



GRADO EN INGENIERÍA EN TECNOLOGÍAS DE LA TELECOMUNICACIÓN

Curso Académico 2018/2019

Trabajo Fin de Grado

TORNEOS DE DR. SCRATCH

Autor : Álvaro Fernández Roig

Tutor : Dr. Gregorio Robles

Trabajo Fin de Grado

Torneos de Dr. Scratch

Autor : Álvaro Fernández Roig

Tutor : Dr. Gregorio Robles Martínez

La defensa del presente Proyecto Fin de Carrera se realizó el día de
de 2019, siendo calificada por el siguiente tribunal:

Presidente:

Secretario:

Vocal:

y habiendo obtenido la siguiente calificación:

Calificación:

Fuenlabrada, a de de 2019

*Dedicado a
mi familia,
por su permanente
e incondicional sostén*

Agradecimientos

Aquí vienen los agradecimientos... Aunque está bien acordarse de la pareja, no hay que olvidarse de dar las gracias a tu madre, que aunque a veces no lo parezca disfrutará tanto de tus logros como tú... Además, la pareja quizás no sea para siempre, pero tu madre sí.

Resumen

Aquí viene un resumen del proyecto. Ha de constar de tres o cuatro párrafos, donde se presente de manera clara y concisa de qué va el proyecto. Han de quedar respondidas las siguientes preguntas:

- ¿De qué va este proyecto? ¿Cuál es su objetivo principal?
- ¿Cómo se ha realizado? ¿Qué tecnologías están involucradas?
- ¿En qué contexto se ha realizado el proyecto? ¿Es un proyecto dentro de un marco general?

Lo mejor es escribir el resumen al final.

Summary

Here comes a translation of the “Resumen” into English. Please, double check it for correct grammar and spelling. As it is the translation of the “Resumen”, which is supposed to be written at the end, this as well should be filled out just before submitting.

Índice general

1. Introducción	1
1.1. Contexto	1
1.2. Estructura de la memoria	1
2. Objetivos	3
2.1. Objetivo general	3
2.2. Objetivos específicos	3
2.3. Planificación temporal	4
3. Estado del arte	5
3.1. Python	6
3.2. Django	6
3.3. HTML	6
3.4. CSS	6
3.5. JavaScript	6
3.6. jQuery	6
3.7. Hairball	6
3.8. Scratch	6
3.9. Bootstrap	6
4. Diseño e implementación	7
4.1. Arquitectura general	7
4.1.1. Cliente	8
4.1.2. Servidor	8
4.2. Modelo de datos	9

4.3. Diseño e implementación por funcionalidad	13
4.3.1. Página principal, login y registro de creadores	13
4.3.2. Funcionalidades para creadores	16
5. Resultados	33
6. Conclusiones	35
6.1. Consecución de objetivos	35
6.2. Aplicación de lo aprendido	35
6.3. Lecciones aprendidas	35
6.4. Trabajos futuros	36
A. Manual de usuario	37
Bibliografía	39

Índice de figuras

4.1. Arquitectura general	7
4.2. Diagrama de entidad-relación	9
4.3. Reto no superado	11
4.4. Reto superado	11
4.5. Diagrama de estados de una partida	13
4.6. Página principal de Dr. Scratch	14
4.7. Página principal de torneos de Dr. Scratch: Menú de login	15
4.8. Página principal de torneos de Dr. Scratch: Registro de creadores	16
4.9. Página principal de creadores	17
4.10. Página principal de la administración de torneos	17
4.11. Listado de retos	18
4.12. Detalle de un reto	19
4.13. Creación de equipo	21
4.14. Ver participantes de un equipo	22
4.15. Creación de torneo	24
4.16. Detalle de torneo	25
4.17. Validación manual: listado de torneos y equipos	27
4.18. Validación manual: ventana modal	27
4.19. Progreso de los torneos	30
4.20. Registrar a nuevos participantes	31

Capítulo 1

Introducción

El siguiente trabajo consta de la realización de un módulo independiente para la aplicación Dr. Scratch. Se trata del módulo de torneos de Dr. Scratch, que permitirá a los docentes la creación de torneos y la participación en ellos de sus alumnos.

1.1. Contexto

Dr. Scratch es una aplicación web que permite la evaluación de proyectos realizados a través de la herramienta Scratch. A través del análisis de los proyectos, Dr. Scratch puntúa en una escala de cero a tres una serie de parámetros relacionados con el pensamiento computacional, e identifica posibles errores en la realización de los proyectos, como duplicidades o código muerto. Los parámetros evaluados son paralelismo, pensamiento lógico, control de flujo, interactividad con el usuario, representación de la información, abstracción y sincronización.

(PENDIENTE)De manera didáctica como ayuda a los niños etc. (Bibliografía)

(PENDIENTE)Autoevaluación, trabajo en equipo, proyectos, independencia(Bibliografía)

1.2. Estructura de la memoria

En esta sección se debería introducir la estructura de la memoria. Así:

- En el primer capítulo se hace una intro al proyecto.
- En el capítulo 2 (ojo, otra referencia automática) se muestran los objetivos del proyecto.

- A continuación se presenta el estado del arte.
- ...

Capítulo 2

Objetivos

2.1. Objetivo general

La realización del módulo de torneos para Dr. Scratch surge bajo la premisa de realizar modificaciones en la aplicación de Dr. Scratch para tratar que ésta resulte más atractiva a los estudiantes que la utilicen y que esto ayude a su vez a fomentar un mayor uso de la aplicación original.

2.2. Objetivos específicos

- Promover el análisis de proyectos Scratch a través de Dr. Scratch proporcionando un apartado de carácter lúdico que motive y atraiga a los estudiantes.
- Proveer a los profesores de una aplicación administrable, que les permita tener el control de la creación, edición y borrado de las distintas entidades relacionadas con la creación de los torneos.
- Mejorar el control y la evaluación por parte de los profesores de los proyectos realizados por sus alumnos, proporcionándoles información centralizada sobre el avance y resultados de los proyectos a través de la aplicación.
- Permitir a los alumnos realizar una autoevaluación de sus proyectos Scratch mostrando información de ayuda para la mejora en los distintos parámetros evaluados.

- Realizar una aplicación intuitiva, cuya información sea comprensible tanto por profesores como por alumnos y cuyos procesos sean sencillos y tengan un aspecto visual atractivo y en consonancia la aplicación de Dr. Scratch existente.

2.3. Planificación temporal

A mí me gusta que aquí pongáis una descripción de lo que os ha llevado realizar el trabajo. Hay gente que añade un diagrama de GANTT. Lo importante es que quede claro cuánto tiempo llevas (tiempo natural, p.ej., 6 meses) y a qué nivel de esfuerzo (p.ej., principalmente los fines de semana).

Capítulo 3

Estado del arte

Descripción de las tecnologías que utilizas en tu trabajo. Con dos o tres párrafos por cada tecnología, vale. Se supone que aquí viene todo lo que no has hecho tú.

Puedes citar libros, como el de Bonabeau et al. sobre procesos estigmérgicos [1].

