Unplagged Developers Manual – Part 2

Building the Plagiarism Detection Cockpit

Term paper for the master project I
Mentoring Teacher: Prof. Dr. Debora Weber-Wulff

Department Economics II
HTW Berlin – University of Applied Sciences

Elsa Mahari (s0534556) <Elsa.Mahari@gmx.de>
Dominik Horb (s0534217) <horb@htw-berlin.de>
Tien Nguyen (s0512510) <s0512510@htw-berlin.de>
Benjamin Oertel (s0522720) <contact@benjaminoertel.com>
Heiko Stammel (s0534218) <heiko.stammel@googlemail.com>

Contents

List of Figures

1. Introduction and Overview

@dominik

2. Features

2.1. Notifications and Comments

The following section describes an important part for registered users, the notification system and the opportunity for adding comments basically on any resource in the system.

2.1.1. Recent activity stream

It displays the most recent events in the portal in the order they happened. Each activity (figure 2.1), one line in the stream, consists of 3 parts: meta information about the activity itself, information about the initiator and comments. The content of each of these parts is being determined automatically, when a new notification is being persisted to the database.

Besides the initiator information and comments, each activity has an own title, description and icon. These meta information texts can be edited in the portal in the 'Administration' > 'Acions' section. Although it is not possible to remove them, because the action references are being used hard-coded within the system. That means a set of notifications is provided and can be ouput anywhere in the workflow.

The following code snippet shows, how to create a new notification when a new automatic plagiarism detection report was created. The static method takes in 3 parameters: a unique name for the notification type, the content object related to the notification and a user object, as the third parameter. A list of all available notification types can either

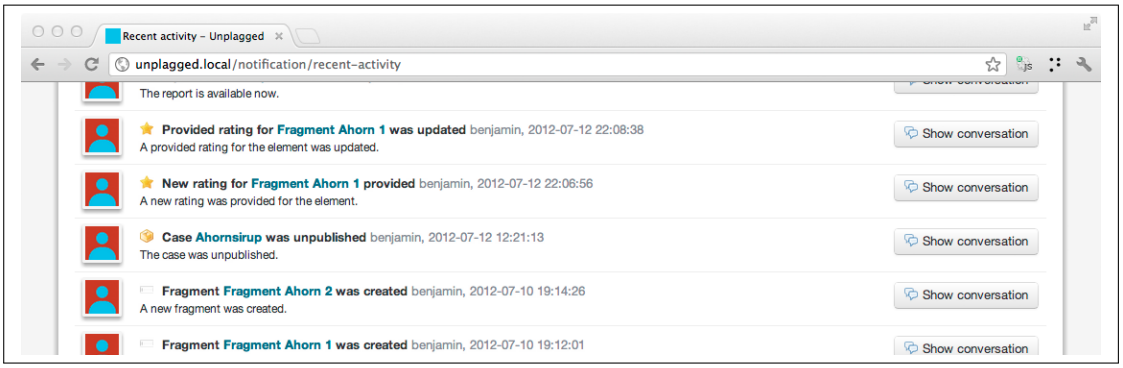

Figure 2.1.: Single activity in the activity stream

be found in the previousely mentioned actions section or in the scripts/build/initdb.php file, where all notifcation types are being declared.

Listing 2.1: Creating a notification for a created report

```
1 Unplagged_Helper::notify("detection_report_created", $report,
      $report->getUser());
```

Unplagged does have an extensive role and permission management. Therefore the grant on each resource is being verified, before it is being displayed in the activity stream. Usually the resource related to the notification is the resource where the permission check is being performed on. Although in some cases, e.g. when rating a fragment, the resource will be the rating itself, but the permission check is done on the fragment. In this case the notify-method is being called with a fourth, optional parameter, another resource, in this case the fragment. When the user has access on the fragment, all ratings can be accessed as well, automatically.

2.1.2. Comments plugin

Comments are simply a small text related to a specific user and a resource. They can be used to share ideas on an object collaboratively. The most prominent part at Unplagged, where comments are being used, is the activity stream. A comment can be added by any user having access to the notification.

Figure 2.2.: Creating a comment on a resource

For providing a better workflow to the user, the comments can be refreshed and added in place. That means, the position where the user scrolled to in the browser does not get affected. The in-place refreshing is being realized through AJAX. The comments container is being loaded empty and displaying a small loading image only. Not before the user clicks the 'show conversation' button, the comments are being fetched through a post request to the server. Whenever the result is being fetched completely, the spinner graphic is being hidden and the comments are being appended. The parsing of the comments markup is being done in Javascript as well. So the server requests are kept small and the server can return JSON only without any HTML.

