Challenge Report

Hackathon Aubay 2024

Challenge 2 - Course Search in Chamilo

Done By the Yellow Team:

- Bruno Costa
- Pedro Appel
- Luís Lemos
- Pedro Santos

Table of contents

Introduction

This report aims to detail the solution implemented to expand the course search functionality on the Chamilo platform.

Challenge Description

- **Challenge:** Enable users to find relevant courses by searching for keywords present in the course content. For example, when searching for "Angular," the user should be directed to the "Frontend" course, avoiding the "no results found" message.

- **Problem:** Current Search Limitation: By default, Chamilo only searches for exact words that appear in the course titles.

Solutions Attempted Without Success:

- Integration with Xapian**: The attempt to use Xapian for full-text search was unsuccessful due to integration challenges.

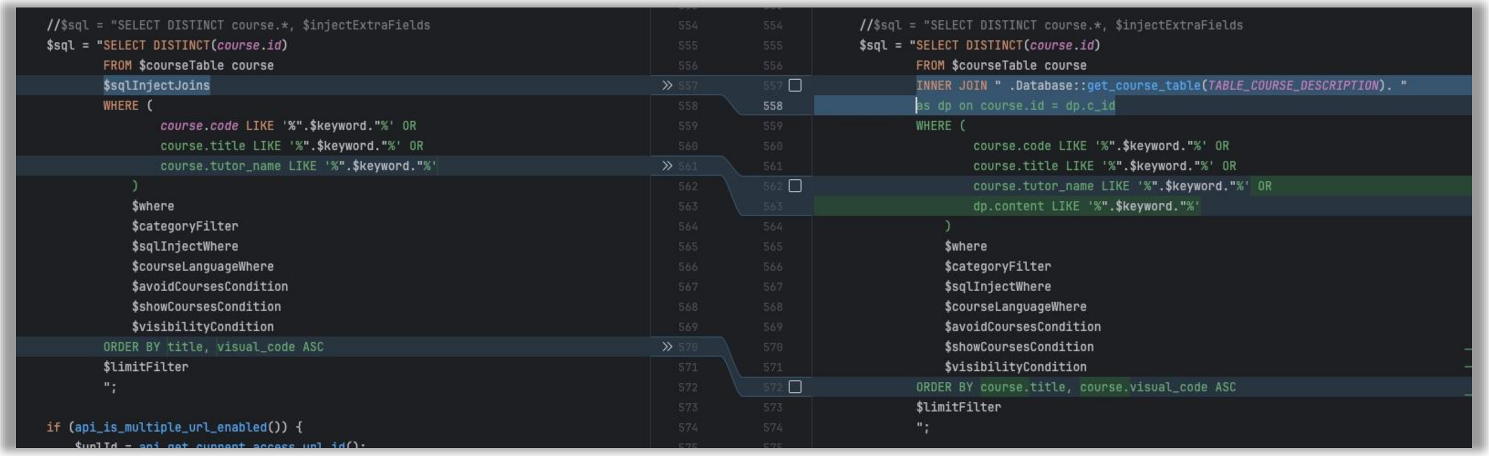
- Use of Tags**: Creating tags with keywords did not result in corresponding courses appearing in search results.

- **Proposed Solution:** To address the issue, the decision was made to directly modify the code in the `CoursesAndSessionsCatalog.class.php` class, expanding the SQL query responsible for course searches.

Implemented Solution

4.1. Code Modification

The SQL query was modified as follows:



```
//$sql = "SELECT DISTINCT course.*, $injectExtraFields
$sql = "SELECT DISTINCT(course.id)
        FROM $courseTable course
        $sqlInjectJoins
        WHERE (
                course.code LIKE '%".$keyword."%' OR
                course.title LIKE '%".$keyword."%' OR
                course.tutor_name LIKE '%".$keyword."%'
            )
            $where
            $categoryFilter
            $sqlInjectWhere
            $courseLanguageWhere
            $avoidCoursesCondition
            $showCoursesCondition
            $visibilityCondition
        ORDER BY title, visual_code ASC
        $limitFilter
        ";

if (api_is_multiple_url_enabled()) {
    $urlId = api_get_current_access_url_id();
```

```
//$sql = "SELECT DISTINCT course.*, $injectExtraFields
$sql = "SELECT DISTINCT(course.id)
        FROM $courseTable course
        INNER JOIN " .Database::get_course_table(TABLE_COURSE_DESCRIPTION). "
        as dp on course.id = dp.c_id
        WHERE (
                course.code LIKE '%".$keyword."%' OR
                course.title LIKE '%".$keyword."%' OR
                course.tutor_name LIKE '%".$keyword."%' OR
                dp.content LIKE '%".$keyword."%'
            )
            $where
            $categoryFilter
            $sqlInjectWhere
            $courseLanguageWhere
            $avoidCoursesCondition
            $showCoursesCondition
            $visibilityCondition
        ORDER BY course.title, course.visual_code ASC
        $limitFilter
        ";
```

4.2. Solution Explanation

- Join with the Course Descriptions Table:
- An INNER JOIN was performed with the `TABLE_COURSE_DESCRIPTION` table (alias `dp`), linking each course to its description through `course.id = dp.c_id`.

