



OpenAni

Alejandro Stoian

Stephano Rayden Leon Estela

Fabio Junior Ninahuanca





1. Introducción.....	3
2. Estudio de mercado.....	4
2.1. DAFO.....	4
2.1.1. Fortalezas.....	4
2.1.2. Oportunidades.....	4
2.2. Cliente.....	5
3. Arquitectura de la aplicación.....	6
3.1. Estructura BB.DD.....	6
3.1.1. Entidades.....	6
3.1.2. Relaciones.....	6
3.2. Android Studio y React Native.....	7
3.2.1. Historias de usuario.....	7
4. Estilo de la aplicación.....	9
4.1. Paleta de colores.....	9
4.2. Fuente.....	9
4.3. Elementos visuales.....	9

1. Introducción

Cada día es mayor el problema que suponen las plataformas de streaming de pago en todo el mundo. En 2025, Netflix aumentó el coste de su suscripción una cantidad considerable, enmascarándola como un “ahorro del 30 %”¹ dando como razón una mejora de servicios.

No fue la única empresa en recurrir a la subida del precio de sus servicios y, sumándose a esto la gran cantidad de plataformas de streaming existentes hoy en día, hace que sea cada vez más difícil para el consumidor mantener todas las suscripciones activas.

El anime se ve afectado por la situación también. La serie “JoJo's Bizarre Adventure” se volvió propiedad de Netflix en 2021, convirtiéndola en la única plataforma en la que se puede ver. Nuestra aplicación ofrece un lugar donde poder encontrar todos los animes que uno se puede imaginar, incluso aquellos que no se encuentran en las plataformas mainstream.

El principal enfoque de la aplicación es funcionar como portal para poder descargar torrents de animes y mangas a través de la API de nyaa.si, una página web con funcionalidad parecida pero sin una versión móvil y con una interfaz primitiva. Todo esto sin tener que preocuparse ni de pérdida de calidad en la imagen, ni de suscripciones.

¹ [Netflix sube el precio de sus tarifas en España a partir de septiembre: este es el dinero que pagarás de más](#)

2. Estudio de mercado

2.1. DAFO

2.1.1. Fortalezas

- Código libre: Todo es transparente, el funcionamiento interno de la aplicación es visible y modificable para que se pueda mejorar y personalizar.
- Sistema de valoración: Poder dar “like” o “dislike” a los torrents ayuda a los usuarios a escoger el mejor de los torrents.
- API duradera: Nyaa Torrents lleva activa desde el 2005, lo cual la convierte en un sistema API de confianza, sin gran probabilidad de caer.
- Sistema de torrents: La propia página de [Nyaa.si](https://nyaa.si) no permite la subida de archivos de calidad baja, manipulados o con codecs cambiados. Todo el contenido que se sube a la API está verificada y moderada muy rigurosamente. Esto implica que los usuarios puedan ver anime en, muchas veces, calidad incluso más alta que la encontrada en plataformas de streaming.

2.1.2. Oportunidades

- Subida de precios: Mencionado en la introducción, la subida de los precios constante de las plataformas de streaming atrae a nuevos usuarios activamente.
- Eventos y convenciones: Nuestra aplicación podría salir promocionada en convenciones locales, ya sea de boca a boca o publicidad directa.

2.1.3. Debilidades

- Incapacidad de visualizar anime directamente en la aplicación: Muchos clientes potenciales se verán repelidos por no poder consumir anime o manga sin previamente descargarlo.
- Necesidad de una aplicación externa para la descarga de torrents
- Almacenamiento móvil: Como los torrents se descargan localmente, el cliente necesita disponer del almacenamiento

necesario.

2.1.4. Amenazas:

- Páginas web que permiten ver anime sin necesidad de descarga.
- Legalidad y Distribución: Google Play y App Store suelen banear aplicaciones que facilitan la descarga de contenido con copyright, por lo que tendríamos que apoyarnos en otros medios para distribuir nuestra aplicación, como GitHub directamente o FDroid.

2.2. Cliente

Esta aplicación está disponible para personas de todo el mundo y está pensada para quienes estén cansados de pagar múltiples suscripciones solo para poder ver una serie, leer, o consumir anime y manga.

Está dirigida a todo público sin una edad específica, ofreciendo una experiencia libre y centrada en los gustos de los amantes de la cultura asiática. Aunque la edad no influye en nuestros usuarios, lo que sí importan son los hobbies. Los usuarios de nuestra aplicación serían principalmente consumidores de este tipo de contenido. Por culpa de usar un sistema de torrents, los usuarios tendrán, en su mayoría, conocimientos tecnológicos básicos.

