

**“MICROROBOT INTELIGENTE CON TECNOLOGIA BASADA EN .NET Y DISPOSITIVOS MOVILES”**

**INFORME TÉCNICO DE TITULACIÓN**

**PARA OBTENER EL GRADO DE:**

**INGENIERO EN SISTEMAS COMPUTACIONALES**

**PRESENTADO POR:**

**ROBERTO ARMANDO BALLEZA MANCILLAS**

**ÁNGEL ALBERTO MEJÍA SÁNCHEZ**

**ALEJANDRO RUIZ VARELA**

**LAGOS DE MORENO JALISCO, MARZO DE 2014**

Instituto Tecnológico Superior de Lagos de Moreno.

Libramiento Tecnológico # 5000.

Colonia Portugalejo de los Romanes.

Tel. y Fax. (52)-01-474-72-52-100 y 101

Lagos de Moreno Jalisco, México.

Superior Technology Institute of Lagos de Moreno.

5000 Libramiento Tecnológico

Portugalejo de los Romanes

Phone & Fax (52)-01-474-72-52-100 y 101

Lagos de Moreno Jalisco, México.

“MICROROBOT INTELIGENTE CON TECNOLOGIA BASADA EN .NET Y DISPOSITIVOS MOVILES”

PRESENTADO POR:

ROBERTO ARMANDO BALLEZA MANCILLAS

ÁNGEL ALBERTO MEJÍA SÁNCHEZ

ALEJANDRO RUIZ VARELA

RESUMEN: Hoy en día, la tecnología ha tenido un gran avance, debido a esto se ha facilitado la integración de tecnologías de la información (T.I.) con el uso de microcontroladores y sensores, esto ha generado grandes beneficios ya que se pueden realizar una serie tareas consideradas de riesgo para las personas. El presente informe ejemplifica un prototipo que involucra innovación y aplicación las nuevas tecnologías para el desarrollo de un micro-robot diseñado para monitorear lugares de difícil acceso y calidad del ambiente.

Palabras Claves: Arduino, Bluetooth, Sensores, Microcontrolador, Dispositivos Móviles.

ABSTRACT: Nowadays, technology has had a breakthrough, this has facilitated the integration of information technology (TI) with the use of microcontrollers and sensors, this has generated huge profits because they can perform a number tasks a hazard to people. This report illustrates a prototype that involves innovation and application of new technologies to development of a micro-robot designed to monitor places of difficult access and environmental quality.

KEYWORDS: Arduino, Bluetooth, Sensors, Microcontroller, Mobile Devices.

Contenido

[AGRADECIMIENTOS 7](#_Toc392794535)

[INTRODUCCIÓN 8](#_Toc392794536)

[JUSTIFICACIÓN 9](#_Toc392794537)

[OBJETIVO GENERAL Y OBJETIVOS ESPECÍFICOS 10](#_Toc392794538)

[Objetivo General 10](#_Toc392794539)

[Objetivos Específicos 10](#_Toc392794540)

[ANTECEDENTES 11](#_Toc392794541)

[CAPÍTULO I: ESBOZO TEÓRICO 14](#_Toc392794542)

[1.1 Ingeniería en Sistemas Computacionales (ISC) 14](#_Toc392794543)

[1.2 Planificación y modelado 14](#_Toc392794544)

[1.3 Circuitos eléctricos y electrónicos 15](#_Toc392794545)

[1.4 Programación orientada a objetos (POO) 15](#_Toc392794546)

[1.5 Taller de Base de Datos 16](#_Toc392794547)

[1.6 Programación Web 16](#_Toc392794548)

[1.7 Arduino 16](#_Toc392794549)

[1.8 Sql Server 2012 19](#_Toc392794550)

[1.9 Visual Studio 2013 21](#_Toc392794551)

[1.10 C# 21](#_Toc392794552)

[1.11 Xamarin 22](#_Toc392794553)

[CAPÍTULO II: DESARROLLO DEL PROYECTO 23](#_Toc392794554)

[2.1 Procedimiento y descripción de las actividades realizadas 23](#_Toc392794555)

[Fichas de tareas. 24](#_Toc392794556)

[2.1.1 Análisis de requerimientos 29](#_Toc392794557)

[2.1.2 Definir herramientas de trabajo 29](#_Toc392794558)

[2.1.3 Diseño de la base de datos 30](#_Toc392794559)

[2.1.4 Estructura de los circuitos electrónicos 32](#_Toc392794560)

[2.1.5 Diagramas UML 46](#_Toc392794561)

[2.1.6 Desarrollo del Software 52](#_Toc392794562)

[2.1.7. Grabar las rutinas en el hardware. 84](#_Toc392794563)

[2.1.8 Pruebas 84](#_Toc392794564)

[2.1.9 Implementación. 84](#_Toc392794565)

[CAPITULO III: LOGROS Y RESULTADOS OBTENIDOS 85](#_Toc392794566)

[CAPITULO IV: MANUAL DE OPERACIONES 87](#_Toc392794567)

[CAPÍTULO V: ANEXOS 102](#_Toc392794568)

[Glosario de términos 102](#_Toc392794569)

[Referencias bibliográficas y virtuales 105](#_Toc392794570)

[Figura 2.1: Tabla tblUsuario 1 32](#_Toc392597339)

[Figura 2.2: Tabla tblToken 2 33](#_Toc392597340)

[Figura 2.3: Tabla tblEvento 3 33](#_Toc392597341)

[Figura 2.4: Diagrama relacional 4 34](#_Toc392597342)

[Figura 2.5: Terminales del puente h L293D. 5 35](#_Toc392597343)

[Figura 2.6: Ejemplo de uso del puente h L293D 6 36](#_Toc392597344)

[Figura 2.7: Placa Arduino Mega 7 37](#_Toc392597345)

[Figura 2.8: Bluetooth HC-05 8 39](#_Toc392597346)

[Figura 2.9: Ultrasónico HC-SR04 9 40](#_Toc392597347)

[Figura 2.10: Arreglo de pilas en paralelo 10 42](#_Toc392597348)

[Figura 2.11: Sensor de gas MQ4 11 44](#_Toc392597349)

[Figura 2.12: Definición del pin 12 46](#_Toc392597350)

[Figura 2.13: Motorreductor doble eje 13 47](#_Toc392597351)

[Figura 2.14: Estructura de Protoboard 14 48](#_Toc392597352)

[Figura 2.15: Ejemplo de conexión de Protoboard 15 49](#_Toc392597353)

[Figura 2.16: Ejemplo de una mejor conexión de Protoboard 16 50](#_Toc392597354)

[Figura 2.17: Conexión de Bluetooth con Arduino 17 51](#_Toc392597355)

[Figura 2.18: Representación final 18 52](#_Toc392597356)

# AGRADECIMIENTOS

Primeramente a Dios, por habernos guiado y acompañado a lo largo de nuestra carrera, ya que nos brindó fortaleza y optimismo en los momentos más difíciles para nunca perder la esperanza.

A nuestros padres, ya que ellos son parte esencial en nuestras vidas, además por brindarnos su apoyo incondicional en todo momento, por esa palabra de aliento en el momento que más la necesitábamos y por aquellos consejos que nos han dado en el transcurso de nuestras vidas, los cuales han sido de gran importancia para poder cumplir nuestras metas.

A mis compañeros del proyecto, por el tiempo compartido, por los consejos, enseñanzas y por la confianza que generamos durante toda la carrera.

# INTRODUCCIÓN

El uso de la tecnología ha ido incrementando a pasos agigantados, esto ha sido de gran ayuda, ya que debido a esta evolución se pueden tener soluciones más sencillas y a menor costo comparado con los productos que están en venta en el mercado, cabe mencionar que para manipular las tecnologías es necesario saber o adquirir información y conocimiento acerca de esta para poder hacer un buen manejo de esta al momento de estar trabajando.

Actualmente en el país se cuenta con sistemas o robots controlados a través de un ordenador, los cuales son de alto costo, además de que deben de ser operadas por personas especializadas y/o personas capacitadas.

Nuestro proyecto ofrece una solución innovadora a un menor costo, y se pretende que pueda ser utilizado por instituciones de auxilio.

Este prototipo de micro robot se diseñó para llevar a cabo tareas específicas. Dichas tareas como son el monitoreo de eventos de alto riesgo, en lugares pequeños o de difícil acceso.

# JUSTIFICACIÓN

El presente proyecto tiene como principal objetivo realizar un prototipo de micro-robot. Este prototipo está diseñado para llevar a cabo tareas específicas, tales como el monitoreo de alto riesgo en lugares pequeños o de difícil acceso para los seres humanos, así como monitorear la calidad del ambiente en lugares donde existe riesgo de emisión de gases.

Nuestro país cuenta con sistemas o robots controlados a través de un ordenador, los cuales son de alto costo, además de que deben de ser operados por personas especializadas, es por ello que hemos propuesto este prototipo que ofrece una opción a un menor costo y es de gran apoyo para grupos de rescate.

Este tipo de investigaciones sin duda aportan grandes beneficios a la sociedad en general, ya que estas tecnologías traen ventajas al evitar poner en riesgo la integridad de las personas.

Por esta razón el proyecto describe en este documento la experiencia adquirida durante el proceso de desarrollo en esta Institución y también es el resultado de una serie de investigaciones previas sobre este tema.

# OBJETIVO GENERAL Y OBJETIVOS ESPECÍFICOS

## Objetivo General

Diseñar un micro robot (prototipo) utilizando dispositivos electrónicos y tecnologías de la información (TI) para monitorear áreas de difícil acceso y de alto riesgo.

## Objetivos Específicos

* Describir los dispositivos electrónicos a utilizar.
* Comunicar los diferentes dispositivos utilizando interfaces y las plataformas de desarrollo.
* Generar los sketchs para la comunicación de los dispositivos móviles con el Arduino.
* Dar a conocer desarrollos personales que hemos obtenido de conocimientos adquiridos, para exponer algunas las ventajas.

# ANTECEDENTES

A pesar de que México ha tenido un papel digno en competencias mundiales de robótica, nuestro país en esta rama está marcada por los escasos recursos destinados a la ciencia y tecnología, así como por la falta de desarrolladores.

Hay soluciones muy ingeniosas, en el concurso de limpiadores de playa, el concurso latinoamericano, se vio que los robots han avanzado, en las primeras competencias los robots se atoraban en la arena. Ahora ya no se atoran fácilmente. También en el reconocimiento de los objetos: hay robots que toman indiscriminadamente la arena y las latas, pero hay otros que son muy selectivos, van con su cámara o con un sensor. Se dirige el robot aunque tome todas las cosas que encuentra a su paso pero ya no camina como loco.

En nuestro país existe la FMR (Federación Mexicana de Robótica) integra a instituciones como el Cinvestav, el Instituto Nacional de Astrófísica, Óptica y Electrónica (INAOE), el Instituto Politécnico Nacional (IPN), el Instituto Tecnológico Autónomo de México (ITAM), el Tecnológico de Monterrey (ITESM), la Universidad La Salle, la Universidad Nacional Autónoma de México y la Universidad Popular Autónoma del Estado de Puebla (UPAEP). La FMR realiza actividades desde el 2007 pero fue hasta el 2010 cuando consiguió su conformación legal.

México ha sido el escenario de graves desastres: huracanes, terremotos, incendios y derrumbes causan cada año la pérdida de cientos de vidas y numerosas víctimas que deben ser sacadas de entre los escombros.

Conscientes de la necesidad de mejorar los sistemas para la localización y rescate de sobrevivientes atrapados en ruinas, terregales o edificios colapsados, así como de contar con equipos especiales para la remoción de objetos peligrosos o contaminantes, estudiantes del Instituto Tecnológico de Nuevo León crearon el robot CuerBot, curioso artefacto similar a un auto todo terreno. La estructura mecánica del CuerBot se basa en un mecanismo de eslabones para evitar que se mueva su centro de gravedad; por ello se adhiere prácticamente al terreno por el que transita. Está construido sobre una estructura de aluminio, tiene seis ruedas y es teledirigido por un operador rescatista, por medio de una computadora remota con la que se controlan los movimientos del robot y se lee la información que éste transmite.

Dispone de sistemas capaces de medir el ritmo de respiración y la temperatura de la víctima; así como la temperatura del ambiente, registro de audio y grabación - transmisión de video.

Con el apoyo de sensores de desplazamiento, el sistema puede elaborar un mapa trazado con la ruta del robot. Lo novedoso de CuerBot es el empleo de un algoritmo que detecta el estado y la posición de la víctima.

El equipo de trabajo creador del CuerBot está conformado por los alumnos del Instituto Tecnológico de Nuevo León: Israel López, Antonio Almaraz, Salvador Rubio, Ángel Díaz y Cristóbal López; además de Axel Candanosa, estudiante de la Universidad TecMilenio, todos ellos bajo la asesoría del Prof. José de Jesús López.

El Instituto Tecnológico de Nuevo León participó con este robot en el Campeonato Mundial World Championship RoboCup en su edición del 2009, que se realizó en la Ciudad de Graz, en Austria. En él tomaron parte 15 equipos mexicanos, provenientes de prestigiosas instituciones de educación superior: UNAM, Instituto Tecnológico Autónomo de México (ITAM), Tecnológico de Monterrey (ITESM) campus Edo de México, ITESM-Monterrey, Universidad La Salle-DF, U. Panamericana-Guadalajara, Instituto Nacional de Astrofísica, Óptica y Electrónica (INAOE), la Universidad Popular Autónoma del Estado de Puebla (UPAEP) y, desde luego, el Instituto Tecnológico de Nuevo León.

Esta competencia cumbre de la inteligencia artificial y la robótica fue organizada por la Universidad de Tecnología de Graz y tuvo como sede el Centro de Convenciones denominado Stadthalle. Se dividió en cuatro grandes categorías: Robots Futboleros, Robots de Servicio, Robots de Rescate y Robots de Jóvenes y Niños.

<http://eleconomista.com.mx/tecnociencia/2012/05/04/robotica-mexico-pocos-recursos-desarrolladores>

<http://www.snit.mx/ciencia-y-tecnologia/alumnos-del-it-de-nuevo-leon-crean-robot-para-rescate-de-victimas-de-desastres>

# CAPÍTULO I: ESBOZO TEÓRICO

## 1.1 Ingeniería en Sistemas Computacionales (ISC)

Es un modo de enfoque permite estudiar y comprender la realidad, con el propósito de implementar u optimizar sistemas informáticos complejos. Puede verse como la aplicación tecnológica de la teoría de sistemas a los esfuerzos de la ingeniería, adoptando en todo este trabajo el paradigma sistémico. La ingeniería en sistemas integra otras disciplinas y grupos de especialidad en un esfuerzo de equipo, formando un proceso de desarrollo estructurado.

Lo primero que debe hacer el ingeniero de sistemas es identificar y entender que problema se quiere resolver, cual es la razón que tiene la organización para creer que eso lo podría solucionar y facilitar el trabajo a los usuarios finales. Es necesario que interactúe permanentemente con los implicados en el problema, ya que son ellos en últimas quienes sienten y perciben que haya demoras en sus procesos, que no hay buen rendimiento por causa de otros procesos y que se requiere de una optimización.

## 1.2 Planificación y modelado

Es una disciplina para desarrollar una especificación completa, consistente no ambigua, la cual servirá como base para los acuerdos comunes entre todas las partes involucradas y en donde se describen las funciones que realizará el sistema.

Características

* Permite gestionar las necesidades del proyecto en forma estructurada.
* Mejora la capacidad de predecir cronogramas de proyectos así como sus resultados.
* Disminuye los costos y retrasos del proyecto.
* Mejora la calidad del software.
* Evita rechazos de usuarios finales.

