CONSEJERÍA DE EDUCACIÓN
Comunidad de Madrid

IES ENRIQUE TIERNO GALVAN

Parla

CFGS DESARROLLO DE APLICACIONES MULTIPLATAFORMA

Curso 2024/2025

Proyecto DAM

TITULO: Desarrollo de un sistema de gestión multiplataforma con Flutter y Odoo

***Alumno*: Laura Salas Ávila**

***Tutor*: Julián Parra Perales**

Junio de 2025

Contenido

(Profe no sé cómo poner el índice, hago salto para que no esté junto?)

Contexto de la aplicación.

Mundo real del problema.

Actualmente las empresas prefieren estar más cerca de lo digital, la comodidad y también la movilidad para poder gestionarse de una manera más eficiente y rápida, ya no sirve con tener las cosas a papel o en un dispositivo fijo en la oficina, es por eso que las empresas buscan herramientas o sistemas que las permitan organizarse y tener de forma más accesible sus tareas, pedidos, reuniones con sus clientes o proveedores, incluso saber qué debe de hacer cada trabajador en cualquier momento y desde cualquier dispositivo. Es por este motivo que las aplicaciones ERP pueden llegar a ser muy utilizadas por las empresas para poder llegar a tener o mejorar esa gestión de información, pero muchas de estas soluciones ERP llegan a ser algo difíciles de usar o no se llegan a necesitar de manera completa por muchos de los trabajadores, ya que ellos no necesitan entrar a un ERP completo, sino que necesitan entrar a una herramienta que les permita ver de forma rápida lo que realmente quieren. Además de que muchas de estas soluciones llegan a ser caras o están pensadas para grandes empresas, muchas de ellas no llegan a poder ser usadas de manera multiplataforma o si lo llegan a ser, no suelen ser muy agradables para los usuarios. Es por esto que se ha decidido solucionar este problema que iremos explicando a lo largo de este documento, donde se creará una solución más sencilla, rápida y eficiente de una aplicación centrada en la gestión diaria de una empresa.

Qué aplicaciones existen.

Se ha realizado un pequeño análisis del mercado para ver qué aplicaciones existen actualmente que intenten realizar o se parezcan a lo que se está intentando desarrollar en este proyecto, es decir, una aplicación que tenga como objetivo ayudar a las empresas en su gestión diaria de una forma más sencilla y accesible. Hay bastantes aplicaciones

en el mercado que permiten a los usuarios organizar sus tareas, hacer seguimientos, repartir trabajos, etc... Pero muchas de ellas no están pensadas para poder conectarse a un ERP, ser multiplataforma o simplemente que no llegan a sr tan prácticas para el tipo de uso que se quiere conseguir en este proyecto. Se han escogido para realizar una comparación de nuestra idea las siguientes aplicaciones de gestión: (esto no sé si te servirá o debo de cambiarlo, no entendí muy bien el comentario profe)

• Trello y Asana: Conocidas para organizar tareas de manera visual, se usan en empresas para proyectos con tableros, asignaciones y todo eso, pero no están pensadas para empresas que tienen pedidos, productos, clientes…

• Notion: Para organizar casi cualquier cosa con ella, pero todo hay que montarlo a mano y no tiene automatización ni conexión con otros sistemas como un ERP. Para cosas más personales va bien, pero para gestionar una empresa con tareas reales no es suficiente.

• Monday.com y Zoho Creator: Permiten montar soluciones un poco más a medida parecidas a pequeñas apps de gestión, pero muchas opciones son de pago, pueden ser complicadas de configurar y no están centradas en tareas como pedidos, trabajadores, entregas...

Como se puede ver en estos ejemplos, falta una aplicación que se centre en tareas reales que pueda llegar a tener una empresa en el día a día, que se pueda usar de manera más fácil e intuitiva, que no tenga muchas opciones inútiles o precios elevados, que tenga un diseño más decente y que además se pueda conectar a un sistema ERP sin complicaciones. Por eso este proyecto se busca ofrecer justo eso: una aplicación de gestión sencilla, que al igual que estas sea multiplataforma, pero con un mejor diseño y sea más intuitiva de usar, que sea accesible para cualquier tipo de empresa y que esté pensada especialmente para quienes necesitan organizar su trabajo diario sin tener que hacer uso de herramientas complicadas o de costosas. Además, a diferencia de otras soluciones, esta aplicación estará diseñada para integrarse fácilmente con un sistema ERP, permitiendo así que las empresas gestionen tareas reales como pedidos, reuniones o asignaciones sin perder tiempo navegando entre opciones que no necesitan.

