A continuación, se detallará los pasos a realizar para obtener el producto del sistema de facturación móvil de la empresa Noroeste SA.C la cual fue desarrollada en Android.

1. Arquitectura
   1. Servidor de base de datos

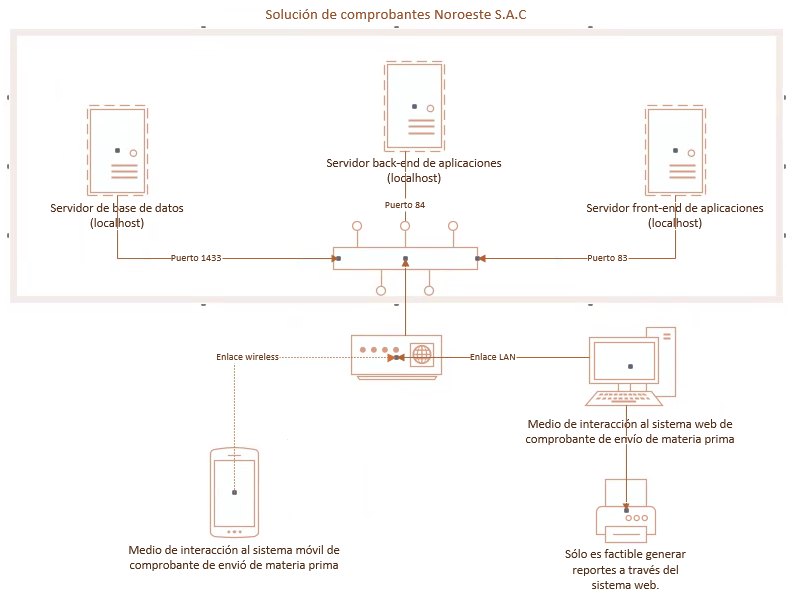
El motor de base de datos es un SQL Express 2014 el cual sale por el puerto 1433 hacia su red interna. Se estableció una IP estática para no ser afectada por el servicio DHCP ya activo en su servidor dedicado.

* 1. Servidor back-end de aplicaciones

El servicio dedicado de back-end cuenta con los webservices (WCF, Restful, otros) de la empresa por lo que para este caso se habilitó el puerto 84 para consumir lo necesario para la solución.

* 1. Servidor front-end de aplicaciones

En ese servidor se habilitó la aplicación web local para la empresa. A futuro se publicará esto a internet.



1. Modelo de la base de datos
   1. Creación de tablas
      1. Tabla de clientes

CREATE TABLE [dbo].[tblCliente](

[ID] [int] IDENTITY(1,1) NOT NULL, [Cliente] [nchar](100) NOT NULL,

PRIMARY KEY CLUSTERED

([ID] ASC) WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY], UNIQUE NONCLUSTERED

([Cliente] ASC) WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]) ON [PRIMARY]

* + 1. Tabla de monedas

CREATE TABLE [dbo].[tblMoneda](

[ID] [nchar](6) NOT NULL, [Moneda] [nvarchar](30) NOT NULL, [Simbolo] [nchar](3) NOT NULL,

PRIMARY KEY CLUSTERED

([ID] ASC) WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]) ON [PRIMARY]

* + 1. Tabla de credenciales

CREATE TABLE [dbo].[tblCredencialPeriferico](

[ID] [nvarchar](128) NOT NULL,

[Clave] [nvarchar](max) NULL,

[Rol] [nvarchar](256) NULL,

[Estado] [bit] NULL DEFAULT ((0)),

[Salt] [nvarchar](250) NULL,

PRIMARY KEY CLUSTERED

([ID] ASC) WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]) ON [PRIMARY] TEXTIMAGE\_ON [PRIMARY]

* + 1. Tabla de tokens

CREATE TABLE [dbo].[tblPerifericoToken](

[ID] [int] IDENTITY(1,1) NOT NULL,

[Token] [nvarchar](250) NOT NULL,

[idUsuario] [nvarchar](128) NOT NULL,

[FechaCreacion] [datetime] NOT NULL,

PRIMARY KEY CLUSTERED

([ID] ASC) WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY], CONSTRAINT [Unique\_Token] UNIQUE NONCLUSTERED

([Token] ASC)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]) ON [PRIMARY]

* + 1. Tabla de usuarios

CREATE TABLE [dbo].[tblUsuario](

[Id] [nvarchar](128) NOT NULL, [ApePaterno] [nvarchar](150) NOT NULL,

[ApeMaterno] [nvarchar](150) NOT NULL, [Nombres] [nvarchar](150) NOT NULL,

[NomCompleto] AS ((((ltrim(rtrim([ApePaterno])) +' ') + ltrim(rtrim([ApeMaterno]))) + ', ') + ltrim(rtrim([Nombres]))), [Sexo] [bit] NOT NULL, CONSTRAINT [PK\_tblUsuario] PRIMARY KEY CLUSTERED

([Id] ASC) WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]) ON [PRIMARY]

* + 1. Tabla de facturas

CREATE TABLE [dbo].[tblFactura](

[ID] [int] IDENTITY(1,1) NOT NULL,

[Com\_Id] AS ('FAC-'+CONVERT([nvarchar](max),[ID])),

[Fecha] [date] NOT NULL,

[idCliente] [int] NULL,

[PuertoOrigen] [nchar](100) NOT NULL,

[PuertoDestino] [nchar](100) NOT NULL,

[Condiciones] [nchar](30) NULL,

[GuiaCarga] [nchar](30) NULL,

[Asegurado] [nchar](30) NULL,

[NumeroPoliza] [nchar](30) NULL,

[Intermediario] [nchar](30) NULL,

[CC] [nchar](15) NULL,

[Flete] [decimal](10, 2) NULL,

[Total] [decimal](10, 2) NULL,

PRIMARY KEY CLUSTERED

([ID] ASC) WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]) ON [PRIMARY]

* + 1. Tabla de detalle factura

CREATE TABLE [dbo].[tblDetalleUsuarioFactura](

[ID] [int] IDENTITY(1,1) NOT NULL,

[idUsuario] [nvarchar](128) NOT NULL,

[idFactura] [int] NOT NULL,

[Bultos] [int] NOT NULL,

[PesoTotal] [nchar](30) NULL,

[Descripcion] [nvarchar](250) NOT NULL,

[idMoneda] [nchar](6) NULL,

[PrecioUnitario] [decimal](10, 2) NOT NULL,

[Subtotal] [decimal](10, 2) NOT NULL,

PRIMARY KEY CLUSTERED

([ID] ASC) WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]) ON [PRIMARY]

* 1. Crear vistas
     1. Crear vista de facturas

SELECT f.ID, f.Com\_Id, f.Fecha, RTRIM(c.Cliente) AS Cliente, RTRIM(f.PuertoOrigen) AS PuertoOrigen, RTRIM(f.PuertoDestino) AS PuertoDestino, ISNULL(RTRIM(f.Condiciones), '-') AS Condiciones, ISNULL(RTRIM(f.GuiaCarga), '-') AS GuiaCarga, ISNULL(RTRIM(f.Asegurado), '-') AS Asegurado, ISNULL(RTRIM(f.NumeroPoliza), '-') AS NumeroPoliza, ISNULL(RTRIM(f.Intermediario), '-') AS Intermediario, ISNULL(RTRIM(f.CC), '-') AS CC, ISNULL(f.Flete, 0) AS Flete, ISNULL(f.Total, 0) AS Total FROM dbo.tblFactura AS f INNER JOIN dbo.tblCliente AS c ON f.idCliente = c.ID

* + 1. Crear vista de los detalles de la factura

SELECT d.ID, d.idFactura, d.Bultos, ISNULL(RTRIM(d.PesoTotal), N'-') AS PesoTotal, RTRIM(d.Descripcion) AS Descripcion, LTRIM({ fn CONCAT(ISNULL(RTRIM(m.Simbolo) + ' ', N' '), CONVERT(nvarchar(MAX),

d.PrecioUnitario)) }) AS PrecioUnitario, LTRIM({ fn CONCAT(ISNULL(RTRIM(m.Simbolo) + ' ', N' '), CONVERT(nvarchar(MAX), d.Subtotal)) }) AS Subtotal

FROM dbo.tblDetalleUsuarioFactura AS d LEFT OUTER JOIN

dbo.tblMoneda AS m ON d.idMoneda = m.ID

1. Modelo del servicio web
   1. Creación del modelo de servicios
      1. ServiceCliente.svc

public class ServiceCliente : IServiceCliente

{

public bool create(tblCliente model)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

try

{

//el atributo ID se autogenera

ClienteEntity fa = new ClienteEntity();

fa.Cliente = model.Cliente;

bd.tblCliente.Add(fa);

bd.SaveChanges();

return true;

}

catch (Exception)

{

return false;

}

}

}

public bool delete(string id)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

try

{

int \_id = Int32.Parse(id);

ClienteEntity pe = bd.tblCliente.Single(p => p.ID == \_id);

bd.tblCliente.Remove(pe);

return (bd.SaveChanges() != 0);

}

catch (Exception)

{

return false;

}

}

}

public bool edit(tblCliente model)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

try

{

ClienteEntity fa = bd.tblCliente.Single(x => x.ID == model.ID);

fa.Cliente = model.Cliente;

bd.tblCliente.Attach(fa);

bd.Entry(fa).State = EntityState.Modified;

return (bd.SaveChanges() != 0);

}

catch (Exception)

{

return false;

}

}

}

public tblCliente find(string id)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

int \_id = Convert.ToInt32(id);

int \_cant = obtenerCantFacturas(bd, \_id);

tblCliente model = bd.tblCliente.Where(x => x.ID == \_id).Select(fac => new tblCliente

{

ID = fac.ID,

Cliente = fac.Cliente.Trim(),

\_Facturas = \_cant

}).FirstOrDefault();

return model;

}

}

public List<tblCliente> findAll()

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

List<tblCliente> model = bd.tblCliente.ToList().Select(fac => new tblCliente

{

ID = fac.ID,

Cliente = find(fac.ID.ToString()).Cliente,

\_Facturas = find(fac.ID.ToString()).\_Facturas

}).ToList();

return model;

}

}

private int obtenerCantFacturas(bdNoroesteEntities bd, int \_id)

{

return bd.tblFactura.ToList()

.Count(x => x.idCliente != null && x.idCliente == \_id);

}

}

* + 1. ServiceDetalleFactura.svc

public class ServiceDetalleFactura : IServiceDetalleFactura

{

public bool create(tblDetalleUsuarioFactura model)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

try

{

DetalleUsuarioFacturaEntity fa = new DetalleUsuarioFacturaEntity();

fa.idUsuario = model.idUsuario;

fa.idFactura = model.idFactura;

fa.Bultos = model.Bultos;

fa.PesoTotal = model.PesoTotal;

fa.Descripcion = model.Descripcion;

fa.idMoneda = model.idMoneda;

fa.PrecioUnitario = model.PrecioUnitario;

fa.Subtotal = model.Subtotal;

bd.tblDetalleUsuarioFactura.Add(fa);

return (bd.SaveChanges() != 0);

}

catch (Exception)

{

return false;

}

}

}

public bool delete(string id)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

try

{

int \_id = Int32.Parse(id);

DetalleUsuarioFacturaEntity pe = bd.tblDetalleUsuarioFactura.SingleOrDefault(p => p.ID == \_id);

bd.tblDetalleUsuarioFactura.Remove(pe);

return (bd.SaveChanges() != 0);

}

catch (Exception ex)

{

throw new FaultException("Error: " + ex.Message, new FaultCode("ServiceDetalleFactura\_DELETE\_Error"));

}

}

}

public bool edit(tblDetalleUsuarioFactura model)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

try

{

DetalleUsuarioFacturaEntity fa = bd.tblDetalleUsuarioFactura.Single(x => x.ID == model.ID);

fa.idUsuario = model.idUsuario;

fa.idFactura = model.idFactura;

fa.Bultos = model.Bultos;

fa.PesoTotal = model.PesoTotal;

fa.Descripcion = model.Descripcion;

fa.idMoneda = model.idMoneda;

fa.PrecioUnitario = model.PrecioUnitario;

fa.Subtotal = model.Subtotal;

bd.tblDetalleUsuarioFactura.Attach(fa);

bd.Entry(fa).State = EntityState.Modified;

return (bd.SaveChanges() != 0);

}

catch (Exception ex)

{

throw new FaultException("Error: " + ex.Message, new FaultCode("ServiceDetalleFactura\_PUT\_Error"));

}

}

}

public tblDetalleUsuarioFactura find(string id)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

int \_id = Convert.ToInt32(id);

return bd.tblDetalleUsuarioFactura.Where(x => x.ID == \_id).Select(fac => new tblDetalleUsuarioFactura

{

ID = fac.ID,

idUsuario = fac.idUsuario,

idFactura = fac.idFactura,

Bultos = fac.Bultos,

PesoTotal = string.IsNullOrEmpty(fac.PesoTotal) ? null : fac.PesoTotal.Trim(),

Descripcion = fac.Descripcion,

idMoneda = string.IsNullOrEmpty(fac.idMoneda) ? null : fac.idMoneda.Trim(),

PrecioUnitario = fac.PrecioUnitario,

Subtotal = fac.Subtotal

}).First();

}

}

public List<tblDetalleUsuarioFactura> findAll()

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

List<tblDetalleUsuarioFactura> model = bd.tblDetalleUsuarioFactura.ToList().OrderBy(x => x.ID).Select(fac => new tblDetalleUsuarioFactura

{

ID = fac.ID,

idUsuario = find(fac.ID.ToString()).idUsuario,

idFactura = find(fac.ID.ToString()).idFactura,

Bultos = find(fac.ID.ToString()).Bultos,

PesoTotal = find(fac.ID.ToString()).PesoTotal,

Descripcion = find(fac.ID.ToString()).Descripcion,

idMoneda = find(fac.ID.ToString()).idMoneda,

PrecioUnitario = find(fac.ID.ToString()).PrecioUnitario,

Subtotal = find(fac.ID.ToString()).Subtotal

}).ToList();

return model;

}

}

public List<tblDetalleUsuarioFactura> findByFactura(string idFactura)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

int \_idFactura = Convert.ToInt32(idFactura);

return bd.tblDetalleUsuarioFactura.Where(x => x.idFactura == \_idFactura).Select(fac => new tblDetalleUsuarioFactura

{

ID = fac.ID,

idUsuario = fac.idUsuario,

idFactura = fac.idFactura,

Bultos = fac.Bultos,

PesoTotal = string.IsNullOrEmpty(fac.PesoTotal) ? null : fac.PesoTotal.Trim(),

Descripcion = fac.Descripcion,

idMoneda = string.IsNullOrEmpty(fac.idMoneda) ? null : fac.idMoneda.Trim(),

PrecioUnitario = fac.PrecioUnitario,

Subtotal = fac.Subtotal

}).ToList();

}

}

}

* + 1. ServiceFactura.svc

public class ServiceFactura : IServiceFactura

{

public bool create(tblFactura model)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

try

{

FacturaEntity fa = new FacturaEntity();

fa.Asegurado = string.IsNullOrEmpty(model.Asegurado) ? null : model.Asegurado.Trim();

fa.CC = string.IsNullOrEmpty(model.CC) ? null : model.CC.Trim();

fa.Condiciones = string.IsNullOrEmpty(model.Condiciones) ? null : model.Condiciones.Trim();

fa.Fecha = DateTime.Parse(model.Fecha);

fa.Flete = model.Flete;

fa.GuiaCarga = string.IsNullOrEmpty(model.GuiaCarga) ? null : model.GuiaCarga.Trim();

fa.idCliente = model.idCliente;

fa.Intermediario = string.IsNullOrEmpty(model.Intermediario) ? null : model.Intermediario.Trim();

fa.NumeroPoliza = string.IsNullOrEmpty(model.NumeroPoliza) ? null : model.NumeroPoliza.Trim();

fa.PuertoDestino = string.IsNullOrEmpty(model.PuertoDestino) ? null : model.PuertoDestino.Trim();

fa.PuertoOrigen = string.IsNullOrEmpty(model.PuertoOrigen) ? null : model.PuertoOrigen.Trim();

fa.Total = model.Total;

bd.tblFactura.Add(fa);

bd.SaveChanges();

return true;

}

catch (Exception)

{

return false;

}

}

}

public bool delete(string id)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

try

{

int \_id = Int32.Parse(id);

FacturaEntity pe = bd.tblFactura.SingleOrDefault(p => p.ID == \_id);

List<DetalleUsuarioFacturaEntity> de = bd.tblDetalleUsuarioFactura.Where(x => x.idFactura == pe.ID).ToList();

if (de != null) {

for (int i = 0; i < de.Count(); i++)

{

bd.tblDetalleUsuarioFactura.Remove(de[i]);

}

}

bd.tblFactura.Remove(pe);

return (bd.SaveChanges() != 0);

}

catch (Exception ex)

{

throw new FaultException("Error: " + ex.Message, new FaultCode("ServiceFactura\_DELETE\_Error"));

}

}

}

//validar cuando ya tiene un total; sino, que agregue el que yo seteo

public bool edit(tblFactura model)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

try

{

FacturaEntity fa = bd.tblFactura.Single(x => x.ID == model.ID);

Decimal \_total = bd.tblDetalleUsuarioFactura.ToList().Where(x => x.idFactura == fa.ID).Sum(x => x.Subtotal);

fa.Asegurado = model.Asegurado;

fa.CC = model.CC;

fa.Condiciones = model.Condiciones;

fa.Fecha = DateTime.Parse(model.Fecha);

fa.Flete = model.Flete;

fa.GuiaCarga = model.GuiaCarga;

fa.idCliente = model.idCliente;

fa.Intermediario = model.Intermediario;

fa.NumeroPoliza = model.NumeroPoliza;

fa.PuertoDestino = model.PuertoDestino;

fa.PuertoOrigen = model.PuertoOrigen;

fa.Total = model.Total.HasValue ? model.Total : \_total;

bd.tblFactura.Attach(fa);

bd.Entry(fa).State = EntityState.Modified;

return (bd.SaveChanges() != 0);

}

catch (Exception ex)

{

throw new FaultException("Error: " + ex.Message, new FaultCode("ServiceFactura\_PUT\_Error"));

}

}

}

public tblFactura find(string id)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

int \_id = Convert.ToInt32(id);

string \_tipo = calcularSimbolo(bd, \_id);

tblFactura model = bd.tblFactura.Where(x => x.ID == \_id).Select(fac => new tblFactura

{

ID = fac.ID,

Com\_Id = fac.Com\_Id,

Asegurado = string.IsNullOrEmpty(fac.Asegurado) ? null : fac.Asegurado.Trim(),

CC = string.IsNullOrEmpty(fac.CC) ? null : fac.CC.Trim(),

Condiciones = string.IsNullOrEmpty(fac.Condiciones) ? null : fac.Condiciones.Trim(),

Fecha = fac.Fecha.ToString(),

Flete = fac.Flete,

GuiaCarga = string.IsNullOrEmpty(fac.GuiaCarga) ? null : fac.GuiaCarga.Trim(),

idCliente = fac.idCliente,

Intermediario = string.IsNullOrEmpty(fac.Intermediario) ? null : fac.Intermediario.Trim(),

NumeroPoliza = string.IsNullOrEmpty(fac.NumeroPoliza) ? null : fac.NumeroPoliza.Trim(),

PuertoDestino = string.IsNullOrEmpty(fac.PuertoDestino) ? null : fac.PuertoDestino.Trim(),

PuertoOrigen = string.IsNullOrEmpty(fac.PuertoOrigen) ? null : fac.PuertoOrigen.Trim(),

Total = fac.Total,

Moneda = \_tipo

}).FirstOrDefault();

return model;

}

}

public List<tblFactura> findAll()

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

List<tblFactura> model = bd.tblFactura.ToList().OrderBy(x => x.ID).Select(fac => new tblFactura

{

ID = fac.ID,

Com\_Id = find(fac.ID.ToString()).Com\_Id,

Asegurado = find(fac.ID.ToString()).Asegurado,

CC = find(fac.ID.ToString()).CC,

Condiciones = find(fac.ID.ToString()).Condiciones,

Fecha = find(fac.ID.ToString()).Fecha,

Flete = find(fac.ID.ToString()).Flete,

GuiaCarga = find(fac.ID.ToString()).GuiaCarga,

idCliente = find(fac.ID.ToString()).idCliente,

Intermediario = find(fac.ID.ToString()).Intermediario,

NumeroPoliza = find(fac.ID.ToString()).NumeroPoliza,

PuertoDestino = find(fac.ID.ToString()).PuertoDestino,

PuertoOrigen = find(fac.ID.ToString()).PuertoOrigen,

Total = find(fac.ID.ToString()).Total.HasValue ? find(fac.ID.ToString()).Total.Value : 0,

