UNIVERSIDAD NACIONAL AUTÓNOMA
DE MÉXICO

FACULTAD DE CIENCIAS

Desarrollo de un sistema para análisis de información obtenida de redes sociales

REPORTE DE TRABAJO PROFESIONAL

QUE PARA OBTENER EL TÍTULO DE:

Licenciado en Ciencias de la Computación

PRESENTA:

Luis Eduardo Miranda Sánchez

TUTORA

Amparo López Gaona

1.Datos del alumno

Miranda
Sánchez
Luis Eduardo
55 41 87 88 36
Universidad Nacional Autónoma de México
Facultad de Ciencias
Ciencias de la Computación
309199814

2. Datos del tutor

Dra.
Amparo
López
Gaona

3. Datos del sinodal 1

M. en I.
Gerardo
Avilés
Rosas

4. Datos del sinodal 2

Mat.
Salvador
López
Mendoza

5. Datos del sinodal 3

L. en C.C.
Oscar
Ruiz
Salinas

6. Datos del sinodal 4

M. en I.A.
Erick Orlando
Matla
Cruz

7.Datos del trabajo escrito.

Desarrollo de un sistema para análisis de información obtenida de redes sociales
N/A
81 p
2019

Índice general

Introducción

En este trabajo se describe el proceso de desarrollo de una aplicación web que recopila la información de distintas redes sociales, la almacena, la procesa, la analiza, la utiliza para crear reportes y los muestra a través de una interfaz web dinámica que actualiza su contenido en tiempo real.

Mini Cooper (MINI) es una marca de automóviles producidos por el fabricante alemán British Motor Company (BMW), esta marca de automóviles cuenta con un gran número de seguidores (followers) y consumidores en todo el mundo; en cuanto a seguidores, tan sólo en Facebook cuenta con más de 11 millones en 107 países y más de 2 millones de seguidores en América Latina (Panamá, México, Paraguay, Costa Rica, Perú, Chile, Guatemala, República Dominicana, Uruguay y Colombia).

MINI contrata a distintas agencias publicitarias en cada país donde tiene presencia para que con la ayuda de Creadores de Contenido Multimedia (Community Managers) atraigan seguidores a sus redes sociales. Los Community Managers crean contenido que da a conocer la marca a sus productos y lo publican en las redes sociales Twitter, Instagram y Facebook en los distintos formatos que permite cada red social, con esto, promueven la marca y atrae posibles consumidores de sus automóviles.

Para MINI es difícil medir el rendimiento por país, porque para revisar si las agencias están dando los resultados esperados, así como si están comunicando adecuadamente lo que representa la marca, si están cumpliendo con el número de publicaciones acordadas al mes o si está creciendo la comunidad, es necesario que ingresen a cada página de Facebook, Twitter o Instagram de MINI por país y "manualmente" cuenten las publicaciones, los seguidores, busquen las publicaciones más relevantes, etc.

La complejidad de la revisión está en que acceder a cada red social por país es tardado y son demasiadas cuentas de redes sociales para hacer la revisión de las redes sociales de América Latina, que consta de 10 perfiles de Facebook, 8 de Instagram y 6 de Twitter lo que da un total de 24 perfiles, de cada uno se debe tener el usuario y contraseña para acceder a los perfiles, ya dentro de cada uno se procede a hacer el conteo de las interacciones, la revisión de los comentarios hechos por los seguidores, el conteo de seguidores y todo esto de forma manual. En todo este trabajo a realizar debe considerarse que se tienen aproximadamente 71,658 seguidores en sus redes de Twitter, 27,524 seguidores en sus redes de Instagram y 1,275,268 en sus redes de Facebook.

Con el fin de dar solución a la necesidad de MINI de gestionar de forma sencilla la información de sus redes sociales se desarrolló una aplicación web que reúne la información de 3 distintas redes sociales en las que MINI tiene comunidad, genera reportes y los muestra mediante una interfaz amigable e intuitiva que actualiza su contenido en tiempo real.

De las redes sociales se obtuvieron los datos de las publicaciones, de los comentarios, de sus seguidores y la información pública de estos, de toda esta información recopilada se analizaron los datos y se hicieron reportes acerca del crecimiento de seguidores, de las publicaciones más relevantes para sus seguidores, palabras más usadas en sus publicaciones y de los seguidores que más interactuaron con las publicaciones.

Este sitio web puede ser visitado en cualquier momento y desde ahí se puede ver el crecimiento de seguidores, publicaciones y otros datos para que así un administrador de MINI pueda medir el desempeño de las agencias publicitarias contratadas en cada país y actuar de inmediato dependiendo de los resultados que observe, produciendo así una mejora en la productividad de la marca.

El objetivo de este sitio es almacenar los datos de los perfiles de Facebook, Twitter e Instagram de MINI que pueden obtenerse a través de consultas a su Interfaz de Programación de Apicaciones (API), los datos obtenidos, son procesados y después se muestran en un sitio web dinámico del cual se garantiza su dispo-

nibilidad y la integridad de los datos mostrados. Esto es de utilidad para el administrador de las redes de MINI, porque tener la información recopilada en un sólo sitio web, agiliza sus procesos y tareas.

CAPÍTULO 1

Definición del problema

1.1 Antecedentes de MINI

Mini Cooper es una empresa dedicada a la producción de automóviles a nivel internacional. En América Latina y los países de habla hispana Mini Cooper tiene presencia en Panamá, México, Paraguay, Costa Rica, Perú, Chile, Guatemala, República Dominicana, Uruguay y Colombia. Para su publicidad, MINI contrata a agencias publicitarias en cada país para dar a conocer sus productos y automóviles. En Latinoamérica cada país tiene diferentes redes sociales, la distribución de perfiles en redes sociales por país es la siguiente:

País	¿Tiene Facebook?	¿Tiene Twitter?	¿Tiene Instagram?
Panamá	Sí	Sí	Sí
México	Sí	Sí	Sí
Paraguay	Sí	Sí	Sí
Perú	Sí	No	Sí
Chile	Sí	Sí	Sí
Guatemala	Sí	No	Sí
República Dominicana	Sí	Sí	Sí
Colombia	Sí	Sí	Sí
Costa Rica	Sí	No	No
Uruguay	Sí	No	No

1.1.1 Redes sociales

Cada vez más gente tiene redes sociales, pasa más tiempo en ellas y da más prioridad a lo que llega a través de ellas frente a lo que llega de otros canales de comunicación. La importancia de su uso en las empresas es mayor cuanto más grande es el uso de estas por parte de los usuarios y su uso se convierte en algo imprescindible.

Las redes sociales son un nuevo canal de negocios para las empresas, en el que captan a nuevos clientes, clientes a los que no podían llegar facilmente por medio de la **publicidad convencional**. Se conoce por publicidad convencional a aquella que utiliza mecanismos tradicionales (comprar espacio en medios como prensa, radio, televisión, cine, publicidad exterior o teléfono) para dar a conocer una marca, servicio o producto con el fin de crear una relación con el consumidor.

El uso de medios no convencionales como las redes sociales responde a la búsqueda de eficacia, a la necesidad de lograr contactactar con el público y que su mensaje se recuerde y acepte. Las redes sociales no sustituyen a la publicidad convencional, por el contrario, la publicidad digital también se usa como un complemento de estos medios. Cada empresa debe elegir cuál o cuáles son las estrategias que se adaptan más a ella, tomando en cuenta su rentabilidad y efectividad.

El uso de redes sociales sólo es una de varias estrategias digitales que han revolucionado la forma de hacer negocios; SEO (Search Engine Optimization), SEM (Search Engine Marketing), Email Marketing, Display (Situar anuncios o banners en un sitio web), Video Marketing son algunas de ellas.

Las empresas encuentran en las redes sociales un canal en el cual anunciar sus productos de forma bidireccional, creando relaciones más sólidas con los clientes potenciales, que les ayudan a realizar cualquier cambio que el mismo podria necesitar para mejorar su posicionamiento.

Las redes sociales, son una excelente herramienta de segmentación del público, ya que de cada persona se pueden conocer los atributos que los subdividen en grupos sociales, atributos tales como: la edad, el sexo, sus aficiones, preferencias de entretenimiento, etc. Toda esta información es tomada por las marcas para que puedan dirigir su mensaje al publico correcto.

Lo que convierten las redes sociales en un canal para los negocios es la cantidad de usuarios que tiene cada una de ellas. Los usuarios activos en un mes de las principales redes sociales a día de hoy[1], son : Instagram que ha superado la cifra de 1,000 millones de usuarios activos mensuales, Twitter que cuenta con más de 326 millones de usuarios activos mensuales, Facebook con cerca de 2,270 millones de usuarios activos mensuales. LinkedIn con 303 millones y YouTube con 1.900 millones.

1.2 Planteamiento del problema

MINI tiene un director de marketing que gestiona el correcto manejo de las redes sociales de la marca en los países de habla hispana de América Latina, pero la gestión se vuelve difícil conforme más redes sociales por país tiene la marca, ya que tener control sobre todas éstas implica: tener las credenciales para acceder a cada una de ellas, ingresar a cada una y hacer reportes manuales sobre lo que se observa en los perfiles (crecimiento de seguidres, comentarios hechos por los seguidores y más).

Por otro lado, el administrador de las redes de MINI no puede visualizar de forma sencilla y en tiempo real las publicaciones que realizan las agencias publicitarias que se contrataron en cada país para publicitar la marca, y no es suficiente el reporte que entregan las agencias sobre la actividad en las redes sociales y el crecimiento de la comunidad de MINI. ya que en caso de ocurrir alguna incidencia o que no se cumpliera con alguna publicación acordada, el administrador no podría actuar de inmediato para resolver los problemas.

[1] El top de páginas de redes sociales fue compilado por Statista

1.3 Análisis de requerimientos

Derivado de lo anterior, surge la importancia y la necesidad de crear una aplicación web dinámica que se alimenta de las publicaciones en redes sociales de Mini Cooper de cada país y las muestra en tiempo real, para que así el administrador pueda agilizar las siguientes tareas por red social:

Twitter :

- Ver los últimos Tweets y ver sus favoritos(favs), comentarios(replies), compartir(retweets).
- Ver los Tweets que más han tenido impacto.
- Ver la cantidad total de seguidores (followers).
- Ver lista de seguidores más activos, es decir, seguidores que más han interactuado con las publicaciones del perfil.
- Ver el total de Tweets, este es la cantidad total de publicaciones hechas por el Community Manager.

Facebook :

- Ver la cantidad de seguidores.
- Ver las últimas publicaciones y ver sus "me gusta" (likes), comentarios y cantidad de veces compartida.
- Ver las publicaciones más que han tenido impacto.
- Ver las publicaciones y comentarios de sus seguidores.
- Ver lista de seguidores más activos, es decir, seguidores que más han interactuado con las publicaciones del perfil.

Instagram :

- **Ver la cantidad total de seguidores.**
    - Ver de las últimas publicaciones sus favoritos y comentarios.
    - Ver las publicaciones más relevantes que más han tenido impacto.

1.4 Propuesta de solución

La propuesta consta del desarrollo de un "sistema de administración de redes sociales" que cuente con un panel desde el cual los usuarios administrativos de MINI puedan ver en tiempo real y en cualquier momento lo que ocurre en las redes sociales de cada país.

El diseño de la interfaz de usuario fue creado por la agencia y consultora creativa Social Noise, la agencia creó propuestas diseño que cubrían las necesidades de MINI, pero los primeros diseños contemplaban funciones que no podían realizarse con las herramientas de consulta de Twitter, Facebook e Instagram. Por lo anterior, se desarrolló una página a la medida que cumpliera con la mayor parte de los requerimientos del cliente y siguiera las propuestas creadas por la agencia.

En cada una de las propuestas y capturas de pantalla se podrá ver lo siguiente:

- La sección principal o central donde se visualiza la información de la red social y país.
- Una barra de desplazamiento entre paises y redes sociales, en esta se indica que red social se está viendo y a que país pertenece. Al interactuar con ella se cambia el contenido de la sección principal a la del país y red social seleccionada.
- Una nube de palabras(*Tag Cloud*) que es una representación visual de las palabras que conforman un texto, en donde el tamaño es mayor para las palabras que aparecen con más frecuencia, en ésta se muestran las palabras que fueron utilizadas en los comentarios por los usuarios o en las publicaciones hechas por la marca.
- Una sección que muestra el total de seguidores del mes actual y del mes pasado con la intención de revisar el crecimiento/decrecimiento de seguidores de manera sencilla.
- Una sección que muestra el crecimiento/decrecimiento de engagement (ver: *Engagement*). El engagement es el compromiso y la forma en que interactúa un seguidor con una marca. Se trata de un término que mide de cierta forma el grado de interacción entre una marca y los usuarios.

A continuación se muestran las primeras propuestas de diseño presentadas por la agencia y capturas de pantalla de la página web desarrollada.

1.4.1 Interfaz para visualizar las publicaciones de Facebook de un país.

Muestra la última publicación, publicación más relevante, publicaciones y comentarios de los usuarios y fans más activos.

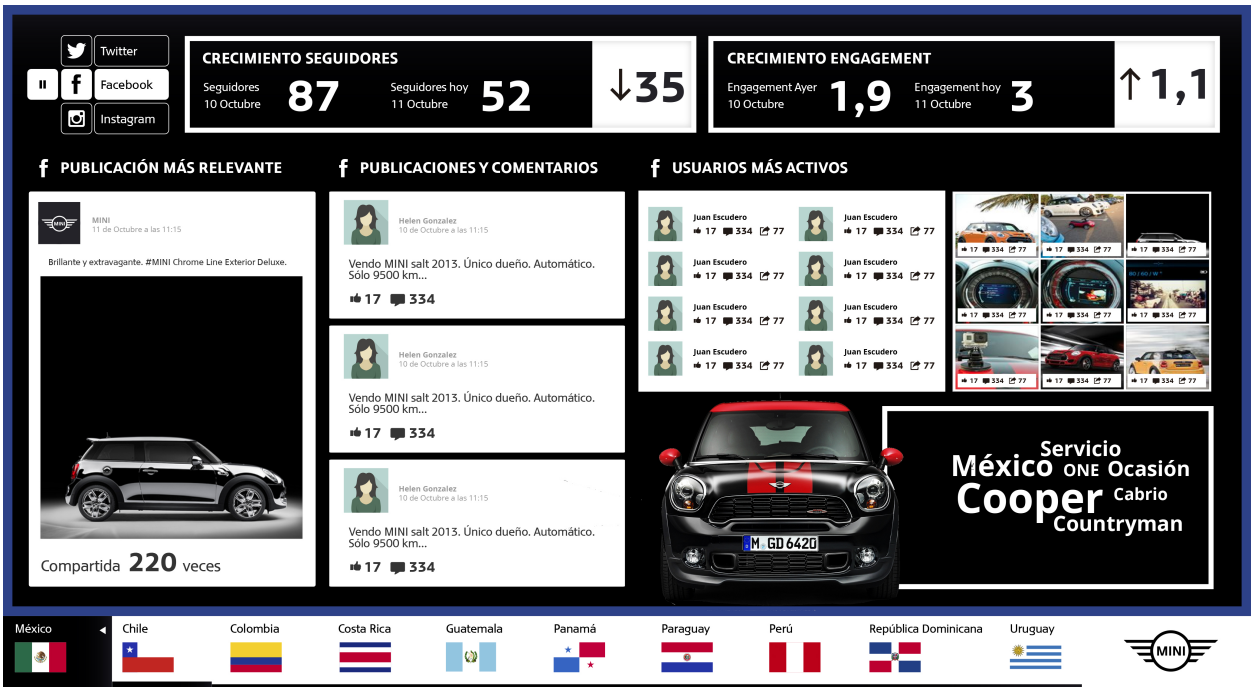


Figura 1.1: Primera propuesta de diseño para Facebook.
Fuente: Social Noise

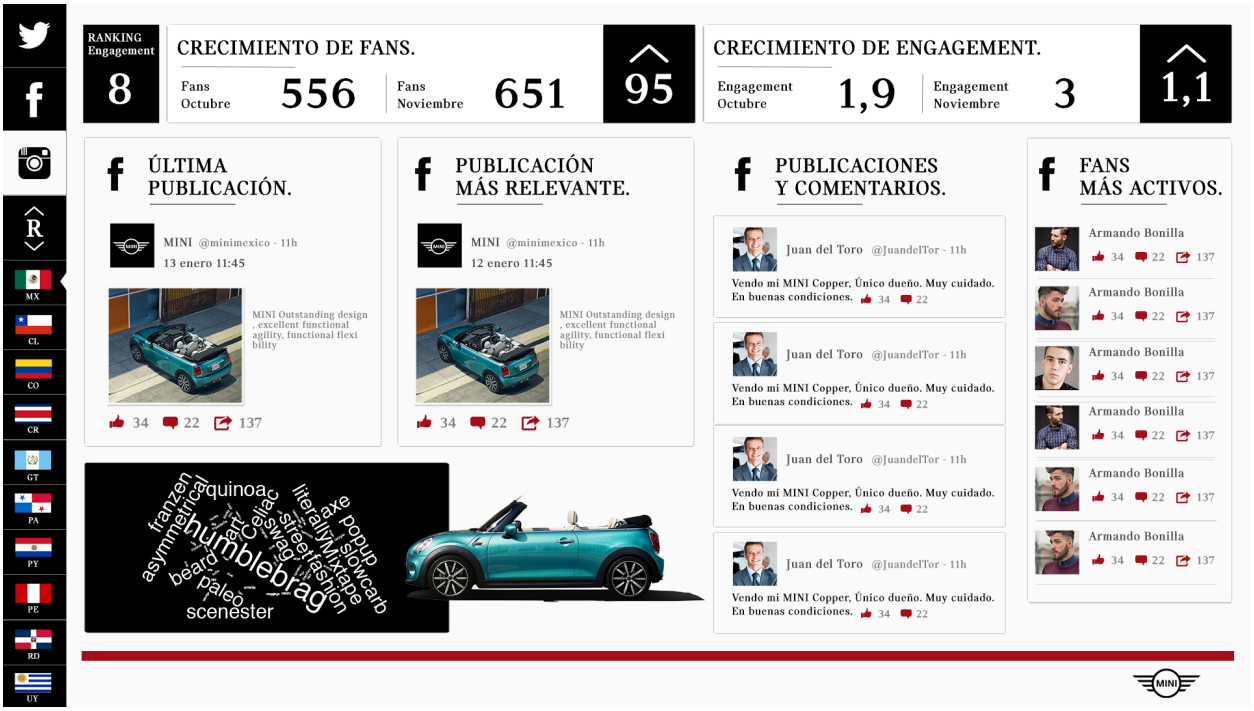


Figura 1.2: Segunda propuesta de diseño para Facebook.
Fuente: Social Noise

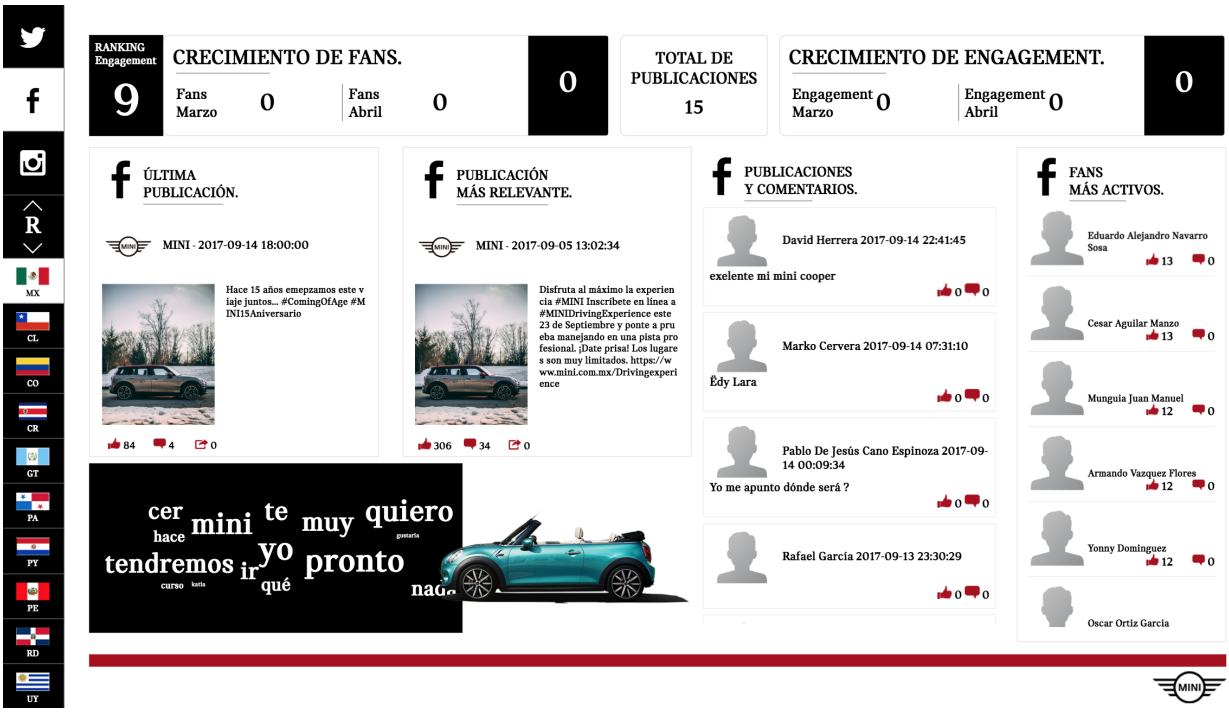


Figura 1.3: Captura de pantalla del sitio desarrollado en la sección de Facebook.

1.4.2 Interfaz para visualizar la información de las publicaciones de Twitter de un país.

Muestra los últimos Tweets, los Tweets más relevantes y seguidores más activos.

