(BORRADOR)

iespereMariaOrts

IES PERE MARIA ORTS I BOSCH
CICLOS FORMATIVOS DE GRADO SUPERIOR

Familia Profesional: Informática y Comunicaciones

Sei Vegan, Sei Grün

Alumno: Alejandro Martínez Morillo

(BORRADOR)

IES PERE MARIA ORTS I BOSCH
CICLOS FORMATIVOS DE GRADO SUPERIOR

Familia Profesional: Informática y Comunicaciones

PROYECTO DE FIN DE CICLO
Desarrollo de Aplicaciones Multiplataforma

Sei Vegan, Sei Grün

AUTOR: Alejandro Martínez Morillo
TUTOR: Jose Caturla Palao

Lugar y fecha.

(PÁGINA PARA AGRADECIMIENTOS, SI SE INCLUYE)

Tabla de contenido

Resumen

Este documento representa la creación y el desarrollo del proyecto final de curso del Ciclo Superior de Desarrollo de Aplicaciones Multiplataforma.

Para la creación del mismo, se han creado distintos elementos ficticios para que el proyecto sea lo más cercano al mundo laboral. Como primer elemento, se ha creado una sociedad limitada de un único socio llamada *SoftDev*. En el apartado 2 de este mismo proyecto, observaremos el proceso completo de lo que es crear una sociedad limitada con unas características concretas. Además, el segundo ficticio al cuál está ligado es el restaurante el cuál necesita nuestros servicios, *Grün.* Este restaurante aporta una oferta vegana al mercado, la cuál es escasa actualmente y refleja unos valores que el creador de dicho proyecto defiende.

Apartando el tema empresarial y económico, en este documento veremos reflejado el desarrollo de 3 aplicaciones, centradas en distintos usuarios finales.

El primer paso de todos fue analizar el proyecto a grandes rasgos y ver cómo enfocar el desarrollo del mismo. Un diagrama de *Gantt* inicial o analizar y crear los posibles casos de usos de las aplicaciones son algunos de los primeros pasos que se deben hacer.

Ya analizada la aplicación y observar las consideraciones a tener en cuenta, llega el momento de preparar el entorno de programación y sus librerías, el control de versiones de los proyectos y el estudio de nuevas tecnologías que se pondrán en uso.

Por último, podremos observar distintos apartados destacables de la aplicación, ya sean por su complejidad o por lo interesante y útil que ha resultado implementar dicho código. Además, pondremos a prueba la aplicación utilizando los casos de uso creados con anterioridad.

Abstract

This document represents the creation and development of the final project of the Higher Cycle of Multiplatform Applications Development.

For the creation of the project, different fictitious elements have been created to make the project as close as possible to the working world. As a first element, a single-partner limited company called SoftDev has been created. In section 2 of this project, we will observe the complete process of creating a limited company with specific characteristics. In addition, the second fictitious company to which it is linked is the restaurant that needs our services, Grün. This restaurant brings a vegan offer to the market, which is currently scarce and reflects the values that the creator of this project defends.

Apart from the business and economic issue, in this document we will see the development of 3 applications, focused on different end users.

The first step of all was to analyse the project in broad strokes and see how to approach its development. An initial Gantt chart or analysing and creating the possible use cases of the applications are some of the first steps to be taken.

Once the application has been analysed and the considerations to be taken into account have been observed, it is time to prepare the programming environment and its libraries, the version control of the projects and the study of new technologies that will be put into use.

Finally, we will be able to observe the different sections of the application, either for their complexity or for how interesting and useful it has been to implement the code. In addition, we will test the application using the use cases created previously.

1. Introducción

Sei Vegan, Sei Grün es un proyecto realizado por la empresa SoftDev, una empresa de desarrollo software que creará y desplegará las aplicaciones que necesita el restaurante Grün, centrado en la comida vegana.

La representación de este proyecto se divide en tres partes, una centrada en el uso del administrador, una segunda centrada en el uso de los trabajadores, tanto los cocineros como los camareros, y una tercera y última parte centrada en los clientes.

En la primera aplicación, centrada en la administración, el dueño o la persona designada en la gestión de recursos podrá encargarse de añadir, borrar y actualizar los datos del restaurante, ya sean los datos relacionados con los trabajadores como los relacionados con los productos del negocio.

