Imagen que contiene exterior, cielo, nieve, agua

Descripción generada con confianza muy alta

**ESCUELA SUPERIOR POLITÉNICA DEL LITORAL**

**PROYECTO DE APLICACIONES MÓVILES Y SERVICIOS TELEMÁTICOS**

**MANUAL TÉCNICO**

RESTAURANTE INTELIGENTE

**MG. ADRIANA COLLAGUAZO**

**GRUPO #4**

CRUZ OCHOA ALBERTO

PAZMIÑO JEFFERSON

SÁNCHEZ VALVERDE JOSUÉ

SANTILLAN PADILLA STEVEN ISAAC

**PARALELO 1**

1T – 2019

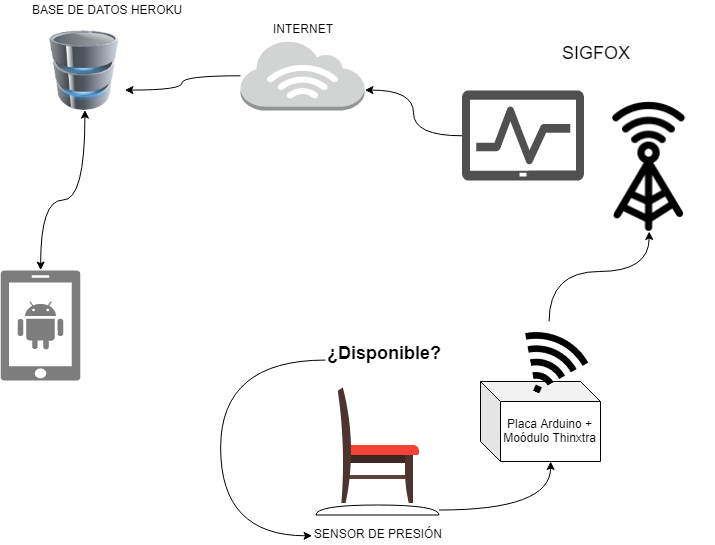
**Manual Implementación Técnica del Proyecto**

El siguiente manual técnico expone los detalles presentes en el proyecto realizado. Este proyecto tiene como fin comunicar la disponibilidad de las mesas de un restaurante mediante el uso de un sensor realizado utilizando velostat y tela conductiva, detalles que serán expuestos a lo largo de este manual.

**Introducción del sistema implementado**

El sistema mostrado a continuación describe la funcionalidad general de la solución. La implementación parte con la base de que al usuario encargado de las mesas de un restaurante, se le debe informar de la disponibilidad de las mismas mediante una aplicación en tiempo real. Para ello fue necesario el desarrollo de un sensor de presión que se ubicará debajo de una mesa, para lo cual fue necesario el establecimiento de umbrales de presión mismos que conforman parte de la estructura lógica de decisión sobre un cambio de disponibilidad en el restaurante. Este sensor se conecta a una placa de desarrollo Arduino y este a su vez a un módulo Thinxtra. Este último modulo nos permite realizar envió de notificaciones mediante la red celular hacia la plataforma de Sigfox, misma que mediante métodos preconfigurados denominados Callbacks realizan el envío de información hacia una base de datos levantada en la nube a través del Herokuapp. El usuario final, es decir el encargado de las mesas, podrá administrar en tiempo real las mesas del restaurante gracias a la información en tiempo real que puede obtener de la base de datos que está en constante cambio debido a la acción del código implementado en Arduino para generar cambios siempre y cuando exista alguna variación importante de disponibilidad.

1. **Diagrama del sistema**



1. **Recursos**

**Para realizar la implementación del proyecto fueron necesarias las siguientes herramientas:**

|  |  |
| --- | --- |
| Hardware | Software |
| Módulo Thinxtra ($109.25) | IDE Arduino |
| 2 PC | Android Studio |
| 30 jumpers ($ 2) | Base de datos en Heroku App |
| 1 placa Arduino ($20) | Herramienta colaborativa Asana |
| 3x10 cm de Velostat ($5) | Plaforma SigFox |
| Dos tiras de tela conductiva de 4x15 ($10 lamina A4) |  |
| Un rollo de cinta aislante ($0.40) |  |
| $146.65 |  |

