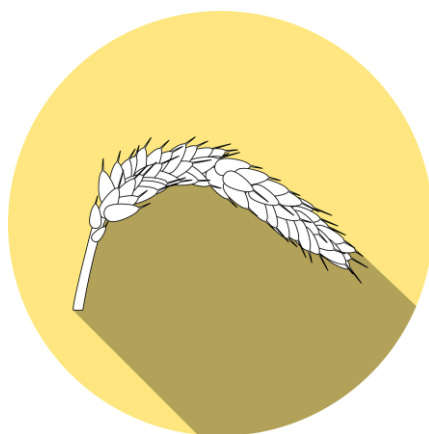


14-6-2019

# MÓDULO DE PROYECTO

C.F.G.S.

DESARROLLO DE APLICACIONES  
MULTIPLATAFORMA



## CELIACOSAS

AUTOR: JUAN DANIEL CEPEDA

TITULO DEL PROYECTO: CELIACOSAS	
AUTOR: J. DANIEL CEPEDA MARÍN	FECHA: 14/06/2019
TUTOR/A: ARANTZAZU FIALLEGA	
TITULACIÓN: C.F.G.S. DESARROLLO DE APLICACIONES MULTIPLATAFORMA	
PALABRAS CLAVE: <ul style="list-style-type: none"><li>• APP, API, Android, Alimento, Gluten, Celiaco</li></ul>	

## Tabla de contenido

1.INTRODUCCIÓN .....	4
INTRODUCCIÓN A LA MEMORIA.....	4
DESCRIPCIÓN .....	4
BENEFICIOS.....	5
MOTIVACIONES PERSONALES .....	5
2.ESTUDIO DE VIABILIDAD.....	5
Objetivos .....	5
ESTUDIO DE LA SITUACIÓN ACTUAL .....	6
Contexto.....	6
Descripción Física .....	7
REQUISITOS DEL SISTEMA .....	8
Requisitos.....	8
Restricciones del sistema.....	8
Catalogación y priorización de los requisitos .....	8
Alternativas y Selección de la solución.....	9
Planificación del Proyecto .....	11
Recursos de Hardware .....	11
Recursos de Software.....	11
Planificación temporal del proyecto.....	12
PRESUPUESTO .....	14
Estimación de costes materiales .....	14
Estimación de costes de software .....	15
Estimación costes personales.....	15
3.ANALISIS.....	15
Requisitos funcionales de usuarios .....	16
Requisitos no funcionales.....	17
Diagramas de casos de uso.....	17
Diagrama de Datos .....	25
Diagrama de Clases.....	27
Diagrama de Actividad.....	29
Menús de Navegación .....	30
EJECUCIÓN DEL PROYECTO .....	31
INTRODUCCIÓN .....	31
EJECUCIÓN DE LA APLICACIÓN .....	32
INICIO .....	32
NAVEGACIÓN.....	33

RECYCLER VIEW .....	34
Alimentos Confirmados.....	35
Alimentos Generales.....	37
Mis Alimentos.....	37
Favoritos.....	38
DISEÑO .....	38
Introducción.....	38
Selección del entorno de desarrollo.....	39
Selección de la base de datos .....	40
Arquitectura de la aplicación .....	40
IMPLEMENTACIÓN.....	43
PRUEBAS.....	44
Pruebas realizadas .....	44
Conclusiones .....	46
4.CONCLUSIONES .....	47
Conclusiones finales .....	47
Posibles ampliaciones y mejoras .....	47
Opinión personal.....	48
BIBLIOGRAFÍA.....	48
GLOSARIO .....	48

# 1. INTRODUCCIÓN

## INTRODUCCIÓN A LA MEMORIA

La intolerancia al gluten o enfermedad celíaca es una intolerancia permanente al gluten. Se estima que un 1% de la población padece esta enfermedad, de hecho, se estima que entre un 80 y un 85% de los celíacos están aún sin diagnóstico.

La dieta que deben seguir las personas que padezcan celiaquía parece fácil, en el fondo simplemente deben evitar los alimentos que contengan gluten. Pero la cosa se complica cuando vemos que, el gluten no está solamente en 4 o 5 cereales, sino que es utilizado por la industria en una gran variedad de productos. Los cuales muchas veces no son etiquetados correctamente por sus fabricantes, muchas veces por no ser el producto español o europeo no haciendo necesaria su correcta indicación.

## DESCRIPCIÓN

La aplicación de Celiacosas se presenta como ayuda a identificar de forma segura si un alimento tiene o no gluten sin la necesidad de poner en riesgo su salud o tener que pasar por la molestia de ponerse en contacto con el fabricante o distribuidor. Desde la aplicación el usuario podrá registrarse y añadir alimentos de los que tenga dudas a una lista común donde otros usuarios podrán votarla, los alimentos que tienen más votos suben en la lista y los más votados serán seleccionados para ser investigados, de confirmarse el que no tengan gluten se añadirán a una lista aparte para consulta de los usuarios.

## BENEFICIOS

Los beneficios que presenta la aplicación al usuario son:

- Accesibilidad a una lista de alimentos sin gluten los cuales han sido confirmados por sus fabricantes.
- Rapidez a la hora de añadir alimentos de dudosa situación.
- La comodidad de poder consultar alimentos y avances.

Además del beneficio de la satisfacción personal que me da el saber que estoy desarrollando una herramienta que va a ayudar a la comunidad de celíacos.

## MOTIVACIONES PERSONALES

Como celíaco en mi día a día es común ir a un supermercado y encontrarme productos sin etiquetar, en zonas de productos sin gluten que no están señalados como sin gluten o que directamente deberían ser sin gluten, pero se encuentran en otros idiomas haciendo imposible es comprender lo que pone en el empaquetado o leer los ingredientes. Con eso en mente me he propuesto el crear un sistema por el cual de manera rápida y sencilla un usuario pueda registrar un alimento del cual no esté seguro, o buscarlo tanto en una lista que se sepa que es 100% segura o dentro de los alimentos que ya han sido publicados buscar el alimento deseado.

## 2. ESTUDIO DE VIABILIDAD

### Objetivos

El objetivo principal del proyecto es que con el uso de los conocimientos obtenidos durante el grado y la fase de investigación del proyecto realice una aplicación de complejidad media para el sistema operativo Android sobre alimentos de los cuales se duda si tienen o no gluten.

Respecto a lo anterior, la aplicación que permitirá:

- Registro de usuarios.
- Login de usuarios.
- Consulta de lista de alimentos sin gluten.
- Introducción de alimentos dudosos nuevos al "foro".
- Modificar alimentos introducidos.
- Eliminar alimentos introducidos.
- Consultar alimentos favoritos del usuario.
- Votar alimentos.
- Filtrar/Buscar entre los alimentos existentes.

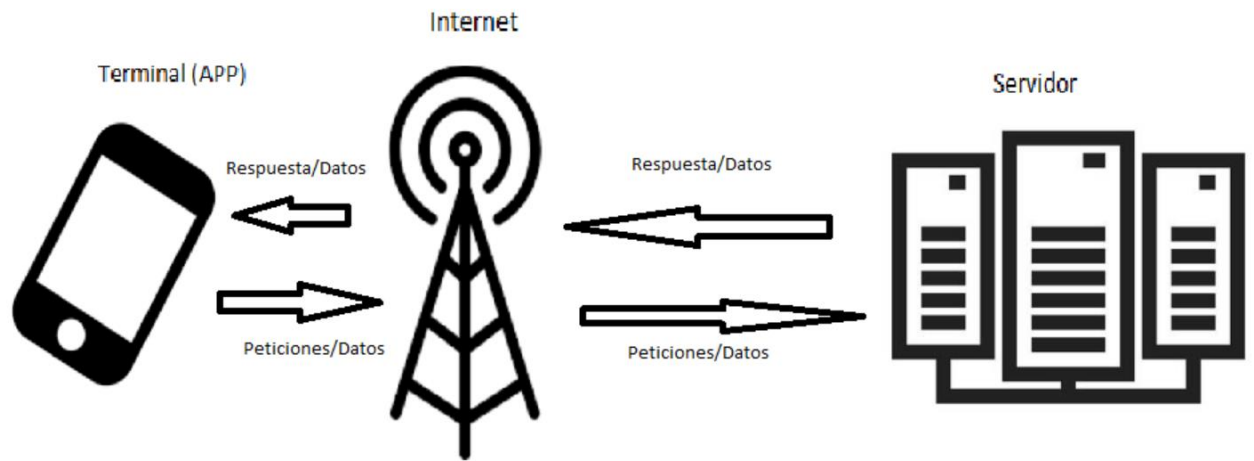
A su vez que la base de datos de la aplicación sea hosteada en un servidor junto a una API Rest en PHP que será a la cual la aplicación hará las peticiones.

## ESTUDIO DE LA SITUACIÓN ACTUAL

### Contexto

Actualmente el mercado de aplicaciones móviles si bien se encuentra saturado con aplicaciones sobre alimentación, pero raro es encontrar una centrada en la alimentación sin gluten desde un punto de vista celíaco en vez de dietético.

## Descripción Física



Físicamente es prácticamente idéntica a la mayoría de entornos de aplicaciones móviles. Los dispositivos se conectan a internet a través de redes 3G/4G/5G o WiFi.

El dispositivo hace peticiones al servidor, ya sea solicitando datos o enviándolos y este envía respuestas al dispositivo tras operar con la base de datos que contiene, ya haya sido bien al operar con datos enviados por el dispositivo o al enviar datos al dispositivo.

## REQUISITOS DEL SISTEMA

### Requisitos

Para el correcto funcionamiento de la aplicación será necesario el uso de:

- Un dispositivo móvil con sistema operativo Android 7 o superior,
- Conexión a internet estable ya sea a través de WiFi o datos móviles,
- 6 MB de almacenamiento ya sea en la memoria interna del dispositivo o en una tarjeta micro-ssd.

### Restricciones del sistema

Necesita de conexión constante y estable a internet para poder operar, solo funciona en dispositivos que usen Android 7 o superior.

### Catalogación y priorización de los requisitos

Los requisitos de la aplicación se pueden dividir en principales (necesarios para el funcionamiento mínimo de la aplicación), secundarios (establecen el funcionamiento óptimo y deseado de la aplicación) y extras (añadidos como refuerzo o añadidos, pero cuya funcionalidad no compromete ni afecta la del resto del sistema).

Los requisitos de principales de la aplicación son:

- Registro de usuarios



- Lista de alimentos sin gluten
- Lista central ("Forum")
- Añadir Alimento
- Eliminar Alimento
- Ver detalle de Alimentos

Los requisitos secundarios de la aplicación son:

- Modificar Alimentos
- Añadir Alimentos a Favoritos
- Quitar Alimentos de Favoritos
- Buscar Alimentos

Requisitos extras de la aplicación:

- Cerrar sesión
- Buscar supermercados más cercanos

## Alternativas y Selección de la solución

Alternativa 1: "Qué puedo comer" (App/Web)

La aplicación ofrece una amplia lista de alimentos sin gluten, más la posibilidad de escanear el código de barras de un alimento y te indique con ello si es o no apto para el consumo.

