Desarrollo de una red social para personas que disfrutan de una comida saludable

Susana Vázquez Corte
Ingeniería de software
Desarrollo de aplicación web

Gregorio Robles Martínez
Santi Caballe Llobet

Fecha de entrega: 12 de junio de 2019

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Desarrollo de una red social para personas que disfrutan de una vida/comida saludable*
Nombre del autor:	*Susana Vázquez Corte*
Nombre del consultor/a:	*Gregorio Robles Martínez*
Nombre del PRA:	*Santi Caballe Llobet*
Fecha de entrega (mm/aaaa):	01/06/19
Titulación::	*Ingeniería de software*
Área del Trabajo Final:	*Desarrollo web*
Idioma del trabajo:	*Español*
Palabras clave	*tecnologías web, aplicaciones web, Django, red social, alimentación saludable.*

Resumen del Trabajo:

La finalidad de este Trabajo Final de Grado es la creación de una red social para personas que llevan o quieren llevar una vida saludable.

Esta red social permite a las personas tener un espacio donde compartir sus dudas sobre si los alimentos o las recetas que consumen son saludables y se pueden consumir habitualmente o no.

Para el desarrollo del proyecto se utiliza una metodología en cascada en la que se establecen cuatro hitos de entrega. La aplicación se ha desarrollado con Django, PostgreSQL y Docker.

Realizar este proyecto ha sido muy satisfactorio, por un lado he podido asentar los conocimientos adquiridos durante todo el grado y por otro lado he aprendido nuevas tecnologías que me ayudarán en el ámbito laboral.

Abstract:

The purpose of this Final Degree Project is the creation of a social network for people who lead or want to lead a healthy life.

This social network allows people to have a space to share their doubts about whether the food or the recipes they consume are healthy and can be consumed habitually or not.

For the development of the project, a cascade methodology is used in which four delivery milestones are established. The application has been developed with Django, PostgreSQL and Docker.

Carry out this project has been very satisfactory, on the one hand I have been able to establish the knowledge acquired during the whole degree and by the other hand I have learned new technologies that will help me in the workplace.

Agradecimientos

Ahora que llega el final, no puedo irme sin aprovechar para expresar todos los agradecimientos que no he podido expresar durante todos estos años de grado.

Muchas gracias a la UOC, por apostar por un formato de estudios no presencial. A su personal, profesores y consultores, gracias a vuestro trabajo, muchas personas como yo pueden estudiar las cosas que les gusta y labrarse un buen futuro laboral.

A mi gran compañero de estudios, Javi, parecía que no pero al final lo hemos conseguido, gracias por los ánimos y por ayudarme siempre que lo he necesitado.

A mis grandes amigos, por vuestra paciencia, vuestros ánimos y por entender mi vida de días de entregas y de exámenes.

Y sobre todo a mis tres pilares fundamentales, Mamá, Iván, Paqui. Gracias por tantos años apoyándome, ayudándome y dándome ánimos, por soportar mis nervios y mis días de cansancio, por adaptar vuestras agendas a mi calendario escolar, gracias por todo! De verdad que sin vosotros esto nunca hubiera sido posible.

Índice

Lista de figuras

1. Introducción

1.1 Contexto y justificación del Trabajo[2,3]

Cada vez es más frecuente ver en restaurantes, en la publicidad o en los supermercados alimentos denominados *Healthy* o saludables. La sociedad está cambiando sus hábitos de alimentación, de no preocuparse por los alimentos que ingiere o su procedencia, a buscar alimentos no procesados, con ingredientes naturales y sin azúcares añadidos.

Esto se produce como consecuencia de la toma de conciencia de las personas sobre los que tiene para la salud este tipo de alimentación según recientes estudios ciéntificos[1]:

- Prevención de la obesidad.
- Prevención de enfermedades crónicas como el colesterol y la diabetes.
- Mejora en el descanso.
- Mejora en el estado de ánimo.
- Mayor concentración.

Para muchas personas el cambiar de un estilo de alimentación a otro no es fácil ya que no disponen de herramientas o de información suficiente para saber si un alimento es saludable o no. No están habituadas a leer los ingredientes de los productos que compran y le surgen dudas de si un producto es realmente saludable o no.

Además, cada vez es más frecuente encontrar productos en los supermercados denominados como “naturales” o “sin azúcares añadidos” que están llenos de ingredientes procesados o de otros ingredientes no denominados azúcar pero que tienen los mismos efectos.

Como consecuencia, cada vez es más frecuente encontrar en internet (blogs, redes sociales,...) preguntas de usuarios sobre si un alimento o incluso una receta es saludable y si se puede consumir cada día, de manera ocasional o nunca.

En definitiva, es necesaria una red social que agrupe a este perfil de usuarios, donde puedan compartir sus inquietudes y buscar información sobre los alimentos y/o recetas que consume.

1.2 Objetivos

El objetivo principal del proyecto es crear una red social para todas las personas que disfrutan o quieren disfrutar de una alimentación saludable donde los usuarios puedan añadir las etiquetas de los productos o de las recetas que quieran cocinar para que el resto de usuarios puedan evaluar si se puede consumir diariamente, ocasionalmente o nunca, además puedan evaluar los productos o recetas de otros usuarios e incluso puedan hacer amigos.

A nivel personal, los objetivos son:

- Asentar los conocimientos adquiridos durante todo el grado.
- Profundizar en Python, uno de mis lenguajes favoritos.
- Conocer Django, un framework que se utiliza cada día más y que tengo mucho interés en aprender.
- Desplegar una aplicación en un entorno de producción.

1.3 Enfoque y método seguido

A pesar de que todos los desarrollos de software tienen la misma finalidad, no todos tienen las mismas características, necesidades, recursos, equipos de desarrollo,... Es por esto que existen diferentes metodologías, según su enfoque se pueden dividir en metodologías tradicionales y metodologías ágiles.

Las metodologías tradicionales se caracterizan por tener una fuerte planificación que es definida en la fase inicial del proyecto. Mientras que las metodologías ágiles están basadas en el trabajo incremental e iterativo y están enfocadas a proyectos que necesiten cierta flexibilidad y puedan cambiar en el tiempo.

Teniendo en cuenta que este proyecto tiene establecido por defecto cuatro hitos y que hasta el último no se realiza la entrega total del proyecto, se considera utilizar una metodología tradicional, en concreto, la metodología en cascada.

En la primera fase se definirá el plan de trabajo, donde se definirá el problema que se pretende resolver, el trabajo que se llevará a cabo y las fases del proyecto.

La segunda fase está enfocada en la parte del diseño de la aplicación, se establecen los requisitos, se definen los casos de uso, se realiza el modelado de datos y la definición de las vistas. Además en esta fase se inicia la implementación y se crea la estructura del proyecto.

La tercera fase será dedica a realizar la implementación del proyecto, las pruebas de test y el despliegue de la aplicación en un entorno de producción.

Por último, la cuarta fase conlleva la elaboración de la memoria, la creación de la presentación así como la entrega del proyecto.

1.4 Planificación del Trabajo

- Hitos:

Fecha inicio	Fecha fin	Hito	Tareas
20/02/19	06/03/19	PEC1: Plan de trabajo	- Creación del plan de proyecto, contiene: • Problema que se pretende resolver. • Trabajo que se llevará a cabo. • Tareas e hitos.
07/03/19	10/04/19	PEC2: Hito intermedio	- Establecimiento de requisitos. - Especificación de los casos de uso. - Modelado de datos. - Inicio de implementación.
11/04/19	29/05/19	PEC3: Hito intermedio	- Desarrollo de la aplicación. - Test.
30/05/19	12/06/19	PEC4: Memoria final	- Creación de memoria. - Creación de la presentación del TFG. - Entrega del proyecto.

- Diagrama de Gantt [4]:

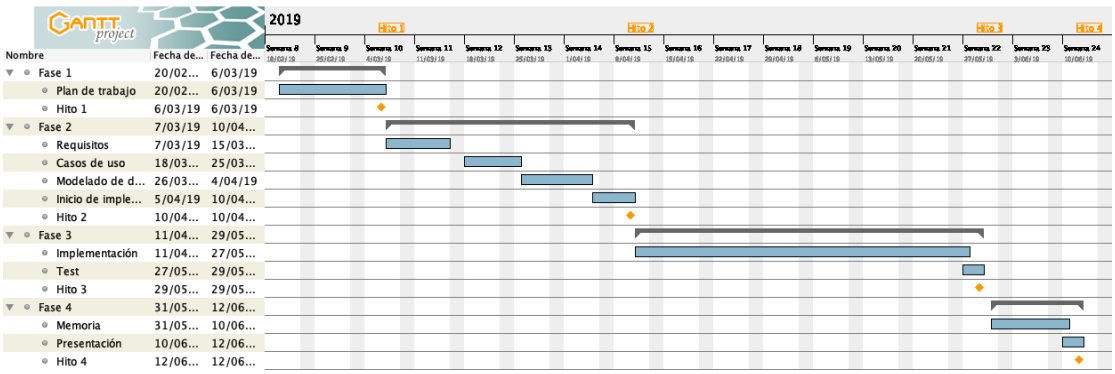

Ilustración 1: Diagrama de Gantt

1.5 Estructura de la memoria

Para finalizar la introducción, se detalla a continuación un breve resumen sobre el contenido de los capítulos de la memoria.

- Capítulo 1: introducción de la memoria, se realiza la presentación del proyecto y los objetivos. Además se determina cual es la metodología a seguir y la planificación del proyecto.
- Capítulo 2: se definen los requisitos del proyecto, casos de uso y el modelado de datos.
- Capítulo 3: se define las tecnologías utilizadas en el proyecto, la estructura y las diferentes maneras de desplegar la aplicación.
- Capítulo 4: se detalla el plan de pruebas a realizar una vez acabada la implementación.
- Capítulo 5: contiene las conclusiones una vez finalizado el proyecto.
- Capítulo 6: se recoge la bibliografía utilizada para el desarrollo de la memoria.

2. Diseño

2.1 Requisitos

A continuación se detallan los requisitos funcionales y no funcionales de la aplicación web.

1. Requisitos funcionales:

    - *RF-01*: el usuario debe tener la posibilidad de registrarse en el sistema mediante un correo electrónico que servirá de identificador, una contraseña y un nombre. Además de manera opcional podrá indicar su año de nacimiento y su tipo de dieta.
    - *RF-02*: el usuario podrá editar sus datos.
    - *RF-03*: el usuario debe estar registrado en el sistema para poder acceder.
    - *RF-04*: el sistema tendrá un tipo de rol, administrador. Puede existir más de un usuario con rol administrador.
    - *RF-05*: el usuario se autenticará con su correo y contraseña, además podrá recuperar su contraseña.
    - *RF-06*: solo un usuario con rol administrador podrá añadir a otros administradores.
    - *RF-07*: el administrador puede añadir tipos de dietas.
    - *RF-08*: el administrador podrá eliminar imágenes y recetas de cualquier usuario.
    - *RF-09*: el administrador podrá eliminar valoraciones.
    - *RF-10*: una vez el usuario inicie sesión en el sistema tendrá acceso a las siguientes secciones:
        - Valoración de productos: imágenes con los ingredientes de los productos que los usuarios podrán valorar como: apto para consumo diario, consumo esporádico / ocasional o no consumir.
        - Recetas: recetas que suben los usuarios y que podrán ser etiquetadas como: aptas para consumo diario, consumo esporádico / ocasional o no consumir.
        - Favoritos: usuarios marcados como favoritos.
    - *RF-11*: el usuario podrá publicar una imagen con los ingredientes de un producto para ser valorado.
    - *RF-12*: el usuario podrá valorar las imágenes de otros usuarios.
    - *RF-13*: el usuario podrá editar y eliminar solo sus propias imágenes.
    - *RF-14*: el usuario tendrá acceso a las imágenes de todos los usuarios, podrá filtrar por tipo de producto.
    - *RF-15*: el usuario podrá publicar una receta indicando nombre, ingredientes, tiempo de preparación y etiqueta.
    - *RF-16*: el usuario podrá editar y eliminar solo sus recetas.
    - *RF-17*: el usuario podrá valorar una receta de cualquier usuario.

- *RF-18*: el usuario tendrá acceso a las recetas de todos los usuarios, además podrá filtrar por usuario y por etiqueta.
- *RF-19*: el usuario se podrá dar de baja.

2. Requisitos no funcionales:

- *RNF-01*: el sistema debe ser compatible con todos los navegadores.
- *RNF-02*: el sistema debe adaptarse a dispositivos móviles.

2.2 Casos de uso

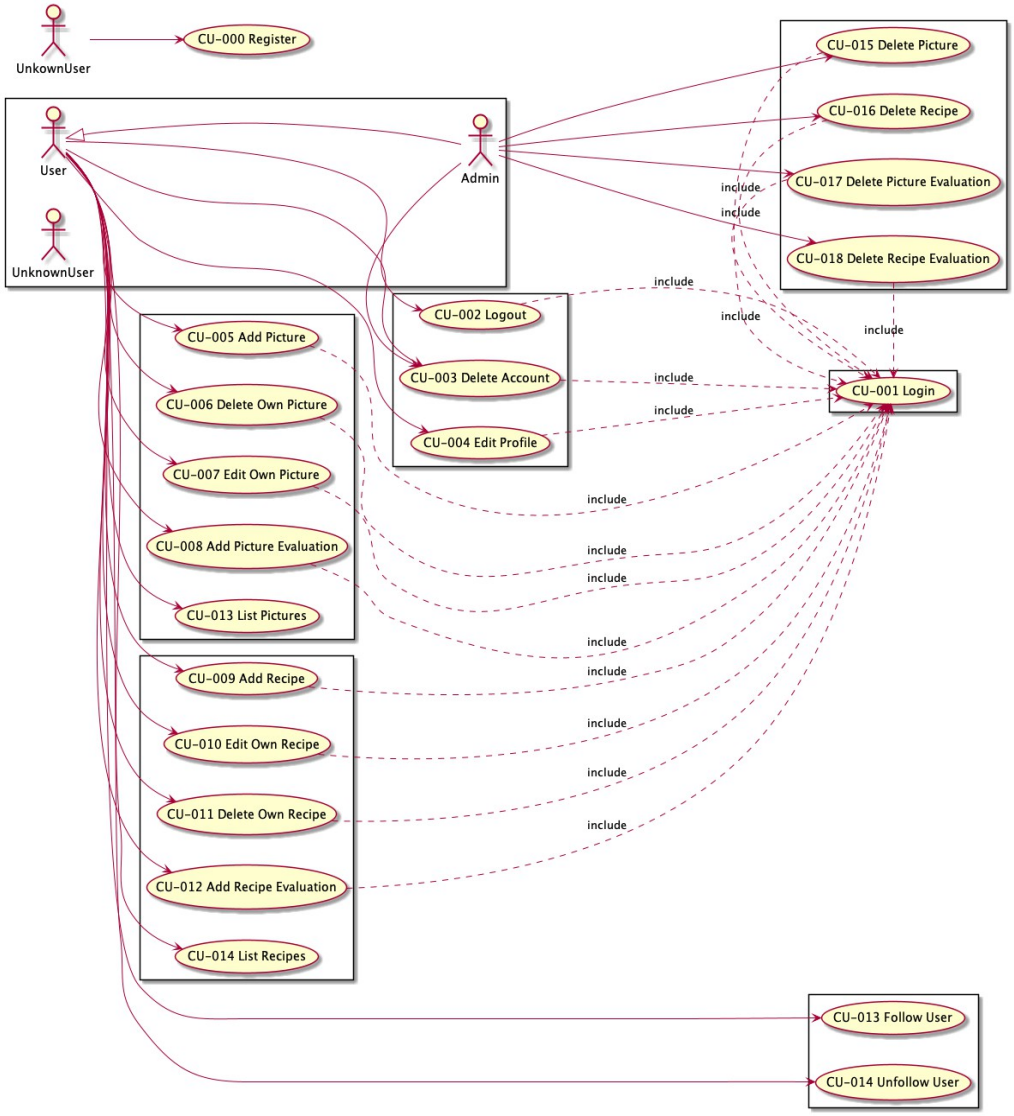

Ilustración 2: Casos de uso

Leyenda:

- UnknownUser: usuario no registrado en el sistema.
- User: usuario registrado en el sistema.
- Admin: usuario registrado en el sistema con perfil de administrador.

CU-000: Register user

- **Caso de uso:** registro de un usuario en el sistema
- **Actor**: usuario desconocido
- **Precondición**: el usuario no debe tener iniciada su sesión en el sistema
- **Escenario principal**:
    1. El usuario accede a la página de registro
    2. El sistema solicita los datos para el registro
    3. El usuario introduce los datos de registro
    4. El sistema notifica que el registro ha sido satisfactorio
- **Extensiones**:
    4.1: el sistema muestra error de registro
- **Post-condición**: se registra el usuario en el sistema

CU-001: Login

- **Caso de uso**: acceder al sistema
- **Actor**: usuario
- **Precondición**: el usuario debe estar registrado en el sistema
- **Escenario principal**:
    1. El sistema solicita correo electrónico y contraseña del usuario
    2. El usuario introduce los datos
    3. El sistema muestra la página de inicio
- **Extensiones**:
    2.1: el sistema muestra mensaje de error, no existe usuario con los datos introducidos
- **Post-condición**: -

CU-002: Logout

- **Caso de uso**: salir del sistema
- **Actor**: usuario
- **Precondición**: el usuario debe haber iniciado sesión en el sistema
- **Escenario principal**:
    1. El usuario solicita salir del sistema
    2. El sistema cierra la sesión del usuario
- **Extensiones**: -
- **Post-condición**: -

CU-003: Delete User Account

- **Caso de uso**: el usuario elimina su cuenta
- **Actor**: usuario
- **Precondición**: el usuario debe haber iniciado sesión en el sistema
- **Escenario principal**:
    1. El usuario solicita dar de baja su cuenta
    2. El sistema solicita la contraseña del usuario
    3. El usuario introduce la contraseña
    4. El sistema solicita confirmación de eliminación de cuenta
    5. El sistema elimina la cuenta del usuario y muestra la pantalla de inicio

- **Extensiones**:
  3.1: el usuario cancela la eliminación de la cuenta
  4.1: el usuario cancela la eliminación de la cuenta
- **Excepciones**:
  3: el usuario introduce una contraseña errónea
  E1: el sistema notifica el error y vuelve al paso 2
- **Post-condición**: -

CU-004: Edit Profile
- **Caso de uso**: el usuario edita los datos de su perfil
- **Actor**: usuario
- **Precondición**: el usuario debe haber iniciado sesión en el sistema
- **Escenario principal**:
  1. El usuario introduce los nuevos datos en el sistema
  2. El sistema modifica los datos del usuario
- **Extensiones**: -
- **Post-condición**: -

CU-005: Add Picture
- **Caso de uso**: el usuario añade una imagen a su perfil
- **Actor**: usuario
- **Precondición**: el usuario debe haber iniciado sesión en el sistema
- **Escenario principal**:
  1. El sistema solicita al usuario la imagen que añadir
  2. El usuario selecciona la imagen
  3. El sistema añade la nueva foto
- **Extensiones**:
  2.1: El usuario cancela la operación
- **Post-condición**: -

CU-006: Delete Own Picture
- **Caso de uso**: el usuario elimina una imagen a su perfil
- **Actor**: usuario
- **Precondición**: el usuario debe haber iniciado sesión en el sistema
- **Escenario principal**:
  1. El sistema muestra las imágenes del usuario
  2. El usuario selecciona la imagen
  3. El sistema pide confirmación
  4. El sistema elimina la imagen
- **Extensiones**:
  2.1: El usuario cancela la operación
  3.1: El usuario cancela la operación
- **Post-condición**: -

CU-007: Edit Own Picture
- **Caso de uso**: el usuario edita una imagen añadida a su perfil
- **Actor**: usuario

- **Precondición**: el usuario debe haber iniciado sesión en el sistema
- **Escenario principal**:
    1. El sistema muestra las imágenes del usuario
    2. El usuario selecciona la imagen
    3. El sistema solicita la nueva imagen
    4. El usuario añade la imagen
    5. El sistema edita la imagen
- **Extensiones**:
    2.1: el usuario cancela la operación
    4.1: el usuario cancela la operación

CU-008: Add Picture Evaluation
- **Caso de uso**: el usuario añade una valoración a una imagen
- **Actor**: usuario
- **Precondición**: el usuario debe haber iniciado sesión en el sistema
- **Escenario principal**:
    1. El usuario selecciona la imagen a valorar
    2. El usuario valora la imagen
    3. El sistema añade la valoración a la imagen
- **Extensiones**:
    2.1: El usuario cancela la operación

CU-009: Add Recipe
- **Caso de uso**: el usuario añade una receta a su perfil
- **Actor**: usuario
- **Precondición**: el usuario debe haber iniciado sesión en el sistema
- **Escenario principal**:
    1. El sistema solicita los datos de la receta (nombre, ingredientes, tiempo de preparación y etiqueta)
    2. El usuario introduce los datos
    3. El sistema añade la receta al sistema
- **Extensiones**:
    2.1: El usuario cancela la operación

CU-010: Edit Own Recipe
- **Caso de uso**: el usuario edita una receta añadida a su perfil
- **Actor**: usuario
- **Precondición**: el usuario debe haber iniciado sesión en el sistema
- **Escenario principal**:
    1. El sistema solicita al usuario la receta que quiere editar
    2. El usuario selecciona la receta
    3. El usuario edita los datos de la receta
    4. El sistema edita la imagen
- **Extensiones**:
    2.1: El usuario cancela la operación
    3.1: El usuario cancela la operación

CU-011: Delete Own Recipe

- **Caso de uso**: el usuario elimina una receta de su perfil
- **Actor**: usuario
- **Precondición**: el usuario debe haber iniciado sesión en el sistema
- **Escenario principal**:
    1. El usuario selecciona la receta a eliminar
    2. El sistema elimina la imagen
- **Extensiones**:
    2.1: El usuario cancela la operación

CU-012: Add Recipe Evaluation

- **Caso de uso**: el usuario añade una valoración a una receta
- **Actor**: usuario
- **Precondición**: el usuario debe haber iniciado sesión en el sistema
- **Escenario principal**:
    1. El usuario selecciona la receta a valorar
    2. El usuario añade su valoración
    3. El sistema añade la valoración a la receta
- **Extensiones**:
    1.1: El usuario cancela la operación
    2.1: El usuario cancela la operación

CU-013: Recipe List

- **Caso de uso**: el usuario lista todas las recetas
- **Actor**: usuario
- **Precondición**: el usuario debe haber iniciado sesión en el sistema
- **Escenario principal**:
    1. El usuario selecciona mostrar todas las recetas
    2. El sistema muestra todas las recetas

CU-014: Picture List

- **Caso de uso**: el usuario lista todas las imágenes
- **Actor**: usuario
- **Precondición**: el usuario debe haber iniciado sesión en el sistema
- **Escenario principal**:
    1. El usuario selecciona mostrar todas las imágenes
    2. El sistema muestra todas las imágenes

CU-015: Delete Picture

- **Caso de uso**: el administrador elimina una imagen de un usuario
- **Actor**: usuario con rol administrador
- **Precondición**: el usuario debe haber iniciado sesión en el sistema
- **Escenario principal**:
    1. El usuario selecciona la imagen que quiere eliminar
    2. El sistema solicita confirmación
    3. El sistema elimina la imagen

- **Extensiones**:
    2.1: El usuario cancela la operación

CU-016: Delete Recipe
- **Caso de uso**: el administrador elimina una receta de un usuario
- **Actor**: usuario con rol administrador
- **Precondición**: el usuario debe haber iniciado sesión en el sistema y haber subido al menos una receta

- **Escenario principal**:
    1. El usuario selecciona la receta que quiere eliminar
    2. El sistema solicita confirmación
    3. El sistema elimina la receta
- **Extensiones**:
    2.1: El usuario cancela la operación

CU-017: Delete Picture Evaluation
- **Caso de uso**: el administrador elimina una valoración de la imagen de un usuario
- **Actor**: usuario con rol administrador
- **Precondición**: el usuario debe haber iniciado sesión en el sistema
- **Escenario principal**:
    1. El usuario selecciona la imagen a la que quiere eliminar la valoración
    2. El usuario selecciona la valoración a eliminar
    3. El sistema solicita confirmación
    4. El sistema elimina la valoración de la imagen
- **Extensiones**:
    2.1: El usuario cancela la operación
    3.1: El usuario cancela la operación

CU-018: Delete Recipe Evaluation
- **Caso de uso**: el administrador elimina una valoración de la receta de un usuario
- **Actor**: usuario con rol administrador
- **Precondición**: el usuario debe haber iniciado sesión en el sistema
- **Escenario principal**:
    1. El usuario selecciona la receta a la que quiere eliminar la valoración
    2. El usuario selecciona la valoración a eliminar
    3. El sistema solicita confirmación
    4. El sistema elimina la valoración de la receta
- **Extensiones**:
    2.1: El usuario cancela la operación
    3.1: El usuario cancela la operación

2.3 Modelado de datos

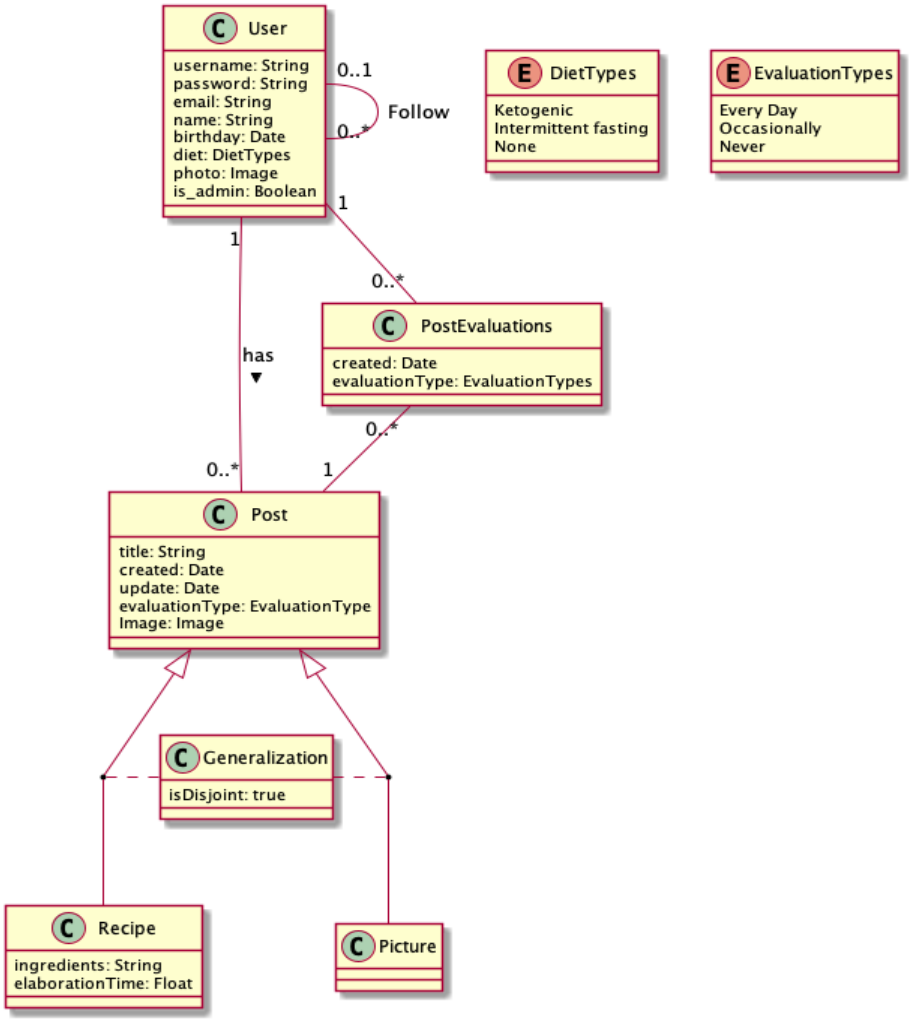

Ilustración 3: Diagrama de clases

Definición de clases:

- User: clase que contiene los datos de los usuarios registrados.
- Post: clase abstracta de la que heredan las class Picture y Recipe.
- Picture: tabla que contiene las imágenes de los usuarios.
- Recipe: tabla que contiene las recetas de los usuarios.
- PostEvaluations: tabla que contiene las valoraciones de los usuarios.
- DietTypes: tabla que contiene los tipos de dietas disponibles.
- EvaluationTypes: tabla que contiene los tipos de valoraciones disponibles.

3. Implementación

A lo largo de este capítulo se detallan las herramientas y tecnologías utilizadas para el desarrollo de la aplicación web, se describe la estructura del proyecto y como poder desplegar la aplicación tanto en un entorno de desarrollo como en un entorno de producción.

3.1 Herramientas y tecnologías utilizadas

El desarrollo de la aplicación se realiza mediante Django y Python como lenguaje de programación. A nivel de diseño se utiliza Bootstrap.

Los datos de la aplicación serán guardados en PostgreSQL. Se decide utilizar esta base de datos relacional debido a que es el sistema más completo con licencia BSD, en contra de MySQL que tiene licencia dual, además es la más compatible con las migraciones de modelos en Django.

Dado que la implementación de la aplicación se realizará en diferentes ordenadores, se decide utilizar Docker como entorno de desarrollo y GitHub como plataforma de desarrollo, de esta manera se podrá tener acceso al código desde cualquier dispositivo.

Por último, se utiliza Heroku como plataforma de despliegue mediante Docker.

A continuación se detallan las características principales de las herramientas y tecnologías utilizadas:

- PyCharm 2019.1.2 (*Professional Edition*)[5] : entorno de desarrollo multiplataforma que facilita el desarrollo de aplicaciones.
- Django[6] : framework web de alto nivel de código abierto para la creación de aplicaciones web escrito en Python.

Está pensado para trabajar bajo un patrón MVT, modelo-vista-template o modelo-vista-plantilla, está basado en el MVC, cada capa tiene su responsabilidad permitiendo la reutilización de código.

En la capa modelo se definen los datos de la aplicación, la capa plantilla contiene la información de como mostrar los datos y en la capa de las vistas se define la lógica que accede al modelo y la delega en la plantilla asignada.

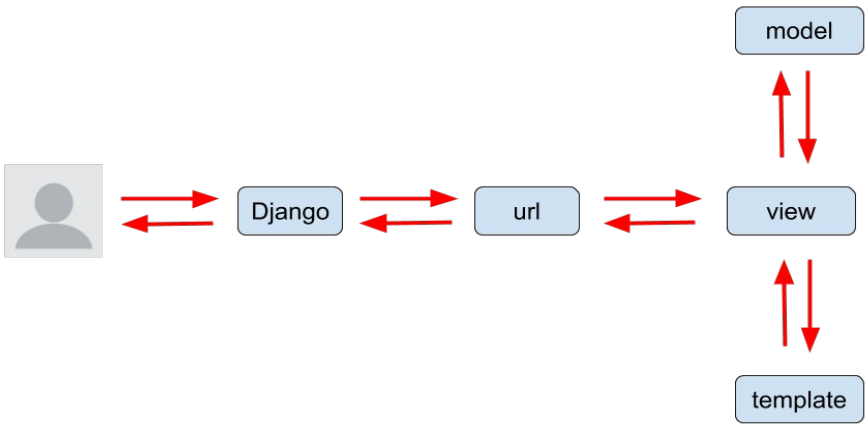

Ilustración 4: Patrón Modelo-Vista-Template

Django permite el desarrollo de una aplicación web de manera rápida y cómoda, facilita gracias a las siguientes características:

- Incluye un ORM (*Oriented Relational Model*) que permite interactuar con la base de datos de forma simple.
- Utiliza un diseño de URL limpio y elegante.
- Fomenta la reutilización de código (DRY).
- Incorpora por defecto un sistema de administración que permite crear, editar, eliminar datos.
- Es escalable y modular.
- Es seguro, incluye por defecto medidas para que no se pueda hacer inyecciones SQL (*SQL Injection*) y ataques de falsificación de petición en sitios cruzados, *Cross Site Request Forgery.*

- Bootstrap[7]: se trata de una framework de desarrollo para diseño web. Entre sus características destaca:
  - Es *responsive.*
  - Incluye Grid System permitiendo maquetar por columnas.
  - Se integra fácilmente con JavaScript.
  - Utiliza Less.
- Python 3.7[8]: lenguaje de programación interpretado, no compilado, tipado dinámico, fuertemente tipado y multiplataforma. Además es multiparadigma, soporta orientación de objetos, programación imperativa y funcional.
- PostgreSQL[9]: Se trata de un sistema gestor de base de datos relacional orientado a objetos. Sus principales características son:

- Permite alta concurrencia.
- Es multiplataforma.
- Usa arquitectura cliente / servidor.
- Es *Full ACID compliant.*
- Es escalable, soporta gran cantidad de peticiones simultaneas.

- Docker[10] / Docker Compose[11]: se puede definir Docker como una herramienta que permite el despliegue de aplicaciones mediante contenedores. Estos contenedores incluyen todo lo necesario para que la aplicación funcione de manera aislada y sin dependencia con el sistema operativo que lo está ejecutando.
- Heroku[12]: se trata de un una plataforma como servicio o PaaS (*Plataform as Service*). Permite a los usuarios el despliegue de una aplicación sin tener que preocuparse por el hardware y software necesario para que funcione.

Internamente funciona con los llamados Dynos, contenedores independientes que se gestionan en tiempo de ejecución.

Soporta múltiples lenguajes de programación y frameworks.

- Github[13]: plataforma de desarrollo que permite alojar proyectos utilizando el sistema de control de versiones de Git.

3.2 Estructura de la aplicación

A continuación se detalla brevemente la estructura utilizada para la aplicación:

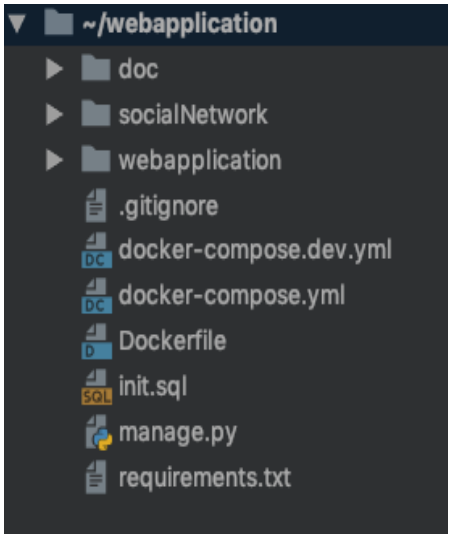

Ilustración 5: Estructura final de la aplicación

- doc: contiene los ficheros en PlantUMLde los casos de uso y el modelo de datos.
- socialNetwork: contiene todos los ficheros y directorios de la aplicación.

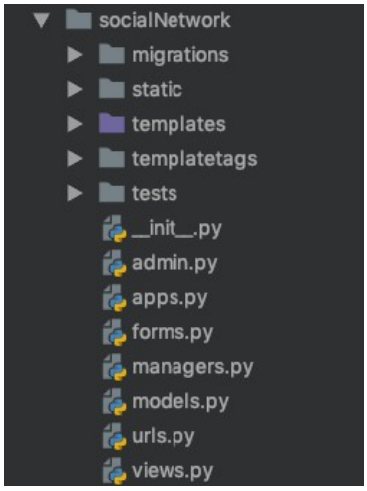

Ilustración 6: Estructura directorio socialNetwork

  - migrations: directorio que contiene los ficheros para la creación de la base de datos y las migraciones.
  - static: ficheros estáticos, contiene el template de Bootstrap.
  - templates: plantillas html de las vistas usadas por el controlador.
  - templatetags: etiquetas para plantillas, permiten insertar elementos Python dentro del html.
  - admin.py: contiene la configuración de los modelos que estarán disponibles desde la interfaz de administración.
  - forms.py: contiene la definición de los formularios para cada uno de los modelos.
  - models.py: contiene las clases con la definición de los atributos de cada clase.
  - urls.py: contiene el mapeo entre las urls y la vista.
  - views.py: incluye la lógica de la aplicación.

- Webapplication:

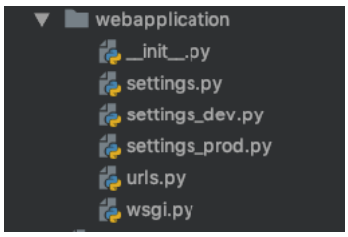

Ilustración 7: Estructura directorio webapplication

  - settings.py: fichero con la configuración básica para iniciar la aplicación. Este fichero es importado desde los ficheros settings_dev.py y settings_prod.py.
  - settings_dev.py: fichero que incluye la configuración específica para iniciar la aplicación en modo desarrollo.
  - settings_prod.py: fichero que incluye la configuración específica para iniciar la aplicación en el entorno de desarrollo.
- .gitignore: ficheros que se excluyen de subir a Github.
- docker-compose.dev.yml: define la configuración de los servicios, redes y volúmenes de los contenedores Docker para el entorno de desarrollo.
- docker-compose.yml: define la configuración de los servicios, redes y volúmenes de los contenedores Docker para el entorno de producción.
- Dockerfile: incluye la definición del entorno de aplicaciones.
- init.sql: script que se utiliza en la creación del contenedor de la base de datos.
- manage.py: script que ayuda con la administración de la aplicación. Algunas de las funciones que podemos realizar son:
  - Crear ficheros de migración (makemigrations).
  - Realizar migraciones en la base de datos (migrate).
  - Iniciar la aplicación (runserver).
  - Crear una aplicación dentro del proyecto (startapp).
- requirements.txt: contiene las librerías necesarias para la aplicación.

3.3 Despliegue de la aplicación

El proyecto está preparado para hacer el despliegue de la aplicación en cualquier entorno de desarrollo, el único requisito es tener instalado en la máquina Docker Compose. La instalación se puede realizar siguiendo la documentación oficial de la página.[14]

A continuación se describen los pasos a seguir para el despliegue de la aplicación según el entorno.

- Entorno local / desarrollo:

Para iniciar el servicio se debe acceder a la ruta que contiene el fichero docker-compose.yml y ejecutar los siguientes comandos en la terminal:

```
$ docker-compose build
$ docker-compose -f docker-compose.dev.yml -f docker-compose.yml up
```

El primer comando construye la imagen según la configuración del fichero Dockerfile y el segundo arranca el contenedor con los servicios indicados en el fichero docker-compose.

Una vez iniciados los contenedores, es necesario crear los modelos en la base de datos, esta operación solo es necesario realizarla la primera vez o en el caso de que se haga alguna modificación en los modelos. Para ello es necesario ejecutar los siguientes comandos:

```
$ docker-compose exec web /bin/bash
$ python manage.py migrate
```

Finalmente, podemos acceder a la aplicación mediante la url http://0.0.0:8000

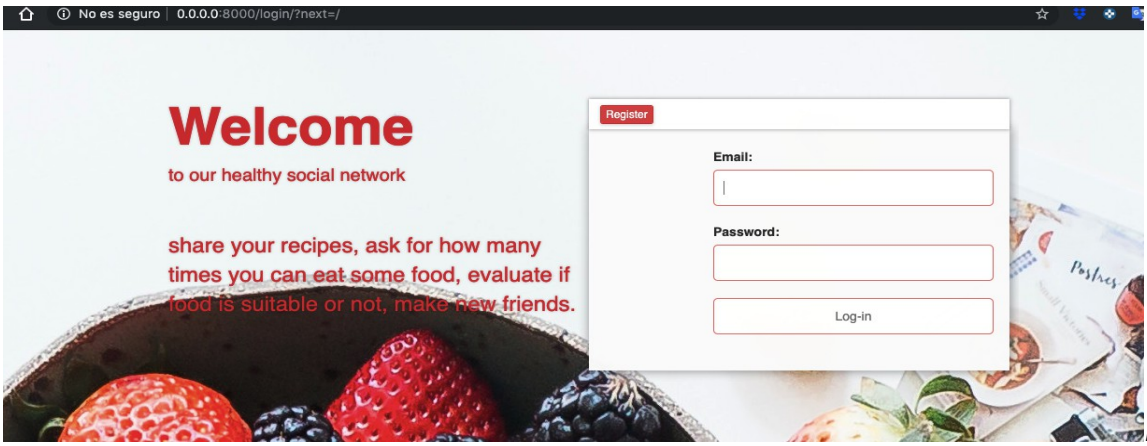

Ilustración 8: Home entorno local

- Entorno de producción:

Tal como se ha comentado en capítulos anteriores, el entorno de producción está desplegado en Heroku.

La primera vez que se crea el entorno de producción es necesario crear una aplicación en Heroku e instalar el *add-on* Heroku Postgres. Heroku se encargará internamente de utilizar esta base de datos con la aplicación.

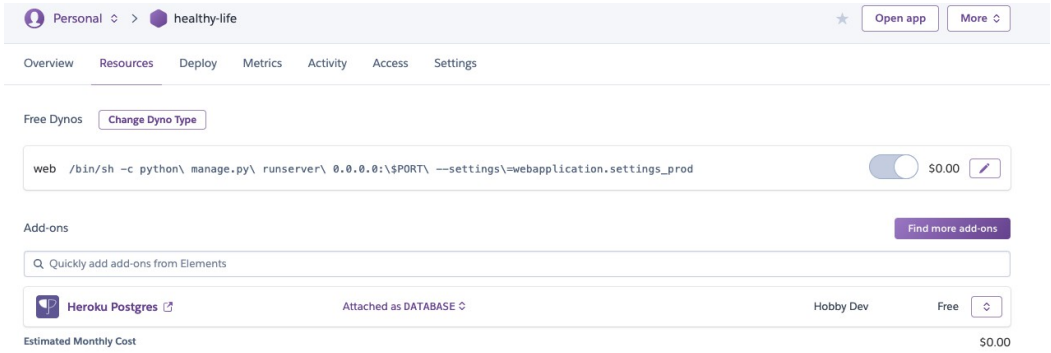

Ilustración 9: Heroku postgres

Una vez añadida la base de datos, es necesario añadir las siguientes variables de entorno:

- DJANGO_SETTINGS_MODULE
- SECRET_KEY

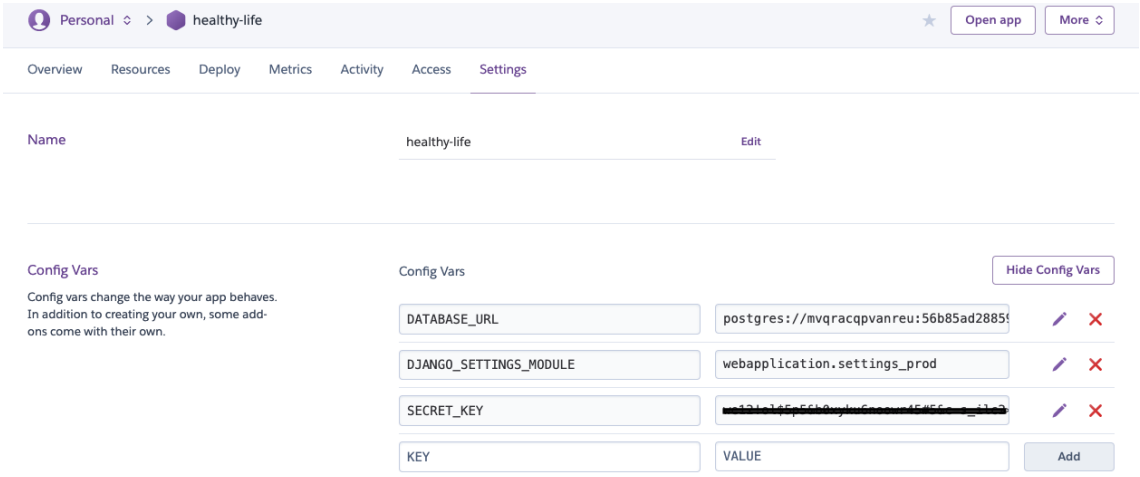

Ilustración 10: Variables de entorno Heroku

Una vez configurada la aplicación se puede proceder a realizar el despliegue de la aplicación mediante los siguientes comandos:

- Iniciar login en Heroku
  ```
  $ heroku login
  ```
- Iniciar login en el contenedor
  ```
  $ heroku container:login
  ```

- Construir el Dockerfile y subir la imagen de Docker
  ```
  $ heroku container:push web –app NombreApp
  ```

- Desplegar los cambios
  ```
  $ heroku container:release web –app NombreApp
  ```

Si es la primera vez que se despliega la aplicación, es necesario crear los modelos en la base de datos, para esto debe ejecutar los siguientes comandos:

```
$ heroku run bash –app NombreApp
$ python manage.py migrate
```

Con el primer comando se accede al bash del Dynos de la aplicación. Con el segundo comando se ejecuta el script para crear los modelos en la base de datos.

Por último, ejecutaremos el siguiente comando para añadir el usuario administrador de la aplicación:

```
$ python manage.py createsuperuser
```

A partir de este momento ya está el entorno de producción preparado para poder acceder mediante el botón “Open app” en Heroku o accediendo directamente a la url → https://nombreApp.herokuapp.com

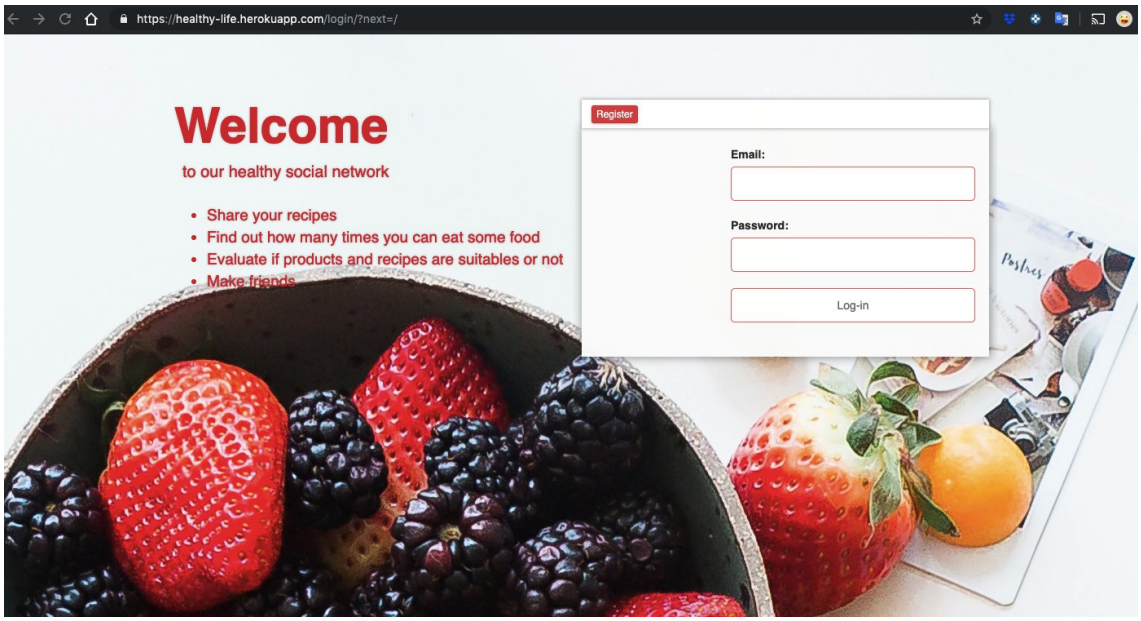

Ilustración 11: Home entorno de producción

4. Plan de pruebas

Núm.	Entrada	Acción esperada	Resultado
1	Registrarse como usuario con campo email, nombre, contraseña, fecha de nacimiento y tipo de dieta	El sistema crea el usuario en el sistema y redirige a la pantalla de login	
2	Registrase como usuario con campo email, nombre y contraseña	El sistema crea el usuario en el sistema y redirige a la pantalla de login	
3	Registrarse como usuario con un formato de email incorrecto	El sistema muestra mensaje de error	
4	Modificar como usuario el email por uno no existente en el sistema	El sistema guarda el nuevo email del usuario	
5	Modificar como usuario el email por uno existente	El sistema muestra mensaje de error y no guarda los cambios	
6	Modificar como usuario el nombre	El sistema guarda el nuevo nombre del usuario	
7	Modificar como usuario el nombre dejando campo nombre en blanco	El sistema muestra mensaje de error y no guarda los cambios	
8	Modificar como usuario la contraseña	El sistema guarda la nueva contraseña	No implementado
9	Modificar como usuario la fecha de nacimiento	El sistema guarda la nueva fecha de nacimiento	

10	Modificar como usuario el tipo de dieta	El sistema guarda el nuevo tipo de dieta	
11	Hacer login con email y contraseña correctas	El sistema muestra página de inicio	
12	Hacer login con email incorrecto	El sistema muestra mensaje de error	
13	Hacer login con email correcto y contraseña incorrecta	El sistema muestra mensaje de error	
14	Intentar acceder a una página sin hacer login	El sistema redirige a pantalla de login	
15	Como usuario administrador añadir tipos de dietas	El sistema guarda el nuevo tipo de dieta	
16	Como usuario añadir tipos de dietas	El sistema no permite añadir tipos de dietas	
17	Como usuario administrador eliminar receta de otro usuario	El sistema elimina la receta y sus valoraciones	
18	Como usuario no administrador eliminar receta de otro usuario	El sistema no permite eliminar la receta	
19	Como usuario administrador eliminar imagen de otro usuario	El sistema elimina la imagen y sus valoraciones	

20	Como usuario no administrador eliminar imagen de otro usuario	El sistema no permite eliminar la imagen	
21	Como usuario administrador eliminar valoración de una receta	El sistema elimina la valoración de la receta	
22	Como usuario no administrador eliminar valoración de una receta	El sistema no permite eliminar la valoración	
23	Como usuario administrador eliminar valoración de una imagen	El sistema elimina la valoración de la imagen	
24	Como usuario no administrador eliminar valoración de una imagen	El sistema no permite eliminar la valoración	
25	Listar recetas	El sistema lista todas las recetas	
26	Listar imágenes	El sistema lista todas las imágenes	
27	Como usuario añadir una imagen	El sistema guarda la imagen	
28	Como usuario editar una imagen	El sistema guarda la imagen con los datos nuevos	
29	Como usuario eliminar una imagen propia	El sistema elimina la imagen y sus valoraciones	

30	Como usuario añadir una receta cumplimentando título, ingredientes, tiempo de preparación, foto	El sistema guarda la receta	
31	Como usuario añadir una receta sin cumplimentar el campo título	El sistema genera un error y no guarda la receta	
32	Como usuario añadir una receta sin cumplimentar campo ingredientes	El sistema genera un error y no guarda la receta	
33	Como usuario añadir una receta sin cumplimentar campo tiempo de preparación	El sistema genera un error y no guarda la receta	
34	Como usuario editar una receta propia	El sistema guarda la receta con los nuevos datos	
35	Como usuario editar una receta de otro usuario	El sistema no permite editar la receta de otra usuario	
36	Como usuario valorar una receta	El sistema guarda la valoración	
37	Darse de baja como usuario	El sistema elimina el usuario	No implementado
38	Abrir aplicación web en Google Chrome	La aplicación se muestra correctamente	
39	Abrir aplicación web en Safari	La aplicación se muestra correctamente	

40	Abrir aplicación web en Internet Explorer	La aplicación se muestra correctamente	
41	Abrir aplicación en teléfono móvil	La aplicación se muestra correctamente	

El resultado del plan de pruebas es el siguiente:

- Satisfactorias: 37 - 90%
- No satisfactorias: 2 – 5%
- No implementadas: 2 – 5%

5. Conclusiones y futuros trabajos

5.1 Conclusiones

Una vez terminado el trabajo, teniendo en cuenta el resultado final y el plan de pruebas, se puede considerar en términos generales que el objetivo principal establecido al inicio, crear una red social, se ha cumplido satisfactoriamente.

En cuanto a los objetivos personales puedo destacar que:

- He aprendido a trabajar con Django, uno de los principales objetivos personales, a pesar de que la inversión de tiempo en conocer el framework ha sido mayor de lo que esperaba.
- He podido desplegar la aplicación en un entorno de producción, además me ha permitido conocer la plataforma Heroku, he disfrutado mucho aprendiendo a usarla y seguramente la utilice en futuros proyectos.
- He profundizado conocimientos en Bootstrap y sacar como conclusión que es un framework muy potente y que permite hacer diseños ágiles a personas que tienen pocos conocimientos sobre desarrollo *Frontend*.

Por otro lado, a pesar de que la valoración global ha sido positiva, hay que tener en cuenta que dos funcionalidades establecidas en los requisitos no se han implementado. En concreto:

- El sistema de seguimiento de usuarios.
- Restablecer contraseña: requería la configuración para el envío de correos que debido a la complejidad de la implementación se decidió aplazarla a una segunda fase.

Una inversión de horas mayor de la prevista para el aprendizaje del framework Django y algunas complicaciones a la hora de desplegar el sistema en producción han provocado que se decidiese dejar la implementación de estás funcionalidades para una segunda fase. Se consideró que, para esta primera versión de la aplicación, estas funcionalidades no eran críticas para el funcionamiento global de la aplicación.

Hay que destacar que la elección de la metodología en cascada ha permitido, a pesar de los imprevistos, reaccionar a tiempo y llegar a las fechas de entrega establecidas. Dadas las limitaciones de tiempo y los recursos disponibles, planificar el proyecto, acotar las funcionalidades al principio del proyecto y sobretodo establecer los hitos intermedios han sido piezas clave para lograr los objetivos.

5.2 Futuros trabajos

A pesar de que la aplicación es funcional y se puede usar sin problemas, en una siguiente fase se deberían de implementar las funcionalidades que han quedado pendientes en esta versión:

- Sistema de seguimiento de usuarios.
- Recuperación de contraseña.

Además, se citan a continuación una serie de mejoras y funcionalidades que harían que la aplicación fuese más atractiva para los usuarios.

A nivel técnico:

- Añadir una servidor de fichero en la nube para el contenido añadido por los usuarios, por ejemplo un *bucket* en S3 de AWS[15].
- Creación de test unitarios y de integración para facilitar el desarrollo de nuevas funcionalidades.

A nivel funcional:

- Mejoras de diseño para el navegador Internet Explorer y para dispositivos móviles.
- Añadir la posibilidad de hacer comentarios en las imágenes y las recetas.
- Añadir paginación en las listas de imágenes, recetas y usuarios.
- Añadir un marco del color de la valoración más votada a la foto o receta que permita a simple vista la frecuencia con la que se puede consumir un alimento o una receta.
- Añadir un sistema de mensajes privados entre usuarios.

6. Bibliografía

- **[1] Artículo**: AJ. Calañas-Continente*, D. Bellido. Bases científicas de una alimentación saludable. Revista de medicina. Universidad de Navarra, Vol 50, Nº 4, 2006.
- **[2] Artículo**: Sociedad Española de Nutrición Comunitaría. Guía de la alimentación saludable. 2004.
- **[3] Artículo**: SENC. Guía práctica sobre hábitos de Alimentación y Salud. Madrid, SENC-Instituto Omega 3, 2002.
- **[4] Web**: GanttProject: https://www.ganttproject.biz
- **[5] Web:** PyCharm: https://www.jetbrains.com/pycharm/documentation/
- **[6] Web:** Django: https://www.djangoproject.com/
- **[7] Web:** Bootstrap: https://getbootstrap.com/docs/3.3/getting-started/
- **[8] Web:** Python: https://docs.python.org/3/
- **[9] Web:** PostgreSQL: https://www.postgresql.org/docs/
- **[10] Web:** Docker: https://docs.docker.com/
- **[11] Web:** Docker Compose: https://docs.docker.com/compose/
- **[12] Web:** Bootstrap: https://www.w3schools.com/bootstrap/bootstrap_buttons.asp
- **[13] Web:** Heroku: https://devcenter.heroku.com/categories/reference
- **[14] Web**: Instalación de Docker Compose: https://docs.docker.com/compose/install/
- **[15] Web**: AWS – S3 Bucket: https://aws.amazon.com/es/s3/
- **Web:** Django Migrations: https://docs.djangoproject.com/en/2.2/topics/migrations/#backend-support
- **Libro**: Russ McKendrick, Scott Gallagher, Mastering Docker – Third Edition, Packt, Octubre 2018.
- **Libro**: Kamon Ayeva, Sakis Kasampalis, Mastering Python Design Patterns – Second Edition, Packt, Agosto 2018.
- **Web:** Pixabay: https://pixabay.com. Imágenes utilizadas en la aplicación.