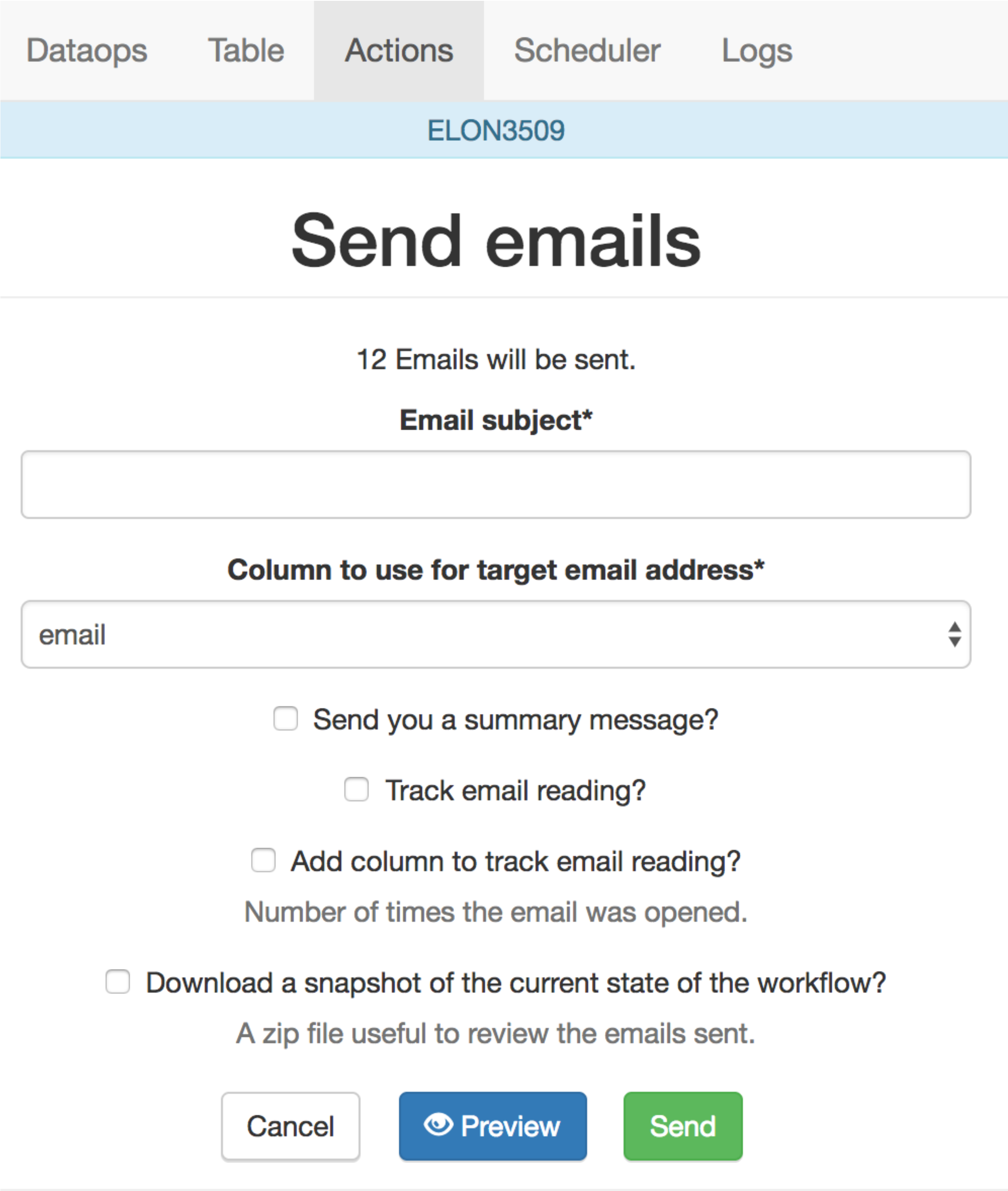
The action (sending the emails with the personalised text) is executed at the selected time. This functionality requires the server to be configured to check the list of pending tasks and scheduling for execution.

The *Schedule* item in the navigation menu shows all the scheduled tasks for the current workflow. The left-most column offers the operations to edit the task or delete it.

3.7 The Logs

The platform keeps a log of most of the operations that are executed when managing a workflow. These records are available through the *Logs* link in the navibation bar at the top of the screen.

You may review the events and download them as a CSV file.



Dataops
Table
Actions
Scheduler
Logs
ELON3509

Send emails

12 Emails will be sent.

Email subject*

Column to use for target email address*

email

☐ Send you a summary message?

☐ Track email reading?

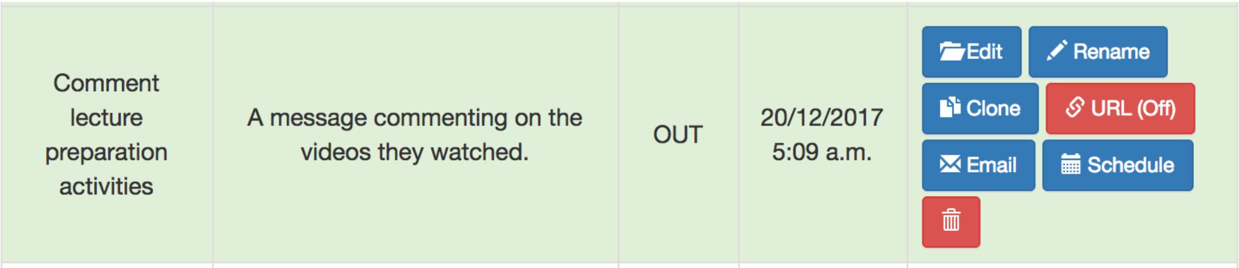
☐ Add column to track email reading?

Number of times the email was opened.

☐ Download a snapshot of the current state of the workflow?

A zip file useful to review the emails sent.

Cancel
Preview
Send

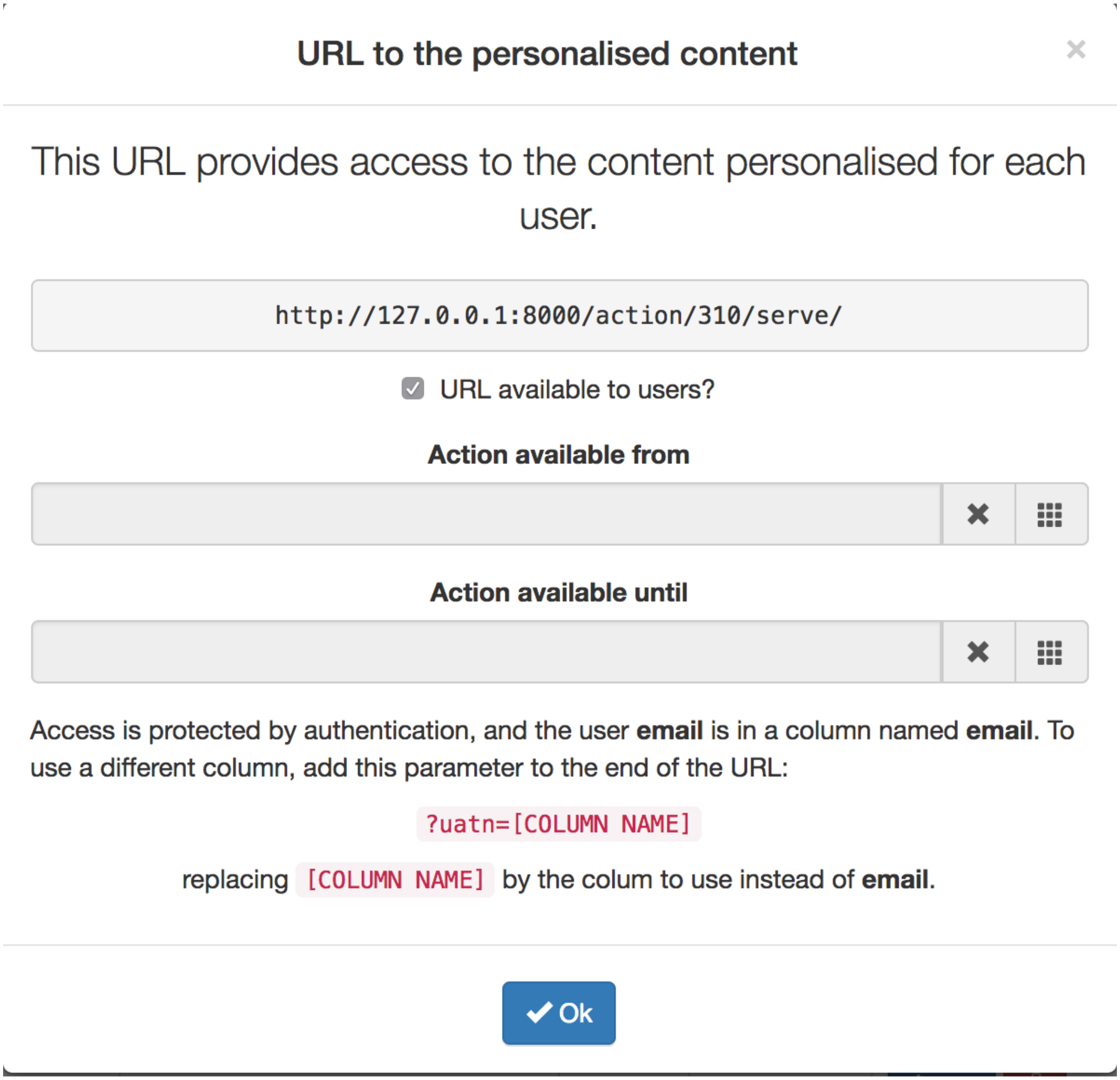


Comment lecture preparation activities

A message commenting on the videos they watched.

