



**GRADO EN INGENIERÍA EN TECNOLOGÍAS DE LA  
TELECOMUNICACIÓN**

Curso Académico 2018/2019

Trabajo Fin de Grado

**TORNEOS CON DR. SCRATCH**

Autor : Álvaro Fernández Roig

Tutor : Dr. Gregorio Robles



# **Trabajo Fin de Grado**

Torneos con 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

En primer lugar, quiero dar las gracias a mis padres. Siempre he sentido por vuestra parte apoyo incondicional en cada una de mis decisiones. Obviamente, sin vuestro apoyo económico no hubiera podido finalizar la carrera, pero sin duda, tampoco sin vuestro apoyo emocional. Sé que estáis muy orgullosos de mi, siempre me lo habéis hecho ver. Habéis sabido respetarme siempre, estuvierais o no de acuerdo conmigo y eso tiene para mí un valor incalculable.

Por supuesto a mi hermano, gracias por tu paciencia en mis ratos de estudio, tus silencios y, por supuesto, todo tu apoyo y cariño.

A mis amigos, en los que siempre me he podido refugiar, gracias por los buenos momentos pero sobre todo gracias por estar ahí en los malos. A cada uno de vosotros os pertenece un cachito de esto.

A Alberto, Rebeca, Itziar y David, gracias por todo. Sin duda sois lo mejor que me ha pasado en mis años de universidad. Jamás lo hubiera conseguido sin vosotros.

Y a Sandra, por todo. Por tu amor todos estos años. Son tantas cosas que no podría enumerarlas, es una década de momentos. Gracias.



# Resumen

Este proyecto consiste en la creación de un módulo de torneos para la aplicación Dr. Scratch. Dr. Scratch es una aplicación de análisis de proyectos Scratch que evalúa mediante parámetros algunos aspectos del pensamiento computacional.

El módulo se ha creado como una aplicación independiente, utilizando un modelo de datos aislado del resto de entidades de la aplicación original. Permite la creación y administración de torneos por parte de profesores y la participación en ellos de sus alumnos.

Con la creación de este módulo se ha tratado de fomentar una mayor utilización de la funcionalidad original, haciendo que su uso resulte atractivo para los estudiantes y práctico para los profesores.

El módulo de torneos con Dr. Scratch se ha desarrollado utilizando Django, un framework en lenguaje Python. Se ha utilizado una base de datos PostgreSQL, y para el diseño de las pantallas se ha utilizado el framework Bootstrap.



# **Summary**

This project consists in the creation of a new module for the Dr. Scratch application named tournaments of Dr. Scratch. Dr. Scratch is a Scratch project analysis application which evaluates some aspects of computational thinking through parameters.

The module has been created as an independent application, using a data model isolated from the other entities of the original application. It allows the creation and administration of tournaments by teachers and the participation of their students in them.

The creation of this module has tried to encourage greater use of the original functionality, making its use attractive for students and practical for teachers.

The tournament module of Dr. Scratch has been developed using Django, a Python framework. A PostgreSQL database has been used, and the Bootstrap framework has been used to create the templates.



# Índice general

<b>1. Introducción</b>	<b>1</b>
1.1. Contexto . . . . .	1
1.2. Estructura de la memoria . . . . .	2
<b>2. Objetivos</b>	<b>3</b>
2.1. Objetivo general . . . . .	3
2.2. Objetivos específicos . . . . .	3
2.3. Planificación temporal . . . . .	4
<b>3. Estado del arte</b>	<b>7</b>
3.1. Python . . . . .	7
3.2. Django . . . . .	8
3.3. HTML . . . . .	8
3.4. CSS . . . . .	8
3.5. JavaScript . . . . .	9
3.6. jQuery . . . . .	9
3.7. Bootstrap . . . . .	9
<b>4. Diseño e implementación</b>	<b>11</b>
4.1. Arquitectura general . . . . .	11
4.1.1. Cliente . . . . .	12
4.1.2. Servidor . . . . .	12
4.2. Modelo de datos . . . . .	13
4.3. Diseño e implementación por funcionalidad . . . . .	17
4.3.1. Página principal, login y registro de creadores . . . . .	17

4.3.2. Funcionalidades para creadores . . . . .	19
4.3.3. Funcionalidades para participantes . . . . .	33
<b>5. Resultados</b>	<b>39</b>
<b>6. Conclusiones</b>	<b>41</b>
6.1. Consecución de objetivos . . . . .	41
6.2. Aplicación de lo aprendido . . . . .	42
6.3. Lecciones aprendidas . . . . .	42
6.4. Trabajos futuros . . . . .	43
<b>A. Manual de usuario</b>	<b>45</b>
<b>Bibliografía</b>	<b>47</b>

# Índice de figuras

2.1. Planificación temporal . . . . .	4
4.1. Arquitectura general . . . . .	11
4.2. Diagrama de entidad-relación . . . . .	13
4.3. Reto no superado . . . . .	15
4.4. Reto superado . . . . .	15
4.5. Diagrama de estados de una partida . . . . .	16
4.6. Página principal de Dr. Scratch . . . . .	17
4.7. Página principal de torneos con Dr. Scratch: Menú de login . . . . .	18
4.8. Página principal de torneos con Dr. Scratch: Registro de creadores . . . . .	19
4.9. Página principal de creadores . . . . .	20
4.10. Página principal de la administración de torneos . . . . .	20
4.11. Listado de retos . . . . .	21
4.12. Detalle de un reto . . . . .	22
4.13. Creación de un equipo . . . . .	23
4.14. Ver participantes de un equipo . . . . .	24
4.15. Creación de un torneo . . . . .	25
4.16. Detalle de un torneo . . . . .	26
4.17. Validación manual: listado de torneos . . . . .	27
4.18. Validación manual: listado de equipos . . . . .	28
4.19. Validación manual: ventana modal . . . . .	28
4.20. Progreso de los torneos . . . . .	30
4.21. Registrar a nuevos participantes . . . . .	31
4.22. Selección de equipo . . . . .	33

4.23. Página principal de juego . . . . .	34
4.24. Mensajes de la página de resultados . . . . .	35
4.25. Página de resultados: Nivel básico, medio y alto . . . . .	36
4.26. Torneo completado . . . . .	37
5.1. Respuestas del cuestionario de creadores . . . . .	40
5.2. Respuestas del cuestionario de participantes . . . . .	40

# **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 con 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 puntuá 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.

La participación de los alumnos en los torneos con Dr. Scratch se realiza a través de la formación de equipos, fomentando de este modo el uso del aprendizaje cooperativo. Mediante el uso del aprendizaje cooperativo, los alumnos trabajan en grupos reducidos, de este modo intervienen no solo en su propio aprendizaje sino en el del resto de miembros del equipo, buscando un resultado beneficioso tanto para ellos mismos como para el resto de sus compañeros [5].

Así mismo, los torneos con Dr. Scratch permiten a los alumnos la autoevaluación en grupo de sus propios proyectos Scratch. A través de la autoevaluación, los alumnos participan activamente en el proceso de aprendizaje, aumentando su responsabilidad en el mismo y ligando el aprendizaje a sus intereses, capacidades y motivaciones [7].

## 1.2. Estructura de la memoria

A continuación se detalla la estructura de la memoria.

