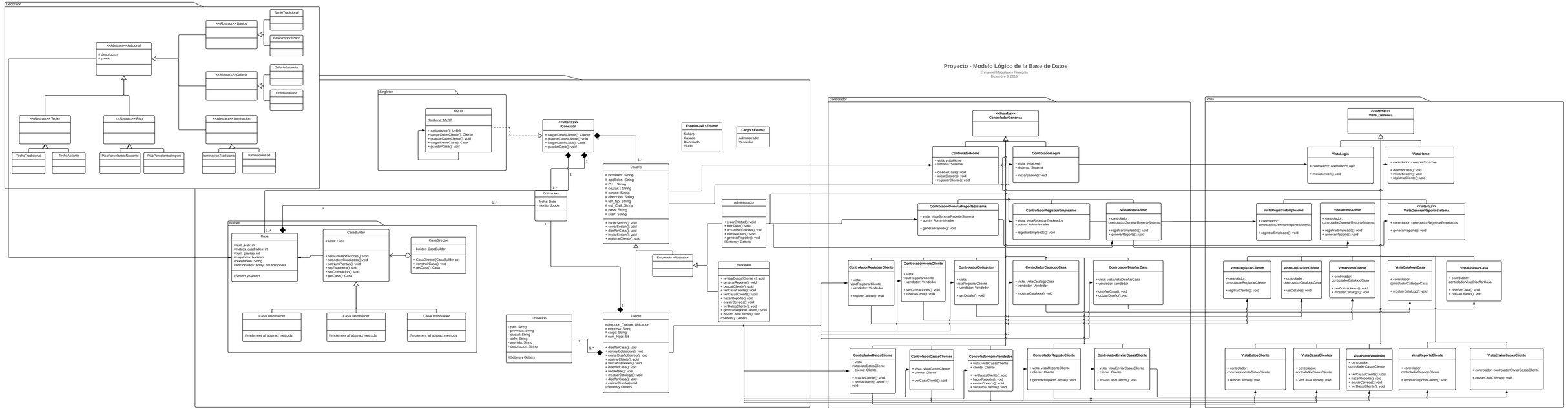
Proyecto - Modelo Lógico de la Base de Datos

