



Arquitectura e Integración de Sistemas Software

Grado de Ingeniería del Software

Curso 2018-2019

Cortés Muñoz, Juan Carlos (juacormun@alum.us.es)

Molino Peña, María Elena (marmolpen3@alum.us.es)

Muñoz Aranda, Alejandro José (alemunara@alum.us.es)

Ruano Fernández, Mario (mruano@us.es)

Tutor: Márquez Chamorro, Alfonso Eduardo.

Número de grupo:

Enlace de la aplicación: <https://foodreams.appspot.com>

Repositorio: <https://repositorio.informatica.us.es/svn/k3ea7nkxmx7n3yg24tr>

HISTORIAL DE VERSIONES

Fecha	Versión	Detalles	Participantes
16/03/2019	1.0	<ul style="list-style-type: none"> - Introducción y motivación del proyecto, prototipos de las interfaces de usuario y diagramas UML de componentes, despliegue y secuencia de alto nivel. 	Juan Carlos Cortés María Elena Molino Alejandro José Muñoz Mario Ruano
26/04/2019	1.0	<ul style="list-style-type: none"> - Nuevos diagramas UML (diagrama de clases y diagramas de secuencia). - Prototipo funcional desplegado en AppEngine en el que se hace un consumo de las APIs integradas. - Documentación de la API REST de FooDreams. 	Juan Carlos Cortés María Elena Molino Alejandro José Muñoz Mario Ruano
13/05/2019	2.0	<ul style="list-style-type: none"> - Implementación de operaciones CRUD para el recurso Playlist de Youtube haciendo uso de su API REST. - Finalización del mashup totalmente funcional. - Implementación de la API REST de FooDreams y su documentación web. 	Juan Carlos Cortés María Elena Molino Alejandro José Muñoz Mario Ruano

Índice

1	Introducción.....	5
1.1	Aplicaciones integradas.....	6
1.2	Evolución del proyecto.....	6
2	Prototipos de interfaz de usuario	8
2.1	Vista desktop de página principal de inicio	9
2.2	Vista desktop de resultados de búsqueda de recetas	10
2.3	Vista desktop de resultados de fotos de recetas relacionadas	11
2.4	Vista desktop de vídeos relacionados con una receta seleccionada.....	12
2.5	Vista desktop del formulario de creación de playlist	13
2.6	Vista desktop de playlists creadas por el usuario.....	14
2.7	Vista mobile de página principal de inicio.....	15
2.8	Vista mobile de resultados de búsqueda de recetas.....	15
2.9	Vista mobile de resultados de fotos de recetas relacionadas	16
2.10	Vista mobile de vídeos relacionados con una receta seleccionada.....	16
2.11	Vista mobile del formulario de creación de playlist	17
2.12	Vista mobile de playlists creadas por el usuario	17
3	Arquitectura.....	18
3.1	Diagrama de componentes	18
3.2	Diagrama de despliegue.....	19
3.3	Diagrama de secuencia de alto nivel.....	20
3.4	Diagrama de clases.....	21
3.4.1	Diagramas de secuencia	22
3.4.2	Búsqueda de información e imágenes de recetas.....	22
3.4.3	Búsqueda de vídeos de recetas	23
3.4.4	Creación de playlist de recetas.....	24
3.4.5	Añadir vídeos a playlist de recetas	25
3.4.6	Editar playlist de recetas	26
3.4.7	Eliminar playlist de recetas.....	27
4	Implementación	28
5	Pruebas	30

6	Manual de usuario	33
6.1	Mashup	33
6.2	Documentación de API REST	39
6.3	Recurso ingrediente	39
6.4	Recurso receta.....	40
7	Referencias	42

1 Introducción

La capacidad de creatividad del ser humano no tiene límite, ni entiende de barreras, y es aplicable a cualquier aspecto del día a día. Todos tenemos dentro a un gran artista en potencia. Artes plásticas, música, ingeniería, actividades sociales...sin olvidarnos de algo que todos compartimos con total probabilidad de éxito, el amor y el gusto por la comida.

Sin embargo, hoy en día, los noticiarios y los titulares de la prensa generalista suelen transmitir la voz de alarma de muchos especialistas nutricionales, los cuales alertan y denuncian los altos índices de obesidad en la población, las malas costumbres a la hora de alimentarnos o la calidad de los ingredientes de muchos alimentos.

Este proyecto nace con la convicción de que toda persona tiene el potencial necesario para crear y dar su toque personal a los platos más sabrosos y a las recetas más famosas del planeta.

Del mismo modo, esta afición por la cocina y por la comida pueden ir de la mano de prácticas saludables en los fogones a través de información nutricional de los ingredientes y los alimentos utilizados en las recetas, sin impedir que se consigan los mejores resultados.

En este contexto es donde se inserta **FooDreams**, una solución pensada para dar rienda suelta a la creatividad en la cocina al mismo tiempo que se tiene la información nutricional necesaria para preparar platos ricos y saludables.

FooDreams ofrece como resultados de la búsqueda de sus usuarios una relación de ingredientes, así como su información nutricional, al tiempo que presenta una serie de fotografías, vídeos y portales de recetas para inspirar nuevas creaciones culinarias.

1.1 Aplicaciones integradas

FooDreams hace uso de tres APIs para prestar al cliente sus servicios de una forma transversal. A continuación, se describe el principal uso y el servicio consumido en cada una de ellas:

- **Edaman:** brinda la información nutricional completa de recetas, facilitando también un listado de los ingredientes necesarios para su elaboración, así como un enlace a una explicación más detallada para su realización.
- **Flickr:** famosa red de imágenes y fotografías que ofrece la posibilidad de ilustrar la búsqueda del usuario con imágenes del plato que solicita en la búsqueda.
- **Youtube:** se hace uso del servicio de búsqueda de vídeos, facilitando al usuario información visual de la elaboración del plato que solicita en la búsqueda. Además, se hace uso de su API para crear playlists con los videos de las recetas que más han gustado al usuario.

Nombre aplicación	URL documentación API
Edamam	https://developer.edamam.com/edamam-docs-recipe-api
Flickr	https://www.flickr.com/services/api/
Youtube	https://developers.google.com/youtube/v3/docs/

TABLA 1. APPLICACIÓN INTEGRADAS

1.2 Evolución del proyecto

La primera fase del proyecto se desarrolló de forma exitosa, cumpliéndose con el plazo establecido para la finalización del primer entregable. El equipo fue capaz de definir una idea, seleccionar varias APIs que permitían llevarla a cabo y modelar un sistema que implementara la funcionalidad que se buscaba para la aplicación.

En un principio se tuvo que volver al punto de partida. Tras tener una idea lo suficientemente buena y original para su desarrollo, la imposibilidad de hacer uso de las APIs que se tenían previstas, algunas por motivos de disponibilidad de los propios servicios, otras por la no gratuitad de estos, hizo que se abortase esa línea de trabajo.

El equipo reaccionó con rapidez y llegó pronto a una nueva idea, la cual ha sido desarrollada en el presente trabajo.

Tal como se ha expresado en la sección de introducción, se pretende desarrollar una aplicación que ofrezca al usuario no sólo la información común y habitual de recetas, sino que ésta esté acompañada por diferentes fuentes que le permitan crear su propia

versión de los platos, además de contar con el valor añadido de tener una información nutricional completa.

En la segunda fase del proyecto, de cara al segundo entregable, fue necesario redefinir el consumo que se haría de las APIs finalmente seleccionadas para integrarse en el mashup. Se partió de la idea de que el usuario hiciera uso de la API de Flickr para publicar sus propias fotos, pero dado que el sistema de autenticación de esta API hacía uso de OAuth 1.0 y a la insuficiente documentación de cómo implementarlo, se decidió pasar a un segundo plan.

El usuario podría crear playlists con los videos que devuelve la búsqueda de recetas, pudiendo seleccionar así sus favoritos. Para ello se ha hecho uso del servicio de creación y manipulación de playlists de videos de Youtube, la cual sí trabaja con el sistema de autenticación OAuth 2.0.

Además, en esta fase también se completaron en su totalidad todos los diagramas UML que faltaban hasta la fecha, como el diagrama de clases, con el que tuvimos claro el mapa general de archivos que formarían la aplicación, a qué capa pertenecía cada uno de ellos y qué dependencias existían entre ellos; o los diagramas de secuencia, los cuales fueron evolucionando conforme íbamos avanzado en la implementación de diferentes servicios.

Aquí también fue el momento en el que se planteó de manera escrita el contrato de la API Rest que brindaría FooDreams.

Esta fase concluyó de manera exitosa y el equipo valoró satisfactoriamente los objetivos marcados y alcanzados.

Por último, la tercera y última fase vino acompañada de algunas funcionalidades añadidas. Aquí se trabajó más en la implementación de la funcionalidad del mashup y en el desarrollo e implementación de nuestra propia API Rest (proyecto paralelo).

Además de ofrecer la posibilidad de crear playlists de videos, se decidió implementar por completo todos los servicios que brinda la API de Youtube para su recurso de playlists. De esta manera, el usuario que utiliza nuestra aplicación tiene el control absoluto de sus propias playlists, ya no sólo añadiéndole videos de su interés, sino editando la información de la de esta y eliminándola por completo.

Estas nuevas funcionalidades requirieron algunas modificaciones en los diagramas de clases y los diagramas de secuencia.

También se implementó la API Rest de FooDreams y se generó su documentación a través de la herramienta de Swagger.

Después de verificar las pruebas y comprobar el correcto funcionamiento del mashup, revisando, al mismo tiempo, el diseño frontend (CSS y JS), se dio por finalizado el proyecto en el plazo marcado de manera satisfactoria.

2 Prototipos de interfaz de usuario

En esta sección se presentan los prototipos de las diferentes vistas de interfaces de usuario que forman parte de la aplicación.

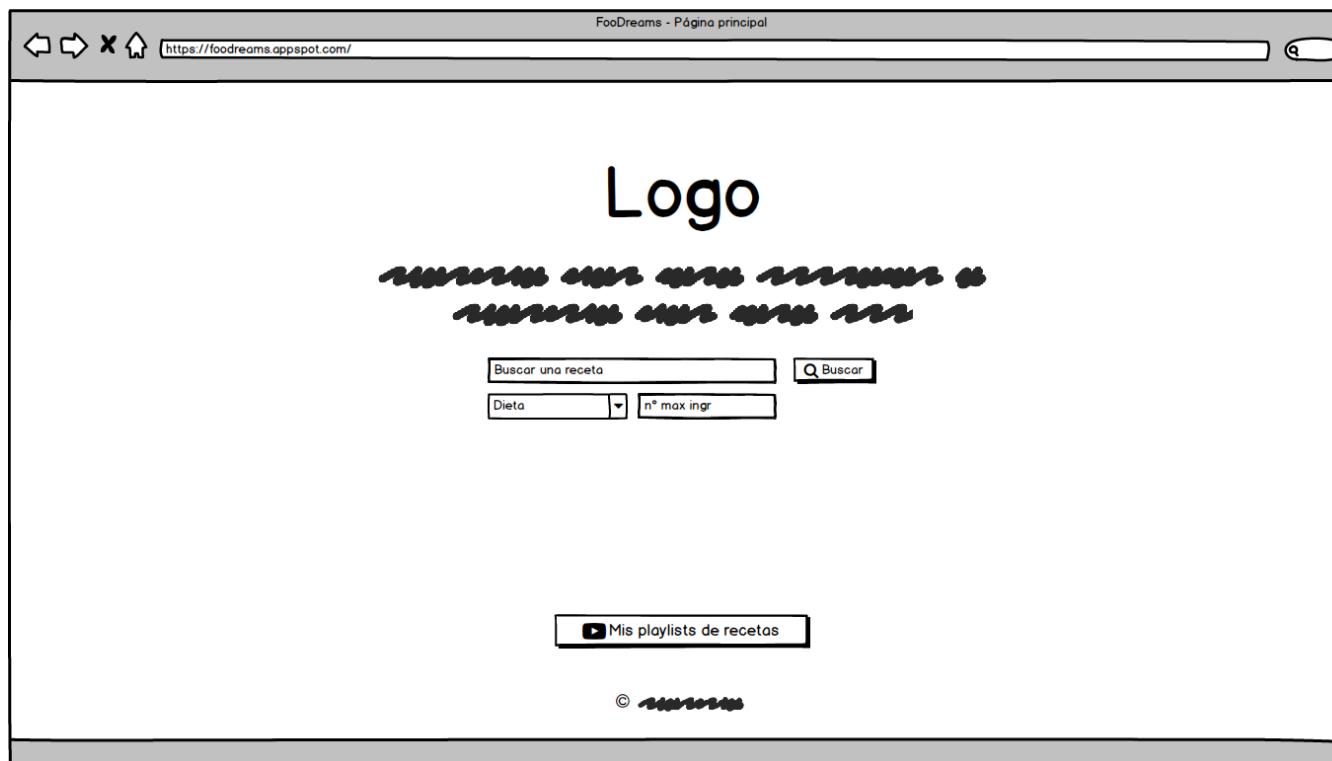
Es importante resaltar que, para llevar a cabo el diseño de los prototipos, se ha seguido una filosofía “mobile-first”, buscado siempre que el diseño pueda ser adaptable a diferentes dispositivos de resolución variable.

