UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE CUENCA

CARRERA DE COMPUTACIÓN

DESARROLLO DE UN INCLINÓMETRO PARA VEHÍCULOS TODO TERRENO CON SISTEMA DE ENTRETENIMIENTO BASADO EN ANDROID

Trabajo de titulación previo a la obtención del título de Ingeniero en Ciencias de la Computación

AUTORES: JOSÉ GUILLERMO QUINDE DELGADO

TONNY ROGER LEMA JARAMILLO

TUTOR: ING. MARCELO ESTEBAN FLORES VÁZQUEZ, MST.

Cuenca - Ecuador

2024

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Nosotros, José Guillermo Quinde Delgado con documento de identificación N° 0301681748 y Tonny Roger Lema Jaramillo con documento de identificación N° 0105723605; manifestamos que:

Somos los autores y responsables del presente trabajo; y, autorizamos a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Cuenca, 29 de enero del 2024

Atentamente,

José Guillermo Quinde Delgado
0301681748

Tonny Roger Lema Jaramillo
0105723605

CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

Nosotros, José Guillermo Quinde Delgado con documento de identificación N° 0301681748 y Tonny Roger Lema Jaramillo con documento de identificación N° 0105723605 expresamos nuestra voluntad y por medio del presente documento cedemos a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que somos autores del Proyecto técnico: "Desarrollo de un inclinómetro para vehículos todo terreno con sistema de entretenimiento basado en Android", el cual ha sido desarrollado para optar por el título de Ingeniero en Ciencias de la Computación, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribimos este documento en el momento que hacemos la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Cuenca, 29 de enero del 2024

Atentamente,

José Guillermo Quinde Delgado
0301681748

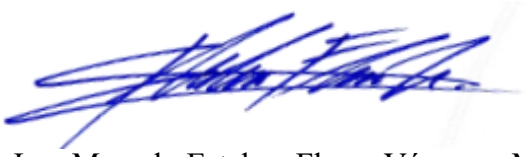
Tonny Roger Lema Jaramillo
0105723605

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Marcelo Esteban Flores Vázquez con documento de identificación N° 0102408978, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: DESARROLLO DE UN INCLINÓMETRO PARA VEHÍCULOS TODO TERRENO CON SISTEMA DE ENTRETENIMIENTO BASADO EN ANDROID, realizado por José Guillermo Quinde Delgado con documento de identificación N° 0301681748 y por Tonny Roger Lema Jaramillo con documento de identificación N° 0105723605 obteniendo como resultado final el trabajo de titulación bajo la opción Proyecto técnico que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Cuenca, 29 de enero del 2024

Atentamente,



Ing. Marcelo Esteban Flores Vázquez, Mst.
0102408978

AGRADECIMIENTOS

Expreso mi profundo agradecimiento a Dios por su constante presencia en la vida de mi familia, brindándonos su protección y amor.

A mi querida madre, le agradezco sinceramente por los cuidados y las enseñanzas que me ha brindado desde mi infancia, así como por ese amor incondicional que ha sido mi motor para nunca rendirme.

A mi padre, le estoy agradecido por el constante apoyo que me ha ofrecido a lo largo de mi vida, siendo un pilar fundamental en mi desarrollo.

A mis hermanos, les doy las gracias por la motivación constante que me han proporcionado, inspirándome a mantenerme firme y no rendirme ante los desafíos.

A mis queridos tíos, agradezco sinceramente el tiempo que han compartido conmigo y sus palabras de aliento, las cuales me han instado a siempre mantener en alto el nombre de mis padres.

Quiero expresar mi gratitud a los diversos profesores que han compartido sus conocimientos, tiempo y apoyo a lo largo de mi trayectoria académica, con especial reconocimiento al Ing. Marcelo Esteban Flores Vázquez, cuya orientación fue fundamental para la realización de este trabajo de investigación.

Tonny Roger Lema Jaramillo

DEDICATORIAS

Dedico con profundo cariño este proyecto de titulación a Dios, mi guía constante y fuente de fortaleza en cada paso de mi vida.