Enmanuel Magallanes Pirargote

Diciembre 3, 2019


Decorator
<<Abstract>> Adicional
descripcion
precio
BanoTradicional
<<Abstract>> Banos
BanoInsonorizado
GriferiaEstandar
<<Abstract>> Griferia
GriferiaItaliana
<<Abstract>> Techo
<<Abstract>> Piso
<<Abstract>> Iluminacion
TechoTradicional
TechoAislante
PisoPorcelanatoNacional
PisoPorcelanatoImport
IluminacionTradicional
IluminacionLed
Singleton
MyDB
database: MyDB
+ setInstanciaD(): MyDB
+ cargarDatosCliente(): Cliente
+ guardarDatosCliente(): void
+ cargarDatosCasa(): Casa
+ guardarCasa(): void
<<Interfaz>>
IConexion
+ cargarDatosCliente(): Cliente
+ guardarDatosCliente(): void
+ cargarDatosCasa(): Casa
+ guardarCasa(): void
EstadoCivil <Enum>
Soltero
Casado
Divorciado
Viudo
Cargo <Enum>
Administrador
Vendedor
Usuario
nombres: String
apellidos: String
C.I. : String
celular : String
correo: String
direccion: String
telf_fijo: String
est_Civil: String
pasa: String
user: String
+ iniciarSesion(): void
+ cerraSesion(): void
+ diseñarCasa(): void
+ iniciarSesion(): void
+ registrarCliente(): void
Cotizacion
- fecha: Date
- monto: double
Administrador
+ crearEntidad(): void
+ leerTabla(): void
+ actualizarEntidad(): void
+ eliminarEntidad(): void
+ generarReporte(): void
//Setters y Getters
Empleado <Abstract>
Vendedor
+ revisarDatosCliente c): void
+ generarReporte(): void
+ buscarCliente(): void
+ verCasaCliente(): void
+ verCasasCliente(): void
+ verCotizaciones(): void
+ enviarCorreo(): void
+ verDatosCliente(): void
+ generarReporteCliente(): void
+ enviarCasaCliente(): void
//Setters y Getters
Builder
Casa
#num_Hab: int
#metros_cuadrados: int
#num_plantas: int
#cantidad_bolas: int
#cantidad_Bombillos: int
#adicionales: ArrayList<Adicional>
//Setters y Getters
CasaBuilder
casa: Casa
+ setNumHabitaciones(): void
+ setMetrosCuadrados():void
+ setNumPlantas(): void
+ setEsquema(): void
+ setOrientacion(): void
+ getCasa(): Casa
CasaDirector
builder: CasaBuilder
+ CasaDirector(CasaBuilder cb)
+ construirCasa(): void
+ getCasa(): Casa
CasaOasisBuilder
//Implement all abstract methods
CasaOasisBuilder
//Implement all abstract methods
CasaOasisBuilder
//Implement all abstract methods
Ubicacion
- pais: String
- provincia: String
- ciudad: String
- calle: Sring
- avenida: String
- descripcion: String
//Setters y Getters
Cliente
#direccion_Trabajo: Ubicacion
empresa: String
cargo: String
num_Hijos: bit
+ diseñarCasa(): void
+ revisarCotizacion(): void
+ enviarDiseñoCorreo(): void
+ registrarCliente(): void
+ verCotizaciones(): void
+ diseñarCasa(): void
+ verDetalle(): void
+ mostrarCatalogo(): void
+ verCasaCliente(): void
+ cotizarDiseño(): void
//Setters y Getters
Controlador
<<Interfaz>>
ControladorGenerica
ControladorHome
+ vista: vistaHome
+ sistema: Sistema
+ diseñarCasa(): void
+ iniciarSesion(): void
+ registrarCliente(): void
ControladorLogin
+ vista: vistaLogin
+ sistema: Sistema
+ iniciarSesion(): void
ControladorGenerarReporteSistema
+ vista: vistaGenerarReporteSistema
+ admin: Administrador
+ generarReporte(): void
ControladorRegistrarEmpleados
+ vista: vistaRegistrarEmpleados
+ admin: Administrador
+ registrarEmpleado(): void
VistaHomeAdmin
+ controlador: controladorGenerarReporteSistema
+ registrarEmpleado(): void
+ generarReporte(): void
ControladorRegistrarCliente
+ vista: vistaRegistrarCliente
+ vendedor: Vendedor
+ registrarCliente(): void
ControladorHomeCliente
+ vista: vistaRegistrarCliente
+ vendedor: Vendedor
+ verCotizaciones(): void
+ diseñarCasa(): void
ControladorCotizacion
+ vista: vistaRegistrarCliente
+ vendedor: Vendedor
+ verDetalle(): void
ControladorCatalogoCasa
+ vista: vistaCatalogoCasa
+ vendedor: Vendedor
+ mostrarCatalogo(): void
ControladorDiseñarCasa
+ vista: vistaVistaDiseñarCasa
+ vendedor: Vendedor
+ diseñarCasa(): void
+ cotizarDiseño():void
ControladorDatosCliente
+ vista: vistaVistaDatosCliente
+ cliente: Cliente
+ buscarCliente(): void
+ revisarDatosCliente c): void
ControladorCasasClientes
+ vista: vistaCasasCliente
+ cliente: Cliente
+ verCasaCliente(): void
ControladorHomeVendedor
+ vista: vistaCasasCliente
+ cliente: Cliente
+ verCasasCliente(): void
+ verCotizaciones(): void
+ enviarCorreo(): void
+ verDatosCliente(): void
ControladorReporteCliente
+ vista: vistaReporteCliente
+ cliente: Cliente
+ generarReporteCliente(): void
ControladorEnviarCasasCliente
+ vista: vistaEnviarCasasCliente
+ cliente: Cliente
+ enviarCasaCliente(): void
Vista
<<Interfaz>>
Vista_Generica
VistaLogin
+ controlador: controladorLogin
+ iniciarSesion(): void
VistaHome
+ controlador: controladorHome
+ diseñarCasa(): void
+ iniciarSesion(): void
+ registrarCliente(): void
VistaRegistrarEmpleados
+ controlador: controladorGenerarReporteSistema
+ registrarEmpleado(): void
VistaHomeAdmin
+ controlador: controladorGenerarReporteSistema
+ registrarEmpleado(): void
+ generarReporte(): void
<<Interfaz>>
VistaGenerarReporteSistema
+ controlador: controladorRegistrarEmpleados
+ generarReporte(): void
VistaRegistrarCliente
+ controlador: controladorRegistrarCliente
+ registrarCliente(): void
VistaCotizacionCliente
+ controlador: controladorCatalogoCasa
+ verDetalle(): void
VistaHomeCliente
+ controlador: controladorCatalogoCasa
+ verCotizaciones(): void
+ mostrarCatalogo(): void
VistaCatalogoCasa
+ controlador: controladorCatalogoCasa
+ mostrarCatalogo(): void
VistaDiseñarCasa
+ controlador: controladorVistaDiseñarCasa
+ diseñarCasa(): void
+ cotizarDiseño(): void
VistaDatosCliente
+ controlador: controladorVistaDatosCliente
+ buscarCliente(): void
VistaCasasClientes
+ controlador: controladorCasasCliente
+ verCasaCliente(): void
VistaHomeVendedor
+ controlador: controladorCasasCliente
+ verCasaCliente(): void
+ hacerReporte(): void
+ enviarCorreo(): void
+ verDatosCliente(): void
VistaReporteCliente
+ controlador: controladorReporteCliente
+ generarReporteCliente(): void
VistaEnviarCasasCliente
+ controlador: controladorEnviarCasasCliente
+ enviarCasaCliente(): void
1
1..*
1..*
1
1
1..*
1..*
1..*
1..*
1
1..*
1..*
1..*
1..*