El usuario una vez registrado puede buscar en una extensa base de datos sobre alimentos, los cuales están clasificados por varias alergias e intolerancias.



#### Alternativa 2: "Super celíaca" (Web)

Permite la descarga de recetas sin gluten y tiene una gama de alimentos sin gluten. Esta página además ofrece un amplio catálogo de alimentos sin gluten a comprar en el acto y guías y buscadores para establecimientos los cuales ofrezcan alimentos sin gluten.



#### Conclusiones:

Si bien el mercado ya está plagado de aplicaciones que tienen funcionalidades básicas tales como listas de alimentos filtrados por intolerancias o alergias.

Mi aplicación lo que pretende es introducir el factor de red social, la posibilidad de que los usuarios sean los que vayan manualmente introduciendo los alimentos, mientras que el resto de usuarios votan por él. Además, la gran mayoría de estas alternativas no suelen contener todos los alimentos que te puedes llegar a encontrar en un supermercado.

## Planificación del Proyecto

### Recursos de Hardware

RECURSO	FUNCIÓN
Portátil Personal	Diseño y programación de la app. Realización del documento. Formación/Investigación
Smatphone personal	Testing de la app durante el desarrollo

### Recursos de Software

Android Studio	Diseño y programación de la app
Adobe XD	Software de diseño de interfaces
Word	Editor de texto
Firebase	Es una plataforma ubicada en la nube, integrada con Google Cloud Platform que usa un conjunto de herramientas para la creación y sincronización de proyectos que serán dotados de alta calidad, haciendo posible el crecimiento del número de usuarios y dando resultado también a la obtención de una mayor monetización.
Hostinger	Hostinger International, Ltd. es un proveedor de alojamiento web de empleados y un

	registrador de dominios de Internet.
Xampp	XAMPP es un paquete de software libre, que consiste principalmente en el sistema de gestión de bases de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script PHP y Perl.
phpMyAdmin	phpMyAdmin es una herramienta escrita en PHP con la intención de manejar la administración de MySQL a través de páginas web, utilizando un navegador web
Visual Studio Code	Visual Studio Code es un editor de código fuente desarrollado por Microsoft para Windows, Linux y macOS, el cual se ha usado para realizar la API de la cual se toman los datos para la app.
MySQL	MySQL es un sistema de gestión de bases de datos relacional.
Slim	Slim es un microframework de PHP que le ayuda a escribir rápidamente aplicaciones web y API simples pero potentes.

## Planificación temporal del proyecto

1-3 semanas de investigación/estudio

MATERIA	TIEMPO
Android Fragments	Primeras semanas
Firebase Auth	Mínimo 1 semana
Firebase Database	Mínimo 1 semana
Ionic	Dos semanas
Adobe XD	Varios días durante el desarrollo

1 semana de diseño

DISEÑO	TIEMPO
Menús	1-2 días
Listas	1 días
Logos	1 día
Gama de colores	1-2 días
Elementos de la lista	1 día

6 semanas de desarrollo

OBJETIVO	TIEMPO
Lista principal	5 días
Lista favoritos	4 días
Lista mis alimentos	4 días
Menu principal	1 día
Opciones	1 día
Autenticación Firebase	4-5 días
Base de datos	4 días
API REST	3-4 días
Otra programación	5-9 días
Firebase Database	1-3 días
Diseño	5-8 días

2 semanas de documentación

MATERIA	TIEMPO
Documento de entrega	7-9 días
Manual de Usuario	1-2 días
Documentación del código	1-2 días
Otros Página Wordpress	1-2 días

## PRESUPUESTO

Estimación de costes materiales

OBJETO	PRECIO
Ordenador personal I5-8250u 8GB RAM 1000 GB	500€
Hosting servicio Hosting Hostinger	16.15€
Cableado vario	10€
Dispositivos de almacenamiento Pendrives x2 Tarjeta micro-sd	20€

**HOSTING PREMIUM**

1 Mes  
**11,95 €**  
/mes

3 Meses  
**4,45 €**  
/mes

12 Meses  
**3,75 €**  
/mes

24 Meses  
**2,95 €**  
/mes

48 Meses  
**2,15 €**  
/mes

Subtotal: **13,35 €**  
Ahorras **22,50 €**

**Resumen del Pedido**

Hosting Premium © **13,35 €**

↓ Activación del Certificado SSL **0,00 €**

¿Tienes un código de cupón? [Clic aquí](#)

Impuestos y Pagos ⓘ **2,80 €**

Total ~~62,64 €~~ **16,15 €**

### Estimación de costes de software

SOFTWARE	PRECIO
Android Studio	0€
Visual Studio Code	0€
Xampp	0€
Word	5€
Adobe XD	0€
FileZile Client	0€

### Estimación costes personales

TAREA	TIEMPO
Investigación y formación	25-35 horas
Diseño	10 horas
Desarrollo interfaces	10-15 horas
Desarrollo API	10 horas
Desarrollo de la lógica de la aplicación	40-50 horas+
Desarrollo base de datos	5-10 horas
Desarrollo pruebas	2-5 horas
Desarrollo documentación	20 horas+

## 3. ANALISIS

## Requisitos funcionales de usuarios

Los requerimientos funcionales de un sistema, son aquellos que describen cualquier actividad que este deba realizar, en otras palabras, el comportamiento o función particular de un sistema o software cuando se cumplen ciertas condiciones.

Por lo general, estos deben incluir funciones desempeñadas por pantallas específicas, descripciones de los flujos de trabajo a ser desempeñados por el sistema y otros requerimientos de negocio, cumplimiento, seguridad u otra índole.

Los requisitos funcionales correspondientes al cumplimiento y negocio que realiza mi aplicación son:

- Visualizar Alimentos añadidos por los usuarios
- Visualizar alimentos confirmados como sin gluten.
- Visualizar Alimentos seleccionados como favoritos.
- Buscar por palabras claves los alimentos.
- Seleccionar alimentos favoritos.
- Deseleccionar alimentos favoritos.
- Votar Alimentos.
- Añadir Alimento.
- Modificar Alimento añadido.
- Eliminar Alimento.



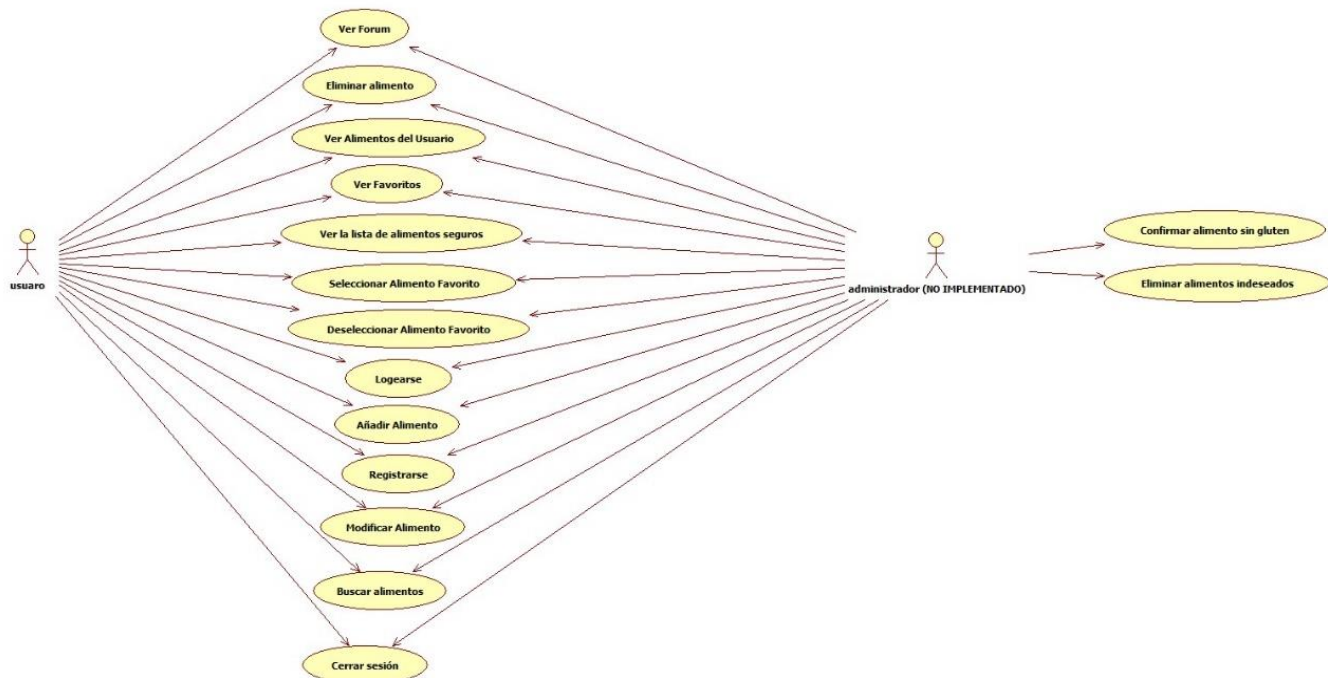
## Requisitos no funcionales

Los requisitos no funcionales representan características generales y restricciones de la aplicación o sistema que se esté desarrollando.

Suelen presentar dificultades en su definición dado que su conformidad o no conformidad podría ser sujeto de libre interpretación, por lo cual es recomendable acompañar su definición con criterios de aceptación que se puedan medir.

