# Manual tecnico

## Sistema interactivo de aperndizaje a distancia

SIAD

Integrantes de equipo:

- Luis Abdul Chávez Bustamante – 256910
- Omar Fernando Garcia Nápoles – 256944
- Darío Andrés Rosales Pérez Rodríguez –251961
- Oscar Ernesto Flores Jasso - 256935

# Contenido

## Introducción

Este documento contiene toda la información sobre los recursos de la plataforma "SIAD" (Sistema Interactivo de Aprendizaje a Distancia) ya que llevan una descripción muy específica sobre las características físicas y técnicas de cada elemento: ya sea el manejo de cursos, de estudiantes, calificaciones, etc.

El SIAD surge de la gran plusvalía del internet además de que es más sencillo que un estudiante pueda tener acceso a este por lo tanto se decidió hacer esta plataforma para que pueda haber aprendizaje sea donde sea

# Organización

## Descripción

El sistema funciona mediante la alta de alumnos y maestros  ya sean uno u otro podrán tener diferentes accesos a los cursos es decir si se es un maestro este podrá agregar un recurso al mismo o por ejemplo de igual manera un examen. Pero en dado caso que sea un estudiante este podrá ver mas no editar los recursos además de hacer los exámenes y no editarlos como lo haría el profesor.

## Antecedentes

Nuestro sistema está basado en la plataforma “moodle” ya que de igual manera nos sirve para la educación en línea, nuestro sistema puede contener tantos alumnos y profesores es decir un usuario puede agregar cursos o participar en ellos.

## Misión y visión

Nuestra empresa es para poder conectar gracias a internet todos los cursos posibles para que podamos realizar una serie de cursos y que todo el mundo pueda aprender más acerca de carreras tecnológicas o más si es así como lo desea.  Nuestra empresa esta para todo el mundo que le guste conocer más allá de los rasgos avanzados, intermedios o más básicos de algún tema en específico o bien si esta persona es muy buena él/ella misma puede iniciar un curso en la página del “SIAD” para poder compartir sus conocimientos con todo aquella persona que le interese.

## Objetivos generales

Nuestro principal objetivo es la unión de los diferentes tipos de cursos en un solo lugar para esto gracias a todos los usuarios que se vallan registrando pueden ir abriendo algún curso de algún tema que deseen enseñar.

# Sistema

## Descripción y Delimitación del sistema

La plataforma SIAD se encargara de cursos en línea para que el estudiante pueda aprender sin tener que salir de casa además esta manejara a los usuarios tanto como alumnos como maestros mostrando un kardex con los diferentes cursos que han tomado hasta la fecha junto con sus respectivas calificaciones del mismo además de que se encargara en dado caso de querer agregar un curso, se podrán agregar recursos para el completo entendimiento del tema y por supuesto se podrán agregar exámenes a los alumnos para que se pueda comprobar si realmente han quedado claros todos los temas, en este se pueden agregar preguntas del tipo selección, selección múltiple, pregunta abierta y una vez publicado el examen aparecerán las calificaciones de los alumnos. En dado caso que sea el estudiante podrá revisar todos los recursos impartidos por el maestro y hacer el emanen debido al curso una vez finalizado el examen nos dará las calificaciones que hemos tenido en los exámenes de ese curso para poder tener un control de las mismas.

## Objetivos generales

Brindar un servicio de cursos en línea el cual sean definidos por los mismos usuarios es decir pueden ser tanto alumnos como maestros si les gustaría compartir el conocimiento que ellos tienen.

## Descripción de usuarios

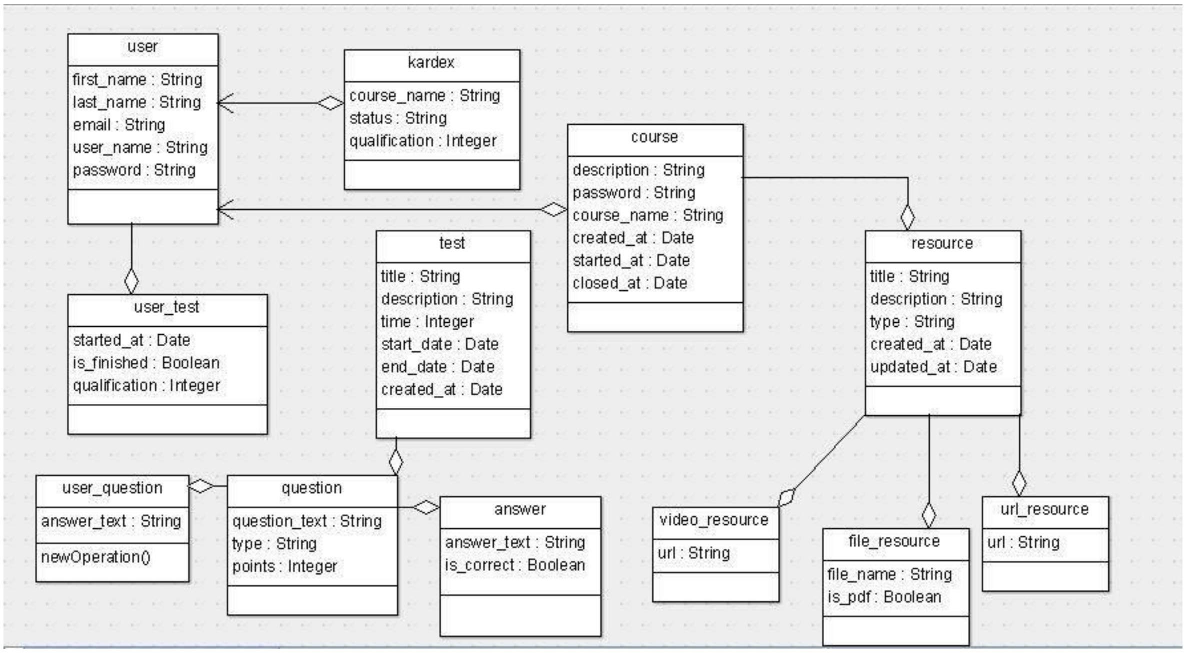
La plataforma SIAD dispondrá de dos tipos de usuarios: los alumnos y los maestros, los alumnos se convertirán en maestros una vez abran un curso

Maestros.- Podrán hacer manejo de los diferentes tipos de recursos que irán publicando durante el transcurso del curso. Además tendrán un control para llevar a cabo algún examen es decir podrán crear sus propios exámenes ya sea con preguntas de selección o de selección múltiple o bien preguntas abiertas puede publicarlo a la fecha y hora que el desee además de que el maestro podrá finalizar el curso cuando este lo desee una vez hecho esto los alumnos recibirán sus calificaciones en el kardex.

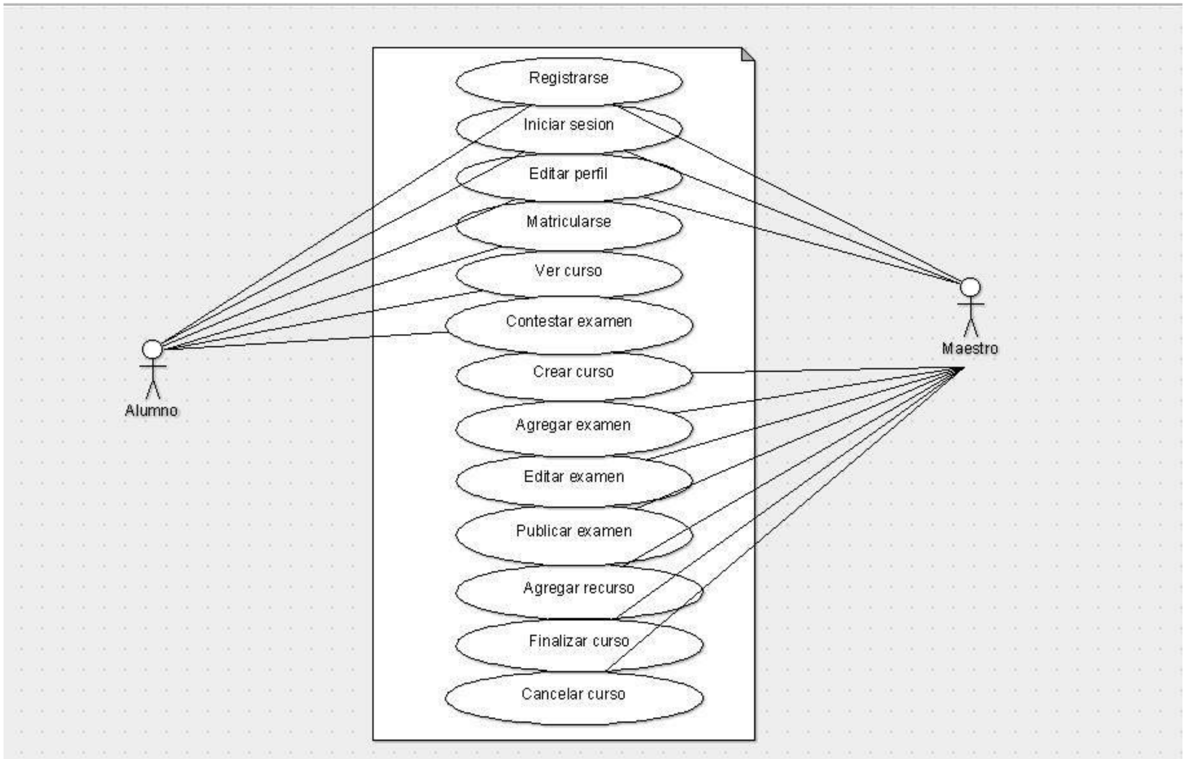
Alumnos.- Los alumnos inscritos en algún curso podrán ver pero no modificar los recursos disponibles en los mismos y a la hora de poder hacer un examen aparecerá un reloj el cual será el límite de tiempo que tienen para hacerlo, una vez finalizado el examen podrán ver su calificación y una vez finalizado un curso los mismos alumnos podrán entrar al kardex y podrán ver sus calificaciones.

# Especificación de requerimientos

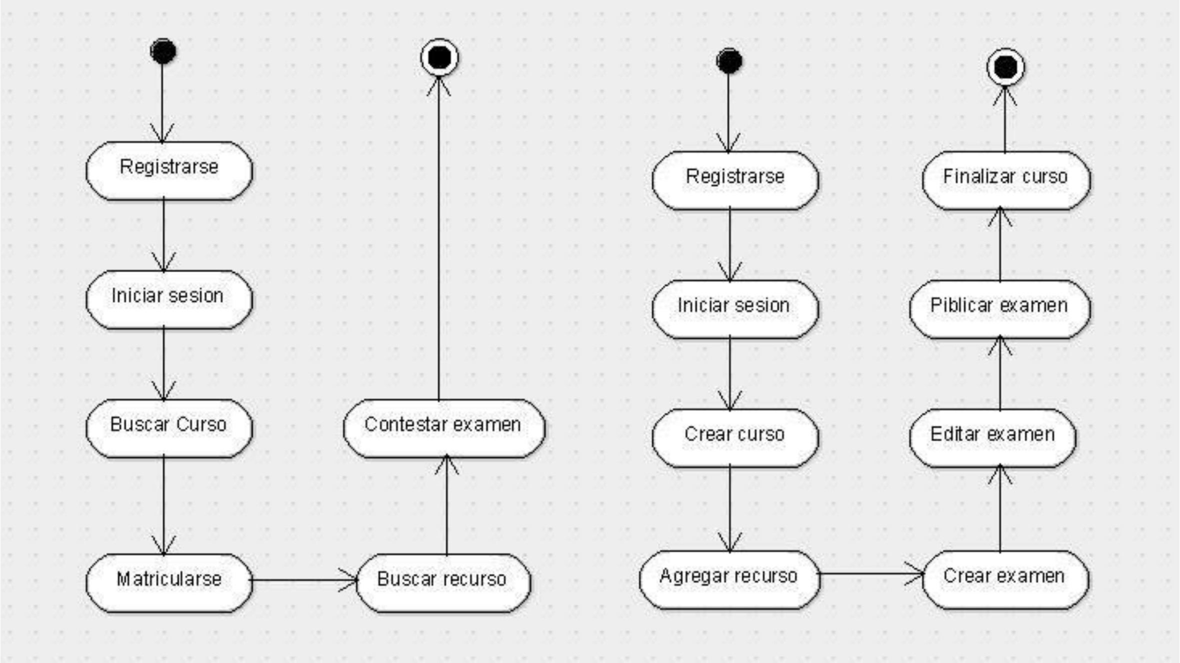
## Diagrama de clases



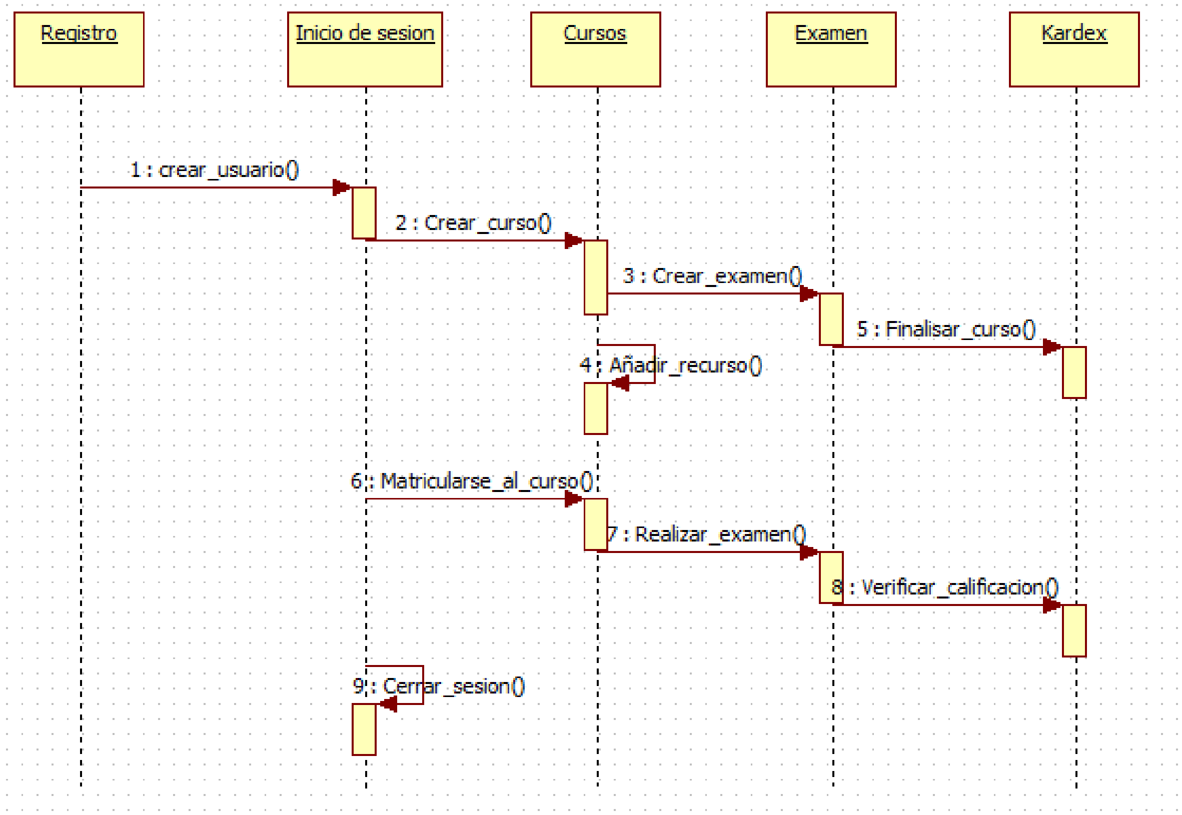
## Diagrama de casos de uso



## Diagrama de actividades

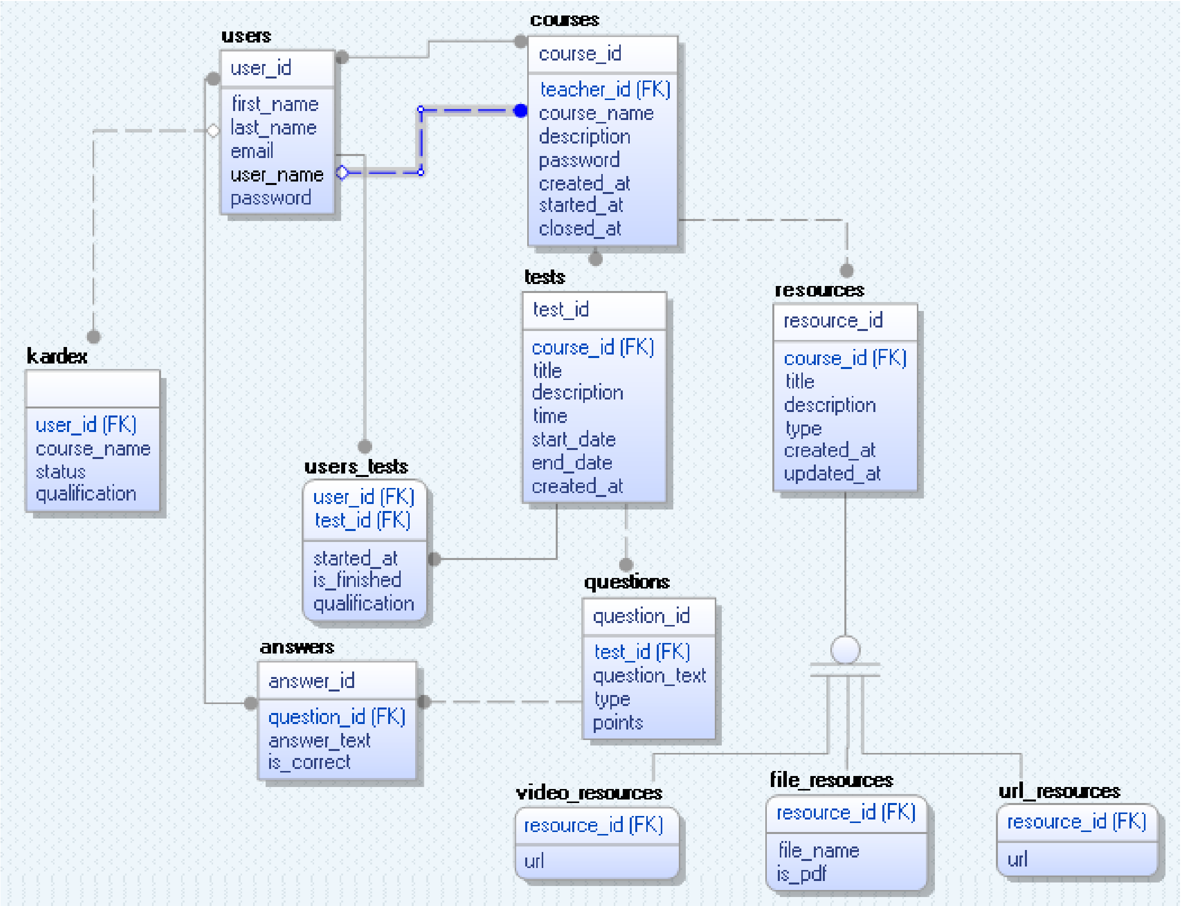


## Diagramas de secuencia



# Diseño de la base de datos

## Modelado de la base de datos con Erwin



## Esquema de la base de datos

```
-- phpMyAdmin SQL Dump
-- version 4.0.4.1
-- http://www.phpmyadmin.net
--
-- Servidor: 127.0.0.1
-- Tiempo de generaciÃ³n: 26-11-2013 a las 03:40:36
-- VersiÃ³n del servidor: 5.5.32
-- VersiÃ³n de PHP: 5.4.19

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";




/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS
*/;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;

--
-- Base de datos: `siad`
--
CREATE DATABASE IF NOT EXISTS `siad` DEFAULT CHARACTER SET utf8
COLLATE utf8_unicode_ci;
USE `siad`;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `answers`
--

DROP TABLE IF EXISTS `answers`;
CREATE TABLE IF NOT EXISTS `answers` (
  `answer_id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `question_id` int(10) unsigned NOT NULL,
  `answer_text` text COLLATE utf8_unicode_ci NOT NULL,
  `is_correct` tinyint(1) NOT NULL DEFAULT '0',
  PRIMARY KEY (`answer_id`),
  KEY `answers_question_id_foreign` (`question_id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO_INCREMENT=46 ;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `courses`
--
```

```
DROP TABLE IF EXISTS `courses`;
CREATE TABLE IF NOT EXISTS `courses` (
  `course_id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `teacher_id` int(10) unsigned NOT NULL,
  `course_name` varchar(120) COLLATE utf8_unicode_ci NOT NULL,
  `description` text COLLATE utf8_unicode_ci NOT NULL,
  `password` varchar(20) COLLATE utf8_unicode_ci DEFAULT NULL,
  `created_at` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',
  `started_at` timestamp NULL DEFAULT NULL,
  `closed_at` timestamp NULL DEFAULT NULL,
  PRIMARY KEY (`course_id`),
  KEY `courses_teacher_id_foreign` (`teacher_id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO_INCREMENT=9 ;


-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `file_resources`
--

DROP TABLE IF EXISTS `file_resources`;
CREATE TABLE IF NOT EXISTS `file_resources` (
  `resource_id` int(10) unsigned NOT NULL,
  `file_name` text COLLATE utf8_unicode_ci NOT NULL,
  `is_pdf` tinyint(1) NOT NULL DEFAULT '0',
  PRIMARY KEY (`resource_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `kardex`
--

DROP TABLE IF EXISTS `kardex`;
CREATE TABLE IF NOT EXISTS `kardex` (
  `user_id` int(10) unsigned NOT NULL,
  `course_name` varchar(50) COLLATE utf8_unicode_ci NOT NULL,
  `status` varchar(40) COLLATE utf8_unicode_ci NOT NULL,
  `qualification` double DEFAULT NULL,
  KEY `kardex_user_id_foreign` (`user_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `migrations`
--
```

```
DROP TABLE IF EXISTS `migrations`;
CREATE TABLE IF NOT EXISTS `migrations` (
  `migration` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `batch` int(11) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `questions`
--

DROP TABLE IF EXISTS `questions`;
CREATE TABLE IF NOT EXISTS `questions` (
  `question_id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `test_id` int(10) unsigned NOT NULL,
  `question_text` text COLLATE utf8_unicode_ci NOT NULL,
  `type` enum('open','choice','multiple') COLLATE utf8_unicode_ci
NOT NULL DEFAULT 'open',
  `points` int(10) unsigned NOT NULL,
  PRIMARY KEY (`question_id`),
  KEY `questions_test_id_foreign` (`test_id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO_INCREMENT=13 ;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `resources`
--

DROP TABLE IF EXISTS `resources`;
CREATE TABLE IF NOT EXISTS `resources` (
  `resource_id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `course_id` int(10) unsigned NOT NULL,
  `title` varchar(120) COLLATE utf8_unicode_ci NOT NULL,
  `description` text COLLATE utf8_unicode_ci,
  `type` enum('video','file','url') COLLATE utf8_unicode_ci NOT
NULL DEFAULT 'url',
  `created_at` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',
  `updated_at` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',
  PRIMARY KEY (`resource_id`),
  KEY `resources_course_id_foreign` (`course_id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO_INCREMENT=11 ;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `tests`
```

```
--

DROP TABLE IF EXISTS `tests`;
CREATE TABLE IF NOT EXISTS `tests` (
  `test_id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `course_id` int(10) unsigned NOT NULL,
  `title` varchar(120) COLLATE utf8_unicode_ci NOT NULL,
  `description` text COLLATE utf8_unicode_ci,
  `time` int(10) unsigned NOT NULL,
  `start_date` timestamp NULL DEFAULT NULL,
  `end_date` timestamp NULL DEFAULT NULL,
  `created_at` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',
  PRIMARY KEY (`test_id`),
  KEY `tests_course_id_foreign` (`course_id`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO_INCREMENT=12 ;



-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `url_resources`
--

DROP TABLE IF EXISTS `url_resources`;
CREATE TABLE IF NOT EXISTS `url_resources` (
  `resource_id` int(10) unsigned NOT NULL,
  `url` text COLLATE utf8_unicode_ci NOT NULL,
  PRIMARY KEY (`resource_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `users`
--

DROP TABLE IF EXISTS `users`;
CREATE TABLE IF NOT EXISTS `users` (
  `user_id` int(10) unsigned NOT NULL AUTO_INCREMENT,
  `first_name` varchar(20) COLLATE utf8_unicode_ci NOT NULL,
  `last_name` varchar(20) COLLATE utf8_unicode_ci NOT NULL,
  `email` varchar(255) COLLATE utf8_unicode_ci NOT NULL,
  `user_name` varchar(15) COLLATE utf8_unicode_ci NOT NULL,
  `password` varchar(60) COLLATE utf8_unicode_ci NOT NULL,
  PRIMARY KEY (`user_id`),
  UNIQUE KEY `users_email_unique` (`email`),
  UNIQUE KEY `users_user_name_unique` (`user_name`)
) ENGINE=InnoDB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO_INCREMENT=7 ;
```