OUT

20/12/2017 5:09 a.m.



URL to the personalised content

This URL provides access to the content personalised for each user.

`http://127.0.0.1:8000/action/310/serve/`

☑ URL available to users?

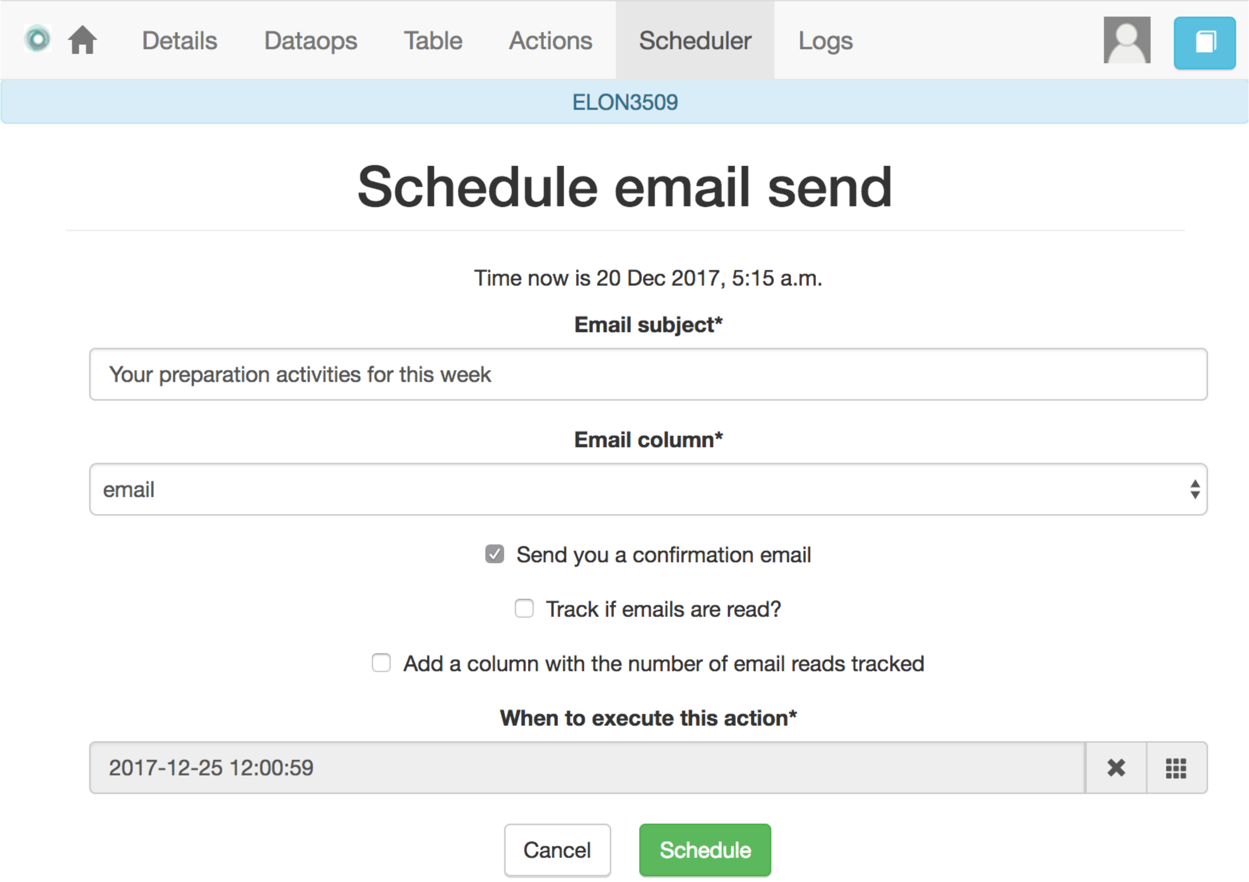
Action available from

Action available until

Access is protected by authentication, and the user **email** is in a column named **email**. To use a different column, add this parameter to the end of the URL:

`?uatn=[COLUMN NAME]`

replacing `[COLUMN NAME]` by the colum to use instead of **email**.


Details
Dataops
Table
Actions
Scheduler
Logs
ELON3509
Cancel
Schedule

Schedule email send

Time now is 20 Dec 2017, 5:15 a.m.

Email subject*

Your preparation activities for this week

Email column*

email

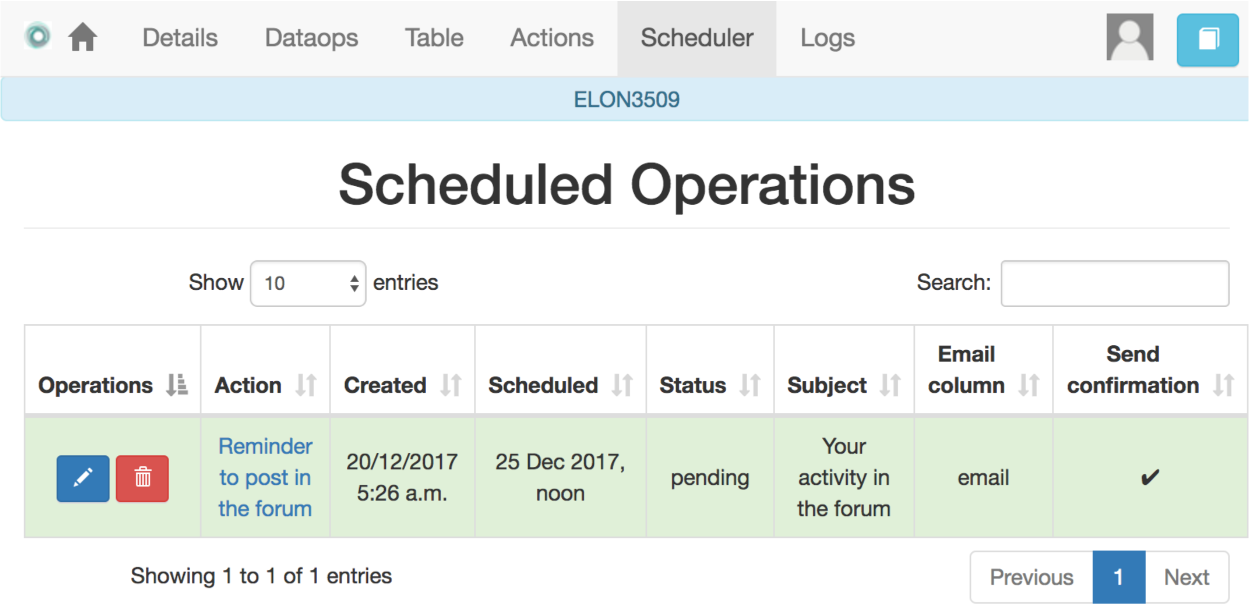
☑ Send you a confirmation email

☐ Track if emails are read?