A mis abuelos y tíos, y de manera especial a todas mis tías, les dedico este logro. Siempre estuvieron presentes cuando los necesitaba, creyeron en mí y me enseñaron que los sueños se hacen realidad, incluso en medio de las dificultades.

A mis queridos padres, María Jaramillo y Patricio Lema, les dedico este proyecto con todo mi agradecimiento. Su esfuerzo, tiempo y amor han sido fundamentales para que hoy cumpla una meta más en mi vida. Agradezco por las lecciones de respeto, esfuerzo y dedicación que me han transmitido, así como por su arduo sacrificio para que pueda alcanzar este nuevo logro.

A mis adoradas hermanas, María José, Gabriela Valentina y Evelin Patricia, les dedico este proyecto con gratitud por su apoyo y cariño incondicional. Son mi impulso para seguir adelante, y quiero transmitirles la convicción de que los sueños se pueden cumplir a pesar de los obstáculos. Son el pilar fundamental que impulsa mis sueños y les animo a alcanzar los suyos.

Tonny Roger Lema Jaramillo

AGRADECIMIENTOS

Agradezco de corazón a toda mi familia, especialmente a mis padres, por su constante apoyo, consejo y cariño que me han brindado a lo largo de esta significativa etapa de mi vida. Gracias a ello, he logrado culminar este proceso que, sin duda, representa una parte fundamental de mi desarrollo.

Quiero expresar mi profundo agradecimiento a todos mis profesores, quienes han contribuido con valiosos conocimientos durante este trayecto. Cada uno de ellos ha aportado su grano de arena, enseñándome aspectos fundamentales para convertirme en un profesional destacado. En especial, agradezco a mi tutor, el Ing. Marcelo Esteban Flores Vázquez, cuyo conocimiento y apoyo en estos últimos meses fueron clave para el éxito de nuestro proyecto.

También quiero extender mi gratitud a mi compañero de tesis, Tonny Roger Lema, ya que juntos, enfrentamos diversos obstáculos a lo largo del camino, pero gracias a nuestra dedicación y trabajo en equipo, logramos superarlos y alcanzar este punto de culminación de manera impecable.

José Guillermo Quinde Delgado

DEDICATORIAS

Esta tesis está dedicada a mis padres, Ignacio Guillermo Quinde y Zoila Eusthela Delgado. Agradezco profundamente su constante apoyo, el cual ha sido fundamental para llegar a esta etapa de mi vida. Ellos han estado a mi lado tanto en momentos buenos como en difíciles, guiándome y brindándome el respaldo necesario para convertirme en un profesional.

Mis padres no solo me han ayudado en esta fase crucial de mi vida, sino que también me han inculcado valores que me han permitido crecer como persona. Quiero expresar mi gratitud a mis hermanas, quienes también han sido un pilar importante, brindándome su apoyo incondicional en cada situación.

Agradezco a todas las personas que he conocido a lo largo de este camino hacia mi meta profesional. Cada uno de ustedes ha dejado una huella significativa en mi desarrollo, contribuyendo de manera fundamental a la persona que soy hoy. Su presencia y apoyo han sido invaluables, y estoy agradecido por haber compartido este viaje con personas tan maravillosas

José Guillermo Quinde Delgado

Resumen

Nos enfocamos en el desarrollo integral de un proyecto técnico altamente especializado que se centra en el monitoreo continuo de la inclinación de vehículos a través de un avanzado sistema de entretenimiento basado en Android. Este sistema en fase de desarrollo proporciona seguimiento constante de la inclinación, ofreciendo información vital en tiempo real a conductores y pasajeros acerca de posibles riesgos de vuelco. El dispositivo colabora eficazmente en la transmisión de datos capturados por sensores hacia la aplicación Android, utilizando tecnología Wi-Fi junto con el sistema embebido ESP32.