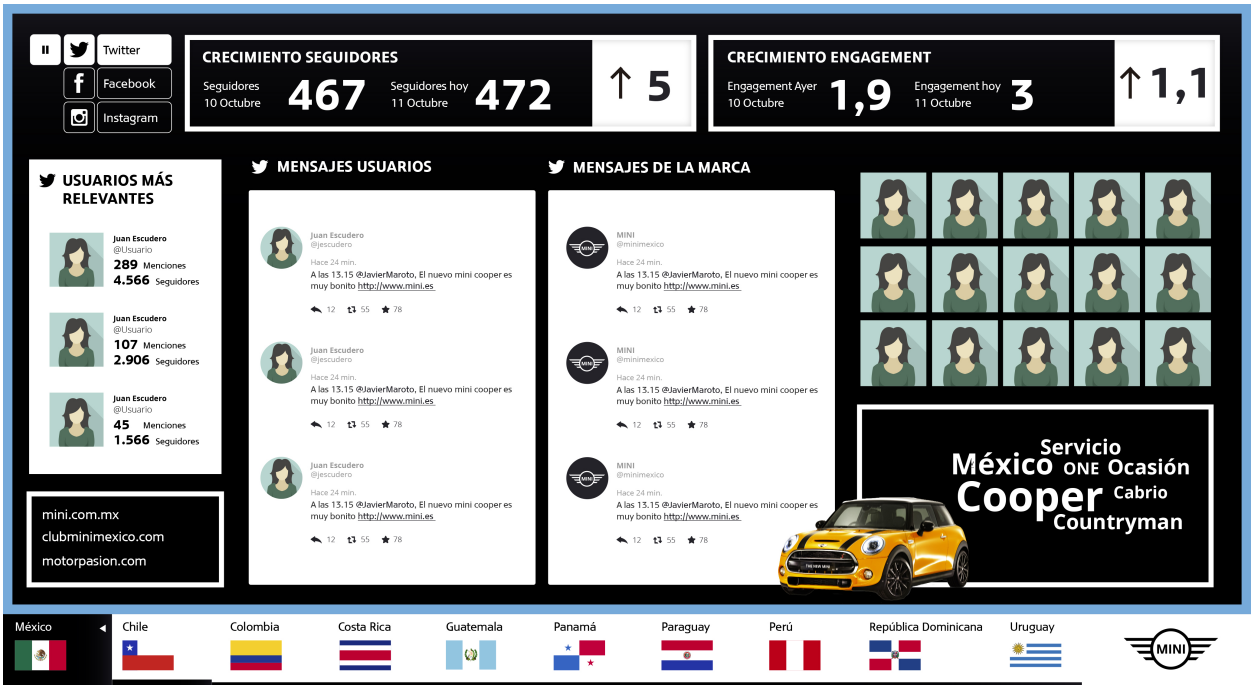


Figura 1.4: Primera propuesta de diseño para Twitter.
Fuente: Social Noise

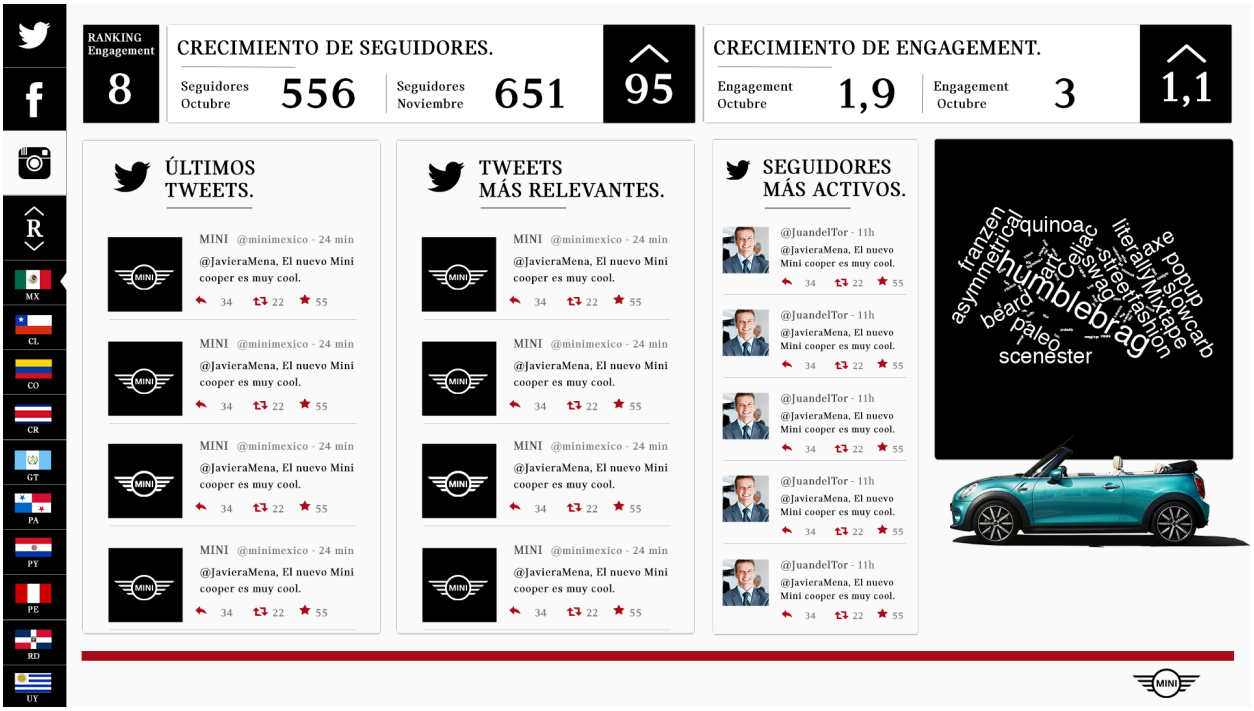


Figura 1.5: Segunda propuesta de diseño para Twitter.
Fuente: Social Noise

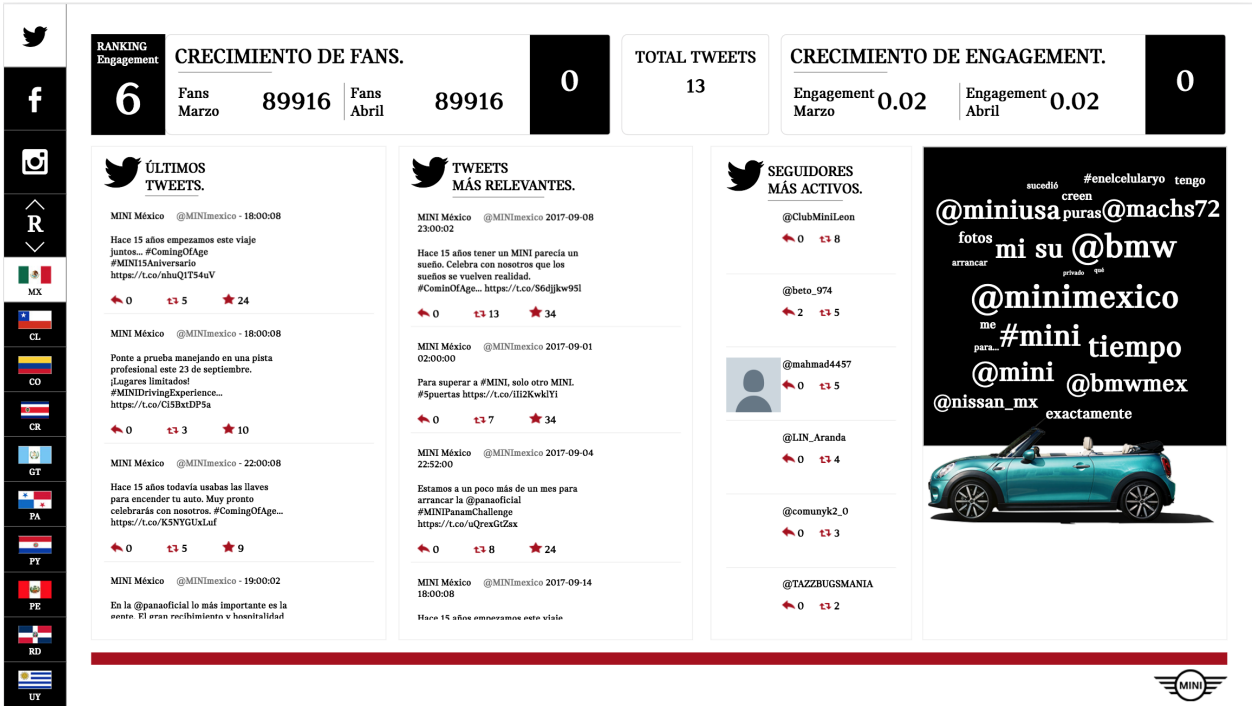


Figura 1.6: Captura de pantalla del sitio desarrollado en la sección de Twitter.

1.4.3 Interfaz para visualizar la información de las publicaciones de Instagram de un país.

Muestra las publicaciones más recientes y la publicación más relevante.

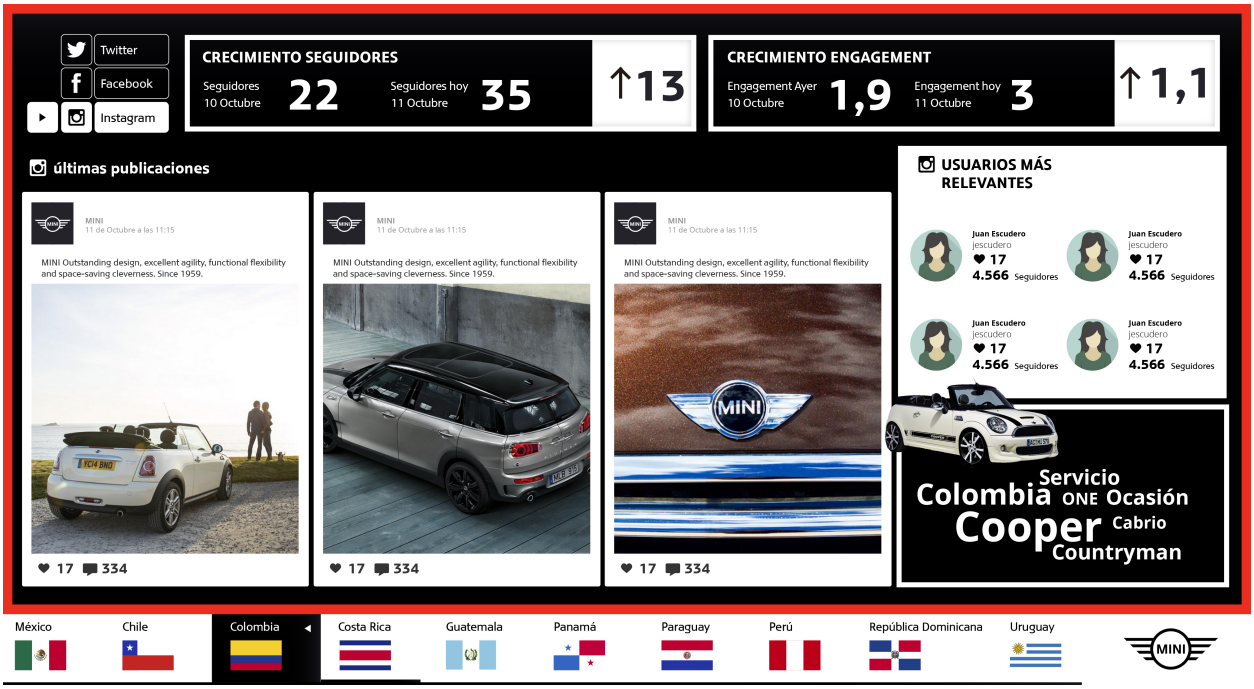


Figura 1.7: Primera propuesta de diseño para Instagram.
Fuente: Social Noise

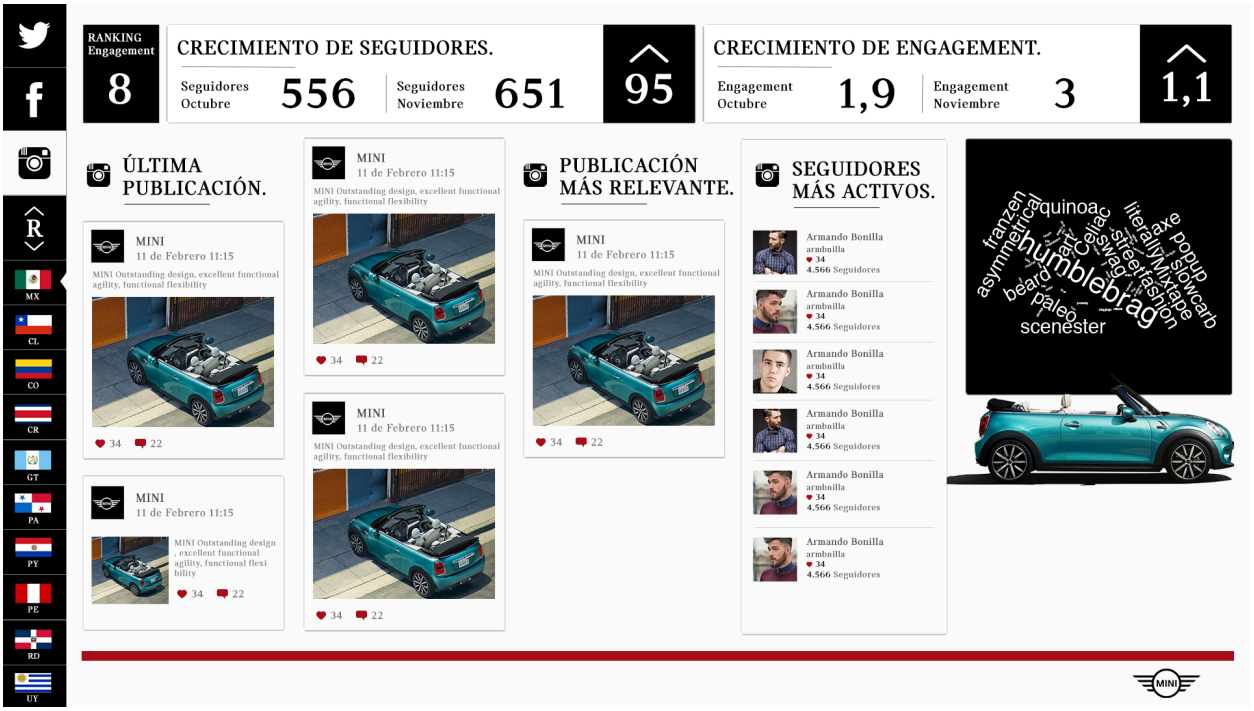


Figura 1.8: Segunda propuesta de diseño para Instagram.
Fuente: Social Noise

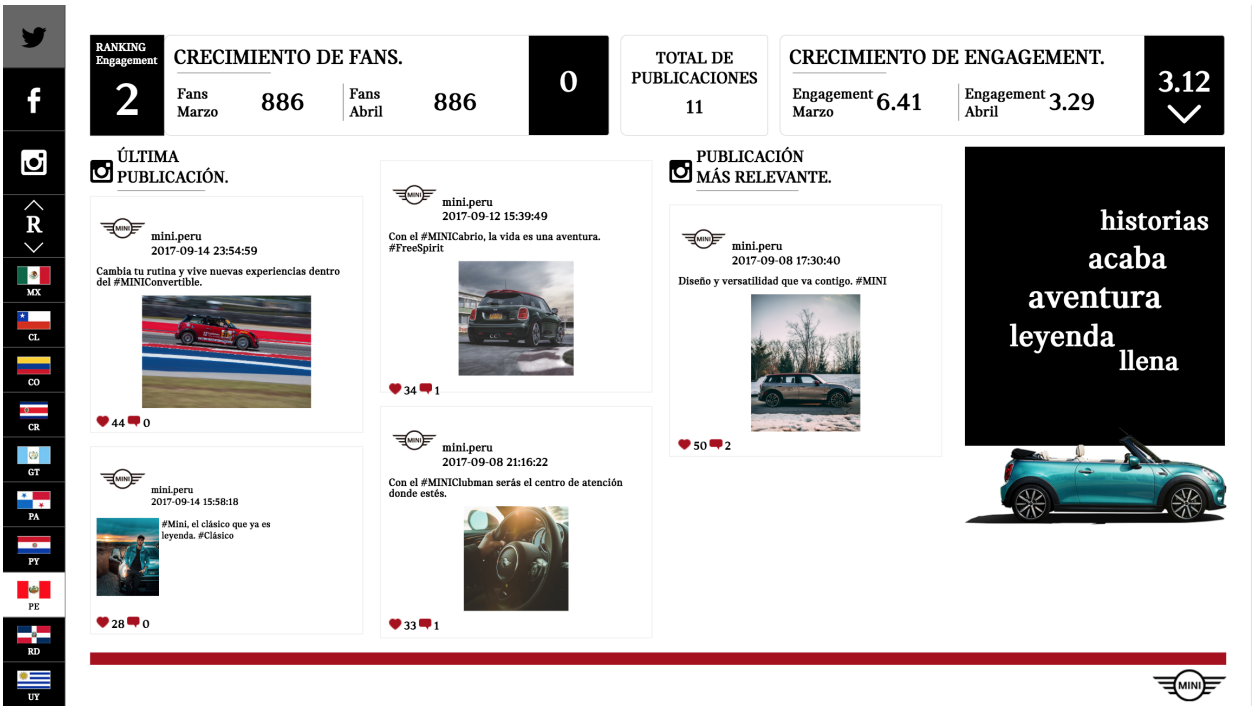


Figura 1.9: Captura de pantalla del sitio desarrollado en la sección de Instagram.

Las figuras 1.1, 1.2, 1.4, 1.5, 1.7 y 1.8 son las primeras propuestas de diseño creadas por Social Noise, no se pudieron seguir estos primeros diseños porque hubo limitantes de las APIs de las redes sociales para obtener el contenido que era necesario que se mostrará, y porque hubo requerimentos extra, por ejemplo:

- Con el API de Instagram[2] no es posible obtener información de los usuarios que publican en la página, dan *me gusta*, comparten o comentan las publicaciones. Para obtener la información de un usuario es necesario tener su autorización explícita sólo para obtener su nombre o imagen de perfil, esto no pasa con Facebook, ya que tan sólo con el identificador de un usuario se pueden obtener los mismo datos.
- Para la sección de Twitter se agregó lógica para poder manejar además de texto una imagen en cada publicación ya que la primera y segunda propuesta se contemplaba únicamente texto.

Nota: En las figuras correspondientes a la segunda propuesta de diseño, en la barra lateral se encuentra seleccionada la red social Instagram aunque el contenido principal corresponda a otra red social, esto fue un error por parte de Social Noise. En las capturas de pantalla del sitio desarrollado, se puede ver como la selección de red social si corresponde con el contenido de la sección principal.

1.4.4 Barra superior

En la parte superior de las figuras anteriores encuentra una barra como la siguiente, la cual muestra el crecimiento/decrecimiento de seguidores y crecimiento/decrecimiento de engagement del mes anterior y el mes actual.

Figura 1.10: Propuesta de diseño para la barra superior.

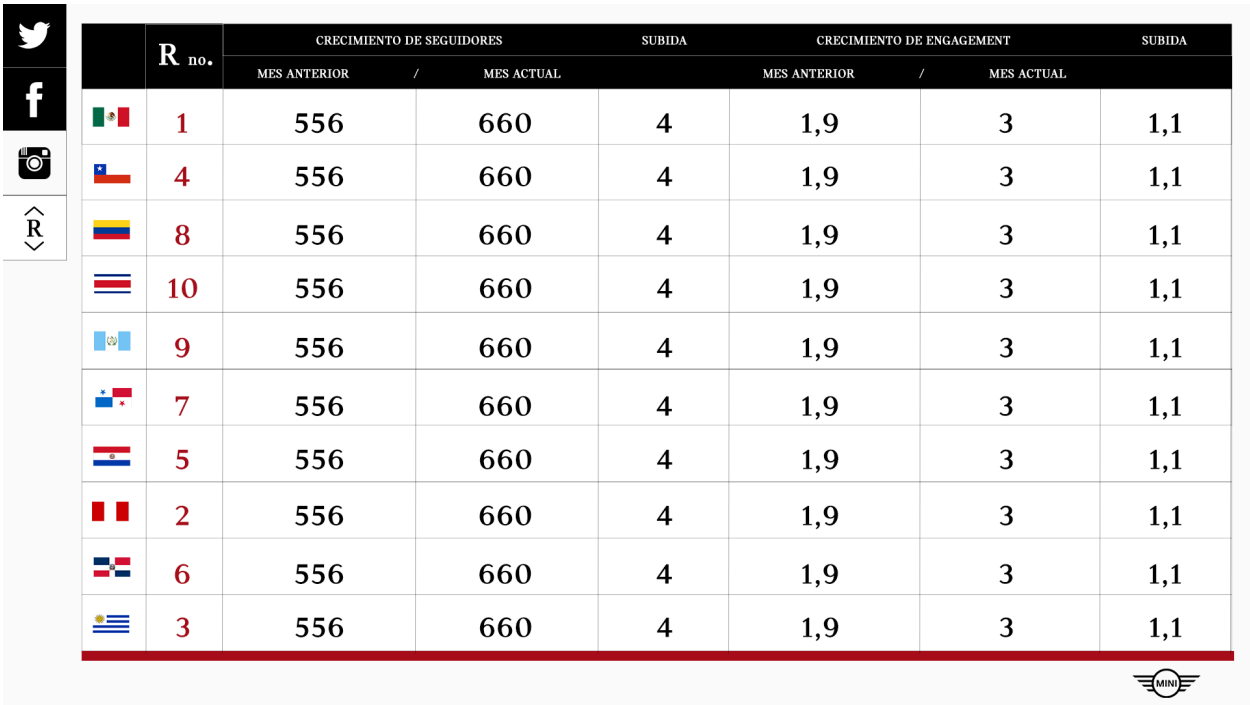
Figura 1.11: Barra desarrollada para la página.

Como se puede ver en la figura anterior, en la página desarrollada se incluyó el **Total de Publicaciones** por solicitud de MINI, ya que no podían ver en ninguna sección cuántas publicaciones se habían hecho y no se podían ver en la sección principal porque sólo se mostraban las publicaciones más recientes o las más populares de cada red social.

[2] Permisos del API de Instagram - https://www.instagram.com/developer/authorization/

1.4.5 Ranking de comunidad

Propuesta de diseño para ver el ranking de comunidad y engagement por red social y país. La intención de esta sección es visualizar de manera simple el desempeño que está teniendo cada red social por país.



	R no.	CRECIMIENTO DE SEGUIDORES		SUBIDA	CRECIMIENTO DE ENGAGEMENT		SUBIDA
		MES ANTERIOR	/ MES ACTUAL		MES ANTERIOR	/ MES ACTUAL	
	1	556	660	4	1,9	3	1,1
	4	556	660	4	1,9	3	1,1
	8	556	660	4	1,9	3	1,1
	10	556	660	4	1,9	3	1,1
	9	556	660	4	1,9	3	1,1
	7	556	660	4	1,9	3	1,1
	5	556	660	4	1,9	3	1,1
	2	556	660	4	1,9	3	1,1
	6	556	660	4	1,9	3	1,1
	3	556	660	4	1,9	3	1,1

Figura 1.12: Primera propuesta para el Ranking de Comunidad.
Fuente: Social Noise

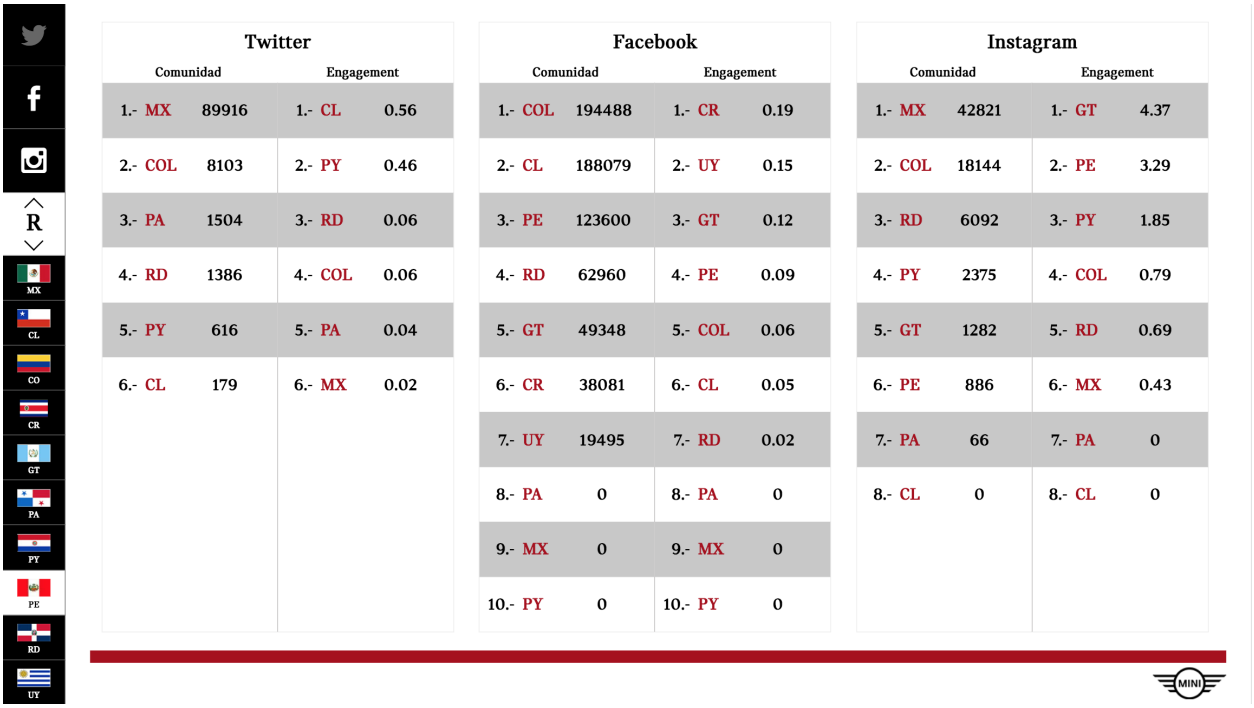


Figura 1.13: Captura de pantalla del sitio desarrollado en la sección del Ranking de Comunidad.

En esta sección se muestra condensada y ordenada toda la información que es visible en la barra superior de cada país y red social. La información se ordena primero por número de seguidores y después por engagement.

1.4.6 Engagement

Las fórmulas para calcular el engagement fueron proporcionadas por Mini Cooper y las creó para cumplir con sus necesidades ya que no existe un concenso para definir este término ni una fórmula universal para calcularlo. Las fórmulas que proporcionó son las siguientes:

Engagement Facebook

```
// Likes = Me gusta
// Fans = Seguidores

(((Σ Comentarios + Σ Likes + Σ Shares) / Σ Publicaciones ) / Fans)*100
```

Engagement Twitter

```
// Retweets = Compartidos
// Replies = Comentarios
// Tweets = Publicaciones

(((Σ Retweets + Σ Replies + Σ Me gusta) / Σ Tweets) / Followers)*100
```

Engagement Instagram

```
// Likes = Me gusta
// Followers = Seguidores

(((Σ Comentarios + Σ Likes) / Σ Publicaciones) / Followers)*100
```

CAPÍTULO 2

Creación de la base de datos

En este capítulo se describe el proceso de creación de la base de datos a partir de lo visto en las redes sociales.

2.1 Almacenamiento

Uno de los pasos más importantes durante la creación de la aplicación web fue el diseño y la elaboración de la base de datos para el almacenamiento de todos los datos obtenidos por las consultas al API de cada red social.

