UCAB UNIVERSIDAD CATÓLICA ANDRÉS BELLO

Facultad de Ingeniería
Escuela de Ingeniería Informática

SYNERGY Vision

Proyecto FBCache

Luis Fuentes

C.I: 26.483.209

Objetivos

Objetivo General

- Desarrollar una librería en NodeJS que implemente un almacenamiento en caché de la información consultada a los sistemas de gestión de datos de Firebase

Objetivos Específicos

1. Diseñar una librería pública para la implementación de una caché de datos consultados sobre los sistemas de gestión de datos de Firebase
2. Diseñar un archivo de configuración donde se describa a qué rutas se le hará guardado en caché y especifique el tiempo de refrescamiento de la información almacenada en caché
3. Desarrollar métodos de consumo de la información almacenada en caché y refrescamiento de la información
4. Implementar un servidor de pruebas que muestre los logs de las consultas

Limitaciones

- Se utilizará NodeJS en su versión 12.18.3 LTS para desarrollar la librería, usando como lenguaje de programación JavaScript en su versión ECMAScript 2016
- El archivo de configuración de la librería tendrá un formato de tipo JSON
- El desarrollo se hará usando un control de versiones con Git, y será almacenado en un repositorio público en GitHub
- El servidor de pruebas será desplegado en Heroku
- La documentación del proyecto se encontrará en un archivo README.md en el repositorio de GitHub

Planificación inicial

	S1	S2	S3	S4	S5	S6
Diseño de la librería						
Diseño del archivo de configuración						
Desarrollo de los métodos de consulta a Firebase						
Desarrollo de métodos de almacenamiento en caché						
Desarrollo de métodos de consulta a la información guardada en caché						
Desarrollo de métodos de refrescamineto de información almacenada en caché						
Implementación del servidor de prueba						
Despliegue del servidor en Heroku						

S1	24/08/2020 - 28/08/2020
S2	31/08/2020 - 04/09/2020
S3	07/09/2020 - 11/09/2020
S4	14/09/2020 - 18/09/2020
S5	21/09/2020 - 25/09/2020
S6	28/09/2020 - 02/10/2020

	Diseño
	Desarrollo
	Despliegue

Semana 1

- Diseño de los métodos de consulta, inserción, actualización y eliminación de la librería a los gestores de datos de Firebase y la memoria caché.
- Diseño inicial del archivo de configuración de la librería
- Elección de las librerías y frameworks a utilizar en la implementación de la librería y el servidor de pruebas
  - Líbrería: firebase-admin, node-cache, moment y uuid
  - Servidor: express, winston, Morgan, fbcache (librería a implementar en la pasantía)

Semana 2

- Diseño del archivo de configuración de la librería
- Diseño e implementación del método de inicialización de la librería

- Firma del método de inicialización:

```
FBCache.init(config, projectURL, credentialType, credential);
```

config es donde se le pasará el archivo de configuración al método de inicialización; *proyectURL* es la ruta al proyecto de Firebase al que nos queramos conectar; *credentialType* es donde se le indica al método cual es la credencial que le vamos a pasar para conectarse al proyecto en Firebase, podemos indicar si será por un archivo de credenciales (“file”), o por un token OAuth (“token”); y *credential* es la credencial que tengamos para conectarnos al proyecto en Firebase

Diseño del archivo de configuración

```
{
    "read_only": true,
    "firestore": [
        {
            "name": "users",
            "refresh": "30 s",
            "read_only": false
        }
    ],
    "realtime": [
        {
            "name": "persons",
            "period": "20 s",
            "start": "2020-10-06 04:30:30 am",
            "read_only": false
        }
    ],
    "max_size": "50 MB"
}
```