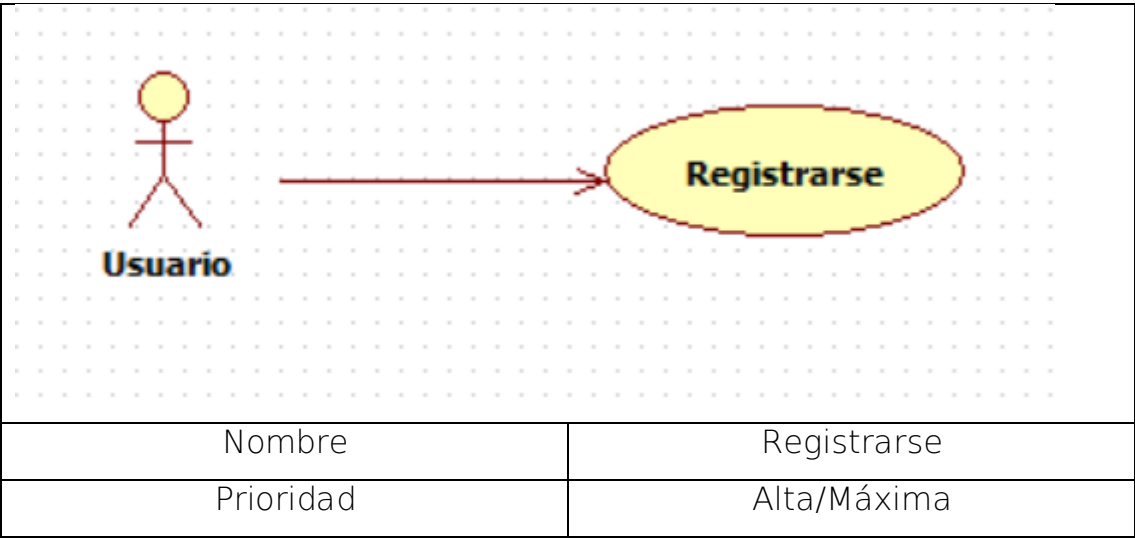
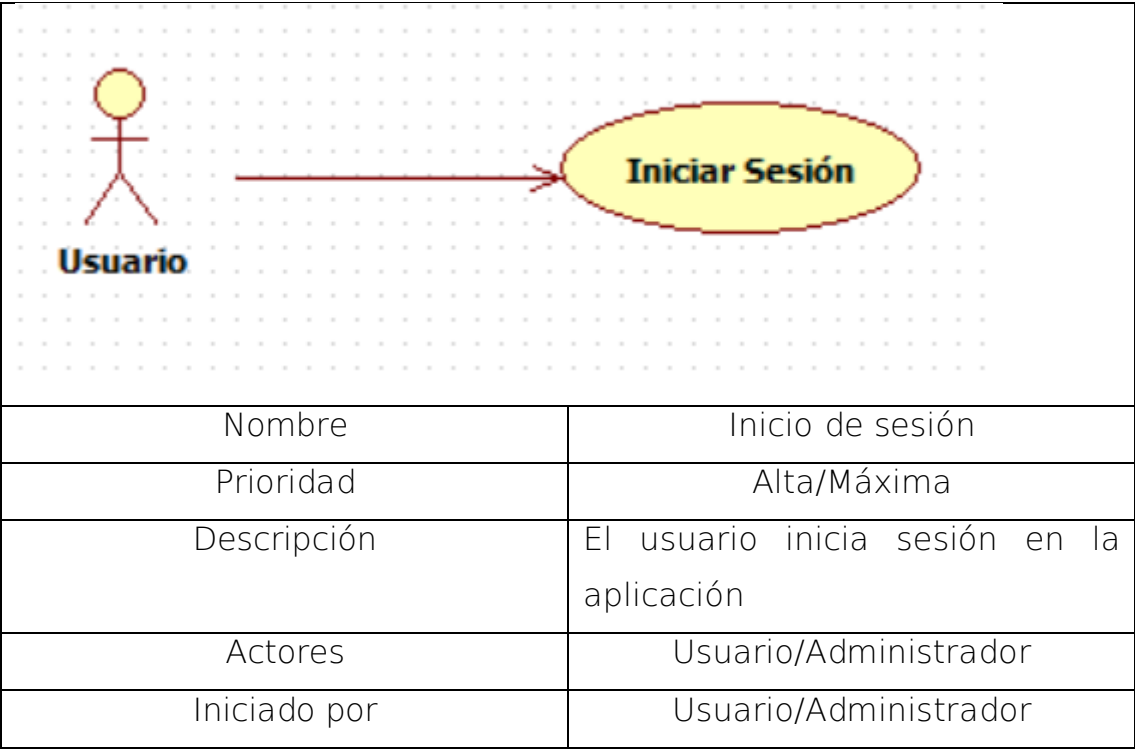
En relación con el requisito de la disponibilidad la aplicación cuenta con varias dependencias, siendo las dos principales conjuntas en el hecho de que ambas necesitan de conexión a internet, la primera es el login/registro con Firebase y la segunda es la misma base de datos y servicio REST, los cuales dependen del correcto funcionamiento del hosting donde estén alojados.

## Diagramas de casos de uso



Hay dos actores, el usuario normal y administrador (no implementado). Si bien ambos tienen los mismos casos de uso, el administrador tiene dos extras,

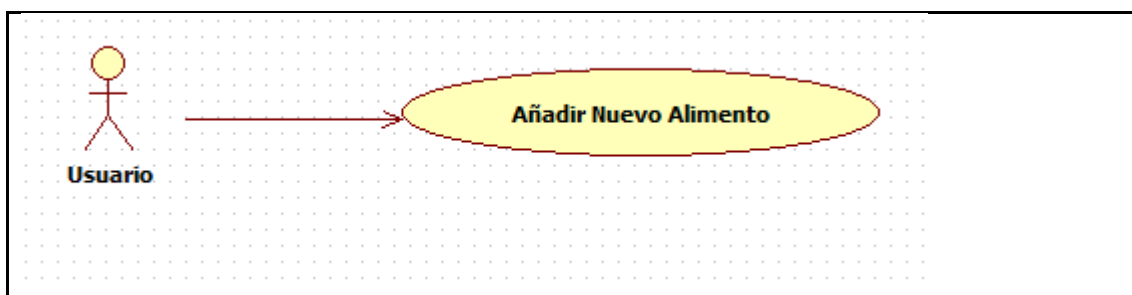
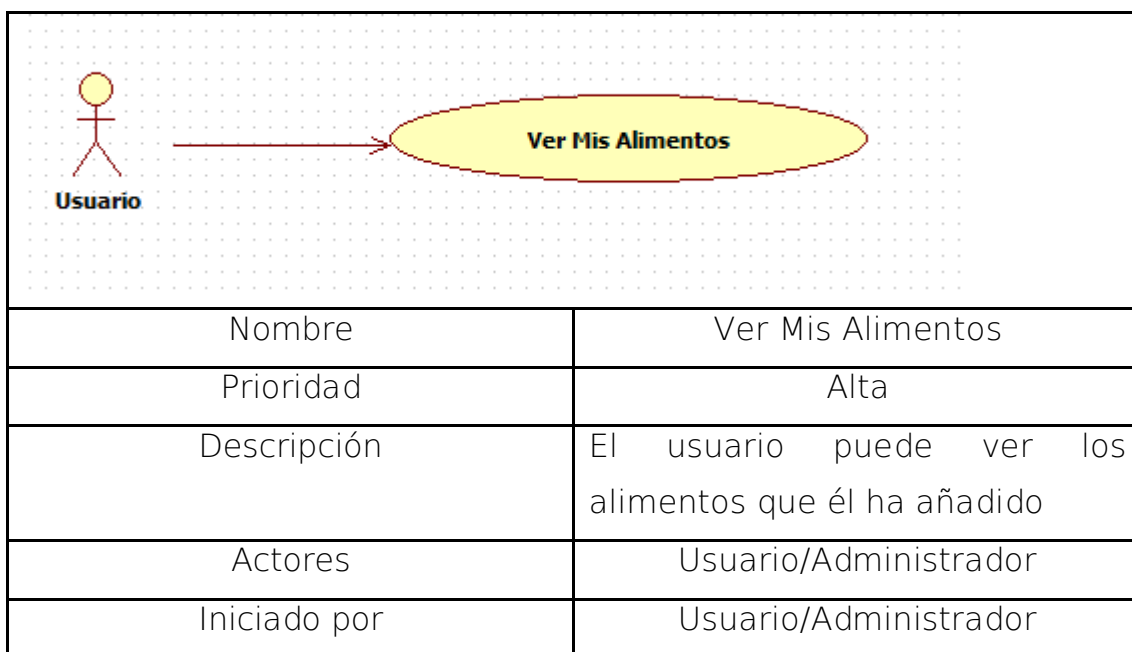
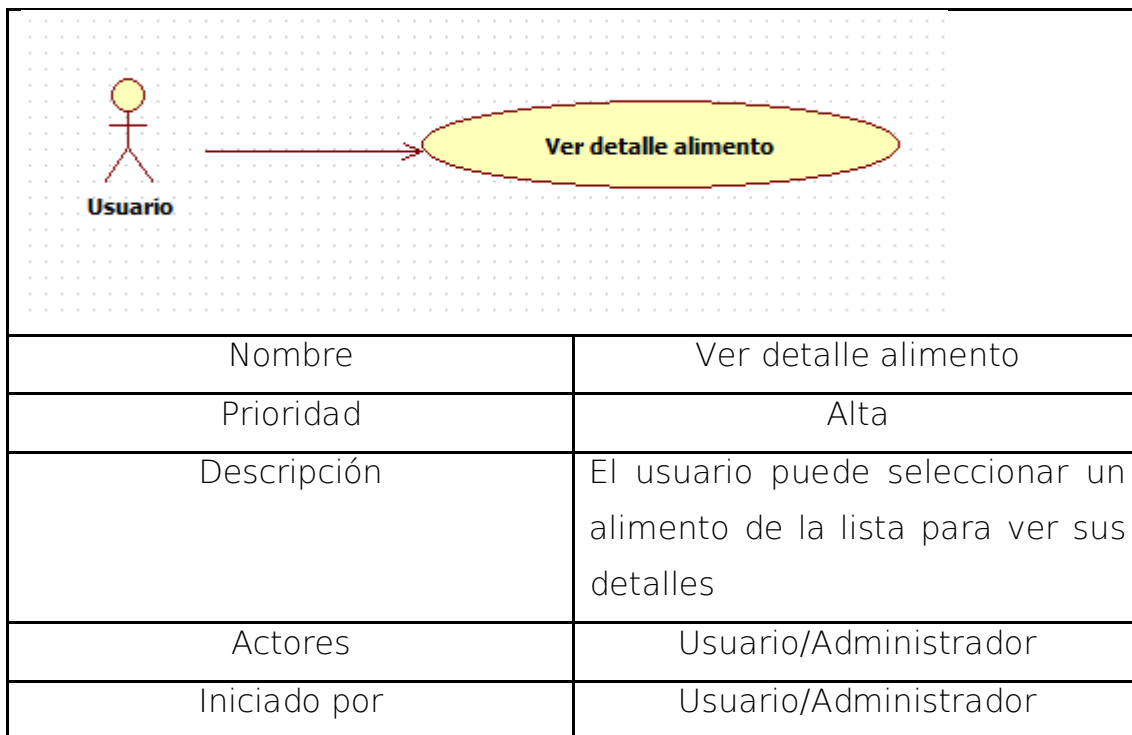
La diferencia entre estos dos es que los administrados son usuarios los cuales han sido elegidos por su experiencia y confianza para filtrar de entre los alimentos añadidos aquellos que son sin gluten o que son nocivos. Esto lo llevaran a cabo a través de una investigación propia.