Se eligió el tipo de base de datos que usaría de acuerdo a su propósito y teniendo en cuenta qué tipos de datos almacenaría. Se decidió utilizar una estructura de datos relacional (SQL) en vez de una no relacional (NoSQL) porque previamente se realizaron pruebas consultando a las API y se investigó en las documentaciones de las redes sociales acerca de los tipos de datos devueltos de las peticiones al API. De las pruebas e investigación, se identificó que había datos devueltos de las consultas que estarían relacionados, por ejemplo, un usuario está relacionado a una publicación si este usuario fue el que la creó y a su vez este usuario está relacionado a otra publicación, si interactuó con ella.

Además, se sabía bien que la cantidad de datos a almacenar no aumentaría, es decir, se sabía bien que datos se tomarían de las respuestas del API y que datos se almacenarán en la base. Si el número de información que hubiera que almacenarse fuera variable, hubiera optado por MongoDB que es una base de datos no relacional, porque permite almacenar los esquemas sin importar el número de atributos tengan y el agregar un atributo más o quitarlo es inmediato.

Durante las pruebas e investigación, se recopilaron todos los tipos de información que se almacenarán en la base de datos, como nombres, mensajes, identificadores de publicación, número de likes etc. Identificar los datos y su tipo ayudó a tener una idea de las entidades que se tenían que crear y cómo podrían relacionarse. A continuación se listan algunas muestras de los datos:

- `facebook_id`

  - Descripción : Identificador de perfil de Facebook.
  - Tipo de Dato : Cadena.
  - Longitud máxima encontrada: 15 caracteres.
  - Ejemplo: `162862883750468`

- `access_token`
  - Descripción : API Access Token.
  - Tipo de Dato : Cadena.
  - Longitud máxima encontrada: 202 caracteres.
  - Ejemplo: *CAAK5tbBqNKEBANy0m0OQ5WUY4Hg1JNtF4ZAt2fNiA42ActrrOo3SRg AFZBxXHi3UnGLKZAqvxTvTZBifaoGKVM41oVJ3ZB0lWd3uIyLcZC6XM4ZAX b5HrZBw1DAroSJP4eMhG6nZAPV4MfGQthcDZAwOrSHJRb6rU428oxSZAMhZBUu4 BPU2g00aPoD6HxlM07N0iy444*

- `picture`
  - Descripción : URL[#f1]_ de las imágenes publicadas.
  - Tipo de Dato : Cadena.
  - Longitud máxima encontrada: 388 caracteres.

- `facebook_post_id`
  - Descripción : Identificador de post de Facebook.
  - Tipo de Dato : Cadena.
  - Longitud máxima encontrada: 33 caracteres.
  - Ejemplo: `156096364439354_10151350209245217`

- `facebook_message`
  - Descripción : Texto de publicación de Facebook.
  - Tipo de Dato : Texto.
  - Longitud máxima encontrada: 1,729 caracteres.
  - Límite de caracteres en publicación en Facebook: 5,000.

- `twitter_message`
  - Descripción : Texto de publicación de Twitter.
  - Tipo de Dato : Cadena.
  - Longitud máxima encontrada: 140 caracteres.
  - Límite de caracteres actual en publicación: 280.
  - Ejemplo: *Visítanos en la Feria Expomóvil Banreservas y pública tu foto en el #MINIwall. La foto con más "likes", gana un bon… https://t.co/umv5VwlNmd*

- `instagram_message`
  - Descripción : Texto de publicación de Instagram.
  - Tipo de Dato : Texto.
  - Longitud máxima encontrada: 498 caracteres.
  - Límite de caracteres actual publicación: 2,200.
- `facebook_username`
  - Descripción : Nombre de Usuario.
  - Tipo de Dato : Cadena.
  - Longitud máxima encontrada: 65 caracteres.
  - Ejemplo: *Save Beach Ventanilla—Peru ; Salvemos la Playa de Ventanilla—Perú*
- `facebook_users_message`
  - Descripción : Mensaje publicado por un usuario a un perfil de Facebook.
  - Tipo de Dato : Texto Mediano.
  - Longitud máxima encontrada: 6,952 caracteres.
  - Límite de caracteres en publicación en Facebook: 8,000.

Se dividieron los elementos de información en 3 grupos, cada grupo contenia tablas que servían para almacenar datos de Twitter, Facebook o Instagram. Cada grupo tiene una tabla principal desde la cual parten las relaciones de su grupo.

La tabla principal de cada grupo contiene el identificador de la red social, país al que pertenece, nombre (MINI México, MINI USA, etc), el token que autoriza las peticiones al API, URL de la imagen de perfil y un atributo que indica el perfil necesita actualización.

Para visualizar y comprender mejor la base de datos que se crearía, se elaboró un diagrama entidad relación (ER) que contiene los elementos que debía contener la base de datos. A continuación se muestra una parte del diagrama donde se puede ver la entidad principal del grupo Facebook y las entidades relacionadas a el.

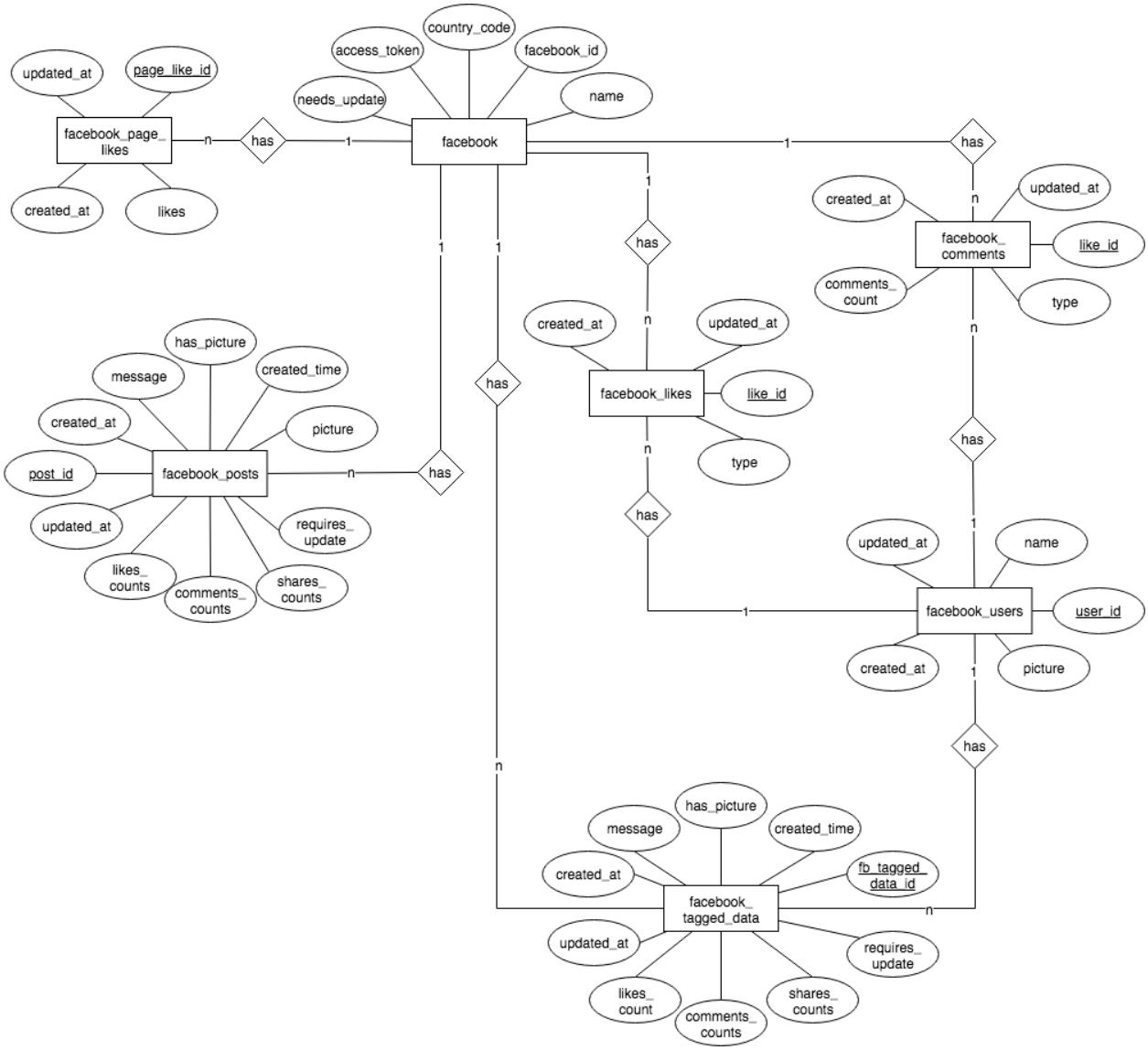


Figura 2.1: Diagrama Entidad Relación de las tablas de Facebook

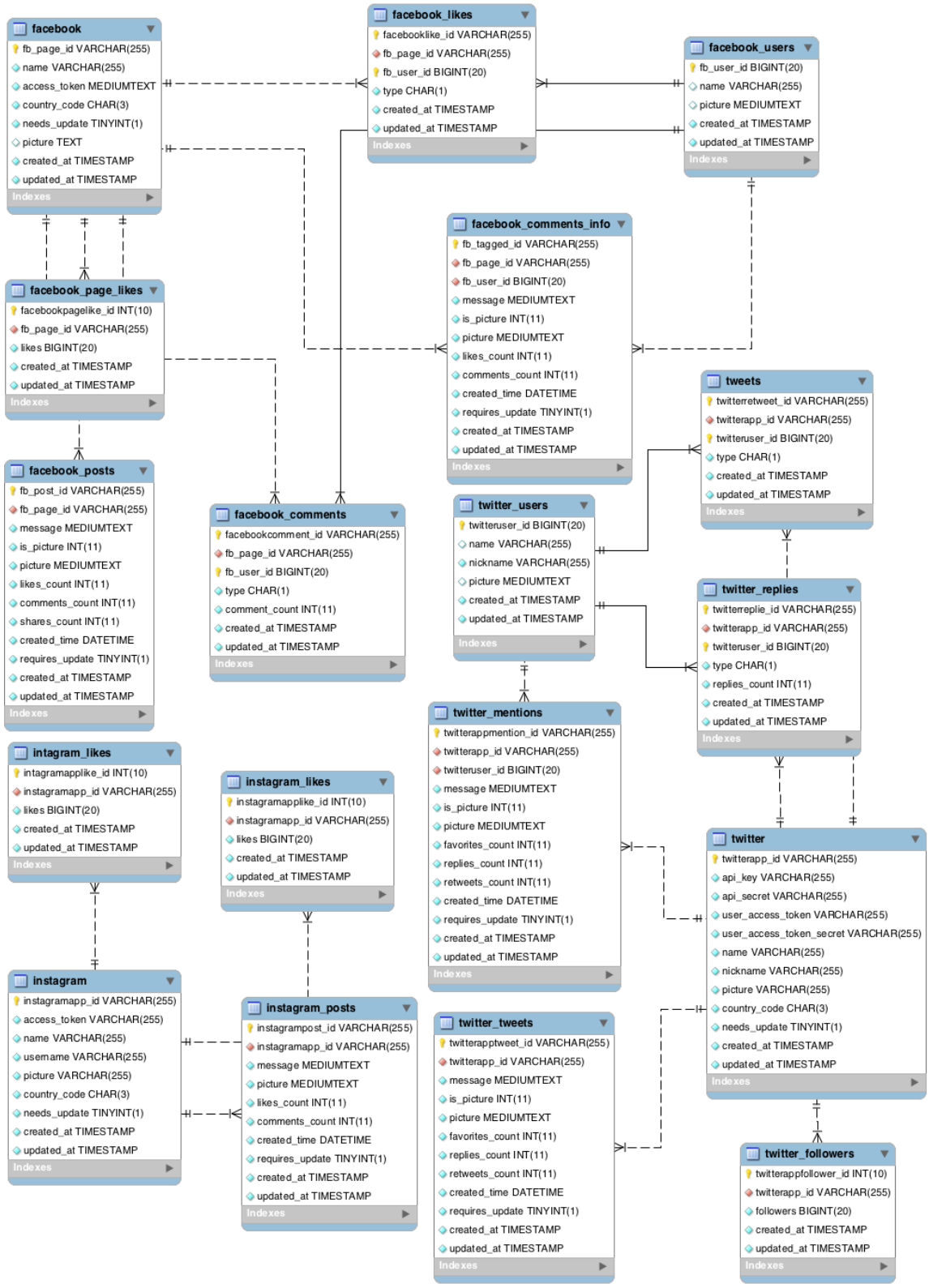


Figura 2.2: Diagrama relacional de la base de datos creada.

Se crearon 18 tablas en la base de datos, tres tablas son las principales y almacenan los datos de las redes sociales por país, a continuación se explica brevemente la función de las tablas principales.

2.1.1 Tablas principales

facebook

La tabla de **facebook** almacena los datos importantes acerca de la red social por cada país y los token de acceso necesarios para poder autenticar las consultas al API.

Esta tabla cuenta con nueve tuplas y los datos correspondientes a las siguientes páginas de Facebook:

Identificador	Código	Nombre
119466091417535	PA	MINI Panama
119626204728343	MX	MINI México
124553437626427	PY	MINI Paraguay
128870127136405	CR	MINI Costa Rica
143395282379300	CL	MINI Chile
156096364439354	GT	MINI Guatemala
158528990851562	RD	MINI República dominicana
162862883750468	UY	MINI Uruguay
55724017509	COL	MINI Colombia

Si se toma el identificador y se le antepone la URL https://www.facebook.com/ se obtiene la URL a la página de fans en Facebook del país correspondiente. Facebook usa la ubicación del usuario para inferir qué página se debe visitar. Por lo que si se intenta ingresar a https://www.facebook.com/158528990851562 que es la URL de página de Facebook de MINI República Dominicana y nuestra ubicación es México esta URL nos redireccionará al perfil de Facebook de MINI México .

twitter

La tabla de **twitter** almacena los datos importantes de cada perfil de Twitter por país y los token de acceso para autenticar las peticiones. Twitter requiere de 4 token: **api_key**, **api_key_secret**, **user_access_token** y **user_access_token_secret**.

Nota: Los **user_access_token** autorizan las peticiones al API para obtener información del perfil propio[2], las **api_key** autorizan las peticiones al API para obtener información de otros perfiles de Twitter[3].

---

[2] Twitter Access Tokens - https://developer.twitter.com/en/docs/basics/authentication/guides/access-tokens.html

[3] Autenticación de Usuarios - https://developer.twitter.com/en/docs/accounts-and-users/subscribe-account-activity/guides/authenticating-users.html

Estos son los identificadores y nombres de los diferentes perfiles de Twitter que se almacenaron:

Identificador	Nickname	Código	Nombre
212376103	MINIRD	RD	MINI RD
2231639696	MINI_Paraguay	PY	MINI Paraguay
333769939	MINIPanama	PA	MINI Panama
366139966	MINImexico	MX	MINI México
39165786	MINIcolombia	COL	MINI Colombia
720747390	ChileMINI	CL	MINI Chile

Twitter cuenta con dos identificadores para cada cuenta que tenga registrada. Una es el **nickname** el cual elige el usuario, y otro un identificador numérico asignado por Twitter.

Para acceder a la página de Twitter de cada país se toma la ruta *https://twitter.com/* y se le concatena el nickname de la página del país a la cual se quiera acceder, por ejemplo, https://twitter.com/MINImexico muestra la página de fans de MINI México en Twitter.

instagram

La tabla de *instagram* almacena la información importante del perfil de Instagram de cada país así como el token de acceso necesario para autenticar las peticiones al API.

Estos son los identificadores y nombres de los diferentes perfiles de Instagram que se almacenaron:

Identificador	username	Código	Nombre
1457996206	minicolombia	COL	MINI Colombia
1107093846	miniparaguay	PY	MINI Paraguay
2041205806	mini.peru	PE	MINI Perú
2218187788	miniguatemala	GT	MINI Guatemala
2261903394	mini.panama	PA	MINI Panamá
231274656	mini.chile	CL	MINI.Chile
235934740	mini_rd	RD	MINI RD
24879301	minimexico	MX	MINI México

Instagram cuenta con dos identificadores para cada cuenta que tenga registrada. Una es el *username*, que es elegido por el usuario, y otro un identificador numérico, asignado por Instagram.

Acceder a la página de Instagram de cada país se toma la ruta https://www.instagram.com/ y se le concatena el username de la página del país a la cual se quiera acceder, por ejemplo, https://www.instagram.com/minimexico muestra la página de fans de MINI México en Instagram.

facebook_posts

Esta tabla almacena los datos de las publicaciones hechas desde el perfil de Facebook del país, cada tupla de esta tabla está relacionada a la tabla **facebook**. De cada publicación se almacena el mensaje, un atributo booleano que indica si es una publicación con imagen o no, la URL de la imagen si es que la hay, el número de likes, comentarios, número de veces que se compartió, fecha de publicación y un atributo booleano que indica si la publicación necesita ser actualizada. A continuación se muestran algunos datos que se almacenaron:

Ejemplo 1:

fb_post_id : *119466091417535_1050178891679579*

fb_page_id : *119466091417535*

message : *Enciende la diversión. #MINI*

is_picture : *1*

picture : https://scontent.xx.fbcdn.net/hphotos-xal1/v/t1.0-9/p720x720/12391076_1050178891679579_929713943975244614_n.jpg?_nc_eui=ARgxCheomStqS6BM7IuguHgv_Kac6vQmn85eLGMfmIqV2VFwUNeKNBzIpYA-&oh=e3682b43e41045b5c896d2c345bfc677&oe=57744BAB

likes_count : *13*

comments_count : *0*

shares_count : *1*

requires_update : *1*

Ejemplo 2:

fb_post_id : *119466091417535_1051101251587343*

fb_page_id : *119466091417535*

message : *Nada define "clásico" como lo hace un #MINI.*

is_picture : *1*

picture : https://scontent.xx.fbcdn.net/hphotos-xfa1/v/t1.0-9/p720x720/10154069_1051101251587343_184920363167753526_n.png?_nc_eui=ARjIm0bFv8H1vj2i7Nv4X5fe8mvmcfC55g-cTeW-oHxS0cqnr67-ZnN7fBHJ&oh=6289b125e3a9c56781e726a3109880a4&oe=577C73E2

likes_count : *30*

comments_count : *0*

shares_count : *1*

requires_update : *1*

Ejemplo 3:

fb_post_id : *119466091417535_1051101384920663*

fb_page_id : *119466091417535*

message : *¡Vamos por la victoria! Los #MINI Countryman ALL4 Racing tienen cuatro victorias consecutivas del 2012-2015 en el Rally Dakar. Apóyalos del 2 al 16 de enero. #Dakar2016*

is_picture : *1*

picture : https://scontent.xx.fbcdn.net/hphotos-xft1/v/l/t1.0-9/p720x720/1931398_1051101384920663_5807919710547577541_n.jpg?_nc_eui=ARgKxtQW_

fVLKbp0q8NKLY6X8CVxOXjuO20zFcbwhbUB3FoP7aBZqQE4S6M2&oh=
8c01b6ce2d06d9a5e356b226017b148a&oe=5778576C

likes_count : *19*

comments_count : *1*

shares_count : *3*

requires_update : *1*

Ejemplo 4:

fb_post_id : *119466091417535_1051101484920653*

fb_page_id : *119466091417535*

message : *Ser puntual es tener actitud #MINI.*

is_picture : *1*

picture : https://scontent.xx.fbcdn.net/hphotos-xpf1/v/t1.0-9/p720x720/
12400504_1051101484920653_3403380515247461012_n.jpg?_nc_eui=
ARivTYOEbt7RIAQOblgqL4tZGwAoWR7TGP-KOwjPZYFr0RTcu0ligCnbP-BP&
oh=14dc49cbb9847b7f219b944058499eb0&oe=577E96B7

likes_count : *19*

comments_count : *0*

shares_count : *1*

requires_update : *1*

facebook_comments_info

Esta tabla sirve para almacenar los datos de los comentarios hechos por los usuarios en el perfil de Facebook del país; esta tabla es similar a **facebook_posts** con la diferencia de que esta almacena el identificador del usuario que hizo el comentario.

facebook_page_likes

Esta tabla sirve para almacenar el conteo de seguidores del perfil de Facebook de un país alcanzado durante un mes, es decir, durante un mes se hacen las actualizaciones sobre la misma tupla perteneciente a un perfil, para hacer esto se busca la tupla del perfil cuya fecha de creación se encuentre entre el primer día del mes y el último, al encontrarla se actualiza con el nuevo conteo de seguidores, si no se encuentra se crea una nueva tupla para almacenar conteo de seguidores del mes en curso.

Esta tabla es de utilidad para poder crear la comparativa del número de seguidores entre el mes actual y el mes anterior que se muestra en la *Figura 1.13*.

facebook_likes

Esta tabla almacena los likes hechos por un usuario a publicaciones o comentarios de la página de Facebook de un país. Se usa para tener registro de la interacción de un usuario con las publicaciones, para después poder calcular los usuarios que más interactúan con el perfil de un país.

Las tablas que almacenan las publicaciones de Facebook, Twitter e Instagram almacenan datos similares y se pensó en elaborar una única tabla que pudiera almacenar la información de las publicaciones pero se tomaron en cuenta las ventajas/desventajas y al final se llegó a la decisión de no hacerla. A continuación se mencionan algunas de las ideas que ayudaron a tomar esa decisión.

La longitud de los atributos que cumplen la misma función para las diferentes redes no era la misma, por ejemplo, la longitud de los mensajes en Twitter (al momento de la creación de la aplicación web) era de 140 caracteres, mientras que en Facebook era mayor a 3,000.

Si se tuviera una tabla llamada **posts** que pudiera almacenar todas las publicaciones independientemente de la red social, tendría los siguientes atributos:

post_id : Identificador de la publicación. Los identificadores de publicación de cada red social son distintos y no se puede calcular un tamaño mínimo, por ejemplo, el identificador de una publicación de Facebook es largo porque se compone del identificador del usuario que hizo la publicación y otro identificador asignado por Facebook, por otro lado, el identificador de las publicaciones de Twitter es corto.

social_type Caracter que identifica a la red social a la que pertenece. Podría declararse como: `ENUM('F','T',I)`.

social_id Identificador del perfil de la red social del país al que pertenece la publicacion.

has_picture Valor booleano que indica si la publicación tiene o no imagen.

message Mensaje de la publicación. Mientras que para mensajes de Twitter la declaración de tipo de esta columna podría ser `VARCHAR(140)`, en los mensajes de Facebook la declaración de tipo es ser `MEDIUM TEXT`, entonces se establecería la longitud de este campo a `MEDIUM TEXT`.

likes Conteo de likes o favoritos que tiene la publicación.

comments Cantidad de comentarios que tiene la publicación.

shares Conteo de veces que la publicación se compartió. El valor de esta columna siempre sería 0 para publicaciones de Instagram ya que el conteo de veces compartido no puede ser obtenido de su API.

created_time Fecha de publicación.

Las desventajas del diseño de esta posible tabla son:

- La llave primaria de esta tabla estaría compuesta por **post_id**, **social_type** y **social_id**. Se considero como una desventaja porque la tecnología que se usó para el manejo de la base de datos no trabaja bien con llaves ternarias.
- No existen llaves foraneas porque una misma columna no puede ser llave foranea de diferentes tablas. **social_id** contendría los identificadores de los perfiles de las redes sociales, estas son tres y

no se podría hacer llave foranea a esta columna para relacionarla con las tablas **facebook**, **twitter** e **instagram**.

- Podría reemplazarse la columna **social_id** por las columnas **facebook_id**, **twitter_id** e **instagram_id** como llaves foraneas pertenecientes a tablas **facebook**, **twitter** e **instagram** respectivamente, pero ni **facebook_id**, **twitter_id** ni **instagram_id** podrían ser únicos ya que al insertar una publicación dos de estos serán `null`.

Por los motivos anteriores y para poder hacer mejor uso de las tecnologías elegidas, se crearon diferentes tablas por red social para almacenar de cada una las publicaciones, comentarios, usuarios e interacciones.

CAPÍTULO 3

Recopilación de datos

En este capítulo se detalla cómo se usaron las APIs de las redes sociales, cómo se crearon las peticiones realizadas, cómo se manejó la información devuelta, cómo se almacenaron los datos útiles, que limitantes hubo con cada red social y como se resolvieron estas incidencias.

3.1 Recopilación de credenciales

Para recopilar el contenido publicado de las redes sociales por país era necesario tener los token de acceso de los perfiles de Instagram, Twitter y Facebook.

Para obtener los token de acceso se necesitan todas las credenciales (correo electrónico / contraseña) que se tengan de cada red social ya que se debe de iniciar sesión en las páginas de desarrolladores de las redes. Estas son las páginas para desarrolladores de Facebook, Twitter e Instagram respectivamente:

- https://developers.facebook.com/
- https://apps.twitter.com/
- https://www.instagram.com/developer/

Para obtener las llaves de Twitter e Instagram, se necesita iniciar sesión en sus páginas para desarrolladores con las credenciales del propietario.