Por otro lado, la segunda aplicación que se desarrollará para este negocio estará centrada en el uso de los trabajadores a la hora de atender a los clientes en el local, pensada para ser utilizada en un teléfono móvil o tablet. Por un lado, los camareros tendrán acceso a un TPV para crear y gestionar los pedidos realizados en el local, ya sean para degustarlos en el lugar o para llevar. Por otro lado, los cocineros tendrán una vista, pensada para que se utilice en una tablet o AIO (All in one). En este apartado, los cocineros podrán ver todos los pedidos, junto a sus detalles, y marcarlos como “Finalizados”.

Por último, se desarrollará la aplicación centrada en el uso del cliente. En dicho programa, los usuarios tendrán dos acciones principales: visualizar el menú y crear un pedido. La primera opción está diseñada y orientada para el uso en el local, de esta manera los clientes pueden visualizar el menú en cualquier momento y no tendrán por qué esperar a que un camarero les entregue la carta físicamente. La segunda función está centrada en el uso desde el domicilio. Desde la comodidad de su casa, podrán pedir cualquier producto de la carta para que lo entreguen en su domicilio.

1.1. Objetivos iniciales

A la hora de presentar la propuesta de proyecto, se estipularon una serie de objetivos que, como veremos más adelante, se cumplen. A grandes rasgos, el objetivo principal del proyecto es desarrollar tres tipos de aplicaciones, las cuáles han sido desarrolladas *grosso modo*. Para la parte administrativa, tenemos como objetivos desarrollar un CRUD (*Create*, *Read*, *Update Delete*) completamente funcional.

Por otro lado, a la hora de desarrollar la aplicación para los trabajadores, concretamente para los camareros, el objetivo es crear distintas funciones, las cuales son:

- ⇒ Creación de pedidos y pagos.
- ⇒ Creación de facturas.
- ⇒ Diseño de un menú e implementarlo en la aplicación actual.

En cambio, en la vista de los cocineros tendremos una única función; visualizar los pedidos y todos sus detalles.

Como objetivo adicional, se desea implementar un *socket* para que dichas aplicaciones sean en tiempo real y que los cocineros no tengan la necesidad de darle un botón para recibir los nuevos pedidos

La tercera aplicación que se desea desarrollar tiene como función principal la de crear pedidos a domicilio. También se quiere desarrollar una vista con el menú completo con el objetivo de ayudar al usuario final en la elección de sus productos. Asimismo, como en la anterior aplicación, se intentará implementar un *socket* para que sea una aplicación en tiempo real, facilitando tanto la llegada de los pedidos como las actualizaciones del mismo.

Como último objetivo prioritario de este proyecto, se desarrollará un software intermediario que tiene como finalidad la comunicación entre aplicaciones y la base de datos. En ella, se desarrollarán distintas funciones que ayudaran en la comunicación del software anteriormente nombrado.

Por último, como objetivo ajeno a la programación en sí, se desea diseñar y recrear el menú del restaurante y realizar las fotografías correspondientes para utilizarlas en las distintas aplicaciones.

1.2. Objetivos cumplidos

1.3. Justificación del proyecto

La decisión de desarrollar este proyecto tiene un trasfondo totalmente personal. Mis padres, desde pequeño, me han transmitido un amor y una pasión por la comida que, a día de hoy, sigo manteniendo. Además, siempre me han ido enseñando a cocinar, desde una simple merienda hasta algún plato que pueda alimentar a una familia entera.

Por otro lado, otra parte característica de este proyecto es que mantenga una dieta vegana. Desde hace unos años, mi curiosidad sobre este tipo de dietas ha ido en auge, siendo el confinamiento uno de los momentos en los que más me centré en ello. Desde entonces, he descubierto un mundo culinario muy interesante, tanto como plantear llevar este tipo de dieta, aunque no es correcto llamarlo de tal manera.

En conclusión, he juntado dos facetas que me hacen feliz, la programación y la cocina, junto a un mundo culinario que no está para nada explotado. Además de crearme un reto en cuánto a la programación, decidí conjuntarlo con la creación de un menú propio para fusionar al máximo posible estas dos facetas propias.

2. Estudio económico

3. Análisis

En el siguiente apartado nos centraremos en el análisis previo que ha de hacerse siempre a la hora de desarrollar un proyecto.