La aplicación, meticulosamente diseñada para integrarse perfectamente con el sistema de entretenimiento vehicular, permite una presentación en tiempo real y una interpretación precisa de los datos recopilados. Especialmente concebida para la conducción en terrenos difíciles, la aplicación posibilita el monitoreo tanto de la inclinación transversal como de la capacidad de ascenso del vehículo, utilizando el avanzado sensor de inclinación (MPU6050) para obtener estos datos en tiempo real.

La interfaz de la aplicación ha sido diseñada estratégicamente para ser intuitiva y fácilmente comprensible por parte del usuario. Desarrollada en Ionic, un marco de código abierto que emplea tecnologías web estándar como HTML, CSS y JavaScript/TypeScript, busca ofrecer al usuario una experiencia placentera y de fácil comprensión.

Adicionalmente, hemos implementado un protector plástico exclusivamente diseñado para resguardar los componentes electrónicos. Esta medida tiene como objetivo prevenir posibles problemas que podrían comprometer la integridad de los componentes, garantizando al mismo tiempo su colocación adecuada dentro del vehículo. Durante las pruebas, se llevó a cabo una evaluación exhaustiva del rendimiento de la aplicación en condiciones reales, lo que resultó en una aplicación optimizada y segura para los conductores de vehículos todo terreno.

Palabras clave: Sistema embebido, Inclinómetro, Android, Sistema de entretenimiento, Internet de las Cosas

Abstract

We focus on the comprehensive development of a highly specialized technical project that revolves around the continuous monitoring of vehicle tilt through an advanced Android-based entertainment system. This system, currently in the development phase, provides constant tilt tracking, offering real-time vital information to drivers and passengers about potential rollover risks. The device effectively collaborates in transmitting data captured by sensors to the Android application, utilizing Wi-Fi technology in conjunction with the embedded ESP32 system.

The meticulously designed application seamlessly integrates with the vehicle entertainment system, allowing real-time presentation and precise interpretation of the collected data. Tailored for driving in challenging terrains, the application enables monitoring of both lateral tilt and vehicle ascent capability, utilizing the advanced inclination sensor (MPU6050) to obtain these real-time data.

The application interface has been strategically crafted to be intuitive and easily understandable for the user. Developed in Ionic, an open-source framework that utilizes standard web technologies such as HTML, CSS, and JavaScript/TypeScript, it aims to provide the user with a pleasant and easily comprehensible experience.

Additionally, we have implemented a specially designed plastic protector to safeguard the electronic components. This measure aims to prevent potential issues that could compromise the integrity of the components while ensuring their proper placement within the vehicle. During testing, a thorough evaluation of the application's performance in real-world conditions was conducted, resulting in an optimized and secure application for off-road vehicle drivers.

Keywords: Embedded System, Inclinometer, Android, Entertainment System, Internet of Things

Índice de contenido

Índice de figuras

Índice de tablas

1 Introducción

En el contexto actual, el desarrollo de sistemas dedicados al monitoreo de la inclinación de vehículos se erige como una respuesta esencial para potenciar la seguridad de los ocupantes. La creciente necesidad de implementar dichos sistemas se refleja en la urgencia de mitigar riesgos de vuelco, lo que se traduce directamente en la prevención de accidentes y la reducción de lesiones y fatalidades. Este proyecto se propone abordar esta imperante demanda mediante la creación de una aplicación Android de monitoreo. Al aprovechar los sistemas de entretenimiento integrados en los vehículos, la aplicación proporcionará, en tiempo real, información detallada sobre la inclinación del vehículo, empoderando a conductores y acompañantes con una herramienta invaluable para mejorar la seguridad durante la conducción en terrenos desafiantes.

La integración de tecnología Android en vehículos ha revolucionado la experiencia de conducción, proporcionando comodidad y funcionalidad avanzada. Sin embargo, es crucial abordar las limitaciones actuales, como la compatibilidad de aplicaciones que afecta la efectividad del monitoreo de diversos valores. A pesar de estas restricciones, este proyecto se enfoca en superar estos desafíos, presentando una solución integral. La aplicación Android desarrollada, diseñada para complementar el sistema de entretenimiento del vehículo, tiene como objetivo proporcionar valores precisos en tiempo real mediante el uso de un sensor de inclinación. Esta iniciativa no solo busca mejorar la seguridad, sino también superar las barreras tecnológicas, brindando a los conductores una herramienta completa y eficiente para optimizar la experiencia de conducción en condiciones difíciles.

