Integrantes:
-Daniel Díez Arias
-María García Sanchez
-Marcos Gómez de Quero
-Brais Rodríguez Calvo

Nombre de la aplicación: LeagueUPSA

1. Descripción funcional de la aplicación

1.1- Descripción general de la aplicación

LeagueUPSA es una aplicación con la cual podemos obtener información sobre los jugadores del popular juego League of Legends, siendo esta información desde información general del perfil hasta detallado de las últimas partidas jugadas.

1.2. Descripción de funcionamiento

1.2.1. Al ejecutarse la aplicación, el usuario verá en primer lugar una ventana con el icono de League of Legends.

1.2.2. El usuario podrá ver la ventana principal de la aplicación, en la cual se vera un carrusel de imágenes junto con el logotipo de League of Legends y un área de texto junto con un botón de búsqueda. El usuario tendrá que introducir un nombre de invocador existente en el área de texto y a continuación pulsar el botón de búsqueda. Si la búsqueda se realiza satisfactoriamente el usuario accederá al grueso de la funcionalidad de la aplicación. SI no existe el invocador el usuario podrá volver a introducir el nombre de invocador.

1.2.3. Se cargará la ventana principal de la aplicación, siendo esta un Tab Bar Controller donde existirán tres ítems entre los que podrá navegar el usuario, explicando a continuación la funcionalidad de cada ítem:

a) Item general: mostrará información general del invocador buscado junto con las ultimas partidas jugadas por el mismo.
En las partidas jugadas, el jugador podrá pulsar el botón con texto “Ver partida” para ver un detallado de la partida.

Al pulsarse el botón de “Ver partida”, el usuario podrá ver en una nueva ventana el detallado de la ultima partida jugada mostrándose aliados y enemigos, nombre de campeón, nombre de invocador y estadísticas sobre la misma partida.

b) Item campeones: mostrará un listado aleatorio de campeones del juego. Si el usuario pulsa sobre el nombre del campeón, podrá ver información ampliada sobre el campeón.

Al pulsarse el botón con el nombre del campeón, se mostrará en una nueva ventana información relativa al campeón como es su nombre, su historia y sus habilidades.

c) Item directo: mostrará si el invocador seleccionado se encuentra en partida en ese momento o esta desconectado. Podremos ver un caja de texto con el texto “Próximamente” haciendo referencia a una ampliación de la funcionalidad de este ítem.

2. Descripción técnica de la aplicación

2.1. Jerarquia de la aplicación

Mostraremos los elementos de los que se encuentra compuesta la aplicación y una breve descripción de la funcionalidad de ambos

2.1.1. Asset.xcassets: carpeta contenedora del icono de aplicación
2.1.2. Assets: carpeta contenedora de todas las imágenes utilizadas en la aplicación
2.1.3. New Group: carpeta que contendrá el framework utilizado para el desarrollo de la aplicación

2.1.3. LeagueUPSA: carpeta contenedora de las vistas, controladores y modelos utilizados en el desarrollo de la aplicación

Controladores

ViewController: controlador inicial de la aplicación.

ViewControllorGeneralView: controlador de la vista de información general de la aplicación.

ViewControllerChampions: controlador de la vista encargada de mostrar información general sobre campeones.

ViewControllerCampeonDetalle: controlador de la vista encargada de mostrar información detallada de cada campeón.

ViewControllerPartidaDetallada: controlador de la vista encargada de mostrar información detallada sobre la partida seleccionada.

ViewControllerDirecto: controlador de la vista encargada de mostrar información acerca de si una partida se encuentra en directo.

Modelos

ModeloPrincipal: modelo que contiene funcionalidades síncronas utilizadas en la aplicación

APIFunctions: modelo que contiene todas las funciones asíncronas necesarias para obtener la información relativa al invocador seleccionado. Utilizará la API para obtener los recursos que sean necesarios

TablaPrincipal: modelo que permitirá la carga de información de manera general en la Table View ubicada en el ítem principal.

TablaCampeones: modelo que permitirá la carga de información sobre los campeones en la Table View ubicada en el ítem campeón.

TablaJugadorDetalle: modelo que permitirá la carga de información relativa a la partida seleccionada en el Table View del ítem principal.

FondoAnimado: modelo encargado de gestionar el carrusel de imágenes mostrada en la vista de inicio de la aplicación.

Vistas

Main: contendrá todas las vistas que se visualizarán en el uso de la aplicación. Constara de siete vistas en total:

-Vista inicial
-Vista de información general
-Vista de partida detallada
-Vista de campeón general
-Vista de campeón detallada
-Vista de directo
-Vista TabBarController que contendrá todos los ítems

LaunchScreen: contendrá la vista que se mostrará al iniciarse la aplicación

Recursos

APIResources: contiene todas la variables utilizadas en la aplicación

2.2. Componentes utilizados

Para el desarrollo de la aplicación se han hecho uso de los siguientes elementos visuales:

-TabBarController: utilizado para contener los tres ítems principales de la aplicación.

-NavigatioBar: utilizado para poder mostrar en que ventana nos encontramos y poder retroceder a la ventana anterior.

-TableView: utilizado para contener la información utilizando de esta manera contenedores generales no haciendo necesaria la declaración repetida de los elementos.

-Segues: utilizados para la navegación entre vistas de la aplicación.

En cuanto al apartado puramente técnico, refiriéndonos al funcionamiento de la aplicación se debe destacar:

API League of Legends: para poder obtener toda la información utilizada y mostrada en el proyecto hemos utilizada la api proporcionada por League of Legends. Hemos optado por una versión descargable ya que de esta manera simplemente debemos incluir la API en el proyecto como una librería, teniendo acceso así a todos los métodos necesarios.

Uso de colas y semáforos: para que el funcionamiento de la aplicación fuera correcto, hemos tenido que incluir uso de colas y semáforos ya que los métodos proporcionados por la API son asíncronos y ademas de ello no devuelven información en su retorno.
Por ello hemos establecido colas de operaciones que nos permiten controlar el flujo de información, asegurándonos que solamente un método se ejecutará cuando el anterior termine de recoger los datos de la aplicación. Usamos semáforos dentro de la propia llamada de los métodos para poder bloquear la salida del método hasta que la información llegue, liberando así los recursos.

3. Modelo-Vista-Controlador

El MVC utilizado queda detallada con la jerarquía descrita anteriormente. Hemos separado el apartado encargado de realización de operaciones principales de la aplicación de las llamadas a métodos y carga de variables para mostrar la información en la aplicación. Se han separado también los recursos utilizados en la aplicación del resto del programa para mejorar la accesibilidad de dichos elementos en toda la aplicación.

4. Dificultades encontradas

El principal problema al que nos hemos enfrentado en el desarrollo de la aplicación es la gestión de las operaciones asíncronas, ya que el ser estas así, al intentar llamar a un método que necesita información proporcionada por uno anterior podía darse la situación de que esta información no estuviera cargada aun. Es por ello que hemos utilizado las colas y semáforos mencionadas anteriormente.

Para que el uso de la aplicación pueda llevarse a termino es necesario renovar la KEY de la API cada 24 horas.