Ejemplo del diseño del archivo de configuración de la librería

Semana 3

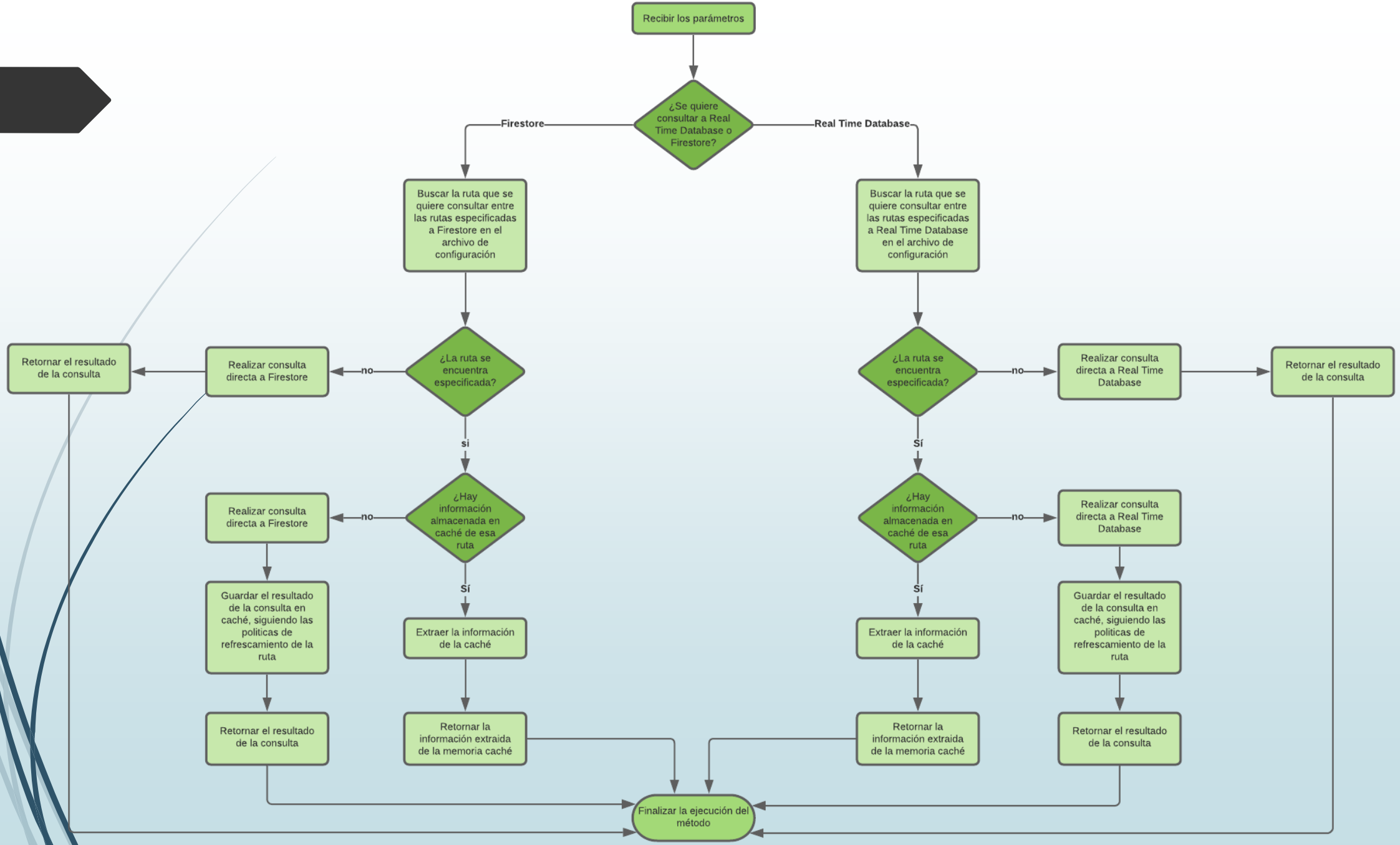
- Implementación del método de consulta de la librería, tomando en cuenta las políticas de refrescamiento de la información en caché
- Implementación de los avances de la librería en un servidor de pruebas
- Despliegue del servidor de pruebas en Heroku

Aunque en la planificación inicial aparezca que la implementación del servidor de pruebas y su despliegue se harían en semana 5 y 6, se decidió adelantar para esta semana para facilitar la presentación de avances al Tutor Empresarial

- Firma del método de consulta:

```
FBCache.get(dbms, route)
```

dbms indica si la consulta se está realizando hacía Real Time Database o hacía Firestore, *route* indica la ruta que se quiera consultar


