Universidad de La Laguna

# Proyecto PAU

## Sistemas y Tecnologías Web

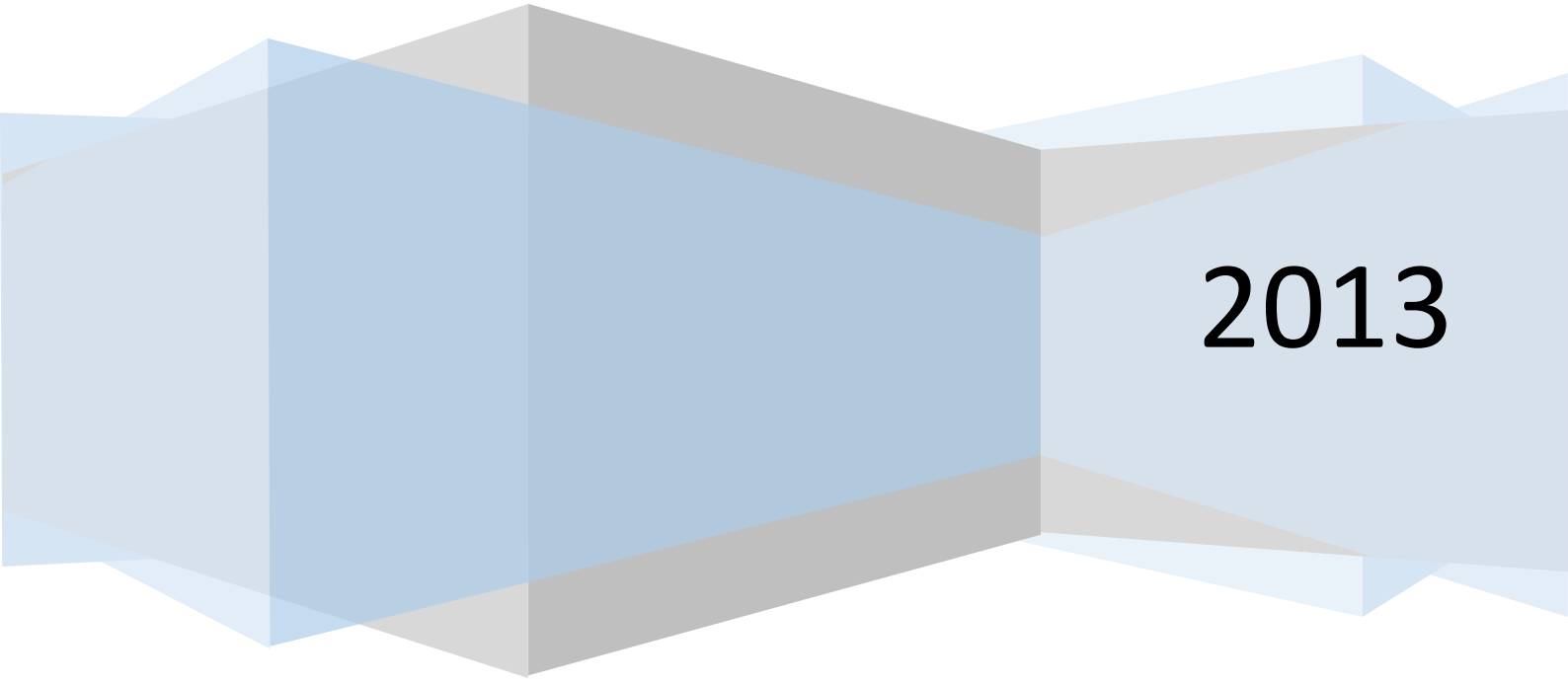
Rodrigo Carpintero Díez

Uriel Sánchez Guindulain

Sergio Jesús García Llanera

Juan José Labrador González

Domingo Yeray Rodríguez Martín



2013

# Índice

# 0. Información básica

- ***Github***
  https://github.com/urisan23/proyectostw

- ***Enlace***
  https://obscure-savannah-8776.herokuapp.com

- ***Administrador***
  Uriel Sánchez Guindulain

- ***Equipo de desarrollo***
  Rodrigo Carpintero Díez
  Sergio Jesús García Llanera
  Juan José Labrador González
  Domingo Yeray Rodríguez Martín

# 1. Descripción general:

La página web desarrollada se trata de un portal para universitarios de la Escuela Técnica Superior de Ingeniería Informática en el que pueden compartir archivos, comentar acerca de las asignaturas y comunicarse con otros usuarios a través de mensajería privada. Dada su estructura, el crecimiento de la plataforma puede llevarse a cabo de manera sencilla y de esta manera aumentar su ámbito a las demás titulaciones de la ULL.

