Instituto Tecnológico de Costa Rica

Escuela de Ingeniería en Computación

Carrera Ingeniería en Computación

IC-8041 Desarrollo de Aplicaciones para Dispositivos Móviles

Profesores: Andrei Fuentes Leiva

Jeff Schmidt Peralta

Grupo 01

Proyecto II

Estudiante:

- González Chacón Juan Carlos 201226896
- Rodríguez Arce Juan José 200944003

Octubre, 2014

Cartago

Tabla de Contenido

Descripción del aplicación

Git-NES; será una aplicación con un entorno lo suficientemente amigable para que cualquier usuario de Google Play, pueda acceder a todas sus funcionalidades de manera sencilla y rápida, retornando de manera inmediata los juegos de super Nintendo de interés del usuario, para su descarga para así sean previamente manipulados al gusto del usuario.

El usuario además puede tener varios roles, tanto como de administrador como de seguidor de videojuegos, con ello teniendo la oportunidad de observar los juegos dependiendo de la categoría que le sea a gusto, entre las experiencias que el usuario podrá vivir dentro de la aplicación, se destaca la búsqueda avanzada de juegos de Nintendo, entre los cuales podrá dar una recomendación de juegos, permitiendo al usuario darle like o dislike a los mismos juegos.

Con ello se tendrá una experiencia social sin dejar de lado un criterio más objetivo en el cual se podrá ir mejorando con el grueso de usuarios la experiencia dentro de la misma. Cabe mencionar que hacer una experiencia al igual que las redes sociales actuales se pueden ver perfiles de juegos y ver videos relacionados de youtube con ese juego, para así el usuario no necesariamente tenga que estar descargando juegos, más bien podrá ir discriminando que le gusta realmente y por ende que será previamente descargado.

Entre otras funcionalidades el agregar juegos favoritos, bajar juegos al teléfono, la facilidad de los usuarios que puedan agregar los mismos juegos, pero sin peligro ya que se dispondrá de un API con antivirus en el cual discrimine qué archivos podrían ser potenciales peligros para el usuario, pero con ello además para delimitar post maliciosos se dispondrá de una interfaz de administrador para aprobar los juegos o no, haciendo más que una experiencia llena de errores y posibles disputas, una red para compartir juegos de manera eficiente, amigable y segura.

Requerimientos Funcionales

- El acceso a la aplicación debe ser controlada, para ello debe de autentificarse el ingreso de usuarios al sistema por medio de la red social Facebook.

- El usuario, autenticado en el sistema, debe ser capaz de visualizar los diferentes juegos de Super Nintendo disponibles en la aplicación, además debe de tener la capacidad de poder realizar búsqueda de los mismos.

- La aplicación debe realizar recomendaciones de juegos disponibles a los usuarios, además debe de permitirles darle like o dislike a las mismas.

- La aplicación debe permitir visualizar el perfil de los juegos, además para los usuarios interesados en conocer más de los mismos debe permitir ver videos de Youtube relacionados con este juego.

- El usuario debe ser capaz de identificar un juego como favorito, para posteriormente visualizarlo en sus publicaciones favoritas.

- La aplicación debe de permitir al usuario descargar la cantidad de juegos que desee al dispositivo, además deber también permitir la carga de juegos al repositorio, para lo cual ha de utilizar el API de transferencia de archivos.

- Para la administración del repositorio debe existir una interfaz, mediante la cual el administrador sea capaz de aprobar o rechazar la solicitud de agregar juegos al repositorio.

- La aplicación debe permitir visualizar juegos aleatoriamente, para tal debe de utilizar el acelerómetro del dispositivo.

Requerimientos No Funcionales

- Para la creación de la aplicación debe de utilizarse un framework de desarrollo nativo para Android.
- El backend debe desarrollarse en Python utilizando Google App Engine.
- En el desarrollo de la aplicación debe de seguirse estándares de calidad para posibles mantenimientos futuros de forma rápida y oportuna.
- La aplicación debe de hacer uso de diferentes patrones de diseño para aplicaciones móviles

Funcionalidades

- **Inicio de Sesión:** mediante la cuenta de Facebook cualquier persona interesada en tener acceso al repositorio, será capaz de utilizar las diferentes funcionalidades que la aplicación ofrece.
- **Visualización General de Juegos:** el usuario de la aplicación podrá ver en la ventana principal de una forma general los diferentes juegos que se han agregado recientemente.
- **Recomendaciones:** el usuario podrá tener acceso a un área de la aplicación donde se le ofrecerán recomendaciones de juegos que el repositorio posee, permitiéndole además darle like o dislike.
- **Perfil de juegos:** el usuario podrá ver como mayor detalle el perfil de los juegos, en el cual se podrá visualizar las diferentes características del mismo.

- **Favoritos:** el usuario tiene la posibilidad de identificar un juego determinado como favorito, para visualizarlo posteriormente en el área respectiva de la aplicación.