## 1.3 Circuitos eléctricos y electrónicos

Consolidar su formación básica como ingeniero y se potencia su capacidad de análisis e interpretación de datos; aportando a su perfil: los conocimientos necesarios para, analizar e integrar equipos y/o sistemas electrónicos para la solución de problemas en el entorno profesional, aplicando normas técnicas y estándares nacionales e internacionales, haciendo uso de la simulación de modelos que permitan predecir el comportamiento de sistemas electrónicos empleando plataformas computacionales Aporta al perfil del Ingeniero, la capacidad de Diseñar, analizar y construir equipos y/o sistemas electrónicos para la solución de problemas en el entorno profesional, aplicando normas, técnicas y estándares nacionales e internacionales.

Crear, innovar y transferir tecnología aplicando métodos y procedimientos en proyectos de ingeniería electrónica tomando en cuenta el desarrollo sustentable.

Simular modelos que permitan predecir el comportamiento de sistemas electrónicos empleando plataformas computacionales.

## 1.4 Programación orientada a objetos (POO)

Es un paradigma de programación que usa los objetos en sus interacciones, para diseñar aplicaciones y programas informáticos. Está basado en varias técnicas, incluyendo herencia, cohesión, abstracción, polimorfismo, acoplamiento y encapsulamiento. Su uso se popularizó a principios de la década de los años 1990. En la actualidad, existe una gran variedad de lenguajes de programación que soportan la orientación a objetos.

Características

* Abstracción
* Encapsulamiento
* Modularidad
* Polimorfismo
* Herencia
* Recolección de basura

## 1.5 Taller de Base de Datos

Aportar al perfil del Ingeniero en Sistemas Computacionales las competencias para diseñar y desarrollar bases de datos conforme a los requerimientos definidos, las normas organizacionales de manejo y seguridad de la información, utilizando tecnologías emergentes con el fin de integrar soluciones computacionales con diferentes tecnologías, plataformas o dispositivos considerando los aspectos legales, éticos, sociales y de desarrollo sustentable.

## 1.6 Programación Web

Aporta al perfil del Ingeniero la capacidad para desarrollar e implementar sistemas de información en ambiente web para la automatización de procesos y toma de decisiones utilizando metodologías basadas en estándares internacionales y tecnologías emergentes, introduciéndonos a la arquitectura de las aplicaciones web, los conceptos básicos del lenguaje de marcas, al lenguaje de presentación de datos, al desarrollo de código de lado cliente y servidor e implementación de servicios web.

## 1.7 Arduino

Es una plataforma de creación de prototipos electrónicos de código abierto basado en hardware y software fácil de usar, flexible. Está dirigido a artistas, diseñadores, aficionados y cualquier persona interesada en la creación de objetos o entornos interactivos.

Funciones

Arduino puede detectar el medio ambiente mediante la recepción de la entrada de una variedad de sensores y puede afectar a sus alrededores por las luces de control, motores, y otros actuadores. El microcontrolador en la placa se programa mediante el lenguaje de programación Arduino (basado en Wiring) y el entorno de desarrollo Arduino (basado en Processing). Proyectos Arduino puede ser independiente o se pueden comunicar con el software que se ejecuta en un ordenador (por ejemplo Flash).

Los tableros pueden ser construidos por la mano o comprados premontado, el software puede ser descargado de forma gratuita. Los diseños de referencia de hardware.

(Archivos CAD) son disponibles bajo una licencia de código abierto, que son libres de adaptarlos a sus necesidades.

Historia

* Arduino nació como un proyecto educativo allá por el año 2005 sin pensar que algunos años más tarde se convertiría en líder del mundo DIY (Do It Yourself).
* Su nombre viene del nombre del bar Bar di Re Arduino donde Massimo Banzi pasaba algunas horas, el cual a su vez viene del nombre de un antiguo rey europeo allá por el año 1002.
* Banzi dice que nunca surgió como una idea de negocio, es más nació por una necesidad de subsistir ante el eminente cierre del Instituto de diseño Interactivo IVREA en Italia. Es decir, al crear productos open hardware (de uso público) no podría ser embargado. Es más hoy en día Arduino tiene la difícil tarea de subsistir comercialmente y continuar en continuo crecimiento.
* A la fecha se han vendido más de 250 mil placas en todo el mundo sin contar las versiones clones y compatibles.
* Para su creación participaron alumnos que desarrollaban sus tesis como Hernando Barragan (Colombia) quien desarrollo la plataforma de programación Wiring con la cual se programa el microcontrolador.
* Hoy en día con Arduino se pueden fabricar infinidad de prototipos y cada vez su uso se viene expandiendo más. Desde cubos de leds, sistemas de automatización en casa (domotica), integración con el Internet, displays Twitter, kit analizadores de ADN.
* Google ha apostado por el proyecto y ha colaborado en el Android ADK (Accesory Development Kit), una placa Arduino capaz de comunicarse directamente con spmartphones Android para obtener las funcionalidades del teléfono (GPS, acelerómetros, GSM, y bases de datos) y viceversa para que el teléfono controle luces, motores y sensores conectados de Arduino.

Tipos de Placas de Arduino

* Arduino Mega.
* Arduino Leonardo.
* Arduino Pro.
* Arduino Nano.
* Arduino Mini.

**¿Porque utilizar Arduino?**

Hay muchos otros microcontroladores y plataformas con microcontroladores disponibles para la computación física. Parallax Basic Stamp, BX-24 de Netmedia, Phidgets, Handyboard del MIT, y muchos otros ofrecen funcionalidades similares.

Todas estas herramientas organizan el complicado trabajo de programar un microcontrolador en paquetes fáciles de usar. Arduino, además de simplificar el proceso de trabajar con microcontroladores, ofrece algunas ventajas respecto a otros sistemas a profesores, estudiantes y amateurs:

* Accesible**:** Las placas Arduino son más accesibles comparadas con otras plataformas de microcontroladores. La versión más cara de un módulo de Arduino puede ser montada a mano, e incluso ya montada cuesta bastante menos de 60€.
* Multi-Plataforma: El software de Arduino funciona en los sistemas operativos Windows, Macintosh OSX y Linux. La mayoría de los entornos para microcontroladores están limitados a Windows.
* Entorno de programación simple y directa: El entorno de programación de Arduino es fácil de usar para principiantes y lo suficientemente flexible para los usuarios avanzados. Pensando en los profesores, Arduino está basado en el entorno de programación de Procesing con lo que el estudiante que aprenda a programar en este entorno se sentirá familiarizado con el entorno de desarrollo Arduino.
* Software ampliable y de código abierto: El software Arduino está publicado bajo una licencia libre y preparada para ser ampliado por programadores experimentados. El lenguaje puede ampliarse a través de librerías de C++, y si se está interesado en profundizar en los detalles técnicos, se puede dar el salto a la programación en el lenguaje AVR C en el que está basado. De igual modo se puede añadir directamente código en AVR C en tus programas si así lo deseas.
* Hardware ampliable y de Código abierto: Arduino está basado en los microcontroladores ATMEGA168, ATMEGA328 y ATMEGA1280. Los planos de los módulos están publicados bajo licencia Creative Commons, por lo que diseñadores de circuitos con experiencia pueden hacer su propia versión del módulo, ampliándolo u optimizándolo. Incluso usuarios relativamente inexpertos pueden construir la versión para placa de desarrollo para entender cómo funciona y ahorrar algo de dinero.

## 1.8 Sql Server 2012

Es un sistema de gestión de base de datos relacional de Microsoft (RDBMS). Es una base de datos con todas las funciones, principalmente diseñado para competir frente a los competidores de Oracle Database (DB) y MySQL. Al igual que todos los grandes RBDMS, SQL Server es compatible con ANSI SQL, el lenguaje SQL estándar. Sin embargo, SQL Server también contiene T-SQL, su propia IMPLEMENTACIÓN SQL. SQL Server Management Studio (SSMS) (anteriormente conocido como Enterprise Manager) es una herramienta de la interfaz principal de SQL Server, y es compatible con entornos de 32 bits y 64 bits.

La liberación de SQL Server 2012 simplificará las opciones para el cliente con tres ediciones principales que están estrechamente alineadas con la manera en cómo los clientes usan SQL Server entre cargas de trabajo. SQL Server 2012 también entregar licenciamiento que permite a los clientes aprovechar más fácilmente los beneficios de la nube pública y privada a través de la virtualización y la movilidad de licencias. Nuestro nuevo modelo proporcionará:

Licenciamiento simplificado

SQL Server 2012 ofrecerá dos opciones de licenciamiento – una que se basa en el poder de la computación, y una que se basa en los usuarios o dispositivos.

Flexibilidad e innovación

Licenciamiento optimizado para la nube con la capacidad para otorgar la licencia a una sola Máquina Virtual (VM) u otorgar la licencia a un servidor para una virtualización máxima así como la flexibilidad para mover sus VMs de servidor a servidor, o a hosters o a la nube.

TCO líder de la industria

Sigue proporcionando valor sobresaliente a cualquier escala entre todos los principales proveedores. SQL Server 2012 ofrecerá características y capacidades de producto atractivas dentro de las tres ediciones principales con complementos económicos que se requieren para ejecutar sus aplicaciones de Misión crítica y de Inteligencia de negocios.

## 1.9 Visual Studio 2013

Es una colección completa de herramientas y servicios que le permitirá crear una gran variedad de aplicaciones, tanto para plataformas de Microsoft como para otras plataformas. Visual Studio conecta también todos los proyectos, equipos y partes interesadas. Ahora su equipo puede trabajar con gran agilidad desde casi cualquier parte, independientemente de la herramienta de desarrollo, incluidas Eclipse y Xcode. Si va a diseñar aplicaciones .NET críticas, escribir código rápido con C++ AMP o probar y depurar una aplicación HTML/JavaScript conectada a la nube que se ejecute en muchos dispositivos, únase a millones de desarrolladores de todo el mundo que han elegido Visual Studio como entorno de desarrollo esencial.

Visual Studio es flexible e integrado con el fin de ayudarle a adoptar prácticas de desarrollo ágiles a su ritmo. Tanto si trabaja en modo local como si trabaja en la nube, las herramientas de administración del ciclo de vida de Visual Studio permiten entregar software en ciclos más cortos y con mayor agilidad, a la vez que se aumenta la calidad del software. Las herramientas integradas sustentan todos los aspectos del proceso de desarrollo, desde la administración del trabajo pendiente hasta el planeamiento de sprints y la entrega continua.

## 1.10 C#

Es un lenguaje de alto nivel, multi-paradigma, el lenguaje de programación moderno, de propósito general para la creación de aplicaciones utilizando el. NET Framework Visual Studio y. C # ha sido diseñado para ser simple, de gran alcance, con seguridad de tipos y orientado a objetos. Las numerosas innovaciones en C # permiten el desarrollo rápido de aplicaciones sin perder la expresividad y elegancia de lenguajes de tipo C.

El nombre C Sharp fue inspirado por la notación musical, donde '#' (sostenido, en inglés sharp) indica que la nota (C es la nota do en inglés) es un semitono más alta, sugiriendo que C# es superior a C/C++. Además, el signo '#' se compone de cuatro signos '+' pegados.

## 1.11 Xamarin

Es una empresa fundada por Miguel de Icaza y Nat Friedman en mayo del 2011 cuyo objetivo es proporcionar una plataforma de desarrollo de aplicaciones móviles especialmente centrada en iOS y Android utilizando la plataforma Mono y con C# como lenguaje de desarrollo.

La zona de componentes nos permite incorporar bibliotecas establecidas de controles que facilitan el desarrollo de aplicaciones, la principal ventaja en usar C# en las aplicaciones móviles es el ahorro significativo de tiempo de desarrollo comparado con usar otros frameworks o usar las plataformas oficiales.

# CAPÍTULO II: DESARROLLO DEL PROYECTO

## 2.1 Procedimiento y descripción de las actividades realizadas

* Análisis de requerimientos.
* Definir herramientas de trabajo.
* Diseño de diagrama para base de datos.
* Realizar base de datos.
* Estructura de los circuitos electrónicos.
* Diseño de Software.
* Desarrollo de Software.
* Grabar las rutinas en hardware.
* Generar Pruebas.
* Implementación.

Fichas de tareas.

*Tabla 2.1: Análisis de requerimientos.*

|  |  |
| --- | --- |
| Descripción | Investigar los componentes para el micro-robot. |
| Esfuerzo Estimado | Miembros del equipo. |
| Recursos | Internet y libros. |
| Duración | 1 semana. |
| Entregable | Análisis de los componentes a utilizar |
| Antecesora | - |
| Sucesora | Definir herramientas de trabajo. |

*Tabla 2.2: Definir herramientas de trabajo.*

|  |  |
| --- | --- |
| Descripción | Analizar cuáles son las mejores herramientas para el desarrollo del micro-robot. |
| Esfuerzo Estimado | Miembros del equipo. |
| Recursos | Internet y libros. |
| Duración | 2 días. |
| Entregable | Herramientas de trabajo. |
| Antecesora | Análisis de requerimientos. |
| Sucesora | Diseño de diagrama para base de datos. |

*Tabla 2.3: Diseño de diagrama para base de datos.*

|  |  |
| --- | --- |
| Descripción | Realización del Diseño de diagrama de base de datos. |
| Esfuerzo Estimado | Miembros del equipo. |
| Recursos | Software para realizar diagramas. |
| Duración | 1 hora. |
| Entregable | Diagrama de la base de datos. |
| Antecesora | Definir herramientas de trabajo. |
| Sucesora | Realizar base de datos. |

*Tabla 2.4: Realizar base de datos.*

|  |  |
| --- | --- |
| Descripción | Creación de la base de datos. |
| Esfuerzo Estimado | Miembros del equipo. |
| Recursos | Sql Server 2012 |
| Duración | 1 hora. |
| Entregable | Base de datos lista para ser usada. |
| Antecesora | Diseño de diagrama para base de datos. |
| Sucesora | Estructura de los circuitos electrónicos. |

*Tabla 2.5: Estructura de los circuitos electrónicos.*

|  |  |
| --- | --- |
| Descripción | Realización de los diagramas de los circuitos electrónicos. |
| Esfuerzo Estimado | Miembros del equipo. |
| Recursos | Software para realizar diagramas. |
| Duración | 3 días. |
| Entregable | Diagramas de los circuitos electrónicos de fácil comprensión. |
| Antecesora | Realizar base de datos. |
| Sucesora | Diseño de Software. |

*Tabla 2.6: Diseño de Software.*

|  |  |
| --- | --- |
| Descripción | Realización de los diagramas de los circuitos electrónicos. |
| Esfuerzo Estimado | Diagramas UML. |
| Recursos | Software para realizar diagramas. |
| Duración | 2 días. |
| Entregable | Diagramas de fácil comprensión para que sea más rápida la programación. |
| Antecesora | Estructura de los circuitos electrónicos. |
| Sucesora | Desarrollo de Software. |

*Tabla 2.7: Desarrollo de Software.*

|  |  |
| --- | --- |
| Descripción | Codificación de los programas y sketch’s del micro-robot. |
| Esfuerzo Estimado | Miembros del equipo. |
| Recursos | Visual Studio 2013. |
| Duración | 4 semanas. |
| Entregable | Programas y sketch’s terminados. |
| Antecesora | Estructura de los circuitos electrónicos. |
| Sucesora | Grabar las rutinas en hardware. |

*Tabla 2.8: Grabar las rutinas en hardware.*

|  |  |
| --- | --- |
| Descripción | Grabar las rutinas en hardware. |
| Esfuerzo Estimado | Miembros del equipo. |
| Recursos | Visual Studio 2013. |
| Duración | 3 días. |
| Entregable | Interfaz lista para usarse. |
| Antecesora | Desarrollo de Software. |
| Sucesora | Generar Pruebas. |

*Tabla 2.9: Generar Pruebas.*

|  |  |
| --- | --- |
| Descripción | Probar el micro-robot ya terminado. |
| Esfuerzo Estimado | Miembros del equipo. |
| Recursos | Micro-robot terminado. |
| Duración | 3 días. |
| Entregable | Micro-robot listo para implementarse. |
| Antecesora | Grabar las rutinas en hardware. |
| Sucesora | Implementación. |

*Tabla 2.10: Implementación.*

|  |  |
| --- | --- |
| Descripción | Implementar micro-robot terminado. |
| Esfuerzo Estimado | Miembros del equipo. |
| Recursos | Micro-robot terminado. |
| Duración | 2 días. |
| Entregable | Micro-robot listo para usarse. |
| Antecesora | Generar pruebas. |
| Sucesora | - |

## 2.1.1 Análisis de requerimientos

Identificación del proyecto a realizar, y de esta manera poder hacer una investigación de los componentes electrónicos, software y accesorios para el desarrollo del mismo.