En la medida en que avanzamos hacia un futuro impulsado por la innovación, este proyecto no solo representa un paso significativo en la mejora de la seguridad vehicular, sino que también destaca la importancia de adaptar tecnologías existentes para abordar las demandas cambiantes de los conductores modernos. La aplicación Android propuesta no solo responde a la necesidad inminente de monitoreo de inclinación, sino

que también demuestra un compromiso continuo con la evolución de soluciones tecnológicas para garantizar una conducción más segura y eficiente en cualquier entorno.

2 Problema

La falta de inclinómetros en la mayoría de los vehículos todo terreno utilizado extensamente en Ecuador, especialmente en las camionetas pick-up, representa una preocupación crucial en términos de seguridad vial. La alta demanda de estos vehículos se debe a las condiciones geográficas y de vialidad del país, donde el transporte de insumos implica el acceso a zonas montañosas y terrenos desafiantes. En este escenario, los vehículos todo terreno se ven sometidos a inclinaciones extremas, llevándolos al límite de sus capacidades y aumentando el riesgo significativo de vuelco.

La situación se agrava al considerar que, a pesar de contar con sistemas de entretenimiento basados en Android, la mayoría de estos vehículos carecen de inclinómetros integrados. Esta carencia limita la capacidad de los conductores para monitorear de manera efectiva la inclinación del vehículo, lo que resulta en un aumento del peligro y la imposibilidad de prevenir accidentes potenciales. La diferencia en la presencia de tecnologías de seguridad, donde solo algunos vehículos todo terreno de alta gama tienen inclinómetros, destaca la necesidad apremiante de abordar esta falta tecnológica.

En consecuencia, el desarrollo de un inclinómetro específicamente diseñado para vehículos todo terreno con sistema de entretenimiento basado en Android se presenta como una solución esencial. Este proyecto no solo aborda una carencia técnica crítica en los vehículos más ampliamente utilizados en el país, sino que también busca mejorar significativamente la seguridad de los conductores y sus cargas, especialmente al enfrentar los desafíos inherentes a terrenos accidentados y montañosos.

En la actualidad, hemos observado la existencia de aplicaciones comerciales para tablets y smartphones basados en Android que facilitan la visualización de inclinómetros en la pantalla de estos dispositivos. Estas aplicaciones aprovechan los sensores internos

disponibles en tablets y smartphones. Sin embargo, surge un desafío cuando consideramos los sistemas de entretenimiento en vehículos que también funcionan con Android; estos carecen de sensores de inclinación incorporados, lo que limita la aplicación de las soluciones comerciales previamente mencionadas.

En este contexto, nuestro trabajo de titulación busca resolver esta limitación al proponer el desarrollo de un inclinómetro específicamente diseñado para vehículos todo terreno con sistemas de entretenimiento basados en Android. La propuesta implica la creación de un sistema embebido que incorpora un sensor de inclinación y la presentación de los datos medidos a través de una aplicación móvil. Esta aplicación se integrará en el sistema de entretenimiento preexistente en el vehículo.

El alcance de nuestro proyecto abordará la implementación de esta solución sin comprometer las funciones esenciales del sistema de entretenimiento, como la conectividad manos libres y la reproducción de música desde el smartphone. Para garantizar la coexistencia armoniosa de nuestro inclinómetro con el sistema ya existente en el vehículo, optaremos por la comunicación mediante wifi en lugar de bluetooth. Este enfoque asegura no solo la compatibilidad técnica, sino también la comodidad y funcionalidad continua de las características ya presentes en los vehículos todo terreno con sistemas de entretenimiento basados en Android.

3 Objetivos Generales Y Específicos

3.1 Objetivo general

Desarrollar un inclinómetro para un vehículo todo terreno con sistema de entretenimiento basado en Android.