☐ Add a column with the number of email reads tracked

When to execute this action*

2017-12-25 12:00:59


Details
Dataops
Table
Actions
Scheduler
Logs
ELON3509
Previous
1
Next

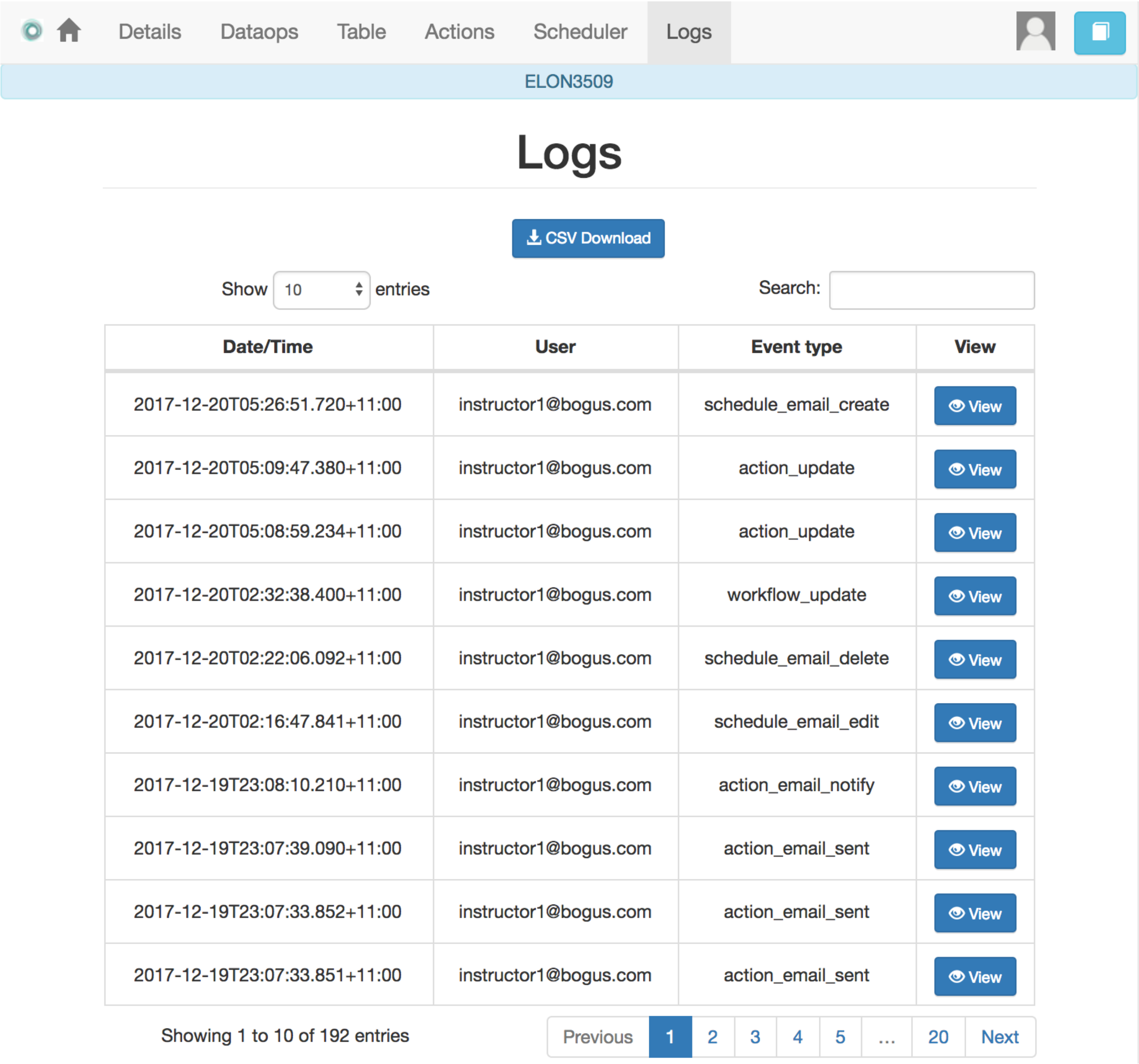
Scheduled Operations

Show 10 entries

Search:

Operations	Action	Created	Scheduled	Status	Subject	Email column	Send confirmation
	Reminder to post in the forum	20/12/2017 5:26 a.m.	25 Dec 2017, noon	pending	Your activity in the forum	email	✔

Showing 1 to 1 of 1 entries


Details
Dataops
Table
Actions
Scheduler
Logs
ELON3509
Logs
CSV Download
Show 10 entries
Search:

Date/Time	User	Event type	View
2017-12-20T05:26:51.720+11:00	instructor1@bogus.com	schedule_email_create	View
2017-12-20T05:09:47.380+11:00	instructor1@bogus.com	action_update	View
2017-12-20T05:08:59.234+11:00	instructor1@bogus.com	action_update	View
2017-12-20T02:32:38.400+11:00	instructor1@bogus.com	workflow_update	View
2017-12-20T02:22:06.092+11:00	instructor1@bogus.com	schedule_email_delete	View
2017-12-20T02:16:47.841+11:00	instructor1@bogus.com	schedule_email_edit	View
2017-12-19T23:08:10.210+11:00	instructor1@bogus.com	action_email_notify	View
2017-12-19T23:07:39.090+11:00	instructor1@bogus.com	action_email_sent	View
2017-12-19T23:07:33.852+11:00	instructor1@bogus.com	action_email_sent	View
2017-12-19T23:07:33.851+11:00	instructor1@bogus.com	action_email_sent	View

Showing 1 to 10 of 192 entries

Previous
1
2
3
4
5
…
20
Next

CHAPTER 4

OnTask Tutorial

Before you start the tutorial make sure you have an account in an OnTask instance and you have instructor privileges (create workflows, actions, etc). Also, download the file `learner_information.csv` that contains a synthetic data set with information about learners, participation in a discussion forum, engagement with activities, and some additional features.

Remember the three central concepts in OnTask:

Workflow A container with the data (table), a set of procedures to manipulate columns, data upload and a set of actions. This container is typically associated with a course, but it could also model an entire institutional degree.

Table A two-dimensional structure in which each row represents a learner, and each column a learner attribute such as the score in an assessment, class attendance, number of interventions in the discussion forum, engagement with videos, etc.

Actions An action is a HTML resource of which certain parts that are included or excluded based on a set of **conditions** created with the learner attributes (for example, number of interventions in the forum is larger than five, and number of times a video was watched is larger than 2).

The following figure represents the high level view of the tool.

The process to create a personalised text is divided into four stages:

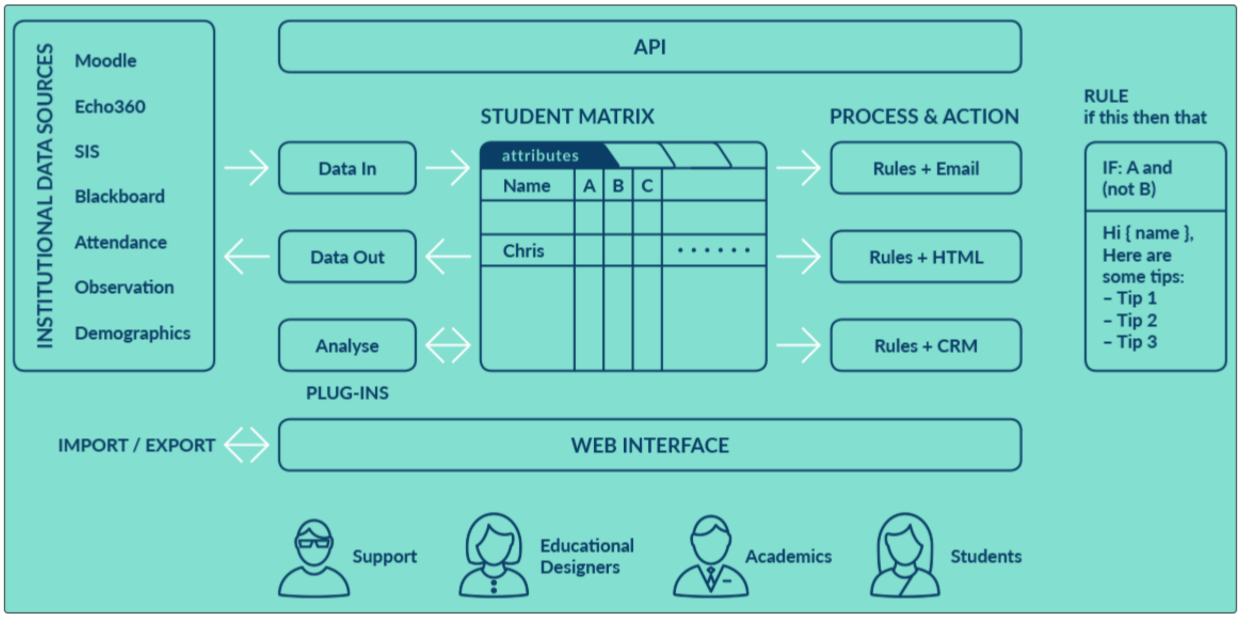
Upload the data We will use a `CSV file with the learners data` about several activities in a course. In this first stage you will upload the data into a table and identify certain special columns (those that provide a unique reference to a learner such as the learner ID)

Review the data table In this stage you will explore the values in the table.

Create an action In this stage you will create an action with various conditions that will be applied to the text of a personalised email message.

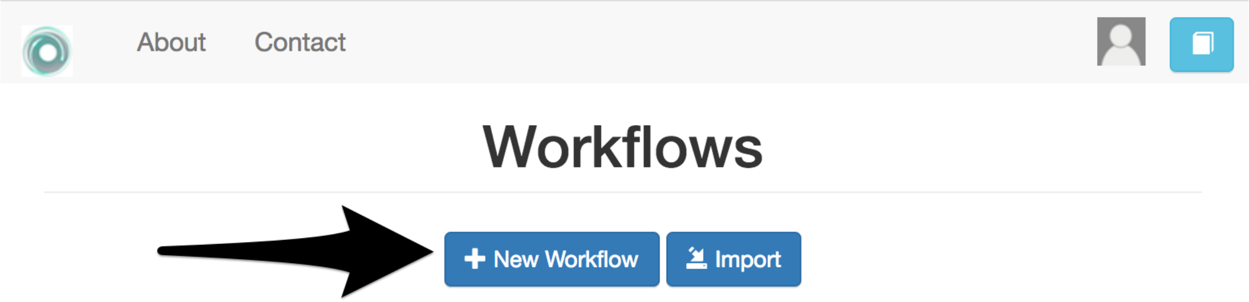
Review the messages And finally you will review the appearance of these messages for different learners.

The following steps describe the required operations in each of these stages.


INSTITUTIONAL DATA SOURCES
Moodle
Echo360
SIS
Blackboard
Attendance
Observation
Demographics
API
STUDENT MATRIX
attributes
Name
A
B
C
Chris
PROCESS & ACTION
RULE
if this then that
Data In
Data Out
Analyse
PLUG-INS
Rules + Email
Rules + HTML
Rules + CRM
IF: A and (not B)
Hi { name }, Here are some tips:
- Tip 1
- Tip 2
- Tip 3
IMPORT / EXPORT
WEB INTERFACE
Support
Educational Designers
Academics
Students

4.1 Create a new workflow

Log into the tool and click in the tool icon on the top left corner of the screen. If you have an instructor account, you will see the button to create a new workflow as shown in the following figure.


About
Contact
Workflows
+ New Workflow
Import

The icon in the top right corner next to your profile image is a link to the OnTask documentation. Click in the button to create a new workflow and enter its name and a description.