Moneda = find(fac.ID.ToString()).Moneda

}).ToList();

return model;

}

}

public tblFactura findByModel(string condiciones, string fecha, string guiacarga, string puertodestino, string puertoorigen, string idcliente)

{

tblFactura model = new tblFactura();

model.Condiciones = condiciones;

model.Fecha = fecha;

model.GuiaCarga = guiacarga;

model.PuertoDestino = puertodestino;

model.PuertoOrigen = puertoorigen;

if (!string.IsNullOrEmpty(idcliente)) model.idCliente = Int32.Parse(idcliente);

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

tblFactura newModel = bd.tblFactura.Where(x =>

x.Condiciones == model.Condiciones &&

x.Fecha.ToString() == model.Fecha &&

x.GuiaCarga == model.GuiaCarga &&

x.PuertoDestino == model.PuertoDestino &&

x.PuertoOrigen == model.PuertoOrigen &&

x.idCliente == model.idCliente

).Select(x => new tblFactura()

{

ID = x.ID,

Com\_Id = x.Com\_Id,

Asegurado = model.Asegurado,

CC = model.CC,

Condiciones = model.Condiciones,

Fecha = model.Fecha,

Flete = model.Flete,

GuiaCarga = model.GuiaCarga,

idCliente = model.idCliente,

Intermediario = model.Intermediario,

NumeroPoliza = model.NumeroPoliza,

PuertoDestino = model.PuertoDestino,

PuertoOrigen = model.PuertoOrigen,

}).FirstOrDefault();

return newModel;

}

}

private string calcularSimbolo(bdNoroesteEntities bd, int \_id)

{

string simbol = null;

//obtener la lista de strings de la tbldetalle

List<string> lista = new List<string>();

var listaTemp = bd.tblDetalleUsuarioFactura.ToList().Where(x => x.idFactura == \_id).ToList();

foreach (var item in listaTemp)

{

lista.Add((string.IsNullOrEmpty(item.idMoneda) ? null : item.tblMoneda.Simbolo.Trim()));

}

lista = lista.Distinct().ToList();

if (lista.Any(o => o != lista[0].Trim()))

simbol = null;

else simbol = (lista.FirstOrDefault() != null ? lista.First().Trim() : null);

return simbol;

}

}

* + 1. ServiceMoneda.svc

public class ServiceMoneda : IServiceMoneda

{

public bool create(tblMoneda model)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

tblMoneda mo = new tblMoneda();

try

{

mo.ID = model.ID;

mo.Moneda = model.Moneda;

mo.Simbolo = model.Simbolo;

bd.tblMoneda.Add(new MonedaEntity() { ID = mo.ID, Moneda = mo.Moneda, Simbolo = mo.Simbolo });

bd.SaveChanges();

return true;

}

catch (Exception)

{

return false;

}

}

}

public bool delete(string id)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

try

{

MonedaEntity pe = bd.tblMoneda.Single(p => p.ID == id);

bd.tblMoneda.Remove(pe);

return (bd.SaveChanges() != 0);

}

catch (Exception)

{

return false;

}

}

}

public bool edit(tblMoneda model)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

try

{

MonedaEntity ma = bd.tblMoneda.Single(x => x.ID.Trim() == model.ID.Trim());

ma.Moneda = model.Moneda;

ma.Simbolo = model.Simbolo;

bd.tblMoneda.Attach(ma);

bd.Entry(ma).State = EntityState.Modified;

return (bd.SaveChanges() != 0);

}

catch (Exception)

{

return false;

}

}

}

public tblMoneda find(string id)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

return bd.tblMoneda.Where(x => x.ID == id).Select(fac => new tblMoneda

{

ID = fac.ID.Trim(),

Moneda = fac.Moneda.Trim(),

Simbolo = string.IsNullOrEmpty(fac.Simbolo) ? null : fac.Simbolo.Trim()

}).First();

}

}

public List<tblMoneda> findAll()

{

var token = HttpContext.Current.Request.Headers["Token"]; // This works whether aspNetCompatibilityEnabled is true of false.

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

ITokenValidator validator = new DatabaseTokenValidator(bd);

//NO puedes regresar un null en una clase que te pide devolver una lista

List<tblMoneda> lista = null;

if (validator.IsValid(token))

{

lista = bd.tblMoneda.Select(fac => new tblMoneda

{

ID = fac.ID.Trim(),

Moneda = fac.Moneda.Trim(),

Simbolo = string.IsNullOrEmpty(fac.Simbolo) ? null : fac.Simbolo.Trim()

}).ToList();

}

else throw new FaultException("Error de autenticación por token.", new FaultCode("TokenFault"));

return lista;

}

}

}

* + 1. ServiceUsuario.svc

public class ServiceUsuario : IServiceUsuario

{

public string Authenticate(Permisson creds)

{

if (creds == null && WebOperationContext.Current != null)

{

creds = new BasicAuth(WebOperationContext.Current.IncomingRequest.Headers["Authorization"]).Creds;

}

using (var dbContext = new bdNoroesteEntities())

{

return new DatabaseTokenBuilder(dbContext).Build(creds);

}

}

public bool ValidateToken(TokenInformation model)

{

bool rpt = false;

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

ITokenValidator validator = new DatabaseTokenValidator(bd);

rpt = validator.IsValid(model.token);

}

return rpt;

}

public bool create(tblUsuario model)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

try

{

UsuarioEntity fac = new UsuarioEntity();

fac.Id = model.Id;

fac.ApeMaterno = model.ApeMaterno;

fac.ApePaterno = model.ApePaterno;

fac.Nombres = model.Nombres;

fac.Sexo = model.Sexo;

bd.tblUsuario.Add(fac);

bd.SaveChanges();

return true;

}

catch (Exception)

{

return false;

}

}

}

public bool delete(tblUsuario model)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

try

{

UsuarioEntity pe = bd.tblUsuario.Single(p => p.Id == model.Id);

bd.tblUsuario.Remove(pe);

bd.SaveChanges();

return true;

}

catch (Exception)

{

return false;

}

}

}

public bool edit(tblUsuario model)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

try

{

UsuarioEntity fac = bd.tblUsuario.Single(x => x.Id == model.Id);

fac.ApeMaterno = model.ApeMaterno;

fac.ApePaterno = model.ApePaterno;

fac.Nombres = model.Nombres;

fac.Sexo = model.Sexo;

bd.tblUsuario.Attach(fac);

bd.Entry(fac).State = EntityState.Modified;

return (bd.SaveChanges() != 0);

}

catch (Exception)

{

return false;

}

}

}

public tblUsuario find(string mail)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

return bd.tblUsuario.Where(x => x.AspNetUsers.Email == mail).Select(fac => new tblUsuario

{

Id = fac.Id,

ApeMaterno = fac.ApeMaterno,

ApePaterno = fac.ApePaterno,

Nombres = fac.Nombres,

NomCompleto = fac.NomCompleto,

Sexo = fac.Sexo,

Email = fac.AspNetUsers.Email

}).First();

}

}

public string findIdByMail(string mail)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

return bd.tblUsuario.FirstOrDefault(x => x.AspNetUsers.Email == mail).Id;

}

}

public List<tblUsuario> findAll()

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

return bd.tblUsuario.Select(fac => new tblUsuario

{

Id = fac.Id,

ApeMaterno = fac.ApeMaterno,

ApePaterno = fac.ApePaterno,

Nombres = fac.Nombres,

NomCompleto = fac.NomCompleto,

Sexo = fac.Sexo,

Email = fac.AspNetUsers.Email

}).ToList();

}

}

public string decryptPassword(string clave)

{

try

{

return Utils.Codificador.StrongDecrypt(clave);

}

catch (Exception)

{

return null;

}

}

public string encryptPassword(string clave)

{

try

{

return Utils.Codificador.StrongEncrypt(clave);

}

catch (Exception)

{ return null; } }

/// <summary>

/// Validate if email is a valid input and exists into AspNetUsers/Usuario table

/// </summary>

/// <param name="cuenta">Email</param>

public valCredential existsCredential(string cuenta)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

valCredential model = new valCredential();

if (IsValidEmail(cuenta) && bd.AspNetUsers.ToList().Any(x => x.Email == cuenta))

{

model.Existe = true;

model.NomCompleto = bd.AspNetUsers.Single(x => x.Email == cuenta).tblUsuario.NomCompleto;

}

else

{

model.Existe = false;

model.NomCompleto = null;

}

return model;

}

}

public string hasPassword(string cuenta)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

if (IsValidEmail(cuenta) && bd.AspNetUsers.ToList().Any(x => x.Email == cuenta))

{

var \_id = bd.AspNetUsers.ToList().Single(x => x.Email == cuenta).Id;

if (bd.tblCredencialPeriferico.ToList().Any(x => x.ID == \_id))

{

string \_clave = bd.tblCredencialPeriferico.ToList().Single(x => x.ID == \_id).Clave;

if (!string.IsNullOrEmpty(\_clave))

{

return "SI";

}

else

{

//La cuenta existe en la BD tblCredenciales pero no tiene una clave asignada

return "NO";

}

}

//Si la cuenta existe como parte del sistema pero no tiene una clave

return "-";

}

//Si la cuenta no existe

return null;

}

}

public bool asignCredential(Credentials model)

{

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

if (bd.AspNetUsers.ToList().Any(x => x.Email == model.Cuenta))

{

try

{

string \_id = bd.AspNetUsers.ToList().Single(x => x.Email == model.Cuenta).Id;

CredencialPerifericoEntity fac = bd.tblCredencialPeriferico.Single(x => x.ID == \_id);

fac.Clave = encryptPassword(model.Clave);

fac.Estado = true;

bd.SaveChanges();

return true;

}

catch (Exception)

{

return false;

}

}

else

{ return false; }

}

}

#endregion

private bool IsValidEmail(string email)

{

try

{

var addr = new System.Net.Mail.MailAddress(email);

return addr.Address == email;

}

catch

{

return false;

}

}

public credentialsResponse validateCredentials(string cuenta, string clave)

{

Credentials c = new Credentials();

c.Cuenta = cuenta;

c.Clave = encryptPassword(clave);

using (bdNoroesteEntities bd = new bdNoroesteEntities())

{

credentialsResponse response = new credentialsResponse();

response.Correcto = false;

response.Rol = null;

if (bd.tblUsuario.ToList().Any(x => x.AspNetUsers.Email == c.Cuenta))

{

var valid = bd.tblCredencialPeriferico.ToList().SingleOrDefault(x => x.tblUsuario.AspNetUsers.Email == c.Cuenta && x.Clave == c.Clave);

if (valid != null)

{

response.Correcto = true;

response.Rol = valid.Rol;

}

}

return response;

}

}

public bool ActualizarClaves(string ID, string oldClave, string newClave)

{

using (bdNoroesteEntities db = new bdNoroesteEntities())

{

if (db.tblCredencialPeriferico.ToList().Any(x => x.ID == ID))

{

string enc\_oldClave = encryptPassword(oldClave);

var currentPass = db.tblCredencialPeriferico.ToList().Single(x => x.ID == ID).Clave;

if (currentPass == enc\_oldClave)

{

CredencialPerifericoEntity fa = db.tblCredencialPeriferico.Single(x => x.ID == ID);

fa.Clave = encryptPassword(newClave);

db.tblCredencialPeriferico.Attach(fa);

db.Entry(fa).State = EntityState.Modified;

return (db.SaveChanges() != 0);

}

}

}

return false;

}

}

* 1. Creación de las url de los servicios

<services>

<service name="WcfFacturacion.Services.\_Factura.ServiceFactura" behaviorConfiguration="WcfFacturacion\_Behavior">

<endpoint address="" binding="webHttpBinding" contract="WcfFacturacion.Services.\_Factura.IServiceFactura"></endpoint>

</service>

<service name="WcfFacturacion.Services.\_Cliente.ServiceCliente" behaviorConfiguration="WcfFacturacion\_Behavior">

<endpoint address="" binding="webHttpBinding" contract="WcfFacturacion.Services.\_Cliente.IServiceCliente"></endpoint>

</service>

<service name="WcfFacturacion.Services.\_Moneda.ServiceMoneda" behaviorConfiguration="WcfFacturacion\_Behavior">

<endpoint address="" binding="webHttpBinding" contract="WcfFacturacion.Services.\_Moneda.IServiceMoneda"></endpoint>

</service>

<service name="WcfFacturacion.Services.\_Usuario.ServiceUsuario" behaviorConfiguration="WcfFacturacion\_Behavior">

<endpoint address="" binding="webHttpBinding" contract="WcfFacturacion.Services.\_Usuario.IServiceUsuario"></endpoint>

</service>

<service name="WcfFacturacion.Services.\_DetalleUsuarioFactura.ServiceDetalleFactura" behaviorConfiguration="WcfFacturacion\_Behavior">

<endpoint address="" binding="webHttpBinding" contract="WcfFacturacion.Services.\_DetalleUsuarioFactura.IServiceDetalleFactura"></endpoint>

</service>

<service name="WcfFacturacion.Services.ServiceReporte" behaviorConfiguration="WcfFacturacion\_Behavior">

<endpoint address="" binding="webHttpBinding" contract="WcfFacturacion.Services.IServiceReporte"></endpoint>

</service>

</services>

* 1. Creación de métodos de los servicios
     1. IServiceCliente

public interface IServiceCliente

{

[OperationContract]

[WebInvoke(Method = "GET", UriTemplate = "findall", ResponseFormat = WebMessageFormat.Json)]

List<tblCliente> findAll();

[OperationContract]

[WebInvoke(Method = "GET", UriTemplate = "find/{id}", ResponseFormat = WebMessageFormat.Json)]

tblCliente find(string id);

[OperationContract]

[WebInvoke(Method = "POST", UriTemplate = "create", ResponseFormat = WebMessageFormat.Json, RequestFormat = WebMessageFormat.Json)]

bool create(tblCliente model);

[OperationContract]

[WebInvoke(Method = "PUT", UriTemplate = "edit", ResponseFormat = WebMessageFormat.Json, RequestFormat = WebMessageFormat.Json)]

bool edit(tblCliente model);

[OperationContract]

[WebInvoke(Method = "DELETE", UriTemplate = "delete/{id}", ResponseFormat = WebMessageFormat.Json)]

bool delete(string id);

}

* + 1. IServiceDetalleFactura

public interface IServiceDetalleFactura

{

[OperationContract]

[WebInvoke(Method = "GET", UriTemplate = "findall", ResponseFormat = WebMessageFormat.Json)]

List<tblDetalleUsuarioFactura> findAll();

/// <summary>

/// Service para encontrar la lista de detalles por factura generada

/// </summary>

/// <param name="idFactura">ID de la factura</param>

/// <returns>lista de elementos</returns>

[OperationContract]

[WebInvoke(Method = "GET", UriTemplate = "findbyfactura/{idFactura}", ResponseFormat = WebMessageFormat.Json, BodyStyle = WebMessageBodyStyle.Bare)]

List<tblDetalleUsuarioFactura> findByFactura(string idFactura);

/// <param name="id">ID del detalle</param>

/// <returns>lista de elementos</returns>

[OperationContract]

[WebInvoke(Method = "GET", UriTemplate = "find/{id}", ResponseFormat = WebMessageFormat.Json)]

tblDetalleUsuarioFactura find(string id);

[OperationContract]

[WebInvoke(Method = "POST", UriTemplate = "create", ResponseFormat = WebMessageFormat.Json, RequestFormat = WebMessageFormat.Json)]

bool create(tblDetalleUsuarioFactura model);

[OperationContract]

[WebInvoke(Method = "PUT", UriTemplate = "edit", ResponseFormat = WebMessageFormat.Json, RequestFormat = WebMessageFormat.Json)]

bool edit(tblDetalleUsuarioFactura model);

[OperationContract]

[WebInvoke(Method = "DELETE", UriTemplate = "delete/{id}", ResponseFormat = WebMessageFormat.Json, RequestFormat = WebMessageFormat.Json, BodyStyle = WebMessageBodyStyle.Bare)]

bool delete(string id);

}

* + 1. IServiceFactura

public interface IServiceFactura

{

[OperationContract]

[WebInvoke(Method = "GET", UriTemplate ="findall", ResponseFormat = WebMessageFormat.Json)]

List<tblFactura> findAll();

[OperationContract]

[WebInvoke(Method = "GET", UriTemplate = "find/{id}", ResponseFormat = WebMessageFormat.Json)]

tblFactura find(string id);

[OperationContract]

[WebInvoke(Method = "GET", UriTemplate = "findbymodel/{condiciones}/{fecha}/{guiacarga}/{puertodestino}/{puertoorigen}/{idcliente}", ResponseFormat = WebMessageFormat.Json)]

tblFactura findByModel(string condiciones, string fecha, string guiacarga, string puertodestino, string puertoorigen, string idcliente );

[OperationContract]

[WebInvoke(Method = "POST", UriTemplate = "create", ResponseFormat = WebMessageFormat.Json, RequestFormat = WebMessageFormat.Json)]

bool create(tblFactura model);

[OperationContract]

[WebInvoke(Method = "PUT", UriTemplate = "edit", ResponseFormat = WebMessageFormat.Json, RequestFormat = WebMessageFormat.Json)]

bool edit(tblFactura model);

[OperationContract]

[WebInvoke(Method = "DELETE", UriTemplate = "delete/{id}", ResponseFormat = WebMessageFormat.Json, RequestFormat = WebMessageFormat.Json, BodyStyle = WebMessageBodyStyle.Bare)]

bool delete(string id);

}

* + 1. IServiceMoneda

public interface IServiceMoneda

{

[OperationContract]

[WebInvoke(Method = "GET", UriTemplate = "findall", ResponseFormat = WebMessageFormat.Json)]

List<tblMoneda> findAll();

[OperationContract]

[WebInvoke(Method = "GET", UriTemplate = "find/{id}", ResponseFormat = WebMessageFormat.Json)]

tblMoneda find(string id);

[OperationContract]

[WebInvoke(Method = "POST", UriTemplate = "create", ResponseFormat = WebMessageFormat.Json, RequestFormat = WebMessageFormat.Json)]

bool create(tblMoneda model);

[OperationContract]

[WebInvoke(Method = "PUT", UriTemplate = "edit", ResponseFormat = WebMessageFormat.Json, RequestFormat = WebMessageFormat.Json)]

bool edit(tblMoneda model);

[OperationContract]

[WebInvoke(Method = "DELETE", UriTemplate = "delete/{id}", ResponseFormat = WebMessageFormat.Json)]

bool delete(string id);

}

* + 1. IServiceUsuario

public interface IServiceUsuario

{

[WebInvoke(Method = "POST", RequestFormat = WebMessageFormat.Json, ResponseFormat = WebMessageFormat.Json, BodyStyle = WebMessageBodyStyle.Bare)]

[OperationContract]

string Authenticate(Permisson creds);

[WebInvoke(Method = "POST", RequestFormat = WebMessageFormat.Json, ResponseFormat = WebMessageFormat.Json, BodyStyle = WebMessageBodyStyle.Bare)]

[OperationContract]

bool ValidateToken (TokenInformation model);

[OperationContract]

[WebInvoke(Method = "GET", UriTemplate = "findall", ResponseFormat = WebMessageFormat.Json)]

List<tblUsuario> findAll();

[OperationContract]

[WebInvoke(Method = "GET", UriTemplate = "find/{mail}", ResponseFormat = WebMessageFormat.Json)]

tblUsuario find(string mail);

[OperationContract]

[WebInvoke(Method = "GET", UriTemplate = "findidbymail/{mail}", ResponseFormat = WebMessageFormat.Json, BodyStyle = WebMessageBodyStyle.Bare)]

string findIdByMail(string mail);

[OperationContract]

[WebInvoke(Method = "POST", UriTemplate = "create", ResponseFormat = WebMessageFormat.Json, RequestFormat = WebMessageFormat.Json)]

bool create(tblUsuario model);

[OperationContract]

[WebInvoke(Method = "PUT", UriTemplate = "edit", ResponseFormat = WebMessageFormat.Json, RequestFormat = WebMessageFormat.Json, BodyStyle = WebMessageBodyStyle.Bare)]

bool edit(tblUsuario model);

[OperationContract]

[WebInvoke(Method = "DELETE", UriTemplate = "delete", ResponseFormat = WebMessageFormat.Json, RequestFormat = WebMessageFormat.Json)]

bool delete(tblUsuario model);

//Por motivo de pruebas estará en GET de forma temporal

[OperationContract]

[WebInvoke(Method = "GET", UriTemplate = "encryptPassword/{clave}", ResponseFormat = WebMessageFormat.Json)]

string encryptPassword(string clave);

//Por motivo de pruebas estará en GET de forma temporal

[OperationContract]

