

Restaurante Inteligente Usando Velostat

Manual Técnico

Profesor:

MG. ADRIANA COLLAGUAZO

Desarrollado por:

CESAR ANTONIO COLOMA GUZMAN

ERICK ANDREE RUIZ CEDEÑO

DANNY FABRICIO UCHO MOROCHO

Versión: v3.0

Fecha: 3/9/2020

[Versión 3.0]

|  |
| --- |
| El desarrollo de este proyecto se llevó a cabo en la asignatura de Aplicaciones Móviles y Servicios Telemáticos de la Escuela Superior Politécnica del Litoral en el Término 1S 2020. |

Contenido

[1 PROYECTO 3](#_Toc50078564)

[1.1 Descripción 3](#_Toc50078565)

[1.2 Problema por resolver 3](#_Toc50078566)

[1.3 Funcionalidades Historias de Usuario 3](#_Toc50078567)

[2 SISTEMA 6](#_Toc50078568)

[2.1 Modelo Lógico 6](#_Toc50078569)

[2.2 Diagrama de Despliegue 6](#_Toc50078570)

[2.3 Modelo de base de Datos 7](#_Toc50078571)

[2.4 Diagrama de Casos de Uso 7](#_Toc50078572)

[2.5 Diagrama del Circuito 8](#_Toc50078573)

[2.6 Implementación del prototipo 8](#_Toc50078574)

[3 DESCRIPCIÓN DEL SISTEMA 9](#_Toc50078575)

[3.1 Implementación Arduino 9](#_Toc50078576)

[3.1.1 Lectura de Valores 9](#_Toc50078577)

[3.1.2 Evaluar valores Censados 9](#_Toc50078578)

[3.1.3 Envío de valores a red sigfox 9](#_Toc50078579)

[3.2 Implementación Aplicación móvil Android 10](#_Toc50078580)

# PROYECTO

## Descripción

En el Restaurante Poly Descanso se presenta la dificultad de monitorear las mesas con las sillas que se encuentran disponibles, por lo cual los clientes forman largas filas fuera del local para ser atendidos.

## Problema por resolver

Grandes restaurantes, cuentan con n sillas y mesas a disposición que contantemente se vacían y se vuelven a llenar. También suelen mover las mesas y sillas a conveniencia de grandes grupos. Algunas empresas logran solucionar parcialmente este problema usando radios para

comunicación entre meseros que logran identificar mesas vacías. Aun así, esta solución es parcial y degrada la calidad del servicio del restaurante.

## Funcionalidades Historias de Usuario

**HU1**

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

Criterios de aceptación

• 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).

• 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”.

**HU2**

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.

Criterios de aceptación

• 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”.

• Dado que existen mesas disponibles cuando seleccione la opción “Notificar cuando una mesa esté disponible” entonces llegara una notificación indicando "Actualmente existen mesas disponibles”.

**HU3**

Como usuario requiero encender mi dispositivo IoT e iniciar comunicaciones para comenzar a utilizar el sistema clasificador.

Criterios de aceptación

• Dado que mi dispositivo tiene una fuente estable cuando se presione el botón de encendido entonces se iniciará comunicación con el sensor y el módulo de comunicación dejando listo todos los parámetros para enviar automáticamente la información censada.

• Dado que mi dispositivo tiene una fuente estable cuando se presione el botón de encendido, pero no puede establecer comunicación con el sensor entonces se mostrará mediante una luz led una alerta de error.

**HU4**

Como usuario requiero apagar mi dispositivo cuando termine de usarlo para poder ahorrar energía de la batería.

Criterios de aceptación

• Dado que mi dispositivo tiene una fuente estable y no necesito que siga encendido cuando se presione el botón de apagado entonces deberá terminar la conexión Sigfox, interrumpir la conexión con la batería.

**HU5**

Como usuario requiero iniciar sesión para poder utilizar la aplicación.

Criterios de aceptación

• Dado que he descargado la aplicación y tengo un usuario registrado cuando inicie la aplicación e ingrese su usuario y contraseña entonces cargara su información de perfil e ingresara a la aplicación.