Para este procedimiento se realizaron varias propuestas del proyecto a realizar, después se hicieron varias investigaciones para ver el material a utilizar y las plataformas del desarrollo.

## 2.1.2 Definir herramientas de trabajo

Para la realización de nuestro proyecto requerimos analizar a fondo las herramientas de trabajo (software, hardware, circuitos eléctricos y electronicos).

Utilizamos:

* Arduino Ide.
* Sql server 2012.
* Xamarin Studio.
* Visual Studio 2013.
* 2 puentes H L293D.
* Arduino Mega.
* Bluetooth HC-05.
* 2 sensores de a proximidad (Ultrasónicos) HC-SR04.
* Arreglo de pilas AA.
* Pila de 9v (pila cuadrada).
* Cámara.
* Sensor de gas MQ4.
* 4 motoreductores de doble eje.
* 4 llantas.
* Chasis acrílico.
* 1 Protoboard o breadbord.

## 2.1.3 Diseño de la base de datos

Se realizó un diagrama para la base de datos que guardara la información del micro-robot. Las tablas estarán relacionadas con una o varias tablas a la vez ya que se manejara búsqueda de información, y de esta manera se pretende tener un mejor control de nuestra información y será más eficaz al momento de consultar algún dato.

Esta herramienta utilizada se le conoce como “Modelo Entidad-Relación” el cual consiste en buscar entidades que describen los objetos que intervienen.

Esto se va graficando es un esquema para de esta manera poder al programador al momento de la codificación, y al usuario comprender el funcionamiento del proyecto.



Figura 2.1: Tabla tblUsuario 1

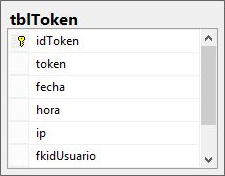
**

Figura 2.2: Tabla tblToken 2

**

Figura 2.3: Tabla tblEvento 3

Y al final quedaría de la siguiente manera la relación de todas las tablas.

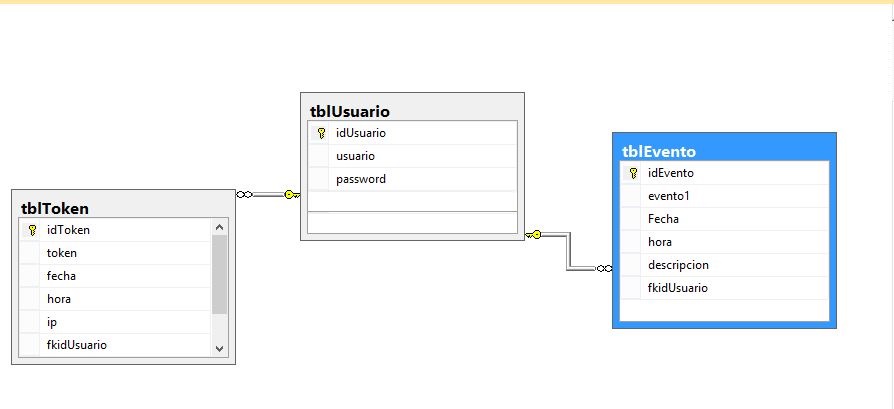


Figura 2.4: Diagrama relacional 4

## 2.1.4 Estructura de los circuitos electrónicos

Puente H L293D

Es un circuito integrado que se usa para el control de dos motores pequeños que pueden ser controladores simultáneamente y, además, en forma bidireccional.

Este integrado está limitado en cuanto a su capacidad de manejar corriente, por lo que se debe consultar la hoja de datos de la versión específica.

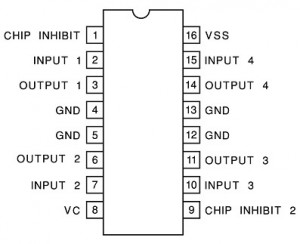
[](http://oficinaenlared.com/robotica101/wp-content/uploads/2011/09/terminalesL293.jpg)

Figura 2.5: Terminales del puente h L293D. 5

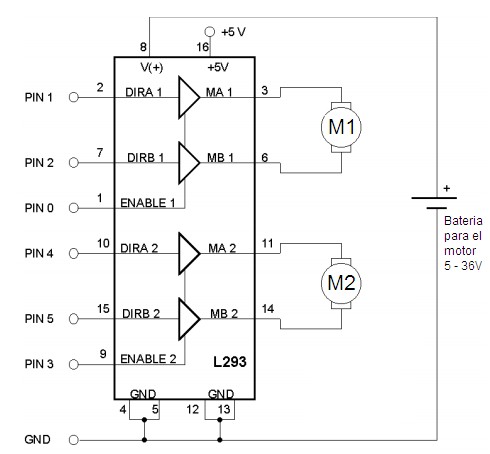


Figura 2.6: Ejemplo de uso del puente h L293D 6

La tierra del L293 se conecta al negativo de la batería y a la tierra del PIC.

Se puede conectar el pin Enable a +5V para usar solo los pines de dirección para encender y apagar el motor. En tal caso, sólo se requieren dos pines por motor. Si se coloca un pin en alto y el otro bajo, el motor girará en una dirección. Al invertir los voltajes, el motor girará en la otra dirección. Para detener el motor se coloca ambos pines en cero.

Arduino Mega

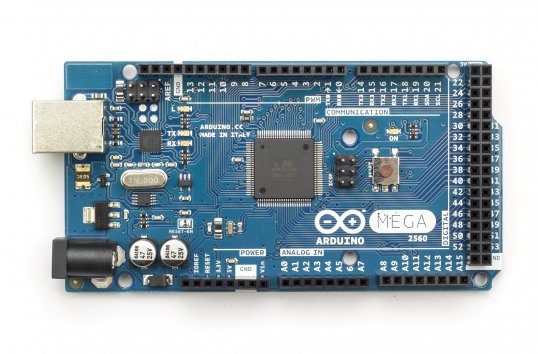
****

Figura 2.7: Placa Arduino Mega 7

Es una placa electrónica basada en los ATmega2560. Lleva 54 entradas / salidas digitales (de los cuales 14 pueden utilizarse para salidas PWM), 16 entradas analógicas, 4 UARTs (puertas seriales), un oscilador de 16MHz, una conexión USB, un conector de alimentación, un header ICSP y un botón de reset. Contiene todo lo necesario para apoyar el microcontrolador, simplemente conectarlo a un ordenador con un cable USB, o alimentarla con un adaptador de corriente AC a DC para empezar. La Mega es compatible con la mayoría de los shield para Arduino Uno, Duemilanove o Diecimila.

*Tabla 2.11: Especificaciones técnicas Arduino Mega.*

|  |  |
| --- | --- |
| Microcontroladores | ATmega2560 |
| Tensión de funcionamiento | 5V |
| Voltaje de entrada (recomendado) | 7-12V |
| Voltaje de entrada (límite) | 6-20V |
| Digital I / O Pins | 54 |
| PWM digital pines I / O | 14 |
| Pines de entrada analógica | 16 |
| Corriente continua para las E / S Pin | 40 Ma |
| Corriente de la CC para Pin 3.3V | 50 mA |
| Memoria Flash | 256 KB |
| Memoria flash para Bootloader | 8 KB |
| SRAM | 8 KB |
| EEPROM | 4 KB |
| Velocidad del reloj | 16 MHz |

Bluetooth HC-05.



Figura 2.8: Bluetooth HC-05 8

Viene configurado de fábrica para trabajar como maestro o esclavo.  En el modo maestro puede conectarse con otros módulos bluetooth, mientras que en el modo esclavo queda a la escucha peticiones de conexión.  Agregando este módulo a tu proyecto podrás controlar a distancia desde un celular o una laptop todas las funcionalidades que desees.

Características:

* 3.3 / 5 v.
* Chip BC417143.
* Alcance 10 metros.
* Nivel TTL.
* 1200bps a 1.3Mbps.

Sensor de a proximidad (Ultrasónicos) HC-SR04.

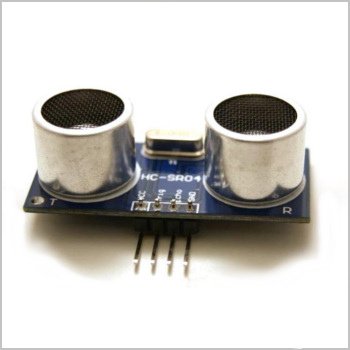


Figura 2.9: Ultrasónico HC-SR04 9

Es un sensor ultrasónico, este utiliza el sonar para determinar la distancia a un objeto. Ofrece una precisión excelente y lecturas estables. Su funcionamiento no se ve afectado por la luz solar o materiales obscuros.

Es una buena opción dentro de la gama de los sensores ultrasónicos. Su comportamiento es estable y tiene mucha exactitud en sus mediciones lo que lo hace muy popular dentro del área de robótica y mini robótica.

Características

* Voltaje: 5V DC.
* Corriente: Menor a 2mA.
* Ángulo detección: menor a 15 grados.
* Rango detección: de 2 a 450 cm.
* Precisión: cerca a los 2mm.

Arreglo de pilas AA.

Cuando las baterías se conectan en paralelo, el voltaje se mantiene. Lo que se logra es incrementar la Capacidad (A-h) y el Poder de Arranque (CCA).

En una conexión de baterías en paralelo, la capacidad del arreglo, es la suma de las capacidades de cada batería individual. El Poder de Arranque del arreglo, es la suma de los Poderes de Arranque de cada batería individual.

Ejemplo:

Si tenemos la batería de modelo PT88 (Norma JIS \*) de características:

Voltaje: 12

Capacidad: 88 A-h (Carga Eléctrica) a 20 horas.

Reserva de Capacidad: 155 minutos.

Poder de Arranque: 630 amperios.

Si conectamos dos de estas baterías en paralelo, tenemos como resultante un arreglo de las siguientes características:

Voltaje: 12

Capacidad: 88 + 88 = 176 A-h (Carga Eléctrica)

Reserva de Capacidad: 155 + 155 = 310 minutos.

Poder de Arranque: 630 + 630 = 1260 amperios.

La conexión en paralelo, consiste en enlazar los terminales positivos entre sí (que vendría a ser el borne positivo del arreglo) y los terminales negativos entre sí (que vendría a ser el borne negativo del arreglo).

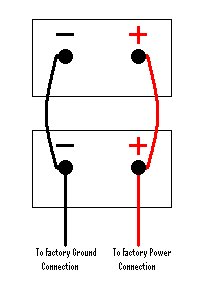


Figura 2.10: Arreglo de pilas en paralelo

Para aplicaciones de tipo estacionario, si con una batería tenemos una autonomía de 1 hora, con dos baterías en paralelo y bajo las mismas condiciones de servicio, podemos doblar el tiempo de autonomía.

Cuando estas dos baterías en paralelo se conectan a un cargador y leemos en el amperímetro del cargador una corriente de carga digamos de 10 amperios, entonces cada batería está recibiendo 5 amperios. La única condición es que las pilas sean del mismo voltaje.

Cámara IP

Es un dispositivo que emite las imágenes directamente sin necesidad de un ordenador con lo que nos brinda la posibilidad de ver lo que está pasando en cualquier lugar donde me encuentre.

Algo muy importante es que a diferencia de cualquier otro tipo de cámara, las cámaras ip no necesitan estar conectadas a una computadora ni dependen de ella, son totalmente independientes y autoadministrables, lo cual incrementa aún más su funcionalidad.

Se pueden instalar en cualquier sitio que disponga de conexión a Internet mediante Router ADSL, o computador en caso que usted quiera ver o grabar en el lugar de la instalación.

Sensor de gas MQ4



Figura 2.11: Sensor de gas MQ4 11

El MQ4 se utiliza en equipos de detección de fugas de gas en los mercados de consumo y de la industria, este sensor es adecuado para detectar CH4, gas natural, gas natural licuado, evitar la exposición a alcohol, humo de la cocina, y el humo del cigarrillo. La sensibilidad puede ser ajustada por el potenciómetro.

El sensor viene con una tarjeta de circuito impreso que tiene provisto un circuito de polarización del sensor y detector de umbral ajustable.

Lo único que usted tiene que hacer es ajustar el umbral de detección de Gas Natural a través del potenciómetro.

El circuito también incluye un LED para avisar la alarma. Es en realidad muy sencillo de usar, sólo requiere la alimentación de 5V a 200mA como mínimo. El sensor estará listo para detectar gas hasta después de unas 10 horas de "calentamiento". A partir de esto, ya puede calibrar el potenciómetro para la detección de fugas.

Especificación

* Las necesidades de suministro de energía: 5V
* Interfaz: Analógica.
* Pin Definición: 1 Salida, 2 GND, 3 VCC.
* Alta sensibilidad al gas natural CH4 £ ¬
* Sensibilidad al alcohol y el humo.
* Respuesta rápida.
* Vida estable y de largo.
* Circuito de accionamiento simple.
* Tamaño: 40x20mm

Definición del Pin

1. Señal de salida
2. GND
3. Potencia

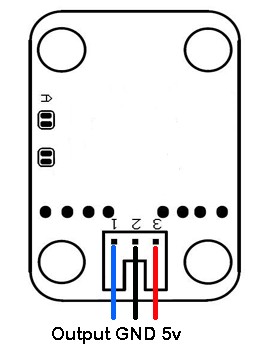


Figura 2.12: Definición del pin 12

Motorreductor de doble eje

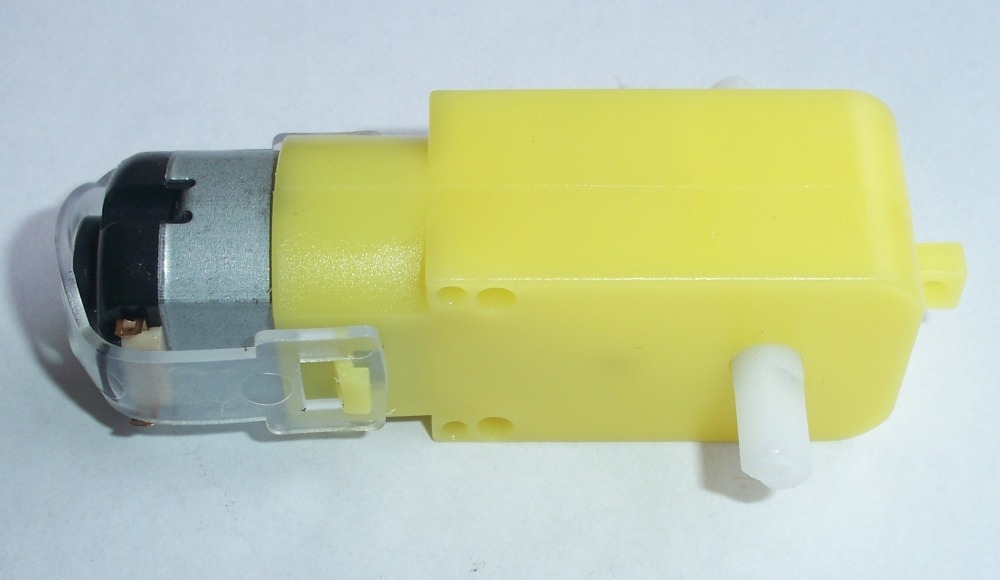


Figura 2.13: Motorreductor doble eje 13

Es apropiado para el accionamiento de toda clase de motores que necesitan reducir su velocidad en una forma segura y eficiente.