Listing 2.2: Refreshing the comments of a resource

```
1   target.show();
2       conversation.hide();
3       loading.slideDown(800, function() {
4         // get the whole conversation
5         $.post('/notification/conversation', {
6           'source': sourceId
7         }, function(data) {
8           if(!data.errorcode) {
9             conversation.html("");
10            $.each(data, function(index, value) {
11              conversation.append(renderConversation(value));
12            });
13            loading.slideUp(800, function() {
14              conversation.slideDown(300);
15            });
16          } else {
17            conversation.html('<div class="comment">' + data.
                  message + '</div>');
```

```
18              loading.slideUp(800, function() {
19                conversation.slideDown(300);
20              });
21            }
22          }, "json");
```

Listing 2.3: Creating the markup of a single comment

```
1  function renderConversation(data, target) {
2      var tpl;
3
4      switch(data.type) {
5        case 'comment':
6          tpl = '<div class="comment">' +
7          '<div class="image"><img class="avatar-small" src="' +
               data.author.avatar + '" /></div>' +
8          '<div class="details">' +
9          '<div class="title"><b>' + data.author.username + '</b
               > ' + data.text +
10         ' <span class="date">' + data.created.humanTiming +
               '</span>' +
11         '</div>' +
12         '</div>' +
13         '</div>';
14         break;
15     }
16     if(!target) {
17       return tpl;
18     } else {
19       target.append(tpl);
20     }
21   }
```

2.2. Fragments

Fragments are the part of the application where found text passages that are plagiairism or potential plagiarism are being documented.

A single fragment contains a candidate and a source document. Each of the two documents is being saved with a starting position (page number / line number combination) and an ending position. These two positions can be used to determine exactly the text being involved in a fragment. To visualize this, the figure 2.3) below shows a sample fragment.

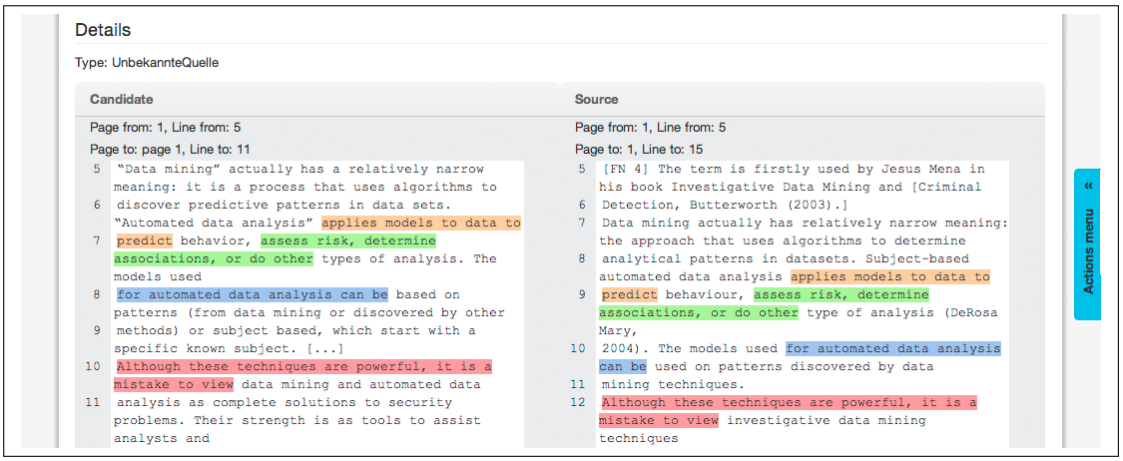

Figure 2.3.: Single fragment with highlighted similarities

2.2.1. Creating a fragment

Such a fragment can be created in two ways, one for people that like using the mouse and another one that can be used with the keyboard only.

The old-fashioned way

The basic way, which can be accessed through the keyboard only, offers a two-column form to the user where the source and potential plagiarism information can be selected by hand. Once a page or line number is being changed, the text shown below is being updated instantly through AJAX and the similarities are highlighted automatically. Although the

big disadvantage of this method is, that the whole page is never displayed and the user actually has to guess where the starting and ending point of the fragment in the text really is. Therefore the values of the line from and line to fields have to be increased or decreased by hand, until they are adjusted properly.