- **Capítulo 1: Introducción.** En este capítulo se desarrolla brevemente el contexto en el que se ha realizado el proyecto.
- **Capítulo 2: Objetivos.** En este capítulo se enumeran los objetivos que se pretenden alcanzar con la realización del proyecto.
- **Capítulo 3: Estado del arte.** En este capítulo se realiza una descripción de alto nivel de las diferentes tecnologías utilizadas en la realización del proyecto.
- **Capítulo 4: Diseño e implementación.** En este capítulo se describe el desarrollo del proyecto incluyendo su arquitectura y modelo de datos.
- **Capítulo 5: Resultados.** En este capítulo se describen y analizan los resultados obtenidos.
- **Capítulo 6: Conclusiones.** En este capítulo se indican las conclusiones a las que se ha llegado una vez finalizado el proyecto.

# **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.
- Fomentar el trabajo cooperativo asociando las puntuaciones obtenidas a todos los participantes de un equipo de manera conjunta sin independizar entre sus miembros.

- 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

Sin duda la planificación del proyecto ha sido muy atípica. Gregorio, mi tutor, me presentó la idea del proyecto un lejano 18 de marzo del 2014. A lo largo de estos años he realizado, sin éxito, varios intentos de desarrollar el proyecto. El más fructífero en el año 2017, donde llegué a realizar parte de la funcionalidad y a reunirme nuevamente con Gregorio. Lamentablemente este intento también quedó en nada.

Finalmente en octubre de 2019 retomé, esta vez con plena dedicación, el desarrollo de la aplicación. De lo realizado en 2017 salvé parte del modelo pero comencé a realizar todo de nuevo, es por esto que el comienzo de la planificación es el 1 de octubre de 2019.

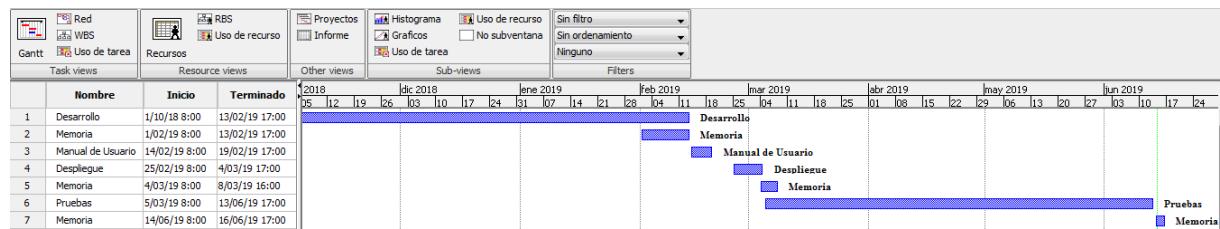


Figura 2.1: Planificación temporal

La duración de las jornadas ha cambiado a lo largo de la duración del proyecto.

- Del 1 de octubre al 31 de enero: 4 horas al día en desarrollo.
- Del 1 de febrero al 13 de febrero: 4 horas al día en desarrollo y 4 horas al día en la memoria.
- Del 14 de febrero al 19 de febrero: 8 horas al día en el manual de usuario.
- Del 25 de febrero al 4 de marzo: 8 horas al día en el despliegue.

- Del 4 de marzo al 20 de marzo: 8 horas al día en la memoria.
- Del 5 de marzo al 13 de junio: pruebas de usuario, asistencia en remoto.
- Del 14 de junio al 16 de junio: 8 horas al día en la memoria. Inclusión de los resultados de las pruebas.



# Capítulo 3

## Estado del arte

A continuación se detallan las tecnologías utilizadas en el desarrollo del proyecto.

### 3.1. Python

Python<sup>1</sup> es un lenguaje de programación desarrollado a principios de los noventa por Guido van Rossum [1].

Se trata de un lenguaje de programación de código abierto cuyas principales características son:

- Es un lenguaje interpretado, por lo que a diferencia de otros lenguajes como Java o C no necesita compilación. Esto lo convierte en un lenguaje con un ciclo de desarrollo más rápido que el de los lenguajes compilados, dado que no requiere la compilación del código fuente cada vez que éste se cambia. Sin embargo, ser un lenguaje interpretado hace que su ejecución sea más lenta que la de los lenguajes compilados.
- Es un lenguaje multiplataforma. Ser un lenguaje de programación interpretado hace que Python pueda ser ejecutado en cualquier sistema operativo que disponga de intérprete.
- Es un lenguaje multiparadigma. Permite diferentes paradigmas de programación: programación orientada a objetos, programación imperativa y programación funcional.

---

<sup>1</sup><https://www.python.org>

## 3.2. Django

Django<sup>2</sup> es un framework de desarrollo web creado en el año 2003 por Adrian Holovaty y Simon Willison. En un principio, Django fue creado para administrar una serie de páginas web relacionadas con noticias, pero posteriormente fue lanzado en internet utilizando un modelo de código abierto.

Django presenta un mapeo objeto-relacional (ORM) que actúa entre el modelo de datos y el motor de base de datos y que permite que la modificación del motor de base de datos se realice simplemente modificando la configuración en su archivo de ajustes.

Así mismo, Django proporciona un servidor web ligero que puede usarse para desarrollo y pruebas, así como mensajes de error detallados en el modo de depuración, que hacen que el desarrollo y la corrección de errores sean sencillos.

Django también permite crear sitios web multilenguaje a través de su sistema de internacionalización, que simplifica la traducción de la interfaz [4].

## 3.3. HTML

HTML son las siglas de HyperText Markup Language, se trata del lenguaje estándar utilizado para la creación de páginas web. Fue creado por Tim Berners-Lee en los años ochenta. Es un lenguaje de marcas, esto quiere decir que para indicar la estructura del documento se utilizan etiquetas. Al ser un estándar, todos los navegadores leen e interpretan el lenguaje HTML y lo presentan de un modo similar.

## 3.4. CSS

CSS son las siglas de Cascading Style Sheets, es un lenguaje que permite establecer la apariencia de un documento escrito mediante un lenguaje de marcas. CSS permite asociar reglas a los elementos que aparecen en una página web, las reglas indican como debe representarse el contenido de esos elementos [6].

---

<sup>2</sup><https://www.djangoproject.com>

## 3.5. JavaScript

JavaScript es un lenguaje de programación creado por Brendan Eich, un trabajador de Netscape, en el año 1995. Se trata de un lenguaje que permite añadir interactividad en páginas web, pudiendo ejecutarse en el navegador del usuario.

Se trata de un lenguaje interpretado y orientado a objetos. Es también un lenguaje débilmente tipado, lo que significa que las variables no tienen que tener un tipo especificado [3].

## 3.6. jQuery

jQuery<sup>3</sup> es una librería de Javascript en código abierto creada en el año 2006 por John Resig, que simplifica la relación entre el documento HTML y Javascript.

jQuery simplifica la manipulación del documento HTML, el manejo de eventos, la realización de animaciones, las interacciones Ajax y el desarrollo de código Javascript aplicable a múltiples navegadores [2].

Así mismo, su amplia comunidad hace que jQuery esté siempre depurada y optimizada y que estén disponibles multitud de plugins, facilitando su uso.

## 3.7. Bootstrap

Bootstrap<sup>4</sup> es un framework font-end que permite la construcción de páginas web adaptativas. Se trata de un software de código abierto creado en el año 2011 por Mark Otto y Jacob Thornton, empleados de Twitter.

Bootstrap ha evolucionado desde ser un proyecto enteramente basado en CSS hasta incluir múltiples plugins Javascript e iconos junto con formularios y botones. Presenta una cuadrícula de doce columnas y 940px de ancho [8].

La creación de páginas web mediante Bootstrap se simplifica al disponer de elementos predefinidos fácilmente incluibles en cualquier página web, tales como menús desplegables, botones, iconos, ventanas emergentes, etc.