Al emplear REDUCTORES O MOTORREDUCTORES se obtiene una serie de beneficios sobre estas otras formas de reducción. Algunos de estos beneficios son:

* Una regularidad perfecta tanto en la velocidad como en la potencia transmitida.
* Una mayor eficiencia en la transmisión de la potencia suministrada por el motor.
* Mayor seguridad en la transmisión, reduciendo los costos en el mantenimiento.
* Menor espacio requerido y mayor rigidez en el montaje.
* Menor tiempo requerido para su instalación.

Protoboard o breadbord

Es una especie de tablero con orificios, en la cual se pueden insertar componentes electrónicos y cables para armar circuitos. Como su nombre lo indica, esta tableta sirve para experimentar con circuitos electrónicos, con lo que se asegura el buen funcionamiento del mismo.

Estructura del protoboard: Básicamente un protoboard se divide en tres regiones:

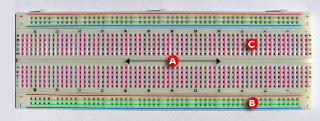


Figura 2.14: Estructura de Protoboard

A) Canal central: Es la región localizada en el medio del protoboard, se utiliza para colocar los circuitos integrados.

B) Buses: Los buses se localizan en ambos extremos del protoboard, se representan por las líneas rojas (buses positivos o de voltaje) y azules (buses negativos o de tierra) y conducen de acuerdo a estas, no existe conexión física entre ellas. La fuente de poder generalmente se conecta aquí.

C) Pistas: La pistas se localizan en la parte central del protoboard, se representan y conducen según las líneas rosas.

Recomendaciones al utilizar el protoboard: A continuación veremos una serie de consejos útiles pero no esenciales.

1.- Hacer las siguientes conexiones:

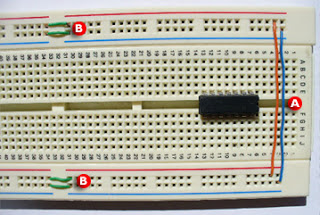


Figura 2.15: Ejemplo de conexión de Protoboard 15

A) Esta conexión nos sirve para que ambos pares de buses conduzcan corriente al agregarles una fuente de poder, así es más fácil manipular los circuitos integrados.

B) Algunos protoboards tienen separada la parte media de los buses, es por eso que se realiza esta conexión para darle continuidad a la corriente.

2.- Coloca los circuitos integrados en una sola dirección, de derecha a izquierda o viceversa.

3.- Evita el cableado aéreo (A), resulta confuso en circuitos complejos. Un cableado ordenado (B) mejora la comprensión y portabilidad.

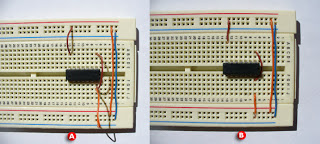


Figura 2.16: Ejemplo de una mejor conexión de Protoboard 16

Bluetooth con Android

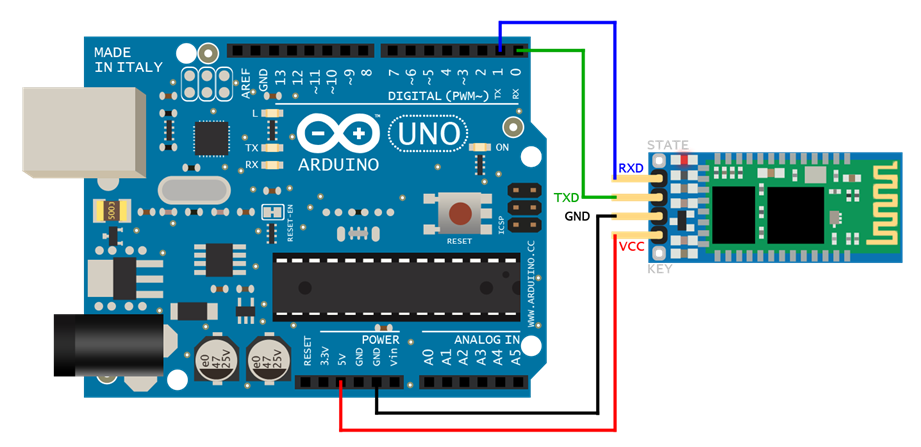


Figura 2.17: Conexión de Bluetooth con Arduino 17

La plataforma Android incluye soporte para la pila de red Bluetooth, que permite a un dispositivo para el intercambio de datos de forma inalámbrica con otros dispositivos Bluetooth. El marco de aplicación proporciona acceso a la funcionalidad Bluetooth a través de las API de Bluetooth de Android. Estas API permiten aplicaciones se conectan de forma inalámbrica a otros dispositivos Bluetooth, de punto a punto que permite y multipunto funciones inalámbricas.

El uso de las API de Bluetooth, una aplicación para Android puede realizar lo siguiente:

* Analizar en busca de otros dispositivos Bluetooth.
* Consultar el adaptador Bluetooth local para los dispositivos con Bluetooth emparejados.
* Establecer canales RFCOMM.
* Conectarse a otros dispositivos a través de descubrimiento de servicios.
* Transferencia de datos hacia y desde otros dispositivos.
* Gestión de múltiples conexiones.

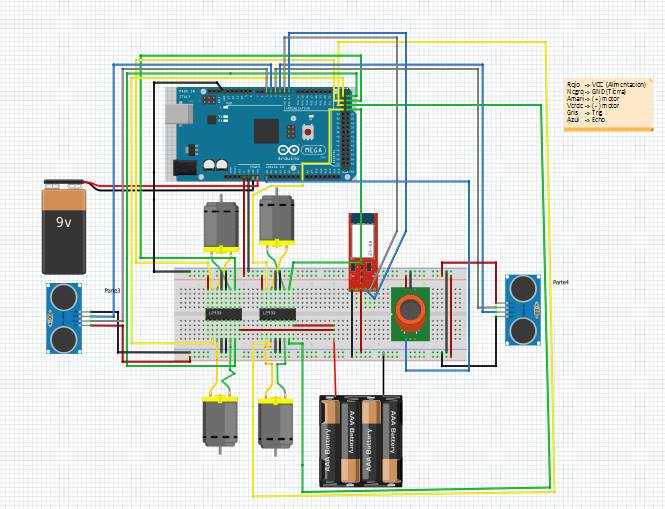
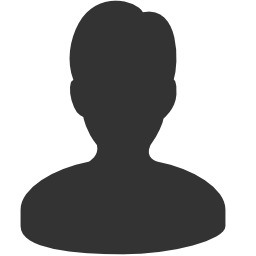


Figura 2.18: Representación final

## 2.1.5 Diagramas UML

**Diagramas de Casos de Uso**



**Diagrama de Clases**

Prueba\_Bluetooth

|  |
| --- |
| Registro |
| Id, Resultado, Fecha. |
| Guardar() |
| Conexión() |
| Entidad() |

|  |
| --- |
| Form1 |
| Bluetooh Client |
| Cargar() |
| CargarDatos() |
| Conectar() |
| ThreadProcSafe() |
| beginListenForData() |
|  |

App for Android

|  |
| --- |
| Principal |
| Adelante  Atrás  Izquierda  Derecha |
| CheckBt() |
| Connect() |
| writeData() |
| OnDestroy() |
| beginListenForData() |
| OnOff\_HandleCheckedChange() |

|  |
| --- |
| WebserviceActivity |
| Encender  Apagar  Buscar |
| Todos\_HandleClick() |
| Encender\_HandleClick() |
| Apagar\_HandleClick() |

|  |
| --- |
| ListaConectar |
| Nombre  Mac |
| OnStart() |
| DeviceListClick() |
| OnCreate() |
| OnDestroy() |
| DoDiscovery() |
| BtDevice |

|  |
| --- |
| ActivityInicial |
|  |
| btnWeb\_HandleClick() |
| btnBlue\_HandleClick() |

Servicio Web

|  |
| --- |
| tblToken |
| int IdToken  string Token  string Fecha  string Hora  string Ip  Nullable<int> fkIdUsuario  Nullable<int> Estado  string Descripcion |
|  |

|  |
| --- |
| tblUsuarios |
| int IdUsuario  string Usuario  string Password |
|  |

|  |
| --- |
| TlbEvento |
| int IdEvento  string Evento  string Fecha  string Hora  string Descripción  Nullable<int> fkIdUsuario |
|  |
|  |

|  |
| --- |
| AutoEntities |
| DbSet<tblEvento> tblEventoes  DbSet<tblToken> tblTokens  DbSet<tblUsuario> tblUsuarios |
| DbContext() |

|  |
| --- |
| SHA1Util |
| No contiene variables |
| string SHA1HashStringForUTF8String(string s) |

|  |
| --- |
| UsuarioController |
|  |
| IEnumerable<tblUsuario> GettblUsuarios()  tblUsuario GettblUsuario(int id)  HttpResponseMessage UpdateUsuario(tblUsuario tblusuario)  HttpResponseMessage PosttblUsuario(tblUsuario tblusuario)  HttpResponseMessage Login(tblUsuario tblusuario)  HttpResponseMessage UserId(tblUsuario tblusuario) |

|  |
| --- |
| EventoController |
| AutoEntities db |
| IEnumerable<tblEvento> GettblEventoes()  IEnumerable<tblEvento> GettblEvento(int id)  HttpResponseMessage PosttblEvento (tblEvento tblevento). |

|  |
| --- |
| TokenController |
| AutoEntities db |
| IEnumerable<tblToken> GettblToken(int id)  HttpResponseMessage downToken(tblToken tbltoken)  HttpResponseMessage IsValid(tblToken tbltoken) |

Página Web

|  |
| --- |
| Funciones |
| Int id |
| function CargarDatos()  function CargarRegistros(id)  function CargarSesiones(id)  function deleteS(id)  function LoadUser(id)  function showLoader()  function closeLoader()  function init():  function LogOK() |

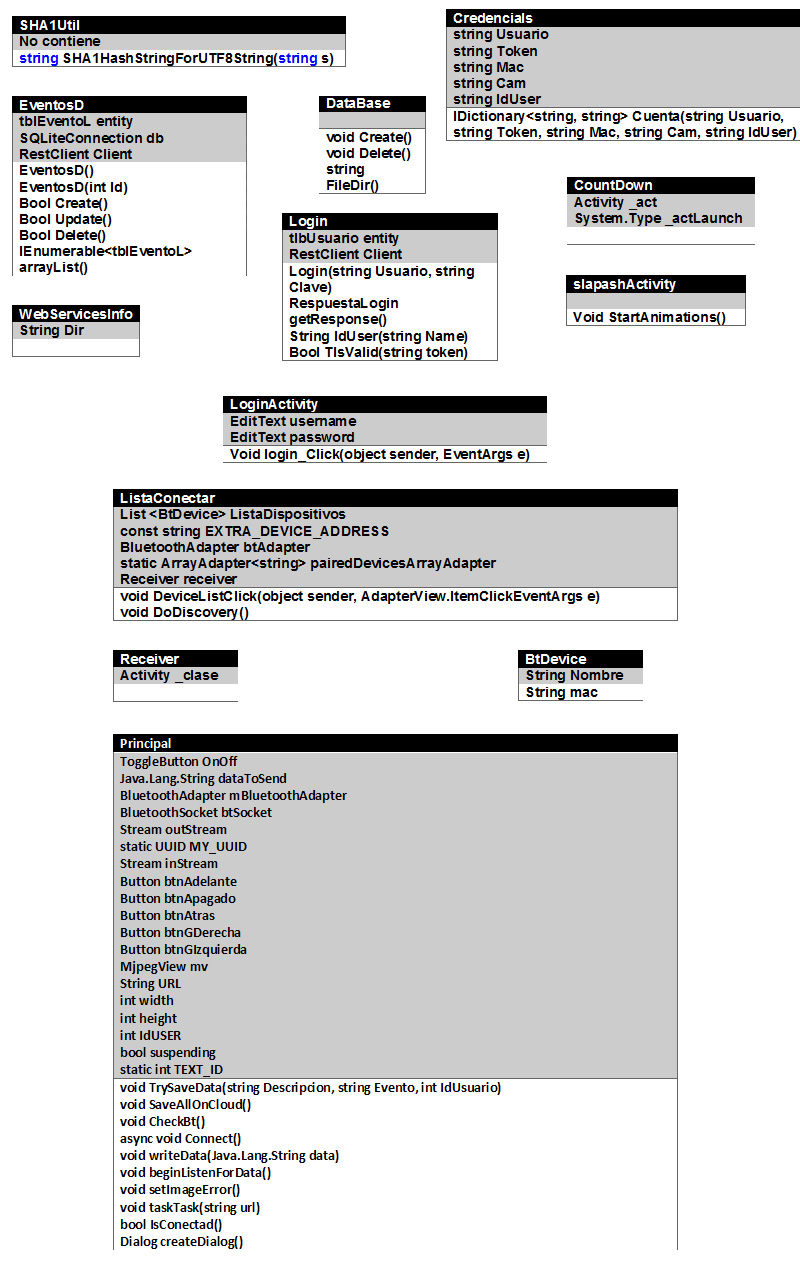
Aplicación Android

Diagrama de interacción



: Base de Datos

: Web Service

: Interface del usuario

: Página Web

Ingresar Datos Despliega la pantalla

Almacena registro

Registro de

Movimientos

Almacena Datos

Confirmación de registro

Registrarme

Muestra información

## 2.1.6 Desarrollo del Software

Servicios Web

EventoController: Clase manejadora de los servicios web para tabla tblEvento.

public class EventoController : ApiController

{

private AutoEntities db = new AutoEntities();

// GET api/Evento

public IEnumerable<tblEvento> GettblEventoes()

{

return db.tblEventoes.AsEnumerable();

}

// GET api/GettblEvento/5

public IEnumerable<tblEvento> GettblEvento(int id)

{

var t = from p in db.tblEventoes where p.fkIdUsuario == id select p;

if (t == null)

{

throw new HttpResponseException(Request.CreateResponse(HttpStatusCode.NotFound));

}

return t;

}

* Metodos:
  + IEnumerable<tblEvento> GettblEventoes (): Método tipo GET devuelve la información de todos los registros en la tabla.
  + IEnumerable<tblEvento> GettblEvento (int id): Método tipo GET devuelve la información de un registro con un id especifico.
  + HttpResponseMessage PosttblEvento (tblEvento tblevento): Método tipo POST ingresa registra datos dentro de la tabla.
* Variables
  + AutoEntities db (Variable que gestiona las transacciones de datos entre los servicios web y la base de datos)

SHA1Util: Clase encargada de generar cadenas encriptadas tipo SHA1.