Justificación de este proyecto y diferencia del mercado.

Es por esto que en este proyecto vamos a dar solución a los problemas explicados anteriormente y vamos a diferenciarnos del resto de sistemas de gestión, permitiendo a las empresas tener una forma mucho más fácil, cómoda y multiplataforma de acceder o interactuar con sus datos desde cualquier sitio y con cualquier dispositivo para poder gestionarse. La idea principal es que cualquier usuario, ya sea un trabajador o un administrador pueda ver sus tareas, reuniones, asignaciones, pedidos o incluso más sin tener que entrar a sistemas o herramientas más grandes o complicadas que muchas veces no sirven para el uso diario.

Esta solución no pretende ser un ERP, sino una aplicación de gestión que será más sencilla y rápida que se conecte a uno existente (que en este caso será Odoo) para facilitar el trabajo de quienes solo necesitan gestionar su parte. Con esto, los usuarios podrán centrarse solo en lo que necesitan hacer, sin tener que navegar por menús complicados con mucha información o entender cómo funciona un ERP. Además, no todas las empresas pueden pagar licencias caras o aplicaciones que exigen tener un gran conocimiento sobre su uso, por eso, en este proyecto también se busca que sea una opción accesible, gratuita y que pueda adaptarse a cualquier tamaño de empresa.

(no sé si esto va aquí o lo borro:)

En resumen, en este proyecto se busca el objetivo de ayudar a las empresas a organizarse mejor en su día a día sin tener que usar sistemas complicados, costosos o poco adaptables y cómodos. Para ello se va a desarrollar una aplicación de gestión multiplataforma con Flutter para la parte visual (frontend), conectada a un backend gestionado con Odoo que se encargará de gestionar la lógica del sistema y almacenará la información en una base de datos PostgreSQL. La comunicación entre ambos se realizará intercambiando datos en formato JSON a través de una conexión segura con HTTPS, mientras que todo el sistema será virtualizado mediante Docker.

Casos de Uso.

(quiero mirarlo contigo profe, es que con las prácticas no podía conectarme a la reunión)

Requisitos funcionales y no funcionales.

Repasando todo lo anterior los requisitos tanto funcionales como no funcionales que son necesarios para este proyecto son los siguientes: (no sé si me faltarán, iré viendo, cambié la forma de escribirlos)

- Funcionales:
  - Gestionar el inicio de sesión de usuarios según su rol (administrador o trabajador).
  - Crear nuevas tareas, reuniones o pedidos.
  - Consultar tareas, reuniones o pedidos asignados.
  - Asignar tareas o pedidos a uno o varios trabajadores.
  - Marcar tareas o pedidos como completados.
  - Consultar el estado de las tareas (pendiente, en curso, completada).
  - Visualizar los detalles de cada tarea o pedido (cliente, trabajador, producto, fecha, etc.).
  - Ver las tareas organizadas en una lista o calendario para una mejor planificación.
  - Eliminar tareas o pedidos que ya no sean necesarios.
  - Editar la información de una tarea si ha habido algún cambio.

- No funcionales:
  - La aplicación debe ser multiplataforma (móvil, escritorio, web…).
  - La aplicación debe ser fácil de entender y usar, incluso para usuarios con poco conocimiento.
  - La interfaz debe estar bien organizada y debe ser visualmente agradable.
  - Las tareas de cualquier tipo deben poder crearse con pocos clics o pasos.
  - El rendimiento debe ser fluido, incluso si hay muchos usuarios o tareas en el sistema.
  - Deben verse mensajes para el usuario en caso de error o cuando una acción se complete de manera correcta.
  - El sistema debe ser flexible y permitir futuras actualizaciones (nuevos tipos de tareas, nuevos roles, mensajes...).

Diseño.

GUI.