Para obtener las llaves de Facebook es necesario ser el propietario de la cuenta, tener acceso a la cuenta del propietario o que el propietario agregue al desarrollador como un nuevo administrador del perfil, para esto es requisito que el desarrollador tenga una cuenta de Facebook.

Una vez que se ingresa a la página de desarrolladores de la red social y se tiene una sesión activa, se debe crear una aplicación. Una aplicación en las redes es una forma de identificar un proyecto. Para crearla se necesita (en la mayoría de las redes): un nombre para la aplicación, descripción, URL desde la que se puede acceder al sitio web (mientras se realizan pruebas la URL puede ser localhost o la dirección IP del ambiente

de pruebas), correo electrónico de contacto y URL de la política de privacidad. Al crear la aplicación la red genera un identificador para la aplicación y las llaves de acceso necesarias.

Las llaves de acceso son cadenas de diferentes longitudes, cada red social genera su token y con ésta se autentican las peticiones a las APIs para obtener los datos de los perfiles y de las publicaciones.

Cada red social tiene su forma de llamar a las llaves de acceso, las llaves en Instagram se llaman:

- Client ID.
- Client Secret.

En Twitter son:

- Consumer Key.
- Consumer Secret.
- Access Token .
- Access Token Secret.

Y en Facebook son:

- Application ID.
- Application Secret Key.

Estas llaves deben permanecer seguras en el servidor y no deben ser compartidas, de lo contrario comprometería la seguridad de los perfiles de las redes.

Con Instagram es necesario un paso extra para obtener el token de acceso definitivo para autorizar las peticiones al API. El Client ID y Client Secret se usan para dar permiso al servidor de mostrar a través de un navegador una pantalla inicio de sesión, en esta pantalla se debe iniciar sesión con las credenciales de cada red social, si el inicio de sesión es satisfactorio la respuesta de Instagram será el token de acceso final con el que sí se pueden hacer consultas a su API.

Algo similar pasa con Facebook a diferencia de que no se necesita iniciar sesión de nuevo si no que con el Application ID y Application Secret Key se realiza una otra petición al API para generar un token de acceso que no caduca.

Tan sólo con los permisos básicos de Facebook, se puede obtener Nombre, Apellidos, si la persona es mayor de edad, correo electrónico e imagen de perfil. En otros proyectos desarrollados se utilizaron diferentes permisos que requerían la autorización del usuario que utilizara la aplicación y necesitaba que Facebook revisara los permisos que solicitaba la aplicación. Estos son algunos de los permisos que se solicitaron para otras aplicaciones y que requirieron revisión de Facebook:

- Obtener edad.
- Obtener *me gusta* del usuario.
- Obtener páginas que sigue el usuario.
- Obtener ubicación.
- Obtener fotografías.
- Obtener videos.

- Obtener lista de amigos : Proporciona acceso a la lista de amigos de una persona que también usan la aplicación.
- Obtener las publicaciones del usuario : Proporciona acceso a las publicaciones de la biografía de una persona, incluidas las que el hizo y las de otras personas.
- Publicar en nombre de un usuario (publicar texto, imágenes y videos).

Para obtener estos permisos se enviaron diferentes solicitudes de revisión a Facebook, cada solicitud requirió capturas de pantalla en diferentes tamaños y resoluciones de la aplicación, explicación en inglés sobre el uso que se le daría, enlace a los términos y condiciones, enlace al aviso de privacidad, un video que muestra cómo funciona la aplicación y otros datos más. Cada revisión de los permisos tardó aproximadamente 3 días hábiles. Hoy en día, con el cambio de sus políticas, puede tardar incluso semanas.

De Twitter se pueden obtener los permisos antes mencionados sin revisión alguna. Lo único que se necesita es la autorización de cualquier usuario de Twitter para obtener toda la información pública que se quiera. Cuando un usuario inicia sesión en Twitter en una aplicación, se le muestra una ventana, esta ventana tiene los permisos que usará la aplicación, es muy común que los usuarios no lean con detenimiento el contenido de la ventana de permisos y otorguen más permisos de los que la aplicación en verdad necesita.

Instagram es más estricto con sus permisos, para obtener información pública de diferentes cuentas es necesario tener el permiso explícito de cada cuenta que se quiere consultar, no existe un permiso general con el que se pueda obtener la información de todas las cuentas públicas ni publicaciones.

El uso de las herramientas que dan las redes puede ser usado de manera inadecuada por lo que es necesario que en cada plataforma se lleve a cabo un estricto proceso de verificación para prevenir que se creen aplicaciones para fines maliciosos o que no tengan nada que ver con el objetivo para el que fueron solicitados.

Desde 2018 las organizaciones y empresas están obligadas a tener en cuenta la privacidad desde el inicio a la hora de crear y desarrollar nuevos productos y servicios para así cumplir con Reglamento General de Protección de Datos (RGPD)[10], de aplicación internacional.

Este reglamento introduce la privacidad por defecto (Privacy by Design), que en la práctica se traduce como la obligación de las empresas y organizaciones de que sólo recolecten y procesen los datos personales que sean necesarios para la actividad que se esté desarrollando, que sólo estén disponibles para los involucrados y minimicen el tiempo por el que están almacenados y disponibles.

Por lo tanto, si una empresa para brindar su servicio sólo necesita nombre, apellido y correo electrónico, no debe solicitar, por ejemplo, RFC ni fecha de nacimiento, además, estos datos personales recolectados no deben ser almacenados por más tiempo del necesario para prestar el servicio para el que fueron recolectados.

La inclusión de estos conceptos de privacidad obliga a las empresas a actualizar sus procesos internos para adaptarlos a estos requerimientos.

Facebook cuenta con una página web sobre sus medidas para acatar la RGPD[11] de donde se puede resumir lo siguiente:

- Cada empresa o aplicación es responsable de cumplir con el RGPD. Las aplicaciones deben ofrecer un enlace claro y visible al que sea fácil acceder desde la aplicación donde se explique claramente que es posible que terceros, incluido Facebook, recopilen información de la aplicación y de otras aplicaciones, y cómo y dónde podrán desactivar los usuarios, la recopilación y uso de información

[10] Reglamento General de Protección de Datos
[11] RGPD en Facebook e Instagram

(Con esto Facebook se deslinda del mal uso que llegaran a hacer las empresas con las herramientas que ofrece).

- Cada usuario de cualquiera de sus productos (Facebook, Messenger, Instagram, Oculus y WhatsApp) debe informarse de el uso que hace de sus datos y las opciones que tiene.

Facebook aclara el uso que se le da a los datos que recopila en su página web de Política de Datos, a grandes rasgos menciona que recopilan todo el contenido, las comunicaciones, datos proporcionados al usar sus productos, información de los dispositivos, información de cómo se usan sus productos, información sobre las personas, las páginas, las cuentas, los hashtags y los grupos a los que el usuario está conectado y más.

Cuando los usuarios aceptan la Política de Datos requerida para crear una cuenta, en resumen aceptan:

- Que se recopilen y se usen todos los datos antes mencionados.
- Que son responsables sobre que se muestra en su perfil y qué ocultan para el resto de usuarios de la plataforma. Que son responsables de su privacidad ya que la plataforma provee todas las herramientas para proteger la información.
- Que pueden ocultar publicaciones de otros usuarios, pero que esto no significa que la información que se publique no sea usada.

Acerca del tiempo que los datos son resguardados se menciona que "los datos se almacenan hasta que ya no son necesarios para brindar sus servicios o hasta que se elimine la cuenta".

En resumen Facebook e Instagram pueden recopilar y usar de la manera que les sea conveniente, la información personal del usuario y la que éste genere dentro de su plataforma, mientras su cuenta no sea eliminada.

Twitter[12] cuenta con su propia página web acerca del RGPD en donde mencionan la recopilación de datos, brindan al usuario una guía para configurar la privacidad y de donde se puede resumir básicamente lo mismo que con Facebook, **la privacidad es responsabilidad del usuario**.

3.2 Vigencia de las llaves de acceso

Para la aplicación web era necesario que los token de acceso tuvieran tiempo de vida ilimitado para no requerir de trabajo manual para crear nuevas llaves en cada red y después actualizarlas en la aplicación, ya que esto provocaría que cada cierto tiempo la aplicación interrumpiera su funcionamiento hasta obtener de nuevo los permisos necesarios para consultar el API. Para prevenir esta situación se investigó en cada red social sobre la vigencia y términos de uso de sus token y sólo con Facebook se tuvo que realizar un paso extra para obtener un token sin caducidad.

Los token de acceso de Twitter no tienen caducidad, en su lugar que tienen un límite de peticiones por rango de tiempo. El token de acceso se invalida si un usuario rechaza explícitamente el uso de éste desde el panel de desarrollo de Twitter o si un administrador de Twitter suspende la aplicación.

El token que se obtiene por primera vez de Facebook tiene una fecha de caducidad. Para obtener un token sin caducidad se debe ingresar al Explorador de la API Graph, ya dentro, se ingresa el primer token obtenido en el campo *Token de Acceso* y se consulta el endpoint `/me/accounts`, esta consulta responderá una lista con las páginas que se administra, cada una contiene el atributo `access_token` el valor de este atributo es el token sin caducidad, este token es el que se almacenó en la base de datos.

[12] RGPD en Twitter

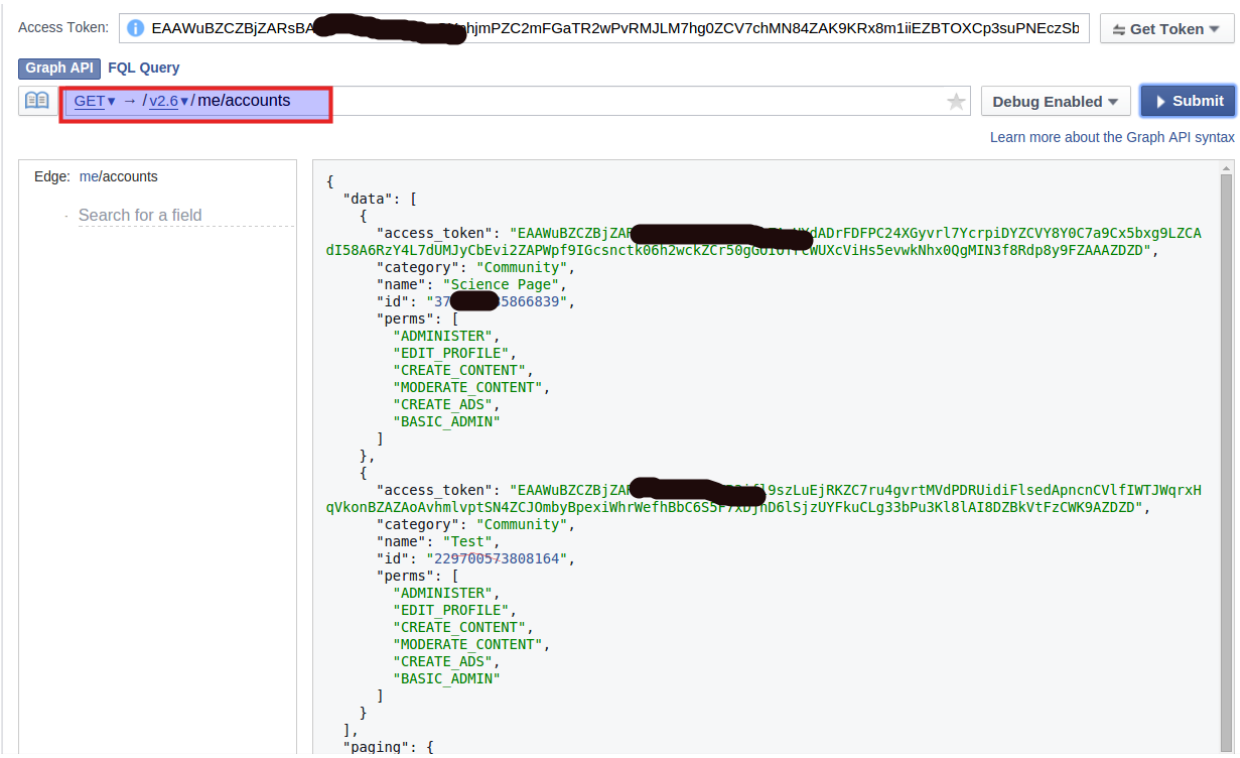


Figura 3.1: Captura de pantalla de la Graph API de Facebook al obtener el token sin caducidad.

Instagram no especifica el tiempo de caducidad de sus llaves de acceso. El token caducará o queda inservible si el un administrador revoca los permisos de la aplicación o si Instagram lo expira después de un periodo de tiempo.

3.3 Recopilación de datos con Facebook

Para obtener los datos y llenar las tablas de Facebook, se deben hacer peticiones a su API, las consultas se deben realizar a los nodos correctos, con los token de acceso correctos y solicitando los campos que se necesite obtener.

URL base del API de Facebook:

```
https://graph.facebook.com/v2.12/
```

A continuación se muestra el formato general de una consulta al API de Facebook:

```
/<nodo>?fields=<primer-nivel>{<segundo nivel>}
```

- **nodo** : Indica la entidad de quien se quiere consultar información. Este puede ser el identificador de cualquier elemento, un usuario, una foto, una página, un comentario, etc.
- **primer-nivel** : Son uno o varios parámetros, separados por coma, que se pueden consultar del nodo.
- **segundo-nivel** : Son uno o varios parámetros que se pueden consultar del *primer-nivel*.

Dentro del primer nivel y en el segundo se puede restringir la cantidad de datos que se quiere obtener con el argumento:

```
.limit(n)
```

este puede concatenar a cualquier parámetro, por ejemplo:

```
// Obtener los n primeros comentarios

comments.limit(n)
```

Las siguientes consultas se realizan al nodo **/me**, para poder obtener los datos de éste nodo se necesita el token de acceso obtenido. Con éste token, Facebook identifica el cliente que hace las petición y si se hacen las consultas correctas devolverá la información solicitada. De aquí en adelante nos referiremos al nodo como **perfil**.

Todas las consultas se hacen sobre el parámetro **fields**, a este parámetro se le indica qué campos se necesitan obtener separados por coma.

Algunos de los campos de primer-nivel y segundo-nivel que se pueden consultar en **fields** son:

Atributo	Descripción
id	Identificador de la página o el campo en el que se está consultando.
post	Información de las publicaciones que ha realizado la página.
name	Nombre de la página.
picture	Información de la imagen de perfil de la página.
country page_likes	Cantidad de likes de la página.
tagged	Información de las publicaciones hechas en el perfil de la página.
comments	Información de los comentarios hechos en la publicación (Sólo aplica para el nodo de publicaciones).
message	Mensaje del campo o nodo (elemento consultado).
full_picture	URL de la imagen asociada al elemento en el tamaño que fue creada.
crea-ted_time	Fecha de creación del campo o nodo.
shares	Número de veces que se compartió el elemento.
comments	Comentarios del elemento. Se puede obtener el conteo de los comentarios además de la información relevante de los comentarios con el atributo .summary(true).
likes	Número de likes del elemento. Se puede obtener el conteo de los likes con el atributo .summary(true).
picture	Información de la imagen asociada al campo o nodo. Se puede restringir el tamaño en el que se devuelve la imagen con los argumentos width(n) y height(m), donde n y m son números enteros que representan la dimensión de la imagen en pixeles.

3.3.1 Obtener las 100 publicaciones más recientes

La siguiente consulta devuelve un arreglo de publicaciones, cada una contendrá los atributos *id*, *from*, *full_picture*, *created_time*, *shares*, *comments* y *likes*.

Consulta

```
// Del nodo actual(me) busca las 100 últimas publicaciones post.limit(100)
// de cada publicación que se encuentre solo se necesitan los atributos
// id, from, message, full_picture, created_time, shares, comments y likes
// De los atributos comments y likes se necesita sólo el resumen, no la lista␣
↪de todos los comentarios
y likes (comments.summary(true) , likes.summary(true))

GET /me?fields=posts.limit(100)
  {
    id,
    from,
    message,
    full_picture,
    created_time,
    shares,
    comments.summary(true),
    likes.summary(true)
  }
```

Respuesta


```
{
  "posts": {
    "data": [
      {
        "id": "117357088671963_264604007280603",
        "from": {
          "name": "MINI",
          "id": "117357088671963"
        },
        "full_picture": "https://scontent.xx.fbcdn.net/v/t15.0-10/s720x720/
        18494059_10155142702461745_1013312939815337984_n.jpg?_nc_cat=0&
        oh=008fd04a4919957e52be79378bde1229&oe=5B60A8E7",
        "created_time": "2017-05-16T03:55:36+0000",
        "likes": {
          "data": [
            {
              "id": "1032952920072873",
              "name": "Alfonso"
            },
            {
              "id": "1390857621206849",
              "name": "Clau"
            },
            {
              "id": "10203652104446098",
              "name": "Omar"
            }
          ],
```

```
          "paging": {
            "cursors": {
              "before": "MTAzMjk1MjkyMDA3Mjg3MwZDZD",
              "after": "MTAyMDM2NTIxMDQ0NDYwOTgZD"
            }
          },
          "summary": {
            "total_count": 3,
            "can_like": true,
            "has_liked": false
          }
        },
        "comments": {
          "data": [
            {
              "created_time": "2017-05-16T05:17:32+0000",
              "from": {
                "name": "Claudia",
                "id": "1390857621206849"
              },
              "message": "Hola",
              "id": "264604007280603_264620500612287"
            }
          ],
          "paging": {
            "cursors": {
              "before":
              "WTI5dGJXVnVkRjlqZAFhKemIzSTZANalkwTmpJd05UQXdOakV5TWpn",
              "after":
              "WTI5dGJXVnVkRjlqZAFhKemIzSTZANalkwTmpJd05UQXdOakV5TWpn"
            }
          },
          "summary": {
            "order": "chronological",
            "total_count": 1,
            "can_comment": true
          }
        }
      }
    ]
  },
  "id": "117357088671963"
}
```


Como se puede ver la consulta devuelve el atributo `comments`, pero los campos que sí se usarán de este son:

- **id** : Identificador del comentario.
- **from.id** : Identificador del usuario de Facebook que comentó la publicación.
- **message** : Comentario del usuario a la publicación.
- **created_time** : Fecha en la que se hizo el comentario.

Los atributos *likes* y *comments* contienen un arreglo de objetos con información de los usuarios que *"dieron me gusta"* o comentaron una publicación. Esta información es necesaria para poder tener registro de actividad de los usuarios y publicaciones para que se puedan generar los reportes de la sección "usuarios más activos" y "publicaciones y comentarios".

Los publicaciones resultantes se almacenarán en la tabla **facebokook_posts** de la siguiente manera:

- **facebook_post_id** : Identificador de la publicación.
- **facebook_page_id** : Identificador de la página que se está consultando.
- **message** : Mensaje publicado.
- **likes_count** : Conteo de likes, se obtuvo de la consulta a el campo `likes.summary.total.count`.
- **comments_count** : Conteo de comentarios, para tener el conteo de comentarios se cuentan las tuplas de facebook_comments.
- **shares_count** : Conteo de las veces compartido, se obtuvo del campo shares.
- **is_picture** : Valor que indica si el mensaje publicado es multimedia o no. Una publicación es multimedia si el atributo **full_picture** es devuelto en la respuesta.
- **picture** : URL de la imagen de la publicación.
- **created_time** : Fecha de creación de la publicación.
- **needs_update** : Valor booleano que indica si la publicación debe ser actualizada o no.

Si la publicación ya existe en la tabla se actualiza su información.

3.3.2 Actualizar o registrar un usuario

Es necesario guardar la información de todos los usuarios que se vayan encontrando para poder mostrar su nombre y foto de perfil en la sección principal en el apartado de *"Seguidores más activos"* y también en *"Publicaciones y comentarios"*.

Para obtener la información de un usuario para que pueda insertarse o actualizarse en la base de datos, se necesita el identificador del perfil de usuario, este ID se puede obtener de los comentarios o de los likes hechos por usuarios en las publicaciones de MINI.

Una vez que obtenido el identificador (`id-usuario`) se realiza la siguiente consulta:

Consulta

```
// De quien corresponda el identificador (id-usuario) devuelve
// id, nombre y la URL de la foto de perfil de ser posible en tamaño 500 x␣
↪500.

GET  /<id-usuario>?fields=id,name,picture.width(500).height(500){url}
```

Respuesta

```
{
  "id": "117357088671963",
  "name": "Lalo Miranda",
  "picture": {
    "data": {
      "url": "https://scontent.xx.fbcdn.net/v/t1.0-1/p720x720/a.jpg"
    }
  }
}
```

Si el usuario ya había sido registrado en la base, se actualiza, ya que puede suceder que el usuario cambie de nombre o de imagen de perfil. Y gracias a que Facebook no cambia los identificadores de usuario, siempre pueden obtenerse los datos de un usuario con su identificador.

3.3.3 Obtener información de los comentarios

El proceso de almacenamiento y actualización para la tabla *facebook_comments_info* es análoga a *Obtener las 100 publicaciones más recientes*. En la actualización de las publicaciones sólo cambia el nodo que se consulta, en vez de **me** se consulta el nodo **id-comentario**, este es el identificador de un comentario, los comentarios se obtienen de la consulta *Obtener las 100 publicaciones más recientes*.

Consulta

```
// Las publicaciones más recientes consulta el nodo me
// Los comentarios se consultan al nodo id-comentario

GET /<id-comentario>?fields=tagged.limit(100){
  id,
  from,
  message,
  full_picture,
  created_time,
  comments.summary(true),
  likes.summary(true)
}
```

Los datos devueltos por esta consulta se almacenan en la tabla **facebook_comments_info**, en ésta se almacenan los comentarios hechos por los usuarios a una publicación o en el muro de Facebook de la marca. Los datos obtenidos se almacenan de la siguiente manera:

- **facebook_comment_id** : Este es el identificador del comentario o mensaje.
- **facebook_page_id** : Identificador de la página de la cual se está consultando la información.
- **facebook_user_id** : Este es el identificador del usuario que realizó el comentario.
- **message** : Comentario o mensaje publicado por el usuario.
- **likes_count** : Número de likes para el mensaje.

- **comments_count** : Número de comentarios para el mensaje.[13]
- **is_picture** : Valor que indica si el mensaje publicado es multimedia o no. Una publicación es multimedia si el atributo **full_picture** es devuelto en la respuesta.
- **picture** : URL de la imagen del comentario.
- **created_time** : Fecha de creación del mensaje o comentario.

Después de cualquier inserción o actualización en las publicaciones o comentarios se debe actualizar la columna **needs_update** a `true` de la tabla **faceboook** que pertenezca a la marca que se consulta, la actualización de esta columna indica a la interfaz que se tiene nueva información que mostrar.

Nota: Cuando `needs_update` de cualquier tupla de la tabla **faceboook** cambia a `true` empieza el proceso de actualización de las páginas de Facebook que se tienen registradas. Primero se actualizan los de datos del perfil de Facebook de MINI, como el nombre y la foto de perfil, después se actualizan las publicaciones, después los comentarios y al final los datos del perfil de los usuarios.

3.3.4 Obtener likes de una página

La siguiente consulta devuelve el número de seguidores del perfil que sea identificado por el token de acceso con el que se hace la petición.

Consulta

```
GET /me?fields=country_page_likes
```

Nota: Esta consulta sólo puede realizarse si el token de acceso pertenece a una página de fans de Facebook.

Se creó la tabla **facebook_page_likes** para guardar los seguidores de cada página por mes. Para actualizar el número de seguidores de una página de Facebook, se busca en la página de **facebook_page_likes** el primer registro que coincida con el identificador de la página que se está actualizando y cuya fecha de creación se encuentre entre la fecha de primer día del mes y la fecha de actualización. Si existe, se actualiza la cantidad de seguidores a la nueva cantidad devuelta por la consulta. Si no existe, significa que la actualización se lleva a cabo el primer día del mes, en este caso se agrega otra tupla a la tabla **facebook_page_likes** con los datos obtenidos de la consulta, así se guardará el registro mes con mes de los seguidores de los distintos perfiles.

3.3.5 Actualización de las publicaciones y comentarios

Cada cierto intervalo de tiempo se debe llamar a la siguiente consulta para poder obtener los datos actualizados de los comentarios y las publicaciones que necesiten ser actualizadas (con *needs_update* igual a *true*).

[13] En el tiempo en que se desarrolló este proyecto Facebook no había implementado los likes o reacciones para comentarios ni las réplicas para estos. Por lo que el almacenamiento de datos en **fb_comments_info** proveniente de un comentario siempre tiene 0 en los campos *likes_count* y *comment_count*.