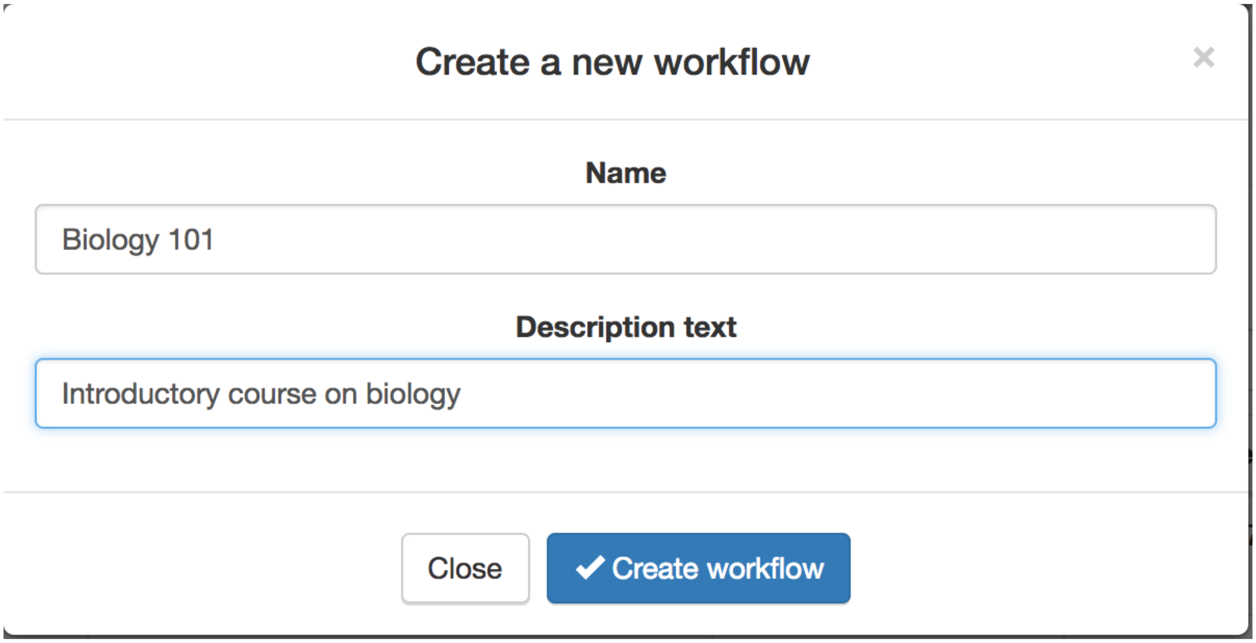
Once you created the workflow the platform shows the list of all the workflows available.

The first step is to select or *open* a workflow by clicking on the name to manipulate it. Once this operation is done, the access to the element is blocked for any other users (in case the workflow is being shared) to prevent two users changing the data or the actions simultaneously. The following screens will show the name of the selected workflow at the top. If you want to select another workflow to manipulate, you simply click in the OnTask icon at the top left corner of the screen to go back to the initial table.

4.2 Open a workflow

When you open a workflow, a page with its details is shown like the one in the following figure

You can see the icon on the top right corner that links to the initial page, and the icon in the top left corner that links to the documentation. So far the page only shows the description of the workflow and the last time it was modified

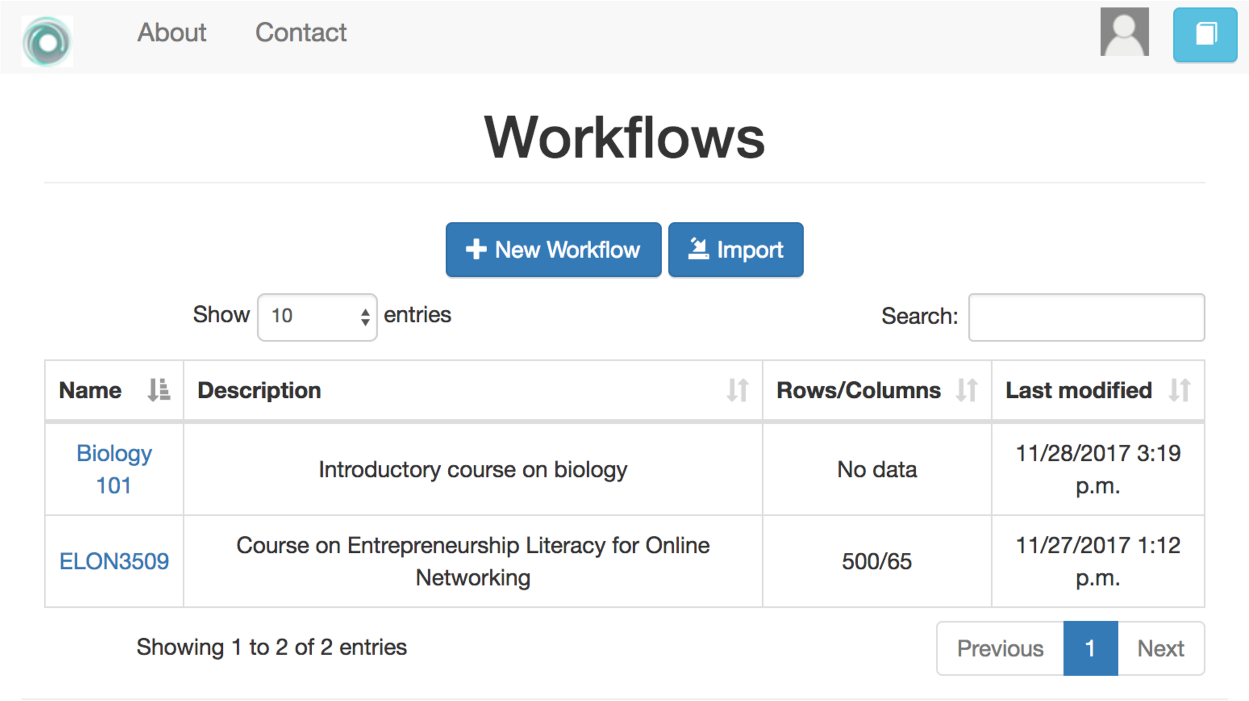


Create a new workflow

Name: Biology 101

Description text: Introductory course on biology

×
Close
✓ Create workflow



About
Contact

Workflows

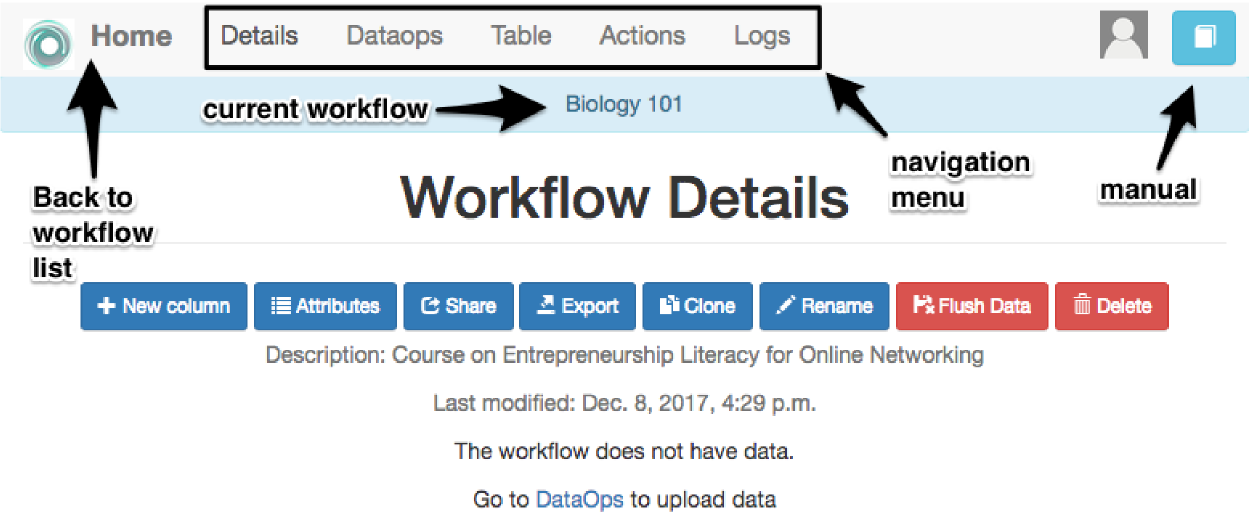
+ New Workflow
Import

Show 10 entries Search:

Name	Description	Rows/Columns	Last modified
Biology 101	Introductory course on biology	No data	11/28/2017 3:19 p.m.
ELON3509	Course on Entrepreneurship Literacy for Online Networking	500/65	11/27/2017 1:12 p.m.

Showing 1 to 2 of 2 entries

Previous
1
Next


Home
Details
Dataops
Table
Actions
Logs
current workflow
Biology 101
Back to workflow list
navigation menu
manual
Workflow Details
New column
Attributes
Share
Export
Clone
Rename
Flush Data
Delete

Description: Course on Entrepreneurship Literacy for Online Networking

Last modified: Dec. 8, 2017, 4:29 p.m.

The workflow does not have data.

Go to DataOps to upload data

because no data has been uploaded.

The top of the screen now shows the sections offering different operations over the workflow:

Details Is the current page with information about the columns, data types, number of actions, etc.

Table Operations to visualise and manipulate the table (search for values, add a row, add a column)

Actions Operations to create the actions and conditions.