Recibir los parámetros
¿Se quiere consultar a Real Time Database o Firestore?
Firestore
Real Time Database
Buscar la ruta que se quiere consultar entre las rutas especificadas a Firestore en el archivo de configuración
Buscar la ruta que se quiere consultar entre las rutas especificadas a Real Time Database en el archivo de configuración
¿La ruta se encuentra especificada?
no
Realizar consulta directa a Firestore
Retornar el resultado de la consulta
si
¿Hay información almacenada en caché de esa ruta
no
Realizar consulta directa a Firestore
Guardar el resultado de la consulta en caché, siguiendo las politicas de refrescamiento de la ruta
Retornar el resultado de la consulta
Sí
Extraer la información de la caché
Retornar la información extraida de la memoria caché
¿La ruta se encuentra especificada?
no
Realizar consulta directa a Real Time Database
Retornar el resultado de la consulta
Sí
¿Hay información almacenada en caché de esa ruta
no
Realizar consulta directa a Real Time Database
Guardar el resultado de la consulta en caché, siguiendo las politicas de refrescamiento de la ruta
Retornar el resultado de la consulta
Sí
Extraer la información de la caché
Retornar la información extraida de la memoria caché
Finalizar la ejecución del método

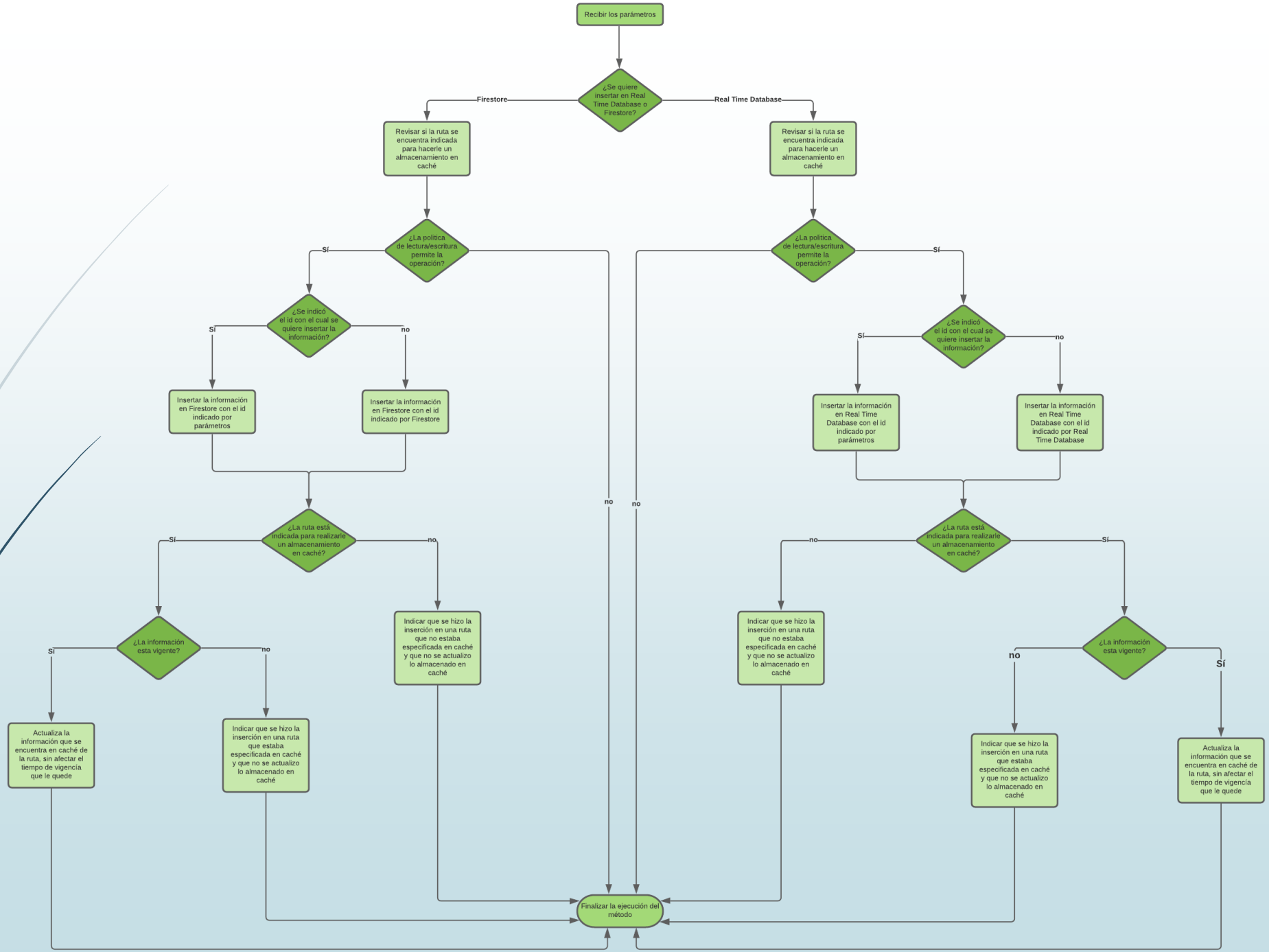
Semana 4

- Implementación del método de inserción de la librería
- Implementación del método que se encargue de actualizar lo almacenado en caché cuando se hace una inserción en una ruta especificada en el archivo de configuración

- Firma del método de inserción:

```
FBCache.insert(dbms,route,data,id)
```

Al igual que en el método anterior, *dbms* indica si la inserción se quiere realizar en Real Time Database o en Firestore y *route* la ruta a la que se le quiere hacer la inserción; *data* es donde se indica la información que se quiere guardar, y *id* es un atributo opcional que nos permite asignarle nosotros mismos el identificador con el que se guardará la información, en caso de no indicarse, Real Time Database o Firestore asignará automáticamente un identificador único.