[WebInvoke(Method = "GET", UriTemplate = "decryptPassword/{clave}", ResponseFormat = WebMessageFormat.Json)]

string decryptPassword(string clave);

[OperationContract]

[WebInvoke(Method = "GET", UriTemplate = "existsCredential/{cuenta}", ResponseFormat = WebMessageFormat.Json)]

valCredential existsCredential(string cuenta);

[OperationContract]

[WebInvoke(Method = "GET", UriTemplate = "hasPassword/{cuenta}", ResponseFormat = WebMessageFormat.Json)]

string hasPassword(string cuenta);

[OperationContract]

[WebInvoke(Method = "GET", UriTemplate = "validateCredentials/{cuenta}/{clave}", ResponseFormat = WebMessageFormat.Json)]

credentialsResponse validateCredentials(string cuenta, string clave);

[OperationContract]

[WebInvoke(Method = "PUT", UriTemplate = "asignCredential", ResponseFormat = WebMessageFormat.Json, RequestFormat = WebMessageFormat.Json)]

bool asignCredential(Credentials model);

[OperationContract]

[WebInvoke(Method = "PUT", UriTemplate = "updatePassword/{ID}/{oldClave}/{newClave}", ResponseFormat = WebMessageFormat.Json, BodyStyle = WebMessageBodyStyle.Bare)]

bool ActualizarClaves(string ID, string oldClave, string newClave);

}

1. Modelo lógico de la aplicación móvil
   1. Diseño y generación de lógica de las interfaces

* Login
  + Activity

**public class** LoginActivity **extends** AppCompatActivity **implements** android.view.View.OnClickListener{  
 **private** EditText **Usuario**, **Clave**;  
 **private** Button **Ingresar**, **Registrar**, **Salir**;  
  
 **private static final** String ***BASE\_URL*** = Constantes.HTTP.Usuario.***BASE\_URL\_Usuario***;  
  
 **private** AlertDialog.Builder **builder**;  
  
 **private** Session **session**;  
 **private static final int *REQUEST\_SIGNUP*** = 0;  
  
 **private** View **layoutLogin**;

@Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_login***);  
 **layoutLogin** = findViewById(R.id.***linearLayoutLogin***);  
  
 **session** = **new** Session(getApplication());  
  
 **Usuario** = (EditText)findViewById(R.id.***txtUsuario***);  
  
 **Clave** = (EditText)findViewById(R.id.***txtClave***);  
 **Ingresar** = (Button)findViewById(R.id.***btnIngresar***);  
 **Ingresar**.setOnClickListener(**this**);  
 **Registrar** = (Button)findViewById(R.id.***btnRegistrar***);  
 **Registrar**.setOnClickListener(**this**);  
 **Salir** = (Button)findViewById(R.id.***btnSalir***);  
 **Salir**.setOnClickListener(**this**);  
  
 **Registrar**.setEnabled(**false**);  
  
 **if** (getIntent().getExtras() != **null**){  
 String \_tempUser = getIntent().getExtras().getString(**"\_accountLogged"**);  
 **if**(\_tempUser != **null**) { **Usuario**.setText(\_tempUser); }  
 }**else** {  
 **Usuario**.setText(**"@noroeste.com"**);  
 **Usuario**.setSelection(0);  
 }  
  
 activeNetworkValidator();  
 }  
  
 **public void** Logeo(View v){  
 Utility.*setContext*(getApplicationContext());  
 **if** (!Utility.*hasNetwork*()) {  
 **if** (Utility.*isAirplaneModeOn*()){  
 mostrarBar(**"El modo avión está activo."**).setDuration(Snackbar.***LENGTH\_LONG***).show();  
 }**else**{  
 **final** WifiManager wifiManager = (WifiManager) getApplicationContext().getSystemService(Context.***WIFI\_SERVICE***);  
  
 mostrarBar(**"Es necesario conectarse a alguna red."**)  
 .setAction(**"Activar Wifi"**, **new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **if** (wifiManager.getWifiState() == WifiManager.***WIFI\_STATE\_DISABLED***){  
 wifiManager.setWifiEnabled(**true**);  
 }**else** mostrarToast(**"No es necesario activar el servicio de wifi. Intente con su paquete de datos."**);  
 }  
 }).setDuration(Snackbar.***LENGTH\_LONG***).show();  
 }  
 }**else if** (!validate()){  
 **return**;  
 }**else**{  
 **final** String \_usuario = **Usuario**.getText().toString();  
 **final** String \_clave = **Clave**.getText().toString();  
  
 **new** accionesTeclado().hide();  
  
 valInfoCredential(\_usuario, \_clave);  
 }  
 }  
  
 @Override  
 **protected void** onActivityResult(**int** requestCode, **int** resultCode, Intent data) {  
 **if** (requestCode == ***REQUEST\_SIGNUP***) {  
 **switch** (resultCode){  
 **case *RESULT\_OK***:  
 mostrarToast(**"Se creó satisfactoriamente el perfil."**);  
 **break**;  
 **case *RESULT\_CANCELED***:  
 mostrarBar(**"No se agregó el registro."**);  
 **break**;  
 }  
 }  
 }  
  
 @Override  
 **public void** onBackPressed() {  
 *// disable going back to the* moveTaskToBack(**true**);  
 }  
  
 *//Validamos que la cuenta ingresada existe* **public void** valInfoCredential(**final** String \_correo, **final** String \_clave){  
 **new** hasCredentialAsync(\_correo, LoginActivity.**this**, **new** hasCredentialAsync.onResponseListener() {  
  
 @Override  
 **public void** showError(String error) {  
 **if** (error != **null** && error.length() > 0) mostrarToast(error);  
 }  
  
 @Override  
 **public void** responseCredentials(valCredential response) {  
 **if** (response != **null**){  
 **if** (response.isExiste()){  
 validarAccesos(\_correo, \_clave, response.getNomCompleto());  
 }**else**{  
 mostrarToast(**"Su cuenta "** + \_correo.toUpperCase() + **" no está registrada en el sistema. Solicite los accesos al administrador."**);  
 **Usuario**.requestFocus();  
 **Usuario**.selectAll();  
 }  
 }  
 }  
 }).execute();  
 }  
  
 **public void** generarToken(Permisson permisson){  
 *//Generar token de conexión* **final** UsuarioService service = UsuarioService.Factory.*getInstance*(getApplicationContext());  
 service.Authenticate(permisson).enqueue(**new** Callback<String>() {  
 @Override  
 **public void** onResponse(Call<String> call, Response<String> response) {  
 **if** (response.isSuccessful()){  
 **if** (!response.body().isEmpty()) {  
 Log.*i*(LoginActivity.**class**.getSimpleName(), response.body());  
 **final** String \_newKey = response.body();  
  
 runOnUiThread(**new** Runnable() {  
 @Override  
 **public void** run() {  
 **session**.setToken(\_newKey);  
  
 enableBroadcastToken();  
 cargarServicios();  
 }  
 });  
 }  
 **else** Toast.*makeText*(getApplicationContext(), **"No se generó un token válido. Habrán servicios que no se puedan acceder por seguridad."**, Toast.***LENGTH\_SHORT***).show();  
 }**else** Toast.*makeText*(getApplicationContext(), **"No se obtuvo el token del servidor."**, Toast.***LENGTH\_SHORT***).show();  
 }  
  
 @Override  
 **public void** onFailure(Call<String> call, Throwable t) {  
 Log.*i*(**"Login"**, **"ERROR: "** + t.getCause().toString());  
 }  
 });  
 }  
  
 **public void** validarAccesos(**final** String correo, **final** String clave, **final** String nombre) {  
 **new** hasPasswordAsync(***BASE\_URL***, **this**, **new** hasPasswordAsync.onResponseListener() {  
 @Override  
 **public void** responsePassword(String response) {  
 **switch** (response){  
 **case "SI"**:  
 getRolAndValidatePassword(correo, clave, nombre);  
 **break**;  
 **case "NO"**:  
 **Registrar**.setEnabled(**true**);  
 mostrarToast(**"Es necesario registrar sus credenciales."**);  
 **break**;  
 **case "-"**:  
 **Registrar**.setEnabled(**true**);  
 mostrarToast(**"Al parecer su cuenta tiene permisos para ingresar al módulo web pero aún no ha generado una clave para este módulo."**);  
 **break**;  
 **default**:  
 mostrarBar(**"Comuníquese con el administrador."**).show();  
 **break**;  
 }  
 }  
 }, correo).execute();  
 }  
  
 **public void** getRolAndValidatePassword(**final** String \_cuenta, **final** String \_clave, **final** String \_nombre){  
 **new** validateCredentialsAsync(***BASE\_URL***, **this**, **new** validateCredentialsAsync.onResponseListener() {  
 @Override  
 **public void** responseCredentials(credentialsResponse response) {  
 **if** (response.isCorrecto()){  
 **session**.setCuenta(\_cuenta);  
 **session**.setRol(response.getRol());  
 **session**.setLoggedIn(**true**);  
  
 **if** (**session**.getFinSesion().isEmpty()){  
 String limite = Utility.CONTROLSESSION.***LIMIT***;  
 **session**.setFinSesion(limite);  
 }  
  
 generarToken(**new** Permisson(\_cuenta, \_clave));  
  
 String \_rol = response.getRol().isEmpty() ? **"Sin rol"** : response.getRol();  
 mostrarToast(**"BIENVENIDO: "** + \_nombre + **" ("** + \_rol + **")"**);  
  
 irMain();  
 }**else**{  
 mostrarBar(**"Credenciales incorrectas"**).setAction(**"Limpiar entradas"**, **new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 limpiarEntradas();  
 **new** accionesTeclado().show();  
 }  
 }).show();  
 }  
 }  
 }, \_cuenta, \_clave).execute();  
 }  
  
 **private void** irMain() {  
 Intent i = **new** Intent(**this**, MainActivity.**class**);  
 i.setFlags(Intent.***FLAG\_ACTIVITY\_NEW\_TASK*** | Intent.***FLAG\_ACTIVITY\_CLEAR\_TASK***);  
 startActivity(i);  
 }  
  
 @Override  
 **public void** onClick(View v) {  
 **if** (v == **Ingresar**){  
 **Registrar**.setEnabled(**false**);  
 Logeo(v);  
 }  
  
 **if** (v == **Registrar**){  
 Intent intent = **new** Intent(**this**, RegistrarActivity.**class**);  
 intent.putExtra(RegistrarActivity.***ARG\_KEY\_CUENTA***, **Usuario**.getText().toString());  
 startActivityForResult(intent, ***REQUEST\_SIGNUP***);  
 }  
  
 **if** (v == **Salir**){  
 **if** (Build.VERSION.***SDK\_INT*** >= Build.VERSION\_CODES.***LOLLIPOP***){  
 **builder** = **new** AlertDialog.Builder(LoginActivity.**this**, android.R.style.***Theme\_Material\_Dialog\_Alert***);  
 }**else**{  
 **builder** = **new** AlertDialog.Builder(LoginActivity.**this**);  
 }  
  
 **builder**.setTitle(**"Mensaje del sistema"**)  
 .setMessage(**"¿Seguro que desea cerrar el aplicativo?"**)  
 .setCancelable(**true**)  
 .setPositiveButton(**"Sí"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 finish();  
  
 Intent intent = **new** Intent(Intent.***ACTION\_MAIN***);  
 intent.addCategory(Intent.***CATEGORY\_HOME***);  
 intent.setFlags(Intent.***FLAG\_ACTIVITY\_NEW\_TASK***);  
 startActivity(intent);  
 }  
 })  
 .setNegativeButton(**"No"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 dialog.cancel();  
 }  
 }).setIcon(android.R.drawable.***ic\_dialog\_alert***);  
 **builder**.show();  
 }  
 }  
  
 **private void** cargarServicios(){  
 Intent a = **new** Intent(getApplicationContext(), VerifyGeneralService.**class**);  
 a.addCategory(Intent.***CATEGORY\_DEFAULT***);  
 **boolean** alarmRunning = (PendingIntent.*getBroadcast*(getApplicationContext(), VerifyGeneralService.***REQUEST\_CODE***, a, PendingIntent.***FLAG\_NO\_CREATE***) != **null**);  
 **if**(!alarmRunning) { *//Valido si la alarma está activa* **final** PendingIntent pendingIntent = PendingIntent.*getBroadcast*(**this**, VerifyGeneralService.***REQUEST\_CODE***, a, PendingIntent.***FLAG\_UPDATE\_CURRENT***);  
 **long** firstMillis = System.*currentTimeMillis*();  
 AlarmManager alarmManager = (AlarmManager) **this**.getSystemService(Context.***ALARM\_SERVICE***);  
alarmManager.setRepeating(AlarmManager.***RTC\_WAKEUP***, firstMillis, 3000, pendingIntent); *//3 minutos* Log.*i*(GlobalActivity.**class**.getSimpleName(), **"Entro al alarmRunning"**);  
 }**else** Log.*i*(GlobalActivity.**class**.getSimpleName(), **"El alarmRunning está ejecutandose."**);  
 }  
  
 **public void** enableBroadcastToken(){  
 ComponentName receiver = **new** ComponentName(**this**, VerifyGeneralService.**class**);  
 PackageManager pm = **this**.getPackageManager();  
 pm.setComponentEnabledSetting(receiver, PackageManager.***COMPONENT\_ENABLED\_STATE\_ENABLED***,  
 PackageManager.***DONT\_KILL\_APP***);  
 }  
   
}

* + Layout(s)

<**LinearLayout  
 android:focusable="true"  
 android:focusableInTouchMode="true"  
 xmlns:fab="http://schemas.android.com/apk/res-auto"  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:tools="http://schemas.android.com/tools"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 android:orientation="vertical"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:paddingTop="45dp"  
 android:id="@+id/linearLayoutLogin"  
 android:paddingLeft="24dp" android:background="@drawable/material\_background3"  
 android:paddingRight="24dp" tools:context="com.proyecto.facturacion.LoginActivity"**>  
  
 <**ImageView  
 android:id="@+id/imageView"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="70dp"  
 android:layout\_gravity="center\_horizontal"  
 android:layout\_marginBottom="24dp"  
 android:src="@drawable/logo"** />  
  
 *<!-- Email Label -->* <**android.support.design.widget.TextInputLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="8dp"  
 android:layout\_marginBottom="8dp"**>  
 <**EditText android:id="@+id/txtUsuario"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:inputType="textEmailAddress"  
 android:hint="Usuario"** />  
 </**android.support.design.widget.TextInputLayout**>  
  
 *<!-- Password Label -->* <**android.support.design.widget.TextInputLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="8dp"  
 android:layout\_marginBottom="8dp"**>  
 <**EditText android:id="@+id/txtClave"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:inputType="textPassword"  
 android:hint="Clave"**/>  
 </**android.support.design.widget.TextInputLayout**>  
  
 <**android.support.v7.widget.AppCompatButton  
 android:id="@+id/btnIngresar"  
 android:layout\_width="fill\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="24dp"  
 android:layout\_marginBottom="24dp"  
 android:padding="12dp"  
 android:text="Ingresar"**/>  
  
 *<!-- Bottom Bar -->* <**LinearLayout  
 android:id="@+id/bottom\_bar"  
 android:layout\_width="match\_parent"  
 android:layout\_height="?attr/actionBarSize"  
 android:layout\_alignParentBottom="true"  
 android:gravity="center\_vertical"  
 android:orientation="horizontal"**>  
  
 <**Button  
 android:id="@+id/btnRegistrar"  
 style="@style/Widget.AppCompat.Button.Borderless.Colored"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:backgroundTint="@color/colorPrimary"  
 android:layout\_weight="1"  
 android:text="Registrar"** />  
  
 <**Button  
 android:id="@+id/btnSalir"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:backgroundTint="@color/colorPrimaryDark"  
 android:text="Salir"  
 android:textColor="@android:color/white"** />  
 </**LinearLayout**>  
  
</**LinearLayout**>

* Main
  + Activity

**public class** MainActivity **extends** AppCompatActivity  
 **implements** NavigationView.OnNavigationItemSelectedListener {  
  
 **private static final int *ARG\_KEY\_WEB*** = 99;  
 **public static final int *NOTIFICATION\_PERMANENT*** = 66;  
 **private static final int *ARG\_KEY\_PERFIL*** = 10;  
 **public static final** String ***ARG\_KEY\_CUENTA*** = **"\_cuentaSession"**;  
 **private** BroadcastReceiver **mWifi**;  
 **private** Session **session**;  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_main***);  
 Utility.*setContext*(getApplicationContext());  
  
 **new** Thread(**new** Runnable() {  
 @Override  
 **public void** run() {  
 **session** = **new** Session(getApplicationContext());  
 }  
 }).start();  
  
 StrictMode.ThreadPolicy policy = **new** StrictMode.ThreadPolicy.Builder().permitAll().build();  
 StrictMode.*setThreadPolicy*(policy);  
  
 Toolbar toolbar = (Toolbar) findViewById(R.id.***toolbar***);  
 setSupportActionBar(toolbar);  
  
 DrawerLayout drawer = (DrawerLayout) findViewById(R.id.***drawer\_layout***);  
 ActionBarDrawerToggle toggle = **new** ActionBarDrawerToggle(  
 **this**, drawer, toolbar, R.string.***navigation\_drawer\_open***, R.string.***navigation\_drawer\_close***);  
 drawer.setDrawerListener(toggle);  
 toggle.syncState();  
  
 NavigationView navigationView = (NavigationView) findViewById(R.id.***nav\_view***);  
 navigationView.setNavigationItemSelectedListener(**this**);  
  
 permisosRol(**session**, navigationView);  
 cargarSesion(**session**, navigationView);  
  
 cargarListaFacturas();  
  
 evaluarNotificaciones();}  
  
 **private void** permisosRol(Session session, NavigationView navigationView) {  
 Menu menu = navigationView.getMenu();  
 String \_roles = session.getRol();  
 **switch** (\_roles){  
 **case "Root"**:  
 **break**;  
 **case "Administrador"**:  
 menu.findItem(R.id.***nav\_reporte***).setEnabled(**false**);  
 **break**;  
 **default**:  
 menu.findItem(R.id.***nav\_cliente***).setVisible(**false**);  
 menu.findItem(R.id.***nav\_moneda***).setVisible(**false**);  
 menu.findItem(R.id.***nav\_reporte***).setVisible(**false**);}  
  
 }  
  
 **private void** cargarSesion(**final** Session session, NavigationView navigationView) {  
 View headerView = navigationView.getHeaderView(0);  
 TextView navUsername = (TextView) headerView.findViewById(R.id.***tvAccountSession***);  
 TextView navRol = (TextView) headerView.findViewById(R.id.***tvRolSession***);  
 navUsername.setText(session.getCuenta());  
 navRol.setText(session.getRol());  
  
 LinearLayout nav\_header = (LinearLayout) headerView.findViewById(R.id.***nav\_header***);  
 nav\_header.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent miPerfil = **new** Intent(MainActivity.**this**, PerfilActivity.**class**);  
 miPerfil.putExtra(***ARG\_KEY\_CUENTA***, session.getCuenta());  
 startActivityForResult(miPerfil, ***ARG\_KEY\_PERFIL***);  
 }  
 });  
 }  
  
 @Override  
 **protected void** onActivityResult(**int** requestCode, **int** resultCode, Intent data) {  
 **if** (requestCode == ***ARG\_KEY\_PERFIL***){  
 **switch** (resultCode){  
 **case** Activity.***RESULT\_OK***:  
 mostrarToast(**"Se aplicaron los cambios."**);  
 **break**;  
 **case** Activity.***RESULT\_CANCELED***:  
 mostrarToast(**"No se aplicaron los cambios en el perfil"**);  
 **break**;  
 }  
 }  
 }  
  
 @Override  
 **protected void** onResume() {  
 **super**.onResume();  
 Utility.*closeGlobalNotification*(***NOTIFICATION\_PERMANENT***);  
 }  
  
 @Override  
 **protected void** onPause() {  
 **super**.onPause();  
 Utility.*showGlobalNotification*(**"El sistema de facturación aún se encuentra ejecutándose."**, MainActivity.**class**, ***NOTIFICATION\_PERMANENT***);  
 }  
   
 @SuppressWarnings(**"StatementWithEmptyBody"**)  
 @Override  
 **public boolean** onNavigationItemSelected(MenuItem item) {  
 *// Handle navigation view item clicks here.* **int** id = item.getItemId();  
 **final** String \_token = **session**.getToken();  
  
 FragmentManager fragmentManager = getSupportFragmentManager();  
 FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();  
 fragmentTransaction.addToBackStack(**null**);  
 fragmentTransaction.isAddToBackStackAllowed();  
  