---

<sup>3</sup><https://jquery.com>

<sup>4</sup><https://getbootstrap.com>



# Capítulo 4

## Diseño e implementación

A continuación se detalla el módulo de torneos con Dr. Scratch, incuyendo su arquitectura y su modelo de datos.

### 4.1. Arquitectura general

El módulo de torneos está desarrollado 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, view<sup>1</sup>.

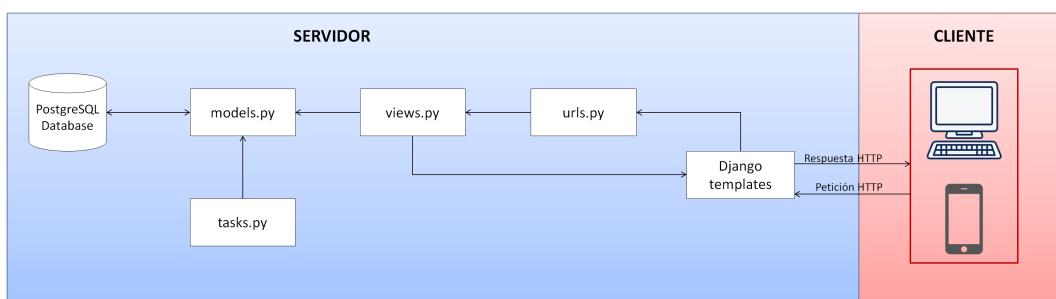


Figura 4.1: Arquitectura general

<sup>1</sup><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 encuentra el navegador web del usuario. El usuario interactuará con la aplicación a través de peticiones HTTP que serán interpretadas en el lado del servidor. La creación de las plantillas con el framework Bootstrap permite que las pantallas sean responsive, es decir, que se ajusten automáticamente a la resolución de pantalla del dispositivo en el que se muestran. De este modo, las diferentes pantallas pueden visualizarse sin problemas tanto desde ordenadores como desde dispositivos móviles o tablets.

### 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. Estas plantillas son archivos en formato HTML en los que se muestra la información de la respuesta recibida desde la vista. En este caso, se ha utilizado el framework Bootstrap para el desarrollo del formato de las plantillas.

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 Tasks<sup>2</sup> para la creación de procesos asíncronos. La función a ejecutar se encuentra definida en el archivo

---

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

tasks.py que actúa como una vista.

## 4.2. Modelo de datos

Para la creación del módulo de torneos con 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:

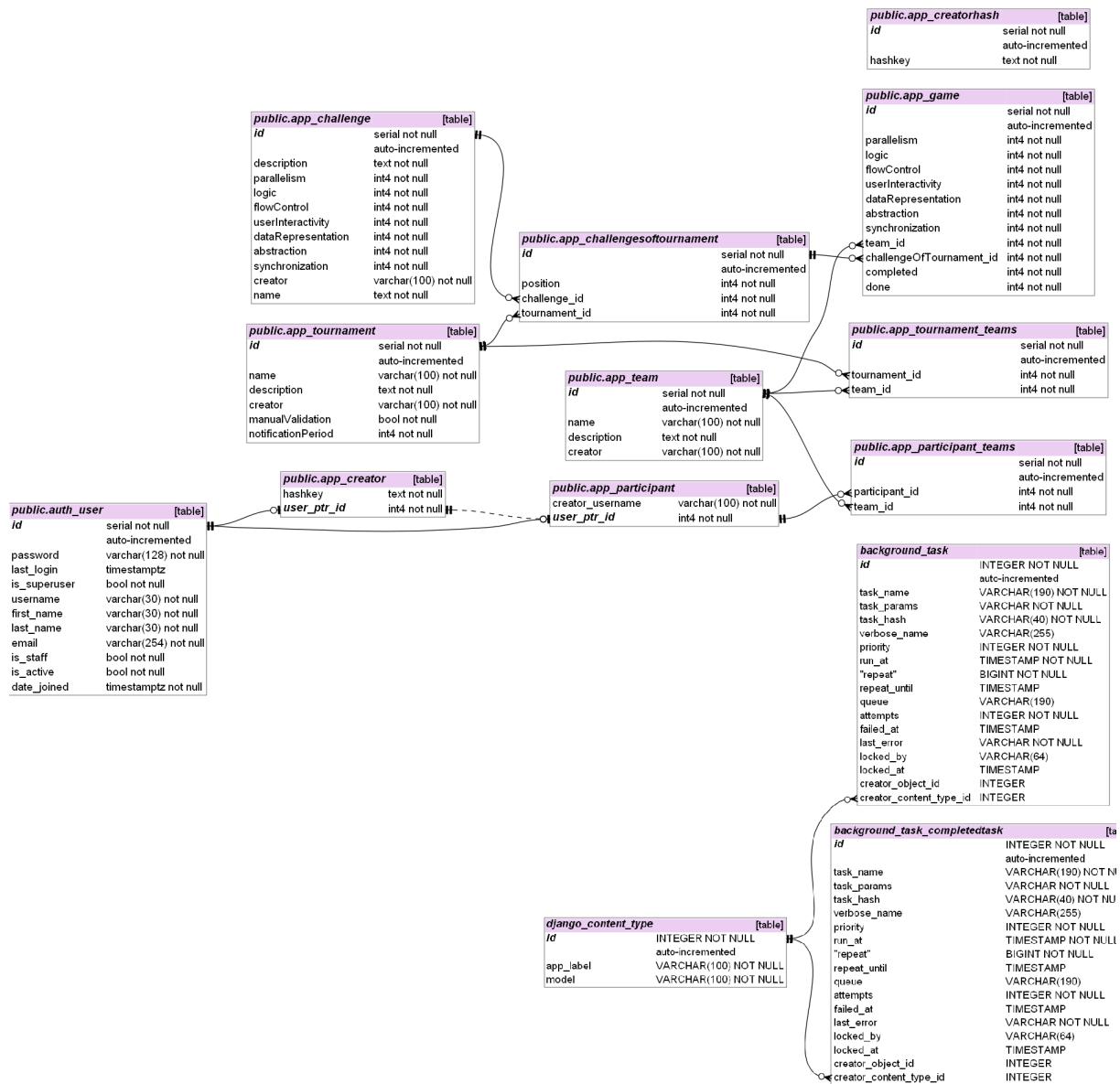


Figura 4.2: Diagrama de entidad-relación

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 con 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 4.3, 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 4.3 y 4.4, 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.

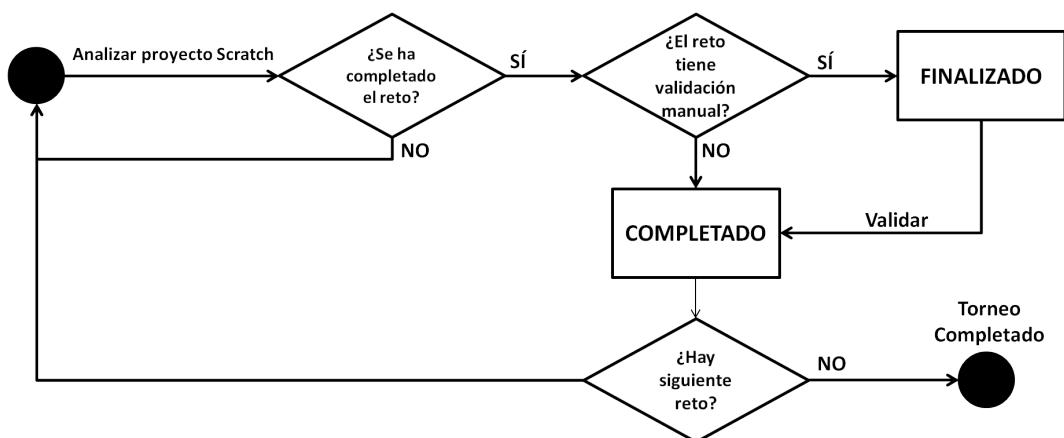


Figura 4.5: Diagrama de estados de una partida

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 con 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/](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.