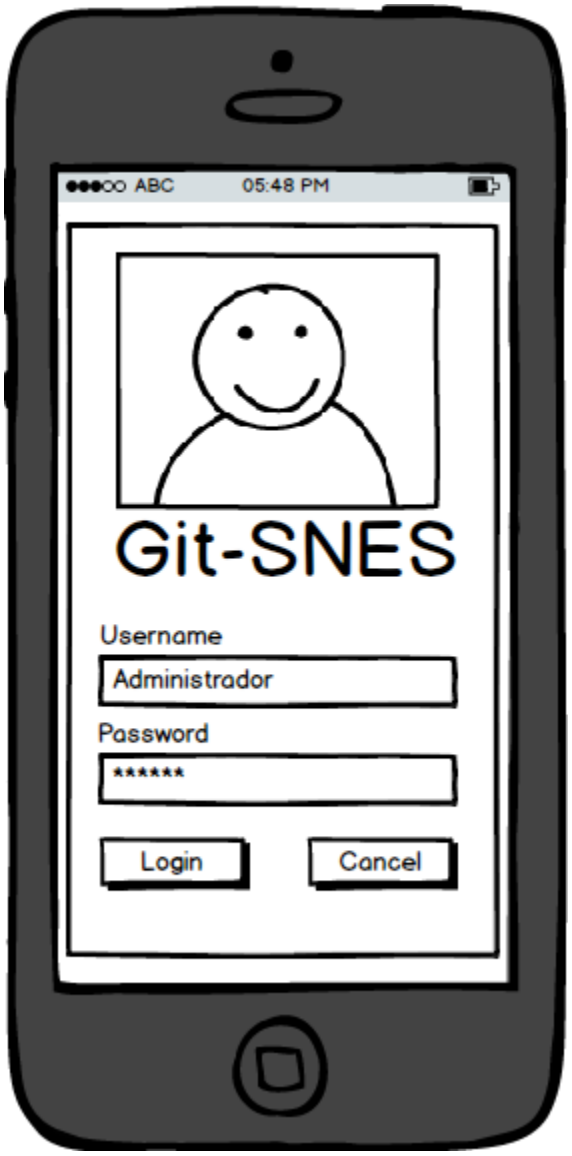
- **Descarga de Juegos:** el usuario de la aplicación puede descargar a sus dispositivos móviles los juegos que desee.

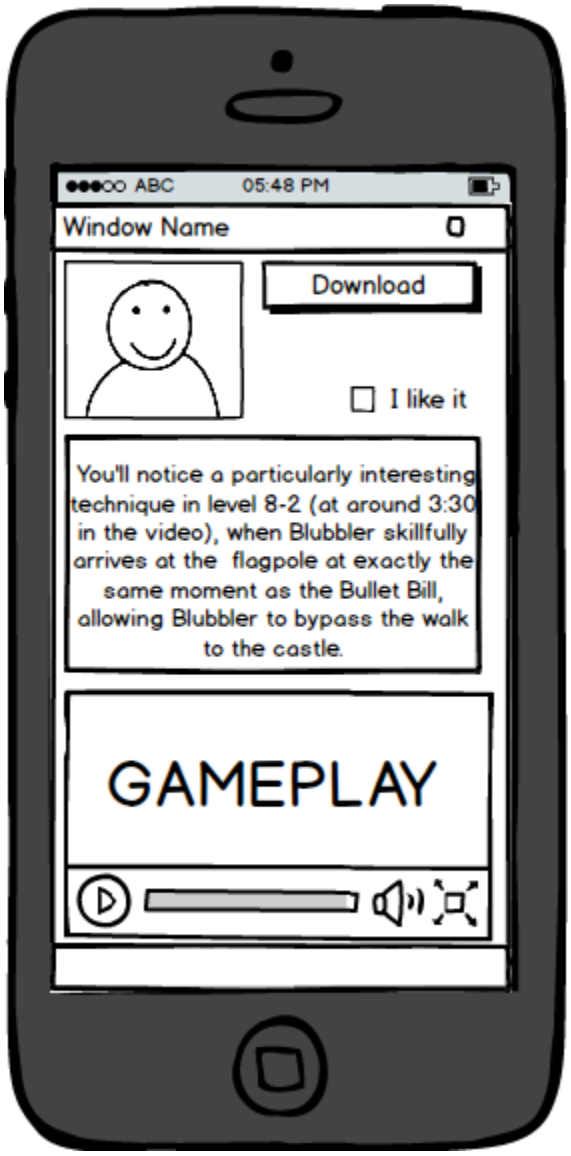
- **Subida de Juegos al Repositorio:** el usuario es capaz de agregar juegos al repositorio, sin embargo estos quedarán pendientes de mostrar en la aplicación hasta que el administrador del sistema lo apruebe.

