* *Capturas de pantalla:*
  + *Interfaz de usuario gráfica, Sitio web

    El contenido generado por IA puede ser incorrecto. Página principal con menú de navegación rápida*
  + *Vista de usuario al iniciar sesión y de citas hechas*

*Interfaz de usuario gráfica, Sitio web

El contenido generado por IA puede ser incorrecto.*

*Registro citas
*

* + *Formularios de: Login, Registro y Reservar cita*Interfaz de inicio de sesión
    

*Interfaz de usuario gráfica

El contenido generado por IA puede ser incorrecto.Interfaz de usuario gráfica, Aplicación, Sitio web

El contenido generado por IA puede ser incorrecto.*

* *Evidencia Base de Datos:*
* *Estructura:*
  + *Diagrama, Esquemático

    El contenido generado por IA puede ser incorrecto.Esquema base de datos*
  + *Listado de tablas con sus atributos (nombre, tipo de dato, relaciones)*

CREATE TABLE Bitacora (

id\_bitacora INTEGER PRIMARY KEY AUTOINCREMENT,

id\_usuario INTEGER NOT NULL,

descripcion\_accion TEXT NOT NULL,

fecha\_bitacora DATETIME NOT NULL,

tipo\_accion TEXT NOT NULL,

FOREIGN KEY(id\_usuario) REFERENCES Usuario(id\_usuario)

)

CREATE TABLE Cita (

id\_cita INTEGER PRIMARY KEY AUTOINCREMENT,

id\_cliente INTEGER NOT NULL,

id\_servicio INTEGER NOT NULL,

id\_estado\_cita INTEGER NOT NULL,

fecha\_cita DATE NOT NULL,

hora\_cita TIME NOT NULL,

FOREIGN KEY(id\_cliente) REFERENCES Usuario(id\_usuario),

FOREIGN KEY(id\_servicio) REFERENCES Servicio(id\_servicio),

FOREIGN KEY(id\_estado\_cita) REFERENCES EstadoCita(id\_estado\_cita)

)

CREATE TABLE Descuento (

id\_descuento INTEGER PRIMARY KEY AUTOINCREMENT,

id\_usuario INTEGER NOT NULL,

descuento REAL NOT NULL,

FOREIGN KEY(id\_usuario) REFERENCES Usuario(id\_usuario)

)

CREATE TABLE EstadoCita (

id\_estado\_cita INTEGER PRIMARY KEY AUTOINCREMENT,

estado\_cita TEXT CHECK(estado\_cita IN ('En progreso', 'Finalizado', 'Cancelado'))

)

CREATE TABLE EstadoUsuario (

id\_estado\_usuario INTEGER PRIMARY KEY AUTOINCREMENT,

estado\_usuario TEXT CHECK(estado\_usuario IN ('Activo', 'Bloqueado'))

)

CREATE TABLE FormaPago (

id\_forma\_pago INTEGER PRIMARY KEY AUTOINCREMENT,

nombre\_forma\_pago TEXT CHECK(nombre\_forma\_pago IN ('Efectivo', 'Débito', 'Crédito'))

)

CREATE TABLE Observacion (

id\_observacion INTEGER PRIMARY KEY AUTOINCREMENT,

nombre\_observacion TEXT CHECK(nombre\_observacion IN ('Puntualidad', 'Trato', 'Servicio', 'Barbero'))

)

CREATE TABLE RegistroPago (

id\_registro\_pago INTEGER PRIMARY KEY AUTOINCREMENT,

id\_cita INTEGER NOT NULL,

id\_forma\_pago INTEGER NOT NULL,

monto\_original REAL NOT NULL,

total\_pagado REAL NOT NULL,

id\_descuento INTEGER,

fecha\_pago DATETIME NOT NULL,

FOREIGN KEY(id\_cita) REFERENCES Cita(id\_cita),

FOREIGN KEY(id\_forma\_pago) REFERENCES FormaPago(id\_forma\_pago),

FOREIGN KEY(id\_descuento) REFERENCES Descuento(id\_descuento)

)

CREATE TABLE Servicio (

id\_servicio INTEGER PRIMARY KEY AUTOINCREMENT,

id\_usuario INTEGER NOT NULL,

id\_subservicio INTEGER NOT NULL,

descripcion TEXT,

duracion\_minutos INTEGER NOT NULL,

precio REAL NOT NULL,

FOREIGN KEY(id\_usuario) REFERENCES Usuario(id\_usuario),

FOREIGN KEY(id\_subservicio) REFERENCES SubServicio(id\_subservicio)