Figura 4.7: Página principal de torneos con 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.



Figura 4.8: Página principal de torneos con 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.

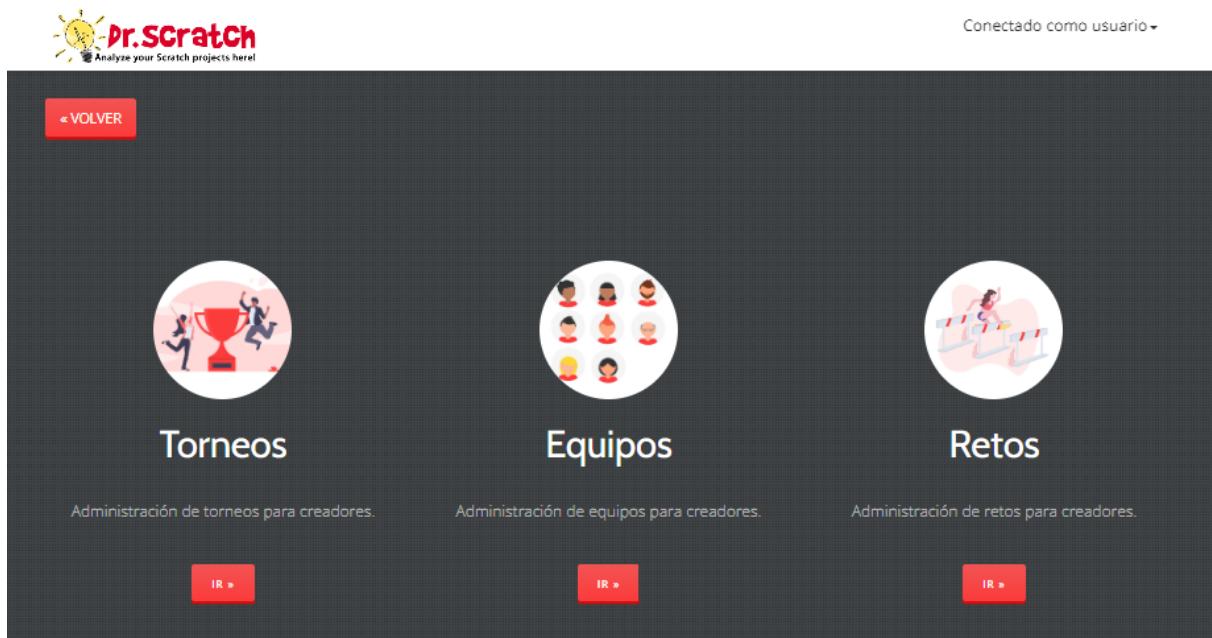


Figura 4.10: Página principal de la administración de torneos

## 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.

Conectado como alvaro▼

« Volver

Retos creados por alvaro

<b>Reto 0</b> Reto valores cero	Detalle	Editar	Borrar
<b>Reto 1</b> Valores: 0011100	Detalle	Editar	Borrar
<b>Reto 2</b> Valores: 1322112	Detalle	Editar	Borrar

« 1 2 3 4 5 »

NUEVO RETO

Programamos

Universidad Rey Juan Carlos

LibreSoft

GOBIERNO DE ESPAÑA

MINISTERIO DE COMUNICACIONES

FECYT

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.

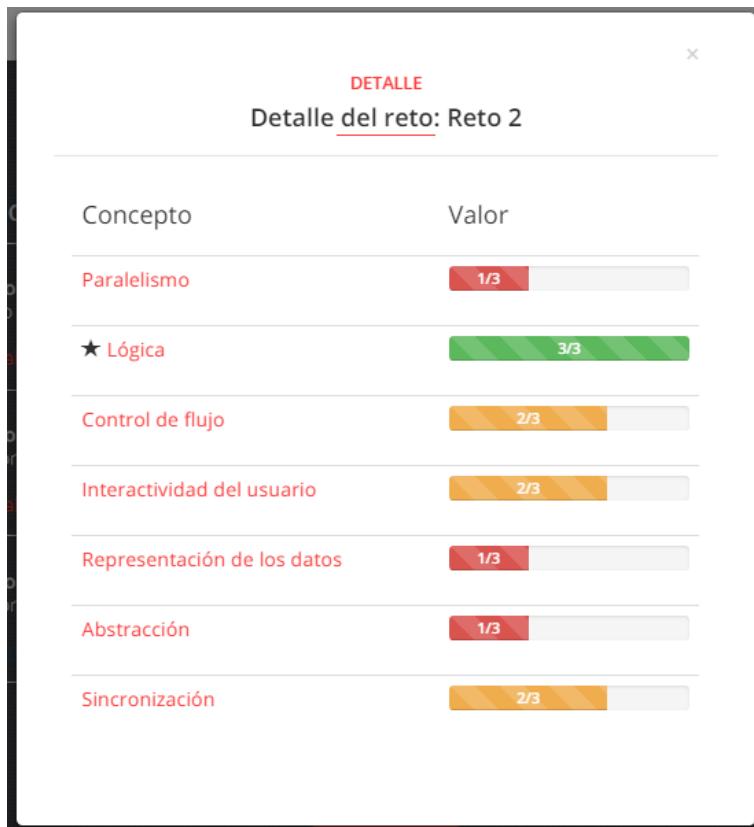


Figura 4.12: Detalle de un reto

## 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 4.11, 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.

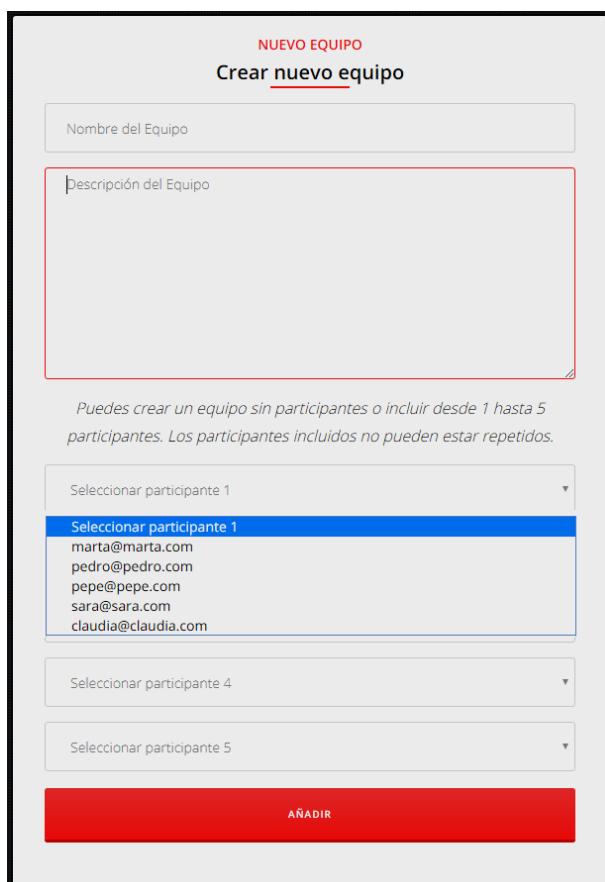


Figura 4.13: Creación de un equipo

El listado de los participantes de un equipo se muestra 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, no pudiendo 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 de la figura 4.11, 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 valide 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

**AÑADIR**

Figura 4.15: Creación de un torneo

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 un torneo.