• Dado que he descargado la aplicación y tengo un usuario registrado cuando inicie la aplicación, pero no ingresa ningún dato y da clic en iniciar sesión entonces aparecerá el mensaje “Por favor ingrese su usuario y contraseña”.

• Dado que he descargado la aplicación y tengo un usuario registrado cuando ingrese un usuario o contraseña inválidos entonces aparecerá el mensaje “Por favor revise que sus credenciales sean correctas”.

• Dado que he descargado la aplicación y tengo un usuario registrado cuando ingrese un usuario o contraseña, pero no dispone de conexión a internet entonces aparecerá el mensaje “No hay conexión a internet”.

• Dado que he cerrado la aplicación y he iniciado previamente mi sesión cuando vuelva a ingresar entonces no me volverá a pedir que ingrese mi usuario y contraseña.

**Requerimientos no funcionales:**

• El sistema debe generar la menor cantidad de ruido posible, por eso probar diferentes configuraciones con el velostat.

• El sistema debe ser de bajo costo, por eso se requiere un informe presupuestario. Observaciones:

• La interfaz gráfica debe ser fácil de usar y agradable a la vista. Debe contar con un icono para la aplicación.

• El iButton es un dispositivo que solo envía un pulso al momento de ser activado, para el estudiante funciona como una caja negra, donde el hardware no puede ser modificado.

• Realizar el circuito con dos mesas e identificar cuál de las dos está siendo ocupada y cual no.

• La interfaz gráfica debe ser fácil de usar y agradable a la vista. Debe contar con un icono para la aplicación.

**Hardware necesario:**

• Arduino uno

• iButton

• Velostat (sensor de peso)

• Tela conductora

**Software necesario:**

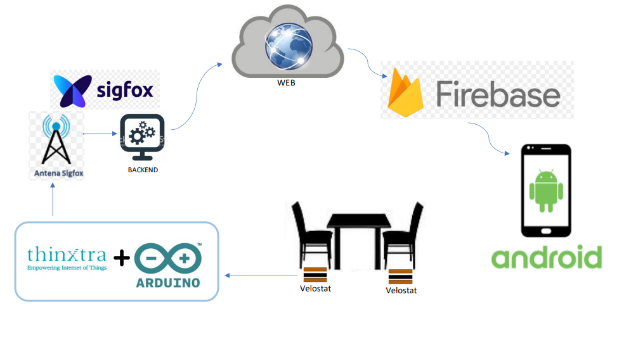
• Android Studio para el desarrollo de una aplicación móvil.

• FireBase database / servicio de autenticación / servicio de notificación.

# SISTEMA

## Modelo Lógico

La aplicación OpenSeat se basa en un Modelo de varios componentes, en el siguiente gráfico se muestra la estructura general del proyecto y como interactúa su dispositivo móvil con los componentes adicionales que permiten su uso.



## Diagrama de Despliegue

La aplicación OpenSeat está diseñada para proveer un servicio en tiempo real. Captura datos de un componente IoT mediante un sensor de velostat que cambia su resistividad mediante la presión y estos datos son enviados a un backend de la red sigfox y mediante callbacks son enviados a un backend de Firebase donde son almacenados y se gestionan para proveer información a los usuarios de la aplicación móvil.

Captura de pantalla de un celular con letras

Descripción generada automáticamente

## Modelo de base de Datos

La base de datos modelada es NOSQL (modelo no relacional), este modelo resulta adecuado por el tipo de información que vamos a almacenar, así como la interacción mediante notificaciones con usuarios de la aplicación.