En primer lugar, se presentan las vistas propias de la versión desktop de la aplicación, acompañándose de una breve descripción textual que aclara algunos aspectos que no han podido tratarse a través del mockup.

Por último, se listan las vistas de interfaces pertenecientes a la versión mobile de la aplicación. En este caso, sólo se aclaran, mediante una descripción textual, los detalles exclusivos de estas vistas, sin reiterar explicaciones que ya se especifican en la versión desktop, dado que se trata de una aplicación responsive.

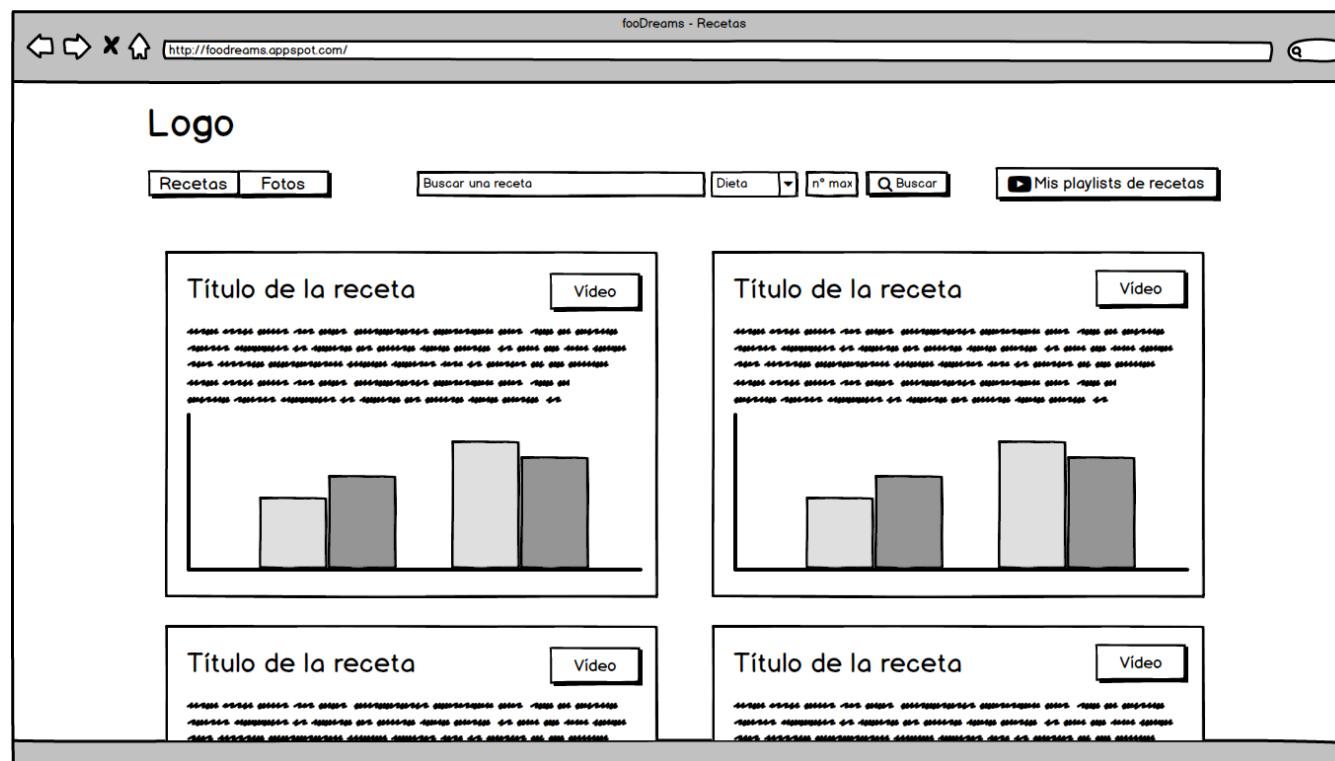
2.1 Vista desktop de página principal de inicio

En esta primera vista nos encontramos ante la pantalla principal, donde el usuario puede buscar la receta que desee. Además, puede hacer uso de un filtrado por varios parámetros como el tipo de dieta (equilibrada, alta en proteínas, etc.) y un número máximo de ingredientes. Cada vez que se pulsa sobre el logo de FooDreams en cualquiera de las vistas, el usuario es redirigido a esta vista. Por último, el usuario puede acceder a sus playlists de vídeos de recetas.



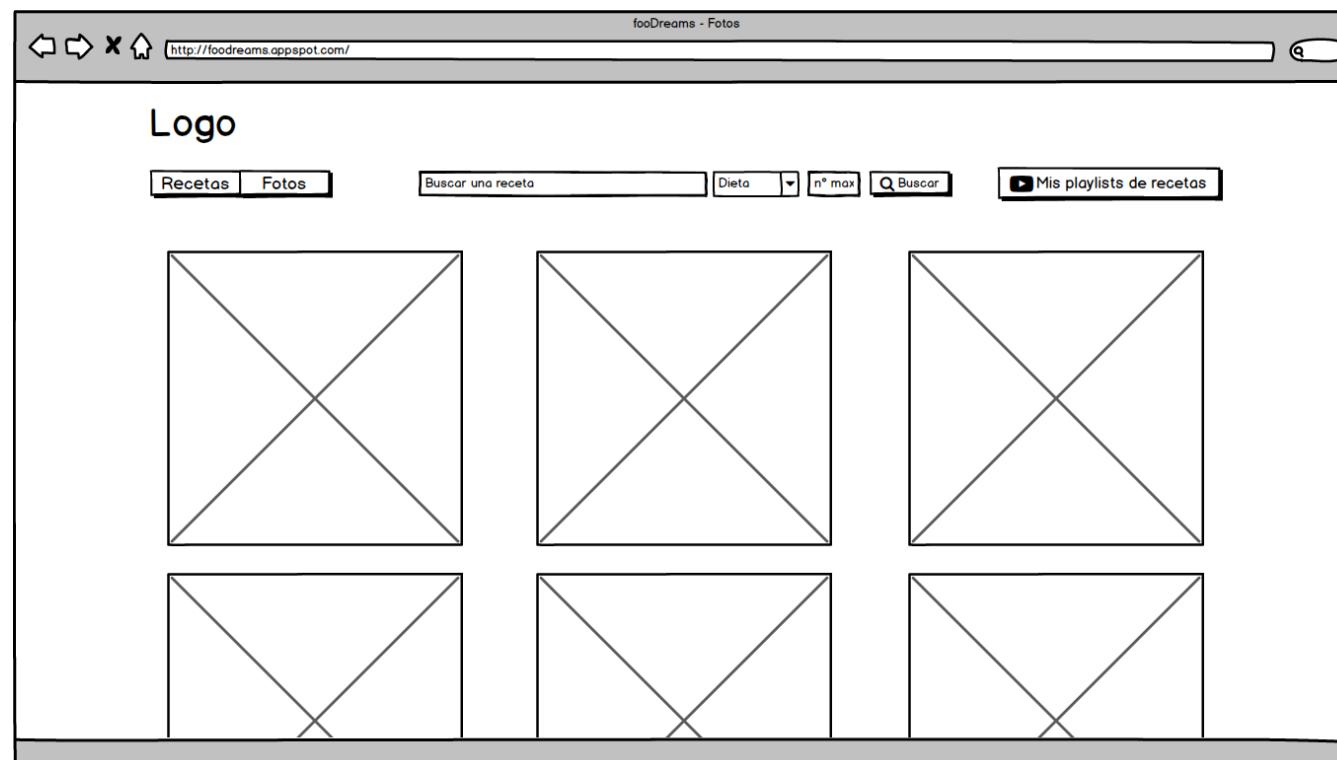
2.2 Vista desktop de resultados de búsqueda de recetas

Una vez realizada la búsqueda, se muestra al usuario una vista con diferentes recetas. Cada una de ellas contiene un título, el cual enlaza a la página fuente de la receta; los ingredientes, información nutricional y un enlace a los vídeos relacionados. A su vez, la parte superior está ocupada por el buscador y un botón para acceder a las playlists del usuario.



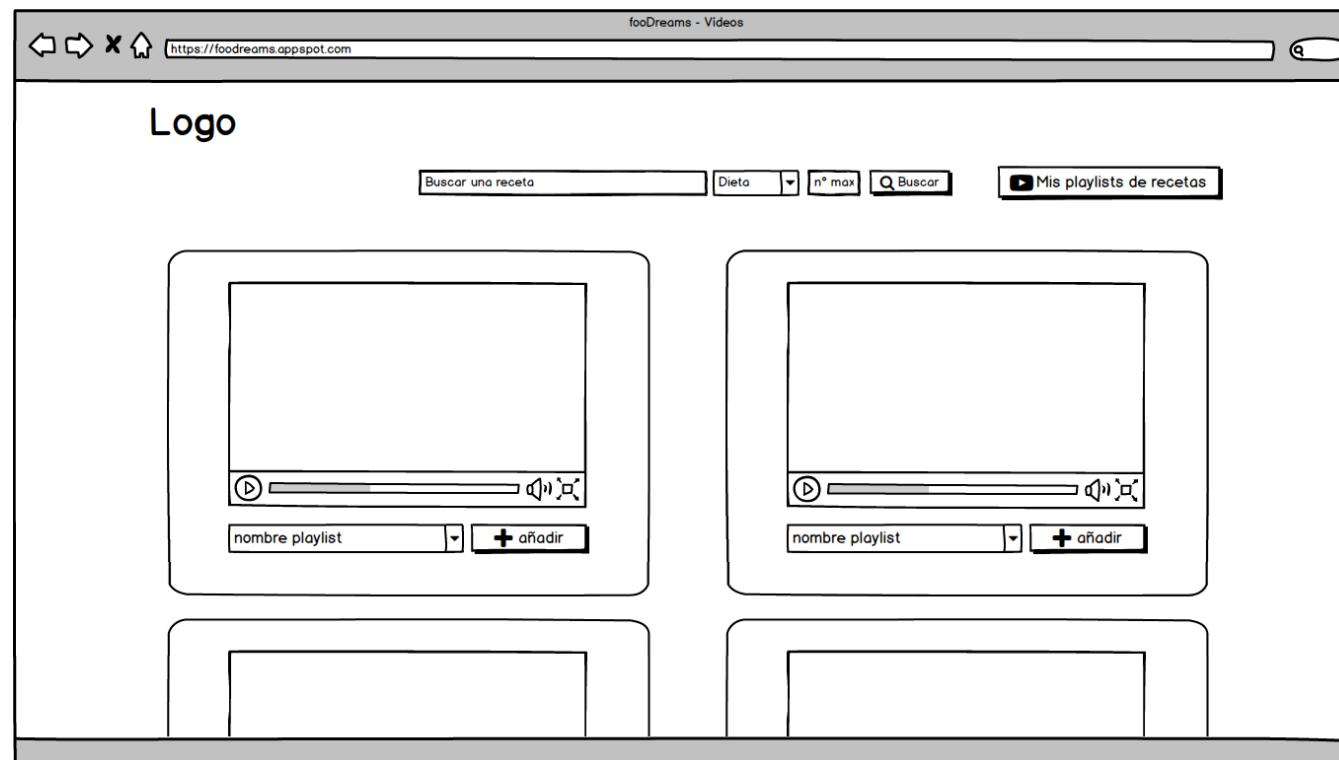
2.3 Vista desktop de resultados de fotos de recetas relacionadas

En esta vista el usuario obtiene fotos relacionadas con la búsqueda realizada. Además, se mantiene la misma cabecera que en la vista anterior, el buscador, un panel de navegación y un botón que le lleva a sus playlists de vídeos de recetas.



