Proyecto Grupo Nº 6

**Plan de proyecto: Procesos de una empresa de marketing de contenido**

**<<Versión 1.0>>**

Apellidos y Nombres:

-Avila Pacheco, Anthony

-Castillo Vilchez, Greysy Nicole

-Figueroa Millan, Daniel Jonas

-Gutiérrez La Serna, Oscar Arturo

-Rupay Olascoaga, Sebastian Eli

<<Agencia de Marketing de Contenido>>

<<29/05/2022>>

**ÍNDICE**

[1.](#_gjdgxs) DESCRIPCIÓN 5

[2.](#_30j0zll) ALCANCE 5

[3.](#_1fob9te) EDT 6

[4.](#_3znysh7) CRONOGRAMA 6

[5.](#_2et92p0) REQUERIMIENTOS 10

[5.1. REQUERIMIENTOS FUNCIONALES 10](#_tyjcwt)

[6.](#_3dy6vkm) CASOS DE USO 10

[CU001: Registrar Nuevo Cliente 11](#_1t3h5sf)

[Diagrama del caso de uso 11](#_2s8eyo1)

[Especificación del caso de uso 11](#_17dp8vu)

[Modelo de Datos 12](#_3rdcrjn)

[Análisis de caja negra 12](#_26in1rg)

[Prototipo 13](#_lnxbz9)

[Diagrama de secuencia 14](#_35nkun2)

[Código del SP 14](#_1ksv4uv)

[Pruebas del SP 16](#_44sinio)

[Prueba 1 16](#_2jxsxqh)

[Prueba 2 16](#_3j2qqm3)

[CU002: Reservar Asesoría 17](#_4i7ojhp)

[Diagrama del caso de uso 17](#_2xcytpi)

[Especificación del caso de uso 17](#_1ci93xb)

[Modelo de Datos 18](#_3whwml4)

[Análisis de caja negra 19](#_2bn6wsx)

[Prototipo 19](#_qsh70q)

[Diagrama de secuencia 20](#_3as4poj)

[Código del SP 20](#_1pxezwc)

[Pruebas del SP 22](#_49x2ik5)

[Prueba 1 22](#_2p2csry)

[Prueba 22](#_147n2zr)

[CU003: Realizar Pedido de Servicio 23](#_3o7alnk)

[Diagrama del caso de uso 23](#_23ckvvd)

[Especificación del caso de uso 23](#_ihv636)

[Modelo de Datos 24](#_32hioqz)

[Análisis de caja negra 24](#_1hmsyys)

[Prototipo 25](#_41mghml)

[Diagrama de secuencia 28](#_2grqrue)

[Código del SP 28](#_vx1227)

[Pruebas del SP 30](#_3fwokq0)

[Prueba 1 30](#_1v1yuxt)

[Prueba 2 30](#_4f1mdlm)

[CU004: Consultar Asesorías 31](#_2u6wntf)

[Diagrama del caso de uso 31](#_19c6y18)

[Especificación del caso de uso 31](#_3tbugp1)

[Modelo de Datos 32](#_28h4qwu)

[Análisis de caja negra 33](#_nmf14n)

[Prototipo 34](#_37m2jsg)

[Diagrama de secuencia 35](#_1mrcu09)

[Código del SP 35](#_46r0co2)

[Pruebas del SP 36](#_2lwamvv)

[Prueba 1 36](#_111kx3o)

[Prueba 2 37](#_3l18frh)

[CU005: Registrar Asesor 37](#_206ipza)

[Diagrama del caso de uso 37](#_4k668n3)

[Especificación del caso de uso 37](#_2zbgiuw)

[Modelo de Datos 38](#_1egqt2p)

[Análisis de caja negra 39](#_3ygebqi)

[Prototipo 39](#_2dlolyb)

[Diagrama de secuencia 40](#_sqyw64)

[Código del SP 40](#_3cqmetx)

[Pruebas del SP 42](#_1rvwp1q)

[Prueba 1 42](#_4bvk7pj)

[Prueba 2 42](#_2r0uhxc)

[CU006: Consultar Pedido 43](#_1664s55)

[Diagrama del caso de uso 43](#_3q5sasy)

[Especificación del caso de uso 43](#_25b2l0r)

[Modelo de Datos 44](#_kgcv8k)

[Análisis de caja negra 44](#_34g0dwd)

[Prototipo 45](#_1jlao46)

[Diagrama de secuencia 46](#_43ky6rz)

[Código del SP 46](#_2iq8gzs)

[Pruebas del SP 47](#_xvir7l)

[Prueba 1 47](#_3hv69ve)

[Prueba 2 47](#_1x0gk37)

[7. MODELO FÍSICO 48](#_4h042r0)

[7.1. Modelo en software erwin 48](#_2w5ecyt)

[modelo en sql server 48](#_1baon6m)

[8.](#_3vac5uf) DICCIONARIO DE DATOS 49

[A.](#_2afmg28) Tabla Gerente 49

[B.](#_pkwqa1) Tabla Ciudad 50

[C.](#_39kk8xu) Tabla Turno 50

[D.](#_1opuj5n) Tabla Cliente 50

[E.](#_48pi1tg) Tabla Asesores 51

[F.](#_2nusc19) Tabla Asesorías 51

[G.](#_1302m92) Tabla Plan 51

[H.](#_3mzq4wv) Tabla Pedido 52

[9.](#_2250f4o) SCRIPT 52

[10.](#_haapch) CONCLUSIONES 69

[11.](#_319y80a) RECOMENDACIONES 69

# **DESCRIPCIÓN**

Dentro del proyecto planteado, se incluye la estructuración de la creación de una prototipado que emplea la base de datos fijado a una agencia de marketing digital ya que no cuenta con una base de datos que le permita trabajar de una manera más fluida, por otro lado este proyecto lo estamos realizando con el propósito de que la empresa pueda contar con una base de datos y asimismo pueda facilitar a los trabajadores y clientes de la agencia de marketing a que puedan realizar sus operaciones de manera más ordenada.

Por otro lado, el cliente podrá realizar distintos procesos a la hora de realizar sus operaciones la cual les facilitará el trabajo a los empleadores de la agencia de marketing para una buena atención y un mejor servicio para los cliente, ya que a la hora de adquirir su servicio será mucho mejor por la manera la cuál ahora lo podrán realizar.

Asimismo, este prototipo que realizaremos para está agencia de marketing digital le será de mucho beneficio ya que los clientes podrán realizar la adquisición de los servicios de una manera más sencilla y factible, la cual evitará que los clientes tengan muchos inconvenientes y reclamos frente a su adquisición de servicios.

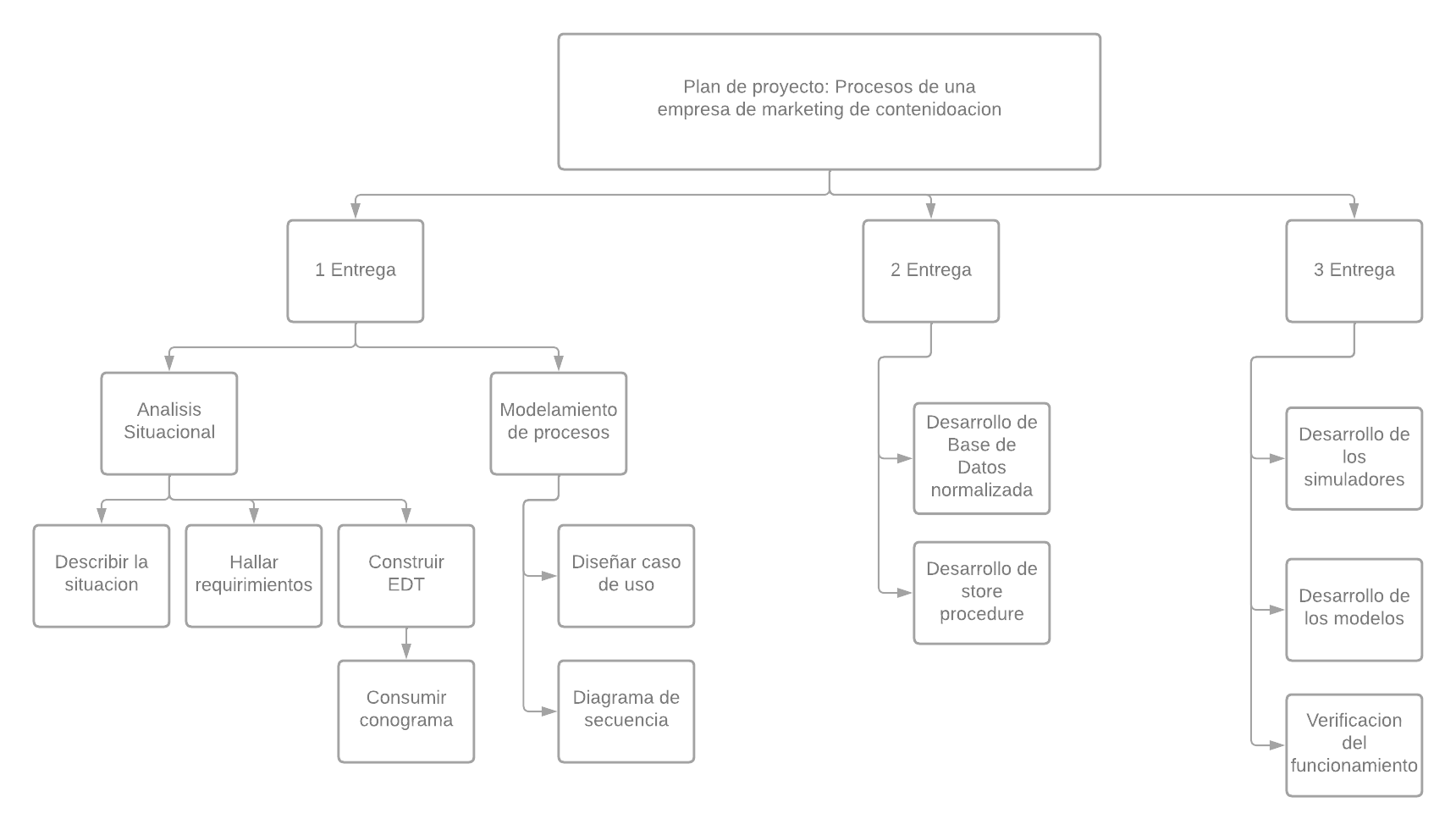
Por último y para finalizar buscamos lo mejor para esa agencia de marketing digital es por ello nosotros queremos realizar este prototipo y proyecto para su mejora en lo que ellos ya están acostumbrados a lo que es el marketing digital.

1. **ALCANCE**

Nuestro objetivo es entregar un prototipo de la base de datos para una agencia de marketing de contenido

Este proyecto está netamente enfocado a realizar la base de datos y el prototipo del software que emplea dicha base de datos. Además de lo mencionado la estructuración de la base de datos se realizará en SQL Server por lo cual el plazo entregable de este proyecto concluye o se llevará a cabo durante el intervalo del mes de julio y agosto. Sin embargo, toda la implementación y elaboración se llevará a cabo con las pruebas respectivas a fin de ofrecer un buen resultado

# **EDT**



# **CRONOGRAMA**

| Tareas | | |  | | |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Código** | **Titúlo** | | **Comprometer tarea** | | | **Estimación horas** | **Estado** |
| TA001 | Crear reunión para realizar un concenso sobre el tema que elegiremos | | Figueroa Millan, Daniel Jonas | | | 1 | Sí |
| TA002 | Elección del tema realizado | | Todos los integrantes | | | 1 | Sí |
| TA003 | Debate de titúlo del proyecto | | Todos los integrantes | | | 2 | Sí |
| TA004 | Creación de titúlo según el proyecto elegido | | Todos los integrantes | | | 2 | Sí |
| TA005 | Distribución de las tareas | | Figueroa Millan, Daniel Jonas | | | 1 | Sí |
| TA006 | Indagar sobre la descripción del proyecto | | Avila Pacheco, Anthony | | | 1 | Sí |
| TA007 | Plantear la descripción del proyecto | | Avila Pacheco, Anthony | | | 1 | Sí |
| TA008 | Efectuar la descripción del proyecto | | Avila Pacheco, Anthony | | | 1 | Sí |
| TA009 | Recopilar información sobre los alcances del proyecto | | Gutiérrez La Serna, Oscar Arturo | | | 1 | Sí |
| TA010 | Sucitar los alcances del proyecto | | Gutiérrez La Serna, Oscar Arturo | | | 1 | Sí |
| TA011 | Desempeñar alcance del proyecto | | Gutiérrez La Serna, Oscar Arturo | | | 1 | Sí |
| TA012 | Averiguar lo que se requiere para desarrollar del EDT | | Rupay Olascoaga, Sebastian | | | 1 | Sí |
| TA013 | Elaborar EDT del proyecto | | Rupay Olascoaga, Sebastian | | | 1 | Sí |
| TA014 | Culminar EDT del proyecto | | Rupay Olascoaga, Sebastian | | | 1 | Sí |
| TA015 | Efectua el listado para el cronograma | | Castillo Vilchez, Greysy Nicole | | | 1 | Sí |
| TA016 | Producir el avance del cronograma | | Castillo Vilchez, Greysy Nicole | | | 1 | Sí |
| TA017 | Realización de cronograma | | Castillo Vilchez, Greysy Nicole | | | 1 | Sí |
| TA018 | Distribución de requerimientos funcionales | | Figueroa Millan, Daniel Jonas | | | 1 | Sí |
| TA019 | Distribución de requerimientos no funcionales | | Figueroa Millan, Daniel Jonas | | | 1 | Sí |
| TA020 | Aceptación de realizar los requerimientos funcionales | | Avila Pacheco, Anthony | | | 1 | Sí |
| TA021 | Realizar cuestionarios sobre los requerimientos al dueño de la empresa | | Avila Pacheco, Anthony | | | 1 | Sí |
| TA022 | Listado de requerimientos funcionales | | Avila Pacheco, Anthony | | | 1 | Sí |
| TA023 | Aceptación de realizar los requerimientos no funcionales | | Castillo Vilchez, Greysy Nicole | | | 1 | Sí |
| TA024 | Realizar cuestionarios sobre los requerimientos al dueño de la empresa | | Castillo Vilchez, Greysy Nicole | | | 1 | Sí |
| TA025 | Listado de requerimientos no funcionales | | Castillo Vilchez, Greysy Nicole | | | 1 | Sí |
| TA026 | Distribución de análisis del proyecto | | Figueroa Millan, Daniel Jonas | | | 1 | Sí |
| TA027 | Distribución de los CU | | Gutiérrez La Serna, Oscar Arturo | | | 1 | Sí |
| TA028 | Averiguar lo requerido para la CU | | Gutiérrez La Serna, Oscar Arturo | | | 1 | Sí |
| TA029 | Identificación de actores en el CU | | Gutiérrez La Serna, Oscar Arturo | | | 1 | Sí |
| TA030 | Identificación de los CU | | Gutiérrez La Serna, Oscar Arturo | | | 1 | Sí |
| TA031 | Reconocimiento de la funcionalidad del CU | | Gutiérrez La Serna, Oscar Arturo | | | 1 | Sí |
| TA032 | Determinar las relaciones de los CU | | Gutiérrez La Serna, Oscar Arturo | | | 1 | Sí |
| TA033 | Culminar con los CU | | Gutiérrez La Serna, Oscar Arturo | | | 1 | Sí |
| TA034 | Distribución de los prototipos del proyecto | | Figueroa Millan, Daniel Jonas | | | 1 | Sí |
| TA035 | Hacer un boceto sobre el prototipo | | Figueroa Millan, Daniel Jonas | | | 1 | Sí |
| TA036 | Realizar el modelo y la prueba | | Figueroa Millan, Daniel Jonas | | | 1 | Sí |
| TA037 | Indagar soluciones sobre el prototipo | | Figueroa Millan, Daniel Jonas | | | 1 | Sí |
| TA038 | Llevar a cabo distintas experimentaciones del prototipo | | Figueroa Millan, Daniel Jonas | | | 1 | Sí |
| TA039 | Finalización del avance del prototipo | | Figueroa Millan, Daniel Jonas | | | 1 | Sí |
| TA040 | Distribución de los diagramas de secuencias | | Rupay Olascoaga, Sebastian | | | 1 | Sí |
| TA041 | Explicar la lógica del algoritmo a realizar | | Rupay Olascoaga, Sebastian | | | 1 | Sí |
| TA042 | Describir los pasos realizados en un caso de uso UML | | Rupay Olascoaga, Sebastian | | | 1 | Sí |
| TA043 | Instruir los procesos de negocios, flujos de trabajo y el sistema | | Rupay Olascoaga, Sebastian | | | 1 | Sí |
| TA044 | Simplificar los procesos de negocios, flujos de trabajo y el sistema | | Rupay Olascoaga, Sebastian | | | 1 | Sí |
| TA045 | Modelación de los elementos del diagrama de secuencias | | Rupay Olascoaga, Sebastian | | | 1 | Sí |
| TA046 | Distribución de diseño del proyecto | | Avila Pacheco, Anthony | | | 1 | Sí |
| TA047 | Distribución de modelo fisico | | Avila Pacheco, Anthony | | | 1 | Sí |
| TA048 | Convertir las entidades en tablas | | Avila Pacheco, Anthony | | | 1 | Sí |
| TA049 | Creación de las entidades en las tablas | | Avila Pacheco, Anthony | | | 1 | Sí |
| TA050 | Convertir relaciones en claves externas | | Avila Pacheco, Anthony | | | 1 | Sí |
| TA051 | Convertir atributos en columnas | | Avila Pacheco, Anthony | | | 1 | Sí |
| TA052 | Realizar modidicaciones del modelo fisico | | Avila Pacheco, Anthony | | | 1 | Sí |
| TA053 | Distribución del diccionario de datos | | Castillo Vilchez, Greysy Nicole | | | 1 | Sí |
| TA054 | Elegir los nombres de los campos a rellenar | | Castillo Vilchez, Greysy Nicole | | | 1 | Sí |
| TA055 | Elegir la clave primaria | | Castillo Vilchez, Greysy Nicole | | | 1 | Sí |
| TA056 | Elegir los tipos de datos | | Castillo Vilchez, Greysy Nicole | | | 1 | Sí |
| TA057 | Realizar una descripción de cada campo | | Castillo Vilchez, Greysy Nicole | | | 1 | Sí |
| TA058 | Distribución del script para SQL Server | | Gutiérrez La Serna, Oscar Arturo | | | 1 | Sí |
| TA059 | Seleccionar el servidor para ejecutar SQL Server | | Figueroa Millan, Daniel Jonas | | | 1 | Sí |
| TA060 | Expandir el nodo de base de datos | | Figueroa Millan, Daniel Jonas | | | 1 | Sí |
| TA061 | Realizar una copia de seguridad | | Figueroa Millan, Daniel Jonas | | | 1 | Sí |
| TA062 | Seleccionar el script en una nueva ventana | | Figueroa Millan, Daniel Jonas | | | 1 | Sí |
| TA063 | Revisar el código en la ventana consulta | | Figueroa Millan, Daniel Jonas | | | 1 | Sí |
| TA064 | Generar la script | | Figueroa Millan, Daniel Jonas | | | 1 | Sí |
| TA065 | Finalización del proyecto | | Todos los integrantes | | | 8 | Sí |

# **REQUERIMIENTOS**

## **5.1. REQUERIMIENTOS FUNCIONALES**

**RF01**: El sistema debe permitir al cliente registrarse con su correo o cuenta de la página.

**RF02:** El cliente puede reservar una asesoría para recibir recomendaciones de acuerdo a sus necesidades.

**RF03:** El cliente puede registrar su pedido de servicio, eligiendo un plan que se adecue a lo que requiere, para luego realizar el pago.