 **if** (id == R.id.***nav\_factura***) {  
 irListaFacturas();  
 } **else if** (id == R.id.***nav\_cliente***) {  
 irListaClientes(fragmentTransaction);  
 } **else if** (id == R.id.***nav\_moneda***) {  
 MonedasFragment monedas = MonedasFragment.*newInstance*(\_token);  
 fragmentTransaction.replace(R.id.***fragment\_container***, monedas, **"MONEDAS"**);  
 fragmentTransaction.commit();  
 } **else if** (id == R.id.***nav\_reporte***) {  
 String rol = **session**.getRol();  
 Toast.*makeText*(**this**, **"ROL: "** + rol, Toast.***LENGTH\_SHORT***).show();  
 }**else if**(id == R.id.***nav\_web***){  
 Intent i = **new** Intent(Intent.***ACTION\_VIEW***);  
 i.setData(Uri.*parse*(Constantes.RESOURCES.***URL\_SISTEMA\_WEB***));  
 String title = getResources().getString(R.string.***chooser\_url\_title***);  
 PackageManager packageManager = getPackageManager();  
 List activities = packageManager.queryIntentActivities(i, PackageManager.***MATCH\_DEFAULT\_ONLY***);  
 **boolean** isIntentSafe = activities.size() > 0;  
  
 **if** (isIntentSafe) startActivity(Intent.*createChooser*(i, title));  
 **else**{  
 Intent intent = **new** Intent(**this**, WebActivity.**class**);  
 startActivityForResult(intent, ***ARG\_KEY\_WEB***);  
 }  
 }**else if** (id == R.id.***nav\_cerrarsesion***) {  
 eliminarTemporales();  
 cancelServicios();  
 disableBroadcastToken();  
 hideNotification(GeneralIntentService.***\_KEY\_NOT\_GENERAL***); *//cerrar la notificación del KEY General* Intent i = **new** Intent(**this**, LoginActivity.**class**);  
 startActivity(i);  
 }  
  
 DrawerLayout drawer = (DrawerLayout) findViewById(R.id.***drawer\_layout***);  
 drawer.closeDrawer(GravityCompat.***START***);  
 **return true**;  
 }  
  
 **private void** irListaClientes(FragmentTransaction ft) {  
 ClientesFragment clientes = **new** ClientesFragment();  
 ft.replace(R.id.***fragment\_container***, clientes, **"CLIENTES"**);  
 ft.commit();  
 }  
  
 **private void** irListaFacturas() {  
 FragmentTransaction transection = getSupportFragmentManager().beginTransaction();  
 FacturasFragment facturas = **new** FacturasFragment();  
 transection.replace(R.id.***fragment\_container***, facturas);  
 transection.isAddToBackStackAllowed();  
 transection.addToBackStack(**null**);  
 transection.commit();  
 }  
  
 **private void** eliminarTemporales(){  
 **session**.setFinSesion(**""**);  
 **session**.setToken(**""**);  
 **session**.setRol(**""**);  
 **session**.setCuenta(**""**);  
 **session**.setLoggedIn(**false**);  
 }  
  
 **public void** cancelServicios(){  
 Intent intent = **new** Intent(**this**, VerifyGeneralService.**class**);  
 PendingIntent pendingIntent = PendingIntent.*getBroadcast*(**this**, VerifyGeneralService.***REQUEST\_CODE***, intent, 0);  
 AlarmManager alarmManager = (AlarmManager) **this**.getSystemService(Context.***ALARM\_SERVICE***);  
 alarmManager.cancel(pendingIntent);  
 }  
  
 **public void** disableBroadcastToken(){  
 ComponentName receiver = **new** ComponentName(**this**, VerifyGeneralService.**class**);  
 PackageManager pm = **this**.getPackageManager();  
 pm.setComponentEnabledSetting(receiver, PackageManager.***COMPONENT\_ENABLED\_STATE\_DISABLED***, PackageManager.***DONT\_KILL\_APP***);  
 }  
}

* + Layout(s)

<**android.support.v4.widget.DrawerLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:id="@+id/drawer\_layout"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:fitsSystemWindows="true"  
 tools:openDrawer="start"**>  
  
 <**include  
 layout="@layout/app\_bar\_main"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"** />  
  
 <**android.support.design.widget.NavigationView  
 android:id="@+id/nav\_view"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="match\_parent"  
 android:layout\_gravity="start"  
 android:fitsSystemWindows="true"  
 app:headerLayout="@layout/nav\_header\_main"  
 app:menu="@menu/activity\_main\_drawer"** />  
  
</**android.support.v4.widget.DrawerLayout**>

* Registrar
  + Activity

**public class** RegistrarActivity **extends** AppCompatActivity {  
 **private** Session **session**;  
 **private static final** String ***TAG*** = **"RegistrarActivity"**;  
 **public static final** String ***ARG\_KEY\_CUENTA*** = **"\_keyCuenta"**;  
 **private static final** String ***BASE\_URL*** = Constantes.HTTP.Usuario.***BASE\_URL\_Usuario***;  
  
 **private** EditText **txtCuenta**, **txtClave**;  
 **private** Button **btnAgregar**;  
 **private** TextView **tvRegresar**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_registrar***);  
  
 getSupportActionBar().setDisplayHomeAsUpEnabled(**true**);  
 getSupportActionBar().setHomeButtonEnabled(**true**);  
  
 **session** = **new** Session(**this**);  
  
 **txtCuenta** = (EditText)findViewById(R.id.***txtCuenta***);  
 **txtClave** = (EditText)findViewById(R.id.***txtClave***);  
 **btnAgregar** = (Button)findViewById(R.id.***btnAgregar***);  
 **tvRegresar** = (TextView)findViewById(R.id.***tvRegresar***);  
  
 String \_cuentaTemp = getIntent().getStringExtra(***ARG\_KEY\_CUENTA***);  
 **txtCuenta**.setText(\_cuentaTemp);  
 **txtCuenta**.setEnabled(**false**);  
  
 **txtClave**.requestFocus();  
  
 **btnAgregar**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **if** (!validate()){  
 Toast.*makeText*(getBaseContext(), **"Falló el registro de la cuenta"**, Toast.***LENGTH\_LONG***).show();  
 **return**;  
 }  
  
 String \_usuario = **txtCuenta**.getText().toString();  
 String \_clave = **txtClave**.getText().toString();  
  
 registrarCuenta(\_usuario, \_clave);  
 }  
 });  
  
 **tvRegresar**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **session**.setCuenta(**""**);  
 finish();  
 }  
 });  
 }  
  
 @Override  
 **public void** onBackPressed() {  
 moveTaskToBack(**true**);  
 }  
  
 **public boolean** validate() {  
 **boolean** valid = **true**;  
  
 String email = **txtCuenta**.getText().toString();  
 String password = **txtClave**.getText().toString();  
 **if** (email.isEmpty() || !android.util.Patterns.***EMAIL\_ADDRESS***.matcher(email).matches()) {  
 **txtCuenta**.setError(**"Ingresar un correo válido"**);  
 valid = **false**;  
 } **else** {  
 **txtCuenta**.setError(**null**);  
 }  
 **if** (password.isEmpty() || password.length() < 4 || password.length() > 10) {  
 **txtClave**.setError(**"Entre 4 y 10 caracteres alfanuméricos"**);  
 valid = **false**;  
 } **else** {  
 **txtClave**.setError(**null**);  
 }  
 **return** valid;  
 }  
  
 **private void** registrarCuenta(String cuenta, String clave){  
 **new** asignCredentialAsync(***BASE\_URL***, cuenta, clave, **this**, **new** asignCredentialAsync.onResponseListener() {  
 @Override  
 **public void** responseAsign(Boolean response) {  
 setResult(response ? Activity.***RESULT\_OK*** : Activity.***RESULT\_CANCELED***);  
 finish();  
 }  
 }).execute();  
 }  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 **switch** (item.getItemId()){  
 **case** android.R.id.***home***:  
 finish();  
 **break**;  
 }  
 **return super**.onOptionsItemSelected(item);  
 }  
}

* + Layout(s)
* <**ScrollView  
   xmlns:android="http://schemas.android.com/apk/res/android"  
   xmlns:app="http://schemas.android.com/apk/res-auto"  
   xmlns:tools="http://schemas.android.com/tools"  
   android:layout\_width="match\_parent"  
   android:layout\_height="match\_parent"  
   android:fitsSystemWindows="true"  
   android:background="@drawable/material\_background3"  
   tools:context="com.proyecto.facturacion.RegistrarActivity"**>  
    
   <**LinearLayout  
   android:orientation="vertical"  
   android:layout\_width="match\_parent"  
   android:layout\_height="wrap\_content"  
   android:paddingTop="56dp"  
   android:paddingLeft="24dp"  
   android:paddingRight="24dp"**>  
    
   <**ImageView android:src="@drawable/logo"  
   android:contentDescription="Logo"  
   android:layout\_width="wrap\_content"  
   android:layout\_height="72dp"  
   android:layout\_marginBottom="24dp"  
   android:layout\_gravity="center\_horizontal"** />  
    
   *<!-- Email Label -->* <**android.support.design.widget.TextInputLayout  
   android:layout\_width="match\_parent"  
   android:layout\_height="wrap\_content"  
   android:layout\_marginTop="8dp"  
   android:layout\_marginBottom="8dp"**>  
   <**EditText android:id="@+id/txtCuenta"  
   android:layout\_width="match\_parent"  
   android:layout\_height="wrap\_content"  
   android:inputType="textEmailAddress"  
   android:hint="Correo"** />  
   </**android.support.design.widget.TextInputLayout**>  
    
   *<!-- Password Label -->* <**android.support.design.widget.TextInputLayout  
   android:layout\_width="match\_parent"  
   android:layout\_height="wrap\_content"  
   android:layout\_marginTop="8dp"  
   android:layout\_marginBottom="8dp"**>  
   <**EditText android:id="@+id/txtClave"  
   android:layout\_width="match\_parent"  
   android:layout\_height="wrap\_content"  
   android:inputType="textPassword"  
   android:hint="Contraseña"**/>  
   </**android.support.design.widget.TextInputLayout**>  
    
   *<!-- Signup Button -->* <**android.support.v7.widget.AppCompatButton  
   android:id="@+id/btnAgregar"  
   android:backgroundTint="@color/colorAccent"  
   android:layout\_width="fill\_parent"  
   android:layout\_height="wrap\_content"  
   android:layout\_marginTop="24dp"  
   android:layout\_marginBottom="24dp"  
   android:padding="12dp"  
   android:text="Aplicar"**/>  
    
   <**TextView android:id="@+id/tvRegresar"  
   android:layout\_width="fill\_parent"  
   android:layout\_height="wrap\_content"  
   android:layout\_marginBottom="24dp"  
   android:text="¿Ya eres un miembro? Inicia sesión"  
   android:gravity="center"  
   android:textSize="16dip"**/>  
   </**LinearLayout**>  
  </**ScrollView**>
  1. Agregar librerías para la conexión entre el dispositivo y los servicios web

dependencies {  
 compile fileTree(include: [**'\*.jar'**], dir: **'libs'**)  
 androidTestCompile(**'com.android.support.test.espresso:espresso-core:2.2.2'**, {  
 exclude group: **'com.android.support'**, module: **'support-annotations'** })  
 compile **'com.android.support:appcompat-v7:25.3.1'** compile **'com.android.support.constraint:constraint-layout:1.0.2'** compile **'com.android.support:design:25.0.0'** compile **'com.squareup.retrofit2:retrofit:2.1.0'** compile **'com.squareup.retrofit2:converter-gson:2.1.0'** compile **'de.hdodenhof:circleimageview:1.3.0'** compile **'com.android.support:support-v4:25.3.1'** compile **'com.android.support:recyclerview-v7:25.3.1'** compile **'com.android.support:cardview-v7:25.3.1'** compile **'com.squareup.okhttp3:logging-interceptor:3.0.1'** compile **'com.squareup.retrofit2:adapter-rxjava:2.1.0'** testCompile **'junit:junit:4.12'**}

* 1. Creación de métodos y clases para la consumir los webservices
* Clientes

**public interface** ClienteService {  
 String ***BASE\_URL*** = Constantes.HTTP.Cliente.***BASE\_URL\_Cliente***;  
  
 @GET(Constantes.HTTP.Cliente.***SRV\_GETALL***)  
 Call<List<Cliente>> getClientes();  
  
 @POST(Constantes.HTTP.Cliente.***SRV\_INSERT***)  
 Call<Boolean> insertClientes(@Body Cliente moneda);  
  
 @DELETE(Constantes.HTTP.Cliente.***SRV\_DELETE***)  
 Call<Boolean> deleteClientes(@Path(**"id"**) String id);  
  
 @PUT(Constantes.HTTP.Cliente.***SRV\_UPDATE***)  
 Call<Boolean> updateClientes(@Body Cliente moneda);  
}

* Credenciales

**public interface** CredencialService {  
 String ***BASE\_URL*** = Constantes.HTTP.Usuario.***BASE\_URL\_Usuario***;  
  
 @GET(Constantes.HTTP.Usuario.***SRV\_EXISTSCREDENTIAL***)  
 **public** Call<valCredential> hasCredential(@Path(**"cuenta"**) String cuenta);  
  
 @GET(Constantes.HTTP.Usuario.***SRV\_VALIDATECREDENTIALS***)  
 Call<credentialsResponse> validateCredentials(@Path(**"cuenta"**) String cuenta, @Path(**"clave"**) String clave);  
  
 @PUT(Constantes.HTTP.Usuario.***SRV\_ASIGNCREDENTIAL***)  
 Call<Boolean> asignCredential(@Body Credentials model);  
}

* Detalle de la factura

**public interface** DetalleFacturaService {  
 String ***BASE\_URL*** = Constantes.HTTP.DetalleFactura.***BASE\_URL\_DetalleFactura***;  
  
 @GET(Constantes.HTTP.DetalleFactura.***SRV\_GETALL***)  
 Call<List<DetalleUsuarioFactura>> getDetalleFacturas();  
  
 @GET(Constantes.HTTP.DetalleFactura.***SRV\_GETBYFACTURA***)  
 Call<List<DetalleUsuarioFactura>> getDetalleByFactura(@Path(**"idFactura"**) String idFactura);  
  
 @POST(Constantes.HTTP.DetalleFactura.***SRV\_INSERT***)  
 Call<Boolean> insertDetalleFacturas(@Body DetalleUsuarioFactura model);  
  
 @DELETE(Constantes.HTTP.DetalleFactura.***SRV\_DELETE***)  
 @Headers(**"Content-Type: application/json; charset=UTF-8"**)  
 Call<Boolean> deleteDetalleFacturas(@Path(**"id"**) String id);  
  
 @PUT(Constantes.HTTP.DetalleFactura.***SRV\_UPDATE***)  
 Call<Boolean> updateDetalleFacturas(@Body DetalleUsuarioFactura model);  
}

* Facturas

**public interface** FacturaService {  
 String ***BASE\_URL*** = Constantes.HTTP.Factura.***BASE\_URL\_Factura***;  
  
 @GET(Constantes.HTTP.Factura.***SRV\_GETALL***)  
 Call<List<Factura>> getFacturas();  
  
 @GET(Constantes.HTTP.Factura.***SRV\_GETCOMBYMODEL***)  
 @Headers(**"Content-Type: application/json; charset=UTF-8"**)  
 Call<Factura> findCOMByModel(@Path(**"condiciones"**) String condiciones, @Path(**"fecha"**) String fecha, @Path(**"guiacarga"**) String guiacarga, @Path(**"puertodestino"**) String puertodestino, @Path(**"puertoorigen"**) String puertoorigen, @Path(**"idcliente"**) String idcliente);  
  
 @POST(Constantes.HTTP.Factura.***SRV\_INSERT***)  
 Call<Boolean> insertFacturas(@Body Factura factura);  
  
 @DELETE(Constantes.HTTP.Factura.***SRV\_DELETE***)  
 @Headers(**"Content-Type: application/json; charset=UTF-8"**)  
 Call<Boolean> deleteFacturas(@Path(**"id"**) String id);  
  
 @PUT(Constantes.HTTP.Factura.***SRV\_UPDATE***)  
 @Headers(**"Content-Type: application/json; charset=UTF-8"**)  
 Call<Boolean> updateFacturas(@Body Factura moneda);  
}

* Monedas

**public interface** MonedaService {  
 String ***BASE\_URL*** = Constantes.HTTP.Moneda.***BASE\_URL\_Moneda***;  
  
 @GET(Constantes.HTTP.Moneda.***SRV\_GETALL***)  
 Call<List<Moneda>> getMonedas();  
  
 @POST(Constantes.HTTP.Moneda.***SRV\_INSERT***)  
 Call<Boolean> insertMonedas(@Body Moneda moneda);  
  
 @DELETE(Constantes.HTTP.Moneda.***SRV\_DELETE***)  
 Call<Boolean> deleteMonedas(@Path(**"id"**) String id);  
  
 @PUT(Constantes.HTTP.Moneda.***SRV\_UPDATE***)  
 @Headers(**"Content-Type: application/json; charset=UTF-8"**)  
 Call<Boolean> updateMonedas(@Body Moneda moneda);

}

* Usuarios

**public interface** UsuarioService {  
 String ***BASE\_URL*** = Constantes.HTTP.Usuario.***BASE\_URL\_Usuario***;  
  
 @POST(Constantes.HTTP.Usuario.***SRV\_AUTH***)  
 Call<String> Authenticate(@Body Permisson creds);  
  
 @POST(Constantes.HTTP.Usuario.***SRV\_VALIDATETOKEN***)  
 Call<Boolean> ValidateToken(@Body TokenInformation model);  
  
 @GET(Constantes.HTTP.Usuario.***SRV\_GETALL***)  
 Call<List<Usuario>> getUsuarios();  
  
 @GET(Constantes.HTTP.Usuario.***SRV\_HASPASSWORD***)  
 Call<String> hasPassword(@Path(**"cuenta"**) String cuenta);  
  
 @GET(Constantes.HTTP.Usuario.***SRV\_GETIDBYMAIL***)  
 Call<String> findIdByMail(@Path(**"mail"**) String mail);  
  
 @GET(Constantes.HTTP.Usuario.***SRV\_GETBYEMAIL***)  
 Call<Usuario> findModelByMail(@Path(**"mail"**) String mail);  
  
 @PUT(Constantes.HTTP.Usuario.***SRV\_UPDATE***)  
 Call<Boolean> updatePerfil(@Body Usuario usuario);  
  
 @PUT(Constantes.HTTP.Usuario.***SRV\_UPDATEPASSWORD***)  
 Call<Boolean> updateClave(@Path(**"ID"**) String id,@Path(**"oldClave"**) String oldclave,@Path(**"newClave"**) String newclave);  
}

* 1. Creación de adaptadores y sus vistas
* Adaptador del cliente

**public class** ClienteAdapter **extends** RecyclerView.Adapter<ClienteAdapter.Holder> **implements** Filterable {  
 **private** List<Cliente> **lCliente**;  
 **private** List<Cliente> **mFiltered**;  
 **private** Context **context**;  
 **private final** ClienteClickListener **mlistener**;  
  
 **private static final** String ***TAG*** = ClienteAdapter.**class**.getSimpleName();  
  
 **public** ClienteAdapter(Context context, List<Cliente> objects, ClienteClickListener listener) {  
 **this**.**context** = context;  
 **this**.**lCliente** = objects;  
 **this**.**mFiltered** = objects;  
 **this**.**mlistener** = listener;  
 }  
  
 @Override  
 **public** Filter getFilter() {  
 **return new** Filter() {  
 @Override  
 **protected** FilterResults performFiltering(CharSequence constraint) {  
 String charString = constraint.toString();  
 **if** (charString.isEmpty()) **mFiltered** = **lCliente**;  
 **else**{  
 ArrayList<Cliente> filteredList = **new** ArrayList<>();  
 **for** (Cliente cliente : **lCliente**) {  
 **if** (cliente.getCliente().toLowerCase().contains(charString)) {  
 filteredList.add(cliente);  
 }  
 }  
 **mFiltered** = filteredList;  
 }  
 FilterResults filterResults = **new** FilterResults();  
 filterResults.**values** = **mFiltered**;  
 **return** filterResults;  
 }  
 @Override  
 **protected void** publishResults(CharSequence constraint, FilterResults results) {  
 **mFiltered** = (List<Cliente>) results.**values**;  
 notifyDataSetChanged();  
 }  
 };  
 }  
  