He realizado unos bocetos sobre cuál sería la idea para la interfaz del usuario, como se va a poder ver, es un diseño bastante agradable y simple, en cada apartado haremos una breve explicación de los aspectos de usabilidad que se han tenido en cuenta:

UI.

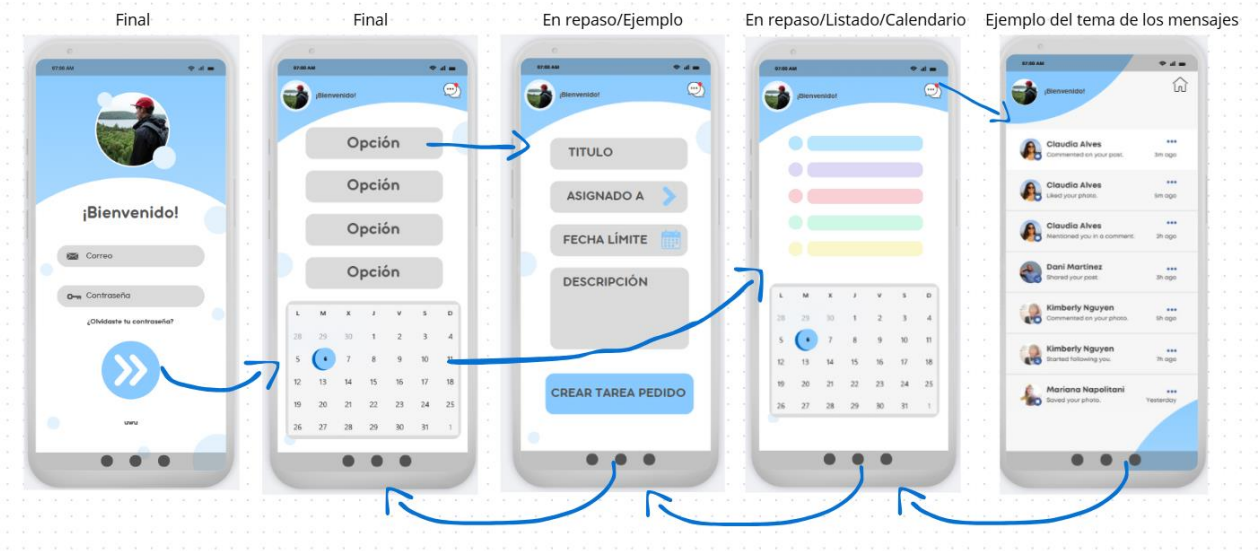
(vistas, no sé si quieres las vistas de diseño o las vistas de cómo se ven en la aplicación al final)

UX.

(experiencia)

Diagrama de navegación.

Este es el diagrama de navegación de las pantallas de nuestra aplicación, en él explicaremos cómo nos desplazaríamos por cada una de ellas. (Está un poco en proceso ya que hay partes del diseño que estoy cambiando aún) (añadir pequeña explicación de navegación)



Reutilización.

(posible en pantallas de las opciones)

Arquitectura.

En este proyecto se ha desarrollado una solución formada por varios sistemas y subsistemas que trabajan juntos para poder permitir la gestión de tareas, pedidos, asignaciones y más de una manera sencilla, eficiente y desde cualquier dispositivo. La arquitectura se ha pensado para que sea clara, escalable y mantenible, y para ello se va a separar lo máximo posible los distintos elementos que formarán este sistema. Aunque algunas partes de esta, como el ERP o la base de datos, no se hayan desarrollado desde cero, sí se han configurado y conectado en el sistema. A continuación, vamos a nombrar y explicar cómo se compone esta arquitectura y cuáles son cada uno de sus componentes.

1. Aplicación (subsistema desarrollado):

   La aplicación ha sido desarrollada en Flutter y es la interfaz principal del sistema. Funciona como una app multiplataforma que se puede ejecutar en dispositivos móviles, escritorios, navegadores…Este subsistema ha sido desarrollado desde cero.

2. Backend – Odoo (subsistema independiente)

   Odoo actúa como sistema de gestión de datos y de lógica empresarial. No se ha programado desde cero, pero se ha desplegado y configurado activando los módulos necesarios para habilitar las funcionalidades necesarias para este proyecto.