2.4 Vista desktop de vídeos relacionados con una receta seleccionada

Una vez pulsado el botón de vídeos desde una receta concreta, se pasa a la vista en la cual se presentan los vídeos relacionados con la misma. Además, se mantiene la misma cabecera que en la vista anterior, el buscador, un panel de navegación y un botón para acceder a las playlists.



2.5 Vista desktop del formulario de creación de playlist

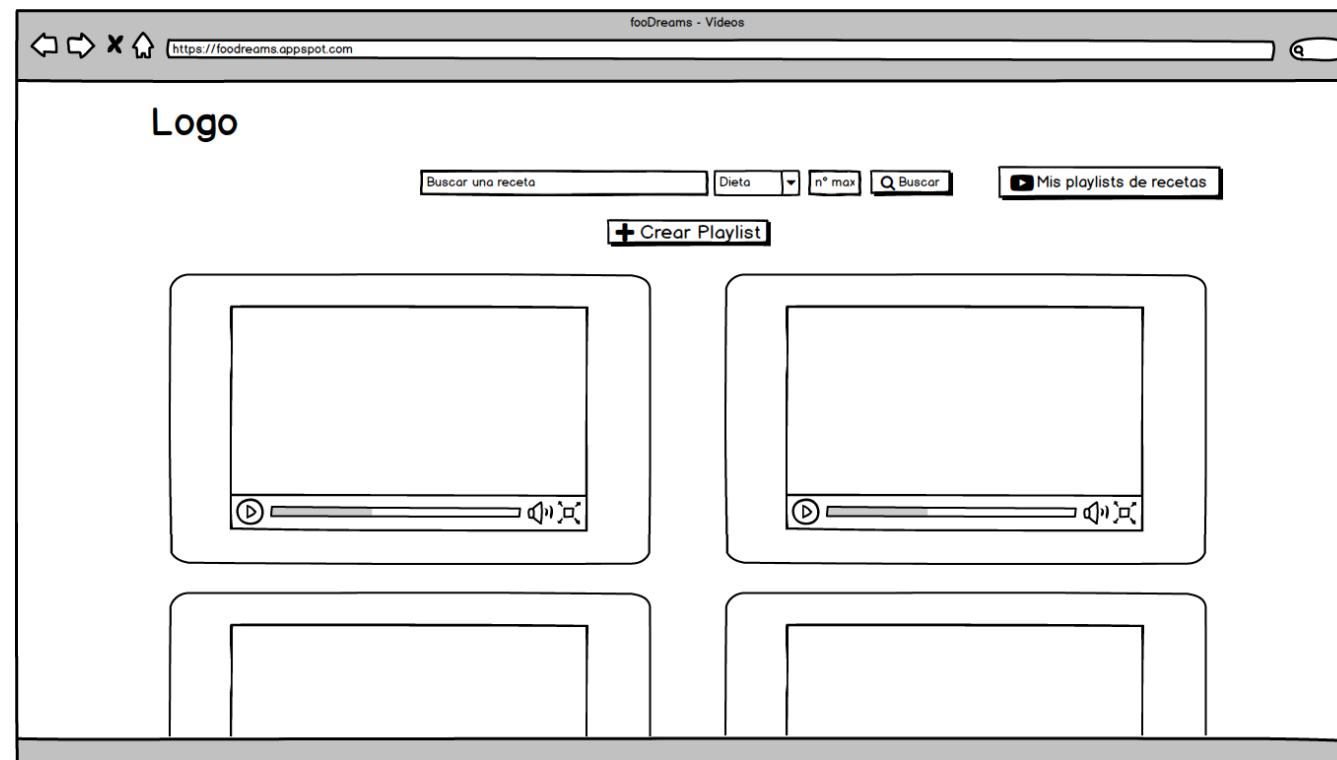
Tras pulsar el botón de “Crear playlist” en cualquiera de las vistas en las que aparece, se muestra un formulario de registro de una nueva playlist. Se hará uso de la API de Youtube para la creación de esta.

The screenshot shows a desktop browser window with the following details:

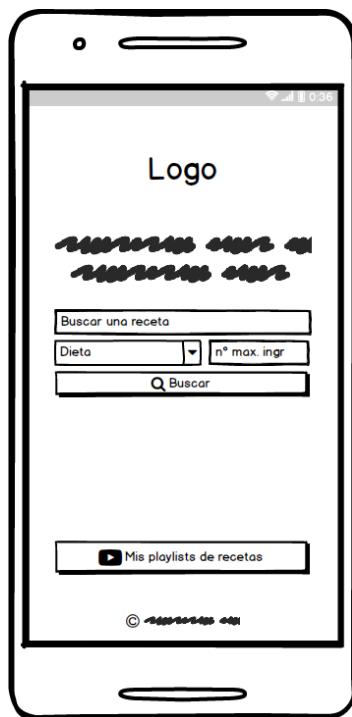
- Title Bar:** fooDreams - Crear playlist
- Address Bar:** https://foodreams.appspot.com
- Content Area:**
 - Logo:** A placeholder for a logo, currently showing the word "Logo".
 - Search Bar:** Buscar una receta, Dieta (dropdown), n° max, Buscar button.
 - Link:** Mis playlists de recetas
 - New Playlist Form:** A modal dialog titled "Nueva playlist" containing:
 - Nombre de la playlist input field
 - Descripción de la playlist... input field
 - A large blue button labeled "+ Crear nueva playlist de recetas"
- Footer:** © foodreams

2.6 Vista desktop de playlists creadas por el usuario

Por último, en la vista del perfil del usuario se listan las playlists creadas por este. En estas playlists el usuario puede ir añadiendo los vídeos de recetas que más le interesen. Además, también tiene la posibilidad de crear nuevas playlists.



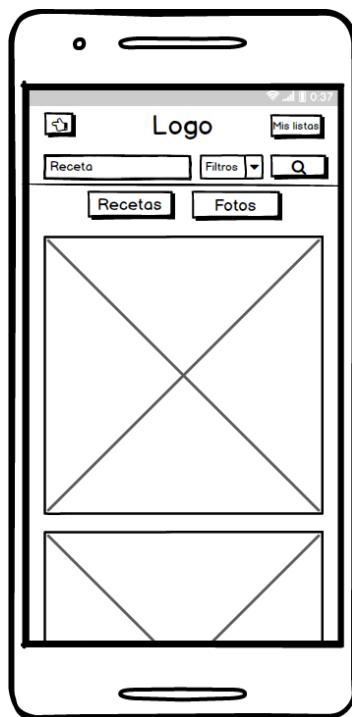
2.7 Vista mobile de página principal de inicio



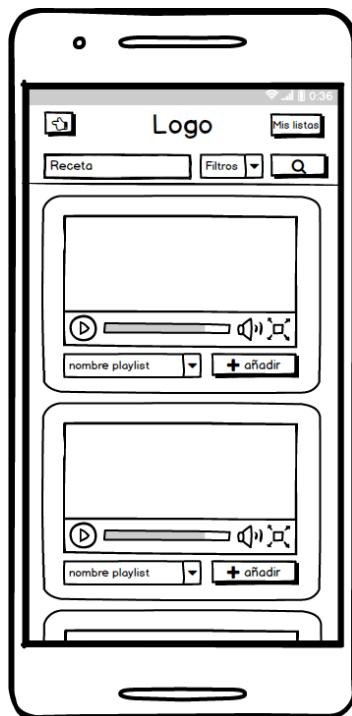
2.8 Vista mobile de resultados de búsqueda de recetas



2.9 Vista mobile de resultados de fotos de recetas relacionadas



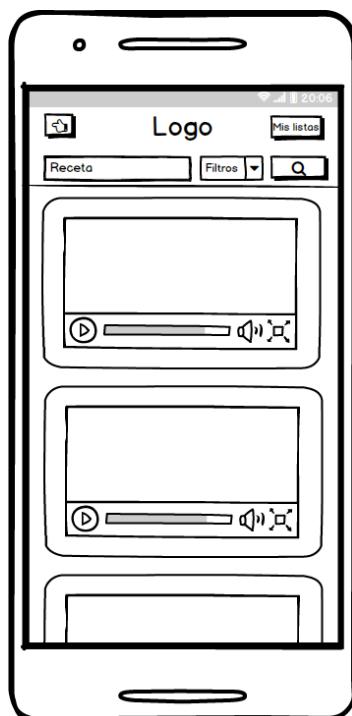
2.10 Vista mobile de videos relacionados con una receta seleccionada



2.11 Vista mobile del formulario de creación de playlist



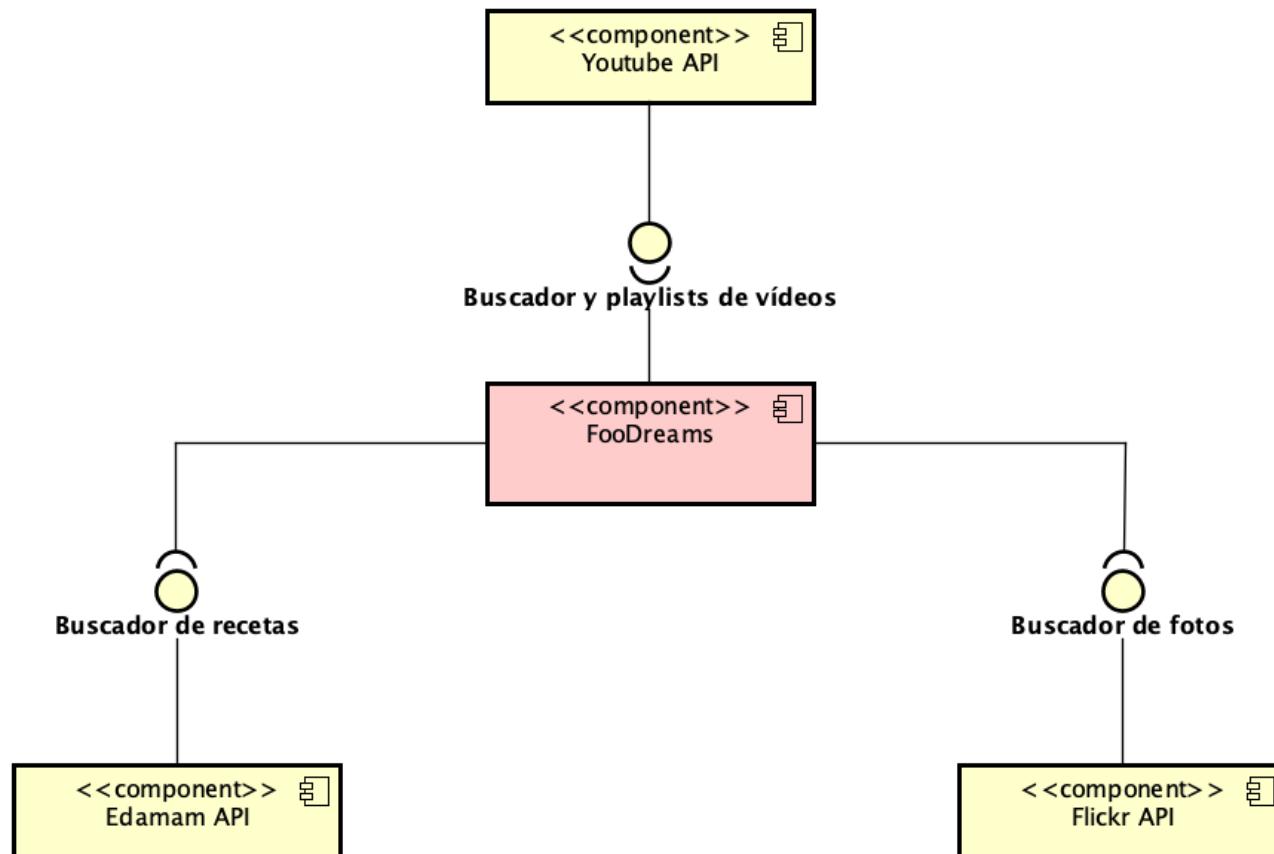
2.12 Vista mobile de playlists creadas por el usuario



