Comenius University in Bratislava
Faculty of mathematics, physics and informatics

Assignments Workflow in Learning Management System

Diploma thesis

2016
Bc. Tomáš Trungel

Comenius University in Bratislava
Faculty of mathematics, physics and informatics

Assignments Workflow in Learning Management System

Diploma thesis

Study programme:	Applied Informatics
Field of study:	2511 Applied Informatics
Department:	Department of Applied Informatics
Supervisor:	RNDr. Martin Homola, PhD.
Consultant:	doc. RNDr. Zuzana Kubincová, PhD.

Bratislava, 2016
Bc. Tomáš Trungel

Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

Meno a priezvisko študenta:	Bc. Tomáš Trungel
Študijný program:	Applied Computer Science (Single degree study, master II. deg., full time form)
Študijný odbor:	9.2.9. Applied Informatics
Typ záverečnej práce:	Diploma Thesis
Jazyk záverečnej práce:	English
Sekundárny jazyk:	Slovak

Title: Assignments Workflow in Learning Management System

Abstrakt:

Dátum odovzdania:

.....

Študent

By this I declare that I wrote this master thesis by myself, only with the help of the referenced literature, under the careful supervision of my thesis advisor.

. . . . . .

Acknowledgment: I would like to say thank You to my master thesis supervisor Martin Homola for his help and professional and valuable advices while I was writing this thesis.

Abstract

English abstract: TODO

Keywords: one, two, three

Abstrakt

Slovenský abstrakt: TODO

Kľúčové slová: jedno, druhé, tretie

Contents

List of Figures

Introduction

Learning management systems are becoming ubiquitous technology adopted at institutions of higher learning. Before these systems can be considered effective the user experience must be studied and analyzed to provide the optimum solution to meet pedagogical needs of both faculty and students. [11]

There are already many open-source learning management system but none of them matched our requirement to be easily extendable, easy to use and lightweight. For example Moodle, is too complicated to extend and does not allow us to do rapid development in new scenarios.

As a result of this, during the year 2014 and 2015, the learning management system Courses 2 was developed by a student, Jakub Culik under supervision of 3 teachers and multiple developers. This system is fully modular, with custom modules for assignments, notes, quises or results. Until now, this system has been used for 4 terms with about 250 users registered and X courses taught. During the usage of this system, some problems that needed to be solved arose.

Problems, solution and contribution

Development of this system was finished in the spring of 2015. Since then, various bugs were discovered that needed to be solved. One of the purposes of this work is to provide support for this system and fix these issues.

During usage of Courses 2 we descovered that it is important to enable peer review and improved submissions in assignments module. The flow is following: student is assigned an assignment and submits his solution. After that, this solution is peer reviewed and student gets feedback from other students. Then, he is enabled to submit an improved solution of this assignemnt. This work was also published as article "Peer Review Support in a Virtual Learning Environment" [8].

Another extension to Courses 2 system was support for team assignment management. Team assignemnts has appeared to be important in the process of learing. This extension was first used by "Modern approaches to webdesign" course and used by about 50 students. It was also aim of one diploma thesis to develop this extension.

Hovewer the most important part of this work was to implement easy to use web mirroring. In some courses, students were submitting URLs of their web projects, not the sources itself. This enabled the students to use technologies which they wanted to use to develop their web pages and teacher reviewed only the rendered page. The problem was that student was able to alter his project anytime he wanted to do, so there were plenty of room for cheating. We needed to develop the technology to auto backup the rendered pages in such cases. In this work, we present simple and elegant solution of this problem.