- **Administración del Repositorio:** en el sistema existirá usuarios administradores del repositorio, los cuales tendrán la responsabilidad de aprobar o rechazar los juegos que diferentes usuarios particulares han agregado a la aplicación.

- **Visualización Aleatoria de Juegos:** el usuario tiene la capacidad de visualizar aleatoriamente un juego cada vez que agitar el dispositivo móvil.

Wireframes

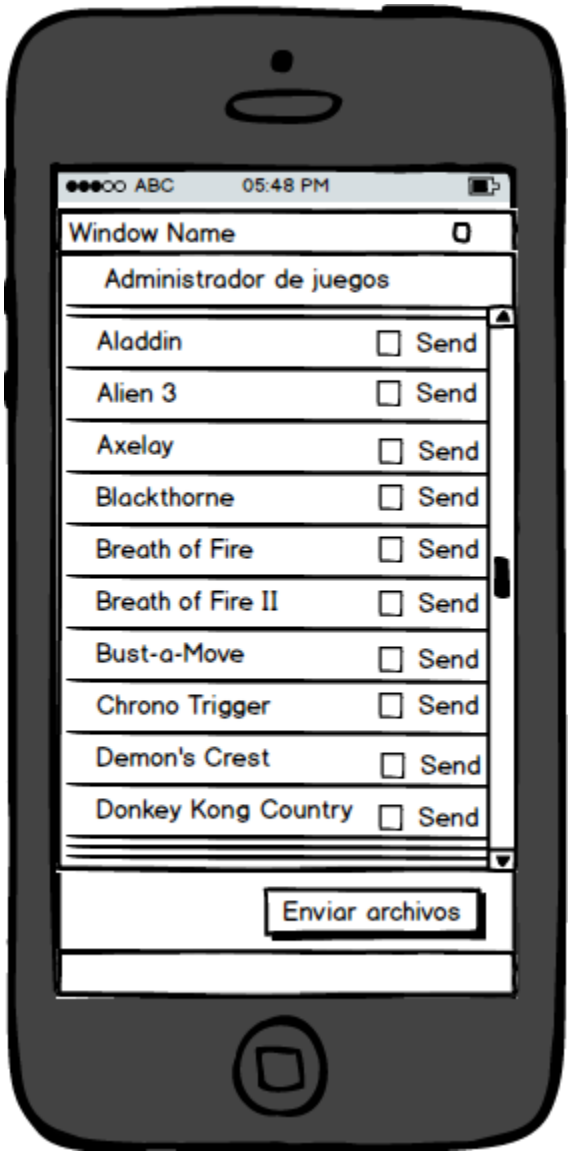

ABC
05:48 PM
Git-SNES
Username
Administrador
Password
******
Login
Cancel


ABC
05:48 PM
Window Name
Download
I like it
You'll notice a particularly interesting technique in level 8-2 (at around 3:30 in the video), when Blubbler skillfully arrives at the flagpole at exactly the same moment as the Bullet Bill, allowing Blubbler to bypass the walk to the castle.
GAMEPLAY


ABC
05:48 PM
Window Name
ActRaiser
Aladdin
Alien 3
Axelay
Blackthorne
Breath of Fire
Breath of Fire II
Bust-a-Move
Chrono Trigger
Contra III: The Alien Wars
Demon's Crest
Donkey Kong Country
Donkey Kong Country 2: Diddy's
Donkey Kong Country 3: Dixie


ABC
05:48 PM
Window Name
Busqueda avanzada
Top populares
Chrono Trigger
Contra III: The Alien Wars
Donkey Kong Country 2:
Donkey Kong Country 3: Dixie
Resultados obtenidos
Chrono Trigger
Contra III: The Alien Wars
Demon's Crest
Donkey Kong Country
Donkey Kong Country 2: Diddy's
Donkey Kong Country 3: Dixie


ABC
05:48 PM
Window Name
Busqueda avanzada
Agregar juego
Nombre
Categorias
Descripción
Seleccionar
Subir juego

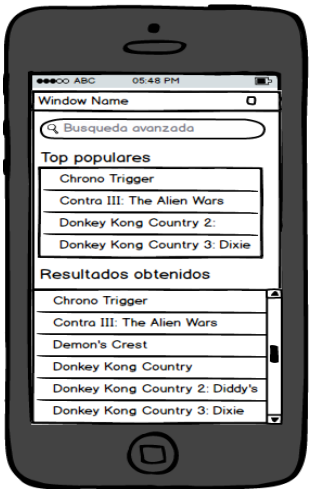

ABC
05:48 PM
Window Name
Administrador de juegos
Aladdin
Send
Alien 3
Send
Axelay
Send
Blackthorne
Send
Breath of Fire
Send
Breath of Fire II
Send
Bust-a-Move
Send
Chrono Trigger
Send
Demon's Crest
Send
Donkey Kong Country
Send
Enviar archivos

Explicación de patrones a utilizar

Patrones de despliegue de datos

Master-Detail

Un modelo master-detail es un concepto de diseño de interfaz mediante el cual se muestra una lista de elementos (en adelante, la lista maestra) al usuario. Al seleccionar un elemento de la lista, información adicional relacionada con ese tema se presenta al usuario dentro de un panel de detalles. En el caso de nuestra aplicación se podría, por ejemplo, consistir en una lista maestra de los juegos mediante el dashboard descrito anteriormente. Tras la selección de un juego de la lista principal, se mostrará todos los datos de cada uno de los juegos con lujo de detalle, además la interfaz para descargarlos, ver videos entre otros funcionalidad de gran importancia para el jugador que desee obtenerlo.