**RF04**: El sistema debe permitir al cliente renovar su contrato de servicio.

**RF05**: El sistema debe mostrar al asesor las asesorías reservadas por los clientes.

**RF06**: El sistema debe mostrar los pedidos de servicio registrados.

* 1. REQUERIMIENTOS NO FUNCIONALES

**RNF01**.El sistema deberá ser desarrollado en PHP

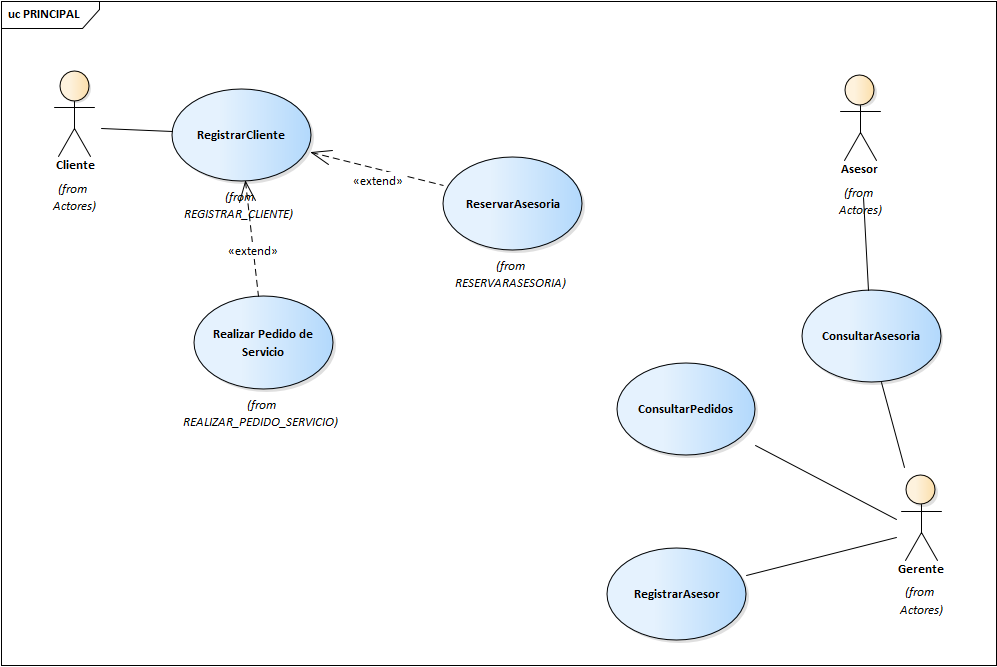
**RNF02**.El sistema deberá ser creado como gestor de base de datos SQL Server

**RNF03**.El sistema mantendrá una estructura UI y UX

**RNF04.**El sistema del prototipo debe contar con colores llamativos y colores fuertes.

# **CASOS DE USO**

| Ítem | Nombre | Descripción |
| --- | --- | --- |
| 1 | Registrar Cliente | El sistema debe permitir al cliente registrarse con su correo o cuenta de la página. |
| 2 | Reservar Asesoría | El cliente puede reservar una asesoría para recibir recomendaciones de acuerdo a sus necesidades. |
| 3 | Registrar Pedido de Servicio | El cliente puede registrar su pedido de servicio, eligiendo un plan que se adecue a lo que requiere, para luego realizar el pago. |
| 4 | Registrar Asesor | El gerente registra un asesor dentro del sistema. |
| 5 | Consultar Asesorías | El sistema debe mostrar, al asesor, las asesorías reservadas por los clientes. |
| 6 | Consultar Pedido de Servicio | El sistema debe mostrar los pedidos de servicio registrados. |



# CU001: Registrar Nuevo Cliente

## 

## Diagrama del caso de uso

## Especificación del caso de uso

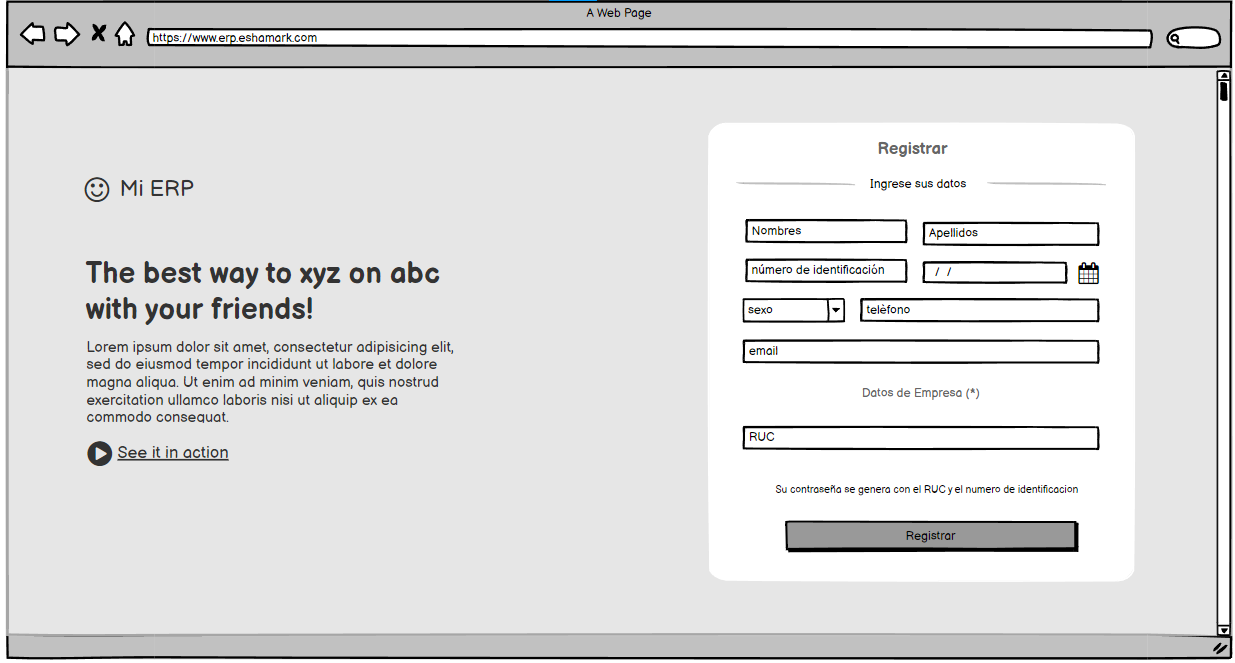
| Código: CUS01 | Registrar Cliente |
| --- | --- |
| Actores: | Cliente |
| Descripción: | El cliente ingresa sus datos dentro de la plataforma y se registra. |
| Precondiciones: | Ingresar a la plataforma |
| Eventos del flujo básico: | 1. Cliente ingresa a la plataforma. 2. Cliente selecciona la opción “Registrarse”. 3. Cliente ingresa los datos que se necesiten. 4. Cliente presiona el botón de “Registrar”. |
| Flujos alternativos: | 1. Si los datos no coinciden no se efectuará el registro. |
| Post Condiciones: | Cliente Registrado |

## Modelo de Datos

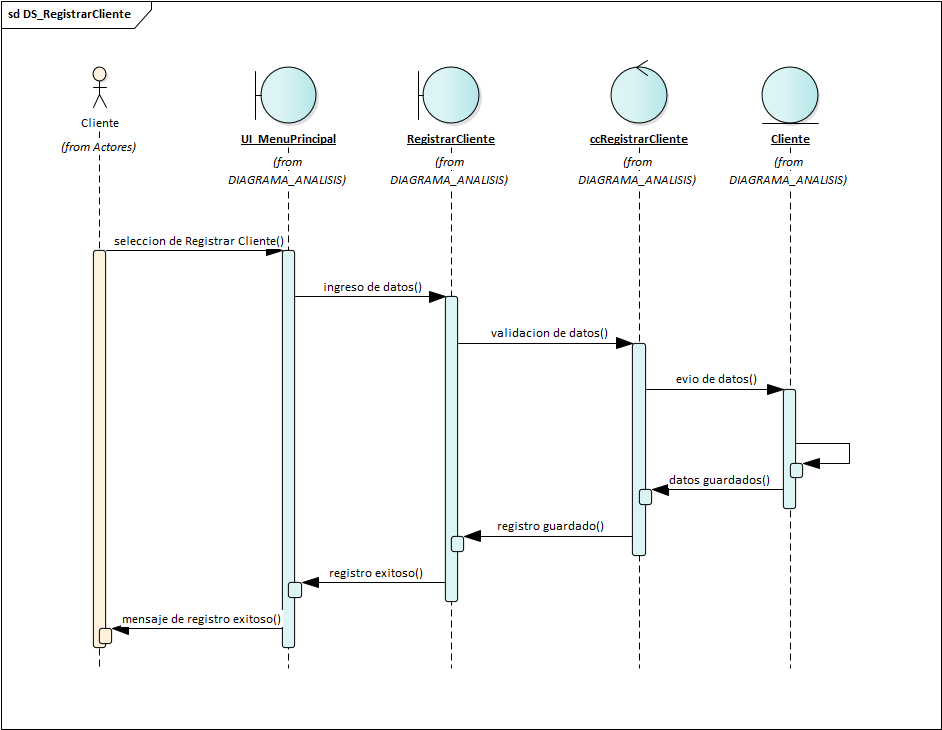
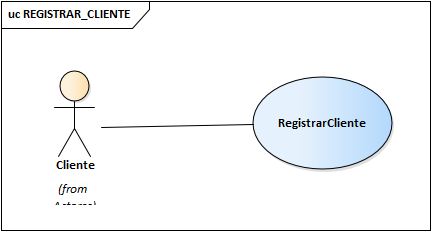
## Análisis de caja negra



## Prototipo



## Diagrama de secuencia



## Código del SP

drop procedure usp\_registrar\_cliente;

go

create procedure usp\_registrar\_cliente

(

@p\_cli\_nombre varchar(30),

@p\_cli\_apellido varchar(30),

@p\_cli\_identificacion char(8),

@p\_cli\_sexo varchar(10),

@p\_cli\_fechanac date,

@p\_cli\_email varchar(50),

@p\_cli\_telefono varchar(50),

@p\_cli\_ruc char(11),

@p\_ciu\_id int,

----------

@p\_codigo char (5) OUT,

@p\_estado int OUT

)

AS

BEGIN

-- Variables

DECLARE @cli\_id int;

-- Validaciones

if(@p\_cli\_apellido is null OR @p\_cli\_nombre is null OR @p\_cli\_identificacion is null)

begin

set @p\_estado = -3;

set @p\_codigo = 'Proceso fue abortado por que no se tienen espacios rellenados';

return;

end;

-- Proceso

SET @p\_estado = 1;

set @p\_codigo = 'Proceso fue realizado';

BEGIN TRY

-- Inicia la Tx

BEGIN TRANSACTION;

-- Generar el nuevo codigo

select @CLI\_ID=MAX(cli\_id) from cliente;

set @cli\_id = @cli\_id+1;

-- Inserta el nuevo cliente

INSERT INTO CLIENTE(cli\_id,cli\_nombre,cli\_apellido,cli\_identificacion,cli\_sexo,cli\_fechanac,cli\_email,cli\_telefono,cli\_ruc,ciu\_Id)

VALUES(@cli\_id,@p\_cli\_nombre,@p\_cli\_apellido,@p\_cli\_identificacion,@p\_cli\_sexo,@p\_cli\_fechanac,@p\_cli\_email,@p\_cli\_telefono,@p\_cli\_ruc,@p\_ciu\_id);

-- Confirma la Tx

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

-- Cancela la Tx

ROLLBACK TRANSACTION;

if(@p\_estado=1) set @p\_estado = -1;

set @p\_codigo = 'Proceso fallido';

END CATCH;

END;

GO

## Pruebas del SP

### Prueba 1

begin

declare @codigo char(5), @estado int;

exec usp\_registrar\_cliente 'Esteban','Bueno','76232221','M','2002-03-01','esteban22@hotmail.com','98009221','576310111',1, @codigo out, @estado out;

print concat('Estado: ', @estado);

print concat('Codigo: ', @codigo);

end;

go

(1 row affected)

Estado: 1

### Prueba 2

begin

declare @codigo char(5), @estado int;

exec usp\_registrar\_cliente 'Emilia','De la Cruz','76890192','F','1997-03-12','emiliasantos@santos.com','988872122','20239911122',1, @codigo out, @estado out;

print concat('Estado: ', @estado);

print concat('Codigo: ', @codigo);

end;

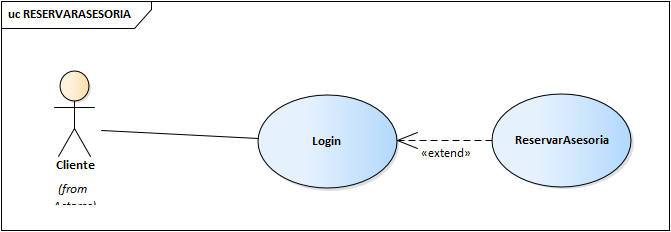
go

(1 row affected)

Estado: 1

# CU002: Reservar Asesoría

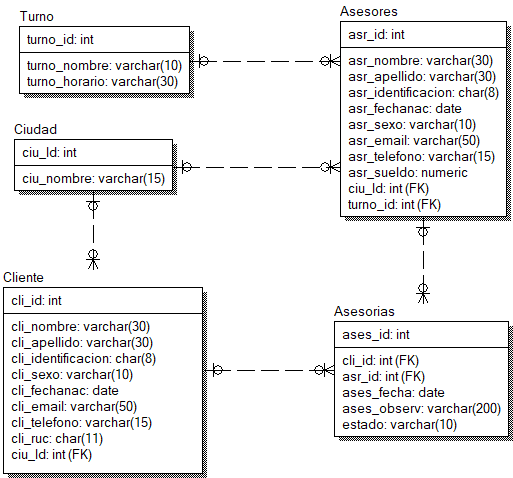
## Diagrama del caso de uso



## Especificación del caso de uso

| Código: CUS02 | Reservar Asesoría |
| --- | --- |
| Actores: | Cliente |
| Descripción: | El cliente reserva una asesoría, seleccionando el horario y asesor e ingresando la fecha disponible. |
| Precondiciones: | El cliente ingresa sus credenciales (RUC y su número de identificación) para poder ingresar a la plataforma. |
| Eventos del flujo básico: | 1. Seleccionar Reservar Asesoría 2. El Cliente visualiza todos los asesores con sus horarios disponibles. 3. El Cliente elige uno de ellos, presionando el botón (+) que se encuentra a su lado. 4. Se muestra una nueva ventana al cliente, en donde éste ingresa la fecha deseada (Día hábil y 2 días de anticipación) 5. El Cliente ingresa alguna observación si es necesario. 6. El Cliente pulsa “Registrar” para registrar su reserva. |
| Flujos alternativos: | 1. Si el Cliente ingresa una fecha que no cumpla con las condiciones, no se realizará la reserva. |
| Post Condiciones: | Asesoría Reservada |

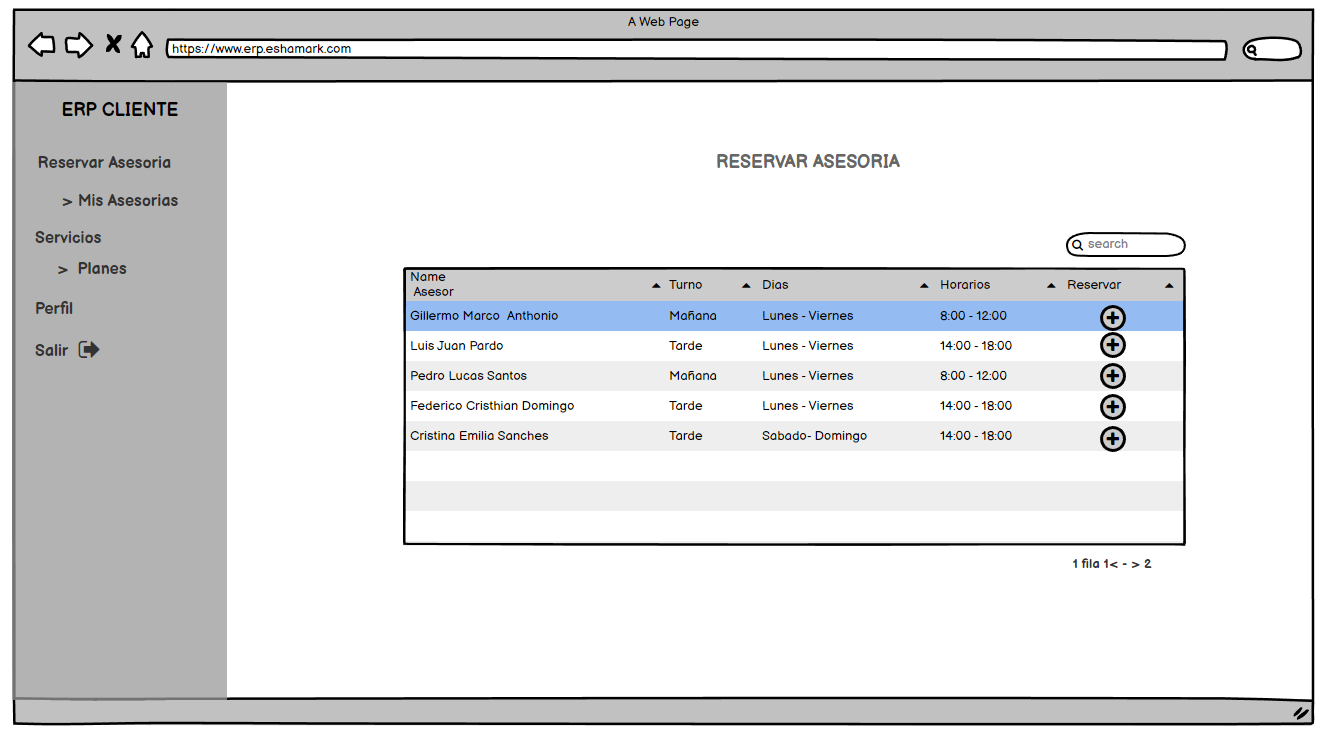
## Modelo de Datos

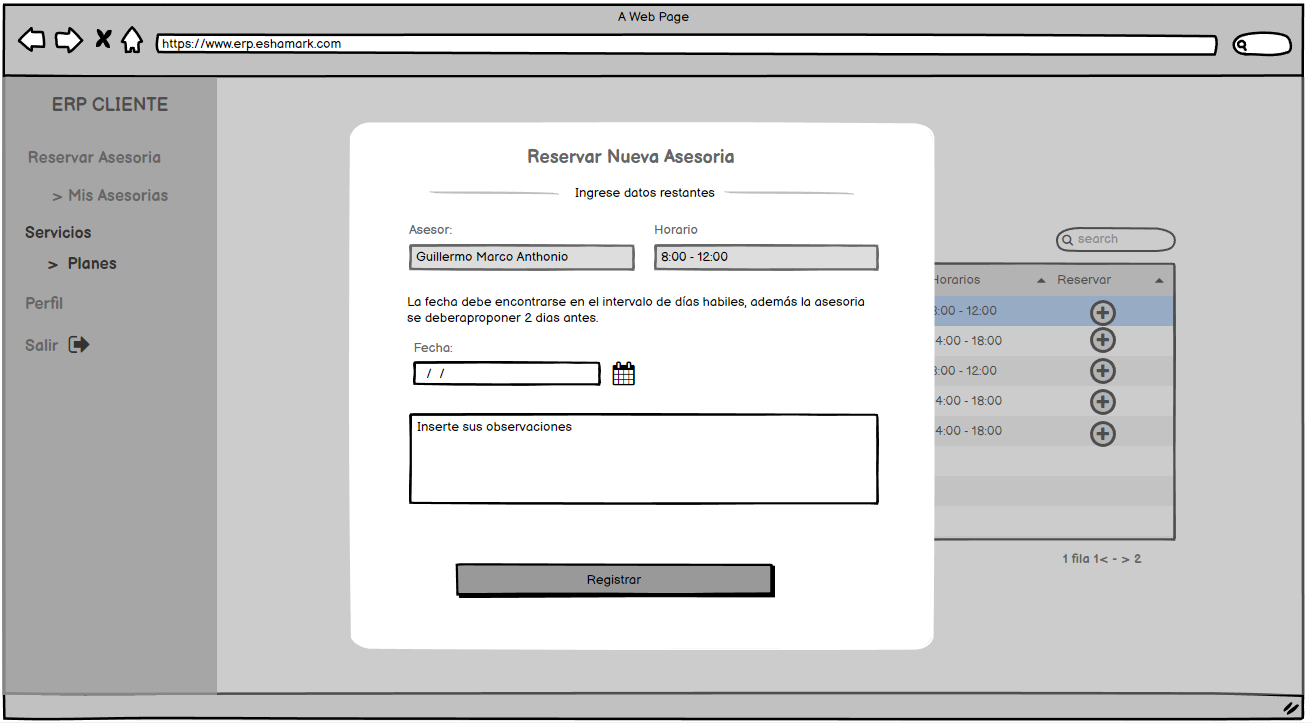


## Análisis de caja negra

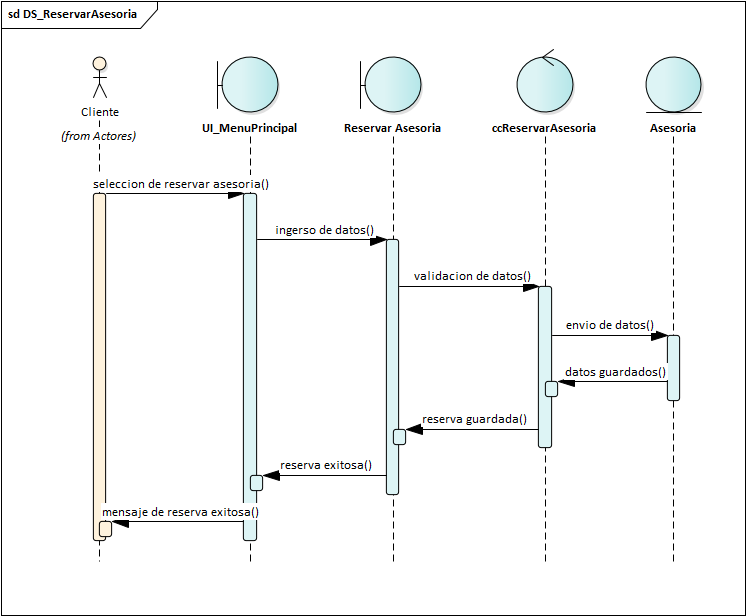


## Prototipo





## Diagrama de secuencia



## Código del SP

drop procedure usp\_reservar\_asesoria;

go

create procedure usp\_reservar\_asesoria

(

@p\_cli\_id int,

@p\_asr\_id int,

@p\_ases\_fecha date,

@p\_ases\_observ varchar(200),

@p\_ases\_estado varchar(10),

----------

@p\_codigo char (50) OUT,

@p\_estado int OUT

)

AS

BEGIN

-- Variables

DECLARE @ases\_id int;

-- Validaciones

if(@p\_cli\_id is null OR @p\_asr\_id is null OR @p\_ases\_estado is null)

begin

set @p\_estado = -3;

set @p\_codigo = 'Proceso fue abortado por que no se tienen espacios rellenados';

return;

end;

-- Proceso

SET @p\_estado = 1;

set @p\_codigo = 'Proceso fue realizado';

BEGIN TRY

-- Inicia la Tx

BEGIN TRANSACTION;

-- Generar el nuevo codigo

select @ases\_id=MAX(ases\_id) from asesorias;

set @ases\_id = @ases\_id+1;