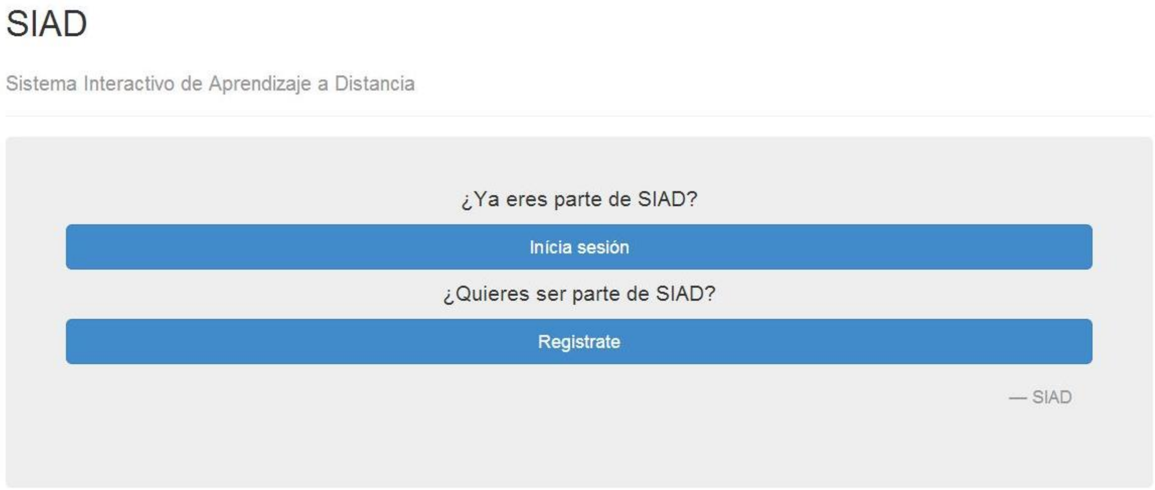
```
-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `users_courses`
--

DROP TABLE IF EXISTS `users_courses`;
CREATE TABLE IF NOT EXISTS `users_courses` (
  `user_id` int(10) unsigned NOT NULL,
  `course_id` int(10) unsigned NOT NULL,
  PRIMARY KEY (`user_id`,`course_id`),
  KEY `users_courses_course_id_foreign` (`course_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;




-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `users_questions`
--


DROP TABLE IF EXISTS `users_questions`;
CREATE TABLE IF NOT EXISTS `users_questions` (
  `user_id` int(10) unsigned NOT NULL,
  `question_id` int(10) unsigned NOT NULL,
  `answer_text` text COLLATE utf8_unicode_ci NOT NULL,
  KEY `users_questions_user_id_foreign` (`user_id`),
  KEY `users_questions_question_id_foreign` (`question_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `users_tests`
--

DROP TABLE IF EXISTS `users_tests`;
CREATE TABLE IF NOT EXISTS `users_tests` (
  `user_id` int(10) unsigned NOT NULL,
  `test_id` int(10) unsigned NOT NULL,
  `started_at` timestamp NULL DEFAULT NULL,
  `is_finished` tinyint(1) NOT NULL DEFAULT '0',
  `qualification` double NOT NULL DEFAULT '0',
  PRIMARY KEY (`user_id`,`test_id`),
  KEY `users_tests_test_id_foreign` (`test_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

-- --------------------------------------------------------

--
-- Estructura de tabla para la tabla `video_resources`
--

DROP TABLE IF EXISTS `video_resources`;
CREATE TABLE IF NOT EXISTS `video_resources` (
  `resource_id` int(10) unsigned NOT NULL,
  `url` text COLLATE utf8_unicode_ci NOT NULL,
  PRIMARY KEY (`resource_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci;

--
-- Restricciones para tablas volcadas
--



--
-- Filtros para la tabla `answers`
--
ALTER TABLE `answers`
  ADD CONSTRAINT `answers_question_id_foreign` FOREIGN KEY
(`question_id`) REFERENCES `questions` (`question_id`) ON DELETE
CASCADE ON UPDATE CASCADE;

--
-- Filtros para la tabla `courses`
--

ALTER TABLE `courses`
  ADD CONSTRAINT `courses_teacher_id_foreign` FOREIGN KEY
(`teacher_id`) REFERENCES `users` (`user_id`) ON DELETE CASCADE ON
UPDATE CASCADE;

--
-- Filtros para la tabla `file_resources`
--
ALTER TABLE `file_resources`
  ADD CONSTRAINT `file_resources_resource_id_foreign` FOREIGN KEY
(`resource_id`) REFERENCES `resources` (`resource_id`) ON DELETE
CASCADE ON UPDATE CASCADE;

--
-- Filtros para la tabla `kardex`
--
ALTER TABLE `kardex`
```

```
  ADD CONSTRAINT `kardex_user_id_foreign` FOREIGN KEY (`user_id`)
REFERENCES `users` (`user_id`) ON DELETE CASCADE ON UPDATE
CASCADE;

--
-- Filtros para la tabla `questions`
--
ALTER TABLE `questions`
  ADD CONSTRAINT `questions_test_id_foreign` FOREIGN KEY
(`test_id`) REFERENCES `tests` (`test_id`) ON DELETE CASCADE ON
UPDATE CASCADE;

--
-- Filtros para la tabla `resources`
--
ALTER TABLE `resources`
  ADD CONSTRAINT `resources_course_id_foreign` FOREIGN KEY
(`course_id`) REFERENCES `courses` (`course_id`) ON DELETE CASCADE
ON UPDATE CASCADE;




--
-- Filtros para la tabla `tests`
--
ALTER TABLE `tests`
  ADD CONSTRAINT `tests_course_id_foreign` FOREIGN KEY
(`course_id`) REFERENCES `courses` (`course_id`) ON DELETE CASCADE
ON UPDATE CASCADE;

--
-- Filtros para la tabla `url_resources`
--
ALTER TABLE `url_resources`
  ADD CONSTRAINT `url_resources_resource_id_foreign` FOREIGN KEY
(`resource_id`) REFERENCES `resources` (`resource_id`) ON DELETE
CASCADE ON UPDATE CASCADE;

--
-- Filtros para la tabla `users_courses`
--
ALTER TABLE `users_courses`
  ADD CONSTRAINT `users_courses_course_id_foreign` FOREIGN KEY
(`course_id`) REFERENCES `courses` (`course_id`) ON DELETE CASCADE
ON UPDATE CASCADE,
  ADD CONSTRAINT `users_courses_user_id_foreign` FOREIGN KEY
(`user_id`) REFERENCES `users` (`user_id`) ON DELETE CASCADE ON
UPDATE CASCADE;

--
```

```
-- Filtros para la tabla `users_questions`
--
ALTER TABLE `users_questions`
  ADD CONSTRAINT `users_questions_question_id_foreign` FOREIGN KEY
(`question_id`) REFERENCES `questions` (`question_id`) ON DELETE
CASCADE ON UPDATE CASCADE,
  ADD CONSTRAINT `users_questions_user_id_foreign` FOREIGN KEY
(`user_id`) REFERENCES `users` (`user_id`) ON DELETE CASCADE ON
UPDATE CASCADE;

--
-- Filtros para la tabla `users_tests`
--
ALTER TABLE `users_tests`
  ADD CONSTRAINT `users_tests_test_id_foreign` FOREIGN KEY
(`test_id`) REFERENCES `tests` (`test_id`) ON DELETE CASCADE ON
UPDATE CASCADE,
  ADD CONSTRAINT `users_tests_user_id_foreign` FOREIGN KEY
(`user_id`) REFERENCES `users` (`user_id`) ON DELETE CASCADE ON
UPDATE CASCADE;




--
-- Filtros para la tabla `video_resources`
--
ALTER TABLE `video_resources`
  ADD CONSTRAINT `video_resources_resource_id_foreign` FOREIGN KEY
(`resource_id`) REFERENCES `resources` (`resource_id`) ON DELETE
CASCADE ON UPDATE CASCADE;

/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

# Diseño de la interfaz de la aplicación

## Inicio



SIAD

Sistema Interactivo de Aprendizaje a Distancia

¿Ya eres parte de SIAD?

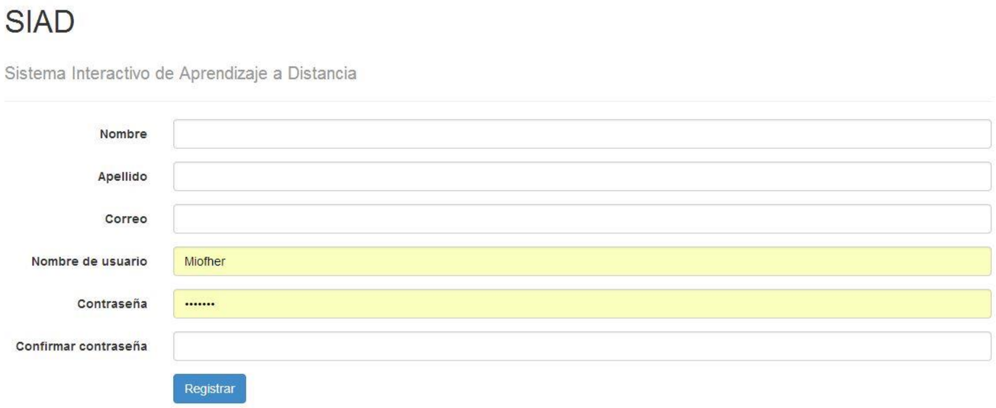
Inícia sesión

¿Quieres ser parte de SIAD?

Registrate

— SIAD

## Registros



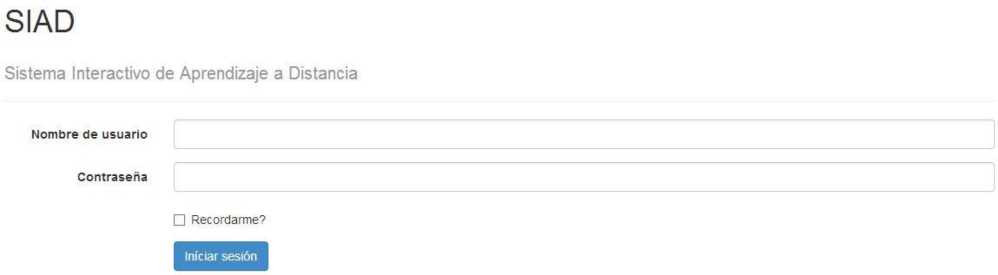
SIAD

Sistema Interactivo de Aprendizaje a Distancia

| | |
|---|---|
| Nombre | |
| Apellido | |
| Correo | |
| Nombre de usuario | Miofher |
| Contraseña | ••••••• |
| Confirmar contraseña | |

Registrar

## Inicio de sesión



SIAD

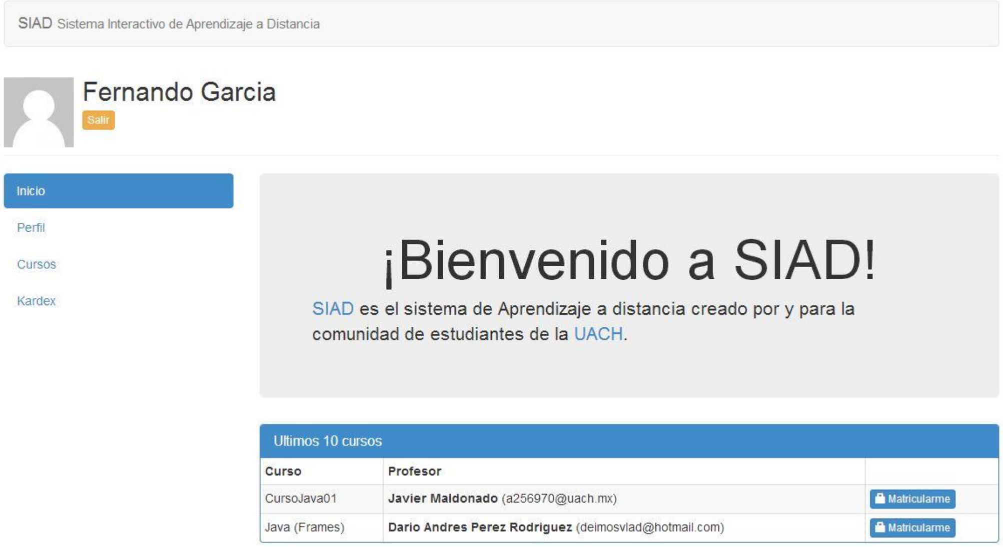
Sistema Interactivo de Aprendizaje a Distancia

| | |
|---|---|
| Nombre de usuario | |
| Contraseña | |

□ Recordarme?

Iniciar sesión

## Menú Principal



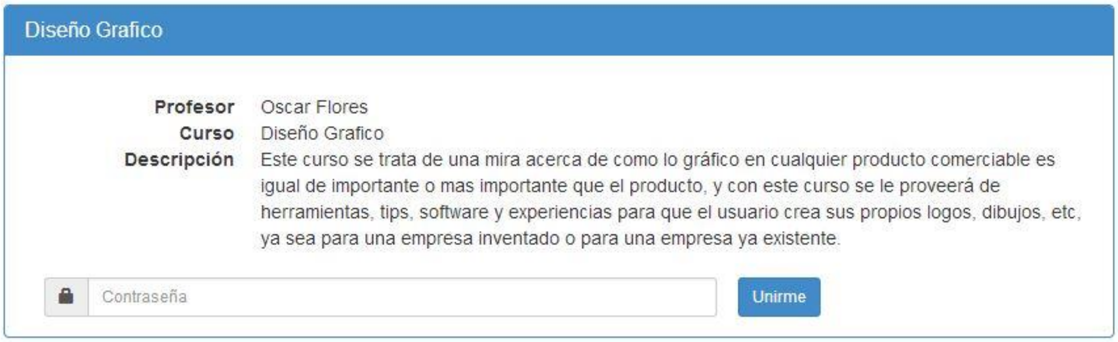
¡Bienvenido a SIAD!

SIAD es el sistema de Aprendizaje a distancia creado por y para la comunidad de estudiantes de la UACH.

Ultimos 10 cursos

| Curso | Profesor | |
|---|---|---|
| CursoJava01 | **Javier Maldonado** (a256970@uach.mx) | Matricularme |
| Java (Frames) | **Dario Andres Perez Rodriguez** (deimosvlad@hotmail.com) | Matricularme |

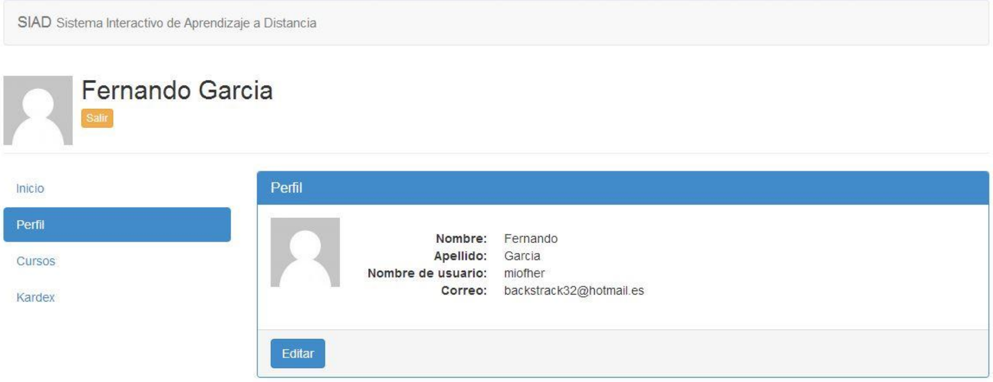
## Curso



Diseño Grafico

| | |
|---|---|
| **Profesor** | Oscar Flores |
| **Curso** | Diseño Grafico |
| **Descripción** | Este curso se trata de una mira acerca de como lo gráfico en cualquier producto comerciable es igual de importante o mas importante que el producto, y con este curso se le proveerá de herramientas, tips, software y experiencias para que el usuario crea sus propios logos, dibujos, etc, ya sea para una empresa inventado o para una empresa ya existente. |

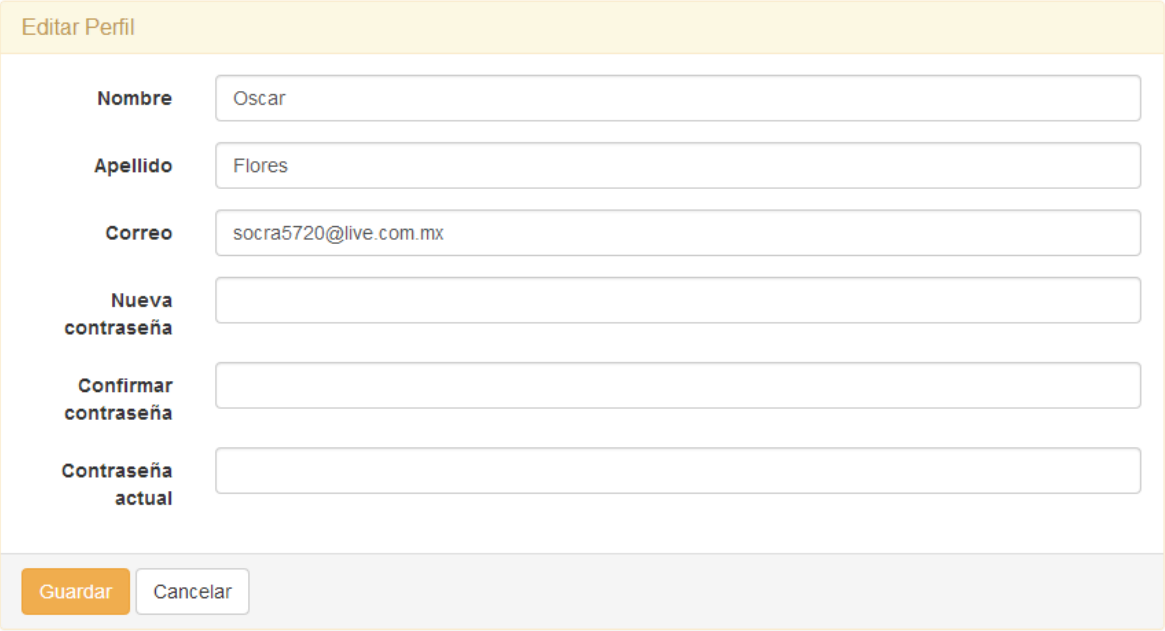
## Perfil del usuario



Perfil

| | |
|---|---|
| **Nombre:** | Fernando |
| **Apellido:** | Garcia |
| **Nombre de usuario:** | miofher |
| **Correo:** | backstrack32@hotmail.es |

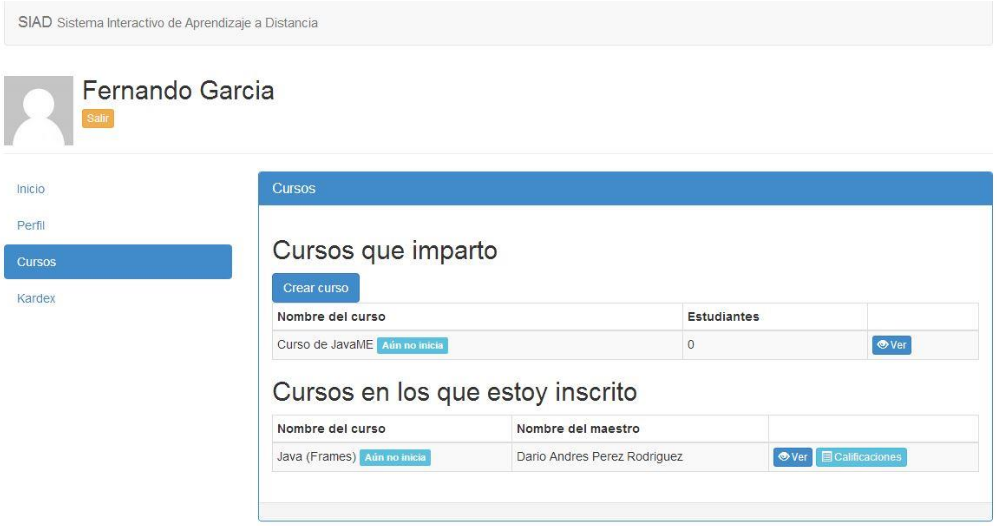
## Editar usuario



Editar Perfil

| | |
|---|---|
| Nombre | Oscar |
| Apellido | Flores |
| Correo | socra5720@live.com.mx |
| Nueva contraseña | |
| Confirmar contraseña | |
| Contraseña actual | |

## Cursos



SIAD Sistema Interactivo de Aprendizaje a Distancia

Fernando Garcia

Cursos

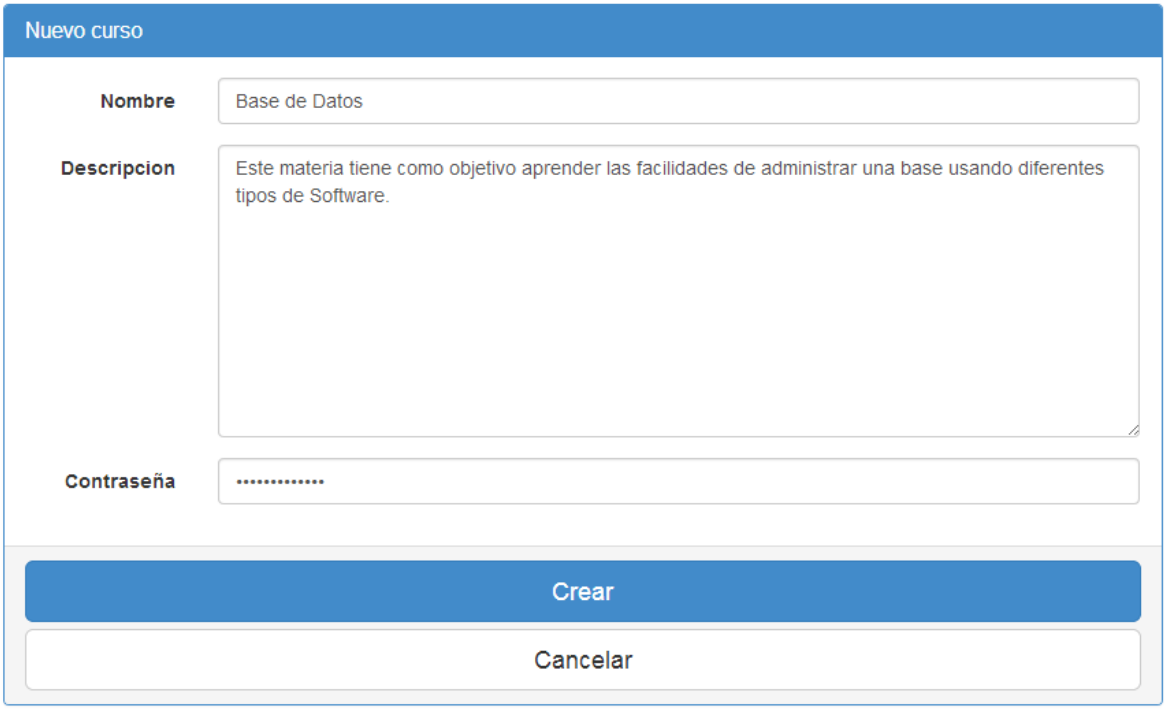
Cursos que imparto

| Nombre del curso | Estudiantes | |
|---|---|---|
| Curso de JavaME Aún no inicia | 0 | Ver |

Cursos en los que estoy inscrito

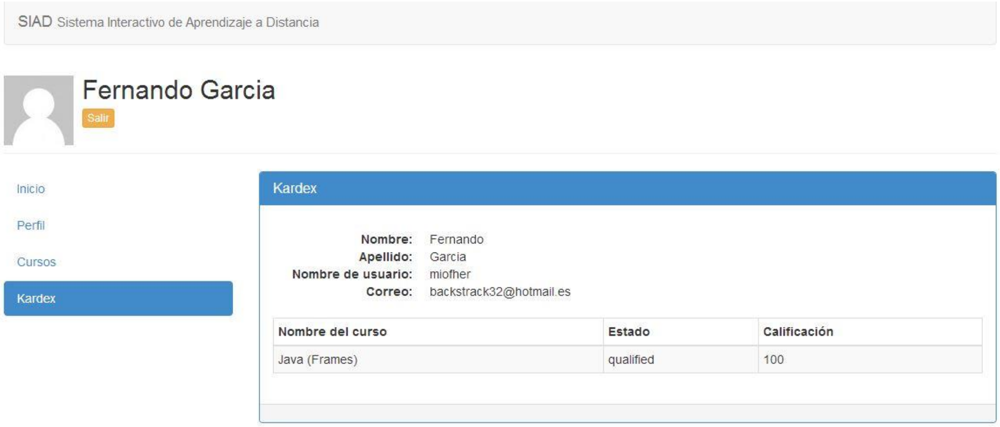
| Nombre del curso | Nombre del maestro | |
|---|---|---|
| Java (Frames) Aún no inicia | Dario Andres Perez Rodriguez | Ver Calificaciones |

## Curso que imparto


Nuevo curso

| | |
|---|---|
| Nombre | Base de Datos |
| Descripcion | Este materia tiene como objetivo aprender las facilidades de administrar una base usando diferentes tipos de Software. |
| Contraseña | •••••••••••••• |

## Kardex


SIAD Sistema Interactivo de Aprendizaje a Distancia

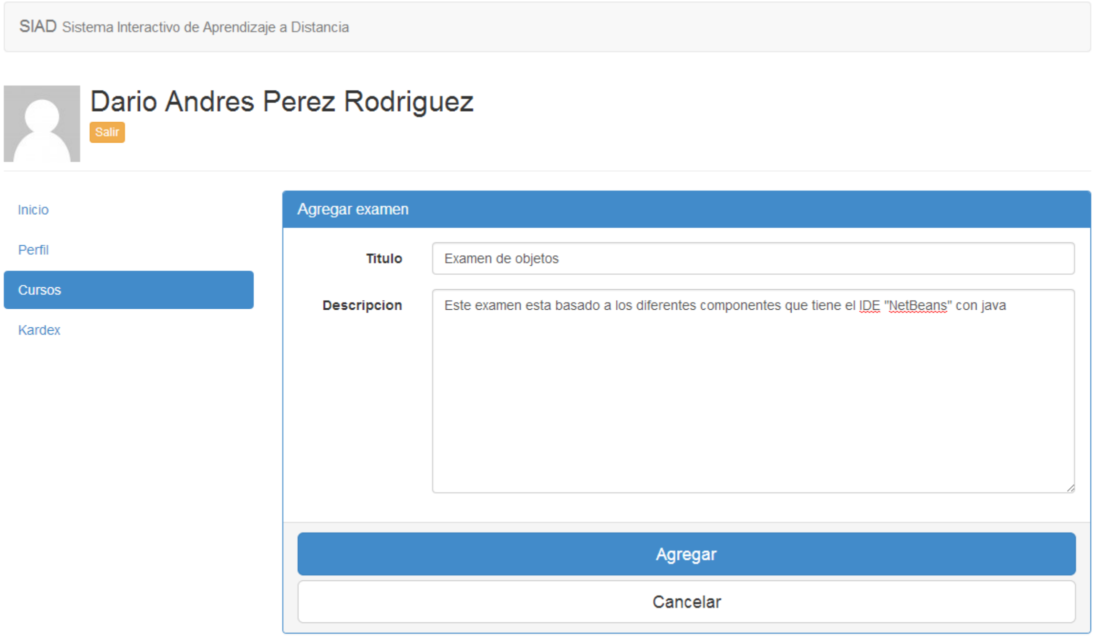
Fernando Garcia

Kardex

| | |
|---|---|
| Nombre: | Fernando |
| Apellido: | Garcia |
| Nombre de usuario: | miofher |
| Correo: | backstrack32@hotmail.es |

| Nombre del curso | Estado | Calificación |
|---|---|---|
| Java (Frames) | qualified | 100 |

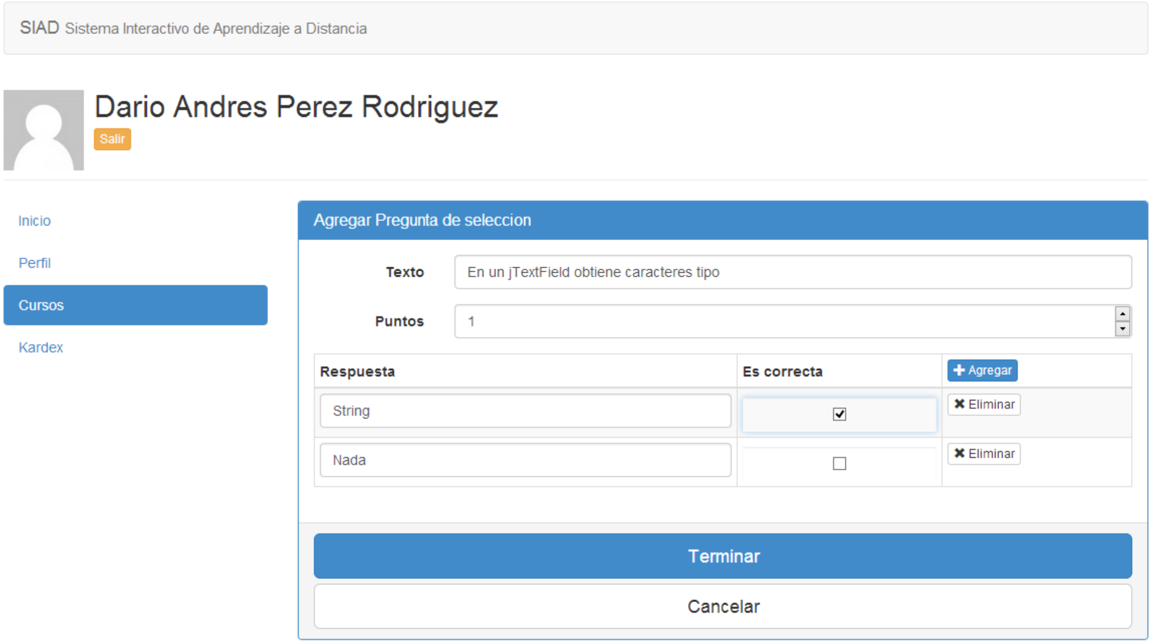
## Examen


SIAD Sistema Interactivo de Aprendizaje a Distancia
Dario Andres Perez Rodriguez
Salir
Inicio
Perfil
Cursos
Kardex
Agregar examen
Agregar
Cancelar

Titulo: Examen de objetos

Descripcion: Este examen esta basado a los diferentes componentes que tiene el IDE "NetBeans" con java

## Agregar pregunta a un examen



SIAD Sistema Interactivo de Aprendizaje a Distancia

Dario Andres Perez Rodriguez

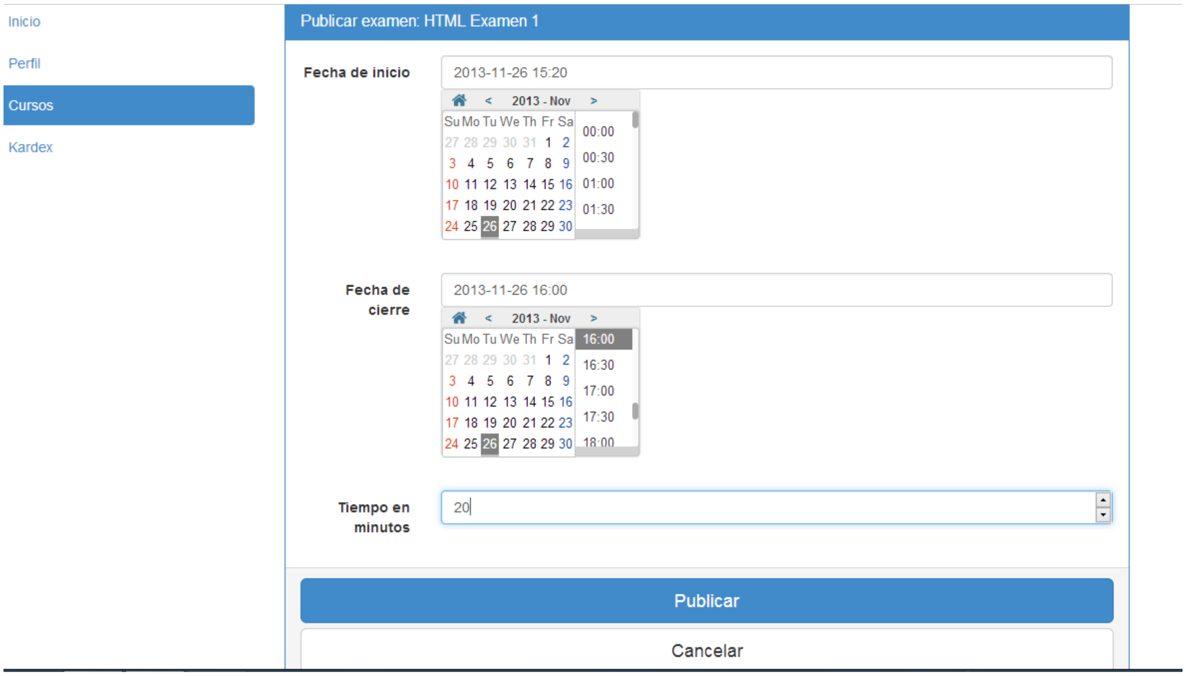
Agregar Pregunta de seleccion

Texto: En un jTextField obtiene caracteres tipo

Puntos: 1

| Respuesta | Es correcta | + Agregar |
|---|---|---|
| String | ☑ | ✖ Eliminar |
| Nada | □ | ✖ Eliminar |

## Publicar un examen



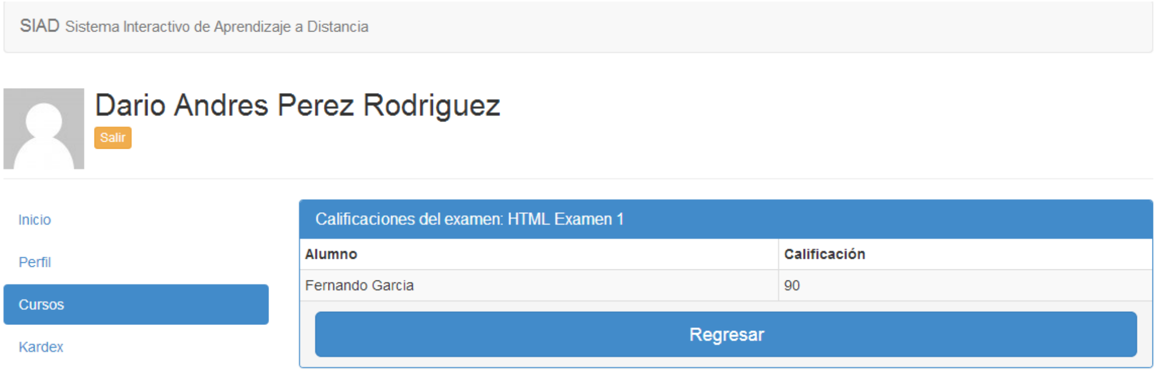
Publicar examen: HTML Examen 1

Fecha de inicio: 2013-11-26 15:20

Fecha de cierre: 2013-11-26 16:00

Tiempo en minutos: 20

## Calificaciones



SIAD Sistema Interactivo de Aprendizaje a Distancia
Dario Andres Perez Rodriguez
Salir
Inicio
Perfil
Cursos
Kardex
Regresar

Calificaciones del examen: HTML Examen 1

| Alumno | Calificación |
|---|---|
| Fernando Garcia | 90 |

## Arquitectura del sistema

Nuestro sistema está basado en un “modelo vista controlador” el cual separa los datos y la lógica de negocio de una aplicación de la interfaz de usuario es decir, por un lado define componentes para la representación de la información, y por otro lado para la interacción del usuario, por lo tanto es perfecto para manejar nuestro sistema además de que gracias a este modelo no estaremos limitados a una sola computadora o algún solo servidor sino que dispondremos de muchas más herramientas para el correcto funcionamiento del sistema.

## Especificación de ayuda en el sistema

Nuestro sistema tiene varios objetos y/o características visuales a la hora de estar dentro de la aplicación alguna de estos ejemplos son al momento de ver los colores de los botones

Rojo.- Significa un estado de “peligro” es decir que si se confirma la acción no se puede deshacer.

Azul.- Conlleva a una acción positiva es decir “cambios” de una ventana a otra

Blanco.- es el color neutro es decir dan una acción pero no es ni positiva ni negativa

Además no solamente contamos con aquellos colores sino que también contamos con pequeños iconos justo antes de la palabra para poder conocer visualmente lo que ese botón puede hacer un ejemplo de esto es “agregar” es un botón que a su lado derecho tiene un icono el cual es una “pluma” esto puede indicar al usuario que lo que va a hacer el botón es crear algo nuevo.

## Conexión de la aplicación cliente con el servidor

Nosotros hemos utilizado Laravel es un framework de código abierto para desarrollar aplicaciones y servicios web con PHP 5.

Además estamos utilizando un sistema de mapeo de datos relacional llamado Eloquent ORM que facilita la creación de modelos. Este ORM se funda en patrón active record y su funcionamiento es muy sencillo. Gracias a esto podemos tener una conexión de la aplicación con el servidor

## Acceso a la base de datos desde la aplicación

Nosotros hemos utilizado MySQL como acceso a la base de datos ya que es más sencillo y liviano al momento de poder utilizar el sistema además que gracias a este manejador de base de datos nos da más funciones para poder controlar y además para poder validar la base de datos y no poder tener ningún problema con ella, además es necesaria este para que el sistema funcione ya que MySQL tiene más opciones para poder personalizar fácilmente la base de datos y como se ha dicho antes es más fácil validarlo.

# Código de la aplicación

## Filtros