Logs A table showing the history of operations performed on this workflow

The buttons immediately under title *Workflow Details* show some of the operations available at this point:

- *New column*
- *Attributes*
- *Share*
- *Export*
- *Rename*
- *Delete*

4.3 Data Upload

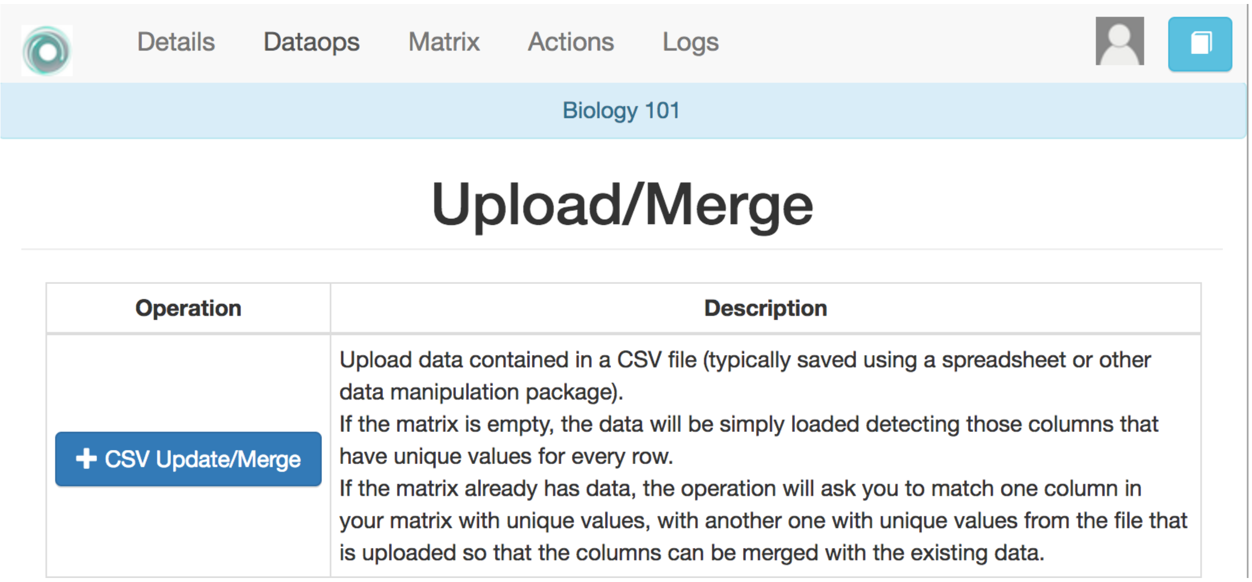
We now upload the data included in the file `learner_information.csv`. Click in the *Dataops* menu, and then in the option to *CSV Update/Merge* as shown in the following figure

The next screen asks you to choose a file to upload the data.

Choose the file `learner_information.csv` and proceed to the next step. The next screen shows a table with the name of the detected columns, the type (also automatically detected), a pre-filled field with the column name (in case you want to change it), and if it is a *key column* (there are no repeated values in all the rows).

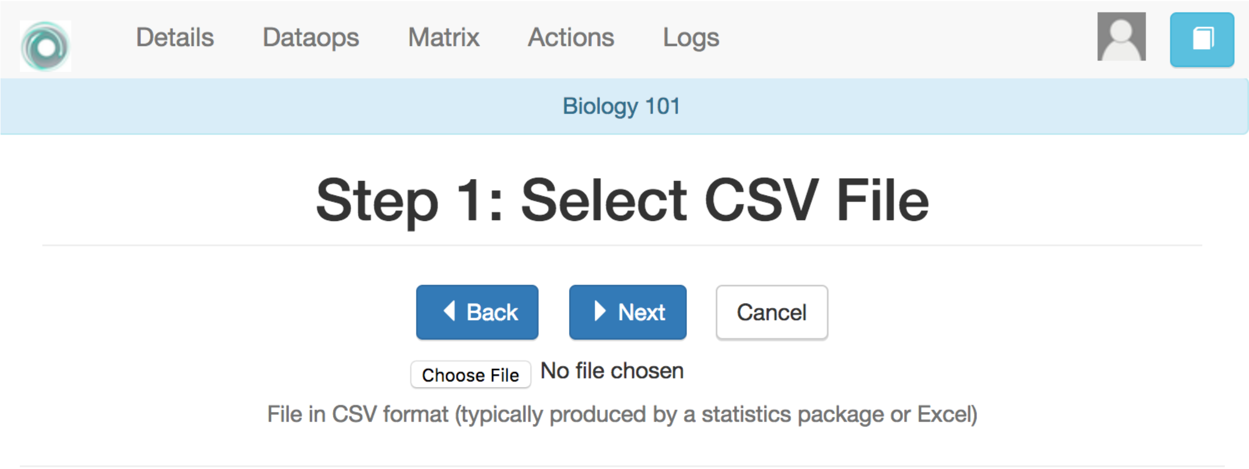
The *key* columns are highlighted because a workflow must have at least one column of this type in its table. Select all the column (clicking in the top element labeled *load*) and click on the *Finish* button, and then back to the *Details* page to see the summary of the information in the workflow.

You can now see the information about the columns present in the workflow as shown in the follogin figure


Details
Dataops
Matrix
Actions
Logs
Biology 101

Upload/Merge

Operation	Description
+ CSV Update/Merge	Upload data contained in a CSV file (typically saved using a spreadsheet or other data manipulation package). If the matrix is empty, the data will be simply loaded detecting those columns that have unique values for every row. If the matrix already has data, the operation will ask you to match one column in your matrix with unique values, with another one with unique values from the file that is uploaded so that the columns can be merged with the existing data.