3. Base de datos – PostgreSQL (componente interno)

   Odoo utiliza PostgreSQL como sistema de gestión de bases de datos en el que almacena todos los datos del sistema: las tareas, pedidos, usuarios, categorías, etc. A esta base de datos no se accede directamente desde la aplicación, sino a través del backend de Odoo.

4. Capa de comunicación (API REST sobre HTTPS)

   Toda la comunicación entre la aplicación y Odoo se realiza mediante peticiones HTTPS gracias a la configuración de un servidor proxy NGINX. Este proxy gestiona el cifrado SSL para asegurar que toda la información intercambiada esté protegida.

Con este uso de HTTPS se asegura de que el sistema sea seguro incluso cuando se accede desde otras redes externas u otros dispositivos móviles.

Despliegue.

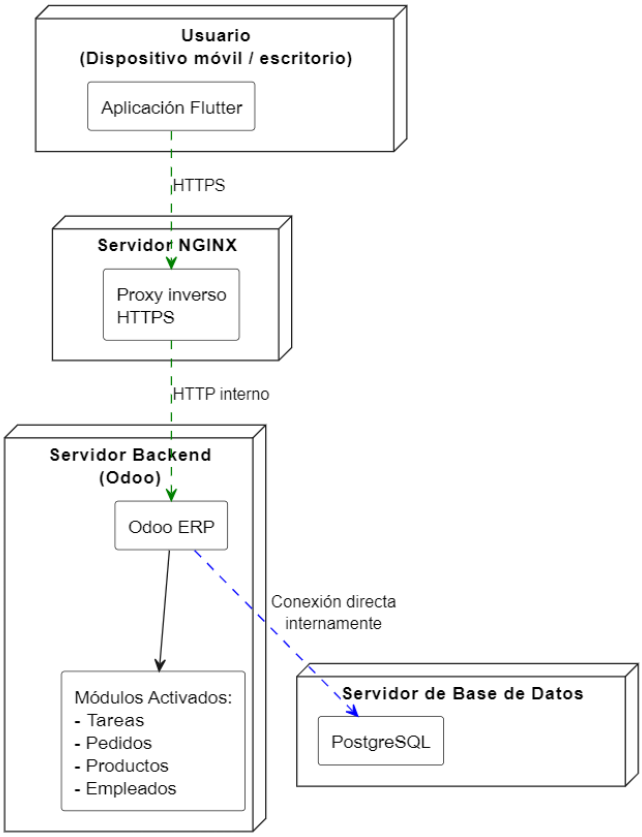
Todo el sistema ha sido desplegado mediante contenedores Docker organizados. Esta virtualización nos permite:

- Ejecutar Odoo, PostgreSQL y NGINX en contenedores separados pero interconectados.
- Asegurar de que el sistema funcione igual en cualquier máquina.

La estructura del despliegue es la siguiente:

- Contenedor de PostgreSQL: que contiene la base de datos.
- Contenedor de Odoo: ejecuta el ERP.
- Contenedor NGINX: actúa como proxy inverso, cifrando todas las comunicaciones del cliente a través de HTTPS.
- Aplicación del cliente: desarrollada en Flutter, esta se ejecuta por separado (osea fuera de Docker), conectándose al backend mediante HTTPS.

Gracias a esta configuración cada parte del sistema puede reiniciarse o actualizarse de forma separada sin afectar a los demás.



Componentes.

A nivel de componentes individuales, el sistema desarrollado se puede dividir en las siguientes partes, cada una con una función específica dentro de esta arquitectura:

- Frontend (Flutter):
    - Es la interfaz principal con la que interactúan los usuarios y está diseñada para funcionar en múltiples plataformas (móvil, web, escritorio...) y permite:
        - Iniciar sesión según el tipo del usuario.
        - Visualizar tareas, pedidos, reuniones, asignaciones…
        - Crear o modificar tareas y asignarlas a otros empleados.
        - Ver la información en formato de lista o calendario.
        - Cambiar el estado de una tarea.

- Controlador de la comunicación (API REST)
    - Es la capa encargada de conectar la app con el backend. Se encarga de:

        - Enviar solicitudes HTTP al servidor (Odoo).
        - Interpretar respuestas JSON.
        - Gestionar errores de conexión o autenticación.
        - Garantizar que los datos estén actualizados entre el cliente y servidor.