Descripción	El usuario se registra como usuario de la aplicación
Actores	Usuario
Iniciado por	Usuario

<p>The diagram shows a stick figure actor labeled 'Usuario' on the left. A red arrow points from the actor to a yellow oval use case labeled 'Ver Alimentos Comprobados'.</p>	
Nombre	Ver Alimentos Comprobados
Prioridad	Alta/Máxima
Descripción	El usuario puede ver la lista de alimentos los cuales han sido confirmados como sin gluten
Actores	Usuario/Administrador
Iniciado por	Usuario/Administrador

<p>The diagram shows a stick figure actor labeled 'Usuario' on the left. A red arrow points from the actor to a yellow oval use case labeled 'Ver el Forum'.</p>	
Nombre	Ver el Forum
Prioridad	Alta
Descripción	El usuario se registra como usuario de la aplicación
Actores	Usuario
Iniciado por	Usuario

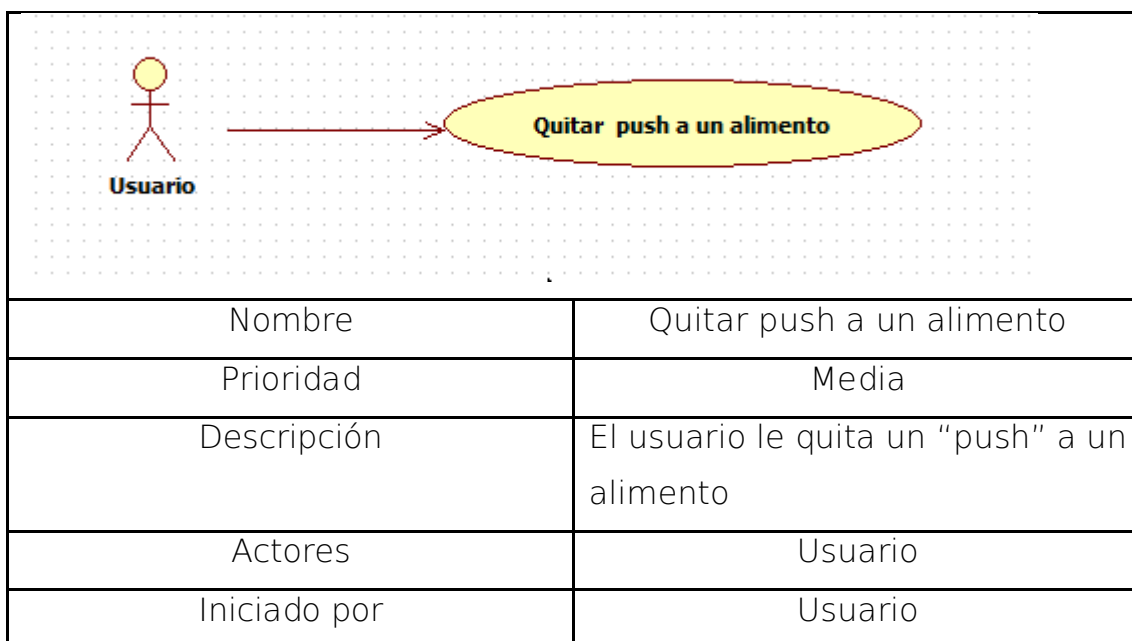
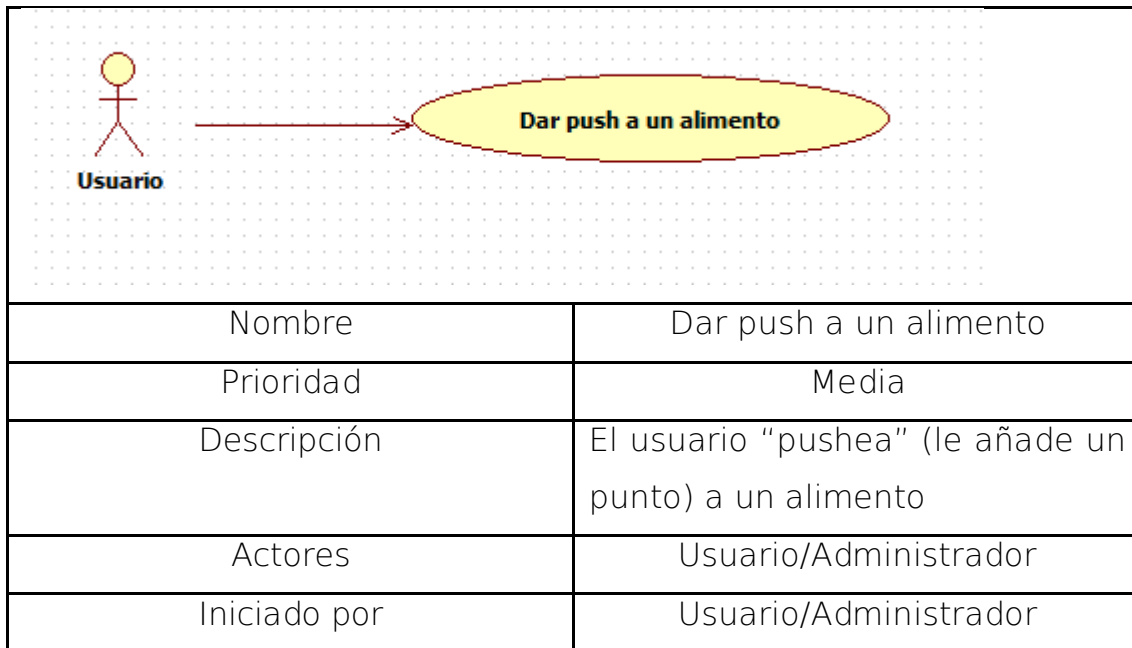


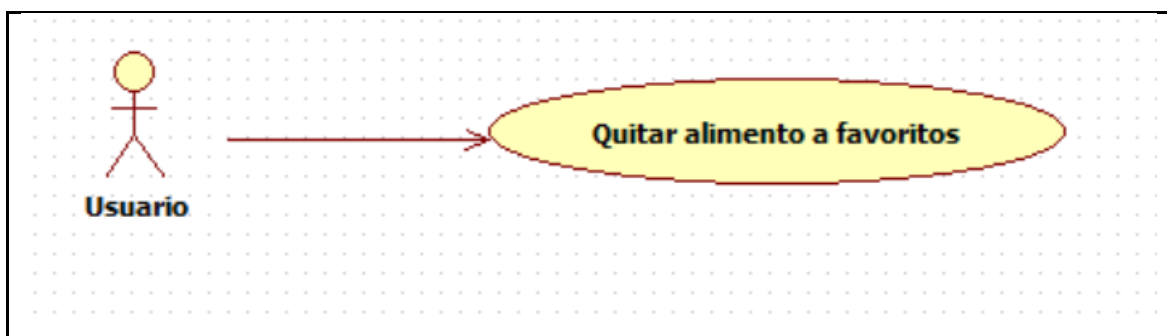
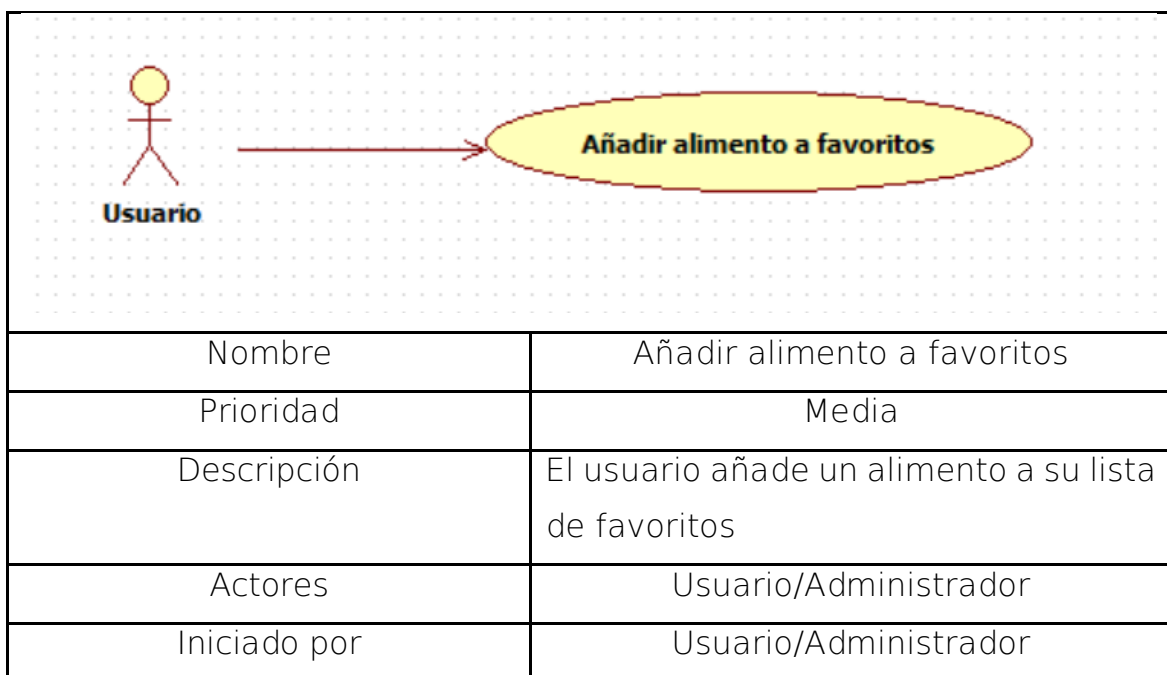
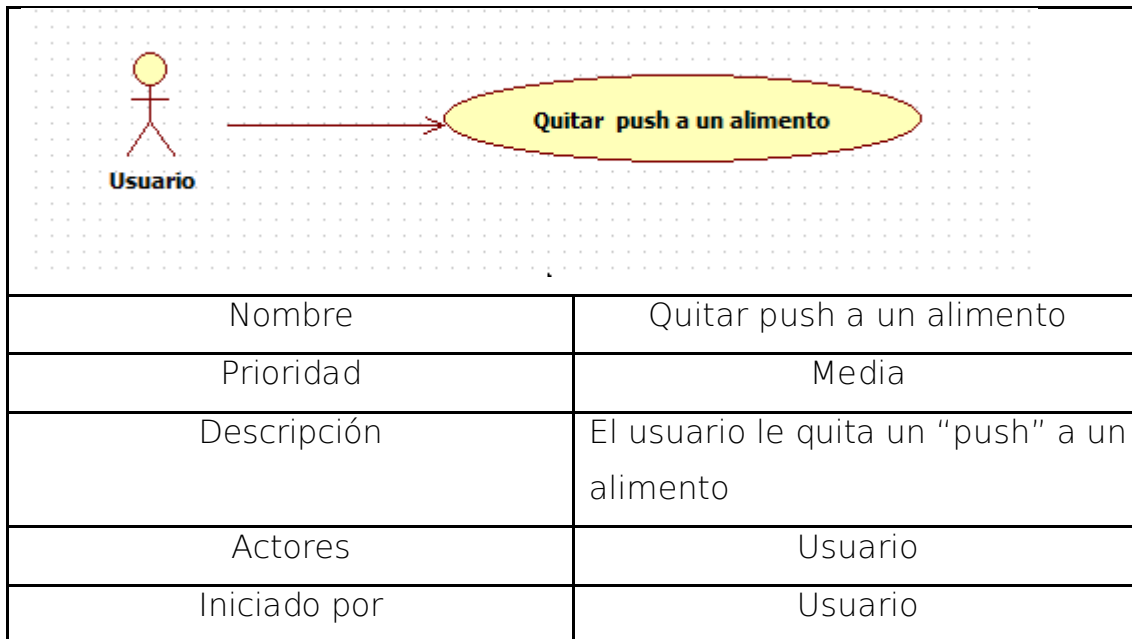
Nombre	Añadir Nuevo Alimento
Prioridad	Alta
Descripción	El usuario añade un nuevo alimento
Actores	Usuario/Administrador
Iniciado por	Usuario/Administrador