Por cada elemento que necesite actualización se hace la siguiente consulta al API.

```
// node-id : Es el identificador de un comentario o publicación (fb_comment_
↪id o fb_post_id).
// Esta consulta indica que del nodo consultado se devuelva el id, el id de␣
↪los primeros 100 mil likes y
// el id(atributo from) y el mensaje de los primeros 100 mil comentarios.

GET /<node-id>?fields=
  id,
  likes.limit(100000){
    id
  },
  comments.limit(100000){
    from,
    message
  }
```

Cada objeto **comments** o **likes** devuelto contiene el identificador del usuario que dio like o comentó. Con cada identificador obtenido se puede realizar la petición **Obtener información de usuario** para tener la información más reciente de los usuarios (nuevo nombre, nueva foto de perfil).

Se actualizan sólo los comentarios y publicaciones durante el mes en el que fueron creados, es decir, al terminar un mes las publicaciones que fueron creadas en ese mes no serán actualizadas para el próximo, ya que esa información no se verá más en el sitio (excepto el crecimiento de engagement y el crecimiento de seguidores), además esto ayuda a disminuir la cantidad de consultas que se harán en el próximo mes, de lo contrario con el paso de los meses se acumularía el número de peticiones que deben realizarse e incrementa el tiempo de actualización. Esto no afecta en la información que MINI necesita, pues requieren cortes al mes y comparativos entre la información recopilada del mes anterior y el mes actual.

3.3.6 Incidencias con Facebook

El tiempo de respuesta de cada actualización de Facebook depende del tamaño de la consulta y de si la respuesta contiene muchos nodos que necesiten actualización. La consulta para **Obtener las 100 últimas publicaciones** se considera grande por la cantidad de datos que devuelve y la gran cantidad de subconsultas que surgen. En general todas las actualizaciones y creaciones en conjunto son un proceso tardado ya que puede haber muchas publicaciones, muchos comentarios y muchos usuarios que necesite ser actualizada y cada uno de los procesos se debe realizar para obtener la información más reciente.

Para reducir el tiempo de actualización, las actualizaciones se hacen por partes, es decir, que en cada proceso de actualización se toman las diez últimas publicaciones y los comentarios de los usuarios hechas en el mes, si de las diez primeras no hay información por actualizar se continúa con las siguientes diez que necesiten actualización y se encuentren en el mes.

Cuando la actualización se lleva a cabo por solicitud del sitio web, se actualizan las primeras diez publicaciones y se responde, se hizo de esta manera para que la página no permaneciera vacía mientras espera a que se actualicen todos los elementos ya que el tiempo que ocupado para actualizar todos los datos de las publicaciones y comentarios puede ser demasiado.

Un aspecto que cabe mencionar de las consultas que pueden realizarse con Facebook, es que no son una

cadena de consulta convencional que componen las URL. Una cadena de consulta se compone por lo siguiente:

- Una serie de parejas clave-valor.
- Cada pareja está separada por un signo de igual '='.
- Las series de parejas están separadas por el ampersand '&'.

Ejemplo

```
``http://www.ejemplo.com?campo1=valor1&campo2=valor2``.
```

Esta es una convención sugerida[14] por la W3C (World Wide Web Consortium).

En las consultas que se realizaron sólo se pudo ver la clave **field** y distintos valores cómo **comments**, **likes**, **id**, pero a estos valores eran posibles agregarles subconsultas.

No es una consulta convencional, porque todos los datos que requería se realizaban subconsultas sobre los valores, por ejemplo:

```
?fields=tagged.limit(100){ id, from, message, full_picture, created_time,␣
↪comments.summary(true), likes.summary(true) }
```

Esta consulta si es válida ya que sigue las reglas anteriores y la clave o valor pueden contener cualquier carácter, sólo que los caracteres especiales pasan a su codificación para URL[15].

```
?fields=tagged.limit%28100%29%7Bid%2Cfrom%2Cmessage%2Cfull_picture%2Ccreated_
↪time%2C+comments.summary%28true%29%2C+likes.summary%28true%29%7D
```

3.4 Recopilación de datos con Twitter

Con Twitter se lleva a cabo un proceso similar para llenar las tablas y actualizar los datos, los procesos realizados son los siguientes:

- Actualizar el conteo de seguidores.
- Actualizar los datos del perfil.
- Obtener nuevos Tweets.
- Obtener un usuario.
- Asociar un comentario a un usuario.
- Actualizar datos de un usuario.
- Obtener las menciones de los usuarios.
- Actualizar las menciones.
- Obtener los usuarios que que compartieron un Tweet.

[14] URL parameter serialization - https://www.w3.org/TR/2012/WD-url-20120524/#url-parameter-serialization
[15] Codificación de URI - https://www.w3.org/International/O-URL-code.html

- Actualizar los tweets.
- Actualizar los comentarios hechos a las menciones o comentarios.

URL base del API de Twitter es:

```
https://api.twitter.com/1.1/
```

3.4.1 Obtener nuevos Tweets

Con esta consulta se obtienen los últimos 100 tweets del usuario que se identifique con los token de acceso con los que se realiza la petición.

Consulta

```
// exclude_replies : Este parámetro previene que no aparezcan comentarios de␣
↪respuesta en la cronología que se responde.
// count : Especifica la cantidad de Tweets que intentará recuperar, que es␣
↪hasta un máximo de 100 por solicitud.
// include_rts : Cuando se envía como valor del parámetro false, la␣
↪cronología removerá cualquier retweet.

      GET /statuses/user_timeline?exclude_replies=true&count=100&include_
↪rts=false
```

Nota: Sólo se puede realizar esta petición 1500 veces en el transcurso de 15 minutos[16].

Si la respuesta a la petición fue exitosa se devuelven al un a lista con al menos 100 tweets. Cada objeto devuelto que representa un tweet tiene la siguiente estructura:


```
{
  "created_at": "Mon Apr 03 16:09:50 +0000 2017",
  "id": 848930551989915648,
  "id_str": "848930551989915648",
  "text": "@TwitterMktg: Starting today, businesses can request and share",
  "truncated": false,
  "entities": {
    "hashtags": [],
    "symbols": [],
    "user_mentions": [
      {
        "screen_name": "TwitterMktg",
        "name": "Twitter Marketing",
        "id": 357750891,
        "id_str": "357750891",
        "indices": [
```


[16] Límite de peticiones de Twitter a `/timeline` - https://developer.twitter.com/en/docs/tweets/timelines/api-reference/get-statuses-user_timeline.html

```
            3,
            15
          ]
        }
    ],
    "urls": []
  }
}
```

Por cada tweet devuelto se debe verificar si existe una tupla en la tabla **twitter_tweets** con identificador igual al identificador del tweet, si no existe se inserta, en otro caso, se actualiza la información de **needs_update**, **retweets**, **favorites** y **replies** si es que difiere de los valores almacenados.

3.4.2 Obtener usuarios que compartieron un Tweet

Esta consulta obtiene los identificadores de los 100 primeros usuarios que compartieron un tweet.

Consulta

```
// id : Identificador numérico del Tweet del cual se quieren obtener los␣
↪comentarios.
// count : Número de registros a obtener. Debe ser menor o igual a 100.

GET /statuses/retweeters/ids?id=[id]&count=100
```

Nota: Sólo se puede realizar esta petición 300 veces en el transcurso de 15 minutos[#8]_.

Para poder realizar esta petición se deben de obtener los identificadores de los de todas las tuplas de la tabla **tweets**, por cada identificador se realiza poniendo en `[id]` el identificador del Tweet.

Respuesta

```
{
  "previous_cursor": 0,
  "ids": [
    "1382021622",
    "931150754",
    "1364953914",
    "92313481",
    "1398853771",
    "268642828",
    "1393765387",
    "417614795",
    "1401282895",
    "1319566290",
    "1398546265",
```

```
    "1332481988",
    "295679474",
    "967380524",
    "1278983791",
    "711759314"
  ],
  "previous_cursor_str": "0",
  "next_cursor": 0,
  "next_cursor_str": "0"
}
```


El arreglo ids contiene los identificadores numéricos de los usuarios que compartieron la publicación. Teniendo estos identificadores se realiza la siguiente tarea.

3.4.3 Obtener información de los usuarios

Con la siguiente petición se obtiene la información de los usuarios indicados en parámetro `user_id`.

Consulta

Si la respuesta fue exitosa la petición devolverá un arreglo con la información detallada de cada usuario consultado.

Respuesta


```
{
  "user": {
    "id": 11348282,
    "id_str": "11348282",
    "name": "NASA",
    "screen_name": "NASA",
    "location": "",
    "description": "Explore the universe and discover
    our home planet with @NASA. We usually post in EST
    (UTC-5)",
    "url": "https://t.co/TcEE6NS8nD",
    "entities": {
      "url": {
        "urls": [
          {
            "url": "https://t.co/TcEE6NS8nD",
            "expanded_url": "http://www.nasa.gov",
            "display_url": "nasa.gov",
            "indices": [
              0,
              23
            ]
          }
        ]
```

```
      },
      "description": {
        "urls": []
      }
    },
    "protected": false,
    "followers_count": 28677827,
    "friends_count": 269,
    "listed_count": 90706,
    "created_at": "Wed Dec 19 20:20:32 +0000 2007",
    "favourites_count": 2958,
    "utc_offset": -18000,
    "time_zone": "Eastern Time (US & Canada)",
    "geo_enabled": false,
    "verified": true,
    "statuses_count": 50797,
    "lang": "en",
    "contributors_enabled": false,
    "is_translator": false,
    "is_translation_enabled": false,
    "profile_background_color": "000000",
    "profile_background_image_url": "http://pbs.twimg.com
    /profile_background_images/590922434682880000/3byPYvqe.jpg",
    "profile_background_image_url_https": "https://pbs.twimg.com
    /profile_background_images/590922434682880000/3byPYvqe.jpg",
    "profile_background_tile": false,
    "profile_image_url": "http://pbs.twimg.com
    /profile_images/188302352/nasalogo_twitter_normal.jpg",
    "profile_image_url_https": "https://pbs.twimg.com
    /profile_images/188302352/nasalogo_twitter_normal.jpg",
    "profile_banner_url": "https://pbs.twimg.com/profile_banners
    /11348282/1518798395",
    "profile_link_color": "205BA7",
    "profile_sidebar_border_color": "000000",
    "profile_sidebar_fill_color": "F3F2F2",
    "profile_text_color": "000000",
    "profile_use_background_image": true,
    "has_extended_profile": true,
    "default_profile": false,
    "default_profile_image": false,
    "following": null,
    "follow_request_sent": null,
    "notifications": null,
    "translator_type": "regular"
  }
}
```


Por cada usuario que respondió la consulta se busca si existe en la tabla **twitter_users**, si existe, se actualiza, si no existe se agrega.

La búsqueda en la tabla **twitter_users** se realiza buscando por el `id` de cada objeto de la respuesta, ya que al almacenar un usuario este identificador queda como llave primaria.

Con las tareas anteriores en Twitter se logra lo siguiente: actualizar la información de los usuarios de Twitter y tener un registro de los usuarios que han tenido interacción con un Tweet publicado por cualquier perfil.

Con los datos de la tabla **tweets** es posible obtener el conteo parcial de los usuarios que más han interactuado con las publicaciones de una marca, es parcial ya que falta considerar las veces que el usuario marcó como favorito o comentó un Tweet.

3.4.4 Incidencias con Twitter

Todas las consultas devuelven la URL de la foto de perfil del usuario en tamaño 48 x 48 y esto resultó ser un problema cuando eran visualizadas en el sitio web, porque en dispositivos con resoluciones grandes se veía mal. Para obtener la imagen en el tamaño en el que fueron subidas por los usuarios fue removido de cada URL el fragmento `_normal`.

Por ejemplo:

```
https://pbs.twimg.com/profile_images/969300881119195136/0y1-sdx__normal.jpg
```

Es una imagen de `48 x 48` pixeles.

Si se requiere una versión más grande que la imagen anterior se quita la palabra `_normal` de la URL.

```
https://pbs.twimg.com/profile_images/969300881119195136/0y1-sdx_.jpg
```

Ésta es una imagen de `360 x 360` pixeles.

Nota: Tambien se pueden obtener las imagenes en tamaño **73 x 73** o **24 x 24** reemplazando `_normal` por `_bigger` o `_mini` respectivamente[18].

- **bigger** : https://pbs.twimg.com/profile_images/969300881119195136/0y1-sdx__mini.jpg
- **mini** : https://pbs.twimg.com/profile_images/969300881119195136/0y1-sdx__bigger.jpg

---

De esta manera todas las URL de imágenes que se guardaron en la base de datos corresponden a imágenes en tamaño normal (que es el tamaño en el que fueron subidas originalmente). En la documentación de Twitter no está debidamente explicado cómo obtener la imagen en tamaño real.

Twitter tiene la limitación de que sólo se pueden obtener los primeros 100 objetos más recientes por consulta ya sean tweets o menciones. Esta limitación provoca que se hagan consultas, con más frecuencia para obtener los datos requeridos y que se haga uso de su paginación.

Debe estar explícito en cada consulta el que no sean devueltos retweets ni las respuestas a los tweets, de lo contrario la respuesta contendrá publicaciones repetidas, esto se debe a que para Twitter cada publicación compartida es una nueva publicación.

Se debe tener especial cuidado con el número de llamadas que se realizan al API de Twitter, ya que cada método tiene un límite de peticiones que se pueden realizar en cierto intervalo de tiempo, si se supera este límite no se pueden realizar más peticiones si no hasta después de que pasen 15 minutos. Por esta razón, en

[18] Imágenes de Perfil de Usuario en Twitter - https://developer.twitter.com/en/docs/accounts-and-users/user-profile-images-and-banners.html

una sola llamada hacia peticiones concisas para obtener la mayor información posible. Esto no hacía que obtuviera menos información o que se perdiera, ya que los atributos de cada objeto de la respuesta siempre es el mismo por petición, sólo se solicitaban filtradas las respuestas para no manejar datos innecesarios y se solicitaba el número máximo posible de datos que al API de Twitter podía responder.

Una vez que se llegaba al límite de llamadas al API de Twitter con una mismo perfil de Twitter, continuaba con el siguiente perfil de Twitter para actualizar su información. Una vez que pasaban los 15 minutos y se podían volver a realizar llamadas, se continuaba con el proceso de llamadas en el que se encontraba el primer perfil antes de que se alcanzara el límite de peticiones. Para saber dónde continuar se usó la columna **needs_update**, con la ayuda de esta columna se sabe a partir de qué tupla se detuvo la actualización.

3.5 Recopilación de datos con Instagram

Con Instagram se realizan tantas tareas para obtener los datos como si pasa en Facebook o Twitter. Son menos tareas a causa de las limitaciones de su API. Los procesos que sí se realizan son los siguientes:

- Actualizar el conteo de seguidores del perfil.
- Obtener las publicaciones de la marca.
- Actualizar los datos de las publicaciones como es el número de likes y número de comentarios.

URL base del API de Instagram:

```
https://api.instagram.com/v1
```

3.5.1 Actualizar conteo de seguidores

Esta consulta devuelve la información de perfil del propietario del token de acceso con el que se hace la petición.

Consulta

```
GET /users/self/?access_token=[TOKEN-DE-ACCESO]
```

Respuesta

```
{
  "data": {
    "id": "1574083",
    "username": "snoopdogg",
    "full_name": "Snoop Dogg",
    "profile_picture": "http://distillery.samazonaws.com
    /profiles/aaa.jpg",
    "bio": "This is my bio",
    "website": "http://snoopdogg.com",
    "is_business": false,
```

```
    "counts": {
      "media": 1320,
      "follows": 420,
      "followed_by": 3410
    }
  }
}
```

Los atributos que contiene son:

- **id** : Identificador del usuario en Instagram.
- **username**: Nombre de usuario, este nombre es único.
- **full_name**: Nombre del usuario.
- **profile_picture**: Imagen de perfil del usuario.
- **bio**: Descripción de la biografía del usuario.
- **website**: Sitio web del usuario.
- **is_bussines**: Si es true es una cuenta de Instagram para una empresa, en otro caso es una cuenta normal.
- **counts**: Objeto que contiene el conteo de los siguientes datos.
  - **counts.media** : Conteo de publicaciones hechas por el usuario.
  - **counts.follows** : Conteo de usuarios a los que se sigue.
  - **counts.followed_by** : Conteo de seguidores del usuario.

Con esta petición se actualiza el número actual de seguidores del perfil de Instagram. Para actualizar, se busca en la tabla **instagram_followers** la tupla con identificador igual a `id` y cuya fecha de creación esté entre el primer día del mes y la fecha de consulta. Si se encuentra, se actualiza el número de likes con el valor de `counts.followed_by` y con `needs_update` igual a `true`. Si no se encuentra, significa no hay registros, ya sea porque a que se inició la aplicación o que es inicio de mes, en cualquier caso se inserta una nueva tupla con los valores **likes** igual a `counts.followed_by` y `needs_update` igual a `false`. Si se encuentra la tupla significa que ya hay un registro del conteo de seguidores en el mes, en este caso se actualiza el la columna *likes* de la tupla encontrada con el valor `followed_by` y la columna `needs_update` a `true`.

3.5.2 Incidencias con Instagram

En Instagram no es posible ver los datos de los usuarios que dieron like a una publicación o comentaron, porque para ver la información los perfiles públicos o privados de los usuarios, se necesita la autorización explícita del usuario. Esta limitación causó que no pudieran crearse en la sección principal de Instagram los apartados *"Publicaciones y Comentarios"* y *"Usuarios más activos"* como en Facebook y Twitter; sólo es posible ver la información del usuario que da su autorización a Instagram.

El límite de peticiones de Instagram es de 5000 peticiones por hora para cada token y ya que las consultas son mucho menos que con Twitter o Facebook no se tuvo problema con este límite.

CAPÍTULO 4

Generación de reportes con los datos recopilados

En este capítulo se muestran las consultas a la base de datos realizadas para poder crear los reportes que se enviaron al sitio web, para que éste fuera capaz de mostrar la información en la interfaz y así cumplir con la propuesta de diseño.

4.1 Consultas para obtener los datos de Facebook

Los datos de Facebook que deben de enviarse a la aplicación web son:

- Código del país: (*MX*, *COL*, *CR*,...).
    - Nombre de la página.
    - Foto de perfil.
    - Información de la página:
        - Número de likes.
        - Número de Comentarios.
        - Número de veces compartido.
        - Número de publicaciones.
    - Información de los Likes:
        - Likes del mes anterior.
        - Likes del mes actual.
        - Diferencia de Likes.
        - Indicador si subió o bajó el número de likes.

- Información del Engagement:
  - Cálculo engagement del mes anterior.
  - Cálculo engagement del mes actual.
  - Diferencia de engagement.
  - Indicador si subió o bajó el engagement.
- Posición de la página en la clasificación del engagement.
- Última publicación de la página:
  - Identificador de la publicación.
  - Indicador para saber si es una publicación con imagen o no.
  - Imagen de la publicación.
  - Mensaje de la publicación.
  - Número de likes.
  - Número de comentarios.
  - Número de veces compartido.
  - Fecha de creación de la publicación.
- Información de los comentarios hechos por los seguidores:
  - Identificador del comentario.
  - Nombre del usuario que realizó la publicación.
  - Foto de perfil del usuario.
  - Indicador para saber si es una publicación con imagen o no.
  - Imagen de la publicación.
  - Mensaje de la publicación.
  - Número de likes.
  - Número de comentarios.
  - Fecha de creación de la publicación.
- Publicación más relevante:
  - Identificador de la publicación.
  - Indicador para saber si es una publicación con imagen o no.
  - Imagen de la publicación.
  - Mensaje de la publicación.
  - Número de likes.
  - Número de comentarios.

- Número de veces compartido.
- Fecha de creación de la publicación.

- Lista de los usuarios más activos:
  - Identificador del usuario.
  - Suma de likes y comentarios hechos.
  - Nombre del usuario.
  - URL de la foto de perfil.
  - Número de comentarios hechos.
  - Número de likes hechos.

La respuesta que se generará con esta información se verá como la siguiente:


```
{
  "PA" : {
    "name" : "MINI Paraguay",
    "picture" : "https://scontent.xx.fbcdn.net/v/t1.0-1/22089730_174219599247
    1348_629246975461527947_n.png?oh=c9014e197910ed7c3ab87f34eb4d3e38
    &oe=5AA1D8B8",
    "page_data" : {
      "likes" : 604690,
      "comments" : 5220,
      "shares" : 7704,
      "posts" : 881,
    },
    "likes" : {
      "difference" : "up",
      "difference_count" : 10,
      "current" : 100,
      "past" : 90
    },
    "engagement" : {
      "difference" : "down",
      "difference_count" : 0.49,
      "current" : 0.4,
      "past" : 0.89
    },
    "last_post":{
      "is_picture" : true,
      "picture":"https://scontent.xx.fbcdn.net/hphotos-xal1/v/t1.0-9/p720x
      720/12391076_1050178891679579_929713943975244614_n.jpg?_nc_eui=
      ARgxCheomStqS6BM7IuguHgv_Kac6vQmn85eLGMfmIqV2VFwUNeKNBzIpYA-
      &oh=e3682b43e41045b5c896d2c345bfc677&oe=57744BAB",
      "message":"Enciende la diversión. #MINI",
      "likes_count":13,
      "shares_count":1,
      "comments_count":2,
      "created_time":"2015-12-24 01:00:00"
    },
```

```
    "tagged_posts":[
      {
        "user_name" : "Fabian da Silva",
        "user_picture" : "https://scontent.xx.fbcdn.net/hprofile-xtp1/v/t1.
        0-1/p720x720/12011200_961453027223781_1443707060904098435_n.jpg
        ?_nc_eui=ARhZz96jerIn7Ku0Ei4Ay6jjPADV_6MsVoYHKaaU5DP1i6zJcM0c6MM
        M1H0h&oh=2cccd0acfcb82c8c4f7e6f52380a890a&oe=577C43AF",
        "is_picture" : true,
        "picture" : "https://scontent.xx.fbcdn.net/v/t1.0-0/p480x480/106896
        65_759412150761204_7875482140630520741_n.jpg?oh=93987604c710e00b4b3
        68107de1c3a7a&oe=58E1937A",
        "message" : "Definitivamente no me queda ninguna palabra mas que
        Gracias! También quiero agradecer a la marca que creyó en mi
        trabajo, gracias por apostar a la moda nacional MINI Paraguay.",
        "likes_count" : 233,
        "comments_count" : 22,
        "created_time" : "2014-09-09 00:46:35"
      },
      {
        "user_name" : "Fayse Del Orbe Rizek",
        "user_picture" : "https://scontent.xx.fbcdn.net/v/t1.0-1/13178742_
        10156988302340046_2418205444230001924_n.jpg?oh=b53dd4006b5a91ecca
        8933bc9d68f1d3&oe=57D1BA3F",
        "is_picture" : false,
        "picture" : null,
        "message" : "HOY POR MI, MAÑANA POR TI!! POR FAVOR dale aquí
        http://p.dry.lt/01irIrLVbNs5 y después dale a VOTAR! Te lo␣
↪agradecería muchísimo!!",
        "likes_count" : 0,
        "comments_count" : 0,
        "created_time" : "2016-04-29 12:10:28"
      }
    ],
    "relevant_post":[
      {
        "is_picture" : true,
        "picture" : "https://scontent.xx.fbcdn.net/v/t1.0-9/s720x720/15420
        903_1393388200685464_8573085976685501301_n.png?oh=95eecae6dc4c97e6
        55c8d87575d39bea&oe=58F22D15",
        "message" : "La historia continúa. #MINI #Seven #MeetTheSeven ",
        "likes_count" : 28753,
        "comments_count" : 80,
        "shares_count" : 0,
        "total" : 28833,
        "created_time" : "2016-12-11 12:36:06"
      }
    ],
    "users":[
      {
        "name" : "Lupis San",
        "picture" : "https://scontent.xx.fbcdn.net/v/t1.0-1/c0.0.720.720/p72
        0x720/13432186_1022062497830241_5206455384346621236_n.jpg
        ?oh=5416b3780cd52e62c0153d09363b724d&oe=58712DC7",
```

```
        "likes" : 17,
        "comments" : 9,
        "total" : 26
      }
    ],
  },
  "COL" : {...},
  "MX" : {...},
  "PY" : {...},
  "CR" : {...},
  "CL" : {...},
  "GT" : {...},
  "RD" : {...},
  "UY" : {...}
}
```

La respuesta anterior es un documento JSON[1]. Toda la comunicación entre cliente y servidor se hace enviando y recibiendo estos documentos.

4.1.1 JSON

JSON es el acrónimo de JavaScript Object Notation, que es un formato de texto ligero de intercambio de datos basado en un subconjunto del lenguaje JavaScript pero es completamente independiente de este.