Part I

Background

Chapter 1

Courses 2 system

1.1 Introduction

In 2010, a new approach of the learning was implemented in the web design course in our faculty. A blog based learning. This approach is based on blog posts about the lectures content written by the students. These blogs are published and they are rated by the teachers and the students. The ratings of the blog are included in the final subject result. This approach forces the students to study lectures more precisely and they have to read the blog posts of their classmates, which make the students focused on the discussed problems.[1]

Since the blog portal [12] was running in the faculty, there was a space to publish the blog posts. Unfortunately, there was not a system to process the reviews and ratings of these posts. The necessity of this system occurred. One possible solution was to integrate mentioned functionality into existing blog portal, other one was to create a separate system with a possible blog integration. For our purpose there was choosen the second solution. The goal of a standalone system development was to create the customizable modular Learning management system. Such system should o er the blog portal integration (and/or provide an API for an integration with the other systems), implement the basic learning management system requirements (course management, user management, content management, etc.) and o er a logical and user-friendly structure for the next developers.

The Courses system was developed as the result of the bachelor thesis from 2013 called Integrated learning management system [8]. Courses was deployed to the production usage in 2013 and 4 courses were running on Courses system for two

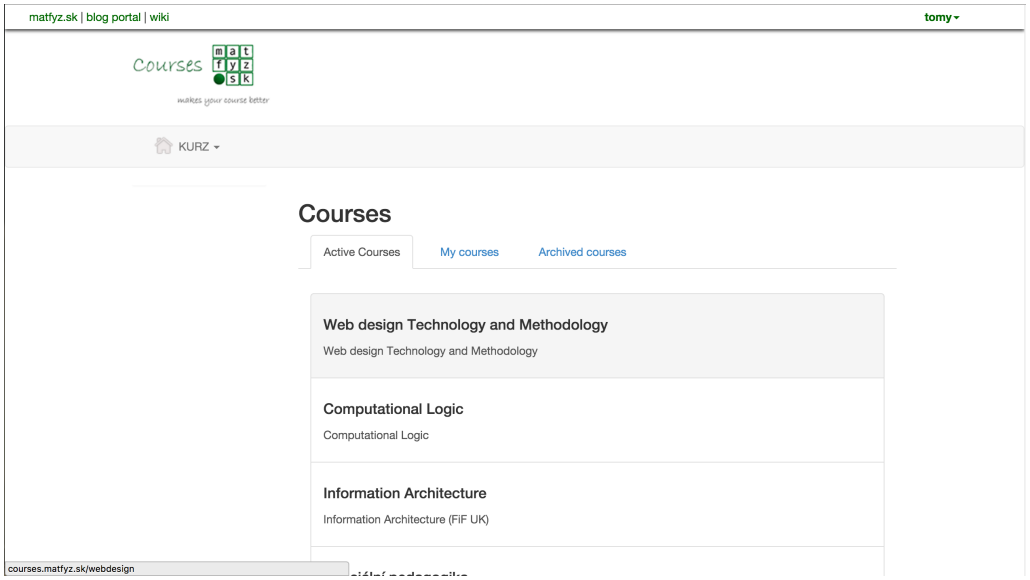

Figure 1.1: Screenshot of Courses 2 main page

semesters. The system was used by 130 users in system and 6 teachers. Although the system was designed as a fully modu- lar system with an integration with blog portal, there appeared some flaws during the production usage.

Courses 2 is a second iteration of learning management system developed at the Comenius University by Jakub Culik. This system aims at modularity with easy module management and usability for students and teachers. During its development it has been already used to teach multiple courses and increasingly gained popularity among teachers. After finishing of base development in autumn of 2014 multiple enhancements of this system and particulary assignments module were needed to be implemented.

Purpose of this chapter is to provide information about this system as it was implemented, with focus on assignments module so that we could later explain our contribution to this system.

1.2 Overview

Aim of the Courses 2 system is to provide all functionality splitted into modules. These modules can then be easily extended an improved separately, which helps reduce number of bugs and issues. After finished development of this system, 3 modules were introduced [1].

Courses 2 base module provide basic functionality in logging of user and basic course selection as in image 1.1. Notes module, on the other hand provides functionality of static content, such as lectures, labs, project tasks and information about courses. The role of results module is to provide scores for all students attending a course. The last one, assignments module is the purpose of this work, so it is described in it's own part.