<pre> graph LR     Usuario((Usuario)) --&gt; ModificarAlimentoPropio(Modificar Alimento propio) </pre> <p>The diagram shows a stick figure actor labeled 'Usuario' with an arrow pointing to a yellow oval use case labeled 'Modificar Alimento propio'.</p>	
Nombre	Modificar Alimento propio
Prioridad	Alta/Media
Descripción	El usuario modifica los datos de un alimento que el mismo ha creado, si el alimento ha sido confirmado esta opción no será posible.
Actores	Usuario/administrador
Iniciado por	Usuario/administrador

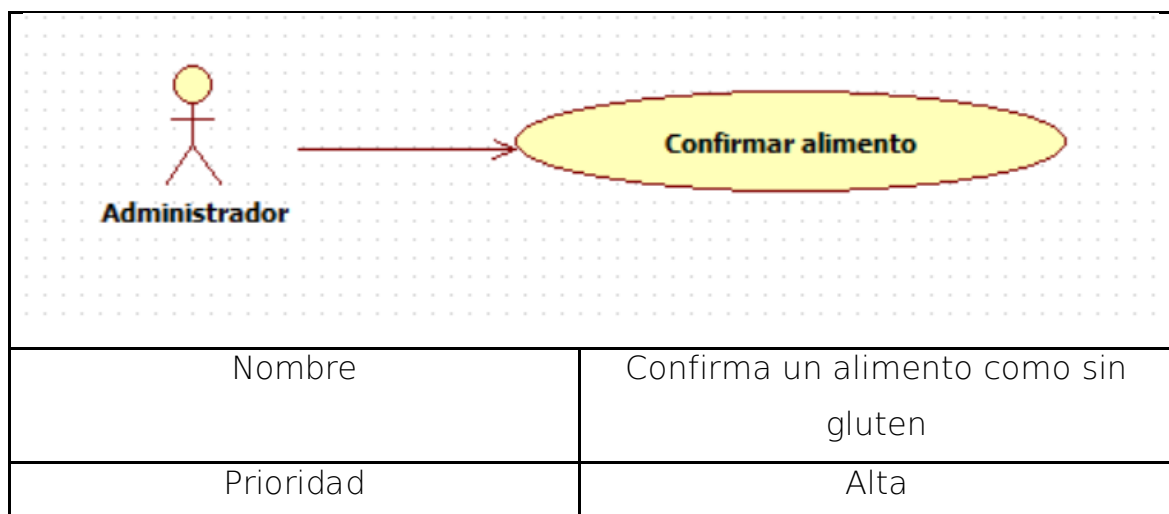
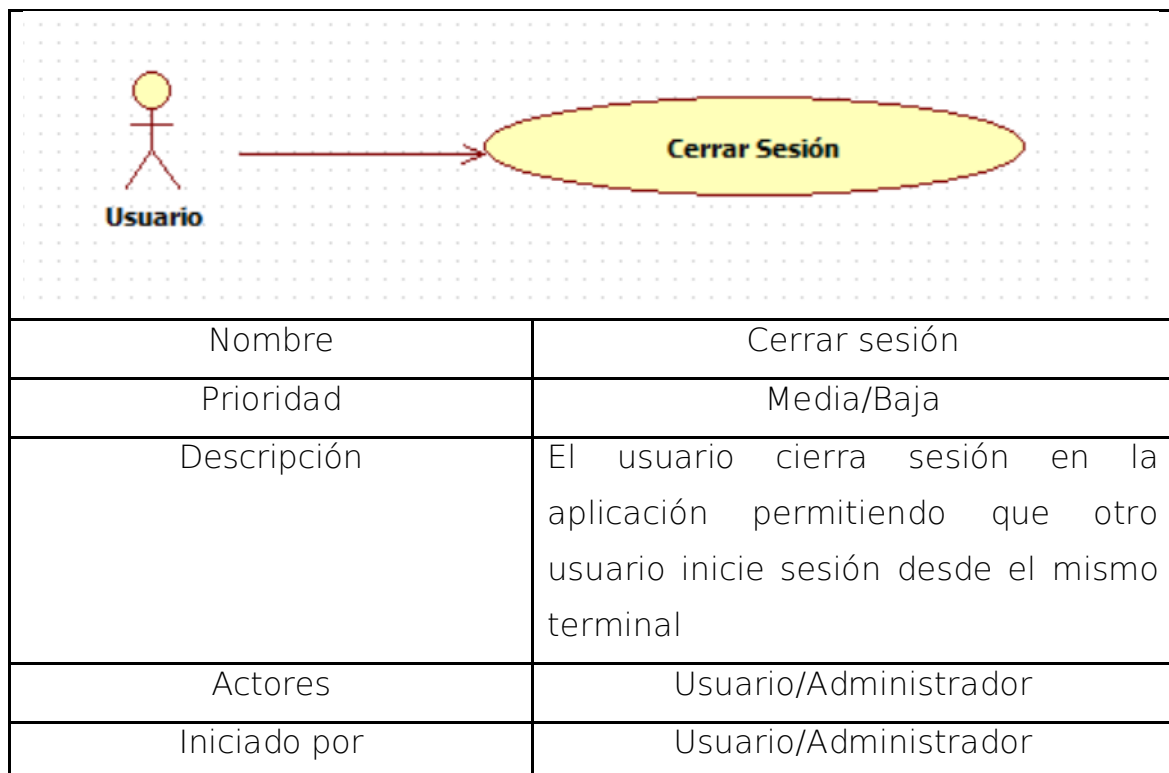
<pre> graph LR     Usuario((Usuario)) --&gt; EliminarAlimentoPropio(Eliminar Alimento propio) </pre> <p>The diagram shows a stick figure actor labeled 'Usuario' with an arrow pointing to a yellow oval use case labeled 'Eliminar Alimento propio'.</p>	
Nombre	Eliminar Alimento Propio
Prioridad	Alta/Media
Descripción	El usuario elimina un alimento que el mismo ha creado, si el

	alimento ha sido confirmado esta opción no será posible.
Actores	Usuario/administrador
Iniciado por	Usuario/administrador





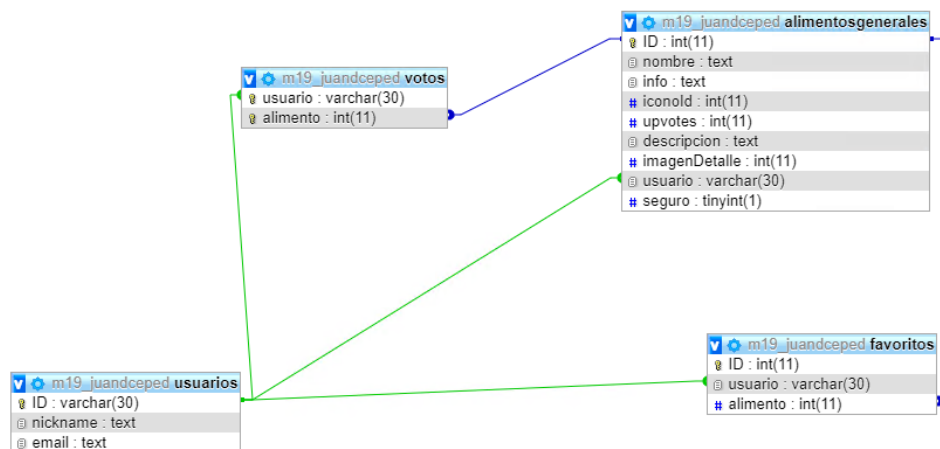
Nombre	Quitar alimento a favoritos
Prioridad	Media
Descripción	El usuario quita un alimento a su lista de favoritos
Actores	Usuario/Administrador
Iniciado por	Usuario/Administrador





Descripción	Un administrador tras una tarea de investigación concluye que un alimento añadido por un usuario es efectivamente sin gluten. (El alimento no puede ser añadido por el mismo).
Actores	Administrador
Iniciado por	Administrador
<pre> graph LR     Admin[Administrador] --&gt; UC(Eliminar alimento inseguro) </pre> <p>The diagram shows a stick figure actor labeled 'Administrador' on the left. A horizontal arrow points from the actor to a yellow oval use case on the right. The use case is labeled 'Eliminar alimento inseguro' in bold black text. The background of the diagram area is a light gray dotted pattern.</p>	
Nombre	Eliminar alimento inseguro
Prioridad	Alta/Media
Descripción	El administrador concluye que un alimento ha sido añadido por un usuario con intenciones maliciosas, y por ende el alimento es eliminado a pesar de que el usuario no lo haya creado.
Actores	Administrador
Iniciado por	Administrador

## Diagrama de Datos



La base de datos consta de tan solo cuatro tables, dos principales y dos que sirven como registros para los favoritos y los votos de los usuarios.

Siendo la tabla *alimentosgenerales* la tabla principal donde se almacenan los alimentos creados por los usuarios. En este cabe destacar el campo seguro, el cual es un tipo booleano que de ser True indicara que el alimento ha sido confirmado como sin gluten

m19_juandceped alimentosgenerales
ID : int(11)
nombre : text
info : text
iconold : int(11)
upvotes : int(11)
descripcion : text
imagenDetalle : int(11)
usuario : varchar(30)
seguro : tinyint(1)

La tabla usuario mantiene un registro de los datos básicos de los usuarios registrados, y se actualiza automáticamente introduciendo

m19_juandceped usuarios
ID : varchar(30)
nickname : text
email : text

una nueva entrada en la tabla cada vez que se crea un nuevo usuario en la aplicación.

Las tablas votos y favoritos hacen las de registros. Ambas añaden o borran una entrada dependiendo de si un usuario de la push o se lo quita a un alimento o si le da me gusta o se lo quita a un alimento respectivamente.