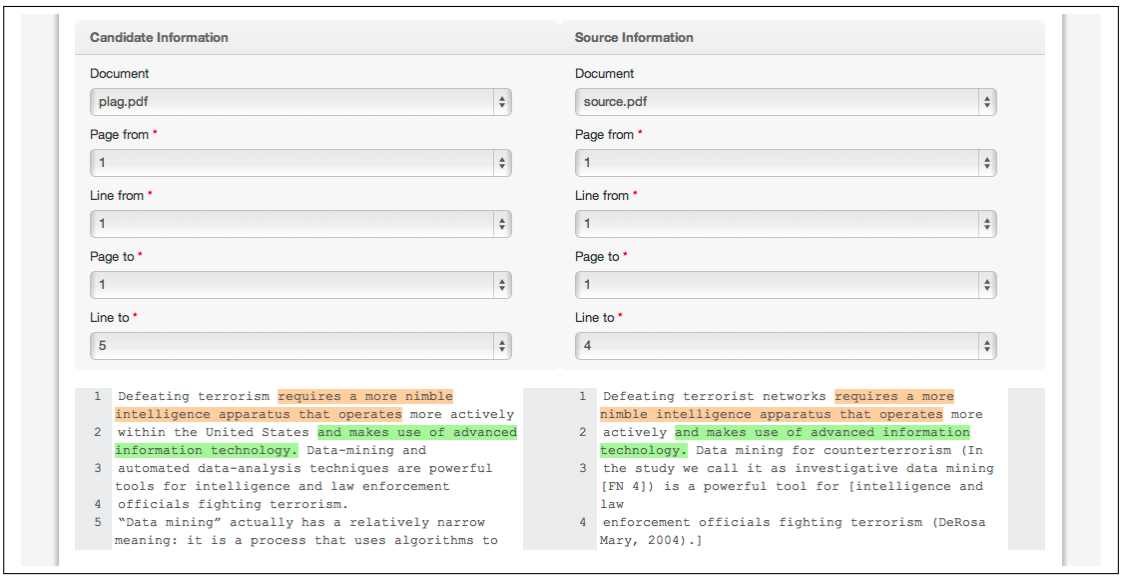

Figure 2.4.: Form for creating a fragment by hand

A more comfortable workflow

Wouldn't it be cool to select text by just marking it with the mouse and having this previousely described form being filled out automatically? We though it would, so we implemented it.

The user has to go to the document being inspected in the current case, select a page to start with and then hit the button 'Switch to two-column view for fragment creation'. At this point a second document can be selected on the right side and the similarities in both texts are once again being highlighted. In this two-column view it is also possible to iterate through the pages of the left-side or right-side document to compare page 1 from the left with page 2 from the right and page 1 on the left with page 3 on the right just by one click.

When there are sufficient similarities in an area of the page, a fragment can be created by marking the the text, then making a click with the right mouse key to open the context

menu and 'set as candidate/source of fragment'. This stores the marked text temporarily until the 'create fragment' button in the context menu is being pressed. The selection of the 'create fragment' button opens the same form as described in the section before and pre-fills it with the start and end values for page and line for both sides automatically.