1.3 Assignments module

Purpose of the assignments module is to provide student and teacher interface for management, submission and reviewing of assignments. This is the most complex module but we found that is still lacking some functionality and therefore must be extended.

1.3.1 Student's part

This module allows students to submit their solutions of assignments and review other students. We found interface of this part counterintuitive so in our future work we decided to refactor it and provide dashboard. This decision was also made after numerous complaints of other students.

1.3.2 Teacher's part

Teacher's part consists of two main functionalities: creating/editing of assignments and evaluating of student submisions. It is also important to note, that assignments module provides also interface for student reviews of other students work as is described in [1].

Creating and editing of assignments is provided by interface shown on 1.2. This interface tries to be minimalistic and intuitive but we found that it must be improved. It is important to note that each assignments can have multiple rounds, multiple review settings, notes and can be set to be FILE or URL.

The second part of teacher's administration is reviewing of user submissions. Teacher can evaluate a submission after any user submits a submission of assignment and results are than converted into notes module. Along submissions, teacher is also allowed to submit evaluation of student reviews.

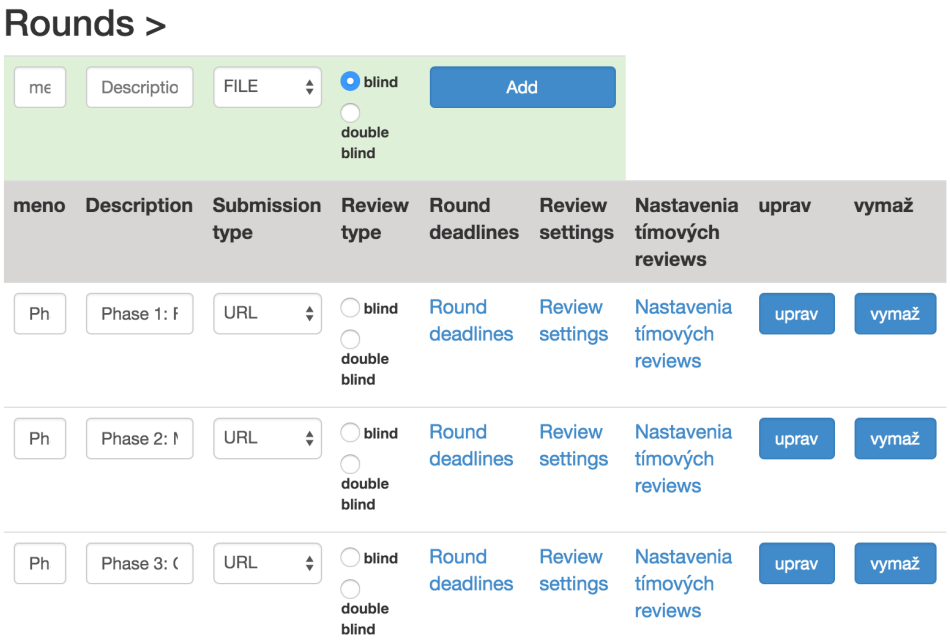

Figure 1.2: Assignment management in courses 2 system

1.4 Implementation

The whole system is split into modules so it could be easily extended or rewritten. At first we need to describe core of this system, CodeIgniter framework and then we can focus on design patterns and assignments module description.

1.4.1 CodeIgniter framework

CodeIgniter is an Application Development Framework - a toolkit - for people who build web sites using PHP [5]. One of it's goals is to provide exceptional performance of web applications along with minimum complexity and fast development. CodeIgniter was born from ExpressionEngine [4] in 2006 and since then, it increasingly gained popularity among PHP developers. In 2008 CodeIginiter became industry leader in an enviroment saturated with PHP frameworks [4] and in 2014 CodeIgniter v 2.2 was released.

This framework is licensed under custom Elislab license, which is basicaly an open source license with only a few restrictions [6]. We provide a copy of this license in medium attached to this work.