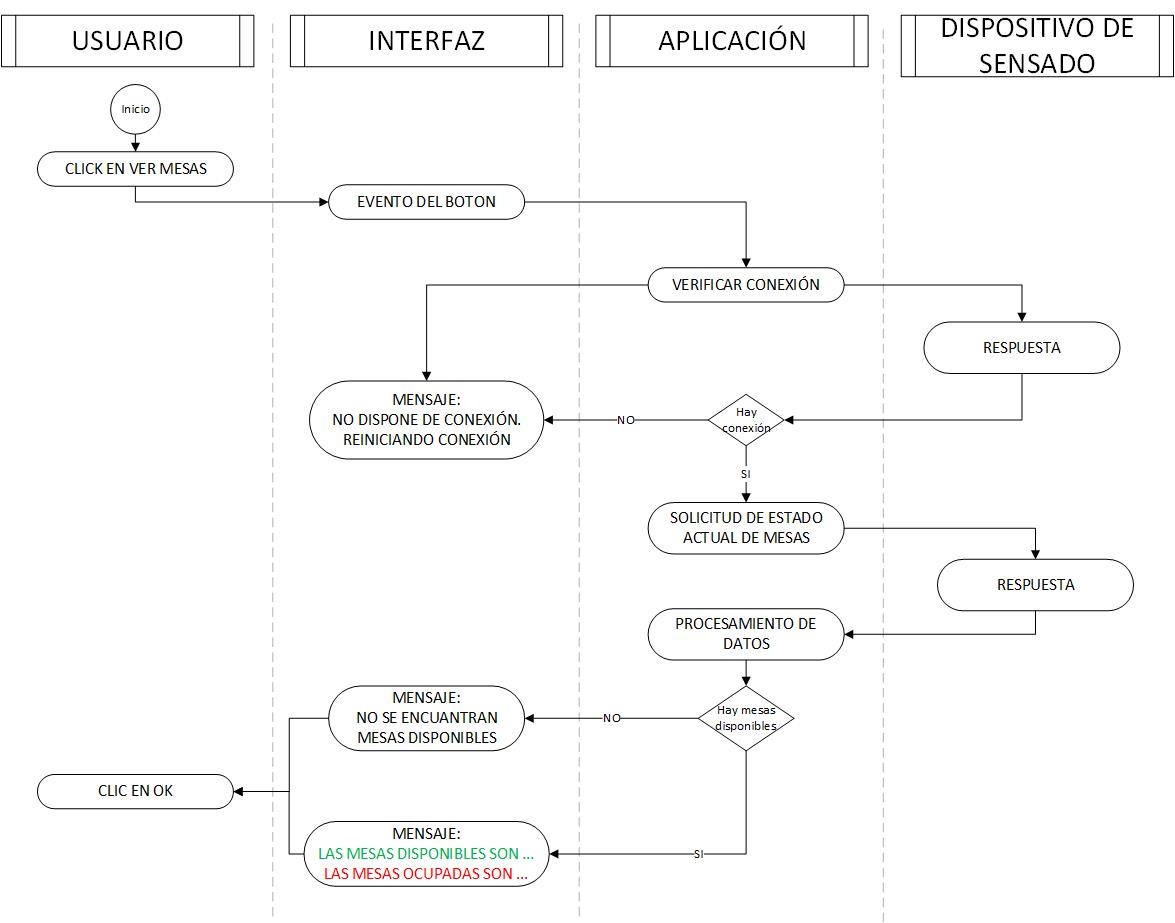
1. **Diagramas de Usuario**

**Historia de usuario 1:** Como encargado de piso requiero poder consultar la cantidad y ubicación de mesas disponibles para poder dirigir a los clientes que esperan ser atendidos a una mesa.

**Criterio de aceptación 1:** Dado que le sistema IoT se encuentra censando el suelo donde se encuentran las sillas (sensor de presión) cuando el usuario de clic en “Ver mesas" entonces aparecerán las ubicaciones de las mesas libres (con verde) y mesas ocupadas (con rojo).

**Criterio de aceptación 2:** Dado que le sistema IoT se encuentra censando el suelo (sensor de presión) y no se encuentras mesas disponibles cuando el usuario de clic en “Ver mesas" entonces aparecerán las ubicaciones de las mesas y el mensaje “No se encuentra mesas disponibles”.

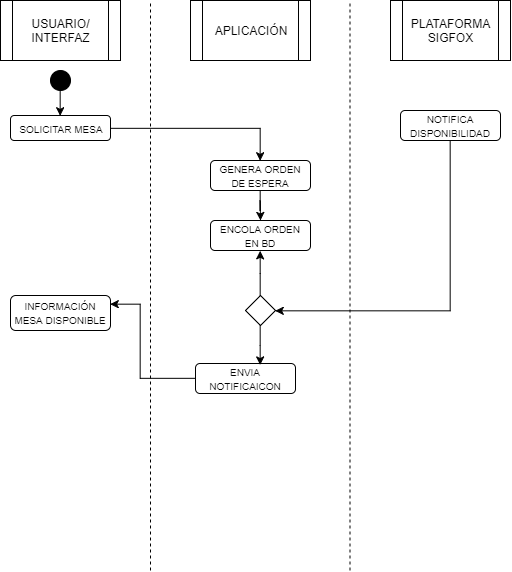
**Criterio de aceptación 3:** Dado que le sistema IoT se encuentra censando el suelo (sensor de presión) y no se dispone de conexión cuando el usuario de clic en “Ver mesas" entonces aparecerá el siguiente mensaje “No se dispone de conexión de internet” y se reinicia la comunicación Sigfox.



**Historia de usuario 2:** Como encargado de piso requiero indicar que estoy esperando una mesa (cualquiera) y recibir una notificación cuando una mesa sea liberada para para ubicar a un nuevo cliente.

**Criterio de aceptación 1:** Dado que todas las mesas se encuentran ocupadas cuando seleccione la opción “Notificar cuando una mesa esté disponible” y una mesa se libera entonces llegara una notificación indicando "Una mesa se encuentra disponible”.

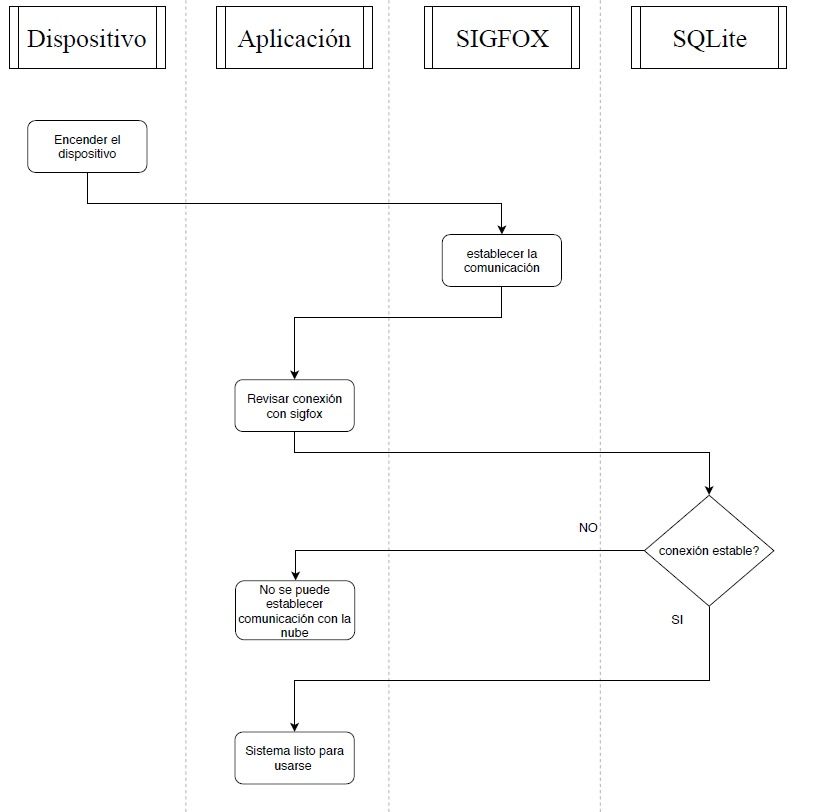
**Criterio de aceptación 2:** Dado que existen mesas disponibles cuando seleccione la opción “Notificar cuando una mesa esté disponible” entonces llegara una notificación indicando "Actualmente existen mesas disponibles”.