Recibir los parámetros
¿Se quiere insertar en Real Time Database o Firestore?
Firestore
Real Time Database
Revisar si la ruta se encuentra indicada para hacerle un almacenamiento en caché
Revisar si la ruta se encuentra indicada para hacerle un almacenamiento en caché
¿La política de lectura/escritura permite la operación?
¿La política de lectura/escritura permite la operación?
Sí
Sí
no
no
¿Se indicó el id con el cual se quiere insertar la información?
¿Se indicó el id con el cual se quiere insertar la información?
Sí
no
Sí
no
Insertar la información en Firestore con el id indicado por parámetros
Insertar la información en Firestore con el id indicado por Firestore
Insertar la información en Real Time Database con el id indicado por parámetros
Insertar la información en Real Time Database con el id indicado por Real Time Database
¿La ruta está indicada para realizarle un almacenamiento en caché?
¿La ruta está indicada para realizarle un almacenamiento en caché?
Sí
no
no
Sí
¿La información esta vigente?
¿La información esta vigente?
Sí
no
no
Sí
Indicar que se hizo la inserción en una ruta que no estaba especificada en caché y que no se actualizo lo almacenado en caché
Indicar que se hizo la inserción en una ruta que no estaba especificada en caché y que no se actualizo lo almacenado en caché
Actualiza la información que se encuentra en caché de la ruta, sin afectar el tiempo de vigencia que le quede
Indicar que se hizo la inserción en una ruta que estaba especificada en caché y que no se actualizo lo almacenado en caché
Indicar que se hizo la inserción en una ruta que estaba especificada en caché y que no se actualizo lo almacenado en caché
Actualiza la información que se encuentra en caché de la ruta, sin afectar el tiempo de vigencia que le quede
Finalizar la ejecución del método