1.4.2 MVC design pattern

In an Object oriented programming (OOP) programmers often run into problems with application complexity and maintanance of code. As application becomes larger and more robust, it is ofter harder to add or modify existing features. Modifying of complex classes and unclear implementation can cause a lot of software bugs. On the other way, design patterns aim to provide reusable soulutions of common problems in software engineering and reduce this complexity by splitting functionality into specific classes with a single purpose. One of the most important desing patterns is Model-View-Controller, which is the performs as the core of the CodeIgniter framework.

Model-View-Controller (MVC) design pattern was first created in 70's by SmallTalk programmers. Since that time, MVC design idiom has become a commonplace, especially in object oriented systems [2]. We specify the motto of this pattern as to "Stop data, program logic and presentation layers being mixed up together".

To do so, MVC has to split application functionality into three layers: models, views and controllers. Each part plays specific, distinctive, non-overlapping function. This also enables programmers to easily replace these layers and extend functionality. To fully understand this pattern, we need to look at each layer separately with context to CodeIgniter framework.

Model

Model is basicaly a class where all of the business logic is stored. Basic model operations are create, update, read, delete (often referenced as CRUD) of application data plus handling of external services (such as API calls). In CodeIgniter model is the only MVC layer supposed to perform operations with database. CRUD operations are forbidden outside models to prevent programming bugs and to keep simplicity of the code base.

In Courses 2, models are connected to MySQL database which stores all the data about courses, users and teachers. There are also models, which use Oauth2 protocol to communicate with external services to authorize users from other matfyz portals. In this way, there is no more need to store passwords in MySQL database because log in of user is handled outside of application and this improves security.

```
class Admin_model extends MY_Model
{
    function __construct()
    {
        parent::__construct();
    }

    public function get_assignment($id)
    {
        $where = array (
            'course_id' => $this->cid,
            'id' => $id,
        );
        return $this->db->get_where('assignment', $where)->row_array();
    }

    public function add_assignment($name)
    {
        $input = array (
            'name'      => $name,
            'course_id' => $this->cid
        );
        return $this->db->insert('assignment', $input);
    }

    public function update_assignment($id, $name)
    {
        $where = array (
            'id'        => $id,
            'course_id' => $this->cid
        );
        $update = array (
            'name'      => $name,
            'course_id' => $this->cid
        );
        return $this->db->where($where)->update('assignment', $update);
    }

     .... (another 200 lines of code)
```

Example above presents a model of assignment in Courses application. This model is used by all of the controllers and serves as the only code accessing or editing assignment in database.

View

Views, unlike controllers and models are not supposed to contain any logic. In object-oriented terms, these will consist of sets of classes which give us "windows (such as GUI, CLI or API) [2]. Althought views are very often graphical, they don't have to be. Example of view from Courses application could be something like this:

```
<h3>Assignments</h3>

<?php foreach ($assignments as $assignment) : ?>
    <div class="alert alert-info">
        <a href="<?= $course->course_base() ?>assignments/<?= $assignment['id'] ?>">
            <?= $assignment['name'] ?>
        </a>
    </div>
<?php endforeach; ?>

<?php if (!$assignments) : ?>
```

```
    <div class="alert alert-warning">
        <?= lang('assignments_found') ?>
    </div>
<?php endif; ?>
```

Controller

A controller is an object that manipulates a view [2]. Over-simplifying a bit, controller knows all the enviroment settings (operating system, display resolution, ...) and according to these loads data from models, perform operations on them and then they are sended to a view. Controller knows the views but views does not. The same applies to models. Example of controller connecting previous model with view could be:

```
    public function run()
    {
        $assignments = $this->assignments_model->get_course_assignments();

        for ($i=0; $i<count($assignments); $i++) {
            $notes = $this->assignment_note_model->get_assigment_notes($assignments[$i]['id']);
            $assignments[$i]['notes'] = $notes;
        }

        $data = array (
            'assignments' => $assignments,
        );

        $this->layout->set_content('../modules/assignments/views/user/assignments', $data);
    }
```

As we can see, in controller, data are loaded from models, then this layer can perform some operations over them and after all, result is sent to model which is then rendered.