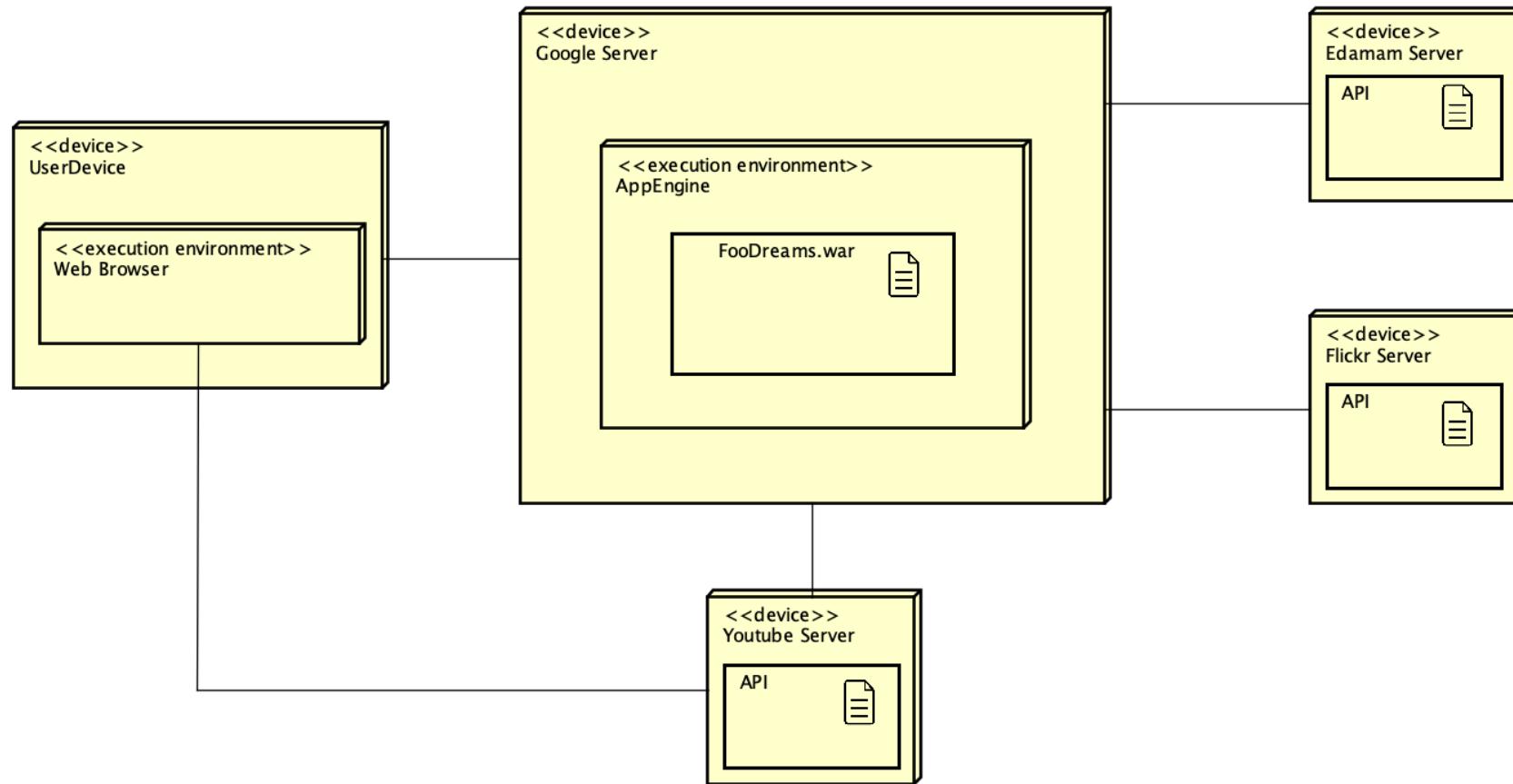
3 Arquitectura

3.1 Diagrama de componentes

La aplicación de FooDreams requiere los servicios de búsqueda y de playlists de vídeos de la API de Youtube, los servicios de buscador de fotos que proporciona la API de Flickr, y el servicio de búsqueda de recetas e información nutricional de la API de Edamam.



3.2 Diagrama de despliegue



El despliegue de la aplicación se realiza a través de la plataforma como servicio AppEngine de Google Cloud Platform. Dicho servicio está alojado en los servidores de Google, por lo que la aplicación es desplegada en un servicio cloud.

A través de dicho servicio se hacen las distintas llamadas a las APIs que se requieran en todo momento, según la operación que se lleve a cabo.

Además, se destaca la conexión que hay entre la API de Youtube y el propio navegador del cliente, ya que se hace uso de la API Javascript de Youtube para poder incrustar el reproductor en el código HTML y poder visualizar los vídeos desde la aplicación.

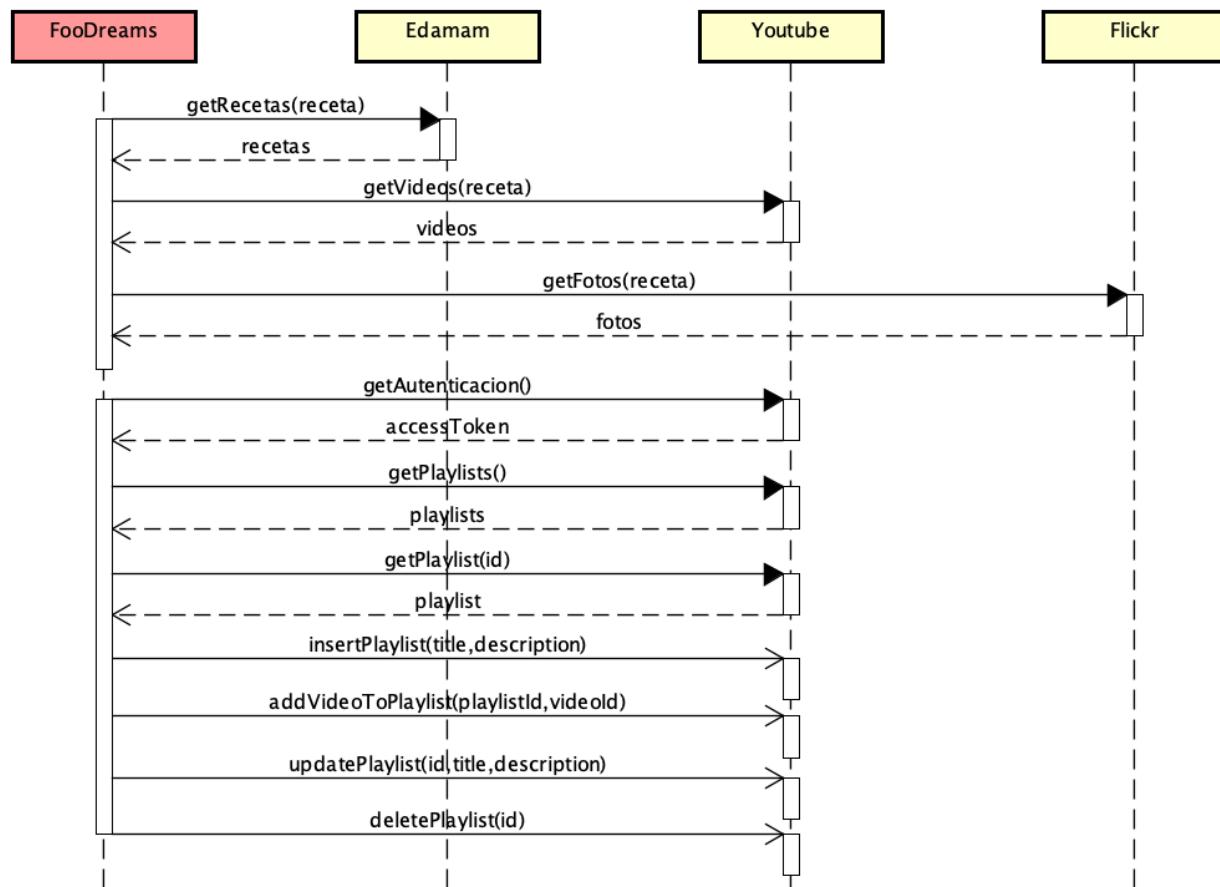
3.3 Diagrama de secuencia de alto nivel

Como se observa en el diagrama, el flujo de trabajo de FooDreams comienza con la búsqueda de una receta por parte del usuario. Se requiere, en este punto, el servicio de Edamam, el cual facilita los resultados de recetas.