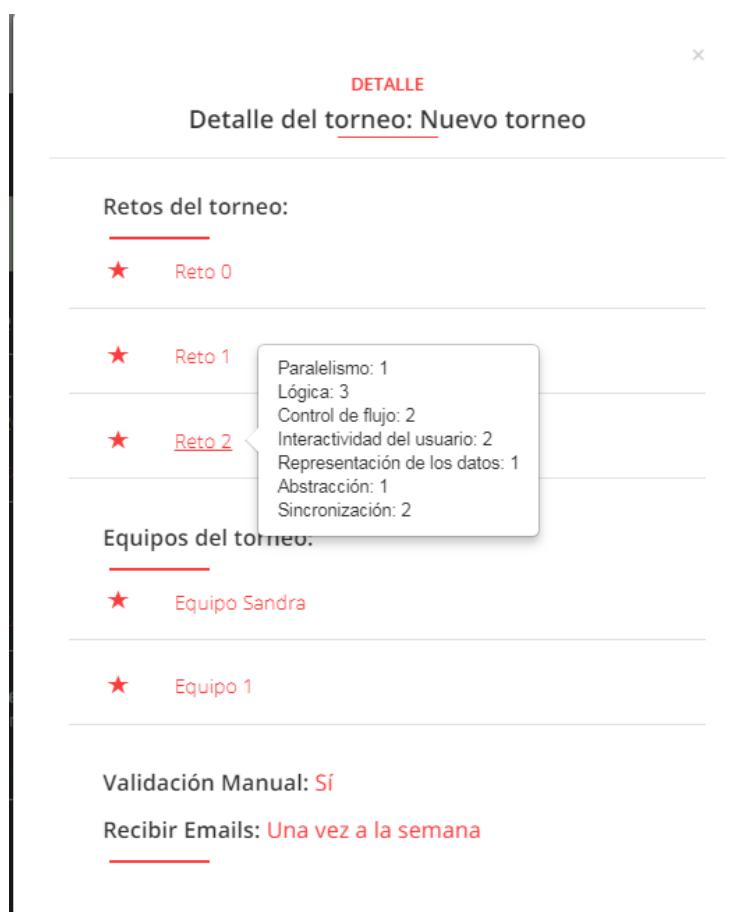


Figura 4.16: Detalle de un torneo

#### 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 4.5), 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.



Figura 4.17: Validación manual: listado de torneos

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 4.18: Validación manual: listado de equipos

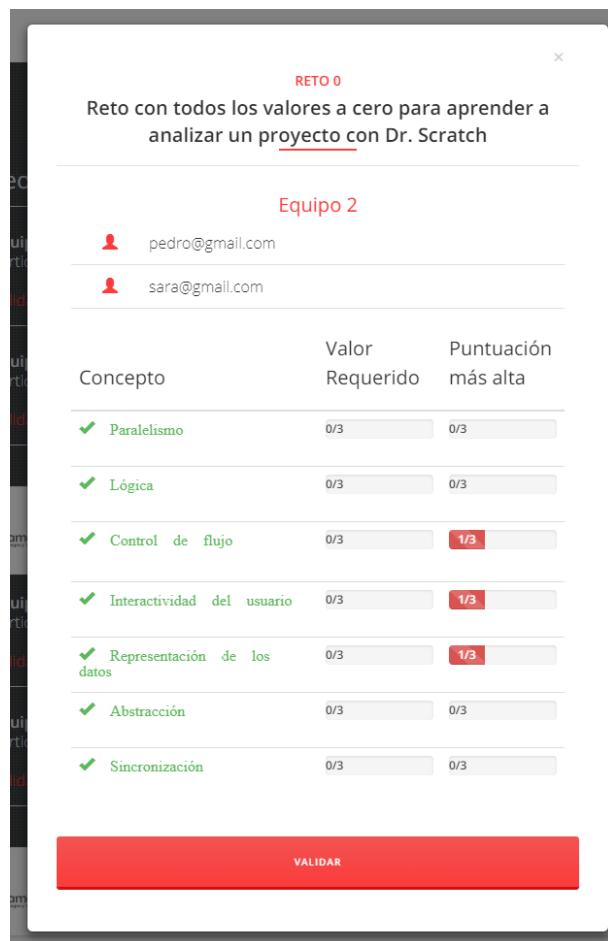


Figura 4.19: Validación manual: ventana modal

#### 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 4.11.

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 4.16).

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 Wizard<sup>3</sup>. En el archivo easyWizard.css se han adaptado los colores para que coincidieran con los tonos usados en

---

<sup>3</sup><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.

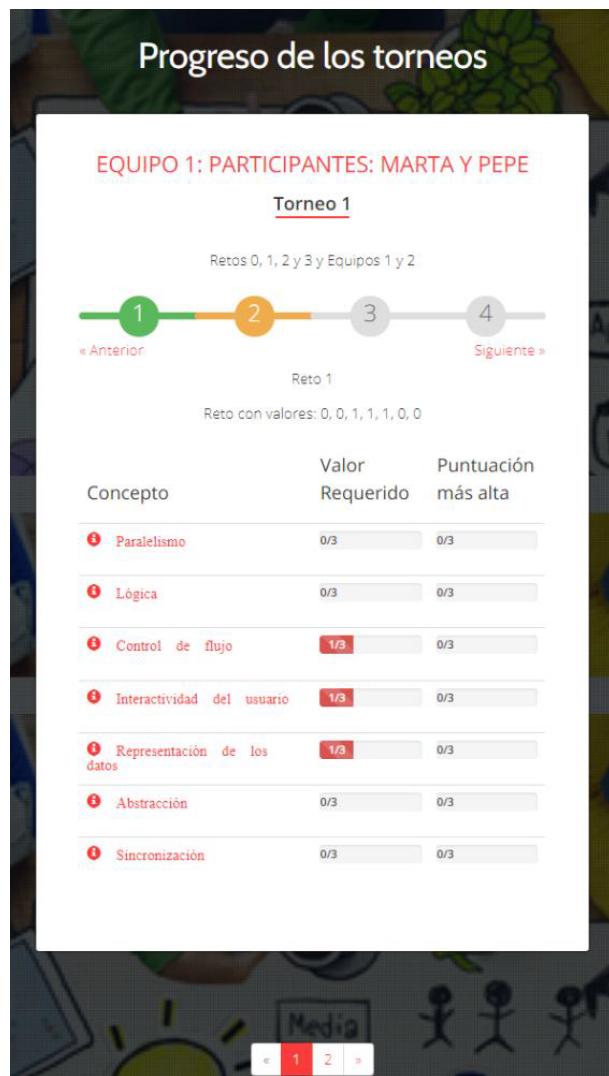


Figura 4.20: Progreso de los torneos

#### 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.

The screenshot shows the 'Registrar nuevos participantes en los torneos de Dr. Scratch' (Register new participants in Dr. Scratch tournaments) page. At the top right, it says 'Conectado como alvaro'. Below the title, there's an info message: 'Info: La contraseña de los usuarios registrados será el usuario o email introducido. El usuario y la contraseña serán iguales.' (Info: The password of the registered users will be the user or email entered. The user and password will be the same). The page lists three options:

1. Puedes incluir hasta 20 participantes al mismo tiempo con un [fichero CSV](#)
2. Puedes incluir hasta 20 participantes al mismo tiempo con un [fichero TXT](#)
3. Puedes incluir hasta 20 participantes al mismo tiempo llenando el siguiente formulario. Pulsa + para incluir múltiples participantes.