MVC summary

Although MVC was originaly developed for desktop computing [2] it is mostly used to build web applications. This fact is reflected in most frameworks and in CodeIgniter too. Splitting the application in three separate layers can bring simplicity, code reusability and easily switchable GUIs.

1.5 HMVC design pattern

Using MVC design pattern is excellent for building small applications. Hovewer, huge application building on pure MVC pattern brings also some disadvantages [1]. For example, large sets of views, models and controllers could be a point of confusion

and failure for many programmers. In case of larger projects, it is very needed to bring order to MVC. Solution to this can be Hierarchical Model-View-Controller design pattern (HMVC) [7].

Hierarchical Model-View-Controller design pattern (HMVC) is a direct extension of the MVC pattern that aims to solve mainly scalability issues mensioned earlier. HMVC was first described in a blog post entitled HMVC: The layered pattern for developing strong client tiers on the JavaWorld web site in July 2000 [7]. In summary, HMVC is a collection of MVC triads operating as one application. Each of this triads is treated as individual and can be rendered on it's own. This resolves into greater scalability, code reusability and most importantly, resolves the problem with hudge views in large projects.

To successfuly implement HMVC, it is crucial for application to be broken into systems [7]. Each system gets it's own views, models and controllers and is fully in control of it's own part and nothing else. In Courses 2 system, these systems are called modules (assignments, note, quiz, ..). Thanks to HMVC model, in our further work, we will only edit assignments module and can ommit descriptions and refactoring of other modules.

We also must note, that CodeIgniter framewor, core of the Course 2 system is not directly supporting HMVC design pattern. This improvement was implemented thanks to Jakub Culik [1].

1.6 Comparision of other learning management systems

In this section, we consider important to describe alternative learning management systems to Courses 2. Although many projects exists, none of them met our criterions and considerations. This section is important to realize why we had to write our own learning management system that is easily extendable and usable.

1.6.1 Moodle

Moodle is a free, online Learning Management system enabling educators to create their own private website filled with dynamic courses that extend learning, any time,