También existe la posibilidad de poner notas al pie de página, por ejemplo, una para indicarte que visite la página de LibreSoft¹.

¹<http://www.libresoft.es>

3.1. Python

3.2. Django

3.3. HTML

3.4. CSS

3.5. JavaScript

3.6. jQuery

3.7. Hairball

3.8. Scratch

3.9. Bootstrap

Capítulo 4

Diseño e implementación

A continuación se detalla la aplicación torneos de Dr. Scratch, incluyendo su arquitectura y su modelo de datos.

4.1. Arquitectura general

La aplicación está desarrollada usando el framework Django, cuya arquitectura está basada en el modelo vista controlador MVC. Sin embargo, la terminología utilizada por Django varía con respecto a otros frameworks. En este caso, el controlador se denomina vista y la vista plantilla o template, por lo que en lugar de utilizar el acrónimo MVC se utiliza el acrónimo MTV: model, template, view¹.

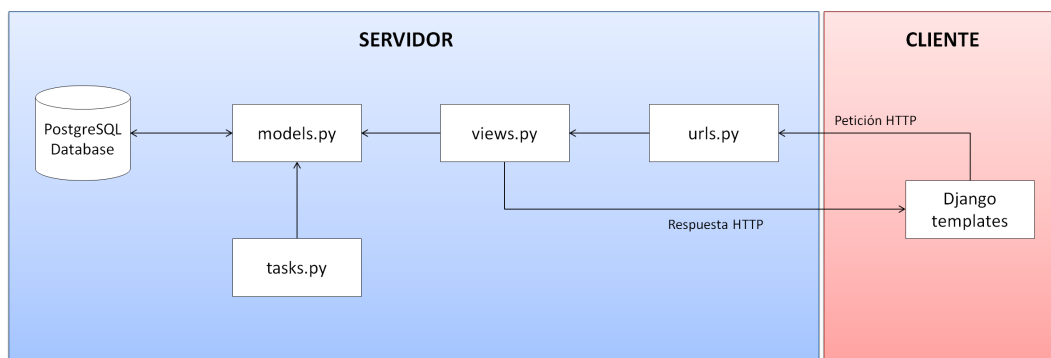


Figura 4.1: Arquitectura general

¹<https://docs.djangoproject.com/es/2.1/faq/general/#django-appears-to-be-a-mvc-framework-but-you-call-the-controller-the-view-and-the-view-the-template-how-come-you-don-t-use-the-standard-names>

4.1.1. Cliente

En el lado del cliente se encuentran las plantillas de Django que serán visualizadas por los usuarios a través de un navegador web. Estas plantillas son archivos en formato HTML en los que se muestra la información de la respuesta recibida desde la vista. Así mismo, las plantillas dan formato a la información de respuesta usando hojas de estilo en formato CSS y scripts en lenguaje Javascript, ejecutados en el lado del cliente. En este caso, se ha utilizado el framework Bootstrap para el desarrollo del formato de las plantillas.

4.1.2. Servidor

En el lado del servidor se realiza el desarrollo back-end al completo, desde la recepción de peticiones HTTP hasta el acceso a datos.

En primer lugar, las peticiones HTTP son recibidas por un despachador de URLs denominado `urls.py` en el que se define un mapeo de expresiones regulares donde cada expresión regular corresponde a un método de la vista `views.py`. De este modo, cuando se recibe una petición HTTP que cumple una expresión regular del despachador de URLs, éste invoca al método de la vista correspondiente.

En la vista `views.py` se encuentra la lógica de la aplicación, en cada método de la vista se recibe una serie de parámetros de entrada, se procesa la información recibida y finalmente, se retorna la información resultante al cliente a través de una plantilla. En la vista se realiza también el acceso a la base de datos, este acceso se realiza a través de las entidades existentes en el modelo `models.py`. Cada entidad del modelo corresponde a una tabla existente en la base de datos y para realizar las consultas se realizan llamadas a los diferentes métodos de la API Queryset de Django. A través de estos métodos pueden realizarse tanto consultas como inserciones, modificaciones o borrado de datos de la base de datos.

Cabe mencionar que en este caso se ha utilizado la librería Django Background Tasks² para la creación de procesos asíncronos. La función a ejecutar se encuentra definida en el archivo `tasks.py` que actúa como una vista, en este caso, la función asíncrona accede a la base de datos y envía un correo electrónico si se cumple una serie de condiciones.

²<https://django-background-tasks.readthedocs.io/en/latest/>

4.2. Modelo de datos

Para la creación del módulo de torneos de Dr. Scratch se ha utilizado un modelo de datos independiente del modelo existente en la aplicación original de Dr. Scratch. El diagrama de entidad-relación utilizado es el siguiente:

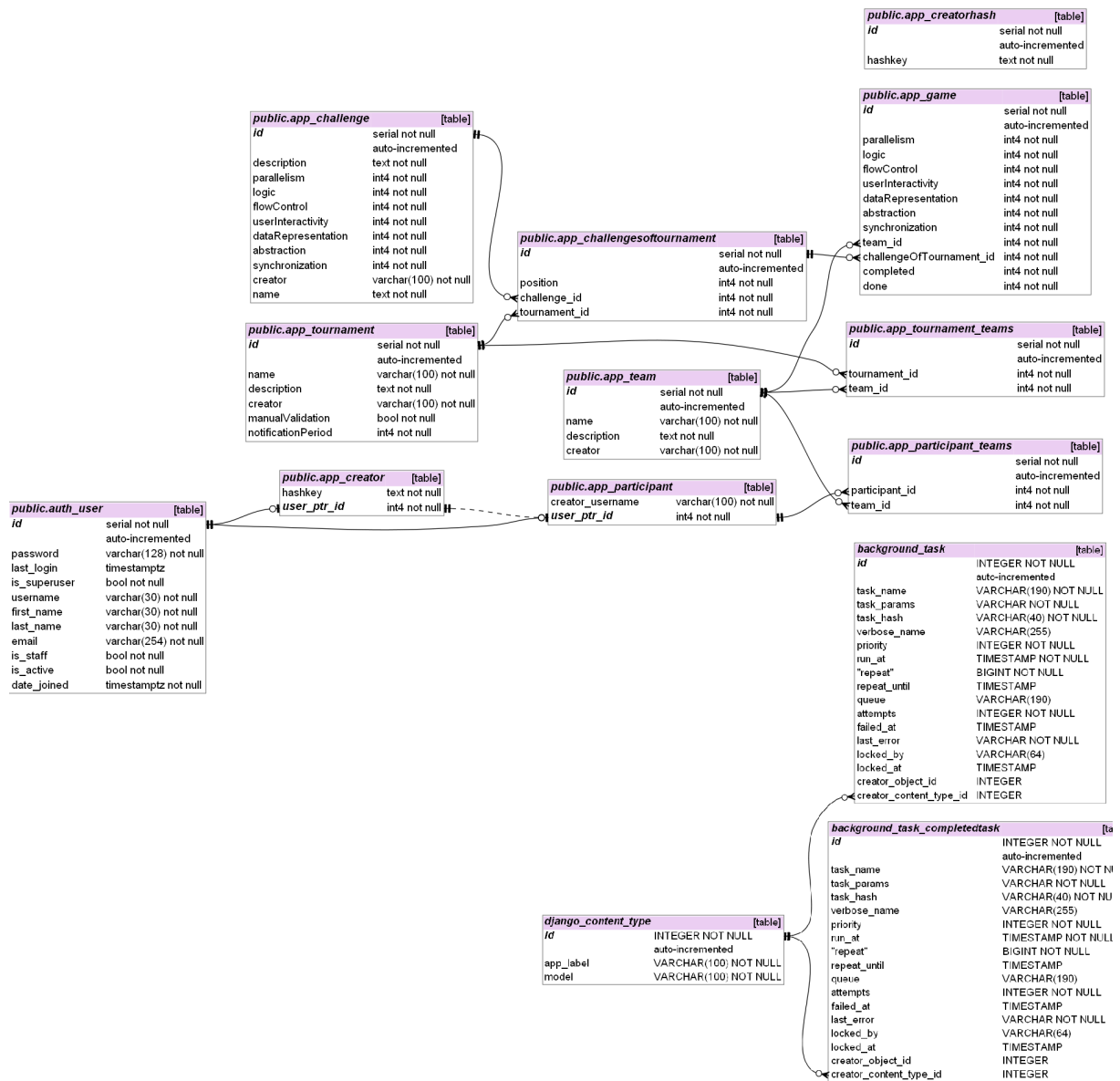


Figura 4.2: Diagrama de entidad-relación

A continuación se detallan las principales entidades del modelo. Estas definiciones ponen en contexto conceptos que deben ser previamente tenidos en cuenta a la hora de detallar el desarrollo de la aplicación.

- **Creator:** La entidad `Creator` representa a los profesores, que serán los creadores de los torneos. Son los encargados de crear y administrar los torneos, de registrar a los participantes y de controlar y validar las partidas realizadas por los equipos. Para ello dispondrán de apartados solo para creadores que les permitan la realización de estas funciones.
- **CreatorHash:** Entidad creada para realizar el registro de creadores. El equipo de Dr. Scratch almacena un código hash en esta tabla y se lo proporciona a los profesores interesados en la utilización de torneos de Dr. Scratch para que puedan introducirlo en el formulario de registro.
- **Participant:** La entidad `Participant` representa a los estudiantes, que serán los participantes de los torneos creados. Su único cometido es jugar. Los participantes no disponen de acceso a las funcionalidades de administración y control destinadas a los creadores.
- **Tournament:** Los torneos son la entidad principal sobre la que se basa el juego y están compuestos por retos y equipos. Cada torneo lo juega uno o varios equipos, cada uno de esos equipos debe superar de uno en uno y en orden los retos que se hayan definido en el torneo. Cuando un equipo haya superado todos los retos superará el torneo, sin embargo, no se ha establecido un procedimiento de ganador.
- **Team:** Los equipos se componen de uno o varios participantes, permitiendo a los profesores establecer juegos individuales o por equipos. Pueden estar asociados a múltiples torneos.
- **Challenge:** Los retos son los desafíos individuales que deberán superar los equipos y pueden estar asociados a uno o más torneos. En ellos se determina un valor para cada uno de los parámetros de evaluación de Dr. Scratch. Para superar el reto, los equipos deberán realizar un análisis de un proyecto Scratch cuyo resultado en cada uno de los parámetros evaluados sea igual o superior a los valores establecidos en el reto.



Figura 4.3: Reto no superado

En la figura X.X, los valores del reto se muestran en la columna *Valor requerido*. El reto se superará cuando los valores obtenidos en los parámetros lógica, control de flujo, representación de los datos y sincronización sean iguales o mayores que los establecidos en el reto.



Figura 4.4: Reto superado

- **ChallengesOfTournament:** Los retos definidos en un torneo deben tener un orden, para ello se ha definido la entidad `ChallengesOfTournament` cuyo atributo `position` ordena los retos de un torneo.
- **Game:** La entidad `Game` representa las partidas de los equipos. Cada partida está asociada a un equipo y a un reto concretos. En las figuras X.X y X.Y, los valores que representan la puntuación de la partida de un equipo se muestran en la columna *Puntuación más alta*.

Las partidas son comunes para todos los participantes de un equipo, de este modo si un equipo tiene varios participantes y uno de ellos analiza un proyecto obteniendo una puntuación concreta, la partida se guarda y la puntuación será visualizada por el resto de participantes.

Así mismo, los valores de la puntuación de la partida se sobrescriben solo si dicho valor en concreto es superior al de la partida existente, los valores que sean inferiores a su correspondiente valor en la partida existente no se actualizarán.

- **Background Task:** Los retos de un torneo pueden ser validados de manera automática, con el procedimiento descrito anteriormente, o de manera manual. Si se selecciona validación manual, cuando un reto haya sido superado por un equipo, se quedará en un estado intermedio denominado `Finalizado`. En ese momento el creador deberá comprobar la puntuación y validar manualmente la partida que pasará al estado final `Completado`. A continuación se muestra un diagrama de estados que describe el proceso que se realiza al jugar una partida.

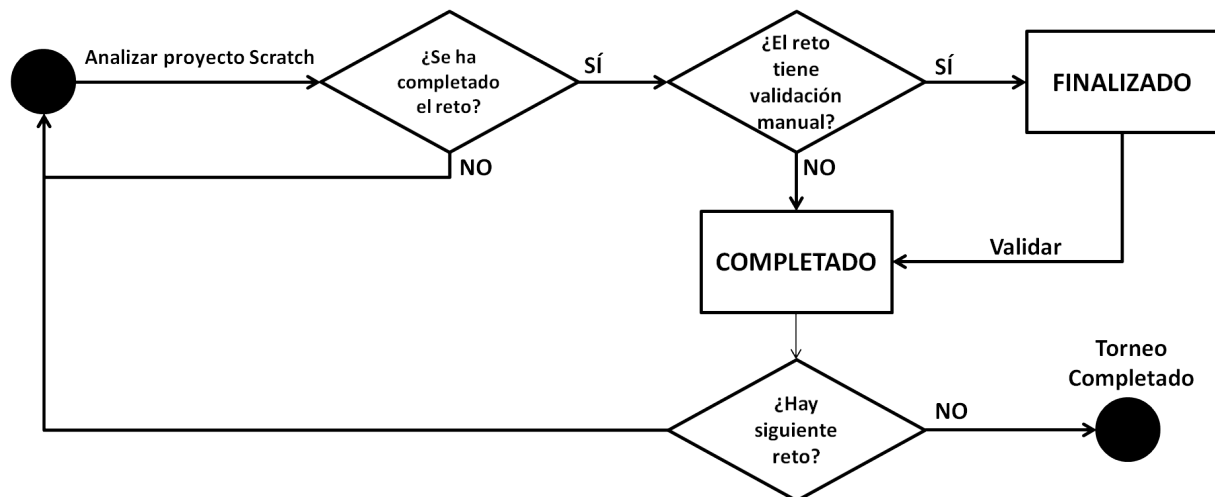


Figura 4.5: Diagrama de estados de una partida

Se ha definido un mecanismo para notificar al creador a través del envío de correos electrónicos cuando un equipo finaliza una partida. Cuando el creador selecciona validación manual, puede también seleccionar la frecuencia en la que quiere recibir correos electrónicos de aviso. Las frecuencias son: nunca, inmediatamente, una vez al día y una vez a la semana.