**Historia de usuario 3:** Como usuario requiero encender mi dispositivo IoT e iniciar comunicaciones para comenzar a utilizar el sistema.

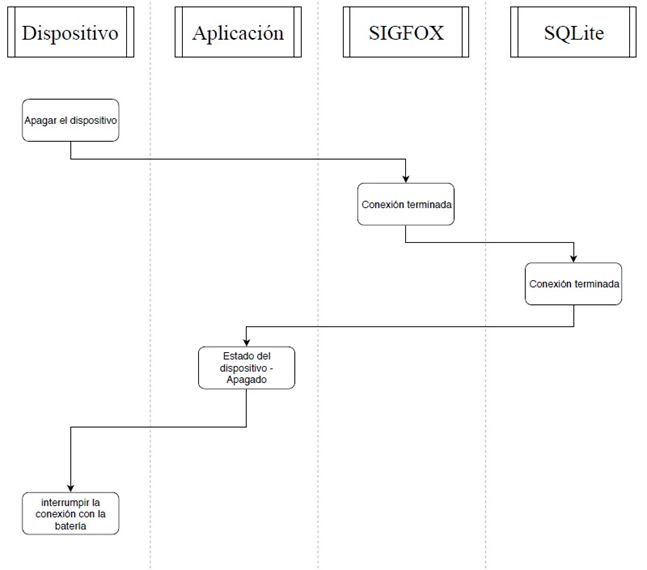
**Criterio de aceptación 1:** Dado que mi dispositivo IoT tiene una fuente de poder estable cuando se presione el botón de encendido en el dispositivo entonces se establecerá la comunicación Sigfox enviando un mensaje indicando el estado del dispositivo y aparecerá en la aplicación móvil “Sistema listo para usar”.

**Criterio de aceptación 2:** Dado que mi dispositivo IoT tiene una fuente de poder estable cuando se presione el botón de encendido en el dispositivo, pero no puede establecer comunicación Sigfox entonces no se enviará el mensaje inicial indicando el estado del dispositivo y se presentará en la aplicación móvil “No se puede establecer comunicación con la nube”.



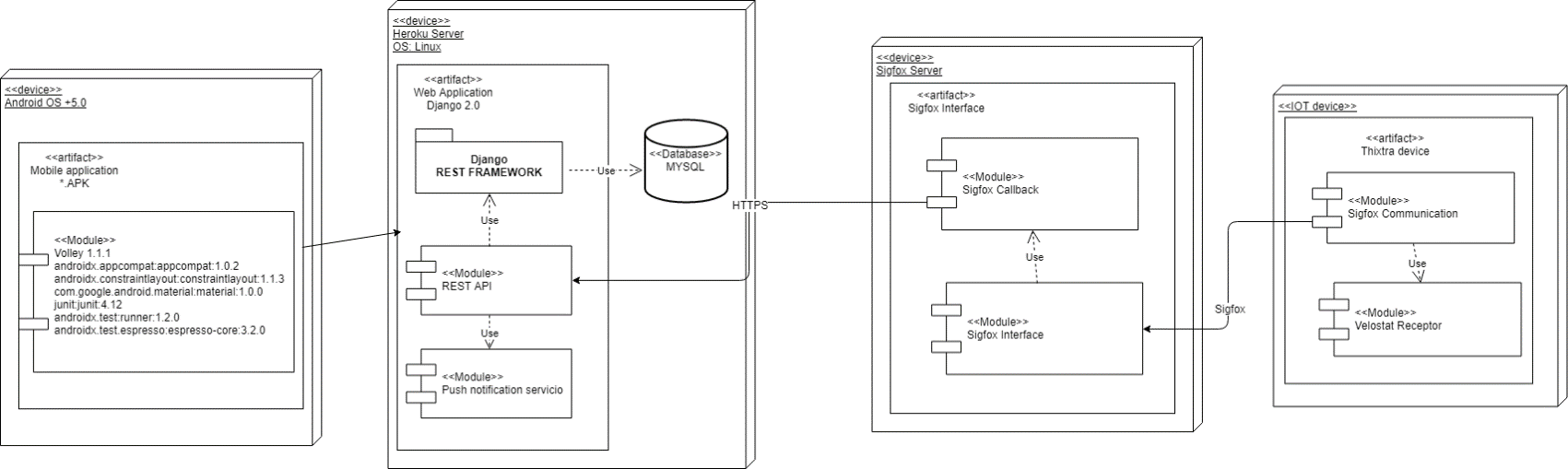
**Historia de usuario 4:** Como usuario requiero apagar mi dispositivo IoT cuando termine de usarlo para poder ahorrar energía de la batería.

**Criterio de aceptación 1:** Dado que mi dispositivo IoT tiene una fuente de poder estable y no necesito que siga encendido cuando se presione el botón de apagado en el dispositivo entonces deberá terminar la conexión Sigfox, interrumpir la conexión con la batería y mostrar en la aplicación móvil “Estado del dispositivo - Apagado”.