Dashboard

Dashboards proporcionan a menudo en un solo vistazo visitas de KPIs (indicadores clave de rendimiento) correspondiente a un proceso objetivo o negocio en particular (por ejemplo, ventas, marketing, recursos humanos, o de producción).

El término dashboard se origina en el tablero de instrumentos del automóvil donde los conductores controlar las principales funciones de un vistazo a través del cuadro de instrumentos. Dashboards dan señales sobre un negocio dejando que el usuario sabe que algo está mal o algo está bien. El mundo empresarial ha intentado durante años para llegar a una solución que les dijera si su negocio necesita mantenimiento o si la temperatura de su negocio se ejecuta encima de lo normal.

Dashboards normalmente se limitan a mostrar los resúmenes, las principales tendencias, comparaciones y excepciones. Hay cuatro elementos clave de un buen cuadro de mandos:

- Simple, comunica fácilmente
- Distracciones mínimas, que podrían causar confusión
- Soporta negocio organizado con los datos de significado y útiles
- Aplica la percepción visual humana a la presentación visual de la información

En este caso será utilizado para el manejo dentro del listview de cada uno de los juegos y mostrar de manera resumida los datos más relevantes de los mismos (nombre, calificación, peso, categoría principal).

Patrones de diseño de búsqueda

SearchView

Un cuadro de diálogo de búsqueda en la parte superior de la pantalla o un widget de búsqueda (SearchView) que se puede incrustar en su diseño de la actividad. En cualquiera de los casos, el sistema Android ayudará a su aplicación de búsqueda mediante la entrega de las consultas de búsqueda a una actividad específica que realiza búsquedas.

Advanced search

Algunas de las opciones que nos ofrece la búsqueda avanzada son la posibilidad de buscar resultados especificando algunas palabras, todas las palabras o frase exacta. Estas opciones no son excluyentes se pueden usar de forma combinada.

Búsqueda aleatoria (Mediante acelerómetro)

Mediante el acelerómetro se puede recurrir a la opción avanzada para obtener de manera aleatoria cualquier juego que esté en la base de datos.

Patrones de diseño de filtro y ordenamiento

Listview (ordenamiento por relevancia/favoritos de la gente)

El control ListView resulta útil para mostrar datos que se encuentran en una estructura que se repite, de manera similar a los controles DataList y Repeater. A diferencia de estos controles, el control ListView admite operaciones de edición, inserción y eliminación, así como ordenación y paginación. La funcionalidad de paginación de ListView la proporciona el nuevo control DataPager.

Advanced search (filtro por categoría o nombre)

Algunas de las opciones que nos ofrece la búsqueda avanzada son la posibilidad de buscar resultados especificando algunas palabras, todas las palabras o frase exacta. Estas opciones no son excluyentes se pueden usar de forma combinada.

Patrones de diseño para navegación: primaria y secundaria

Listview

El control ListView resulta útil para mostrar datos que se encuentran en una estructura que se repite, de manera similar a los controles DataList y Repeater. A diferencia de estos controles, el control ListView admite operaciones de edición, inserción y eliminación, así como ordenación y paginación. La funcionalidad de paginación de ListView la proporciona el nuevo control DataPager.

El control ListView es un control que permite personalizarse al máximo mediante plantillas y estilos para definir la interfaz de usuario (IU) del control. Al igual que los controles Repeater, DataList y FormView, las plantillas del control ListView no están predefinidas para representar la UI específica en el explorador.

Master-Detail

Un modelo master-detail es un concepto de diseño de interfaz mediante el cual se muestra una lista de elementos (en adelante, la lista maestra) al usuario. Al seleccionar un elemento de la lista, información adicional relacionada con ese tema se presenta al usuario dentro de un panel de detalles. En el caso de nuestra aplicación se podría, por ejemplo, consistir en una lista maestra de los juegos mediante el dashboard descrito anteriormente. Tras la selección de un juego de la lista principal, se mostrará todos los datos de cada uno de los juegos con lujo de detalle, además la interfaz para descargarlos, ver videos entre otros funcionalidad de gran importancia para el jugador que desee obtenerlo.