JSON está constituido por dos estructuras:

- Una colección de pares de nombre/valor.
- Una lista ordenada de valores.

Elegí usar documentos JSON para las respuestas por las siguientes razones:

- Puede ser leído por cualquier lenguaje de programación.
- Proporciona gran sencillez para usar estructuras como objetos y arreglos.
- Es simple, ligero y se necesita menos tiempo para estructurarlo.
- Actualmente JSON es un formato que domina la web.

A continuación se explican las consultas a la base que devuelven la información para poder crear este JSON y tambíen se explica el significado de cada atributo y como se obtuvo cada valor.

4.2 Obtener la publicación más reciente

De la tabla **facebook_posts** se obtiene la primera publicación que pertenece a la cuenta de la que se quiere obtener información y cuya fecha de creación se encuentre entre la fecha de inicio del mes y la fecha de consulta.

[1] JSON - http://json.org/

```
SELECT * FROM facebook_posts
WHERE fb_page_id =  page_id
AND created_time
BETWEEN(first_date, current_date)
ORDER BY created_time DESC LIMIT 1;
```

4.3 Obtener la publicación más relevante

La publicación más relevante es la que en el transcurso de un mes, tiene el mayor número de *me gusta*, comentarios y número de veces compartida.

De la tabla **facebook_posts** se obtienen los registros entre la fecha de inicio del mes y la fecha actual, se suman las columnas `likes_count`, `comments_count` y `shares_count` se le asigna el nombre **total**, las tuplas obtenidas se ordenan por total de manera descendente y se obtiene el primer registro.

```
SELECT fb_post_id, is_picture, picture, message, likes_count, comments_count,
shares_count, likes_count+comments_count+shares_count AS total, created_time
FROM fb_posts
WHERE fb_page_id =  page_id
AND created_time
BETWEEN  first_date
AND current_date
ORDER BY total
DESC LIMIT 1;
```

4.4 Obtener los usuarios más relevantes

Un usuario relevante es aquel que interactúa más con las publicaciones del perfil de MINI. La relevancia de los usuarios se calcula a partir del número de me gusta y comentarios que ha hecho a cualquier publicación de un perfil de MINI creada entre el primer día del mes y la fecha actual.

Esta consulta se explica por partes ya que es grande:

Primero obtiene el conteo de los *me gusta* dados por los usuarios a las publicaciones entre el primer día del mes y la fecha actual. Se nombra al resultado **likes**.

```
-- Esta consulta se nombra likes

SELECT likes.fb_page_id, fb_user_id, COUNT(fb_user_id) AS likes
FROM fb_likes AS likes , 0 AS comments
INNER JOIN fb_posts AS fb_post
ON likes.fb_likes_id = fb_post.fb_post_id
WHERE likes.fb_page_id = page_id
AND fb_post.created_time
BETWEEN first_date
AND current_date
GROUP BY fb_user_id;
```

Después se obtiene el conteo de los comentarios hechos por los usuarios entre el primer día del mes y la fecha actual. Se nombra al resultado **comments**.

```
-- Esta consulta se nombra comments

SELECT fb_comments.fb_page_id, fb_comments.fb_user_id, fb_comments.comment_
↪count
AS comments, 0 AS likes
FROM facebook_comments_info AS fb_comments
INNER JOIN facebook_posts AS fb_post
ON facebook_comments.facebookcomment_id = fb_post.fb_post_id
WHERE facebook_comments.fb_page_id = page_id
AND fb_post.created_time
BETWEEN first_date
AND current_date;
```

Se hace la unión de las consultas **likes** y **comments** creadas en los pasos anteriores y se agrupan por identificador de usuario para que se puedan sumar los *me gusta* y comentarios de cada usuario correctamente. Se nombra al resultado **sum_total**.

```
-- Esta consulta se nombra sum_total

SELECT fb_page_id,fb_user_id, SUM(likes) AS likes , SUM(comments) AS comments,
SUM(likes)+SUM(comments)
AS total
FROM ( likes )
UNION ( comments ) AS all
GROUP BY fb_user_id;
```

Por último, se necesitan los datos del usuario importantes del usuario como nombre, URL de la foto de perfil, la suma de sus *me gusta* y comentarios, por lo que se realiza un `INNER JOIN` de la consulta **sum_total** con la tabla **facebook_users** y del resultado se obtienen los 6 primeros registros.

```
SELECT fb_page_id, total.fb_user_id, users.name, users.picture,
likes, comments, total
FROM sum_total
INNER JOIN facebookb_users AS users
ON users.fb_user_id = total.fb_user_id
ORDER BY total DESC
LIMIT 6;
```

4.5 Obtener los datos necesarios para el cálculo de la fórmula de engagement

Para obtener el engagement visto en el *Capítulo 1* se necesita el conteo de me gusta, comentarios, número de veces compartido y total de publicaciones realizadas en un mes.

```
// Likes = Me gusta
// Fans = Seguidores
```

```
(((Σ Comentarios + Σ Likes + Σ Shares) / Σ Publicaciones ) / Fans) * 100
```

Advertencia: Si la suma de publicaciones es 0 es imposible obtener la suma de comentarios, likes y shares. Si no hay publicaciones no se realiza el cálculo anterior y se devuelve 0 como valor del engagement.

Se obtiene *Comentarios*, *Likes*, *Shares* y *Likes* con la siguiente consulta:

```
SELECT SUM(likes_count) AS likes, SUM(comments_count) AS comments,
SUM(shares_count) AS shares, SUM(1) AS posts
FROM facebookpostdatas
WHERE fb_post_id = page_id
AND created_time
BETWEEN first_date
AND current_date
LIMIT 1;
```

Para crear la respuesta también se deben obtener el número total de interacciones del mes anterior para poder hacer un comparativo. Usando la consulta anterior y cambiando la fecha de inicio y fecha actual de la sentencia `BETWEEN` se obtiene esa información.

Se calcula el engagement del mes anterior y el mes actual y se envían ambos datos a la aplicación web para poder hacer un comparativo visual.

Para obtener los datos de la vista ranking (*Figura 1.13 del Capítulo 1*) se obtiene el número de seguidores del mes anterior y el mes actual por cada país y se calcula la diferencia entre estos, el resultado se almacena en un diccionario en donde el país es la llave y la diferencia el valor, después se ordenan respecto a la diferencia en orden descendiente. Con el engagement se sigue el mismo proceso.

4.6 Creación de la respuesta

Con los datos obtenidos de las consultas anteriores se crea el JSON que se responde cuando el sitio web hace una petición con método **GET** a la siguiente URI (Identificador uniforme de recursos):

```
/facebook/info
```

Si no hubo error en las consultas y todos los requeridos se obtuvieron, el servidor responde el estado HTTP `200` y el siguiente JSON:

```
{
  "PA" : {
    "name" : "MINI Paraguay",
    "picture" : "https://scontent.xx.fbcdn.net/v/t1.0-1/22089730_174219599247
    1348_629246697546152794 7_n.png?oh=c9014e197910ed7c3ab87f34eb4d3e38
    &oe=5AA1D8B8",
    "page_data" : {
      "likes" : 604690,
```

```
      "comments" : 5220,
      "shares" : 7704,
      "posts" : 881,
    },
    "likes" : {
      "difference" : "up",
      "difference_count" : 10,
      "current" : 100,
      "past" : 90
    },
    "engagement" : {
      "difference" : "down",
      "difference_count" : 0.49,
      "current" : 0.4,
      "past" : 0.89
    },
    "last_post":{
      "is_picture" : true,
      "picture":"https://scontent.xx.fbcdn.net/hphotos-xal1/v/t1.0-9/p720x
      720/12391076_1050178891679579_929713943975244614_n.jpg?_nc_eui=
      ARgxCheomStqS6BM7IuguHgv_Kac6vQmn85eLGMfmIqV2VFwUNeKNBzIpYA-
      &oh=e3682b43e41045b5c896d2c345bfc677&oe=57744BAB",
      "message":"Enciende la diversión. #MINI",
      "likes_count":13,
      "shares_count":1,
      "comments_count":2,
      "created_time":"2015-12-24 01:00:00"
    },
    "tagged_posts":[
      {
        "user_name" : "Fabian da Silva",
        "user_picture" : "https://scontent.xx.fbcdn.net/hprofile-xtp1/v/t1.
        0-1/p720x720/12011200_961453027223781_1443707060904098435_n.jpg
        ?_nc_eui=ARhZz96jerIn7Ku0Ei4Ay6jjPADV_6MsVoYHKaaU5DP1i6zJcM0c6MM
        M1H0h&oh=2cccd0acfcb82c8c4f7e6f52380a890a&oe=577C43AF",
        "is_picture" : true,
        "picture" : "https://scontent.xx.fbcdn.net/v/t1.0-0/p480x480/106896
        65_759412150761204_7875482140630520741_n.jpg?oh=93987604c710e00b4b3
        68107de1c3a7a&oe=58E1937A",
        "message" : "Definitivamente no me queda ninguna palabra mas que
        Gracias! También quiero agradecer a la marca que creyó en mi
        trabajo, gracias por apostar a la moda nacional MINI Paraguay.",
        "likes_count" : 233,
        "comments_count" : 22,
        "created_time" : "2014-09-09 00:46:35"
      },
      {
        "user_name" : "Fayse Del Orbe Rizek",
        "user_picture" : "https://scontent.xx.fbcdn.net/v/t1.0-1/13178742_
        10156988302340046_2418205444230001924_n.jpg?oh=b53dd4006b5a91ecca
        8933bc9d68f1d3&oe=57D1BA3F",
        "is_picture" : false,
        "picture" : null,
```

```
        "message" : "HOY POR MI, MAÑANA POR TI!! POR FAVOR dale aquí
        http://p.dry.lt/01irIrLVbNs5 y después dale a VOTAR! Te lo␣
↪agradecería muchísimo!!",
        "likes_count" : 0,
        "comments_count" : 0,
        "created_time" : "2016-04-29 12:10:28"
      }
    ],
    "relevant_post":[
      {
        "is_picture" : true,
        "picture" : "https://scontent.xx.fbcdn.net/v/t1.0-9/s720x720/15420
        903_1393388200685464_8573085976685501301_n.png?oh=95eecae6dc4c97e6
        55c8d87575d39bea&oe=58F22D15",
        "message" : "La historia continúa. #MINI #Seven #MeetTheSeven ",
        "likes_count" : 28753,
        "comments_count" : 80,
        "shares_count" : 0,
        "total" : 28833,
        "created_time" : "2016-12-11 12:36:06"
      }
    ],
    "users":[
      {
        "name" : "Lupis San",
        "picture" : "https://scontent.xx.fbcdn.net/v/t1.0-1/c0.0.720.720/p72
        0x720/13432186_1022062497830241_5206455384346621236_n.jpg
        ?oh=5416b3780cd52e62c0153d09363b724d&oe=58712DC7",
        "likes" : 17,
        "comments" : 9,
        "total" : 26
      }
    ],
  },
  "COL" : {...},
  "MX" : {...},
  "PY" : {...},
  "CR" : {...},
  "CL" : {...},
  "GT" : {...},
  "RD" : {...},
  "UY" : {...}
}
```


Se explican los atributos que contiene esta respuesta:

- `MX` es el código del país del cual se devuelve la información. En total esta respuesta devolverá 9 objetos con la información obtenida de las consultas para cada perfil de Facebook. Los códigos que se devuelven son: **MX**, **PA**, **PY**, **CR**, **PE**, **CL**, **GT**, **RD**, **UY** y **COL**.

`MX` contiene diferentes atributos que tienen la información de la página de Facebook de MINI México.

- **name** : Nombre público de la página de Facebook.

- **picture** : Imagen de perfil pública de la página de Facebook.
- **page_data** : Objeto con la información de la página. Los atributos son:
  - **likes** : Número de *me gusta* en las publicaciones de la página.
  - **comments** : Comentarios que usuarios han hecho a las publicaciones de la página.
  - **shares** : Número de veces que se compartieron las publicaciones de la página.
  - **posts** : Número de publicaciones.
- **likes** : Objeto con la información de los "me gusta" de la página. Los atributos son:
  - **difference** : Esta cadena indica si los seguidores aumentaron (`up`), disminuyeron (`down`) o permanecieron igual (`equal`) del mes anterior al mes en curso.
  - **difference_count** : Diferencia de seguidores entre el mes anterior y el mes en curso al momento de crear la respuesta.
  - **current** : Número de seguidores al momento de crear la respuesta.
  - **past** : Número de seguidores hasta el último día del mes anterior.
- **engagement** : Objeto con la información del engagement. Los atributos son:
  - **difference** : Esta cadena indica si el engagement aumentó (`up`), disminuyó (`down`) o permaneció igual (`equal`) del mes anterior al mes en curso.
  - **difference_count** : Diferencia de engagement entre el mes anterior y el mes en curso al momento de crear la respuesta.
  - **current** : Engagement al momento de crear la respuesta.
  - **past** : Cálculo de engagement hasta el último día del mes anterior.
- **last_post** : Objeto que contiene la información de la última publicación hecha por el perfil de Facebook. La información de este objeto se obtiene de la consulta **Obtener la publicación más reciente**. Los atributos son:
  - **is_picture** : Este es `true` si la publicación tiene imagen, `false` en otro caso.
  - **picture** : URL de la imagen si es que la publicación contiene, si no el valor es `null`.
  - **message** : Mensaje de la publicación.
  - **likes_count** : Número de *me gusta*.
  - **shares_count** : Número de veces que se compartió la publicación.
  - **comments_count** : Número de comentarios.
  - **created_time** : Fecha de publicación.
- **tagged_posts** : Arreglo de objetos con la información de comentarios hechos por usuarios al perfil de Facebook de MINI. Los atributos son:
  - **user_name** : Nombre público del usuario que comentó en el perfil.
  - **user_picture** : Imagen de perfil del usuario que comentó.

  - **is_picture** : Este es `true` si su comentario tiene imagen, `false` en otro caso.
  - **picture** : URL de la imagen si es que el comentario contiene, si no el valor es `null`.
  - **message** : Mensaje de la publicación.
  - **likes_count** : Número de *me gusta*.
  - **comments_count** : Número de comentarios.
  - **created_time** : Fecha de publicación.
- **relevant_post** : Objeto con la información de la publicación más relevante de la página en el transcurso del mes actual al momento de crear la respuesta. La información de este objeto se obtiene de la consulta **Obtener la publicación más relevante**. Los atributos son:
  - **is_picture** : Este es `true` si la publicación tiene imagen, `false` en otro caso.
  - **picture** : URL de la imagen si es que la publicación contiene, si no el valor es `null`.
  - **message** : Mensaje de la publicación.
  - **likes_count** : Número de *me gusta*.
  - **shares_count** : Número de veces que se compartió la publicación.
  - **comments_count** : Número de comentarios.
  - **created_time** : Fecha de publicación.
- **users** : Arreglo de objetos con la información pública de los usuarios que más interacción tuvieron con la página en el transcurso del mes actual. La información de este objeto se obtiene de la consulta *Obtener las usuarios más relevantes*. Los atributos de cada objeto son:
  - **name** : Nombre del usuario.
  - **picture** : URL de la foto de perfil del usuario.
  - **likes** : Número de *me gusta* hechos.
  - **comments** : Número de comentarios hechos.
  - **total** : Suma de los *me gusta* y comentarios.

Nota: A diferencia del objeto **relevant_post**, ningún objeto de **tagged_posts** contiene el atributo `shares_count` ya que los comentarios de usuarios a una página de Facebook no se pueden compartir.

4.7 Obtención de los datos para crear la nube de términos

La **nube de términos** muestra las palabras más usadas en las publicaciones y comentarios en un perfil de Facebook, Twitter e Instagram, puede encontrarse en las figuras 1.1 a 1.9 como un recuadro negro con palabras en color blanco.

Figura 4.1: Captura de pantalla de la nube de términos.

Para obtener las palabras, se obtienen los mensajes de **facebook_posts** y **facebook_comments_info** cuya fecha de creación está entre la fecha de inicio del mes y la fecha actual. Cada mensaje obtenido se separa por palabras y se descartan las siguientes:

- Artículos.
- Preposiciones.
- Signos de exclamación.
- Saltos de línea.
- Pronombres.
- Conjunciones.
- Disyunciones.
- Palabras extra.

También se descartan los signos de puntuación, exclamación y algunos caracteres especiales (`, . ? ¿ $ * ¡ ! / \ ( ) ' : " & = - +`).

Nota: No se implementó un filtrado de las palabras mal escritas. En algún momento MINI lo solicitó pero por costos y tiempo de desarrollo no se llegó a un acuerdo para desarrollar un algoritmo que identificara estas palabras y las descarta.

Cada palabra que no se descartó se almacena en un diccionario donde los elementos son del tipo (palabra, contador). Cada que se inserta una palabra al diccionario, si la palabra es nueva se inserta con el contador en 1, si la palabra ya existe en el diccionario aumenta en uno el contador.

Nota: El filtrado de las palabras se realizó creando un arreglo con las palabras y caracteres a descartar, después se iteró sobre las palabras del arreglo, si alguna de estas existía como llave en el diccionario, se elimina esa llave de el. La eliminación y búsqueda en el diccionario son de orden constante, y la iteración sobre las palabras es O(1) pues se sabe exactamente el número de palabras y caracteres a descartar, que es 132 y este número no es modificado.

Al tener todas las palabras útiles almacenadas en el diccionario se ordenan respecto al contador de cada palabra en orden descendente. Por último se toman los 50 primeros registros del diccionario y estos son los que se envían a la página web en un objeto JSON para que sean visualizadas.

Finalmente, con los datos resultantes de lo anterior se crea un JSON que se responde cuando la aplicación web hace una petición con método GET a la siguiente URI:

```
/facebook/tag_cloud
```

Si se pudieron obtener los datos de la nube de palabras se responde el estado HTTP `200` y un JSON como el que sigue:

```
{
  "MX" : {
    "mini" : 100,
    "clásico" : 40,
    "diversion" : 2,
    ...
  },
  "COL" : {...},
  "PA" : {...},
  "PY" : {...},
  "CR" : {...},
  "CL" : {...},
  "GT" : {...},
  "RD" : {...},
  "UY" : {...}
}
```

Este JSON contiene las palabras más usadas en las publicaciones del perfil de Facebook del país. La clave es la palabra y el valor es la cantidad de apariciones de esta.

CAPÍTULO 5

Tecnologías de desarrollo y tareas complementarias

5.1 Tareas programadas

Para mantener actualizada la información de las publicaciones, likes, comentarios, número de veces compartido, número de seguidores, nube de términos, comentarios de seguidores y sus likes, se crearon tareas en el servidor que se ejecutaron periódicamente. Estas tareas son llamadas **Cron Jobs**.

En sistemas operativos de tipo Unix, un Cron es un programador de tareas en basado en el tiempo. Estas tareas se ejecutan periódicamente, con base en fechas o intervalos de tiempo.

Los Cron son manejados por medio de un archivo **crontab**, este es un archivo de configuración que especifica los comandos a ejecutar periódicamente a partir de un horario o programación dado.

Cada usuario del sistema puede crear su propio Cron con el siguiente comando desde la terminal:

```
crontab -e
```

Este comando abre un archivo temporal en la terminal donde se escribe la expresión cron. En este archivo cada línea del archivo representa una tarea por lo que se pueden declarar tantas tareas como se necesiten.

Cada entrada del archivo **crontab** tiene cinco atributos para especificar el momento ejecución seguidos del comando a ejecutar.

Los comandos se ejecutan por el cron cuando el minuto, la hora y el mes coincidan con la fecha actual y al menos uno de los dos campos de día coincidan con el día actual.

El *daemon* verifica cada minuto si el crontab contiene una expresión cron que se cumpla.

5.1.1 Campos de fecha y hora

Campos	Valores permitidos
`minuto`	0-59
`hora`	0-23
`día del mes`	1-31
`mes`	1-12
`día de la semana`	0-7

En todos los campos se pueden poner el caracter (∗), lo que significa que toma todos los valores permitidos. Por ejemplo, poner ∗ en la posición de hora significaría *"cada hora"* o *"a toda hora"*.

La expresión cron que se creó para que se ejecutara la tarea en PHP que actualiza la información de la base de datos con los datos de las redes sociales es la siguiente:

```
*/5 * * * * php /var/www/html/mini/artisan schedule:run >> /dev/null 2>&1
```

`*/5 * * * *` significa que la tarea se ejecutará cada 5 minutos, cada hora, todos los meses, cada día de la semana.

La tarea consiste en que PHP ejecute la tarea `schedule:run` del programa `artisan`.

`>> /dev/null 2>&1` significa que se redirige la salida estándar y la salida de error del programa a `/dev/null` el cual es un objeto especial del sistema de archivos que descarta todo lo que está escrito en él. Al redirigir la salida del programa a este archivo no significa que se pierda el registro de error. Todos los mensajes de error se guardan en archivos `.log` organizados por fecha.

`artisan` es la interfaz de línea de comandos (**CLI** por sus siglas en inglés) de **Laravel** (Ver: *PHP y Laravel*).

El comando `schedule:run` ejecuta un programa que indica las tareas que deben realizarse para la actualización. Las tareas son:

- Facebook
  - Actualizar el conteo de seguidores de la página.
  - Actualizar nuevos comentarios hechos por seguidores.
  - Buscar nuevas publicaciones.
  - Actualizar el conteo de likes, comentarios y shares de las publicaciones existentes.
  - Actualizar los comentarios hechos a las publicaciones.
  - Actualizar los likes de las publicaciones.
  - Actualizar el registro de los seguidores de la página y sus likes o comentarios.
  - Actualizar los datos de perfil de un seguidor activo.
- Twitter
  - Actualizar el conteo de seguidores.
  - Actualizar menciones de seguidores.

- Buscar nuevos Tweets.
- Actualizar el conteo de favoritos, comentarios y compartidos de los Tweets.
- Actualizar el registro de los seguidores de la página, sus favoritos y menciones.
- Actualizar la los datos de perfil de un seguidor activo.

- Instagram
  - Actualizar el conteo de seguidores.
  - Buscar nuevas publicaciones.
  - Actualizar el conteo de likes y comentarios de las publicaciones existentes.

5.2 Tecnologías empleadas para la creación de la solución

5.2.1 HTML5 y CSS3

HTML5[1] es la versión más reciente del estándar que define HTML (Hyper Text Markup Language). Éste es el sistema que permite la modificación de la apariencia de las páginas web. Con HTML5 los navegadores actuales saben cómo mostrar una página web, como acomodar los elementos, como mostrar imágenes o videos y donde mostrar los textos.

HTML5 tiene muchos nuevos elementos y una larga lista de tecnologías que permiten desde la construcción de páginas web sencillas (texto, colores, formularios) hasta las páginas más complejas (galerías de imágenes, juegos, videos, streams, chats ,etc). La gran mayoría, si no es que la totalidad de las páginas web hoy en día están hechas usando este estándar.

5.2.2 CSS3

CSS3 es la versión más reciente del lenguaje de Hojas de Estilos en Cascada mejor conocido como CSS por sus siglas en inglés. Ésta versión contiene una gran variedad de propiedades que permiten la creación de una página web visualmente completa, con animaciones, transiciones, acciones sin la necesidad de usar Javascript. CSS es al día de hoy un lenguaje en desarrollo que va liberando nuevos módulos, con nuevas funcionalidades periódicamente.

CSS como su nombre lo indica es lo que le da estilos a cada página web. Sin esta herramienta las páginas estarían aún hechas en fondo blanco con letras negras. Actualmente CSS3 consta de poco más de 545 propiedades[2], con la combinación de estas y sus valores está construida la maqueta de cada página que se encuentre en internet.

5.2.3 JavaScript (JS)

JavaScript un lenguaje de programación ligero interpretado con funciones de primera clase, es decir, que las variables pueden ser tomadas como cualquier otra variable. Esto hace que se puedan crear callbacks, que

[1] HTML5 - https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/HTML5
[2] CSS3 - https://www.w3.org/Style/CSS/all-properties

son funciones que reciben como parámetros otras funciones para usarlas dentro, esto es muy común en las bibliotecas y frameworks.

JS es un lenguaje dinámico, basado en prototipos, multiparadigma, compatible con orientado a objetos, imperativo y declarativo. Es mayormente conocido por ser el lenguaje usado para el desarrollo de páginas web del lado del cliente, aunque también se usa del lado del servidor con Node.js, que es un ambiente de ejecución de JS multiplataforma. Node.js actualmente es uno de los ambientes de servidores más veloces que existen, ya que ejecuta programas de manera asíncrona en un solo hilo, sin bloqueo, que es eficiente para la memoria.