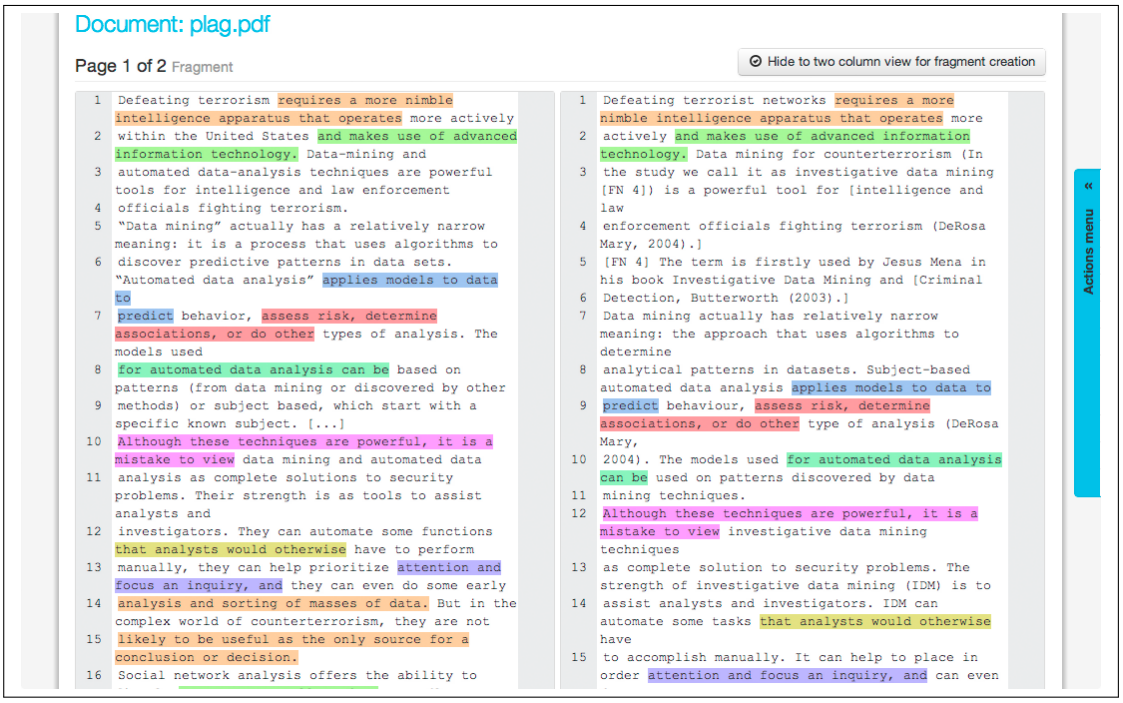

Figure 2.5.: Creating a fragment the modern way - Step 1

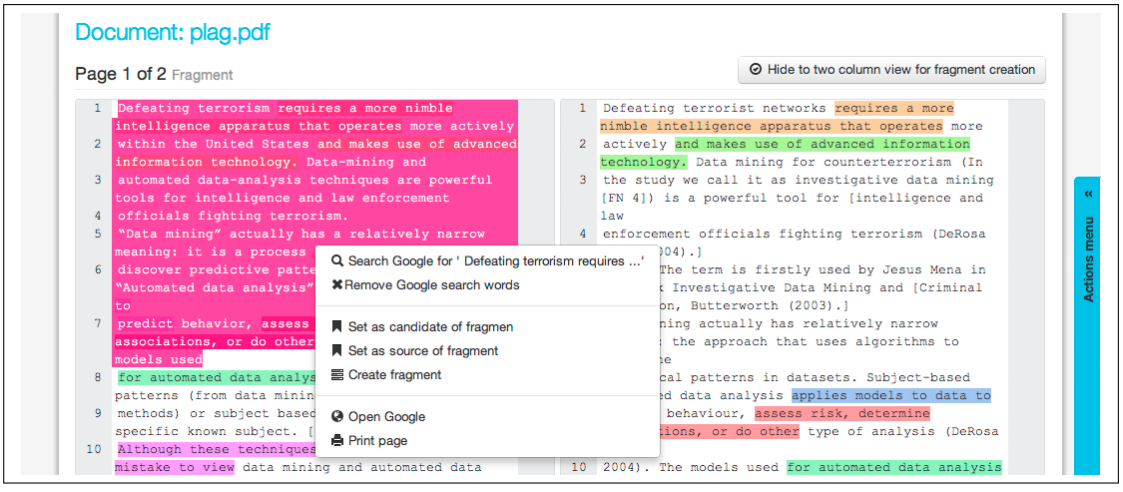

Figure 2.6.: Creating a fragment the modern way - Step 2

2.2.2. Rating a fragment

@benjamin

2.3. Barcode

@benjamin

2.4. User avatar

@elsa

2.4.1. Avatar cropping

@benjamin

2.5. Automatic Plagiarism Detection Webservice

@benjamin

2.5.1. PlagAware

@benjamin

2.6. Permission and role management

@dominik types of permission, types of roles

2.6.1. Collaborators

@benjamin

new role created from global case role

3. Showtime

4. Summary and Outlook

summary and outlook

A. Meetings

The following tables show the minutes of most of the team meetings.

Bibliography

[Google 2012] Google: *News Search Interest: plagiat*. http://www.google.com/insights/search/#q=Plagiat&geo=DE&date=1%2F2011%2013m&gprop=news&cmpt=q, 2012. – [Online; accessed 07-February-2012]