Patrones de diseño de formularios

Login-Form

Se utilizará la manera de login de Facebook mediante una interfaz sencilla sin muchos requerimientos para el usuario, siendo así minimalista e inmediata de ingreso.

Add game

Se utilizará la manera de login de Facebook mediante una interfaz sencilla sin muchos requerimientos para el usuario, siendo así minimalista e inmediata de ingreso

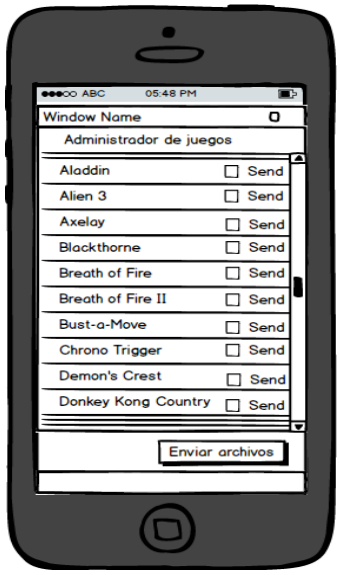
Patrones de herramientas

Modificar datos en un ListView

Se pueden crear plantillas para el control ListView que permitan a los usuarios editar, insertar o eliminar un único elemento de datos, en el caso específico de la aplicación será para aceptar o rechazar juegos, en ambos casos ya dado el hecho se eliminarán de la lista.

La plantilla debe incluir los controles enlazados a datos donde un usuario puede editar los valores. Por ejemplo, la plantilla puede incluir cuadros de texto donde los usuarios editen los valores existentes.

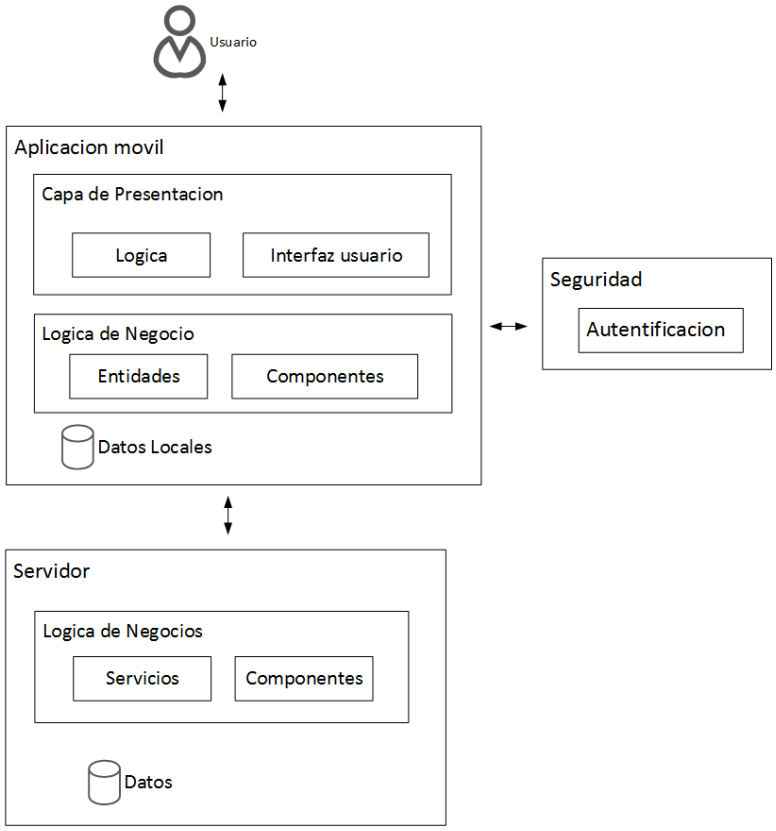
Normalmente se agregan botones a las plantillas para permitir a los usuarios especificar qué acción desea realizar. Por ejemplo, se puede agregar el botón Send (enviar) a una plantilla de elemento para permitir a los usuarios eliminar ese elemento. Se puede agregar el botón Editar a una plantilla de elemento para permitir a los administradores cambiar al modo de edición.



Descripción de Diseño de Alto Nivel

Arquitectura de la Aplicación

La implementación de la aplicación se desarrollará bajo un modelo de aplicación Mobile donde el backend es el responsable de mantener toda la información relacionada al repositorio, pero además se encargará de realizar otras funciones de lógica de negocios como el manejo de búsqueda de juegos y likes. Por otra parte, el frontend, además de encargarse de la visualización de la aplicación se encargará de la conexión con APIs externos al sistema, así como la comunicación con hardware propio del dispositivo móvil para su utilización en funcionalidades del sistema



Descripción detallada

- El almacenamiento de los datos estará a cargo de una base de datos relacional. Esta estará a cargo de mantener toda la información del repositorio de juegos.

- El proyecto será implementado utilizando el patrón de diseño MVC, de tal forma que cada parte del sistema se mantenga independiente de los demás y no exista una alta cohesión y bajo acoplamiento.