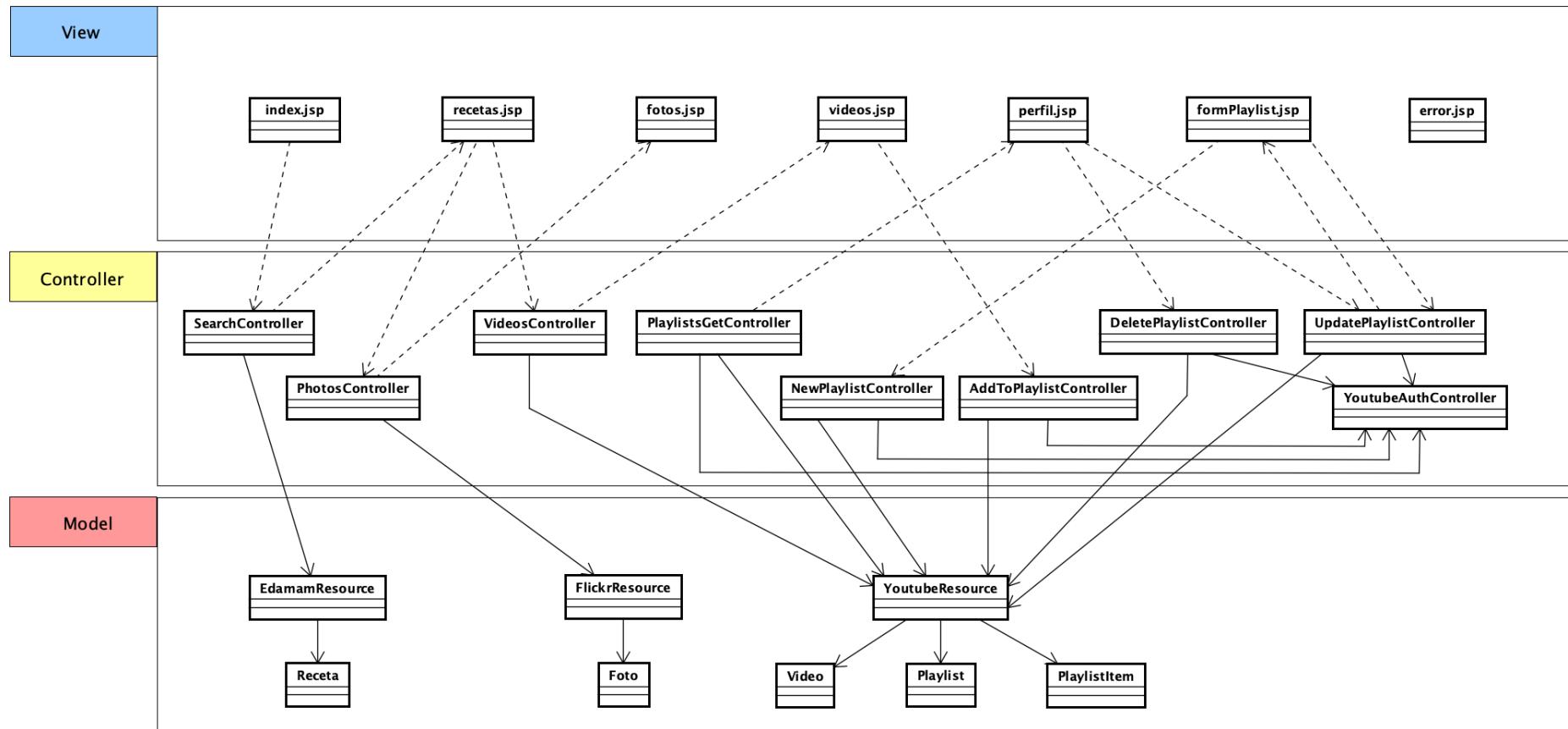
Acto seguido, por cada receta, se realizan búsquedas a través de la API de Youtube, de manera que se obtienen videos específicos para cada uno de los resultados ofrecidos por la API de Edamam.

De forma paralela, y bajo los mismos criterios de búsqueda del usuario, desde FooDreams se solicita al servicio de búsqueda de Flickr una serie de fotos relacionadas con la receta.

Por último, el usuario podrá listar playlists de videos, crearlas, editarlas y eliminarlas, haciendo uso de la API de Youtube, y añadirles los videos que más le puedan interesar de los mostrados tras la búsqueda.



3.4 Diagrama de clases

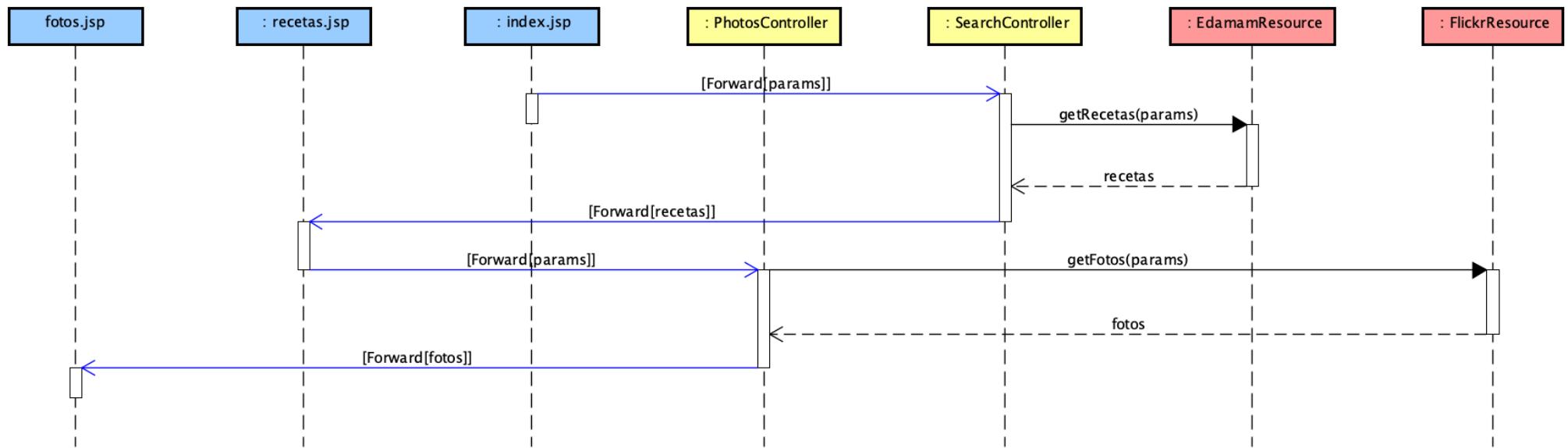


El mashup se estructura en una serie de vistas principales: index.jsp (página principal de búsqueda), recetas.jsp (resultados de Edamam API), fotos.jsp (resultados de Flickr API) y videos.jsp (resultados de Youtube API). Además, cuenta con el listado de playlists del usuario en perfil.jsp y un formulario para la creación y edición de playlists, formPlaylists.jsp.

En cuanto a los controladores, existe uno para cada acción: búsqueda de recetas en Edamam, fotos en Flickr, vídeos en Youtube y las operaciones CRUD sobre el recurso Playlist. Los recursos del mashup son: recetas, fotos, vídeos, playlists y vídeos de playlists (PlaylistItem).

3.4.1 Diagramas de secuencia

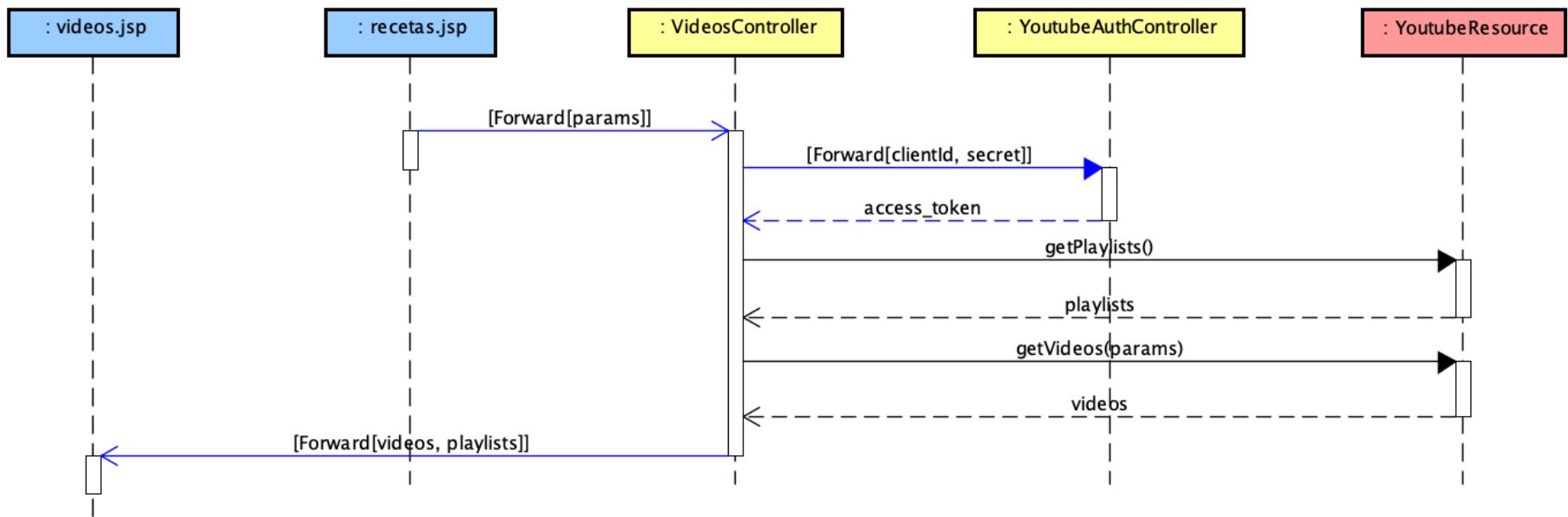
3.4.2 Búsqueda de información e imágenes de recetas



Aunque la búsqueda puede comenzarse desde cualquier búsqueda, ya que el formulario está presente en el header, el flujo de búsqueda principal parte de la vista `index.jsp`. El controlador `SearchController` se encarga de tratar la petición al recurso Receta de Edamam. Una vez se obtienen las recetas, el controlador las envía a la vista `recetas.jsp`, donde son mostradas en el documento.

Una vez en la vista `recetas.jsp`, es posible solicitar las fotos, las cuales se buscarán bajo el mismo parámetro que las recetas. En este caso, el controlador `PhotosController` se encarga de tratar la petición al recurso Foto de Flickr. Una vez se obtienen las fotos, el controlador las envía a la vista `fotos.jsp`, donde son mostradas en el documento.

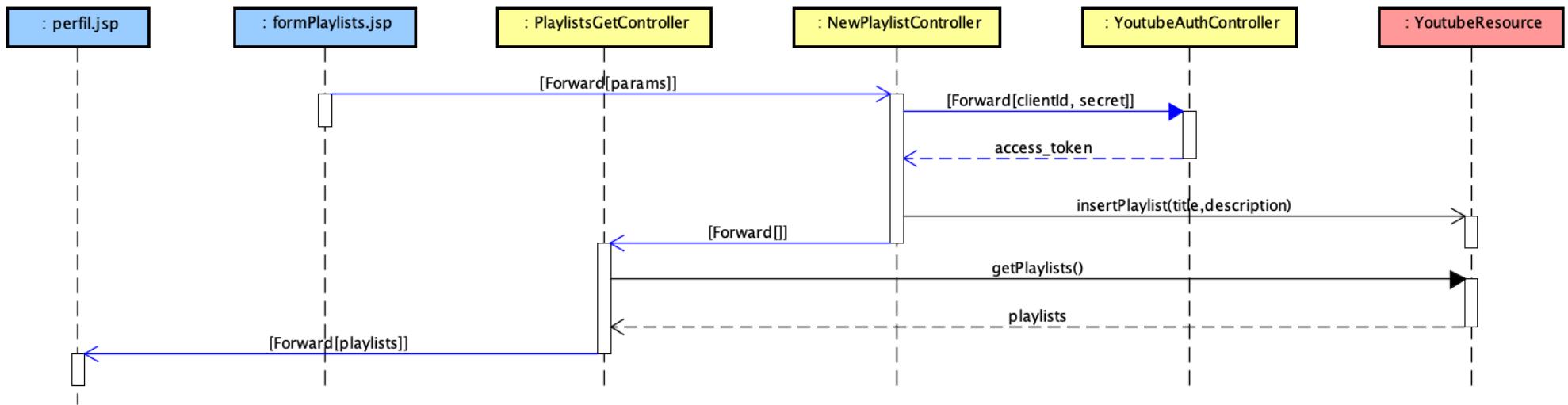
3.4.3 Búsqueda de vídeos de recetas



Partiendo de la vista `recetas.jsp`, el usuario puede acceder a una búsqueda por cada receta encontrada por la API de Edamam. El criterio de búsqueda en Youtube será el nombre de cada receta. Esta búsqueda será tratada por el controlador `VideosController`, el cual requiere la autenticación del usuario mediante OAuth 2.0, ya que se necesitarán listar sus playlists en la vista de resultados de vídeos para poder elegir la playlist a la cual añadir vídeos de los ofrecidos por la búsqueda.

Una vez que se obtienen tanto las playlists del usuario como los vídeos de la búsqueda, el controlador envía todos los datos a la vista `videos.jsp`, donde se mostrará la información al completo.