Imagen que contiene texto, mapa

Descripción generada automáticamente

## Diagrama de Casos de Uso

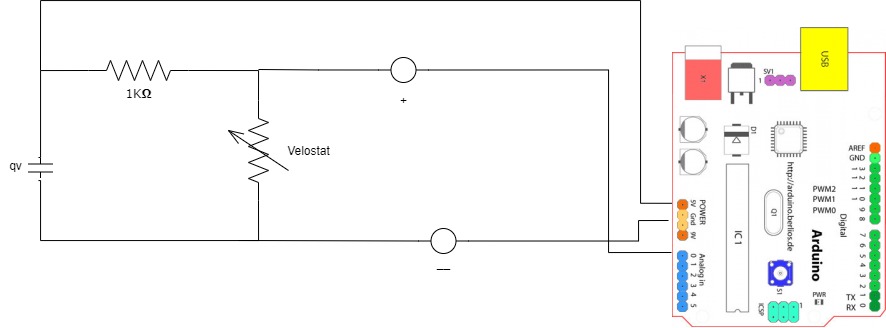
EL Diagrama a continuación muestra las diferentes actividades que puede realizar el sistema con todos los actores que interactúan con este, algunas de las actividades que se muestran no son implementadas por el alcance que tiene la aplicación.

Texto en fondo blanco

Descripción generada automáticamente

## Diagrama del Circuito

El diagrama del circuito mostrado a continuación refleja el mínimo de recurso utilizados para conectar el sensor velostat con el controlador Arduino Uno.



## Implementación del prototipo

La propuesta de prototipo se muestra a continuación, es necesario especificar que según nuestra propuesta este prototipo debe colocarse debajo del tapizado de la silla y la esponja del asiento, de esta manera el prototipo puede protegerse de posibles daños y censar correctamente el cambio de presión en las sillas.



# DESCRIPCIÓN DEL SISTEMA

## Implementación Arduino

Se detallan 3 procesos necesarios para el funcionamiento del dispositivo, la lectura constante de cambios de presión, la evaluación de los valores censados y el envió de datos al backend de sigfox y

### Lectura de Valores

|  |
| --- |
| Voidloop() { |
|  | // Lectura de sensores |
|  | ValoresSensados(); |
|  | if (voltajeMedido > 1.30) { |
|  | if (mesa != "DE") { |
|  | Serial.println("desocupada"); |
|  | mesa = ObtenerEstadoDeMesa(voltajeMedido); |
|  | EnviarSigfox(mesa); |
|  | delay(10000); |
|  | } |
|  | } |
|  | else { |
|  | if (mesa != "OC") { |
|  | Serial.println("Ocupada"); |
|  | mesa = ObtenerEstadoDeMesa(voltajeMedido); |
|  | EnviarSigfox(mesa); |
|  | delay(10000); |
|  | } |
|  | } |

### Evaluar valores Censados

|  |
| --- |
| voidValoresSensados() { |
|  | voltajeAlfombra = analogRead(pin0); |
|  | voltajeMedido = (((float)voltajeAlfombra) \* 5.0) / 1023.0; |
|  | Serial.println(voltajeMedido); |
|  | } |

### Envío de valores a red sigfox

|  |
| --- |
| void EnviarSigfox(String mesa) { |
|  | byte \*stringt\_byte2 = (byte \*)&mesa; |
|  | const uint8\_t payloadSize = 3; |
|  | uint8\_t buf\_stri[payloadSize]; |
|  | buf\_stri[0] = mesa.charAt(0); |
|  | buf\_stri[1] = mesa.charAt(1); |
|  | uint8\_t \*sendData = buf\_stri; |
|  | int len = 3; |
|  | 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); |
|  | } |

## Implementación Aplicación móvil Android

Las clases creadas para el proyecto se detallan a continuación:

Buzón: activity que permite almacenar las sugerencias en el sistema.

Database: se gestiona la conexión de la base de datos de Firebase

EstadoMesa: muestra la estadística de las mesas ocupadas y disponibles

Login: permite a los usuarios registrarse en el sistema

Main: Carga la interfaz de inicio de la aplicación y permite iniciar sesión

MesasDisponibles: muestra las mesas disponibles en el restaurante

Registro: permite a los usuarios registrarse en la aplicación

Userdata: controla el registro del usuario

Usuario: interfaz de usuario Cliente.