Para el envío de correos con frecuencia una vez al día o una vez a la semana se ha creado un proceso asíncrono que revisa si existen partidas finalizadas y sin validar.

En la entidad `Background Task` se almacena un registro identificado con el email del creador y el identificador del torneo para cada ejecución asíncrona. Los registros se actualizan en cada una de las iteraciones del proceso.

4.3. Diseño e implementación por funcionalidad

4.3.1. Página principal, login y registro de creadores

La aplicación se ha integrado como un módulo independiente dentro de Dr. Scratch. Para acceder se ha creado un enlace en la barra principal superior de Dr. Scratch que redirige a la página principal de torneos.



Figura 4.6: Página principal de Dr. Scratch

La página principal de torneos consta de un registro para creadores y de una sección en la barra superior con un menú de *login*. La página principal se ha creado manteniendo el diseño existente en la sección de organizaciones de Dr. Scratch.

La aplicación torneos de Dr. Scratch presenta funcionalidades independientes que dependen de si el usuario es un profesor o un alumno. Para ello se ha definido un *login* que identifica si el usuario es un creador o un participante y le redirige a una pantalla principal diferente en cada caso.

En el menú de *login* también se encuentra la funcionalidad de recuperación de contraseña. En el proceso de recuperación de contraseña el usuario deberá introducir su email, que sirve para identificarle en la base de datos y enviarle un correo electrónico con un enlace único en el que se incluirá un token generado y que seguirá el siguiente formato: `http://torneosdrscratch01.azurewebsites.net/reset_password_tournaments/MQ-54f-0bd3d0c01b4209132767/`

Al pulsar en el enlace el usuario será dirigido a una página en la que podrá establecer una nueva contraseña.

Dado que los participantes pueden ser registrados con nombre de usuario y sin correo electrónico, si el participante no tiene asociada una cuenta de correo electrónico se envía el correo de recuperación de contraseña al creador que le haya registrado.



The image shows a login form for the Dr. Scratch tournaments page. At the top, there is a red button labeled 'INGRESAR' with a downward arrow. Below this, the form is contained within a white box with a thin border. It features two input fields: the first is labeled 'Nombre de usuario' and the second is for a password, indicated by a series of dots. Below the password field, there is a red link that says '¿HAS OLVIDADO TU CONTRASEÑA?'. At the bottom of the form is a grey button labeled 'Ingresar' in red text.

Figura 4.7: Página principal de torneos de Dr. Scratch: Menú de login

En cuanto al registro de creadores, se trata de un registro que sigue el mismo proceso que el utilizado para el registro de organizaciones de Dr. Scratch. El equipo de Dr. Scratch debe darle una clave al profesor para que la introduzca en el formulario de registro junto con su nombre de usuario, correo electrónico y contraseña. La clave proporcionada al profesor deberá encontrarse almacenada en el campo `hashkey` de la tabla `app_creatorhash` y cuando el profesor se registre se eliminará de esta tabla y pasará a ser el campo `hashkey` de la tabla `app_creator`.

TORNEOS DE DR. SCRATCH

Cómo participar en los torneos de Dr. Scratch?

Si quieres ser uno de los 10 primeros profesores que prueban Dr. Scratch envíanos un email a team@drscratch.org y te daremos una clave para poder registrarte en la web



Nombre de usuario

E-mail


Contraseña

Clave


¡REGÍSTRATE!



team@drscratch.org



[@DrScratchTool](https://twitter.com/DrScratchTool)



[GitHub\(issues\)](https://github.com/DrScratchTool/issues)

Figura 4.8: Página principal de torneos de Dr. Scratch: Registro de creadores

4.3.2. Funcionalidades para creadores

Los creadores dispondrán de un menú principal con cuatro funcionalidades independientes: administración de torneos, validación manual, progreso de los torneos y registrar nuevos participantes. La página principal de torneos se ha creado usando el componente `carousel` de Bootstrap con el que se puede acceder a cada una de las funcionalidades existentes.



Figura 4.9: Página principal de creadores

4.3.2.1. Administración de torneos

La administración de torneos consiste en un módulo de administración para poder consultar, crear, editar y borrar torneos, equipos y retos. En la pantalla principal se encuentra un enlace a la administración de cada una de las entidades que son administrables por separado.

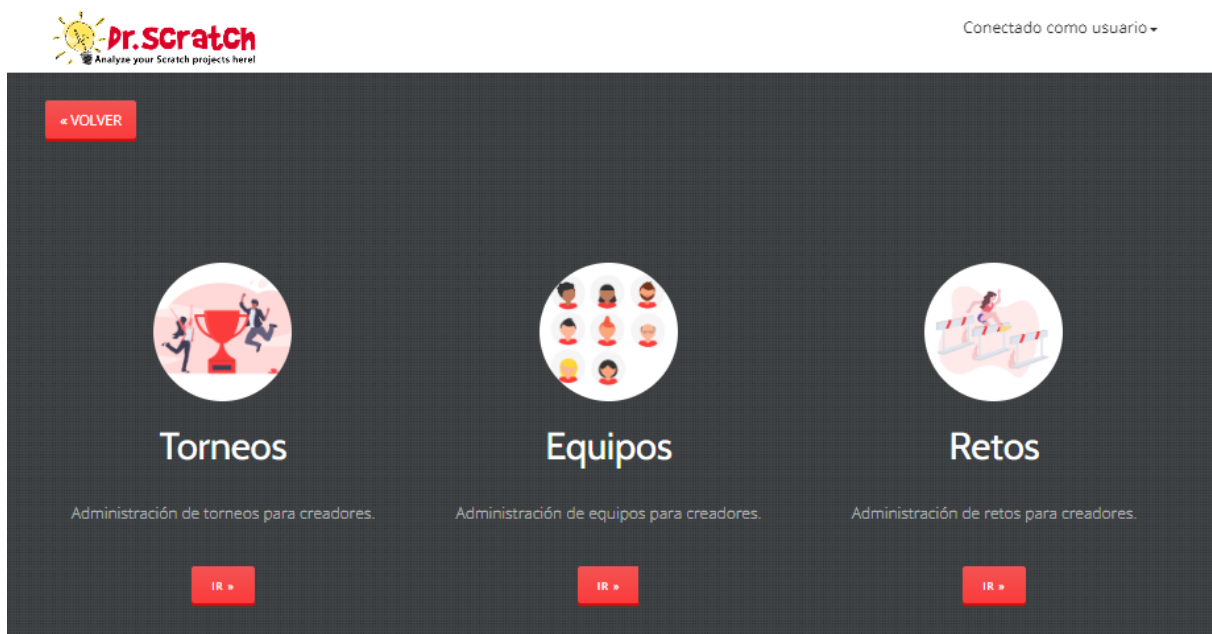


Figura 4.10: Página principal de la administración de torneos

Retos

Al acceder a la administración de retos se muestra un listado con los retos creados por el usuario. Por cada reto se muestra su nombre y descripción y tres botones que permiten ver el detalle del reto, editarlo o eliminarlo. El listado se divide en páginas de tres elementos como máximo. La navegación entre páginas se realiza a través de un paginador. Así mismo, en la parte inferior se encuentra un botón que permite crear un nuevo reto.