3.4.4 Creación de playlist de recetas

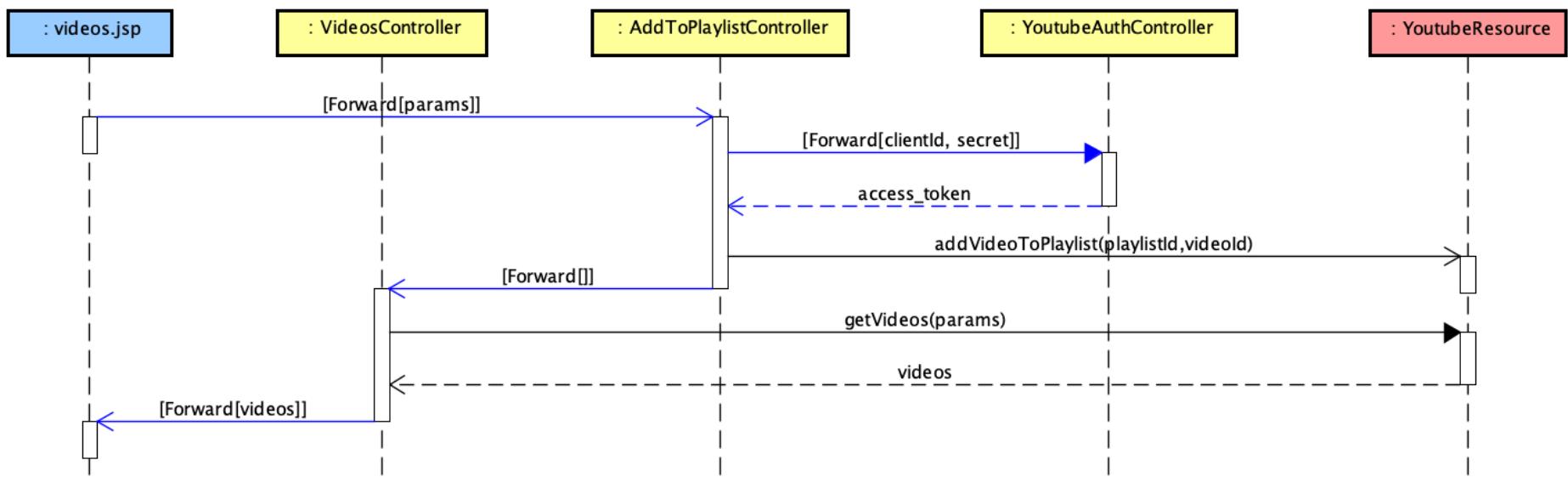


A través del formulario de la vista `formPlaylists.jsp`, donde se va a evaluar si se llega para una edición o, en cambio, para la creación de una nueva playlist, se pasan los parámetros al controlador `NewPlaylistController`. Será necesaria la autenticación OAuth 2.0.

Una vez autenticado, el controlador realiza la operación de inserción, creando así la nueva playlist y redirigiendo el flujo al controlador encargado de listar las playlists, `PlaylistsGetController`.

Por último, cuando el controlador recibe las playlists del recurso, las envía a la vista `perfil.jsp`, donde ya se mostrará la nueva playlist creada.

3.4.5 Añadir vídeos a playlist de recetas

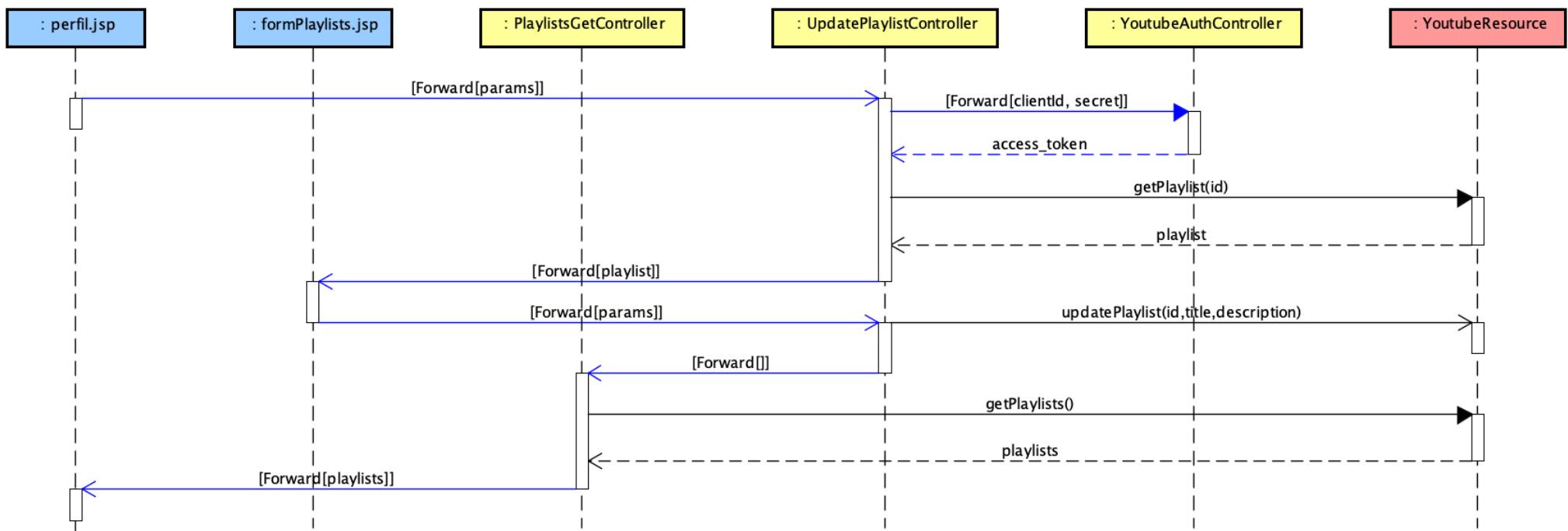


Cuando se tienen los resultados de vídeos de Youtube, el usuario tiene un pequeño formulario por cada resultado, el cual consiste en un select con sus playlists creadas y un botón de añadir. De este modo, haciendo click en el botón, el vídeo se añade en la playlist seleccionada.

Para ello, desde la vista `videos.jsp` se pasan los parámetros al controlador `AddToPlaylistController` tanto los datos del vídeo como los de la búsqueda misma, para poder regresar a la vista de partida. El controlador requerirá autenticación OAuth 2.0. Una vez autenticado, realizará la operación de adición al recurso.

Acto seguido redirige al controlador de búsqueda de vídeos, `VideosController`, el cual tomará de nuevo el parámetro de búsqueda y listará los resultados que son enviados a la vista de partida, `videos.jsp`. De esta manera se permite al usuario continuar con la navegación y seguir añadiendo vídeos de su interés.

3.4.6 Editar playlist de recetas

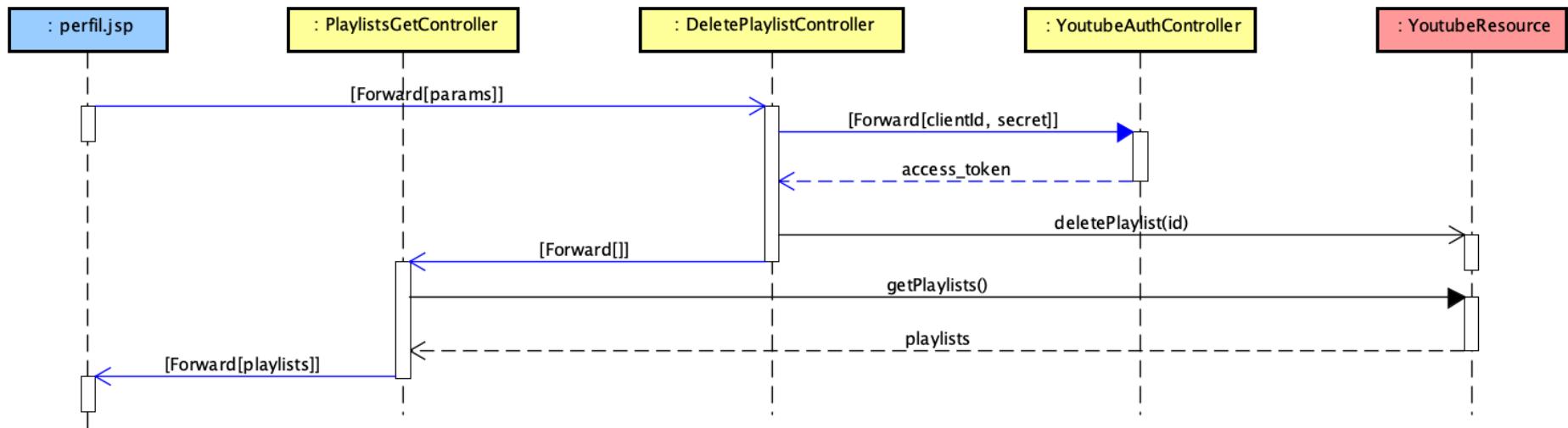


En la vista de `perfil.jsp`, el usuario obtiene todas sus playlists. Haciendo click en el botón de editar de una de ellas, se enviarán los parámetros de esta al controlador `UpdatePlaylistController`, el cual necesitará autenticación OAuth 2.0. Una vez autenticado, obtendrá la playlist a editar, enviando los datos al formulario de la vista `formPlaylists.jsp`.

Llegados al punto de encontrarnos en la vista del formulario, desde aquí se evalúa afirmativamente de que se trata de una edición, ya que se ha recibido una playlist. De este modo, los datos de la playlist aparecerán en el formulario, pudiendo editarse.

Cuando se hace click en el botón de guardar, los nuevos datos pasan de nuevo al controlador `UpdatePlaylistController`, el cual ejecutará el método `update` del recurso y redirigirá al controlador `PlaylistsGetController`, el cual solicitará todas las playlists del usuario de nuevo y las reenviará a la vista de `perfil.jsp`, donde se encontrará ya la playlist editada, entre las demás.

3.4.7 Eliminar playlist de recetas



El flujo que sigue la eliminación de playlists es muy parecido al que se sigue en el caso de la actualización. Partiendo desde la vista de perfil.jsp, haciendo click en el botón eliminar, se envía el parámetro de playlist al controlador DeletePlaylistController, el cual necesita autenticación OAuth 2.0.

Una vez autenticado, se ejecuta la eliminación del recurso y se redirige el flujo hacia el controlador PlaylistsGetController. Este controlador solicitará las playlists del usuario y enviará los recursos de nuevo a la vista de perfil.jsp.

En la vista perfil.jsp se obtienen las playlists del usuario menos la que acaba de ser eliminada.

4 Implementación

A continuación, en esta sección se describen las principales características de la implementación los aspectos que se consideran a destacar de esta.

En el directorio *main* del proyecto, quedan estructuradas las carpetas que componen la totalidad de la implementación del mashup.

En el directorio *java* se implementa la lógica de backend, en lenguaje Java. Aquí están los diferentes controladores y los modelos en los que se basa el funcionamiento de la aplicación y la API REST de FooDreams.

Por otro lado, en el directorio *webapp* queda reservada para el frontend del mashup. Aquí están las diferentes vistas que componen la aplicación, el código CSS, así como el código de librerías de terceros y recursos como las imágenes de logotipo.

Pasando a los aspectos a destacar en la implementación de la aplicación, en primer lugar cabe resaltar el uso del framework CSS *Bootstrap*, el cual permite aplicar estilos de forma sencilla y adaptable, por tanto, su uso ha simplificado el proceso de maquetación y ha ayudado a agilizar el desarrollo del proyecto. Además, proporciona un diseño adaptable que se ajusta a cualquier dispositivo y resolución de pantalla.

Partiendo de *Bootstrap* como base, se han creado los estilos más específicos que dan personalidad al mashup en diferentes hojas de estilo.