JavaScript es usado del lado del cliente en la web, para programar el comportamiento de las páginas web ante algún evento. Aunque el lenguaje es sencillo de aprender y de usar, es muy común que las páginas estén desarrolladas usando un framework o una o muchas bibliotecas en este lenguaje, la biblioteca más popular es JQuery que es una biblioteca que se enfoca en la manipulación del DOM, llamadas AJAX y el manejo de eventos.

JQuery es frecuentemente utilizado para desarrollar páginas web de poco contenido, ya que, solo deberían ser necesarios pocos scripts. Y es recomendable usarlo en páginas pequeñas porque, si bien la sintaxis de JQuery es fácil de aprender, las líneas de código requeridas para modificar diferentes componentes del DOM (si se escribe una sentencia por línea) son muchas, lo que da como resultado que, si no se desarrolla adecuadamente una página se termina con muchos scripts de muchas líneas para modificar poco contenido del sitio.

A continuación se muestra el fragmento de un script en JS con JQuery que al recibir el evento `submit`, crea un nuevo elemento dentro de una lista con el texto capturado en un formulario, y que le asocia un evento al elemento creado, tal que, al dar clic sobre éste se destruye.

```
$(function() {

  var $list, $newItemForm;
  $list = $('ul');
  $newItemForm = $('#newItemForm');

  $newItemForm.on('submit', function(e) {
    e.preventDefault();
    var text = $('input:text').val();
    $list.append('<li>' + text + '</li>');
    $('input:text').val('');
  });

  $list.on('click', 'li', function() {
    var $this = $(this);
    $this.remove();
  });

});
```

Si se controlan más eventos con acciones más complejas como: animar el elemento creado, ocultarlo aumentando la opacidad, agregando estilos, validar el texto ingresado, etc. el script quedaría mucho más largo.

Para desarrollar páginas web grandes es recomendable usar un framework, ya que proveen una estructura de directorios que organiza las partes del sitio y contiene las funciones necesarias para crear una página

web robusta. La ventaja de usar los frameworks de JavaScript es la eficiencia y la organización general que aportan a un proyecto.

5.2.4 AJAX

AJAX es el acrónimo de *Asynchronous JavaScript And XML*, es decir: *Javascript y XML Asíncrono*. AJAX no es una tecnología o una biblioteca, es una técnica que permite, mediante programas escritos en Javascript, que un servidor y un navegador intercambien información, de forma asíncrona[3].

AJAX permite que el contenido de una página web sea actualizado y simultáneamente se puedan hacer solicitudes al servidor cuando un usuario realiza una acción.

El funcionamiento de AJAX en el sitio web de MINI es el siguiente: primero el cliente a través del navegador solicita el contenido estático de la página con el método GET del protocolo HTTP, el contenido que envía el servidor es documentos HTML, los CSS, imágenes y scripts Javascript. Una vez que se terminan de interpretar los scripts de Javascript, se realiza una petición HTTP asíncrona al servidor para obtener la información inicial de todas las redes sociales, simultáneamente el navegador termina de cargar en la página las imágenes, inserta el contenido indicado por AngularJS en la página, calcula las animaciones, etc. Una vez que se obtiene respuesta del servidor, se actualiza el contenido de la página (este cambio en el contenido de la página no provoca que se recargue la página), y mientras sucede esta actualización del contenido, se realiza la siguiente petición al servidor para solicitar información más fresca para la página (información de otra red social o país).

5.2.5 AngularJS

AngularJS[4] es un framework MVC (Modelo Vista Controlador) de JavaScript para el desarrollo web de la aplicación front-end[5], es de código libre y fue desarrollado por un grupo de desarrolladores de Google.

AngularJS permite usar HTML como lenguaje plantilla y permite extender la sintaxis HTML con directivas y atributos, para expresar los componentes de la aplicación de manera clara, manteniendo la semántica y sin necesidad de emplear librerías externas como jQuery.

Desde su segunda versión AngularJS permite el todo el desarrollo con el lenguaje TypeScript. TypeScript es un superconjunto de JavaScript, lo que significa que contiene toda la funcionalidad de JavaScript. Por lo tanto, cualquier programa escrito en JavaScript válido, funciona también en TypeScript. De hecho, TypeScript transpila a JavaScript puro usando el compilador *tsc*. Con TypeScript toda la sintaxis y la manera de hacer las cosas en el código es la misma, lo que agrega coherencia a la información y a la forma de leer el código.

En AngularJS se programó que cada cierto intervalo de tiempo se realizará la petición para obtener la información más reciente de las redes sociales si durante un lapso de tiempo no había interacción del cliente con el sitio. Cuando se recibe información actual nuevamente se crean etiquetas HTML, se eliminan las etiquetas antiguas y se insertan las nuevas. Para el cliente es imperceptible la eliminación e inserción, este sólo ve las animaciones de entrada de elementos que indican que hay información reciente.

[3] AJAX - https://developer.mozilla.org/en-US/docs/Web/Guide/AJAX/Getting_Started#What's_AJAX

[4] AngularJS - https://docs.angularjs.org/guide

[5] Este término es frecuentemente usado en el desarrollo web y se refiere a la interfaz de la aplicación con la cuál un cliente interactúa directamente.

5.2.6 PHP y Laravel

PHP es un lenguaje de programación del lado del servidor, independiente de plataforma, de código abierto muy popular para el desarrollo web, con un gran catálogo de funciones y mucha documentación.

Las páginas que se ejecutan en el servidor pueden realizar accesos a las bases de datos, conexiones de red y otras tareas para crear la página final que verá el cliente. Lo que distingue a PHP de algo del lado del cliente como JavaScript es que el código es ejecutado en el servidor generando HTML y lo envía al cliente. El cliente sabe el resultado del código ejecutado pero no sabe el código generador.

Este lenguaje de programación está preparado para realizar muchos tipos de aplicaciones web gracias a la extensa biblioteca de funciones que tiene. La biblioteca de funciones cubre desde cálculos matemáticos hasta tratamiento de conexiones de red. Una de las más importantes capacidades de PHP es la compatibilidad con las bases de datos más comunes, como MySQL. Oracle, Postgres.

Laravel[6] es uno de los frameworks basado en MVC de PHP, que permite el uso de una sintaxis refinada y expresiva para crear código de manera sencilla. Las herramientas más destacables de Laravel son:

- **Eloquent**: Eloquent[7] es el nombre del **ORM** (Object Relational Mapping) que incluye Laravel para manejar de forma fácil y sencilla los procesos correspondientes al manejo de bases de datos del proyecto. Las consultas se realizan por medio de llamadas a métodos del ORM y el ORM transforma a la sintaxis de la base de datos usada la consulta solicitada, lo que no permite procesar consultas SQL directamente y así protege la aplicación de inyección SQL.
- **Blade**: Es un sistema de plantillas para crear vistas, que permite extender plantillas creadas y secciones en otras vistas. Puede ser parecido a el uso de `include` para añadir en un archivo php el contenido de otro. Pero la diferencia radica en que las plantillas blade están especializadas en HTML y minimizan el código usado para crear condicionales, ciclos, la impresión de variables y más. Blade es parte de la capa de vista.
- **Routing**: Es un sistema de organización y gestión de rutas que permite controlar de manera eficiente las rutas del sistema. Permite la creación de rutas con métodos, GET, POST, PUT, PATCH, DELETE y puede asociarse cada una a acciones del controlador. Así tras una petición se ejecutará la acción con base en el método y la ruta.

En Laravel se programaron los controladores y tareas necesarias para la sincronización de la base de datos con la información de las redes sociales. La base de datos era actualizada y se obtenían los datos de ella usando el ORM. Por medio del API y usando el *Router* se accedía a los controladores, en estos se organizaban los datos obtenidos por país y eran devueltos como una respuesta JSON. Con blade se crearon plantillas para la barra superior, la barra lateral, y 4 vistas, una para cada red social y una para el ranking. Al servir la página, se unía la información de cada vista y sección, se generaba el HTML y se enviaba al cliente. Desde el momento que la página web era recibida por el cliente ésta era controlada con AngularJS y con AJAX se llevaba a cabo toda la comunicación con el servidor.

[6] Laravel - https://laravel.com/
[7] Eloquent ORM - https://laravel.com/docs/5.8/eloquent#introduction

5.3 Interfaz como aplicación de una sola página

La interfaz web está desarrollada con el framework **AngularJS** en su primera versión, este es un framework de Javascript mantenido por Google, que se utiliza para crear aplicaciones web de una sola página **SPA**[8].

Se desarrolló el front-end como un SPA para minimizar las llamadas al servidor al cargar el contenido. De haberse desarrollado como una **MPA** (Multiple-Page Application), en cada cambio de sección se pediría la información de la página al servidor y este respondería la página completa para que fuera rendereada de nueva cuenta por el navegador, así cada acción podría tardar demasiado por lo que la disponibilidad y experiencia de usuario estarían comprometidas. Debe mencionarse que existe la posibilidad de usar MPA con AJAX lo que mejora la experiencia y permite actualizar solo partes particulares de la aplicación. Por otro lado, agrega más complejidad y es más difícil de desarrollar que una aplicación de una sola página.

Al tener la interfaz web como una aplicación de una página, sólo se necesita cargar una vez la información como íconos, tipografías, estilos, HTML, imágenes fijas y JavaScript. La información que hace dinámica la página, como información de países y publicaciones se obtiene mediante peticiones AJAX al **API REST** con ayuda del módulo HTTP de Angular.

5.4 Arquitecturas orientadas a servicios

Se eligió el tipo de arquitectura de software que se basa en la integración de aplicaciones mediante servicios; Arquitectura Orientada a Servicios (SOA por sus siglas en inglés).

5.4.1 SOA

SOA[9] es un estándar que presenta todos los procesos de un negocio de un modo orientado a servicios. El objetivo de esta arquitectura es separar la lógica de integración de negocio de la implementación. A grandes rasgos, para lograr ésta arquitectura, se descompone la lógica de negocio de una organización (o partes de ella) en pequeñas unidades de funcionalidad. Estas pequeñas unidades son los servicios.

Para lograr esta arquitectura pueden crearse servicios web[10] basados en SOAP o REST. SOAP y REST son siglas asociadas a protocolos para el diseño y desarrollo de servicios web. El uso más común que se suele dar a estos servicios es el de integrar diferentes sistemas o componentes de una o varias plataformas, los servicios web, por sus especiales características de interoperabilidad, reutilización, modularidad, etc., se ajustan a las necesidades y principios de las arquitecturas SOA, por esto se ha convertido en la aproximación más extendida de implementar esta arquitectura.

[8] SPA - Single Page Application o "Aplicaciones de una sola página", son sitios web donde los usuarios perciben una experiencia similar a la que se tiene con las aplicaciones de escritorio. En este tipo de sitios la página no se recarga, no existe una navegación de una página a otra totalmente diferente, sino que se van intercambiando las "vistas".

[9] SOA - https://www.ibm.com/support/knowledgecenter/es/SSFTN5_8.5.7/com.ibm.wbpm.wid.main.doc/prodoverview/topics/csoa.html

[10] El organismo W3C define servicio web como un sistema software diseñado para soportar la interacción máquina-a-máquina a través de una red y de forma interoperable.

5.4.2 SOAP

SOAP[11] son las siglas de Simple Object Access Protocol, es un protocolo de comunicación de aplicaciones, la comunicación se realiza por medio del intercambio de datos XML.

SOAP se basa exclusivamente en XML para proporcionar servicios de mensajería que pueden llegar a ser extremadamente complejos en algunos casos, por ejemplo, accediendo al servicio web a través de Javascript. Todas las solicitudes SOAP se envían a través de la solicitud POST.

Ejemplo de una respuesta XML de un Web Service con protocolo SOAP que responde categorías.

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <soap:Body>
    <LeeTodasLasCategoriasResponse xmlns="http://ejemplo.com/categoriasWS">
      <LeeTodasLasCategoriasResult>
        <Estatus />
        <Lista>
          <Categorias>
            <DatosDeLaCategoria>
              <IdCategoria>1</IdCategoria>
              <NomCategoria>Pastor</NomCategoria>
            </DatosDeLaCategoria>
          </Categorias>
          <Categorias>
            <DatosDeLaCategoria>
              <IdCategoria>9</IdCategoria>
              <NomCategoria>Servicio Domicilio</NomCategoria>
            </DatosDeLaCategoria>
          </Categorias>
          <Categorias>
            <DatosDeLaCategoria>
              <IdCategoria>10</IdCategoria>
              <NomCategoria>Cervezas de Importacion 58</NomCategoria>
            </DatosDeLaCategoria>
          </Categorias>
          <Categorias>
            <DatosDeLaCategoria>
              <IdCategoria>27</IdCategoria>
              <NomCategoria>Aguas Frescas</NomCategoria>
            </DatosDeLaCategoria>
          </Categorias>
        </Lista>
      </LeeTodasLasCategoriasResult>
    </LeeTodasLasCategoriasResponse>
  </soap:Body>
</soap:Envelope>
```

Hoy en día SOAP se utiliza principalmente, para la comunicación de Servidor a Servidor pues es un protocolo mucho más robusto, tiene un tipado más fuerte, permite agregar metadatos mediante los atributos y

[11] SOAP - https://www.w3schools.com/xml/xml_soap.asp

más, por lo mismo, SOAP es un formato más pesado, tanto en tamaño como en procesamiento.

5.4.3 REST

REST (abreviación de Representational State Transfer) por otro lado, es una tecnología mucho más flexible que transporta datos por medio del protocolo HTTP, permite utilizar los diversos métodos que proporciona HTTP para comunicarse, como son: `GET`, `POST`, `PUT`, `PATCH`, `DELETE` y a la vez utiliza los códigos de respuesta nativos de HTTP (200, 404, 409, 500, etc).

Con REST es posible transmitir prácticamente cualquier tipo de datos legible por máquina, ya que el tipo de dato está definido por la cabecera `Content-Type`, lo que permite estructurar datos en XML, YAML, etc. aunque generalmente se prefiere JSON.

Los servicios web basados en REST son completamente libres de estado. El manejo de estado de una conversación es responsabilidad únicamente del cliente. El servidor no guarda de ninguna forma el estado.

Ejemplo de SOAP transformado a una respuesta JSON.