)

CREATE TABLE SubServicio (

id\_subservicio INTEGER PRIMARY KEY AUTOINCREMENT,

nombre\_servicio TEXT NOT NULL,

imagenes TEXT

)

CREATE TABLE TipoUsuario (

id\_tipo\_usuario INTEGER PRIMARY KEY AUTOINCREMENT,

tipo TEXT CHECK(tipo IN ('Barbero', 'Cliente', 'Barbería'))

)

CREATE TABLE Usuario (

id\_usuario INTEGER PRIMARY KEY AUTOINCREMENT,

id\_tipo\_usuario INTEGER NOT NULL,

nombre TEXT NOT NULL,

apellido TEXT NOT NULL,

telefono TEXT,

correo TEXT,

contraseña\_hash TEXT NOT NULL,

descripcion\_usuario TEXT,

imagen TEXT,

id\_estado\_usuario INTEGER NOT NULL,

FOREIGN KEY(id\_tipo\_usuario) REFERENCES TipoUsuario(id\_tipo\_usuario),

FOREIGN KEY(id\_estado\_usuario) REFERENCES EstadoUsuario(id\_estado\_usuario)

)

CREATE TABLE ValoracionObservacion (

id\_valoracion\_observacion INTEGER PRIMARY KEY AUTOINCREMENT,

valoracion INTEGER CHECK(valoracion BETWEEN 1 AND 5),

id\_observacion INTEGER NOT NULL,

id\_cita INTEGER NOT NULL,

FOREIGN KEY(id\_observacion) REFERENCES Observacion(id\_observacion),

FOREIGN KEY(id\_cita) REFERENCES Cita(id\_cita)

)

* + *Capturas de panel de administrador Base de Datos (Django)*

*Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.*

* + *Contenido:*
    - *Interfaz de usuario gráfica, Aplicación

      El contenido generado por IA puede ser incorrecto.Usuarios*
    - *Reservas*

Interfaz de usuario gráfica, Teams

El contenido generado por IA puede ser incorrecto.