```php
<?php

App::before(function($request) {
});


App::after(function($request, $response) {
});

Route::filter('auth', function() {
    if (Auth::guest()) return Redirect::guest('login');
});


Route::filter('auth.basic', function() {
    return Auth::basic();
});

Route::filter('guest', function() {
    if (Auth::check()) return Redirect::to('/');
});

Route::filter('csrf', function() {
    if (Session::token() != Input::get('_token')) {
        throw new Illuminate\Session\TokenMismatchException;
    }
});

Route::filter('welcome', function() {
    if (!Auth::check()) {
        return Redirect::route('welcome');
    }
});

Route::filter('course.teacher', function($route) {
    $user = Auth::user();

    $course = $route->getParameter('course');

    if (!$user->teacherCourses->contains($course->course_id)) {
        return Redirect::route('course.lists');
    }
});
Route::filter('course.student', function($route) {
    $user = Auth::user();

    $course = $route->getParameter('course');

    if (!$user->studentCourses->contains($course->course_id)) {
        return Redirect::route('course.lists');
    }
});
Route::filter('course.candidat', function($route) {
    $user = Auth::user();

    $course = $route->getParameter('course');

    if ($user->teacherCourses->contains($course->course_id)) {
        return Redirect::route('course.manager', $course->course_id);
    }

    if ($user->studentCourses->contains($course->course_id)) {
        return Redirect::route('course.view', $course->course_id);
    }
});
Route::filter('course.available', function($route) {
    $course = $route->getParameter('course');

    if ($course->started_at) {
        return Redirect::route('course.lists');
    }
});
```

```php
Route::filter('course.closed', function($route) {
    $course = $route->getParameter('course');

    if (NULL != $course->closed_at) {
        Session::push('messages', array(
            'type' => 'info',
            'text' => "El curso '$course->course_name' todavia no termina"
        ));

        return Redirect::route('course.lists');
    }
});
Route::filter('course.started', function($route) {
    $course = $route->getParameter('course');

    if (!$course->started_at) {
        Session::push('messages', array(
            'type' => 'info',
            'text' => "El curso '$course->course_name' todavia no inicia"
        ));

        return Redirect::route('course.lists');
    }
});

Route::filter('resource.both', function($route) {
    $user = Auth::user();

    $course = $route->getParameter('resource')->course;

    if (!$user->teacherCourses->contains($course->course_id)) {
        if (!$user->studentCourses->contains($course->course_id)) {
            return Redirect::route('course.lists');
        }

        if (!$course->started_at) {
            return Redirect::route('course.lists');
        }
    }
});
Route::filter('resource.teacher', function($route) {
    $user = Auth::user();

    $course = $route->getParameter('resource')->course;

    if (!$user->teacherCourses->contains($course->course_id)) {
        return Redirect::route('course.lists');
    }
});
Route::filter('resource.student', function($route) {
    $user = Auth::user();

    $course = $route->getParameter('resource')->course;

    if (!$user->studentCourses->contains($course->course_id)) {
        return Redirect::route('course.lists');
    }

    if (!$course->started_at) {
        return Redirect::route('course.lists');
    }
});

Route::filter('test.teacher', function($route) {
    $user = Auth::user();

    $course = $route->getParameter('test')->course;

    if (!$user->teacherCourses->contains($course->course_id)) {
        return Redirect::route('course.lists');
    }
});
Route::filter('test.student', function($route) {
    $user = Auth::user();

    $course = $route->getParameter('test')->course;

    if (!$user->studentCourses->contains($course->course_id)) {
        return Redirect::route('course.lists');
    }

    if (!$course->started_at) {
        return Redirect::route('course.lists');
    }
});
Route::filter('test.valid', function($route) {
    $test = $route->getParameter('test');

    if (!$test->canStart(Auth::user())) {
        return Redirect::route('test.info', $test->test_id);
    }
});
Route::filter('test.editable', function($route) {
    $test = $route->getParameter('test');

    if (!$test->isEditable()) {
        return Redirect::route('course.manager', $test->course->course_id);
    }
});
Route::filter('test.finalized', function($route) {
    $test = $route->getParameter('test');

    if (!$test->isFinalized()) {
        return Redirect::route('course.manager', $test->course->course_id);
    }
});
Route::filter('test.filled', function($route) {
    $test = $route->getParameter('test');

    if (!$test->hasQuestions()) {
        return Redirect::route('course.manager', $test->course->course_id);
    }
});
```

## Rutas

```php
<?php

Route::model('user', 'User');
Route::model('course', 'Course');
Route::model('resource', 'Resource');
Route::model('test', 'Test');
Route::model('question', 'Question');

Route::group(array('prefix' => 'auth'), function() {
    Route::get('', array(
        'as' => 'login',
        'before' => 'guest',
        'uses' => 'AuthController@login'
    ));
    Route::get('welcome', array(
        'as' => 'welcome',
        'before' => 'guest',
        'uses' => 'AuthController@welcome'
    ));
    Route::get('register', array(
        'as' => 'register',
        'before' => 'guest',
        'uses' => 'AuthController@register'
    ));
    Route::post('create', array(
        'as' => 'user.create',
        'before' => 'guest',
        'uses' => 'AuthController@create'
    ));
    Route::post('login', array(
        'as' => 'do.login',
        'before' => 'guest',
        'uses' => 'AuthController@doLogin'
    ));
    Route::get('logout', array(
        'as' => 'do.logout',
        'before' => 'auth',
        'uses' => 'AuthController@doLogout'
    ));
});

```