public static class SHA1Util

{

public static string SHA1HashStringForUTF8String(string s)

{

byte[] bytes = Encoding.UTF8.GetBytes(s);

var sha1 = SHA1.Create();

byte[] hashBytes = sha1.ComputeHash(bytes);

return HexStringFromBytes(hashBytes);

}

public static string HexStringFromBytes(byte[] bytes)

{

var sb = new StringBuilder();

foreach (byte b in bytes)

{

var hex = b.ToString("x2");

sb.Append(hex);

}

return sb.ToString();

}

}

* Métodos:
  + string SHA1HashStringForUTF8String (string) (Método encargado de generar la cadena encriptada)
* Variables: no contiene

TokenController: Clase manejadora de los servicios web para tabla tblToken.

public class TokenController : ApiController

{

private AutoEntities db = new AutoEntities();

// GET api/GettblToken/5

public IEnumerable<tblToken> GettblToken(int id)

{

var t = from p in db.tblTokens where p.fkIdUsuario == id && p.Estado == 1 select p;

if (t == null)

{

throw new HttpResponseException(Request.CreateResponse(HttpStatusCode.NotFound));

}

return t;

}

// POST api/Token/downToken

public HttpResponseMessage downToken(tblToken tbltoken)

{

if (ModelState.IsValid)

{

try

{

var user = db.tblTokens.Find(tbltoken.IdToken);

if (user != null)

{

user.Estado = 0;

using (AutoEntities d = new AutoEntities())

{

d.Entry(user).State = EntityState.Modified;

d.SaveChanges();

}

return Request.CreateResponse<bool>(HttpStatusCode.OK, true);

}

else

{

return Request.CreateResponse<bool>(HttpStatusCode.OK, false);

}

}

catch (DbUpdateConcurrencyException ex)

{

return Request.CreateResponse<bool>(HttpStatusCode.OK, false);

}

}

else

{

return Request.CreateResponse<bool>(HttpStatusCode.OK, false);

}

}

// POST api/Token/IsValid

public HttpResponseMessage IsValid(tblToken tbltoken)

{

if (ModelState.IsValid)

{

try

{

tblToken t = db.tblTokens.FirstOrDefault(tok => tok.Token == tbltoken.Token);

if (t.Estado == 1)

{

tblUsuario u = db.tblUsuarios.FirstOrDefault(use => use.IdUsuario == t.fkIdUsuario);

if(u!=null)

return Request.CreateResponse<bool>(HttpStatusCode.OK, true);

else

return Request.CreateResponse<bool>(HttpStatusCode.OK, false);

}

else

{

return Request.CreateResponse<bool>(HttpStatusCode.OK, false);

}

}

catch (Exception)

{

return Request.CreateResponse<bool>(HttpStatusCode.OK, false);

}

}

else

{

return Request.CreateResponse<bool>(HttpStatusCode.BadRequest, false);

}

}

* Métodos:
  + IEnumerable<tblToken> GettblToken (int id): Método tipo GET encargado de devolver los tokens de un usuario en específico.
  + HttpResponseMessage downToken (tblToken tbltoken): Método tipo POST encargado de eliminar tokens.
  + HttpResponseMessage IsValid(tblToken tbltoken): Método tipo POST encargado de verificar la validez de un token.
* Variables
  + AutoEntities db (Variable que gestiona las transacciones de datos entre los servicios web y la base de datos)

UsuarioController: Clase manejadora de los servicios web para la tabla tblUsuario.

public class UsuarioController : ApiController

{

private AutoEntities db = new AutoEntities();

// GET api/Usuario/GettblUsuarios

public IEnumerable<tblUsuario> GettblUsuarios()

{

return db.tblUsuarios.AsEnumerable();

}

// GET api/Usuario/GettblUsuario/5

public tblUsuario GettblUsuario(int id)

{

tblUsuario tblusuario = db.tblUsuarios.Find(id);

if (tblusuario == null)

{

throw new HttpResponseException(Request.CreateResponse(HttpStatusCode.NotFound));

}

return tblusuario;

}

//POST api/Usuario/UpdateUsuario

public HttpResponseMessage UpdateUsuario(tblUsuario tblusuario)

{

if (ModelState.IsValid)

{

try

{

var user = db.tblUsuarios.Find(tblusuario.IdUsuario);

if (user != null)

{

var claves = tblusuario.Password.Split('\*');

if (claves[0] == user.Password)

{

tblusuario.Password = claves[1];

using(AutoEntities d = new AutoEntities())

{

d.Entry(tblusuario).State = EntityState.Modified;

d.SaveChanges();

}

return Request.CreateResponse<bool>(HttpStatusCode.OK, true);

}

else

{

return Request.CreateResponse<bool>(HttpStatusCode.OK, false);

}

}

else

{

return Request.CreateResponse<bool>(HttpStatusCode.OK, false);

}

}

catch (DbUpdateConcurrencyException ex)

{

return Request.CreateResponse<bool>(HttpStatusCode.OK, false);

}

}

else

{

return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ModelState);

}

}

// POST api/Usuario/PosttblUsuario

public HttpResponseMessage PosttblUsuario(tblUsuario tblusuario)

{

if (ModelState.IsValid)

{

tblUsuario myUser = db.tblUsuarios.SingleOrDefault(user => user.Usuario == tblusuario.Usuario);

if (myUser == null)

{

db.tblUsuarios.Add(tblusuario);

db.SaveChanges();

HttpResponseMessage response = Request.CreateResponse(HttpStatusCode.OK, tblusuario);

return response;

}

else

{

HttpResponseMessage response = Request.CreateResponse(HttpStatusCode.OK, tblusuario);

return response;

}

}

else

{

return Request.CreateErrorResponse(HttpStatusCode.BadRequest, ModelState);

}

}

// POST api/Usuario/Login

public HttpResponseMessage Login(tblUsuario tblusuario)

{

if (ModelState.IsValid)

{

string u = tblusuario.Usuario;

tblUsuario myUser = db.tblUsuarios.SingleOrDefault(user => user.Usuario == u);

if (myUser != null)

{

if (myUser.Password == tblusuario.Password)

{

HttpRequestBase r = ((HttpContextBase)Request.Properties["MS\_HttpContext"]).Request;

tblToken t = new tblToken();

t.Descripcion = "User-Agent:"+r.UserAgent+", OS:"+r.Browser.Platform;

t.Estado = 1;

t.Fecha = DateTime.Now.ToShortDateString();

t.fkIdUsuario = myUser.IdUsuario;

t.Hora = DateTime.Now.ToShortTimeString();

t.Ip = HttpContext.Current.Request.ServerVariables["REMOTE\_ADDR"] ?? "No Identificada";

t.Token = SHA1Util.SHA1HashStringForUTF8String(t.fkIdUsuario+DateTime.Now.ToString());

db.tblTokens.Add(t);

db.SaveChanges();

return Request.CreateResponse<RespuestaLogin>(HttpStatusCode.OK, new RespuestaLogin() { Respuesta = true, Token = t.Token });

}

else

{

return Request.CreateResponse<RespuestaLogin>(HttpStatusCode.OK, new RespuestaLogin() { Respuesta= false, Token = "" });

}

}

else

{

return Request.CreateResponse<RespuestaLogin>(HttpStatusCode.OK, new RespuestaLogin() { Respuesta = false, Token = "" });

}

}

else

{

return Request.CreateResponse<RespuestaLogin>(HttpStatusCode.BadRequest, new RespuestaLogin() { Respuesta = false, Token = "" });

}

}

// POST api/Usuario/UserId

public HttpResponseMessage UserId(tblUsuario tblusuario)

{

if (ModelState.IsValid)

{

string u = tblusuario.Usuario;

tblUsuario myUser = db.tblUsuarios.SingleOrDefault(user => user.Usuario == u);

if (myUser != null)

{

return Request.CreateResponse<RespuestaLogin>(HttpStatusCode.OK, new RespuestaLogin { Respuesta = true, Token = myUser.IdUsuario.ToString() });

}

else

{

return Request.CreateResponse<RespuestaLogin>(HttpStatusCode.Forbidden, new RespuestaLogin() { Respuesta = false, Token = "" });

}

}

else

{

return Request.CreateResponse<RespuestaLogin>(HttpStatusCode.Forbidden, new RespuestaLogin() { Respuesta = false, Token = "" });

}

}

* Métodos:
  + IEnumerable<tblUsuario> GettblUsuarios (): Método tipo GET devuelve un listado de los usuarios registrados.
  + tblUsuario GettblUsuario (int id): Método tipo GET devuelve la información de un Usuario en específico.
  + HttpResponseMessage UpdateUsuario (tblUsuario tblusuario): Método tipo POST registra cambios a un usuario en específico.
  + HttpResponseMessage PosttblUsuario (tblUsuario tblusuario): Método tipo POST encargado de registrar usuarios nuevos.
  + HttpResponseMessage Login (tblUsuario tblusuario) Método tipo POST encargado de validar el acceso al sistema.
  + HttpResponseMessage UserId (tblUsuario tblusuario): Método tipo POST encargado de retornar el identificador de un usuario en específico.

Aplicación Android

SHA1Util: Clase encargada de generar cadenas encriptadas tipo SHA1.

public static class SHA1Util

{

public static string SHA1HashStringForUTF8String(string s)

{

byte[] bytes = Encoding.UTF8.GetBytes(s);

var sha1 = SHA1.Create();

byte[] hashBytes = sha1.ComputeHash(bytes);

return HexStringFromBytes(hashBytes);

}

public static string HexStringFromBytes(byte[] bytes)

{

var sb = new StringBuilder();

foreach (byte b in bytes)

{

var hex = b.ToString("x2");

sb.Append(hex);

}

return sb.ToString();

}

}

* Métodos:
  + String SHA1HashStringForUTF8String (string) (Método encargado de generar la cadena encriptada).
* Variables: no contiene.

Credencials: Clase encargada de generar diccionario de cuenta de usuario.

public static class Credencials

{

public static IDictionary<string, string> Cuenta(string Usuario, string Token, string Mac, string Cam, string IdUser)

{

IDictionary<string, string> retorno = new Dictionary<string, string>();

retorno.Add("User", Usuario);

retorno.Add("Token", Token);

retorno.Add("Mac", Mac);

retorno.Add("Cam", Cam);

retorno.Add("IdUser", IdUser);

return retorno;

}

}

* Métodos:
  + IDictionary<string, string> Cuenta (string Usuario, string Token, string Mac, string Cam, string IdUser) (Método que genera un diccionario de cuenta de usuario).

DataBase: Clase encargada de crear y eliminar la base de datos en el dispositivo.

public static class DataBase

{

public static void Create()

{

try

{

var conn = new SQLiteConnection(FileDir());

conn.CreateTable<tblTokenL>();

conn.CreateTable<tblEventoL>();

conn.CreateTable<DateL>();

}

catch (Exception e) { Console.WriteLine(e.Message); }

}

public static void Delete()

{

try

{

var conn = FileDir();

if (File.Exists(conn))

File.Delete(conn);

}

catch (Exception e) { Console.WriteLine(e.Message); }

}

public static string FileDir()

{

string folder = System.Environment.GetFolderPath(System.Environment.SpecialFolder.Personal);

var conn = System.IO.Path.Combine(folder, "DB.db");

return conn;

}

}

* Métodos
  + void Create() (Método encargado de crear base de datos local en el dispositivo móvil).
  + void Delete() (Método encargado de eliminar la base de datos local)
  + string FileDir() (Método encardado retornar la cadena de conexión de la base de datos).

EventosD: Clase encargada de manejar los datos de la base de datos del dispositivo.

public class EventosD

{

public tblEventoL entity = new tblEventoL();

SQLiteConnection db = new SQLiteConnection(DataBase.FileDir());

RestClient Client = new RestClient(WebServicesInfo.Dir);

public EventosD()

{

entity.Descripcion = string.Empty;

entity.Evento = string.Empty;

entity.Fecha = string.Empty;

entity.fkIdUsuario = 0;

entity.Hora = string.Empty;

entity.IdEvento = 0;

entity.IdEventoS = 0;

entity.Sync = 0;

}

public bool UploadData()

{

try

{

tblEvento entityS = new tblEvento();

entityS.Descripcion = entity.Descripcion;

entityS.Evento = entity.Evento;

entityS.Fecha = entity.Fecha;

entityS.fkIdUsuario = entity.fkIdUsuario;

entityS.Hora = entity.Hora;

entityS.IdEvento = entity.IdEventoS;

bool salida = false;

var request = new RestRequest("api/Evento/PosttblEvento", Method.POST);

request.AddHeader("Content-Type", "application/json; charset=utf-8");

request.AddParameter("text/json", JsonConvert.SerializeObject(entityS), ParameterType.RequestBody);

request.RequestFormat = DataFormat.Json;

var t = Client.Execute(request);

if (string.IsNullOrWhiteSpace(t.Content) || t.StatusCode != System.Net.HttpStatusCode.Created)

{

salida = false;

}

else

{

if (t.Content.ToLower() == "true")

{

salida = true;

}

}

return salida;

}

catch (Exception ex)

{

return false;

}

}

public EventosD(int Id)

{

var n = db.Query<tblEventoL>("SELECT \* FROM tblEventoL WHERE IdEvento = ?", Id).FirstOrDefault();

if (n != null)

entity = n;

db.Close();

}

public bool Create()

{

try

{

db.Insert(entity);

db.Close();

return true;

}

catch (Exception)

{

db.Close();

return false;

}

}

public bool Update()

{

try

{

db.Update(entity);

db.Close();

return true;

}

catch (Exception)

{

db.Close();

return false;

}

}

public bool Delete()

{

try

{

db.Delete(entity);

db.Close();

return true;

}

catch (Exception)

{

db.Close();

return false;

}

}

public static IEnumerable<tblEventoL> arrayList()

{

SQLiteConnection db = new SQLiteConnection(DataBase.FileDir());

IEnumerable<tblEventoL> salida = db.Query<tblEventoL>("select \* from tblEventoL");

db.Close();

return salida;

}

}

* Métodos:
  + EventosD() (Constructor instancia todas las variables globales del método).
  + EventosD(int Id) (Constructor instancia un registro con el id del registro)
  + bool UploadData() (Método encargado de subir registros locales al servidor).
  + bool Create() (Método encargado de crear un registro nuevo local).
  + bool Update() (Método encargado de modificar un registro local).
  + bool Delete() (Método encargado de eliminar un registro local).
  + IEnumerable<tblEventoL> arrayList() (Método que devuelve una lista de todos los registros locales).
* Variables
  + tblEventoL entity (Variable de entidad).
  + SQLiteConnection db (Variable de conexión a base de datos).
  + RestClient Client (Variable de cliente web).