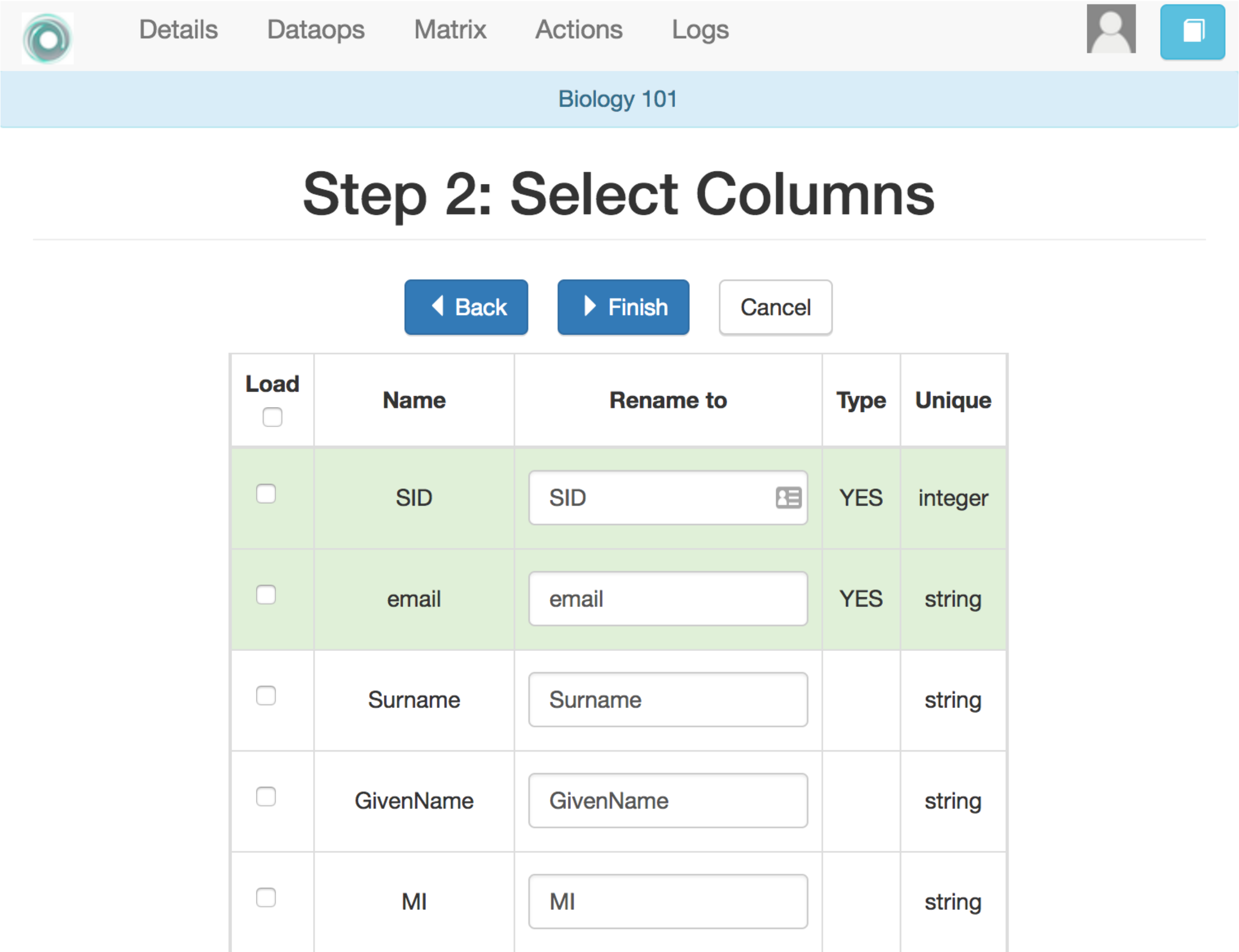

Details
Dataops
Matrix
Actions
Logs
Biology 101

Step 1: Select CSV File

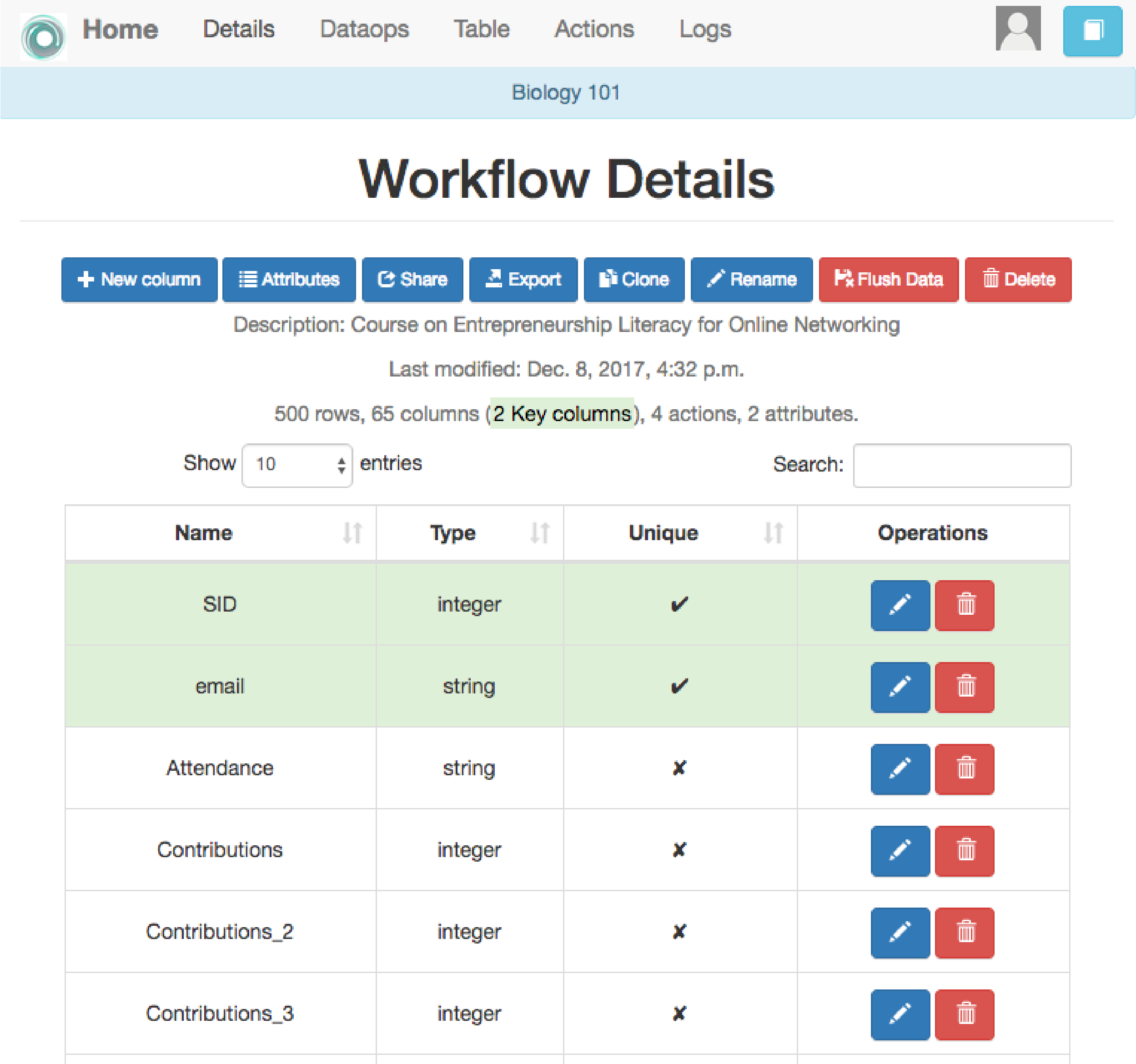
Back
Next
Cancel
Choose File

No file chosen

File in CSV format (typically produced by a statistics package or Excel)


Details
Dataops
Matrix
Actions
Logs
Biology 101
Step 2: Select Columns
Back
Finish
Cancel

Load ☐	Name	Rename to	Type	Unique
☐	SID	SID	YES	integer
☐	email	email	YES	string
☐	Surname	Surname		string
☐	GivenName	GivenName		string
☐	MI	MI		string


Home
Details
Dataops
Table
Actions
Logs
Biology 101

Workflow Details

New column
Attributes
Share
Export
Clone
Rename
Flush Data
Delete

Description: Course on Entrepreneurship Literacy for Online Networking

Last modified: Dec. 8, 2017, 4:32 p.m.

500 rows, 65 columns (2 Key columns), 4 actions, 2 attributes.

Show 10 entries

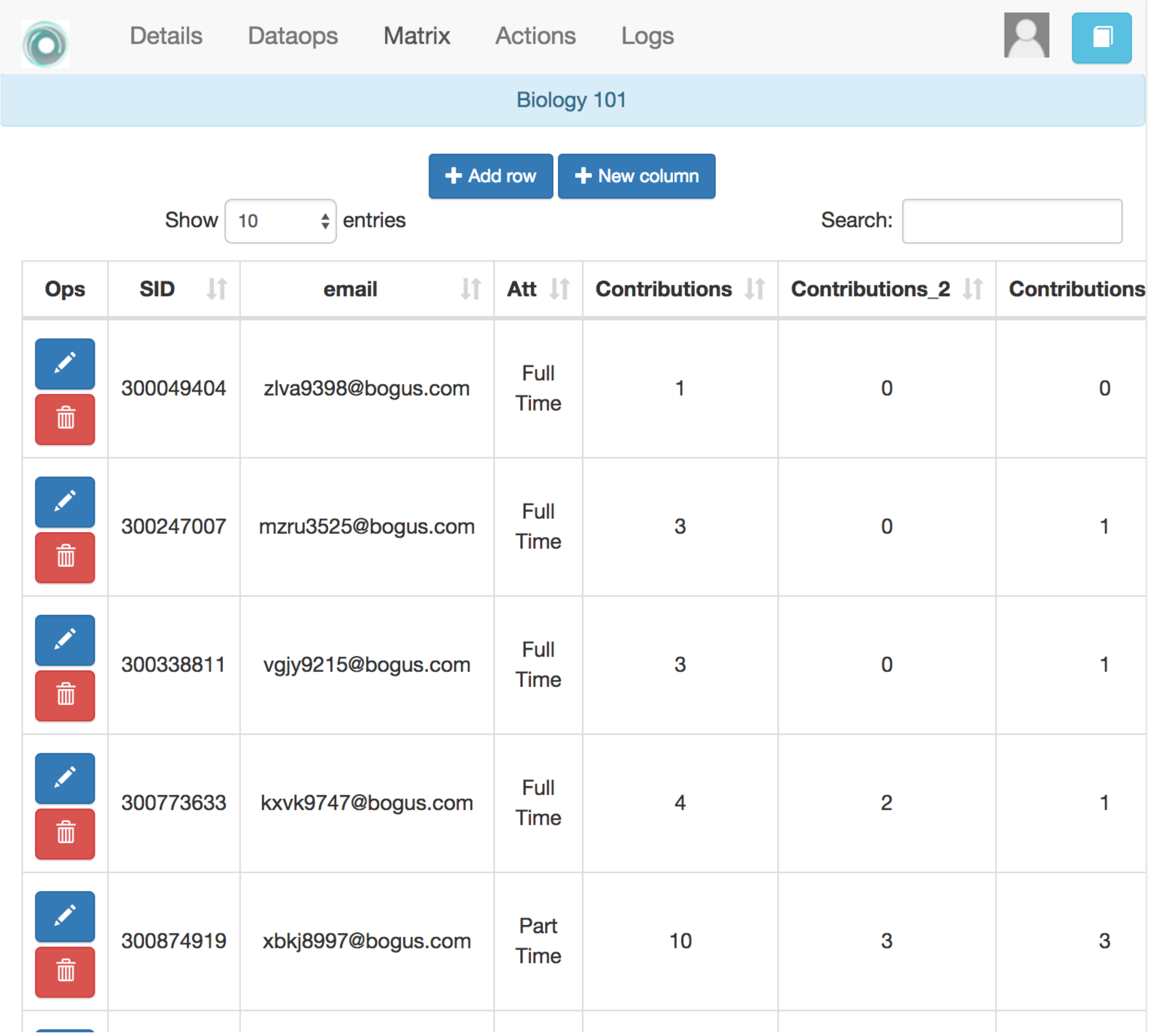
Search:

Name	Type	Unique	Operations
SID	integer	✔	
email	string	✔	
Attendance	string	✘	
Contributions	integer	✘	
Contributions_2	integer	✘	
Contributions_3	integer	✘	

For each column you can change its name, description, type and key attributes, or delete it from the workflow (icons in the left most column of the table).

4.4 Browsing the table

Once the data has been uploaded, click in the *Table* link at the top of the screen. The following screen shows the values stored in the table


Details
Dataops
Matrix
Actions
Logs
Biology 101
Add row
New column
Show 10 entries
Search:

Ops	SID	email	Att	Contributions	Contributions_2	Contributions
✎ 🗑	300049404	zlva9398@bogus.com	Full Time	1	0	0
✎ 🗑	300247007	mzru3525@bogus.com	Full Time	3	0	1
✎ 🗑	300338811	vgjy9215@bogus.com	Full Time	3	0	1
✎ 🗑	300773633	kxvk9747@bogus.com	Full Time	4	2	1
✎ 🗑	300874919	xbkj8997@bogus.com	Part Time	10	3	3

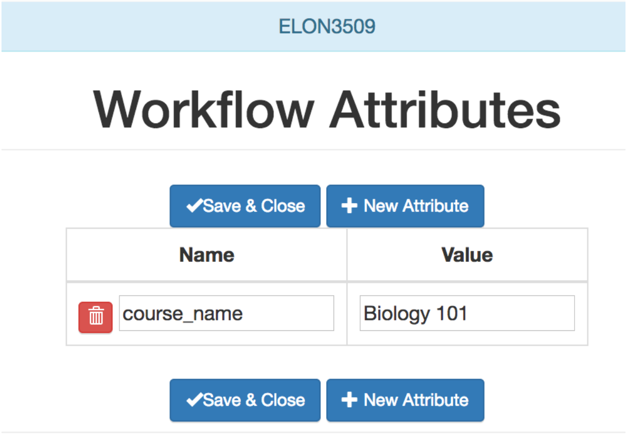
4.5 Workflow Operations

There are several operations available at the details page.