-- Inserta el nuevo cliente

INSERT INTO asesorias(ases\_id,cli\_id,asr\_id,ases\_fecha,ases\_observ,estado)

VALUES(@ases\_id,@p\_cli\_id,@p\_asr\_id,@p\_ases\_fecha,@p\_ases\_observ,@p\_ases\_estado);

-- Confirma la Tx

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

-- Cancela la Tx

ROLLBACK TRANSACTION;

if(@p\_estado=1) set @p\_estado = -1;

set @p\_codigo = 'Proceso fallido';

END CATCH;

END;

GO

## Pruebas del SP

### Prueba 1

begin

declare @codigo char(5), @estado int;

exec usp\_reservar\_asesoria 1,3,'2022-06-10','La reunion se realizara a travez de este zoom: https://meet.google.com/xua-by-iuhg','PENDIENTE', @codigo out, @estado out;

print concat('Estado: ', @estado);

print concat('Codigo: ', @codigo);

end;

go

(1 row affected)

Estado: 1

### Prueba

begin

declare @codigo char(5), @estado int;

exec usp\_reservar\_asesoria 1,1,'2022-04-22','La reunion se realizara a travez de este meet: https://meet.google.com/uid-wme-ldk','ATENDIDO', @codigo out, @estado out;

print concat('Estado: ', @estado);

print concat('Codigo: ', @codigo);

end;

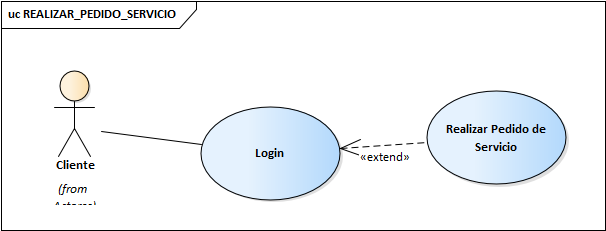
go

(1 row affected)

Estado: 1

# CU003: Realizar Pedido de Servicio

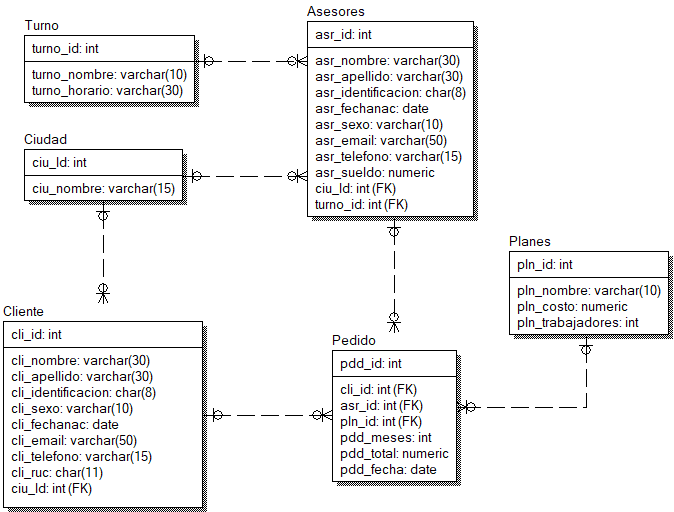
## Diagrama del caso de uso



## Especificación del caso de uso

| Código: CUS03 | Realizar Pedido de Servicio |
| --- | --- |
| Actores: | Cliente |
| Descripción: | El cliente realiza el pedido de un plan. Ingresando al apartado de servicios. |
| Precondiciones: | El cliente debe de haber iniciado sesión. |
| Eventos del flujo básico: | 1. El cliente debe de seleccionar servicios en el menú. 2. El cliente deberá de seleccionar el tipo de plan de servicio para poder proceder. 3. El cliente deberá de ingresar los datos de cantidad de meses y seleccionar el asesor que tendrá su equipo de trabajo. 4. El cliente deberá de seleccionar una pasarela de pago automático. 5. El cliente podrá descargar la factura y tendrá que esperar hasta que activen su pedido y establezcan un contacto con el cliente para poder definir la fecha de inicio. |
| Flujos alternativos: | 1. Si la pasarela rechaza el pago, se elimina la selección. |
| Post Condiciones: | Pedido de servicio realizado |

## Modelo de Datos

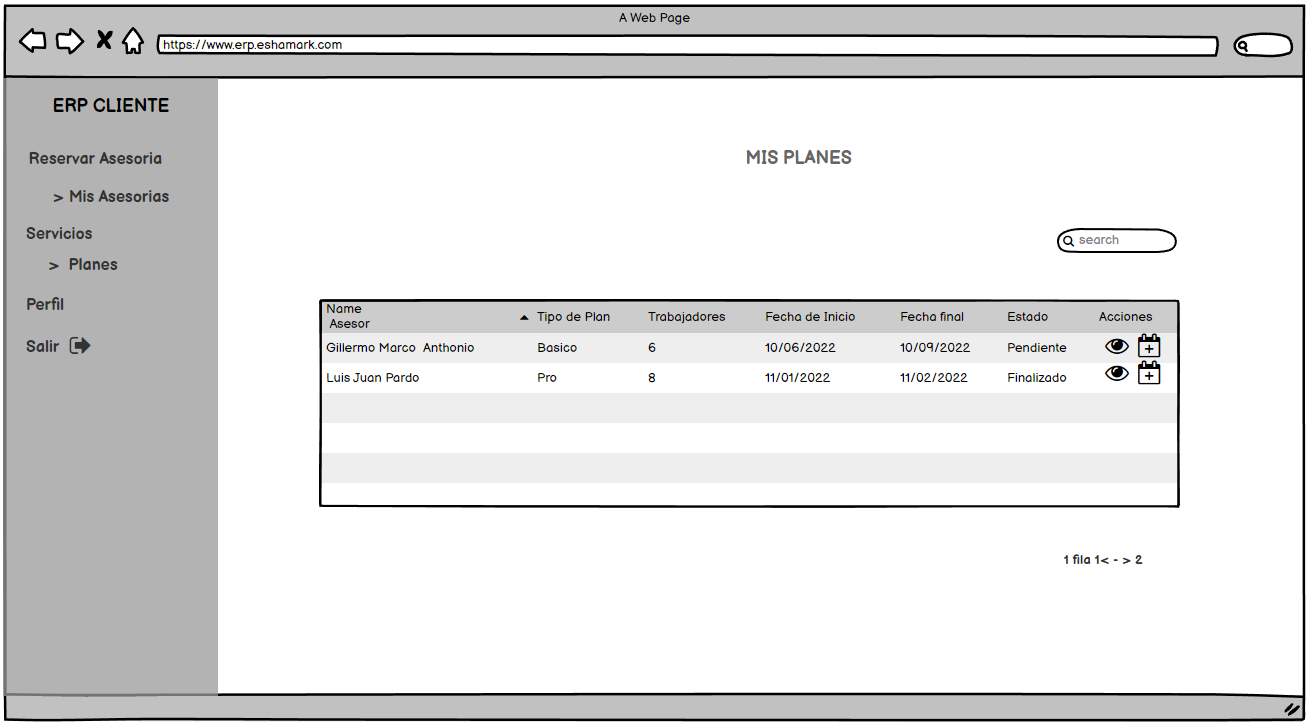
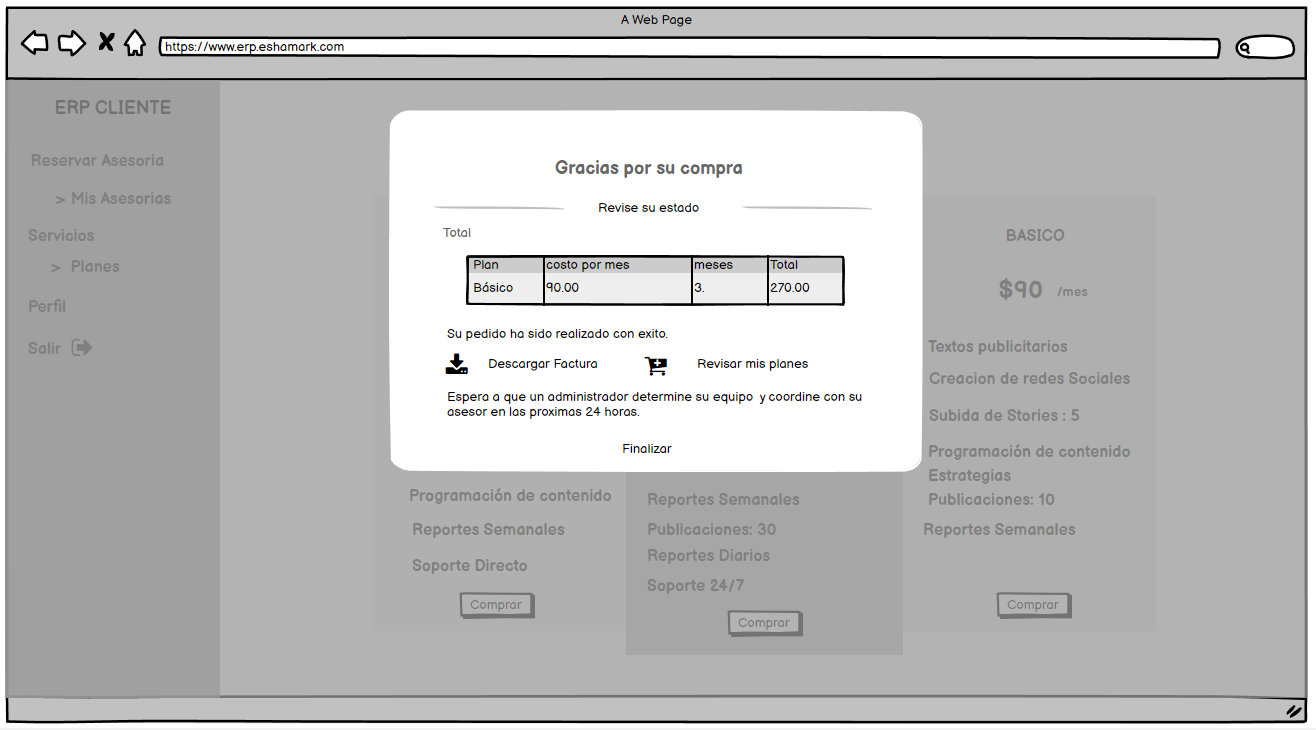
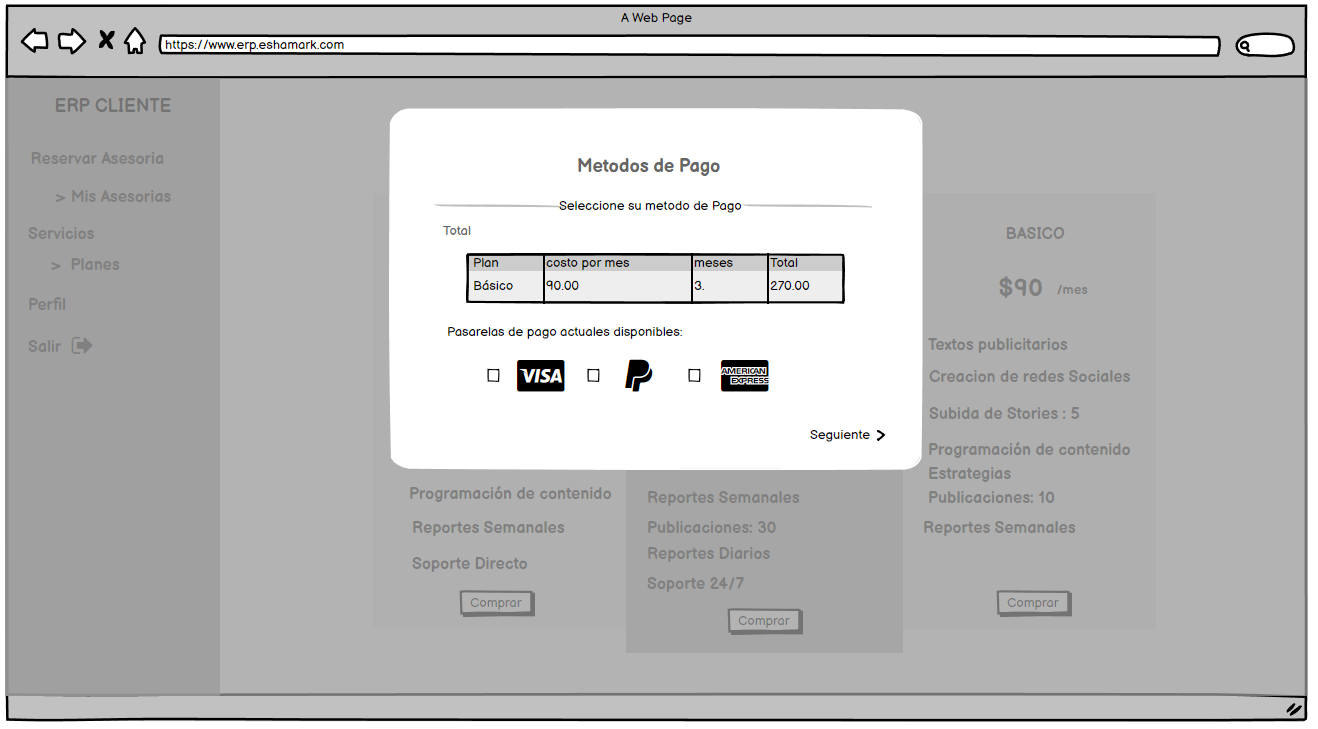
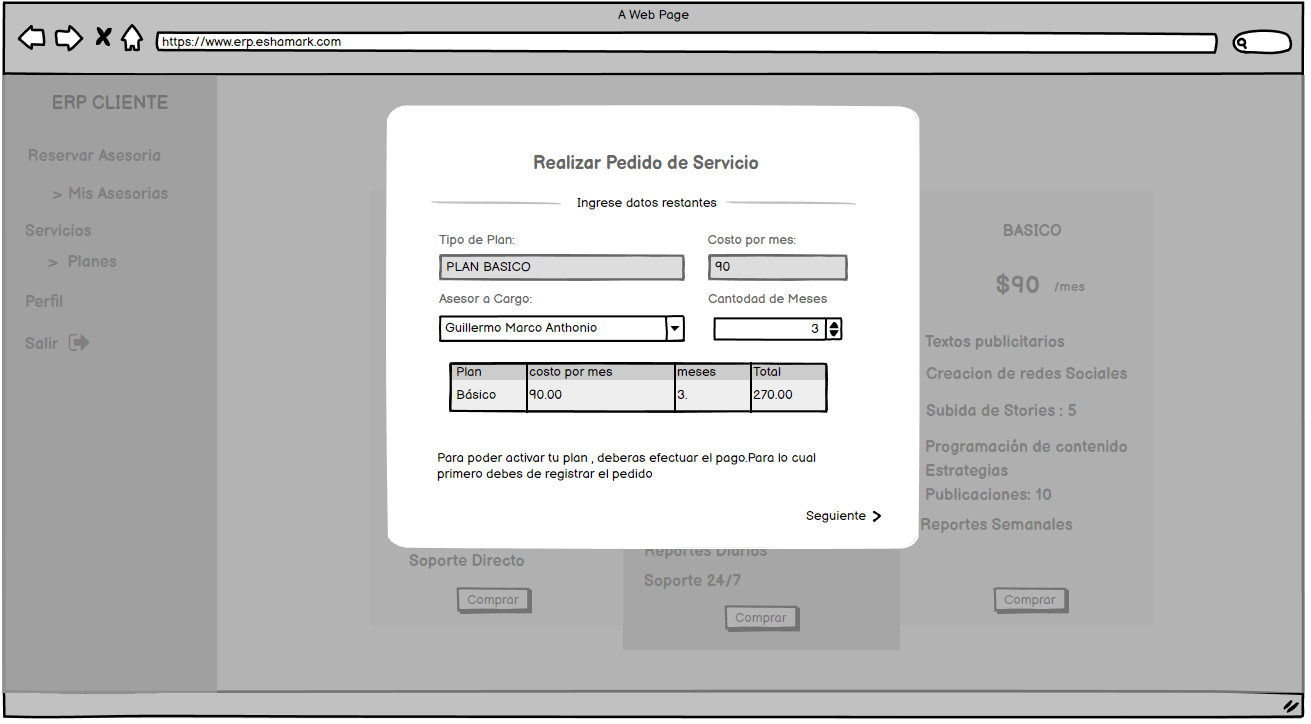


## Análisis de caja negra

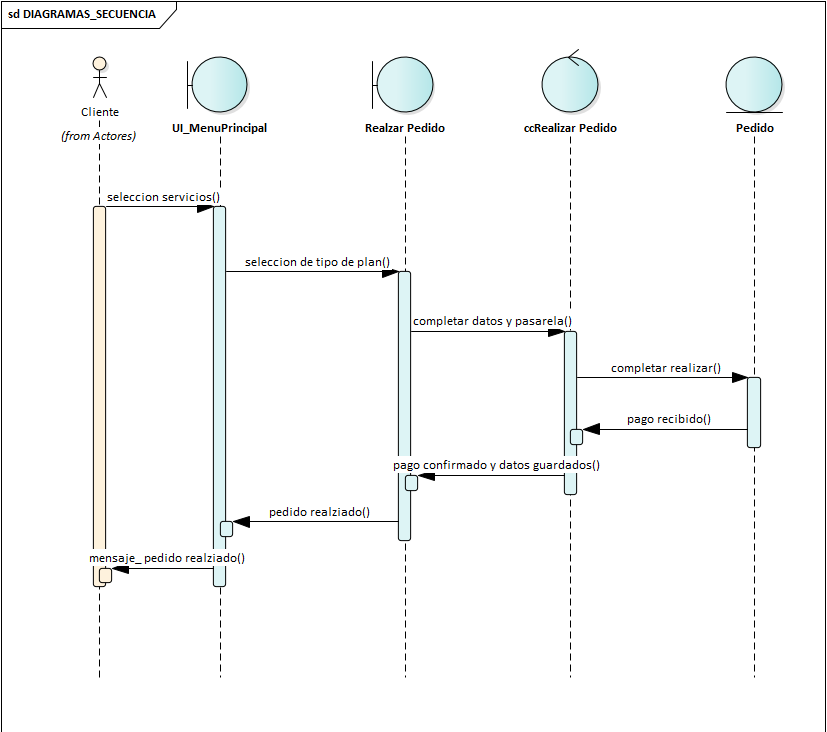


## Prototipo





## Diagrama de secuencia



## Código del SP

create procedure usp\_registrar\_pedido

(

@p\_cli\_id int,

@p\_asr\_id int,

@p\_pln\_id int,

@p\_pdd\_meses int,

@p\_pdd\_fechaini date,

----------

@p\_pdd\_fechafin date out,

@p\_pdd\_total numeric(18,2) OUT,

@p\_msg varchar (100) OUT,

@p\_estado int OUT

)

AS

BEGIN

-- Variables

DECLARE @asr\_id int;

-- Validaciones

if(

@p\_cli\_id is null OR

@p\_asr\_id is null OR

@p\_pln\_id is null OR

@p\_pdd\_meses is null OR

@p\_pdd\_fechaini is null

)

begin

set @p\_estado = -2;

set @p\_msg = 'Proceso fue abortado por que no se tienen espacios rellenados';

return;

end;

-- Proceso

SET @p\_estado = 1;

set @p\_msg = 'Proceso fue realizado';

BEGIN TRY

-- Inicia la Tx

BEGIN TRANSACTION;

-- Generar el nuevo codigo

set @p\_pdd\_total = Cast(@p\_pdd\_meses as numeric(18,2)) \* (Select p.pln\_costo from Planes p where p.pln\_id = @p\_pln\_id)

set @p\_pdd\_fechafin = (Select DATEADD(MONTH,@p\_pdd\_meses,@p\_pdd\_fechaini))

-- Inserta el nuevo cliente

INSERT INTO Pedido VALUES(@p\_cli\_id,@p\_asr\_id,@p\_pln\_id,@p\_pdd\_meses,@p\_pdd\_total,@p\_pdd\_fechaini,@p\_pdd\_fechafin);

-- Confirma la Tx

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

-- Cancela la Tx

ROLLBACK TRANSACTION;

if(@p\_estado=1) set @p\_estado = -1;

set @p\_msg = 'Proceso fallido';

END CATCH;

END;

GO

## Pruebas del SP

### Prueba 1

begin

declare @fechaini date = (select GETDATE()),@fechafin date, @total numeric(18,2),@msg varchar(100), @estado int;

exec usp\_registrar\_pedido 1,1,1,1,@fechaini,@fechafin out, @total out,@msg out, @estado out;

print concat('Estado: ', @estado);

print concat('Mensaje: ', @msg);

print concat('Total: ',@total);

print concat('Fecha final: ',@fechafin)

end;

go

(1 row affected)

Estado: 1

Mensaje: Proceso fue realizado

Total: 90.00

Fecha final: 2022-07-29

Completion time: 2022-06-29T09:47:37.4704734-05:00

### Prueba 2

begin

declare @fechaini date = (select GETDATE()),@fechafin date, @total numeric(18,2),@msg varchar(100), @estado int;

exec usp\_registrar\_pedido 2,1,2,1,@fechaini,@fechafin out, @total out,@msg out, @estado out;

print concat('Estado: ', @estado);

print concat('Mensaje: ', @msg);

print concat('Total: ',@total);

print concat('Fecha final: ',@fechafin)

end;

go

(1 row affected)