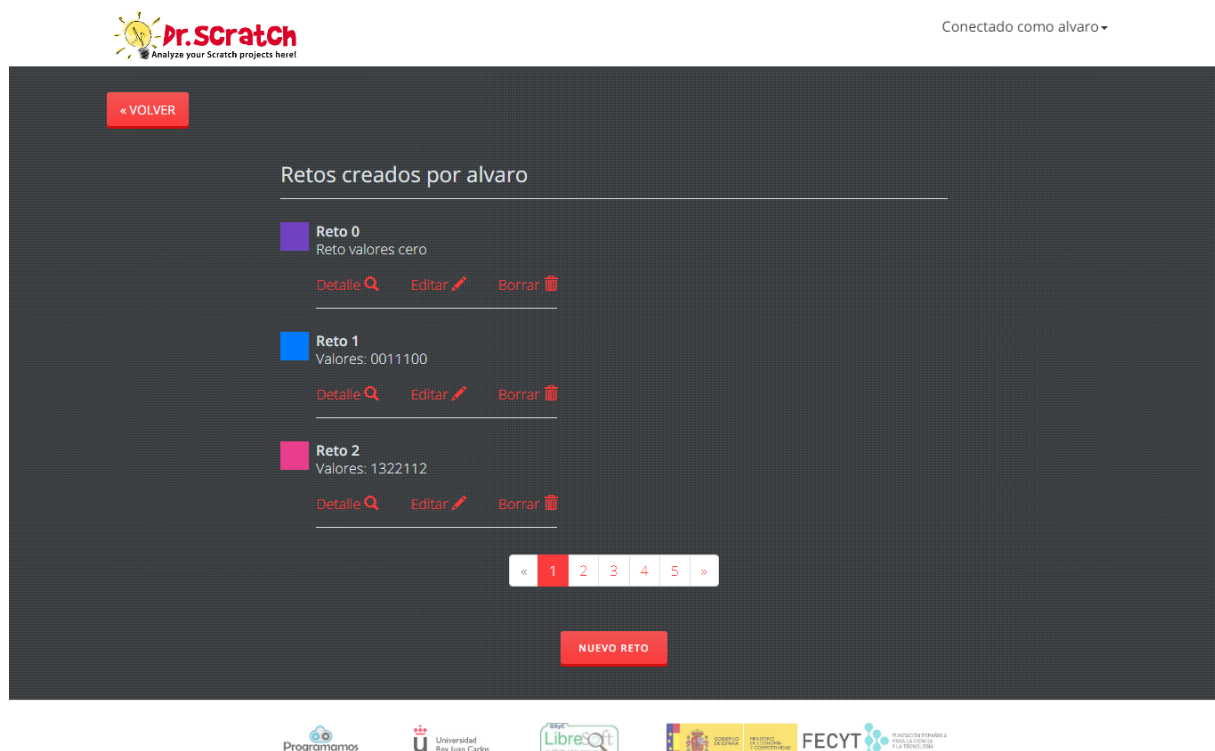


Figura 4.11: Listado de retos

Para crear o editar un reto se ha definido un formulario de Django denominado `ChallengeForm` en el archivo `forms.py`.

```
DIFFICULTY_CHOICES = [
    (0, _('None: 0')),
    (1, _('Easy: 1')),
    (2, _('Medium: 2')),
    (3, _('Hard: 3')),
]

class ChallengeForm (forms.Form):
    name = forms.CharField(max_length=50, widget=forms.TextInput(), required=True)
    description = forms.CharField(max_length=500, widget=forms.Textarea(), required=True)
```

```

parallelism = forms.ChoiceField(widget=forms.RadioSelect, choices=DIFFICULTY_CHOICES, initial=0)
logic = forms.ChoiceField(widget=forms.RadioSelect, choices=DIFFICULTY_CHOICES, initial=0)
flowControl = forms.ChoiceField(widget=forms.RadioSelect, choices=DIFFICULTY_CHOICES, initial=0)
userInteractivity = forms.ChoiceField(widget=forms.RadioSelect, choices=DIFFICULTY_CHOICES, initial=0)
dataRepresentation = forms.ChoiceField(widget=forms.RadioSelect, choices=DIFFICULTY_CHOICES, initial=0)
abstraction = forms.ChoiceField(widget=forms.RadioSelect, choices=DIFFICULTY_CHOICES, initial=0)
synchronization = forms.ChoiceField(widget=forms.RadioSelect, choices=DIFFICULTY_CHOICES, initial=0)

```

Cuando se edita un reto, se recogen los valores del reto seleccionado de la base de datos y se precargan en el formulario `ChallengeForm` sobrescribiendo el parámetro `initial`.

Al pulsar en el botón `Borrar` aparecerá una ventana modal en la que se solicita confirmación para borrar el reto. Al confirmar, el reto se borrará siempre y cuando no esté asociado a ningún torneo.

El detalle del reto despliega una ventana modal en la que, de manera visual a través de *progress bars* de Bootstrap, se muestran los valores de cada uno de los parámetros del reto. Cada uno de los conceptos tendrá un enlace al tutorial de Dr. Scratch por si el profesor quiere revisar el significado de alguno de los conceptos de pensamiento computacional evaluables.