 **public class** Holder **extends** RecyclerView.ViewHolder **implements** View.OnClickListener, View.OnLongClickListener {  
 **private** TextView **main**, **detail**;  
 **public** Holder(View itemView) {  
 **super**(itemView);  
 **main** = (TextView) itemView.findViewById(R.id.***txtMain***);  
 **detail** = (TextView) itemView.findViewById(R.id.***txtDetail***);  
 itemView.setOnClickListener(**this**);  
 itemView.setOnLongClickListener(**this**);  
 }  
 @Override  
 **public void** onClick(View v) {  
 **mlistener**.onClick(getLayoutPosition());  
 }  
 @Override  
 **public boolean** onLongClick(View v) {  
 **mlistener**.onLongClick(getLayoutPosition());  
 **return true**;  
 }  
 }  
  
 @Override  
 **public** ClienteAdapter.Holder onCreateViewHolder(ViewGroup parent, **int** viewType) {  
 View row = LayoutInflater.*from*(**context**).inflate(R.layout.***clientes\_item***, parent, **false**);  
 **return new** ClienteAdapter.Holder(row);  
 }  
  
 @Override  
 **public void** onBindViewHolder(ClienteAdapter.Holder holder, **int** position) {  
 Cliente model = **mFiltered**.get(position);  
 holder.**main**.setText(model.getCliente());  
 holder.**detail**.setText(**"Cantidad de facturas: "** + model.get\_Facturas());  
 }  
  
 @Override  
 **public int** getItemCount() {  
 **return mFiltered**.size();  
 }  
  
 **public interface** ClienteClickListener{  
 **void** onClick(**int** position);  
 **void** onLongClick(**int** position);  
 }  
  
 **public** Cliente getSelectedCliente(**int** position) {  
 **return mFiltered**.get(position);  
 }  
}

* + Layout

<**android.support.v7.widget.CardView  
 xmlns:card\_view="http://schemas.android.com/apk/res-auto"  
 card\_view:cardCornerRadius="5dp"  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="5dp"  
 android:layout\_marginLeft="5dp"  
 android:layout\_marginRight="5dp"**>  
 <**LinearLayout  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"  
 android:layout\_margin="5dp"  
 android:orientation="vertical"**>  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_marginTop="5dp" android:layout\_marginBottom="5dp"  
 android:layout\_height="wrap\_content" android:textAppearance="@style/Base.TextAppearance.AppCompat.Title"  
 android:text="Large Text" android:id="@+id/txtMain"**/>  
 <**TextView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content" android:textAppearance="@style/Base.TextAppearance.AppCompat.Body1"  
 android:text="Medium Text" android:id="@+id/txtDetail"**/>  
 </**LinearLayout**>  
</**android.support.v7.widget.CardView**>

* Adaptador del detalle del usuario
  + Clase

**public class** DetalleUsuarioFacturaAdapter **extends** RecyclerView.Adapter<DetalleUsuarioFacturaAdapter.Holder> {  
 **private** List<DetalleUsuarioFactura> **lDetalleFactura**;  
 **private** Context **context**;  
 **private final** DetalleFacturaClickListener **mlistener**;  
  
 **public** DetalleUsuarioFacturaAdapter(Context context, List<DetalleUsuarioFactura> objects, DetalleFacturaClickListener listener) {  
 **this**.**context** = context;  
 **this**.**lDetalleFactura** = objects;  
 **this**.**mlistener** = listener;  
 }  
  
 @Override  
 **public** DetalleUsuarioFacturaAdapter.Holder onCreateViewHolder(ViewGroup parent, **int** viewType) {  
 View row = LayoutInflater.*from*(**context**).inflate(R.layout.***adapter\_detalleusuariofactura\_item***, parent, **false**);  
 **return new** DetalleUsuarioFacturaAdapter.Holder(row);  
 }  
  
 @Override  
 **public void** onBindViewHolder(DetalleUsuarioFacturaAdapter.Holder holder, **int** position) {  
 DetalleUsuarioFactura model = **lDetalleFactura**.get(position);  
 holder.**main**.setText(model.getDescripcion());  
 holder.**detail**.setText(model.getPesoTotal());  
 holder.**monto**.setText(model.getIdMoneda() + **" "** + model.getSubtotal().toString());  
 }  
  
 @Override  
 **public int** getItemCount() {  
 **return lDetalleFactura**.size();  
 }  
  
 **public** DetalleUsuarioFactura getSelected(**int** position) {  
 **return lDetalleFactura**.get(position);  
 }  
  
 **public class** Holder **extends** RecyclerView.ViewHolder **implements** View.OnClickListener{

**private** TextView **main**, **detail**, **monto**;  
 **public** Holder(View itemView) {  
 **super**(itemView);  
 **main** = (TextView) itemView.findViewById(R.id.***txtMain***);  
 **detail** = (TextView) itemView.findViewById(R.id.***txtDetail***);  
 **monto** = (TextView) itemView.findViewById(R.id.***txtMonto***);  
 itemView.setOnClickListener(**this**);  
 }

@Override  
 **public void** onClick(View v) {  
 **mlistener**.onClick(getLayoutPosition());  
 }  
 }  
  
 **public interface** DetalleFacturaClickListener{  
 **void** onClick(**int** position);  
 }  
}

* + Layout

<**LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"  
 android:orientation="horizontal" android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content" android:padding="5dp"**>  
 <**ImageView  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:background="@mipmap/ic\_launcher"  
 android:minHeight="80dp"  
 android:minWidth="80dp"  
 android:scaleType="centerCrop"**/>  
 <**LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical"**>  
 <**TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content" android:textAppearance="@style/Base.TextAppearance.AppCompat.Title"  
 android:text="Title" android:id="@+id/txtMain"**/>  
 <**TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content" android:textAppearance="@style/Base.TextAppearance.AppCompat.Caption"  
 android:text="Body" android:id="@+id/txtDetail"**/>  
 <**TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent" android:textAppearance="@style/Base.TextAppearance.AppCompat.Small"  
 android:text="Body 2" android:id="@+id/txtMonto"**/>  
 </**LinearLayout**>  
</**LinearLayout**>

* Adaptador de las facturas
  + Clase

**public class** FacturaAdapter **extends** RecyclerView.Adapter<FacturaAdapter.Holder> **implements** Filterable {  
 **private** List<Factura> **lFactura**;  
 **private** List<Factura> **mFiltered**;  
 **private** Context **context**;  
 **private final** FacturaClickListener **mlistener**;  
  
 **private static final** String ***TAG*** = FacturasFragment.**class**.getSimpleName();  
  
 **public** FacturaAdapter(Context context, List<Factura> objects, FacturaClickListener listener) {  
 **this**.**context** = context;  
 **this**.**lFactura** = objects;  
 **this**.**mlistener** = listener;  
 **this**.**mFiltered** = objects;  
 }  
  
 @Override  
 **public** FacturaAdapter.Holder onCreateViewHolder(ViewGroup parent, **int** viewType) {  
 View row = LayoutInflater.*from*(**context**).inflate(R.layout.***facturas\_item***, parent, **false**);  
 **return new** FacturaAdapter.Holder(row);  
 }  
  
 @Override  
 **public void** onBindViewHolder(FacturaAdapter.Holder holder, **int** position) {  
 Factura model = **mFiltered**.get(position);  
 holder.**main**.setText(model.getCom\_Id());  
  
 String \_format = model.getTotal().toString();  
 **if**(model.getMoneda() != **null**){  
 \_format = model.getMoneda() +**" "**+ model.getTotal().toString();  
 }  
  
 String \_origen = model.getPuertoOrigen().length() > 18 ? model.getPuertoOrigen().substring(0, 17) + **"..."** : model.getPuertoOrigen();  
 String \_destino = model.getPuertoDestino().length() > 18 ? model.getPuertoDestino().substring(0, 17) + **"..."** : model.getPuertoDestino();  
  
 holder.**detail**.setText(\_format);  
 holder.**fecha**.setText(model.getFecha());  
 holder.**origen**.setText(\_origen);  
 holder.**destino**.setText(\_destino);  
 }  
  
 @Override  
 **public int** getItemCount() {  
 **return mFiltered**.size();  
 }  
 **public** Factura getSelectedFactura(**int** position) {  
 **return mFiltered**.get(position);  
 }  
 @Override  
 **public** Filter getFilter() {  
 **return new** Filter() {  
 @Override  
 **protected** FilterResults performFiltering(CharSequence constraint) {  
 String charString = constraint.toString();  
 **if** (charString.isEmpty()) **mFiltered** = **lFactura**;  
 **else**{  
 ArrayList<Factura> filteredList = **new** ArrayList<>();  
 **for** (Factura factura : **lFactura**) {  
 **if** (factura.getCom\_Id().toLowerCase().contains(charString.toLowerCase()) || factura.getFecha().contains(charString) || factura.getPuertoOrigen().toLowerCase().contains(charString.toLowerCase()) || factura.getPuertoDestino().toLowerCase().contains(charString.toLowerCase())) { filteredList.add(factura); } }  
 **mFiltered** = filteredList;  
 }  
  
 FilterResults filterResults = **new** FilterResults();  
 filterResults.**values** = **mFiltered**;  
 **return** filterResults;  
 }  
 @Override  
 **protected void** publishResults(CharSequence constraint, FilterResults results) {  
 **mFiltered** = (List<Factura>) results.**values**;  
 notifyDataSetChanged();  
 }  
 };  
 }  
 **class** Holder **extends** RecyclerView.ViewHolder **implements** View.OnClickListener{  
 **private** ImageView **imageView**;  
 **private** TextView **main**, **detail**, **fecha**, **origen**, **destino**, **linearLayoutActualizar**, **linearLayoutEliminar**, **linearLayoutAdicional**;  
 **private** Holder(View itemView) {  
 **super**(itemView);  
 **imageView** = (ImageView) itemView.findViewById(R.id.***factImagen***);  
 **main** = (TextView) itemView.findViewById(R.id.***txtMain***);  
 **detail** = (TextView) itemView.findViewById(R.id.***txtDetail***);  
 **fecha** = (TextView) itemView.findViewById(R.id.***txtFecha***);  
 **origen** = (TextView) itemView.findViewById(R.id.***txtOrigen***);  
 **destino** = (TextView) itemView.findViewById(R.id.***txtDestino***);  
 **linearLayoutEliminar** = (TextView) itemView.findViewById(R.id.***linearEliminar***);  
 **linearLayoutActualizar** = (TextView) itemView.findViewById(R.id.***linearActualizar***);  
 **linearLayoutAdicional** = (TextView) itemView.findViewById(R.id.***linearAdicional***);  
 itemView.setOnClickListener(**this**);  
 **linearLayoutEliminar**.setOnClickListener(**this**);  
 **linearLayoutActualizar**.setOnClickListener(**this**);  
 **linearLayoutAdicional**.setOnClickListener(**this**);  
 }  
 @Override  
 **public void** onClick(View v) {  
 **switch** (v.getId()){  
 **case** R.id.***linearActualizar***:  
 **mlistener**.onClickActualizar(getLayoutPosition());  
 **break**;  
 **case** R.id.***linearEliminar***:  
 **mlistener**.onClickEliminar(getLayoutPosition());  
 **break**;  
 **case** R.id.***linearAdicional***:  
 **mlistener**.onClickAdicional(getLayoutPosition());  
 **break**;  
 **default**:  
 **mlistener**.onClick(getLayoutPosition());  
 }  
 }  
 }

**public interface** FacturaClickListener{  
 **void** onClick(**int** position);  
 **void** onClickEliminar(**int** position);  
 **void** onClickActualizar(**int** position);  
 **void** onClickAdicional(**int** position);  
 }  
}

* + Layout

<**android.support.v7.widget.CardView  
 xmlns:card\_view="http://schemas.android.com/apk/res-auto"  
 card\_view:cardCornerRadius="5dp"  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="10dp"  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginRight="10dp"**>  
 <**LinearLayout  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"  
 android:minHeight="70dp"  
 android:minWidth="70dp"  
 android:maxHeight="80dp"  
 android:maxWidth="80dp"  
 android:orientation="horizontal"**>  
 <**ImageView  
 android:id="@+id/factImagen"  
 android:background="@mipmap/ic\_launcher"  
 android:scaleType="centerCrop"  
 android:layout\_width="0dip"  
 android:layout\_height="fill\_parent"  
 android:layout\_weight="0.25"**/>  
 <**LinearLayout  
 android:layout\_width="0dip"  
 android:layout\_height="fill\_parent"  
 android:layout\_weight="0.75"  
 android:gravity="center\_vertical|center\_horizontal"  
 android:orientation="vertical"**>  
 <**LinearLayout  
 android:orientation="horizontal"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"**>  
 <**TextView  
 android:id="@+id/txtMain"  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:text="Large Text" android:textAppearance="@style/Base.TextAppearance.AppCompat.Title"** />  
 <**TextView  
 android:id="@+id/txtFecha"  
 android:text="Fecha"  
 android:layout\_marginRight="10dp"  
 android:gravity="center\_vertical"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="match\_parent"** />  
 </**LinearLayout**>  
 <**TextView  
 android:id="@+id/txtDetail"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:text="Medium Text"  
 android:textAppearance="@style/Base.TextAppearance.AppCompat.Subhead"** />  
 <**View  
 android:layout\_width="wrap\_content"  
 android:layout\_height="3dp"  
 android:background="#DDFFDD"** />  
  
 <**LinearLayout  
 android:orientation="horizontal"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"**>  
  
 <**TextView  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:text="Origen:"  
 android:textAppearance="@style/Base.TextAppearance.AppCompat.Body2"** />  
  
 <**TextView  
 android:id="@+id/txtOrigen"  
 android:text="getOrigen"  
 android:layout\_marginStart="10dp"  
 android:gravity="center\_vertical"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"** />  
  
 </**LinearLayout**>  
 <**LinearLayout  
 android:orientation="horizontal"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"**>  
 <**TextView  
 android:layout\_width="0dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_weight="1"  
 android:text="Destino:"  
 android:textAppearance="@style/Base.TextAppearance.AppCompat.Body2"** />  
 <**TextView  
 android:id="@+id/txtDestino"  
 android:text="getDestino"  
 android:layout\_marginStart="4dp"  
 android:gravity="center\_vertical"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"**/>  
 </**LinearLayout**>  
 <**View  
 android:layout\_width="wrap\_content"  
 android:layout\_height="3dp"  
 android:background="#DDFFDD"** />  
 <**LinearLayout  
 android:orientation="horizontal"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="30dp"**>  
 <**TextView  
 android:id="@+id/linearEliminar"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:gravity="center"  
 android:textSize="15sp"  
 android:drawablePadding="8dp"  
 android:text="Eliminar"  
 android:textAlignment="center"** />  
 <**Space  
 android:layout\_marginStart="10dp"  
 android:layout\_marginEnd="10dp"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="match\_parent"** />  
 <**TextView  
 android:id="@+id/linearActualizar"  
 android:text="Actualizar"  
 android:textSize="15sp"  
 android:drawablePadding="8dp"  
 android:gravity="center"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:textAlignment="center"**/>  
 <**Space  
 android:layout\_marginStart="10dp"  
 android:layout\_marginEnd="10dp"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="match\_parent"** />  
 <**TextView  
 android:id="@+id/linearAdicional"  
 android:text="Adicional"  
 android:textSize="15sp"  
 android:drawablePadding="8dp"  
 android:gravity="center"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:textAlignment="center"**/>  
 </**LinearLayout**>  
 </**LinearLayout**>  
 </**LinearLayout**>  
</**android.support.v7.widget.CardView**>

* Adaptador de la moneda
  + Clase

**public class** MonedaAdapter **extends** RecyclerView.Adapter<MonedaAdapter.Holder> **implements** Filterable {  
 **private** List<Moneda> **lMoneda**;  
 **private** List<Moneda> **mFiltered**;  
 **private** Context **context**;  
 **private final** MonedaClickListener **mlistener**;  
 **private static final** String ***TAG*** = ClienteAdapter.**class**.getSimpleName();  
 **public** MonedaAdapter(Context context, List<Moneda> objects, MonedaClickListener listener) {  
 **this**.**context** = context;  
 **this**.**lMoneda** = objects;  
 **this**.**mFiltered** = objects;  
 **this**.**mlistener** = listener;  
 }  
 @Override  
 **public** Filter getFilter() {  
 **return new** Filter() {  
 @Override  
 **protected** FilterResults performFiltering(CharSequence constraint) {  
 String charString = constraint.toString();  
 **if** (charString.isEmpty()) **mFiltered** = **lMoneda**;  
 **else**{  
 ArrayList<Moneda> filteredList = **new** ArrayList<>();  
 **for** (Moneda moneda : **lMoneda**) {  
 **if** (moneda.getMoneda().toLowerCase().contains(charString)) {  
 filteredList.add(moneda);  
 }  
 }  
 **mFiltered** = filteredList;  
 }  
 FilterResults filterResults = **new** FilterResults();  
 filterResults.**values** = **mFiltered**;  
 **return** filterResults;  
 }  
 @Override  
 **protected void** publishResults(CharSequence constraint, FilterResults results) {  
 **mFiltered** = (List<Moneda>) results.**values**;  
 notifyDataSetChanged();  
 }  
 };  
 }  
 **public class** Holder **extends** RecyclerView.ViewHolder **implements** View.OnClickListener, View.OnLongClickListener{  
 **private** TextView **main**, **detail**;  
 **public** Holder(View itemView) {  
 **super**(itemView);  
 **main** = (TextView) itemView.findViewById(R.id.***txtMain***);  
 **detail** = (TextView) itemView.findViewById(R.id.***txtDetail***);  
 itemView.setOnClickListener(**this**);  
 itemView.setOnLongClickListener(**this**);  
 }  
 @Override  
 **public void** onClick(View v) {  
 **mlistener**.onClick(getLayoutPosition());  
 }  
 @Override  
 **public boolean** onLongClick(View v) {  
 **mlistener**.onLongClick(getLayoutPosition());  
 **return true**;  
 }  
 }  
 **public interface** MonedaClickListener{  
 **void** onClick(**int** position);  
 **void** onLongClick(**int** position);  
 }  
 **public** Moneda getSelectedMoneda(**int** position) {  
 **return mFiltered**.get(position);  
 }  
  
 @Override  
 **public** MonedaAdapter.Holder onCreateViewHolder(ViewGroup parent, **int** viewType) {  
 View row = LayoutInflater.*from*(**context**).inflate(R.layout.***monedas\_item***, parent, **false**);  
 **return new** MonedaAdapter.Holder(row);  
 }  
 @Override  
 **public void** onBindViewHolder(Holder holder, **int** position) {  
 Moneda model = **mFiltered**.get(position);  
 holder.**main**.setText(model.getMoneda() + **" ("**+model.getID()+**")"**);  
 holder.**detail**.setText(**"Simbolo: "** + model.getSimbolo());  
 }  
 @Override  
 **public int** getItemCount() {  
 **return mFiltered**.size();  
 }  
}

* + Layout

<**android.support.v7.widget.CardView  
 xmlns:card\_view="http://schemas.android.com/apk/res-auto"  
 card\_view:cardCornerRadius="5dp"  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="5dp"  
 android:layout\_marginLeft="5dp"  
 android:layout\_marginRight="5dp"**>  
 <**LinearLayout  
 android:layout\_width="fill\_parent"  
 android:layout\_height="fill\_parent"  
 android:layout\_margin="5dp"  
 android:orientation="vertical"**>  
 <**TextView  
 android:layout\_width="match\_parent"  
 android:layout\_marginTop="5dp" android:layout\_marginBottom="5dp"  
 android:layout\_height="wrap\_content" android:textAppearance="@style/Base.TextAppearance.AppCompat.Title"  
 android:text="Large Text" android:id="@+id/txtMain"**/>  
 <**TextView  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content" android:textAppearance="@style/Base.TextAppearance.AppCompat.Caption"  
 android:text="Medium Text" android:id="@+id/txtDetail"**/>  
 </**LinearLayout**>  
</**android.support.v7.widget.CardView**>

* 1. Creación de métodos CRUD por fragmento
* Clientes
  + Activity

**public class** DetalleClienteActivity **extends** AppCompatActivity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 Utility.*setContext*(getApplicationContext());  
  
 setContentView(R.layout.***activity\_detalle\_cliente***);  
 Toolbar toolbar = (Toolbar) findViewById(R.id.***toolbar***);  
 setSupportActionBar(toolbar);  
  
 getSupportActionBar().setDisplayHomeAsUpEnabled(**true**);  
 getSupportActionBar().setHomeButtonEnabled(**true**);  
  
 Cliente model = (Cliente)getIntent().getSerializableExtra(ClientesFragment.***KEY\_DETALLE\_FLAG\_CLIENTE***);  
  
 DetalleClienteFragment fragment = (DetalleClienteFragment) getSupportFragmentManager().findFragmentById(R.id.***frameDetalleCliente***);  
 **if** (fragment == **null**){  
 fragment = DetalleClienteFragment.*newInstance*(model);  
 getSupportFragmentManager()  
 .beginTransaction()  
 .add(R.id.***frameDetalleCliente***, fragment)  
 .commit();  
 }  
 }  
  
 @Override  
 **public boolean** onCreateOptionsMenu(Menu menu) {  
 getMenuInflater().inflate(R.menu.***menu\_detalle\_cliente***, menu);  
 **return super**.onCreateOptionsMenu(menu);  
 }  
  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 **switch** (item.getItemId()){  
 **case** android.R.id.***home***:  
 setResult(Activity.***RESULT\_CANCELED***);  
 finish();  
 **break**;  
 }  
 **return super**.onOptionsItemSelected(item);  
 }  
  
 @Override  
 **protected void** onResume() {  
 **super**.onResume();  
 Utility.*closeGlobalNotification*(MainActivity.***NOTIFICATION\_PERMANENT***);  
 }  
  
 @Override  
 **protected void** onPause() {  
 **super**.onPause();  
 Utility.*showGlobalNotification*(**"El sistema de facturación aún se encuentra ejecutándose."**, MainActivity.**class**, MainActivity.***NOTIFICATION\_PERMANENT***);  
 }  
}

* + Fragment

**public class** ClientesFragment **extends** Fragment **implements** ClienteAdapter.ClienteClickListener {  
 **private** RecyclerView **rvCliente**;  
 **private** ClienteAdapter **aCliente**;  
 **private** SwipeRefreshLayout **refreshLayout**;  
 **private** FloatingActionButton **btnAgregar**;  
  
 **private** SearchView **searchView**;  
  
 **public static final int *KEY\_DIALOG\_ACTUALIZAR*** = 2;  
 **public static final int *KEY\_DIALOG\_AGREGAR*** = 3;  
 **public static final int *KEY\_DIALOG\_DETALLE\_ACTUALIZAR*** = 4;  
 **public static final** String ***KEY\_DETALLE\_FLAG\_CLIENTE*** = **"\_serializableCliente"**;  
 **public static final** String ***KEY\_DIALOG\_FLAG\_ACTUALIZAR*** = **"dialogActualizar"**;  
  
 **private** Session **session**;  
 **private** BroadcastReceiver **brUpdate**;  
  
 **public** ClientesFragment() {  
 }  
  
 @Override  
 **public** View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {  
 */\*\* Inflating the layout for this fragment \*\*/* View v = inflater.inflate(R.layout.***fragment\_clientes***, container, **false**);  
  
 setHasOptionsMenu(**true**);  
  
 Utility.*setContext*(getActivity());  
  
 **session** = **new** Session(getActivity());  
  
 **refreshLayout** = (SwipeRefreshLayout)v.findViewById(R.id.***activity\_swipe\_refresh\_layout\_clientes***);  
  
 **refreshLayout**.setOnRefreshListener(**new** SwipeRefreshLayout.OnRefreshListener() {  
 @Override  
 **public void** onRefresh() {  
 cargarData();  
 **refreshLayout**.setRefreshing(**false**);  
 }  
 });  
  
 **rvCliente** = (RecyclerView) v.findViewById(R.id.***RVClientes***);  
 **rvCliente**.setHasFixedSize(**true**);  
 **rvCliente**.setRecycledViewPool(**new** RecyclerView.RecycledViewPool());  
 **rvCliente**.setLayoutManager(**new** LinearLayoutManager(getActivity().getApplicationContext(), LinearLayoutManager.***VERTICAL***, **false**));  
  
 **btnAgregar** = (FloatingActionButton) v.findViewById(R.id.***fabAgregar***);  
  
 cargarData();  
  
 *//Background* Intent a = **new** Intent(getActivity(), VerifyClientes.**class**);  
 **boolean** alarmRunning = (PendingIntent.*getBroadcast*(getActivity(), 0, a, PendingIntent.***FLAG\_NO\_CREATE***) != **null**);  
 **if**(!alarmRunning) {  
 PendingIntent pendingIntent = PendingIntent.*getBroadcast*(getActivity(), 0, a, 0);  
 AlarmManager alarmManager = (AlarmManager) getActivity().getSystemService(Context.***ALARM\_SERVICE***);  
 alarmManager.setRepeating(AlarmManager.***ELAPSED\_REALTIME\_WAKEUP***, SystemClock.*elapsedRealtime*(), Constantes.BACKGROUND.Cliente.***TIME\_UPDATE\_UI***, pendingIntent);  
 }  
  
 **brUpdate** = **new** BroadcastReceiver() {  
 @Override  
 **public void** onReceive(Context context, Intent intent) { cargarData(); }  
 };  
  
 **btnAgregar**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 Intent i = **new** Intent(getActivity(), AgregarClienteActivity.**class**);  
 startActivityForResult(i, ***KEY\_DIALOG\_AGREGAR***);  
 }  
 });  
  
 **return** v;  
 }  
  
 @Override  
 **public void** onCreateOptionsMenu(Menu menu, MenuInflater inflater) {  
 menu.clear();  
 getActivity().getMenuInflater().inflate(R.menu.***menu\_search***, menu);  
  
 MenuItem search = menu.findItem(R.id.***search***);  
 **searchView** = (SearchView) MenuItemCompat.*getActionView*(search);  
 **searchView**.setQueryHint(**"Buscar..."**);  
 search(**searchView**);  
 }  
  
 **private void** search(SearchView searchView) {  
 searchView.setOnQueryTextListener(**new** SearchView.OnQueryTextListener() {  
 @Override  
 **public boolean** onQueryTextSubmit(String query) {  
 **return false**;  
 }  
  
 @Override  
 **public boolean** onQueryTextChange(String newText) {  
  
 **aCliente**.getFilter().filter(newText);  
 **return true**;  
 }  
 });  
 }  
  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 **return super**.onOptionsItemSelected(item);  
 }  
  
 **private void** cargarData() {  
 **new** findAllAsync(**false**, getActivity(), **new** findAllAsync.onResponseListener() {  
 @Override  
 **public void** responseClientes(List<Cliente> response) {  
 **if** (response != **null**){  
 **aCliente** = **new** ClienteAdapter(getActivity(), response, ClientesFragment.**this**);  
 **aCliente**.notifyDataSetChanged();  
 **rvCliente**.setAdapter(**aCliente**);  
  
 *//almacenar la lista en una sesion para trabajar en bg* **session**.setList(***SESSION\_SAVE\_LIST***,response);  
 }  
 }  
  
 @Override  
 **public void** showError(String error) {  
 **if** (error != **null** && error.length() > 0){  
 mostrarToast(error);  
 }  
 }  
 }).execute();  
 }  
  
 @Override  
 **public void** onClick(**int** position) {  
 Cliente selected = **aCliente**.getSelectedCliente(position);  
 irDetalle(selected);  
 }  
  
 **private void** irDetalle(Cliente selected) {  
 Intent i = **new** Intent(getActivity(), DetalleClienteActivity.**class**);  
 i.putExtra(***KEY\_DETALLE\_FLAG\_CLIENTE***, selected);  
 startActivityForResult(i, ***KEY\_DIALOG\_DETALLE\_ACTUALIZAR***);  
 }  
  
 @Override  
 **public void** onLongClick(**final int** position) {  
 **final** CharSequence[] items = {**"Seleccionar"**, **"Eliminar"**, **"Actualizar"**};  
 AlertDialog.Builder builder = **new** AlertDialog.Builder(getActivity());  
 builder.setTitle(**"Acciones: "**);  
 builder.setIcon(android.R.drawable.***ic\_input\_get***);  
 builder.setItems(items, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 **final** Cliente selected = **aCliente**.getSelectedCliente(position);  
 **switch** (which){  
 **case** 0:  
 irDetalle(selected);  
 **break**;  
 **case** 1:  
 **new** AlertDialog.Builder(getContext()).setTitle(**"Eliminar"**)  
 .setIcon(android.R.drawable.***ic\_dialog\_alert***)  
 .setMessage(**"¿Desea eliminar el siguiente registro?"**)  
 .setPositiveButton(**"Aceptar"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 ClienteService service = ClienteService.Factory.*getInstance*(getActivity());  
 service.deleteClientes(String.*valueOf*(selected.getID())).enqueue(**new** Callback<Boolean>() {  
 @Override  
 **public void** onResponse(Call<Boolean> call, Response<Boolean> response) {  
 **if** (response.isSuccessful()){  
 **if** (response.body()){  
 mostrarToast(**"Se actualizó el cliente."**);  
 cargarData();  
 }**else**{  
 mostrarToast(**"No se pudo actualizar el elemento."**);  
 }  
 }  
 **else**{  
 mostrarToast(**"Inconvenientes para consumir el servicio"**);  
 }  
 }  
  
 @Override  
 **public void** onFailure(Call<Boolean> call, Throwable t) {  
 Toast.*makeText*(getActivity(), **"Error: "**+ t.getCause().getMessage(), Toast.***LENGTH\_SHORT***).show();  
 }  
 });  
 }  
 })  
 .setNegativeButton(**"Cancelar"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 dialog.cancel();  
 }  
 }).show();  
 **break**;  
 **case** 2:  
 showActualizarDialog(***KEY\_DIALOG\_ACTUALIZAR***, selected);  
 **break**;  
 **default**:  
 Toast.*makeText*(getActivity(), **"Ninguna acción"**, Toast.***LENGTH\_SHORT***).show();  
 }  
 }  
 });  
  
 AlertDialog alertDialog = builder.create();  
 alertDialog.show();  
 }  
  
 **private void** showActualizarDialog(**int** keyDialogActualizar, Cliente selected) {  
 android.app.FragmentTransaction fragmentTransaction = getActivity().getFragmentManager().beginTransaction();  
 android.app.Fragment prev = getActivity().getFragmentManager().findFragmentByTag(***KEY\_DIALOG\_FLAG\_ACTUALIZAR***);  
 **if** (prev != **null**) {fragmentTransaction.remove(prev);}  
 fragmentTransaction.addToBackStack(**null**);  
  
 **switch** (keyDialogActualizar){  
 **case *KEY\_DIALOG\_ACTUALIZAR***:  
 DialogFragment dialogFragment = DialogActualizarCliente.*newInstance*(selected);  
 dialogFragment.setTargetFragment(**this**, ***KEY\_DIALOG\_ACTUALIZAR***);  
 dialogFragment.show(getFragmentManager().beginTransaction(), ***KEY\_DIALOG\_FLAG\_ACTUALIZAR***);  
 **break**;  
 }  
 }  
  
 **private void** showNotification(String texto){  
 NotificationCompat.Builder notification = **new** NotificationCompat  
 .Builder(getActivity())  
 .setSmallIcon(android.R.drawable.***stat\_sys\_warning***)  
 .setLargeIcon((((BitmapDrawable)getResources()  
 .getDrawable(R.drawable.***ic\_truck***)).getBitmap()))  
 .setContentTitle(**"Evento Cliente"**)  
 .setContentText(texto)  
 .setTicker(**"Visualizar"**);  
  
 Intent i = **new** Intent(getActivity().getApplicationContext(), MainActivity.**class**);  
 i.putExtra(**"menuFragment"**, **"ClientesFragment"**);  
  
 PendingIntent p = PendingIntent.*getActivity*(getActivity(), 0, i, 0);  
 notification.setContentIntent(p);  
  
 NotificationManager nm = (NotificationManager)getActivity().getSystemService(getActivity().***NOTIFICATION\_SERVICE***);  
 nm.notify(10, notification.build());  
  
 }  
  
 @Override  
 **public void** onResume() {  
 **super**.onResume();  
 getActivity().registerReceiver(**brUpdate**, **new** IntentFilter(Constantes.BACKGROUND.Cliente.***UPDATE\_UI\_FILTER***));  
 getView().setFocusableInTouchMode(**true**);  
 getView().requestFocus();  
 getView().setOnKeyListener(**new** View.OnKeyListener() {  
 @Override  
 **public boolean** onKey(View v, **int** keyCode, KeyEvent event) {  
 **if** (event.getAction() == KeyEvent.***ACTION\_UP*** && keyCode == KeyEvent.***KEYCODE\_BACK***){  
 cargarData();  
 **return true**;  
 }  
 **return false**;  
 }  
 });  
 }  
  
 @Override  
 **public void** onPause() {  
 **super**.onPause();  
 getActivity().unregisterReceiver(**brUpdate**);  
 }  
  
 @Override  
 **public void** onDestroy() {  
 **super**.onDestroy();  
 **session**.setList(***SESSION\_SAVE\_LIST***, **null**);  
 }  
  
 @Override  
 **public void** onActivityResult(**int** requestCode, **int** resultCode, Intent data) {  
 **switch** (requestCode){  
 **case *KEY\_DIALOG\_AGREGAR***:  
 **if** (resultCode == Activity.***RESULT\_OK***) cargarData();  
 **break**;  
 **case *KEY\_DIALOG\_DETALLE\_ACTUALIZAR***:  
 **if** (resultCode == Activity.***RESULT\_OK***) cargarData();  
 **break**;  
 **case *KEY\_DIALOG\_ACTUALIZAR***:  
 **if** (resultCode == Activity.***RESULT\_OK***){  
 Cliente getDialog = (Cliente) data.getSerializableExtra(DialogActualizarCliente.***KEY\_MODEL***);  
 ClienteService service = ClienteService.Factory.*getInstance*(getActivity());  
 service.updateClientes(getDialog).enqueue(**new** Callback<Boolean>() {  
 @Override  
 **public void** onResponse(Call<Boolean> call, Response<Boolean> response) {  
 **if** (response.isSuccessful()){  
 **if** (response.body()){  
 mostrarToast(**"Se actualizó el cliente."**);  
 cargarData();  
 }**else**{  
 mostrarToast(**"No se pudo actualizar el elemento."**);  
 }  
 }  
 **else**{  
 mostrarToast(**"Inconvenientes para consumir el servicio"**);  
 }  
 }  
  
 @Override  
 **public void** onFailure(Call<Boolean> call, Throwable t) {  
 Toast.*makeText*(getActivity(), **"Error: "**+ t.getCause().getMessage(), Toast.***LENGTH\_SHORT***).show();  
 }  
 });  
 }  
 **break**;  
 }  
 }  
  
 **public void** mostrarToast(String mensaje){  
 Toast.*makeText*(getActivity(), mensaje, Toast.***LENGTH\_LONG***).show();  
 }  
}

* Detalle de la factura
  + Activity

**public class** AdicionalFacturaActivity **extends** AppCompatActivity {  
  
 **private** Boolean **\_flagTabAgregar**;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_adicional\_factura***);  
  
 getSupportActionBar().setHomeButtonEnabled(**true**);  
 getSupportActionBar().setDisplayHomeAsUpEnabled(**true**);  
  
 Factura model = (Factura) getIntent().getSerializableExtra(DetalleFacturaFragment.***KEY\_ARG\_ADICIONAL***);  
  
 **\_flagTabAgregar** = getIntent().getBooleanExtra(TabAgregarFacturas.***FLAG\_ARG***, **false**);  
  
 setTitle(**"Información: "** + model.getCom\_Id());  
  
 AdicionalFacturaFragment fragment = (AdicionalFacturaFragment) getSupportFragmentManager().findFragmentById(R.id.***frameAdicionalFactura***);  
 **if** (fragment == **null**){  
 fragment = AdicionalFacturaFragment.*newInstance*(model);  
 getSupportFragmentManager().beginTransaction()  
 .add(R.id.***frameAdicionalFactura***, fragment)  
 .commit();  
 }  
 }  
  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 **switch** (item.getItemId()){  
 **case** android.R.id.***home***:  
 **if** (**\_flagTabAgregar**) *//Si viene del tab de agregar factura* irMainActivity();  
 **else** goBack(TabListaAdicionalFactura.*modified*);  
 **break**;  
 }  
 **return super**.onOptionsItemSelected(item);  
 }  
  
 **private void** goBack(Boolean been){  
 **if** (been){ *//Si se modifico en la lista anterior de detalles* TabListaAdicionalFactura.*modified* = **false**; *//volvemos a inicializar el var global* irMainActivity();  
 }**else**{  
 finish();  
 }  
 }  
  
 **private void** irMainActivity() {  
 Intent i = **new** Intent(**this**, MainActivity.**class**);  
 i.addFlags(Intent.***FLAG\_ACTIVITY\_CLEAR\_TASK*** | Intent.***FLAG\_ACTIVITY\_NEW\_TASK***);  
 startActivity(i);  
 }  
}

* + Fragment

**public class** AdicionalFacturaFragment **extends** Fragment{  
 **private** Factura **mFactura**;  
  
 **public** AdicionalFacturaFragment() {}  
  
 **public static** AdicionalFacturaFragment newInstance(Factura model) {  
 AdicionalFacturaFragment fragmentDemo = **new** AdicionalFacturaFragment();  
 Bundle args = **new** Bundle();  
 args.putSerializable(**"\_data"**, model);  
 fragmentDemo.setArguments(args);  
 **return** fragmentDemo;  
 }  
  
 @Override  
 **public void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 **if**(getArguments() != **null**){  
 *// Get back arguments* **mFactura** = (Factura)getArguments().getSerializable(**"\_data"**);  
 }  
 }  
@Override  
 **public** View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {  
 */\*\* Inflating the layout for this fragment \*\*/* View root = inflater.inflate(R.layout.***tab\_layout\_adicionalfacturas***, container, **false**);  
 TabLayout tabLayout = (TabLayout) root.findViewById(R.id.***tab\_layout\_adicionalfacturas***);  
 tabLayout.addTab(tabLayout.newTab().setText(**"Agregar"**));  
 tabLayout.addTab(tabLayout.newTab().setText(**"Lista"**));  
 tabLayout.setTabGravity(TabLayout.***GRAVITY\_FILL***);  
  
 **final** ViewPager viewPager = (ViewPager) root.findViewById(R.id.***pager\_adicionalfacturas***);  
 **final** PagerAdapter adapter = **new** PagerAdapterAdicionalFactura(getActivity().getSupportFragmentManager(), tabLayout.getTabCount(), **mFactura**);  
 viewPager.setAdapter(adapter);  
 viewPager.addOnPageChangeListener(**new** TabLayout.TabLayoutOnPageChangeListener(tabLayout));  
  
 tabLayout.setOnTabSelectedListener(**new** TabLayout.OnTabSelectedListener() {  
 @Override  
 **public void** onTabSelected(TabLayout.Tab tab) {  
 viewPager.setCurrentItem(tab.getPosition());  
 }  
  
 @Override  
 **public void** onTabUnselected(TabLayout.Tab tab) {  
  
 }  
  
 @Override  
 **public void** onTabReselected(TabLayout.Tab tab) {  
  
 }  
 });  
 **return** root;  
 }  
}

* Facturas
  + Activity

**public class** DetalleFacturaActivity **extends** AppCompatActivity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_detalle\_factura***);  
 Toolbar toolbar = (Toolbar) findViewById(R.id.***toolbar***);  
 setSupportActionBar(toolbar);  
  
 getSupportActionBar().setDisplayHomeAsUpEnabled(**true**);  
  
 Factura model = (Factura) getIntent().getSerializableExtra(TabListaFacturas.***KEY\_DETALLE\_FLAG\_CLIENTE***);  
  
 DetalleFacturaFragment fragment = (DetalleFacturaFragment) getSupportFragmentManager().findFragmentById(R.id.***frameDetalleFactura***);  
 **if** (fragment == **null**){  
 fragment = DetalleFacturaFragment.*newInstance*(model);  
 getSupportFragmentManager()  
 .beginTransaction()  
 .add(R.id.***frameDetalleFactura***, fragment)  
 .commit();  
 }  
  
 Utility.*setContext*(getApplicationContext());  
 }  
  
 @Override  
 **public boolean** onCreateOptionsMenu(Menu menu) {  
 getMenuInflater().inflate(R.menu.***menu\_detalle\_factura***, menu);  
 **return super**.onCreateOptionsMenu(menu);  
 }  
  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 **switch** (item.getItemId()){  
 **case** android.R.id.***home***:  
 finish();  
 **break**;  
 }  
 **return super**.onOptionsItemSelected(item);  
 }  
  
 @Override  
 **protected void** onResume() {  
 **super**.onResume();  
 Utility.*closeGlobalNotification*(MainActivity.***NOTIFICATION\_PERMANENT***);  
 }  
  
 @Override  
 **protected void** onPause() {  
 **super**.onPause();  
 Utility.*showGlobalNotification*(**"El sistema de facturación aún se encuentra ejecutándose."**, MainActivity.**class**, MainActivity.***NOTIFICATION\_PERMANENT***);  
 }  
}

* + Fragment

**public class** DetalleFacturaFragment **extends** Fragment **implements** View.OnClickListener {  
  
 **private** EditText **txtIdentificador**, **txtiFecha**, **txtiPOrigen**, **txtiPDestino**, **txtiCondiciones**, **txtiGCarga**;  
 **private** Spinner **spCliente**;  
 **private** Button **btniEliminar**, **btniActualizar**, **btniRegresar**, **btniAdicional**;  
  
 **private** CollapsingToolbarLayout **mCollapsing**;  
  
 **private** Factura **iFactura**;  
  
 **private** DatePickerDialog **fromDatePickerDialog**;  
 **private** SimpleDateFormat **dateFormatter**;  
  
 **public static final** String ***KEY\_ARG\_ADICIONAL*** = **"\_serializableFactura"**;  
 **public static final int *RESPONSE\_ARG\_ADICIONAL*** = 1;  
  
 **public** DetalleFacturaFragment() {  
 *// Required empty public constructor* }  
  
 **public static** DetalleFacturaFragment newInstance(Factura model) {  
 DetalleFacturaFragment fragmentDemo = **new** DetalleFacturaFragment();  
 Bundle args = **new** Bundle();  
 args.putSerializable(**"data"**, model);  
 fragmentDemo.setArguments(args);  
 **return** fragmentDemo;  
 }  
  
 @Override  
 **public void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 **if**(getArguments() != **null**){  
 *// Get back arguments* **iFactura** = (Factura)getArguments().getSerializable(**"data"**);  
 }  
 setHasOptionsMenu(**true**);  
 }  
  
 @Override  
 **public** View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {  
 */\*\* Inflating the layout for this fragment \*\*/* View root = inflater.inflate(R.layout.***fragment\_detalle\_facturas***, container, **false**);  
 Locale spanish = **new** Locale(**"es"**, **"ES"**);  
 Locale.*setDefault*(spanish);  
  
 **mCollapsing** = (CollapsingToolbarLayout) getActivity().findViewById(R.id.***toolbar\_layout***);  
 **mCollapsing**.setTitle(**"Detalle de: "** + **iFactura**.getCom\_Id());  
  
 **dateFormatter** = **new** SimpleDateFormat(**"dd-MM-yyyy"**, spanish);  
 **txtiFecha** = (EditText) root.findViewById(R.id.***txtFecha***);  
 **txtiFecha**.setInputType(InputType.***TYPE\_NULL***);  
  
 **txtIdentificador** = (EditText) root.findViewById(R.id.***txtIdentificador***);  
 **txtiPOrigen** = (EditText) root.findViewById(R.id.***txtPOrigen***);  
 **txtiPDestino** = (EditText) root.findViewById(R.id.***txtPDestino***);  
 **txtiCondiciones** = (EditText) root.findViewById(R.id.***txtCondiciones***);  
 **txtiGCarga** = (EditText) root.findViewById(R.id.***txtGCarga***);  
  
 **spCliente** = (Spinner) root.findViewById(R.id.***spCliente***);  
  
 **btniEliminar** = (Button) root.findViewById(R.id.***btnEliminarFacturas***);  
 **btniActualizar** = (Button) root.findViewById(R.id.***btnActualizarFacturas***);  
 **btniRegresar** = (Button) root.findViewById(R.id.***btnRegresar***);  
  
 **btniAdicional** = (Button) root.findViewById(R.id.***btnIrAdicional***);  
 **btniAdicional**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 irAAdicional();  
 }  
 });  
  
 **btniEliminar**.setOnClickListener(**this**);  
 **btniActualizar**.setOnClickListener(**this**);  
 **btniRegresar**.setOnClickListener(**this**);  
  
 **txtiFecha**.setText(**iFactura**.getFecha());  
 **txtIdentificador**.setText(**iFactura**.getCom\_Id());  
 **txtiPOrigen**.setText(**iFactura**.getPuertoOrigen());  
 **txtiPDestino**.setText(**iFactura**.getPuertoDestino());  
 **txtiCondiciones**.setText(**iFactura**.getCondiciones());  
 **txtiGCarga**.setText(**iFactura**.getGuiaCarga());  
  
 setDateTimeField();  
 cargarItems();  
 bloquearEntradas();  
 **return** root;  
 }  
  
 **private void** irAAdicional() {  
 String \_cliente = **spCliente**.getSelectedItem().toString();  
 **int** \_idCliente = Integer.*parseInt*(\_cliente.substring(\_cliente.indexOf(**"["**) + 1, \_cliente.indexOf(**"]"**)));  
 String \_fecha = **txtiFecha**.getText().toString();  
 String \_po = **txtiPOrigen**.getText().toString();  
 String \_pd = **txtiPDestino**.getText().toString();  
 String \_co = **txtiCondiciones**.getText().toString();  
 String \_gc = **txtiGCarga**.getText().toString();  
  
 Factura model = **new** Factura();  
 model.setID(**iFactura**.getID());  
 model.setCom\_Id(**iFactura**.getCom\_Id());  
 model.setIdCliente(\_idCliente);  
 model.setFecha(\_fecha);  
 model.setPuertoOrigen(\_po);  
 model.setPuertoDestino(\_pd);  
 model.setCondiciones(\_co);  
 model.setGuiaCarga(\_gc);  
  
 Intent i = **new** Intent(getActivity(), AdicionalFacturaActivity.**class**);  
 i.putExtra(***KEY\_ARG\_ADICIONAL***, model);  
 startActivityForResult(i, ***RESPONSE\_ARG\_ADICIONAL***);  
 }  
  
 **private void** bloquearEntradas() {  
 **spCliente**.setEnabled(**false**);  
 **txtiFecha**.setEnabled(**false**);  
 **txtIdentificador**.setEnabled(**false**);  
 **txtiPOrigen**.setEnabled(**false**);  
 **txtiPDestino**.setEnabled(**false**);  
 **txtiCondiciones**.setEnabled(**false**);  
 **txtiGCarga**.setEnabled(**false**);  
 }  
  
 **private void** desbloquearEntradas() {  
 **spCliente**.setEnabled(**true**);  
 **txtiFecha**.setEnabled(**true**);  
 **txtiPOrigen**.setEnabled(**true**);  
 **txtiPDestino**.setEnabled(**true**);  
 **txtiCondiciones**.setEnabled(**true**);  
 **txtiGCarga**.setEnabled(**true**);  
 }  
  
 **private void** setDateTimeField() {  
 **txtiFecha**.setOnFocusChangeListener(**new** View.OnFocusChangeListener() {  
 @Override  
 **public void** onFocusChange(View v, **boolean** hasFocus) {  
 **if**(hasFocus) **fromDatePickerDialog**.show();  
 }  
 });  
  
 **txtiFecha**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **if**(!**txtiFecha**.getText().toString().isEmpty()){  
 String \_date = **txtiFecha**.getText().toString();  
  
 SimpleDateFormat format = **new** SimpleDateFormat(**"dd-MM-yyyy"**, **new** Locale(**"es"**, **"ES"**));  
 **try** {  
 Date date = format.parse(\_date);  
 } **catch** (ParseException e) {  
 e.printStackTrace();  
 }  
 }  
  
 **fromDatePickerDialog**.show();  
 }  
 });  
  
 Calendar c = Calendar.*getInstance*();  
 **int** mYear = c.get(Calendar.***YEAR***);  
 **int** mMonth = c.get(Calendar.***MONTH***);  
 **int** mDay = c.get(Calendar.***DAY\_OF\_MONTH***);  
  
 **fromDatePickerDialog** = **new** DatePickerDialog(getActivity(), **new** DatePickerDialog.OnDateSetListener() {  
  
 @Override  
 **public void** onDateSet(DatePicker view, **int** year, **int** month, **int** dayOfMonth) {  
 Calendar newDate = Calendar.*getInstance*();  
 newDate.set(year, month, dayOfMonth);  
 **txtiFecha**.setText(**dateFormatter**.format(newDate.getTime()));  
 }  
  
 },mYear, mMonth, mDay);  
 **fromDatePickerDialog**.getDatePicker().setMaxDate(c.getTimeInMillis());  
 }  
  
 @Override  
 **public void** onClick(View v) {  
 **switch** (v.getId()){  
 **case** R.id.***btnEliminarFacturas***:  
 EliminarFactura(String.*valueOf*(**iFactura**.getID()));  
 **break**;  
 **case** R.id.***btnActualizarFacturas***:  
 **if** (**btniEliminar**.isEnabled()){  
 *//deshabilito txtSimbolo y abreviatura* desbloquearEntradas();  
  
 **btniEliminar**.setEnabled(**false**);  
 **btniEliminar**.setVisibility(View.***GONE***);  
  
 **btniActualizar**.setText(**"APLICAR"**);  
 }**else**{  
 **if** (!validate()){  
 Toast.*makeText*(getActivity(), **"Error al agregar el nuevo elemento."**, Toast.***LENGTH\_SHORT***).show();  
 **return**;  
 }  
  
 String \_cliente = **spCliente**.getSelectedItem().toString();  
 **int** \_idCliente = Integer.*parseInt*(\_cliente.substring(\_cliente.indexOf(**"["**) + 1, \_cliente.indexOf(**"]"**)));  
 String \_fecha = **txtiFecha**.getText().toString();  
 String \_po = **txtiPOrigen**.getText().toString();  
 String \_pd = **txtiPDestino**.getText().toString();  
 String \_co = **txtiCondiciones**.getText().toString();  
 String \_gc = **txtiGCarga**.getText().toString();  
  
 Factura model = **new** Factura();  
 model.setID(**iFactura**.getID());  
 model.setIdCliente(\_idCliente);  
 model.setFecha(\_fecha);  
 model.setPuertoOrigen(\_po);  
 model.setPuertoDestino(\_pd);  
 model.setCondiciones(\_co);  
 model.setGuiaCarga(\_gc);  
  
 ActualizarFactura(model);  
 }  
 **break**;  
 **case** R.id.***btnRegresar***:  
 getActivity().setResult(Activity.***RESULT\_CANCELED***);  
 getActivity().finish();  
 **break**;  
 }  
 }  
  
 **private boolean** validate() {  
 **boolean** rpt = **true**;  
  
 **if** (**txtiPOrigen**.getText().toString().isEmpty()){  
 **txtiPOrigen**.setError(**"Asignar puerto de origen"**);  
 **txtiPOrigen**.requestFocus();  
 rpt = **false**;  
 }  
  
 **if** (**txtiPDestino**.getText().toString().isEmpty()){  
 **txtiPDestino**.setError(**"Asignar puerto de destino"**);  
 **txtiPDestino**.requestFocus();  
 rpt = **false**;  
 }  
  
 **return** rpt;  
 }  
  
 **public void** irALista(){  
 getActivity().setResult(Activity.***RESULT\_OK***);  
 getActivity().finish();  
 }  
  
 **public void** mostrarToast(String mensaje){  
 Toast.*makeText*(getActivity(), mensaje, Toast.***LENGTH\_LONG***).show();  
 }  
  
 **private void** EliminarFactura(**final** String \_id){  
 AlertDialog.Builder builder = **new** AlertDialog.Builder(getActivity(), android.R.style.***Theme\_Material\_Dialog\_Alert***);  
 builder.setTitle(**"Mensaje del sistema"**)  
 .setMessage(**"¿Seguro que desea eliminar la factura "**+**iFactura**.getCom\_Id()+**"?"**)  
 .setCancelable(**true**)  
 .setPositiveButton(**"Sí"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
  
 **new** deleteAsync(getActivity(), \_id, **new** deleteAsync.onResponseListener() {  
 @Override  
 **public void** getResponse(Boolean response) {  
 **if** (response) irALista();  
 **else** mostrarToast(**"No se pudo eliminar la factura."**);  
 }  
 }).execute();  
  
 }  
 })  
 .setNegativeButton(**"No"**, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 dialog.cancel();  
 }  
 }).setIcon(android.R.drawable.***ic\_dialog\_alert***);  
 builder.show();  
 }  
  
 **private void** ActualizarFactura(**final** Factura model){  
 **new** updateAsync(getActivity(), model, **new** updateAsync.onResponseListener() {  
 @Override  
 **public void** getResponse(Boolean response) {  
 **if** (response) irALista();  
 **else** mostrarToast(**"No se pudo actualizar la factura."**);  
 }  
 }).execute();  
 }  
  
 *//Cargar cliente spinner* **public void** cargarItems(){  
 Handler mainHandler = **new** Handler(getActivity().getMainLooper());  
  
 Runnable s = **new** Runnable() {  
 @Override  
 **public void** run() {  
 ClienteService service = ClienteService.Factory.*getInstance*(getActivity());  
 **try** {  
 List<Cliente> lista = service.getClientes().execute().body();  
 Cliente[] it = **new** Cliente[lista.size()];  
 it = lista.toArray(it);  
  
 String[] m = **new** String[it.**length**];  
  
 **int** idCliente = **iFactura**.getIdCliente();  
 **int** \_savPosition = -1;  
  
 **for** (**int** i = 0; i < lista.size(); i++){  
 m[i] = **"["**+lista.get(i).getID()+**"] "** + lista.get(i).getCliente();  
 **if** (lista.get(i).getID() == idCliente) \_savPosition = i;  
 }  
  
 ArrayAdapter<String> adapter = **new** ArrayAdapter<String>(getActivity(),  
 android.R.layout.***simple\_spinner\_item***, m);  
  
 adapter.setDropDownViewResource(android.R.layout.***simple\_spinner\_dropdown\_item***);  
  
 **spCliente**.setAdapter(adapter);  
  
 **spCliente**.setSelection(\_savPosition);  
  
 } **catch** (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 };  
  
 mainHandler.post(s);  
 }  
  
 @Override  
 **public void** onResume() {  
 **super**.onResume();  
 getView().setFocusableInTouchMode(**true**);  
 getView().requestFocus();  
 getView().setOnKeyListener(**new** View.OnKeyListener() {  
 @Override  
 **public boolean** onKey(View v, **int** keyCode, KeyEvent event) {  
 **if** (event.getAction() == KeyEvent.***ACTION\_UP*** && keyCode == KeyEvent.***KEYCODE\_BACK***){  
 irALista();  
 **return true**;  
 }  
 **return false**;  
 }  
 });  
 }  
  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 **switch** (item.getItemId()) {  
 **case** R.id.***action\_regresar***:  
 getActivity().setResult(Activity.***RESULT\_CANCELED***);  
 getActivity().finish();  
 **break**;  
 **case** R.id.***action\_adicional***:  
 irAAdicional();  
 **break**;  
 }  
 **return super**.onOptionsItemSelected(item);  
 }  
}

* Monedas
  + Activity

**public class** DetalleMonedaActivity **extends** AppCompatActivity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_detalle\_moneda***);  
 Toolbar toolbar = (Toolbar) findViewById(R.id.***toolbar***);  
 setSupportActionBar(toolbar);  
  
 getSupportActionBar().setDisplayHomeAsUpEnabled(**true**);  
 getSupportActionBar().setHomeButtonEnabled(**true**);  
  
 Moneda model = (Moneda) getIntent().getSerializableExtra(MonedasFragment.***KEY\_DETALLE\_FLAG\_MONEDA***);  
  
 DetalleMonedaFragment mfragment = (DetalleMonedaFragment) getSupportFragmentManager().findFragmentById(R.id.***frameDetalleMoneda***);  
 **if** (mfragment == **null**){  
 mfragment = DetalleMonedaFragment.*newInstance*(model);  
 getSupportFragmentManager().beginTransaction()  
 .addToBackStack(**null**)  
 .add(R.id.***frameDetalleMoneda***, mfragment)  
 .commit();  
 }  
  
 Utility.*setContext*(getApplicationContext());  
 }  
  
 @Override  
 **public boolean** onCreateOptionsMenu(Menu menu) {  
 getMenuInflater().inflate(R.menu.***menu\_detalle\_moneda***, menu);  
 **return super**.onCreateOptionsMenu(menu);  
 }  
  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 **switch** (item.getItemId()){  
 **case** android.R.id.***home***:  
 setResult(Activity.***RESULT\_CANCELED***);  
 finish();  
 **break**;  
 }  
 **return super**.onOptionsItemSelected(item);  
 }  
  
 @Override  
 **protected void** onResume() {  
 **super**.onResume();  
 Utility.*closeGlobalNotification*(MainActivity.***NOTIFICATION\_PERMANENT***);  
 }  
  
 @Override  
 **protected void** onPause() {  
 **super**.onPause();  
 Utility.*showGlobalNotification*(**"El sistema de facturación aún se encuentra ejecutándose."**, MainActivity.**class**, MainActivity.***NOTIFICATION\_PERMANENT***);  
 }  
}

* + Fragment

**public class** DetalleMonedaFragment **extends** Fragment {  
 **private** EditText **iMoneda**, **iSimbolo**, **iAbreviatura**;  
 **private** Moneda **uMoneda**;  
 **private** Button **btnEliminar**, **btnActualizar**, **btnRegresar**;  
  
 **public** DetalleMonedaFragment (){}  
  
 **public static** DetalleMonedaFragment newInstance(Moneda model) {  
 DetalleMonedaFragment fragmentDemo = **new** DetalleMonedaFragment();  
 Bundle args = **new** Bundle();  
 args.putSerializable(**"data"**, model);  
 fragmentDemo.setArguments(args);  
 **return** fragmentDemo;  
 }  
  
 @Override  
 **public void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 **if**(getArguments() != **null**){  
 *// Get back arguments* **uMoneda** = (Moneda)getArguments().getSerializable(**"data"**);  
 }  
 setHasOptionsMenu(**true**);  
 }  
  
 @Override  
 **public** View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {  
 */\*\* Inflating the layout for this fragment \*\*/* View root = inflater.inflate(R.layout.***fragment\_detalle\_monedas***, container, **false**);  
  
 **iMoneda** = (EditText)root.findViewById(R.id.***txtMoneda***);  
 **iSimbolo** = (EditText)root.findViewById(R.id.***txtSimboloMoneda***);  
 **iAbreviatura** = (EditText)root.findViewById(R.id.***txtMonedaAbreviatura***);  
 **btnEliminar** = (Button)root.findViewById(R.id.***btnEliminarMonedas***);  
 **btnActualizar** = (Button)root.findViewById(R.id.***btnActualizarMonedas***);  
 **btnRegresar** = (Button)root.findViewById(R.id.***btnRegresar***);  
  
 **iMoneda**.setText(**uMoneda**.getMoneda());  
 **iSimbolo**.setText(**uMoneda**.getSimbolo());  
 **iAbreviatura**.setText(**uMoneda**.getID());  
  
 **iMoneda**.setEnabled(**false**);  
 **iSimbolo**.setEnabled(**false**);  
 **iAbreviatura**.setEnabled(**false**);  
  
 **btnRegresar**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 getActivity().setResult(Activity.***RESULT\_CANCELED***);  
 getActivity().finish();  
 }  
 });  
  
 **btnEliminar**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 AlertDialog.Builder builder = **new** AlertDialog.Builder(getActivity(), android.R.style.***Theme\_Material\_Light\_Dialog\_Alert***);  
 builder.setTitle(**"Mensaje del sistema"**)  
 .setMessage(**"¿Seguro que desea eliminar la moneda "**+**uMoneda**.getMoneda().toUpperCase()+**"?"**)  
 .setCancelable(**true**)  
 .setPositiveButton(android.R.string.***yes***, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 **new** deleteAsync(getActivity(), **uMoneda**.getID().trim(), **new** deleteAsync.onResponseListener() {  
 @Override  
 **public void** getResponse(Boolean response) {  
 **if** (!response){  
 Toast.*makeText*(getActivity(), **"Error al eliminar la moneda: "**+ **uMoneda**.getMoneda().toUpperCase(), Toast.***LENGTH\_SHORT***).show();  
 **return**;  
 }**else**{  
 Toast.*makeText*(getActivity(), **"Se eliminó la moneda: "**+ **uMoneda**.getMoneda().toUpperCase(), Toast.***LENGTH\_SHORT***).show();  
 irALista();  
 }  
 }  
 }).execute();  
 }  
 })  
 .setNegativeButton(android.R.string.***no***, **new** DialogInterface.OnClickListener() {  
 @Override  
 **public void** onClick(DialogInterface dialog, **int** which) {  
 dialog.cancel();  
 }  
 }).setIcon(android.R.drawable.***ic\_dialog\_alert***);  
 builder.show();  
 }  
 });  
  
 **btnActualizar**.setOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 **if** (**btnEliminar**.isEnabled()){  
 *//deshabilito txtSimbolo y abreviatura* **iSimbolo**.setEnabled(**true**);  
 **iMoneda**.setEnabled(**true**);  
  
 **btnEliminar**.setEnabled(**false**);  
 **btnEliminar**.setVisibility(View.***GONE***);  
  
 **btnActualizar**.setText(**"APLICAR"**);  
 }**else**{  
 **if** (!validate()){  
 **return**;  
 }  
  
 MonedaService service = MonedaService.Factory.*getInstance*(getActivity(), **null**);  
 Moneda model = **new** Moneda();  
 model.setID(**uMoneda**.getID());  
 model.setSimbolo(**iSimbolo**.getText().toString());  
 model.setMoneda(**iMoneda**.getText().toString());  
  
 service.updateMonedas(model).enqueue(**new** Callback<Boolean>() {  
 @Override  
 **public void** onResponse(Call<Boolean> call, Response<Boolean> response) {  
 **if** (response.isSuccessful()){  
 **if** (response.body()){  
 mostrarToast(**"Se actualizó la moneda."**);  
 irALista();  
 }**else** mostrarToast(**"No se pudo actualizar el elemento."**);  
 }  
 **else** mostrarToast(**"Inconvenientes para consumir el servicio."**);  
 }  
  
 @Override  
 **public void** onFailure(Call<Boolean> call, Throwable t) {  
 mostrarToast(**"Error: "**+ t.getCause().getMessage());  
 }  
 });  
  
 **iSimbolo**.setEnabled(**false**);  
 **iMoneda**.setEnabled(**false**);  
 **btnEliminar**.setVisibility(View.***VISIBLE***);  
 **btnEliminar**.setEnabled(**true**);  
 **btnActualizar**.setText(**"ACTUALIZAR"**);  
 }  
 }  
 });  
  
 **return** root;  
 }  
  
 **public boolean** validate() {  
 **boolean** valid = **true**;  
  
 String \_moneda = **iMoneda**.getText().toString();  
 String \_simbolo = **iSimbolo**.getText().toString();  
 String \_abreviatura = **iAbreviatura**.getText().toString();  
  
 **if** (\_moneda.isEmpty()) {  
 **iMoneda**.setError(**"Debe completar el campo nombre."**);  
 **iMoneda**.requestFocus();  
 valid = **false**;  
 } **else if**(\_moneda.length() > 30) {  
 **iMoneda**.setError(**"El tamaño máximo de caracteres es 30."**);  
 **iMoneda**.requestFocus();  
 valid = **false**;  
 }**else**{  
 **iMoneda**.setError(**null**);  
 }  
  
 **if** (\_simbolo.isEmpty() || \_simbolo.length() > 3) {  
 **iSimbolo**.setError(**"Máximo 3 caracteres."**);  
 **iSimbolo**.requestFocus();  
 valid = **false**;  
 } **else** {  
 **iSimbolo**.setError(**null**);  
 }  
  
 **if** (\_abreviatura.isEmpty() || \_abreviatura.length() > 6) {  
 **iAbreviatura**.setError(**"Máximo 6 caracteres."**);  
 **iAbreviatura**.requestFocus();  
 valid = **false**;  
 } **else** {  
 **iAbreviatura**.setError(**null**);  
 }  
  
 **return** valid;  
 }  
  
 **public void** irALista(){  
 getActivity().setResult(Activity.***RESULT\_OK***);  
 getActivity().finish();  
 }  
  
 **public void** mostrarToast(String mensaje){  
 Toast.*makeText*(getActivity(), mensaje, Toast.***LENGTH\_LONG***).show();  
 }  
  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 **switch** (item.getItemId()) {  
 **case** R.id.***action\_regresar***:  
 getActivity().setResult(Activity.***RESULT\_CANCELED***);  
 getActivity().finish();  
 **break**;  
 }  
 **return super**.onOptionsItemSelected(item);  
 }  
}

* Perfil
  + Activity

**public class** PerfilActivity **extends** AppCompatActivity {  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_perfil***);  
 Toolbar toolbar = (Toolbar) findViewById(R.id.***toolbar***);  
 setSupportActionBar(toolbar);  
  
 String \_cuenta = getIntent().getStringExtra(MainActivity.***ARG\_KEY\_CUENTA***);  
  
 **if**(\_cuenta.isEmpty()) finish();  
  
 getSupportActionBar().setDisplayHomeAsUpEnabled(**true**);  
 getSupportActionBar().setDisplayHomeAsUpEnabled(**true**);  
  
 PerfilFragment fragment = (PerfilFragment) getSupportFragmentManager().findFragmentById(R.id.***framePerfil***);  
 **if** (fragment == **null**){  
 fragment = PerfilFragment.*newInstance*(\_cuenta);  
 getSupportFragmentManager().beginTransaction().add(R.id.***framePerfil***, fragment).commit();  
 }  
  
 Utility.*setContext*(getApplicationContext());  
 }  
  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 **switch** (item.getItemId()){  
 **case** android.R.id.***home***:  
 setResult(Activity.***RESULT\_CANCELED***);  
 finish();  
 **break**;  
 }  
 **return super**.onOptionsItemSelected(item);  
 }  
  
 @Override  
 **public boolean** onCreateOptionsMenu(Menu menu) {  
 getMenuInflater().inflate(R.menu.***menu\_perfil***, menu);  
 **return super**.onCreateOptionsMenu(menu);  
 }  
  
 @Override  
 **protected void** onResume() {  
 **super**.onResume();  
 Utility.*closeGlobalNotification*(MainActivity.***NOTIFICATION\_PERMANENT***);  
 }  
  
 @Override  
 **protected void** onPause() {  
 **super**.onPause();  
 Utility.*showGlobalNotification*(**"El sistema de facturación aún se encuentra ejecutándose."**, MainActivity.**class**, MainActivity.***NOTIFICATION\_PERMANENT***);  
 }  
}

* + Fragment

**public class** PerfilFragment **extends** Fragment **implements** View.OnClickListener {  
 **private** Session **\_session**;  
 **private** EditText **txtiNombre**, **txtiApePaterno**, **txtiApeMaterno**;  
 **private** RadioGroup **rgbtnSexo**;  
 **private** Button **btniActualizar**;  
 **private** Usuario **getModel**;  
  
 *//* ***TODO: Rename parameter arguments, choose names that match*** *// the fragment initialization parameters, e.g. ARG\_ITEM\_NUMBER* **private static final** String ***ARG\_PARAM1*** = **"param1"**;  
  
 *//* ***TODO: Rename and change types of parameters* private** String **\_cuentaGET**;  
  
 **public static final int *KEY\_DIALOG\_ACTUALIZAR*** = 1;  
 **public static final** String ***KEY\_DIALOG\_FLAG\_ACTUALIZAR*** = **"dialogActualizar"**;  
  
 **public** PerfilFragment() {  
 *// Required empty public constructor* }  
  
 **public static** PerfilFragment newInstance(String param1) {  
 PerfilFragment fragment = **new** PerfilFragment();  
 Bundle args = **new** Bundle();  
 args.putString(***ARG\_PARAM1***, param1);  
 fragment.setArguments(args);  
 **return** fragment;  
 }  
  
 @Override  
 **public void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 **if** (getArguments() != **null**) {  
 **\_cuentaGET** = getArguments().getString(***ARG\_PARAM1***);  
 }  
 setHasOptionsMenu(**true**);  
 }  
  
 @Override  
 **public** View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
 *// Inflate the layout for this fragment* View root = inflater.inflate(R.layout.***fragment\_perfil***, container, **false**);  
  
 **txtiNombre** = (EditText) root.findViewById(R.id.***txtNombrePerfil***);  
 **txtiApePaterno** = (EditText) root.findViewById(R.id.***txtApePaternoPerfil***);  
 **txtiApeMaterno** = (EditText) root.findViewById(R.id.***txtApeMaternoPerfil***);  
 **rgbtnSexo** = (RadioGroup) root.findViewById(R.id.***rbtgSexo***);  
 **btniActualizar** = (Button) getActivity().findViewById(R.id.***btnActualizarPerfil***);  
 **btniActualizar**.setOnClickListener(**this**);  
 **btniActualizar**.setEnabled(**false**);  
  
 **new** cargarPerfilTask().execute(**\_cuentaGET**);  
  
 **return** root;  
 }  
  
 @Override  
 **public void** onClick(View v) {  
 **switch** (v.getId()){  
 **case** R.id.***btnActualizarPerfil***:  
 String \_nombres = **txtiNombre**.getText().toString();  
 String \_appaterno = **txtiApePaterno**.getText().toString();  
 String \_apmaterno = **txtiApeMaterno**.getText().toString();  
  
 **int** \_rbtnID = **rgbtnSexo**.getCheckedRadioButtonId();  
 View radioButton = **rgbtnSexo**.findViewById(\_rbtnID);  
 **int** index = **rgbtnSexo**.indexOfChild(radioButton);  
 RadioButton radio = (RadioButton) **rgbtnSexo**.getChildAt(index);  
 String selected = radio.getText().toString();  
 **boolean** \_sexo = selected.startsWith(**"M"**);  
  
 Usuario usuario = **new** Usuario();  
 usuario.setId(**getModel**.getId());  
 usuario.setEmail(**getModel**.getEmail());  
 usuario.setNombres(\_nombres);  
 usuario.setApePaterno(\_appaterno);  
 usuario.setApeMaterno(\_apmaterno);  
 usuario.setSexo(\_sexo);  
  
 **if**(!validar(usuario)) **return**;  
  
 UsuarioService service = UsuarioService.Factory.*getInstance*(getActivity());  
 service.updatePerfil(usuario).enqueue(**new** Callback<Boolean>() {  
 @Override  
 **public void** onResponse(Call<Boolean> call, Response<Boolean> response) {  
 **if** (response.isSuccessful()){  
 getActivity().setResult(response.body() ? Activity.***RESULT\_OK*** : Activity.***RESULT\_CANCELED***);  
 getActivity().finish();  
 }  
 }  
  
 @Override  
 **public void** onFailure(Call<Boolean> call, Throwable t) {  
 mostrarToast(**"Error: "** + t.getMessage());  
 }  
 });  
 **break**;  
 }  
 }  
  
 **private boolean** validar(Usuario model){  
 **boolean** rpt = **true**;  
 **if** (model.getNombres().isEmpty()){  
 **txtiNombre**.setError(**"Agregar campo nombre"**);  
 **txtiNombre**.requestFocus();  
 rpt = **false**;  
 }  
  
 **if** (model.getApePaterno().isEmpty()){  
 **txtiApePaterno**.setError(**"Agregar campo apellido paterno"**);  
 **txtiApePaterno**.requestFocus();  
 rpt = **false**;  
 }  
  
 **if** (model.getApeMaterno().isEmpty()){  
 **txtiApeMaterno**.setError(**"Agregar campo apellido materno"**);  
 **txtiApeMaterno**.requestFocus();  
 rpt = **false**;  
 }  
  
 **return** rpt;  
 }  
  
 **private class** cargarPerfilTask **extends** AsyncTask<String, Void, Usuario>{  
 ProgressDialog **dialog**;  
 String **mensaje** = **""**;  
 **private** cargarPerfilTask(){  
 **dialog** = **new** ProgressDialog(getActivity());  
 }  
  
 @Override  
 **protected void** onPreExecute() {  
 **dialog**.setMessage(**"Obteniendo información"**);  
 **dialog**.setIndeterminate(**true**);  
 **dialog**.setCancelable(**false**);  
 **dialog**.show();  
 }  
  
 @Override  
 **protected** Usuario doInBackground(String... params) {  
 UsuarioService service = UsuarioService.Factory.*getInstance*(getActivity());  
 Usuario model = **null**;  
 **try** {  
 Response<Usuario> response = service.findModelByMail(params[0]).execute();  
 **if** (response.isSuccessful()){  
 model = response.body();  
 }**else**{  
 **mensaje** = **"No se pudo mostrar la información."**;  
 }  
 } **catch** (IOException e) {  
 **mensaje** = e.getMessage();  
 }  
 **return** model;  
 }  
  
 @Override  
 **protected void** onPostExecute(Usuario usuario) {  
 **dialog**.dismiss();  
 **if** (usuario != **null** && **mensaje**.isEmpty()){  
 **getModel** = usuario;  
 **txtiNombre**.setText(usuario.getNombres());  
 **txtiApePaterno**.setText(usuario.getApePaterno());  
 **txtiApeMaterno**.setText(usuario.getApeMaterno().isEmpty() ? **null** : usuario.getApeMaterno());  
 **int** \_sexo = usuario.isSexo() ? 1 : 0;  
 ((RadioButton) **rgbtnSexo**.getChildAt(\_sexo)).setChecked(**true**);  
 **btniActualizar**.setEnabled(**true**);  
 }**else if**(!**mensaje**.isEmpty()){  
 mostrarToast(**mensaje**);  
 }  
 **mensaje** = **""**;  
 }  
 }  
  
 @Override  
 **public boolean** onOptionsItemSelected(MenuItem item) {  
 **switch** (item.getItemId()) {  
 **case** R.id.***action\_cambiarclave***:  
 mostrarDialogActualizarPerfilClave(***KEY\_DIALOG\_ACTUALIZAR***, **getModel**.getId());  
 **break**;  
 **case** R.id.***action\_regresar***:  
 getActivity().finish();  
 **break**;  
 }  
 **return super**.onOptionsItemSelected(item);  
 }  
  
 **private void** mostrarToast(String texto){  
 Toast.*makeText*(getActivity(), texto, Toast.***LENGTH\_SHORT***).show();  
 }  
  
 **private void** mostrarDialogActualizarPerfilClave(**int** keyDialog, String IDPERFIL){  
 android.app.FragmentTransaction fragmentTransaction = getActivity().getFragmentManager().beginTransaction();  
 android.app.Fragment prev = getActivity().getFragmentManager().findFragmentByTag(***KEY\_DIALOG\_FLAG\_ACTUALIZAR***);  
 **if** (prev != **null**) {fragmentTransaction.remove(prev);}  
 fragmentTransaction.addToBackStack(**null**);  
  
 **switch** (keyDialog){  
 **case *KEY\_DIALOG\_ACTUALIZAR***:  
 DialogFragment dialogFragment = DialogActualizarClavePerfil.*newInstance*(IDPERFIL);  
 dialogFragment.setTargetFragment(**this**, ***KEY\_DIALOG\_ACTUALIZAR***);  
 dialogFragment.show(getFragmentManager().beginTransaction(), ***KEY\_DIALOG\_FLAG\_ACTUALIZAR***);  
 **break**;  
 }  
 }  
  
 @Override  
 **public void** onActivityResult(**int** requestCode, **final int** resultCode, Intent data) {  
 **switch** (requestCode){  
 **case *KEY\_DIALOG\_ACTUALIZAR***:  
 **if**(resultCode == Activity.***RESULT\_OK***){  
 String \_ID = data.getStringExtra(DialogActualizarClavePerfil.***KEY\_ID***);  
 String \_oldClave = data.getStringExtra(DialogActualizarClavePerfil.***KEY\_OLD***);  
 String \_newClave = data.getStringExtra(DialogActualizarClavePerfil.***KEY\_NEW***);  
  
 UsuarioService service = UsuarioService.Factory.*getInstance*(getActivity());  
 service.updateClave(\_ID, \_oldClave, \_newClave).enqueue(**new** Callback<Boolean>() {  
 @Override  
 **public void** onResponse(Call<Boolean> call, Response<Boolean> response) {  
 **if** (response.isSuccessful()){  
 getActivity().setResult(response.body() ? Activity.***RESULT\_OK*** : Activity.***RESULT\_CANCELED***);  
 getActivity().finish();  
 }**else** mostrarToast(**"Hubo un problema al actualizar la clave"**);  
 }  
  
 @Override  
 **public void** onFailure(Call<Boolean> call, Throwable t) {  
 mostrarToast(**"Error: "** + t.getMessage());  
 }  
 });  
  
 }  
 **break**;  
 }  
 }  
}

* Web
  + Activity

**public class** WebActivity **extends** AppCompatActivity {  
  
 **private final static** String ***URL*** = Constantes.RESOURCES.***URL\_SISTEMA\_WEB***;  
  
 @Override  
 **protected void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_web***);  
  
 Toolbar toolbar = (Toolbar) findViewById(R.id.***toolbarViewWeb***);  
 setSupportActionBar(toolbar);  
 getSupportActionBar().setDisplayHomeAsUpEnabled(**true**);  
 toolbar.setNavigationOnClickListener(**new** View.OnClickListener() {  
 @Override  
 **public void** onClick(View v) {  
 finish();  
 }  
 });  
  
 WebFragment fragment = (WebFragment)getSupportFragmentManager().findFragmentById(R.id.***frameWeb***);  
  
 **if** (fragment == **null**){  
 fragment = WebFragment.*newInstance*(***URL***);  
 getSupportFragmentManager()  
 .beginTransaction()  
 .add(R.id.***frameWeb***, fragment)  
 .commit();  
 }  
  
 Utility.*setContext*(getApplicationContext());  
 }  
  
 @Override  
 **protected void** onResume() {  
 **super**.onResume();  
 Utility.*closeGlobalNotification*(MainActivity.***NOTIFICATION\_PERMANENT***);  
 }  
  
 @Override  
 **protected void** onPause() {  
 **super**.onPause();  
 Utility.*showGlobalNotification*(**"El sistema de facturación aún se encuentra ejecutándose."**, MainActivity.**class**, MainActivity.***NOTIFICATION\_PERMANENT***);  
 }  
}

* + Fragment

**public class** WebFragment **extends** Fragment {  
 **private** WebView **web**;  
 **private** ProgressBar **progressBar**;  
 **private int scrollbarsInsideOverlay** = View.***SCROLLBARS\_INSIDE\_OVERLAY***;  
 **private static final** String ***ARG\_PARAM1*** = **"param1"**;  
  
 *//* ***TODO: Rename and change types of parameters* private** String **mParam1**;  
  
 **public** WebFragment() {  
 *// Required empty public constructor* }  
  
 **public static** WebFragment newInstance(String \_url) {  
 WebFragment fragment = **new** WebFragment();  
 Bundle args = **new** Bundle();  
 args.putString(***ARG\_PARAM1***, \_url);  
 fragment.setArguments(args);  
 **return** fragment;  
 }  
  
 @Override  
 **public void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 **if** (getArguments() != **null**) {  
 **mParam1** = getArguments().getString(***ARG\_PARAM1***);  
 }  
 setHasOptionsMenu(**true**);  
 }  
  
 @Override  
 **public** View onCreateView(LayoutInflater inflater, ViewGroup container,  
 Bundle savedInstanceState) {  
  
 *// Inflate the layout for this fragment* View root = inflater.inflate(R.layout.***fragment\_web***, container, **false**);  
  
 **progressBar** = (ProgressBar)root.findViewById(R.id.***progressBar***);  
 **progressBar**.setVisibility(View.***GONE***);  
 **progressBar**.setMax(100);  
  
 **web** = (WebView) root.findViewById(R.id.***idWebView***);  
 *//web.clearCache(true);  
 //web.getSettings().setRenderPriority(WebSettings.RenderPriority.HIGH);* **web**.getSettings().setCacheMode(WebSettings.***LOAD\_NO\_CACHE***);  
  
 **if** (Build.VERSION.***SDK\_INT*** >= 19) **web**.setLayerType(View.***LAYER\_TYPE\_HARDWARE***, **null**);  
 **else web**.setLayerType(View.***LAYER\_TYPE\_SOFTWARE***, **null**);  
  
 **web**.getSettings().setLoadsImagesAutomatically(**true**);  
 **web**.getSettings().setJavaScriptEnabled(**true**);  
 **web**.setScrollBarStyle(**scrollbarsInsideOverlay**);  
  
 **this**.**progressBar**.setProgress(0);  
  
 Handler main = **new** Handler(getActivity().getMainLooper());  
 Runnable run = **new** Runnable() {  
 @Override  
 **public void** run() {  
 **web**.setWebViewClient(**new** browser());  
 **web**.loadUrl(**mParam1**);  
 }  
 };  
 main.post(run);  
  
 **return** root;  
 }  
  
 **public void** setValue(**int** progress){  
 **this**.**progressBar**.setProgress(progress);  
 }  
  
 **private class** browser **extends** WebViewClient {  
 @Override  
 **public boolean** shouldOverrideUrlLoading(WebView view, String url) {  
 view.loadUrl(url);  
 **return true**;  
 }  
  
 @Override  
 **public void** onPageStarted(WebView view, String url, Bitmap favicon) {  
 **progressBar**.setVisibility(View.***VISIBLE***);  
 WebFragment.**this**.setValue(100);  
 **super**.onPageStarted(view, url, favicon);  
 }  
  
 @Override  
 **public void** onPageFinished(WebView view, String url) {  
 **progressBar**.setVisibility(View.***GONE***);  
 WebFragment.**this**.setValue(0);  
 **super**.onPageFinished(view, url);  
 }  
 }  
}

Con estos datos sería suficiente para recrear el sistema en su etapa básica ya que no se está considerando agrerar las validaciones por una posible pérdida de conectividad de los web-services.