- El web service será el medio disponible para la comunicación entre la aplicación móvil y el backend.

- La administración de los juegos cargados al sistema, por usuarios particulares, se llevará a cabo por medio de una página web que permita visualizar los juegos pendientes de mostrar en la aplicación y además aprobar o rechazar los mismos.

- El backend de la aplicación será desarrollado en Python, utilizando para su alojamiento la tecnología de Google App Engine, mientras que el frontend se implementará nativamente por medio de Android Studio.

- Además para el diseño de la interfaz gráfica y visualización de las diferentes funcionalidades de la aplicación se utilizarán diferentes patrones de diseño para dispositivos móviles.

Diagrama Relacional de la Base de Datos

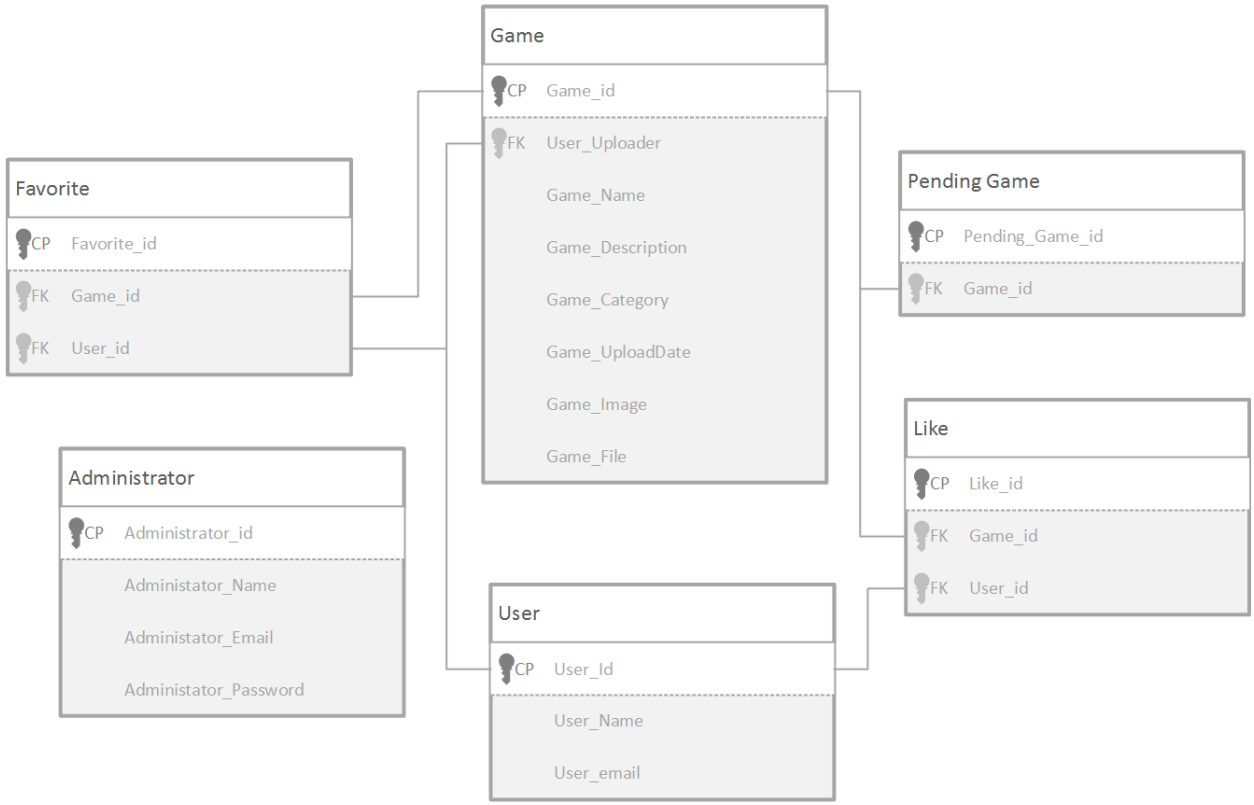
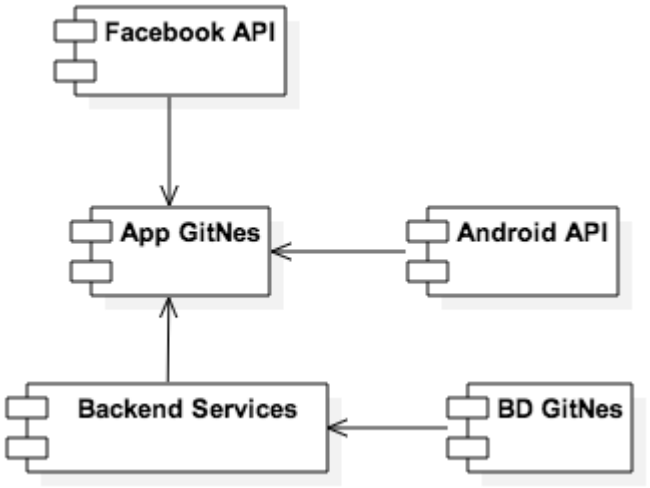