En segundo lugar, se ha hecho uso de la librería *C3* de JavaScript, con la que se generan gráficos y se representan datos de manera visual. Esta funcionalidad se ha utilizado para mostrar una gráfica por cada resultado de la búsqueda de recetas, sirviéndose como fuente de datos de los valores nutricionales de los diferentes nutrientes de cada una de las recetas. De manera breve y descriptiva, se expone el uso de esta librería:

Para cada resultado de la búsqueda de recetas se genera un id aleatorio, que lo hace único. De manera iterativa, se guardan los valores de los distintos nutrientes de cada receta en inputs ocultos. Estos valores son el resultado de dividir el total de dichos nutrientes (dato que viene expresado para la totalidad de la elaboración de la receta) entre el número de comensales para el cual está dirigida la receta. Así se obtiene el dato de los nutrientes por persona. Tras esto, con JavaScript, se selecciona el lugar donde se muestra la gráfica con un selector (por el id único) y se almacenan los datos de cada input en una variable. Finalmente, con el uso de la librería *C3*, se genera una gráfica a la cual se le pasan, mediante un esquema *JSON*, el selector ya mencionado, los datos de las variables y otras opciones de customización y visualización de los resultados.

En tercer lugar, se han implementado las operaciones *CRUD* del recurso Playlist de la API REST de Youtube, ofreciendo al usuario del mashup la posibilidad de crear playlists, editarlas, eliminarlas y añadirles vídeos. Para que las playlists que se muestran al usuario desde la aplicación sean única y exclusivamente las relacionadas con

FooDreams, internamente se ha agregado un prefijo que se añade por defecto al crear una playlist (“fooDreams – ”). Posteriormente, a la hora de requerir las playlists sólo se listan las que corresponden al mashup, siendo todo esto transparente para el usuario. De esta manera se logra una mayor concordancia.

Finalmente, destaca la documentación de la API REST de FooDreams. Para ello se ha definido un documento en lenguaje *YAML* con el que, gracias a la herramienta *Swagger*, se obtiene un diseño estandarizado que permite una visualización y testeo de la API de forma sencilla y accesible.

5 Pruebas

Para la realización de las pruebas de integración que ayuden a detectar posibles errores en la implementación y desarrollo del mashup se han ejecutado pruebas unitarias, que testean el funcionamiento de los métodos encargados de generar las peticiones a las diferentes APIs que se consumen.

La estrategia que se ha seguido ha sido la de pruebas incrementales. Conforme se llevaba a cabo la implementación de los determinados componentes que hacían consumo de las APIs, se generaban los tests pertinentes.

Resumen	
Número total de pruebas realizadas	7
Número de pruebas automatizadas con JUnit	6 (86%)

ID	Test Edamam 1
Descripción	Prueba para la detección de errores al implementar búsquedas en Edamam haciendo uso de su API, aplicando filtros de tipo de dieta y número máximo de ingredientes.
Entrada	Se hace uso de la librería Reslet para invocar al servicio usando la URI <code>https://test-es.edamam.com/search?app_id=EDAMAM_APP_ID&app_key=EDAMAM_API_KEY&to=10&q=RECETA&diet=DIETA&ingr=MAX</code> desde nuestra aplicación.
Salida esperada	Los datos devueltos en formato JSON son mapeados a la clase Java RecetaSearch.
Resultado	EXITO
Automatizada	Sí

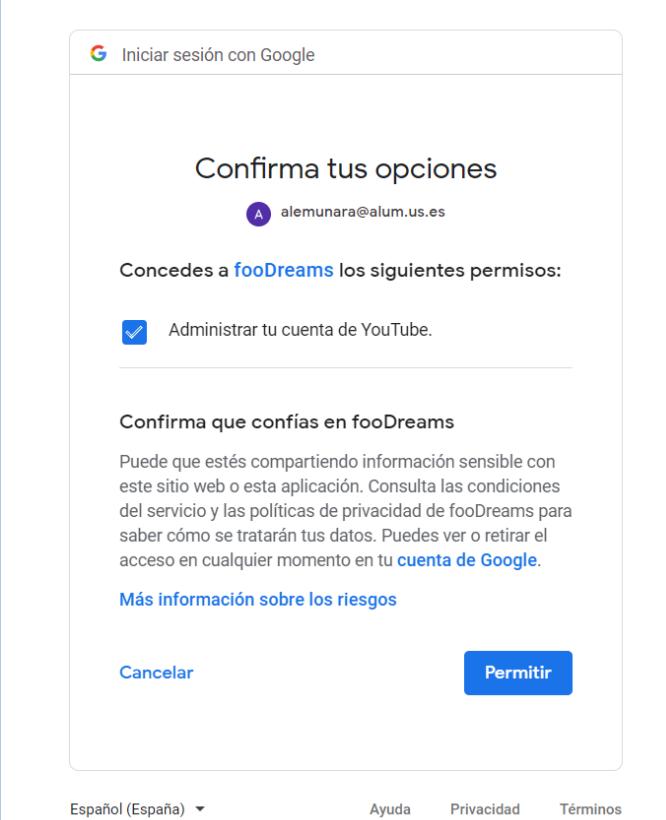
ID	Test Edamam 2
Descripción	Prueba para la detección de errores al implementar búsquedas en Edamam haciendo uso de su API, aplicando filtro de tipo de dieta.
Entrada	Se hace uso de la librería Reslet para invocar al servicio usando la URI <code>https://test-es.edamam.com/search?app_id=EDAMAM_APP_ID&app_key=EDAMAM_API_KEY&to=10&q=RECETA &diet=DIETA</code> desde nuestra aplicación.
Salida esperada	Los datos devueltos en formato JSON son mapeados a la clase Java RecetaSearch.
Resultado	EXITO
Automatizada	Sí

ID	Test Edamam 3
Descripción	Prueba para la detección de errores al implementar búsquedas en Edamam haciendo uso de su API, aplicando filtro de número máximo de ingredientes.
Entrada	Se hace uso de la librería Reslet para invocar al servicio usando la URI https://test-es.edamam.com/search?app_id=EDAMAM_APP_ID&app_key=EDAMAM_API_KEY&to=10&q=RECETA &ingr=MAX desde nuestra aplicación.
Salida esperada	Los datos devueltos en formato JSON son mapeados la clase Java RecetaSearch.
Resultado	EXITO
Automatizada	Sí

ID	Test Edamam 4
Descripción	Prueba para la detección de errores al implementar búsquedas en Edamam haciendo uso de su API, aplicando filtro de número máximo de ingredientes.
Entrada	Se hace uso de la librería Reslet para invocar al servicio usando la URI https://test-es.edamam.com/search?app_id=EDAMAM_APP_ID&app_key=EDAMAM_API_KEY&to=10&ingr=MAX desde nuestra aplicación.
Salida esperada	Los datos devueltos en formato JSON son mapeados a la clase Java RecetaSearch.
Resultado	EXITO
Automatizada	Sí

ID	Test Flickr 1
Descripción	Prueba para la detección de errores al implementar búsquedas en Flickr haciendo uso de su API.
Entrada	Se hace uso de la librería Reslet para invocar al servicio usando la URI https://api.flickr.com/services/rest/?method=flickr.photos.search&text=TEXT&api_key=FLICKR_API_KEY&per_page=12&tags=comida,food,receta&sort=relevance&content-type=1&format=json&nojsoncallback=1
Salida esperada	Los datos devueltos en formato JSON son mapeados a la clase Java FotoSearch.
Resultado	EXITO
Automatizada	Sí

ID	Test Youtube 1
Descripción	Prueba para la detección de errores al implementar búsquedas en Youtube haciendo uso de su API.
Entrada	Se hace uso de la librería Reslet para invocar al servicio usando la URI https://www.googleapis.com/youtube/v3/search?key=YOUTUBE_API_KEY&part=snippet&q=BUSQUEDA&maxResults=8
Salida esperada	Los datos devueltos en formato JSON son mapeados la clase Java VideoSearch.
Resultado	EXITO
Automatizada	Sí

ID	Test Youtube 2
Descripción	Prueba para la detección de errores al implementar la autenticación OAuth 2.0 a través del servicio ofrecido por la API de Youtube.
Entrada	<p>Se hace uso del código facilitado en la plantilla del proyecto para la autenticación mediante OAuth 2.0. https://accounts.google.com/o/oauth2/v2/auth</p> 
Salida esperada	Se obtiene el token de acceso que habilita para la realización de acciones que requieren autenticación en el servicio.
Resultado	EXITO
Automatizada	No

6 Manual de usuario

6.1 Mashup

Vista principal



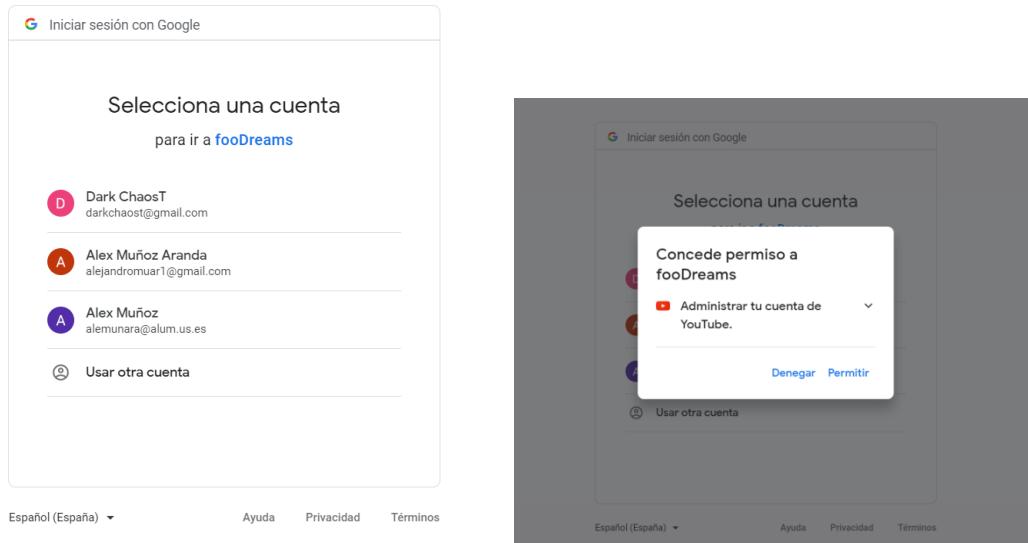
En la vista principal, el usuario se encuentra con un buscador, en el que puede teclear la receta que le interese buscar y también indicar un tipo de dieta concreto al que le gustaría que se adaptaran los resultados, así como establecer un número máximo de ingredientes por receta.

Esta acción de búsqueda desencadena el proceso de autenticación que, aunque directamente esta acción no la requiere, sí es necesario para el grueso de acciones que lleva a cabo la aplicación, por lo que se ha considera conveniente iniciar el flujo de navegación con la autenticación del usuario, obteniéndose así el token de acceso.

El usuario también tiene acceso en la parte inferior de la vista a la documentación de la propia API de la aplicación.

Autenticación

La autenticación produce cuando se realiza la primera búsqueda desde la vista principal. El usuario deberá permitir que la aplicación acceda a su cuenta de YouTube.



Vista de resultado de recetas

The screenshot shows the fooDreams application interface. At the top, there is a navigation bar with a logo, a search bar, a dropdown for diet type, a button for ingredients, a search button, and a button for playlists. Below the navigation bar, a blue button labeled 'Fotos' is visible. The main content area is titled 'Resultados sobre: "Pasta"'. It displays two cards for pasta recipes: 'Pasta quesosísima vegetariana' and 'Pasta con tomate seco'. Each card has tabs for 'Nutrientes' (Nutrients) and 'Ingredientes' (Ingredients). Below the tabs are line graphs showing the nutritional content per 100g. The first graph for 'Pasta quesosísima vegetariana' shows values: Carbohydrates (~42), Azúcares (~1), Grasas (~1), Fibra (~2), and Proteínas (~8). The second graph for 'Pasta con tomate seco' shows values: Carbohydratos (~48), Azúcares (~6), Grasas (~2), Fibra (~2), and Nutrientes (~7).

