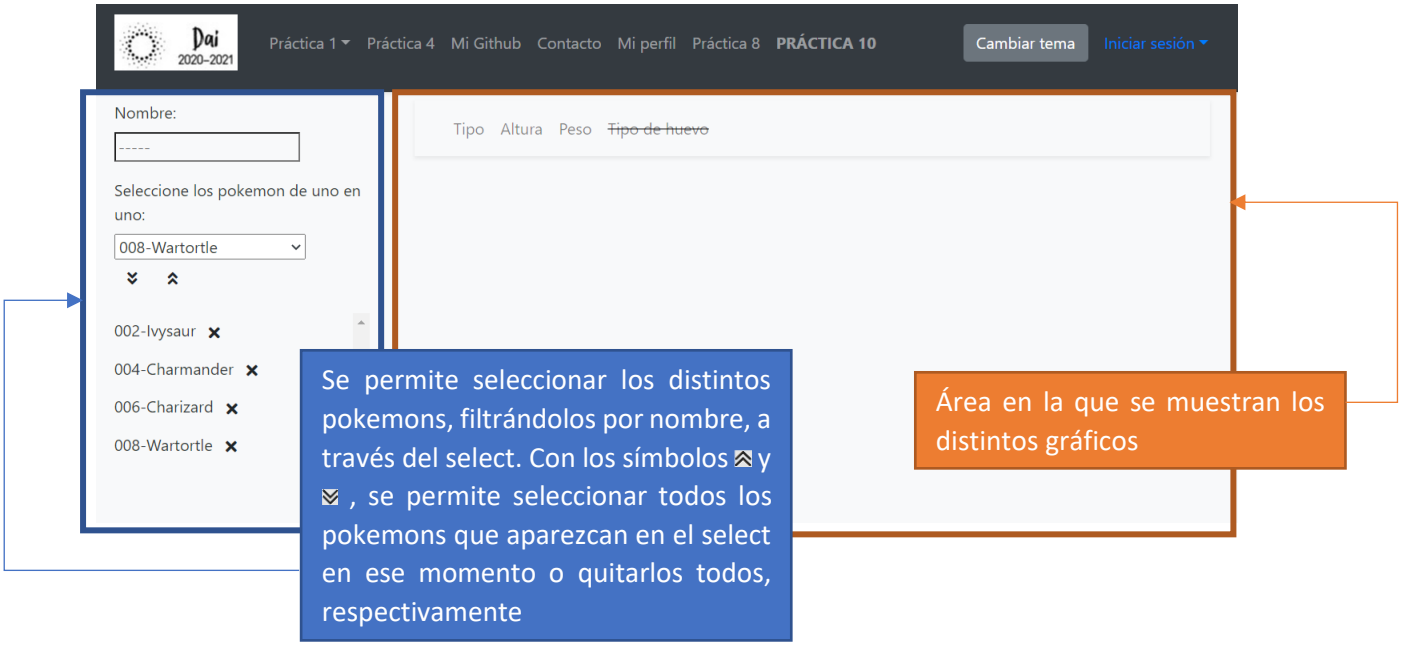
Práctica 10 – Práctica libre sobre Javascript

En esta práctica se añade cierta funcionalidad al frontend de la práctica basada en Flask relativa a la base de datos del juego Pokemon GO. En cuanto a la funcionalidad añadida, se resume en dos aspectos. Por un lado, un análisis descriptivo de los datos de la base de datos citada, y, por otro, se desarrolla un sistema de notificaciones, de forma que se avise al usuario a la hora a la que "sale" un pokemon.

Análisis descriptivo de los datos

Se utiliza la librería de código abierto Chart.js, que permite generar distintos tipos de gráficos para mostrar los datos de manera descriptiva. Además, para seleccionar los distintos pokemons sobre los cuales se desea mostrar los datos de manera gráfica, así como para obtener los propios datos, se utiliza Ajax para llamar a la api desarrollada en prácticas anteriores.