3.2 Objetivos específicos

- **OE1.** Realizar un estudio de los sensores de inclinación disponibles y su comunicación a través de wifi con una app móvil, a través de la revisión de bibliografía especializada a fin de seleccionar el dispositivo más adecuado.

- **OE2.** Diseñar y construir un sistema embebido que permita a la aplicación móvil leer los datos proporcionados por el sensor de inclinación, considerando requerimientos de inmunidad a la interferencia electromagnética presente en el vehículo, a fin de obtener un dispositivo fiable.

- **OE3.** Desarrollar una aplicación móvil a ser instalada en un sistema de entretenimiento basado en el sistema operativo Android, que permita establecer la comunicación a través de wifi con el sistema embebido y visualizar la información proporcionada, mediante el uso de herramientas informáticas especializadas, en aras de sentar las bases para la implementación.

- **OE4.** Diseñar y ejecutar un plan de experimentación, en el cual a través de la realización de pruebas de funcionamiento se puede determinar los límites de operación, el desempeño y posibles mejoras.

4 Revisión de la literatura o fundamentos teóricos

En esta sección, nos sumergiremos en la exploración detallada de los diversos principios esenciales que sustentan el desarrollo del proyecto. Para este propósito, se llevó a cabo la revisión bibliográfica de las distintas tecnologías que serán empleadas, proporcionando así los conocimientos fundamentales requeridos para la consecución exitosa del proyecto.

4.1 Sensor de inclinación

Los sensores de inclinación, comúnmente referidos como sensores de ángulo de inclinación o sensores de nivelación vehicular, son dispositivos especializados diseñados para cuantificar la inclinación o ángulo de inclinación de un vehículo en relación con la superficie del suelo o un punto de referencia establecido. Estos dispositivos desempeñan un papel crucial al proporcionar mediciones precisas que permiten evaluar la orientación del vehículo, siendo esenciales para diversas aplicaciones, desde sistemas de seguridad hasta la mejora del rendimiento en terrenos irregulares.

En el área automotriz (Lara Hernández,2023), nos explica sobre el funcionamiento del inclinómetro, que permite el monitoreo del estado de inclinación y la nivelación, así como los parámetros para una correcta nivelación antes, durante y después del proceso de nivelación y estabilización. Esto contribuye a disminuir la probabilidad de volcadura.

4.2 Giroscopio

Los giroscopios desempeñan un papel fundamental en dos esferas clave: como sensores, suministrando datos precisos sobre posiciones angulares, y como dispositivos activos que desempeñan un papel crucial en la estabilización de vehículos y otros dispositivos. Los giroscopios controlados se destacan por su versatilidad superior en comparación con sus contrapartes de lazo abierto, ya

que permiten ajustes dinámicos en su comportamiento. Esta capacidad de adaptación dinámica amplía significativamente las posibilidades de aplicación, brindando una mayor flexibilidad en diversos contextos y escenarios operativos.

4.3 Sensor MPU6050

El sensor MPU6050 se presenta como un componente polifacético utilizado en entornos electrónicos y de robótica para la medición de aceleración y velocidad angular en objetos. Este sensor fusiona un acelerómetro con un giroscopio en un solo chip, permitiendo la detección tridimensional de cambios en posición y movimiento.

En detalle, el acelerómetro del MPU6050 capta la aceleración lineal, lo que habilita al sensor para determinar la orientación espacial y evaluar las variaciones en la velocidad de un objeto. Paralelamente, el giroscopio registra la velocidad angular, brindando información sobre la velocidad de rotación.

Este componente encuentra aplicaciones diversas, desde proyectos de robótica hasta el desarrollo de drones, y se inserta en dispositivos electrónicos de consumo como juguetes y dispositivos de realidad virtual. Su popularidad radica en su tamaño compacto, bajo consumo energético y la capacidad de proporcionar datos precisos sobre movimiento y posición, haciendo del MPU6050 una elección frecuente en contextos donde se buscan mediciones eficientes en este ámbito.