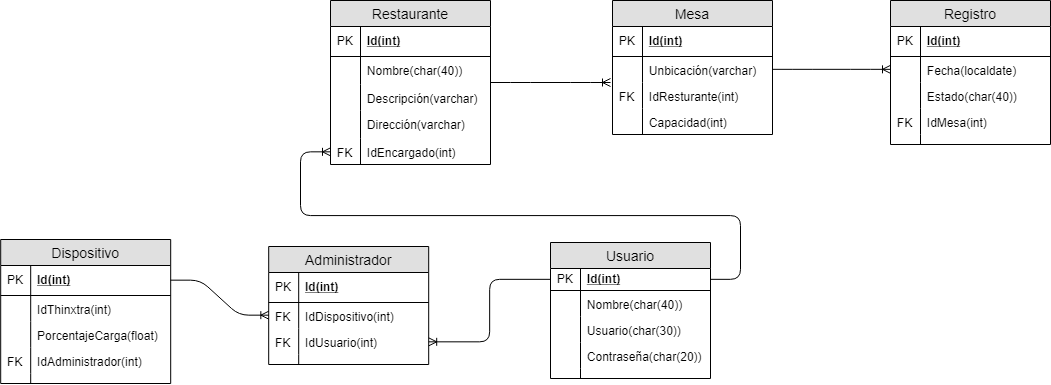
1. **Diagrama de despliegue**

El diagrama de despliegue mostrado a continuación explica acerca de la interacción entre los diferentes módulos que forman parte del proyecto. El primer módulo que se observa es el dispositivo Android, el cual contendrá la aplicación y se podrán observar los datos almacenados en la base de datos. Segundo se encuentra la base de datos la cual almacenara los datos enviados por la plataforma de Sigfox. Tercero, está la plataforma de Sigfox la cual enviara los datos capturados por el dispositivo IoT mediante los callbacks a la base de datos. Por último, se encuentra el dispositivo IoT que permite obtener los valores de los sensores y los enviara como mensajes a la plataforma de Sigfox.



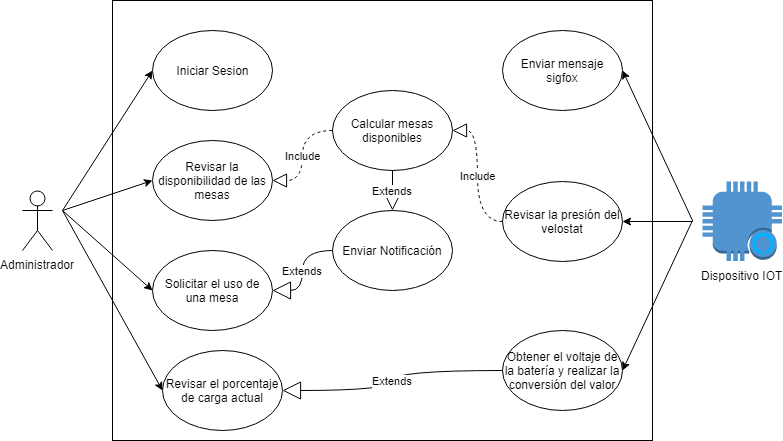
1. **Modelo de entidad-relación de la base de datos.**

El diagrama a continuación muestra las relaciones entre las diferentes entidades de las tablas de la base de datos que se utilizó para almacenar la información obtenida por los sensores implementados. Además, en el diagrama se puede observar los campos que poseen cada tabla, así como su tamaño y el tipo de variable con el que almacenaran los datos.