La web tiene 3 módulos principales que son: mensajes, asignaturas y archivos; además de una página de perfil donde poder ver toda tu información:

En la parte de mensajes, se pueden visualizar los mensajes privados enviados y recibidos, organizados por conversaciones, desde el más antiguo al más reciente. Para iniciar una nueva conversación hay que mandar un mensaje desde la página de perfil del usuario mientras que si la conversación ya existe se puede responder en la página de mensajes.

En la parte de asignaturas encontramos dos páginas, una que muestra el listado de asignatura, donde puedes matricularte y desmatricularte, y por otra parte una página para cada asignatura, que contiene el listado de archivos y los comentarios sobre la asignatura.

En la parte de archivos tenemos la página donde vemos la vista previa del archivo (si está disponible) y podemos descargar el archivo en cuestión.

# 2. Descripción de la Web

*PAU.RB*

***Descripción general.*** Gestiona las acciones básicas de la web, como son el login, el logout, el registro, la activación de la cuenta o el cambio de password. También incluye las rutas de las páginas del perfil propio y del resto de usuarios, así como para la ruta de la página de resultados de la búsqueda de usuarios y los apartados de ayuda y contacto.

- **'/':** ruta raíz. Te redirige a '/login' si no estás logueado y a '/profile' si ya estás logueado.
- **'/login':** el método get comprueba si el usuario está logueado, si es así redirige a la página del perfil, si no carga la vista con el formulario de login. El método Post comprueba que el usuario existe, ya sea su nombre de usuario o su email, y compara la contraseña introducida con la almacenada. Si coinciden, se realiza el login activando las cookies y se redirige al usuario a su perfil.
- **'/signup':** el método get recopila todos los nombres de usuarios e email registrados en la plataforma, y posteriormente carga el formulario de registro. Si se introduce un email o nombre de usuario en uso, no se llevará a cabo el registro. Introducidos todos los datos satisfactoriamente, el método post se encargará de guardar todos los datos y se enviará un email a la dirección de correo introducida con un enlace de activación de cuenta. Posteriormente se nos redirige a '/login'.
- **'/activeaccount':** se encarga de activar la cuenta del usuario y lo redirecciona a su perfil dentro de la plataforma.
- **'/logout':** cierra la sesión del usuario y redirecciona a la página principal de la plataforma.
- **'/forgotten_pass':** el método get carga todos los emails registrados en la plataforma y muestra un formulario para introducir el email de la cuenta cuya contraseña se ha olvidado. El método post se encarga de buscar ese usuario, reestablecer su contraseña con una dada por defecto y mandar un email notificando el cambio. Acto seguido, redirecciona de nuevo a '/login'.
- **'/profile':** muestra el perfil del usuario logueado, con sus datos, asignaturas matriculadas, mensajes, etc.
- **'/edit_profile':** permite modificar los datos del usuario, a excepción de su nombre de usuario y su email. También es posible añadir una foto de perfil.
- **'/change_pass':** permite cambiar la contraseña actual al usuario por una nueva.
- **'/search':** permite realizar búsqueda de usuarios mediante nombre de usuario o nombre y/o apellido.
- **'/user/:id':** permite mostrar la página de perfil de cualquier otro usuario de la plataforma.
- **'/help':** muestra el apartado de ayuda y el FAQs.
- **'/contact':** muestra información sobre los desarrolladores de la plataforma, así como la manera para ponerse en contacto con ellos.

## ADMIN.RB

***Descripción general.*** Este archivo contiene todos los métodos para la gestión de usuarios, asignaturas, comentarios, archivos y de la base de datos por parte de las personas autorizadas.

