### **Arquitectura basadas en Vista-Servidor y en MVC**

#### **1. Capa de Presentación (Front-End) (Basado en MVC)**

* **Tecnología**: Angular/Ionic
* **Descripción**:
  + Se encargará de las interfaces de usuario tanto para clientes como para mecánicos y administrador
  + Diferentes vistas para cada rol:
    - **Clientes (Móvil)** : login, historial de mantenimientos, consulta de cotizaciones, reservas, panel de estado del vehículo.
    - **Mecánicos(Móvil)**: Gestión de mantenimientos, creación de cuentas cliente, historial de clientes y vehículos, propuesta de servicios, agenda diaria.
* **Interacción**: Esta capa consume APIs RESTful para interactuar con el back-end.

#### **2. Capa de Lógica de Negocio (Back-End) (Basado en Vista-Servidor)**

* **Tecnología sugerida**: PHP Laravel
* **Descripción**:
  + Aquí se encuentra toda la lógica de negocio de la aplicación.
  + Gestiona las funcionalidades como creación de cuentas, manejo de sesiones, notificaciones por email o SMS, creación de reservas, actualización del estado de los vehículos, etc.
* **Módulos principales**:
  + **Módulo de Usuarios**: Gestiona el registro, login, y roles (cliente, mecánico, administrador).
  + **Módulo de Mantenimientos**: Gestiona los historiales de mantenimiento y paneles de control de mecánicos.
  + **Módulo de Reservas**: Manejo de reservas de horas para clientes.
  + **Módulo de Vehículos**: Información y gestión de los vehículos en el taller. (Por ejemplo: marca y modelos).
  + **Módulo de Cotización:** Gestiona los historiales de cotizaciones, consulta de precios y panel de control de cotizaciones.
  + **Módulo de empleados:** Información y gestión de los empleados del taller.
  + **Módulo de estados:** Creación ygestión de los estados de cada una de las mantenciones realizadas.
  + **Módulo de servicios:** Información y gestión de los tipos de servicios disponibles de un taller especifico.

#### **3. Capa de Datos (Base de Datos)**

* **Tecnología ocupada**: MySQL
* **Descripción**:
  + Almacena toda la información de usuarios, vehículos, mantenimientos, reservas, y notificaciones.

### **Flujo de Trabajo (Ejemplo de Flujo para el Cliente):**

1. El **cliente** accede a la app móvil o web.
2. **Front-End** (Angular/Ionic) envía una solicitud a la API RESTful /api/users/login/ para iniciar sesión.
3. **Back-End** valida el login mediante el módulo de **usuarios** y devuelve un token de autenticación.
4. El **cliente** puede ver su historial de mantenimientos accediendo a /api/mantenimientos/, y el Front-End muestra los resultados a la vista.
5. Si el **cliente** desea hacer una reserva, el Front-End envía los datos a /api/reservas/ para guardar la nueva reserva en la base de datos.
6. El módulo de **notificaciones** envía un correo o SMS de confirmación.

### **Diagrama Simplificado de la Arquitectura**