```
{
  "estatus" : "",
  "categorias" : [
    {
       "id" : 1,
       "nombre" : "Pastor"
    },
    {
       "id" : 10,
       "nombre" : "Cervezas de Importación 58"
    },
    {
       "id" : 27,
       "nombre" : "Aguas Frescas"
    }
  ]
}
```

La principal razón por la que decidí crear los servicios web basados en **REST** es porque pueden transmitir cualquier tipo de dato y no solamente XML, en especial porque lo común es transmitir la información por JSON que son significativamente más livianos y más rápidos de procesar. JSON es interpretado de manera natural por JavaScript, lo que hace que frameworks de Javascript se aprovechen al máximo, pues se pueden enviar peticiones directas al servidor por medio de AJAX y obtener los datos de forma nativa.

Si bien con SOAP seguridad está bien estandarizada, la información transmitida entre cliente y servidor en esta aplicación es completamente pública y no existen servicios con los que el cliente pueda modificar información del sistema, por lo que para este proyecto no era prioritaria la seguridad entre cliente y servidor, por lo que se optó por encriptar la comunicación a través de TLS.

5.5 Metodología de desarrollo

5.5.1 Modelo de proceso incremental

"El modelo de proceso incremental se centra en que en cada incremento se entrega un producto que ya opera. Los primeros incrementos son versiones desnudas del producto final, pero proporcionan capacidad que sirve al usuario y también le dan una plataforma de evaluación."[12]

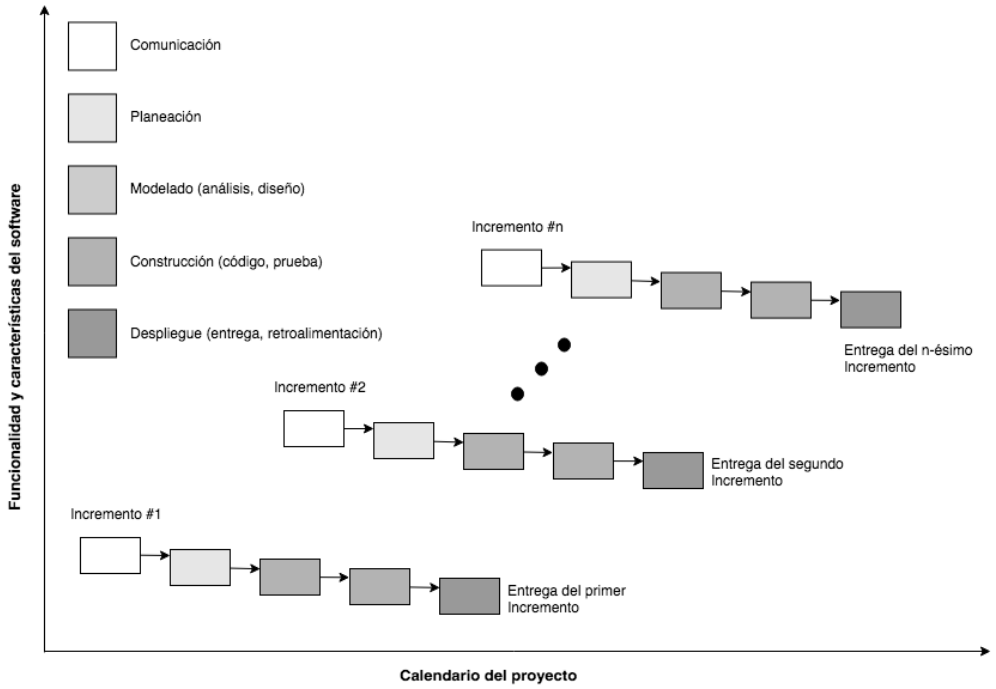


Figura 5.1: Proceso Incremental.

El primer incremento fue el más largo y fue un trabajo en conjunto entre el equipo de desarrollo, el equipo de UX (Experiencia de Usuario por sus siglas en Inglés) y el equipo de UI (Interfaz de Usuario por sus siglas en Inglés), UI y UX pertenecientes a Social Noise.

Durante este incremento se crearon las primeras propuestas de diseño mostradas en el Capítulo 1, se investigó sobre las limitantes de las APIs de las redes sociales para poder cumplir con este diseño, se desarrolló la maqueta de la propuesta de diseño en HTML y CSS, se solicitaron los accesos a las cuentas de las redes sociales para poder crear las llaves de acceso, se comenzó con el modelado de la base de datos según lo que se iba obteniendo de la investigación, se configuró un servidor de aplicaciones para poder mostrar la maqueta desarrollada.

Para el segundo incremento cambió el diseño al que sería el más parecido al final, el cambio vino de la retroalimentación del cliente y de las posibilidades que ofrecía el API de cada red social, en este incrementó se creó el proyecto con Laravel, los modelos de Eloquent para la comunicación con la base de datos y se crearon las plantillas base en Blade.

[12] Roger S. Pressman, Ph.D.; Ingeniería de software Un Enfoque Práctico; McGraw-Hill, 7.ª edición. México, 2010, pág. 36

Las entregas se enfocan en lo visual ya que el cliente no era capaz de ver la lógica de negocio que se iba desarrollando. Las entregas para el cliente de los siguientes incrementos fueron:

- Desarrollo de plantilla para Facebook en Blade.
- Desarrollo de plantilla para Twitter en Blade.
- Desarrollo de plantilla para Instagram en Blade.
- Desarrollo del Ranking de Comunidad en Blade.
- Conexión al REST API para obtener la información de cada red social.
- Conexión de la nube de términos.
- Conexión del Ranking de Comunidad.
- Transición automática entre cada país, cada red social y el ranking.

Las entregas consistieron en actualizar el desarrollo en el servidor de aplicaciones, así el cliente podía hacer su revisión en línea y devolvía la retroalimentación.

Durante cada incrementó se iban desarrollando a la par componentes para la presentación y la lógica de negocio, por lo que, para cuando fue necesaria la comunicación de la interfaz con el servidor, ya estaban listos los servicios web de consulta de Facebook, Twitter e Instagram.

El último incremento consistió en la planeación y despliegue del proyecto en ambiente productivo.

5.6 Instalación

La página web de redes sociales de MINI se despachó desde un servidor de nube virtual (VSP) de Rackspace (compañía de gestión de computación en la nube) con sistema operativo CentOs 7. En este servidor se instaló y configuró Apache, MySQL, PHP (el conjunto de estas tecnologías es conocido como LAMP) y se creó el Cron Job.

Se creó un servidor en la nube con el sistema manejador de base de datos MySQL. En este servidor se ejecutó el script para la creación de la base de datos y las tablas de la aplicación. Se configuró el servidor para que hiciera un respaldo de la base de datos cada día por la noche.

Se tomó la decisión de tener 2 servidores en la nube, uno para la aplicación y otro para la base de datos para no comprometer la información de la base de datos si llegara a ocurrir un error con el servidor de aplicaciones. Y si llegase a ocurrir un error con el servidor de aplicaciones tan sólo se tendría que crear una instancia del servidor a partir de una imagen de respaldo y la aplicación web estaría corriendo de nuevo. Si el error hubiera llegado a ocurrir en el servidor de base de datos se crearía una instancia y se reestablecería la base de datos con el respaldo del día anterior. Ningún dato se perdería ya que al iniciar la aplicación se actualiza la base de datos con la información que no se obtuvo al momento cuando el servidor estuvo fuera de línea.

La instalación del proyecto incluyó el montaje de una pantalla en la que se mostrará la página web desarrollada. La instalación de la pantalla se realizó en las oficinas de corporativas de BMW Group México.

Para montar la pantalla en las instalaciones:

- Se conectó una computadora Mac Mini por medio de un cable de ethernet a internet para que pudiera tener acceso al servidor donde se instaló la aplicación.
- Se conectó la computadora a una pantalla de televisión Full HD y se tuvo que configurar para que se viera la interfaz correctamente a toda la resolución de la pantalla.
- Se instaló el navegador Google Chrome en la computadora. Se optó por este navegador ya que es el que soporta la mayor cantidad de propiedades CSS[13] con respecto a otros navegadores y soporta las versiones más recientes de AngularJS[14]. Además en mi experiencia, es en este navegador en el que se encuentran menos fallas de HTML, CSS y JavaScript, con el otros navegadores se encuentran errores o comportamientos inesperados al momento de renderizar las páginas.
- Se configuró la computadora para que no fuera suspendida o se cerrara la sesión, después de cierto tiempo de inactividad. Se realizó esta configuración para que la pantalla siempre estuviera mostrando la página web.
- Adicionalmente se instaló TeamViewer en el equipo para que, si llegara a haber una incidencia pudiéramos acceder al equipo para poder revisar los registros de error del navegador y resolver los errores que pudieran llegar a ocurrir durante la ejecución de la aplicación.

Una vez finalizada la configuración del ambiente LAMP en los servidores y finalizado el montaje de la pantalla en las oficinas de BMW, se inició el servidor web, se inició el servicio cron y con esto se empezó ejecutar la aplicación.

[13] CSS Browser Support - https://www.w3schools.com/cssref/css3_browsersupport.asp
[14] AngularJS Browser Support - https://angular.io/guide/browser-support

CAPÍTULO 6

Conclusiones

6.1 ¿Qué ventajas brinda el desarrollo a MINI?

La aplicación web se usó durante poco menos de 2 años en las oficinas de Mini Cooper. Durante este tiempo se usó el propósito que fue mencionado, el cual fue medir el desempeño de las empresas de *Marketing Digital* contratadas para gestión de sus redes sociales y actuar acorde a los resultados obtenidos de ellas.

Mini Cooper contrata a las empresas de Marketing Digital para que los especialistas que trabajan en ella, puedan realizar un correcto manejo de todas las herramientas digitales necesarias para conseguir cumplir un mejor posicionamiento de la marca.

Al contratar una empresa de marketing se esperaría que el posicionamiento de la marca crezca, que haya nuevos clientes interesados en los productos que se ofrecen y se mantengan los clientes que ya se tenían. Al área de publicidad de MINI le pareció correcto evaluar el desempeño de las empresas de marketing que contratan, viendo por ellos mismos diariamente el contenido que era publicado en sus redes sociales, sus likes, comentarios, número de veces compartido, el crecimiento de seguidores, etc.

Por ejemplo, si MINI pagó por la creación de cierto número de publicaciones que anunciaran un evento importante que se llevaría a cabo y que cada una pudiera alcanzar cierto número de interacciones de parte de los clientes, lo correcto sería que el contenido fuera publicado en tiempo y forma y cumpliera el objetivo deseado. Si se había convenido que cierto contenido fuera publicado a una cierta hora, con la aplicación web desarrollada se podía corroborar que dicho contenido fuera liberado tan pronto la empresa de marketing lo hubiera publicado en las redes sociales.

Si las empresas de marketing contratadas no hubieran llegado a tener el desempeño esperado o no cumplieran con sus objetivos, MINI podría tomar acciones para remediar esta situación y lo haría tan pronto viera que los datos en paneles de redes sociales de la aplicación no mostraran los datos que ellos esperaban.

Analizando muy a grandes rasgos los datos almacenados se notó que el número de publicaciones aumentó y el seguidores creció desde que la aplicación web se ejecutó. No es posible saber si este incremento se puede atribuir a que la aplicación web desarrollada cumplió su propósito y como consecuencia de la visualización

frecuente de las redes sociales por parte de Mini hubo más atención en las agencias que las manejaban o si fue por algún motivo distinto.

A continuación se muestra una pequeña comparativa de los datos que se tenían al momento del inicio de la aplicación contra los últimos datos obtenidos por la aplicación.

6.1.1 Diferencia total de seguidores de Chile, México y Colombia

Red Social	Pais	Seguidores Iniciales	Seguidores Finales	Diferencia	Total de Publicaciones
Facebook	MX	604,690	764,523	159,833	881
Facebook	CH	184,344	191,274	6,930	675
Facebook	COL	177,107	199,027	21,920	789
Instagram	MX	10,923	45,028	34,105	477
Instagram	CH	1,081	1,864	783	91
Instagram	COL	8,758	18,631	9,873	405
Twitter	MX	61,491	91,310	29,819	820
Twitter	CH	126	179	53	411
Twitter	COL	6,977	8,181	1,204	693

6.2 Posibles mejoras al sitio desarrollado

Desde que la página web de Redes MINI se desarrolló, al día de hoy, las tecnologías han tenido muchas mejoras que añaden nuevas funcionalidades y que facilitan su uso, también han mejorado los frameworks desarrollados para estas.

Cuando se desarrolló el sitio AngularJS estaba en su primera versión y la versión 2 estaba en fase Beta, ahora se encuentra en su séptima versión. Por otro lado, se usó PHP 5.6, ahora se encuentra en la versión 7 (no existió la versión 6).

Si se tuviera que desarrollar la página de web en este momento se elegiría para desarrollar del lado del servidor la tecnología **Sails.js**. Sails.js es un framework para NodeJS. Facilita el desarrollo de APIs REST, servidores de archivos, seguridad y lo más importante **WebSockets**[1]. Los **WebSockets** por definición es una alternativa a la comunicación HTTP que ofrece un canal bidireccional de larga duración entre el cliente y el servidor. Se usaría para establecer un canal de comunicación con la interfaz y mediante notificaciones se informaría inmediatamente a la interfaz si llega a actualizarse una publicación, de esta forma solo se actualizará el objeto JSON que cambió y no todo el documento. No sería necesario que la interfaz realizará peticiones cada cierto intervalo de tiempo para consultar si hay nueva información para mostrar, el servidor notificará inmediatamente a la interfaz si ocurre algún cambio en los datos. No solo habría mejoras a nivel de lógica, también las habría a nivel de diseño y experiencia de usuario. Por ejemplo, cada conteo de likes, comentarios y compartir podría incrementar de manera independiente. Podría verse como crece o decrece el número de seguidores o como incrementan los contadores de cada publicación, incluso con Facebook podría verse como una publicación pasa a ser más relevante reacomodando los elementos de la plantilla, lo mismo con la nube de términos, alguna palabra podría pasar a una mejor posición si se detecta que sus ocurrencias aumentaron tras la inserción de nuevas publicaciones.

[1] WebSockets - https://developer.mozilla.org/en-US/docs/Web/API/WebSocket

Se puede usar la asincronía nata de NodeJS para actualizar los datos de cada perfil del país por red social a la vez, es decir, actualizar la información del perfil de Facebook de MINI México y del perfil de Facebook MINI de Uruguay en paralelo, esto no provocaría problemas de concurrencia ya que los datos almacenados no guardan una relación de orden y tienen dependencias entre sí, lo que los relaciona a otras tablas son los identificadores y estos son proporcionados por las redes sociales.

Se usaría también el manejador de procesos PM2[2], este manejador permitiría que la aplicación estuviera siempre ejecutándose en segundo plano y facilitara el monitoreo de la aplicación ya que incorpora un gestor de registros lo que permite la lectura de los registros en tiempo real para ver qué sucede con la aplicación. Se podrían separar las tareas de actualización de Facebook, Twitter e Instagram y con PM2 se haría que cada tarea fuera ejecutada como un proceso diferente. PM2 lleva a cabo esto de manera automática usando el API de Clúster de Node.js para generar múltiples procesos.

Del lado del cliente la aplicación se desarrollaría también en AngularJS, pero esta vez se usaría la versión 7 que es la más reciente. Las vistas serían programadas como componentes de Angular, por lo que al igual que con Blade, serían programados diferentes componentes, uno para la barra superior, uno para la barra lateral, 4 componentes para cada una de las redes sociales y una para el ranking.

El desarrollo del front-end se llevaría a cabo en menor tiempo debido a las ventajas que fueron incorporadas al framework. Se reduciría la lógica necesaria para el cambio de información entre países y redes sociales y la lógica necesaria para solicitar la información de las interfaces en caso de haber un cambio. Se podría invertir la mayor parte del tiempo en mejorar las animaciones de la interfaz.

La arquitectura de la aplicación sería de 3 capas y 3 niveles. Los niveles son componentes físicos separados del mismo sistema, los niveles serían:

- **Nivel de Almacén de Datos**: El servidor de aplicaciones donde está alojado el proyecto.
- **Nivel de Aplicación**: El servidor donde está alojada la base de datos.
- **Nivel de Cliente**: La máquina del usuario que accede a la aplicación.

Las capas se refieren a la organización lógica del sistema, las capas serían:

Capa de Acceso a Datos:

La capa de acceso a datos está conformada por el ORM de Sails.js llamado **Sequelize** que al igual que con Eloquent cada tabla de la base de datos tiene correspondencia a un Modelo, el cual se usa para interactuar con las tablas de la capa de datos.

Capa de Negocio:

La capa de negocio es la capa que gestiona la lógica de la aplicación y está conformada por todo lo desarrollado en Sails.js, PM2 y Nginx[3]. Esta capa se comunicaría con la capa de acceso a datos a través del uso de las funciones de los Modelos. También se comunicaría con la capa de presentación a través de los WebSockets y el API REST.

Capa de Presentación:

La capa de presentación estaría conformada por la aplicación en Angular 7, que sería compilada y después enviada al navegador de cliente para que fuera ejecutada.

[2] PM2 - https://www.npmjs.com/package/pm2
[3] Nginx - https://www.nginx.com/resources/glossary/nginx/

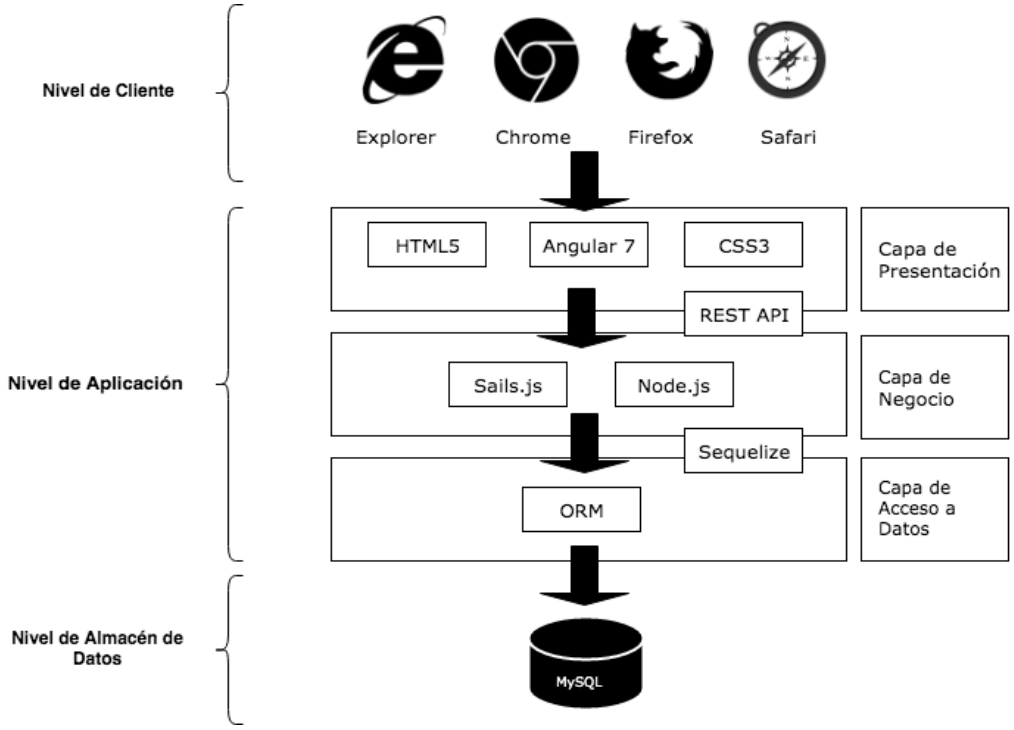


Figura 6.1: Diagrama de arquitectura con diferentes tecnologías compuesto de 3 capas y 3 niveles.

La aplicación web que se desarrolló consta de 3 niveles y tiene 2 capas lógicas que no están bien desacopladas: la capa de presentación y la capa de negocio. No están bien desacopladas porque se usa el sistema de plantillas de Blade junto con Angular para poder crear la interfaz, entonces se tiene lógica correspondiente a la capa de presentación en la capa de negocio. A continuación se muestra el diagrama de la arquitectura del sistema.

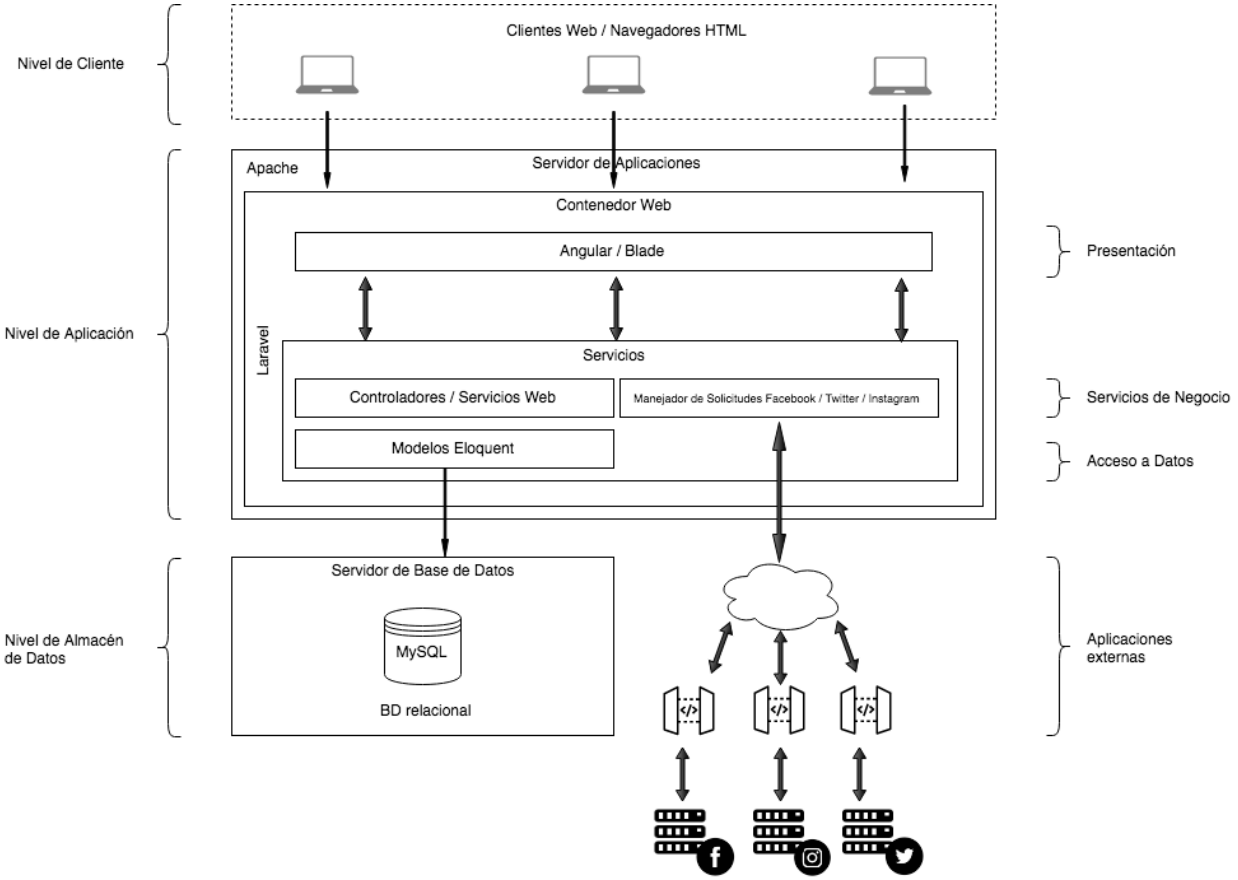


Figura 6.2: Diagrama de arquitectura de la aplicación web desarrollada.

En la siguiente figura se muestra la arquitectura deseada con la tecnología que se usó. En esta, la capa de presentación está compuesta por Angular y es independiente de la lógica de negocio que está desarrollada en Laravel y solo hay comunicación entre ellas a través de los servicios web.

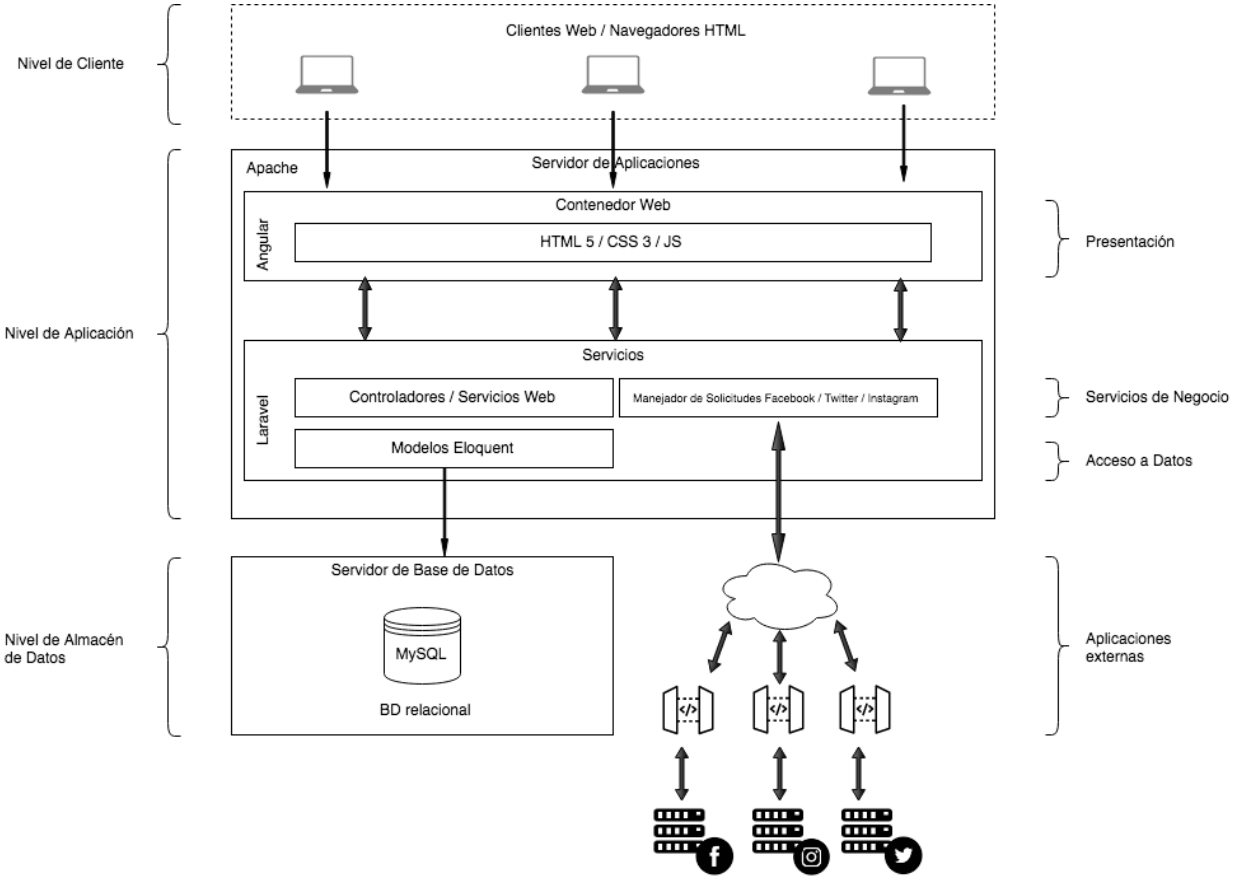


Figura 6.3: Diagrama de arquitectura deseada.

CAPÍTULO 7

Reflexiones

Durante el desarrollo del proyecto Redes MINI aprendí más sobre las tecnologías usadas en el desarrollo Web, sobre la gestión de las redes sociales, la manera en que organizan su información, sobre los datos de acceso público que proporcionan, sus políticas de privacidad, la información que se puede obtener de ellas, sobre el uso de sus APIs y sobre la gran utilidad de las redes sociales para las empresas y para el desarrollo de aplicaciones. Hoy en día me sigo actualizando sobre las nuevas herramientas que liberan las redes sociales, que van cambiando con frecuencia por las nuevas tecnologías y las políticas. Así, si se me solicita analizar un proyecto que use las redes sociales, puedo mencionar las limitantes, posibles riesgos, que se puede y que no se puede hacer con ellas.

Por mi formación en la licenciatura en Ciencias de la Computación, me fue sencillo entender y aprender distintas tecnologías ya que tengo una base sólida de lógica y de estándares de programación. A mi parecer, a diferencia de otras carreras que se enfocan en Tecnologías de la Información (TI), en la licenciatura de Ciencias de la Computación comprendemos las implicaciones del desarrollo de software, i.e, vemos desde otra perspectiva todo lo que implica realizar un proyecto de la mejor manera, desde análisis de requerimientos, calcular tiempos, escoger las tecnologías que mejor sea adapten al proyecto, curva de aprendizaje del lenguaje a usar, el planeamiento, repartir tareas, elegir los patrones de arquitectura de software, el uso de diagramas para entender mejor los componentes del sistema, elegir el sistema manejador de bases de datos adecuado, mantenimiento del sistema, etc.

Los fundamentos que aprendí en esta carrera como: tener buenas prácticas de programación, entender los tipos de datos y hacer correcto uso de ellos, el uso del álgebra Booleana de manera eficiente, el análisis de algoritmos para hacer programas eficientes, todo lo relacionado a la elaboración de bases de datos, el entender las metodologías ágiles, conocer los sistemas operativos y la seguridad de la información, son fundamentos que de alguna forma lo aplico en mi día a día de manera natural en mi trabajo y a mi parecer es conocimiento que debería tener toda persona con algún cargo en un puesto en TI.

Durante el tiempo que llevo trabajando, que es casi 5 años, he tenido de compañeros a personas egresadas de diferentes universidades o carreras afines y siempre he trabajado con ellas sin tener algún problema, aunque tenemos formas diferentes de pensar y trabajar los proyectos. Por ejemplo, no todos conocen bien y aplican los fundamentos que mencioné antes. Lo más común es que no conozcan mucho sobre análisis de algoritmos. Muchos son excelentes programando pero sólo conocen un par de lenguajes, no suelen cuestionarse por qué hacen lo que hacen y si lo hacen de la mejor manera. En lo que va de mi experiencia laboral no he tenido mucho desafío de su parte cuando hay planeación de algún proyecto, por lo regular suelo ser yo quien ponga los lineamientos de desarrollo que se deberán seguir.

Por otro lado, cuando trabajo con gente con formación en Ciencias de la Computación, siempre obtengo más conocimiento y nuevas ideas. Las personas son muy críticas de sí mismas y de lo que ya está desarrollado, de lo que ellos están desarrollando y lo que está desarrollado alguien más. Buscan tiempo libre para poder mejorar el código que desarrollaron antes, buscan si algún componente de código que desarrollaron puede ser mejorado para poder crear un módulo y usarlo en sus futuros proyectos, siempre hay debate sobre la mejor manera de implementar algo o sobre la mejor manera de llevar un proyecto, pocos se quedan callados si una idea no les agrada o no la entienden. En algunos momentos se me hizo pesado que siempre hubiera alguien que cuestionara todo lo que programaba y le encontrara detalles, pero me di cuenta que al final que eso es lo que me hizo crecer y mejorar mis habilidades. Si no hubiera sido así, mi conocimiento hubiera quedado el que tenía cuando comencé a trabajar y de ahí sólo seguiría reforzar lo malo.

Creo que nuestro amplio conocimiento en tecnologías, el conocer cómo y por qué funcionan y no usarlas solo porque sí, nos da ventaja sobre los que aprendieron de cursos por internet u otras escuelas técnicas, en varios lugares me he encontrado que los líderes técnicos o jefes de desarrollo son egresados de Ciencias de la Computación y tienen a su cargo a desarrolladores de Computación u otras carreras.

Durante mi tiempo trabajando, profundicé en mi conocimiento acerca de las diferentes tecnologías involucradas en el desarrollo web. He trabajado en varios proyectos de diferentes dificultades y he trabajado con distintas tecnologías, algunas veces he tenido la oportunidad de elegirlas y otras veces el cliente especifica que tecnologías hay que usar dependiendo de las especificaciones del proveedor de hosting que contrataron o de los estándares de las marcas para las que trabajan. El no conocer sobre las tecnologías que solicita el cliente nunca ha sido un impedimento para poder empezar a desarrollar un proyecto ya que me considero eficiente para dominar en poco tiempo una nueva tecnología. Cuando se aprueba un proyecto y no conozco las tecnologías, necesito un par de días antes de empezar a programar para investigar sobre las tecnologías requeridas, y debatir con el equipo de desarrollo sobre la cuál es la manera más eficiente el problema y cómo llevaremos el proyecto. Por lo regular, yo empiezo a conocer y aprender las nuevas tecnologías requeridas y despúes el resto de mis compañeros de equipo, hasta el momento esto ha funcionado bastante bien, porque si llegan a tener dudas cuando empiezan a abordar la tecnología puedo explicarles con ejemplos prácticos cuál es su funcionamiento y cómo se usará en el proyecto.

Me considero un desarrollador *Full Stack Web*, esto se puede definir como aquel que es capaz de desarrollar software de cliente y servidor. Y también pero en menor medida, me puedo desenvolver como Administrador de Sistemas de TI que es aquel que instala, configura el software, hardware y redes, monitorea el desempeño, soluciona errores del sistema, se asegura de la seguridad y eficiencia de la infraestructura.

Algunas de las tecnologías para el desarrollo de software y administración de sistemas que he manejado son:

Tecnologías de lado de Cliente, Servidor, Sistemas Operativos y Hostings		
HTML	Socket.io	SOAP
CSS	Swift	MySQL
Bootstrap	Apache Tomcat	PostgreSQL
JavaScript	IIS	SQL Server
ES6	PM2	MongoDB
Vue.js	Docker	Nginx
JSON	PHP	Apache
XML	ASP	Git
jQuery	C++	Linux
Angular	C#	Red Hat Enterprise Linux
React	Java	Windows Server
Grunt	Python	Amazon EC2
Gulp	Node.js	Rackspace
Sass	Sails.js	Azure
Less	Express	Hostgator
Ionic	REST	

En la mayor parte de los proyectos me he desempeñado como líder técnico, debido a mi conocimiento funcional de los sistemas y de su arquitectura. Y esto es gracias a que he trabajado en diferentes roles en el proceso de desarrollo de software, principalmente en el grupo de desarrolladores como son:

- Analista de sistemas.
- Arquitecto de Software.
- Revisor de la Arquitectura.
- Diseñador de Base de Datos.
- Revisor del diseño de base de datos.
- Programador.
- Revisor de Código.
- Integrador.
- Administrador de sistema.
- Documentador técnico.
- Especialista en herramientas.

Actualmente trabajo como Arquitecto de Software y es un trabajo que me gusta porque me involucro desde la concepción del proyecto hasta que es liberado. Como arquitecto de software debo considerar los requisitos empresariales, los requisitos del usuario y los requisitos del sistema, actuo como intermediario entre los clientes y el equipo de desarrollo, uso mis habilidades técnicas para producir un mejor plan inicial y requisitos más precisos para ahorrar tiempo y esfuerzo más adelante en el proyecto. Me gusta involucrarme desde la planeación del proyecto y para poder visualizarlo desde el principio como un sistema. A diferencia de un jefe de proyectos, cuyas tareas se centran en los elementos operativos del proyecto, yo tengo la capacidad de dar mi opinión sobre las ventajas y desventajas técnicas que pudiera tener el sistema que el cliente propone, para que se pueda mejorar y se llegue a una solución ideal.