- **'/admin':** el método get comprueba si el usuario que accede al panel de login de administración es un administrador. Si es así, se le redirecciona a una página para que ingrese su contraseña. Por el contrario, si no es un administrador, aparecerá un error al usuario indicando que no puede acceder al contenido.El método post comprueba que la contraseña de acceso al panel de administración sea correcta. En ese caso, se redirecciona al panel principal de administración. En otro caso, aparecerá un mensaje indicando que la contraseña es inválida.
- **'/admin/panel':** para evitar el acceso no deseado de usuario a esta ruta, se comprueba que el usuario que accede es un administrador y se carga la página de administración de usuarios, asignaturas, archivos, y se muestra la base de datos. En otro caso, si no es un administrador, aparecerá un error al usuario indicando que no puede acceder al contenido.
- **'/admin/users':** para evitar el acceso no deseado de usuario a esta ruta, se comprueba que el usuario que accede es un administrador y se carga la página de administración de usuarios. En otro caso, si no es un administrador, aparecerá un error al usuario indicando que no puede acceder al contenido.
- **'/admin/users/edit/:id':** carga la vista con el formulario de los campos a modificar del usuario encontrado con el id pasado en la ruta.
- **'/admin/users/admin_edit_user/:id':** modifica los campos recibidos del formulario, guarda los cambios y redirecciona de nuevo a la página principal de administración de usuarios.
- **'/admin/users/delete_user/:id':** borra al usuario cuyo id es pasado en la ruta, además de sus asignaturas matriculadas y mensajes. Luego redirecciona de nuevo a la página principal de administración de usuarios.
- **'/admin/users/delete_user_subject/:id/:sub':** permite desmatricular a un usuario de una asignatura. Se busca a partir del id pasado en la ruta y se elimina la relación que tiene con la asignatura cuyo id es :sub. Luego se redirecciona a la página de edición de ese usuario de nuevo.
- **'/admin/subjects':** para evitar el acceso no deseado de usuario a esta ruta, se comprueba que el usuario que accede es un administrador y se carga la página de administración de asignaturas, ordenadas por curso. En otro caso, si no es un administrador, aparecerá un error al usuario indicando que no puede acceder al contenido.
- **'/admin/subjects/add_subject':** el método get carga el formulario para añadir una asignatura. El método post comprueba que la asignatura tiene asignado un curso correcto, se crea y se guardan sus campos. Por último, se redirecciona de nuevo a la página de asignaturas.
- **'/admin/subjects/delete_subject/:id':** borra la asignatura cuyo id es pasado en la ruta, además de los archivos que contenga y los comentarios. Luego redirecciona de nuevo a la página principal de administración de asignaturas.

- **‘/admin/subjects/edit_comment’:** se carga un formulario con el comentario seleccionado y se permite modificarlo o, en caso de dejar vacío, borrarlo. Luego redirecciona de nuevo a la página de administración de esa asignatura.
- **‘/admin/subjects/edit_subject/:id’:** el método get carga la vista con el formulario de los campos a modificar de la asignatura encontrada con el id pasado en la ruta. El método post modifica los campos recibidos del formulario, guarda los cambios y redirecciona de nuevo a la página principal de administración de asignaturas.
- **‘/admin/files’:** para evitar el acceso no deseado de usuario a esta ruta, se comprueba que el usuario que accede es un administrador y se carga la página de administración de archivos, ordenadas por asignatura. En otro caso, si no es un administrador, aparecerá un error al usuario indicando que no puede acceder al contenido.
- **‘/admin/files/delete/:id’:** borra el archivo cuyo id es pasado en la ruta. Luego redirecciona de nuevo a la página principal de administración de archivos.
- **‘/admin/files/delete_all’:** borra todos los archivos de todas las asignaturas. Luego redirecciona de nuevo a la página principal de administración de archivos.

### BBDD.RB

***Descripción general.*** Este archivo contiene los métodos que inicializa la base de datos y las tablas que contiene. Además existen los métodos relacionados con cambios en la base de datos, ya sea actualizarla o destruir tablas enteras. El acceso a estas páginas queda restringido únicamente a aquellos usuarios que tengan privilegios de administrador.