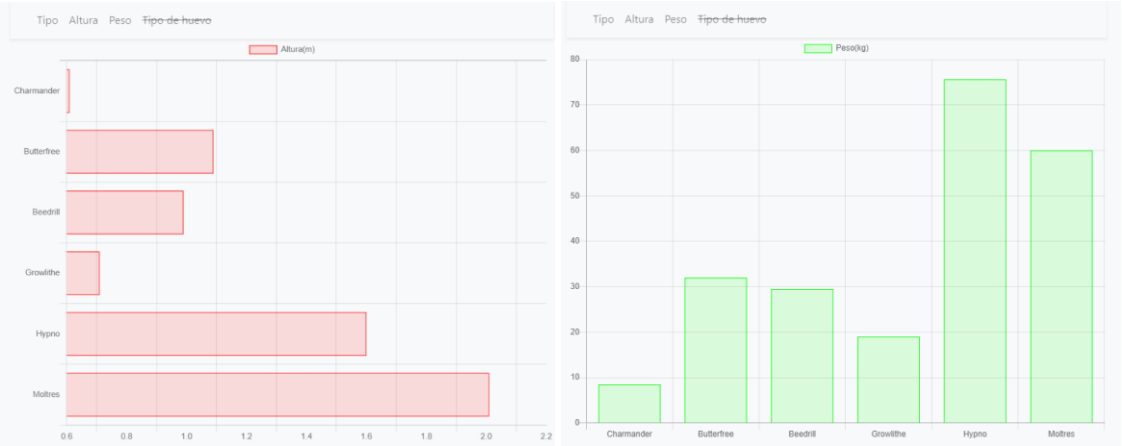
A continuación, se muestra la estructura de la página:



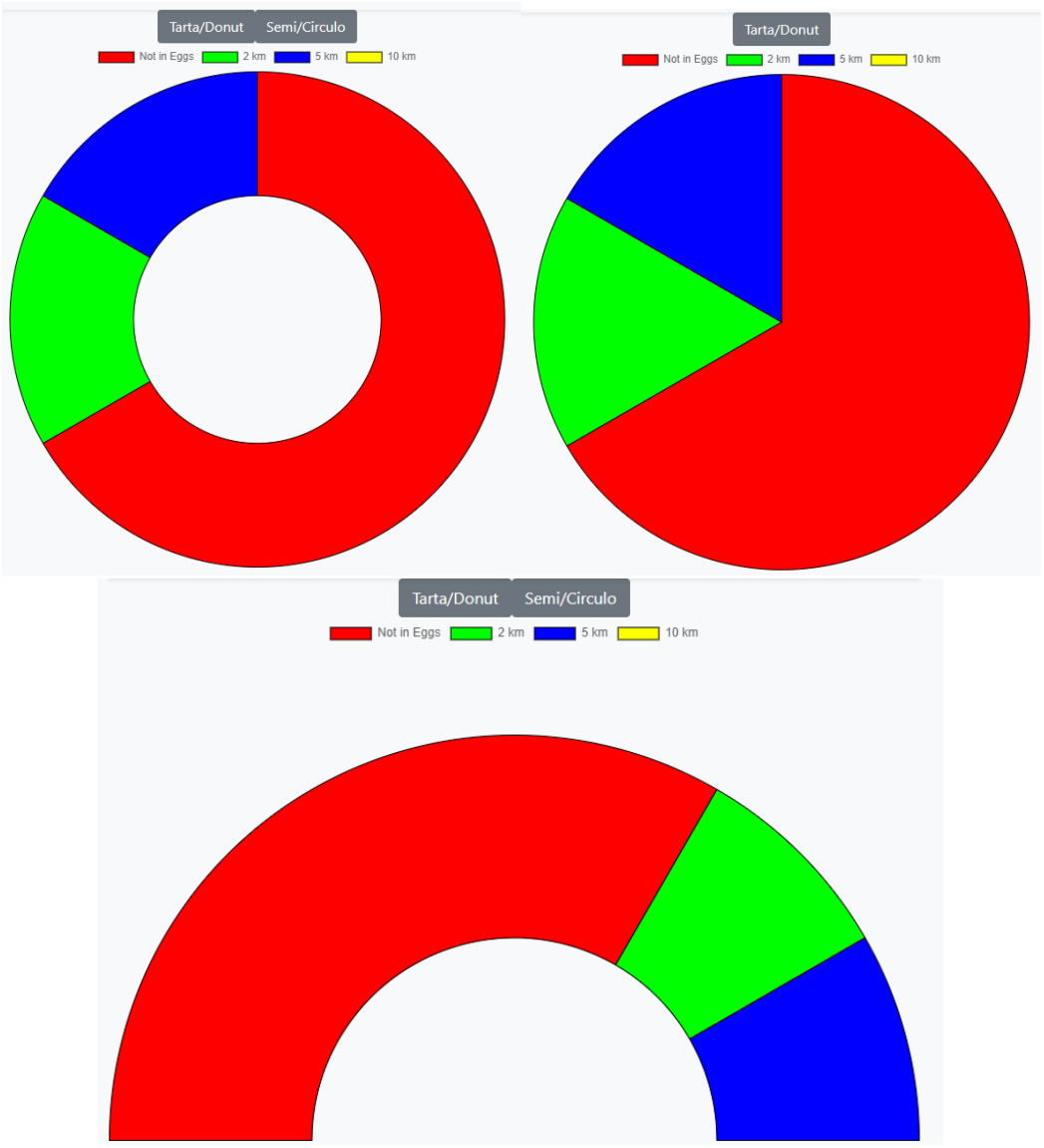
Antes de mostrar los distintos gráficos que se pueden obtener, debe observarse que el enlace de alguno de los tipos de gráfico aparece, en ocasiones, tachado. Esto es porque, pese a que el gráfico se genera igual, la información mostrada no es muy relevante si el número de pokemons seleccionados es muy grande o muy pequeño. Es decir, se pueden seleccionar los gráficos tachados, pero no se podrán extraer muchas conclusiones. Dichos gráficos se resumen a continuación:

- Gráfico por tipo de pokemon: se desarrolla un gráfico "radar" en el que se asigna a cada pokemon un valor de 0, 0.5 y 1 en función de si el pokemon es débil, si no es ni fuerte ni débil o si es fuerte en un determinado "poder". De esta forma puede verse el poder que debería utilizar cada pokemon al enfrentarse a otro u otros para ganar un combate. Por ejemplo, en el que se muestra a continuación, "Charmander" es fuerte (1) en "fire" y "Butterfree" es débil (0), por lo que el primero ganaría, mientras que el segundo no es fuerte en ninguno de los ámbitos en los que "Charmander" es débil, por lo que tendría pocas opciones:

- Gráfico por altura y peso de los pokemons: utilizando dos gráficos de barras, vertical y horizontal, se pueden comparar dos o más pokemons en cuanto a dichas medidas. Esto podría ser interesante en un combate en el que… ¿el grande debería ganar al pequeño?:



- Gráfico por “tipo de huevo”: en la base de datos hay cuatro tipos de huevo, que indican los kilómetros que debe recorrer un jugador para que el huevo se habrá y el pokemon nazca. Para mostrar esto de manera gráfica se generan (con los mismos datos) tres tipos de gráficos, uno de tipo donut con el círculo entero, otro de tipo donut en semicírculo y otro de tipo tarta. Esto podría dar idea de cuánto hay que recorrer, en general, para que se habrán los huevos de distinto tipo (los mayoritarios son los de tipo “not in eggs”, que son los pokemon que no aparecen en huevos, entiendo que el jugador tendría que cazarlos de otra manera):

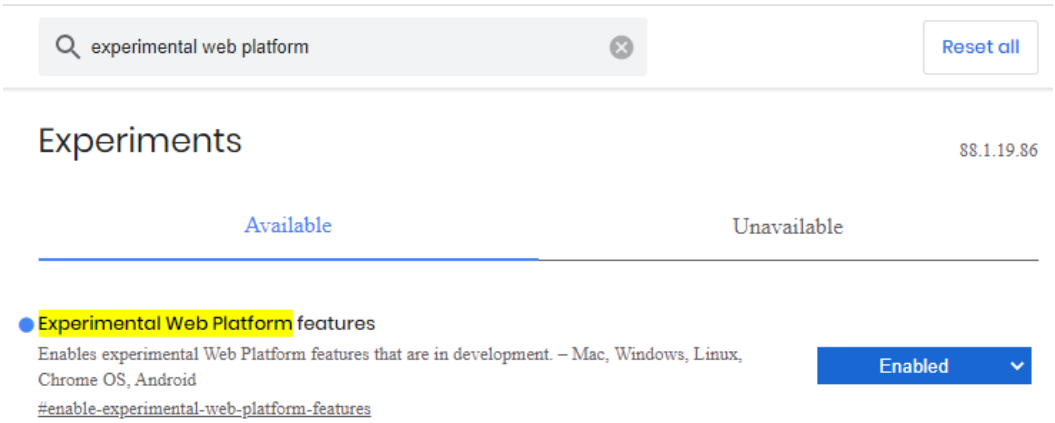


A nivel de código, la base de la página esta en el fichero /templates/practica10.html, aunque toda la funcionalidad se desarrolla en /static/js/practica10.js. En este último, cada vez que se modifica el campo nombre de la columna izquierda, se hace una llamada a la api vía Ajax para obtener los datos de todos los pokemons que cumplan con dicho filtro. Así, una vez se van seleccionando los distintos pokemons y los distintos tipos de gráfico, los datos necesarios ya están en el cliente y el gráfico se genera rápidamente. En este fichero se generan también los distintos tipos de gráfico, con ayuda de la ya comentada librería Chart.js.

Notificaciones push en el cliente

Se permite configurar notificaciones push, de manera que el usuario reciba una notificación de este tipo en su dispositivo (ya sea móvil, ordenador, etc.), para lo cual se hace uso de una de las funcionalidades de las llamadas PWA o aplicaciones web progresivas.

