Requerimientos No Funcionales

Usabilidad:

* El sistema debe proporcionar mensajes de error que sean informativos y orientados a usuario final.
* La aplicación web debe poseer un diseño “Responsive” a fin de garantizar la adecuada visualización en múltiples computadores personales, dispositivos tableta y teléfonos inteligentes.
* El sistema debe poseer interfaces gráficas bien formadas, amigables para el usuario.

Disponibilidad:

* Es condiciones de estrés a la app se le pasaran mas de 200 peticiones GET y esta seguirá funcionando correctamente
* El feedback del sistema puede ser visto hasta cierto porcentaje cargado cuando no haya internet

Rendimiento:

* Los datos modificados en la base de datos deben ser actualizados para todos los usuarios que acceden en menos de 2 segundos, para el cumplimiento de este requerimiento se supone que hay datos almacenados en cache, de lo contrario el tiempo de respuesta podría ser mayor.
* El sistema debe ser capaz de operar adecuadamente sesiones concurrentes.

Elasticidad:

* El sistema debe estar diseñado con una máxima flexibilidad de manera que con el efecto de la demanda este sea capaz de agregar o quitar recursos utilizando la escalabilidad todo esto en constancia de un administrador funcional del sistema.

Cache

Mantendremos en cache, datos sobre publicaciones para poder cumplir con los requerimientos de disponibilidad, así el usuario podrá tener ciertas imágenes y algunos eventos a los que él vaya a participar, para que pueda tener algo de feedback, así mientras el usuario se da cuenta que está pasando algo, tendrá acceso a un evento que tenga y no fallar por no saber dónde es, a qué hora, para que animales está orientado.

Estrategia:

Carga diferida

Como el nombre indica, carga perezosa es una estrategia de almacenamiento en caché que carga datos en la caché solo cuando es necesario. Funciona como se describe a continuación.

Amazon ElastiCache es un almacén de clave-valor en memoria que se encuentra entre su aplicación y el almacén de datos (base de datos) al que accede. Siempre que su aplicación solicite datos, primero realizará una solicitud a la caché de ElastiCache. Si los datos existen en la caché y son actuales, ElastiCache devuelve los datos a su aplicación. Si los datos no existen en la caché o han caducado, la aplicación solicita los datos del almacén de datos. A continuación, el almacén de datos devuelve los datos a la aplicación. A continuación, la aplicación escribe los datos recibidos del almacén en la caché. De esta forma, se puede recuperar más rápidamente la próxima vez que se solicite.

El acierto de caché se produce cuando los datos están en la caché y no han caducado:

La aplicación solicita datos de la caché de.

La caché devuelve los datos a la aplicación.

El error de caché se produce cuando los datos no están en la caché o han caducado:

La aplicación solicita datos de la caché de.

La caché no tiene los datos solicitados, por lo que devuelve un null.

La aplicación solicita y recibe los datos de la base de datos.

La aplicación actualiza la caché con los nuevos datos.

El diagrama siguiente ilustra ambos procesos.

Ventajas y desventajas de la carga diferida

Las ventajas de la carga perezosa son las siguientes

* Solo se almacenan en la caché los datos solicitados.
* Dado que la mayoría de los datos nunca se solicitan, la carga perezosa evita llenar la caché con datos que no se solicitan.
* Los errores de nodo no son fatales para su aplicación.
* Cuando un nodo produce un error y se sustituye por un nuevo nodo vacío, la aplicación sigue funcionando, aunque con mayor latencia. Cuando se realizan solicitudes al nuevo nodo, cada error de caché da como resultado una consulta de la base de datos. Al mismo tiempo, la copia de datos se añade a la caché para que las solicitudes posteriores se recuperen de la caché.

Las desventajas de la carga perezosa son las siguientes:

* Existe una penalización de errores de caché. Cada error de caché da como resultado tres viajes:
  + Solicitud inicial de los datos a la caché
  + Consulta de los datos en la base de datos
  + Escritura de los datos en la caché
  + Estos errores pueden causar un retraso notable en los datos que llegan a la aplicación.
  + Datos obsoletos.

Si los datos se escriben en la caché solo cuando hay un error de caché, los datos de la caché pueden quedar obsoletos. Este resultado se produce porque no hay actualizaciones en la caché cuando se cambian los datos en la base de datos. Para solucionar este problema, puedes utilizar la opción Escritura indirecta y Añadir TTL estrategias.