Universitat Oberta de Catalunya

uoc.edu

TASTANT

Cristina de las Heras Santamaría
Grado en Ingeniería Informática
Desarrollo web

Vicenç Font Sagrista
Santi Caballé Llobet

09/06/2022

Agradecimientos

A Chus por apoyarme y confiar en mí de forma incondicional. Contigo al lado todo es más fácil.

A mi madre por hacerme quien soy.

FITXA DEL TREBALL FINAL

Títol del treball:	*Tastant, una red social gastronómica*
Nom de l'autor:	*Cristina de las Heras Santamaría*
Nom del consultor/a:	*Vicenç Font Sagrista*
Nom del PRA:	*Santi Caballé Llobet*
Data de lliurament (mm/aaaa):	*06/2022*
Titulació o programa:	*Grau d'Enginyeria Informàtica*
Àrea del Treball Final:	*Desarrollo web*
Idioma del treball:	*Castellà*
Paraules clau	*Laravel, React, Red Social*

Resum del Treball (màxim 250 paraules): *Amb la finalitat, context d'aplicació, metodologia, resultats i conclusions del treball*

El objetivo de este trabajo es desarrollar una aplicación web para compartir información de carácter gastronómico entre un grupo de personas con intereses comunes. Esta aplicación permite interactuar a usuarios con intereses comunes. Los usuarios pertenecen a grupos y la información compartida en un grupo sólo es visible por los miembros del grupo.

Es una aplicación RESTful con el backend API desarrollado con Laravel y el frontal con React.

Abstract (in English, 250 words or less):

The aim of this project is building a web application to share gastronomic information between people who have the same interests. This application allows the users to interact with other people with the same interests. Users belong to groups and the information shared on the group only is visible by the group's members.

This is a RESTful application building with Laravel and the frontend with React.

Índice

1. Introducción

1.1 Contexto y justificación del Trabajo

Este proyecto busca cubrir la necesidad de compartir información de carácter gastronómico entre un grupo de personas con intereses comunes.

El antecedente de este proyecto es un grupo de personas que utilizan la herramienta de mensajería instantánea WhatsApp para compartir recomendaciones gastronómicas (restaurantes, vinos, recetas, bares...) en un grupo. Sin embargo, este método no es completamente satisfactorio, ya que, a medida que el número de recomendaciones realizadas crece, también crece la dificultad para localizar una en concreto.

Uno de los objetivos fundamentales de las redes sociales es compartir información e interactuar con otros usuarios. Como en esta plataforma se va a compartir información de tipo gastronómico, podríamos definir esta aplicación como una red social gastronómica cuyo objetivo principal es la compartición de información gastronómica e interacción con otros usuarios con intereses comunes.

Actualmente, existen aplicaciones como TripAdvisor, en las que los usuarios opinan y recomiendan, pero estas opiniones y recomendaciones, generalmente, provienen de personas que no conocemos. Este hecho, en ocasiones, provoca desconfianza, bien sea porque dudamos de la imparcialidad o del criterio que se ha seguido para confeccionar la crítica.

El punto fuerte de esta aplicación es que la información se comparte entre personas que ya conocemos y con las que, previamente, tenemos establecido un nivel de confianza en sus recomendaciones. Sin duda, daremos más relevancia a las recomendaciones recibidas de una persona que ya conocemos y tenemos un buen criterio de sus recomendaciones y gusto, que si la recibimos de una persona desconocida.

Las personas interesadas del proyecto son todas aquellos usuarios con intereses gastronómicos, que quieran compartir con otros usuarios sus lugares favoritos para comer, beber o pasar el rato. Además, las críticas realizadas por los usuarios podrían afectar a las costumbres de consumo de otros usuarios, afectando positiva o negativamente a restauradores.

1.2 Objetivos del Trabajo

El objetivo principal de este proyecto es el desarrollo de una aplicación web que permita la interacción de usuarios con intereses gastronómicos comunes. Los usuarios pertenecen a grupos y la información compartida en un grupo sólo es visible por los miembros del grupo.

Los usuarios pueden recomendar restaurantes, bares, recetas y vinos, comentar, puntuar, guardar y buscar recomendaciones de otros usuarios de sus grupos y organizar eventos.

1.3 Enfoque y método seguido

Dado que en el mercado no había ningún producto que satisficiera las necesidades planteadas por los stakeholders, se ha optado por el desarrollo de un producto nuevo, un desarrollo desde cero.

1.4 Planificación del Trabajo

Para la realización de este proyecto se contará con un único recurso, que se encargará de la realización de todas las tareas involucradas en el proyecto.

Para la realización de la planificación se ha tenido en cuenta la programación temporal y la carga de las diferentes entregas de las PACS.

En la PAC1 se definirá el plan de trabajo del proyecto, las tecnologías que se van a utilizar y los requisitos a alto nivel. Después, en la PAC2 se realizará el análisis de requisitos y el diseño técnico y visual. En la PAC3, se llevará a cabo la fase de codificación y testing. Por último, el objetivo de la PAC4 es preparar la memoria y la defensa.

Las fases principales del proyecto son:

- Planificación
- Análisis de requisitos y diseño
- Codificación y *testing*
- Memoria final

Para gestionar el proyecto se va a utilizar la herramienta Trello. Esta herramienta nos va a permitir tener una visión global del estado del proyecto y del estado de las diferentes tareas. Vamos a crear un tablero con las siguientes columnas: TO DO, DOING, DONE and REVIEW.

Id	Nombre	Inicio	Finalización	Duración (días)
1	Plan de trabajo:	16/02/2022	01/03/2022	14
1.1	Valoración diferentes propuestas	16/02/2022	21/02/2022	6
1.2	Investigación y selección tecnologías, frameworks y herramientas	16/02/2022	21/02/2022	6
1.3	Elaboración PAC1	16/02/2022	01/03/2022	14
1.4	Recogida de requisitos a alto nivel	22/02/2022	26/02/2022	5
1.5	Planificación del proyecto	27/02/2022	01/03/2022	3
2	Análisis y Diseño:	02/03/2022	01/04/2022	31
2.1	Análisis de requisitos	02/03/2022	05/03/2022	4
2.2	Elaboración documento PAC2	02/03/2022	01/04/2022	31
2.3	Diseño de clases	06/03/2022	09/03/2022	4
2.4	Diseño de la Base de Datos	10/03/2022	14/03/2022	5
2.5	Adquirir conocimiento sobre Laravel y React	13/03/2022	01/04/2022	20
2.6	Diseño arquitectónico	15/03/2022	22/03/2022	8
2.7	Diseño de la interfaz gráfica de usuario y navegación	23/03/2022	29/03/2022	7
3	Codificación y testing:	02/04/2022	23/05/2022	52
3.1	Creación Bases de Datos	02/04/2022	02/04/2022	1
3.2	Elaboración documento PAC3	02/04/2022	23/05/2022	52
3.3	Codificar frontend REACT	03/04/2022	23/04/2022	21
3.4	Desarrollo pruebas unitarias	03/04/2022	14/05/2022	42
3.5	Codificar backend LARAVEL	24/04/2022	14/05/2022	21
3.6	Desarrollo pruebas de integración	14/05/2022	17/05/2022	4
3.7	Desarrollo de pruebas de sistema y aceptación	18/05/2022	18/05/2022	1
3.8	Despliegue en el entorno de producción	19/05/2022	21/05/2022	3
4	Memoria final:	24/05/2022	09/06/2022	17
4.1	Elaboración documento final	24/05/2022	09/06/2022	17
4.2	Propuesta de mejoras y nuevas funcionalidades	24/05/2022	26/05/2022	3
4.3	Preparación presentación virtual del trabajo	05/06/2022	09/06/2022	5

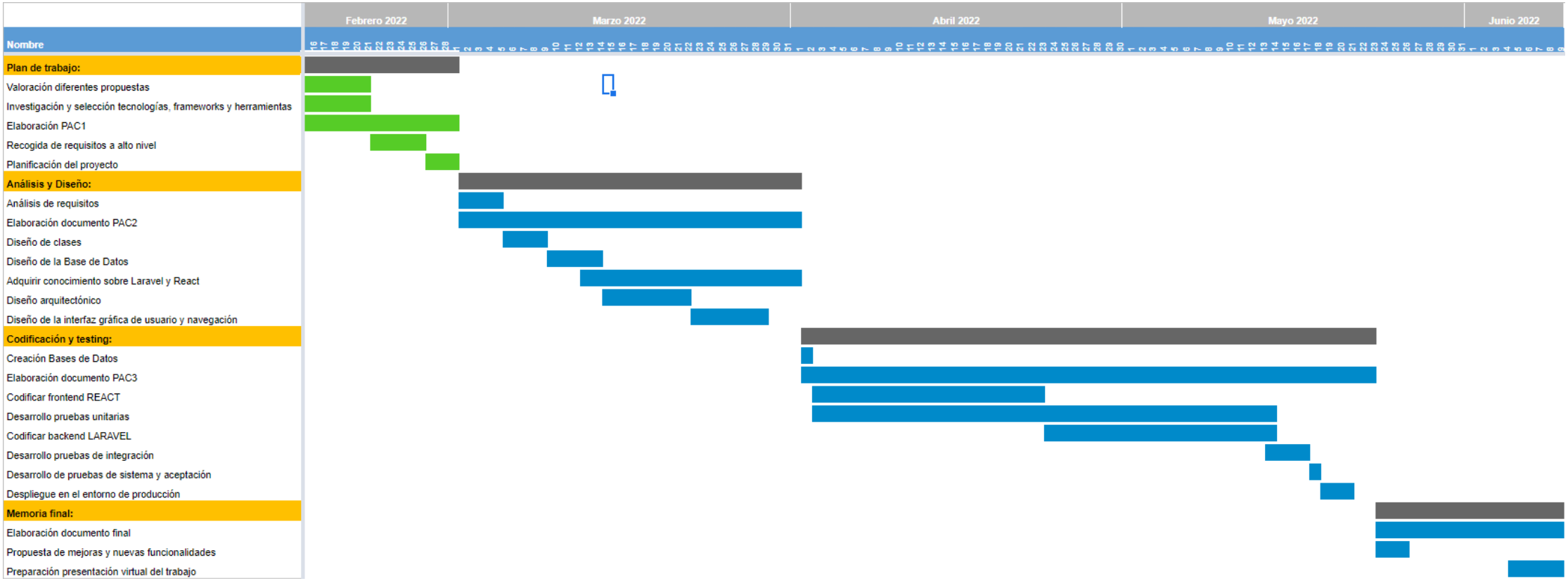


Ilustración 1: Diagrama de Gantt realizado con https://instagantt.com/

1.5 Breve resumen de productos obtenidos

Como producto final del trabajo se va a obtener una aplicación web, que permitirá a los usuarios registrados crear grupos en los que compartir con el resto de miembros recomendaciones gastronómicas.

A continuación vamos a listar todos los diferentes resultados de este proyecto:

- El frontal de la aplicación desarrollado en React: https://github.com/ayoira/tfg-app-frontend.git
- El backend API de la aplicación desarrollado en Laravel: https://github.com/ayoira/tfg-app-backend.git
- Memoria del proyecto: memoria_tfg_cde_las_heras.pdf
- Presentación del proyecto: presentacion_tfg_cde_las_heras.pdf
- Vídeo presentación del proyecto

1.6 Herramientas y tecnología

Las herramientas y tecnologías que se van a utilizar son las siguientes:

Herramienta / Tecnología	Objetivo
React js	Desarrollo frontend
Laravel, framework de PHP	Desarrollo backend API
CSS	Maquetación
Javascript	Interacción usuarios
Git	Control de versión
Github/Gitlab	Gestor de repositorio
Heroku / DigitalOcean	Aprovisionamiento del entorno producción
Trello	Gestión de proyectos
Studio Code / PHPStorm	Entorno de desarrollo
Postman	Herramienta de testing APIs
Docker	Aprovisionamiento del entorno local
Bootstrap 5.2	Front End framework

MySQL Workbench	Cliente SQL
Chrome	Navegador
Chrome: Developer tools	Debug y testing
MailHog	Email testing
Sendgrid	Proveedor email del entorno producción
S3 AWS	Almacenamiento estático en la nube

React JS. Es una librería de JavaScript para desarrollar interfaces de usuario basadas en componentes.

Laravel. Es un framework de PHP para desarrollar aplicaciones web. Implementa el patrón de diseño MVC.

MySQL. Es un sistema de gestión de bases de datos relacionales. Es un sistema rápido, no supone ningún coste su uso, multiplataforma y ampliamente usado.

Bootstrap 5.2. Es un *framework* de *frontend* de código abierto que contiene plantillas de diseño basadas en HTML, CSS y JavaScript.

Nginx. Es un servidor web.

PHP 8.1. Es un lenguaje de programación usado principalmente para el desarrollo web.

La máquina en la que se ejecutará la aplicación tendrá como mínimo los siguientes módulos y librerías: compose, build-essential, libpng-dev, libjpeg62-turbo-dev, libfreetype6-dev, zip, vim, libzip-dev, unzip, git, libonig-dev, curl, pdo_mysql, mbstring, exif y pcntl.

Se ha seleccionado este stack tecnológico porque tanto Laravel como React son tecnologías modernas y muy populares en el desarrollo web y, por este motivo, podremos encontrar documentación extensa sobre estos *frameworks*. Además, la curva de aprendizaje de ambos *frameworks* es menor con respecto a otros como Symfony o Angular.

1.7 Breve descripción de los demás capítulos de la memoria

Los demás capítulos de la memoria se centran en documentar las fases de diseño, implementación y testing.

En primer lugar, se exponen los requisitos de la aplicación. En este apartado se describe el comportamiento y funcionalidades que ha de satisfacer la aplicación.

Después, nos centraremos en el diseño de la aplicación. Aquí se definirán los casos de uso, el diagrama de clase, el diseño de la Base de Datos, los prototipos de las diferentes pantallas de la aplicación y la identidad corporativa de la aplicación.

El siguiente capítulo es el más técnico, aquí se detalla la arquitectura de los diferentes entornos de la aplicación, las principales decisiones que se han tomado con respecto a la implementación, las instrucciones para levantar la aplicación en el entorno local y las pruebas que se han realizado.

A continuación, se realiza una presentación de los resultados obtenidos, haciendo un recorrido por las principales pantallas de la aplicación destacando las principales funcionalidades en cada una de ellas.

Por último, las conclusiones, el glosario y la bibliografía.

2. Requisitos del proyecto

2.1 Requisitos principales

- Crear un sistema de autenticación que permita a los usuarios de la aplicación registrarse, iniciar sesión, cerrar sesión y recuperar credenciales. El acceso a la aplicación se realizará mediante correo electrónico y contraseña.

- El sistema debe identificar de manera unívoca a los usuarios por su cuenta de correo electrónico.

- Crear un sistema de autorización que permita únicamente a los usuarios autorizados el acceso a los recursos.

- El sistema tiene que ofrecer distintos perfiles de usuarios (administrador portal, administrador grupo y usuario normal).

- El sistema ha de disponer de un panel de administración en la que los usuarios administradores podrán visualizar y gestionar todos los usuarios registrados, los grupos existentes en el sistema, las entradas existentes y las categorías de recomendaciones.

- Crear una página principal, tipo *landing page*, en la se ofrece información sobre la funcionalidad y objetivo de la plataforma y en la que los usuarios podrán registrarse e iniciar sesión.

- Crear una página privada de usuario en la que el usuario pueda visualizar y gestionar su información personal, los grupos a los que pertenece, la información que ha publicado y compartido con otros usuarios y visualizar las recomendaciones guardadas.

- Crear una página privada de grupo en la que los usuarios miembros del grupo puedan visualizar las últimas recomendaciones, los miembros del grupo, buscar recomendaciones, invitar nuevos usuarios al grupo.

- Crear una página de detalle de una recomendación en la que se podrán visualizar el detalle de la recomendación, su puntuación media y los comentarios asociados.

- El sistema ha de permitir personalizar la información de cada usuario y de cada grupo.

- Crear un sistema que permita a los usuarios realizar recomendaciones a los miembros de sus grupos. Estas recomendaciones tendrán visibilidad a nivel de grupo. Los usuarios han de poder comentar, puntuar y guardar las recomendaciones de otros usuarios.

- Crear un buscador avanzado que permita buscar recomendaciones en el ámbito del grupo. Estas recomendaciones se podrán filtrar y ordenar por tipología y puntuación.

- El sistema notificará a los miembros del grupo cuando algún miembro realice una nueva recomendación.
- El sistema ha de ofrecer diferentes vistas para las recomendaciones y eventos (mapa, lista...).
- El sistema ha de ofrecer la posibilidad de organizar eventos a nivel de grupo. El sistema notificará a todos los miembros la existencia de un nuevo evento.
- Los usuarios han de poder aceptar o rechazar la asistencia a un evento al que haya sido invitado.
- Solo los usuarios registrados pueden crear grupos. Cuando un usuario crea un grupo, el sistema automáticamente le asignará el rol de administrador del grupo.
- El sistema debe permitir al administrador del grupo añadir, eliminar usuarios del grupo, otorgar el rol de administrador a otros usuarios del grupo, editar la información del grupo.
- Cuando un usuario es invitado a un grupo, el sistema debe enviar un correo electrónico al usuario invitado.
- Un usuario debe tener como mínimo los siguientes campos: un identificador, un nombre, un correo electrónico y un rol.
- Un grupo debe tener como mínimo los siguientes campos: un identificador, un nombre, una descripción y una fotografía.
- Un comentario debe tener como mínimo los siguientes campos: un identificador, comentario.
- Una recomendación debe tener como mínimo los siguientes campos: un identificador, un título, una descripción, una fotografía.
- Un evento debe tener como mínimo los siguientes campos: un identificador, una fecha o rango de fechas, una ubicación, un título, una descripción y una fotografía.

2.2 Requisitos secundarios

- Poner en práctica los conocimientos adquiridos durante el estudio de las diferentes asignaturas que componen el grado.
- Aprender nuevas tecnologías, arquitecturas y metodologías web.

2.3 Requisitos no funcionales

- El sistema ha de ser fácil de usar.
- El sistema ha de cumplir con los estándares de usabilidad y accesibilidad.
- La aplicación debe ser accesible desde los navegadores más usados (Firefox, Safari, Explorer) y distintos dispositivos (pc, tablet, teléfono…).
- El sistema ha de garantizar las propiedades de confidencialidad y autenticación.
- El sistema ha de garantizar la privacidad de la información compartida en un grupo, la cual solo puede ser visible por los miembros del grupo.

3. Diseño

3.1 Casos de uso

Una vez definidos los requisitos y la funcionalidad de la aplicación, vamos a describir las principales acciones y actividades que se pueden realizar en la aplicación a través del diagrama de casos de uso. En este diagrama se indica quién realiza la acción.

En Tastant podemos encontrar los siguientes actores:

- Usuario no autenticado
- Usuario
    - Autor de una entrada
    - Miembro de un grupo
    - Administrador de grupo
- Administrador de la aplicación

Además, los diferentes casos se han agrupado por componentes.

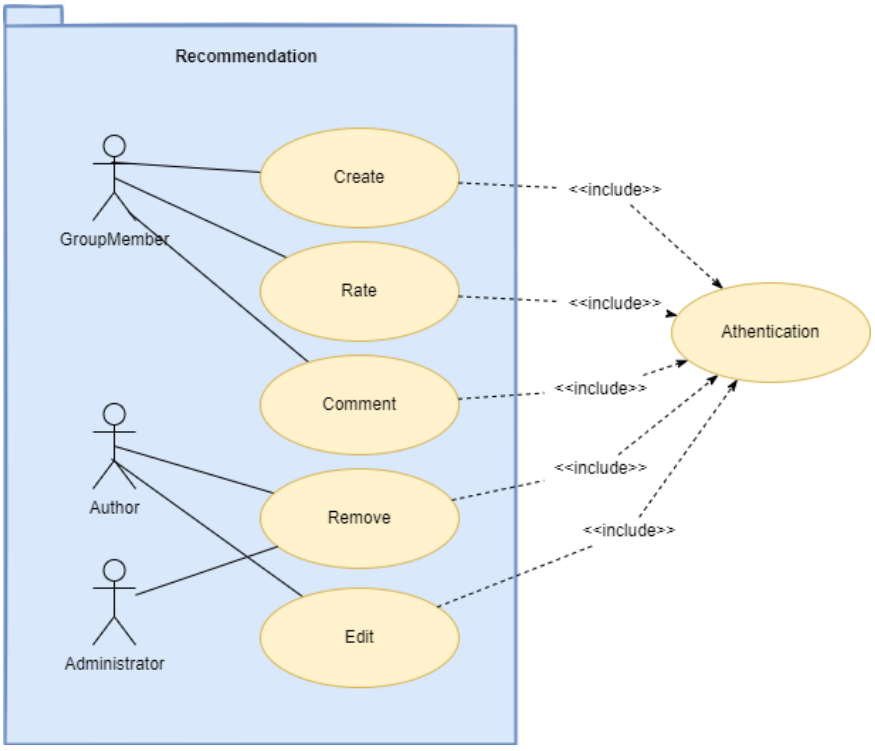


Ilustración 2: Casos de uso - Componente Recommendation

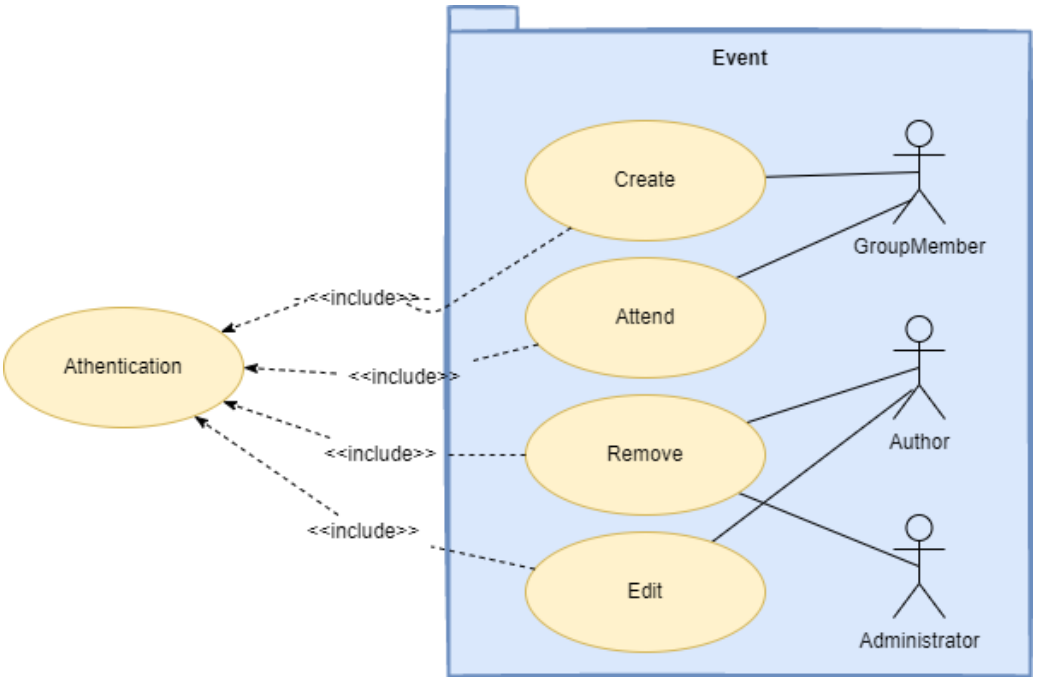


Ilustración 2: Casos de uso - Componente Event

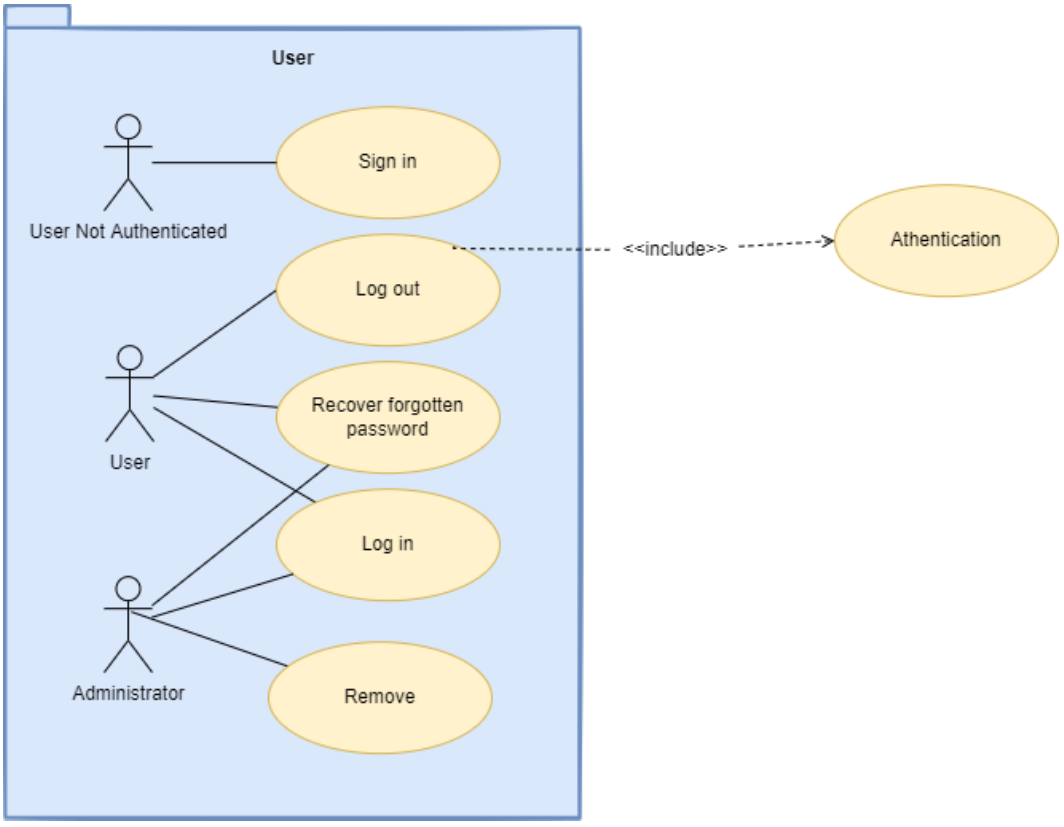


Ilustración 3: Casos de uso - Componente User

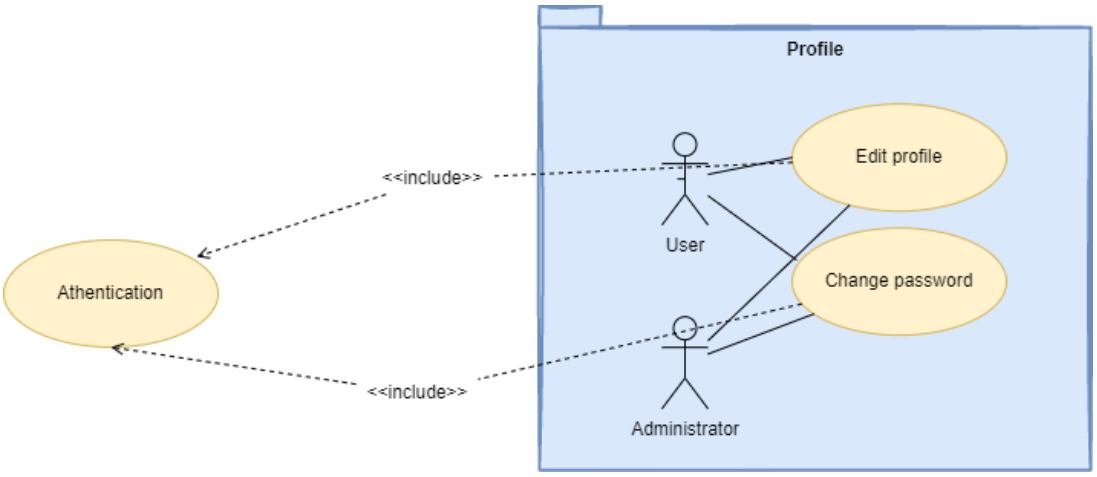


Ilustración 4: Casos de uso - Componente Profile

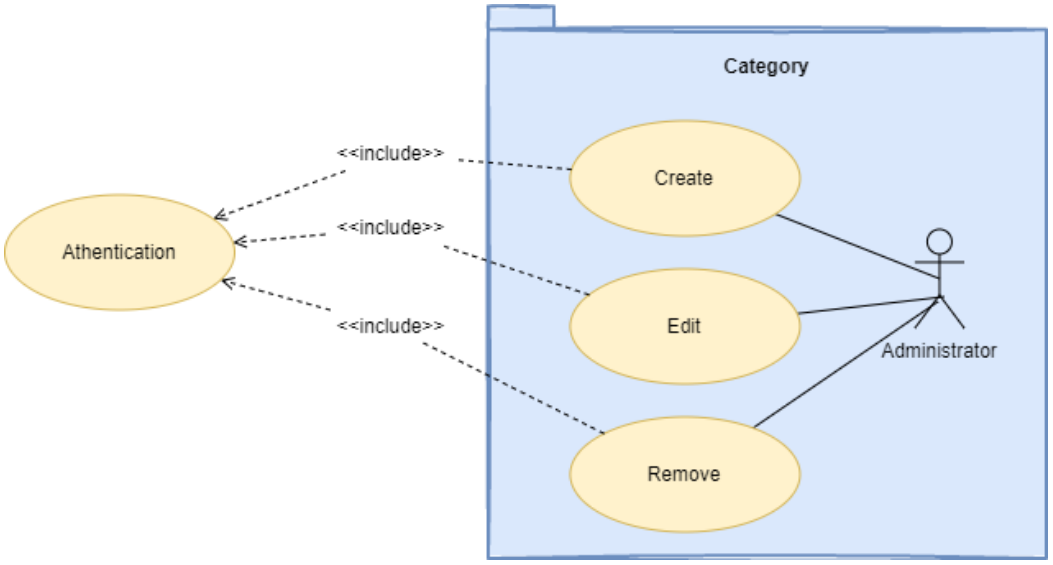


Ilustración 5: Casos de uso - Componente Category

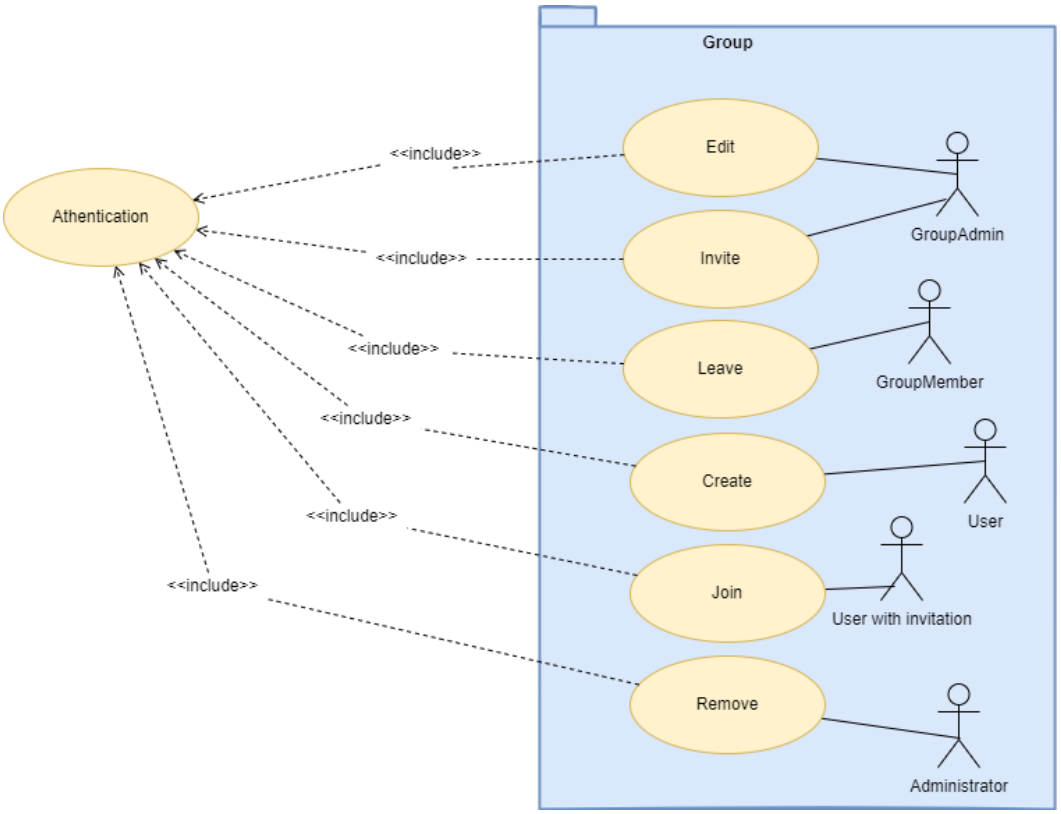


Ilustración 6: Casos de uso - Componente Group

3.2 Diagrama de clases

En este apartado vamos a describir de forma estática la estructura del sistema, sus relaciones y principales atributos.

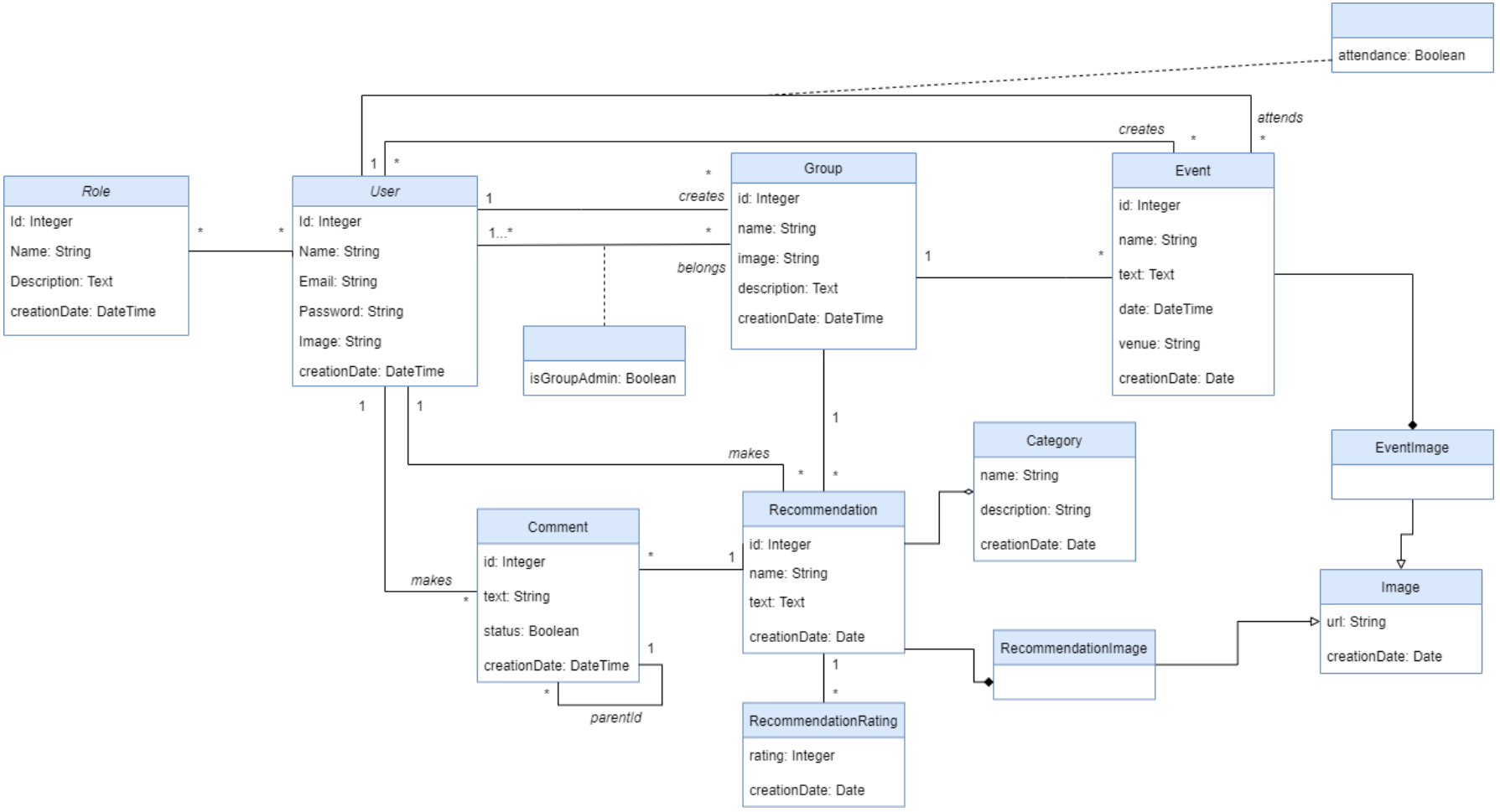


Ilustración 7: Diagrama de clases

3.3 Diseño de la base de datos

Laravel incluye Eloquent un ORM (Object Relational Mapping) que permite asociar las estructuras de una base de datos relacional con una estructura lógica de clases/entidades. Esto permite simplificar las tareas de persistencia de datos.

A continuación se muestra el modelo generado por el ORM:

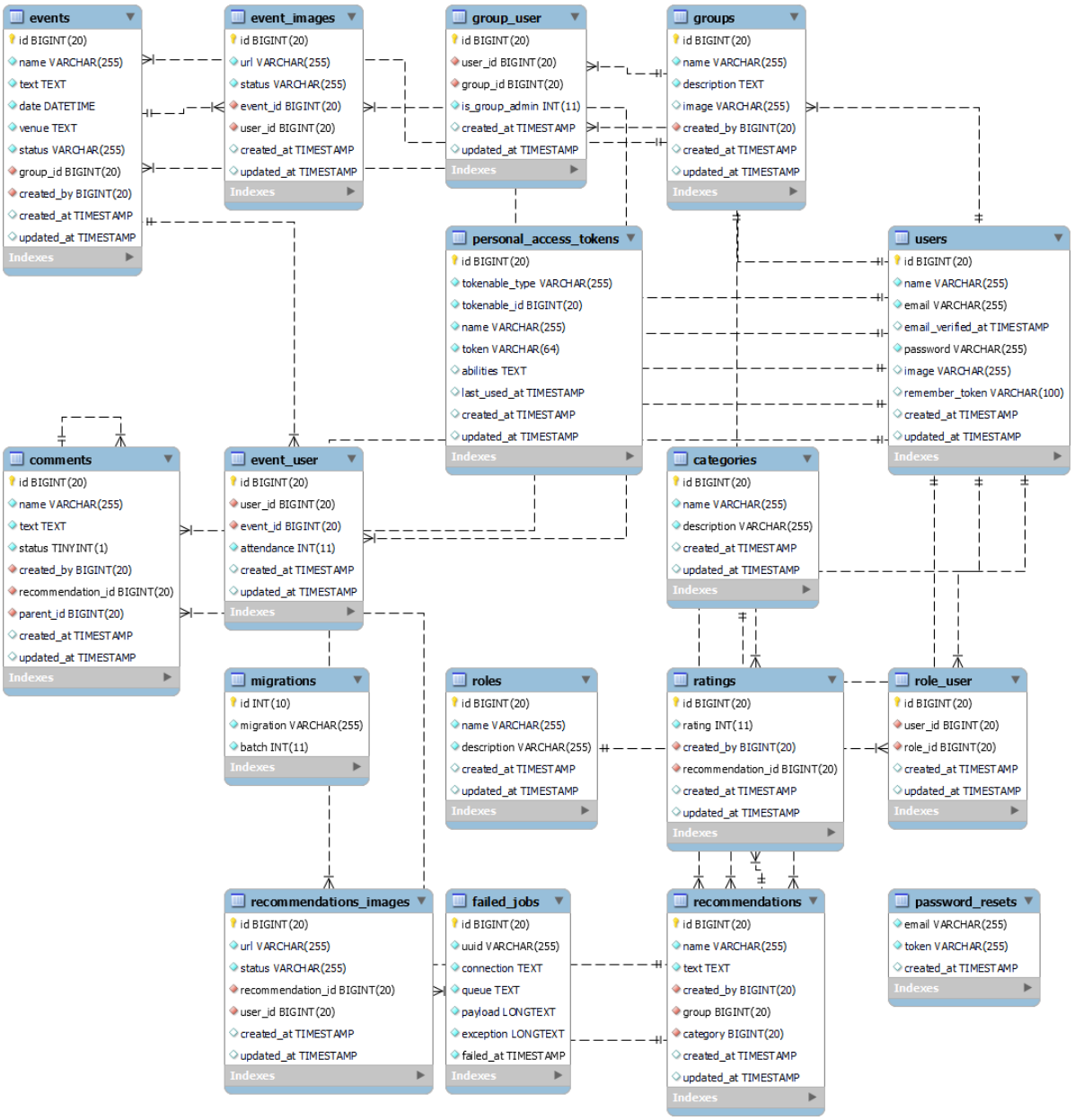


Ilustración 8: Diseño de la Base de Datos

Esto se consigue a través de la definición de modelos y migraciones.

3.4 Prototipos

Para el desarrollo de estos prototipos se ha utilizado la herramienta: https://www.mockflow.com/apps/wireframepro/.

3.4.1 Home

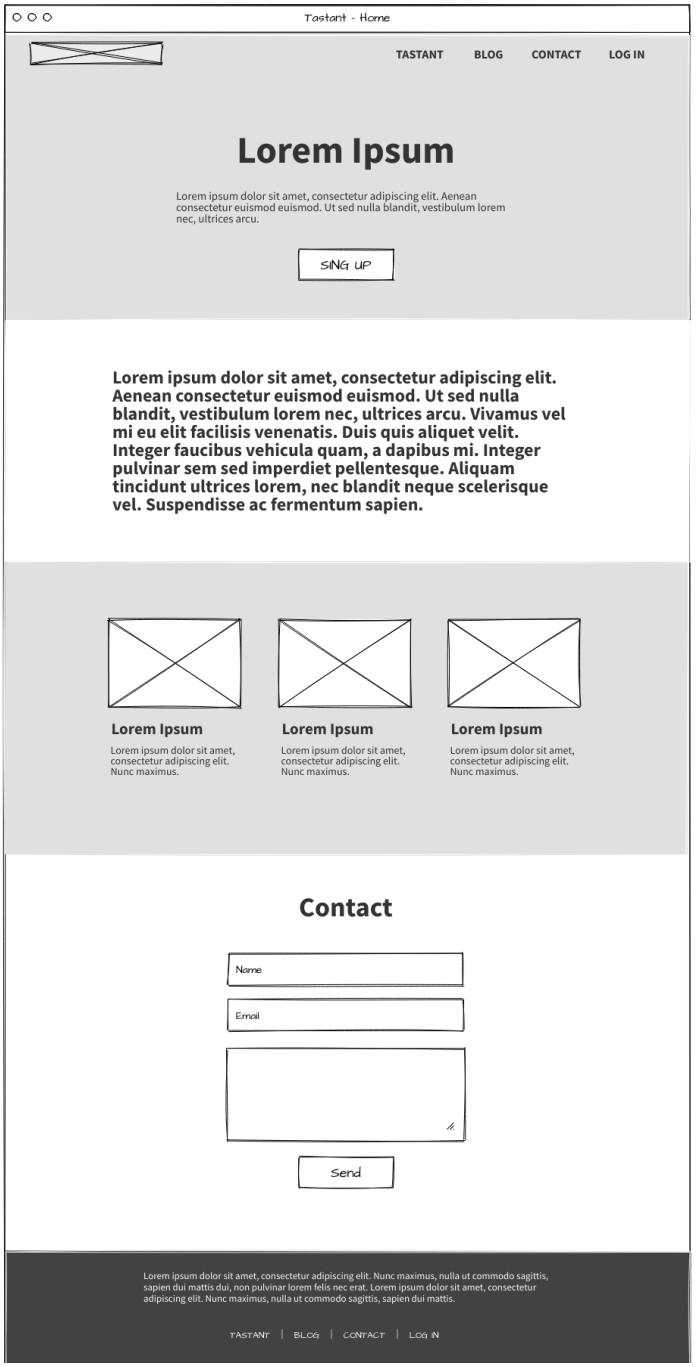


Ilustración 9: Wireframes - Página principal

3.4.2 Autenticación

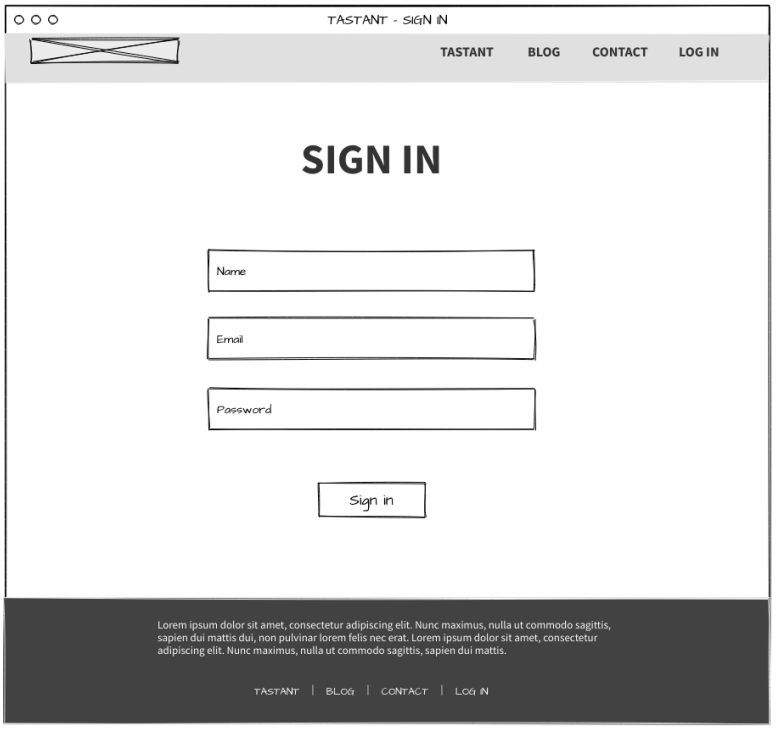


Ilustración 10: Wireframes - Página de registro

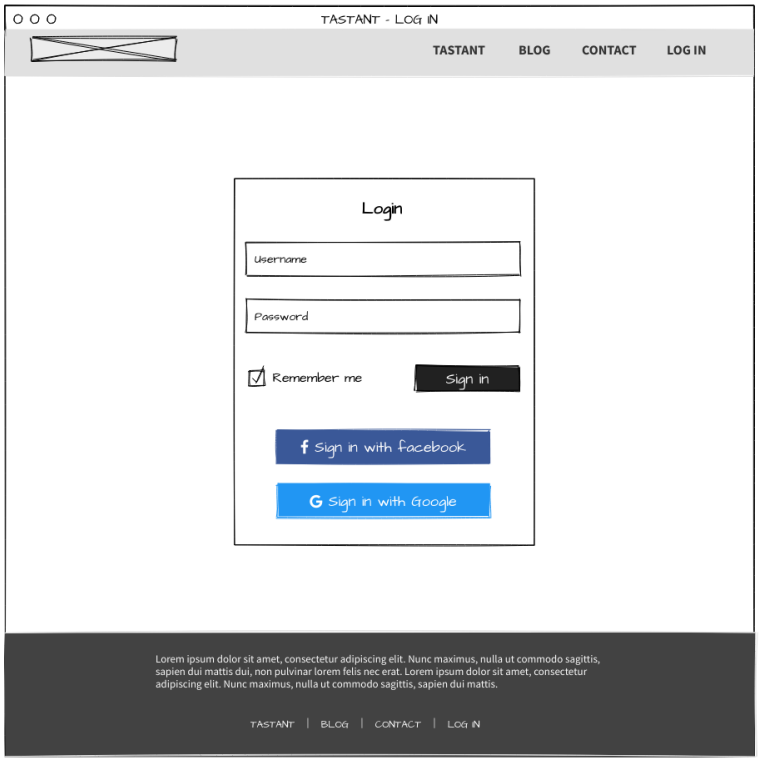


Ilustración 11: Wireframes - Página de Log in

3.4.3 Página de grupo

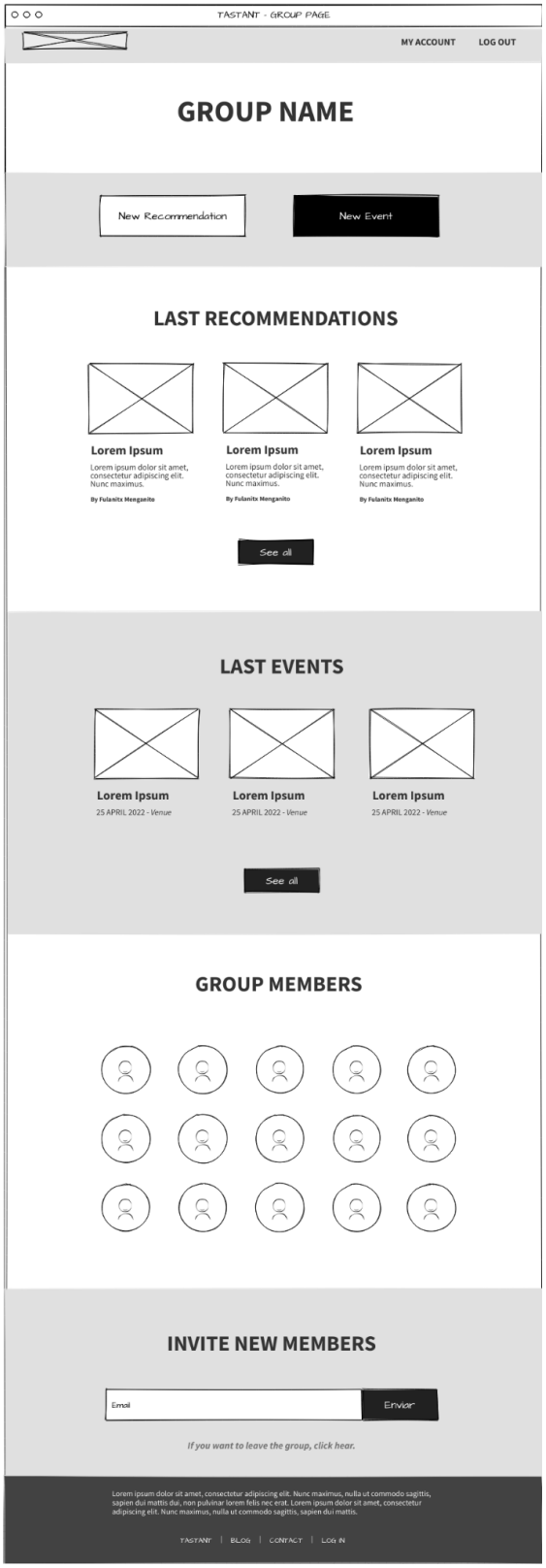


Ilustración 12: Wireframes - Página de grupo

3.4.4 Página de recomendación

Ilustración 13: Wireframes - Página de una recomendación

3.4.5 Página de evento

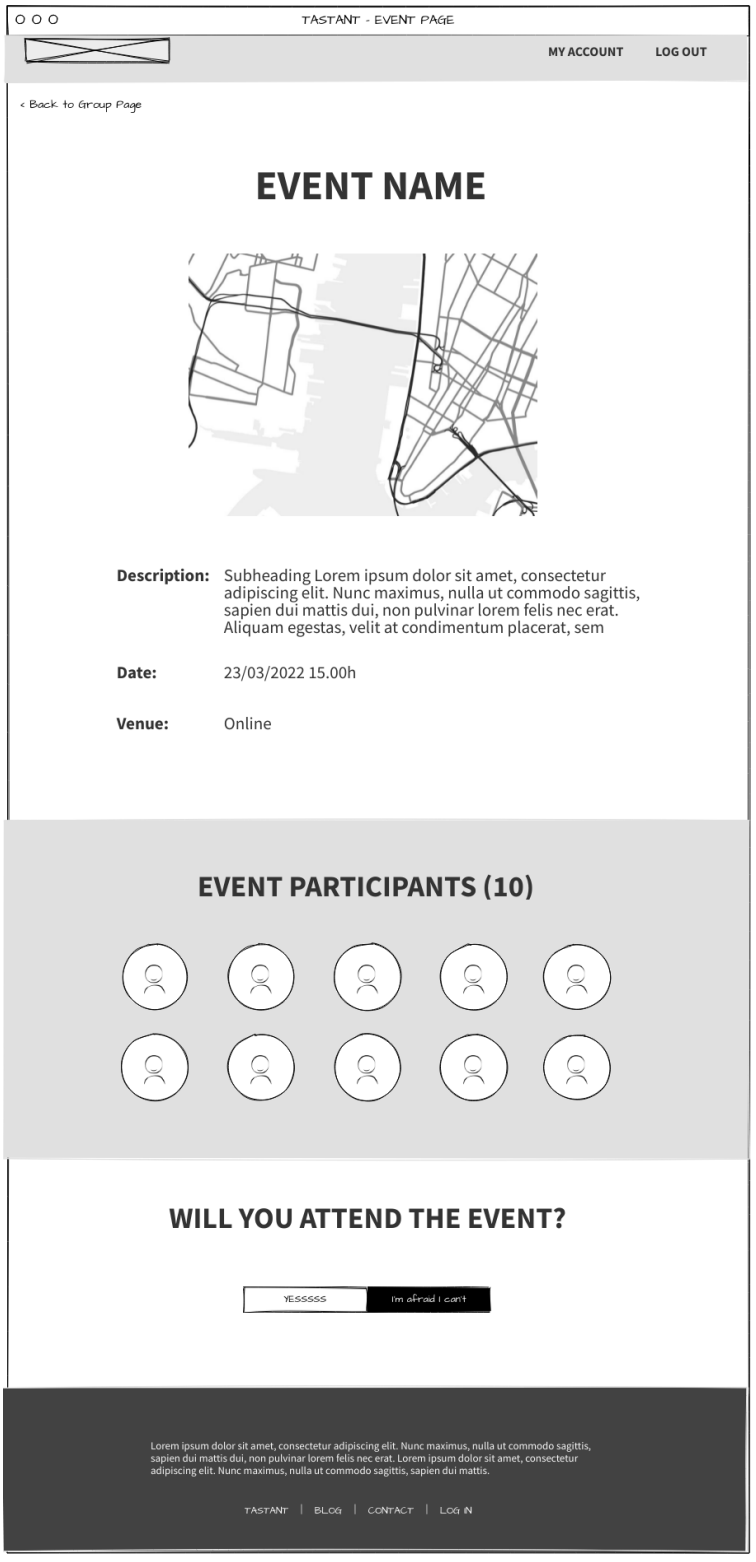


Ilustración 14: Wireframes - Página de un evento

3.4.6 Página cuenta usuario

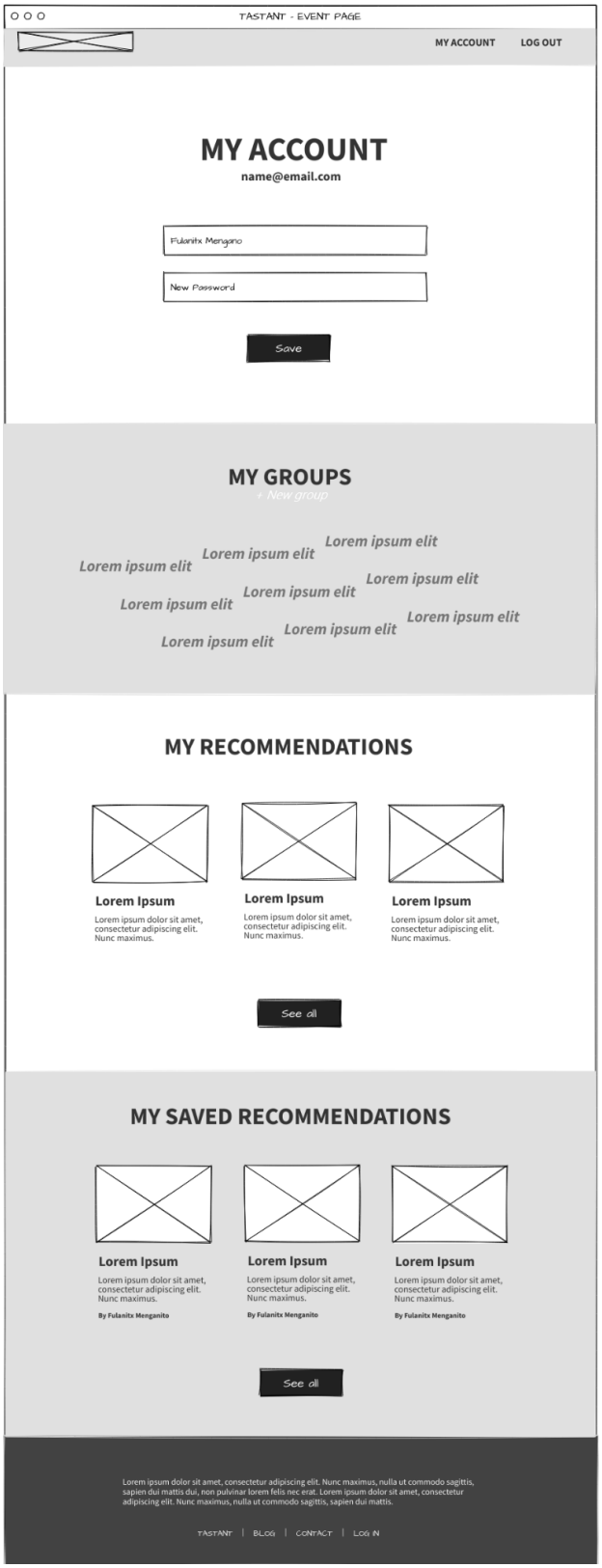


Ilustración 15: Wireframes - Página personal de usuario

3.4.7 Página panel de administración

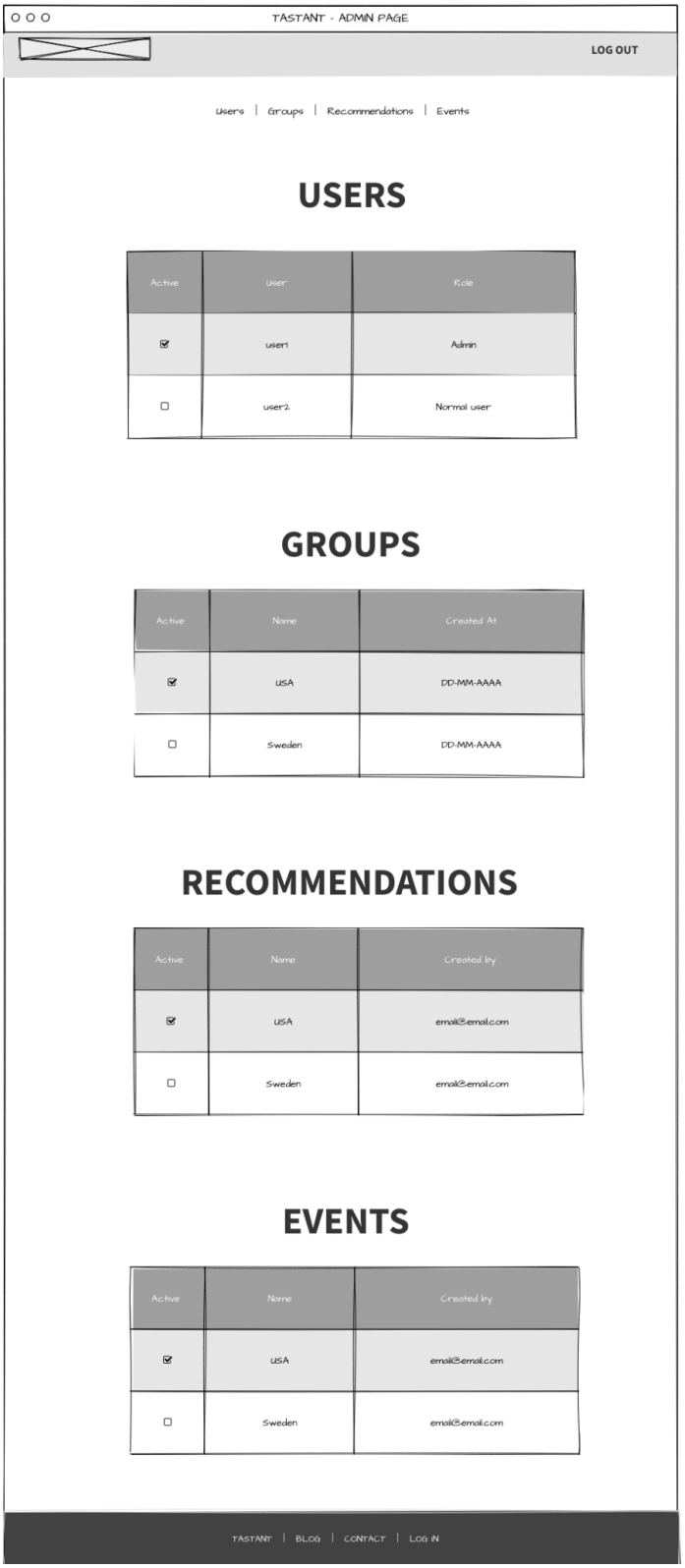


Ilustración 16: Wireframes - Página panel de administración

3.4.8 Página listado de las recomendaciones del grupo

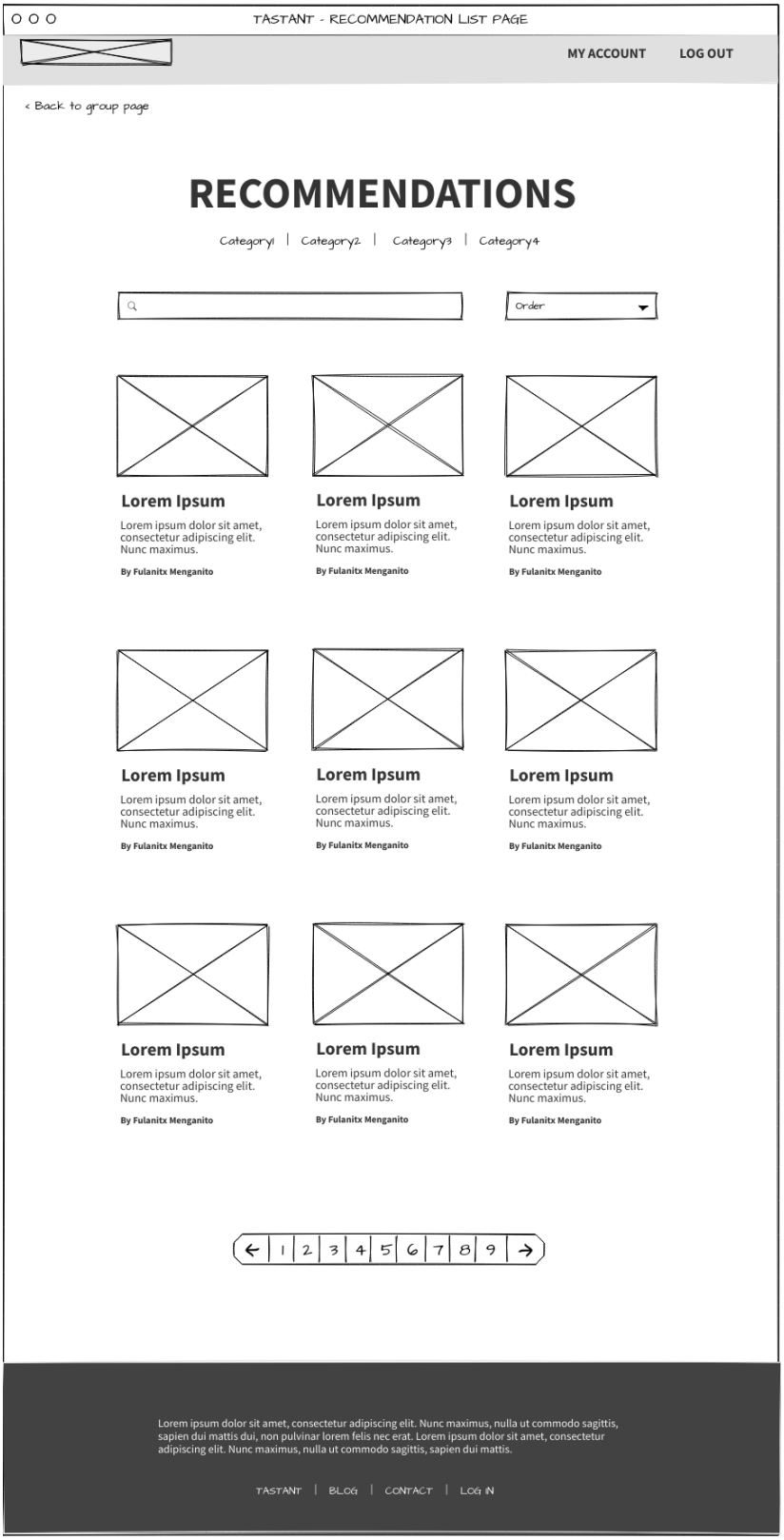


Ilustración 17: Wireframes - Página listado de recomendaciones

3.4.9 Página listado de las eventos del grupo

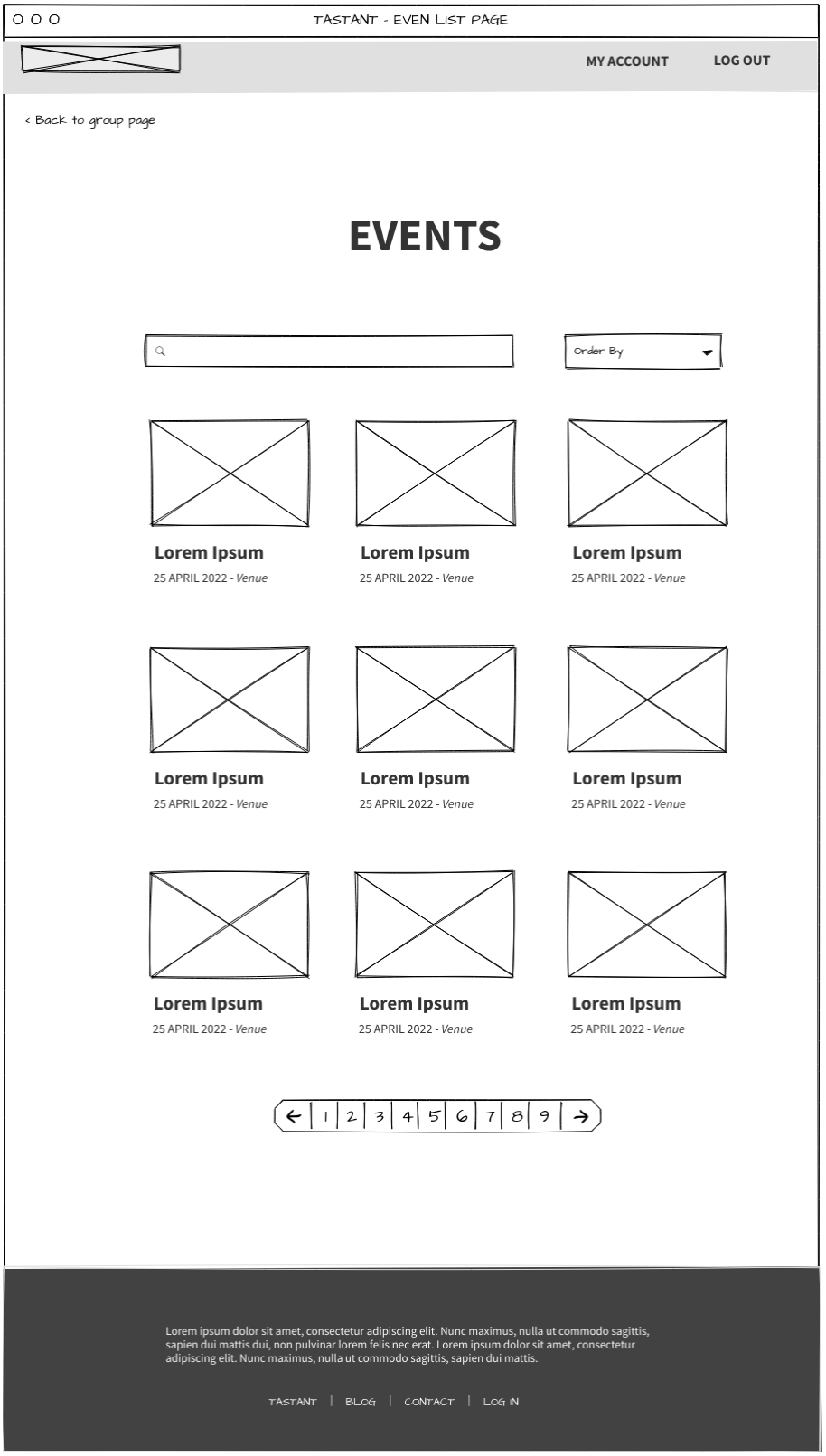


Ilustración 18: Wireframes - Página listado de eventos

3.4.10 Páginas para añadir una nueva recomendación, grupo y evento

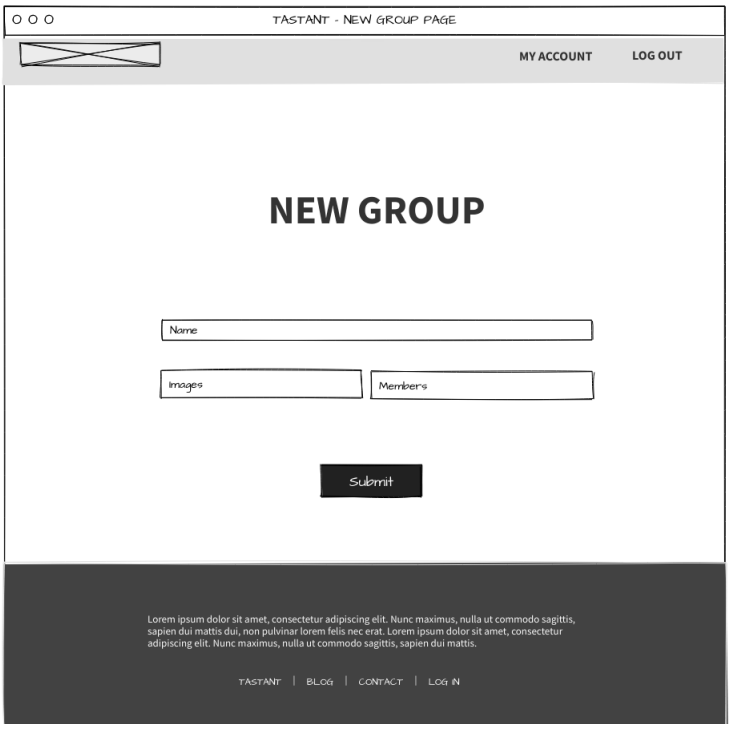


Ilustración 19: Wireframes - Página nuevo grupo

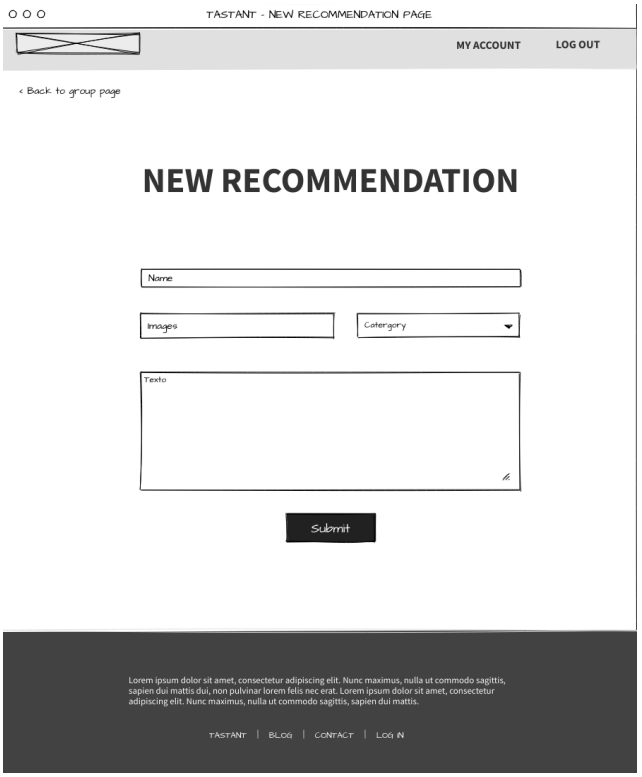


Ilustración 20: Wireframes - Página nueva recomendación

Ilustración 21: Wireframes - Página nuevo evento

3.5 Identidad corporativa

Mediante los diferentes elementos gráficos se ha buscado transmitir la sensación de desenfado de una ligera sobremesa sin formalismos entre amigos y familia.

Para conseguir este objetivo nos apoyamos en el esquema de color, en el logotipo e imágenes inspiracionales.

La tipografía que se ha utilizado en el proyecto es: https://fonts.google.com/specimen/Open+Sans?query=open+sans.

3.5.1 Esquema de color

El esquema de color ha sido generado aleatoriamente con la aplicación https://coolors.co/5d2a42-fb6376-fcb1a6-ffdccc-fff9ec.

El color principal de este esquema de color es el color del vino. Esta gama cromática transmite alegría y calidez.

#5D2A42	#FB6376	#FCB1A6	#FFDCCC	#FFF9EC

3.5.2 Logotipo

El logo es una producción propia y a medida. Este logo representa la mancha que puede dejar una copa de vino sobre un mantel. El color de la mancha es granate como el vino. En el centro la ‘t’ de Tastant en minúscula y del mismo color que el rodal.

Se ha buscado que transmita familiaridad y cercanía, ya que estos valores son los que se busca promover dentro de la aplicación.

Ilustración 22: Logo tastant

3.5.3 Imágenes

Las imágenes usadas en la aplicación tienen derecho de libre uso. Han sido obtenidas a través de la aplicación https://unsplash.com/es/licencia, excepto:

- Las dos imágenes de las pizarras son https://www.shutterstock.com/ (descarga gratuita) (https://www.shutterstock.com/es/search/pizarra%2Bvacia).
- Las imágenes de los usuarios han sido generadas con la aplicación https://getavataaars.com/.

4. Implementación

4.1 Arquitectura

4.1.1 Visión global

Siguiendo la tendencia actual de las aplicaciones web, vamos a desacoplar el *frontend* y el *backend*. Los lenguajes utilizados en ambos son independientes, únicamente tienen que respetarse los contratos definidos. Es decir, vamos a seguir una arquitectura RESTful.

El *backend* API se limitará a gestionar los datos y la lógica de negocio. Para ello se expondrá una REST API con determinados *endpoints*. El *back* recibirá peticiones del *front* a través de los diferentes *endpoints* y el servidor devolverá una respuesta al *front* en formato json.

El *frontend* se encargará de construir la interfaz de usuario, interactuar con el usuario y realizar peticiones al *back* a través de los endpoints definidos, es decir, consumirá datos del *back*. Se ejecuta en el cliente.

Esta arquitectura permite desarrollar aplicaciones web robustas.

4.1.2 Entorno de Desarrollo

Para el desarrollo en local del *backend* se va a utilizar Docker, que permite construir, desplegar y ejecutar aplicaciones con contenedores de forma ágil.

Estos son los contenedores que definiremos para este proyecto:

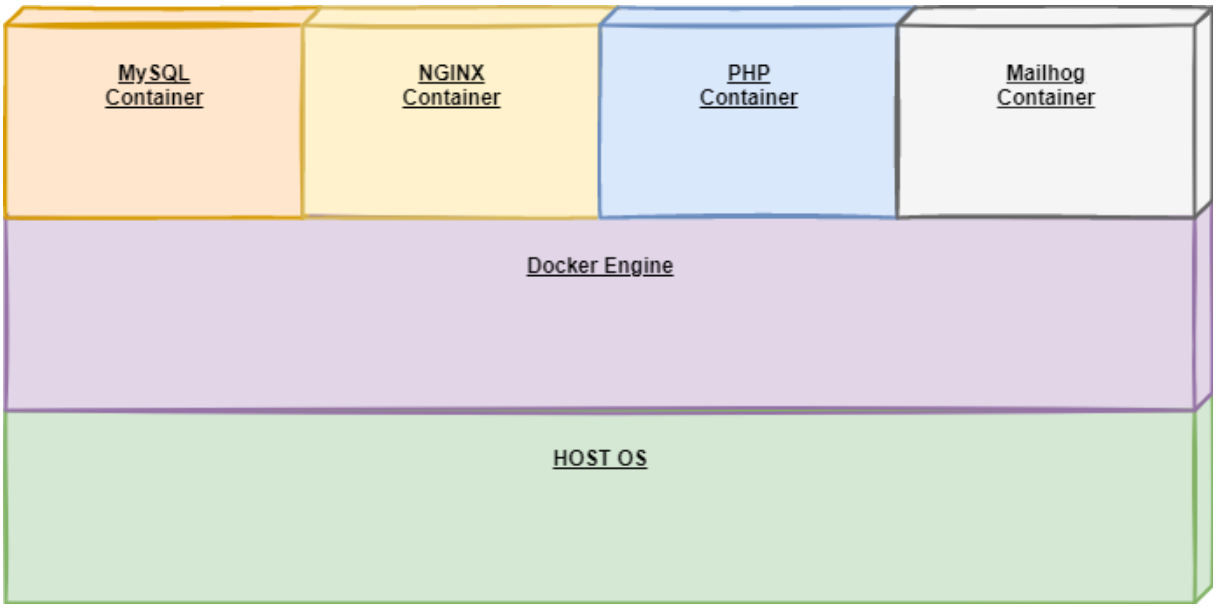


Ilustración 23: Arquitectura desarrollo

En el archivo docker-compose.yml hemos definido los servicios y los contenedores que tendrá nuestra aplicación, que serán los que se levantaran cuando ejecutemos `docker-compose up -d`.

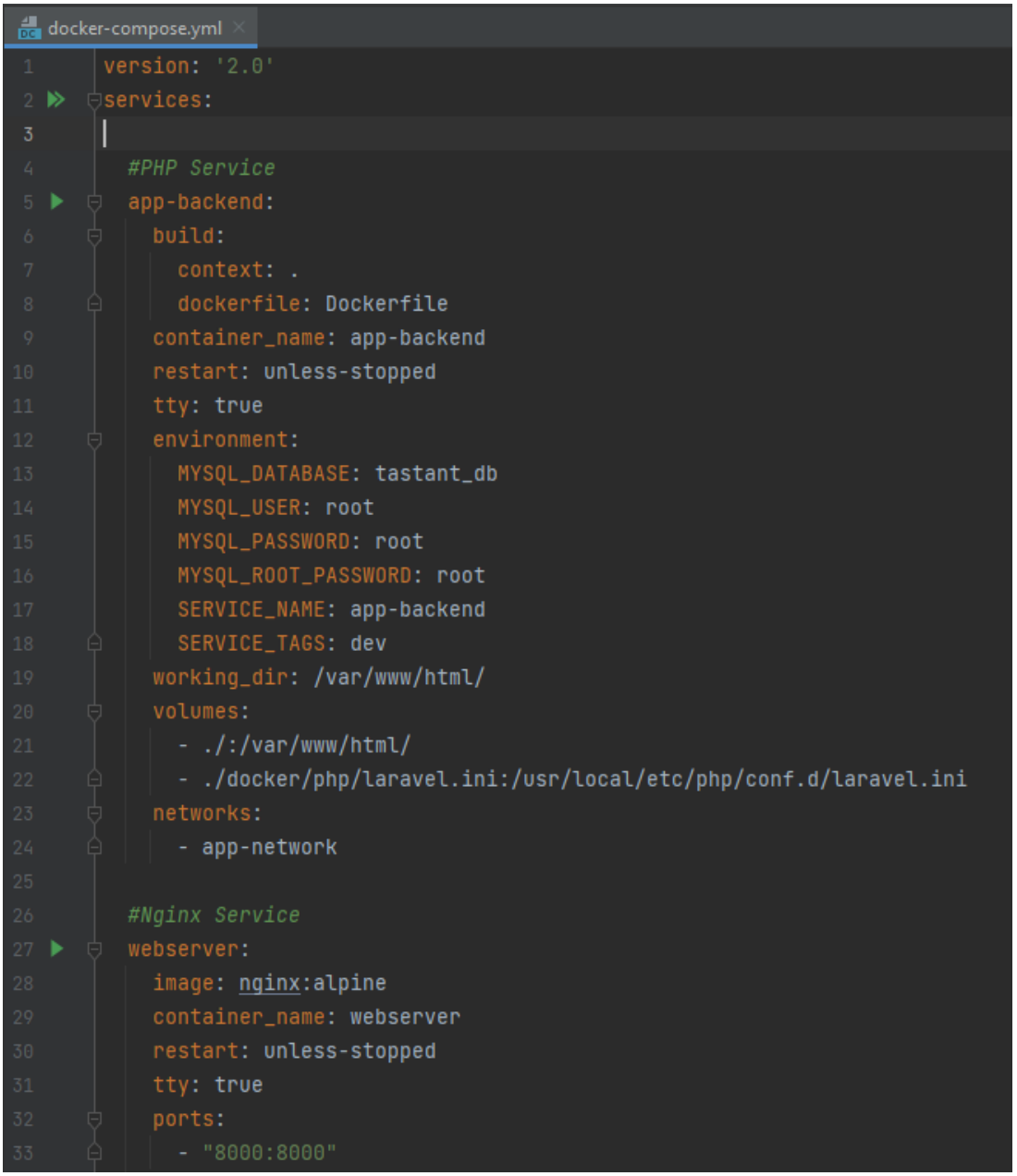


Ilustración 24: Definición docker-compose.yml

Para definir el servicio de PHP se ha utilizado Dockerfile y para el resto de servicios imágenes ya definidas.

En directorio /docker del proyecto backend están definidas las configuraciones específicas para cada uno de los contenedores.

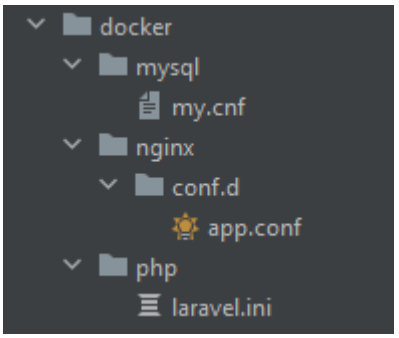


Ilustración 25: Directorios de configuración de los contenedores de Docker

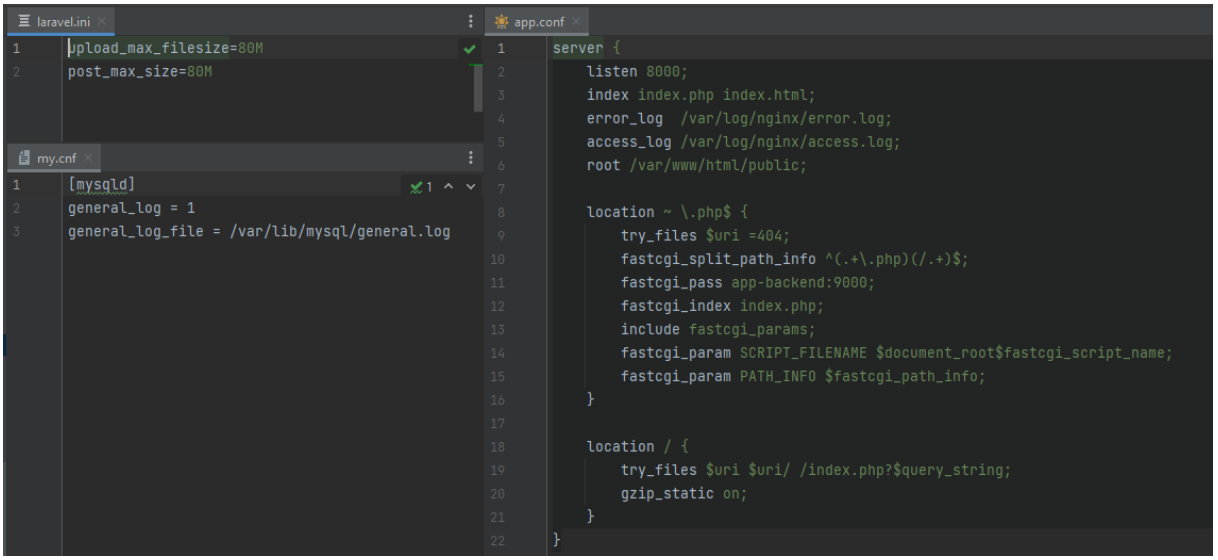


Ilustración 26: Configuraciones específicas de los contenedores de Docker

Si ejecutamos el comando `docker ps` en el directorio en el que tenemos instalado el proyecto del backend vemos los contenedores que se levantan.

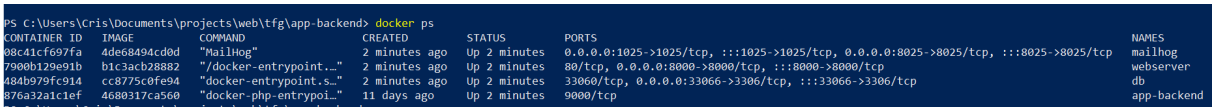


Ilustración 27: Si ejecutamos el comando docker ps podemos ver los contenedores levantados

Con respecto al *frontend*, solo vamos a necesitar tener node.js y npm instalado en nuestro equipo. Con esto ya podremos levantar y probar el frontal.

4.1.3 Entorno de Producción

La aplicación va a estar integrada con dos servicios externos: AWS S3 y Sendgrid.

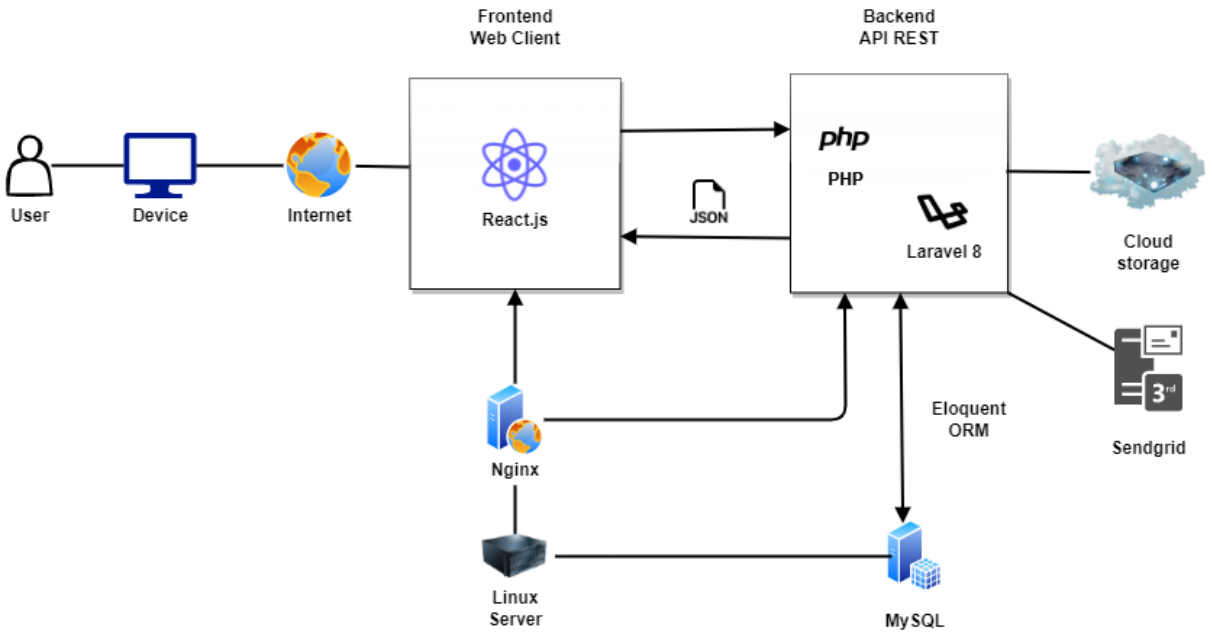


Ilustración 28: Arquitectura producción

4.2 Interfaz HTTP

Se han definido los siguientes *endpoints*:

Método	URI	Descripción
GET	api/accept-invitation/{token}	Aceptar la invitación a un grupo
GET	api/admin/categories	Listar las categorías
POST	api/admin/category	Crear una categoría
GET	api/admin/category/{category}	Recuperar una categoría
PUT	api/admin/category/{category}	Editar una categoría
DELETE	api/admin/category/{category}	Eliminar una categoría
GET	api/admin/dashboard	Recuperar los datos estadísticos de la aplicación usados en el dashboard del backoffice
GET	api/admin/group/{group}	Recuperar un grupo

PUT	api/admin/group/{group}	Editar un grupo
GET	api/admin/groups	Listar los grupos
DELETE	api/admin/groups/{group}	Eliminar un grupo
DELETE	api/admin/recommendation/{recommendation}	Eliminar una recomendación
GET	api/admin/recommendations	Listar las recomendaciones
POST	api/admin/user	Crear un nuevo usuario
GET	api/admin/user/{user}	Recuperar un usuario
PUT	api/admin/user/{user}	Editar un usuario
DELETE	api/admin/user/{user}	Eliminar un usuario
GET	api/admin/users	Listar los usuarios
GET	api/categories	Listar las categorías
POST	api/forgot-password	Recuperar la contraseña
POST	api/group	Crear un nuevo grupo
GET	api/group/{group}	Recuperar un grupo
PUT	api/group/{group}	Editar un grupo
GET	api/group/{group}/admin	Mostrar la información de un grupo
POST	api/group/{group}/invite-member	Invitar a un usuario al grupo
POST	api/group/{group}/leave	Abandonar un grupo
POST	api/group/{group}/make-administrator/{user}	Hacer administrador de grupo a un usuario
POST	api/group/{group}/new-recommendation	Crear una recomendación
GET	api/group/{group}/recommendation/{recommendation}	Recuperar una recomendación
POST	api/group/{group}/recommendation/{recommendation}/rate	Valorar una recomendación
POST	api/group/{group}/recommendation/{recommendation}/save	Guardar una recomendación
GET	api/group/{group}/recommendations	Listar las recomendaciones de un grupo
DELETE	api/group/{group}/remove-user/{user}	Eliminar un usuario de un grupo

POST	api/login	Autenticarse en la aplicación
POST	api/logout	Cerrar la sesión de usuario
GET	api/me	Recuperar la información del usuario autenticado
POST	api/register	Registrarse en la aplicación
GET	api/reset-password	Resetear la contraseña
DELETE	api/user/delete-my-account	Eliminar la cuenta de usuario
PUT	api/user/{user}	Editar mi usuario
POST	api/user/{user}/update-image	Editar mi imagen de usuario

4.2.1 Cross-Origin Resource Sharing (CORS)

CORS define una forma segura y aceptada por navegador de solicitar recursos a otro dominio o puerto.

En nuestro caso, vamos a solicitar recursos desde el frontal a un endpoint del mismo dominio, pero alojado en otro puerto.

Con Laravel podemos fácilmente definir la manera de responder a las cabeceras HTTP CORS en el archivo config/cors.php.

Laravel va un paso más allá y, además, ofrece protección CSRF para las aplicaciones. Este mecanismo requiere que en el proceso de autenticación se realice una llamada previa a /sanctum/csrf-cookie para inicializar CSRF. Con esta llamada Laravel creará una cookie llamada XSRF-TOKEN que contiene el token actual CSRF. Este token se pasará en el encabezado de las siguientes llamadas.

Disponemos de esta protección para todas las llamadas a la API.

4.2.2 Ejemplo de una llamada desde el front a la API

A continuación vamos a ver varios ejemplos de llamadas desde el *front* a la API.

Esta es la llamada que realizamos cuando un usuario se quiere autenticar. En primer lugar, hacemos la llamada CRSF previa necesaria.

Con esta llamada, se va a generar una cookie con el nombre XSRF-TOKEN, tal y como se puede ver en la captura:

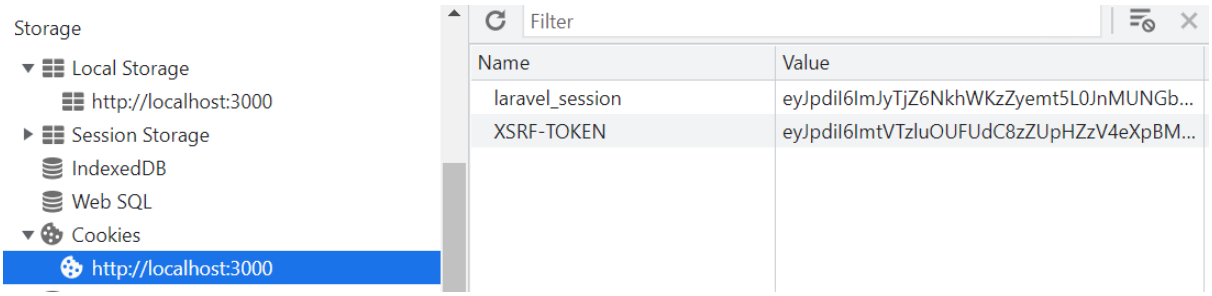


Ilustración 29: Developer tools > application > storage > cookies: XSRF-TOKEN

Una vez que hemos obtenido este token, ya podemos realizar la llamada que gestionará el login a nuestra aplicación.

Ilustración 30: React servidor web puerto 3000

Si la autenticación es satisfactoria en las siguientes peticiones a la API se añadirá automáticamente el token CSRF. El usuario está autenticado vía este token y, además, por el token de autorización.

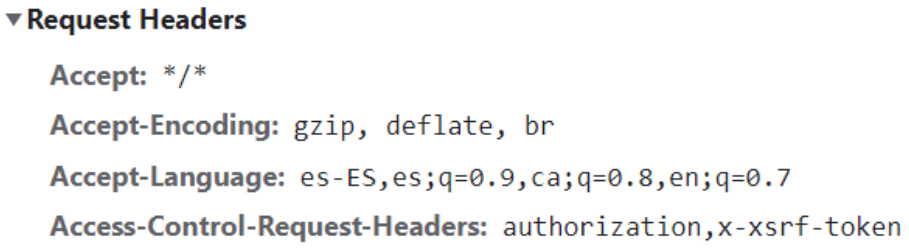


Ilustración 31: Request header XSRF-TOKEN

El frontal se guarda el token de autorización y lo añadirá en el encabezado de cada llamada que haga a la API:

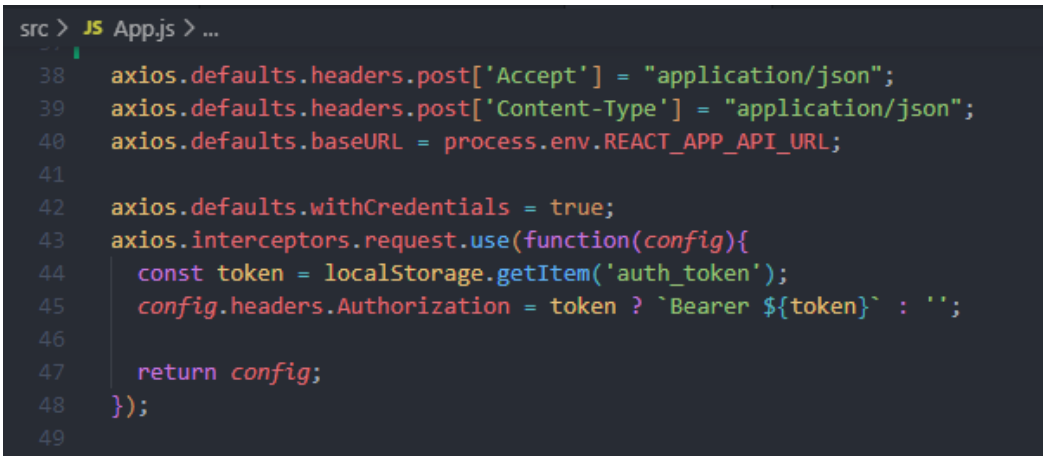


Ilustración 32: React configuración Axios

4.2 Decisiones de implementación

Estas son algunas de las principales decisiones que se han tomado a la hora de codificar:

- Para evitar acceso a recursos a los que un usuario no está autorizado se han utilizado Middlewares.
- Para implementar las funcionalidades se ha intentado aprovechar toda la potencia que ofrece Laravel y características ya construidas, como:
    - Crear modelos, migraciones, seeders y controladores para una entidad: `php artisan make:model Group -m -s -c`
    - Crear migraciones para las tablas intermedias: `php artisan make:migration create_relation_user_groups_table`

    - Crear reglas de validación: `php artisan make:rule IsUserAlreadyMemberOfGroup`
    - Crear notificaciones: `php artisan make:mail InvitationToJoinAGroup`

- Laravel utiliza por defecto el ORM Eloquent. Este ORM hace realmente sencillo interactuar con la DB y los modelos de negocio.
- Para la autenticación usaremos el componente nativo de Laravel, Laravel Sanctum.

4.3 Levantar la aplicación en local

Para levantar la aplicación necesitaremos levantar tanto el frontal como el backend. En los siguientes apartados se indicarán las instrucciones a seguir.

4.3.1 Frontend

El frontal está desarrollado con React.

Para poder ejecutar la aplicación es necesario:

- Node.js (https://nodejs.org/en/)
- El gestor de paquetes npm (https://www.npmjs.com/)
- Git (https://git-scm.com/)

Para levantar el frontal hay que seguir los siguientes pasos:

- `git clone https://github.com/ayoira/tfg-app-frontend.git` o descomprimir el zip tfg-app-frontend-main.zip
- `cd tfg-app-frontend`
- `npm install`
- `npm start`

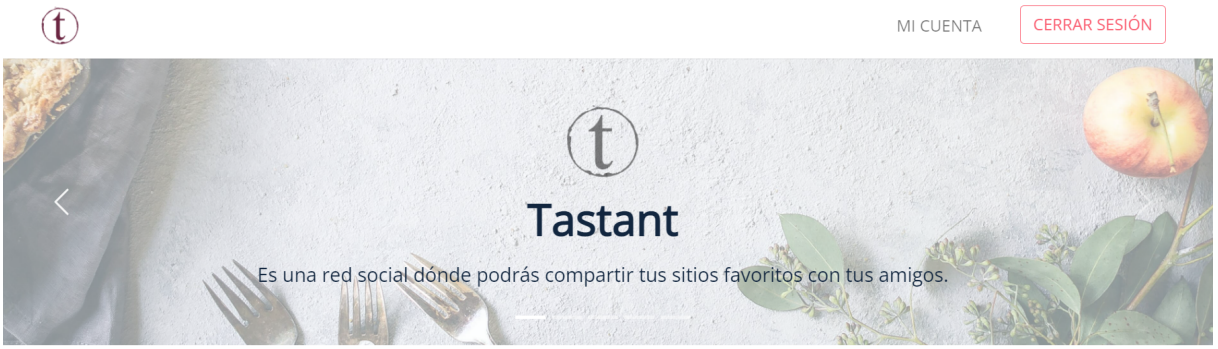
Si las imágenes que vienen del backend no se muestran ejecutar en PowerShell: `($env:REACT_APP_API_URL = "http://localhost:8000") -and (npm start)`

Se espera que la aplicación sea levantada en el puerto 3000. Si se levanta en otro puerto será necesario modificar la variable de entorno FRONTEND_URL de .env del **backend** indicando la url correcta del front.

```
FRONTEND_URL=http://localhost:3000
```

Ilustración 33: React servidor web puerto 3000

La aplicación se levantada en http://localhost:3000/.



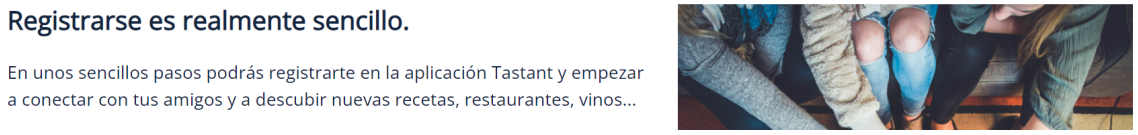


Ilustración 34: React servidor web puerto 3000

No se requiere modificar el archivo de entorno .env.

La rama activa es main.

4.3.2 Backend

El backend está desarrollado con el Framework Laravel.

Para poder ejecutar la aplicación es necesario:

- Docker (https://www.docker.com/)
- Docker compose (https://docs.docker.com/compose/)

Para levantar el backend hay que seguir los siguientes pasos:

- `git clone https://github.com/ayoira/tfg-app-backend.git` o descomprimir el zip tfg-app-backend-main.zip
- `cd tfg-app-backend`
- `docker-compose up -d`
- `docker exec -it app-backend bash`
- `composer install`
- `php artisan migrate`
- `php artisan db:seed`
- `php artisan storage:link`

Si es necesario refrescar las migraciones y seeders se puede utilizar: `php artisan migrate:fresh --seed`.

La aplicación está levantada en http://localhost:8000/.

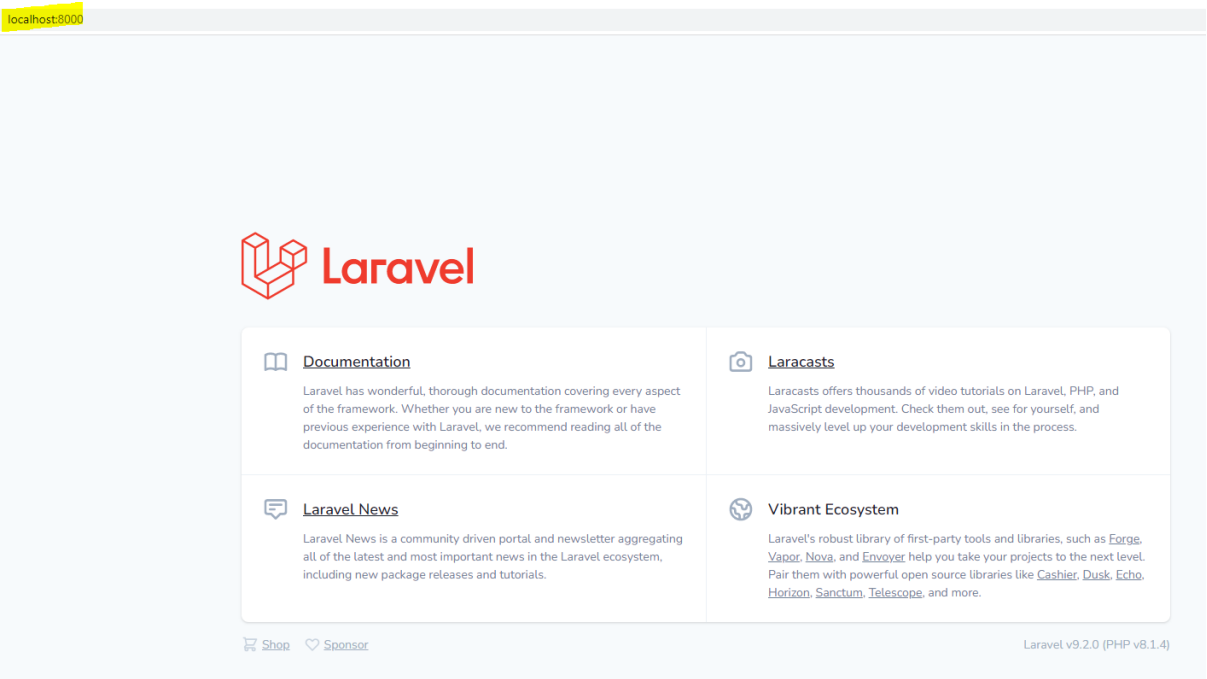


Ilustración 35: Laravel servidor web puerto 8000

La Base de datos escucha el puerto 33066.

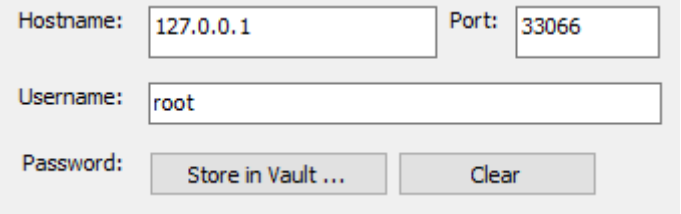


Ilustración 36: Configuración conexión Base de Datos

Las credenciales para acceder a la Base de datos son:

- Hostname:127.0.0.1
- Port:33066
- User: root
- Password: rooot
- DB: tastant_db

Para acceder al contenedor de Mailhog http://localhost:8025/#.

Se espera que la aplicación sea levantada en el puerto 8000. Si se levanta en otro puerto será necesario modificar la variable de entorno REACT_APP_API_URL de .env del **frontal** indicando la url correcta del backend.

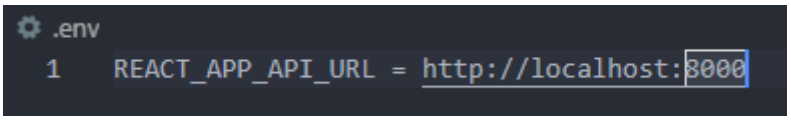


Ilustración 37: Laravel servidor web puerto 8000

No se requiere modificar el archivo de entorno .env.

La rama activa es main.

4.4 Testing

4.4.1 Tests de integración con Postman

Para todas las acciones que requieran autenticación necesitamos un token válido. Este token identifica de manera unívoca al usuario autenticado.

Este token podemos conseguirlo, bien registrándose o bien entrando en la aplicación.

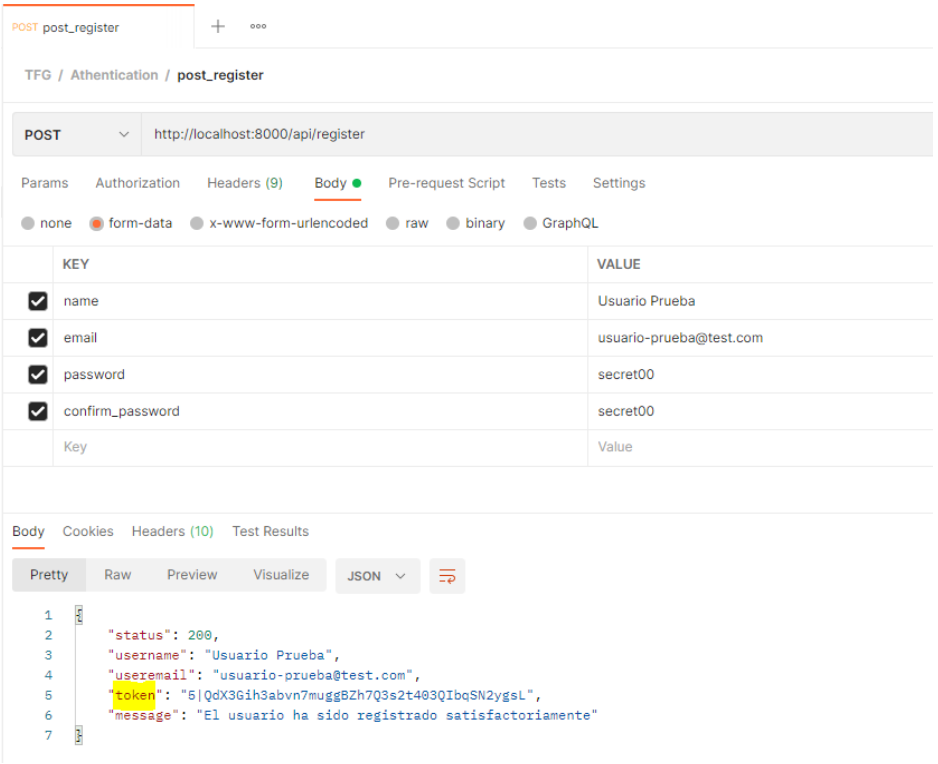


Ilustración 38: Endpoint register

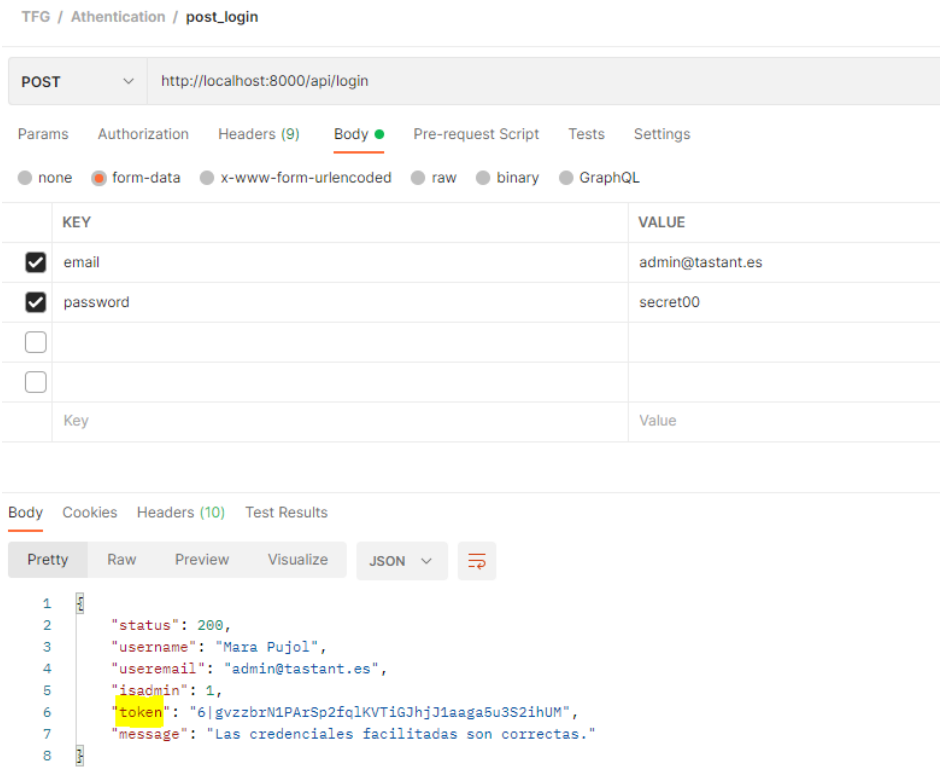


Ilustración 39: Endpoint login

Cuando vayamos a realizar una petición a un endpoint que requiera autenticación, usaremos el token obtenido en el paso anterior. Iremos a la pestaña “Authorization”, en Type seleccionaremos “Bearer Token” y en Token pegaremos el token obtenido en el paso anterior.

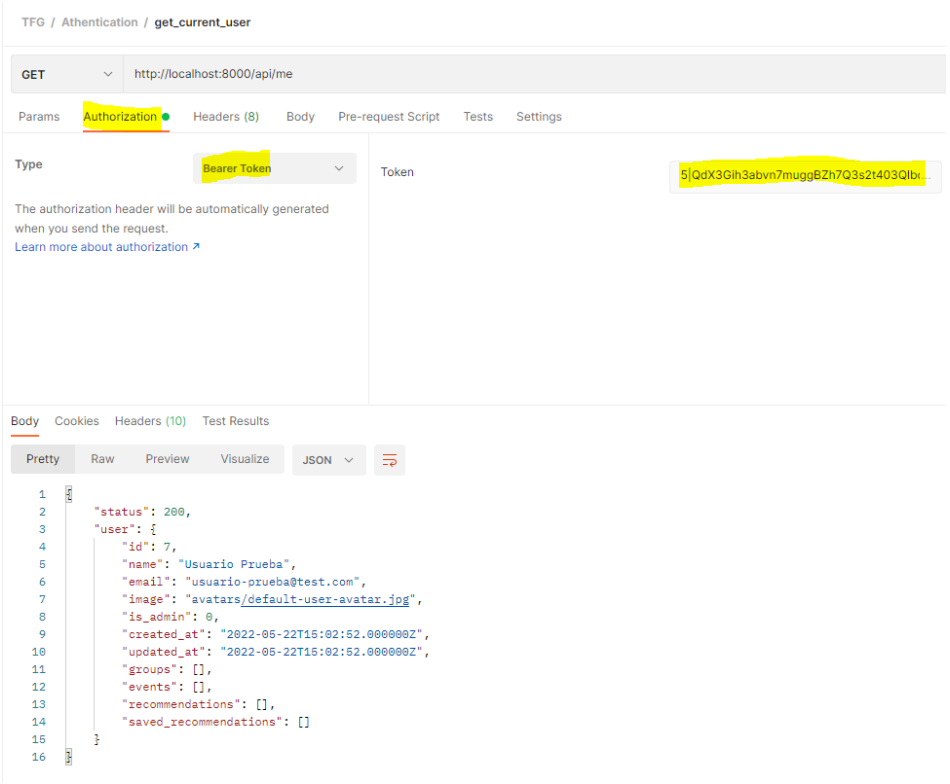


Ilustración 40: Consulta que requiere autenticación

Si usáramos un token no válido o caducado la petición nos devolverá 401.

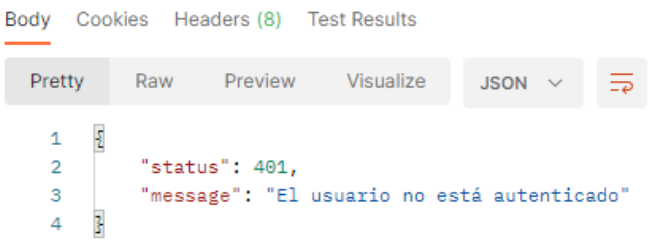


Ilustración 41: Token no válido

Se adjunta un JSON (TFG.postman_collection) con las consultas a los endpoints más relevantes de la aplicación.

4.4.2 Testing email

Para testear el envío de emails utilizamos un contenedor de Mailhog, que está levantado en el puerto 8025.

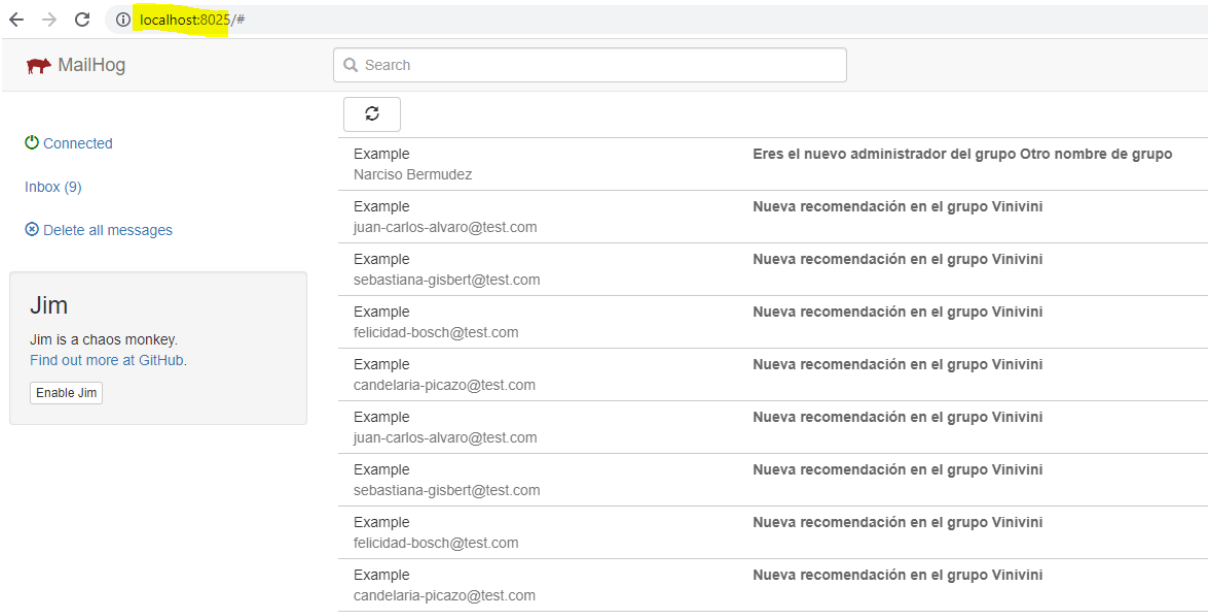


Ilustración 42: Contenedor de Mailhog

4.5 Usuarios de prueba

Al ejecutar las migraciones y los seeders se puebla la Base de Datos con datos de prueba. Hay seis usuarios de prueba, totalmente ficticios. Sus nombres han sido generados con la aplicación https://generadordenombres.online/.

ID	Email	Contraseña	Es administrador
1	admin@tastant.es	secret00	Sí
2	candelaria-picazo@test.com	secret00	No
3	sebastiana-gisbert@test.com	secret00	No
4	juan-carlos-alvaro@test.com	secret00	No
5	felicidad-bosch@test.com	secret00	No
6	narciso-bermudez@test.com	secret00	No

4.6 Gitflow

Gitflow establece unas directrices de qué ramas tenemos que crear y usar y cómo fusionarlas.

Las ramas principales, que registran el historial de cambios del proyecto:

- *Develop*: Es la rama de la cuál partirán las ramas creadas para la implementación de funcionalidades.
- *Main*: Guarda el historial de versiones oficiales de despliegues.

Las ramas secundarias:

- *Feature*: Cada nueva característica tiene que tener su propia rama que partirá de develop. Una vez que la funcionalidad ha sido completada se tiene que fusionar con develop.
- *Release*: Rama creada a partir de *develop* con el objetivo de desplegar el nuevo código en producción. Esto se llevará a cabo cuando haya bastante funcionalidad sin desplegar en desarrollo. Esta rama release se tiene que fusionar con main con una etiqueta de número de versión.
- *Hotfix*: Rama creada a partir de *main* para solucionar errores de forma rápida de la aplicación en producción.

En ambos subproyectos la rama activa es main.

5. Recorrido por la aplicación

5.1. Home

http://localhost:3000/

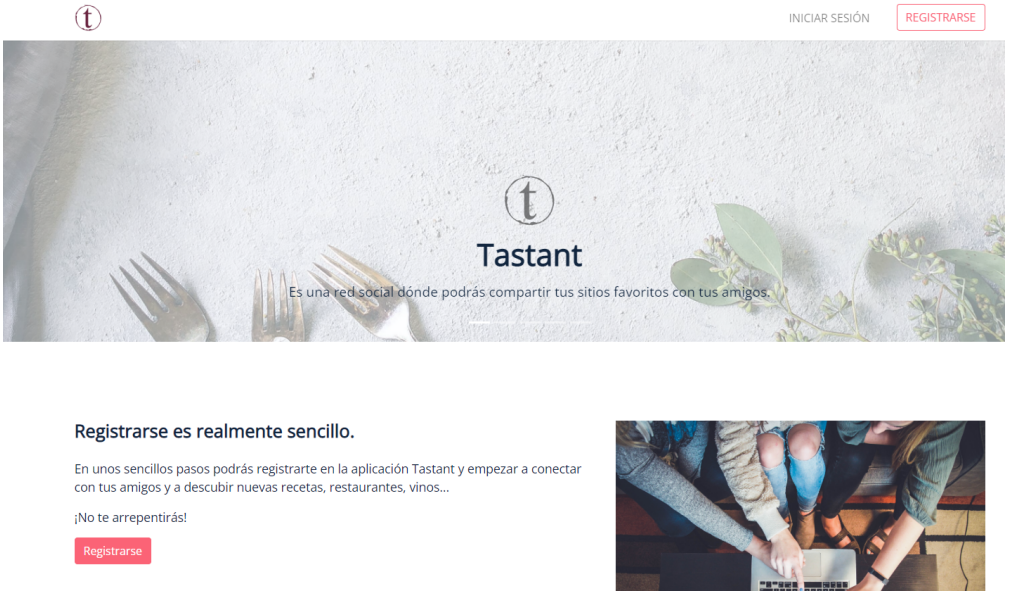


Ilustración 43: Home

5.2. Autenticación

http://localhost:3000/login
http://localhost:3000/forgot-password
http://localhost:3000/register

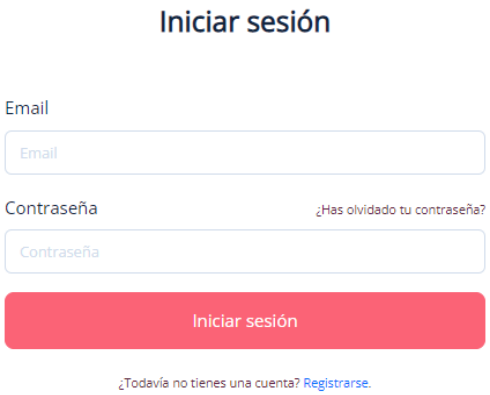


Ilustración 44: Login

5.3. Cuenta de usuario

http://localhost:3000/mi-cuenta

En este espacio los usuarios podrán modificar su información personal, consultar los grupos a los que pertenecen, crear un grupo nuevo, consultar las recomendaciones que han creado y aquellas que se han guardado. También podrán eliminar su cuenta de la aplicación.

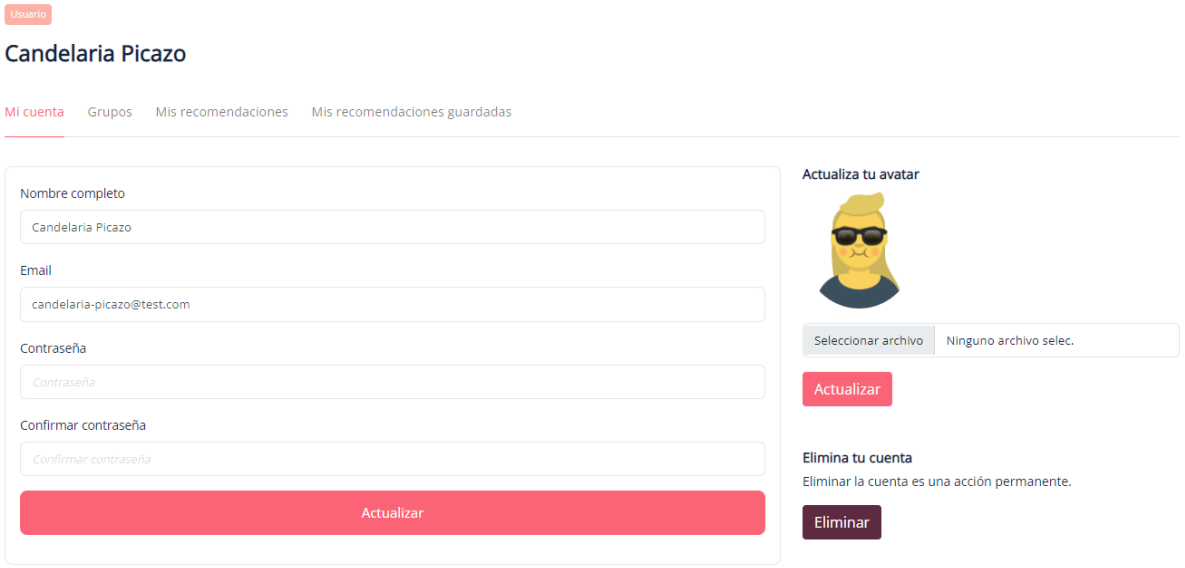


Ilustración 45: Página de usuario

5.4. Página de grupo

http://localhost:3000/grupo/1

En este espacio los usuarios podrán consultar la información del grupo (miembros, recomendaciones, nombre y descripción) e invitar nuevos miembros al grupo. Además los usuarios administradores del grupo podrán editar el grupo.

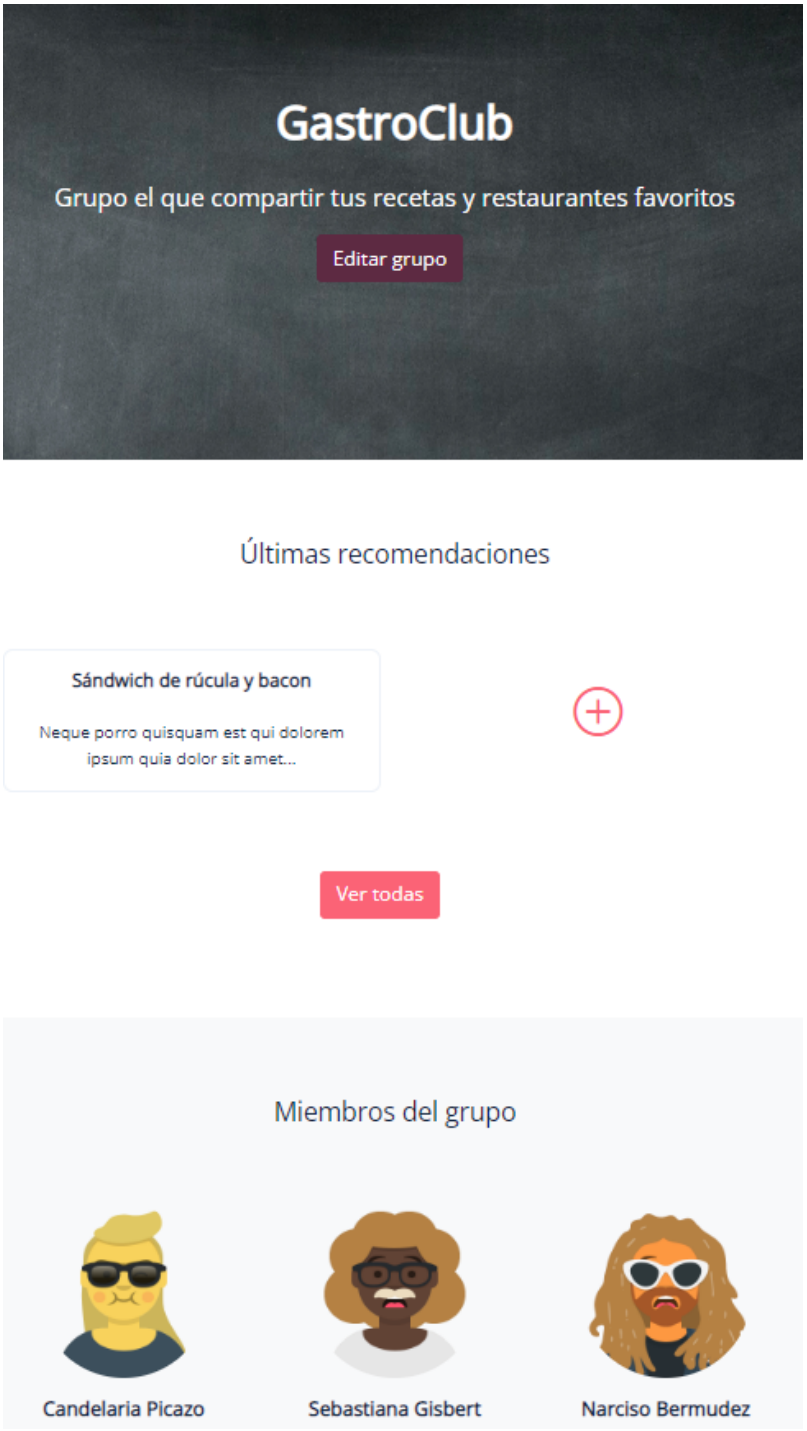


Ilustración 46: Página de grupo

5.5. Página de recomendación

http://localhost:3000/grupo/1/recomendacion/1

En esta página los usuarios podrán ver la recomendación, guardarla y puntuarla.

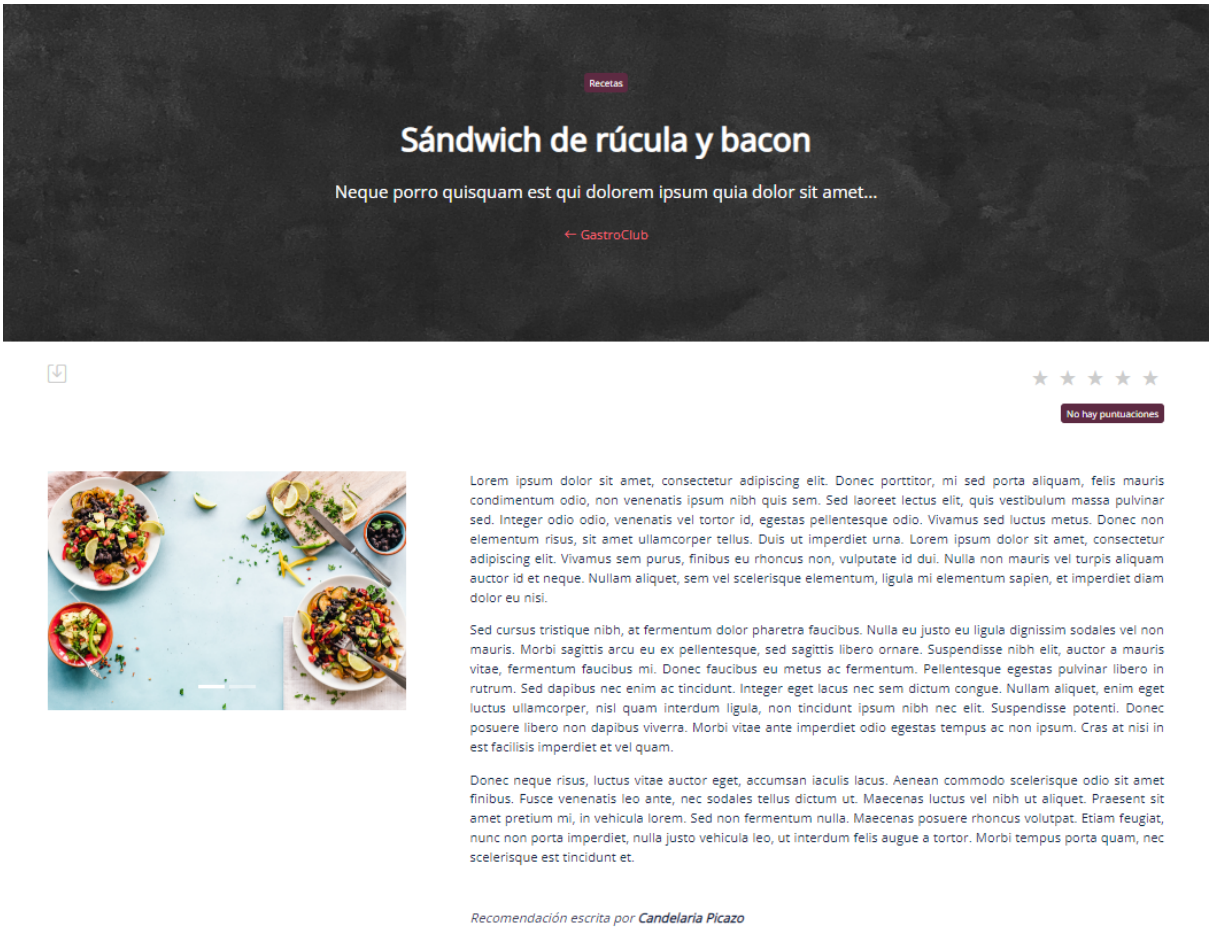


Ilustración 47: Página de recomendación

5.6. Backoffice

http://localhost:3000/admin/dashboard

A este espacio solo pueden acceder los administradores de la plataforma. Los administradores podrán editar la mayoría de las entidades del sistema.

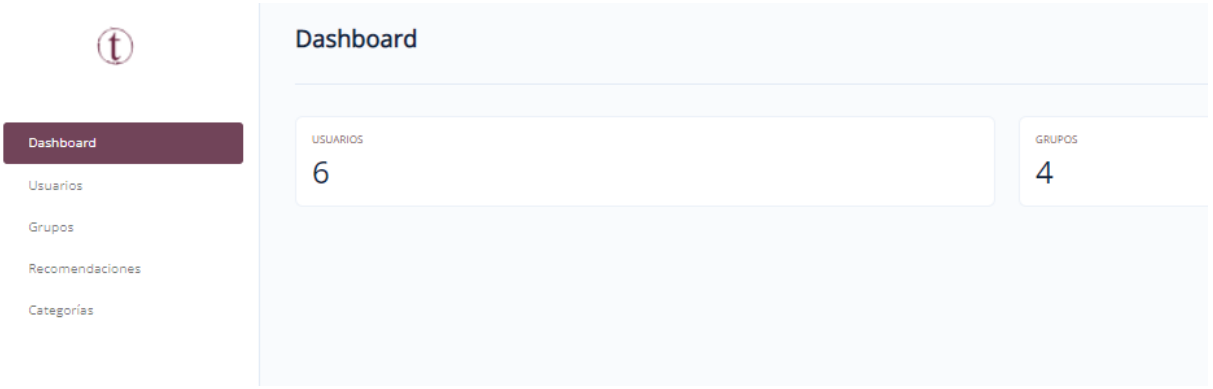


Ilustración 48: Backoffice

6. Conclusiones

Se ha conseguido un producto que podría salir al mercado cumpliendo con los principales requisitos. Sin embargo, no se ha podido implementar todas las funcionalidades definidas por los *stakeholders*.

Debido a la extensión de los requisitos de la aplicación no se han implementado todos los requisitos, quedando pendientes para futuras iteraciones:

- Módulo eventos.
- Sistema de comentarios de las recomendaciones de un grupo.
- Filtro, buscador y paginación de las recomendaciones de un grupo.
- Sistema de almacenamiento en la nube para las imágenes (S3).
- Los autores de una recomendación puedan editar/eliminar una recomendación.
- Subir la aplicación a un entorno de producción.
- Testing unitario.
- Testing automatizado de integración.
- Testing de aceptación.

Por otro lado, uno de los mayores riesgos que se han asumido en este proyecto es la falta de conocimiento de las tecnologías usadas. Debido a la falta de conocimiento, las tareas a realizar han requerido muchos más recursos de los estimados inicialmente, teniendo como consecuencia la no consecución de todos los objetivos marcados inicialmente.

Sin embargo, el haber asumido este riesgo ha supuesto la adquisición de nuevos conocimientos, que actualmente están muy demandados en el mercado laboral actual.

Además, en la planificación el testing unitario se planteó como una actividad transversal, pero no se ha llevado a cabo de tal manera. Después de la codificación de cada caso de uso, se debería haber realizado el correspondiente testing unitario. Es decir, que el testing forme parte importante de la codificación y no una fase aislada. Quizás esto se podría haber evitado siguiendo metodologías TDD.

Con respecto a posibles mejoras que añadiría a la aplicación:

- Uso de UUID, en lugar de identificadores incrementales.
- Uso de arquitecturas limpias y seguir los principios SOLID que hacen que el código esté menos acoplado y sea más mantenible y escalable.

7. Glosario

API: Es una interfaz de programación de aplicaciones es un conjunto de definiciones y protocolos que sirven para ofrecer servicios a otros.

API RESTful: Es una API que cumple con las restricciones de las arquitecturas REST.

Bootstrap: Framework para el diseño y maquetación de aplicaciones web.

CORS: Es un mecanismo de seguridad para poder solicitar recursos a una aplicación desde un dominio o puerto diferente.

CSS: Lenguaje de marcas usado para maquetar aplicaciones web.

CSRF: Es un tipo de vulnerabilidad de seguridad que debido a la confianza en otro extremo este transmite comandos no autorizados por el primero.

Docker: Software que automatiza el despliegue de aplicaciones dentro de contenedores de software.

Eloquent: ORM usado por Laravel.

Framework: Es un marco de trabajo que sirve como referencia.

Gitflow: Flujo de trabajo de git.

HTML: Lenguaje de marcas para el desarrollo de páginas web.

JavaScript: Lenguaje de programación que se ejecuta en el cliente (navegador).

Laravel: Es un framework de código abierto para desarrollar aplicaciones y servicios web con PHP.

Middleware: Es una capa de software que ofrece servicios a las aplicaciones.

MVC: Patrón de diseño de software.

MySQL: Sistema de gestión de Bases de Datos relacional.

MySQL Workbench: Cliente de MySQL.

Node.js: Entorno de ejecución para la capa del servidor de código abierto.

NPM: Gestor de paquetes de node.js.

ORM: Es una técnica para convertir los datos entre el sistema negocio (objetos) y la base de datos.

PHP: Lenguaje de programación usado principalmente para el desarrollo de aplicaciones web.

PHPStorm: Entorno de desarrollo para PHP.

React.js: Es una framework de JavaScript de código abierto para crear interfaces de usuario que sigan el patrón SPA

Responsive: Diseño web que se adapta al dispositivo.

Single Page Application (SPA): Aplicación web de una sola página.

Token: Cadena de caracteres que tiene un significado coherente en cierto contexto o lenguaje de programación.

8. Bibliografía

[1] «Trello» 20 02 2022. [En línia]. Available: https://trello.com/.

[2] «Laravel» 01 03 2022. [En línia]. Available: https://laravel.com/.

[3] «React.js» 01 03 2022. [En línia]. Available: https://reactjs.org/.

[4] «Bootstrap» 29 02 2020. [En línia]. Available: https://getbootstrap.com/.

[5] «Docker» 01 03 2022. [En línia]. Available: https://www.docker.com/.

[6] «PHP» 01 03 2022. [En línia]. Available: https://www.php.net/.

[7] «Nginx» 01 03 2022. [En línia]. Available: https://www.nginx.com/.

[8] «MySql» 01 03 2022. [En línia]. Available: https://www.mysql.com/.

[9] «HTML» 01 03 2022. [En línia]. Available: https://www.w3schools.com/html/.

[10] «CSS» 01 03 2022. [En línia]. Available: https://www.w3schools.com/css/default.asp.

[11] «JavaScript» 01 03 2022. [En línia]. Available:https://www.w3schools.com/js/default.asp.

[12] «SQL» 01 03 2022. [En línia]. Available: https://www.w3schools.com/sql/default.asp.

[13] «Node.js» 01 03 2022. [En línia]. Available: https://nodejs.org/en/.

[14] «NPM» 01 03 2022. [En línia]. Available: https://www.npmjs.com/.

[15] «PHPStorm» 01 03 2022. [En línia]. Available: https://www.jetbrains.com/phpstorm/.

[16] «Visual Studio Code» 01 03 2022. [En línia]. Available: https://code.visualstudio.com/.

[17] «Stackoverflow» 01 03 2022. [En línia]. Available: https://stackoverflow.com/.

[18] «Laracasts» 01 03 2022. [En línia]. Available: https://laracasts.com/discuss/.

[19] «Postman» 01 03 2022. [En línia]. Available: https://www.postman.com/.

[20] «Github» 01 03 2022. [En línia]. Available: https://github.com/.

[21] «Git» 01 03 2022. [En línia]. Available: https://git-scm.com/.

[22] «Instagantt» 01 03 2022. [En línia]. Available: https://instagantt.com/.

[23] «Mockflow» 01 03 2022. [En línia]. Available: https://mockflow.com/.

[24] «Coolors» 01 03 2022. [En línia]. Available: https://coolors.co/.

[25] «Unsplash» 01 03 2022. [En línia]. Available: https://unsplash.com/es.

[26] «TyneMCE» 01 03 2022. [En línia]. Available: https://www.tiny.cloud/.

[27] «Diagrams» 01 03 2022. [En línia]. Available: https://app.diagrams.net/.

[27] «Google Fonts» 01 03 2022. [En línia]. Available: https://fonts.google.com/specimen/Open+Sans?query=open+sans.

[28] «Shutterstock» 01 03 2022. [En línia]. Available: https://www.shutterstock.com/.

[29] «GetAvataars» 01 03 2022. [En línia]. Available: https://getavataaars.com/.

[30] «Generador de nombres online» 01 03 2022. [En línia]. Available: https://generadordenombres.online/.

[31] «Oreilly» 01 03 2022. [En línia]. Available: https://www.oreilly.com/.

[32] «Wikipedia» 01 03 2022. [En línia]. Available: https://www.wikipedia.org/.

[33] «Drawio» 01 03 2022. [En línia]. Available: https://drawio-app.com/.