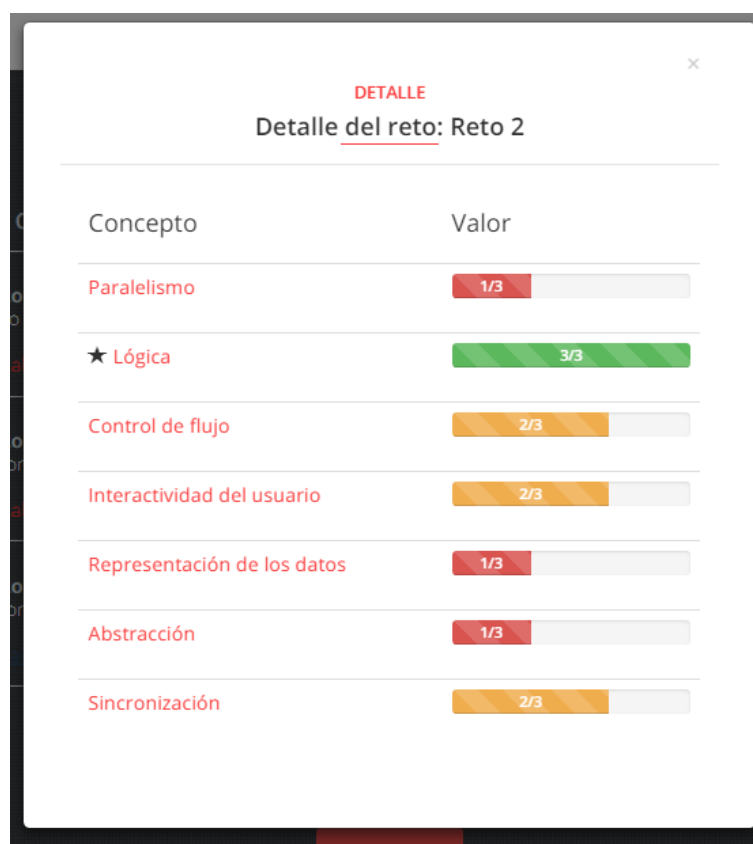


Figura 4.12: Detalle de un reto

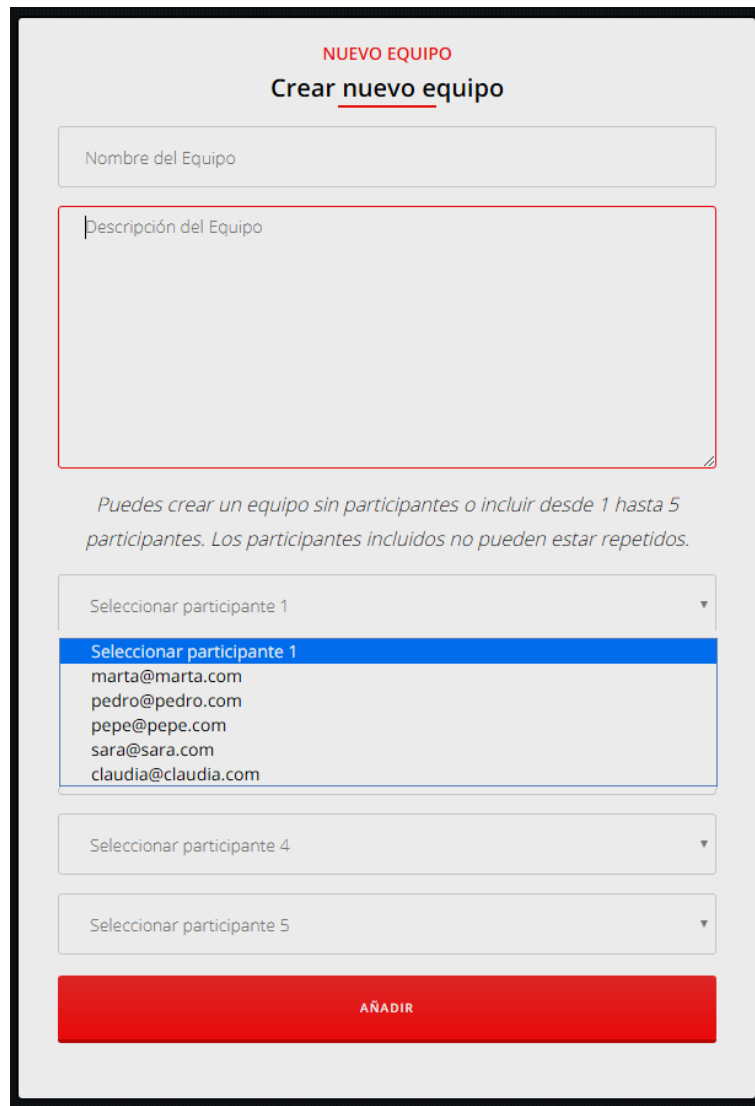
Equipos

Al acceder a la administración de equipos, se muestra una pantalla con un listado de equipos, esta pantalla tiene el mismo formato que el listado de retos de la figura X.X, pudiendo crear, editar o borrar equipos y, además, consultar los participantes de los equipos creados.

El formulario de creación y edición de equipos se denomina `TeamForm` y sus campos son únicamente nombre y descripción.

```
class TeamForm (forms.Form):  
    name = forms.CharField(max_length=50, widget=forms.TextInput(), required=True)  
    description = forms.CharField(max_length=500, widget=forms.Textarea(), required=True)
```

Para añadir participantes al equipo, se incluyen listas desplegables con los participantes que haya registrado el creador que está administrando el equipo. Podrán incluirse hasta un máximo de cinco participantes por equipo.



NUEVO EQUIPO

Crear nuevo equipo

Nombre del Equipo

Descripción del Equipo

Puedes crear un equipo sin participantes o incluir desde 1 hasta 5 participantes. Los participantes incluidos no pueden estar repetidos.

Seleccionar participante 1

- Seleccionar participante 1
- marta@marta.com
- pedro@pedro.com
- pepe@pepe.com
- sara@sara.com
- claudia@claudia.com

Seleccionar participante 4

Seleccionar participante 5

AÑADIR

Figura 4.13: Creación de equipo

La consulta de participantes sigue el mismo formato que en la administración de retos, pudiendo consultar los participantes de los equipos creados a través de una ventana modal.



Figura 4.14: Ver participantes de un equipo

El borrado de un equipo se realiza del mismo modo que el borrado de retos. Se solicita confirmación a través de una ventana modal y no pueden borrarse equipos que se encuentren asociados a algún torneo.

Torneos

Al acceder a la administración de torneos, se muestra una pantalla con un listado de torneos, esta pantalla tiene el mismo formato que el listado de retos o equipos de la figura X.X, pudiendo crear torneos o editar, borrar y ver el detalle de los torneos creados.

La creación de torneos consta de tres secciones. En primer lugar se define el nombre y la descripción del torneo. En segundo lugar, se establecen los equipos y retos del torneo. Podrán incluirse hasta un máximo de cuatro equipos y cuatro retos. Los retos incluidos se ordenarán de arriba a abajo con respecto al formulario. Para facilitar la creación de torneos, se han definido dos enlaces que abren sendas ventanas modales con los formularios de creación de retos y equipos respectivamente. Y en tercer lugar debe seleccionarse la configuración de validación de los retos del torneo y la periodicidad de recepción de correos electrónicos de notificación.

Para la creación y edición de torneos se ha creado el formulario denominado `TournamentForm`.

```
NOTIFICATION_CHOICES= [  
    (0, _('Never')),  
    (1, _('Immediately')),
```

```
(2, _('Once a day')),
(3, _('Once a week')),
]

class TournamentForm(forms.Form):
    name = forms.CharField(max_length=50, widget=forms.TextInput(), required=True)
    description = forms.CharField(max_length=500, widget=forms.Textarea(), required=True)
    manualValidation = forms.BooleanField(widget=forms.CheckboxInput(), required=False)
    notificationPeriod = forms.ChoiceField(widget=forms.Select(), choices=NOTIFICATION_CHOICES, initial=0)
```

NUEVO TORNEO

Crear un nuevo torneo

Nombre del Torneo

Descripción del Torneo

Puedes crear un torneo sin retos o incluir desde 1 hasta retos. Los retos incluidos no pueden estar repetidos.