Diagrama de Componentes



Descripción de los Web Services

Método	URI	Descripción
GET	/games/{hashing}	Obtiene todos los juegos del repositorio.
GET	/users/{hashing}	Obtiene todos los usuarios registrados en la base de datos.
GET	/token/{user_emailP}	Obtiene el token necesario para tener acceso a la API
GET	/gamesrecomendation/{user_idP}	Obtiene recomendaciones de juegos de acuerdo a la categoría de los juegos que han sido agregado a favoritos
GET	/moreliked/{hashing}	Obtiene recomendaciones de juegos de acuerdo a los primeros diez juegos con la mayor cantidad de likes.
GET	/recomendedliker/{hashing}	Obtiene recomendaciones de juegos de acuerdo a la juegos que tienen la mayor cantidad de likes.
GET	/gamesfavorite/{user_id}	Obtiene todos los juegos favoritos de un usuario determinado.
GET	/GamesSearch/{application_key}	Realiza búsqueda de juegos por su id
POST	/gamesfavoriteadd/{game_idP}/{user_idP}	Agrega un juego a los favoritos de un usuario si este no ha sido agregado.
POST	/createuser/{user_nameP}/{user_emailP}/{hashing}	Crea un usuario en la BD si este no existe
POST	/game/{uploader}/{game_name}/{game_description}/{game_category}/{_image}/{_file}/{hashing}	Agrega un juego a la BD.
PUT	/likegame/{application_key}/{hashing}	Agrega un like a un juego.
PUT	/changestate/{application_key}/{hashing}	Cambia el estado de un juego de pendiente de aprobar a aprobado.
DELETE	/deletegame/{application_key}/{hashing}	Elimina un juego especifico.

GETS

/games/{hashing}

Parámetro	Tipo	Descripción
hashing	string	Token necesario para tener acceso al API

/users/{hashing}

Parámetro	Tipo	Descripción
hashing	string	Token necesario para tener acceso al API

/token/{user_emailP}

Parámetro	Tipo	Descripción
user_emailP	string	Email del usuario que desea obtener un token

/gamesrecomendation/{user_idP}

Parámetro	Tipo	Descripción
user_idP	string	Id del usuario que desea obtener recomendaciones

/moreliked/{hashing}

Parámetro	Tipo	Descripción
hashing	string	Token necesario para tener acceso al API

/recomendedliker/{hashing}

Parámetro	Tipo	Descripción
hashing	string	Token necesario para tener acceso al API

/gamesfavorite/{user_id}

Parámetro	Tipo	Descripción
user_id	string	Id del usuario que desea obtener todos sus juegos favoritos

/GamesSearch/{application_key}

Parámetro	Tipo	Descripción
application_key	string	Id del juego que desea jugar

POSTS

/gamesfavoriteadd/{game_idP}/{user_idP}

Parámetro	Tipo	Descripción
game_idP	string	Id del juego que desea agregarse a favoritos
user_idP	string	Id del usuario que desea agregar el juego a favoritos

/createuser/{user_nameP}/{user_emailP}/{hashing}

Parámetro	Tipo	Descripción
user_nameP	string	Nombre del usuario que se desea agregar
user_emailP	string	Email del usuario que se desea agregar
hashing	string	Token necesario para tener acceso al API

/game/{uploader}/{game_name}/{game_description}/{game_category}/{_image}/{_file}/{hashing}

Parámetro	Tipo	Descripción
uploader	string	Usuario que esta cargando el juego
game_name	string	Nombre del juego a agregar
game_description	string	Descripción del juego a agregar
game_category	string	Categoría del juego a agregar
_image	string	Url de donde se almacena a imagen
_file	string	Url de donde se almacena el archivo
hashing	string	Token necesario para tener acceso al API

PUTS

/likegame/{application_key}/{hashing}

Parámetro	Tipo	Descripción
application_key	string	Id del juego al que se le desea like
hashing	string	Token necesario para tener acceso al API

/changestate/{application_key}/{hashing}

Parámetro	Tipo	Descripción
application_key	string	Id del juego al que se le desea cambiar su condición de pendiente a aprobado
hashing	string	Token necesario para tener acceso al API

DELETE

/deletegame/{application_key}/{hashing}

Parámetro	Tipo	Descripción
application_key	string	Id del juego que desea eliminar
hashing	string	Token necesario para tener acceso al API

Interacción con sistemas externos(APIs)

API - Transferencia de archivos

Una representación "abstracta" de una entidad del sistema de archivos identificado por un nombre de ruta. La ruta puede ser absoluta (relativa al directorio raíz del sistema de archivos) o relativa al directorio actual en el que se ejecuta el programa. El archivo real al que hace referencia un archivo puede o no existir. Se puede también, a pesar del nombre de archivo, un directorio o de otro archivo que no sea regular.
Esta clase proporciona una funcionalidad limitada para obtener / establecer permisos de archivo, tipo de archivo, y la última vez modificada.