- Backend (Odoo ERP)
    - Odoo funciona como backend. En este componente se han configurado:
        - Módulos de tareas, productos, empleados, contactos...
        - Control de acceso por tipo de usuario.

- Base de datos (PostgreSQL)
    - Gestionada internamente por Odoo, es el almacenamiento de todos los datos de este proyecto:
        - Tareas, productos, usuarios, reuniones, asignaciones....
        - Solo accesible mediante Odoo.

- Servidor NGINX
    - Funciona como proxy inverso y se encarga de:
        - Redirigir el tráfico externo al backend de Odoo.
        - Asegurar que las comunicaciones sean HTTPS.
        - Mejorar la seguridad del sistema al controlar el acceso.

- Entorno de despliegue (Docker)
    - Todos los servicios anteriores se han desplegado en contenedores Docker, y esto permite:
        - Ejecutar Odoo, PostgreSQL y NGINX.
        - Iniciar el entorno de desarrollo fácilmente.
        - Asegurar que todo funcione igual en distintos entornos.

Base de datos.

Nuestro proyecto hace uso de Odoo como ERP, y este utiliza una base de datos PostgreSQL que se encarga de guardar toda la información de los módulos que tenga Odoo. Esta base de datos no fue creada desde cero ya que al hacer la dockerización de Odoo, este se encargó de realizar las tablas y las relaciones necesarias dependiendo de los módulos que se van activando. Asi que, el diseño de la base de datos no forma parte directa del trabajo que se está desarrollando, pero sí es importante saber qué datos utiliza la aplicación y cómo se accede a ellos.

Módulos de Odoo utilizados

Para que la aplicación funcione correctamente y tenga acceso a los datos que se van a necesitar, se han activado varios módulos dentro de Odoo. Estos módulos son los encargados de crear y mantener las tablas que el sistema usará. Los principales módulos que van a ser utilizados son los siguientes:

- Contactos: Para la gestión de clientes, proveedores y empleados.
- Inventario o Productos: Para la gestión de productos, categorías y cantidades.
- Ventas/Pedidos: Para registrar pedidos de los clientes y su estado.
- Proyectos o Tareas: Para modelar las tareas que se asignan a trabajadores.

Cada uno de estos módulos se activa y mantiene un conjunto de tablas en PostgreSQ que son utilizadas por la aplicación. Aunque no se ha diseñado directamente esta base de datos, se han analizado estas tablas para entender qué campos son las que se usarán en la app.

Uso de la base de datos en la aplicación

Desde la aplicación, se accede a los datos de Odoo mediante su API. Estos datos son de cada una de las tablas internas que han sido creadas por los módulos que se han activado. Como ejemplo:

- Cuando un usuario consulta sus tareas asignadas, la app accede a la información de los módulos de pedidos y empleados.
- Al crear una nueva tarea, se registra automáticamente en el sistema de Odoo.

- Si se actualiza el estado de una entrega, se modifica directamente el campo necesario en la base de datos de Odoo.

En resumen, la aplicación no tiene control de una manera directa sobre la estructura de la base de datos, pero sí la utiliza a través del backend. Por lo tanto, el diseño de base de datos en este proyecto se basa en saber qué módulos se van a necesitar, qué entidades se van a utilizar y cómo se relacionan entre ellas dentro de Odoo.

Paquetes, Interfaces y Clases.

(ir viendo)

Plan de pruebas.

El plan de pruebas que se usará para poder darle el visto bueno a nuestra solución está formado por una serie de pruebas que se deben de cumplir y funcionar correctamente, estas pruebas que explicaremos a continuación nos ayudarán a descubrir posibles errores en la aplicación:

1. Pruebas de arranque y conexión:
    - Verificar que la aplicación inicia correctamente desde distintos dispositivos (móvil, web, escritorio…).
    - Comprobar que se establece la conexión con el sistema de gestión (backend).
    - En el caso de error de conexión, la aplicación debe mostrar un mensaje al usuario.
2. Pruebas de acceso y control de usuarios:
    - El sistema debe permitir el inicio de sesión con usuarios registrados.
    - Según el tipo de usuario (administrador o trabajador), se debe ver el contenido y las funcionalidades que le correspondan.
    - Probar que un trabajador no puede acceder a funciones que no le correspondan.
3. Pruebas de gestión de tareas:
    - Crear nuevas tareas de distintos tipos (pedido, reunión…).