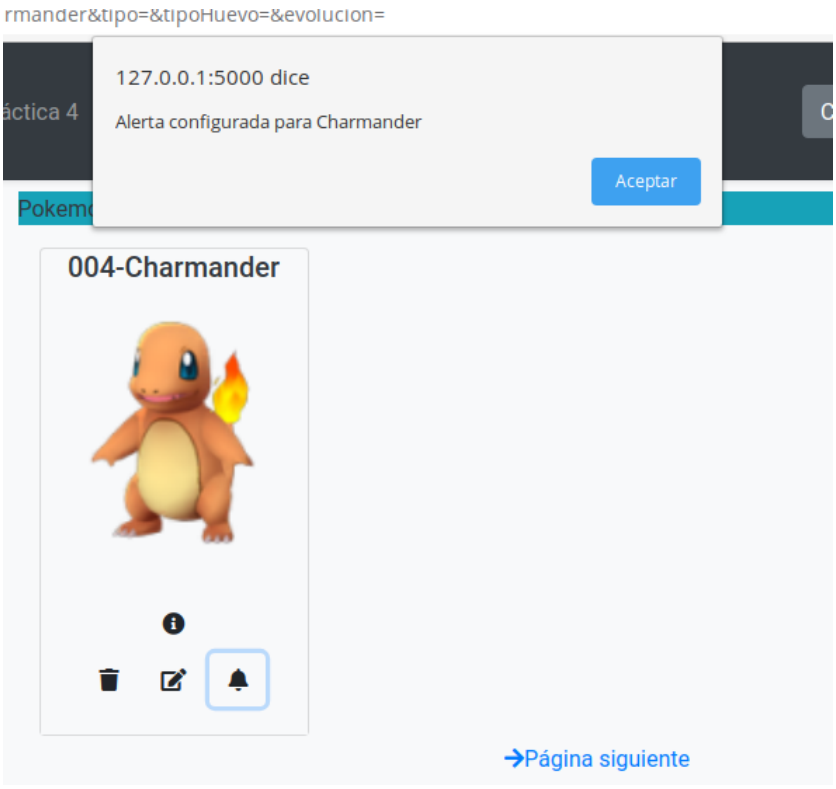
Se trata de una funcionalidad que no está completamente extendida, por lo que podría no funcionar en todos los navegadores. En mi caso, lo he probado en Chromium. Además, para que se pueda utilizar deben habilitarse las opciones de desarrollo (Chrome://flags):

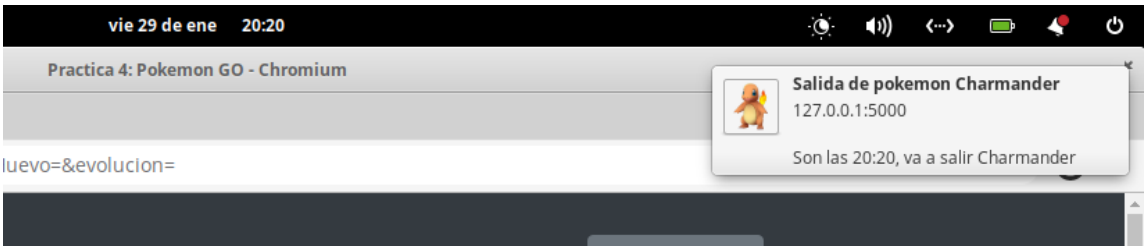


En este caso, las notificaciones se implementan en la página de la práctica 4, donde se añade en la ficha de los pokemon el icono de "notificación". El procedimiento sería el siguiente:

- Filtrar los pokemons de manera que en la página esté aquel sobre el cual se tiene interés.
- Hacer click en el icono "notificación" (la campana).

La notificación aparecería como una notificación del sistema siempre que el navegador esté activo, aunque la página no lo esté. De hecho, en el móvil, bastaría con que el navegador estuviera en segundo plano. En ordenador aparecería así (al activarla y la propia notificación):





Por último, en cuanto al código, me he basado para el desarrollo en el repositorio de github https://github.com/nico-martin/notification-trigger-demo. Los tres ficheros implicados en dicho desarrollo son:

- /templates/practica4.html: se utiliza como base, como se ha dicho, para incluir la campana de activación de la notificación. Además, se incluye el script correspondiente de javascript, /static/js/notificaciones.js.
- /static/js/notificaciones.js: en primer lugar, se "pregunta" al navegador si está soportada la emisión de notificaciones, y, en caso de que lo esté, se procede a registrar el service-worker. Tras esto, se configura el botón de notificaciones para que, al pulsar, se configure la notificación.
- /static/js/service-worker.js: es el service-worker en cuestión, permite que la página web funcione como una aplicación (habría que hacer más para que esto fuera realmente así, cumpliendo con los requisitos de una PWA, aunque para esto serviría).