4.5.1 Attributes

If you click again in the *Details* link at the top of the screen you will see again the page with the workflow details, but this time it will include the information about the columns just loaded.

You can define a set of *attributes* in the workflow. This is simply a set of pairs *name, value* that you can use to have a single place where a value is defined and then reused in several other locations. For example, the name of the course is probably going to appear in various communications with the learners. If you define the attribute *Course_name* with that value, you can then refer to the attribute and it will be replaced by its value.

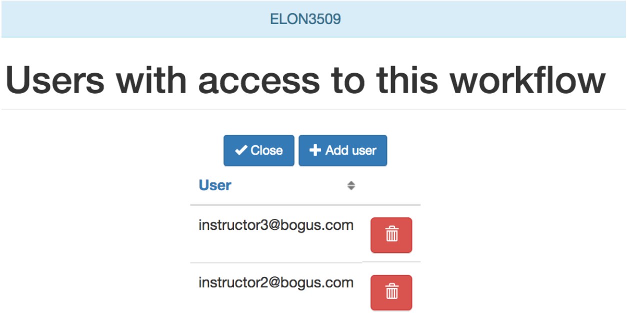


Workflow Attributes

Name	Value
course_name	Biology 101

4.5.2 Share

You may share a workflow with other instructors in the platform. The *Share* button will allow you to add/remove other users to this list.



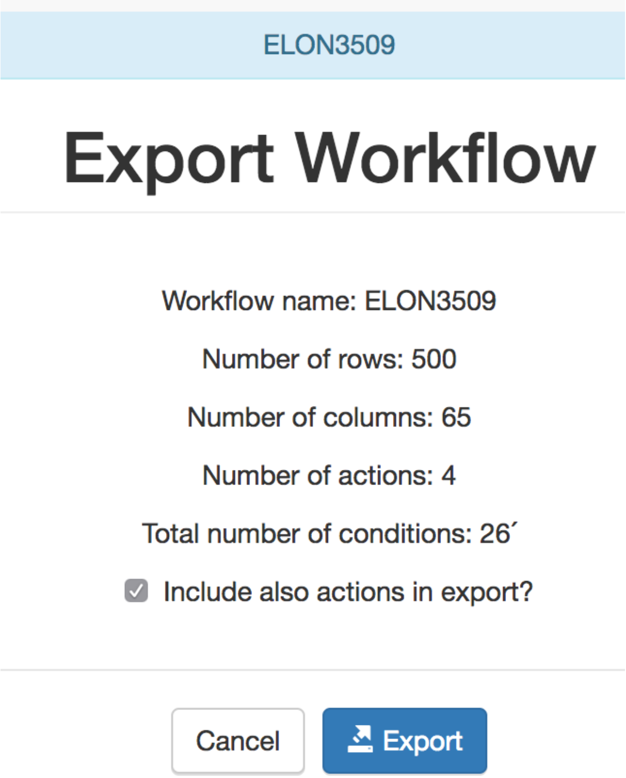
Users with access to this workflow

User
instructor3@bogus.com
instructor2@bogus.com

Remember that whenever you open a workflow, it becomes unavailable for the other users with whom it is being shared.

4.5.3 Export

This functionality allows you to take all the information included in a workflow and export it. The functionality offers the option of including in the export only the data, or the data and the actions.



Workflow name: ELON3509

Number of rows: 500

Number of columns: 65

Number of actions: 4

Total number of conditions: 26´

☑ Include also actions in export?

4.5.4 Clone

This button creates a clone of the workflow with the a name containing the prefix "*Copy of*". Once the operation is executed, the workflow is available in the home screen (link in the upper left corner of the screen).

4.5.5 Rename

Use this function to change the name and description of the workflow

4.5.6 Flush Data

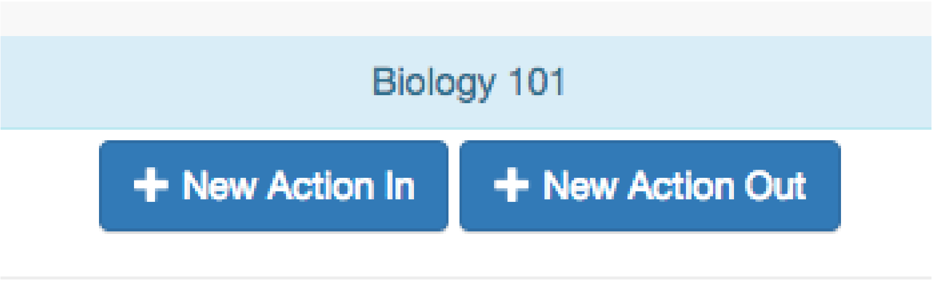
This function deletes the data associated with the workflow. It maintains the set of attributes and the actions, but it removes the conditions and filters from all the actions.

4.5.7 Delete

This function deletes completely the workflow from the platform.

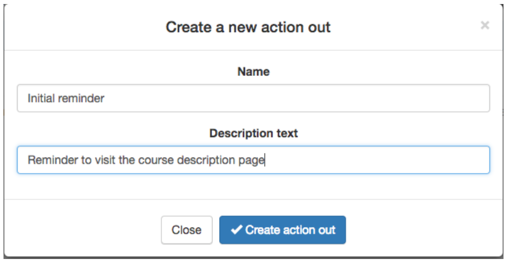
4.6 Actions

Click in the *Actions* link at the top of the screen. The next screen shows the list of actions that are part of the workflow, and if there is none, you will only see the buttons to crate a new ones.



4.6.1 Actions Out

Click on the button to create a new *Action Out* and provide a name and a description.



Name: Initial reminder

Description text: Reminder to visit the course description page

The next screen is the *action editor*. The functions are divided into three areas. The one at the top allows you to specify a condition to select or filter a subset of students. The second contains the conditions to be used in the personalised text. The third is a HTML text editor with the content to personalise.

Place the cursor in the text area and start the text with a salutation, then select the name of a column from the pull-down menu right above the text editor and select the column *GivenName*. The string *{{ GivenName}}* appears in the text area. This notation is to instruct the next steps to replace the value among double curly braces with the name of each student.

Click now in the button *New* in the condition area. A form appears to introduce the name, description and formula. The formula may contain any combination of Boolean operators with respect to the column values. For example, the condition:

```
Q01 is equal to 0 AND Q02 is equal to 0
```

can be encoded in the formula widget as shown in the following figure

We can now use this condition to control the appearance of text in the text area. Write a sentence that you would like to appear, select it and then click in the arrow button in the condition.

The text area is then surrounded by two marks:

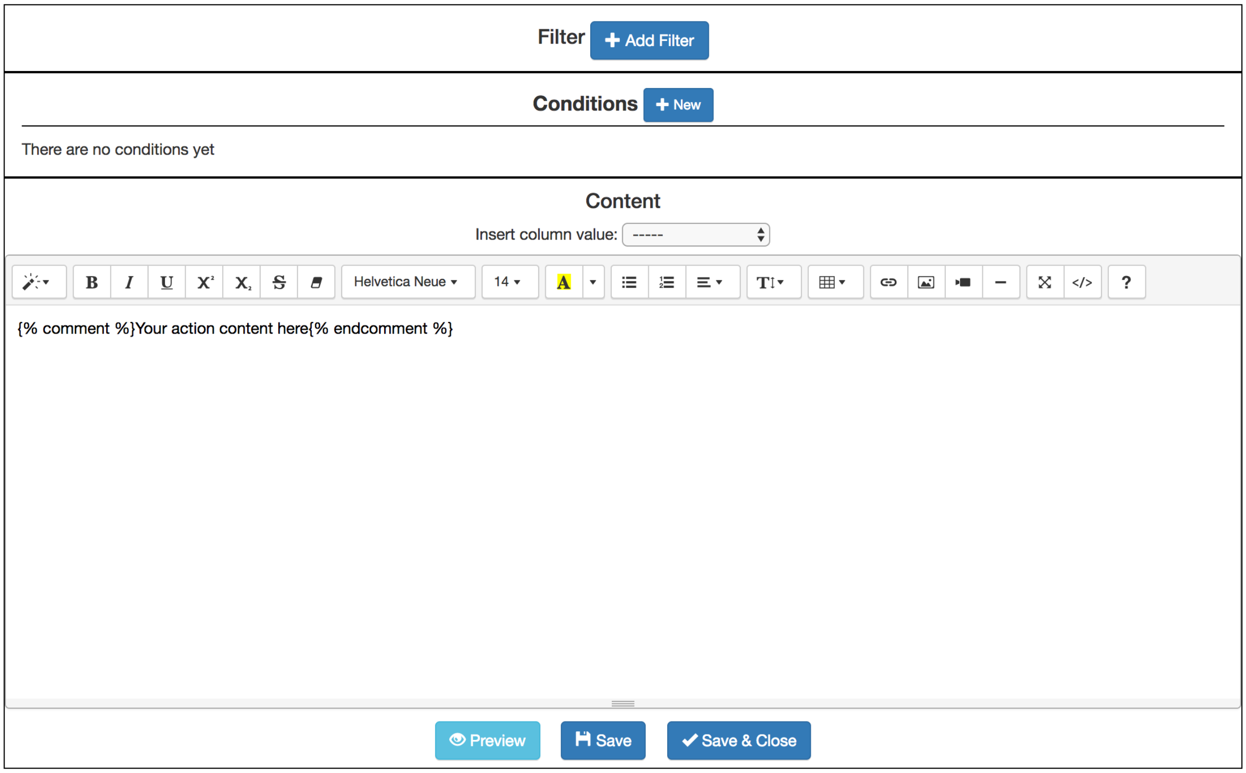
```
{% if Todo_review_T1 %}You need to review Topic 1{% endif %}
```

This is the encoding to instruct the processing step to check the value of the condition `Topic_1_incorrect` and include the surrounded text only if the condition is true.