En la vista de recetas, el usuario dispone de una cabecera común, que está disponible en el resto de las vistas de la aplicación, en la cual puede volver a iniciar un proceso de búsqueda de recetas, ir a sus propias playlists, o bien regresar a la vista principal haciendo click en el logotipo.

Desde aquí también se podrá navegar a las fotografías de recetas similares a las que el usuario buscó y consultar los resultados de la búsqueda principal.

Como se observa en la imagen de arriba, se presentan los resultados en módulos que contienen la gráfica de los valores nutricionales de los ingredientes, en concreto las cantidades de carbohidratos, azúcares, grasas, fibra y proteínas por persona.

Haciendo click en la pestaña “Ingredientes” de cada uno de los módulos de resultados, se muestran los ingredientes utilizados para esa receta concreta, tal como se muestra en la siguiente imagen.

This screenshot shows the 'Ingrediente' (Ingredients) tab for the two pasta recipes from the previous screenshot. The 'Pasta quesosísima vegetariana' card lists ingredients: 1 paquete (12 oz) de conchitas de pasta con queso VELVEETA Shells & Cheese Dinner, 1 taza de verduras mixtas congeladas (brócoli, coliflor y zanahoria), 1/3 taza de pimiento (pimentón) rojo picado, and 1/4 cucharadita de hojas secas de tomillo. The 'Pasta con tomate seco' card lists ingredients: 400 g de papardelle (o pasta al gusto), 100 g de tomate seco, Aceite de oliva, Sal, 10 bolitas de pimienta variada, 2 dientes de ajo, and 1 cucharada de orégano. Both cards also show their respective calorie counts: 212.94622226666664 and 256.1375, and a 'Videos' button.

Además de todo esto, el usuario dispone de un botón, por cada resultado, que lo lleva a buscar vídeos en Youtube para esa receta concreta.

Vista de fotos

The screenshot shows the fooDreams website interface. At the top, there is a navigation bar with a chef's hat icon, the logo 'fooDreams', a search bar ('Buscar otra receta...'), a dropdown menu ('-- Tipo de dieta --'), a 'Ingredientes' button, a 'Buscar' button with a magnifying glass icon, and a 'Mis playlists' button. Below the navigation bar, the text 'Fotos sobre: "Pasta"' is displayed. Three images of pasta dishes are shown: a bowl of spaghetti with a blue and white patterned rim, a bowl of pasta with vegetables and herbs, and a red plate filled with penne pasta and seafood.

Suponiendo que el usuario desea ver fotografías relacionadas con la receta que busca, si hace click en el botón “Fotos” llega a una vista similar a la anterior. Aquí contará con el encabezado común del mashup y un máximo de 12 imágenes sobre la búsqueda realizada, para reforzar la información e ilustrar los platos.

Vista de vídeos

The screenshot shows the fooDreams website interface again. The top navigation bar is identical to the previous one. Below it, the text 'Resultados sobre: "Pasta con tomate seco"' is displayed. Four video thumbnail cards are shown: 1. 'Pasta con tomates secos | Recetas Rockeras' showing a man cooking. 2. 'Pasta con pollo y tomates deshidratados' showing a plate of pasta with the word 'PRETA' overlaid. 3. 'Pesto rojo con tomates secos - Receta italiana d...' showing a close-up of pesto. 4. 'TOMATE SECO | PASTA AND ROLL' showing a plate of pasta.

Si por el contrario el usuario prefiere ir a vídeos que estén relacionados concretamente con un de los resultados de la búsqueda, haciendo click en el botón de “Vídeos”, accede a una vista como la anterior.

Aquí quedan listados los resultados en módulos en los que se podrá seleccionar alguna playlists a la que añadir los vídeos que más gusten al usuario.

A continuación, se pasa a la parte más personalizada por usuario del mashup.

Vista de playlists de usuario

The screenshot shows the fooDreams website interface. At the top, there is a navigation bar with a search bar ('Buscar otra receta...'), a dropdown for 'Tipo de dieta', a 'Ingredientes' button, a 'Buscar' button with a magnifying glass icon, and a 'Mis playlists' button with a YouTube icon. Below the navigation bar, the page title 'Mis playlists' is displayed. A blue button labeled '+ Nueva playlist' is located at the top right of the playlist area. Two playlists are listed: 'fooDreams - Ensaladas' featuring a Greek salad video thumbnail, and 'fooDreams - Pizzas' featuring a pepperoni pizza video thumbnail. Each playlist card includes an 'Editar' (Edit) button and an 'Eliminar' (Delete) button.

En la vista de playlists, el usuario dispone de todas las listas de reproducción que haya creado desde FooDreams. Se le da también la posibilidad de editar tanto el título como la descripción de la lista, así como la oportunidad de borrarla. Además, existe un botón en la parte superior para que pueda crear nuevas playlists.

Vista de creación de playlist

The screenshot shows the fooDreams website interface for creating a new playlist. At the top, there is a navigation bar with a search bar ('Buscar otra receta...'), a dropdown for 'Tipo de dieta', a 'Ingredientes' button, a 'Buscar' button with a magnifying glass icon, and a 'Mis playlists' button with a YouTube icon. Below the navigation bar, the page title 'Nueva playlist' is displayed. The form fields for creating a new playlist are shown: 'Título' (Title) with an input field containing a placeholder, 'Descripción' (Description) with an input field containing a placeholder, and a blue 'Crear playlist' (Create playlist) button with a play icon.

En esta vista el usuario introduce el nombre que desea que tenga su nueva playlist y alguna pequeña descripción o anotación que quiera darle.

Acto seguido a la creación, esta playlist estará disponible en su perfil de playlists y lista para recibir vídeos.

Vista de edición de playlist

The screenshot shows the fooDreams website interface. At the top, there is a navigation bar with a logo featuring a chef's hat and the text "fooDreams". To the right of the logo are search and filter options: "Buscar otra receta...", "-- Tipo de dieta --", "Ingredientes", a search button with a magnifying glass icon, and a "Mis playlists" button.

The main content area is titled "Editar playlist" (Edit Playlist). It contains two input fields: "Título" (Title) with the value "fooDreams - Ensaladas" and "Descripción" (Description) with the value "Mejores y variadas ensaladas". Below these fields is a blue "Guardar" (Save) button with a small icon.

6.2 Documentación de API REST

FooDreams puede implementar una API REST que ofrece una serie de servicios para diferentes recursos.

URI de la API: <http://foodreams.appspot.com/api>

La documentación web completa puede consultarse [aquí](#).

6.3 Recurso ingrediente

HTTP	URI	Descripción
GET	/ingredientes	Devuelve todos los ingredientes de la aplicación.
GET	/ingredientes/{ingrID}	Devuelve el ingrediente con identificador único {ingrID}. Si el ingrediente no existe, devuelve un código de estado de error 404 “Not Found”.
POST	/ingredientes	Añade un nuevo ingrediente. Los datos del ingrediente se deben pasar en el cuerpo de la petición en formato JSON. En caso de que [datos] no sean válidos, devuelve un código de estado de error 400 “Bad Request”. En caso de que el ingrediente se añada de forma exitosa, devuelve un código de estado 201 “Created” con la referencia a la URI y al contenido del nuevo ingrediente.
PUT	/ingredientes	Actualiza los datos del ingrediente cuyos datos se pasan por el cuerpo de la petición en formato JSON, incluyendo el identificador único “ingrID” como criterio de selección del recurso a actualizar. Si el ingrediente no existe, devuelve un código de estado de error 404 “Not Found”. Si la actualización se realiza de forma exitosa, devuelve un código de estado 204 “No Content”.
DELETE	/ingredientes/{ingrID}	Elimina el ingrediente con identificador único {ingrID}. Si el ingrediente no existe, devuelve un código de estado de error 404 “Not Found”. Si la eliminación se realiza de forma exitosa, devuelve un código de estado 204 “No Content”.

Un ingrediente tiene un identificador único, un nombre, cantidad de ingrediente utilizado, unidad de medida de dicha cantidad y calorías aportadas. A continuación, se muestra un ejemplo de la representación JSON del recurso:

```
{
  "id": "5",
  "nombre": "Garbanzos",
  "cantidad": "300.0",
  "unidad": "gramos",
  "calorías": "540.0"
}
```

6.4 Recurso receta

HTTP	URI	Descripción
GET	/recetas	Devuelve todas las recetas de la aplicación.
GET	/recetas/{recetaID}	Devuelve la receta con identificador único {recetaID}. Si la receta no existe, devuelve un código de estado de error 404 “Not Found”.
POST	/recetas	Añade una nueva receta. Los datos de la receta se deben pasar en el cuerpo de la petición en formato JSON. En caso de que [datos] no sean válidos, devuelve un código de estado de error 400 “Bad Request”. En caso de que la receta se añada de forma exitosa, devuelve un código de estado 201 “Created” con la referencia a la URI y al contenido de la receta.
POST	/recetas/{recetaID}/{ingrID}	Añade el ingrediente con identificador único {ingrID} a la receta con identificador único {recetaID}. En caso de que la receta o el ingrediente no existan, devuelve un código de estado de error 404 “Not Found”. En caso de que el ingrediente ya esté incluido en la receta, devuelve un código de estado de error 400 “Bad Request”. En caso de que la inserción se realice de forma exitosa, devuelve un código de estado 201 “Created” con la referencia a la URI y el contenido de la receta.
PUT	/recetas	Actualiza los datos de la receta cuyos datos se pasan por el cuerpo de la petición en formato JSON, incluyendo el identificador único “recetaID” como criterio de selección del recurso a actualizar. Si la receta no existe, devuelve un código de estado de error 404 “Not Found”. Si la actualización se realiza de forma exitosa, devuelve un código de estado 204 “No Content”.

DELETE	/recetas/{recetaID}	Elimina la receta con identificador único {recetaID}. Si la receta no existe, devuelve un código de estado de error 404 “Not Found”. Si la eliminación se realiza de forma exitosa, devuelve un código de estado 204 “No Content”.
DELETE	/recetas/{recetaID}/{ingrID}	Elimina el ingrediente con identificador único {ingrID} de la receta con identificador único {recetaID}. En caso de que la receta o el ingrediente no existan, devuelve un código de estado de error 404 “Not Found”. En caso de que la eliminación se realice de forma exitosa, devuelve un código de estado 204 “No Content”.

Una receta tiene un identificador único, una fecha de publicación, un nombre, descripción, recurso fotográfico, número de comensales, listado de enumerado de alérgenos y listado de ingredientes. A continuación, se muestra un ejemplo de la representación JSON del recurso:

```
{
  "id": "3",
  "fechaPublicacion": "2018-03-02",
  "nombre": "Cocido madrileño",
  "descripcion": "Remojar los garbanzos la noche anterior en agua templada con un poco de sal. A la mañana siguiente, sacarlos y escurrirlos. Colocar en una cazuela grande...",
  "foto": "http://foodreams.appspot.com/media/cocido-madrid-2018-03-02.jpg",
  "nComensales": "6",
  "alergenos": ["huevo", "gluten"],
  "ingredientes": [
    {
      "id": "5",
      "nombre": "Garbanzos",
      "cantidad": "300.0",
      "unidad": "gramos",
      "calorías": "540.0"
    },
    {
      "id": "6",
      "nombre": "Tocino",
      "cantidad": "200.0",
      "unidad": "gramos",
      "calorías": "1082.0"
    }
  ]
}
```

7 Referencias

Balsamiq. <http://balsamiq.com/>. Accedido en Mayo 2019.

Bootstrap. <https://getbootstrap.com/>. Accedido en Mayo 2019

C3.js <https://c3js.org/>. Accedido en Mayo 2019