Estado: 1

Mensaje: Proceso fue realizado

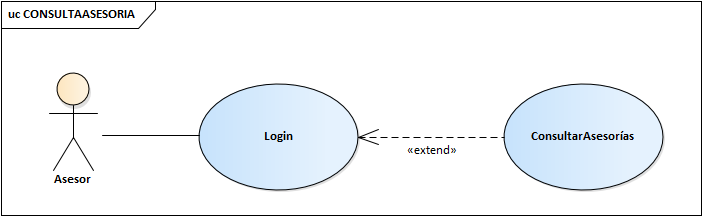
Total: 130.00

Fecha final: 2022-07-29

Completion time: 2022-06-29T09:51:42.3308504-05:00

# CU004: Consultar Asesorías

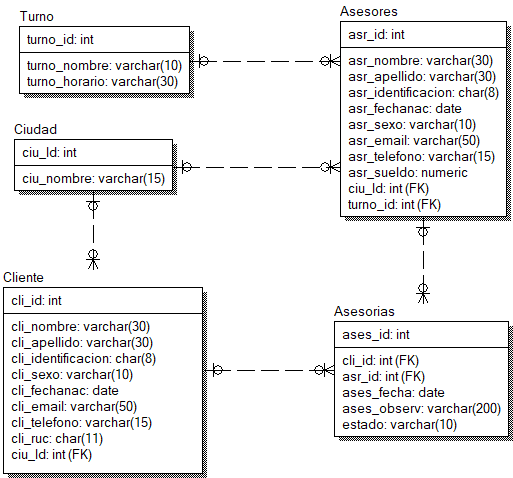
## Diagrama del caso de uso



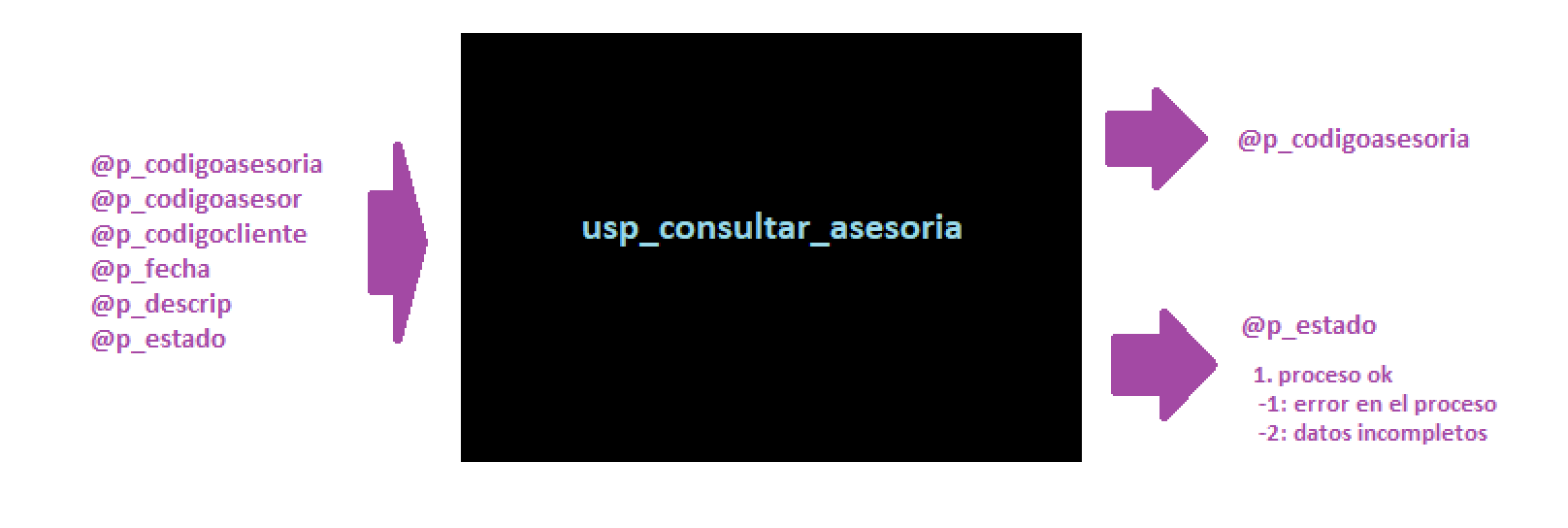
## Especificación del caso de uso

| Código: CUS04 | Consultar Asesorías |
| --- | --- |
| Actores: | Asesor |
| Descripción: | El asesor puede acceder a todas las asesorías que ha ingresado y consultar una en concreto, seleccionando el horario e ingresando la fecha disponible. |
| Precondiciones: | El asesor ingresa sus credenciales (RUC y su número de identificación) para poder ingresar a la plataforma. |
| Eventos del flujo básico: | 1. Seleccionar Consultar Asesoría 2. Asesor visualiza las asesorías realizadas a su nombre 3. Asesor ingresa alguna descripción de la reunión 4. Asesor elige entre las acciones “visualizar los detalles o eliminar la asesoría ” |
| Flujos alternativos: | 1. Si se cancela la asesoría, no se va a poder realizar. |
| Post Condiciones: | Asesoría Consultada |

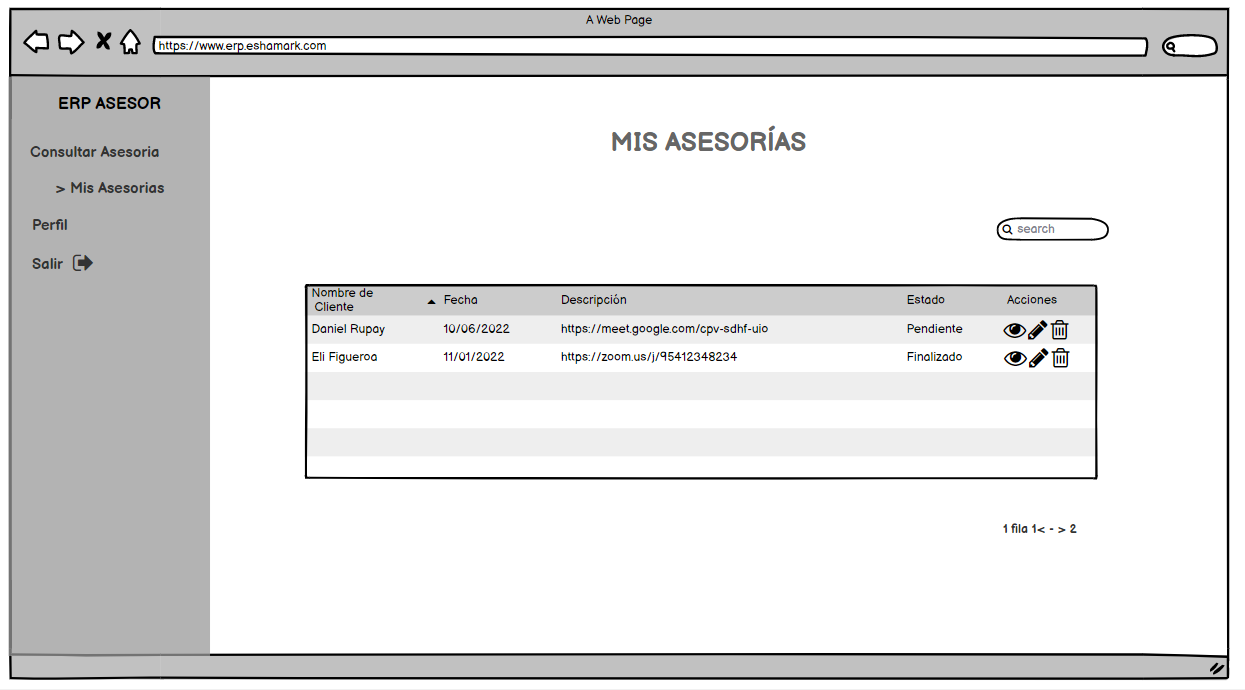
## Modelo de Datos

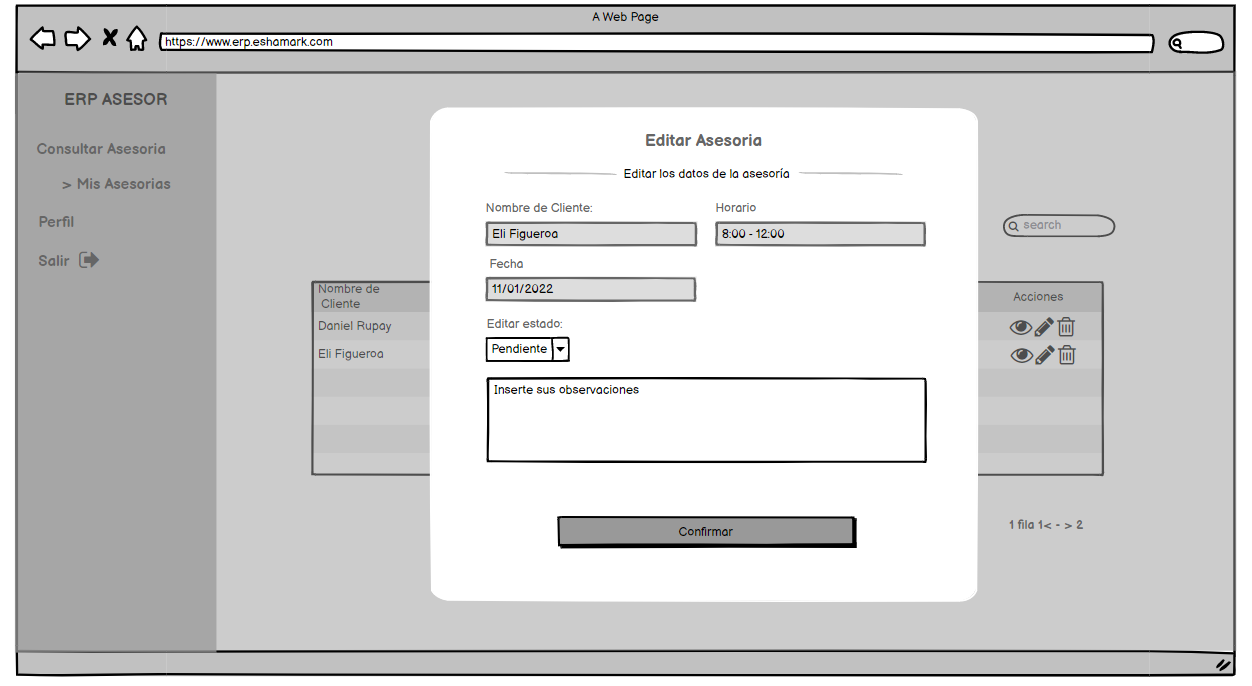


## Análisis de caja negra

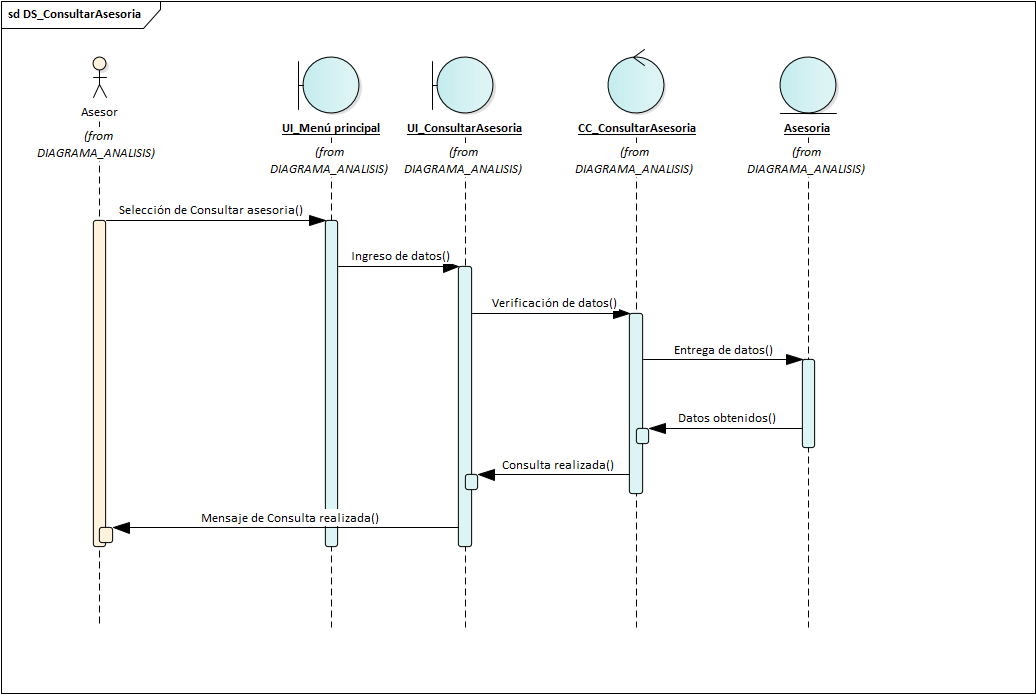


## Prototipo





## Diagrama de secuencia



## Código del SP

create procedure usp\_consultar\_asesores

(

@p\_mensaje varchar(100) OUT,

@p\_estado int OUT

)

AS

BEGIN

-- Variables

DECLARE @msg\_error varchar(50);

-- Proceso

if(@p\_estado is NULL)

BEGIN

SET @p\_estado = 1;

END

SET @p\_estado = 1;

BEGIN TRY

-- Inicia la Tx

BEGIN TRANSACTION;

-- Llenar las variables

select a.asr\_nombre,a.asr\_apepaterno,a.asr\_apematerno,t.turno\_nombre,t.turno\_horario from Asesores a

inner join Turno t ON t.turno\_id = a.turno\_id

set @p\_mensaje = 'Consulta realizada exitosamente.'

-- Confirma la Tx

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

-- Cancela la Tx

ROLLBACK TRANSACTION;

set @p\_estado = -1;

END CATCH;

END;

GO

## Pruebas del SP

### Prueba 1

declare @estado int, @mensaje varchar(100)

exec usp\_consultar\_asesores @mensaje out,@estado out;

print concat('Estado: ', @estado);

print concat('Mensaje: ', @mensaje);

go

(3 rows affected)

Estado: 1

Mensaje: Consulta realizada exitosamente.

Completion time: 2022-06-29T10:08:41.3127230-05:00

### Prueba 2

declare @estado int, @mensaje varchar(100)

exec usp\_consultar\_asesores @mensaje out,@estado out;

print concat('Estado: ', @estado);

print concat('Mensaje: ', @mensaje);

go

(3 rows affected)

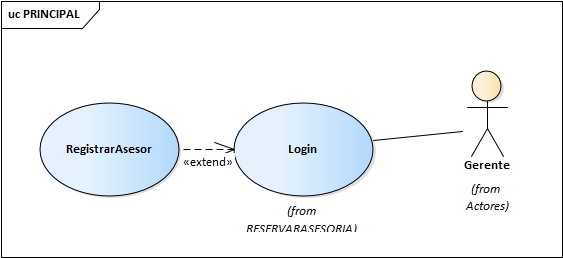
Estado: 1

Mensaje: Consulta realizada exitosamente.

Completion time: 2022-06-29T10:08:41.3127230-05:00

# CU005: Registrar Asesor

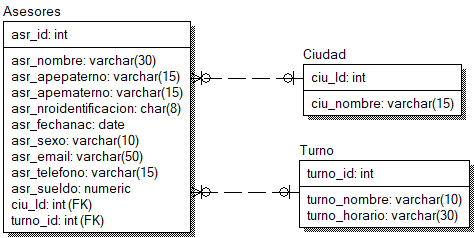
## Diagrama del caso de uso



## Especificación del caso de uso

| Código: CUS05 | Registrar Asesor |
| --- | --- |
| Actores: | Gerente |
| Descripción: | El gerente registra a un nuevo asesor, |
| Precondiciones: | El gerente debe haber iniciado sesión en la plataforma |
| Eventos del flujo básico: | 1. El gerente ingresa a la interfaz de asesores 2. El gerente presiona el botón de “registrar asesor” 3. El gerente rellena los datos del nuevo asesor 4. El gerente pulsa en resitrar para grabar. |
| Flujos alternativos: | 1. Los datos deben de estar completos. |
| Post Condiciones: | Asesor Registrado |

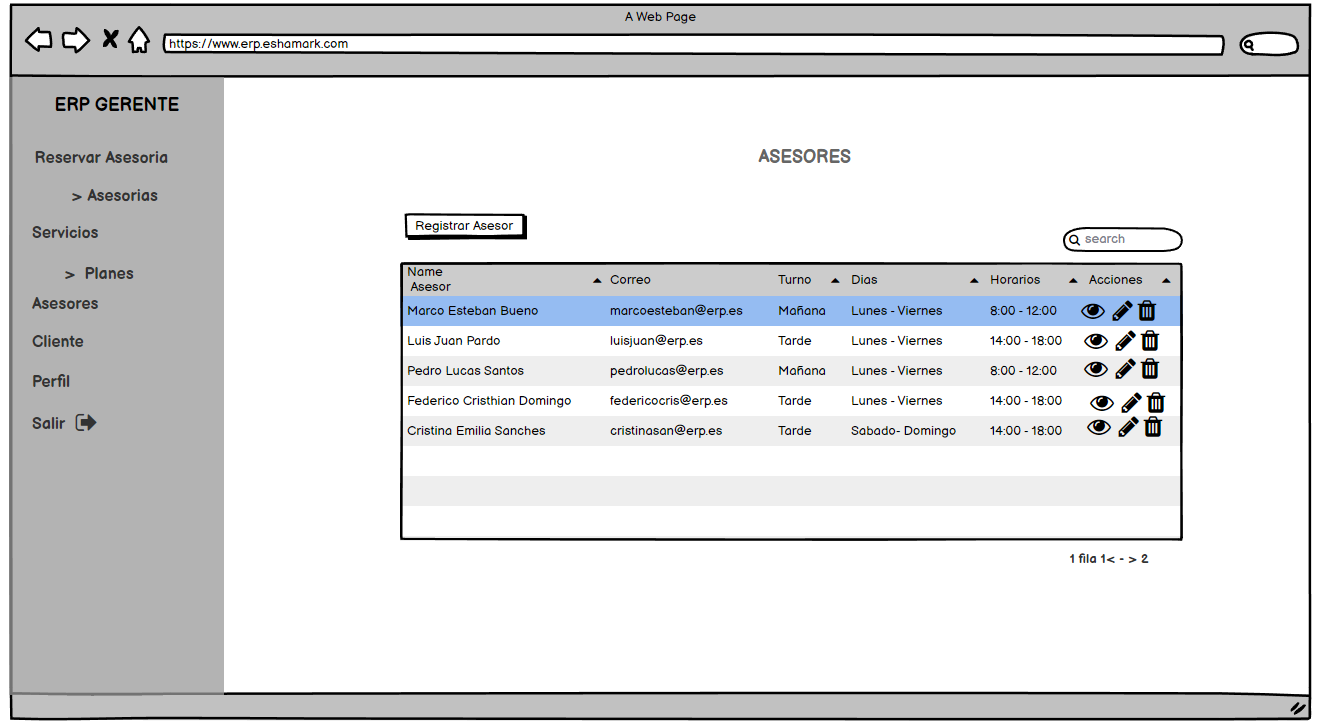
## Modelo de Datos

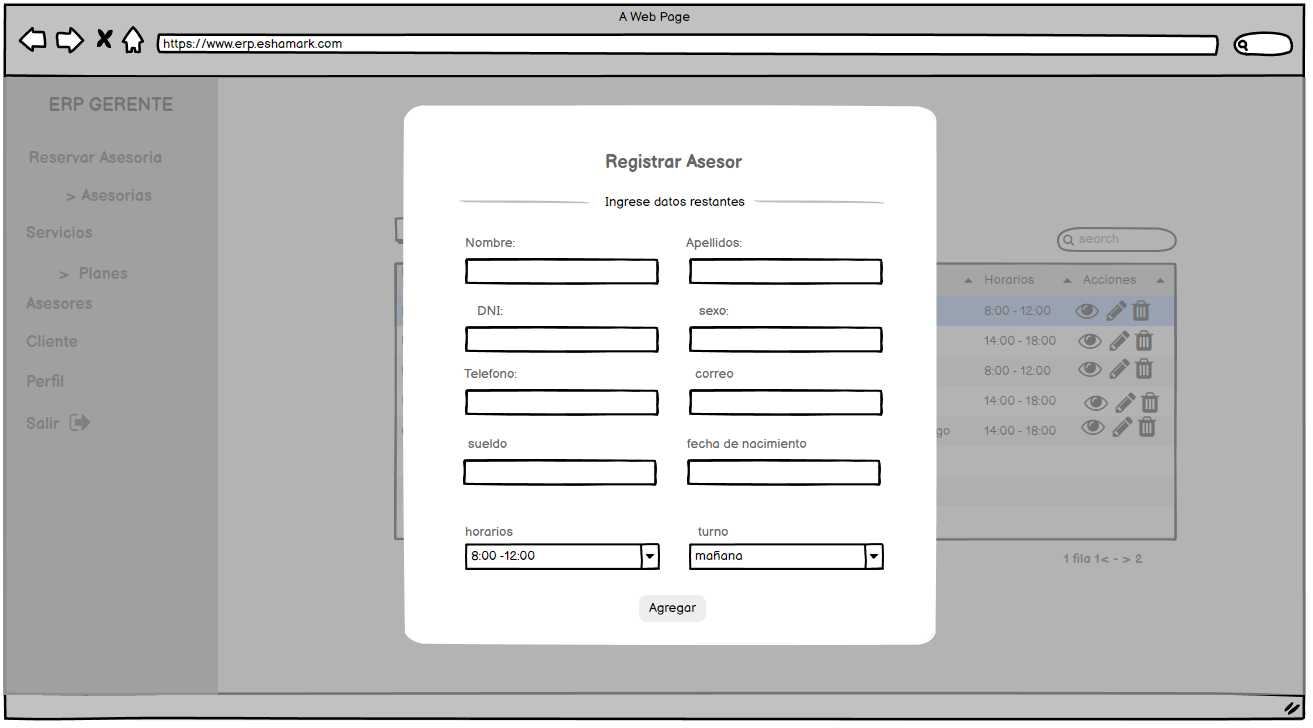


## Análisis de caja negra

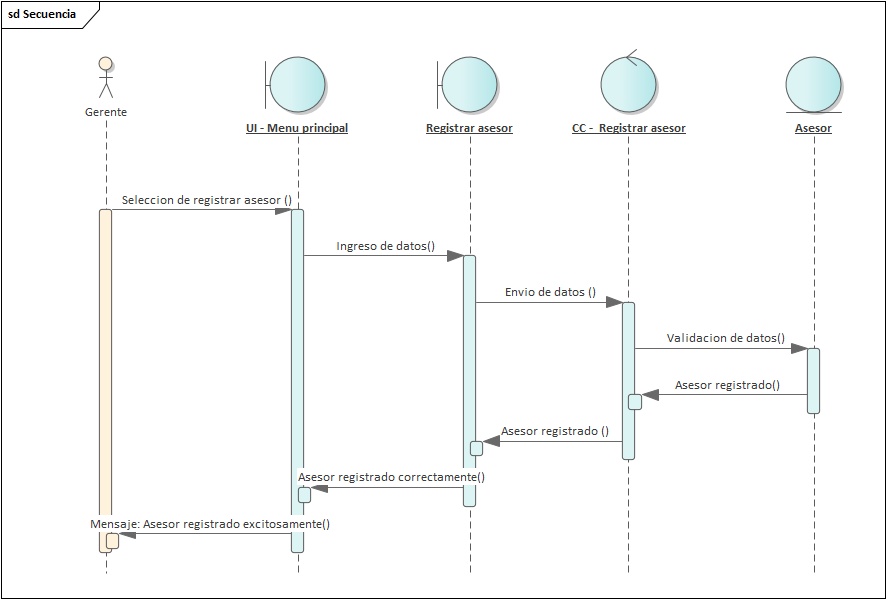


## Prototipo





## Diagrama de secuencia



## Código del SP

drop procedure usp\_registrar\_asesor;

go

create procedure usp\_registrar\_asesor

(

@p\_asr\_nombre varchar(30),

@p\_asr\_apellido varchar(30),

@p\_asr\_identificacion char(8),

@p\_asr\_fechanac date,

@p\_asr\_sexo varchar(10),

@p\_asr\_email varchar(50),

@p\_asr\_telefono varchar(15),

@p\_asr\_sueldo numeric,

@p\_ciu\_id int,

@p\_turno\_id int,

----------

@p\_codigo char (50) OUT,

@p\_estado int OUT

)