- **‘/bbdd/populate’:** Este es el método más importante ya que es el que se encarga de inicializar la base de datos y almacenar los elementos básicos para el correcto funcionamiento de la web. Elimina todo el contenido de la base de datos y a continuación crea unos usuarios con el atributo de administradores. Del mismo modo se crea la tabla de asignaturas y se puebla con las asignaturas de los 4 cursos.
- **‘/bbdd/show_all’:** Este método solo tiene uso para, durante el desarrollo del proyecto, poder ver en todo momento cual es el estado de la base de datos, con la finalidad de detectar errores en el código.
- **‘/bbdd/destroy_users’:** Este método se encarga de eliminar de la base de datos a todos los usuarios registrados.
- **‘/bbdd/destroy_subjects’:** Este método se encarga de eliminar de la base de datos todas las asignaturas que existen.
- **‘/bbdd/destroy_files’:** Del mismo modo que “destroy_users”, este método se encarga de eliminar de la base de datos todos los archivos que se han subido.

### ERROR.RB

***Descripción general:*** En este archivo se establecen los métodos que se ejecutarán cuando se dé algún fallo en el procesamiento de algunas de las solicitudes que llegan al servidor. Básicamente se tratan de redirecciones a páginas estáticas en html que muestran un mensaje de error y el motivo por el que se ha producido. Los errores que se manejan son el 403 (acceso denegado), 404 (página no encontrada) y los errores internos del servidor que van desde el 500 al 510.

### FILES.RB

***Descripción general:*** En este archivo se encuentran todos los métodos necesarios para procesar las solicitudes relacionadas con los archivos: subir, descargar, visualizar y puntuar.

- **'/upload':** el método post se encarga de recibir y procesar el archivo para subirlo al servidor. Para ello se genera un archivo temporal y a continuación se establece una conexión sftp usando la gema Net::Sftp. Se realiza el envío al servidor y a continuación se almacenan los datos del nuevo archivo en la base de datos.
- **'/download/:id':** Con este get se procesa la solicitud de un archivo. Con el parámetro que se le pasa a través de la url se busca la información del archivo en la base de datos. A continuación se establece una conexión con el servidor de ficheros y se almacena el fichero en un archivo temporal. Por último se averigua cual es el tipo de archivo gracias a la ayuda de la función "filetype" y se hace el envío al cliente con el método send_file.
- **'/file/:s/:id':** Con el método get se carga la vista de un archivo en concreto. En primer lugar se cuentan los votos que tiene ese archivo, ya que serán usados en la vista para calcular la puntuación que tiene el archivo. A continuación se carga la vista usando los parámetros recibidos en la uri de la petición.
- **'/file/vote/:s/:id/:stars':** Con el método get se procesa el voto que realiza el usuario sobre un archivo determinado. En primer lugar se busca el fichero y se comprueba si ese usuario ya ha votado con anterioridad. Si es así quiere decir que se realiza un cambio de puntuación y no una nueva puntuación. En el caso de una nueva puntuación se crea un voto y se añade a la lista de votos que tiene asociada el archivo, mientras que si se trata de una actualización simplemente de almacena la nueva puntuación en el voto del usuario.

### MESSAGES.RB

***Descripción general:*** este archivo contiene los métodos necesarios para acceder al buzón de correos para leer mensajes, enviarlos y borrarlos.

- **'/inbox':** muestra el buzón de mensajes recibidos del usuario, con el nombre del emisor y la fecha de envío. Además aparece la opción de borrar cada mensaje.

- **‘/send_message/:id’:** permite enviar un mensaje al usuario cuyo id es pasado en la ruta (:id). Se crean dos copias del mensaje, una para el receptor y otra se almacena en el emisor. De este modo, se puede ver toda la conversación de mensajes entre ambos. Si se envía un mensaje desde la página de perfil de ese usuario, se nos vuelve a redirigir a ella. En caso de que contestemos un mensaje recibido desde nuestro buzón, se nos redirige de nuevo al buzón.
- **‘/delete_message/:id’:** borra el mensaje cuyo identificador es pasado por la ruta (:id). A continuación se nos redirige de nuevo al buzón de mensajes.

#### SUBJECTS.RB

Descripción general: este archivo contiene todos los métodos relacionados con las asignaturas de la titulación: mostrar listado completo, matricularse, desmatricularse y vista de cada asignatura.