* + - *Servicios*

*Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.*

* + - *Observaciones*

*Tabla

El contenido generado por IA puede ser incorrecto.*

* *Migraciones o Scripts*
  + *Código de migración (Django) para creación de tablas*
* class TipoUsuario(models.Model):
* id\_tipo\_usuario = models.AutoField(primary\_key=True)
* TIPO\_CHOICES = [
* ('Barbero', 'Barbero'),
* ('Cliente', 'Cliente'),
* ('Barbería', 'Barbería'),
* ]
* tipo = models.CharField(max\_length=20, choices=TIPO\_CHOICES)
* def \_\_str\_\_(self):
* return self.tipo
* class EstadoUsuario(models.Model):
* id\_estado\_usuario = models.AutoField(primary\_key=True)
* ESTADO\_CHOICES = [
* ('Activo', 'Activo'),
* ('Bloqueado', 'Bloqueado'),
* ]
* estado\_usuario = models.CharField(max\_length=20, choices=ESTADO\_CHOICES)
* def \_\_str\_\_(self):
* return self.estado\_usuario
* class Usuario(models.Model):
* id\_usuario = models.AutoField(primary\_key=True)
* id\_tipo\_usuario = models.ForeignKey(TipoUsuario, on\_delete=models.CASCADE)
* nombre = models.CharField(max\_length=100)
* apellido = models.CharField(max\_length=100)
* telefono = models.CharField(max\_length=20, null=True, blank=True)
* correo = models.EmailField(null=True, blank=True)
* contraseña\_hash = models.CharField(max\_length=255)
* descripcion\_usuario = models.TextField(null=True, blank=True)
* imagen = models.ImageField(upload\_to='usuarios/', null=True, blank=True)
* id\_estado\_usuario = models.ForeignKey(EstadoUsuario, on\_delete=models.CASCADE)
* def \_\_str\_\_(self):
* return f"{self.nombre} {self.apellido}"
* class SubServicio(models.Model):
* id\_subservicio = models.AutoField(primary\_key=True)
* nombre\_servicio = models.CharField(max\_length=100)
* imagenes = models.ImageField(upload\_to='subservicios/', null=True, blank=True)
* def \_\_str\_\_(self):
* return self.nombre\_servicio
* class Servicio(models.Model):
* id\_servicio = models.AutoField(primary\_key=True)
* id\_usuario = models.ForeignKey(Usuario, on\_delete=models.CASCADE)
* id\_subservicio = models.ForeignKey(SubServicio, on\_delete=models.CASCADE)
* descripcion = models.TextField(null=True, blank=True)
* duracion\_minutos = models.PositiveIntegerField()
* precio = models.FloatField()
* def \_\_str\_\_(self):
* return f"{self.id\_subservicio.nombre\_servicio} - {self.id\_usuario.nombre}"
* class EstadoCita(models.Model):
* id\_estado\_cita = models.AutoField(primary\_key=True)
* ESTADO\_CITA\_CHOICES = [
* ('En progreso', 'En progreso'),
* ('Finalizado', 'Finalizado'),
* ('Cancelado', 'Cancelado'),
* ]
* estado\_cita = models.CharField(max\_length=20, choices=ESTADO\_CITA\_CHOICES)
* def \_\_str\_\_(self):
* return self.estado\_cita
* class Cita(models.Model):
* id\_cita = models.AutoField(primary\_key=True)
* id\_cliente = models.ForeignKey(Usuario, on\_delete=models.CASCADE, related\_name='citas\_cliente')
* id\_servicio = models.ForeignKey(Servicio, on\_delete=models.CASCADE)
* id\_estado\_cita = models.ForeignKey(EstadoCita, on\_delete=models.CASCADE)
* fecha\_cita = models.DateField()
* hora\_cita = models.TimeField()
* def \_\_str\_\_(self):
* return f"Cita {self.id\_cita} - {self.fecha\_cita} {self.hora\_cita}"
* class FormaPago(models.Model):
* id\_forma\_pago = models.AutoField(primary\_key=True)
* FORMA\_PAGO\_CHOICES = [
* ('Efectivo', 'Efectivo'),
* ('Débito', 'Débito'),
* ('Crédito', 'Crédito'),
* ]
* nombre\_forma\_pago = models.CharField(max\_length=20, choices=FORMA\_PAGO\_CHOICES)
* def \_\_str\_\_(self):
* return self.nombre\_forma\_pago
* class Descuento(models.Model):
* id\_descuento = models.AutoField(primary\_key=True)
* id\_usuario = models.ForeignKey(Usuario, on\_delete=models.CASCADE)
* descuento = models.FloatField()
* def \_\_str\_\_(self):
* return f"{self.descuento}% para Usuario {self.id\_usuario}"
* class RegistroPago(models.Model):
* id\_registro\_pago = models.AutoField(primary\_key=True)
* id\_cita = models.ForeignKey(Cita, on\_delete=models.CASCADE)
* id\_forma\_pago = models.ForeignKey(FormaPago, on\_delete=models.CASCADE)
* monto\_original = models.FloatField()
* total\_pagado = models.FloatField()
* id\_descuento = models.ForeignKey(Descuento, on\_delete=models.SET\_NULL, blank=True, null=True)
* fecha\_pago = models.DateTimeField()
* def \_\_str\_\_(self):
* return f"Pago {self.id\_registro\_pago} - Cita {self.id\_cita} - Total monto {self.total\_pagado}"
* class Observacion(models.Model):
* id\_observacion = models.AutoField(primary\_key=True)
* OBSERVACION\_CHOICES = [
* ('Puntualidad', 'Puntualidad'),
* ('Trato', 'Trato'),
* ('Servicio', 'Servicio'),
* ('Barbero', 'Barbero'),
* ]
* nombre\_observacion = models.CharField(max\_length=20, choices=OBSERVACION\_CHOICES)
* def \_\_str\_\_(self):
* return self.nombre\_observacion
* class ValoracionObservacion(models.Model):
* id\_valoracion\_observacion = models.AutoField(primary\_key=True)
* valoracion = models.IntegerField(choices=[(i, str(i)) for i in range(1, 6)])
* id\_observacion = models.ForeignKey(Observacion, on\_delete=models.CASCADE)
* id\_cita = models.ForeignKey(Cita, on\_delete=models.CASCADE)
* def \_\_str\_\_(self):
* return f"Valoración {self.valoracion} - Cita {self.id\_cita}"
* class Bitacora(models.Model):
* id\_bitacora = models.AutoField(primary\_key=True)
* id\_usuario = models.ForeignKey(Usuario, on\_delete=models.CASCADE)
* descripcion\_accion = models.TextField()
* fecha\_bitacora = models.DateTimeField()
* tipo\_accion = models.CharField(max\_length=100)
* def \_\_str\_\_(self):
* return f"Bitácora {self.id\_bitacora} - Usuario {self.id\_usuario}"