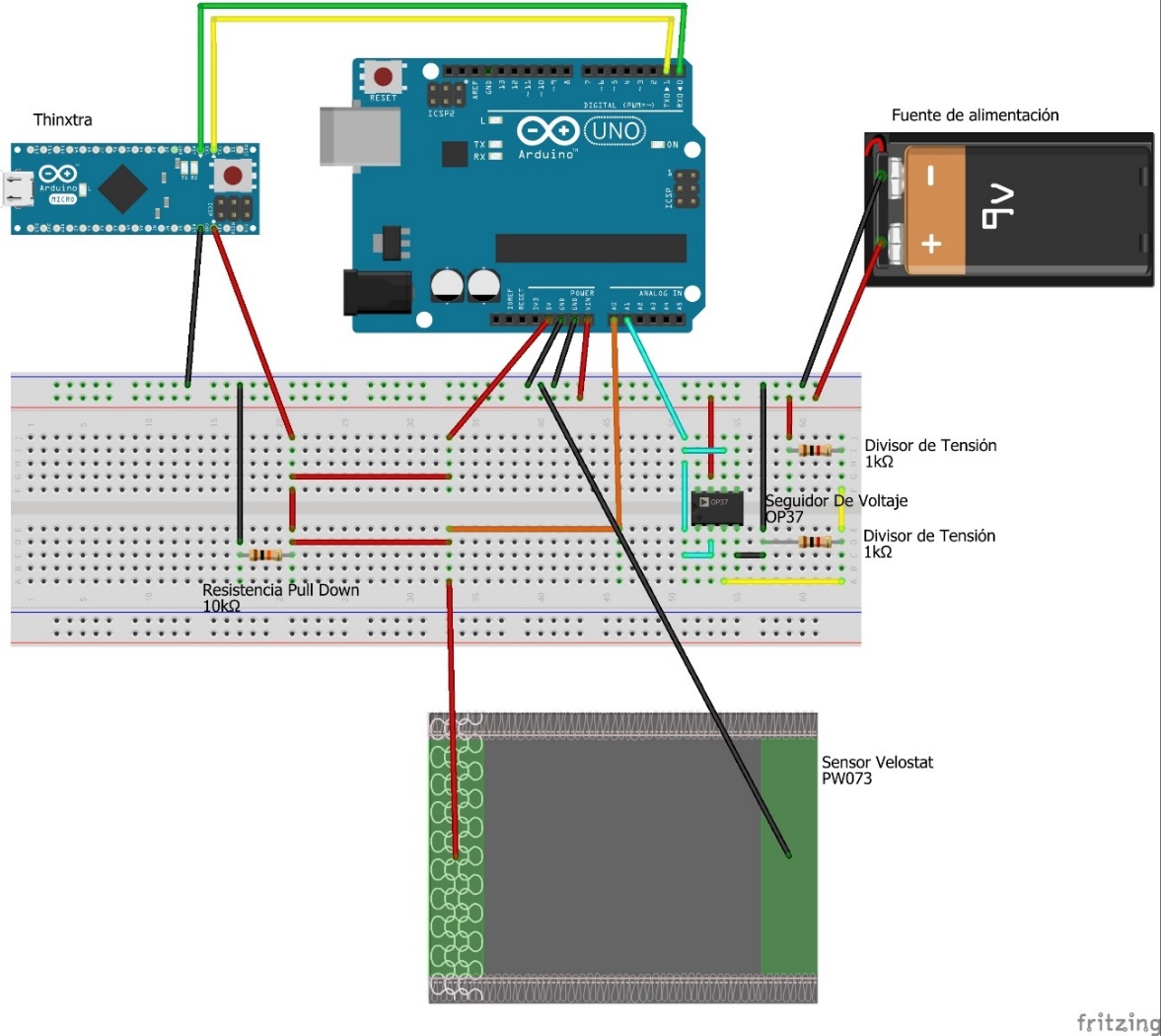
1. **Diagrama de actividades**

El presente diagrama nos describe la interacción del usuario mediante el aplicativo móvil con el dispositivo IoT, pudiendo identificar las acciones que mediante la aplicación se espera el usuario pueda realizar. Citando una de ellas, cada usuario debe poder ingresar en la aplicación, y una vez validado su ingreso podrá hacer solicitudes indirectas a la base de datos sobre la información en tiempo real del estado de las mesas en el restaurante, pero también podrá revisar la batería del sistema que permite dicha notificación de información remota.



1. **Diagrama de circuito**

El siguiente diagrama describe la implementación de hardware del proyecto. Se utilizo la placa Arduino junto con el módulo Thinxtra mismos que son energizados por una batería de 9V. Una configuración de división de tensión fue necesario introducir para poder notificar también el estado de la batería de la implementación. El sensor realizado es conectado directamente a la placa Arduino ya que a través de ella los valores de voltaje variantes debido a la presión del sensor son receptados y evaluados por el código implementado y cargado en la placa. Fue necesario el uso de una residencia en serie con el sensor para bajar la potencia consumida y evitar que genere calor el sensor por ser una capa muy delgada. Así también se tomaron consideración de seguridad colocando un diodo rectificador junto con el sensor.



1. **Bases de datos**

Para la realización de este proyecto las bases de datos utilizadas para obtener información desde el aplicativo móvil, pero también para poder realizar publicaciones de estados de las mesas son descritas a continuación.

* Base de datos para el almacenamiento de estados de las mesas:
  + <https://amstdb.herokuapp.com/db/registroEstadoMesa>
* Base de datos para el registro de mesas nueva y su capacidad:
  + <https://amstdb.herokuapp.com/db/mesa>
* Base de datos para almacenar el nivel de batería del sistema:
  + <http://amstdb.herokuapp.com/db/dispositivo/9>

Cabe mencionar que las bases más utilizadas para este proyecto fueron la de registro de estado de mesas, misma que nos permitía evaluar la disponibilidad de las mesas en tiempo real gracias a la variable “estado” de cada registro. Así mismo la base de datos que almacenada los registros de la batería fue la que nos permitió realizar la verificación del estado del sistema y de notificar al usuario si se ha llegado a un nivel de batería crítico.

1. **Explicación del código fuente Arduino desarrollado con los comentarios correspondientes**

Durante la realización del proyecto se logró efectuar de manera correcta las respectivas historias de usuario planteadas en la rúbrica mediante un programa que permite al obtener el estado de la mesa y validar todos los posibles casos de error con herramientas como una base de datos obteniendo así resultados esperados.

* 1. Explicación por archivo:
     1. **Mesas.ino**
* El programa permite obtener el estado actual de la mesa y el porcentaje restante de la batería, una vez los valores hayan sido censados se procederá a enviar dicha información a la base de datos, mediante el backend de Sigfox.

**void setup**: Función que solo se ejecuta una sola vez al inicio del programa y se inicializan las variables declaradas.

void setup() {

 // Inicializacion de variables

 pin0 = 0;

 pin1 = 1;

 tiempoAnterior = 0;

 intervalo = 600000;

 mesa = "";

 bateriaEnviar = 100;

 //Inicio de la comunicacion I2C

 Wire.begin();

 Wire.setClock(100000);

 // Velocidad del puerto serial

**Serial**.begin(9600);

 // Inicializacion de la comunicacion con sigfox

 Isigfox->initSigfox();

 Isigfox->testComms();

}

**void loop:** Función que se ejecutara de manera continua, siempre y cuando el Arduino se encuentre encendido. En esta función se lee los valores de los sensores y se envía al backend de sigfox.

void loop() {

 // Funcion de lectura de todos los sensores

 ValoresSensados();

 // Funcion que permite obtener el ultimo valor de la bateria

 BateriaMenor((int)porcentajeBateria);

 /\*

   Condicion que solo enviara a sigfox si existe un cmabio de

   estado en la mesa.

 \*/

 if (voltajeMedido > 2.90) {

   if (mesa != "DE") {

**Serial**.println("desocupada");

**Serial**.println(bateriaEnviar);

     mesa = ObtenerEstadoDeMesa(voltajeMedido);

     EnviarSigfox(mesa, bateriaEnviar);

     delay(10000);

   }

 }

 else {

   if (mesa != "OC") {

**Serial**.println("Ocupada");

**Serial**.println(bateriaEnviar);

     mesa = ObtenerEstadoDeMesa(voltajeMedido);

     EnviarSigfox(mesa, bateriaEnviar);

     delay(10000);

   }

 }

 /\*

   Envia a sigfox el valor de la bateria en porcentaje

   a sigfox cada 10 minutos.

 \*/

 tiempoActual = millis();

 if (tiempoActual - tiempoAnterior > intervalo) {

   tiempoAnterior = tiempoActual;

**Serial**.println(bateriaEnviar);

   EnviarSigfox(mesa, (int)porcentajeBateria);

   delay(10000);

 }

}

**void EnviarSigfox:** La función EnviarSigfox recive como parámetros el estado de la mesa y la batera agregándolas en el constructor como bytes y luego enviando a sigfox.

void EnviarSigfox(String mesa, int bateria) {

 byte \*stringt\_byte2 = (byte \*)&mesa;

 const uint8\_t payloadSize = 4;

 uint8\_t buf\_stri[payloadSize];

 buf\_stri[0] = mesa.charAt(0);

 buf\_stri[1] = mesa.charAt(1);

 buf\_stri[2] = bateria;

 uint8\_t \*sendData = buf\_stri;

 int len = 4;

 recvMsg \*RecvMsg;

 RecvMsg = (recvMsg \*)malloc(sizeof(recvMsg));

 Isigfox->sendPayload(sendData, len, 0, RecvMsg);

 for (int i = 0; i < RecvMsg->len; i++) {

**Serial**.print(RecvMsg->inData[i]);

 }

**Serial**.println("");

 free(RecvMsg);

}

**void ObtenerEstadoDeMesa:** La función ObtenerEstadoDeMesa obtiene como parámetro el valor del sensor en voltios y luego retorna DE su es mayor a 2.90 y OC si es menor a dicho valor.

