

Nombre: MBOApp (Maintenance Back Office APP)

* Logo aún por definir

Índice

Introducción	3
Entrevista	3
Diseño del sistema	4
Validación y pruebas	7
Links de apoyo	7

Introducción

La aplicación se trata de una PWA (Progressive Web App) que será creada en GoormIDE mediante el Framework de Laravel, Vue.js, Ionic, con base de datos en Firebase. Constará de un diseño para uso web en plataforma de PC con todas las funcionalidades y un formato para móvil que permitirá el apartado de notificación y partes.

Entrevista

Día 15/03/2021

Necesita de una aplicación web móvil (PWA) que cumpla la funcionalidad de gestión de Backoffice, con funcionalidad web para una empresa de mantenimiento.

Debe cubrir los apartados de facturación y tarifas, creación y modificación de partes sobre averías, notificación de los servicios realizados y selección de un particular (empresa).

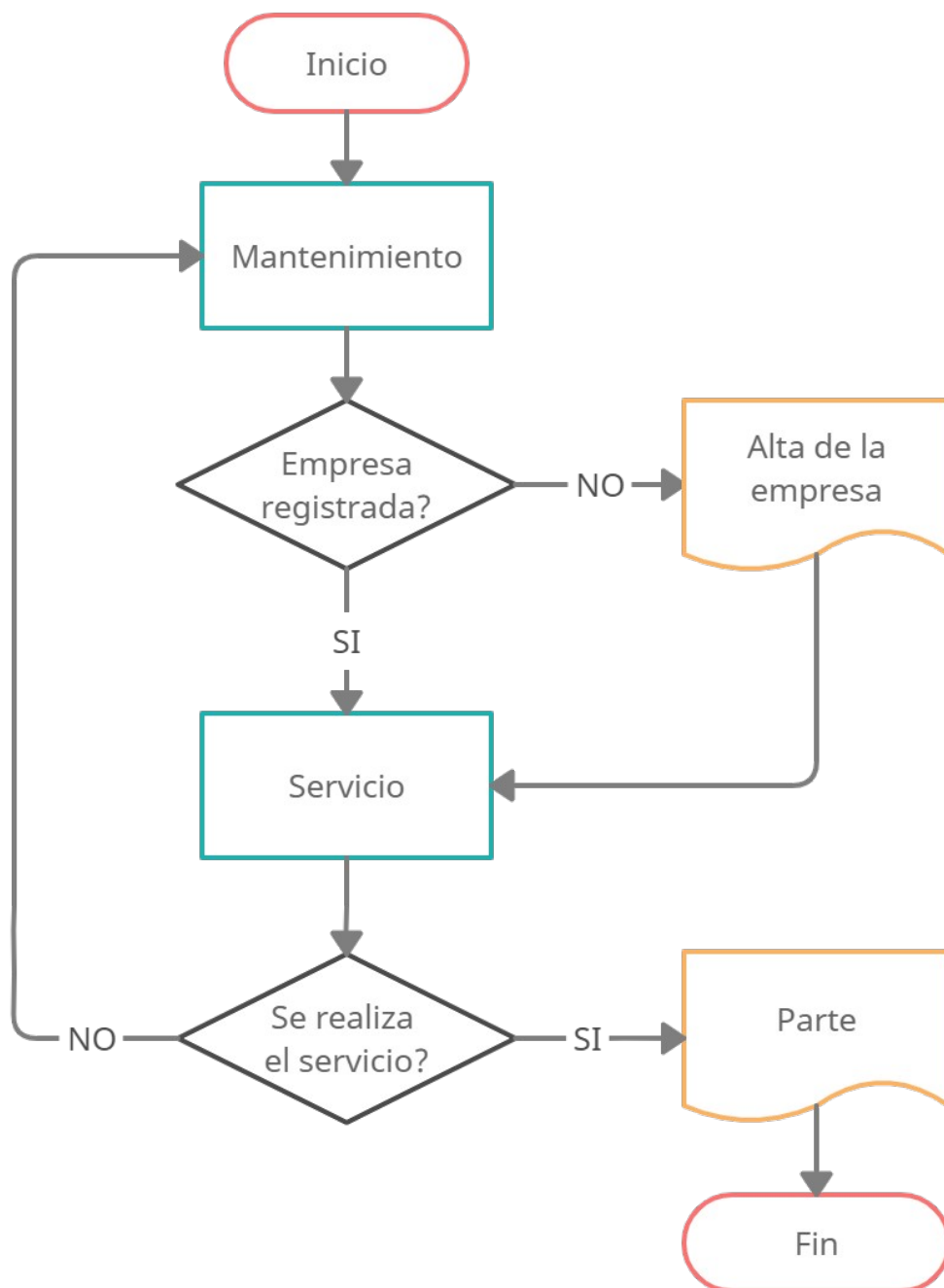
En la aplicación, debe permitir el uso de tres tipos de usuarios (Administrador, Encargado y empleado) que tendrán sus respectivos permisos:

1. Administrados: Tendrá los permisos de crear y editar los demás usuarios del sistema, dar de alta a las empresas, modificar los partes, realización de las facturas (enfoque a lo económico), modificación de las tarifas (los precios según los servicios realizados a las empresas).
2. Encargado: Tendrá permisos a dar de alta a la empresas y modificar/crear los partes.
3. Empleado: Solo podrá realizar partes.