Login: Clase encargada de la conexión para validad los datos de usuario con el servidor web.

public class Login

{

private tblUsuario entity = new tblUsuario();

RestClient Client = new RestClient(WebServicesInfo.Dir);

public Login(string Usuario, string Clave)

{

entity.Usuario = Usuario;

entity.Password = SHA1Util.SHA1HashStringForUTF8String(Clave);

}

public RespuestaLogin getResponse()

{

try

{

RespuestaLogin salida = new RespuestaLogin();

var request = new RestRequest("api/Usuario/Login", Method.POST);

request.AddHeader("Content-Type", "application/json; charset=utf-8");

request.AddParameter("text/json", JsonConvert.SerializeObject(entity), ParameterType.RequestBody);

request.RequestFormat = DataFormat.Json;

var t = Client.Execute(request);

if (string.IsNullOrWhiteSpace(t.Content) || t.StatusCode != System.Net.HttpStatusCode.OK)

{

salida = JsonConvert.DeserializeObject<RespuestaLogin>(t.Content);

}

else

{

if (JsonConvert.DeserializeObject<RespuestaLogin>(t.Content).Respuesta.ToString().ToLower() == "true")

{

salida = JsonConvert.DeserializeObject<RespuestaLogin>(t.Content);

}

}

return salida;

}

catch(Exception ex)

{

return new RespuestaLogin() { Respuesta= false, Token="" };

}

}

public string IdUser(string Name)

{

try

{

RespuestaLogin salida = new RespuestaLogin();

var request = new RestRequest("api/Usuario/UserId", Method.POST);

request.AddHeader("Content-Type", "application/json; charset=utf-8");

request.AddParameter("text/json", JsonConvert.SerializeObject(entity), ParameterType.RequestBody);

request.RequestFormat = DataFormat.Json;

var t = Client.Execute(request);

if (string.IsNullOrWhiteSpace(t.Content) || t.StatusCode != System.Net.HttpStatusCode.OK)

{

salida = JsonConvert.DeserializeObject<RespuestaLogin>(t.Content);

}

else

{

if (JsonConvert.DeserializeObject<RespuestaLogin>(t.Content).Respuesta.ToString().ToLower() == "true")

{

salida = JsonConvert.DeserializeObject<RespuestaLogin>(t.Content);

}

}

return salida.Token;

}

catch (Exception ex)

{

return "";

}

}

public static bool TIsValid(string token)

{

try

{

bool salida = true;

if (token != "None")

{

tblToken tok = new tblToken() { Token = token };

RestClient Client = new RestClient(WebServicesInfo.Dir);

var request = new RestRequest("api/Token/IsValid", Method.POST);

request.AddHeader("Content-Type", "application/json; charset=utf-8");

request.AddParameter("text/json", JsonConvert.SerializeObject(tok), ParameterType.RequestBody);

request.RequestFormat = DataFormat.Json;

var t = Client.Execute(request);

if (string.IsNullOrWhiteSpace(t.Content) || t.StatusCode != System.Net.HttpStatusCode.OK)

{

salida = false;

}

else

{

if (t.Content.ToLower() == "true")

{

salida = true;

}

else

{

salida = false;

}

}

}

else

salida = true;

return salida;

}

catch (Exception ex)

{

return true;

}

}

}

* Métodos:
  + Login(string Usuario, string Clave) (Constructor inicializa las variables globales de la clase).
  + RespuestaLogin getResponse() (Método encargado de validar las credenciales del usuario).
  + string IdUser(string Name) (Método encargado de retornar el id de usuario).
  + bool TIsValid(string token) (Método encargado de validad la sesión del usuario).
* Variables
  + tblUsuario entity (Variable de entidad)
  + RestClient Client (Variable de cliente web)

WebServicesInfo: Clase que retorna información del servidor web.

public static class WebServicesInfo

{

public static string Dir { get { return "http://audomoticar.no-ip.org"; } }

}

* Métodos: No contiene.
* Variables:
  + string Dir (Retorna la dirección web de los servicios).

CountDown: Clase encargada de generar un timer para la pantalla de splash de la aplicación.

class CountDown : CountDownTimer

{

private Activity \_act;

private System.Type \_actLaunch;

public CountDown(long millisInFuture, long countDown, Activity act, System.Type actLaunch) :

base(millisInFuture, countDown)

{

\_act = act;

\_actLaunch = actLaunch;

}

public override void OnFinish()

{

\_act.StartActivity(\_actLaunch);

\_act.Finish();

}

public override void OnTick(long millisUntilFinished)

{

}

}

* Método: No contiene solo los heredados del tipo de clase.
* Variables:
  + Activity \_act (Variable que contiene el contexto en el que corre la aplicacion).
  + System.Type \_actLaunch (Tipo de actividad a correr después de la pantalla de splash).

SplashActivity: Clase de actividad inicial.

public class SplashActivity : Activity

{

public override void OnAttachedToWindow()

{

base.OnAttachedToWindow();

Window window = this.Window;

window.SetFormat(Android.Graphics.Format.Rgb888);

}

protected override void OnCreate(Bundle bundle)

{

base.OnCreate(bundle);

// Create your application here

RequestWindowFeature(WindowFeatures.NoTitle);

this.Window.SetFlags(WindowManagerFlags.Fullscreen, WindowManagerFlags.Fullscreen);

SetContentView(Resource.Layout.Splash);

CountDown \_tik;

\_tik = new CountDown(4000, 1000, this, typeof(LoginActivity));// It delay the screen for 1 second and after that switch to YourNextActivity

\_tik.Start();

StartAnimations();

}

private void StartAnimations()

{

Animation anim = AnimationUtils.LoadAnimation(this, Resource.Animation.alpha);

anim.Reset();

LinearLayout l = FindViewById<LinearLayout>(Resource.Id.lin\_lay);

l.ClearAnimation();

l.StartAnimation(anim);

anim = AnimationUtils.LoadAnimation(this, Resource.Animation.translate);

anim.Reset();

ImageView iv = FindViewById<ImageView>(Resource.Id.logo);

iv.ClearAnimation();

iv.StartAnimation(anim);

}

}

* Métodos:
  + void StartAnimations() (Inicia animación de splash)
* Variables: No contiene.

LoginActivity: Clase de actividad de pantalla de login.

public class LoginActivity : Activity

{

private EditText username = null;

private EditText password = null;

private Button login;

protected override void OnCreate(Bundle bundle)

{

base.OnCreate(bundle);

// Set our view from the "main" layout resource

SetContentView(Resource.Layout.Main);

// Get our button from the layout resource,

// and attach an event to it

username = FindViewById<EditText>(Resource.Id.editText1);

password = FindViewById<EditText>(Resource.Id.editText2);

login = FindViewById<Button>(Resource.Id.button1);

login.Click += login\_Click;

}

protected override void OnStart()

{

base.OnStart();

IEnumerable<Account> accounts = AccountStore.Create(this).FindAccountsForService("Arduino");

foreach (var n in accounts)

{

StartActivity(typeof(ListaConectar));

Finish();

break;

}

}

void login\_Click(object sender, EventArgs e)

{

if (username.Text == "admin" && password.Text == "12345")

{

DataBase.Create();

Account ac = new Account(username.Text, Credencials.Cuenta(username.Text, "None", string.Empty, string.Empty, "0"));

AccountStore.Create(this).Save(ac, "Arduino");

RunOnUiThread(() =>

{

Toast.MakeText(this, "Bienvenido", ToastLength.Short).Show();

StartActivity(typeof(ListaConectar));

Finish();

});

}

else{

if (username.Text.Trim() != "" && password.Text.Trim() != "")

{

RunOnUiThread(() => {

AndHUD.Shared.Show(this, "Ingresando");

});

Task.Factory.StartNew(() =>{

Login l = new Login(username.Text, password.Text);

RespuestaLogin r = l.getResponse();

if (r.Respuesta)

{

string id = l.IdUser(username.Text);

DataBase.Create();

Account ac = new Account(username.Text, Credencials.Cuenta(username.Text, r.Token, string.Empty, string.Empty, id));

AccountStore.Create(this).Save(ac, "Arduino");

RunOnUiThread(() =>

{

AndHUD.Shared.Dismiss(this);

Toast.MakeText(this, "Bienvenido", ToastLength.Short).Show();

StartActivity(typeof(ListaConectar));

Finish();

});

}

else

{

RunOnUiThread(() =>

{

AndHUD.Shared.Dismiss(this);

Toast.MakeText(this, "Usuario o Clave Incorrectos", ToastLength.Short).Show();

});

}

});

}

else

{

RunOnUiThread(() => {

Toast.MakeText(this, "Campos Vacios", ToastLength.Short).Show();

});

}

}

}

}

* Metodos:
  + void login\_Click(object sender, EventArgs e) (Método encargado de verificar si existe alguna sesión activa o iniciar la sesión con los datos proporcionados por el usuario).
* Variables:
  + EditText username (Variable de campo Usuario).
  + EditText password (Variable de campo clave).

ListaConectar: Clase encargada de mostrar los dispositivos bluetooth cercanos a conectar.

public class ListaConectar : Activity

{

private static List<BtDevice> ListaDispositivos = new List<BtDevice>();

// Debugging

private const string TAG = "DeviceListActivity";

private const bool Debug = true;

// Return Intent extra

public const string EXTRA\_DEVICE\_ADDRESS = "device\_address";

// Member fields

private BluetoothAdapter btAdapter;

private static ArrayAdapter<string> pairedDevicesArrayAdapter;

private Receiver receiver;

protected override void OnStart()

{

base.OnStart();

try

{

IEnumerable<Account> accounts = AccountStore.Create(this).FindAccountsForService("Arduino");

if (accounts.ToList().Count == 0)

{

StartActivity(typeof(LoginActivity));

Finish();

}

foreach (var n in accounts)

{

if (n.Properties["Mac"] != string.Empty)

{

StartActivity(typeof(Principall));

Finish();

break;

}

}

ListaDispositivos.Clear();

btAdapter = BluetoothAdapter.DefaultAdapter;

if (!btAdapter.IsEnabled)

{

btAdapter.Enable();

}

var pairedDevices = btAdapter.BondedDevices;

if (pairedDevices.Count > 0)

{

foreach (var device in pairedDevices)

{

BtDevice dis = new BtDevice();

dis.Nombre = device.Name;

dis.Mac = device.Address;

ListaDispositivos.Add(dis);

pairedDevicesArrayAdapter.Add(device.Name + "\n" + device.Address);

}

}

else

{

}

DoDiscovery();

}

catch (Exception)

{

}

}

void DeviceListClick(object sender, AdapterView.ItemClickEventArgs e)

{

btAdapter.CancelDiscovery();

IEnumerable<Account> accounts = AccountStore.Create(this).FindAccountsForService("Arduino");

foreach (var n in accounts)

{

Account ac = new Account(n.Username, Credencials.Cuenta(n.Username, n.Properties["Token"], ListaConectar.ListaDispositivos[e.Position].Mac, n.Properties["Cam"], n.Properties["IdUser"]));

AccountStore.Create(this).Save(ac, "Arduino");

StartActivity(typeof(Principall));

Finish();

break;

}

}

protected override void OnCreate(Bundle bundle)

{

base.OnCreate(bundle);

RequestWindowFeature(WindowFeatures.IndeterminateProgress);

SetContentView(Resource.Layout.Encontrar);

SetResult(Result.Canceled);

pairedDevicesArrayAdapter = new ArrayAdapter<string>(this, Android.Resource.Layout.SimpleListItem1);

var pairedListView = FindViewById<ListView>(Resource.Id.listadescubrir);

pairedListView.Adapter = pairedDevicesArrayAdapter;

pairedListView.ItemClick += DeviceListClick;

receiver = new Receiver(this);

var filter = new IntentFilter(BluetoothDevice.ActionFound);

RegisterReceiver(receiver, filter);

filter = new IntentFilter(BluetoothAdapter.ActionDiscoveryFinished);

RegisterReceiver(receiver, filter);

}

protected override void OnDestroy()

{

base.OnDestroy();

if (btAdapter != null)

{

btAdapter.CancelDiscovery();

}

UnregisterReceiver(receiver);

}

private void DoDiscovery()

{

if (Debug)

Console.WriteLine("doDiscovery()");

SetProgressBarIndeterminateVisibility(true);

if (btAdapter.IsDiscovering)

{

btAdapter.CancelDiscovery();

}

btAdapter.StartDiscovery();

}

public class Receiver : BroadcastReceiver

{

Activity \_clase;

public Receiver(Activity clase)

{

\_clase = clase;

}

public override void OnReceive(Context context, Intent intent)

{

string action = intent.Action;

if (action == BluetoothDevice.ActionFound)

{

BluetoothDevice device = (BluetoothDevice)intent.GetParcelableExtra(BluetoothDevice.ExtraDevice);

if (device.BondState != Bond.Bonded)

{

BtDevice dis = new BtDevice();

dis.Nombre = device.Name;

dis.Mac = device.Address;

ListaConectar.ListaDispositivos.Add(dis);

pairedDevicesArrayAdapter.Add(device.Name + "\n" + device.Address);

}

}

else if (action == BluetoothAdapter.ActionDiscoveryFinished)

{

\_clase.SetProgressBarIndeterminateVisibility(false);

}

}

}

}

* Métodos
  + void DeviceListClick(object sender, AdapterView.ItemClickEventArgs e) (Método en el cual se inicializa y se selecciona el bluetooth por el usuario).
  + void DoDiscovery() (Método mediante el cual la aplicación incia el descubrimiento de dispositivos bluetooth cercanos).
* Variables
  + List<BtDevice> ListaDispositivos
  + const string EXTRA\_DEVICE\_ADDRESS
  + BluetoothAdapter btAdapter
  + static ArrayAdapter<string> pairedDevicesArrayAdapter
  + Receiver receiver

Receiver: Clase que en background verifica la existencia de dispositivos bluetooth.

* Metodos: No contiene solo los heredados por el tipo de clase.
* Variables
  + Activity \_clase (Variable encargada de saber cuál el contexto principal).

BtDevice: Clase auxiliar para almacenaje de la información de dispositivos.

* Métodos: No contiene.
* Variables:
  + string Nombre (Variable que almacena el nombre público del dispositivo bluetooth).
  + string Mac (variable que almacena la dirección mac del dispositivo).

Principal**:** Clase principal de mando de micro robot.

* Métodos:
  + void TrySaveData(string Descripcion, string Evento, int IdUsuario) (Método en el que se intenta guardar un evento en el servidor web en caso contrario lo guarda localmente).
  + void SaveAllOnCloud() (Método que guarda todos los registros locales en el servidor web).
  + void CheckBt() (Método en el cual se verifica la disponibilidad del dispositivo bluetooth).
  + async void Connect() (Método de conexión a micro robot mediante bluetooth).
  + void writeData(Java.Lang.String data) (Método mediante el cual se manda información al microrobot).
  + void beginListenForData() (Método que genera un hilo por el cual se espera la lectura de datos del micro robot).
  + void setImageError() (Método mediante el cual se pone una imagen de error si el streaming de video falla).
  + void taskTask(string url) (Método que genera una tarea asíncrona de streaming de video).
  + bool IsConectad() (Método que retorna el estado de la conexión de red del dispositivo).
  + Dialog createDialog() (Método que genera la venta de dialogo de introducción de URL de video).
* Variables
  + ToggleButton OnOff (Variable de botón de conexión de bluetooth).
  + Java.Lang.String dataToSend (Cadena para envio de datos al bluetooth).
  + BluetoothAdapter mBluetoothAdapter (Variable de adaptador bluetooth).
  + BluetoothSocket btSocket (Variable de socket bluetooth).
  + Stream outStream (Variable de stream de salida).
  + static UUID MY\_UUID (Variable de identificador único de conexión).
  + Stream inStream (Variable de stream de entrada).
  + Button btnAdelante (Variable de botón de avance hacia adelante).
  + Button btnApagado (Variable de botón de apagado de micro robot).
  + Button btnAtras (Variable de botón de retroceso del micro robot).
  + Button btnGDerecha (Variable de botón de giro hacia la derecha).
  + Button btnGIzquierda (Variable de botón de giro hacia la izquierda).
  + MjpegView mv (Variable de stream de video de cámara web).
  + String URL (Variable de cadena url de stream de video).
  + int width (variable de ancho de streaming de video).
  + int height (variable de alto de streaming de video).
  + int IdUSER (Variable de id de usuario).
  + bool suspending (Variable de suspensión de streaming de video).
  + static int TEXT\_ID (Variable de valor único para campo de URL de streaming de video).