For option 3, there is a text input field labeled 'Nombre de usuario / Correo electrónico' with a placeholder '#1' and a red '+' button to its right. Below the input field is a red 'REGISTRAR' button. At the bottom of the page, there are logos for Programamos, Universidad Rey Juan Carlos, LibreSoft, and FECYT.

Figura 4.21: 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.readlines().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(){
    $('.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.

### 4.3.3. Funcionalidades para participantes

Los participantes cuentan con una única función, que es la de jugar. En su caso no existirá un menú con secciones, sino que al acceder al módulo de torneos se mostrará la página de selección de equipo.

La página de selección de equipo consta de un panel central en el que se muestra el nombre, la descripción y los participantes del equipo. En rojo se resalta el nombre del participante que ha iniciado sesión.

En la parte inferior aparecerá un paginador si el participante está incluido en más de un equipo, en este caso cada página corresponderá a un equipo.

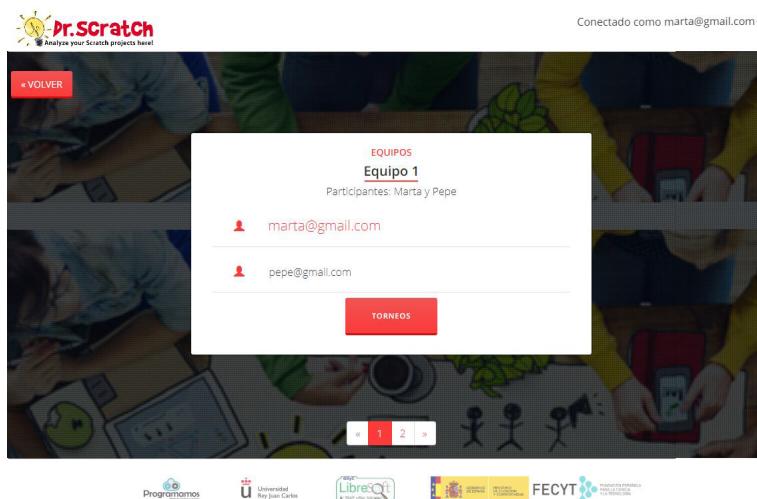


Figura 4.22: Selección de equipo

Pulsando el botón *Torneos*, el participante seleccionará el equipo con el que quiere participar.

Al seleccionar el equipo, se accede a la página principal de juego. A su vez, si el equipo seleccionado por el jugador estuviera incluido en más de un torneo, a través de un paginador podría cambiar de torneo en el que participar.

La página principal de juego consta de dos paneles.

A la izquierda se muestra un panel con el progreso de la partida del jugador. Este panel tiene el mismo formato que el mostrado en la figura 4.20. Se mantiene el mismo código de colores, pero en este caso, los participantes no pueden pasar al siguiente reto hasta que no hayan superado el reto en curso, por lo que el botón *Siguiente* no se mostrará. Sin embargo, si

el participante ha superado el reto sí dispondrá del botón *Anterior* para permitirle mejorar la puntuación obtenida en el reto superado.

A la derecha, se muestra el panel de análisis de proyectos Scratch. Este panel coincide con el panel original que se encuentra en la página principal de Dr. Scratch. Analiza proyectos Scratch en versión 3.0 a través de su URL o adjuntándolos mediante un archivo de tipo *File*.

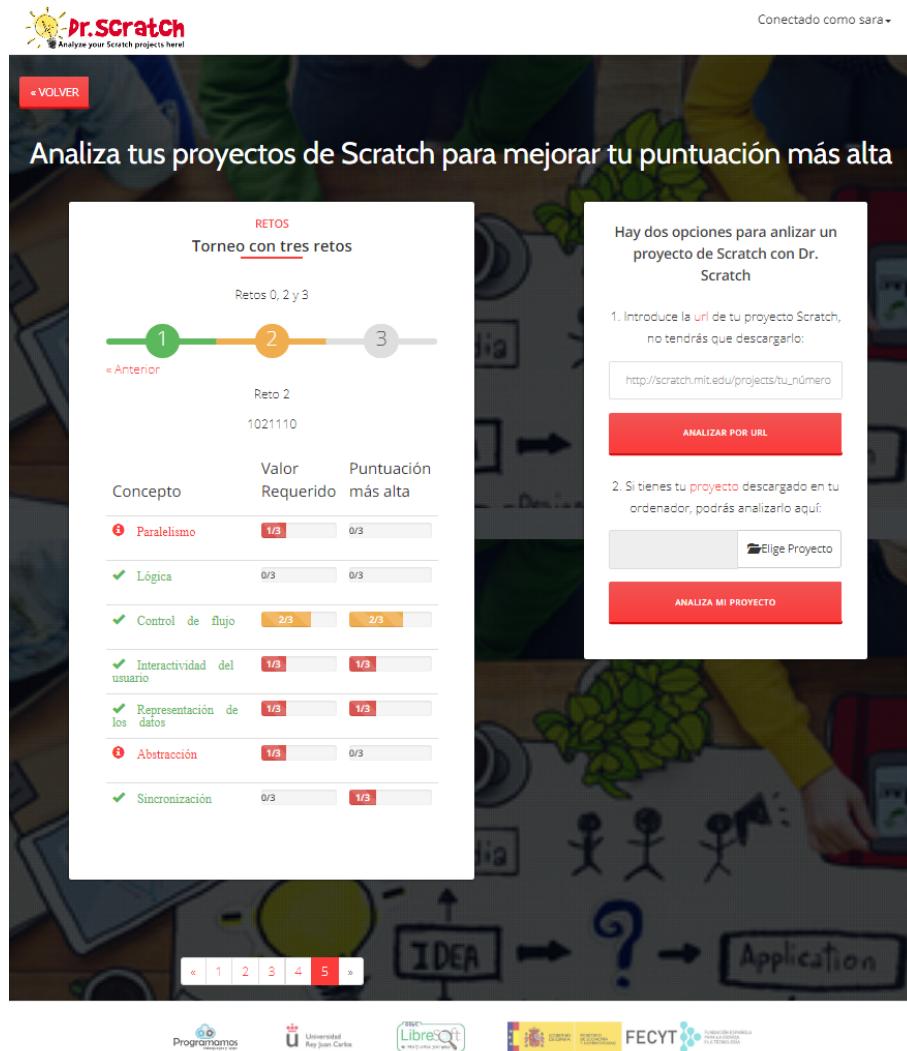


Figura 4.23: Página principal de juego

Se ha modificado la función de análisis de los proyectos Scratch para incluir la evaluación de las partidas en los torneos. La función de evaluación de la partida es la siguiente:

```

game.done = True
if (tournament.manualValidation):
    game.completed = False
else:
    game.completed = True
for key, value in d["mastery"].items():
    if (key != 'points' and key != 'maxi'):
        if (value > getattr(game, trans_keys[key])):
            setattr(game, trans_keys[key], value)
        if (getattr(game, trans_keys[key]) < getattr(challenge, trans_keys[key])):
            game.completed = False
            game.done = False
        game_value = getattr(game, trans_keys[key])
        total_points = total_points + game_value
        d["mastery"][key] = game_value
d["mastery"]["points"] = total_points

```

La función comprueba uno por uno los parámetros analizados, por un lado comprueba si el valor del parámetro analizado es mayor que el existente en la partida y si es así, lo sobrescribe. Por otro, comprueba si algún valor de la partida es menor a su correspondiente valor en el reto, en cuyo caso establece la partida como no finalizada. La partida solo se establece como completada si la validación es automática.