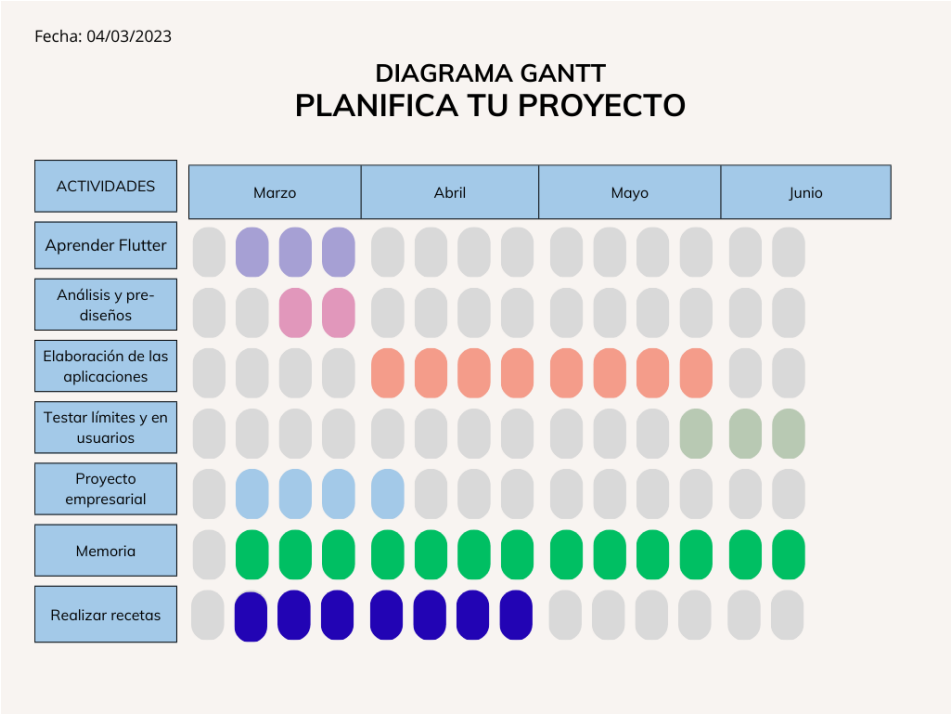
A continuación, desarrollaremos distintos puntos, desde una planificación inicial, con el Diagrama de Gantt, hasta casos de usos y diagrama de clases.

3.1. Planificación temporal

La planificación temporal es un apartado más que importante a la hora de desarrollar un proyecto de cualquier tipo. De esta manera, podemos tener un control sobre las acciones que realizamos durante el mismo. Además, realizar esta planificación nos puede ayudar en futuros proyecto a la hora de organizarlo mejor en el tiempo.

Uno de los procedimientos más conocidos y utilizados en este ámbito es el "Diagrama de Gantt". Este diagrama te proporciona una vista general de las tareas programadas. Además, muestra datos de interés como la fecha de inicio y final del proyecto, en cuántas tareas está dividido el proyecto, una estimación aproximada de cuánto tiempo se llevará a cabo en cada tarea y la manera en la que las tareas están relacionadas entre sí.

A continuación, mostraré el diagrama de Gantt realizado al comienzo de este proyecto.



COLOCAR EL DIAGRAMA DE GANTT FINAL

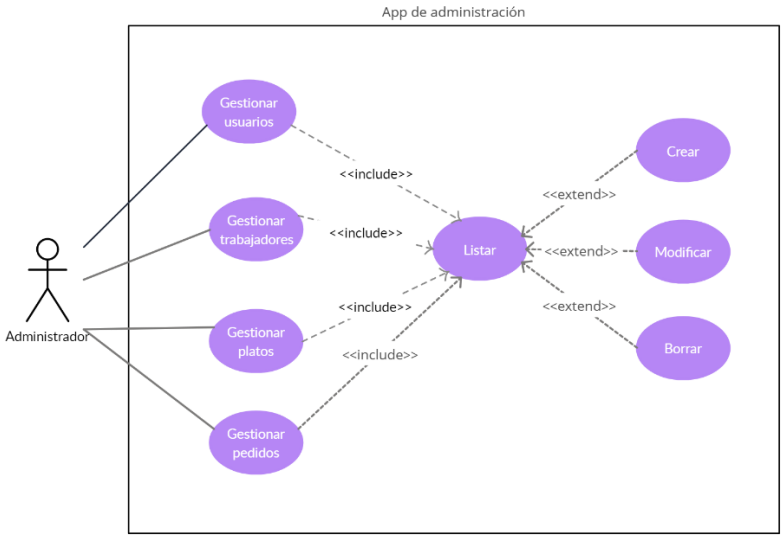
3.2. Casos de uso

Antes que nada, se debe definir correctamente que es un caso de uso. Como su nombre indica, un caso de uso es una secuencia de acciones que debe realizar alguien o algo, llamado actor, para llevar a cabo un proceso.