AS

BEGIN

-- Variables

DECLARE @asr\_id int;

-- Validaciones

if(@p\_asr\_apellido is null OR @p\_asr\_nombre is null OR @p\_asr\_identificacion is null)

begin

set @p\_estado = -3;

set @p\_codigo = 'Proceso fue abortado por que no se tienen espacios rellenados';

return;

end;

-- Proceso

SET @p\_estado = 1;

set @p\_codigo = 'Proceso fue realizado';

BEGIN TRY

-- Inicia la Tx

BEGIN TRANSACTION;

-- Generar el nuevo codigo

select @asr\_id=MAX(asr\_id) from asesores;

set @asr\_id = @asr\_id+1;

-- Inserta el nuevo cliente

INSERT INTO asesores(asr\_id,asr\_nombre,asr\_apellido,asr\_identificacion,asr\_fechanac,asr\_sexo,asr\_email,asr\_telefono,asr\_sueldo,ciu\_Id,turno\_id)

VALUES(@asr\_id,@p\_asr\_nombre,@p\_asr\_apellido,@p\_asr\_identificacion,@p\_asr\_fechanac,@p\_asr\_sexo,@p\_asr\_email,@p\_asr\_telefono,@p\_asr\_sueldo,@p\_ciu\_id,@p\_turno\_id);

-- Confirma la Tx

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

-- Cancela la Tx

ROLLBACK TRANSACTION;

if(@p\_estado=1) set @p\_estado = -1;

set @p\_codigo = 'Proceso fallido';

END CATCH;

END;

GO

## Pruebas del SP

### Prueba 1

begin

declare @codigo char(5), @estado int;

exec usp\_registrar\_asesor 'Arturo','Millan','20668712','1989-01-12','M','arturomillan@hotmail.com','987678564','1700',3,2, @codigo out, @estado out;

print concat('Estado: ', @estado);

print concat('Codigo: ', @codigo);

end;

go

(1 row affected)

Estado: 1

### Prueba 2

begin

declare @codigo char(5), @estado int;

exec usp\_registrar\_asesor 'Anthonela','Ramirez','68271113','1999-11-10','F','mantillaaa@hotmail.com','9879876111','1800',3,2, @codigo out, @estado out;

print concat('Estado: ', @estado);

print concat('Codigo: ', @codigo);

end;

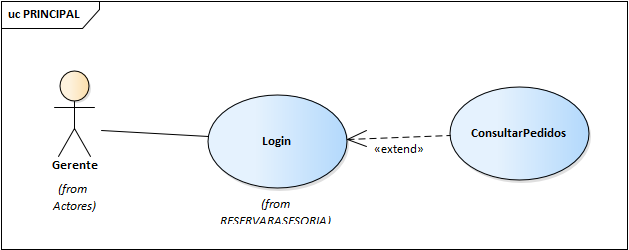
go

(1 row affected)

Estado: 1

# CU006: Consultar Pedido

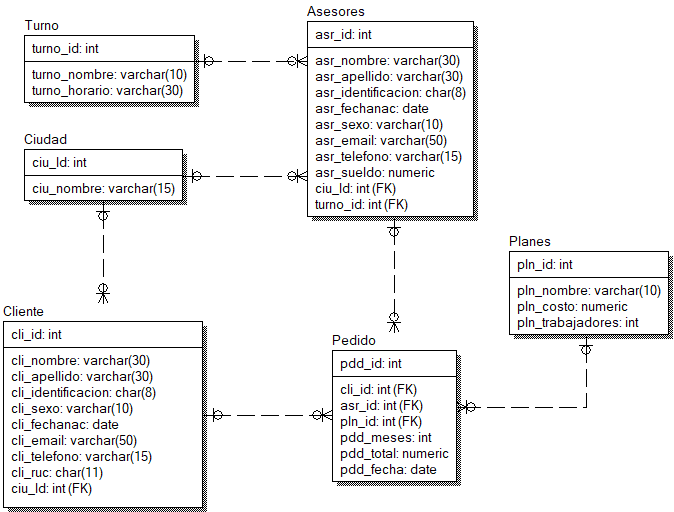
## Diagrama del caso de uso



## Especificación del caso de uso

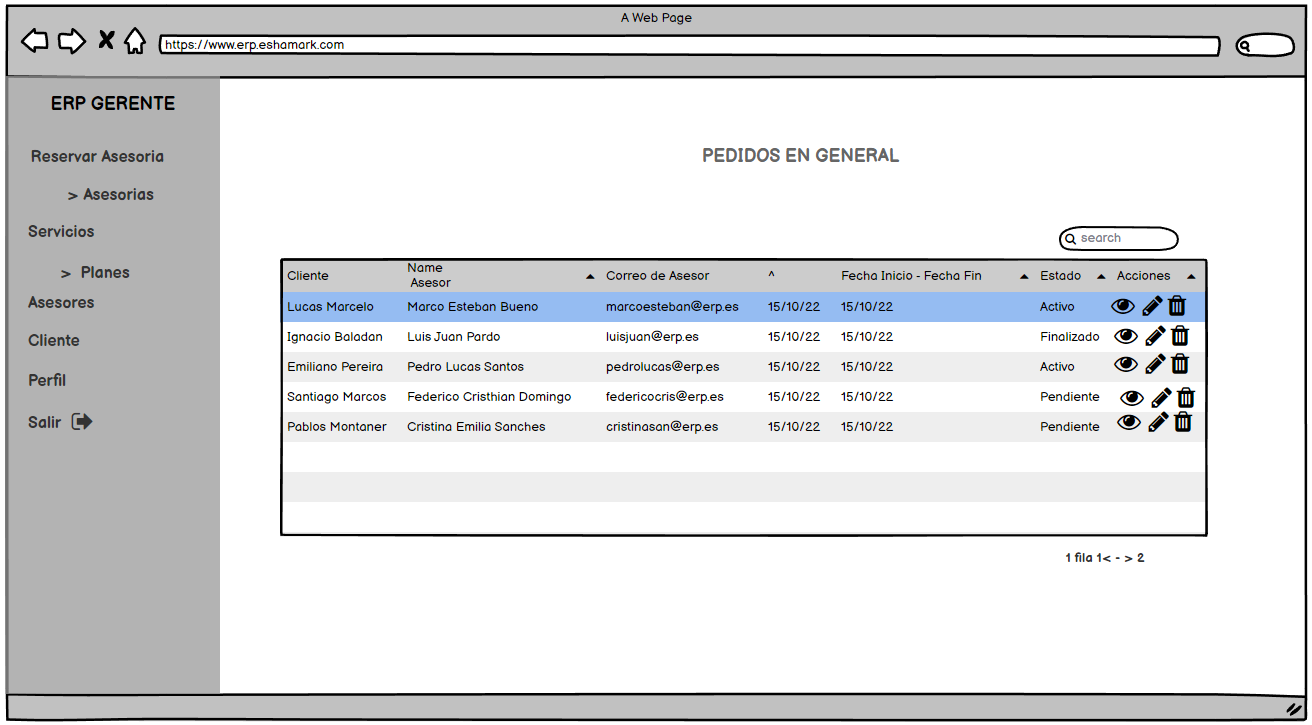
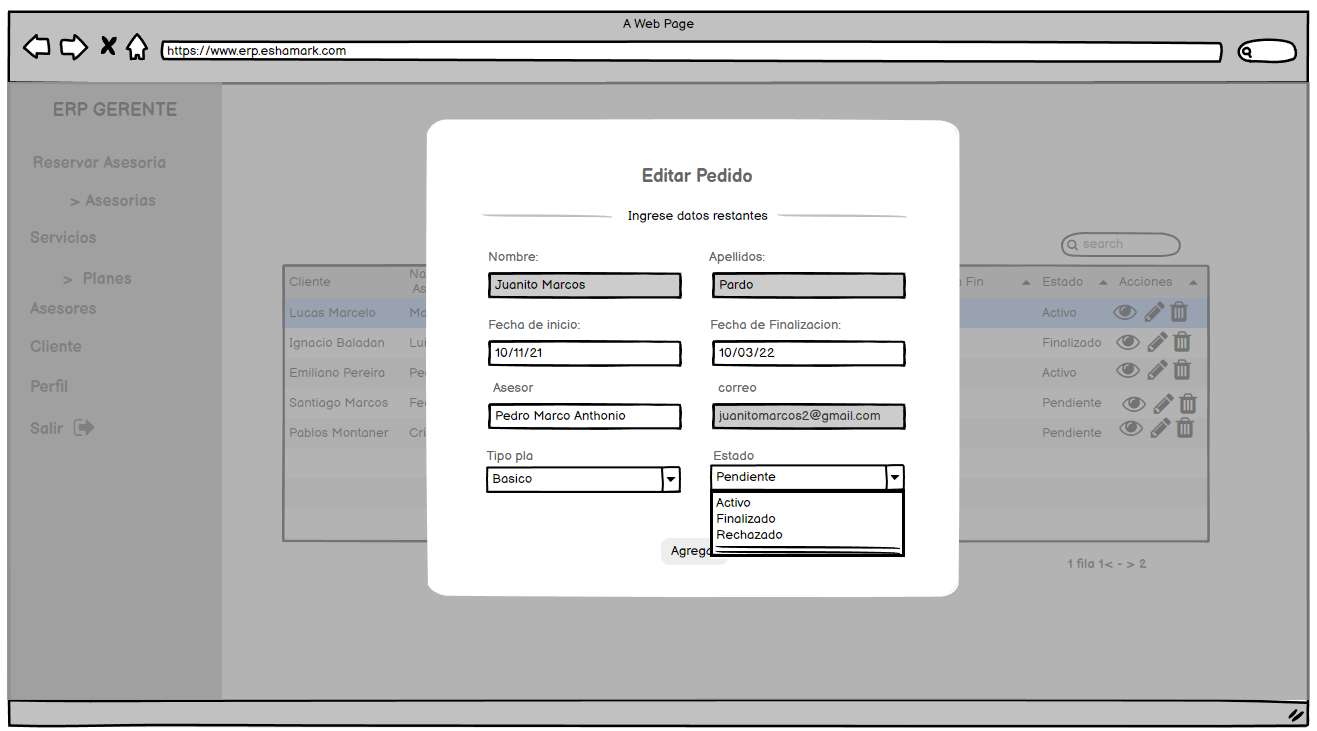
| Código: CUS06 | Consultar Pedido |
| --- | --- |
| Actores: | Gerente |
| Descripción: | El gerente consulta o lista cualquier tipo de pedido realizado dentro del sistema. |
| Precondiciones: | El gerente deberá de ingresar a la plataforma. |
| Eventos del flujo básico: | 1. El gerente ingresa al apartado de pedidos 2. El gerente puede visualizar todos los pedidos de planes 3. El gerente prodra manipular el estado del pedido. 4. El asesor manipula la fecha de inicio y la fecha de fin de acuerdo a lo que pide el cliente. |
| Flujos alternativos: | 1. Se debe de tener al menos un pedido realizado. |
| Post Condiciones: | Pedido consultado y modificado |

## Modelo de Datos

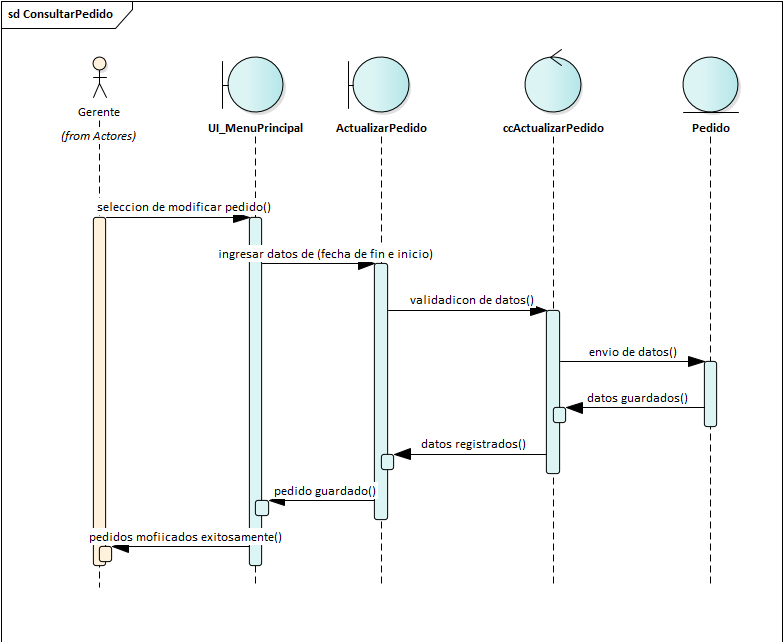


## Análisis de caja negra

## Prototipo

## Diagrama de secuencia



## Código del SP

create procedure usp\_consultar\_pedidos

(

@codigo int,

@p\_mensaje varchar(100) OUT,

@p\_estado int OUT

)

AS

BEGIN

-- Variables

DECLARE @msg\_error varchar(50);

-- Proceso

if(@p\_estado is NULL)

BEGIN

SET @p\_estado = 1;

END

SET @p\_estado = 1;

BEGIN TRY

-- Inicia la Tx

BEGIN TRANSACTION;

-- Llenar las variables

select \* from Pedido where pdd\_id=@codigo;

set @p\_mensaje = 'Consulta realizada exitosamente.'

-- Confirma la Tx

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

-- Cancela la Tx

ROLLBACK TRANSACTION;

set @p\_estado = -1;

END CATCH;

END;

GO

## Pruebas del SP

### Prueba 1

begin;

declare @estado int, @mensaje varchar(100)

exec usp\_consultar\_pedidos 2,@mensaje out,@estado out;

print concat('Estado: ', @estado);

print concat('Mensaje: ', @mensaje);

end;

go

(1 row affected)

Estado: 1

Mensaje: Consulta realizada exitosamente.

### Prueba 2

begin;

declare @estado int, @mensaje varchar(100)

exec usp\_consultar\_pedidos 1,@mensaje out,@estado out;

print concat('Estado: ', @estado);

print concat('Mensaje: ', @mensaje);

end;

go

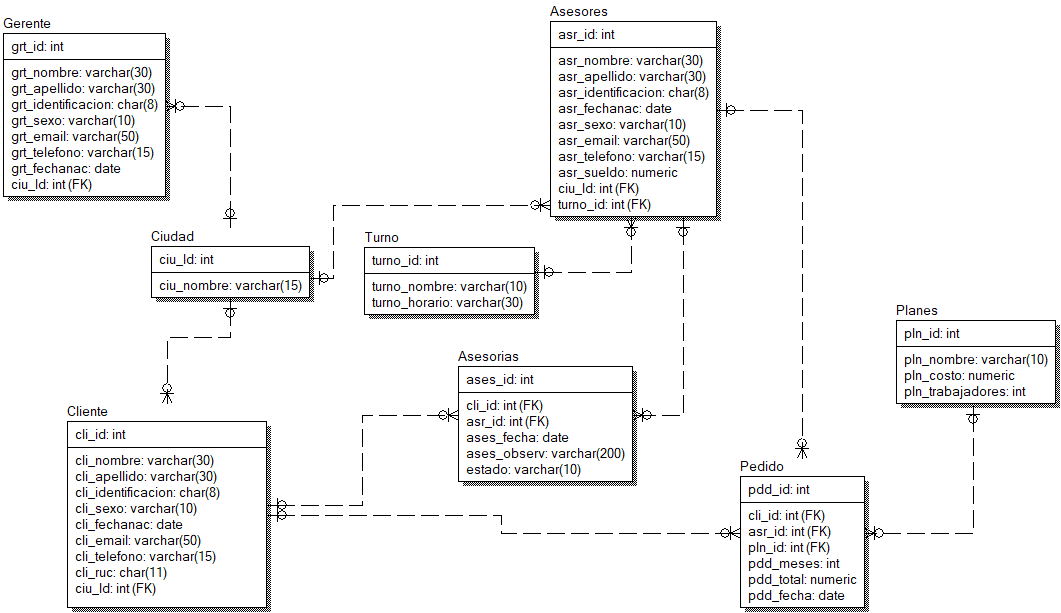
(1 row affected)

Estado: 1

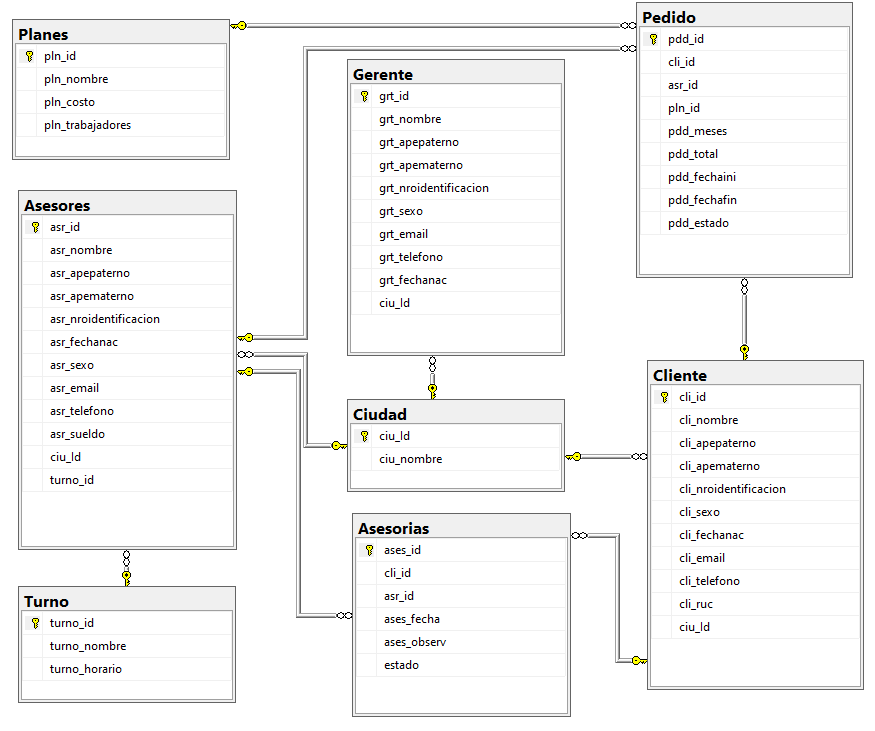
Mensaje: Consulta realizada exitosamente.

