

Ejercicio de programación

Un taller de mantenimiento de vehículos se encuentra a día de hoy gestionando sus procesos usando papel, cuadernos y algún que otro software de cálculo, sin embargo, en los últimos meses el taller ha visto un crecimiento constante en su base de clientes y la necesidad de ofrecerles nuevos servicios, pero la forma actual de gestionar sus procesos no es para nada eficiente y el taller está empezando a tener problemas para organizar sus servicios y gestionar sus recursos.

El dueño del taller, proyectando un crecimiento del negocio a largo plazo, desea implementar un sistema de software web que permita gestionar eficientemente los clientes, el manejo de los servicios, el control de repuestos y la facturación, sin embargo, el dueño que es un mecánico automotriz desconoce sobre temas de tecnología y ha decidido consultar con el equipo de ventas Sitecpro para buscar una solución, el dueño del taller dejó en claro las siguientes consideraciones:

- **El taller es un negocio pequeño (4~6 empleados) conformado totalmente por mecánicos quienes no son muy hábiles con el uso de la tecnología.**
- **El taller funciona casi la totalidad del día turnando mecánicos durante la jornada que adicional a su trabajo de mecánica en sí, se tienen que hacer cargo de la atención al cliente, control de repuestos y registro de vehículos, por lo que el sistema debe ser “global” para todos.**
- **Considerando lo anterior, el dueño quiere que se implemente un log para llevar un control sobre las actividades del taller, este log solo debe ser accesible por él.**

El equipo de ventas traslada el caso al equipo de desarrollo del cual usted ha sido el asignado para plantear una solución y levantar un **prototipo** para demostrárselo al dueño del taller y otros potenciales clientes de Sitecpro.

Adicionalmente, el equipo de ventas y gerencia le solicitan que el prototipo esté listo en un máximo de 25 días de calendario a partir del día en el que usted fue notificado.

CONSIDERACIONES TECNICAS

La solución debe plantearse considerando la siguiente arquitectura de software:



La base de datos debe conformarse obligatoriamente con los siguientes modelos de datos (pueden agregar campos adicionales y tablas si lo consideran adecuado):

Cientes

- Tipo
 - Individual
 - Empresa
- Nombres (individuales)
- Apellidos (individuales)
- Nombre de empresa
- Direccion
 - Direccion ("X calle 29-42")
 - Zona
 - Municipio (depende del departamento)
 - Departamento (depende del país)
 - País
- NIT
- DPI
- Teléfono fijo
- Teléfono móvil

Municipio, Departamento y país deben tener tabla propia y los datos deben estar precargados para cualquier instancia. Adicional a Guatemala agregue los datos de otros dos países de Centroamérica. Estas tablas solo deberían ser editables en base de datos

Vehículos

- Placa
- Propietario (un cliente de los registrados previamente)
- Marca (Toyota, Kia, Isuzu, etc.)
- Tipo (depende de la marca, Hilux, Corolla, D-MAX, etc.)
- Modelo
- Color

Marca y tipo deben tener tabla propia y los datos deben estar precargados para cualquier instancia. Considere agregar 3 marcas y al menos 2 tipos de vehículos de cada marca. Estas tablas solo deberían ser editables en base de datos

Repuestos

- Nombre ("Aceite W10-40")
- Stock
- Vehículos (Vehículos que son compatibles, si aplica)

Servicio

- Tipo de servicio
- Vehículo (de los registrados previamente)
- Kilometraje
- Trabajos (lista, "cambio de aceite", "limpieza de frenos", "cambio de pastillas", etc.)
- Repuestos (lista, "Aceite", "Frenos")
- Estado del servicio
 - Ingresado
 - En Proceso
 - En revisión
 - Terminado
 - Facturado
 - Entregado

Tipo de servicio y trabajos deben tener tabla propia aparte y a los tipos de servicio se le debería poder parametrizar una lista de trabajos predeterminados. Queda a criterio del desarrollador como implementar estas dos tablas.

Facturas

- Cliente a facturar
- NIT
- Monto
- Detalle
 - “Repuesto 1”
 - “Repuesto 2”
 - “Servicio 1”
 - “Servicio 2”

FASES

- ❑ Análisis del problema, creación del diagrama y generación del script que construye la base de datos.
 - ❑ PostgreSQL
 - ❑ SQL Server
- ❑ Para el manejo de los datos (BACK-END): agregar, eliminar, modificar se utilizará un ORM
 - ❑ Construcción de una API (C#)
 - ❑ Utilizar Entity framework como ORM, en un proyecto de tipo biblioteca de clases
 - ❑ Crear una API, que utilice la biblioteca de clases con el ORM
 - ❑ Comunicarse con la base de datos, con métodos que agreguen registros, modifiquen y eliminen, en el caso de eliminación se debe de tomar en cuenta que los registros nunca se eliminan, solo se cambia su estado.
- ❑ Consumo de los datos (FRONT-END)
 - ❑ Crear una vista, que tenga dos opciones
 - ❑ Opción A: Búsqueda de repuestos
 - ❑ Opción B:
 - ❑ Crear un servicio de vehículo
 - ❑ Revisar las características del servicio y estado

OTRAS CONSIDERACIONES

1. Para el desarrollo de las APIs utilice las convenciones y buenas prácticas que se mencionan en esta página: <https://restfulapi.net/resource-naming/>
2. Puede usar cualquier herramienta o librería que le facilite el desarrollo.
3. Queda totalmente a su criterio el diseño de UI/UX, solo tome en cuenta que la pagina debe poder utilizarse tanto en PC como en Móvil.
4. Toda operación a base de datos debe hacerse únicamente a través del ORM, query-strings no están permitidos.
5. No es necesaria la implementación de testing.
6. Tenga siempre en mente las necesidades del cliente y la mantenibilidad del sistema.