Al ser una aplicación móvil, está dirigida a aquellas personas que disponen de un dispositivo móvil inteligente, que sea capaz de manejar torrents.

3. Arquitectura de la aplicación

3.1. Estructura BB.DD.

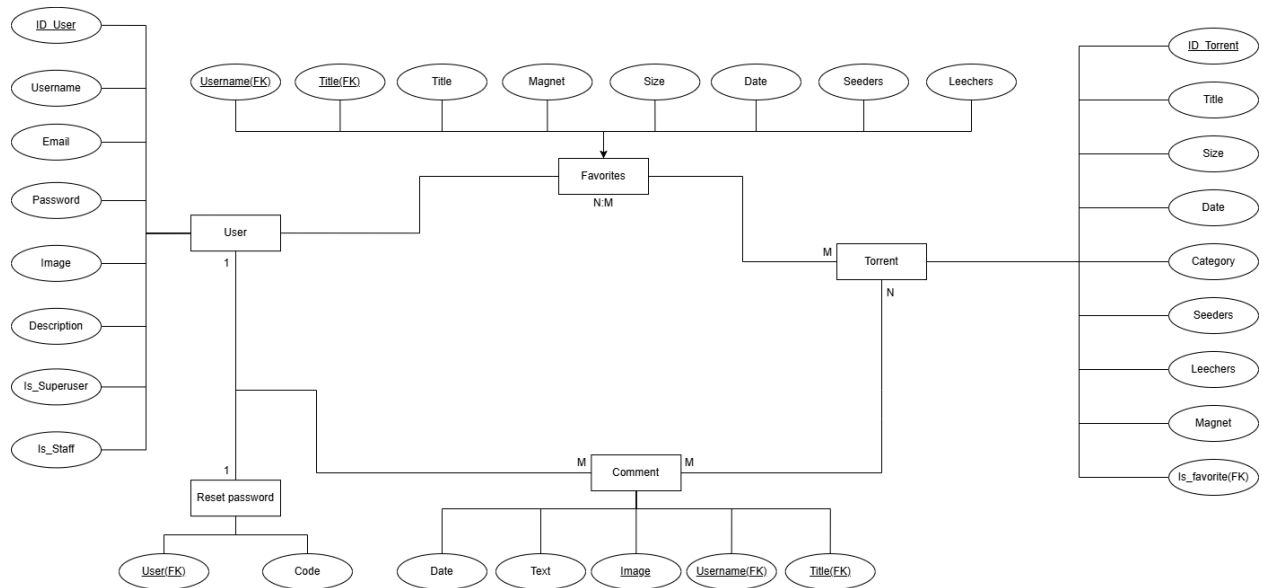
3.1.1. Entidades

- **User:** Almacena los datos de registro y perfil. Cada usuario puede tener muchos torrents como favoritos, y este mismo puede escribir muchos comentarios. El ID único es Key (ID_User, Username, Email, Password, Description, Image)
- **Torrent:** Desde la API se rescatan datos y se añaden a la base de datos, cada torrent tiene muchos comentarios y muchos usuarios pueden guardarlo como favorito. El ID único es Key (ID_Torrent, Title, Size, Date, Category, Seeders, Leechers, Likes, Dislikes, Enlace)
- **Comment:** Cada usuario puede hacer uno o muchos comentarios, y en esta entidad se guardan datos, algunos de los datos son recogidos por acciones de usuarios, como el escribir, el dar like o dislike, y automáticamente se genera la fecha en la que se publicó el comentario. El ID único es Key (ID_Comment, Text, Like, Dislike, Date).

3.1.2. Relaciones

- **Favorites:** Para que cada Torrent tenga la selección de favorito un User debe elegirlo, y cuando esto suceda las tablas se relacionan, los IDs que se agreguen en estas relaciones son Foreign Key. (ID_User, ID_Torrent)
- **Writes:** Cada User puede escribir un comentario, y cuando esto suceda se crea la entidad de Comment. Los IDs únicos son Foreign Key. (ID_User, ID_Comment)
- **Have:** Cada Torrent tiene comentarios que son creados por usuarios.

3.2. Android Studio y React Native



La aplicación se presenta en dos versiones. Una hecha en Android Studio y otra en React Native, intentando mantener el estilo lo más constante posible.