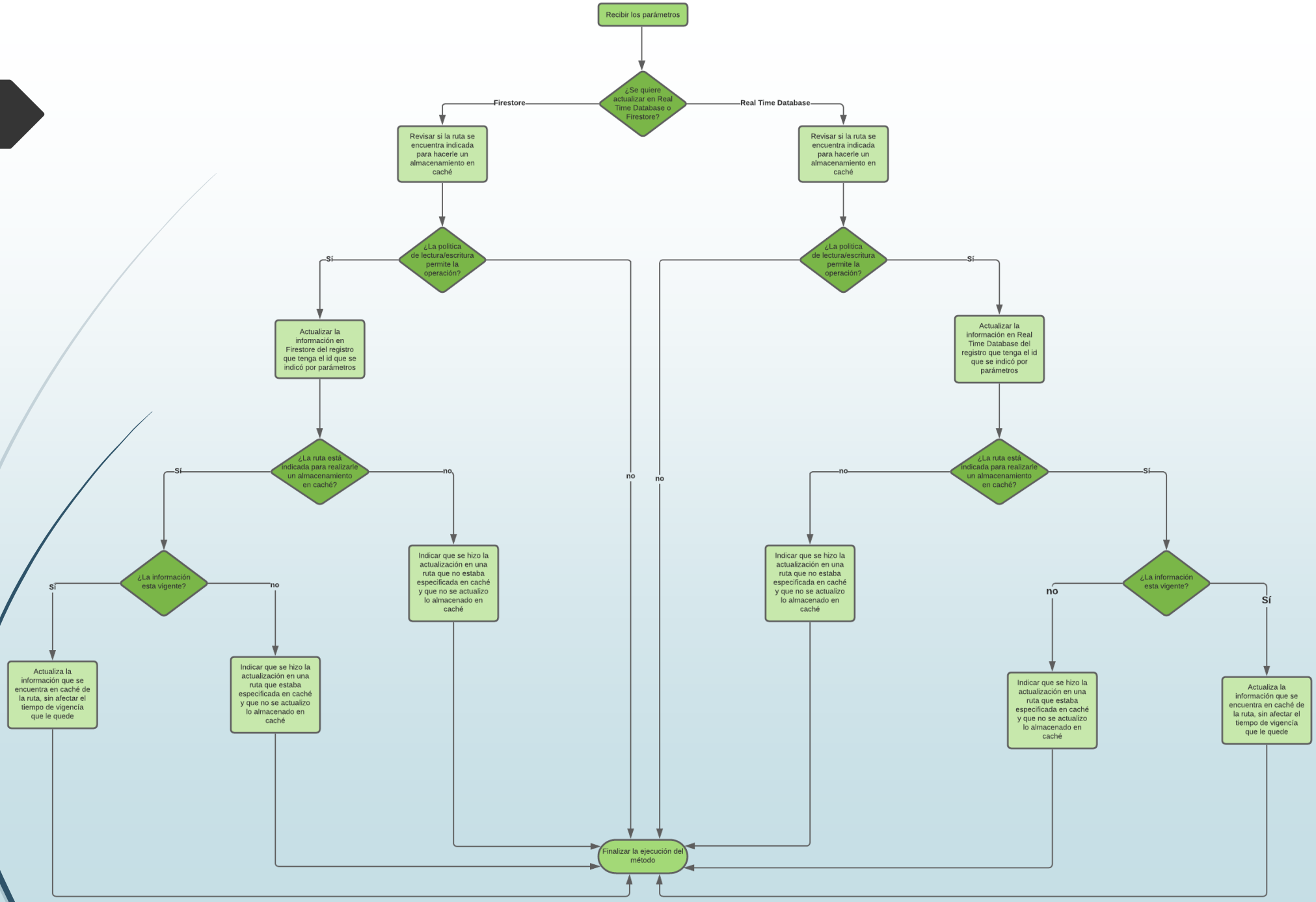
Semana 5

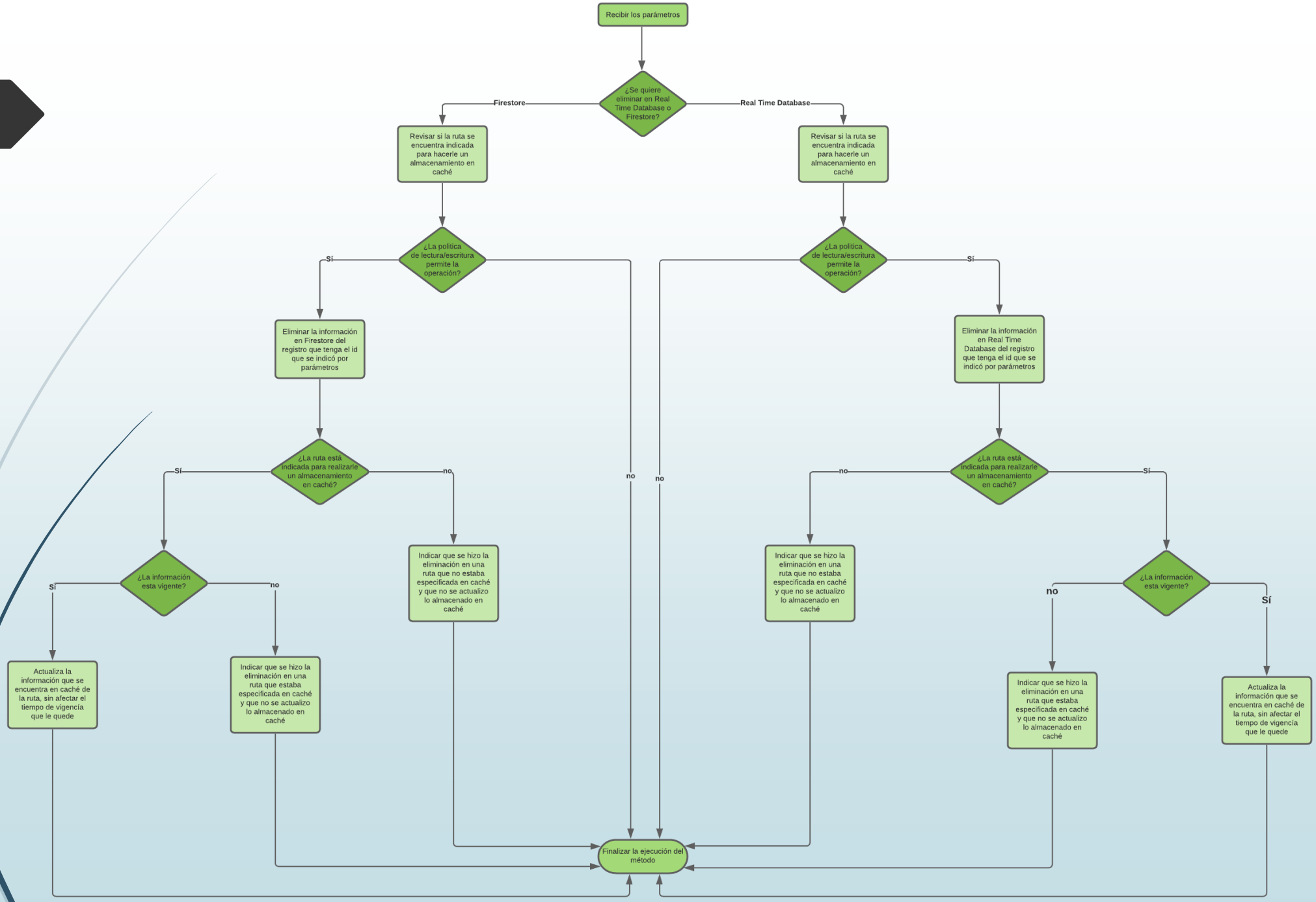
- Implementación del método de actualización y eliminación de la librería
- Implementación de los métodos que permitan actualizar lo almacenado en caché con los cambios de las actualizaciones o eliminaciones
- Mejora en la implementación de lo logs del servidor
- Implementa una nueva firma en la librería más parecida a la firma de firebase-admin que facilite implementar a FBCache en proyectos que usen firebase-admin

- Firma de los métodos de actualización y eliminación

FBCache.update(dbms,route,data,id)
FBCache.delete(dbms,route,id)

dbms y route cumplen la misma función que en los métodos de las semanas anteriores; en update, *data* es la información que se quiere actualizar y *id* el identificador donde se quieren hacer dichos cambios; en el caso de delete, el *id* es el identificador que se quiere eliminar


Recibir los parámetros
¿Se quiere actualizar en Real Time Database o Firestore?
Firestore
Real Time Database
Revisar si la ruta se encuentra indicada para hacerle un almacenamiento en caché
Revisar si la ruta se encuentra indicada para hacerle un almacenamiento en caché
¿La política de lectura/escritura permite la operación?
¿La política de lectura/escritura permite la operación?
Sí
Sí
no
no
Actualizar la información en Firestore del registro que tenga el id que se indicó por parámetros
Actualizar la información en Real Time Database del registro que tenga el id que se indicó por parámetros
¿La ruta está indicada para realizarle un almacenamiento en caché?
¿La ruta está indicada para realizarle un almacenamiento en caché?
Sí
no
no
Sí
Indicar que se hizo la actualización en una ruta que no estaba especificada en caché y que no se actualizo lo almacenado en caché
Indicar que se hizo la actualización en una ruta que no estaba especificada en caché y que no se actualizo lo almacenado en caché
¿La información esta vigente?
¿La información esta vigente?
Sí
no
no
Sí
Actualiza la información que se encuentra en caché de la ruta, sin afectar el tiempo de vigencía que le quede
Indicar que se hizo la actualización en una ruta que estaba especificada en caché y que no se actualizo lo almacenado en caché
Indicar que se hizo la actualización en una ruta que estaba especificada en caché y que no se actualizo lo almacenado en caché
Actualiza la información que se encuentra en caché de la ruta, sin afectar el tiempo de vigencía que le quede
Finalizar la ejecución del método