String ObtenerEstadoDeMesa(float voltajeMedido) {

 if (voltajeMedido > 2.90) {

   return "DE";

 } else {

   return "OC";

 }

}

**void ValoresSensados:** Se obtiene todos los valores de los sensores y se le asigna a una variable los valores censados son la alfombra y la batería.

void ValoresSensados() {

 voltajeAlfombra = analogRead(pin0);

 voltajeMedido = (((float)voltajeAlfombra) \* 5.0) / 1023.0;

**Serial**.println(voltajeMedido);

 bateria = analogRead(pin1);

 porcentajeBateria = (((bateria) \* 5.0) / 1023.0) \* (100 /4.5);

}

**void BateriaMenor:** La función BateriaMenor recibe el valor de la batería actual y si es menor a la anterior actualiza la variable caso contrario no hace nada.

void BateriaMenor(int bateriaActual) {

 if (bateriaActual < bateriaEnviar) {

   bateriaEnviar = bateriaActual;

 }

}

1. **Explicación del código Java desarrollado en Android Studio con los comentarios correspondientes**

**Clase Login**

**IniciarSesion:** Se encarga de construir un objeto JSONObject con los valores de usuario y contraseña para luego hacer un POST a la url indicada y, si todo va bien, obtener el token e iniciar la actividad Menu.

private void iniciarSesion (String usuario, String password);

**GuardarCredencial:** Guarda los datos de credenciales.

private void guardarCredencial();

**CargarCredencial:** Carga los datos de credenciales guardados.

private boolean cargarCedencial();

**ValidarCredencial:** Hace una verificación de que las credenciales esten correctas.

private void validarPaso();

**Clase Estado**

**CalcularPorcentajeMesas:** Método que recibe un HashMap de los datos obtenidos de la base de datos y hace un recorrido de este para contar las mesas ocupadas y conociendo el total de claves del HashMap, se haya el porcentaje de mesas ocupadas y disponibles para guardarlos en el arreglo porcentajes.

private void calcularPorcentajeMesas(HashMap<String,String[]> registros, float[] porcentajes);

**CrearPastel:** Establece características del gráfico de pastel e introduce los datos que se encuentran en un PieData.

private void crearPastel();

**LlenarScrolling:** Método que permite llenar la pantalla con los datos de los estados de las mesas obtenidos del HashMap.

private void llenarScrolling(HashMap<String,String[]> registros);

**Refresh:** Método que ejecuta un hilo para refrescar la gráfica de pastel y los datos de los estados de las mesas.

public void refresh();

**Clase Disponibilidad**

**LlenarScrolling:** Método que permite llenar un RecyclerView para presentar la disponibilidad de las mesas extraídas de un HashMap.

void llenarScrolling(HashMap<String, String[]> registros);

**Refresh:** Método que ejecuta un hilo para refrescar los datos de disponibilidad de mesas llamando al método LlenarScrolling.

public void refresh();

**Clase Menú**

**ObtenerEstadoMesas:** Método que se encarga de Crear un JSONArray y que con dos url’s donde se alojan las dos bases de datos hace un GET para obtener todos los datos.

private void obtenerEstadoMesa(final String urlMesa, final String urlEstado);

**Next\_disponibilidad:** refresca los datos del HashMap en u hilo y accede a la actividad disponibilidad

public void next\_disponibilidad(View view);

**Next\_estado:** Accede a la actividad donde se muestra el estado del restaurante.

public void next\_estado(View view);

**Next\_disponibilidad:** Borra las credenciales y lleva la app a la actividad de Login.

public void next\_salir(View view);

**Next\_batería:** Lleva a la actividad donde se muestra el estado de la batería del dispositivo implementado.

public void next\_bateria(View view);

**BorrarCredenciales:** Borra las credenciales de acceso a la aplicación.

public void borrarCredencial();

**RefreshB:** Ejecuta un hilo que actualiza constantemente el estado de la batería del dispositivo implementado.

public void refreshB();

**BorrarCredenciales:** Método que se encarga de Crear un JSONObject y que con la url donde se aloja la base de datos hace un GET.

private void obtenerB (String urlB);

**ValidarBatería:** Valida si la notificación va a ser o no enviada de acuerdo al estado de la batería

public void validarBateria();

**EnviarNotificación:** Envía una notificación por medio de un POST alertando que el nivel de la batería está muy bajo

private void enviarNotificacion();

**XML LogoInicial:** contiene el formato para presentar el logo inicial de la app al momento de acceder a ella

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout 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:background="@drawable/gradient"  
 tools:context=".LogoInicial">  
  
 <LinearLayout  
 android:id="@+id/root\_layoutLogin"  
 android:layout\_width="0dp"  
 android:layout\_height="0dp"  
 android:background="@drawable/gradient"  
 android:gravity="center"  
 android:orientation="vertical"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent">  
  
 <ImageView  
 android:layout\_width="263dp"  
 android:layout\_height="197dp"  
 android:src="@drawable/spashimagen"  
 android:contentDescription="@string/SplashLogo" />  
 </LinearLayout>  
</androidx.constraintlayout.widget.ConstraintLayout>

**XML Login:** Contiene un LinearLayout donde se ubican los componentes como EditTexts y Buttons para dar forma a la interfaz de inicio de sesión.