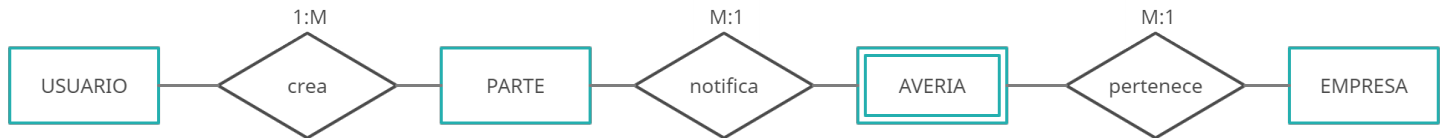
Diseño del sistema

Flujo de datos

Representación gráfica de la realización de un notificación de un servicio.



Base de datos



USUARIO	
id_usuario (PK)	int
tipo_usuario	varchar
nombre_usuario	varchar
apellido_usuario	varchar
email_usuario	varchar
telefono_usuario	int
estatus_usuario	varchar
fecha_alta_usuario	date
fecha_modificacion_usuario	date
fecha_baja_usuario	date

PARTE	
id_mantenimiento (PK)	int
id_usuario (FK)	int
id_averia (FK)	int
fecha_parte	date
hora_parte	time
horas_realizadas	int
observaciones	varchar
fecha_alta_parte	date
fecha_modificacion_parte	date
fecha_baja_parte	date

AVERIA	
id_averia (PK)	int
id_empresa (FK)	int
descripcion_averia	varchar
estatus_averia	varchar
fecha_alta_averia	date
fecha_modificacion_averia	date
fecha_baja_averia	date

EMPRESA	
id_empresa (PK)	int
CIF	varchar
nombre_empresa	varchar
ciudad_empresa	varchar
direccion_empresa	varchar
telefono_empresa	int
email_empresa	varchar
fecha_alta_empresa	date
fecha_modificacion_empresa	date
fecha_baja_empresa	date

USUARIO	
id_usuario (PK)	int
tipo_usuario	varchar
nombre_usuario	varchar
apellido_usuario	varchar
email_usuario	varchar
telefono_usuario	int
estatus_usuario	varchar
fecha_alta_usuario	date
fecha_modificacion_usuario	date
fecha_baja_usuario	date

PARTE	
id_mantenimiento (PK)	int
id_usuario (FK)	int
id_averia (FK)	int
fecha_parte	date
hora_parte	time
horas_realizadas	int
observaciones	varchar
fecha_alta_parte	date
fecha_modificacion_parte	date
fecha_baja_parte	date

AVERIA	
id_averia (PK)	int
id_empresa (FK)	int
descripcion_averia	varchar
estatus_averia	varchar
fecha_alta_averia	date
fecha_modificacion_averia	date
fecha_baja_averia	date

EMPRESA	
id_empresa (PK)	int
CIF	varchar
nombre_empresa	varchar
ciudad_empresa	varchar
direccion_empresa	varchar
telefono_empresa	int
email_empresa	varchar
fecha_alta_empresa	date
fecha_modificacion_empresa	date
fecha_baja_empresa	date

Razonamiento de la relación PARTE-AVERIA (M:1)

En un hipotético caso de que el cliente desconozca la causa del error, el técnico puede ir para inspeccionar y descubrir la avería, por lo que puede realizar un parte con intención de informar. Posteriormente, puede regresar una segunda vez con las herramientas/materiales correspondientes a la resolución del problema.

*Otra posible solución a esta: modificar la descripción de la avería.
Dependiendo del problema, se deja abierto la posibilidad a ambas.

Funcionalidad de la interfaz

Diseño para PC	Diseño para móvil
Estará orientado al control de los diferentes datos almacenados, con los permisos correspondientes según el tipo de usuario. Tendrá una barra superior con las diferentes opciones que mostrará un listado CRUD de los datos almacenados.	Su principal función es la de notificar partes de los mantenimientos realizados, indicando la empresa, fecha, hora, empresa, horas empleadas y un campo a describir el servicio. En caso de no disponer de conexión a internet, podrá crear los partes y tenerlos guardados en la caché del móvil hasta que vuelva a tener internet. Incluye el apartado de notificaciones.

Tiempo estimado

Base de datos	5h
Creación de código	50h
Pruebas y validaciones	10h
Control de errores	5h

Requisitos mínimos

Dispositivo móvil, PC y conectividad a internet.

Validación y pruebas

Se tendrán controles de los datos insertados y modificados, control de permisos, seguridad al acceder a las diferentes vistas (requerido un login). Se restringirá el borrado en cascada en caso de borrar/modificar algún campo de las tablas de base de datos. No se declara una copia de seguridad automatizada.

Se realizarán pruebas con datos aleatorios, pruebas de rendimiento, comprobación de la efectividad de las notificaciones a tiempo real, entre otros.

Para el mantenimiento de la aplicación, se garantiza la resolución de posibles errores futuros durante 1 año.

Links de apoyo

[Tutorial con Ionic/Vue](#)
[Requisitos para una PWA](#)
[Vue](#)
[PWA básico](#)