API

Alejandro Hernández Cano
Castillo Barrón Fernando Daniel
Saavedra Escalona Braulio Rubén

1 ¿Cómo usar la aplicación?

La API se encuentra montada en un servidor público, con la dirección: `dogspott.000webhostapp.com`
Funciona mediante peticiones HTTP con parámetros GET y POST, y devuelve como respuesta un objeto en formato JSON.
Idealmente, la base de datos debe contar con 3 millones de registros en la tabla dog, que representa la información de los perros registrados en dogspott. Sin embargo, como usamos un servidor público, solo se pudieron subir 10 000 registros debido a la limitado ancho de banda que éste nos proporciona. La lista completa de los 3 millones de registros se encuentra guardada en 30 archivos SQL con 100 000 registros cada uno, y pueden descargarse desde el siguiente enlace: `https://drive.google.com/drive/folders/1S230ay_FQc971mHUiF7H6zGfMapUHqPQ?usp=sharing`

1.1 Endpoints

Los endpoints son representados por la API como archivos PHP que ejecutan la lógica de negocio y realizan la conexión con la base de datos. Actualmente cuenta con 6 endpoints ubicados en la carpeta raíz del dominio:

1.1.1 Login

- Endpoint: login.php
- Método: GET

- Parámetros
  - (GET) username: String - Nombre de usuario
  - (GET) password: String - Contraseña asociada al usuario
- Respuesta: Si el usuario con la contraseña existen en la base de datos,

```
{
    "status": "ok",
    "message": "--key--"
}
```

  Donde

```
--key--
```

  es a llave de autenticación.

- Descripción: recibe las credenciales de un usuario y determina si dichas credenciales son válidas para un usuario previamente registrado en la base de datos. La llave de autenticación de devuelve es única para dicho usuario, por lo que es importante para futuras peticiones.

1.1.2 Signup

- Endpoint: signup.php
- Método: GET
- Parámetros
  - (GET) username: String - Nombre de usuario a registrar.
  - (GET) password: String - Contraseña a asociar con el usuario.
- Respuesta: Si el usuario no existe en la base de datos,

```
{
    "status": "ok",
}
```

- Descripción: registra en la base de datos el nombre de usuario y la contraseña dada.

1.1.3 Feed

- Endpoint: index.php
- Método: GET
- Parámetros:
  - (GET) key: String - Llave de autenticación.
- Respuesta: Una lista con 2000 perros, cada uno con la siguiente estructura:

```
{
    "idDog": 1,
    "name": "Egret, great",
    "image" : "http:\/\/dummyimage.com\/227x121.png",
    "likes": 2
}
```

- Descripción: Devuelve una lista aleatoria de 2000 perros registrados en la base de datos. No incluye los comentarios de cada perro.

1.1.4 Detalles

- Endpoint: detalles.php
- Método: GET
- Parámetros:
  - (GET) key: String - Llave de autenticación.

  - (GET) dog_id: String - Id del perro a obtener.

- Respuesta: Un JSON con información del perro, con la siguiente estructura:

```
{
    "idDog": 1,
    "name": "Egret, great",
    "image" : "http:\/\/dummyimage.com\/227x121.png",
    "likes": 2,
    "comentarios": [
        {
            "date": "2019-12-12",
            "user_id": 2,
            "text": "Soy un comentario"
        }
    ]
}
```

- Descripción: Devuelve la información completa de un perro dentro de la base de datos, obtenido a través de su id.

1.1.5 Like

- Endpoint: like.php
- Método: GET
- Parámetros
  - (GET) key: String - Llave de autenticación.
  - (GET) dog_id: Number - Id del perro a dar like.
- Respuesta: un JSON vacío
- Descripción: Aumenta el contador de likes del perro en la base de datos.

1.1.6 Comentario

- Endpoint: comentar.php
- Método: GET
- Parámetros:
  - (GET) key: String - Llave de autenticación.
  - (POST) comentario: String - Contenido del comentario.
- Respuesta: un JSON vacío.
- Descripcion: registra un comentario dato en la base de datos, asociado a un perro previamente registrado.

2 Comentarios sobre la implementación

Login
Para la implemetación del login se requirieron de dos clases: la clase *conexion* y la principal llamada *login*. La primera enunciada se encarga de hacer una conexión con el servidor que tiene la base de datos; en cuanto a la segunda: ésta verifica que los valores de entrada (usuario y contraseña) pertenezcan a algún registro alojado en la base.
Login manda un mensaje de error en el caso de que no haya coincidencias entre la base de datos y la entrada; en caso contrario, la salida es un mensaje de entrada exitosa y la llave *hash* producida con la función criptográfica *sha256* que recibe como parámetros la concatenación del nombre de usuario y su contraseña.

Detalles

Para acceder a los detalles de una publicación se requieren dos llaves: la llave `key` del usuario con el que se está accediendo a la publicación y la llave `dog_id`, que es el identificador del perro al que se quiere buscar. El resultado será una cadena en formato `JSON` que contendrá el nombre del perro especificado, la candidad de likes que tiene la imagen y los detalles de cada comentario que tiene, entre otros.