4.4 Sistema de entretenimiento Android

En la era actual, los sistemas operativos Android han arraigado profundamente su presencia, destacándose de manera aún más prominente en el ámbito automotriz, especialmente en las unidades de entretenimiento de los vehículos. La abrumadora mayoría de estas unidades integran diversas aplicaciones específicamente desarrolladas para Android, asegurando un funcionamiento coherente y eficiente en el entorno vehicular.

Como señala (Paul Pucurucu, Paul Tinizhañay, 2023), los sistemas de entretenimiento basados en Android han revolucionado la experiencia de conducción, aunque presentan desafíos en términos de integración y compatibilidad con aplicaciones para el monitoreo de valores. Android, concebido originalmente para dispositivos móviles, ha ampliado su alcance a tabletas, televisores y discos multimedia. Este sistema operativo, con su núcleo basado en Linux, se destaca como un sistema abierto y multiplataforma, accesible de manera gratuita.

Los sistemas de entretenimiento incorporados en los vehículos se diseñan meticulosamente para brindar una experiencia multimedia enriquecedora. Su función va más allá, cumpliendo roles esenciales como proporcionar información pertinente y facilitar una interacción sin problemas entre diversas aplicaciones que el vehículo puede albergar. Este enfoque integral tiene como objetivo central preservar la seguridad del conductor, creando un entorno armonioso que mejora la experiencia de conducción y reduce distracciones innecesarias.

Siguiendo la visión de (Luis Herrera, Cesar Hernández,2022), los sistemas de entretenimiento Android buscan fundamentalmente ofrecer monitoreo y rastreo en tiempo real de vehículos, elevando así el nivel de seguridad para los usuarios. Estas herramientas no solo proporcionan seguridad adicional, sino que también simplifican procesos legales mediante la grabación de videos en situaciones críticas.

En otro plano, según las explicaciones de K. Omerovic (2016), la unidad de información y entretenimiento en el vehículo constituye un conjunto de hardware y software destinado a proporcionar entretenimiento de audio y video. Sus características comunes incluyen radio, multimedia, control climático, datos del vehículo, navegación y conexión de segunda pantalla.

En la actualidad, los sistemas operativos Android han consolidado su presencia, especialmente en el ámbito automotriz y las unidades de entretenimiento vehicular. Estas unidades incorporan aplicaciones específicas para Android, asegurando un funcionamiento eficiente. Aunque han revolucionado la experiencia de conducción, los sistemas de entretenimiento Android enfrentan desafíos en integración y compatibilidad para el monitoreo de valores. Android, inicialmente para dispositivos móviles, se ha expandido a diversas plataformas, siendo un sistema abierto y gratuito con núcleo basado en Linux.

4.5 Aplicaciones Android

Las aplicaciones desarrolladas en Android para vehículos son sistemas en su mayoría diseñados para el infoentretenimiento. Cumplen funciones que proporcionan a los usuarios una amplia variedad de opciones, ofreciendo una experiencia multimedia en el vehículo y facilitando la entrega de información al usuario.

Según (Hilda Villacres,2022), las aplicaciones Android han sido diseñados principalmente en Java durante muchos años, un lenguaje reconocido por su versatilidad multiplataforma, facilidad y robustez al crear aplicaciones, lo que lo convierte en uno de los lenguajes más influyentes en la actualidad. Las aplicaciones Android híbridas son construidas con tecnologías web como HTML, CSS y JavaScript, brindando la ventaja de un desarrollo más accesible y económico

Enfocándose en la movilidad, los dispositivos Android juegan un papel fundamental al proporcionar una herramienta que mantiene a los profesionales informados en tiempo real, mejorando su eficiencia al permitir respuestas rápidas y la utilización de diversas funcionalidades directamente desde sus dispositivos móviles. (Alexis Tarupi,2020)

Como lo explica (Pedro Sorriguieta,2017), las aplicaciones Android están integradas cada vez en más funciones del vehículo, y, tras su implantación en el automóvil, los coches conectados están llamados a ser el siguiente paso en la evolución del llamado Internet de las Cosas.