```
Route::group(array('before' => 'welcome'), function() {

    View::share('user', Auth::user());

    View::composer('layouts.default', function($view) {
        $links = array(
            array('name' => 'Inicio', 'route' => 'dashboard'),
            array('name' => 'Perfil', 'route' => array(
                'profile.view',
                'profile.edit',
            )),
            array('name' => 'Cursos', 'route' => array(
                'course.lists',
                'course.register',
                'course.add',
                'course.edit',
                'course.manager',
                'course.view',
                'course.qualifications',
                'resource.add',
                'resource.edit',
                'resource.view',
                'test.add',
                'test.edit',
                'test.info',
                'test.view',
                'test.manager',
                'test.publish',
                'test.qualifications',
                'question.add',
                'question.edit',
            )),
            array('name' => 'Kardex', 'route' => 'kardex'),
        );

        $view->with('links', $links);
    });

    Route::get('', array(
        'as' => 'dashboard',
        'uses' => 'DashboardController@dashboard'
    ));

    Route::get('kardex', array(
        'as' => 'kardex',
        'uses' => 'DashboardController@kardex'
    ));

    Route::group(array('prefix' => 'profile'), function() {
        Route::get('', array(
            'as' => 'profile.view',
            'uses' => 'ProfileController@view'
        ));
        Route::get('edit', array(
            'as' => 'profile.edit',
            'uses' => 'ProfileController@edit'
        ));
        Route::post('update', array(
            'as' => 'profile.update',
            'uses' => 'ProfileController@update'
        ));
    });

    Route::group(array('prefix' => 'courses'), function() {
        Route::get('', array(
            'as' => 'course.lists',
            'uses' => 'CourseController@lists'
        ));
        Route::get('add', array(
            'as' => 'course.add',
            'uses' => 'CourseController@add'
        ));
        Route::post('create', array(
            'as' => 'course.create',
            'uses' => 'CourseController@create'
        ));
        Route::get('edit/{course}', array(
            'as' => 'course.edit',
            'before' => 'course.teacher|course.available',
            'uses' => 'CourseController@edit'
        ));
        Route::post('update/{course}', array(
            'as' => 'course.update',
            'before' =>
'course.teacher|course.available',
            'uses' => 'CourseController@update'
        ));
        Route::get('register/{course}', array(
            'as' => 'course.register',
            'before' =>
'course.candidat|course.available',
            'uses' => 'CourseController@register'
        ));
        Route::post('subscribe/{course}', array(
            'as' => 'course.subscribe',
            'before' =>
'course.candidat|course.available',
            'uses' => 'CourseController@subscribe'
        ));
        Route::get('manager/{course}', array(
            'as' => 'course.manager',
            'before' => 'course.teacher',
            'uses' => 'CourseController@manager'
        ));
        Route::get('cancel/{course}', array(
            'as' => 'course.cancel',
            'before' => 'course.teacher',
            'uses' => 'CourseController@cancel'
        ));
        Route::get('start/{course}', array(
            'as' => 'course.start',
            'before' =>
'course.teacher|course.available',
            'uses' => 'CourseController@start'
        ));
        Route::get('close/{course}', array(
            'as' => 'course.close',
            'before' => 'course.teacher|course.started',
            'uses' => 'CourseController@close'
        ));
        Route::get('view/{course}', array(
            'as' => 'course.view',
            'before' => 'course.student|course.started',
            'uses' => 'CourseController@view'
        ));
        Route::get('unsubscribe/{course}', array(
            'as' => 'course.unsubscribe',
            'before' => 'course.student|course.started',
            'uses' => 'CourseController@unsubscribe'
        ));
        Route::get('qualifications/{course}', array(
            'as' => 'course.qualifications',
            'before' => 'course.student|course.started',
            'uses' => 'CourseController@qualifications'
        ));
    });

    Route::group(array('prefix' => 'resources'), function() {
        Route::get('{course}/add/{type}', array(
            'as' => 'resource.add',
            'before' => 'course.teacher',
            'uses' => 'ResourceController@add'
        ))->where('type', 'video|file|url');
        Route::post('{course}/create', array(
            'as' => 'resource.create',
            'before' => 'course.teacher',
            'uses' => 'ResourceController@create'
        ));
        Route::get('edit/{resource}', array(
            'as' => 'resource.edit',
            'before' => 'resource.teacher',
            'uses' => 'ResourceController@edit'
        ));
        Route::post('update/{resource}', array(
            'as' => 'resource.update',
            'before' => 'resource.teacher',
            'uses' => 'ResourceController@update'
        ));
        Route::get('delete/{resource}', array(
            'as' => 'resource.delete',
            'before' => 'resource.teacher',
            'uses' => 'ResourceController@delete'
        ));
        Route::get('view/{resource}', array(
            'as' => 'resource.view',
            'before' => 'resource.both',
            'uses' => 'ResourceController@view'
        ));
        Route::get('download/{resource}', array(
            'as' => 'resource.download',
            'before' => 'resource.both',
            'uses' => 'ResourceController@download'
        ));
    });

    Route::group(array('prefix' => 'tests'), function() {
        Route::get('{course}/add', array(
            'as' => 'test.add',
            'before' => 'course.teacher',
            'uses' => 'TestController@add',
        ));
        Route::post('{course}/create', array(
            'as' => 'test.create',
            'before' => 'course.teacher',
            'uses' => 'TestController@create',
        ));
        Route::get('edit/{test}', array(
            'as' => 'test.edit',
            'before' => 'test.teacher|test.editable',
            'uses' => 'TestController@edit',
        ));
        Route::post('update/{test}', array(
            'as' => 'test.update',
            'before' => 'test.teacher|test.editable',
            'uses' => 'TestController@update',
        ));
        Route::get('delete/{test}', array(
            'as' => 'test.delete',
            'before' => 'test.teacher|test.editable',
            'uses' => 'TestController@delete',
        ));
        Route::get('manager/{test}', array(
            'as' => 'test.manager',
            'before' => 'test.teacher|test.editable',
            'uses' => 'TestController@manager',
        ));
        Route::get('{course}/publish/{test}', array(
            'as' => 'test.publish',
            'before' =>
'test.teacher|test.editable|test.filled|course.started',
            'uses' => 'TestController@publish',
        ));
        Route::post('{course}/publish/{test}', array(
            'as' => 'test.do.publish',
            'before' =>
'test.teacher|test.editable|test.filled|course.started',
            'uses' => 'TestController@doPublish',
        ));
        Route::get('info/{test}', array(
            'as' => 'test.info',
            'before' => 'test.student',
            'uses' => 'TestController@info'
        ));
        Route::get('view/{test}', array(
            'as' => 'test.view',
            'before' => 'test.student|test.valid',
            'uses' => 'TestController@view'
        ));
        Route::post('submit/{test}', array(
            'as' => 'test.submit',
            'before' => 'test.student|test.valid',
            'uses' => 'TestController@submit'
        ));
        Route::get('qualifications/{test}', array(
            'as' => 'test.qualifications',
            'before' => 'test.teacher|test.finalized',
            'uses' => 'TestController@qualifications'
        ));
    });

    Route::group(array('prefix' => 'questions'),
function() {
        Route::get('{test}/add/{type}', array(
            'as' => 'question.add',
            'before' => 'test.teacher|test.editable',
            'uses' => 'QuestionController@add'
        ))->where('type', 'open|choice|multiple');
        Route::post('{test}/create', array(
            'as' => 'question.create',
            'before' => 'test.teacher|test.editable',
            'uses' => 'QuestionController@create'
        ));
        Route::get('{test}/edit/{question}', array(
            'as' => 'question.edit',
            'before' => 'test.teacher|test.editable',
            'uses' => 'QuestionController@edit'
        ));
        Route::post('{test}/update/{question}', array(
            'as' => 'question.update',
            'before' => 'test.teacher|test.editable',
            'uses' => 'QuestionController@update'
        ));
        Route::get('{test}/delete/{question}', array(
            'as' => 'question.delete',
            'before' => 'test.teacher|test.editable',
            'uses' => 'QuestionController@delete'
        ));
    });
});
```

# Controladores

## Control de autentificación

```php
<?php

class AuthController extends BaseController {

    protected $layout = 'layouts.guest';

    public function welcome() {
        $this->layout->content = View::make('auth.welcome');
    }

    public function register() {
        $this->layout->content = View::make('auth.register');
    }

    public function login() {
        $this->layout->content = View::make('auth.login');
    }

    public function create() {
        $input = Input::only('first_name', 'last_name', 'email', 'user_name', 'password', 'password_confirmation');

        $validator = Validator::make($input,
            array(
                'first_name' => 'required|alpha|min:4|max:20',
                'last_name' => 'required|alpha|min:4|max:20',
                'email' => "required|email|unique:users",
                'user_name' => 'required|alpha_dash|min:5|max:15|unique:users',
                'password' => 'required|confirmed|min:5|max:20'
            )
        );

        if ($validator->fails()) {
            return Redirect::back()->withErrors($validator)->withInput($input);
        }

        if (User::create($input)) {
            Session::push('messages', array('type' => 'info', 'text' => 'Usuario registrado correctamente!'));

            return Redirect::route('login');
        }

        Session::push('messages', array('type' => 'danger', 'text' => 'No se puede completar el registro!'));

        return Redirect::back()->withInput($input);
    }

    public function doLogin() {
        $input = Input::only('user_name', 'password');

        $validator = Validator::make($input,
            array(
                'user_name' => 'required',
                'password' => 'required'
            )
        );

        if ($validator->fails()) {
            return Redirect::back()->withErrors($validator)->withInput($input);
        }

        $input['user_name'] = strtolower($input['user_name']);

        if (!Auth::attempt($input, Input::has('remember'))) {
            Session::push('messages', array('type' => 'danger', 'text' => 'Usuario/Contraseña invalido[s]!'));
```

```php

            return Redirect::back()->withInput($input);
        }

        return Redirect::route('dashboard');
    }

    public function doLogout() {
        if (Auth::logout()) {
            return Redirect::route('welcome');
        }

        return Redirect::back();
    }
}
```

### Controlador base

```php
<?php

class BaseController extends Controller {

    /**
     * Setup the layout used by the controller.
     *
     * @return void
     */
    protected function setupLayout()
    {
        if ( ! is_null($this->layout))
        {
            $this->layout = View::make($this->layout);
        }
    }

}
```

### Controlador de curso

```php
<?php

class CourseController extends BaseController {

    protected $layout = 'layouts.default';

    public function lists() {
        $teacherCourses = Auth::user()->teacherCourses;
        $studentCourses = Auth::user()->studentCourses;

        $this->layout->content = View::make('courses.lists')
            ->with('teacherCourses', $teacherCourses)
            ->with('studentCourses', $studentCourses);
    }

    public function add() {
        $this->layout->content = View::make('courses.add');
    }

    public function create() {
        $input = Input::only('course_name', 'description', 'password');

        $user = Auth::user();

        $validator = Validator::make($input,
            array(
                'course_name' => 'required|min:10|max:120',
                'description' => 'max:500',
                'password' => 'max:20'
            )
        );

        if ($validator->fails()) {
            return Redirect::back()->withErrors($validator)->withInput($input);
        }

        $input['created_at'] = Carbon::now();

        $course = new Course($input);

        $user->teacherCourses()->save($course);

        return Redirect::route('course.manager', $course->course_id);
    }

    public function edit(Course $course) {
        $this->layout->content = View::make('courses.edit')
            ->with('course', $course);
    }

    public function update(Course $course) {
        $input = Input::only('course_name', 'description', 'password');

        $validator = Validator::make($input,
            array(
                'course_name' => 'required|min:10|max:120',
                'description' => 'max:500',
                'password' => 'max:20'
            )
        );

        if ($validator->fails()) {
            return Redirect::back()->withErrors($validator)->withInput($input);
        }

        $course->update($input);

        return Redirect::route('course.manager', $course->course_id);
    }

    public function start(Course $course) {
        $course->started_at = Carbon::now();
        $course->closed_at = NULL;
        $course->save();

        return Redirect::route('course.manager', $course->course_id);
    }

    public function close(Course $course) {
        $course->students->each(function($student) use ($course) {
            $totalTests = 0;
            $qualification = 0;

            $course->tests->each(function($test) use ($student, &$totalTests, &$qualification) {
                if ($test->hasQualification($student) || $test->isFinalized()) {
                    $totalTests++;

                    if (!$test->students->contains($student->user_id)) {
                        $test->students()->attach($student->user_id);
                    } else {
                        $qualification += $test->students->find($student->user_id)->pivot->qualification;
                    }
                }
            });

            $student->questions->each(function($question) use ($student, $course) {
                if ($course->course_id == $question->test->course->course_id) {
                    $student->questions()->detach($question->question_id);
                }
            });

            $student->tests->each(function($test) use ($student, $course) {
                if ($course->course_id == $test->course->course_id) {
                    $student->tests()->detach($test->test_id);
                }
            });

            if ($totalTests) {
                $qualification /= $totalTests;
            }

            Kardex::create(array(
                'user_id' => $student->user_id,
                'course_name' => $course->course_name,
                'status' => $totalTests ? 'qualified' : 'completed',
                'qualification' => $qualification
            ));
        });

        $course->students()->detach();

        $course->tests->each(function($test) {
            $test->time = 0;
            $test->start_date = NULL;
            $test->end_date = NULL;

            $test->save();
        });

        $course->started_at = NULL;
        $course->closed_at = Carbon::now();
        $course->save();

        return Redirect::route('course.manager', $course->course_id);
    }

    public function manager(Course $course) {
        $this->layout->content = View::make('courses.manager')
            ->with('course', $course);
    }

    public function view(Course $course) {
        $this->layout->content = View::make('courses.view')
            ->with('course', $course);
    }

    public function register(Course $course) {
        $this->layout->content = View::make('courses.register')
            ->with('course', $course);
    }

    public function subscribe(Course $course) {
        if ($course->password) {
            $input = Input::only('password');

            $validator = Validator::make($input,
                array(
                    'password' => 'required',
                )
            );

            if ($validator->fails()) {
                return Redirect::back()->withErrors($validator)->withInput($input);
            }

            if ($course->password != $input['password']) {
                Session::push('messages', array('type' => 'danger', 'text' => 'Contraseña invalida!'));

                return Redirect::back()->withInput($input);
            }
        }

        $course->students()->attach(Auth::user()->user_id);

        return Redirect::route('course.view', $course->course_id);
    }

    public function unsubscribe($course) {
        $user = Auth::user();

        if ($course->students()->detach($user->user_id)) {
            Session::push('messages', array('type' => 'info', 'text' => "Diste de baja el curso '$course->course_name'"));

            $user->questions->each(function($question) use ($user, $course) {
                if ($course->course_id == $question->test->course->course_id) {
                    $user->questions()->detach($question->question_id);
                }
            });

            $user->tests->each(function($test) use ($user, $course) {
                if ($course->course_id == $test->course->course_id) {
                    $user->tests()->detach($test->test_id);
                }
            });

            Kardex::create(array(
                'user_id' => $user->user_id,
                'course_name' => $course->course_name,
                'status' => 'unsubscribe',
                'qualification' => 0
            ));
        } else {
            Session::push('messages', array('type' => 'warning', 'text' => "El curso '$course->course_name' no se pudo dar de baja"));
        }
```

```php

        return Redirect::route('course.lists');
    }

    public function cancel(Course $course) {
        $resources = $course->resources;

        if ($course->delete()) {
            Session::push('messages', array('type' => 'info', 'text' => "Cancelaste el curso '$course->course_name'"));

            $resources->each(function($resource) {
                if (File::exists('uploads/resources/' . $resource->resource_id)) {
                    File::delete('uploads/resources/' . $resource->resource_id);
                }
            });
        } else {
            Session::push('messages', array('type' => 'warning', 'text' => "El curso '$course->course_name' no se pudo cancelar"));
        }

        return Redirect::route('course.lists');
    }

    public function qualifications(Course $course) {
        $user = Auth::user();

        $course->tests->each(function($test) use ($user) {
            if (!$test->students->contains($user->user_id)) {
                if ($test->isExpired($user)) {
                    $test->students()->attach($user->user_id);
                }
            }
        });

        $this->layout->content = View::make('courses.qualifications')
            ->with('course', $course);
    }
}
```

### Pagina principal

```
<?php

class DashboardController extends BaseController {

    protected $layout = 'layouts.default';

    public function dashboard() {
        $user = Auth::user();

        $courses = Course::where('teacher_id', '!=', $user->user_id)
            ->whereNull('started_at')
            ->orderBy('created_at', 'DESC')
            ->take(10)
            ->get();

        $this->layout->content = View::make('dashboard')
            ->with('courses', $courses);
    }

    public function kardex() {
        $this->layout->content = View::make('kardex');
    }
}
```

### Controlador de perfiles

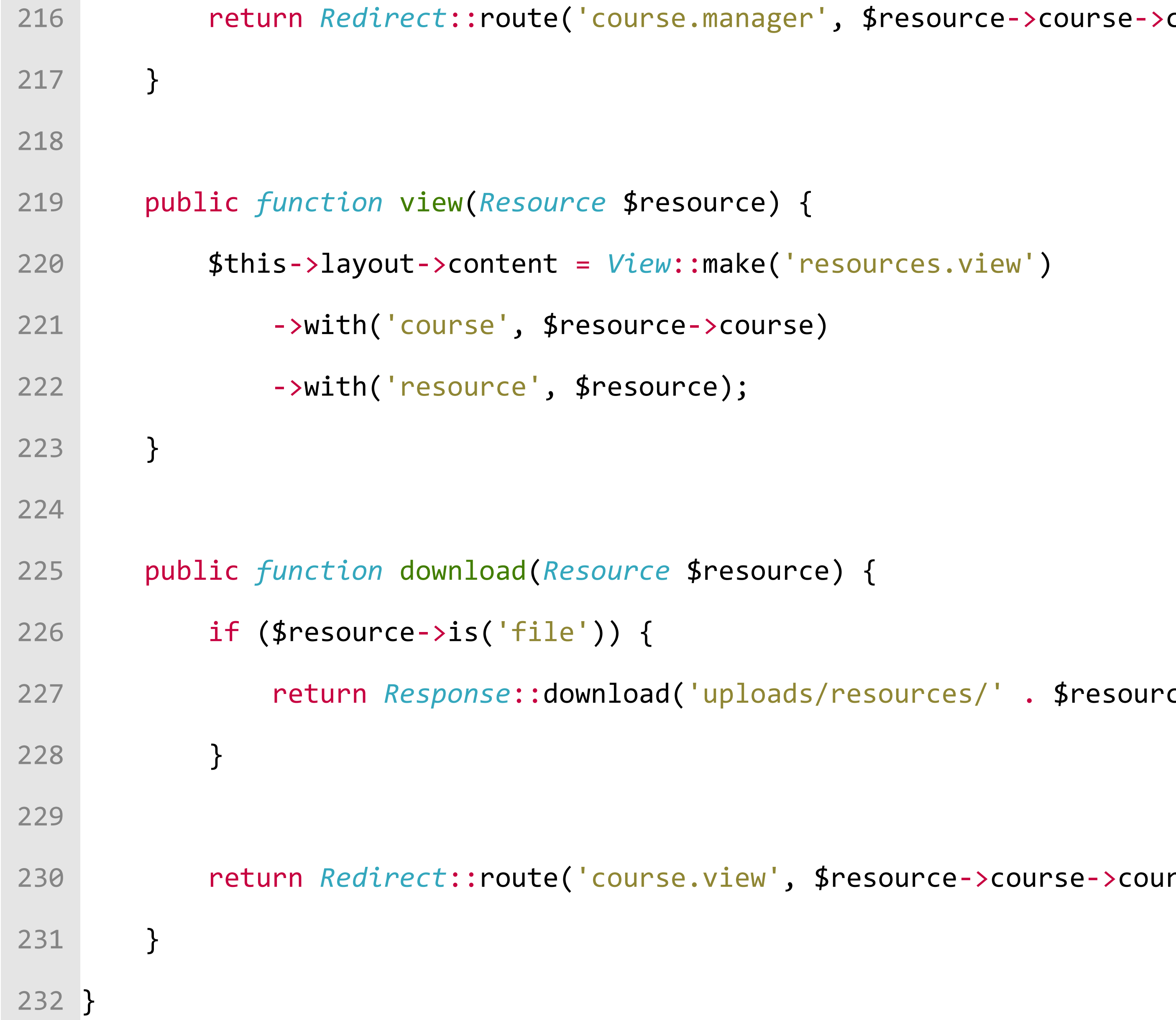
```php
<?php

class ProfileController extends BaseController {

    protected $layout = 'layouts.default';

    public function view() {
        $this->layout->content = View::make('profiles.view');
    }

    public function edit() {
        $this->layout->content = View::make('profiles.edit');
    }

    public function update() {
        $input = Input::only('first_name', 'last_name', 'email', 'password', 'new_password', 'new_password_confirmation');

        $user = Auth::user();

        $validator = Validator::make($input,
            array(
                'first_name' => 'alpha|min:4|max:20',
                'last_name' => 'alpha|min:4|max:20',
                'email' => "email|unique:users,email,$user->user_id,user_id",
                'password' => 'required',
                'new_password' => 'confirmed|min:5|max:20'
            )
        );

        if ($validator->fails()) {
            return Redirect::back()->withErrors($validator)->withInput($input);
        }

        if (!Hash::check($input['password'], $user-
>password)) {
            Session::push('messages', array('type' => 'danger', 'text' => 'Contraseña invalida!'));

            return Redirect::back()->withInput($input);
        }

        if ($input['new_password']) {
            $user->password = Hash::make($input['new_password']);
        }