# **7. MODELO FÍSICO**

## 7.1. Modelo en software erwin



## 7.2. modelo en sql server

****

# **DICCIONARIO DE DATOS**

### Tabla Gerente

| **Campo** | **Tamaño** | **Tipo de Dato** | **Descripción** | | |
| --- | --- | --- | --- | --- | --- |
| grt\_id |  | INT | Identificador del Gerente | | |
| grt\_nombre | 30 | VARCHAR | Nombres del Gerente | | |
| grt\_apepaterno | 15 | VARCHAR | Apellido paterno del Gerente | | |
| grt\_apematerno | 15 | VARCHAR | Apellido materno del Gerente | | |
| grt\_identificación | 8 | CHAR | Documento de identidad del Gerente | | |
| grt\_sexo | 10 | VARCHAR | Sexo del Gerente | | |
| grt\_email | 50 | VARCHAR | Email del Gerente | | |
| grt\_telefono | 15 | VARCHAR | Teléfono del Gerente | | |
| grt\_fechanac |  | DATE | Fecha de nacimiento del Gerente | | |
| ciu\_Id |  | INT | Identificador de la Ciudad | | |
|  |  |  |  |  |  |
| **Relaciones**: | | | **Campos Clave:** | |  |
| TablaGerente con TablaCiudad | |  | grt\_id, ciu\_id, grt\_nombre, grt\_apellido | | |

### Tabla Ciudad

| **Campo** | **Tamaño** | **Tipo de Dato** | **Descripción** | | |
| --- | --- | --- | --- | --- | --- |
| ciu\_id |  | INT | Identificador de la Ciudad | | |
| ciu\_nombre | 15 | VARCHAR | Nombre de la Ciudad | | |
|  |  |  |  |  |  |
| **Relaciones**: | | | **Campos** **Clave**: | |  |
| TablaCiudad con TablaCliente | |  | ciu\_id | | |
| TablaCiudad con TablaAsesores | |  |  |  |  |
| TablaCiudad con TablaCliente | |  |  |  |  |

1. **Tabla Turno**

| **Campo** | **Tamaño** | **Tipo de Dato** | **Descripción** | | |
| --- | --- | --- | --- | --- | --- |
| turno\_id |  | INT | Identificador del Turno | | |
| turno\_nombre | 10 | VARCHAR | Nombre del Turno | | |
| turno\_horario | 30 | VARCHAR | Horario del Turno | | |
|  |  |  |  |  |  |
| **Relaciones**: | | | **Campos** **Clave**: | |  |
| TablaTurno con TablaAsesores | |  | turno\_id | | |
|  | |  |  |  |  |
|  | |  |  |  |  |

1. **Tabla Cliente**

| **Campo** | **Tamaño** | **Tipo de Dato** | **Descripción** | | |
| --- | --- | --- | --- | --- | --- |
| cli\_id |  | INT | Identificador del Cliente | | |
| cli\_nombre | 30 | VARCHAR | Nombres del Cliente | | |
| cli\_apellido | 15 | VARCHAR | Apellidos del Cliente | | |
| cli\_identificación | 8 | CHAR | Documento de identidad del Cliente | | |
| cli\_sexo | 10 | VARCHAR | Sexo del Cliente | | |
| cli\_email | 50 | VARCHAR | Email del Cliente | | |
| cli\_telefono | 15 | VARCHAR | Telefono del Cliente | | |
| cli\_fechanac |  | DATE | Fecha de nacimiento del Cliente | | |
| cli\_ruc | 11 | CHAR | Ruc del Cliente | | |
| ciu\_Id |  | INT | Identificador de la Ciudad | | |
|  |  |  |  |  |  |
| **Relaciones**: | | | **Campos Clave:** | |  |
| TablaCliente con TablaCiudad | |  | cli\_id, ciu\_id, cli\_nombre, cli\_apellido | | |
| TablaCliente con TablaAsesorias | |  |  |  |  |
| TablaCliente con TablaPedido | |  |  |  |  |

1. **Tabla Asesores**

| **Campo** | **Tamaño** | **Tipo de Dato** | **Descripción** | | |
| --- | --- | --- | --- | --- | --- |
| asr\_id |  | INT | Identificador del Asesor | | |
| asr\_nombre | 30 | VARCHAR | Nombres del Asesor | | |
| asr\_apepaterno | 15 | VARCHAR | Apellido paterno del Asesor | | |
| asr\_apematerno | 15 | VARCHAR | Apellido materno del Asesor | | |
| asr\_identificación | 8 | INT | Documento de Identidad del Asesor | | |
| asr\_fechanac |  | DATE | Fecha de nacimiento del Asesor | | |
| asr\_sexo | 10 | VARCHAR | Sexo del Asesor | | |
| asr\_email | 50 | VARCHAR | Email del Asesor | | |
| asr\_telefono | 15 | VARCHAR | Teléfono del Asesor | | |
| asr\_sueldo | 18,2 | NUMERIC | Sueldo del Asesor | | |
| ciu\_id |  | INT | Identificador de la Ciudad | | |
| turno\_id |  | INT | Identificador del Turno | | |
|  |  |  |  |  |  |
| **Relaciones**: | | | **Campos Clave:** | |  |
| TablaAsesores con TablaCiudad | |  | asr\_id, ciu\_id, asr\_nombre, asr\_apellido | | |
| TablaAsesores con TablaAsesorias | |  |  |  |  |
| TablaAsesores con TablaPedido | |  |  |  |  |

### Tabla Asesorías

| **Campo** | **Tamaño** | **Tipo de Dato** | **Descripción** | | |
| --- | --- | --- | --- | --- | --- |
| ases\_id |  | INT | Identificador de la Asesoria | | |
| cli\_id |  | INT | Identificador del Cliente | | |
| asr\_id |  | INT | Identificador del Asesor | | |
| ases\_fecha |  | DATE | Fecha de la Asesoria | | |
| ases\_observ | 200 | VARCHAR | Observación de la Asesoria | | |
| estado | 10 | VARCHAR | Estado de la Asesoria | | |
|  |  |  |  |  |  |
| **Relaciones:** | | | **Campos Clave:** | |  |
| TablaAsesorias con TablaAsesores | |  | ases\_id, cli\_id, asr\_id, ciu\_id | | |
| TablaAsesorias con TablaCliente | |  |  |  |  |

1. **Tabla Planes**

| **Campo** | **Tamaño** | **Tipo de Dato** | **Descripción** | | |
| --- | --- | --- | --- | --- | --- |
| pln\_id |  | INT | Identificador del Plan | | |
| pln\_nombre | 20 | VARCHAR | Nombre del Plan | | |
| pln\_costo | 18,2 | NUMERIC | Costo del Plan | | |
| pln\_trabajadores |  | INT | Trabajadores del Plan | | |
|  |  |  |  |  |  |
| **Relaciones:** | | | **Campos Clave:** | |  |
| TablaPlanes con TablaPedido | |  | pln\_id, pln\_nombre | | |

### Tabla Pedido

| Campo | Tamaño | Tipo de Dato | Descripción | | |  |
| --- | --- | --- | --- | --- | --- | --- |
| pdd\_id |  | INT | Identificador del Pedido | | |  |
| cli\_id |  | INT | Identificador del Cliente | | |  |
| asr\_id |  | INT | Identificador del Asesor | | |  |
| pln\_id |  | INT | Identificador del Plan | | |  |
| pdd\_meses |  | INT | Meses de Pedido | | |  |
| pdd\_total | 18,2 | NUMERIC | Total de Pedido | | |  |
| pdd\_fechaini |  | DATE | Fecha de Pedido | | |  |
| pdd\_fechafin |  | DATE | Fecha de Pedido | | |  |
| pdd\_estado | 15 | VARCHAR | Estado del Pedido | | |  |
|  |  |  |  |  |  |  |
| **Relaciones:** | | | **Campos Clave:** | |  |  |
| TablaPedido con TablaPlanes | |  | pdd\_id, pln\_id, cli\_id, asr\_id, ciu\_id, pdd\_total | | | |
| TablaPedido con TablaAsesores | |  |  |  |  |  |
| TablaPedido con TablaCliente | |  |  |  |  |  |

1. **SCRIPT**

CREATE DATABASE Proyecto\_GDI

GO

USE Proyecto\_GDI

GO

CREATE TABLE dbo.Ciudad

(

ciu\_Id int NOT NULL,

ciu\_nombre varchar(15) NULL ,

CONSTRAINT XPKciudad PRIMARY KEY (ciu\_Id ASC)

)

go

CREATE TABLE dbo.Gerente

(

grt\_id int NOT NULL,

grt\_nombre varchar(30) NULL ,

grt\_apepaterno varchar(15) NULL ,

grt\_apematerno varchar(15) NULL ,

grt\_nroidentificacion char(8) NULL ,

grt\_sexo varchar(10) NULL ,

grt\_email varchar(50) NULL ,

grt\_telefono varchar(15) NULL ,

grt\_fechanac date NULL ,

ciu\_Id int NULL ,

CONSTRAINT XPKGerente PRIMARY KEY (grt\_id ASC),

CONSTRAINT R\_12 FOREIGN KEY (ciu\_Id) REFERENCES dbo.ciudad(ciu\_Id)

ON DELETE NO ACTION

ON UPDATE NO ACTION

)

go

CREATE TABLE dbo.Cliente

(

cli\_id int NOT NULL,

cli\_nombre varchar(30) NOT NULL ,

cli\_apepaterno varchar(15) NULL ,

cli\_apematerno varchar(15) NULL ,

cli\_nroidentificacion char(8) NULL ,

cli\_sexo varchar(10) NOT NULL ,

cli\_fechanac date NOT NULL ,

cli\_email varchar(50) NOT NULL ,

cli\_telefono varchar(15) NOT NULL ,

cli\_ruc char(11) NOT NULL ,

ciu\_Id int NULL ,

CONSTRAINT XPKcliente PRIMARY KEY (cli\_id ASC),

CONSTRAINT R\_16 FOREIGN KEY (ciu\_Id) REFERENCES dbo.ciudad(ciu\_Id)

ON DELETE NO ACTION

ON UPDATE NO ACTION

)

go

CREATE TABLE dbo.Turno

(

turno\_id int NOT NULL,

turno\_nombre varchar(10) NULL ,

turno\_horario varchar(30) NULL ,

CONSTRAINT XPKTurno PRIMARY KEY (turno\_id ASC)

)

go

CREATE TABLE dbo.Asesores

(

asr\_id int NOT NULL,

asr\_nombre varchar(30) NULL ,

asr\_apepaterno varchar(15) NULL ,

asr\_apematerno varchar(15) NULL ,

asr\_nroidentificacion char(8) NULL ,

asr\_fechanac date NULL ,

asr\_sexo varchar(10) NULL ,

asr\_email varchar(50) NULL ,

asr\_telefono varchar(15) NULL ,

asr\_sueldo numeric(18,2) NULL ,

ciu\_Id int NULL ,

turno\_id int NULL ,

CONSTRAINT XPKasesores PRIMARY KEY (asr\_id ASC),

CONSTRAINT R\_13 FOREIGN KEY (ciu\_Id) REFERENCES dbo.ciudad(ciu\_Id)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT R\_18 FOREIGN KEY (turno\_id) REFERENCES dbo.Turno(turno\_id)

ON DELETE NO ACTION

ON UPDATE NO ACTION

)

go

CREATE TABLE dbo.Planes

(

pln\_id int NOT NULL,

pln\_nombre varchar(10) NOT NULL ,

pln\_costo numeric(18,2) NOT NULL ,

pln\_trabajadores int NOT NULL ,

CONSTRAINT XPKplan PRIMARY KEY (pln\_id ASC)

)

go

CREATE TABLE dbo.Pedido

(

pdd\_id int NOT NULL,

cli\_id int NULL ,

asr\_id int NULL ,

pln\_id int NULL ,

pdd\_meses int NULL ,

pdd\_total numeric(18,2) NULL ,

pdd\_fechaini date NULL ,

pdd\_fechafin date NULL ,

pdd\_estado varchar(15) NULL,

CONSTRAINT XPKpedido PRIMARY KEY (pdd\_id ASC),

CONSTRAINT R\_22 FOREIGN KEY (cli\_id) REFERENCES dbo.cliente(cli\_id)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT R\_23 FOREIGN KEY (asr\_id) REFERENCES dbo.asesores(asr\_id)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT R\_24 FOREIGN KEY (pln\_id) REFERENCES dbo.planes(pln\_id)

ON DELETE NO ACTION

ON UPDATE NO ACTION

)

go

CREATE TABLE dbo.Asesorias

(

ases\_id int NOT NULL,

cli\_id int NULL ,

asr\_id int NULL ,

ases\_fecha date NOT NULL ,

ases\_observ varchar(200) NOT NULL ,

estado varchar(10) NOT NULL ,

CONSTRAINT XPKasesorias PRIMARY KEY (ases\_id ASC),

CONSTRAINT R\_19 FOREIGN KEY (asr\_id) REFERENCES dbo.asesores(asr\_id)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT R\_20 FOREIGN KEY (cli\_id) REFERENCES dbo.cliente(cli\_id)

ON DELETE NO ACTION

ON UPDATE NO ACTION

)

go

use Proyecto\_GDI

--Inserción de datos

--INSERT PLAN

INSERT INTO Planes VALUES (1,'Básico','90','4')

INSERT INTO Planes VALUES (2,'Pro','130','8')

INSERT INTO Planes VALUES (3,'Premium','260','16')

--INSERT CIUDAD

INSERT INTO ciudad(ciu\_Id,ciu\_nombre)

values(1,'Lima');

go

INSERT INTO ciudad(ciu\_Id,ciu\_nombre)

values(2,'Callao');

go

INSERT INTO ciudad(ciu\_Id,ciu\_nombre)

values(3,'Arequipa');

go

INSERT INTO ciudad(ciu\_Id,ciu\_nombre)

values(4,'Tacna');

go

INSERT INTO ciudad(ciu\_Id,ciu\_nombre)

values(5,'Junin');

go

INSERT INTO ciudad(ciu\_Id,ciu\_nombre)

values(6,'Loreto');

go

--INSERT TURNO

INSERT INTO Turno(turno\_id,turno\_nombre,turno\_horario)

values(1,'Mañana','8:00 - 12:00');

go

INSERT INTO Turno(turno\_id,turno\_nombre,turno\_horario)

values(2,'Tarde','13:00 - 18:00');

go

--INSERT CLIENTE ORDEN

INSERT INTO CLIENTE(cli\_id,cli\_nombre,cli\_apepaterno,cli\_apematerno,cli\_nroidentificacion,cli\_sexo,cli\_fechanac,cli\_email,cli\_telefono,cli\_ruc,ciu\_Id)

VALUES(1,'Daniel','Figueroa','Millan','79213122','M',GETDATE(),'danielfiguero@gmail.com',92456781,2022091232,1);

--INSERT ORDEN ASESORES

INSERT INTO asesores(asr\_id,asr\_nombre,asr\_apepaterno,asr\_apematerno,asr\_nroidentificacion,asr\_fechanac,asr\_sexo,asr\_email,asr\_telefono,asr\_sueldo,ciu\_Id,turno\_id)

VALUES(1,'Marco','Caballero','Perez','73022690',GETDATE(),'M','marcoestrada@hotmail.com',92456781,2000,2,1);

--INSERT EN ASESORIAS ORDEN

INSERT INTO asesorias(ases\_id,cli\_id,asr\_id,ases\_fecha,ases\_observ,estado)

VALUES(1,1,1,GETDATE(),'La reunion se realizara a travez de este zoom: https://zoom.us/9293182731','Atendido');

--SP START

-----------------------------------------------

drop procedure usp\_registrar\_cliente;

go

create procedure usp\_registrar\_cliente

(

@p\_cli\_nombre varchar(30),

@p\_cli\_apepaterno varchar(30),

@p\_cli\_apematerno varchar(30),

@p\_cli\_nroidentificacion char(8),

@p\_cli\_sexo varchar(10),

@p\_cli\_fechanac date,

@p\_cli\_email varchar(50),

@p\_cli\_telefono varchar(50),

@p\_cli\_ruc char(11),

@p\_ciu\_id int,

----------

@p\_msg varchar (100) OUT,

@p\_estado int OUT

)

AS

BEGIN

-- Variables

-- Validaciones

if(@p\_cli\_apepaterno is null OR @p\_cli\_apematerno is null OR @p\_cli\_nombre is null OR @p\_cli\_nroidentificacion is null)

begin

set @p\_estado = -3;

set @p\_msg = 'Proceso fue abortado por que no se tienen espacios rellenados';

return;

end;

-- Proceso

SET @p\_estado = 1;

set @p\_msg = 'Proceso fue realizado';

BEGIN TRY

-- Inicia la Tx

BEGIN TRANSACTION;

-- Generar el nuevo codigo

Declare @id int = (Select ISNULL(MAX(cli\_id),0) from Cliente)+1;

-- Inserta el nuevo cliente

INSERT INTO CLIENTE(cli\_id,cli\_nombre,cli\_apepaterno,cli\_apematerno,cli\_nroidentificacion,cli\_sexo,cli\_fechanac,cli\_email,cli\_telefono,cli\_ruc,ciu\_Id)

VALUES(@id,@p\_cli\_nombre,@p\_cli\_apepaterno,@p\_cli\_apematerno,@p\_cli\_nroidentificacion,@p\_cli\_sexo,@p\_cli\_fechanac,@p\_cli\_email,@p\_cli\_telefono,@p\_cli\_ruc,@p\_ciu\_id);

-- Confirma la Tx

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

-- Cancela la Tx

ROLLBACK TRANSACTION;

if(@p\_estado=1) set @p\_estado = -1;

set @p\_msg = 'Proceso fallido';

END CATCH;

END;

GO

begin

declare @msg varchar(100), @estado int;

exec usp\_registrar\_cliente 'Esteban','Bueno','Martinez','76232221','M','2002-03-01','esteban22@hotmail.com','98009221','576310111',1, @msg out, @estado out;

print concat('Estado: ', @estado);

print concat('Mensaje: ', @msg);

end;

go

begin

declare @msg varchar(100), @estado int;

exec usp\_registrar\_cliente 'Emilia','De la Cruz','Pereira','76890192','F','1997-03-12','emiliasantos@santos.com','988872122','20239911122',1, @msg out, @estado out;

print concat('Estado: ', @estado);

print concat('Mensaje: ', @msg);

end;

go

--------------------------------------------

drop procedure usp\_registrar\_asesor;

go

create procedure usp\_registrar\_asesor

(

@p\_asr\_nombre varchar(30),

@p\_asr\_apepaterno varchar(30),

@p\_asr\_apematerno varchar(30),

@p\_asr\_nroidentificacion char(8),

@p\_asr\_fechanac date,

@p\_asr\_sexo varchar(10),

@p\_asr\_email varchar(50),

@p\_asr\_telefono varchar(15),

@p\_asr\_sueldo numeric,

@p\_ciu\_id int,

@p\_turno\_id int,

----------

@p\_msg varchar (100) OUT,

@p\_estado int OUT

)

AS

BEGIN

-- Variables

-- Validaciones

if(@p\_asr\_apepaterno is null OR @p\_asr\_apematerno is null OR @p\_asr\_nombre is null OR @p\_asr\_nroidentificacion is null)

begin

set @p\_estado = -3;

set @p\_msg = 'Proceso fue abortado por que no se tienen espacios rellenados';

return;

end;

-- Proceso

SET @p\_estado = 1;

set @p\_msg = 'Proceso fue realizado';

BEGIN TRY

-- Inicia la Tx

BEGIN TRANSACTION;

-- Generar el nuevo codigo

Declare @id int = (Select ISNULL(MAX(asr\_id),0) from Asesores)+1;

-- Inserta el nuevo cliente

INSERT INTO asesores(asr\_id,asr\_nombre,asr\_apepaterno,asr\_apematerno,asr\_nroidentificacion,asr\_fechanac,asr\_sexo,asr\_email,asr\_telefono,asr\_sueldo,ciu\_Id,turno\_id)

VALUES(@id,@p\_asr\_nombre,@p\_asr\_apepaterno,@p\_asr\_apematerno,@p\_asr\_nroidentificacion,@p\_asr\_fechanac,@p\_asr\_sexo,@p\_asr\_email,@p\_asr\_telefono,@p\_asr\_sueldo,@p\_ciu\_id,@p\_turno\_id);

-- Confirma la Tx

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

-- Cancela la Tx

ROLLBACK TRANSACTION;

if(@p\_estado=1) set @p\_estado = -1;

set @p\_msg = 'Proceso fallido';

END CATCH;

END;

GO

begin

declare @msg varchar(100), @estado int;

exec usp\_registrar\_asesor 'Arturo','Millan','Estrada','20668712','1989-01-12','M','arturomillan@hotmail.com','987678564','1700',3,2, @msg out, @estado out;

print concat('Estado: ', @estado);

print concat('Mensaje: ', @msg);

end;

go

begin

declare @msg varchar(100), @estado int;

exec usp\_registrar\_asesor 'Anthonela','Ramirez','Garces','68271113','1999-11-10','F','mantillaaa@hotmail.com','9879876111','1800',3,2, @msg out, @estado out;

print concat('Estado: ', @estado);

print concat('Mensaje: ', @msg);

end;

go

-------------------------------------------------

drop procedure usp\_reservar\_asesoria;

go

create procedure usp\_reservar\_asesoria

(

@p\_cli\_id int,

@p\_asr\_id int,

@p\_ases\_fecha date,

@p\_ases\_observ varchar(200),

----------

@p\_msg varchar (100) OUT,

@p\_estado int OUT

)

AS

BEGIN

-- Variables

DECLARE @ases\_id int;

-- Validaciones

if(@p\_cli\_id is null OR @p\_asr\_id is null)

begin

set @p\_estado = -3;

set @p\_msg = 'Proceso fue abortado por que no se tienen espacios rellenados';

return;

end;

-- Proceso

SET @p\_estado = 1;

set @p\_msg = 'Proceso fue realizado';

BEGIN TRY

-- Inicia la Tx

BEGIN TRANSACTION;

-- Generar el nuevo codigo

Declare @id int = (Select ISNULL(MAX(ases\_id),0) from Asesorias)+1;

-- Inserta el nuevo cliente

INSERT INTO asesorias(ases\_id,cli\_id,asr\_id,ases\_fecha,ases\_observ,estado)