Como en cualquier aplicación, antes de su desarrollo se deben crear distintos casos de usos ajustado al, valga la redundancia, su uso. Por ello, he realizado los distintos casos de uso para las aplicaciones que desarrollaré.

Para comenzar, desarrollaremos los casos de uso de la aplicación orientada en el uso administrativo.

3.2.1. Aplicación administrativa



Caso de uso – Gestionar usuarios, trabajadores y platos

Antes de empezar en sí con este caso de uso, agruparemos las tres acciones que puede realizar el usuario administrador en nuestra aplicación porque realmente es la misma acción, únicamente cambia la entidad a la que hace efecto.

Descripción: El administrador desea gestionar una entidad, ya sea para crear, borrar o modificar.

Actor: Administrador.

Precondiciones: El usuario sea ha registrado y tiene el rol “Administrador”.

Curso normal de la acción:

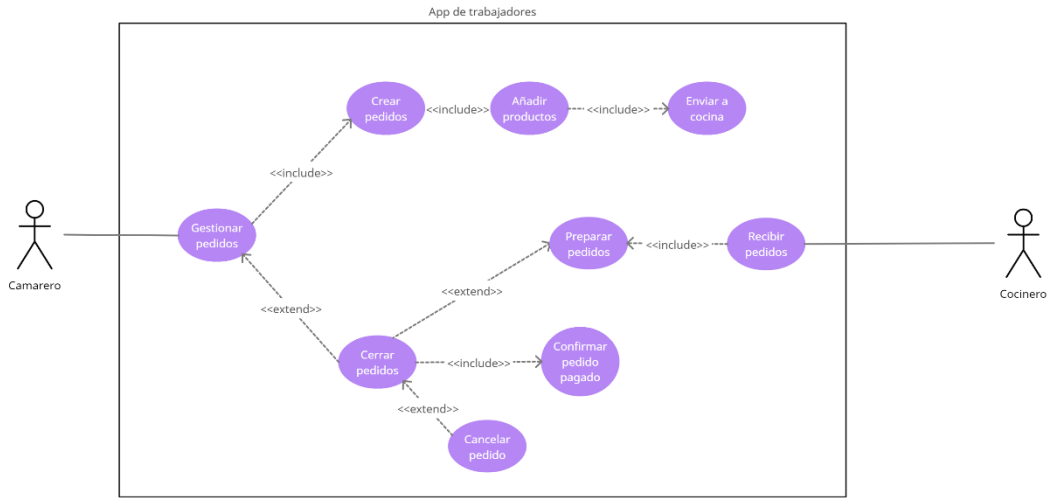
1. El administrador accede a la pestaña correspondiente a la entidad seleccionada.
2. La aplicación lista todas las entidades de la base de datos y el administrador elige una opción (Crear, Borrar o Modificar).
3. Si elige *Crear*, existirán los siguientes sucesos:
   3.1. El administrador introduce los datos de la nueva entidad.
   3.2. El sistema comprueba si los datos son correctos.
   3.3. Ingresa la nueva entidad en la base de datos.
4. Si elige *Modificar*, ocurrirá lo siguiente:
   4.1. El administrador introduce los nuevos datos.
   4.2. El sistema comprueba si los datos son correctos.
   4.3. Actualiza la entidad en la base de datos.

5. Si elige *Borrar*, los sucesos serán los siguientes:
   5.1. El sistema comprueba que esa entidad exista actualmente
   5.2. El sistema elimina toda la información relacionada con esas entidad en concreto.

Errores/Alternativas:

3. El sistema muestra, con una ventana emergente, que los datos son incorrectos, ya sea porque no cumplen un formato o porque ya están en uso.
4. El sistema muestra, con una ventana emergente, que los datos son incorrectos, ya sea porque no cumplen un formato o porque ya están en uso.
5. Algún dato no se borra correctamente.

3.2.2. Aplicación trabajadores



Caso de uso – Camarero

Descripción: El camarero gestiona los pedidos.

Actor: Camarero

Precondiciones: El usuario tiene que haberse registrado y tener el rol de “Camarero”

Curso normal de la acción:

1. El camarero gestiona los pedidos
2. El camarero crea un pedido
3. El camarero añade productos según quiera el cliente
4. Por último, el camarero envía el correo a cocina.