        $user->update($input);

        return Redirect::route('profile.view');
    }
}
```

### Controlador de preguntas

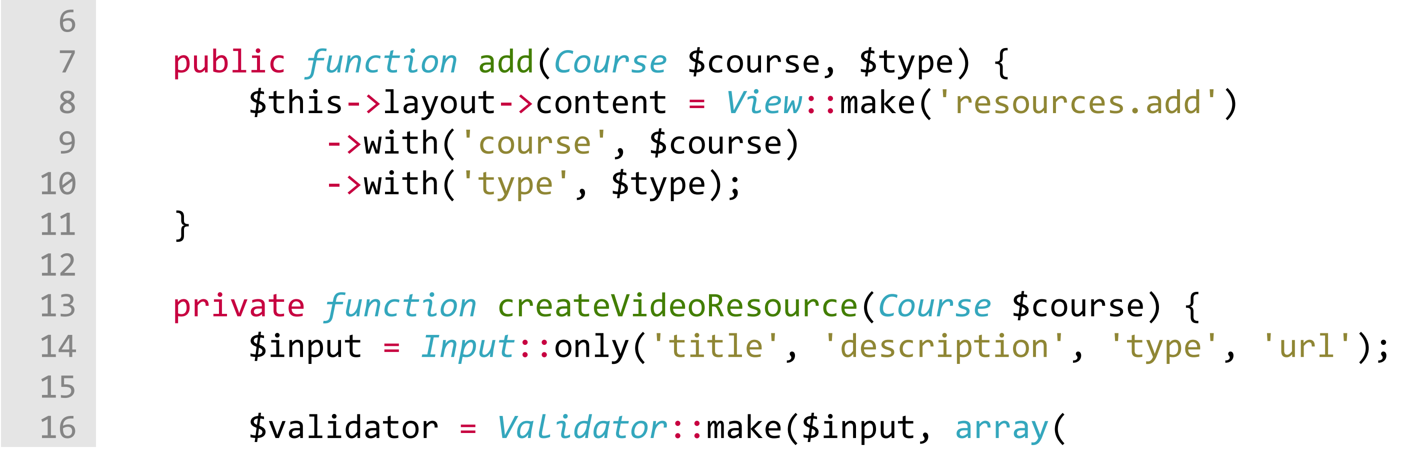
```
<?php

class QuestionController extends BaseController {

    protected $layout = 'layouts.default';

    public function add(Test $test, $type) {
        $this->layout->content = View::make('questions.add')
            ->with('course', $test->course)
            ->with('test', $test)
            ->with('type', $type);
    }

    public function create(Test $test) {
        $input = Input::only('question_text', 'type', 'points', 'answer_text', 'is_correct');

        $validator = Validator::make($input, array(
            'question_text' => 'required|min:3|max:200',
            'points' => 'required|numeric|min:1'
        ));
        if ($validator->fails()) {
            return Redirect::back()->withErrors($validator)->withInput($input);
        }

        $min = 'open' == $input['type'] ? 1 : 2;

        $answerText = Input::get('answer_text', array());
        $isCorrect = Input::get('is_correct', array());

        if (!is_array($answerText) || !is_array($isCorrect)) {
            Session::push('messages', array(
                'type' => 'danger',
                'text' => 'No se puede agregar la pregunta'
            ));

            return Redirect::back()->withInput($input);
        }

        if (count($answerText) !== count($isCorrect)) {
            Session::push('messages', array(
                'type' => 'danger',
                'text' => 'No se puede agregar la pregunta'
            ));

            return Redirect::back()->withInput($input);
        }

        if ($min > count($answerText)) {
            Session::push('messages', array(
                'type' => 'danger',
                'text' => "Tiene que existir por lo menos $min respuesta(s)"
            ));

            return Redirect::back()->withInput($input);
        }

        $correctCount = 0;

        for ($i = 0; $i < count($answerText); $i++) {
            if (1 == $isCorrect[$i] && '' != $answerText[$i]) {
                $correctCount++;
            }
        }

        if (0 === $correctCount) {
            Session::push('messages', array(
                'type' => 'danger',
                'text' => 'Tiene que existir por lo menos una respuesta correcta'
            ));

            return Redirect::back()->withInput($input);
        }

        $question = new Question($input);

        $test->questions()->save($question);

        for ($i = 0; $i < count($answerText); $i++) {
            if ('' != $answerText[$i]) {
                $answer = new Answer(array(
                    'answer_text' => $answerText[$i],
                    'is_correct' => 1 == $isCorrect[$i]
                ));

                $question->answers()->save($answer);
            }
        }

        return Redirect::route('test.manager', $test->test_id);
    }

    public function edit(Test $test, Question $question) {
        $this->layout->content =
View::make('questions.edit')
            ->with('course', $test->course)
            ->with('test', $test)
            ->with('question', $question);
    }

    public function update(Test $test, Question $question) {
        $input = Input::only('question_text', 'points', 'answer_text', 'is_correct');

        $validator = Validator::make($input, array(
            'question_text' => 'required|min:3|max:200',
            'points' => 'required|numeric|min:1'
        ));

        if ($validator->fails()) {
            return Redirect::back()->withErrors($validator)->withInput($input);
        }

        $answerText = Input::get('answer_text', array());
        $isCorrect = Input::get('is_correct', array());

        if (!is_array($answerText) || !is_array($isCorrect)) {
            Session::push('messages', array(
                'type' => 'danger',
                'text' => 'No se puede agregar la pregunta'
            ));

            return Redirect::back()->withInput($input);
        }

        if (count($answerText) !== count($isCorrect)) {
            Session::push('messages', array(
                'type' => 'danger',
                'text' => 'No se puede agregar la pregunta'
            ));

            return Redirect::back()->withInput($input);
        }

        if (0 === count($answerText)) {
            Session::push('messages', array(
                'type' => 'danger',
                'text' => 'Tiene que existir por lo menos una respuesta'
            ));

            return Redirect::back()->withInput($input);
        }

        $correctCount = 0;

        for ($i = 0; $i < count($answerText); $i++) {
            if (1 == $isCorrect[$i] && '' != $answerText[$i]) {
                $correctCount++;
                break;
            }
        }

        if (0 === $correctCount) {
            Session::push('messages', array(
                'type' => 'danger',
                'text' => 'Tiene que existir por lo menos una respuesta correcta'
            ));

            return Redirect::back()->withInput($input);
        }

        $question->update($input);
        $question->answers()->delete();

        for ($i = 0; $i < count($answerText); $i++) {
            $answer = new Answer(array(
                'answer_text' => $answerText[$i],
                'is_correct' => 1 == $isCorrect[$i]
            ));

            $question->answers()->save($answer);
        }

        return Redirect::route('test.manager', $test->test_id);
    }

    public function delete(Test $test, Question $question) {
        $question->delete();

        return Redirect::route('test.manager', $test->test_id);
    }
}
```

### Controlador de recursos (tareas, exámenes, etc.)

```
<?php

class ResourceController extends BaseController {

    protected $layout = 'layouts.default';
<?php

class ResourceController extends BaseController {

    protected $layout = 'layouts.default';

    public function add(Course $course, $type) {
        $this->layout->content = View::make('resources.add')
            ->with('course', $course)
            ->with('type', $type);
    }

    private function createVideoResource(Course $course) {
        $input = Input::only('title', 'description', 'type', 'url');

        $validator = Validator::make($input, array(
            'title' => 'required|min:5|max:120',
            'description' => 'max:500',
            'url' => array('required', 'url',
"regex:/^(?:http(?:s)?:\/\/)?(?:www\.)?(?:youtu\.be\/|youtube\.com\/(
        ));

        if ($validator->fails()) {
            return Redirect::back()->withErrors($validator)->withInpu
        }

        $resource = new Resource($input);

        $course->resources()->save($resource);

        $videoResource = new VideoResource($input);

        $resource->toVideo()->save($videoResource);

        return Redirect::route('course.manager', $course->course_id);
    }

    private function createFileResource(Course $course) {
        $input = Input::only('title', 'description', 'type');
        $input = array_add($input, 'file', Input::file('upload'));

        $validator = Validator::make($input, array(
            'title' => 'required|min:5|max:120',
            'description' => 'max:500',
            'file' => 'required|max:10240'
        ));

        if ($validator->fails()) {
            $input = array_forget($input, 'file');

            return Redirect::back()->withErrors($validator)->withInpu
        }

        $resource = new Resource($input);

        $course->resources()->save($resource);

        $file = Input::file('upload');

        $fileResource = new FileResource(array(
            'file_name' => $file->getClientOriginalName(),
            'is_pdf' => 'application/pdf' === $file->getMimeType()
        ));

        if ($file->move('uploads/resources', $resource->resource_id))
            $resource->toFile()->save($fileResource);
        } else {
            $resource->delete();
        }

        return Redirect::route('course.manager', $course->course_id);
    }

    private function createUrlResource(Course $course) {
        $input = Input::only('title', 'description', 'type', 'url');

        $validator = Validator::make($input, array(
            'title' => 'required|min:5|max:120',
            'description' => 'max:500',
            'url' => 'required|url'
        ));

        if ($validator->fails()) {
            return Redirect::back()->withErrors($validator)->withInpu
        }

        $resource = new Resource($input);

        $course->resources()->save($resource);

        $urlResource = new UrlResource($input);

        $resource->toUrl()->save($urlResource);

        return Redirect::route('course.manager', $course->course_id);
    }

    public function create(Course $course) {
        if (Input::has('type')) {
            switch (Input::get('type')) {
                case 'video':
                    return $this->createVideoResource($course);
                case 'file':
                    return $this->createFileResource($course);
                case 'url':
                    return $this->createUrlResource($course);
            }
        }

        return Redirect::route('course.manager', $course->course_id);
    }

    public function edit(Resource $resource) {
        $this->layout->content = View::make('resources.edit')
            ->with('course', $resource->course)
            ->with('resource', $resource);
    }

    private function updateVideoResource(Resource $resource) {
        $input = Input::only('title', 'description', 'attached');

        $validator = Validator::make($input, array(
            'title' => 'required|min:5|max:120',
            'description' => 'max:500',
            'attached' => array('required', 'url',
"regex:/^(?:http(?:s)?:\/\/)?(?:www\.)?(?:youtu\.be\/|youtube\.com\/(
        ));

        if ($validator->fails()) {
            return Redirect::back()->withErrors($validator)->withInpu
        }

        $resource->update($input);

        $resource->toVideo()->update(array(
            'url' => Input::get('attached')
        ));

        return Redirect::route('course.manager', $resource->course->c
    }

    private function updateFileResource(Resource $resource) {
        $input = Input::only('title', 'description');
        $input = array_add($input, 'file', Input::file('upload'));

        $validator = Validator::make($input, array(
            'title' => 'required|min:5|max:120',
            'description' => 'max:500',
            'file' => 'max:10240'
        ));

        if ($validator->fails()) {
            $input = array_forget($input, 'file');

            return Redirect::back()->withErrors($validator)->withInpu
        }

        $resource->update($input);

        if (Input::hasFile('upload')) {
            $file = Input::file('upload');

            if (File::exists('uploads/resources/' . $resource->resour
                File::delete('uploads/resources/' . $resource->resour
            }

            $resource->toFile()->update(array(
                'file_name' => $file->getClientOriginalName(),
                'is_pdf' => 'application/pdf' === $file->getMimeType(
            ));

            $file->move('uploads/resources', $resource->resource_id);
        }

        return Redirect::route('course.manager', $resource->course->c
    }

    private function updateUrlResource(Resource $resource) {
        $input = Input::only('title', 'description', 'attached');

        $validator = Validator::make($input, array(
            'title' => 'required|min:5|max:120',
            'description' => 'max:500',
            'attached' => 'required|url'
        ));

        if ($validator->fails()) {
            return Redirect::back()->withErrors($validator)->withInpu
        }

        $resource->update($input);

        $resource->toUrl()->update(array(
            'url' => Input::get('attached')
        ));

        return Redirect::route('course.manager', $resource->course->c
    }

    public function update(Resource $resource) {
        switch ($resource->type) {
            case 'video':
                return $this->updateVideoResource($resource);
            case 'file':
                return $this->updateFileResource($resource);
            case 'url':
                return $this->updateUrlResource($resource);
        }
    }

    public function delete(Resource $resource) {
        if (File::exists('uploads/resources/' . $resource->resource_i
            File::delete('uploads/resources/' . $resource->resource_i
        }

        $resource->delete();

```



```
        return Redirect::route('course.manager', $resource->course->c
    }

    public function view(Resource $resource) {
        $this->layout->content = View::make('resources.view')
            ->with('course', $resource->course)
            ->with('resource', $resource);
    }

    public function download(Resource $resource) {
        if ($resource->is('file')) {
            return Response::download('uploads/resources/' . $resourc
        }

        return Redirect::route('course.view', $resource->course->cour
    }
}
```



```

    public function add(Course $course, $type) {
        $this->layout->content = View::make('resources.add')
            ->with('course', $course)
            ->with('type', $type);
    }

    private function createVideoResource(Course $course) {
        $input = Input::only('title', 'description', 'type', 'url');

        $validator = Validator::make($input, array(
            'title' => 'required|min:5|max:120',
            'description' => 'max:500',
            'url' => array('required', 'url',
"regex:/^(?:http(?:s)?:\/\/)?(?:www\.)?(?:youtu\.be\/|youtube\.com\/(
        ));

        if ($validator->fails()) {
            return Redirect::back()->withErrors($validator)->withInpu
        }

        $resource = new Resource($input);

        $course->resources()->save($resource);

        $videoResource = new VideoResource($input);

        $resource->toVideo()->save($videoResource);

        return Redirect::route('course.manager', $course->course_id);
    }

    private function createFileResource(Course $course) {
        $input = Input::only('title', 'description', 'type');
        $input = array_add($input, 'file', Input::file('upload'));

        $validator = Validator::make($input, array(
            'title' => 'required|min:5|max:120',
            'description' => 'max:500',
            'file' => 'required|max:10240'
        ));

        if ($validator->fails()) {
            $input = array_forget($input, 'file');

            return Redirect::back()->withErrors($validator)->withInpu
        }

        $resource = new Resource($input);

        $course->resources()->save($resource);

        $file = Input::file('upload');
        $fileResource = new FileResource(array(
            'file_name' => $file->getClientOriginalName(),
            'is_pdf' => 'application/pdf' === $file->getMimeType()
        ));

        if ($file->move('uploads/resources', $resource->resource_id))
            $resource->toFile()->save($fileResource);
        } else {
            $resource->delete();
        }

        return Redirect::route('course.manager', $course->course_id);
    }

    private function createUrlResource(Course $course) {
        $input = Input::only('title', 'description', 'type', 'url');

        $validator = Validator::make($input, array(
            'title' => 'required|min:5|max:120',
            'description' => 'max:500',
            'url' => 'required|url'
        ));

        if ($validator->fails()) {
            return Redirect::back()->withErrors($validator)->withInpu
        }

        $resource = new Resource($input);

        $course->resources()->save($resource);

        $urlResource = new UrlResource($input);

        $resource->toUrl()->save($urlResource);

        return Redirect::route('course.manager', $course->course_id);
    }

    public function create(Course $course) {
        if (Input::has('type')) {
            switch (Input::get('type')) {
                case 'video':
                    return $this->createVideoResource($course);
                case 'file':
                    return $this->createFileResource($course);
                case 'url':
                    return $this->createUrlResource($course);
            }
        }

        return Redirect::route('course.manager', $course->course_id);
    }

    public function edit(Resource $resource) {
        $this->layout->content = View::make('resources.edit')
            ->with('course', $resource->course)
            ->with('resource', $resource);
    }

    private function updateVideoResource(Resource $resource) {
        $input = Input::only('title', 'description', 'attached');

        $validator = Validator::make($input, array(
            'title' => 'required|min:5|max:120',
            'description' => 'max:500',
            'attached' => array('required', 'url',
"regex:/^(?:http(?:s)?:\/\/)?(?:www\.)?(?:youtu\.be\/|youtube\.com\/(
        ));

        if ($validator->fails()) {
            return Redirect::back()->withErrors($validator)->withInpu
        }

        $resource->update($input);

        $resource->toVideo()->update(array(
            'url' => Input::get('attached')
        ));

        return Redirect::route('course.manager', $resource->course->c
    }

    private function updateFileResource(Resource $resource) {
        $input = Input::only('title', 'description');
        $input = array_add($input, 'file', Input::file('upload'));

        $validator = Validator::make($input, array(
            'title' => 'required|min:5|max:120',
            'description' => 'max:500',
            'file' => 'max:10240'
        ));

        if ($validator->fails()) {
            $input = array_forget($input, 'file');

            return Redirect::back()->withErrors($validator)->withInpu
        }

        $resource->update($input);

        if (Input::hasFile('upload')) {
            $file = Input::file('upload');

            if (File::exists('uploads/resources/' . $resource->resour
                File::delete('uploads/resources/' . $resource->resour
            }

            $resource->toFile()->update(array(
                'file_name' => $file->getClientOriginalName(),
                'is_pdf' => 'application/pdf' === $file->getMimeType(
            ));

            $file->move('uploads/resources', $resource->resource_id);
        }

        return Redirect::route('course.manager', $resource->course->c
    }

    private function updateUrlResource(Resource $resource) {
        $input = Input::only('title', 'description', 'attached');

        $validator = Validator::make($input, array(
            'title' => 'required|min:5|max:120',
            'description' => 'max:500',
            'attached' => 'required|url'
        ));

        if ($validator->fails()) {
            return Redirect::back()->withErrors($validator)->withInpu
        }

        $resource->update($input);

        $resource->toUrl()->update(array(
            'url' => Input::get('attached')
        ));

        return Redirect::route('course.manager', $resource->course->c
    }

    public function update(Resource $resource) {
        switch ($resource->type) {
            case 'video':
                return $this->updateVideoResource($resource);
            case 'file':
                return $this->updateFileResource($resource);
            case 'url':
                return $this->updateUrlResource($resource);
        }
    }

    public function delete(Resource $resource) {
        if (File::exists('uploads/resources/' . $resource->resource_i
            File::delete('uploads/resources/' . $resource->resource_i
        }

        $resource->delete();

        return Redirect::route('course.manager', $resource->course->c
    }

    public function view(Resource $resource) {
        $this->layout->content = View::make('resources.view')
            ->with('course', $resource->course)
            ->with('resource', $resource);
    }

    public function download(Resource $resource) {
        if ($resource->is('file')) {
            return Response::download('uploads/resources/' . $resourc
        }

        return Redirect::route('course.view', $resource->course->cour
    }
}
```

### Controlador de pruebas

```
<?php

class TestController extends BaseController {

    protected $layout = 'layouts.default';

    public function add(Course $course) {
        $this->layout->content = View::make('tests.add')
            ->with('course', $course);
    }

    public function create(Course $course) {
        $input = Input::only('title', 'description');

        $validator = Validator::make($input, array(
            'title' => 'required|min:5|max:120',
            'description' => 'max:500'
        ));

        if ($validator->fails()) {
            return Redirect::back()->withErrors($validator)->withInput($input);
        }

        $input = array_add($input, 'time', 0);
        $input = array_add($input, 'created_at', Carbon::now());

        $test = new Test($input);

        $course->tests()->save($test);

        return Redirect::route('course.manager', $course->course_id);
    }

    public function edit(Test $test) {
        $this->layout->content = View::make('tests.edit')
            ->with('course', $test->course)
            ->with('test', $test);
    }

    public function update(Test $test) {
        $input = Input::only('title', 'description');

        $validator = Validator::make($input, array(
            'title' => 'required|min:5|max:120',
            'description' => 'max:500'
        ));

        if ($validator->fails()) {
            return Redirect::back()->withErrors($validator)->withInput($input);
        }

        $test->update($input);

        return Redirect::route('course.manager', $test->course->course_id);
    }

    public function delete(Test $test) {
        $test->delete();

        return Redirect::route('course.manager', $test->course->course_id);
    }

    public function manager(Test $test) {
        $this->layout->content = View::make('tests.manager')
            ->with('course', $test->course)
            ->with('test', $test);
    }

    public function publish(Course $course, Test $test) {
        $this->layout->content = View::make('tests.publish')
            ->with('course', $test->course)
            ->with('test', $test);
    }

    public function doPublish(Course $course, Test $test) {
        $input = Input::only('start_date', 'end_date', 'time');

        $validator = Validator::make($input, array(
            'start_date' => 'required|date',
            'end_date' => 'required|date',
            'time' => 'required|numeric|min:1',
        ));

        if ($validator->fails()) {
            return Redirect::back()->withErrors($validator)->withInput($input);
        }

        $now = Carbon::now();
        $startDate = Carbon::createFromFormat('Y-m-d H:i', $input['start_date']);
        $endDate = Carbon::createFromFormat('Y-m-d H:i', $input['end_date']);

        if ($now->gt($startDate)) {
            Session::push('messages', array(
                'type' => 'danger',
                'text' => "La fecha de inicio no puede ser menor a la fecha actual."
            ));

            return Redirect::back()->withInput($input);
        }

        if ($startDate->gt($endDate)) {
            Session::push('messages', array(
                'type' => 'danger',
                'text' => "La fecha de cierre no puede ser menor a la fecha de inicio."
            ));

            return Redirect::back()->withInput($input);
        }

        if ($startDate->eq($endDate)) {
            Session::push('messages', array(
                'type' => 'danger',
                'text' => "Las fechas de inicio y cierre no pueden ser iguales."
            ));

            return Redirect::back()->withInput($input);
        }

        $test->start_date = $startDate;
        $test->end_date = $endDate;
        $test->time = $input['time'];
        $test->save();

        return Redirect::route('course.manager', $test->course->course_id);
    }

    public function info(Test $test) {
        $user = Auth::user();

        if ($test->isExpired($user)) {
            $test->students()->attach($user->user_id);

            return Redirect::route('test.info', $test->test_id);
        }

        $this->layout->content = View::make('tests.info')
```

```
            ->with('course', $test->course)
            ->with('test', $test);
    }

    public function view(Test $test) {
        $now = Carbon::now();

        if (!$test->students->contains(Auth::user()->user_id)) {
            $startedAt = $now->format('Y-m-d H:i:s');

            $test->students()->attach(Auth::user()->user_id, array(
                'started_at' => $startedAt
            ));
        } else {
            $startedAt = $test->students->find(Auth::user()->user_id)->pivot->started_at;
        }

        $start = Carbon::createFromFormat('Y-m-d H:i:s', $startedAt);
        $end = $start->copy()->addMinutes($test->time);

        $seconds = ($test->time * 60) - $now->diffInSeconds($start);

        $hours = floor($seconds / 3600);
        $minutes = floor(($seconds - ($hours * 3600)) / 60);
        $seconds = $seconds - ($hours * 3600) - ($minutes * 60);

        if (1 === strlen($hours)) $hours = "0$hours";
        if (1 === strlen($minutes)) $minutes = "0$minutes";
        if (1 === strlen($seconds)) $seconds = "0$seconds";

        $time = "$hours:$minutes:$seconds";

        $userQuestions = Auth::user()->questions-
```

```
>filter(function($question) use ($test) {
      if ($test->test_id === $question->test->test_id) {
        return $question;
      }
    });

    $this->layout->content = View::make('tests.view')
      ->with('course', $test->course)
      ->with('test', $test)
      ->with('time', $time)
      ->with('userQuestions', $userQuestions);
  }

  public function submit(Test $test) {
    $user = Auth::user();