- Asignar tareas a trabajadores desde el administrador.
- Cambiar el estado de una tarea (por ejemplo de pendiente a completada) y comprobar que se actualiza correctamente.
- Visualizar tareas asignadas por fecha o por tipo. (ir viendo)

4. Visualización y navegación:
   - Asegurarse de que los datos (usuarios, tareas, productos...) se ven correctamente en la aplicación.
   - Confirmar que el calendario o la lista de tareas muestra la información que se espera.
   - Probar la navegación entre pantallas para asegurar que no hay errores ni bloqueos.
5. Comunicación entre cliente y servidor:
   - Confirmar que los datos creados en la app se ven correctamente en Odoo.
   - Comprobar que los datos modificados desde Odoo se actualizan en la aplicación.
   - Simular una desconexión o un fallo en la red para ver cómo funcionaría el sistema.
6. Pruebas en diferentes plataformas:
   - Ejecutar la aplicación en dispositivos móviles, navegadores web y escritorio para comprobar la compatibilidad y el correcto funcionamiento de cada uno de ellos.
   - Verificar que el diseño se adapta bien a los distintos tamaños de pantalla.
7. Pruebas de rendimiento:
   - Crear una gran cantidad de tareas y comprobar que la aplicación sigue funcionando sin problemas.
   - Medir el tiempo que tarda en cargar una lista de tareas larga.

8. Pruebas de usabilidad:
    - Validar que la interfaz es clara y fácil de usar para un usuario sin conocimientos.
    - Comprobar que las acciones se intentan realizar en pocos pasos.
9. Pruebas de seguridad y control:
    - Confirmar que los datos de los usuarios no se comparten entre perfiles distintos.
    - Comprobar que solo los usuarios autorizados puedan realizar ciertas acciones (crear tareas, asignar, eliminar...).

Implementación de la aplicación.

La implementación de este proyecto ha seguido una planificación ordenada basada en el análisis y diseño realizados previamente. Aunque no se han desarrollado todos los casos de uso al completo, se ha creado la base funcional necesaria para comprobar que el sistema cumple su objetivo y se puede ampliar en el futuro sin problemas. A continuación, se explican dos aspectos clave de la implementación: el entorno de desarrollo utilizado y la futura puesta en producción del sistema.

Entorno de desarrollo

Para desarrollar esta aplicación se ha utilizado Flutter como framework principal y Dart como lenguaje, el entorno de desarrollo ha sido Visual Studio Code que se ha configurado con las extensiones necesarias para trabajar con Flutter, Dart y GitHub.

(realizar desarrollo con buenas prácticas siguiendo una estructura limpia y organizada del código, uso de control de versiones mediante GitHub, y organización del trabajo. Ir añadiendo y explicando)

La estructura del proyecto incluye:

- Carpeta lib/ con (añadir las carpertas del código)

El backend utiliza Odoo y se ha desplegado junto con la base de datos PostgreSQL mediante Docker usando un archivo docker-compose.yml, lo que facilita el arranque y la configuración del entorno.

Implantación/Puesta en producción

El proyecto ha sido diseñado para que su despliegue y ejecución sea sencillo y puedes ejecutarse en distintos entornos. Para ello, se han tomado en cuenta los siguientes aspectos: (ir añadiendo)

Capturas de la ejecución de la funcionalidad.

Ejecución de pruebas

Herramientas usadas en el proyecto para su ejecución.

Para el desarrollo de este sistema de gestión he utilizado una serie de herramientas que permiten que la aplicación funcione correctamente y pueda cumplir con los objetivos propuestos en los apartados anteriores. A continuación, se mencionarán cada una de ellas con una pequeña explicación de su uso: (por el momento estas, ir explicándolas más adelante)

- Flutter
- Dart
- Odoo
- PostgreSQL
- Docker
- NGINX

Además de estas herramientas para la ejecución, durante el desarrollo del proyecto se han utilizado otras como Visual Studio Code o GitHub, aunque no son necesarias en la ejecución, han sido importantes en el proceso de implementación.

Elementos destacables del desarrollo.