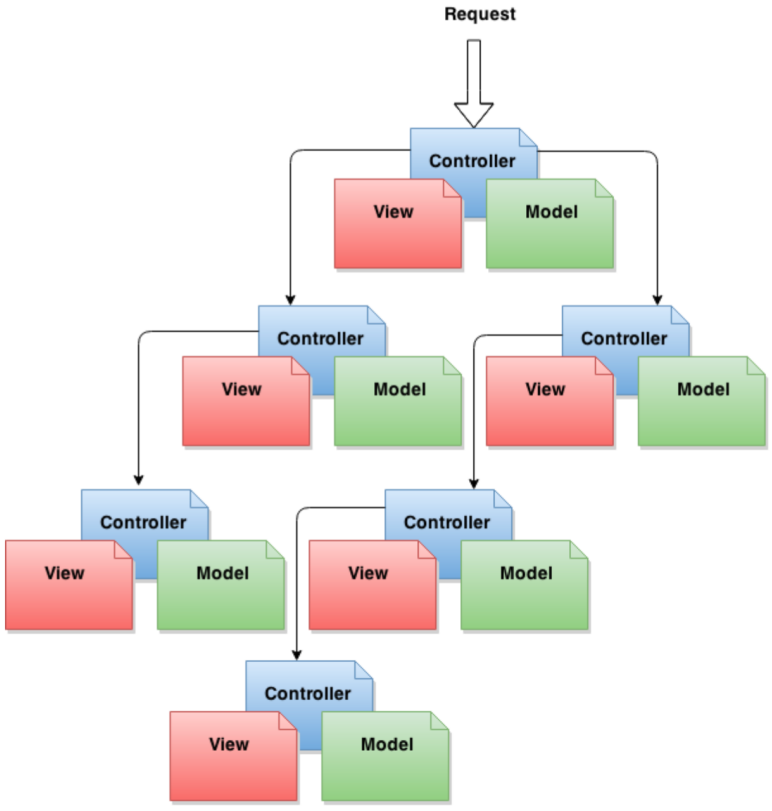

Figure 1.3: Scheme of HMVC design pattern

anywhere [3]. This system is trying to deliver full palette of functionalities, tools, modules, language support with focus on security and usability. It was developped by its Lead Developer Martin Dougiamas in 1999 with first public release in 2002. Since than moodle quickly gained popularity among famous universities like Oxford (2004) and became leading and award-winning open source LMS in 2007. [3]

Moodle as a learning platform can enhance existing learning environments. As an E-learning tool, Moodle has a wide range of standard and innovative features such as calendar and Gradebook. Moodle is a leading virtual learning environment and can be used in many types of environments such as education, training and development and in business settings [3]

Hovewer, there are few considerations to make. Over the time, Moodle has grown to be huge piece of software with hundreds of thousands lines of code and large number of extensions. Other side of this is that this makes future development very hard and time-consuming. The next disadvantage is that new pieces of code are hardly maintained and require skilled developers.

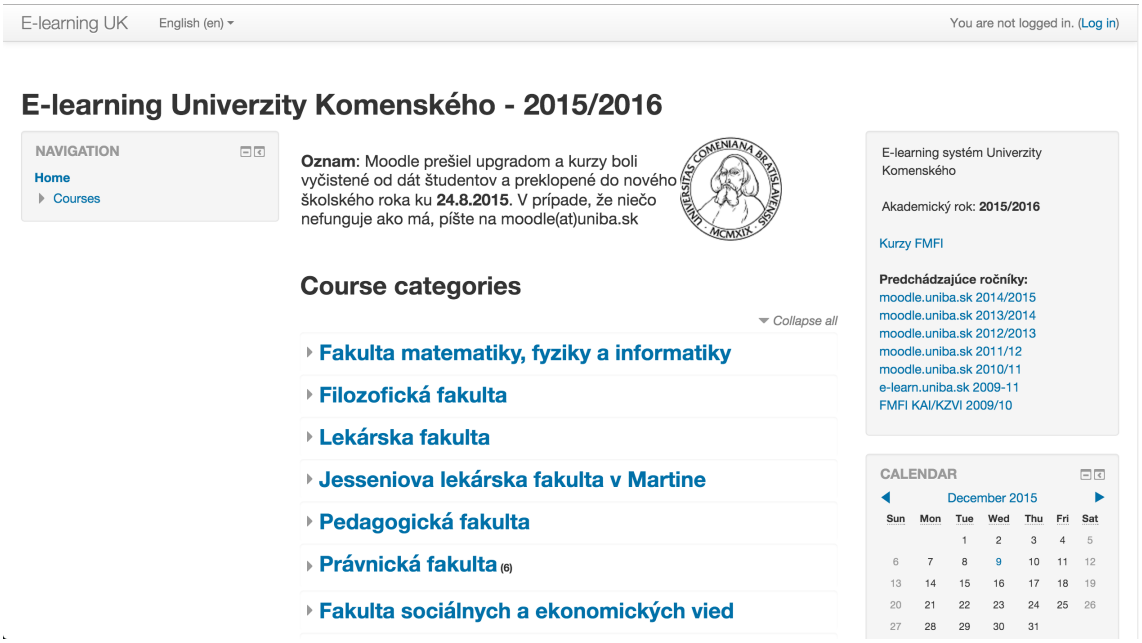

Figure 1.4: Screenshot of Comenius University Moodle

The next thing Moodle is currently not supporting are assignment rounds, assignment reviews and team assignments. There may be some plugins for this functionality but we do not consider them useful or maintained. We also found that writing plugins for moodle is not an easy task. Hovewer, on the end of this work we are going to present a Moodle plugin for mirroring URLs and saving mirrored files as user's submission.