Al analizar un proyecto Scratch la aplicación redirige automáticamente a la página de resultados. Esta página de resultados es una modificación de la página del resultado del análisis, original de Dr. Scratch.

Se ha incluido un panel superior que indica a los usuarios si han superado o no el reto a través de un mensaje. En este panel superior se incluye el botón *Volver* para regresar a la página principal de juego y un enlace al tutorial de Dr. Scratch mediante el botón *Aprende cómo*.

Los mensajes mostrados pueden ser tres, un mensaje que indica que se ha completado el reto, un mensaje que indica que no se ha completado el reto y un mensaje que indica que la puntuación debe ser validada por el profesor para completar el reto.

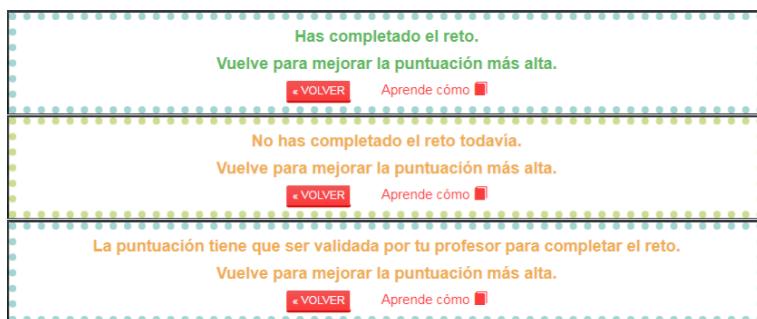


Figura 4.24: Mensajes de la página de resultados

Se ha modificado también el panel principal, situado a la derecha, en el que se muestran, en lugar del resultado del análisis, los valores de los parámetros del reto y la puntuación más alta de la partida del equipo. De la pantalla del resultado del análisis de Dr. Scratch se han mantenido los paneles situados a la izquierda indicando al participante el nivel y los detalles a corregir del resultado de su análisis. También se ha mantenido el formato, variando la apariencia de la pantalla de resultados entre los tres niveles posibles: básico, medio y alto.

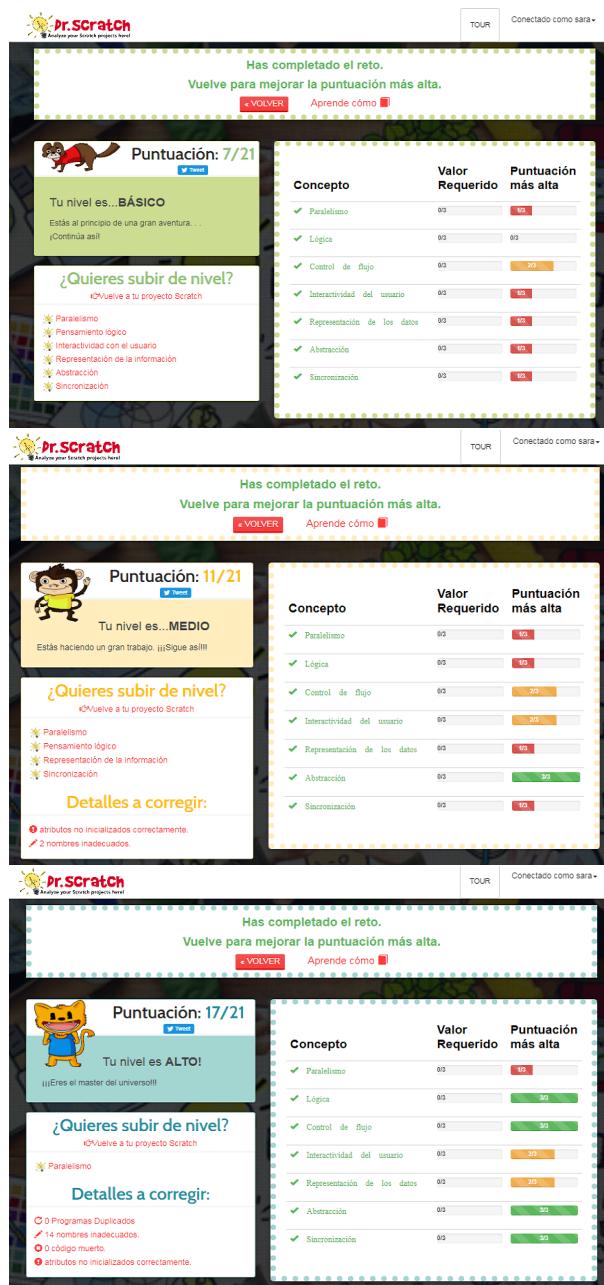


Figura 4.25: Página de resultados: Nivel básico, medio y alto

Por último, cuando un participante haya completado el torneo, en la página principal de juego aparecerán los iconos de los distintos retos en verde y un mensaje que indique que el torneo se ha completado.



Figura 4.26: Torneo completado



# Capítulo 5

## Resultados

Para comprobar la viabilidad del proyecto, se ha probado la aplicación en un entorno real. Se ha contado con la ayuda de Jorge, un profesor del C.E.I.P. Lope de Vega de Madrid, que ha realizado varios torneos con alumnos de 4º de primaria.

Para que la aplicación estuviera disponible, primero se ha realizado un despliegue a través de Azure, tanto de la aplicación web como de la base de datos.

Para la base de datos se ha optado por crear un recurso de Servidor de Azure Database for PostgreSQL. Originariamente se contaba con una instancia de base de datos de MySQL por lo que se ha modificado el atributo DATABASES del archivo `settings.py`.

```
'ENGINE': 'django.db.backends.postgresql_psycopg2'
```

Para el despliegue de la aplicación se ha creado un recurso App Service gratuito de un núcleo de 1GB de almacenamiento.

Para poder desplegar correctamente la aplicación en el recurso creado, se ha realizado una serie de modificaciones entre las que se incluye la creación de un archivo `requirements.txt` que permite, mediante la consola de Kudu de Azure, la instalación de los plugins de Python necesarios para el funcionamiento de la aplicación, tales como Hairball o Kurt.

Así mismo, se incluyen los archivos `web.config` y `run_waitress_server.py` para la correcta publicación de la aplicación incluyendo sus archivos estáticos<sup>1</sup>.

---

<sup>1</sup>[https://blogs.msdn.microsoft.com/azureosssds/2017/09/01/django-app-with-  
httpplatformhandler-in-azure-app-services-windows](https://blogs.msdn.microsoft.com/azureosssds/2017/09/01/django-app-with-httpplatformhandler-in-azure-app-services-windows)

Una vez desplegada la aplicación, Jorge y sus alumnos utilizaron el módulo de torneos con Dr. Scratch para crear torneos y evaluar sus proyectos Scratch. Estas pruebas duraron tres meses y finalmente tanto Jorge como sus alumnos contestaron a un breve cuestionario para ofrecer su impresión sobre la aplicación. Los resultados de los cuestionarios son los siguientes.