    $user->questions->each(function($question) use ($user, $test) {
      if ($test->test_id == $question->test->test_id) {
        $user->questions()->detach($question->question_id);
      }
    });

    $input = Input::all();

    foreach ($input as $key => $value) {
      if (starts_with($key, 'question')) {
        $questionId = substr($key, strlen('question'));

        $question = Question::find($questionId);

        if ($question) {
          if (is_array($value)) {
            foreach ($value as $text) {
              $user->questions()->attach($questionId, array('answer_text' => $text));
            }
```

```
                    } else {
                        if ($value !== '') {
                            $user->questions()->attach($questionId, array('answer_text' => $value));
                        }
                    }
                }
            }
        }

        if (Input::has('save')) {
            return Redirect::back();
        }

        $user = User::find($user->user_id);

        $pivot = $user->tests->find($test->test_id)->pivot;

        $pivot->is_finished = true;

        $userPoints = $test->userPoints($user);
        $totalPoints = $test->totalPoints();

        $qualification = ($userPoints / $totalPoints) * 100.0;

        $pivot->qualification = $qualification;
        $pivot->save();

        return Redirect::route('test.info', $test->test_id);
    }

    public function qualifications(Test $test) {
        $this->layout->content = View::make('tests.qualifications')
            ->with('course', $test->course)
            ->with('test', $test);
    }
}
```

## Modelos

### Respuestas

```php
<?php

class Answer extends Eloquent {

    protected $table = 'answers';

    protected $primaryKey = 'answer_id';

    protected $fillable = array('answer_text', 'is_correct');

    public $timestamps = false;

    public function question() {
        return $this->belongsTo('Question');
    }
}
```

### Cursos

```php
<?php

class Course extends Eloquent {

    protected $table = 'courses';

    protected $primaryKey = 'course_id';

    protected $fillable = array('course_name', 'description', 'password', 'created_at');

    public $timestamps = false;

    public function getTestsAttribute() {
        return $this->tests()->getQuery()->orderBy('start_date', 'ASC')->get();
    }

    public function getResourcesAttribute() {
        return $this->resources()->getQuery()->orderBy('created_at', 'ASC')->get();
    }

    public function teacher() {
        return $this->belongsTo('User', 'teacher_id');
    }

    public function students() {
        return $this->belongsToMany('User', 'users_courses');
    }

    public function tests() {
        return $this->hasMany('Test');
    }

    public function resources() {
        return $this->hasMany('Resource');
    }
}
```

### Recurso “archivo”

```php
<?php

class FileResource extends Eloquent {

    protected $table = 'file_resources';

    protected $primaryKey = 'resource_id';

    protected $fillable = array('file_name', 'is_pdf');

    public $timestamps = false;

    public function resource() {
        return $this->belongsTo('Resource');
    }
}
```

### Kardex

```php
<?php

class Kardex extends Eloquent {

    protected $table = 'kardex';

    protected $fillable = array('user_id', 'course_name', 'status', 'qualification');

    public $timestamps = false;

    public function student() {
        return $this->belongsTo('User');
    }
}
```

### Preguntas

```php
<?php

class Question extends Eloquent {

    protected $table = 'questions';

    protected $primaryKey = 'question_id';

    protected $fillable = array('question_text', 'type', 'points');

    public $timestamps = false;

    public function is($type) {
        return $this->type === $type;
    }

    public function test() {
        return $this->belongsTo('Test');
    }

    public function answers() {
        return $this->hasMany('Answer');
    }

    public function students() {
        return $this->belongsToMany('User', 'users_questions')->withPivot('answer_text');
    }
}
```

### Recursos

```
<?php

class Resource extends Eloquent {

    protected $table = 'resources';

    protected $primaryKey = 'resource_id';

    protected $fillable = array('title', 'description', 'type');

    public $timestamps = true;

    public function getAttachedAttribute() {
        switch ($this->type) {
            case 'video': return $this->toVideo->url;
            case 'file': return $this->toFile->file_name;
            case 'url': return $this->toUrl->url;
        }
    }

    public function is($type) {
        return $this->type === $type;
    }

    public function course() {
        return $this->belongsTo('Course');
    }

    public function toVideo() {
        return $this->is('video') ? $this->hasOne('VideoResource') : NULL;
    }

    public function toFile() {
        return $this->is('file') ? $this->hasOne('FileResource') : NULL;
    }

    public function toUrl() {
        return $this->is('url') ? $this->hasOne('UrlResource') : NULL;
    }
}
```

### Examen

```
<?php

class Test extends Eloquent {

    protected $table = 'tests';

    protected $primaryKey = 'test_id';

    protected $fillable = array('title', 'description', 'time', 'created_at');

    public $timestamps = false;

    public function isStartGreaterThanNow() {
        if ($this->start_date) {
            $now = Carbon::now();
            $start = Carbon::createFromFormat('Y-m-d H:i:s', $this->start_date);

            if ($start->gt($now)) {
                return true;
            }
        }

        return false;
    }

    public function isEndGreaterThanNow() {
        if ($this->end_date) {
            $now = Carbon::now();
            $end = Carbon::createFromFormat('Y-m-d H:i:s', $this->end_date);

            if ($end->gt($now)) {
                return true;
            }
        }

        return false;
    }

    public function isBetweenNow() {
        if ($this->start_date && $this->end_date) {
            $now = Carbon::now();
            $start = Carbon::createFromFormat('Y-m-d H:i:s', $this->start_date);
            $end = Carbon::createFromFormat('Y-m-d H:i:s', $this->end_date);

            if ($now->between($start, $end)) {
                return true;
            }
        }

        return false;
    }

    public function isFinalized() {
        return $this->isPublished() && !$this->isEndGreaterThanNow();
    }

    public function isEditable() {
        return !$this->start_date && !$this->end_date;
    }

    public function isPublished() {
        return $this->start_date && $this->end_date;
    }

    public function hasQualification(User $user) {
        return $user->tests->contains($this->test_id);
    }

    public function isExpired(User $user) {
        if ($user->tests->contains($this->test_id)) {
            return false;
        }

        if ($this->start_date && $this->end_date) {
            $now = Carbon::now();
            $end = Carbon::createFromFormat('Y-m-d H:i:s', $this->end_date);

            if ($now->gt($end)) {
                return true;
            }
        }

        return false;
    }

    public function canStart(User $user) {
        if ($this->start_date && $this->end_date) {
            $now = Carbon::now();
            $start = Carbon::createFromFormat('Y-m-d H:i:s', $this->start_date);
            $end = Carbon::createFromFormat('Y-m-d H:i:s', $this->end_date);

            if ($user->tests->contains($this->test_id)) {
                $pivot = $user->tests->find($this->test_id)->pivot;

                if ($pivot->is_finished) {
                    return false;
                }

                if ($pivot->started_at) {
                    $started =
Carbon::createFromFormat('Y-m-d H:i:s', $pivot->started_at);

          if ($now->diffInMinutes($started) < $this->time) {
            return true;
          }

          return false;
        }
      }

      if ($now->between($start, $end)) {
        return true;
      }
    }

    return false;
  }

  public function isFinished(User $user) {
    if ($user->tests->contains($this->test_id)) {
      $pivot = $user->tests->find($this->test_id)->pivot;

      if ($pivot->is_finished) {
        return true;
      }
    }

    return false;
  }

  public function userPoints(User $user) {
    $points = 0;

    if (!$this->isFinished($user)) {
      return $points;
    }

    $user->questions->each(function($question) use (&$points) {
            if ($question->test->test_id === $this->test_id) {
                $question->answers->each(function($answer) use ($question, &$points) {
                    if ($answer->answer_text == $question->pivot->answer_text) {
                        if ($answer->is_correct) {
                            $points += $question->points;
                        } else {
                            if ($question->is('multiple')) {
                                $points -= $question->points;
                            }
                        }
                    }
                });
            }
        });

        return $points;
    }

    public function totalPoints() {
        $points = 0;

        $this->questions->each(function($question) use (&$points) {
            if ($question->type === 'open' || $question->type === 'choice') {
                $points += $question->points;
            } else {
                $question->answers->each(function($answer) use ($question, &$points) {
                    if ($answer->is_correct) {
                        $points += $question->points;
                    }
                });
            }
        });

```

```php
        return $points;
    }

    public function hasQuestions() {
        return 0 !== $this->questions->count();
    }

    public function getStartDateFormatAttribute() {
        if ($this->attributes['start_date']) {
            $date = Carbon::createFromFormat('Y-m-d H:i:s', $this->attributes['start_date']);

            return $date->formatLocalized('%A %d %B %Y') . ' - ' . $date->format('h:i:s A');
        }
    }

    public function getEndDateFormatAttribute($value) {
        if ($this->attributes['end_date']) {
            $date = Carbon::createFromFormat('Y-m-d H:i:s', $this->attributes['end_date']);

            return $date->formatLocalized('%A %d %B %Y') . ' - ' . $date->format('h:i:s A');
        }
    }

    public function course() {
        return $this->belongsTo('Course');
    }

    public function questions() {
        return $this->hasMany('Question')->with('answers');
    }

    public function students() {
        return $this->belongsToMany('User', 'users_tests')->withPivot('started_at', 'is_finished', 'qualification');
    }
}
```

### Recurso “URL”

```php
<?php

class UrlResource extends Eloquent {

    protected $table = 'url_resources';

    protected $primaryKey = 'resource_id';

    protected $fillable = array('url');

    public $timestamps = false;

    public function resource() {
        return $this->belongsTo('Resource');
    }
}
```

### Usuario

```php
<?php

use Illuminate\Auth\UserInterface;
use Illuminate\Auth\Reminders\RemindableInterface;

class User extends Eloquent implements UserInterface, RemindableInterface {

    protected $table = 'users';

    protected $primaryKey = 'user_id';

    protected $hidden = array('password');

    protected $fillable = array('first_name', 'last_name', 'email', 'user_name', 'password');

    public $timestamps = false;

    public function getAvatarAttribute() {
        return "http://www.gravatar.com/avatar/" . md5(strtolower(trim($this->email))) . "?d=mm";
    }

    public function getAnswer(Question $question) {
        if ($this->questions->contains($question->question_id)) {
            return $this->questions->find($question->question_id)->pivot->answer_text;
        }

        return '';
    }

    public function hasAnswer(Question $question, $answer_text) {
        foreach ($this->questions as $userQuestion) {
            if ($question->question_id === $userQuestion->question_id) {
                if ($userQuestion->pivot->answer_text == $answer_text) {
                    return true;
                }
            }
        }

        return false;
    }

    public function setUserNameAttribute($value) {
        $this->attributes['user_name'] = strtolower($value);
    }

    public function setPasswordAttribute($value) {
        $this->attributes['password'] = Hash::make($value);
    }

    public function getFullNameAttribute() {
        return $this->first_name . ' ' . $this->last_name;
    }

```

```php
    public function getAuthIdentifier() {
        return $this->getKey();
    }

    public function getAuthPassword() {
        return $this->password;
    }

    public function getReminderEmail() {
        return $this->email;
    }

    public function teacherCourses() {
        return $this->hasMany('Course', 'teacher_id');
    }

    public function studentCourses() {
        return $this->belongsToMany('Course',
'users_courses');
    }

    public function kardex() {
        return $this->hasMany('kardex');
    }

    public function questions() {
        return $this->belongsToMany('Question',
'users_questions')->withPivot('answer_text');
    }

    public function tests() {
        return $this->belongsToMany('Test',
'users_tests')->withPivot('started_at', 'is_finished',
'qualification');
    }
}
```

### Recurso “Video”

```php
<?php

class VideoResource extends Eloquent {

    protected $table = 'video_resources';

    protected $primaryKey = 'resource_id';

    protected $fillable = array('url');

    public $timestamps = false;

    public function getVideoIdAttribute() {
        if (preg_match("/^(?:http(?:s)?:\/\/)?(?:www\.)?(?:youtu\.be\
            return $id[1];
        }

        return '';
    }

    public function resource() {
        return $this->belongsTo('Resource');
    }
}
```

# Vistas

## Autentificacion

### Inicar sesión

```
@section('content')
    <form class="form-horizontal" role="form" action="{{ route('do.login') }}" method="POST">
        @include('includes.field', array('name' => 'user_name', 'label' => 'Nombre de usuario', 'type' => 'text'))
        @include('includes.field', array('name' => 'password', 'label' => 'Contraseña', 'type' => 'password'))
        <div class="form-group">
            <div class="col-sm-offset-2 col-sm-10">
                <div class="checkbox">
                    <label>
                        <input type="checkbox" name="remember"> Recordarme?
                    </label>
                </div>
            </div>
        </div>
        <div class="form-group">
            <div class="col-sm-offset-2 col-sm-10">
                <button type="submit" class="btn btn-primary">Iníciar sesión</button>
            </div>
        </div>
    </form>
@stop
```

### Registro

```
@section('content')
    <form class="form-horizontal" role="form" action="{{ route('do.login') }}" method="POST">
        @include('includes.field', array('name' => 'user_name', 'label' => 'Nombre de usuario', 'type' => 'text'))
        @include('includes.field', array('name' => 'password', 'label' => 'Contraseña', 'type' => 'password'))
        <div class="form-group">
            <div class="col-sm-offset-2 col-sm-10">
                <div class="checkbox">
                    <label>
                        <input type="checkbox" name="remember"> Recordarme?
                    </label>
                </div>
            </div>
        </div>
        <div class="form-group">
            <div class="col-sm-offset-2 col-sm-10">
                <button type="submit" class="btn btn-primary">Iníciar sesión</button>
            </div>
        </div>
    </form>
@stop
```

### Bienvenida

```
@section('content')
    <div class="jumbotron">
        <div class="text-center">
            <p>¿Ya eres parte de SIAD?</p>
            <p>
                <a class="btn btn-primary btn-lg btn-block" href="{{ route('login') }}">
                    Inícia sesión
                </a>
            </p>
            <p>¿Quieres ser parte de SIAD?</p>
            <p>
                <a class="btn btn-primary btn-lg btn-block" href="{{ route('register') }}">
                    Registrate
                </a>
            </p>
        </div>
        <blockquote align="right">
            <small>SIAD</small>
        </blockquote>
    </div>
@stop
```

## Cursos

### Añadir

```
@section('content')
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h3 class="panel-title">Nuevo curso</h3>
        </div>
        <form role="form" class="form-horizontal" action="{{ route('course.create') }}" method="POST">
        <div class="panel-body">
            @include('includes.field', array('name' => 'course_name', 'label' => 'Nombre', 'type' => 'text'))
            @include('includes.field', array('name' => 'description', 'label' => 'Descripcion', 'type' => 'textarea'))
            @include('includes.field', array('name' => 'password', 'label' => 'Contraseña', 'type' => 'password'))
        </div>
        <div class="panel-footer">
            <button type="submit" class="btn btn-primary btn-lg btn-block">Crear</button>
            <a class="btn btn-default btn-lg btn-block" href="{{ route('course.lists') }}">Cancelar</a>
        </div>
        </form>
    </div>
@stop
```

### Editar

```
@section('content')
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h3 class="panel-title">Editar curso: {{ $course->course_name }}</h3>
        </div>
        {{ Form::model($course, array('route' => array('course.update', $course->course_id))) }}
        <div class="panel-body">
            @include('includes.field', array('name' => 'course_name', 'label' => 'Nombre', 'type' => 'text'))
            @include('includes.field', array('name' => 'description', 'label' => 'Descripcion', 'type' => 'textarea'))
            @include('includes.field', array('name' => 'password', 'label' => 'Contraseña', 'type' => 'password'))
        </div>
        <div class="panel-footer">
            <button type="submit" class="btn btn-primary btn-lg btn-block">Guardar</button>
            <a class="btn btn-default btn-lg btn-block" href="{{ route('course.manager', $course->course_id) }}">Cancelar</a>
        </div>
        {{ Form::close() }}
    </div>
@stop
```

### Listas

```
@section('content')
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h1 class="panel-title">Cursos</h1>
        </div>
        <div class="panel-body">
            <h2>Cursos que imparto</h2>
            <a class="btn btn-primary" href="{{ route('course.add') }}">Crear curso</a>
            <div class="table-responsive">
```

```
                <table class="table table-striped table-hover table-condensed table-bordered">
                    <thead>
                        <tr>
                            <th>Nombre del curso</th>
                            <th>Estudiantes</th>
                            <th></th>
                        </tr>
                    </thead>
                    <tbody>
                        @foreach ($teacherCourses as $course)
                            <tr>
                                <td>
                                    {{ $course->course_name }}
                                    @if (!$course->started_at)
                                        <span class="label label-info">Aún no inicia</span>
                                    @endif
                                </td>
                                <td>{{ $course->students->count() }}</td>
                                <td>
                                    <a class="btn btn-primary btn-xs" href="{{ route('course.manager', $course->course_id) }}">
                                        <span class="glyphicon glyphicon-eye-open"></span> Ver
                                    </a>
                                </td>
                            </tr>
                        @endforeach
                    </tbody>
                </table>
            </div>
            <h2>Cursos en los que estoy inscrito</h2>
            <div class="table-responsive">
                <table class="table table-striped table-hover table-condensed table-bordered">
                    <thead>
```

```
                        <tr>
                            <th>Nombre del curso</th>
                            <th>Nombre del maestro</th>
                            <th></th>
                        </tr>
                    </thead>
                    <tbody>
                        @foreach ($studentCourses as $course)
                            <tr>
                                <td>
                                    {{ $course->course_name }}
                                    @if (!$course->started_at)
                                        <span class="label label-info">Aún no inicia</span>
                                    @endif
                                </td>
                                <td>{{ $course->teacher->full_name }}</td>
                                <td>
                                    <a class="btn btn-primary btn-xs" href="{{ route('course.view', $course->course_id) }}">
                                        <span class="glyphicon glyphicon-eye-open"></span> Ver
                                    </a>
                                    <a class="btn btn-info btn-xs" href="{{ route('course.qualifications', $course->course_id) }}">
                                        <span class="glyphicon glyphicon-list-alt"></span> Calificaciones
                                    </a>
                                </td>
                            </tr>
                        @endforeach
                    </tbody>
                </table>
            </div>
        </div>
        <div class="panel-footer">
        </div>
    </div>
@stop
```

### Administrador

```
@section('content')
    <h1 class="text-center">{{ $course->course_name }}</h1>
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h3 class="panel-title">
                Información
                <span class="pull-right">
                    @if (!$course->started_at)
                        <a class="btn btn-default btn-xs" href="{{ route('course.edit', $course->course_id) }}">
                            <span class="glyphicon glyphicon-edit"></span>
                            Editar
                        </a>
                    @endif
                </span>
            </h3>
        </div>
        <div class="panel-body">
            @if ($course->started_at)
                @if (0 === $course->students->count())
                    <div class="alert alert-info">
                        Todos los estudiantes se dieron de baja
                    </div>
                @endif
            @else
                <p class="lead">
                    <small class="text-muted text-primary">
                        Comparte este enlace para permitir la entrada al curso
                    </small>
                    <input type="text" class="form-control" value="{{ route('course.register', $course->course_id) }}">
                    @if ($course->description)
                        <p><b>Descripcion:</b> {{ $course->description }}</p>
                    @endif
                    @if ($course->password)
                        <p><b>Contraseña:</b> {{ $course->password }}</p>
                    @endif
                </p>
                @if (0 === $course->students->count())
                    <p class="text-info">Actualmente el curso no tiene estudiantes, necesitas tener por lo menos un estudiante para comenzar el curso</p>
                @endif
            @endif
            @if (0 < $course->students->count())
                <p>
                    <button type="button" class="btn btn-primary btn-xs" data-toggle="collapse" data-target="#students">
                        <span class="glyphicon glyphicon-user"></span>
                        <b>Estudiantes:</b> {{ $course->students->count() }}
                    </button>
                </p>
                <div id="students" class="table-responsive collapse">
                    <table class="table table-striped table-bordered table-hover table-condensed">
                        <thead>
                            <tr>
                                <th>Nombre del estudiante</th>
                                <th>Correo</th>
                            </tr>
                        </thead>
                        <tbody>
                            @foreach ($course->students
as $student)
                                <tr>
                                    <td>{{ $student->full_name }}</td>
                                    <td>{{ $student->email }}</td>
                                </tr>
                            @endforeach
                        </tbody>
                    </table>
                </div>
            @endif
            <div class="text-center">
                @if (!$course->started_at && 0 < $course->students->count())
                    <a class="btn btn-primary btn-lg btn-block confirm tip" href="{{ route('course.start', $course->course_id) }}" data-toggle="tooltip" title="" data-original-title="Una vez iniciado el curso no se podran inscribir mas estudiantes">
                        Iniciar curso
                    </a>
                @endif
                @if ($course->started_at && !$course->closed_at)
                    <a class="btn btn-primary btn-lg btn-block confirm tip" href="{{ route('course.close', $course->course_id) }}" data-toggle="tooltip" title="" data-original-title="Al finalizar el curso se calificaran a los estudiantes, posteriormente podras reabrir el curso">
                        Finalizar curso
                    </a>
                @endif
                <a class="btn btn-default btn-lg btn-block confirm tip" href="{{ route('course.cancel', $course->course_id) }}" data-toggle="tooltip" title="" data-original-title="Esto eliminara totalmente el curso del sistema">
                    Cancelar curso
                </a>
            </div>
```

```
        </div>
    </div>
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h3 class="panel-title">
                Examenes
                <span class="pull-right">
                    <a class="btn btn-default btn-xs" href="{{ route('test.add', $course->course_id) }}">
                        <span class="glyphicon glyphicon-pencil"></span>
                        Agregar
                    </a>
                </span>
            </h3>
        </div>
        <table class="table table-striped table-bordered table-hover table-condensed">
            <thead>
                <tr>
                    <th>Titulo</th>
                    <th>Estado</th>
                    <th></th>
                </tr>
            </thead>
            <tbody>
                @foreach ($course->tests as $test)
                    <tr>
                        <td>{{ $test->title }}</td>
                        <td>
                            @if ($test->isPublished())
                                Publicado
                                <p><b>Inicio:</b> {{ $test->start_date_format }}</p>
                                <p><b>Cierre:</b> {{ $test->end_date_format }}</p>
                            @endif
                            @if ($test->isEditable())
                                Editable
                            @endif
                            @if ($test->isFinalized())
                                Finalizado
                            @endif
                        </td>
                        <td>
                            @if ($test->isEditable())
                                <a class="btn btn-primary btn-xs" href="{{ route('test.edit', $test->test_id) }}">
                                    <span class="glyphicon glyphicon-edit"></span> Editar
                                </a>
                                <a class="btn btn-primary btn-xs" href="{{ route('test.manager', $test->test_id) }}">
                                    <span class="glyphicon glyphicon-edit"></span> Administrar
                                </a>
                                @if ($test->hasQuestions() && $course->started_at)
                                    <a class="btn btn-primary btn-xs" href="{{ route('test.publish', array($course->course_id, $test->test_id)) }}">
                                        <span class="glyphicon glyphicon-send"></span> Publicar
                                    </a>
                                @endif
                                <a class="btn btn-danger btn-xs confirm" href="{{ route('test.delete', $test->test_id) }}">
                                    <span class="glyphicon glyphicon-remove"></span> Eliminar
                                </a>
                            @endif
                            @if ($test->isFinalized())
                                <a class="btn btn-default btn-xs" href="{{ route('test.qualifications', $test->test_id) }}">
                                    <span class="glyphicon glyphicon-list-alt"></span> Calificaciones
                                </a>
                            @endif
                        </td>
```