Moodle Workshop

Workshop is a peer assessment activity with many options. Students submit their work via an on line text tool and attachments. There are two grades for a student: their own work and their peer assessments of other students' work [13]. This plugin is avaible on official Moodle site and can work with many versions of Moodle.

We found, that this plugin aims to support same functionality as we in Moodle. We appreciate this opportunity but does not consider it sufficient for our purposes.

1.6.2 Blackboard

Blackboard Learn (previously the Blackboard Learning Management System), is a virtual learning environment and course management system developed by Blackboard Inc. It is Web-based server software which features course management,

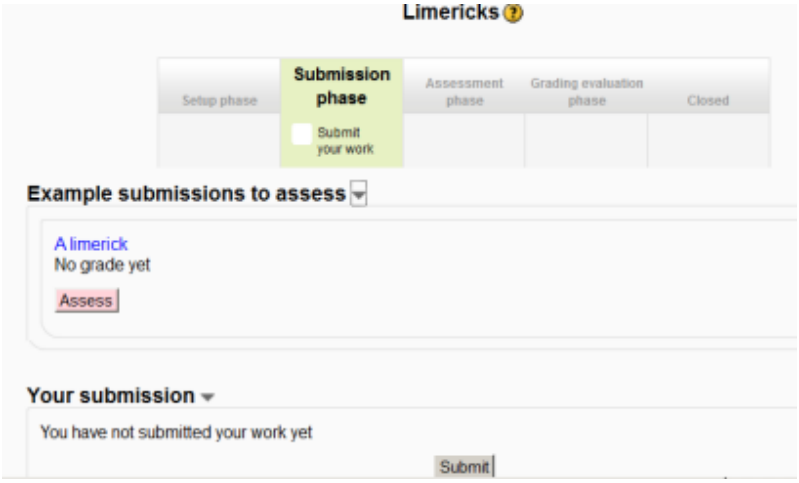

Figure 1.5: Moodle workshop plugin

customizable open architecture, and scalable design that allows integration with student information systems and authentication protocols. It may be installed on local servers or hosted by Blackboard ASP Solutions. Its main purposes are to add online elements to courses traditionally delivered face-to-face and to develop completely online courses with few or no face-to-face meetings. [9]

According to research, Blackboard received 41 percent market share as of fall 2013 among other learning management system. This makes Blackboard the most used tool in industry

Hovewer, Blackboard brings many disadvantages. Mainly, Blackboard is not avaible as open source software and is sold under custom license. Other issues with Blackboard LMS are several legal issues, including faulty patent rights claims. We consider this facts to be enough to not use Blackboard.

1.6.3 Desire2Learn

Desire2Learn was founded in 1999 by president and CEO John Baker [10]. This LMS aims to be online teaching and learning platform that's easy, flexible, and smart. This technology is used by more than 1,100 clients and 15 million learners in higher education, K–12, healthcare, government, and the enterprise sector.

Hovewer, Desire2Learn is not an option for our purposes. This system is commercial and not open source so it is imposible to use it for our future work.