## 2.1.7. Grabar las rutinas en el hardware.

Una vez terminado el circuito, el software necesario y los servicios web ya levantados se procede a grabar las rutinas a la placa Arduino, de esta se podrá pasar a la fase de pruebas.

## 2.1.8 Pruebas

Estas fueron realizadas en un periodo de una semana por los miembros del equipo para de esta manera esta seguros de que todos las funciones programadas al micro robot funcionan perfectamente, después de esto se decidió que algunas personas ajenas probaran el micro robot, para así tener una perspectiva diferente y poder obtener un panorama más amplio de las mejoras que se podrían implementar.

## 2.1.9 Implementación.

Por último, después de haber realizado las pruebas necesarias se procede a la implementación del micro robot, el cual ya está listo para ser utilizado en campo con toda confianza.

# CAPITULO III: LOGROS Y RESULTADOS OBTENIDOS

Mejor Calidad de vida

En base a los resultados obtenidos podemos darnos cuenta que con la aplicación de las nuevas tecnologías puedes desarrollar proyectos que sean de gran utilidad para personas que requieran realizar tareas específicas, este es un ejemplo de un buen uso dichas tecnologías combinadas con el conocimiento adquirido.

Este micro robot puede ser de gran ayuda para los equipos de rescate en tareas como:

* Monitoreo de áreas de difícil acceso.
* Monitoreo de áreas peligrosas.
* Monitoreo de la calidad del ambiente.

Mayor autoestima

Durante el proceso de la realización de nuestro proyecto, se obtuvieron los resultados de manera muy favorable lo cual causo una gran satisfacción en nuestra persona. Sin embargo aún nos queda mucho camino por recorrer, probablemente para para poder llegar a nuestra meta y realizar nuestros objetivos es necesario seguir preparándonos y aprender de nuestros errores y fracasos, así poder triunfar en la vida.

Desarrollo profesional

En cuanto a desarrollo profesional nosotros sentimos un gran avance ya que pudimos darnos cuenta que aplicando de manera correcta cada uno de los factores a nuestro favor, se pude dar solución a cualquier problema que se presente durante el transcurso de nuestra vida profesional. Y cuando es necesario pedir ayuda a los demás de esta manera al trabajar en forma colaborativa se complementan las ideas de manera más clara y concisa para llegar a un mismo fin.

Desarrollo intelectual

Al pasar el tiempo fuimos adquiriendo nuevos conocimiento de diferentes áreas no solamente sobre IT, sino también sobre electrónica la cual fue de gran utilidad para llegar a nuestro objetivo, así como la implementación de herramientas para el desarrollo de la aplicación. En conclusión se consiguió un gran conjunto de nuevos conocimiento.



# CAPITULO IV: MANUAL DE OPERACIONES

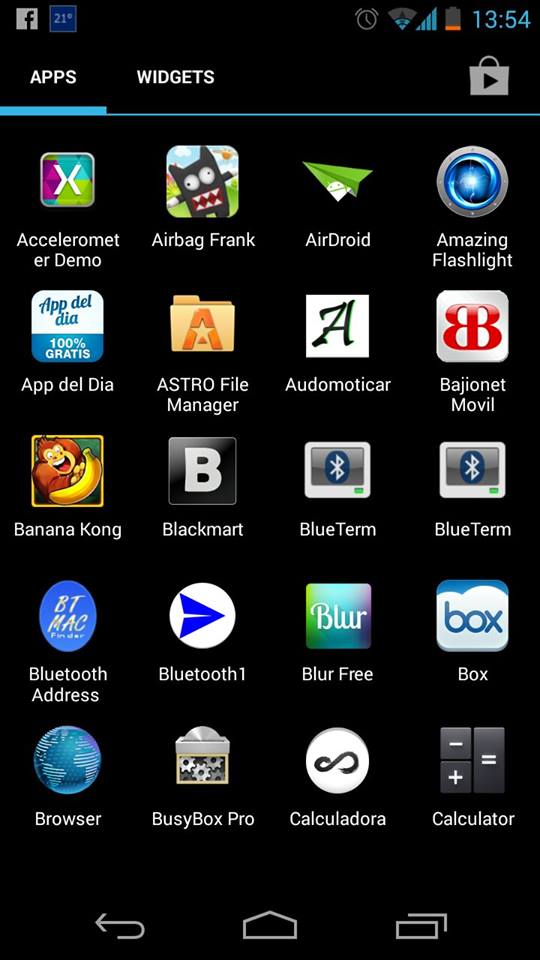


Figura 4.1: Búsqueda de la aplicación.

Al instalar la Aplicación de Audomoticar se prosigue a identificar donde se instaló para iniciarla.



Figura 4.2: Inicio de la aplicación, presentación del logotipo.

Una vez que iniciada, aparece el logotipo de la aplicación como se observa en la figura anterior (Figura 4.2) este proceso lleva acabo por unos momentos antes de mostrar la siguiente pantalla.

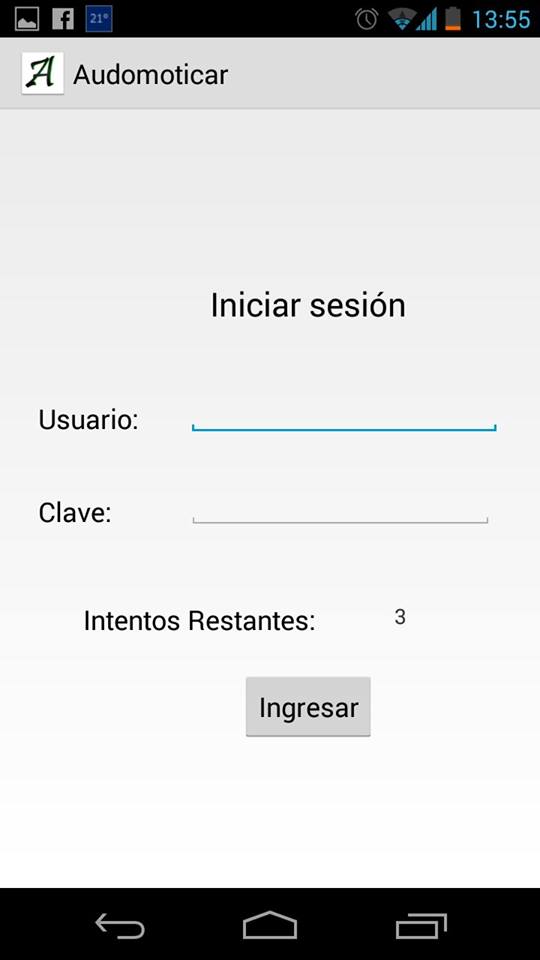


Figura 4.3: pantalla de Login de la aplicación.

Login: Una vez que se ha iniciado la aplicación y finalizado la vista del logotipo de inicio aparece una pantalla donde se solicita ingresar el usuario y la contraseña para poder tener acceso a la misma. Es importante mencionar que cuentas con tres intentos de acceso, los usuarios deben estar registrados en la base de datos para poder iniciar sesión.



Figura 4.4: Ejemplo de Login

En la imagen anterior (figura 4.4) se muestra como debes de ingresar los datos que se te piden. Una vez hecho esto solo de clic en el botón de ingresar. Si tu usuario y tu contraseña son correctos te iniciara una sesión, en caso contrario se te pedirá de nuevo que ingreses tus datos.

Figura 4.5: Visualización de dispositivos disponibles.

Selección de bluetooth: Una vez que ingreses del login, a continuación inicia el proceso de búsqueda de dispositivos disponibles para la conexión, una vez visualizado el dispositivo continuamos a seleccionarlo (bluetooth), con tan solo dar un clic en el nombre del dispositivo bluetooth. Para acceder al dispositivo es necesario hacer una conexión inalámbrica.

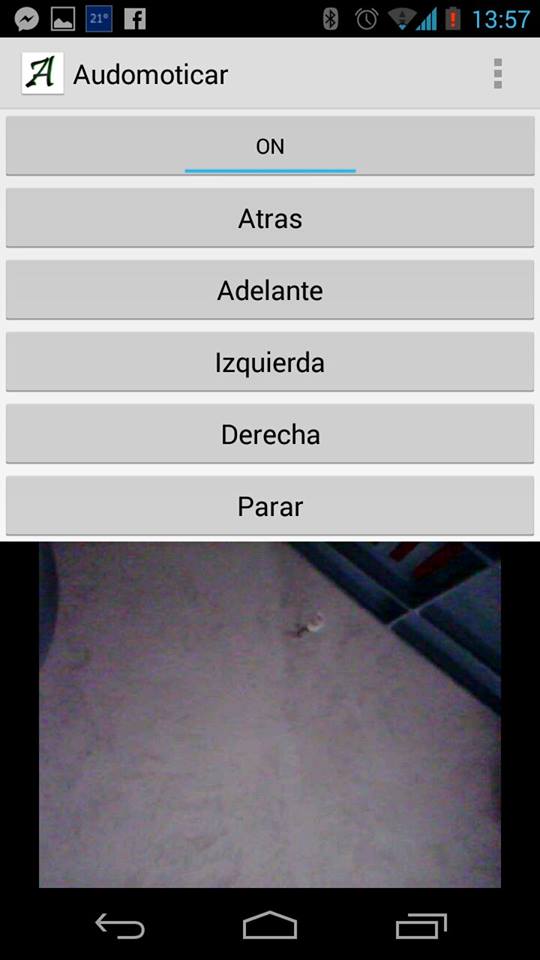


Figura4.6: Vista general de los controles

En la siguiente imagen (figura 4.6) se presenta con una vista previa de las opciones de la aplicación, en donde se encuentran los botones de mando para el control del microrobot los cuales son:

Adelante

Atrás

Izquierda

Derecha

Parar

Y la opción de poder deshabilitar la conexión el dispositivo bluetooth.

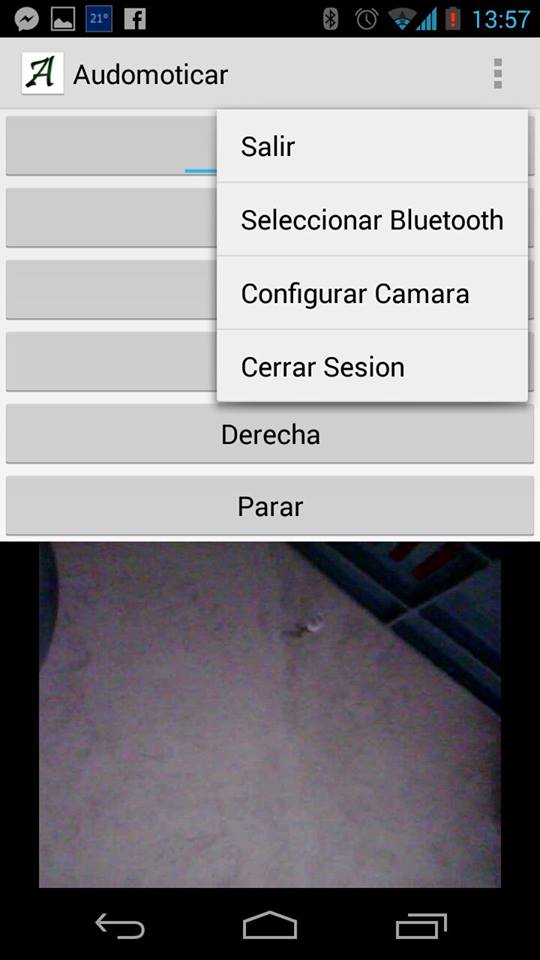


Figura 4.7: Panel de configuraciones opcionales.

Menú de opciones de la aplicación:

Para acceder a los elementos de configuración únicamente presione el botón que se encuentra en la parte superior derecha.

Salir: permite al usuario cerrar la aplicación.

Seleccionar bluetooth: ayuda a vincular el dispositivo con un nuevo bluetooth.

Configuración de la cámara: es opción de ingresar la dirección de la cámara.

Cerrar sesión: Esta opción permite al usuario cerrar su sesión y abrir la aplicación con un nuevo usuario.



Figura 4.8: Ingreso de dirección de acceso a la cámara.

Configuración de la cámara: permite visualizar la imagen que captura la cámara, en la pantalla del dispositivo móvil, para esto tendrá que ingresar la dirección de referencia de la cámara, de este modo tener acceso a la misma. De lo contrario el solo se muestra una pantalla en color negro.

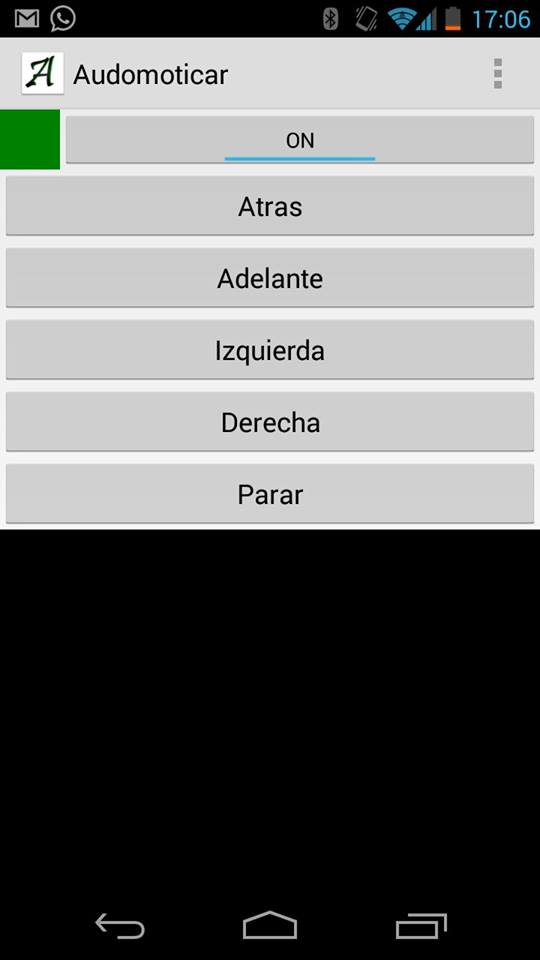
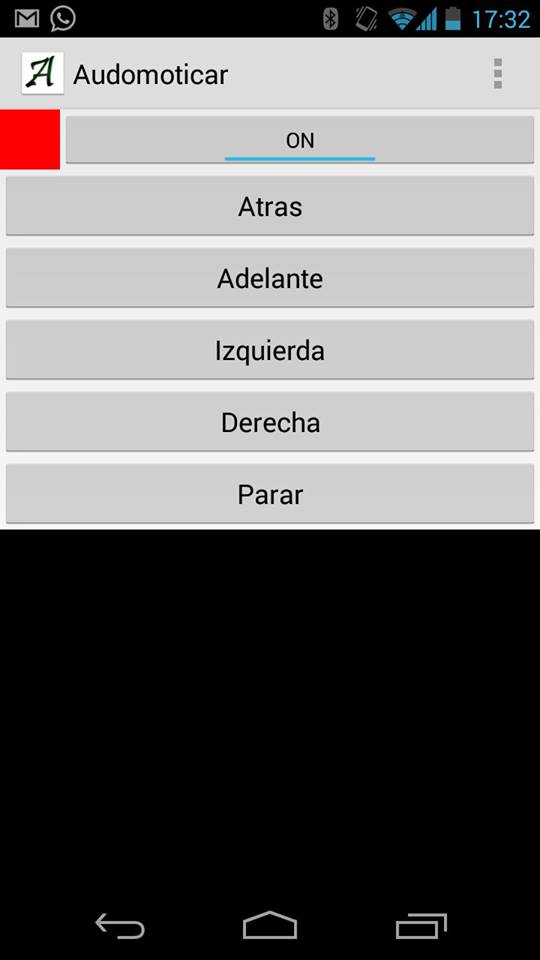


Figura 4:9: sensor de gas Figura 4:10: Sensor de gas 2

Sensor de gas: el programa cuenta con un indicador de presencia de gas Metano se puede observar en el lado superior derecho (pequeño cuadro de color verde), al momento que el micro-robot detecta gas automáticamente este cambia el color la pantalla a rojo.