VALUES(@id,@p\_cli\_id,@p\_asr\_id,@p\_ases\_fecha,@p\_ases\_observ,'Pendiente');

-- Confirma la Tx

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

-- Cancela la Tx

ROLLBACK TRANSACTION;

if(@p\_estado=1) set @p\_estado = -1;

set @p\_msg = 'Proceso fallido';

END CATCH;

END;

GO

begin

declare @msg varchar(100), @estado int;

exec usp\_reservar\_asesoria 1,2,'2022-06-10','La reunion se realizara a travez de este zoom: https://meet.google.com/xua-by-iuhg', @msg out, @estado out;

print concat('Estado: ', @estado);

print concat('Mensaje: ', @msg);

end;

go

begin

declare @msg varchar(100), @estado int;

exec usp\_reservar\_asesoria 1,1,'2022-04-22','La reunion se realizara a travez de este meet: https://meet.google.com/uid-wme-ldk', @msg out, @estado out;

print concat('Estado: ', @estado);

print concat('Mensaje: ', @msg);

end;

go

------------------------------

create procedure usp\_registrar\_pedido

(

@p\_cli\_id int,

@p\_asr\_id int,

@p\_pln\_id int,

@p\_pdd\_meses int,

@p\_pdd\_fechaini date,

----------

@p\_pdd\_fechafin date out,

@p\_pdd\_total numeric(18,2) OUT,

@p\_msg varchar (100) OUT,

@p\_estado int OUT

)

AS

BEGIN

-- Variables

DECLARE @asr\_id int;

-- Validaciones

if(

@p\_cli\_id is null OR

@p\_asr\_id is null OR

@p\_pln\_id is null OR

@p\_pdd\_meses = 0 OR

@p\_pdd\_fechaini is null

)

begin

set @p\_estado = -2;

set @p\_msg = 'Proceso fue abortado por que no se tienen espacios rellenados';

return;

end;

-- Proceso

SET @p\_estado = 1;

set @p\_msg = 'Proceso fue realizado';

BEGIN TRY

-- Inicia la Tx

BEGIN TRANSACTION;

-- Establecer datos

Declare @id int = (Select ISNULL(MAX(pdd\_id),0) from Pedido)+1;

set @p\_pdd\_total = Cast(@p\_pdd\_meses as numeric(18,2)) \* (Select p.pln\_costo from Planes p where p.pln\_id = @p\_pln\_id)

set @p\_pdd\_fechafin = (Select DATEADD(MONTH,@p\_pdd\_meses,@p\_pdd\_fechaini))

-- Inserta el nuevo Pedido

INSERT INTO Pedido VALUES(@id,@p\_cli\_id,@p\_asr\_id,@p\_pln\_id,@p\_pdd\_meses,@p\_pdd\_total,

@p\_pdd\_fechaini,@p\_pdd\_fechafin,'Pendiente');

-- Confirma la Tx

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

-- Cancela la Tx

ROLLBACK TRANSACTION;

if(@p\_estado=1) set @p\_estado = -1;

set @p\_msg = 'Proceso fallido';

END CATCH;

END;

GO

begin

declare @fechaini date = (select GETDATE()),@fechafin date, @total numeric(18,2),@msg varchar(100), @estado int;

exec usp\_registrar\_pedido 1,1,1,1,@fechaini,@fechafin out, @total out,@msg out, @estado out;

print concat('Estado: ', @estado);

print concat('Mensaje: ', @msg);

print concat('Total: ',@total);

print concat('Fecha final: ',@fechafin)

end;

go

------------------------------

create procedure usp\_consultar\_asesores

(

@p\_mensaje varchar(100) OUT,

@p\_estado int OUT

)

AS

BEGIN

-- Variables

DECLARE @msg\_error varchar(50);

-- Proceso

if(@p\_estado is NULL)

BEGIN

SET @p\_estado = 1;

END

SET @p\_estado = 1;

BEGIN TRY

-- Inicia la Tx

BEGIN TRANSACTION;

-- Llenar las variables

select a.asr\_id,a.asr\_nombre,a.asr\_apepaterno,a.asr\_apematerno,a.asr\_nroidentificacion,

a.asr\_fechanac,a.asr\_sexo,a.asr\_email,a.asr\_telefono,a.asr\_sueldo,c.ciu\_nombre,

t.turno\_nombre,t.turno\_horario

from Asesores a

inner join Turno t ON t.turno\_id = a.turno\_id

inner join Ciudad c ON c.ciu\_Id = a.ciu\_Id

set @p\_mensaje = 'Consulta realizada exitosamente.'

-- Confirma la Tx

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

-- Cancela la Tx

ROLLBACK TRANSACTION;

set @p\_estado = -1;

END CATCH;

END;

GO

--2. Verificar su funcionamiento.

-- Prueba 1:

declare @estado int, @mensaje varchar(100)

exec usp\_consultar\_asesores @mensaje out,@estado out;

print concat('Estado: ', @estado);

print concat('Mensaje: ', @mensaje);

go

-------------------------

create procedure usp\_consultar\_asesorias

(

@p\_mensaje varchar(100) OUT,

@p\_estado int OUT

)

AS

BEGIN

-- Variables

DECLARE @msg\_error varchar(50);

-- Proceso

if(@p\_estado is NULL)

BEGIN

SET @p\_estado = 1;

END

SET @p\_estado = 1;

BEGIN TRY

-- Inicia la Tx

BEGIN TRANSACTION;

-- Llenar las variables

select ases.cli\_id,cli.cli\_nombre,cli.cli\_apepaterno,cli.cli\_apematerno,

ases.asr\_id,a.asr\_nombre,a.asr\_apepaterno,a.asr\_apematerno,

ases.ases\_fecha,ases.ases\_observ,ases.estado

from Asesorias ases

inner join Cliente cli ON cli.cli\_id = ases.cli\_id

inner join Asesores a ON a.asr\_id = ases.asr\_id

set @p\_mensaje = 'Consulta realizada exitosamente.'

-- Confirma la Tx

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

-- Cancela la Tx

ROLLBACK TRANSACTION;

set @p\_estado = -1;

END CATCH;

END;

GO

--2. Verificar su funcionamiento.

-- Prueba 1:

declare @estado int, @mensaje varchar(100)

exec usp\_consultar\_asesorias @mensaje out,@estado out;

print concat('Estado: ', @estado);

print concat('Mensaje: ', @mensaje);

go

---------------------------------

create procedure usp\_consultar\_pedido

(

@p\_mensaje varchar(100) OUT,

@p\_estado int OUT

)

AS

BEGIN

-- Variables

DECLARE @msg\_error varchar(50);

-- Proceso

if(@p\_estado is NULL)

BEGIN

SET @p\_estado = 1;

END

SET @p\_estado = 1;

BEGIN TRY

-- Inicia la Tx

BEGIN TRANSACTION;

-- Llenar las variables

select p.pdd\_id,c.cli\_nombre,c.cli\_apepaterno,c.cli\_apematerno,

a.asr\_nombre,a.asr\_apepaterno,a.asr\_apematerno,

pln.pln\_nombre,pln.pln\_trabajadores,

pdd\_meses,pdd\_fechaini,pdd\_fechafin,pdd\_estado

from Pedido p

inner join Cliente c ON c.cli\_id = p.cli\_id

inner join Asesores a ON a.asr\_id = p.asr\_id

inner join Planes pln ON p.pln\_id = pln.pln\_id

set @p\_mensaje = 'Consulta realizada exitosamente.'

-- Confirma la Tx

COMMIT TRANSACTION;

END TRY

BEGIN CATCH

-- Cancela la Tx

ROLLBACK TRANSACTION;

set @p\_estado = -1;

END CATCH;

END;

GO

--2. Verificar su funcionamiento.

-- Prueba 1:

declare @estado int, @mensaje varchar(100)

exec usp\_consultar\_pedido @mensaje out,@estado out;

print concat('Estado: ', @estado);

print concat('Mensaje: ', @mensaje);

go

--------------------------------------------

select \* from Asesorias

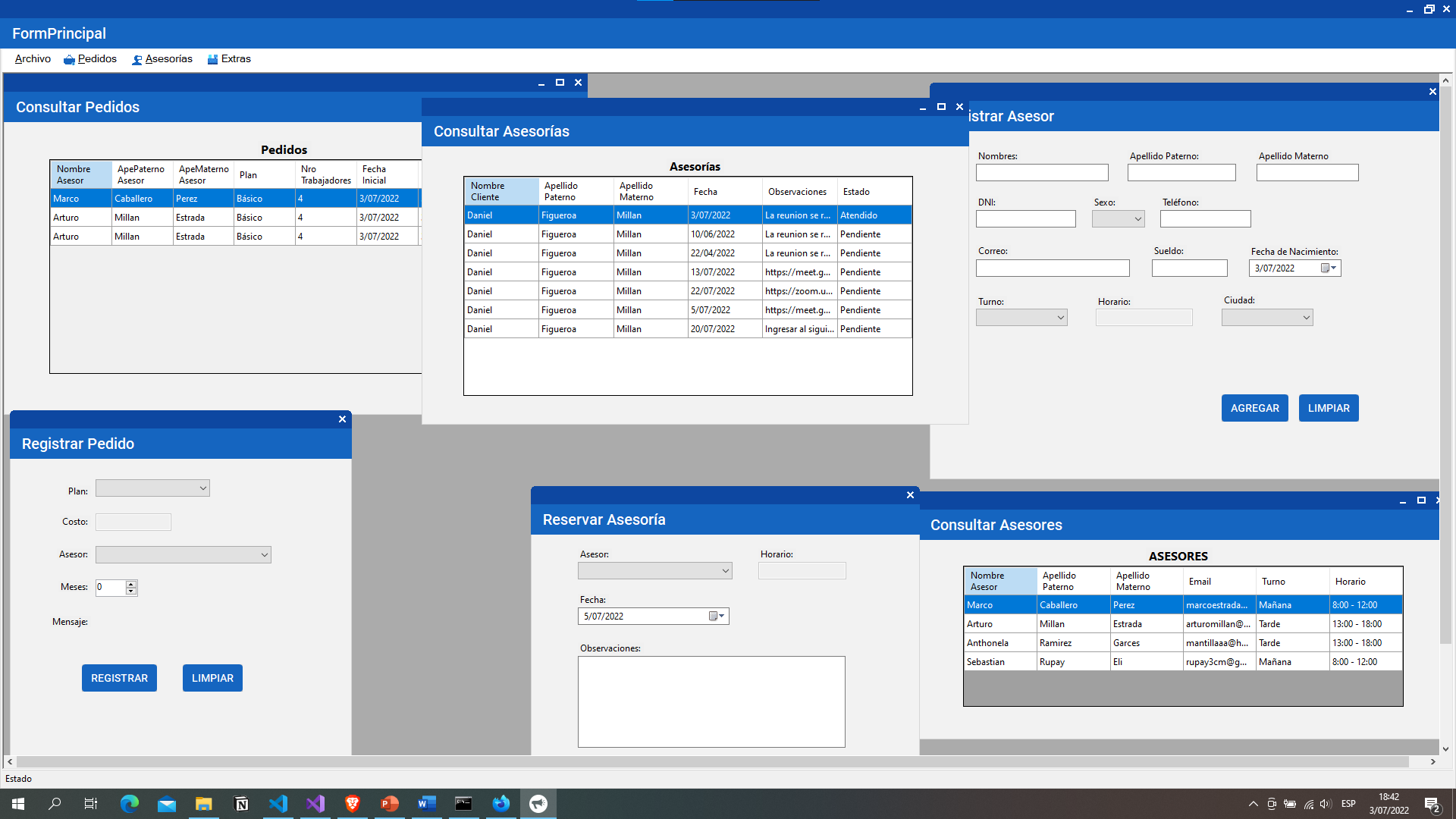
select \* from Asesores

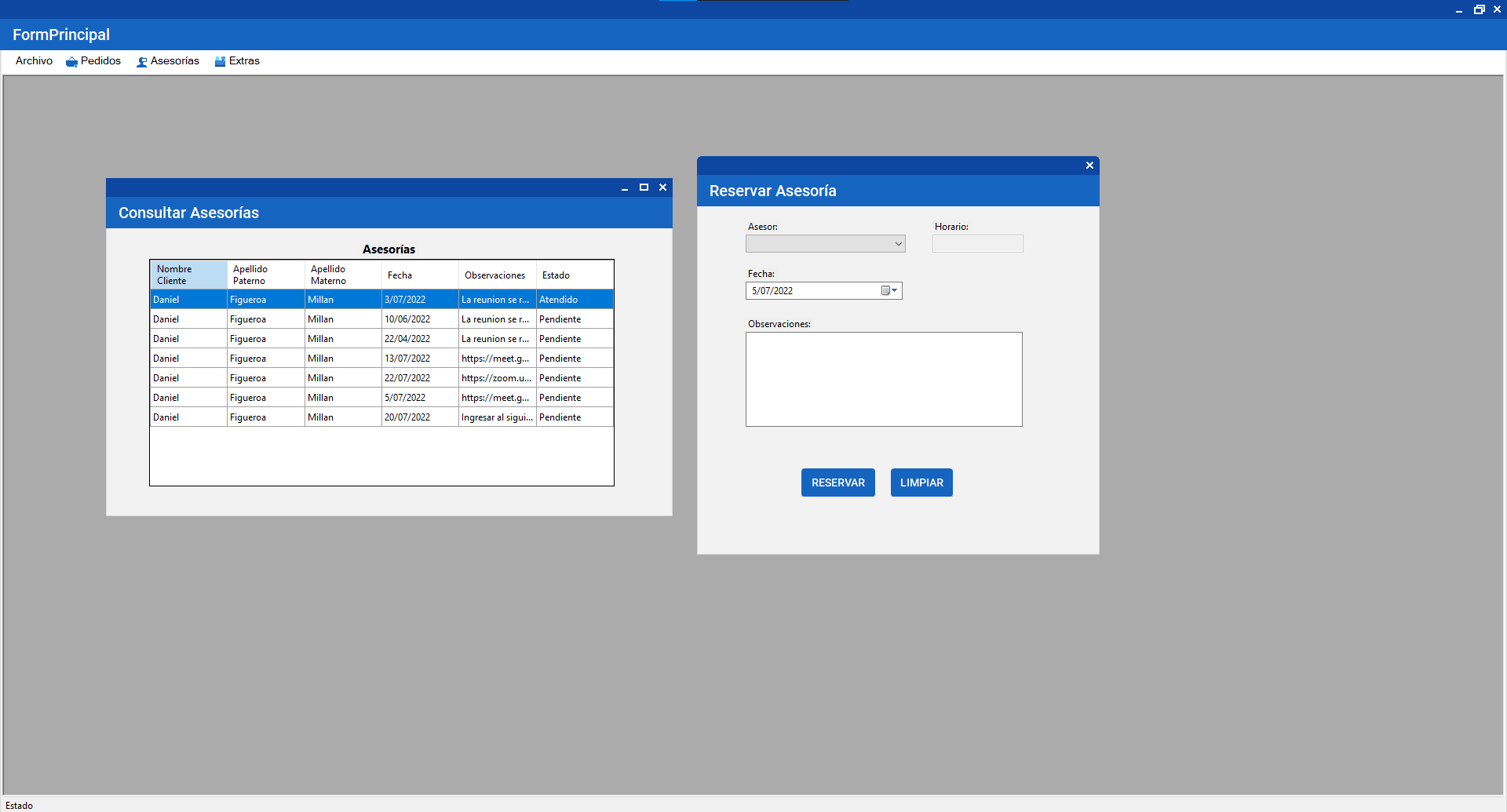
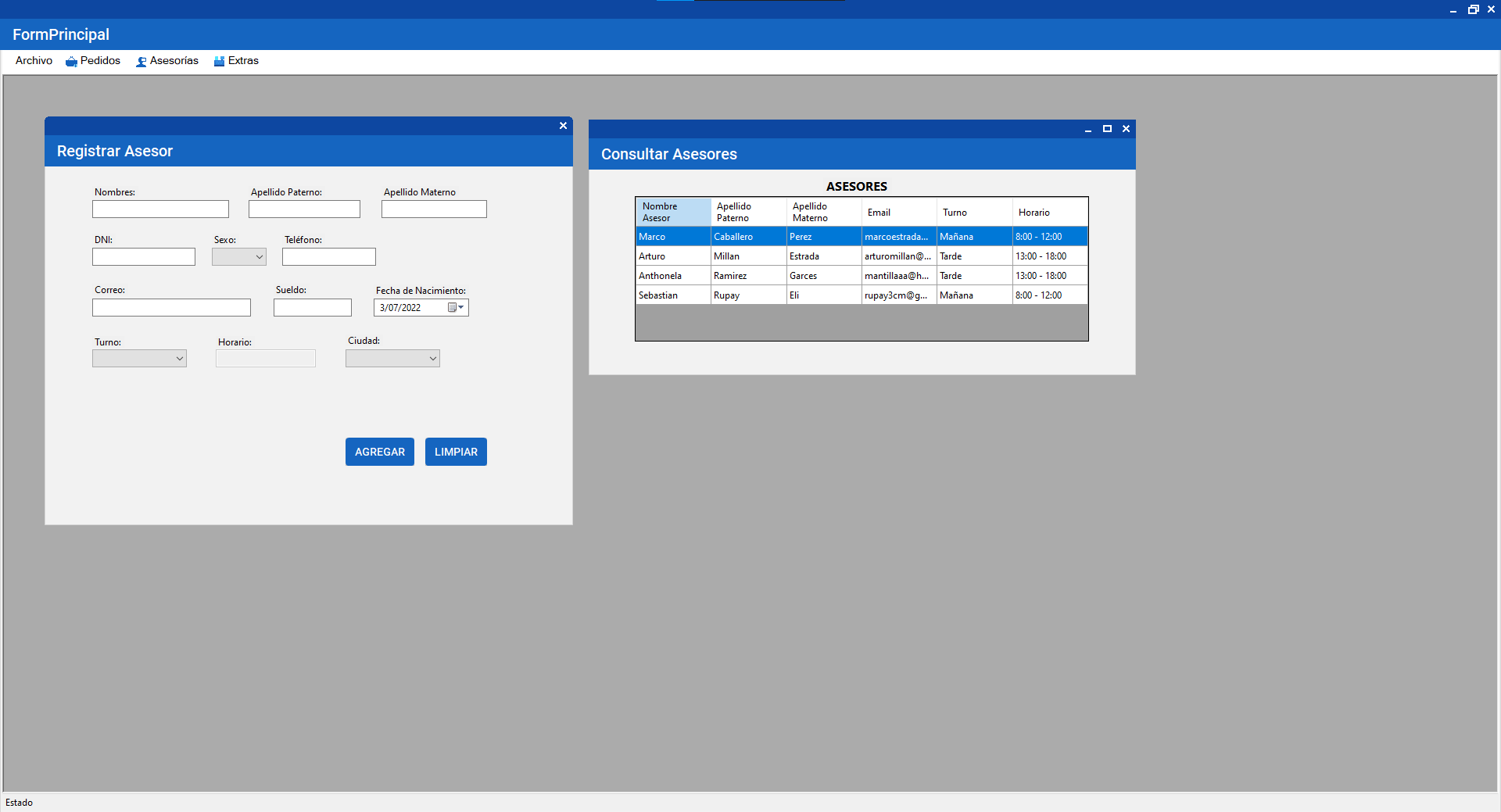
select \* from Pedido

1. **PROCESAMIENTO DE INTERFACES**

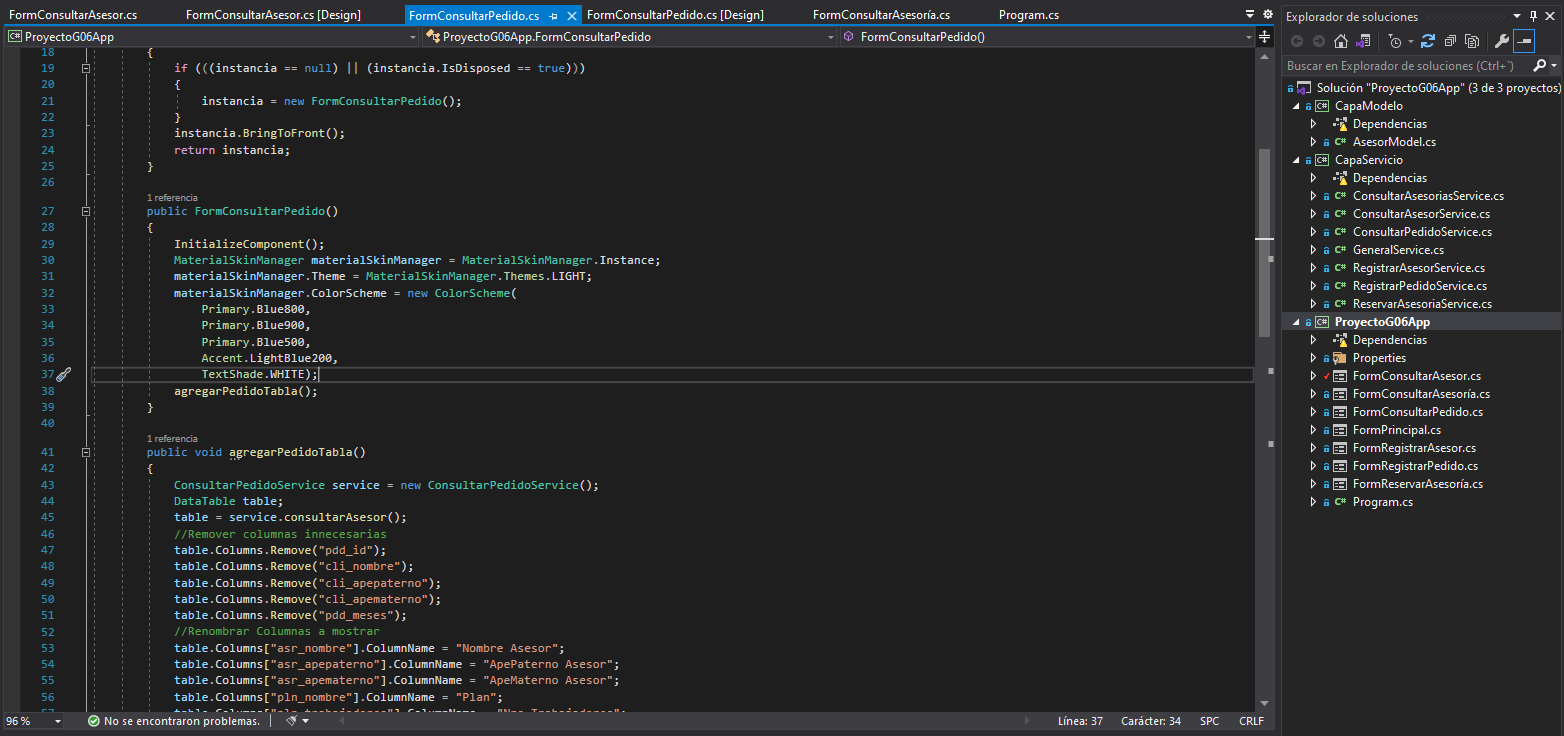
**10.1. VISUAL STUDIO**

Se diseñó unas interfaces con el empleo del lenguaje C#, a fin de utilizar los procedimientos almacenados construidos.Además de ello se empleó el uso de la librería Marial Skin2 para el diseño de la interfaz.