Figura 5.1: Respuestas del cuestionario de creadores

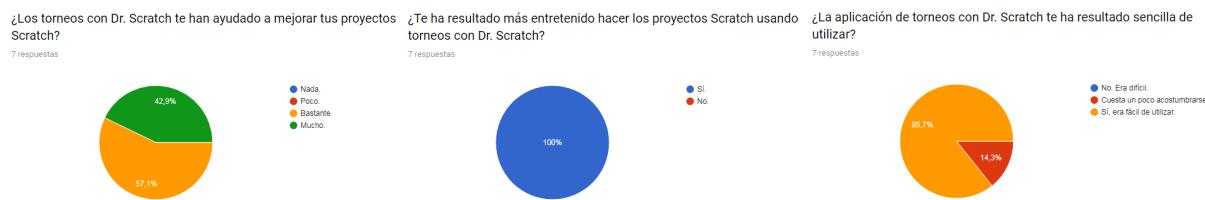


Figura 5.2: Respuestas del cuestionario de participantes

# Capítulo 6

## Conclusiones

### 6.1. Consecución de objetivos

Pese a haber realizado las pruebas en un entorno reducido y disponer de escasa retroalimentación, los resultados se consideran satisfactorios.

El principal objetivo, que era proporcionar una aplicación que resultara atractiva a los alumnos y fomentara el uso de Dr. Scratch, parece haberse conseguido, ya que a la pregunta *¿Te ha resultado más entretenido hacer los proyectos Scratch usando torneos con Dr. Scratch?* el 100 % de los alumnos ha respondido que sí.

Por otro lado, los estudiantes han considerado que la aplicación es suficientemente intuitiva, un 87.5 % consideran que era fácil de utilizar. Esta respuesta resulta especialmente significativa si tenemos en cuenta que se trata de alumnos de 4º de primaria, de edades en torno a 9 o 10 años. Resulta interesante que el rango de edades de uso de la aplicación sea lo más amplio posible.

Y por último, los alumnos consideran que la aplicación les ha ayudado a mejorar sus proyectos Scratch, contestando con *Bastante* (57,1 %) o *Mucho* (42,9 %). Por lo tanto, en base a las respuestas obtenidas consideramos que de parte de los estudiantes se han cumplido a grandes rasgos los objetivos establecidos en una primera fase del proyecto.

En cuanto a los resultados de la encuesta para profesores, pese a contar con un solo caso de uso, se ha obtenido información suficiente para la mejora de la aplicación en trabajos futuros.

Gracias a la información proporcionada por Jorge, el profesor encargado de realizar las pruebas, se ha llegado a la conclusión de que la aplicación, en su estado actual, permite el control por parte del profesor de los torneos y equipos creados y de los proyectos realizados por los

alumnos. Sin embargo, la administración y el control del registro y del perfil de los participantes y un histórico de su participación en los distintos torneos son algunos de los aspectos a mejorar.

También se llega a la conclusión de que el menú de creadores, más complejo y amplio que el de participantes, no resulta del todo intuitivo por lo que es otro punto a mejorar.

## 6.2. Aplicación de lo aprendido

A lo largo del proyecto se han aplicado los conocimientos de programación de manera general, y en particular los conocimientos previos obtenidos en las tecnologías utilizadas en el proyecto, tales como Python y Django o HTML, CSS, JavaScript, etc.

Las asignaturas que más han contribuido en el conocimiento previo a la hora de realizar este proyecto son las siguientes:

- Fundamentos de programación. Conceptos generales de la programación orientada a objetos.
- Aplicaciones telemáticas. Conocimiento de HTML, CSS, JavaScript, JQuery y Bootstrap.
- Servicios y aplicaciones telemáticas y Desarrollo de aplicaciones telemáticas. Creación de aplicaciones en lenguaje Python con el uso de Django.

## 6.3. Lecciones aprendidas

Entre las lecciones aprendidas en este proyecto, sin duda destacaría dos.

Por un lado, el aprendizaje del lenguaje de programación Python y el framework Django. Con una base ya obtenida a lo largo de la titulación de grado he podido afrontar con garantías la realización del proyecto y aprender con mayor profundidad el uso tanto de Python como de Django.

Por otro, el uso de Azure, la creación de recursos y el despliegue de aplicaciones en la nube. Esto lo considero de especial relevancia para mi carrera profesional dado que en estos momentos se encuentra orientada hacia el desarrollo de aplicaciones con arquitectura de microservicios.

## 6.4. Trabajos futuros

Dadas las conclusiones obtenidas tras la realización de las pruebas, existen una serie de aspectos a mejorar en la aplicación actual.

- Simplificación de contenidos y mejora estética para hacer más intuitiva la aplicación sin que esto conlleve una pérdida en la información proporcionada.
- Inclusión de imágenes en los perfiles de los participantes, permitiendo que los participantes tengan asociado un personaje con una imagen y así aumentar el carácter lúdico de la aplicación.
- Incluir un menú de administración de participantes para el creador en el que pueda eliminar participantes registrados, modificar sus contraseñas.
- Un apartado para creadores que incluya un registro histórico de las partidas de cada participante incluyendo las URLs de los proyectos evaluados en esos torneos.
- Una opción para creadores que permita, sin necesidad de registrar otro correo electrónico, acceder él mismo como participante y de este modo comprobar el menú al que pueden acceder los alumnos.
- Hacer parametrizable el número de alumnos por equipo y el número de equipos por torneo.



# Apéndice A

## Manual de usuario

Para utilizar la aplicación de Dr. Scratch que incluye el módulo de torneos es necesario realizar los siguientes pasos:

- Instalar Python 2.7<sup>1</sup>.
- Descargar el código de la aplicación<sup>2</sup>.
- Modificar los valores del parámetro DATABASES en el archivo settings.py para configurar la base de datos local del usuario.
- Abrir una consola en el directorio raíz con el código descargado.
- Ejecutar el comando `python -m pip install -r requirements.txt`.
- Ejecutar el comando `python manage.py migrate`.
- Ejecutar el comando `python manage.py runserver`.
- Abrir el navegador y escribir en la barra de dirección la siguiente URL: `http://localhost:8000`.

El manual de usuario completo de la aplicación se encuentra en la siguiente dirección:

[https://github.com/alvarofdez/drScratch/blob/master/Torneos\\_Dr\\_Scratch\\_Manual\\_De\\_Usuario.pdf](https://github.com/alvarofdez/drScratch/blob/master/Torneos_Dr_Scratch_Manual_De_Usuario.pdf)

---

<sup>1</sup><https://www.python.org/download/releases/2.7>

<sup>2</sup><https://github.com/alvarofdez/drScratch>



# Bibliografía

- [1] T. Donaldson. *Python*. Peachpit Press, 2008.
- [2] J. C. Experts. *JQuery Cookbook*. O'Reilly Media, Inc., 2009.
- [3] D. Flanagan. *Javascript: The Definitive Guide, Fifth Edition*. O'Reilly Media, Inc., 2005.
- [4] S. Jaiswal and R. Kumar. *Learning Django Web Development*. Packt Publishing, 2015.
- [5] D. W. Johnson, R. T. Johnson, and E. J. Holubec. *Cooperative Learning in the Classroom*. Association For Supervision and Curriculum Development, Virginia, 1994.
- [6] R. Larsen. *Beginning HTML and CSS*. Wrox, 2013.
- [7] D. R. Muñoz and P. Troncoso. Autoevaluación de los alumnos: una estrategia participativa orientada al “aprender a valorar”. *REXE*, 2(4):111–120, 2003.
- [8] J. Spurlock. *Bootstrap*. O'Reilly Media, Inc., 2013.