Seleccionar reto 1

Seleccionar reto 2

Seleccionar reto 3

Seleccionar reto 4

Nuevo Reto

Puedes crear un torneo sin equipos o incluir desde 1 hasta equipos. Los equipos incluidos no pueden estar repetidos.

Seleccionar equipo 1

Seleccionar equipo 2

Seleccionar equipo 3

Seleccionar equipo 4

Nuevo Equipo

Puedes validar los retos del torneo manualmente. El reto no se completará hasta que no valides la puntuación del estudiante en la sección de Validación manual.

Validar los retos manualmente ☒

Puedes ser notificado con un email cuando un participante complete el reto. Selecciona la frecuencia de notificación.

Nunca

AGADIR

El borrado de torneos se realizará del mismo modo que el borrado de equipos y retos. Al pulsar en el botón **Borrar** se abre una ventana modal para pedir confirmación al creador.

Al pulsar el botón **Detalle** se abre una ventana modal con la información de los retos y equipos establecidos así como de la configuración de validación.

En la ventana modal aparece un listado con los nombres de los retos y los equipos, sin embargo, si se pasa el ratón por encima del nombre de un reto aparecerá el valor de sus parámetros y si se pasa por encima del nombre de un equipo aparecerán los nombres de usuario de sus participantes. De este modo el creador dispondrá de toda la información necesaria en el detalle de torneo.



Figura 4.16: Detalle de torneo

4.3.2.2. Validación manual

En la sección de validación manual se podrán comprobar y validar las partidas de los torneos seleccionados con la opción de validación manual.

Para discriminar entre las partidas en estado *Finalizado* y las partidas en estado *Completado* (figura X.X), se han incluido dos campos en la entidad `Game` denominados `completed` y `done`.

```
completed = models.IntegerField(default=int(0))
done = models.IntegerField(default=int(0))
```

Cuando una partida pasa al estado *Finalizado*, se marca el campo `done` con un 1. En el caso de que el torneo en cuestión sea de validación automática el campo `completed` también se marca con un 1 y se pasa al siguiente reto, pero si el torneo es de validación manual el campo `completed` será 0.

Por lo tanto, en la sección de validación manual aparecerán las partidas (`Game`) cuyo campo `done` sea 1 y su campo `completed` sea 0, y al validar una partida se marcará el campo `completed` a 1, pasando el equipo correspondiente al siguiente reto, si lo hubiera, o superando el torneo si fuera el último reto.

Para la sección de validación manual se han creado dos pantallas, la primera contiene un listado con todos los torneos que tienen alguna partida superada y sin validar. En este listado cada torneo tiene un botón denominado *Validar Partidas* que lleva al creador a otro listado con los equipos de ese torneo que tienen partidas pendientes de validación.

En el segundo listado, cada equipo tiene a su vez un botón denominado *Validar* que abre una ventana modal en la que el creador puede ver los miembros del equipo seleccionado, los valores de los parámetros del reto y la puntuación de la partida y en la que podrá finalmente validar la partida.

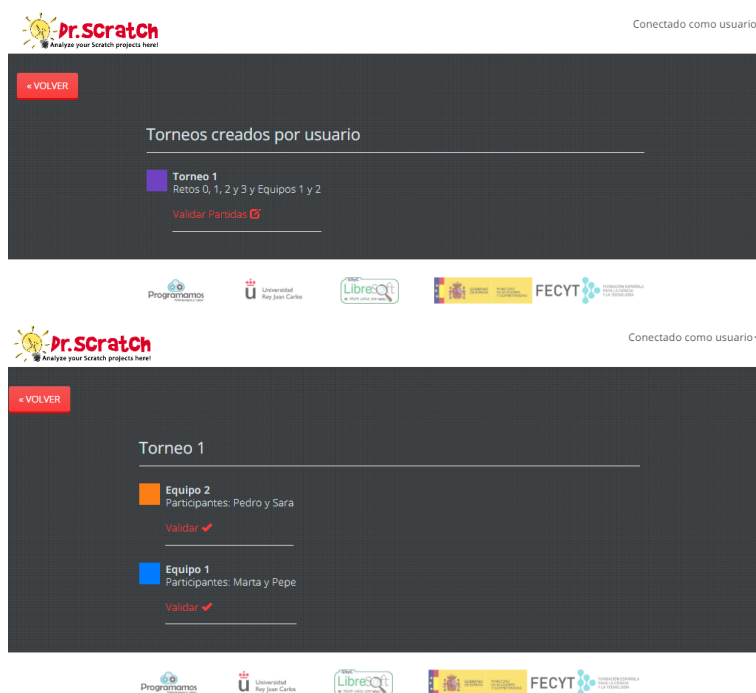


Figura 4.17: Validación manual: listado de torneos y equipos

Conectado como usuario

Torneos creados por usuario

Torneo 1

Retos 0, 1, 2 y 3 y Equipos 1 y 2

Validar Paradas: ☒

Conectado como usuario

Torneo 1

Equipo 2

Participantes: Pedro y Sara

Validar ☒

Equipo 1

Participantes: Marta y Pepe

Validar ☒

Figura 4.18: Validación manual: ventana modal

4.3.2.3. Progreso de los torneos

En la sección de progreso de los torneos, los creadores podrán realizar un seguimiento de todas las partidas que se encuentren en curso para cada uno de los torneos que hayan creado.

Al acceder a la sección de progreso de los torneos se muestra un listado con los torneos creados por el usuario, que mantiene el formato usado en la figura X.X.

En este caso, cada torneo dispone de dos operaciones, la primera es poder ver el detalle de los torneos. El detalle tiene el mismo formato que el definido en la sección de administración (figura X.X).

La segunda operación muestra la pantalla de progreso de los torneos. Esta pantalla tiene una finalidad meramente informativa y no presenta ninguna operación. En ella se muestra la información de las partidas del mismo modo que se muestra a los equipos a la hora de jugar.

Con un título en rojo se muestra el nombre del equipo que ha realizado la partida y debajo, el nombre y descripción del torneo.

Bajo la descripción del torneo se muestra una barra de progreso con pasos, en la que cada paso corresponde a un reto del torneo. Con los botones *Anterior* y *Siguiente* el creador podrá navegar por los distintos retos del torneo. A diferencia de los participantes, que no podrán avanzar al siguiente reto a no ser que completen el reto en el que se encuentran, el creador podrá navegar por todos los retos, estén superados o no por el equipo, para disponer de la información completa del torneo.

Los números de la barra de progreso siguen el siguiente código de colores:

- El reto marcado en **amarillo** será el reto seleccionado por el creador. Por lo tanto, la información que aparecerá en el panel será la correspondiente a ese reto.
- Los retos marcados en **verde** serán los retos superados por el equipo.
- Los retos marcados en **gris** serán los retos que aún no se encuentran superados por el equipo.