```
                    </tr>
                @endforeach
            </tbody>
        </table>
        <div class="panel-footer">
        </div>
    </div>
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h3 class="panel-title">
                Recursos
                <span class="pull-right">
                    <div class="btn-group">
                        <button type="button" class="btn btn-default btn-xs dropdown-toggle" data-toggle="dropdown">
                            <span class="glyphicon glyphicon-pencil"></span>
                            Agregar <span class="caret"></span>
                        </button>
                        <ul class="dropdown-menu pull-right" role="menu">
                            <li>
                                <a href="{{ route('resource.add', array($course->course_id, 'video')) }}">
                                    <span class="glyphicon glyphicon-play"></span>
                                    Video
                                </a>
                            </li>
                            <li>
                                <a href="{{ route('resource.add', array($course->course_id, 'file')) }}">
                                    <span class="glyphicon glyphicon-file"></span>
                                    Archivo
                                </a>
                            </li>
                            <li>
```

```
                                <a href="{{ route('resource.add', array($course->course_id, 'url')) }}">
                                    <span class="glyphicon glyphicon-link"></span>
                                    Enlace
                                </a>
                            </li>
                        </ul>
                    </div>
                </span>
            </h3>
        </div>
        <table class="table table-striped table-bordered table-hover table-condensed">
            <thead>
                <tr>
                    <th>Titulo</th>
                    <th>Tipo</th>
                    <th></th>
                </tr>
            </thead>
            <tbody>
                @foreach ($course->resources as $resource)
                    <tr>
                        <td>{{ $resource->title }}</td>
                        <td>
                            @if ($resource->type === 'video')
                                <span class="glyphicon glyphicon-play"></span>
                                Video
                            @endif
                            @if ($resource->type === 'file')
                                <span class="glyphicon glyphicon-file"></span> Archivo
                            @endif
                            @if ($resource->type === 'url')
                                <span class="glyphicon
```

```
glyphicon-link"></span> Enlace
                            @endif
                        </td>
                        <td>
                            <a class="btn btn-primary
btn-xs" href="{{ route('resource.edit', $resource-
>resource_id) }}">
                                <span class="glyphicon
glyphicon-edit"></span> Editar
                            </a>
                            <a class="btn btn-default
btn-xs" href="{{ route('resource.view', $resource-
>resource_id) }}">
                                <span class="glyphicon
glyphicon-eye-open"></span> Ver
                            </a>
                            <a class="btn btn-danger
btn-xs confirm" href="{{ route('resource.delete',
$resource->resource_id) }}">
                                <span class="glyphicon
glyphicon-remove"></span> Eliminar
                            </a>
                        </td>
                    </tr>
                @endforeach
            </tbody>
        </table>
        <div class="panel-footer">
        </div>
    </div>
@stop
```

### Calificaciones

```
@section('content')
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h1 class="panel-title">Calificaciones</h1>
        </div>
        <div class="panel-body">
            <dl class="dl-horizontal">
                <dt>Profesor</dt>
                <dd>{{ $course->teacher->full_name }}</dd>
                <dt>Curso</dt>
                <dd>{{ $course->course_name }}</dd>
                <dt>Descripción</dt>
                <dd>{{ $course->description }}</dd>
            </dl>
            <div class="table-responsive">
                <table class="table table-striped table-hover table-condensed table-bordered">
                    <thead>
                        <tr>
                            <th>Examen</th>
                            <th>Fecha</th>
                            <th>Calificación</th>
                        </tr>
                    </thead>
                    <tbody>
                        @foreach ($course->tests as $test)
                            @if ($test->hasQualification($user) || $test->isFinalized())
                                <tr>
                                    <td>{{ $test->title }}</td>
                                    <td>
                                        <p><b>Inicio</b> {{ $test->start_date_format }}</p>
                                        <p><b>Fin</b> {{ $test->end_date_format }}</p>
                                    </td>
                                    <td>
                                        @if ($test-
>students->contains($user->user_id))
                                            {{
round($test->students->find($user->user_id)->pivot-
>qualification, 2) }}
                                        @else
                                            0
                                        @endif
                                    </td>
                                </tr>
                            @endif
                        @endforeach
                    </tbody>
                </table>
            </div>
        </div>
        <div class="panel-footer">
        </div>
    </div>
@stop
```

### Registros

```
@section('content')
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h1 class="panel-title">{{ $course-
>course_name }}</h1>
        </div>
        <div class="panel-body">
            <dl class="dl-horizontal">
                <dt>Profesor</dt>
                <dd>{{ $course->teacher->full_name
}}</dd>
                <dt>Curso</dt>
                <dd>{{ $course->course_name }}</dd>
                <dt>Descripción</dt>
                <dd>{{ $course->description }}</dd>
            </dl>
            <form class="form form-inline" action="{{
route('course.subscribe', $course->course_id) }}"
method="POST">
                @if ($course->password)
```

```
                    @if ($errors->has('password'))
                        <div class="form-group col-xs-8 has-error">
                            <div class="input-group input-group-sm">
                                <span class="input-group-addon">
                                    <span class="glyphicon glyphicon-lock"></span>
                                </span>
                                <input type="password" class="form-control" name="password" placeholder="Contraseña">
                            </div>
                            <span class="help-block alert alert-danger">
                                {{ $errors->first('password') }}
                            </span>
                        </div>
                    @else
                        <div class="form-group col-xs-8">
                            <div class="input-group input-group-sm">
                                <span class="input-group-addon">
                                    <span class="glyphicon glyphicon-lock"></span>
                                </span>
                                <input type="password" class="form-control" name="password" placeholder="Contraseña">
                            </div>
                        </div>
                    @endif
                    <button type="submit" class="btn btn-primary btn-sm">Unirme</button>
                @else
                    <div class="text-center">
                        <button type="submit" class="btn btn-primary btn-sm">Unirme</button>
                    </div>
                @endif
            </form>
        </div>
    </div>
@stop
```

**Vista**

```
@section('content')
    <div class="text-center">
        <h1>
            {{ $course->course_name }}
            <a class="btn btn-warning btn-xs confirm" href="{{ route('course.unsubscribe', $course->course_id) }}">Dar de baja</a>
        </h1>
    </div>
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h3 class="panel-title">Examenes</h3>
        </div>
        <div class="panel-body">
            <div class="list-group">
            @foreach ($course->tests as $test)
                @if ($test->isPublished())
                    <a href="{{ route('test.info', $test->test_id) }}" class="list-group-item">
                        <h4 class="list-group-item-heading">
                            {{ $test->title }}
                            @if ($test->hasQualification($user))
                                <span class="glyphicon glyphicon-ok-circle"></span>
                            @endif
                        </h4>
                        <p class="list-group-item-text">{{ $test->description }}</p>
                    </a>
                @endif
            @endforeach
            </div>
        </div>
    </div>
    <div class="panel panel-info">
        <div class="panel-heading">
            <h3 class="panel-title">Recursos</h3>
        </div>
        <div class="panel-body">
            <div class="list-group">
            @foreach ($course->resources as $resource)
                @if ($resource->is('url'))
                    <a href="{{ $resource->toUrl->url }}" target="_blank" class="list-group-item">
                        <h4 class="list-group-item-heading">
                            {{ $resource->title }}
                            <span class="glyphicon glyphicon-link"></span>
                        </h4>
                        <p class="list-group-item-text">{{ $resource->description }}</p>
                    </a>
                @endif
                @if ($resource->is('file'))
                    <a href="{{ route('resource.view', $resource->resource_id) }}" target="_blank" class="list-group-item">
                        <h4 class="list-group-item-heading">
                            {{ $resource->title }}
                            @if ($resource->toFile->is_pdf)
                                <span class="glyphicon glyphicon-file"></span>
                            @else
                                <span class="glyphicon glyphicon-download"></span>
                            @endif
                        </h4>
                        <p class="list-group-item-text">{{ $resource->description }}</p>
                    </a>
                @endif
                @if ($resource->is('video'))
                    <a href="{{ route('resource.view', $resource->resource_id) }}" target="_blank" class="list-group-item">
                        <h4 class="list-group-item-heading">
                            {{ $resource->title }}
                            <span class="glyphicon glyphicon-play"></span>
                        </h4>
                        <p class="list-group-item-text">{{ $resource->description }}</p>
                    </a>
                @endif
            @endforeach
            </div>
        </div>
    </div>
@stop
```

## Examenes

### Añadir

```
@section('content')
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h3 class="panel-title">Agregar examen</h3>
        </div>
        <form role="form" class="form-horizontal" action="{{ route('test.create', $course->course_id) }}" method="POST">
        <div class="panel-body">
            @include('includes.field', array('name' => 'title', 'label' => 'Titulo', 'type' => 'text'))
            @include('includes.field', array('name' => 'description', 'label' => 'Descripcion', 'type' => 'textarea'))
        </div>
        <div class="panel-footer">
            <button type="submit" class="btn btn-primary btn-lg btn-block">Agregar</button>
            <a class="btn btn-default btn-lg btn-block" href="{{ route('course.manager', $course->course_id) }}">Cancelar</a>
        </div>
        </form>
    </div>
@stop
```

### Editar

```
@section('content')
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h3 class="panel-title">Editar examen</h3>
        </div>
        {{ Form::model($test, array('route' => array('test.update', $test->test_id))) }}
        <div class="panel-body">
            @include('includes.field', array('name' => 'title', 'label' => 'Titulo', 'type' => 'text'))
            @include('includes.field', array('name' => 'description', 'label' => 'Decripcion', 'type' => 'textarea'))
        </div>
        <div class="panel-footer">
            <button type="submit" class="btn btn-primary btn-lg btn-block">Guardar</button>
            <a class="btn btn-default btn-lg btn-block" href="{{ route('course.manager', $course->course_id) }}">Cancelar</a>
        </div>
        {{ Form::close() }}
    </div>
@stop
```

### Informacion

```
@section('content')
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h3 class="panel-title">
                {{ $test->title }}
                @if ($test->hasQualification($user))
                    <span class="glyphicon glyphicon-ok-circle"></span>
                @endif
            </h3>
        </div>
        <div class="panel-body">
            <dl class="dl-horizontal">
                <dt>Descripción</dt>
                <dd>{{ $test->description }}</dd>
                <dt>Fecha de inicio</dt>
                <dd>{{ $test->start_date_format }}</dd>
                <dt>Fecha de cierre</dt>
                <dd>{{ $test->end_date_format }}</dd>
                <dt>Tiempo</dt>
                <dd>{{ $test->time }} Minutos</dd>
                <dt>Numero de preguntas</dt>
                <dd>{{ $test->questions->count() }}</dd>
            </dl>
            @if ($test->canStart($user))
                <p class="text-center">
                    <a class="btn btn-lg btn-warning" href="{{ route('test.view', $test->test_id) }}">
                        Comenzar examen
                    </a>
                </p>
            @else
                @if ($test->isFinished($user))
                    <table class="table table-striped table-bordered table-hover table-condensed">
                        <thead>
                            <tr>
                                <th>Puntos</th>
                                <th>Calificacion</th>
                            </tr>
                        </thead>
                        <tbody>
                            <tr>
                                <td>{{ $test->userPoints($user) }}/{{ $test->totalPoints() }}</td>
                                <td>{{ round($user->tests->find($test->test_id)->pivot->qualification, 2) }}</td>
                            </tr>
                        </tbody>
                    </table>
                @endif
            @endif
        </div>
    </div>
@stop
```

### Administrador

```
@section('content')
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h3 class="panel-title">
                Preguntas: {{ $test->questions->count() }}
                <span class="pull-right">
                    <div class="btn-group">
                        <button type="button" class="btn btn-default btn-xs dropdown-toggle" data-toggle="dropdown">
                            <span class="glyphicon glyphicon-pencil"></span>
                            Agregar <span class="caret"></span>
                        </button>
                        <ul class="dropdown-menu pull-right" role="menu">
                            <li>
                                <a href="{{ route('question.add', array($test->test_id, 'open')) }}">
                                    <span class="glyphicon glyphicon-pencil"></span>
                                    Abierta
                                </a>
                            </li>
                            <li>
                                <a href="{{ route('question.add', array($test->test_id, 'choice')) }}">
                                    <span class="glyphicon glyphicon-check"></span>
                                    Seleccion
                                </a>
                            </li>
                            <li>
                                <a href="{{ route('question.add', array($test->test_id, 'multiple')) }}">
                                    <span class="glyphicon glyphicon-th-list"></span>
                                    Multiple
                                </a>
                            </li>
                        </ul>
                    </div>
                </span>
            </h3>
        </div>
        <table class="table table-striped table-bordered table-hover table-condensed">
            <thead>
                <tr>
                    <th>Pregunta</th>
                    <th>Tipo</th>
                    <th>Puntos</th>
                    <th>Respuestas</th>
                    <th></th>
                </tr>
            </thead>
            <tbody>
                @foreach ($test->questions as $question)
                    <tr>
                        <td>{{ $question->question_text }}</td>
                        <td>
```

```
                            @if ('open' === $question->type)
                                Abierta
                            @endif
                            @if ('choice' === $question->type)
                                Seleccion
                            @endif
                            @if ('multiple' === $question->type)
                                Multiple
                            @endif
                        </td>
                        <td>{{ $question->points }}</td>
                        <td>
                            @foreach ($question->answers as $answer)
                                <p>
                                    {{ $answer->answer_text }}
                                    @if ($answer->is_correct)
                                        <span class="glyphicon glyphicon-check"></span>
                                    @endif
                                </p>
                            @endforeach
                        </td>
                        <td>
                            <a class="btn btn-primary btn-xs" href="{{ route('question.edit', array($test->test_id, $question->question_id)) }}">
                                <span class="glyphicon glyphicon-edit"></span> Editar
                            </a>
                            <a class="btn btn-danger btn-xs confirm" href="{{ route('question.delete', array($test->test_id, $question->question_id)) }}">
                                <span class="glyphicon glyphicon-remove"></span> Eliminar
                            </a>
                        </td>
```

```
          </tr>
        @endforeach
      </tbody>
    </table>
    <div class="panel-body"></div>
    <div class="panel-footer">
      <a class="btn btn-primary btn-lg btn-block" href="{{ route('course.manager', $course->course_id) }}">Finalizar</a>
    </div>
  </div>
@stop
```

### Publicar

```
@section('content')
  <div class="panel panel-primary">
    <div class="panel-heading">
      <h3 class="panel-title">Publicar examen: {{ $test->title }}</h3>
    </div>
    <form role="form" class="form-horizontal" action="{{ route('test.do.publish', array($course->course_id, $test->test_id)) }}" method="POST">
    <div class="panel-body">
      @include('includes.field', array('name' => 'start_date', 'label' => 'Fecha de inicio', 'type' => 'text'))
      @include('includes.field', array('name' => 'end_date', 'label' => 'Fecha de cierre', 'type' => 'text'))
      @include('includes.field', array('name' => 'time', 'label' => 'Tiempo en minutos', 'type' => 'number'))
    </div>
    <div class="panel-footer">
```

```
            <button type="submit" class="btn btn-primary btn-lg btn-block">Publicar</button>
            <a class="btn btn-default btn-lg btn-block" href="{{ route('course.manager', $course->course_id) }}">Cancelar</a>
        </div>
        </form>
    </div>
@stop

@section('css')
    @parent
    {{ HTML::style('css/jquery.simple-dtpicker.css') }}
@stop

@section('js')
    @parent
    {{ HTML::script('js/jquery.simple-dtpicker.js') }}
    <script>
        (function($) {
            $('*[name=start_date]').appendDtpicker({"inline": true});
            $('*[name=end_date]').appendDtpicker({"inline": true});
        })(jQuery);
    </script>

@stop
```

### Calificaciones

```
@section('content')
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h1 class="panel-title">
                Calificaciones del examen: {{ $test->title }}
            </h1>
        </div>
        <table class="table table-striped table-hover table-condensed table-bordered">
            <thead>
                <tr>
                    <th>Alumno</th>
                    <th>Calificación</th>
                </tr>
            </thead>
            <tbody>
                @foreach ($course->students as $student)
                    <tr>
                        <td>{{ $student->full_name }}</td>
                        <td>
                            @if ($student->tests->contains($test->test_id))
                                {{ $student->tests->find($test->test_id)->pivot->qualification }}
                            @else
                                0
                            @endif
                        </td>

                    </tr>
                @endforeach
            </tbody>
        </table>
        <div class="panel-footer">
```

```
      <a class="btn btn-primary btn-lg btn-block" href="{{ route('course.manager', $course->course_id) }}">Regresar</a>
    </div>
  </div>
@stop
```

### Vista

```
@section('content')
  <div class="row">
    <div class="col-lg-6 col-lg-offset-3">
      <div id="counter"></div>
      <div class="desc">
        <div>Horas</div>
        <div>Minutos</div>
        <div>Segundos</div>
      </div>
    </div>
  </div>
  <div class="panel panel-default">
    <form role="form" class="form form-horizontal" action="{{ route('test.submit', $test->test_id) }}" method="POST">
    <div class="panel-body">
      @foreach ($test->questions as $key => $question)
      <div class="row">
```

```
                <div class="col-lg-8 col-lg-offset-2">
                    <div id="item-{{ $key }}" class="item">
                        <label>{{ $key + 1 }}.- {{ $question->question_text }}</label>
                        @foreach ($question->answers as $answer)
                            @if ($question->is('open'))
                                <div class="input-group input-group-sm">
                                    <span class="input-group-addon">
                                        <span class="glyphicon glyphicon-pencil"></span>
                                    </span>
                                    <input type="text" name="question{{ $question->question_id }}" value="{{ $user->getAnswer($question) }}" class="form-control">
                                </div>
                            @endif
                            @if ($question->is('choice'))
                                <div class="input-group input-group-sm">
                                    <span class="input-group-addon">
                                        @if ($user->hasAnswer($question, $answer->answer_text))
                                            <input type="radio" name="question{{ $question->question_id }}" value="{{ $answer->answer_text }}" checked>
                                        @else
                                            <input type="radio" name="question{{ $question->question_id }}" value="{{ $answer->answer_text }}">
                                        @endif
```

```
                  </span>
                  <b class="form-control">{{ $answer->answer_text }}</b>
                </div>
              @endif
              @if ($question->is('multiple'))
                <div class="input-group input-group-sm">
                  <span class="input-group-addon">
                    @if ($user->hasAnswer($question, $answer->answer_text))
                      <input type="checkbox" name="question{{ $question->question_id }}[]" value="{{ $answer->answer_text }}" checked>
                    @else
                      <input type="checkbox" name="question{{ $question->question_id }}[]" value="{{ $answer->answer_text }}">
                    @endif
                  </span>
                  <b class="form-control">{{ $answer->answer_text }}</b>
                </div>
              @endif
            @endforeach
          </div>
        </div>
      </div>
```

```
            @endforeach
        </div>
        <div class="panel-footer">
            <div class="row">
                <div class="col-xs-4-col col-xs-offset-4">
                    <div id="paginator"></div>
                </div>
            </div>
            <button type="submit" class="btn btn-primary btn-lg btn-block" name="save" value="1">Guardar sin enviar</button>
            <button type="submit" class="btn btn-default btn-lg btn-block">Enviar</button>
        </div>
        </form>
    </div>
@stop

@section('css')
    @parent
    <style type="text/css">
        .cntSeparator {
            font-size: 54px;
            margin: 10px 7px;
            color: #000;
        }
        .desc div {
            float: left;
            font-family: Arial;
            width: 70px;
            margin-right: 65px;
            font-size: 13px;
            font-weight: bold;
            color: #000;
        }
    </style>
@stop

@section('js')
    @parent
    {{ HTML::script('js/jquery.countdown.min.js') }}
    <script>
        (function($) {
            $('#counter').countdown({
                stepTime: 1,
                startTime: "{{ $time }}",
                timerEnd: function() {
                    window.location.replace('{{ route('test.info', $test->test_id) }}');
                },
                image: "{{ URL::to('img/digits.png') }}"
            });

            var itemsOnPage = 5;

            $('#paginator').pagination({
                prevText: 'Anterior',
                nextText: 'Siguiente',
                items: {{ $test->questions->count() }},
                itemsOnPage: itemsOnPage,
                cssStyle: 'light-theme',
                onInit: function() {
                    var pages = $('#paginator').pagination('getPagesCount');

                    if (1 == pages) {
                        $('#paginator').pagination('disable');
            $('#paginator').hide();
          } else {
            $('.item').hide();

            var current =
$('#paginator').pagination('getCurrentPage');
            var first = (current - 1) *
itemsOnPage;
            var end = current * itemsOnPage;

            for (var i = first; i < end; i++)
{
              $('#item-' + i).show();
            }
          }
        },
        onPageClick: function(page, event) {
          event.preventDefault();

          $('.item').hide();

          var current =
$('#paginator').pagination('getCurrentPage');
          var first = (current - 1) *
itemsOnPage;
          var end = current * itemsOnPage;
```

```

                    for (var i = first; i < end; i++) {
                        $('#item-' + i).show();
                    }
                }
            });
        })(jQuery);
    </script>
@stop
```

## Macros

### Depuración

```
@if (Config::get('app.debug', false))
    <div class="list-group">
        @foreach (DB::getQueryLog() as $log)
            <a href="#" class="list-group-item">
                <h4 class="list-group-item-heading"><b>Query:</b> {{ $log['query'] }}</h4>
                <p class="list-group-item-text">
                    <b>Bindings:</b>
                    {@foreach ($log['bindings'] as $binding)
                        {{ $binding }}
                    @endforeach}
                </p>
                <b>Time: <small>{{ $log['time'] }}s</small></b>
            </a>
        @endforeach
    </div>
@endif
```

### Campo