You may also insert any attributes attached to the workflow. The attribute name will be replaced by its value when processing the text.

4.6.2 Actions In

To be written.

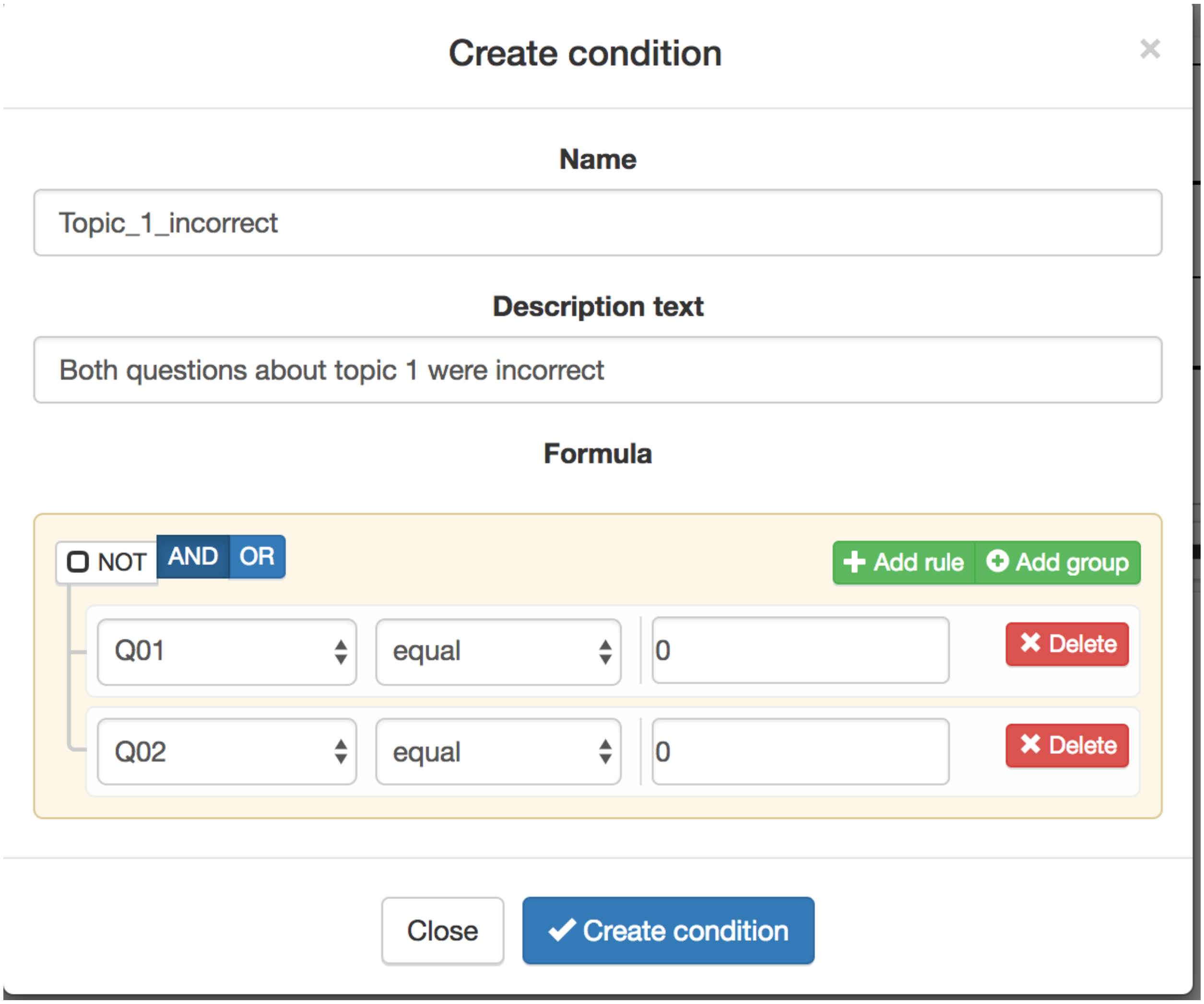

Filter
+ Add Filter
Conditions
+ New

There are no conditions yet

Content
Insert column value: -----
Helvetica Neue
14

{% comment %}Your action content here{% endcomment %}

Preview
Save
Save & Close



Create condition
×
Add rule
Add group
Delete
Delete
Close
Create condition

Name

Topic_1_incorrect

Description text

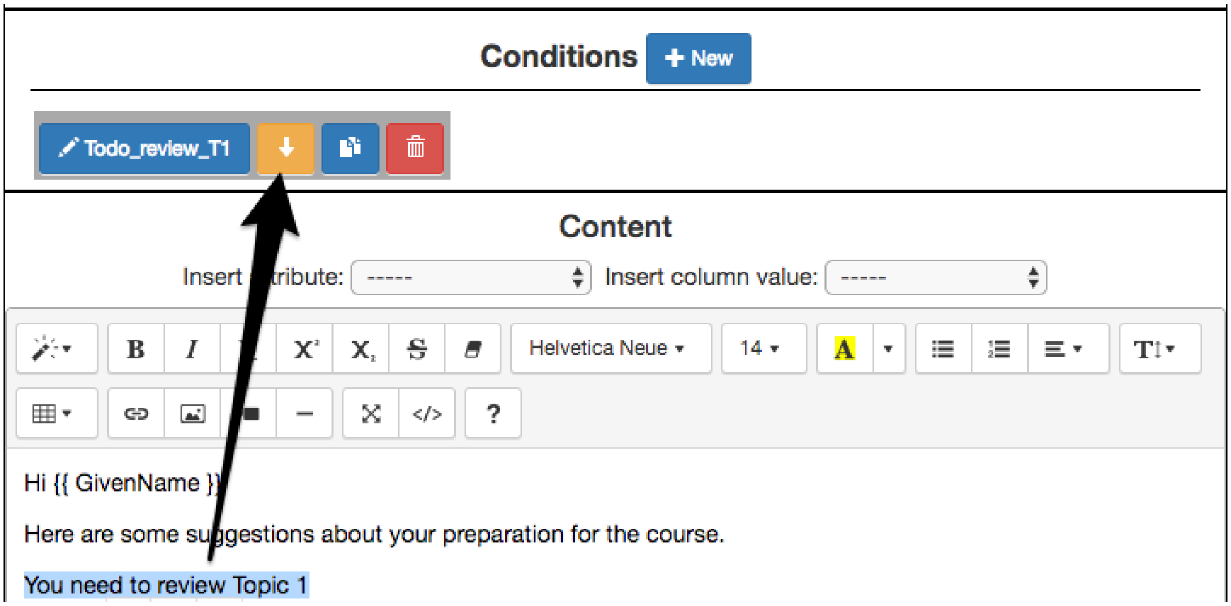
Both questions about topic 1 were incorrect

Formula

☐ NOT AND OR

Q01 equal 0

Q02 equal 0



Conditions
New
Todo_review_T1
Content
Helvetica Neue
14
?

Insert attribute: -----

Insert column value: -----

Hi {{ GivenName }}

Here are some suggestions about your preparation for the course.

You need to review Topic 1

CHAPTER 5

Advanced Features

OnTask is built in Python using the web framework Django in combination with some additional libraries such as Django REST Framework, Pandas, etc. The application is available as open source with MIT License. This means that the advanced users can get a copy of the source code and modify it to suite their needs using the already existing models and functions.

5.1 The API (Application Programming Interface)

OnTask is a platform that facilitates the connection between data and the provision of personalised learner support actions. The higher the quality of the data the higher number of possible effective support actions. This means that OnTask should facilitate the connection with already existing data sources so that it can combine data sets and create a comprehensive view of how a learning experience is evolving.

The API is documented online through the url suffix `apidoc`. The page contains the description of every entry point available with the required parameters.

When manipulating the elements in the table there are two versions of the basic operations (create a table, update a table, merge).

1. Pandas Version. This version handles the encoding of data frames using the pandas pickle encoding. This encoding has the advantage that maintains the elements of the dataframe intact. In other words, when the data frame is decoded from the pandas pickle format back to a regular dataframe, the same initial dataframe is obtained.
2. JSON Versoin. This version encodes the dataframes in JSON. The problem with this format is that values such as NaN and NaT (not a time) are not allowed by JSON, and they are substituted by empty strings. This change may have significant effects on the dataframe, specially on the types of the columns. If there is a dataframe with a column of type datetime and with any element with value NaT and it is first extracted through the JSON interface, and then uploaded again, the NaT value is transformed into an empty string, and Pandas will no longer recognise that column as datetime, but instead it will render the column of type string. This my have also an effect on how rules and actions are evaluated.

Changelog