Recibir los parámetros
¿Se quiere eliminar en Real Time Database o Firestore?
Firestore
Real Time Database
Revisar si la ruta se encuentra indicada para hacerle un almacenamiento en caché
Revisar si la ruta se encuentra indicada para hacerle un almacenamiento en caché
¿La política de lectura/escritura permite la operación?
¿La política de lectura/escritura permite la operación?
Sí
Sí
no
no
Eliminar la información en Firestore del registro que tenga el id que se indicó por parámetros
Eliminar la información en Real Time Database del registro que tenga el id que se indicó por parámetros
¿La ruta está indicada para realizarle un almacenamiento en caché?
¿La ruta está indicada para realizarle un almacenamiento en caché?
Sí
no
no
Sí
¿La información esta vigente?
¿La información esta vigente?
Sí
no
no
Sí
Indicar que se hizo la eliminación en una ruta que no estaba especificada en caché y que no se actualizo lo almacenado en caché
Indicar que se hizo la eliminación en una ruta que no estaba especificada en caché y que no se actualizo lo almacenado en caché
Actualiza la información que se encuentra en caché de la ruta, sin afectar el tiempo de vigencía que le quede
Indicar que se hizo la eliminación en una ruta que estaba especificada en caché y que no se actualizo lo almacenado en caché
Indicar que se hizo la eliminación en una ruta que estaba especificada en caché y que no se actualizo lo almacenado en caché
Actualiza la información que se encuentra en caché de la ruta, sin afectar el tiempo de vigencía que le quede
Finalizar la ejecución del método