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout 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"  
  
 tools:context=".Login">  
  
 <LinearLayout  
 android:id="@+id/root\_layoutLogin"  
 android:layout\_width="0dp"  
 android:layout\_height="0dp"  
 android:background="@drawable/gradient"  
 android:gravity="center"  
 android:orientation="vertical"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent">  
  
  
 <ScrollView  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:ignore="UselessParent">  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center"  
 android:gravity="center"  
 android:orientation="vertical">  
  
 <TextView  
 android:id="@+id/textViewTituloEstadoMesas"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:fontFamily="casual"  
 android:gravity="center"  
 android:text="@string/TituloLoginSmarTable"  
 android:textColor="#090808"  
 android:textSize="60sp"  
 android:textStyle="bold" />  
  
 <ImageView  
 android:layout\_width="323dp"  
 android:layout\_height="177dp"  
 android:adjustViewBounds="false"  
 android:contentDescription="@string/ImagenTableLogin"  
 app:srcCompat="@drawable/mesa" />  
  
 <EditText  
 android:id="@+id/txtUsuario"  
 android:layout\_width="223dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center"  
 android:background="#40009688"  
 android:drawableStart="@drawable/ic\_action\_user"  
 android:fontFamily="aldrich"  
 android:gravity="center"  
 android:hint="@string/txtUsuario"  
 android:inputType="text"  
 android:textColor="#FAFAFA"  
 android:textColorHint="#FAFAFA"  
 android:importantForAutofill="no"  
 tools:targetApi="o" />  
  
 <EditText  
 android:id="@+id/txtPassword"  
 android:layout\_width="223dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center"  
 android:layout\_marginTop="15dip"  
 android:layout\_marginBottom="30dip"  
 android:background="#40009688"  
 android:drawableStart="@drawable/ic\_action\_pass"  
 android:fontFamily="aldrich"  
 android:gravity="center"  
 android:hint="@string/txtClave"  
 android:inputType="textPassword"  
 android:textColor="#FCFBFB"  
 android:textColorHint="#FFFFFF"  
 android:importantForAutofill="no"  
 tools:targetApi="o" />  
  
 <Button  
 android:layout\_width="149dp"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center\_horizontal"  
 android:background="@drawable/custom\_button"  
 android:backgroundTint="#FFFFFF"  
 android:fontFamily="aldrich"  
 android:onClick="irMenuPrincipal"  
 android:paddingLeft="15dip"  
 android:paddingRight="15dip"  
 android:text="@string/BotonIngresar"  
 android:textColor="#030303" />  
 </LinearLayout>  
 </ScrollView>  
  
 </LinearLayout>  
  
</androidx.constraintlayout.widget.ConstraintLayout>

**XML Estado:** Contiene un LinearLayout donde primero se ubica el gráfico de pastel y luego el listado de mesas que será de manera programable en código java.

?xml version="1.0" encoding="utf-8"?>  
<!--suppress ALL -->  
<androidx.constraintlayout.widget.ConstraintLayout 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:layout\_gravity="center"  
 android:background="#FFEECF"  
 android:orientation="vertical"  
 tools:context=".Estado"  
 tools:ignore="Overdraw">  
  
 <TextView  
 android:id="@+id/textEti"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginStart="12dp"  
 android:layout\_marginTop="7dp"  
 android:layout\_marginEnd="8dp"  
 android:layout\_marginBottom="11dp"  
 android:text="ESTADO DEL RESTAURANTE"  
 android:textColor="#000000"  
 android:textSize="24sp"  
 app:layout\_constraintBottom\_toTopOf="@+id/Linear123"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
 <LinearLayout  
 android:id="@+id/Linear123"  
 android:layout\_width="0dp"  
 android:layout\_height="0dp"  
 android:gravity="center"  
 android:orientation="vertical"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/textEti">  
  
 <com.github.mikephil.charting.charts.PieChart  
 android:id="@+id/pieChartGraficaEstadoMesas"  
 android:layout\_width="200dp"  
 android:layout\_height="200dp"  
 android:layout\_margin="10dp" />  
  
 <ScrollView  
 android:id="@+id/scroll"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:layout\_gravity="center"  
 android:foregroundGravity="center">  
  
 <androidx.recyclerview.widget.RecyclerView  
 android:id="@+id/reciclador2"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center"  
 android:foregroundGravity="center"  
 android:padding="3dp"  
 android:paddingLeft="60dp"  
 android:paddingBottom="0dp"  
 android:scrollbars="vertical" />  
 </ScrollView>  
 </LinearLayout>  
</androidx.constraintlayout.widget.ConstraintLayout>

**XML Disponibilidad:** Contiene un ScrollView que contiene a un RecyclerView con el fin de dar una interfaz con datos bien organizados los que serán creados dinámicamente en código java.

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout 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:background="#FFD780"  
 android:orientation="vertical"  
 tools:context=".Disponibilidad"  
 tools:ignore="Overdraw">  
  
 <TextView  
 android:id="@+id/textDtitu"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="9dp"  
 android:text="DISPONIBILIDAD DE MESAS"  
 android:textColor="#000000"  
 android:textSize="24sp"  
 app:layout\_constraintBottom\_toTopOf="@+id/linearLayout"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintHorizontal\_bias="0.495"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toTopOf="parent" />  
  
 <LinearLayout  
 android:id="@+id/linearLayout"  
 android:layout\_width="0dp"  
 android:layout\_height="0dp"  
 android:orientation="horizontal"  
 app:layout\_constraintBottom\_toBottomOf="parent"  
 app:layout\_constraintEnd\_toEndOf="parent"  
 app:layout\_constraintStart\_toStartOf="parent"  
 app:layout\_constraintTop\_toBottomOf="@+id/textDtitu">  
  
 <androidx.recyclerview.widget.RecyclerView  
 android:id="@+id/reciclador"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center"  
 android:padding="3dp"  
 android:paddingBottom="0dp"  
 android:scrollbars="vertical" />  
  
 </LinearLayout>  
  
</androidx.constraintlayout.widget.ConstraintLayout>

**XML Menú:** Contiene un LinearLayout donde se ubican una serie de componentes como Buttons los cuales permitirán el acceso a las diferentes actividades.