```
@if ($errors->has($name))
    <div class="form-group has-error">
        {{ Form::label($name, $label, array('class' => 'col-sm-2 control-label')) }}
        <div class="col-sm-10">
            @if ('textarea' === $type)
                {{ Form::textarea($name, null, array('class' => 'form-control')) }}
            @else
                {{ Form::input($type, $name, null, array('class' => 'form-control')) }}
            @endif
            <span class="help-block alert alert-danger">{{ $errors->first($name) }}</span>
        </div>
    </div>
@else
    <div class="form-group">
        {{ Form::label($name, $label, array('class' => 'col-sm-2 control-label')) }}
        <div class="col-sm-10">
            @if ('textarea' === $type)
                {{ Form::textarea($name, null, array('class' => 'form-control')) }}
            @else
                {{ Form::input($type, $name, null, array('class' => 'form-control')) }}
            @endif
        </div>
    </div>
@endif
```

### Barra lateral

```
<ul class="nav nav-pills nav-stacked">
    @foreach ($links as $link)
        @if (!isset($link['route']))
            <li><a href="#">{{ $link['name'] }}</a></li>
        @else
            @if (is_array($link['route']))
                @if (in_array(Route::currentRouteName(), $link['route']))
                    <li class="active" ><a href="{{ route($link['route'][0]) }}">{{ $link['name'] }}</a></li>
                @else
                    <li><a href="{{ route($link['route'][0]) }}">{{ $link['name'] }}</a></li>
                @endif
            @else
                @if (strcasecmp(Route::currentRouteName(), $link['route']) == 0)
                    <li class="active"><a href="{{ route($link['route']) }}">{{ $link['name'] }}</a></li>
                @else
                    <li><a href="{{ route($link['route']) }}">{{ $link['name'] }}</a></li>
                @endif
            @endif
        @endif
    @endforeach
</ul>
```

## Perfiles

### Editar

```
@section('content')
    {{ Form::model($user, array('route' => 'profile.update', 'class' => 'form-horizontal', 'role' => 'form')) }}
        <div class="panel panel-warning">
            <div class="panel-heading">
                <h1 class="panel-title">Editar Perfil</h1>
            </div>
            <div class="panel-body">
                @include('includes.field', array('name' => 'first_name', 'label' => 'Nombre', 'type' => 'text'))
                @include('includes.field', array('name' => 'last_name', 'label' => 'Apellido', 'type' => 'text'))
                @include('includes.field', array('name' => 'email', 'label' => 'Correo', 'type' => 'email'))
                @include('includes.field', array('name' => 'new_password', 'label' => 'Nueva contraseña', 'type' => 'password'))
                @include('includes.field', array('name' => 'new_password_confirmation', 'label' => 'Confirmar contraseña', 'type' => 'password'))
                @include('includes.field', array('name' => 'password', 'label' => 'Contraseña actual', 'type' => 'password'))
            </div>
            <div class="panel-footer">
                <button type="submit" class="btn btn-warning">Guardar</button>
                <a type="button" class="btn btn-default" href="{{ route('profile.view') }}">Cancelar</a>
            </div>
        </div>
    {{ Form::close() }}
@stop
```

### Vista

```
@section('content')
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h1 class="panel-title">Perfil</h1>
        </div>
        <div class="panel-body">
            <div class="media">
                <div class="pull-left">
                    <img class="media-object" src="{{
$user->avatar }}">
                </div>
                <div class="media-body">
                    <dl class="dl-horizontal">
                        <dt>Nombre:</dt>
                        <dd>{{ $user->first_name }}</dd>
                        <dt>Apellido:</dt>
                        <dd>{{ $user->last_name }}</dd>
                        <dt>Nombre de usuario:</dt>
                        <dd>{{ $user->user_name }}</dd>
                        <dt>Correo:</dt>
                        <dd>{{ $user->email }}</dd>
                    </dl>
                </div>
            </div>
        </div>
        <div class="panel-footer">
            @if (Auth::user() === $user)
                <a type="button" class="btn btn-primary"
href="{{ route('profile.edit') }}">Editar</a>
            @endif
        </div>
    </div>
@stop
```

## Plantillas

### Por defecto

```
<!doctype html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>@yield('title', 'SIAD')</title>
    <meta name="viewport" content="width=device-width,
initial-scale=1.0, maximum-scale=1.0, user-scalable=no">
    @section('css')
        {{ HTML::style('css/bootstrap.min.css') }}
        {{ HTML::style('css/simplePagination.css') }}
    @show
</head>
<body>
    <div class="container">
        <div class="row">
            <div class="col-xs-12">
                <div class="row">
                    <div class="col-xs-12">
                        <nav class="navbar navbar-default"
role="navigation">
                            <div class="navbar-header">
                                <button type="button"
```

```
class="navbar-toggle" data-toggle="collapse" data-target="#user-navbar">
                                    <span class="sr-only">Toggle navigation</span>
                                    <span class="icon-bar"></span>
                                    <span class="icon-bar"></span>
                                    <span class="icon-bar"></span>
                                </button>

                                <a class="navbar-brand" href="#">SIAD <small>Sistema Interactivo de Aprendizaje a Distancia</small></a>
                            </div>

                            <div class="collapse navbar-collapse" id="user-navbar">
                                <!-- Menu -->

                            </div>

                        </nav>

                    </div>

                </div>

                <div class="media page-header">

                    <div class="pull-left">

                        <img class="media-object" src="{{ $user->avatar }}">
                    </div>

                    <div class="media-body">

                        <h2 class="media-heading">{{ $user->first_name . ' ' . $user->last_name }}</h2>
                        <a type="button" class="btn btn-
```

```
warning btn-xs" href="{{ route('do.logout') }}">Salir</a>
                    </div>
                </div>
            </div>
        </div>
        <div class="row">
            <div class="col-xs-3">
                @section('sidebar')
                    @include('includes.sidebar')
                @show
            </div>
            <div class="col-xs-9">
                @foreach (Session::get('messages',
array()) as $message)
                    <div class="alert alert-{{
$message['type'] }}">
                        <button type="button"
class="close" data-dismiss="alert" aria-
hidden="true">&times;</button>
                        {{ $message['text'] }}
                    </div>
                    {{ Session::forget('messages') }}
                @endforeach
                @yield('content')
            </div>
        </div>
        <div class="row">
            <div class="col-xs-12">
                @include('includes.debug')
            </div>
        </div>
    </div>

    @section('js')
        {{ HTML::script('js/jquery.min.js') }}
        {{ HTML::script('js/bootstrap.min.js') }}
        {{ HTML::script('js/holder.js') }}
        {{ HTML::script('js/jquery.simplePagination.js') }}
        {{ HTML::script('js/bootbox.min.js') }}
        <script>
            (function($) {
                $(document).on("click", ".confirm", function(e) {
                    e.preventDefault();

                    var location = $(this).attr('href');

                    bootbox.confirm("¿Esta seguro?", function(confirm) {
                        if (confirm) {
                            window.location.replace(location);
                        }
                    });
                });
            })(jQuery);
        </script>
        <script type="text/javascript">
            $('.tip').tooltip();
        </script>
    @show
</body>
</html>
```

### Invitado

```
<!doctype html>
<html lang="es">
<head>
    <meta charset="UTF-8">
    <title>@yield('title', 'SIAD')</title>
    <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-scalable=no">
    @section('css')
        {{ HTML::style('css/bootstrap.min.css') }}
    @show
</head>
<body>
    <div class="container">
        <div class="row">
            <div class="col-xs-12">
                <div class="page-header">
                    <h1>SIAD</h1>
                    <h2><small>Sistema Interactivo de Aprendizaje a Distancia</small></h2>
                </div>
            </div>
        </div>
        <div class="row">
```

```
      <div class="col-xs-12">
        @foreach (Session::get('messages', array()) as $message)
          <div class="alert alert-{{ $message['type'] }}">
            <button type="button" class="close" data-dismiss="alert" aria-hidden="true">&times;</button>
            {{ $message['text'] }}
          </div>
          {{ Session::forget('messages') }}
        @endforeach
        @yield('content')
      </div>
    </div>
    <div class="row">
      <div class="col-xs-12">
        @include('includes.debug')
      </div>
    </div>
  </div>
  @section('js')
    {{ HTML::script('js/jquery.min.js') }}
    {{ HTML::script('js/bootstrap.min.js') }}
    {{ HTML::script('js/holder.js') }}
    {{ HTML::script('js/bootbox.min.js') }}
    <script>
      (function($) {
        $(document).on("click", ".confirm", function(e) {
          e.preventDefault();

          var location = $(this).attr('href');

          bootbox.confirm("¿Esta seguro?", function(confirm) {
            if (confirm) {
              window.location.replace(location);
            }
          });
        });
      })(jQuery);
    </script>
  @show
</body>
</html>
```

## Preguntas

### Añadir

```
@section('content')
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h3 class="panel-title">
                Agregar Pregunta
                @if ('open' === $type)
                    abierta
                @endif
                @if ('choice' === $type)
                    de seleccion
                @endif
                @if ('multiple' === $type)
                    multiple
                @endif
            </h3>
        </div>
        <form role="form" class="form-horizontal" action="{{ route('question.create', $test->test_id) }}" method="POST">
        <div class="panel-body">
            <input type="hidden" name="type" value="{{ $type }}">
            @include('includes.field', array('name' => 'question_text', 'label' => 'Texto', 'type' => 'text'))
            @include('includes.field', array('name' => 'points', 'label' => 'Puntos', 'type' => 'number'))
            @if ('multiple' === $type)
                <p><b>Cada respuesta valdra los mismos puntos</b></p>
            @endif
            <table class="table table-striped table-bordered table-hover table-condensed">
                <thead>
                    <tr>
                        <th>Respuesta</th>
                        <th>Es correcta</th>
                        <th>
                            <button type="button" class="btn btn-primary btn-xs" id="add-question">
```

```
                                <span class="glyphicon glyphicon-plus"></span> Agregar
                            </button>
                        </th>
                    </tr>
                </thead>
                <tbody id="answers">
                    @if (($answerText = Input::old('answer_text')) && ($isCorrect = Input::old('is_correct')))
                        @for ($i = 0; $i < count($answerText); $i++)
                            <tr>
                                <td>
                                    <input type="text" name="answer_text[]" value="{{ $answerText[$i] }}" class="form-control"/>
                                </td>
                                <td>
                                    @if (1 == $isCorrect[$i])
                                        <input type="checkbox" class="form-control checkable" checked/>
                                    @else
                                        <input type="checkbox" class="form-control checkable"/>
                                    @endif
                                    <input type="hidden" name="is_correct[]" value="{{ $isCorrect[$i] }}"/>
                                </td>
                                <td>
                                    <button type="button" class="btn btn-default btn-xs removable"><span class="glyphicon glyphicon-remove"></span> Eliminar</button>
                                </td>
                            </tr>
                        @endfor
                    @endif
                </tbody>
            </table>
        </div>
        <div class="panel-footer">
            <button type="submit" class="btn btn-primary btn-lg btn-block">Terminar</button>
            <a class="btn btn-default btn-lg btn-block" href="{{ route('test.manager', $test->test_id) }}">Cancelar</a>
        </div>
        </form>
    </div>
@stop

@section('js')
    @parent
    <script>
        (function($) {
            $('#answers').on('click', '.removable', function() {
                $(this).closest('tr').remove();
            });
            $('#answers').on('click', '.checkable', function() {
                var val = $(this).next().val();

                val = 0 == val ? 1 : 0;

                $(this).next().val(val);
            });
            $('#add-question').bind('click', function(event) {
                $('#answers').append("<tr><td><input type=\"text\" name=\"answer_text[]\" class=\"form-control\"/></td><td><input type=\"checkbox\" class=\"form-control checkable\"/><input type=\"hidden\" name=\"is_correct[]\" value=\"0\"/></td><td><button type=\"button\" class=\"btn btn-default btn-xs removable\"><span class=\"glyphicon glyphicon-remove\"></span> Eliminar</button></td></tr>");
            });
        })(jQuery);
    </script>
@stop
```

### Editar

```
@section('content')
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h3 class="panel-title">
                Editar Pregunta
                @if ('open' === $question->type)
                    abierta
                @endif
                @if ('choice' === $question->type)
                    de seleccion
                @endif
                @if ('multiple' === $question->type)
                    multiple
                @endif
            </h3>
        </div>
        {{ Form::model($question, array('route' => array('question.update', $test->test_id, $question->question_id))) }}
        <div class="panel-body">
            @include('includes.field', array('name' => 'question_text', 'label' => 'Texto', 'type' => 'text'))
            @include('includes.field', array('name' => 'points', 'label' => 'Puntos', 'type' => 'number'))
            @if ('multiple' === $question->type)
                <p><b>Cada respuesta valdra los mismos puntos</b></p>
            @endif
            <table class="table table-striped table-bordered table-hover table-condensed">
                <thead>
                    <tr>
                        <th>Respuesta</th>
                        <th>Es correcta</th>
                        <th>
                            <button type="button" class="btn btn-primary btn-xs" id="add-question">
                                <span class="glyphicon glyphicon-plus"></span> Agregar
                            </button>
                        </th>
                    </tr>
                </thead>
                <tbody id="answers">
                    @foreach ($question->answers as
$answer)
                        <tr>
                            <td>
                                <input type="text"
name="answer_text[]" value="{{ $answer->answer_text }}"
class="form-control"/>
                            </td>
                            <td>
                                @if ($answer->is_correct)
                                    <input
type="checkbox" class="form-control checkable" checked/>
                                    <input type="hidden"
name="is_correct[]" value="1"/>
                                @else
                                    <input
type="checkbox" class="form-control checkable"/>
                                    <input type="hidden"
name="is_correct[]" value="0"/>
                                @endif
                            </td>
                            <td>
                                <button type="button"
class="btn btn-default btn-xs removable"><span
class="glyphicon glyphicon-remove"></span>
Eliminar</button>
                            </td>
                        </tr>
                    @endforeach
                </tbody>
            </table>
        </div>
        <div class="panel-footer">
            <button type="submit" class="btn btn-primary
btn-lg btn-block">Guardar</button>
            <a class="btn btn-default btn-lg btn-block"
href="{{ route('test.manager', $test->test_id)
}}">Cancelar</a>
        </div>
        {{ Form::close() }}
    </div>
@stop

@section('js')
    @parent
    <script>
        (function($) {
            $('#answers').on('click', '.removable', function() {
                $(this).closest('tr').remove();
            });
            $('#answers').on('click', '.checkable', function() {
                var val = $(this).next().val();

                val = 0 == val ? 1 : 0;

                $(this).next().val(val);
            });
            $('#add-question').bind('click', function(event) {
                $('#answers').append("<tr><td><input type=\"text\" name=\"answer_text[]\" class=\"form-control\"/></td><td><input type=\"checkbox\" class=\"form-control checkable\"/><input type=\"hidden\" name=\"is_correct[]\" value=\"0\"/></td><td><button type=\"button\" class=\"btn btn-default btn-xs removable\"><span class=\"glyphicon glyphicon-remove\"></span> Eliminar</button></td></tr>");
            });
        })(jQuery);
    </script>
@stop
```

## Recursos

### Añadir

```
@section('content')
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h3 class="panel-title">
                Agregar
                @if ($type === 'video')
                    Video <span class="glyphicon glyphicon-play"></span>
                @endif
                @if ($type === 'file')
                    Archivo <span class="glyphicon glyphicon-file"></span>
                @endif
                @if ($type === 'url')
                    enlace <span class="glyphicon glyphicon-link"></span>
                @endif
            </h3>
        </div>
        <form role="form" class="form-horizontal" enctype="multipart/form-data" action="{{ route('resource.create', $course->course_id) }}" method="POST">
        <div class="panel-body">
            <input type="hidden" name="type" value="{{ $type }}">
            @include('includes.field', array('name' => 'title', 'label' => 'Titulo', 'type' => 'text'))
            @include('includes.field', array('name' => 'description', 'label' => 'Descripcion', 'type' => 'textarea'))
            @if ($type === 'video')
                @include('includes.field', array('name' => 'url', 'label' => 'Enlace de youtube', 'type' => 'text'))
            @endif
            @if ($type === 'file')
                @include('includes.field', array('name' => 'upload', 'label' => 'Archivo', 'type' => 'file'))
                <span>
                    <b>Limite: 10mb</b>
                </span>
            @endif
            @if ($type === 'url')
                @include('includes.field', array('name' => 'url', 'label' => 'Enlace', 'type' => 'text'))
            @endif
        </div>
        <div class="panel-footer">
            <button type="submit" class="btn btn-primary btn-lg btn-block">Agregar</button>
            <a class="btn btn-default btn-lg btn-block" href="{{ route('course.manager', $course->course_id) }}">Cancelar</a>
        </div>
        </form>
    </div>
@stop
```

### Editar

```
@section('content')
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h3 class="panel-title">
                Editar
                @if ($resource->type === 'video')
                    Video <span class="glyphicon glyphicon-play"></span>
                @endif
                @if ($resource->type === 'file')
                    Archivo <span class="glyphicon glyphicon-file"></span>
                @endif
                @if ($resource->type === 'url')
                    Enlace <span class="glyphicon glyphicon-link"></span>
                @endif
            </h3>
        </div>
        {{ Form::model($resource, array('route' =>
array('resource.update', $resource->resource_id), 'files' => true)) }}
        <div class="panel-body">
            @include('includes.field', array('name' => 'title', 'label' => 'Titulo', 'type' => 'text'))
            @include('includes.field', array('name' => 'description', 'label' => 'Descripcion', 'type' => 'textarea'))
            @if ($resource->type === 'video')
                @include('includes.field', array('name' => 'attached', 'label' => 'Enlace de youtube', 'type' => 'text'))
            @endif
            @if ($resource->type === 'file')
                @include('includes.field', array('name' => 'upload', 'label' => 'Archivo', 'type' => 'file'))
                <p>Limite: 10mb</p>
                <p>{{ $resource->attached }}</p>
            @endif
            @if ($resource->type === 'url')
                @include('includes.field', array('name' => 'attached', 'label' => 'Enlace', 'type' => 'text'))
            @endif
        </div>
        <div class="panel-footer">
            <button type="submit" class="btn btn-primary btn-lg btn-block">Guardar</button>
            <a class="btn btn-default btn-lg btn-block" href="{{ route('course.manager', $course->course_id) }}">Cancelar</a>
        </div>
        {{ Form::close() }}
    </div>
@stop
```

Vista

```
@section('content')
    <div class="panel panel-default">
        <div class="panel-heading">
            <h3 class="panel-title">{{ $resource->title }}</h3>
        </div>
        <div class="panel-body">
            <p class="lead text-muted">{{ $resource->description }}</p>
            @if ($resource->is('file'))
                @if ($resource->toFile->is_pdf)
                    <embed src="{{ URL::to('uploads/resources/' . $resource->resource_id) }}" class="col-xs-12" height="500" alt="pdf" pluginspage="http://www.adobe.com/products/acrobat/readstep2.html">
                @endif
                <div class="text-center">
                    <a class="btn btn-primary btn-lg" href="{{ route('resource.download', $resource->resource_id) }}">Descargar</a>
                </div>
            @endif
            @if ($resource->is('video'))
                <iframe type="text/html" class="col-xs-12" height="385" src="http://www.youtube.com/embed/{{ $resource->toVideo->video_id }}" frameborder="0">
                </iframe>
```

```
      @endif

      @if ($resource->is('url'))

        <div class="text-center">

          <a class="btn btn-primary btn-lg" href="{{ $resource->toUrl->url }}">Ir</a>
        </div>

      @endif

    </div>

  </div>

@stop
```

### Principal

```
@section('content')
  <div class="jumbotron">
    <h1 class="text-center">¡Bienvenido a SIAD!</h1>
    <p>
      <a class="tip" href="#" data-toggle="tooltip" title="" data-original-title="Sistema Interactivo de Aprendizaje a Distancia">SIAD</a>
      es el sistema de Aprendizaje a distancia creado por y para la comunidad de estudiantes de la <a class="tip" href="http://uach.mx" target="_blank" data-toggle="tooltip" title="" data-original-title="Universidad Autonoma de Chihuahua">UACH</a>.
    </p>
  </div>
  <div class="panel panel-primary">
    <div class="panel-heading">
      <h3 class="panel-title">Ultimos 10 cursos</h3>
    </div>
    <table class="table table-striped table-bordered table-hover table-condensed">
```

```
            <thead>
                <tr>
                    <th>Curso</th>
                    <th>Profesor</th>
                    <th></th>
                </tr>
            </thead>
            <tbody>
                @foreach ($courses as $course)
                    <tr>
                        <td>
                            {{ $course->course_name }}
                        </td>
                        <td><b>{{ $course->teacher->full_name }}</b>
                            ({{ $course->teacher->email }})</td>
                        <td>
                            <a class="btn btn-primary btn-xs" href="{{ route('course.register', $course->course_id) }}">
                                @if ($course->password)
                                    <span class="glyphicon glyphicon-lock"></span>
                                @endif
                                Matricularme
                            </a>
                        </td>
                    </tr>
                @endforeach
            </tbody>
        </table>
    </div>
@stop
```

### Kardex

```
@section('content')
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h1 class="panel-title">Kardex</h1>
        </div>
        <div class="panel-body">
            <dl class="dl-horizontal">
                <dt>Nombre:</dt>
                <dd>{{ $user->first_name }}</dd>
                <dt>Apellido:</dt>
                <dd>{{ $user->last_name }}</dd>
                <dt>Nombre de usuario:</dt>
                <dd>{{ $user->user_name }}</dd>
                <dt>Correo:</dt>
                <dd>{{ $user->email }}</dd>
            </dl>
            <table class="table table-striped table-hover table-condensed table-bordered">
                <thead>
                    <tr>
                        <th>Nombre del curso</th>
                        <th>Estado</th>
                        <th>Calificación</th>
                    </tr>
                </thead>
                <tbody>
                    @foreach ($user->kardex as $kardex)
                        <tr>
                            <td>{{ $kardex->course_name }}</td>
                            <td>{{ $kardex->status }}</td>
                            <td>{{ $kardex->qualification }}</td>
                        </tr>
                    @endforeach
                </tbody>
            </table>
        </div>
        <div class="panel-footer">
        </div>
```

```
    </div>
@stop
```

## Conclusión

Nuestro sistema o “plataforma” es un excelente método de estudio online para aquellas personas que deseen compartir de sus conocimientos además de que gracias a este tipo de sistemas será más fácil dar a conocer todos los conocimientos que tenemos y nos gustaría compartirlos es muy importante no confundirlo con una escuela es simplemente una serie de cursos en los cuales los alumnos pueden ser tanto maestros como alumnos.

## Bibliografía

Modelo vista controlador

- http://es.wikipedia.org/wiki/Modelo_Vista_Controlador

Laravel

- http://es.wikipedia.org/wiki/Laravel

HTTP

- http://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol