Documentación Proyecto TFGS : Foodsynk

Introducción

A la hora de hacer la compra semanal, es muy común olvidar algún ingrediente o calcular mal las cantidades necesarias, especialmente cuando no se ha planificado con claridad qué recetas se van a preparar o cuántas comidas se realizarán. Esto provoca que muchas veces haya que volver al supermercado o improvisar platos con lo que se tiene en casa, lo que genera pérdida de tiempo y desorganización.

El proyecto “Gestor web de menús semanales, recetas y lista de la compra” surge precisamente para resolver ese problema. Su objetivo es facilitar la planificación de las comidas de toda la semana y automatizar la generación de la lista de la compra, teniendo en cuenta las recetas elegidas, los ingredientes necesarios y el número de personas que van a comer.

De esta manera, el usuario podrá organizar sus menús de forma práctica y visual, evitando olvidos y compras innecesarias. Además, la aplicación pretende fomentar una alimentación más variada y equilibrada, reduciendo el tiempo dedicado a pensar cada día qué cocinar y optimizando el proceso de compra.

Título:
Gestor web de menús semanales,recetas y lista de la compra.

Descripción:

Este proyecto tiene como idea principal crear una aplicación web que ayude a organizar las comidas de una forma más sencilla y práctica. La aplicación estaría pensada para personas que, por ejemplo, no quieren complicarse todos los días pensando qué hacer de comer o de cenar. Un caso típico podría ser una madre que necesita planear el menú semanal para su familia, con varias comidas al día y para varias personas.

La idea es que el usuario pueda crear menús personalizados por días y por comidas (desayuno, almuerzo, cena) y adaptarlos al número de personas. Para eso tendría un recetario consultable, donde se pueden buscar recetas, filtrarlas por categorías y leer comentarios de otros usuarios. Además, cada usuario registrado podría guardar sus recetas favoritas, subir las suyas propias, editarlas o eliminarlas, así como comentar en las de otros.

La parte más característica del proyecto es la lista de la compra automática. Una vez hecho el menú, la aplicación calcularía qué ingredientes se necesitan según el número de platos, de días y de personas, y generaría la lista de la compra. Esta lista se podría editar a mano (por ejemplo, si ya se tiene un ingrediente en casa), y después el usuario podría descargarla o imprimirla para usarla directamente al hacer la compra.

En cuanto a los usuarios, habría tres tipos:

Usuario no registrado, que solo podría ver el recetario, filtrar recetas y leer comentarios.

Usuario registrado, que tendría acceso a todas las funciones: guardar recetas, crearlas, comentarlas, organizar menús semanales y editar o descargar su lista de la compra.

Administrador, que además de todo lo anterior tendría permisos para eliminar recetas, comentarios o usuarios que incumplan las normas, por ejemplo en caso de publicaciones ofensivas o dañinas.

El proyecto también está pensado para ser escalable, de forma que en un futuro se le puedan añadir más funciones, como el cálculo de los valores nutricionales de cada plato o un sistema de recomendaciones según los gustos del usuario.

Objetivo:

El objetivo de la aplicación es crear una aplicación útil y práctica que ayude a las personas a organizar su alimentación de forma rápida y sencilla, evitando la carga de tener que pensar menús todos los días y facilitando la compra con listas automáticas que ahorran tiempo y esfuerzo.
A nivel personal, mi objetivo con este proyecto es desarrollar una aplicación web completa y funcional que refleje lo aprendido durante el curso y que me sirva como parte de mi portafolio profesional, mostrando mis capacidades de diseño, desarrollo y organización de

un proyecto real.

Enlaces de referencia:

1. Plan to Eat: Permite importar recetas, organizarlas, planificar las comidas y generar automáticamente la lista de la compra según lo que hayas puesto en tu calendario. Es de pago, pero es una aplicación muy interesante por la interfaz simple que muestra.
2. Mealime: Es una app que simplifica la planificación de comidas saludables, con recetas de rápido preparo y listas de compra vinculadas.
   Esta app me ha gustado por los filtros (dieta, tiempo…) y el diseño UX.
3. Eat This Much: Genera planes de comidas automáticos según preferencias, presupuesto, dieta, etc., y crea la lista de compras. Es más “avanzada”: ya empieza a mezclar nutrición con planificación.

Hay muchas más aplicaciones por el estilo, pero al final es un simple planteamiento de que funcionalidades me gustaría que tuviera e ideas para la interfaz.

Viabilidad:

Es un proyecto totalmente viable porque puede desarrollarse con tecnologías web básicas como HTML, CSS, JavaScript, PHP y MySQL. No requiere recursos complicados, ya que se puede empezar con una base de datos con recetas de ejemplo e ir ampliándola con la participación de los usuarios. Además, al estar planteado de forma modular, permite empezar con lo básico (recetas + menús + lista de la compra) e ir mejorándolo poco a poco sin rehacer todo el sistema.

Stack tecnológico e infraestructura:

- Frontend:
    - **React** se utilizará para desarrollar la interfaz de usuario, ya que actualmente es una de las librerías más utilizadas en el desarrollo web. Además, me gustaría aprender a trabajar con ella porque considero que es una herramienta muy potente y con gran demanda en el mercado laboral.
    - Se combinará con **Tailwind CSS** para la parte visual. Aunque ya tengo experiencia con Bootstrap, prefiero los acabados que ofrece Tailwind y la flexibilidad que proporciona para adaptar los diseños. Creo que ambos elementos, React y Tailwind, pueden combinar muy bien para obtener una interfaz moderna, funcional y atractiva.

- Backend:
    - **Laravel** (PHP) será el framework elegido, ya que es una herramienta muy completa, bien documentada y con una curva de aprendizaje más accesible que otros frameworks como Angular o Nest. Además, cuenta con librerías como **Laravel Breeze**, que facilitan enormemente el desarrollo de funciones importantes como la autenticación de usuarios. También es un framework muy utilizado en el ámbito profesional, lo que lo convierte en una elección útil tanto para este proyecto como para mi futuro profesional.

- Base de datos:
    - Se usará **MySQL** como sistema de base de datos, por su estabilidad, compatibilidad con Laravel y facilidad de configuración.
    - Para la conexión entre la base de datos y el backend se usará **Eloquent**, el ORM que viene integrado en Laravel, ya que ambos funcionan de forma muy fluida y simplifican mucho el trabajo con las tablas y las relaciones.

- Despliegue:
    - La aplicación se desplegará en un VPS con Nginx, con copias de seguridad automatizadas.
    - Se utilizará GitHub para control de versiones y GitHub Actions para automatizar los despliegues (CI/CD).
    - Como alternativa, se podrá alojar el frontend en Vercel y el backend en un servidor VPS o servicio como Railway.

Material extra necesario:

- Dominio propio con certificado SSL (Let's Encrypt).
- Correo SMTP para notificaciones y recuperación de contraseñas.
- Almacenamiento de imágenes en el servidor.
- Iconos y gráficos de Flaticon, Heroicons o Tabler Icons.

- Tipografías de Google Fonts, como Inter o Poppins.
- Logo y favicon creados con Canva o Figma.

Nota final:

Aunque este es el stack tecnológico que prefiero utilizar, es posible que algunos elementos cambien dependiendo de las necesidades técnicas del proyecto o de los requisitos específicos establecidos en el protocolo de evaluación. La idea es mantener la flexibilidad para adaptar la aplicación a las circunstancias y asegurar que sea viable y funcional en cualquier entorno de desarrollo.

Justificación

La idea de este proyecto surge de una necesidad cotidiana muy común: la dificultad de planificar las comidas semanales de forma eficiente y práctica. Aunque existen aplicaciones similares, muchas resultan confusas, poco intuitivas o están pensadas únicamente para usuarios con conocimientos avanzados de cocina. Mi intención con este proyecto es crear una herramienta sencilla, clara y visual, que cualquier persona pueda utilizar sin complicaciones.

Lo que diferencia a mi gestor de menús semanales de otras propuestas es su enfoque centrado en la comodidad del usuario y la personalización de la experiencia. Quiero que el usuario pueda adaptar completamente la aplicación a su estilo de vida, eligiendo entre distintos tipos de filtros según sus preferencias: recetas rápidas, fáciles, elaboradas o incluso saludables. De este modo, no solo se facilita la planificación, sino que también se motiva a las personas a variar su alimentación sin invertir tanto tiempo en decidir qué cocinar.

Además, la generación automática de la lista de la compra, ajustada al número de comidas y personas, aporta un valor añadido muy práctico. Evita los olvidos frecuentes al hacer la compra y ahorra tiempo, reduciendo la necesidad de volver al supermercado por haber olvidado algo. En conjunto, el proyecto busca equilibrar simplicidad, utilidad y personalización, convirtiéndose en una herramienta realmente útil para la organización diaria.

A nivel técnico y personal, elegí este proyecto porque reúne varios elementos que me permiten aplicar de forma práctica los conocimientos adquiridos durante el curso. Se trata de un desarrollo que abarca múltiples operaciones CRUD —para la gestión de recetas, menús y listas de la compra—, lo que lo convierte en un proyecto completo desde el punto de vista del backend y del diseño de base de datos, pero sin perder la sencillez ni la claridad en su estructura.

Además, su carácter escalable lo hace especialmente interesante, ya que podría evolucionar fácilmente con nuevas funcionalidades, como el cálculo automático de calorías o valores nutricionales orientados a distintos perfiles de usuario. Considero que es un proyecto equilibrado: suficientemente amplio para demostrar un dominio sólido de la programación web, pero lo bastante accesible como para desarrollarlo de forma eficiente con una buena organización.

Objetivos

Objetivo general:

Desarrollar una aplicación web práctica y sencilla que permita al usuario planificar sus menús semanales, consultar recetas y generar automáticamente una lista de la compra adaptada a sus necesidades, facilitando así la organización de las comidas y el ahorro de tiempo en el día a día.

Objetivos específicos:

A continuación, se detallan los objetivos específicos del proyecto, organizados por categorías.
Para su desarrollo se establecen dos niveles de prioridad:

- **MVP (Minimum Viable Product)**: funcionalidades esenciales que deben estar implementadas para que la aplicación sea completamente funcional.

- **V2 (Versión 2)**: funcionalidades adicionales o ampliaciones que pueden incorporarse en una segunda fase, una vez finalizada la versión principal.

1. Base funcional (cuentas y seguridad)

- **[MVP]** Permitir el registro de nuevos usuarios.

- **[MVP]** Implementar el inicio y cierre de sesión (login/logout) de forma segura.

- **[MVP]** Permitir la edición de los datos personales del usuario.

- **[MVP]** Ofrecer la posibilidad de eliminar la cuenta de usuario.

- **[MVP]** Implementar un sistema para restablecer la contraseña mediante correo electrónico.

- **[V2]** Añadir verificación de correo electrónico tras el registro.

---

2. Experiencia de usuario

- **[MVP]** Diseñar una interfaz simple, clara e intuitiva que facilite la navegación por las distintas secciones.

- **[MVP]** Incorporar mensajes visuales de confirmación o error al realizar acciones (guardar, eliminar, comentar, etc.).

- **[V2]** Mejorar la accesibilidad general de la aplicación (contraste, etiquetas ARIA, tamaño de texto, etc.).

---

3. Roles y vistas de usuario

- **[MVP]** Crear una vista para el usuario no registrado, que permita consultar y filtrar recetas sin poder modificarlas.

- **[MVP]** Desarrollar una vista para el usuario registrado, que le permita crear, guardar, comentar recetas, generar menús y listas de la compra.

- **[MVP]** Incluir un rol de administrador con acceso para gestionar usuarios, recetas y comentarios.

- **[V2]** Desarrollar un panel visual unificado para el administrador que centralice la gestión de la aplicación.

---

4. Lista de la compra

- **[MVP]** Calcular automáticamente los ingredientes necesarios a partir de las recetas incluidas en el menú semanal.

- **[MVP]** Adaptar las cantidades en función del número de personas y comidas seleccionadas.

- **[MVP]** Permitir eliminar uno o varios ingredientes de la lista en caso de que el usuario ya los tenga en casa.

- **[MVP]** Incluir la opción de imprimir o descargar la lista de la compra.

- **[V2]** Organizar los ingredientes por categorías (verduras, carnes, lácteos, etc.) para facilitar la compra.

---

5. Recetario

- **[MVP]** Mostrar todas las recetas disponibles.

- **[MVP]** Permitir filtrar recetas por categoría, tiempo de preparación o dificultad.

- **[MVP]** Permitir crear, editar y eliminar recetas propias.
- **[MVP]** Añadir sistema de comentarios en las recetas.
- **[MVP]** Permitir guardar recetas y marcarlas como favoritas.
- **[V2]** Añadir etiquetas personalizadas (rápidas, saludables, económicas, etc.).

---

6. Gestión de menús

- **[MVP]** Incluir un formulario inicial que solicite datos para la planificación del menú:
    - Número de días.
    - Número de personas.
    - Número de comidas diarias.
    - Platos por comida.
    - Si desea repetir alguna receta.
- **[MVP]** Generar automáticamente una hoja de menú editable según las respuestas del formulario.
- **[MVP]** Desarrollar un asistente guiado que permita al usuario elegir recetas paso a paso (por comidas o platos).
- **[MVP]** Permitir guardar y reutilizar menús ya creados.
- **[V2]** Incorporar un sistema de sugerencias personalizadas según los menús o recetas más utilizadas.

---

7. Calidad técnica y mantenimiento

- **[MVP]** Mantener una estructura clara del proyecto separando frontend, backend y base de datos.
- **[MVP]** Validar los datos tanto en el cliente como en el servidor para evitar errores y mejorar la seguridad.

- **[MVP]** Crear un conjunto inicial de recetas de ejemplo (semillas de datos) para realizar pruebas.

- **[V2]** Implementar tests básicos para verificar el correcto funcionamiento de los cálculos y las principales funcionalidades.

- **[V2]** Configurar un entorno de despliegue sencillo con control de versiones y actualizaciones automatizadas.

Temporalización

La temporalización del proyecto se ha planteado en varias fases que permiten un desarrollo ordenado y progresivo. Cada fase incluye objetivos concretos, de manera que el proyecto avance de forma equilibrada entre la parte técnica, funcional y visual.

Fase 1: Análisis y diseño (Semana 1 - Semana 2)

- Definir los requisitos funcionales y técnicos del proyecto.
- Elaborar los primeros bocetos de la interfaz y estructura de navegación.
- Diseñar el modelo E/R y la base de datos.
- Planificar los roles de usuario y las funcionalidades básicas (registro, login, CRUDs principales).

Fase 2: Configuración del entorno y estructura base (Semana 3)

- Configurar los entornos de desarrollo (frontend y backend).
- Crear la estructura del proyecto en React, Laravel y MySQL.
- Implementar el control de versiones con Git y GitHub.
- Definir rutas principales y conexión inicial entre las partes del sistema.

Fase 3: Módulo de usuarios y seguridad (Semana 4 - Semana 5)

- Implementar el registro, inicio y cierre de sesión de usuarios.
- Crear el sistema de edición de perfil y eliminación de cuenta.
- Configurar la recuperación de contraseñas por correo.
- Añadir validaciones básicas de datos.

Fase 4: Gestión de recetas (Semana 6 - Semana 7)

- Crear la base de datos y los CRUDs de recetas.
- Permitir crear, editar, eliminar y visualizar recetas.
- Implementar los filtros por categoría, tiempo o dificultad.
- Añadir el sistema de comentarios y guardado de recetas favoritas.

Fase 5: Creación de menús personalizados (Semana 8 - Semana 9)

- Desarrollar el formulario inicial para generar menús personalizados.
- Programar el asistente guiado para la selección de recetas según comidas y platos.
- Permitir la edición de los menús generados.
- Implementar la opción de guardar y reutilizar menús anteriores.

Fase 6: Generación automática de la lista de la compra (Semana 10 - Semana 11)

- Desarrollar la lógica para calcular los ingredientes a partir de los menús seleccionados.
- Implementar el cálculo de cantidades según personas y comidas.
- Permitir eliminar ingredientes de la lista y marcar los ya disponibles.
- Añadir la función de imprimir o descargar la lista.

Fase 7: Interfaz y experiencia de usuario (Semana 12)

- Aplicar el diseño visual con Tailwind CSS.
- Mejorar la usabilidad y coherencia entre pantallas.

- Añadir mensajes de confirmación, alertas y pequeños detalles visuales.

Fase 8: Pruebas, ajustes y despliegue (Semana 13 - Semana 14)

- Realizar pruebas funcionales completas.
- Corregir errores y optimizar la base de datos.
- Añadir datos de prueba (recetas de ejemplo).
- Configurar el despliegue en servidor o entorno público.

Fase 9: Mejoras futuras (Versión 2)

- Implementar panel de administrador completo.
- Añadir nuevos filtros (por calorías, tipo de dieta, etc.).
- Incluir agrupación de ingredientes por categoría.
- Optimizar la aplicación con pequeñas automatizaciones y pruebas adicionales.

Definición de la Base de datos:

Para definir la base de datos seguiremos los siguientes pasos:

1. Análisis:
    a. Descripción de la app y sus actividades
    b. Descripción de cada elemento de la base de datos: entidades, relaciones y atributos. (descripción similar a los enunciados de clase para los esquemas E/R)
    c. Realizar el esquema E/R, representando las entidades, relaciones, atributos y cardinalidades, en formato draw.io y png/jpeg
2. Diseño:
    a. Aplicar las reglas de transformación y obtener las tablas del esquema relacional
    b. Estudiar de forma razonada la normalización y obtener las nuevas tablas normalizadas
    c. Estudiar la integridad referencial para cada clave foránea
    d. Implementar el diagrama del modelo relacional completo en MySQL WorkBench
3. Desarrollo:
    a. Crear el script de la base de datos y las tablas ajustando los tipos de datos e incluyendo las claves primarias, foráneas y alternativas en cada caso. Añade las restricciones más adecuadas según cada campo de la tabla (Not null, default, check,…) y adecuándose al caso real planteado.
    b. Crear otro script de datos para añadir al menos 4 registros de cada tabla para realizar las pruebas necesarias de comprobación de las restricciones.
4. Política de acceso:
    a. Política de usuarios (y roles).
    b. Asignación de privilegios.
    c. Gestión de cuentas de usuario, credenciales y control de accesos
    d. Mejora futura: Automatización de la asignación, revisión y gestión de roles
5. Automatización:
    a. Al menos un procedimiento almacenado que haga uso de cursores y del manejo de errores mediante handler.
    b. Funciones
    c. Triggers.

1. Análisis:
    a. **Descripción general del sistema:**
    FoodSynk es una plataforma digital enfocada en la gestión, organización y creación de recetas culinarias.
    Su principal objetivo es ofrecer a los usuarios un espacio intuitivo donde puedan descubrir nuevas recetas, crear las suyas propias, organizar menús personalizados y compartir experiencias gastronómicas con otros usuarios.
    La plataforma busca fomentar la creatividad culinaria y mejorar la planificación alimentaria mediante herramientas digitales accesibles y eficientes.

    Principales tareas y actividades:

    - Creación y gestión de recetas por parte de los usuarios, incluidas descripciones, pasos, imágenes, dificultad y tiempo de preparación.
    - Organización de recetas en menús personalizados, permitiendo planificar comidas semanales, temáticas o de eventos especiales.
    - Gestión estructurada de ingredientes, facilitando su uso en múltiples recetas y permitiendo consultas eficientes por tipo, nombre o presencia en platos concretos.
    - Interacción social entre usuarios, permitiendo que cada usuario pueda valorar recetas mediante “me gusta” y comentar platos creados por otros.
    - Exploración y consulta de contenido culinario, con un sistema que facilita descubrir nuevas recetas según categorías, preferencias o popularidad.
    - Gestión del perfil de usuario, donde cada miembro puede administrar su información, sus recetas creadas, sus menús y su actividad dentro de la plataforma.
    - Almacenamiento centralizado y seguro de los datos, asegurando la integridad de recetas, ingredientes, interacciones y menús para un funcionamiento fluido de la aplicación.

b. Descripción de cada elemento de la base de datos: entidades, relaciones y atributos. (descripción similar a los enunciados de clase para los esquemas E/R)

Vamos a definir el modelo de datos de FoodSynk, una aplicación web para gestionar recetas, ingredientes y menús, con funciones sociales de comentarios y “me gusta” entre usuarios.

- De cada usuario se guarda un identificador único, su nombre, su correo electrónico, una contraseña y la fecha en la que se registró.
  El correo electrónico debe ser único en toda la base de datos, es decir, no puede haber dos usuarios distintos con el mismo email.

- Un usuario puede crear recetas.
  También puede crear menús personalizados con sus propias recetas o con recetas de otros usuarios.

- De cada receta se guarda un identificador único, un título, una descripción, los pasos de preparación, el tiempo aproximado que tarda en elaborarse, el nivel de dificultad, una posible imagen ilustrativa y la fecha en la que se creó la receta.
  Interesa saber qué usuario ha creado cada receta, por lo que se almacenará la referencia al usuario correspondiente.

- Cada receta utiliza uno o varios ingredientes.
  De cada ingrediente se guarda un identificador único, el nombre del ingrediente y, opcionalmente, una categoría (por ejemplo, verdura, carne, lácteo, especia, etc.).
  Como un mismo ingrediente puede aparecer en muchas recetas y una receta puede tener muchos ingredientes, será necesario guardar también, para cada combinación receta–ingrediente, la cantidad utilizada y, en su caso, la unidad de medida (gramos, mililitros, unidades, etc.).

- Un usuario puede agrupar varias recetas en un menú.
  De cada menú se almacena un identificador único, un nombre, una breve descripción y la fecha de creación.
  Un menú puede contener muchas recetas y una misma receta puede formar parte de varios menús distintos.

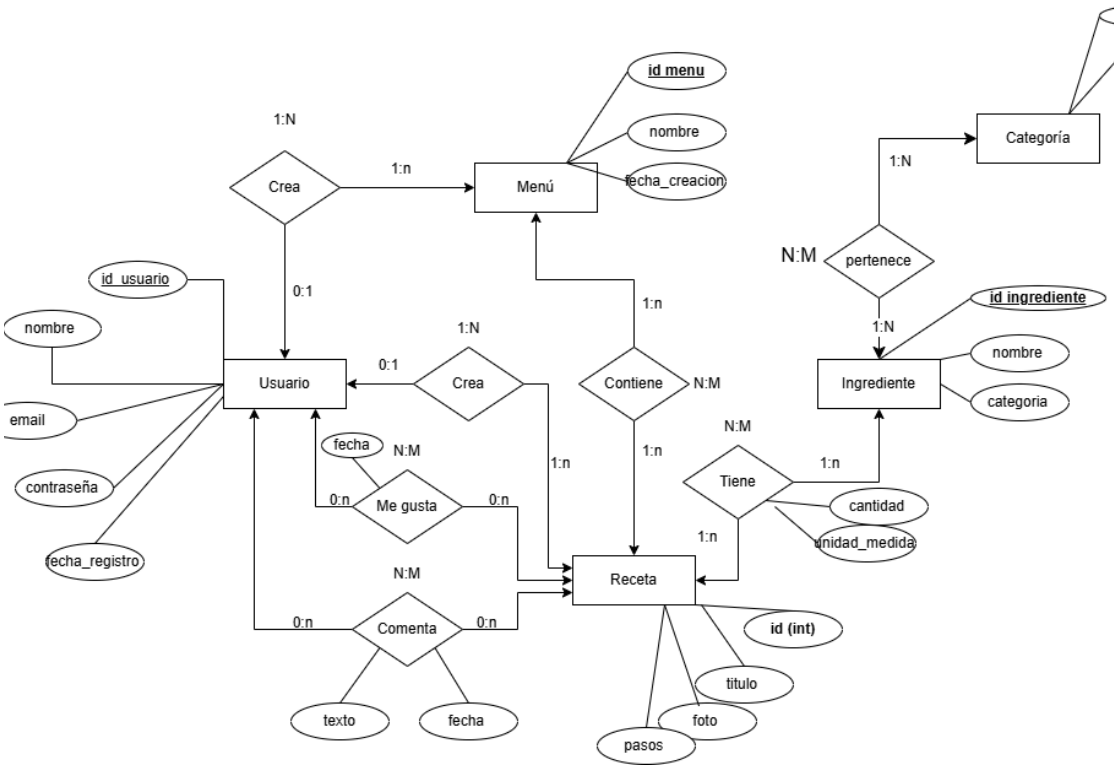
- Un usuario puede marcar una receta como que le gusta (“me gusta”).
  Un usuario solo puede darle “me gusta” a una receta una única vez.
  Habrá que llevar un registro de qué usuario ha dado “me gusta” a qué receta y en qué fecha/hora lo ha hecho.

- Un usuario puede escribir comentarios en una receta determinada.
  De cada comentario se guarda un identificador único, el texto del

comentario, la fecha y hora en la que se realizó, el usuario que escribió el comentario y la receta sobre la que se ha comentado. Una receta puede tener muchos comentarios y un usuario puede comentar muchas recetas diferentes.

Este conjunto de elementos y relaciones servirá de base para construir el esquema Entidad–Relación de FoodSynk, donde se representarán formalmente todas las entidades, sus atributos y las cardinalidades entre ellas.

c. **Realizar el esquema E/R, representando las entidades, relaciones, atributos y cardinalidades, en formato draw.io y png/jpeg**



2. Diseño.
    a. **Aplicar las reglas de transformación y obtener las tablas del esquema relacional**

Usuario(ID_Usuario(**PK**), nombre, email (UQ), contraseña, fecha_registro)

Receta(ID_Receta(PK), título, foto, pasos, ID_Usuario(FK))

Ingrediente(ID_Ingrediente(PK), nombre)

Categoría (ID_Categoria(PK),nombre)

Menu(ID_Menu(PK), nombre, fecha_creación, ID_Usuario(FK))

Receta_Ingrediente(cantidad, unidad_medida, ID_Receta(FK)(PK), ID_Ingrediente(FK)(PK))
Menu_Receta(ID_Menu(FK)(PK), ID_Receta(FK)(PK))
Me_gusta(fecha, ID_Usuario(FK)(PK), ID_Receta(FK)(PK))
Comentario(texto, fecha, ID_Usuario(FK)(PK), ID_Receta(FK)(PK))
Ingrediente_Categoría(ID_categoría(PK)(FK),ID_categoría(PK)(FK))

- 
Inicio > Dominios >
Añadir un subdominio
Los subdominios son direcciones de Internet para las distintas secciones de su sitio web. Estos utilizan su nombre de dominio principal y un prefijo. Por ejemplo, si su dominio es domain.com, un subdominio podría ser store.domain.com. También puede crear un subdominio wildcard introduciendo el símbolo * en vez del nombre. En este caso, los visitantes del sitio serán redireccionados a este subdominio sin tener en cuenta el nombre de subdominio que hayan introducido en su navegador.
Información:
ATENCION:
Recuerda! Antes de manipular o actualizar una web es imprescindible HACER UNA COPIA DE SEGURIDAD.
Nombre del subdominio *
Foodsynk
drogon.online
Introduzca * para crear un subdominio wildcard.
Configuración de hosting
Raíz del documento *
/ Foodsynk.drogon.online
Ruta al directorio principal del sitio web.
* Campos obligatorios
Aceptar
Cancelar

Sitio relacionado: Foodsynk.drogon.online

Usuarios

Cree un usuario predeterminado para la base de datos. Plesk accederá a la base de datos en nombre de este usuario. Si no se asigna ningún usuario de base de datos a la base de datos, no podrá accederse a la misma.

- [x] Crear un usuario de la base de datos

Nombre de usuario de la base de datos *: nurlopbor09

- No se permite el nombre de usuario de base de datos "root".
- Ya existe un nombre de usuario de la base de datos denominado root.

Contraseña *: g66qbY1Kkc. Generar

- Su contraseña no es suficientemente compleja. Conforme a la directiva del servidor, la seguridad mínima es de Segura (recomendado).

- [x] El usuario tiene acceso a todas las bases de datos de la suscripción seleccionada

Control de acceso:

- ( ) Solo permitir conexiones locales
- (•) Permitir conexiones remotas desde cualquier host
- ( ) Permitir conexiones remotas desde

Contacte con su proveedor de hosting y compruebe que la opción especificada arriba cumplirá con las reglas actuales del firewall para el acceso entrante a MariaDB/MySQL.