- **‘/subjects’:** permite mostrar el listado completo de asignaturas, ordenadas por curso, y da la posibilidad de matricularse/desmatricularse de las que se desee.
- **‘/register/:sub’:** permite matricular al usuario logueado en la asignatura cuyo id es pasado en la ruta (:sub). Posteriormente, se redirige de nuevo a la página de asignaturas.
- **‘/unregister/:sub’:** permite desmatricular al usuario logueado de la asignatura cuyo id es pasado en la ruta (:sub), eliminando la relación que existe entre ambos. Posteriormente, se redirige de nuevo a la página de asignaturas.
- **‘/subjects/:idsub’:** el método get permite mostrar la página de la asignatura cuyo id es pasado en la ruta (:idsub). Se muestra el nombre, el curso al que pertenece, los archivos que se han subido con sus puntuaciones correspondientes y los comentarios realizados. Además permite la opción de subir archivos. El método post recibe el texto del formulario de comentarios y lo añade a la página de esa asignatura junto con el usuario que lo realizó y la fecha de creación.

# 3. Historia de Desarrollo

Incluimos en este informe un resumen de la distribución del trabajo que hemos realizado a lo largo de todo el desarrollo. Las fechas son aproximadas y se basan en el historial de actualizaciones del proyecto almacenado en github.

- **01/10/2012 - PRIMERA REUNIÓN DEL PROYECTO**
  De entre varias ideas, elegimos finalmente como proyecto una página web de ámbito universitario (en principio) en el que los estudiantes puedan comunicarse y compartir archivos, exámenes y material de estudio de interés. Planteamos como características más atractivas la compartición de archivos y opciones de mensajería y comunicación. Tomamos las primeras decisiones sobre el diseño. El nombre provisional de la web es PAU (Plataforma Académica Universitaria). Planificamos y repartimos el trabajo sobre el inicio de la página, el sistema de login y signup y la creación del repositorio en github.

- **03/10/2012** - Apertura del repositorio en github. Creación de proyecto inicial.
- **10/10/2012** - Configuración e implementación de la base de datos con la gema datamapper. Añadido signup y primer listado de usuarios. Login implementado y funcionando. Empezamos con el diseño básico, creamos el primer layaout, header y footer. Elegimos como hoja de estilo el .css de twitter Bootstrap.

- **17/10/2012 - SEGUNDA REUNIÓN DEL PROYECTO**
  Acordamos usar Amazon como servidor de almacenamiento de archivos, donde poder almacenar los archivos y documentos que los usuarios suban a la web. Creamos la cuenta de Amazon y acordamos implementar la API mediante una gema de Ruby. También acordamos implementar la gema Pony para el envío de emails a los usuarios.
- **23/10/2012** - Conseguimos implementar Pony con éxito y creamos los primeros emails, uno de bienvenida y otro con la opción de generar una contraseña nueva. No tenemos la misma suerte con la gema de Amazon, que da varios problemas y resulta ser más difícil de implementar de lo que a primera vista creíamos. Añadimos la primera vista del perfil, con opciones de edición incluidas.

- **26/10/2012 - TERCERA REUNIÓN DEL PROYECTO**
  Acordamos abandonar la idea de la gema de Amazon, paralizamos el tema de la subida de archivos y nos centramos en la siguiente etapa del proyecto, añadir la categoría de asignaturas junto con las opciones de matriculación asociadas. Aunque inicialmente pensamos en almacenar las imágenes del usuario en el servidor, decidimos que es mejor que el usuario pueda usar imágenes de cualquier sitio web. Nuevamente planificamos el trabajo y marcamos lo siguiente a desarrollar. También elegimos el servidor en el que desplegaremos la página, Heroku.
- **08/11/2012** - Llevamos a cabo con éxito el despliegue de la web en Heroku, con un primera versión funcional de la página.
- **10/11/2012** - Añadidas las asignaturas y las opciones de matriculación, así como las relaciones en la base de datos. Añadida también la imagen de perfil de cada usuario, usando la API que proporciona Gravatar. Hacemos vistas para listar las asignaturas y las matriculadas dentro del perfil.

- **14/11/2012 - CUARTA REUNIÓN DEL PROYECTO**
  Completada la etapa de las asignaturas, iniciamos la siguiente iteración en la que añadiremos a la web el módulo de mensajería. Acordamos implementar un buscador de usuarios en la web, junto a otras modificaciones para facilitar el envío de mensajes. Tomamos también varias decisiones en cuanto al estilo de la web.
- **17/11/2012** - Implementamos las opciones de mensajería de la web.
- **20/11/2012** - Añadido el formulario de búsqueda en el footer de la web. Llevamos a cabo los cambios y mejoras en el diseño de la web acordados en la reunión. Agregamos una presentación de imágenes a la página principal.