-Expansion of Search Criteria:

- The condition `dp.content LIKE '%".$keyword."%'` was added to the WHERE clause, allowing the search to include keywords found within the course description content.

-Preservation of Original Conditions:

- Other filters and conditions (`$where`, `$categoryFilter`, etc.) were preserved to ensure that other functionalities remained unaffected.

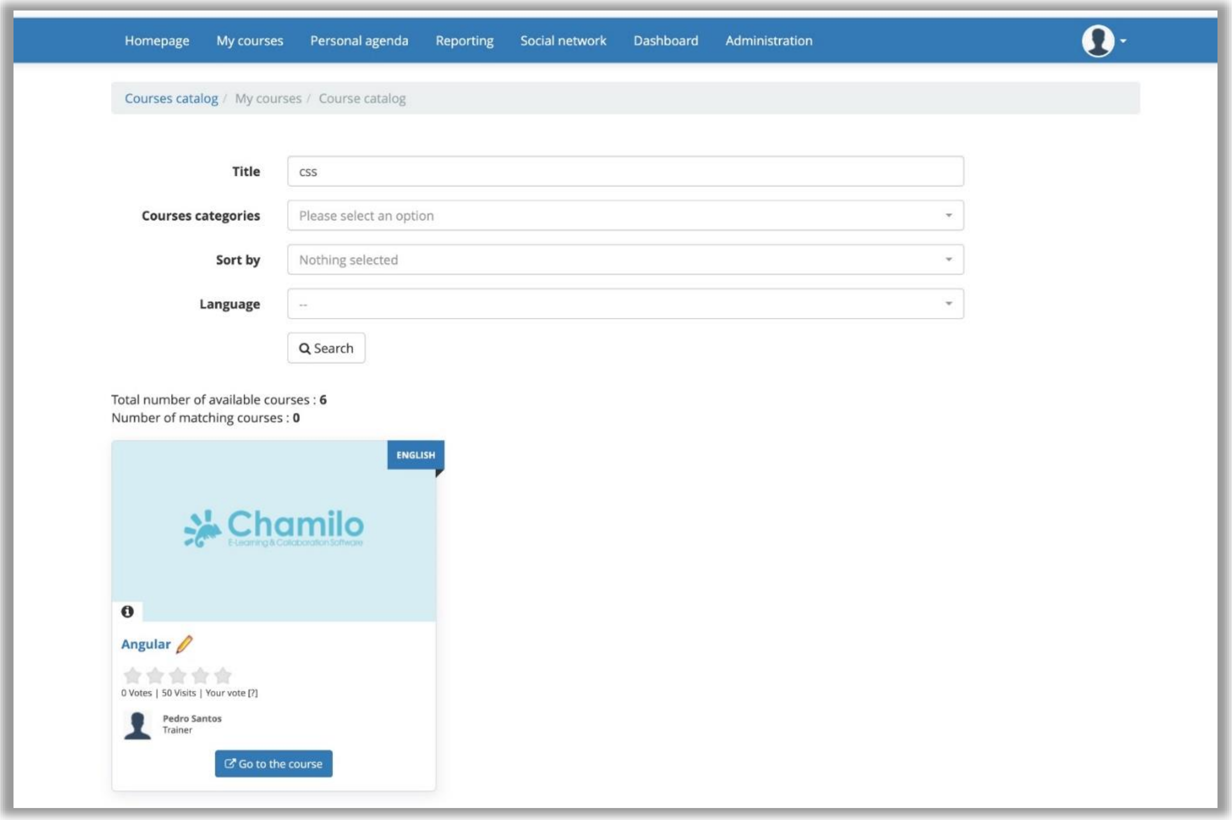
Results Obtained

Improvement in User Experience:

- Users can now find courses by searching for terms present in the course content, no longer limited to the title only.

Challenge Resolution:

- The solution fully met the initial goal, enabling the discovery of relevant courses based on specific keywords.



Title: css

Courses categories: Please select an option

Sort by: Nothing selected

Language: --

Total number of available courses : **6**
Number of matching courses : **0**

Angular

0 Votes | 50 Visits | Your vote [?]

Pedro Santos
Trainer

Conclusion

Directly modifying the SQL query proved to be an effective solution for expanding search functionality in Chamilo without the need to integrate external systems like Xapian or rely on tags. This approach allows greater flexibility and significantly improves the platform's usability.

References

Chamilo Association. (2023). Documentação do Chamilo LMS.

Oracle Corporation. (2023). Manual de Referência do MySQL 8.0: Funções de Pesquisa Full-Text.

BeezNest. (2023). Guia do Desenvolvedor Chamilo.