Errores/Alternativas:

2. Si hay pedidos ya realizados, el camarero puede cerrar un pedido.

    2.1 Debe comprobar si el pago está pagado correctamente.

    2.2 Puede cancelar el pedido si este no ha sido pagado correctamente o el cliente desea cancelarlo.

Caso de uso – Cocinero

Descripción: El cocinero recibe y prepara pedidos.

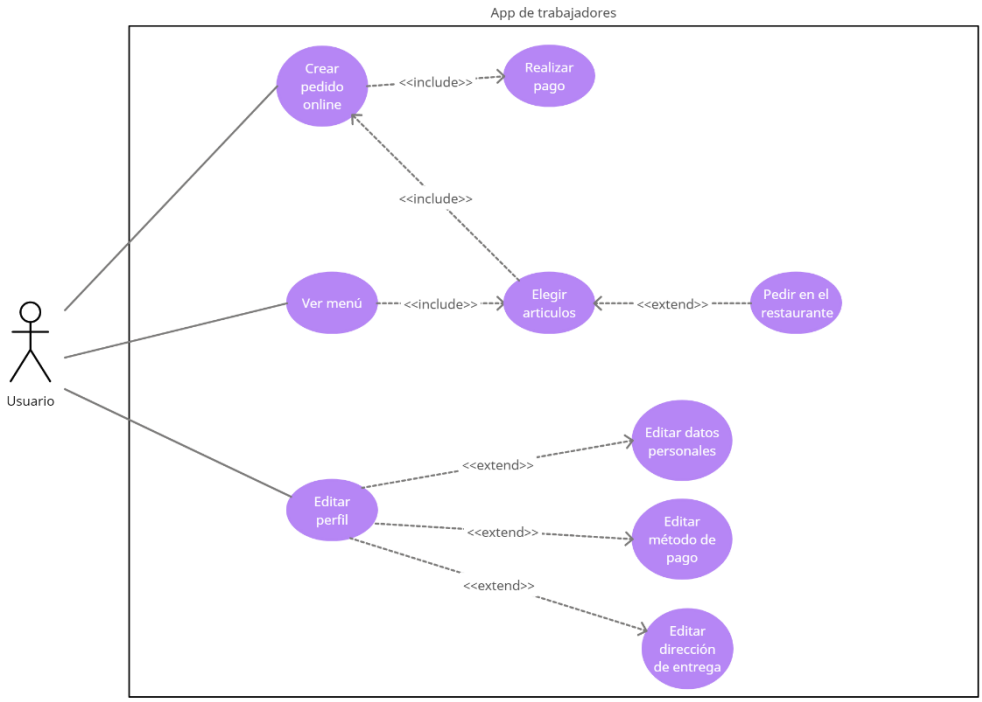
Actor: Cocinero.

Curso normal de la acción:

1. El cocinero recibe pedidos.
2. Tras esto, los prepara.
3. Cuando el pedido está acabado, lo finaliza.
4. En este caso, no es necesario comprobar si el pago es correcto, dado que cuando llega a cocina este tiene que ser correcto.

Alternativas/Errores: Nada.

3.2.3. Aplicación clientes



Caso de uso – Usuario- Crear pedido online

Descripción: El usuario crea un pedido online

Actor: Usuario

Precondiciones: El usuario debe de estar registrado con el rol de “Usuario”

Curso normal de la acción:

1. El usuario empieza a crear el pedido
2. Elige artículos para su pedido
3. -Paga dicho pedido y se envía a su domicilio.

Alternativas/Errores:

3. Error en el pago, por lo que se cancela el pedido

Caso de uso – Usuario- Ver menú

Descripción: El usuario observa el menú.

Actor: Usuario

Precondiciones: El usuario debe de estar registrado con el rol de “Usuario”

Curso normal de la acción:

1. El usuario observa el menú
2. A continuación, elige los productos que desea
3. Le comunica los productos que desea al camarero

Errores/Alternativas:

3. En el caso de comunicarle los productos al camarero, crea un pedido y los añade a el.

Caso de uso – Usuario- Editar perfil

Descripción: El usuario desea cambiar sus datos personales.

Actor: Usuario.

Precondiciones: El usuario debe de estar registrado con el rol de "Usuario".

Curso normal de la acción:

1. El usuario selecciona su perfil.
2. Selecciona el dato que quiera cambiar.
3. Depende de lo que seleccione, podrá realizar los siguiente:
   3.1. Si elige datos personales, podrá cambiar datos como el correo, el teléfono móvil o su nombre y apellidos.
   3.2. Si elige el método de pago, podrá cambiar los datos de la tarjeta guardada o añadir una.
   3.3. Si elige la dirección de envío, podrá cambiar o añadir la dirección donde se enviará su pedido.

   Alternativas/Errores:

   Los datos ya están en uso o son incorrectos.

3.3. Diagrama de clases

3.4. Modelo relacional

4. Entorno de programación

En el punto 4 de esta memoria se va a exponer y explicar los distintos lenguajes y librerías que se van a emplear en el desarrollo de todo el software. Además, hablaremos ligeramente del control de versiones y cómo lo hemos empleado.

4.1. Lenguajes y tecnologías que vamos a utilizar

4.1.1. Base de datos

4.1.2. Backend

4.1.3. Frontend

El *frontend* es la capa con la que el usuario interactúa y todo lo que ello conlleva, como son los diseños, funcionalidad que se encarga de la interactividad con los usuarios, etc.

En el caso del proyecto *Sei Vegan, Sei Grün*, el *frontend* está desarrollado únicamente en el *framework Flutter*, con base del lenguaje de programación *Dart*, el cual va a ser explicado en el siguiente punto.

Dart

Dart es un lenguaje de programación de código abierto desarrollado por Google, publicado por primera vez en 2011. Este lenguaje, en sus inicios, era una alternativa para JavaScript, siendo Dart un lenguaje orientado a objetos (POO) y de tipado estático, un estilo parecido a Java. Actualmente, Dart puede ser utilizado en distintos ámbitos, como aplicaciones web, aplicaciones multiplataforma o servidores.

Dart, como hemos comentado levemente anteriormente, es un lenguaje de programación con similitudes con JavaScript, Java o C++. Por lo tanto, al tener esta base tan similar con lenguajes tan conocidos, a los nuevos estudiantes, en los que me incluyo, se les hace muy sencillo aprender de él.

Flutter

Como hemos nombrado anteriormente, vamos a utilizar el *framework Flutter*, basado en un único lenguaje, *Dart*. A diferencia de otros frameworks de otros lenguajes o del propio Dart, Flutter compila a código nativo, consiguiendo así un rendimiento mayor que en otras aplicaciones basadas en web-views.

El mayor potencial, y unas de las razones por las que se ha elegido este SDK, es la facilidad en la que se pueden crear aplicaciones en distintas plataformas con un único código. Flutter permite crear aplicaciones nativas para *Android*, *iOS*, aplicaciones web

y aplicaciones en *Windows*, *Linux* y *MacOS*. Además, también tiene integrado el *Hot Reload*, una característica muy útil a la hora de desarrollar. A continuación, indagaremos en características de *Flutter*, y a su vez de *Dart*, y veremos qué ventajas tiene.

De la primera ventaja que vamos a hablar es sobre su rápido desarrollo. Cómo hemos avanzado anteriormente, *Dart* compila en lenguaje 100% nativo, por lo que su velocidad de compilado es una de las más óptimas actualmente. Además de todo esto, tiene implementado la función *Stateful Hot Reload*, que te permite ver, de forma instantánea, los cambios realizados con la aplicación corriendo.

Otra de las ventajas que tiene *Flutter* es tener una amplia librería de widgets. Los widgets son los componentes que conforman nuestra aplicación. Estos componentes, mediante clases y *builders*, pueden hacerse reutilizables y personalizables, algo muy interesante a la hora de optimizar y ahorrar código. Algunos de los *widgets* más básicos pueden ser:

- *Text* ⇨ Nos permite ver texto
- *Appbar/TabBar* ⇨ Son la parte superior e inferior de la aplicación.
- *Button* ⇨ Componente qué, al dar click, realiza una función.

Los *widgets*, además, tienen dos tipos: Con estado y sin estado. El estado es la información que se puede utilizar una vez se crea el *widget* y que cambie durante su vida útil.

5. (Apartado 5: Ejemplo bibliografía)

[Identificador] Inicial Nombre 1er Autor. Apellido 1º Autor, Inicial Nombre 2º Autor. Apellido 2º. Autor, , ... y Inicial Nombre Último Autor. Apellido Último Autor, *Nombre Completo del Libro en cursiva*. Ciudad de edición: Editorial, Año.

6. (Apartado 6)