<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout 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:background="#F1c981"  
 tools:context=".Menu">  
  
 <ScrollView  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent">  
  
 <LinearLayout  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:orientation="vertical">  
  
 <TextView  
 android:id="@+id/textLog"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content" />  
  
 <TextView  
 android:id="@+id/textMenu"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="30dp"  
 android:fontFamily="@font/aldrich"  
 android:gravity="center"  
 android:text="MENÚ"  
 android:textColor="#000000"  
 android:textSize="50sp"  
 tools:ignore="HardcodedText" />  
  
 <LinearLayout  
 android:id="@+id/lyResta"  
 android:layout\_width="match\_parent"  
 android:layout\_height="222dp"  
 android:layout\_margin="10dp"  
 android:gravity="center"  
 android:orientation="horizontal">  
  
 <LinearLayout  
 android:layout\_width="213dp"  
 android:layout\_height="match\_parent"  
 android:orientation="vertical">  
  
 <TextView  
 android:id="@+id/textView4"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="10dp"  
 android:text="Restaurante"  
 android:textColor="#050505"  
 android:textSize="30sp"  
 tools:ignore="HardcodedText,RtlHardcoded" />  
  
 <TextView  
 android:id="@+id/textView5"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginTop="5dp"  
 android:text="Lucia´s"  
 android:textColor="#2B2121"  
 android:textSize="24sp"  
 tools:ignore="HardcodedText,RtlHardcoded" />  
  
 <TextView  
 android:id="@+id/textView6"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="10dp"  
 android:layout\_marginTop="20dp"  
 android:text="Encargado"  
 android:textColor="#050505"  
 android:textSize="30sp"  
 tools:ignore="HardcodedText,RtlHardcoded" />  
  
 <TextView  
 android:id="@+id/txtEncargado"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginLeft="10dp"  
 android:text="Marcos Pincay"  
 android:textColor="#2B2121"  
 android:textSize="24sp"  
 tools:ignore="HardcodedText,RtlHardcoded" />  
 </LinearLayout>

**XML EstadoBatería:** Únicamente contenida de una ImageView y Textviews los que presentan el estado de la batería tanto gráficamente como en palabras.

<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout 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:gravity="center"  
 android:orientation="vertical"  
 tools:context=".EstadoBateria">  
  
 <TextView  
 android:id="@+id/textEstadoActualBateria"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginTop="151dp"  
 android:text="@string/TituloEstadoActualBateria"  
 android:textSize="36sp" />  
  
 <ImageView  
 android:id="@+id/imageEstadoActualBateria"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:contentDescription="@string/imagenestadoactual"  
 app:srcCompat="@drawable/b100" />  
  
 <TextView  
 android:id="@+id/textViewPorcentajeBateria"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_marginBottom="186dp"  
 android:text="@string/PorcentajeVisualBateria"  
 android:textSize="30sp" />  
</LinearLayout>

**XML Mesa:** Formato del CardView que hará posible la organización de los datos en la clase disponibilidad.

<androidx.cardview.widget.CardView xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:card\_view="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 card\_view:cardCornerRadius="4dp"  
 card\_view:cardElevation="4dp"  
 card\_view:cardUseCompatPadding="true">  
  
  
 <RelativeLayout  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:layout\_gravity="center">  
  
 <ImageView  
 android:id="@+id/imagen"  
 android:layout\_width="100dp"  
 android:layout\_height="70dp"  
 android:scaleType="centerCrop"  
 android:contentDescription="TODO"  
 tools:ignore="ContentDescription,HardcodedText" />  
  
 <TextView  
 android:gravity="center"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:textAppearance="?android:attr/textAppearanceLarge"  
 android:text="Large Text"  
 android:id="@+id/mesaID"  
 android:layout\_toRightOf="@+id/imagen"  
 android:layout\_alignParentTop="true"  
 android:layout\_marginLeft="10dp"  
 tools:ignore="RtlHardcoded" />  
  
 <TextView  
 android:gravity="center"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:textAppearance="?android:attr/textAppearanceSmall"  
 android:text="Small Text"  
 android:id="@+id/capacidad"  
 android:layout\_below="@+id/mesaID"  
 android:layout\_alignLeft="@+id/mesaID"  
 tools:ignore="RtlHardcoded" />  
  
 <TextView  
 android:gravity="center"  
 android:layout\_width="wrap\_content"  
 android:layout\_height="wrap\_content"  
 android:textAppearance="?android:attr/textAppearanceSmall"  
 android:text="Large Text"  
 android:id="@+id/estado"  
 android:layout\_below="@+id/capacidad"  
 android:layout\_alignLeft="@+id/capacidad"  
 tools:ignore="RtlHardcoded" />  
  
 </RelativeLayout>  
</androidx.cardview.widget.CardView>

**XML Mesados:** Formato del RelativeLayout que se usa en la clase estado para presentar los datos del estado del restaurante.

**<RelativeLayout**  
  
 **xmlns:android="http://schemas.android.com/apk/res/android"**  
 **xmlns:tools="http://schemas.android.com/tools"**  
 **android:layout\_width="wrap\_content"**  
 **android:layout\_height="wrap\_content"**  
 **android:gravity="center">**  
  
  
 **<LinearLayout**  
 **android:layout\_width="wrap\_content"**  
 **android:layout\_height="wrap\_content"**  
 **android:orientation="horizontal"**  
 **android:gravity="center"**  
 **tools:ignore="UselessParent">**  
  
 **<TextView**  
 **android:id="@+id/mesaID"**  
 **android:layout\_width="wrap\_content"**  
 **android:layout\_height="wrap\_content"**  
 **android:gravity="center" />**  
  
 **<TextView**  
 **android:id="@+id/capacidad"**  
 **android:layout\_width="wrap\_content"**  
 **android:layout\_height="wrap\_content"**  
 **android:gravity="center" />**  
  
 **<TextView**  
 **android:id="@+id/estado"**  
 **android:layout\_width="wrap\_content"**  
 **android:layout\_height="wrap\_content"**  
 **android:gravity="center" />**  
 **</LinearLayout>**  
**</RelativeLayout>**