Para crear la barra de progreso se ha utilizado el plugin jQuery Easy Wizard³. En el archivo `easyWizard.css` se han adaptado los colores para que coincidieran con los tonos usados en

³<https://www.jqueryscript.net/other/Simple-Wizard-Modal-Plugin-with-jQuery-Bootstrap-Easy-Wizard.html>

las *progress bars* de Dr. Scratch, y el archivo `easyWizard.js` se ha sustituido por completo utilizando un script con un comportamiento totalmente personalizado.

En la parte inferior de la barra de progreso se muestra el nombre y la descripción del reto seleccionado y bajo estos, los valores establecidos en el reto en la columna *Valor requerido* y los valores de la partida en la columna *Puntuación más alta*.

Finalmente, en la parte inferior se encuentra un paginador, en el que cada página corresponde a un equipo incluido en el torneo.

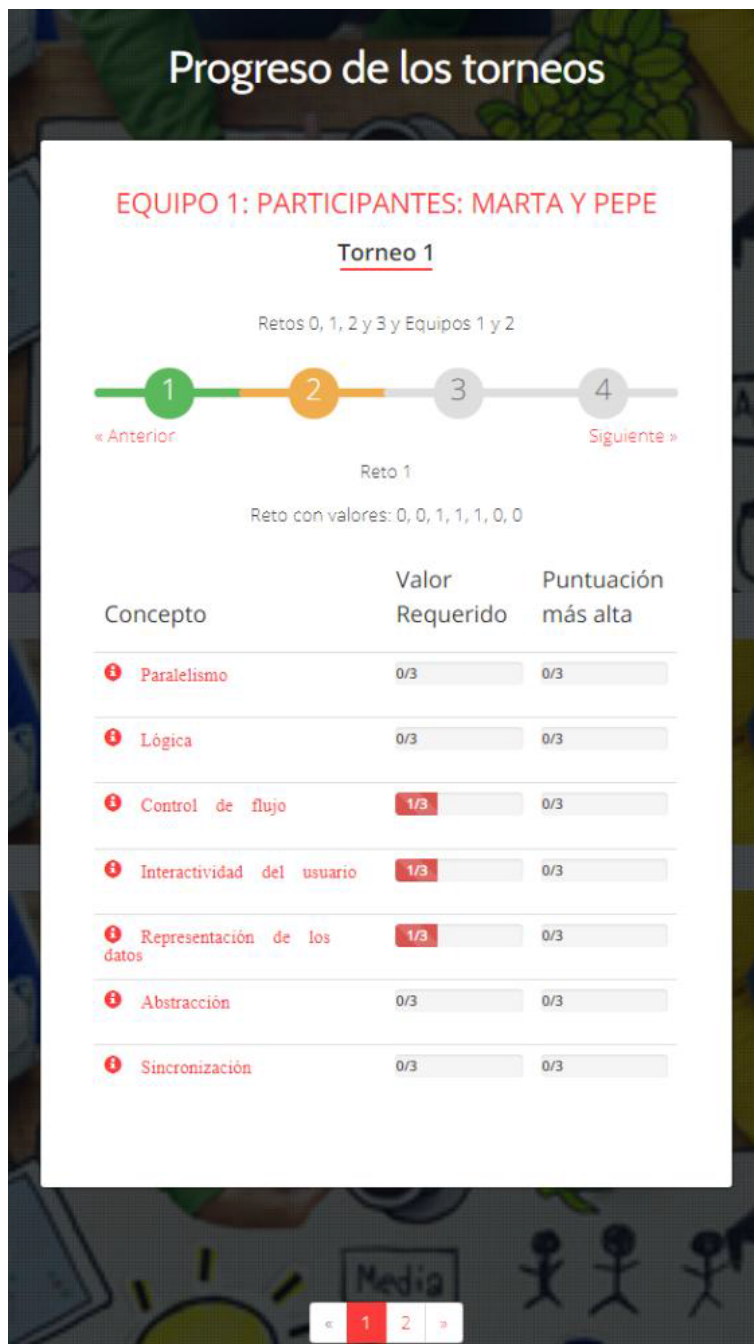


Figura 4.19: Progreso de los torneos

4.3.2.4. Registrar a nuevos participantes

La sección de registro de participantes consta de una única pantalla que incluye tres formularios. En ellos, los creadores podrán registrar de uno a veinte participantes al mismo tiempo pudiendo usar cualquiera de las tres opciones indistintamente.

Figura 4.20: Registrar a nuevos participantes

Los dos primeros formularios constan cada uno de un elemento de tipo *File*. Al recibir el fichero adjunto, se comprueba que la extensión del archivo sea CSV y TXT respectivamente y se leen las primeras veinte líneas de los archivos utilizando la siguiente función:

```
username = file.readline().rstrip()
```

En el tercer formulario se establecen veinte campos de tipo *input*. Mediante jQuery se ocultan todos los *inputs* salvo el primero y cuando el creador pulsa el botón + se muestra el siguiente *input*. La función jQuery utilizada es la siguiente:

```
$(document).on("ready", function(){
    $('<div>.row-line').eq(0).show();
});
$(".plus-link").on("click", function(){
```

```
newId = parseInt(this.id)+1
if ($('#row-line').eq(newId).length > 0){
    $('#row-line').eq(newId).show();
}
$(this).hide();
});
```

Posteriormente, se registran uno por uno los participantes comprobando que aún no se encuentran registrados y utilizando el nombre de usuario recibido también como contraseña.

Existe una diferencia entre los participantes registrados usando una dirección de correo electrónico y los registrados usando un nombre de usuario. Si los participantes no tienen dirección de correo electrónico propia, el email de recuperación de contraseña se enviará a la dirección del creador que los haya registrado.

Capítulo 5

Resultados

En este capítulo se incluyen los resultados de tu trabajo fin de grado.

Si es una herramienta de análisis lo que has realizado, aquí puedes poner ejemplos de haberla utilizado para que se vea su utilidad.

Capítulo 6

Conclusiones

6.1. Consecución de objetivos

Esta sección es la sección espejo de las dos primeras del capítulo de objetivos, donde se planteaba el objetivo general y se elaboraban los específicos.

Es aquí donde hay que debatir qué se ha conseguido y qué no. Cuando algo no se ha conseguido, se ha de justificar, en términos de qué problemas se han encontrado y qué medidas se han tomado para mitigar esos problemas.

6.2. Aplicación de lo aprendido

Aquí viene lo que has aprendido durante el Grado/Máster y que has aplicado en el TFG/TFM. Una buena idea es poner las asignaturas más relacionadas y comentar en un párrafo los conocimientos y habilidades puestos en práctica.

1. a

2. b

6.3. Lecciones aprendidas

Aquí viene lo que has aprendido en el Trabajo Fin de Grado/Máster.

1. a

2. b

6.4. Trabajos futuros

Ningún software se termina, así que aquí vienen ideas y funcionalidades que estaría bien tener implementadas en el futuro.

Es un apartado que sirve para dar ideas de cara a futuros TFGs/TFM.

Apéndice A

Manual de usuario

Bibliografía

- [1] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. Oxford University Press, Inc., 1999.