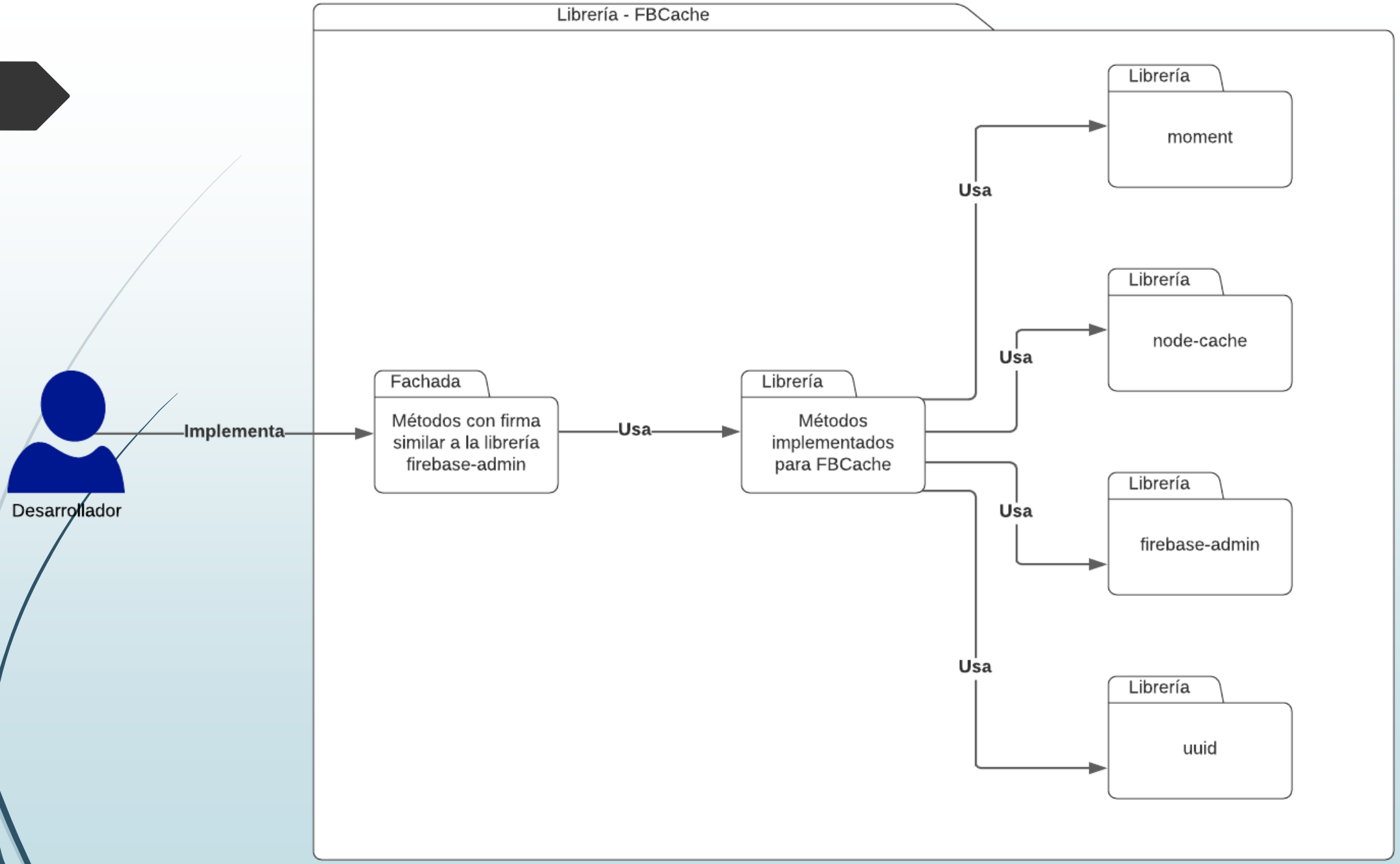

Librería - FBCache
Librería
moment
Usa
Librería
node-cache
Usa
Fachada
Métodos con firma similar a la librería firebase-admin
Usa
Librería
Métodos implementados para FBCache
Usa
Librería
firebase-admin
Usa
Librería
uuid
Implementa
Desarrollador

Tabla de comparación entre la nueva firma de FBCache y la antigua

	Firma antigua	Firma nueva
Inicialización	FBCache.init(config,projectURL,credentialType,credential)	initFBCache(config,projectURL,credentialType,credential)
Consulta	FBCache.get(dbms,route)	FBCache.database().ref(route).once()
		FBCache.firestore().collection(route).get()
Inserción	FBCache.insert(dbms,route,data,id)	FBCache.database().ref(route).child(id).set(data)
		FBCache.database().ref().child(route).push(data)
		FBCache.firestore().collection(route).doc(id).set(data)
		FBCache.firestore().collection(route).add(data)
Actualización	FBCache.update(dbms,route,data,id)	FBCache.database().ref(route).child(id).update(data)
		FBCache.firestore().collection(route).doc(id).update(data)
Eliminación	FBCache.delete(dbms,route,id)	FBCache.database().ref(route).child(id).remove()
		FBCache.firestore().collection(route).doc(id).delete()

Semana 6

- Implementación de pruebas E2E
- Realizar la documentación de la librería

Las pruebas E2E se realizaron de 2 formas, usando la librería desde el servidor de prueba e implementando pruebas con Jest, las pruebas realizadas con Jest dejó como resultado el siguiente coverage:

All files lib

93.87% Statements 337/359 **85.21%** Branches 219/257 **100%** Functions 49/49 **93.82%** Lines 334/356

Press *n* or *j* to go to the next uncovered block, *b*, *p* or *k* for the previous block.

File	Statements	Branches	Functions	Lines
fbcache.js	92.76% 282/304	81.09% 163/201	100% 33/33	92.69% 279/301
index.js	100% 55/55	100% 56/56	100% 16/16	100% 55/55

En pruebas realizadas junto con el Tutor Empresarial se consiguieron algunos detalles que se pasaron por alto, por lo que se paso a corregir dichos detalles en la brevedad posible

Link a los repositorios

- Repositorio de la librería FBCache: https://github.com/synergyvision/fbcache
- Repositorio de prueba de la librería FBCache: https://github.com/synergyvision/fbcacheserver