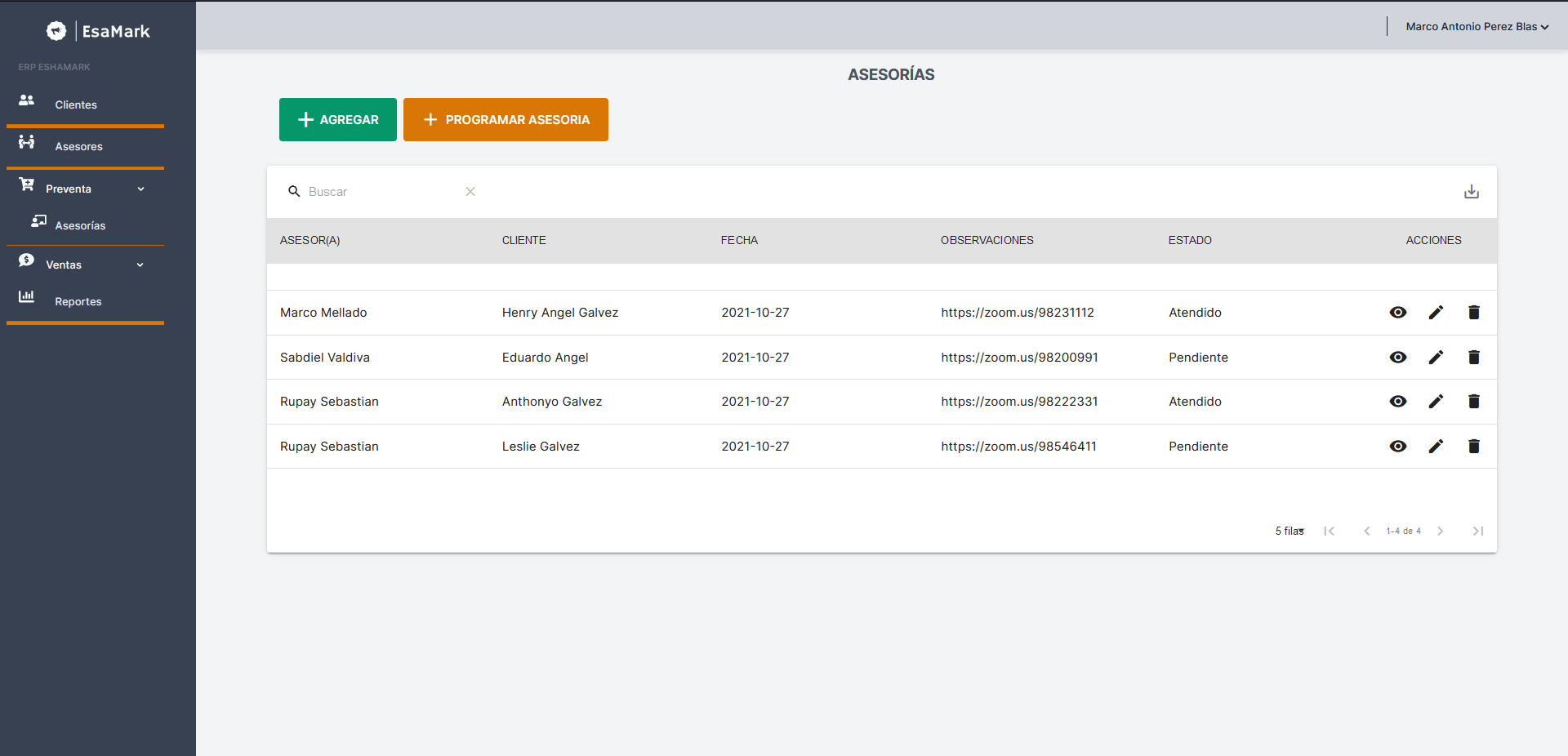
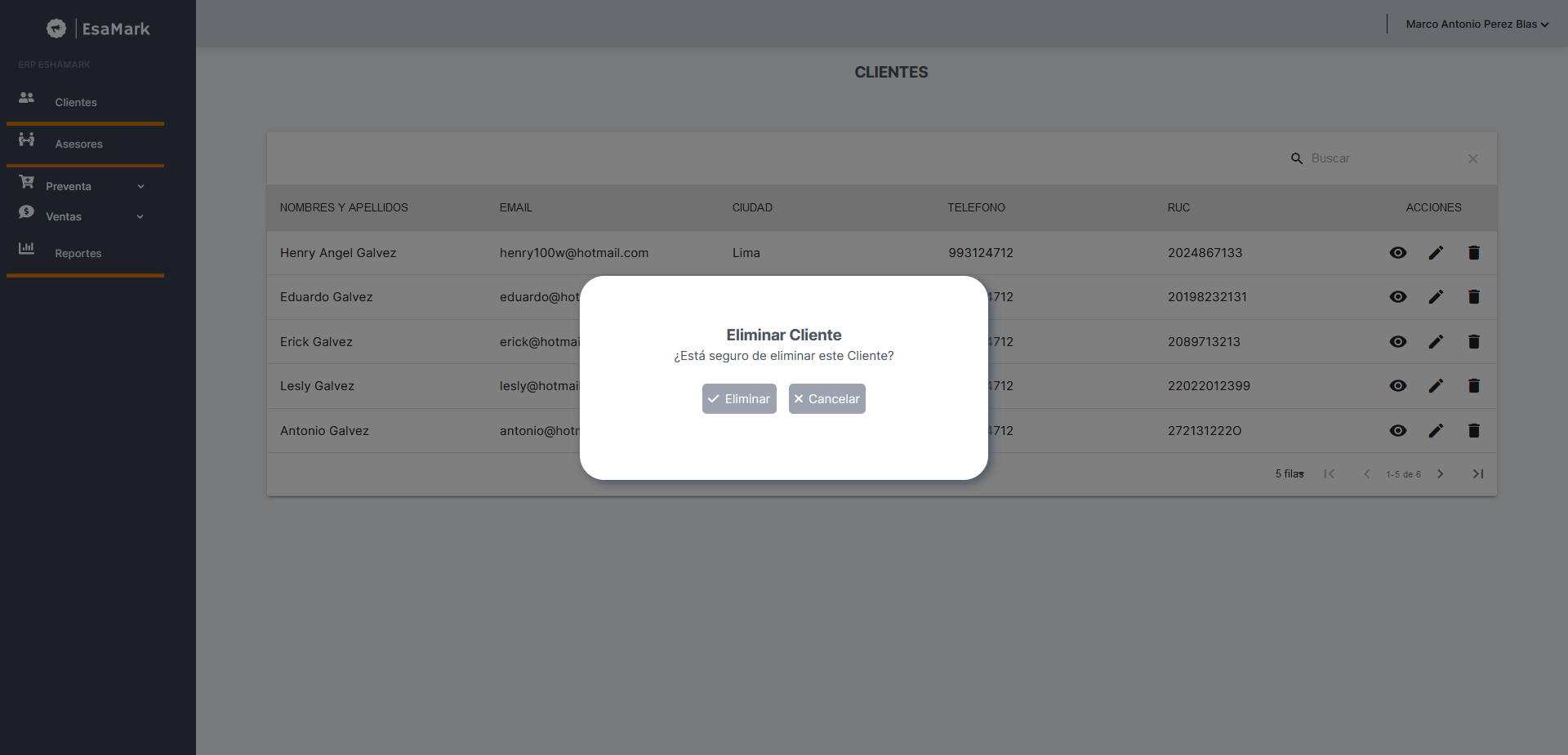
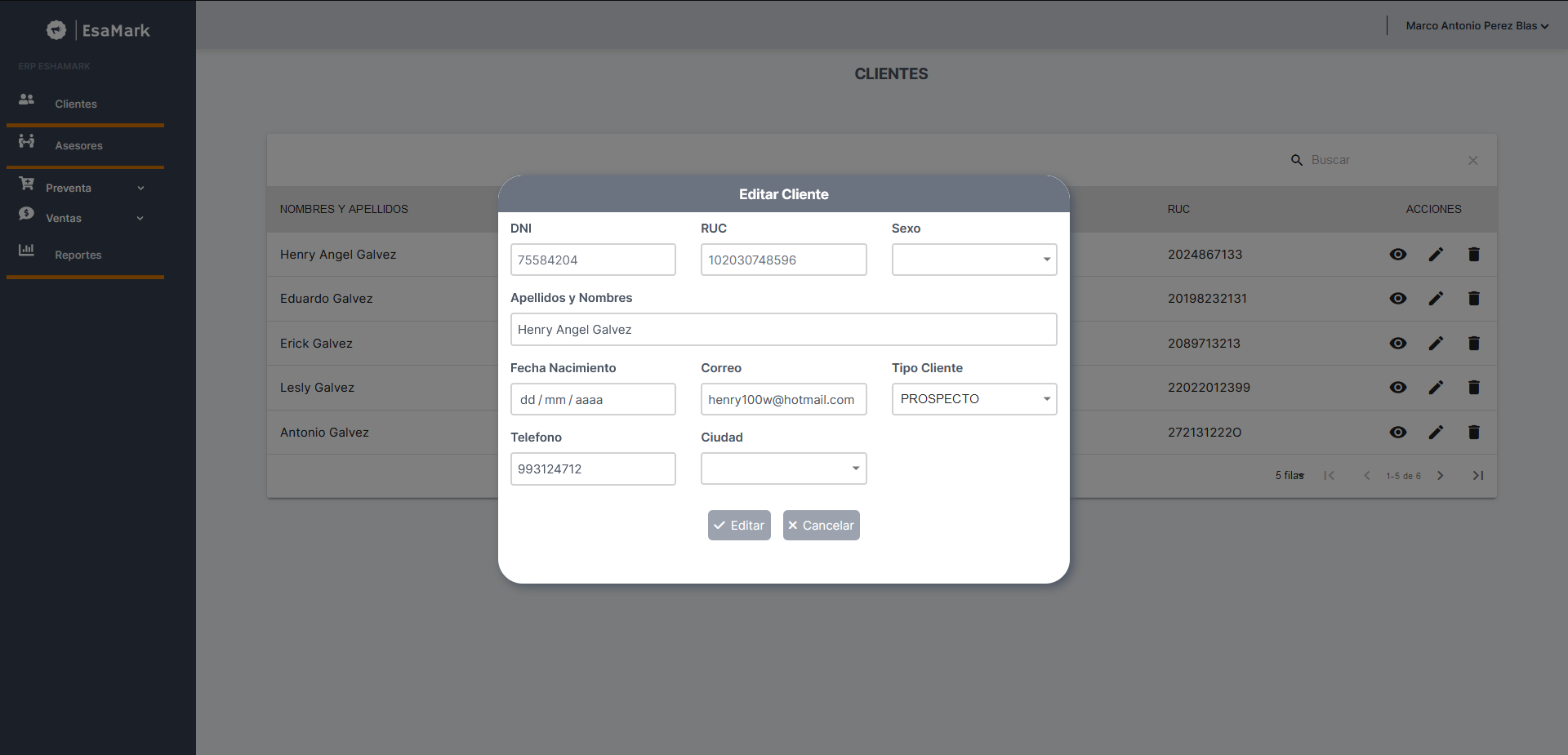
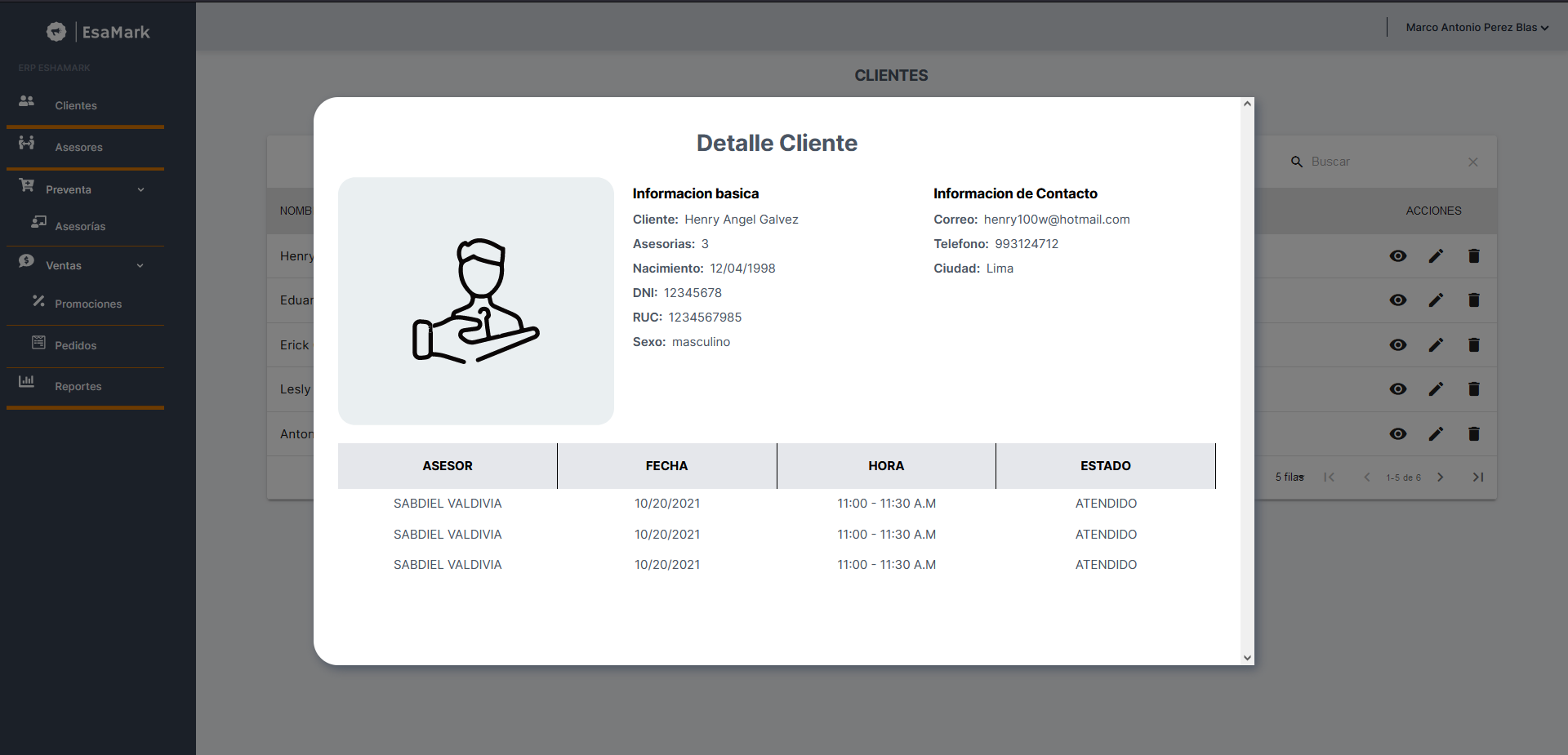
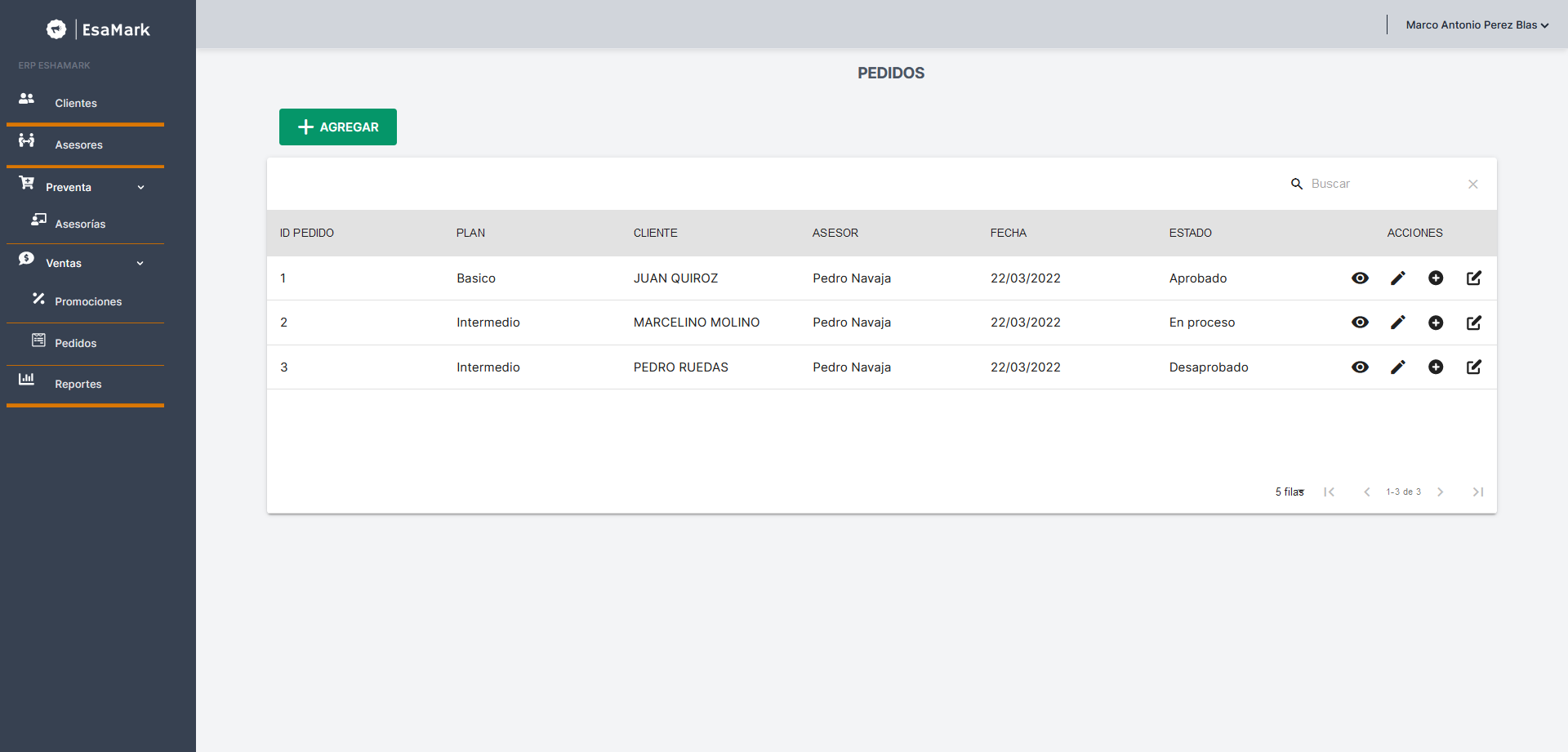


Además de lo mencionado anteriormente, se emplearon las capas Modelo y Servicio respectivamente para una mejor integración y estructuración del sistema.



**10.2. SIMULACIÓN**

Nuestro equipo desarrolló, una simulación empleando html,css,react,tailwind ss para demostrar una breve simulación del funcionamiento del sistema en un servidor web.



1. **CONCLUSIONES**

En conclusión, podemos decir que se cumplió con lo requerido y con lo plasmado en el proyecto de investigación la estructuración de la construcción de una prototipado que emplea la base de datos fijado a una agencia de marketing digital debido a que no cuenta con una base de datos que le posibilite laborar de una forma más dinámica, asimismo cumplimos con nuestro propósito que es que la compañía logre disponer de una base de datos y asimismo logre facilitar a los trabajadores y consumidores de la agencia de marketing a que logren hacer sus operaciones de forma más ordenada.

1. **RECOMENDACIONES**

En la estructuración de la construcción de un prototipo que emplea la base de datos fijado a la agencia de marketing, se recomienda a la organización tener un mejor control de su comprador y personal debido a que contará con una base de datos y asimismo esto le facilitará a los trabajadores y compradores de la agencia de marketing a hacer de mejor forma sus operaciones. Sin embargo, al comprador se le sugiere en el momento de hacer sus operaciones y obtener su servicio lo realice de la forma indicada y de una forma responsable para que esto no perjudique al desarrollo de la organización.