En Android, usando Java, la funcionalidad se gestiona con la utilización de varias librerías y dependencias. Para el manejo de la base de datos creada en Django, se necesita trabajar con JSONs. Para ello se usa Retrofit con el gestor Gson. Además se hará uso de Glide, elementos como RecyclerView e Intent implícitos para mandar el enlace .torrent a una aplicación externa (BitTorrent®, uTorrent®, Flud®, etc.)

En React se opta por usar TypeScript por su sencilla sintaxis. En esta versión se utiliza React Navigator para desplazarse por las distintas pantallas y Axios para la gestión de la base de datos Django. Para enviar el enlace .torrent a otra aplicación se usa la clase Share.

3.2.1. Historias de usuario

Se hacen peticiones constantes a la API, nuestra aplicación está constantemente en línea. Las peticiones posibles a la API de Nyaa son para recibir todos los animes en la pantalla principal. Esta después se podrá filtrar por categorías, indicadas en un menú lateral en la aplicación. Por una barra de búsqueda se podrá filtrar por nombre.

De la base de datos servidor Django que montaremos se harán peticiones para mostrar información de la cuenta de usuario, los comentarios, contador de likes y dislikes de los torrents. Los torrents solo se guardan en la base de datos solamente si tienen actividad social, para no sobrecargar la API Rest.

3.2.2. Definición de componentes

En Android Studio las tarjetas que definen los torrents están hechas usando el componente CardView. Estos se reciclan en un RecyclerView con scroll para ahorrar recursos. Hay distintos Intents para cada una de las 8 pantallas. Para los inputs de la pantalla de cuenta o de la barra de búsqueda se usan TextInput EditText, los botones están hechos con Button simplemente. Para la foto de perfil, como es un elemento circular, usamos un .xml personalizado que lo forme.

En React, reutilizamos los elementos de las tarjetas y los botones. Para las tarjetas implementamos [React Native CardViews](#) y lo utilizamos en conjunto con [RecyclerListView](#). Utilizamos ScrollView en vertical, ahí es donde utilizamos el RecyclerView.

Para los símbolos de las tarjetas utilizamos FontAwesome. Los botones son TouchableOpacity. Los inputs de los formularios de registro, la cuenta o la barra de búsqueda son TextInput.

4. Estilo de la aplicación

4.1. Paleta de colores

#48457C (Azul pizarra): Este color, además de ser principal, es utilizado porque representa ese “lado oscuro” de las aplicaciones y páginas piratas, dándole un aire a esas páginas. Este color se utiliza para los headers, haciendo que resalten.

Siguiendo la teoría del color, complementamos con otros colores que combinen bien y que no sean malos visualmente.

#FF8400 (Naranja brillante): Este color también es principal, hace un cambio un poco brusco pero para nada desagradable, pues

estos colores se utilizan para botones principalmente o también para fondos de entradas de texto. Es un color complementario del azul pizarra. Este color busca ser ese cambio de imagen de la aplicación.

#EEAD4D (Naranja claro): Esta es una variación del naranja brillante, siendo un poco más tranquila, utilizándose para los comentarios que los usuarios hagan.

#E0D222 (Amarillo mostaza): Este color se utiliza únicamente para el icono de favorito, es para un detalle pequeño, pero que sirve un poco para resaltar y mostrar diferencia de todo lo demás.

#000000 (Negro): Utilizado para bordes, sombras, texto, iconos, aportando contraste y profundidad a la aplicación.

#FFFFFF (Blanco): Utilizado para fondos, texto, iconos. Muestra claridad contrastando el header y acompañando el naranja y resaltando este mismo con el blanco.

4.2. Fuente

La principal fuente utilizada en la aplicación es Zud Juice, una de las tres fuentes usadas en las “scanlations” de los mangas². Esto va a la par con algunos elementos visuales que se mencionan en el siguiente apartado para crear una experiencia más cercana a nuestro usuario promedio.

4.3. Elementos visuales

Las tarjetas de los torrents son cuadradas y llenas de información como si fueran paneles de manga. Las casillas de los inputs tienen bordes ligeramente redondeados para parecer más ligeros a la vista. Los headers presentan un fondo estilizado con escenas de anime o de manga, dependiendo de la pantalla en la que se encuentre el usuario. Los iconos son de Material 3 en el caso de Android y de FontAwesome en el caso de React. Encima de la barra de búsqueda hay un pequeño detalle, una ilustración del Detective Conan con su lupa elevada, como si estuviera buscando. En la pantalla de perfil, el espacio donde se puede escribir la descripción está representado por una burbuja de texto, como si el usuario estuviese escribiendo un diálogo él mismo.

² [AnonBlack's Typesetting Guide](#)