- **26/11/2012 - QUINTA REUNIÓN DEL PROYECTO**
  Retomamos el tema de la subida de archivos, esta vez decidimos utilizar la herramienta proporcionada por Dropbox. Decidimos implementar también un tipo de usuario especial (administrador) que pueda modificar directamente la base de datos, con opción para edición de asignaturas y usuarios, entre otras.
- **30/11/2012** - Nuevamente encontramos dificultades a la hora de implementar la gema de Dropbox, por problemas con la interfaz y con la autentificación de cada usuario. Decidimos consultar el asunto con el profesor para buscar una solución.
- **09/12/2012** - Implementado el admin junto con los usuarios que pueden acceder a él y con las opciones de modificación de la base de datos y las vistas asociadas.

- **10/12/2012 - SEXTA REUNIÓN DEL PROYECTO**
  Tras hablar con el profesor, conseguimos acceso a un servidor de la ETSII en el que poder almacenar y recuperar los archivos mediante sftp. Tomamos las decisiones oportunas y acordamos de qué manera lo implementaremos. También decidimos, tras la experiencia de algunos de nosotros el año pasado en la asignatura de DSI, incluir un módulo de tests en el proyecto.
- **14/12/2012** - Terminamos la implementación de las opciones sftp para subir y descargar archivos. Añadimos las opciones de subida y descarga dentro de cada asignatura.
- **15/12/2012** - Añadimos al proyecto la gema rspec para llevar a cabo las pruebas y realizamos el primer test. Realizamos mejoras en la vista de mensajes. Además añadimos la validación de cuenta a través de email al registrarse en la web.

- **20/12/2012 - SÉPTIMA REUNIÓN DEL PROYECTO**
  Iniciamos la última etapa del proyecto, en la que añadiremos un sistema de puntuaciones a los archivos, además de un apartado de comentarios por cada asignatura. Decidimos añadir también páginas para cada error, además de un apartado de ayuda para el uso de la web. Tomamos las últimas decisiones sobre el diseño.
- **26/12/2012** - Añadidos sistemas de puntuaciones y páginas de error. Añadimos también el apartado de ayuda.
- **02/01/2013** - Añadimos los comentarios a las asignaturas, con las vistas correspondientes.
- **08/01/2013** - Testeo de la web y corrección de errores. Añadimos los últimos tests. Incluimos también opción para navegación desde móviles.