Figura 4.11: Inicio de la página web

La aplicación web está diseñada para registrar los usuarios y visualizar los movimientos que se realizarán con el robot.



Figura 4.12: Login

Al inicio de página web se encuentran dos casillas donde se pedirá que ingreses un usuario y una contraseña para acceder al sitio web. Una vez agregados los datos, tan solo hay que dar clic en el botón (Ingresar).



Figura 4.13: Cargado la pagina

En la imagen anterior se puede observar el momento en que se carga la página web.

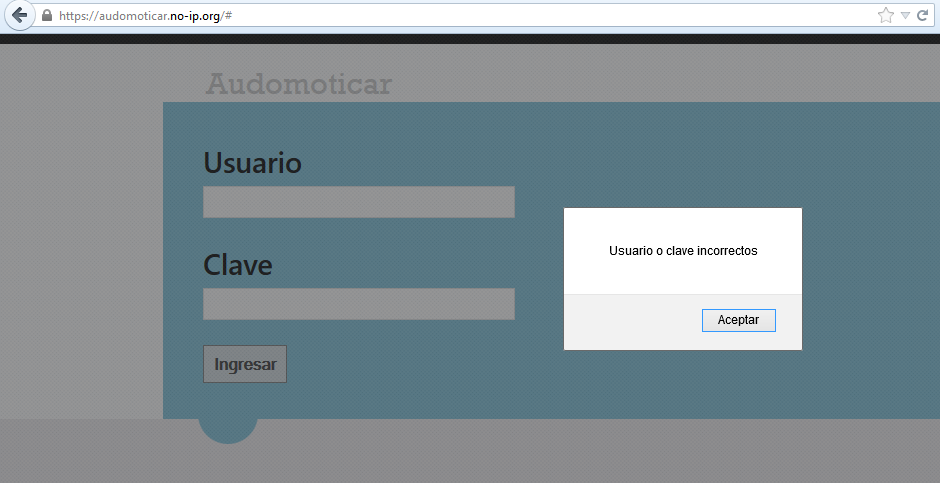


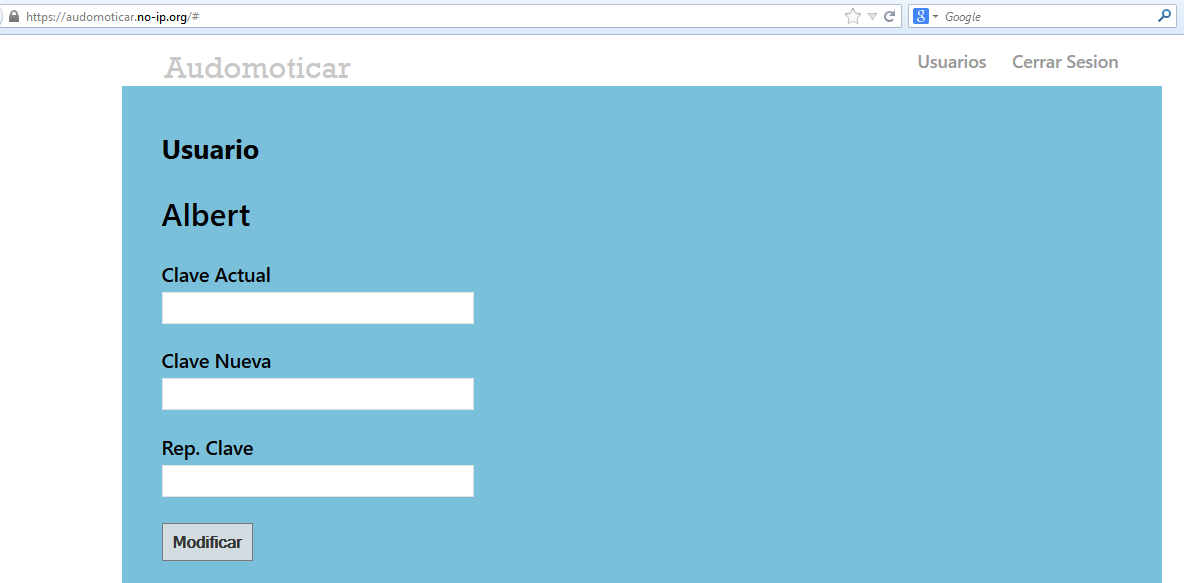
Figura 4.14: Error de Login.

Si se ingresa de manera incorrecta su nombre de usuario o la contraseña aparecerá una pantalla como la imagen anterior (Figura 4.14), donde mostrara el error. Asegúrese de ingresar de manera correcta los datos.



Figura 4.15: Inicio de sesión

Una vez ingresado el usuario y la contraseña de manera correcta aparecerá la siguiente ventana (figuro 4.15), donde se realiza el registro de usuarios, además que se pueden visualizar los usuarios activos.

Figura 4.16: Usuarios

Ya iniciada la sesión el usuario puede revisar los registros que realizo con el microrobot a su vez tiene la opción de modificar su contraseña como se muestra en la imagen (Figura 4.16).

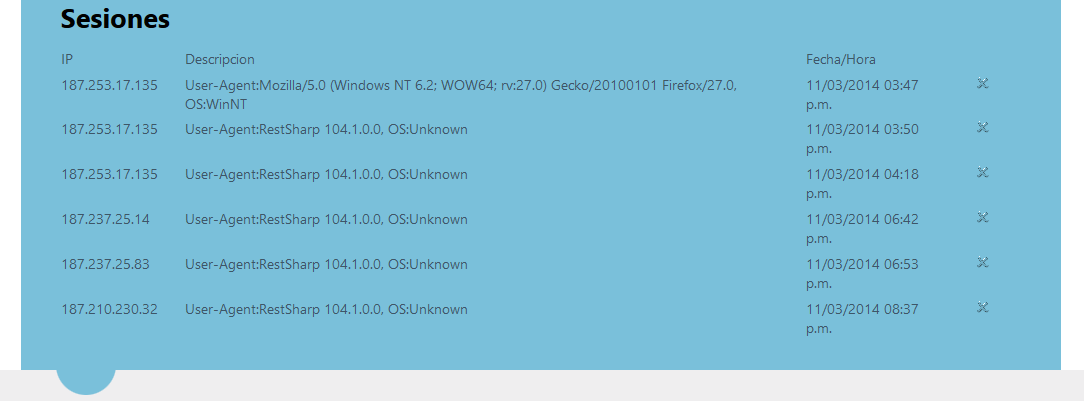


Figura 4.17: Visualización de sesiones.

Cada vez que inicie sesión se guarda un registro donde se registra el navegador, el sistema y la IP desde donde se realiza la conexión. A demás de la hora y fecha exactas. Pero el usuario puede eliminar estos registros solo con hacer clic en la “x” que se encuentra al lado derecha del registro.

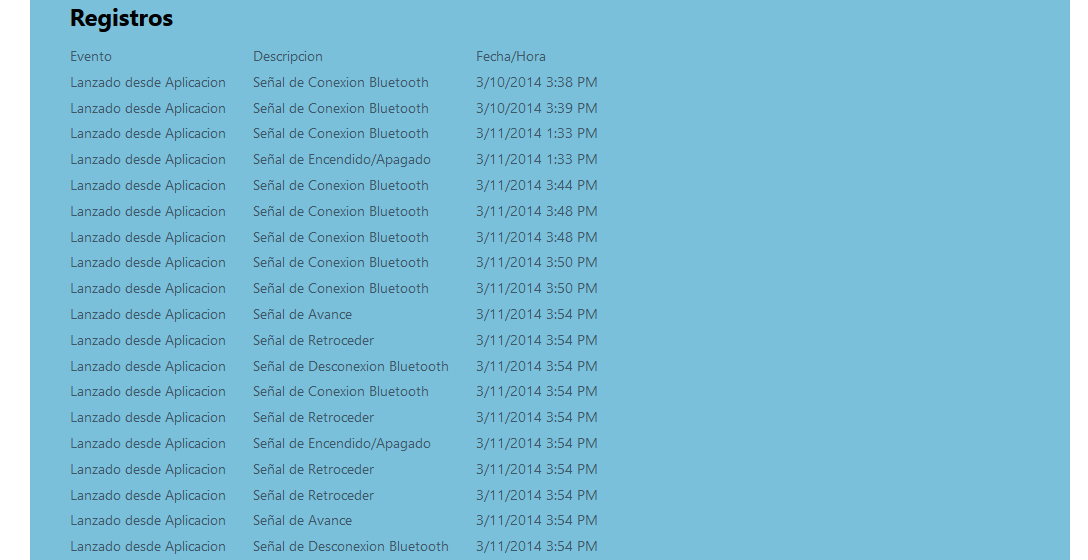


Figura 4.18: Visualización de Registros.

En la página web existe un apartado con el nombre de Registros donde se muestra cada uno de los eventos que se realiza con el microrobot a través de la aplicación del dispositivo móvil, las cuales consta con un evento, una descripción, fecha y hora del movimiento como son:

* Señal de conexión de Bluetooth.
* Señal de encendido/apagado.
* Señal de desconexión de Bluetooth.
* Señal de avance.
* Señal de retroceder.
* Señal de giro a la izquierda.
* Señal de giro a la derecha.

# CAPÍTULO V: ANEXOS

## Glosario de términos

**A**

**API:** Del inglés *Application Programming Interface* es el conjunto de funciones y procedimientos (o métodos, en la programación orientada a objetos) que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción. Son usadas generalmente en las bibliotecas.

**Arduino:** Arduino es una herramienta para hacer que los ordenadores puedan sentir y controlar el mundo físico a través de tu ordenador personal. Es una plataforma de desarrollo de computación física (physical computing) de código abierto, basada en una placa con un sencillo microcontrolador y un entorno de desarrollo para crear software (programas) para la placa.

**B**

**Bluetooth:** es una especificación industrial para Redes Inalámbricas de Área Personal (WPAN) que posibilita la transmisión de voz y datos entre diferentes dispositivos mediante un enlace por radiofrecuencia en la banda ISM de los 2,4 GHz.

**C**

**Código fuente:** El código fuente de un programa informático es un conjunto de líneas de texto que son las instrucciones que debe seguir la computadora para ejecutar dicho programa.

**E**

**Encapsulamiento:** se denomina encapsulamiento al ocultamiento del estado, es decir, del dato miembro de un objeto de manera que sólo se pueda cambiar mediante las operaciones definidas para ese objeto.

**H**

**Hardware:** Conjunto de elementos físicos o materiales que constituyen una computadora o un sistema informático.

**I**

**Inteligencia Artificial:** es la rama de las ciencias de la computación que se ocupa de construir sistemas que permitan exhibir un comportamiento cada vez más inteligente.

**Interface:** En informática se utiliza para nombrar a la conexión física y funcional entre dos sistemas o dispositivos de cualquier tipo dando una comunicación entre distintos niveles. Su plural es interfaces.

Además, la palabra interfaz se utiliza en distintos contextos:

1. **Interfaz como instrumento:** desde esta perspectiva la interfaz es una "prótesis" o "extensión" (McLuhan) de nuestro cuerpo. El ratón es un instrumento que extiende las funciones de nuestra mano y las lleva a la pantalla bajo forma de cursor. Así, por ejemplo, la pantalla de una computadora es una interfaz entre el usuario y el disco duro de la misma.
2. **Interfaz como superficie:** algunos consideran que la interfaz nos trasmite instrucciones ("affordances") que nos informan sobre su uso. La superficie de un objeto (real o virtual) nos habla por medio de sus formas, texturas, colores, etc.
3. **Interfaz como espacio:** desde esta perspectiva la interfaz es el lugar de la interacción, el espacio donde se desarrollan los intercambios y sus manualidades.

**Ip:** es una etiqueta numérica que identifica, de manera lógica y jerárquica, a una interfaz (elemento de comunicación/conexión) de un dispositivo (habitualmente una computadora) dentro de una red que utilice el protocolo IP (*Internet Protocol*).

**P**

**Processing:** Es un lenguaje de programación, entorno de desarrollo, y la comunidad en línea.

**Prototipo:** es la primera versión o modelo de su producto, en que ha incorporado algunas características del producto final. Se crean con rapidez y a un costo bajo para explorar la factibilidad del concepto preliminar.

**Programación**: es el proceso de diseñar, codificar, depurar y mantener el código fuente de programas computacionales. El propósito de la programación es crear programas que exhiban un comportamiento deseado.

**R**

**Router:** es un dispositivo que proporciona conectividad a nivel de red o nivel tres en el modelo OSI. Su función principal consiste en enviar o encaminar paquetes de datos de una red a otra, es decir, interconectar subredes, entendiendo por subred un conjunto de máquinas IP que se pueden comunicar sin la intervención de un encaminador

**S**

**Sistema:** Conjunto ordenado de normas y procedimientos que regulan el funcionamiento de un grupo o colectividad.

**Sketch:** Un sketch es el nombre que usa Arduino para un programa. Es la unidad de código que se sube y ejecuta en la placa Arduino.

**Software:** Conjunto de programas y rutinas que permiten a la computadora realizar determinadas tareas.

**Sensor**: Un sensor es un dispositivo capaz de detectar magnitudes físicas o químicas, llamadas variables de instrumentación, y transformarlas en variables eléctricas. Las variables de instrumentación pueden ser por ejemplo: temperatura, intensidad lumínica, distancia, aceleración, inclinación, desplazamiento, presión, fuerza, torsión, humedad, movimiento, pH, gas, etc.

**W**

**Wiring:** Es un marco de programación de código abierto para los microcontroladores.

## Referencias bibliográficas y virtuales

Revista: IEEE Spectrum

Artículo: The Making of Arduino

Autor del artículo: David Kushner 26 octubre del 2011

<http://www.arduino.cc/>

<http://botscience.wordpress.com/2012/06/05/historia-de-arduino-y-su-nacimiento/>

<http://www.store.arduino.cc>

<http://www.processing.org/>

<http://wiring.org.co/>

<http://developer.android.com/guide/topics/connectivity/bluetooth.html>

<http://searchhealthit.techtarget.com/definition/Kinect>

<http://www.microsoft.com/es-xl/sqlserver/default.aspx>

<http://www.visualstudio.com/>

[http://www.itvillahermosa.edu.mx/](http://www.itvillahermosa.edu.mx/docs/oferta/ingtic/temario2010/4semestre/CircuitosElectricosyElectronicos.pdf)

<http://eleconomista.com.mx/tecnociencia/2012/05/04/robotica-mexico-pocos-recursos-desarrolladores>

<http://www.snit.mx/ciencia-y-tecnologia/alumnos-del-it-de-nuevo-leon-crean-robot-para-rescate-de-victimas-de-desastres>

<http://mjpeg.sourceforge.net/>

<http://energicentro.blogspot.mx/2007/11/bateras-en-paralelo-conexionado-de.html>

<http://articulo.mercadolibre.com.mx/MLM-440854485-chassis-carro-robot-4wd-smart-car-arduino-pic-atmel-_JM>

<http://www.circuitoselectronicos.org/>

<http://www.dfrobot.com/wiki/index.php/Analog_Gas_Sensor(MQ4)_(SKU:SEN0129)>