<b>m19_juandceped votos</b>
usuario : varchar(30)
alimento : int(11)

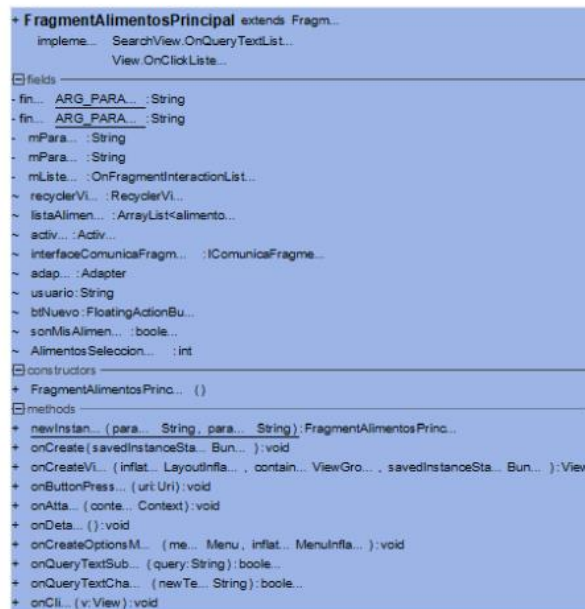
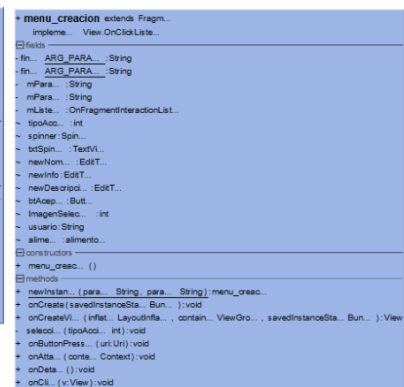
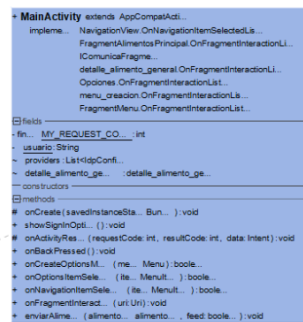
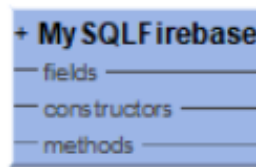
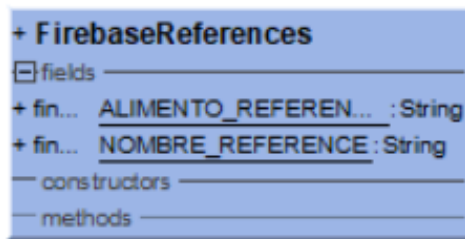
<b>m19_juandceped favoritos</b>
ID : int(11)
usuario : varchar(30)
alimento : int(11)

## Diagrama de Clases

### Diagrama de clases completo



## Diagrama de clases por partes





## Menús de Navegación



Para navegar por la aplicación se usan tanto un menú inicial y un menú lateral. Siendo el primero el que se muestra nada más iniciar la aplicación y la segunda la que el usuario debería de usar para navegar por la aplicación de manera general de forma más rápida y sencilla.



Imágenes de los Diagramas en mejor tamaño y calidad



## EJECUCIÓN DEL PROYECTO

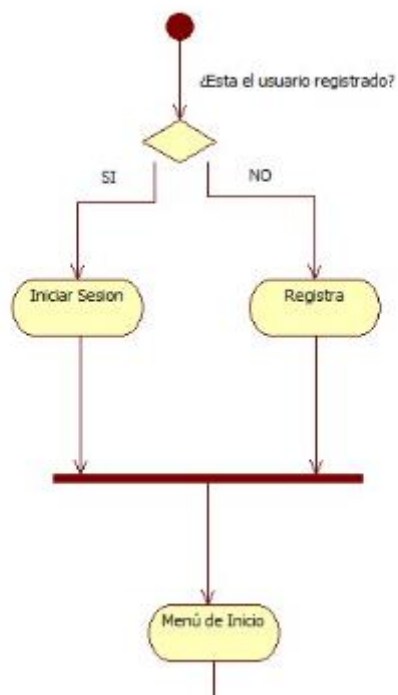
### INTRODUCCIÓN

La aplicación está formada por un total de 17 clases de java, siendo 1 una Activity, 1 una Interface, 6 fragments y 9 clases de java básicas.

Una Activity es un componente de la aplicación que contiene una pantalla con la que los usuarios pueden interactuar para realizar una acción, mientras que un Fragment representa un comportamiento o una parte de la interfaz de usuario en una Activity. Se pueden combinar múltiples fragmentos en una sola actividad para crear una IU multipanel.

## EJECUCIÓN DE LA APLICACIÓN

### INICIO



El usuario al iniciar la aplicación se ejecuta la clase MainActivity, esta al iniciarse la aplicación toma al usuario actual de la clase de Firebase. Si no se devuelve nada, entonces no hay ningún usuario conectado y se inicia todo el proceso para el inicio de sesión/registro de Firebase.

Una vez ya hay un usuario conectado, se carga el fragment del menú de inicio.





## NAVEGACIÓN

La navegación entre pantallas de la aplicación se realiza principalmente entre el menú de inicio y el menú lateral. El menú de inicio se encuentra en un Fragment el cual contiene los botones que se ven en la pantalla. Desde el Fragment declaro los botones del fragment, junto con el método OnClick para ellos.

```
btInicio = vista.findViewById(R.id.btIniciom);
btLista = vista.findViewById(R.id.btLitam);
btForum= vista.findViewById(R.id.btForumm);
btFav = vista.findViewById(R.id.btFavm);
btMis= vista.findViewById(R.id.btMism);
btOpciones = vista.findViewById(R.id.btOpcm);

btInicio.setOnClickListener(this);
btLista.setOnClickListener(this);
btForum.setOnClickListener(this);
btFav.setOnClickListener(this);
btMis.setOnClickListener(this);
btOpciones.setOnClickListener(this);
```

En el método OnClick uso un switch para filtrar el elemento en el cual se ha realizado la acción. En si todos los botones realizan lo mismo al ser pulsados, abren el fragment correspondiente a susodicho botón y le paso los datos necesarios tales como el nombre del usuario o el código correspondiente a la acción que el usuario desea realizar.

```
@Override
public void onClick(View v) {
    Fragment fragment;
    Fragment sustit;
    Bundle bundle = new Bundle();
    String usuario = FirebaseAuth.getInstance().getCurrentUser().getUid();

    switch (v.getId()) {
        case R.id.btIniciom:
            Toast.makeText(getContext(), "Ya se encuentra en esta pantalla", Toast.LENGTH_SHORT).show();
            break;
        case R.id.btLitam:
            fragment = new FragmentAlimentosPrincipal();
            bundle.putString("usuario", usuario);
            bundle.putInt("tipo", 0);
            fragment.setArguments(bundle);

            getActivity().getSupportFragmentManager().beginTransaction()
                .replace(R.id.content_main, fragment)
                .commit();
            break;
    }
```

## RECYCLER VIEW

Un RecyclerView es un contenedor de elementos en forma de lista al igual que la clase ListView. Aunque ambos tienen la misma función, este nuevo elemento permite “reciclar” los ítems que ya no son visibles por el usuario debido al scrolling. Por lo que es ideal para proyectos que manejan grandes volúmenes de ítems que se actualizan constantemente, limitando la visibilidad de elementos.

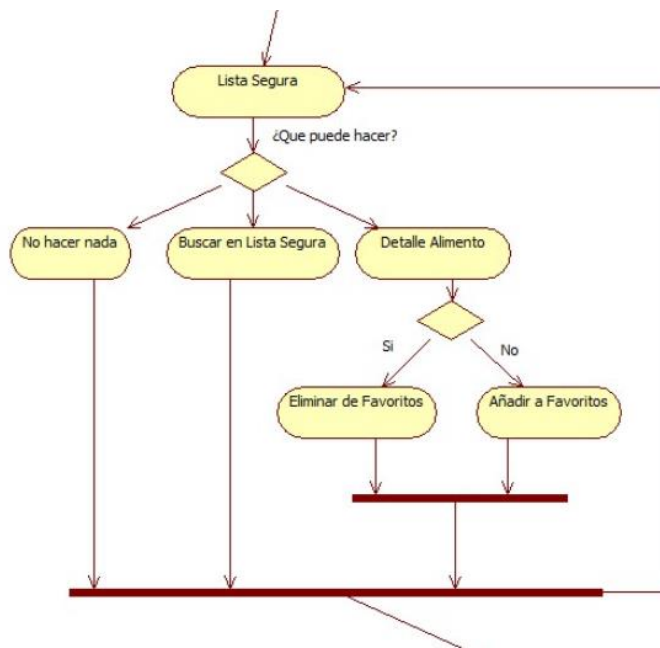
En si un RecyclerView está formado por las partes de la vista, compuestas por la vista donde se verá el RecyclerView y la vista de los objetos que poblarán el mismo, y las clases de java del adaptador y de cuya vista se hayan declarado los datos.

Un adaptador o Adapter es un objeto que comunica a un RecyclerView los datos necesarios para crear los elementos de la lista. Es decir, conecta la lista con una fuente de información como si se tratase de un adaptador de corriente que alimenta a un televisor.

La aplicación solo cuenta con un único RecyclerView, y funciona a través de un Fragment desde el que toda la aplicación pivota. Desde este Fragment iremos filtrando o seleccionando la información que queremos cargar en el RecyclerView.

En mi aplicación el RecyclerView se alimenta a través de la clase Listar.

## Alimentos Confirmados



La lista segura es aquella que muestra los alimentos que han sido confirmados como sin gluten. Cuando se selecciona esta opción se le ejecuta el Fragment AlimentosPrincipal con los datos necesarios.

En este fragment primero declaro todos los elementos visuales tales como el mismo RecyclerView o botones. Dependiendo de la acción que haya decidido hacer el usuario (el botón pulsado) se mostrara un título en pantalla u otro y a su vez se ajustarán otros parámetros tales como la visibilidad de botones, etc...

Antes de terminar el método onCreateView, creo una instancia de la clase Listar con los parámetros correspondientes a la pantalla que desea ver el usuario. En este caso la clase Listar hace una llamada desde un método asíncrono a la API Rest con la cual obtiene los alimentos deseados para este caso, es decir, los confirmados como sin gluten.

```
switch (AlimentosListar) {
    case 0:
        del = new HttpGet( uri: "http://damnation.ddns.net/daniel/phpRestPFG/public/api/inicial");
        resul = HttpGet(resul, httpClient, del);
        break;
}
```

Estos datos serán cargados luego en el RecyclerView en la clase AlimentosPrincipales usando la clase Adapter, la cual en este caso toma y cambia el color del texto.

```
@Override
public void onBindViewHolder(@NonNull AlimentoViewHolder alimentoViewHolder, int i) {
    if(listaAlimentos.get(i).isSeguro()){
        alimentoViewHolder.txtNombre.setTextColor(Color.parseColor( "colorString: "#D4AF37"));
    }
}
```

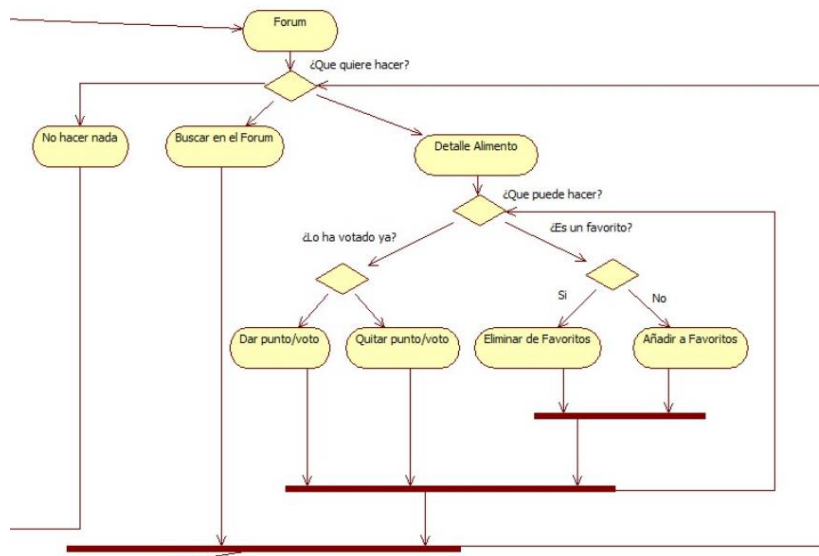
A su vez con los métodos onCreateOptionsMenu y OnQueryTextChange ajusto que quiero que se vea en la barra superior y como filtrar los alimentos según lo que pongan los usuarios en la barra de búsqueda respectivamente.

```
@Override
public void onCreateOptionsMenu(Menu menu, MenuInflater inflater) {
    super.onCreateOptionsMenu(menu, inflater);
    menu.clear();
    inflater.inflate(R.menu.main, menu);
    MenuItem mod = menu.findItem(R.id.idMenuModificar);
    MenuItem eliminar = menu.findItem(R.id.idMenuEliminar);
    mod.setVisible(false);
    eliminar.setVisible(false);
    MenuItem menuItem = menu.findItem(R.id.idBuscar);
    SearchView searchView = (SearchView) menuItem.getActionView();
    searchView.setOnQueryTextListener(this);
}
```

En caso de que cualquier elemento del RecyclerView sea pulsado, se ejecutara el método OnClick, el cual mostrara la pantalla de detalle del alimento.