Implementación

Al guardar un archivo en la memoria interna, puede adquirir el directorio adecuado como un archivo llamando a uno de dos métodos:
getFilesDir () Devuelve un archivo que representa un directorio interno para su aplicación.
getCacheDir () Devuelve un archivo que representa un directorio interno para los archivos caché temporal de su aplicación.
Asegúrese de eliminar cada archivo una vez que ya no es necesario y poner en práctica un límite de tamaño razonable para la cantidad de memoria que se utiliza en un momento dado, por ejemplo, de 1 MB. Si el sistema comienza a funcionar bajo el almacenamiento, puede eliminar los archivos de caché sin previo aviso.

Para crear un nuevo archivo en uno de estos directorios, puede utilizar el constructor File (), pasándole el archivo proporcionado por uno de los métodos anteriores que especifica el directorio de almacenamiento interno. Por ejemplo:

```
File file = new File(context.getFilesDir(), filename);
```

Alternativamente, usted puede llamar openFileOutput () para obtener un FileOutputStream que escribe en un archivo en el directorio interno. Por ejemplo, aquí se explica cómo escribir un texto en un archivo:

```
String filename = "myfile";
String string = "Hello world!";
FileOutputStream outputStream;

try {
  outputStream = openFileOutput(filename,
Context.MODE_PRIVATE);
  outputStream.write(string.getBytes());
  outputStream.close();
} catch (Exception e) {
  e.printStackTrace();
}
```

API - Facebook

Facebook Login permite obtener un token para acceder a la API de Facebook en nombre de alguien usando su aplicación. Se puede utilizar esta función en lugar de la construcción de su propio sistema de cuenta o para agregar servicios de Facebook a sus cuentas existentes.
El SDK de Facebook para Android proporciona métodos para implementar Facebook Entrar para su aplicación.

Requisitos previos

Para utilizar Facebook Iniciar sesión, primero debe agregar el SDK de Facebook para su proyecto. Crear un nuevo proyecto Android. No añadir el código relacionado con un proyecto viable mínima. Hay que asegurar de que se ha configurado una nueva aplicación para Android, vinculado su proyecto para el SDK de Facebook y configurado el Manifiesto de Android para el ID de la aplicación y el LoginActivity.

Una vez que haya pasado por estos pasos, puede utilizar métodos en el SDK para implementar FacebookIn en su aplicación.

Implementar autenticación

Al final de este paso, alguien que use su aplicación va a ser capaz de hacer clic en un botón para iniciar sesión con Facebook.

En primer lugar, vamos a añadir un botón *LoginButton* que la persona puede hacer clic para conectarse. También configura un fragmento de configurar la interfaz de usuario.

Abrir archivo de diseño de XML de su actividad principal: res / layout / main.xml, y agregar el botón LoginButton especificando el nombre completo de la clase: com.facebook.widget.LoginButton. El resultado debe ser similar a esto:

```
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/andro
id"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <com.facebook.widget.LoginButton
    android:id="@+id/authButton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center_horizontal"
    android:layout_marginTop="30dp"/>
</LinearLayout>
```

API - Youtube

El API del reproductor de YouTube permite incorporar la funcionalidad de reproducción de video en las aplicaciones de Android. Dicho API define métodos para cargar y reproducir videos de YouTube para personalizar y controlar la experiencia de reproducción de video.

Al usar la API se puede cargar o insertar videos en una vista de reproductor secuencial en la interfaz de usuario de la aplicación. Con esto se puede controlar la reproducción posteriormente de forma programada. Por ejemplo, se puede reproducir, pausar o buscar un punto específico en el video que está cargado.

¿Cómo funciona el API de YouTube?

La biblioteca cliente de API interactúa con un servicio que se distribuye como parte de la aplicación de YouTube para la plataforma Android. La biblioteca cliente es deja una huella moderada, lo que significa que no repercutirá negativamente en el tamaño del archivo de tu aplicación si desea utilizar *ProGuard* como parte de tu proceso de compilación.

Implementación

Vista para mostrar videos de YouTube. El uso directo de esta vista es una alternativa al uso de *YouTubePlayerFragment*. Si usted lo decide puede usar esta vista directamente, su actividad debe ampliar *YouTubeBaseActivity.*

Para comenzar, se debe ubicar la vista en la jerarquía de vistas e invocar *initialize(String, OnInitializedListener)* para crear un YouTubePlayer, el que puede usar para cargar videos en esta vista.

Esta vista guardará y restaurará el estado del YouTubePlayer asociado, como parte del flujo *onSaveInstanceState/onRestoreInstanceState*. Si lo desea puede consultar la documentación de YouTubePlayer para obtener más información.