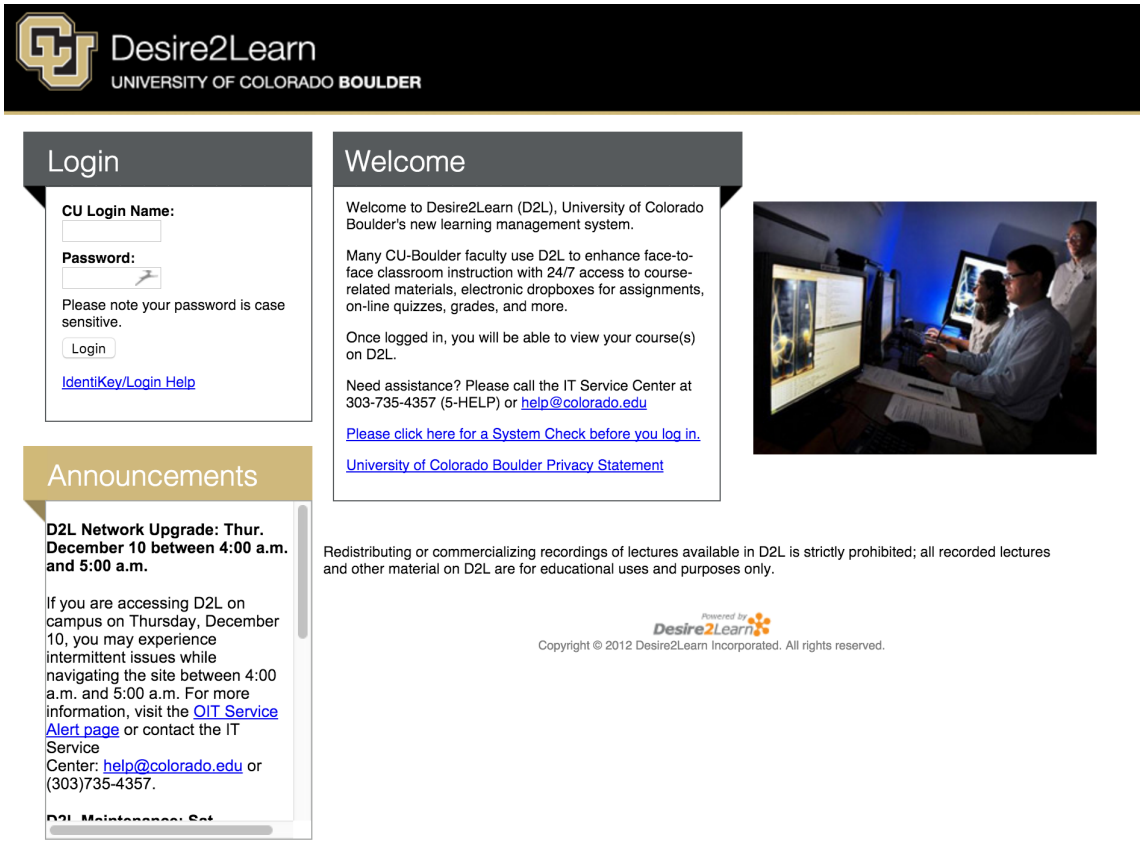

Figure 1.6: Desire2Learn Screenshot

1.6.4 Conclusion

We presented multiple alternative learning management systems. All of them were robust, secure and maintained systems hovewer the complexicity was fairly great so we consider developing our own LMS to be the best option. The next important fact in our decision process was that many of these LMS are not licensed under an open source license.

End

TODO

Bibliography

[1] Jakub Culik. Learning management system for open web-based learning. 2015.

[2] John Deacon. Model-view-controller (mvc) architecture. *Online][Citado em: 10 de março de 2006.] http://www.jdl.co.uk/briefings/MVC.pdf*, 2009.

[3] Moodle Docs. Moodle features. 2015.

[4] Elislab. A brief history of codeigniter, 2014.

[5] Elislab. Codeigniter framework user guide, 2014.

[6] Elislab. Codeigniter license agreement, 2014.

[7] Sam Freysnett. Scaling web applications with hmvc. 2010.

[8] Martin Homola, Zuzana Kubincová, Jakub Čulík, and Tomáš Trungel. Peer review support in a virtual learning environment. 2016.

[9] Blackboard Inc. Blackboard. 2015.

[10] Brightspace Inc. Desire2learn, 2015.

[11] Michael Machado and Eric Tao. Blackboard vs. moodle: Comparing user experience of learning management systems. pages S4J–7, 2007.

[12] Martin Rejda. Redesign of blog portal. 2008.

[13] Moodle Workshop. Moodle workshop module. 2015.