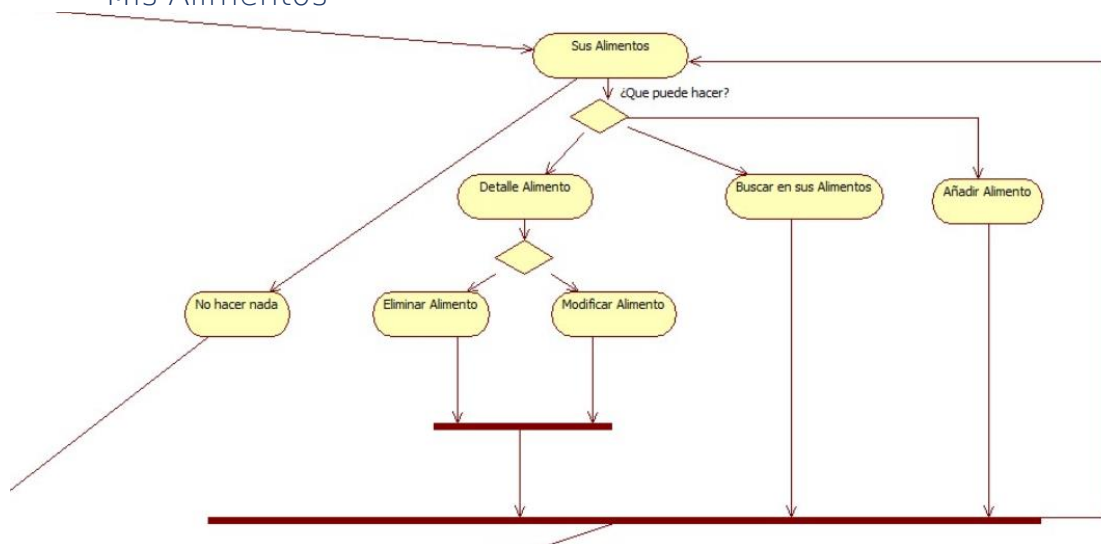
Por motivos de tiempo, el resto de la explicación carecerá de código.

## Alimentos Generales




Alimentos generales viene a seguir la misma formula que el punto anterior, pero esta vez se tomaran todos los alimentos de la base de datos. A su vez los detalles de los alimentos que no hayan sido confirmados tendrán el botón para votar el alimento.

## Mis Alimentos



De la misma manera que en los anteriores se toman los datos deseados de la base de datos y se ajustan botones y opciones de menú, para esta pantalla son los alimentos creados por el usuario

actual. En esta pantalla también se activa el botón de añadir alimentos,  la cual inicia el Fragment de creación. Desde este se mostrarán los campos a rellenar para añadir un nuevo alimento.

A su vez en el detalle de los alimentos creados por el usuario se verán las opciones para modificarlos y eliminarlos. Si se selecciona modificar se llevará al usuario a la pantalla del menú de creación, solo que esta vez ya se mostraran en los campos los datos del alimento seleccionado. En caso de seleccionar Eliminar el alimento seleccionado será borrado.

Modificar

Eliminar

## Favoritos

De la misma manera que en los anteriores se toman los datos deseados de la base de datos y se ajustan botones y opciones de menú, en este caso se cargan los alimentos a los que el usuario haya dado como favoritos.

## DISEÑO

### Introducción

Para el diseño de la aplicación se han tenido en cuenta los patrones clásicos de diseño de apps, en dispositivos móviles tales como la navegación entre pantallas con menús laterales desplegados o la disposición de elementos de forma lineal, ocupándonos de no sobrecargar cada línea con la intención de que el usuario pueda distinguir sin problemas cada elemento en pantalla.



## Selección del entorno de desarrollo

Android Studio es el IDE oficial de Android. Está diseñado para que Android pueda acelerar el desarrollo y te permita crear las apps de mejor calidad para todos los dispositivos de Android ya sea en Java o en Kotlin. A su vez Android Studio está basado en IntelliJ y trae con ello todas las herramientas y comodidades de este entorno.

Ofrece herramientas personalizadas para programadores de Android. Se incluyen herramientas completas de edición, depuración, pruebas y perfilamiento de códigos.



Para la API REST se ha usado Visual Studio Code es un editor de código fuente ligero pero potente disponible para Windows, macOS y Linux. Viene con soporte incorporado para JavaScript, TypeScript y Node.js y tiene un rico ecosistema de extensiones para otros idiomas (como C ++, C #, Java, Python, PHP, Go). Esto permite que sea ideal para la realización de la API usando el framework Slim.



## Selección de la base de datos

Para la base de datos he usado MySQL, la cual he manejado con phpMyAdmin.

phpMyAdmin es una herramienta escrita en PHP con la intención de manejar la administración de MySQL a través de páginas web, utilizando un navegador web. Esta aplicación está implementada hoy en día en la mayoría de servidores y hosting compartidos y puedes acceder a ella a través de su panel de control.



MySQL es un sistema de gestión de base de datos relacional (RDBMS) de código abierto, basado en lenguaje de consulta estructurado (SQL) siendo es muy utilizado en aplicaciones web, como Joomla, Wordpress, Drupal, phpBB, en plataformas (Linux/Windows-Apache-MySQL-PHP/Perl/Python), y por herramientas de seguimiento de errores como Bugzilla. Su popularidad como aplicación web está muy ligada a PHP, que a menudo aparece en combinación con MySQL.



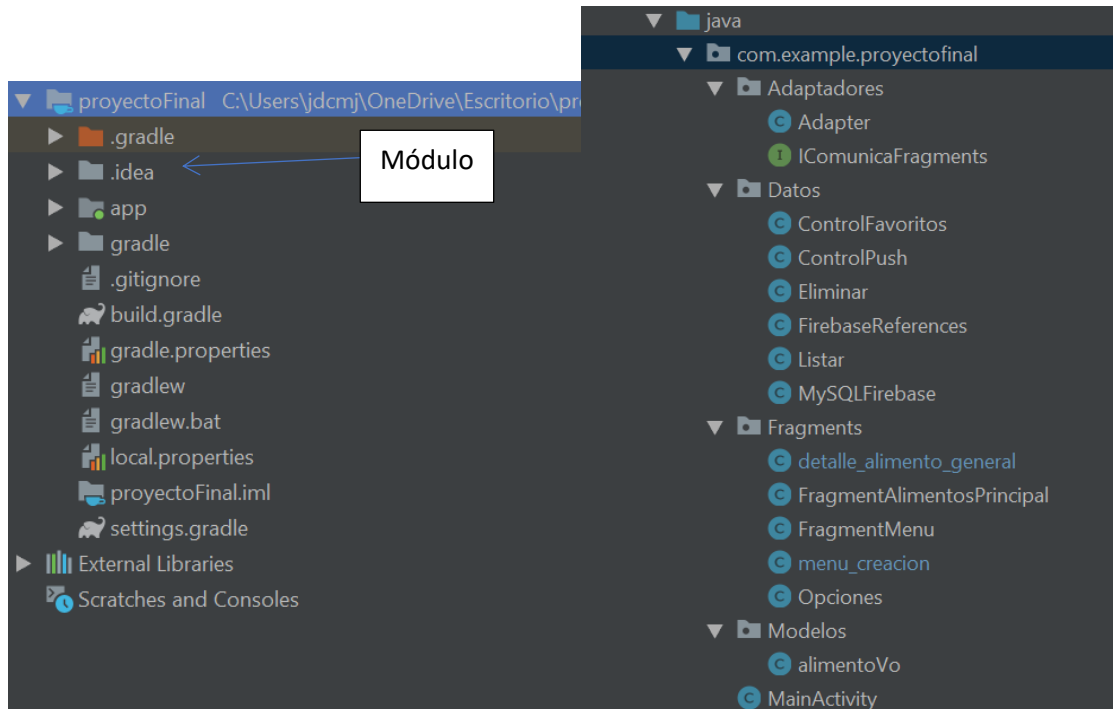
## Arquitectura de la aplicación

Un proyecto en de Android Studio suele siempre la misma arquitectura y distribución de archivos. El proyecto es la entidad única encargada de englobar al resto de elementos, dentro de el suelen estar los módulos, que son la aplicación o aplicaciones o versiones de esta que contenga. Normalmente solo hay un módulo.

Dentro de un módulo las dos partes más importantes son las carpetas *java* y *res*.



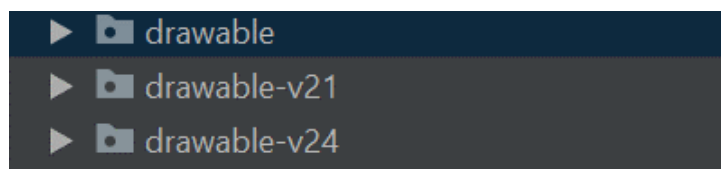
En la carpeta *java* se encuentran todos los códigos fuentes de la aplicación junto o dentro de sus paquetes.



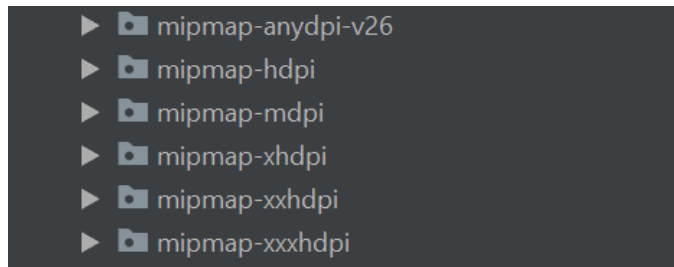
La carpeta *res* contiene todos los ficheros de recursos para la aplicación, ya sean imágenes, pantallas, textos, ...