- **11/01/2013 - PRESENTACIÓN DEL PROYECTO EN CLASE.**

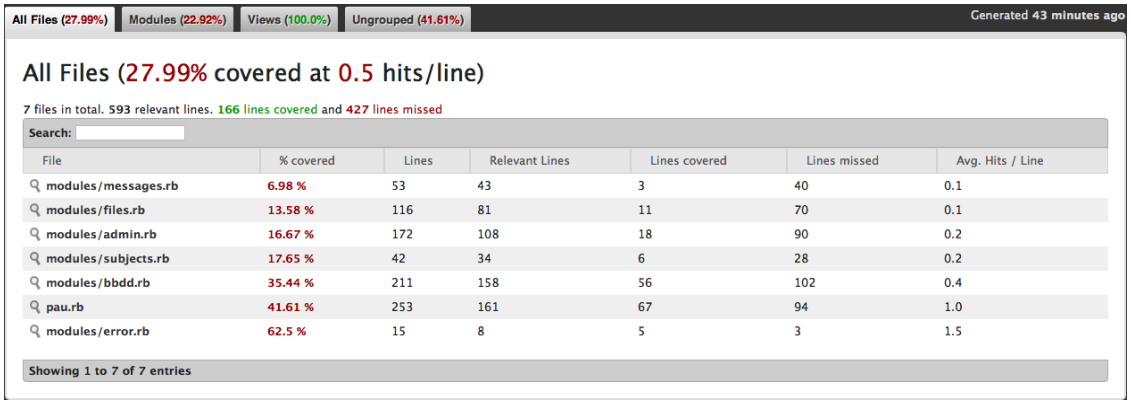
- **15/01/2013 - ENTREGA DEL INFORME.**

# 4. Lenguajes de programación empleados

- ***Desarrollado*** en el framework Sinatra de Ruby.
- ***Gemas usadas:***
  - *dm-postgres-adapter:* adaptador de postgreSQL para DataMapper
  - *pg:* interfaz de postgreSQL para Ruby
  - *dm-sqlite-adapter:* adaptador de sqlite para DataMapper
  - *data_mapper:* permite crear objetos relacionales.
  - *sinatra:* framework de desarrollo de la aplicación.
  - *haml*: lenguaje de marcado que simplifica la escritura de HTML.
  - *pony:* permite el envío de emails.
  - *heroku:* permite realizar las gestión de la aplicación en el servidor web de Heroku.
  - *net-sftp:* permite realizar conexiones sftp para el envío y recepción de archivos.
  - *rspec:* permite realizar los test de la aplicación.
  - *simplecov:* permite realizar el coverage de los tests y mostrar los resultados en forma de tabla.
  - *rack-test:* API para realizar test sobre aplicaciones que usen Rack como middleware.
  - *webrat:* permite crear test para aplicaciones Ruby.
  - *factory_girl:* proporciona un framework y DSL para facilitar cómo se desarrollan las pruebas y cuáles son los modelos que se van a probar.
- ***Tests***

Como se indica en la lista de gemas, RSpec ha sido la herramienta principal para realizar los tests. Es una herramienta basada en BDD (Desarrollo Guiado por Comportamiento). Tras la experiencia previa en otra asignatura con RSpec, decidimos que era la herramienta apropiada para probar el código. Es una herramienta muy versátil, ya que aunque anteriormente se trabajó con Rails, no ha dado problemas con Sinatra.

Se han incluido gemas adicionales para corregir errores surgidos inicialmente con las pruebas, todas ellas indicadas arriba. Además, se usa la gema 'simplecov' para obtener un coverage e indicar qué partes del código están cubiertas por las pruebas. No se ha obtenido un porcentaje significativamente alto, pero cubre las partes más importantes del código. Debajo figura una captura de pantalla con los resultados finales de las pruebas.



- ***Javascripts usados:***
  - Bootstrap, necesario para el correcto funcionamiento de la página junto con las hojas de estilo y el responsive design.
  - Score: permite mostrar las puntuaciones de los archivos de forma dinámica e intuitiva para el usuario.
  - Validation: Controla todo lo referente a los formularios de la página y evita que información no deseada pase al servidor. También controla que los campos de los formularios contienen datos válidos mostrando mensajes de error si no es así.
  - Signup: Javascript que controla el formulario de signup y muestra mensajes de error según sea necesario.
  - EasyPaginate: Este javascript se encarga de mostrar y ordenar los archivos de las asignaturas, creando un listado de páginas donde se muestran únicamente 10 archivos en cada una, de forma que la navegación sea más cómoda para el usuario. Fue tomado de la web y adaptado al proyecto.
  - Prettify: Javascript tomado de google y adaptado a la aplicación para mostrar los archivos de texto o archivos de código con diferentes colores y estilos, de forma que se resalta la sintaxis del código.

- ***Además de:***
  - Bootstrap: hojas de estilo de Twitter usadas en toda la plataforma.
  - Responsive Design: permite ajustar dinámicamente los contenidos de la web a cualquier tamaño de ventana e incluso en dispositivos móviles.
  - Iconos de Glyphicons Free: usados en algunas vistas de la plataforma.
  - Práctica de Syntax Highlight: usada en la vista previa de archivos cuando éstos contienen código de algún lenguaje de programación.

# 5. Referencias


- Página web de Sinatra
  **http://www.sinatrarb.com/intro**

- Sinatra, the book
  **http://sinatra-book.gittr.com/**

- Tutorial de datamapper
  **http://datamapper.org/getting-started.html**

- Documentación de haml
  **http://haml.info/docs/yardoc/file.REFERENCE.html**

- Bootstrap. Libros de estilos, javascripts, iconos,…
  **http://twitter.github.com/bootstrap/base-css.html**

- API de Gravatar
  **https://es.gravatar.com/site/implement/images/**