A medida que se ha ido realizando el proyecto, fueron surgiendo innovaciones y varios problemas que se fueron solucionando, vamos a explicar cada uno de ellos en diferentes apartados y en el caso de los problemas, serán explicados en el orden en el que aparecieron mientras se iba avanzando.

Problemas a la hora del desarrollo.

Error 500 en Odoo.

(cambiado para que lo entiendan los demás profes, luego borro esto)

Al terminar de montar Odoo y su base de datos en Docker e intentar acceder desde el navegador me aparecía el error 500 (un error interno del servidor) que no me dejaba acceder. Después de revisar los mensajes de error que salían en la consola e investigar un poco, me di cuenta de que el problema venía por dos cosas:

- Por algunos puertos que estaba intentando usar y ya estaban ocupados, así que Odoo no podía iniciarse bien.
- Y por la base de datos que no estaba configurándose del todo bien al arrancar, así que Odoo no podía conectarse a ella correctamente.

Después de comprobarlo, lo que hice para solucionarlo fue cambiar los puertos para que no hubiera conflicto y añadir unas líneas de código que me faltaban en el archivo de configuración que usé para el montaje (llamado docker-compose) para arrancar de forma correcta la base de datos.

Cuando terminé de configurarlo, volví a arrancar el sistema y ahora sí funcionaba todo como debería, permitiéndome acceder a Odoo y tener la base de datos.

Error al intentar realizar conexión HTTPS.

Al intentar configurar la conexión segura mediante HTTPS para mi aplicación utilizando un proxy inverso con NGINX, surgieron varios problemas que hicieron que la conexión no funcionara. El objetivo era poder acceder a Odoo de forma segura, usando un dominio personalizado con certificado SSL, pero esto resultó ser más complicado y terminó en error. Después de revisar los mensajes de error e investigar un poco, me di cuenta de que el problema venía por estas dos cosas:

- Por los puertos de nuevo, los puertos que se estaban intentando usar no estaban abiertos por lo que terminaba en error todo el rato.
- Por el fallo en la herramienta certbot que nos permitiría tener un certificado SSL gratuito usando Let’s Encrypt, este error era debido a los puertos, ya que al no estar abiertos no se podía validar el dominio por la falta de acceso a estos y el certificado no pudo generarse.

Este problema por el momento aún no se ha terminado de solucionar, pero esta todo configurado (NGINX, Docker, Certbot y dominio con DuckDNS) y preparado para su funcionamiento.

Problemas al realizar pantallas.

(poner otro día)

Innovaciones del proyecto.

En este proyecto hemos usado varias innovaciones que iremos puntualizando y explicando sobre por qué se usaron y dónde se usaron. (se pondrán ya al final)

Líneas de trabajo futuro.

Requisitos pendientes e ideas pensadas para las siguientes versiones.

En esta primera versión de la aplicación se pondrá en funcionamiento alguno de los apartados de esta, pero en un desarrollo futuro todas las opciones estarían disponibles, más optimizadas e incluso se podrían añadir más, mejorando así la aplicación. Algunos de los requisitos o ideas pensadas (por el momento) para las futuras versiones de la aplicación son las siguientes:

• Mejorar los niveles de seguridad de la aplicación.

• Que la aplicación sea muy escalable y no dé errores.

• Tener una mejor accesibilidad para personas con discapacidades.

• Poder realizar actualizaciones y añadir nuevas opciones/módulos.

• Añadir automatización en ciertos apartados de la aplicación, incluso hacer

uso de inteligencia artificial para crear diagramas o analizar datos que den reportes.

• Tener notificaciones en tiempo real e incluso chats entre usuarios de la misma empresa para el paso de información.

• Mejorar el seguimiento de las tareas y sus estados.

• Mejorar los roles de los usuarios y todos sus permisos.

Conclusiones.

(Hacer DAFO para esto)

Bibliografía/Enlaces de interés.

(configuración flutter/dart)

(investigación del mercado)

(conexión https) + (Docker odoo + PostgreSQL)

(documentación flutter para arquitectura limpia)

(voy añadiendo para recordar, ya lo pondré bien con APA)

Anexos.

GitHub

Repositorio de GitHub con el documento subido: LauraSA29 (Laura Salas)