Ha destacar dentro de *res* están las carpetas:

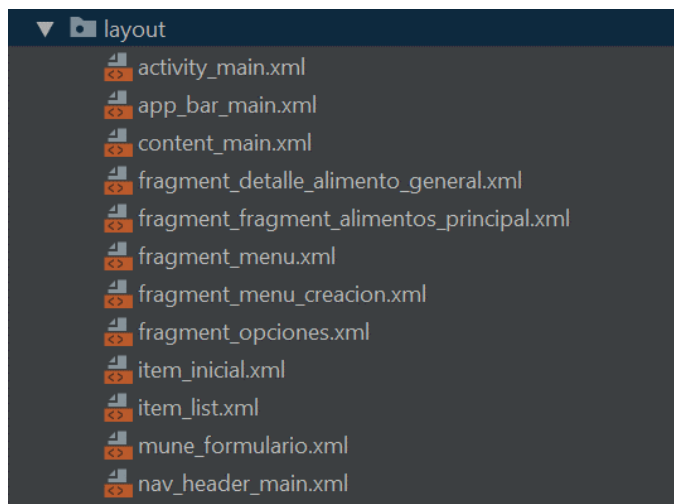
- Drawable: Contiene las imágenes y otros elementos gráficos usados por la aplicación, esta a su vez suele tener otras subcarpetas dependiendo de la resolución.



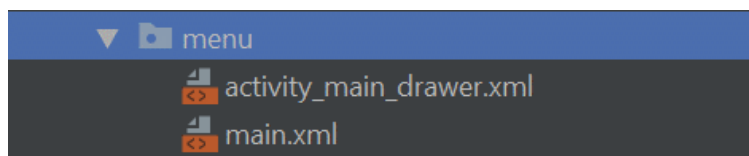
- Mipmap: Contiene los iconos de arranque de la aplicación. Esta también tiene su propio set de subcarpetas.



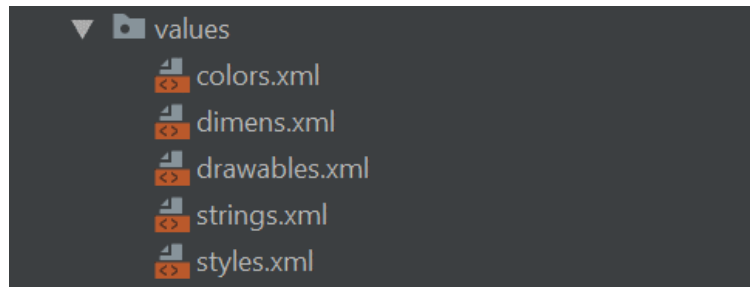
- Layout: Contiene los XML que forman las pantallas de la aplicación.



- Menu: Contiene los XMLs que conforman los diferentes menús y cabeceras de la aplicación.

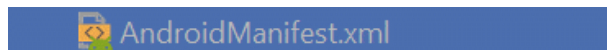


- Values: Contiene XMLs con distintos recursos como colores (colors.xml), texto (strings.xml), estilos (styles.xml) y más.



Otros elementos y carpetas a destacar del proyecto son:

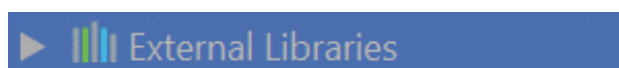
AndroidManifest: Contiene la definición XML de los elementos de la aplicación tales como su identificación o sus componentes.



Build.gradle: Contiene la información necesaria como para compilar el proyecto.



External Libraries: Contiene las librerías externas de Java que utiliza la aplicación. Normalmente no se incluyen las librerías directamente aquí, sino que se referencian en el build.gradle.



## IMPLEMENTACIÓN

Debido a la falta de tiempo, se adjunta un enlace al proyecto con el fin de que puede ser revisada libremente:

<https://github.com/jdanielc/proyectoofp>

## PRUEBAS

### Pruebas realizadas

Debido a la naturaleza de la app y de su desarrollo sumado a la falta de tiempo han llevado a que las únicas pruebas realizadas sean pruebas de usabilidad.

La prueba de usuario (también conocida como de usabilidad) consiste en asegurar que la interfaz de usuario (GUI) de la aplicación sea amigable, intuitiva y que funcione correctamente. Se determina la calidad del software en la forma en la que el usuario interactúa con el sistema, considerando la facilidad de uso y satisfacción del cliente.

Para la fase de pruebas se ha seleccionado a dos usuarios de prueba (no precisamente voluntarios) para que realicen un total de 10 acciones consideradas claves y básicas para el funcionamiento de la app.

Los terminales en los que se probaran son:

Terminal 1:

NOMBRE	REDMI NOTE 6 PRO
CPU	Snapdragon 636
RAM	3GB
ROM	32GB
SO	Android 9

Terminal 2:

NOMBRE	POCOPHONE
CPU	Snapdragon 845
RAM	6GB

ROM	128GB
SO	Android 8.1

Las pruebas a realizar son las siguientes:

NÚMERO PRUEBA	DESCRIPCIÓN
Prueba 1	Registrarse con éxito
Prueba 2	Iniciar con éxito
Prueba 3	Añadir alimento
Prueba 4	Modificar Alimento
Prueba 5	Eliminar Alimento
Prueba 6	Añadir Favorito
Prueba 7	Dar Favoritos
Prueba 8	Quitar Favoritos
Prueba 9	Darle Push a un alimento
Prueba 10	Navegar todas las pantallas
Prueba 11	Buscar alimento
Prueba 12	Cerrar sesión

Esta lista de pruebas fue entregada junto con el manual de usuario y la apk, siendo estos sus resultados:

Terminal 1:

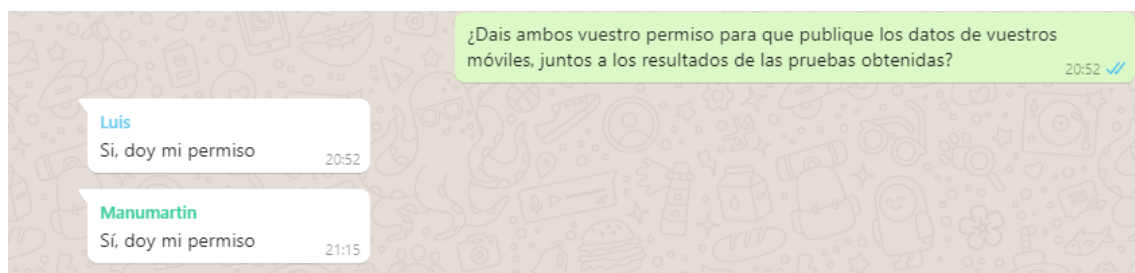
NÚMERO DE PRUEBA	RESULTADO
Prueba 1	POSITIVO
Prueba 2	POSITIVO
Prueba 3	POSITIVO
Prueba 4	POSITIVO
Prueba 5	POSITIVO
Prueba 6	POSITIVO
Prueba 7	POSITIVO

Prueba 8	POSITIVO
Prueba 9	DATOS INCONCLUSOS
Prueba 10	POSITIVO
Prueba 11	POSITIVO
Prueba 12	POSITIVO

Terminal 2:

NÚMERO DE PRUEBA	RESULTADO
Prueba 1	POSITIVO
Prueba 2	POSITIVO
Prueba 3	POSITIVO
Prueba 4	POSITIVO
Prueba 5	POSITIVO
Prueba 6	POSITIVO
Prueba 7	POSITIVO
Prueba 8	POSITIVO
Prueba 9	DATOS INCONCLUSOS
Prueba 10	POSITIVO
Prueba 11	POSITIVO
Prueba 12	POSITIVO

\*Las pruebas pueden haberse realizado bajo la directa supervisión y guiado del desarrollador y bajo chantaje emocional



## Conclusiones

De las pruebas se puede comprender que, las funcionalidades básicas implementadas son funcionales en situaciones de uso común bajo condiciones normales.

## 4. CONCLUSIONES

### Conclusiones finales

La aplicación ha sido llevada a cabo con toda una vida de experiencias como celíaco a mis espaldas la cual me ha ayuda a encaminar este proyecto y adonde quería que llegara y a donde quiero que llegue. Con esto en mente y el deseo de llegar un paso más lejos de lo que habíamos dado en clase me dispuse aprender e investigar con el fin de usar los recursos y tecnologías correctas.

Como celíaco he sabido el calvario que es ir a la compra, la indiferencia de los supermercados y fabricantes y como la mayoría de las veces solo te quedan como recursos la ayuda y experiencia de otros celiacos.

El desarrollo ha sido largo, difícil y exhaustivo tanto física como mentalmente, pero creo que los resultados han sido óptimos.

### Posibles ampliaciones y mejoras

La entrega del proyecto no será el fin de la aplicación, en el futuro cercano tengo pensado continuar con su desarrollo mejorándola y ampliándola con, por ejemplo:

- Implementar administradores con todos sus sistemas
- Opción para que los usuarios puedan usar sus propias fotos.
- Comentarios de usuarios en los alimentos.
- Mensajes privados entre usuarios
- Salas de chat.
- Opciones de personalizar los datos del usuario.
- Mejoras generales de diseño

- Vistas para pantallas en horizontal
- Vistas para Tablet.
- Versión para iPhone.

### Opinión personal

Para mí el proyecto ha sido todo un reto el cual he aceptado con placer ante la expectación de tomarme de frente con un proyecto de un calibre como este. Me alegra haber podido ir superando los obstáculos del proyecto y sobre todo el conocimiento que he conseguido tanto estudiando e investigando como por la experiencia de realizar semejante obra. Me alegra saber que todo este trabajo ha ido enfocado a ayudar a otros celíacos como mi hermana. Si bien es cierto que tengo la espinita clavado de no haber implementado todo lo deseado, he de aceptar que mi idea original era demasiado para el tiempo a mi disposición o como mi falta de conocimiento en el uso de programas de edición y diseño han hecho que visualmente ciertas partes de la aplicación no sean tan atractivas como me hubiera gustado. Pero con todo, he de confesar que me encuentro más que satisfecho con el resultado final.

### BIBLIOGRAFÍA

RecyclerView:

<https://www.youtube.com/watch?v=fnavhYGgZD4>

<https://www.youtube.com/watch?v=X-hYIQcmXUw>

<https://www.youtube.com/watch?v=qpLDlgjngyA>

<https://www.youtube.com/watch?v=ZBEtt2rNS6M>

Fragments:

<https://developer.android.com/guide/components/fragments?hl=es-419>

<https://www.youtube.com/watch?v=wopVTAeaew8>



Firebase

<https://www.youtube.com/watch?v=qvOG7A7zb68>

[https://www.youtube.com/watch?v=EO-\\_vwfVi7c](https://www.youtube.com/watch?v=EO-_vwfVi7c)

<https://firebase.google.com/docs/android/setup>

API REST

<https://www.youtube.com/watch?v=jlwDehC2sEM>

Documentación

[http://openaccess.uoc.edu/webapps/o2/bitstream/10609/66085/3/mromerop\\_oTFG0617memoria.pdf](http://openaccess.uoc.edu/webapps/o2/bitstream/10609/66085/3/mromerop_oTFG0617memoria.pdf)

## GLOSARIO

API REST: Servicio de comunicación para el protocolo HTTP

RecyclerView: El widget RecyclerView es una versión más avanzada y flexible de ListView. En el modelo RecyclerView, varios componentes diferentes trabajan juntos para mostrar sus datos en una lista.

Fragment: Un Fragment representa un comportamiento o una parte de la interfaz de usuario en una Activity.

Acitivity: Una actividad es una cosa única y enfocada que el usuario puede hacer.