Las aplicaciones Android diseñadas para vehículos se centran en el infoentretenimiento, ofreciendo a los usuarios una experiencia multimedia y facilitando el acceso a información variada. Estas aplicaciones, desarrolladas principalmente en Java, destacan por su versatilidad, facilidad y robustez. Se ha observado un aumento en la integración de funciones de movilidad, donde los dispositivos Android desempeñan un papel crucial al mantener a los profesionales informados en tiempo real, mejorando la eficiencia y permitiendo respuestas rápidas.

4.6 Sistema embebido

Un sistema embebido según (Pablo Garcia,2019) “se le denomina a un sistema electrónico con capacidad de cómputo diseñado para ejecutar una o varias tareas bien específicas, a diferencia de una computadora personal que es para usos múltiples. En general los sistemas embebidos poseen menos capacidad de cómputo que las computadoras personales y se diseñan optimizando el tamaño y el consumo de energía. Se denominan embebidos porque forman parte de un dispositivo mayor con partes mecánicas o electromecánicas”.

Estos sistemas desempeñan un papel fundamental al proporcionar tanto entretenimiento como conectividad dentro del vehículo. En algunos casos, estos sistemas ofrecen acceso a Internet mediante conexiones Wifi, permitiendo a los ocupantes aprovechar una serie de servicios diversificados. Entre estos servicios se incluyen la transmisión de música, la navegación en tiempo real y una amplia variedad de aplicaciones de entretenimiento, todo con el fin de mejorar y enriquecer la experiencia a bordo del vehículo. Este enfoque integral no solo potencia el aspecto lúdico del trayecto, sino que también proporciona una

conectividad avanzada que responde a las demandas modernas de entretenimiento y comunicación en el entorno automotriz.

5 Marco Metodológico

En esta sección, se abordará de manera exhaustiva la metodología delineada para la creación del dispositivo encargado de monitorear la inclinación del vehículo. El proceso se organiza en Sprint, donde cada uno será analizado detalladamente para alcanzar el objetivo principal del proyecto: Desarrollar un inclinómetro para un vehículo todo terreno con sistema de entretenimiento basado en Android. Cada sprint de la metodología se presenta como una parte esencial en la consecución de este propósito, asegurando una ejecución precisa y efectiva del proyecto en su totalidad.

5.1 Sprint 1

Para la ejecución del proyecto, estamos iniciando con una investigación exhaustiva sobre los sensores disponibles actualmente en el mercado para medir la inclinación. El objetivo principal de estos sensores es medir la inclinación o ángulo de un objeto en relación con la gravedad o un punto de referencia específico.

A continuación, presentamos algunos de los sensores de inclinación que podríamos considerar para integrar en nuestro proyecto:

- Inclinómetros digitales
- Inclinómetros capacitivos
- Inclinómetros biaxiales
- inclinómetros MEMS

Luego de revisar la documentación necesaria sobre las características de varios inclinómetros, llegamos a la conclusión de que el inclinómetro perteneciente al sistema microelectromecánico (MEMS) MPU6050 es el sensor más adecuado para la ejecución del proyecto.

5.2 Sprint 2

En este periodo, se describirá más a detalle sobre el sensor de inclinación MPU6050 y el sistema embebido ESP32 que se hará uso. El sensor de inclinación nos permite adquirir datos sobre el vehículo, es un sensor de movimiento de 6 ejes, combina de manera integrada un giroscopio de 3 ejes y un acelerómetro de 3 ejes en un único chip. Este dispositivo se caracteriza por su versatilidad y eficiencia, y se destaca por sus especificaciones actuales:

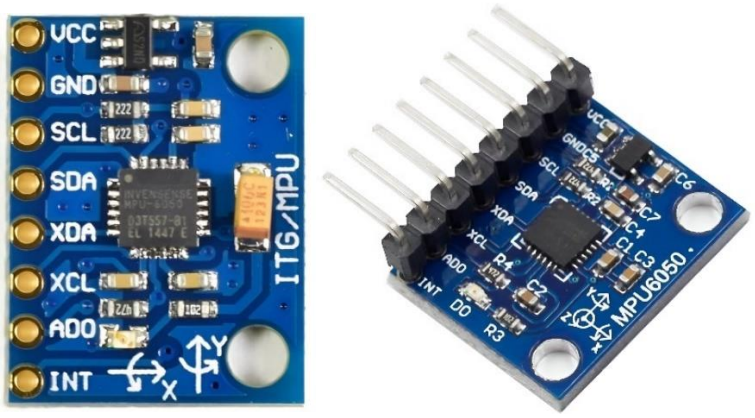


Figura 1: Sensor de inclinación MPU6050

- Especificaciones:

  - ❖ Interfaz de comunicación I2C.
  - ❖ Rango de medición giroscópica de ±250, ±500, ±1000 o ±2000 grados por segundo.
  - ❖ Rango de medición Acelerométrica de ±2g, ±4g, ±8g o ±16g.

- ❖ Datos precisos de orientación y movimiento.
- ❖ Diseño compacto,
- ❖ Bajo consumo de energía.
- ❖ Conexión directa al microcontrolador o Arduino.
- ❖ Voltaje de operación 3.3V hasta 5V.
- ❖ Frecuencia de muestreo hasta 8 kHz.
- ❖ Regulador de voltaje.

De la misma manera se presenta un resumen del diagrama de pines del MPU6050, destacando las conexiones clave necesarias para su correcto funcionamiento, con información proveniente de la hoja de datos oficial proporcionada por InvenSense, el fabricante del sensor.

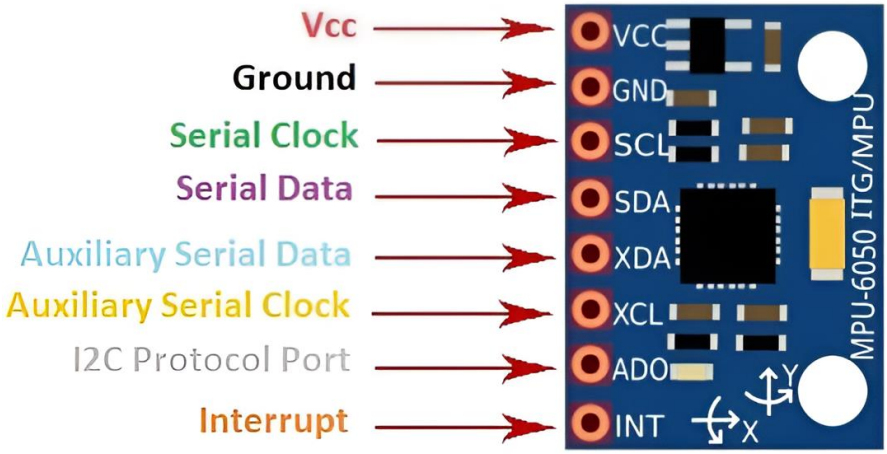


Figura 2:Diagrama de Pines del Sensor de inclinación MPU6050

- Diagrama de Pines:

  - ❖ VCC: Alimentación (Conectar a 3.3V o 5V según las especificaciones del dispositivo).
  - ❖ GND: Conexión a tierra (Conectar a GND).
  - ❖ SCL: Línea de Reloj I2C (Conectar al pin SCL del microcontrolador).
  - ❖ SDA: Línea de Datos I2C (Conectar al pin SDA del microcontrolador).

- ❖ XDA, XCL: No se utilizan en la mayoría de las aplicaciones y deben conectarse a tierra.
- ❖ ADO: Puede usarse para cambiar la dirección I2C si es necesario.
- ❖ INT: Pin de interrupción, utilizado para indicar la disponibilidad de nuevos datos.

El sistema embebido ESP32 es desarrollado por Espressif Systems, se destaca como un microcontrolador altamente versátil, que integra un procesador de doble núcleo potente, conectividad Wi-Fi y Bluetooth, así como diversos periféricos esenciales para el desarrollo de soluciones IoT (Internet de las cosas). Durante este proceso, exploramos las capacidades y características específicas del sistema embebido ESP32, resaltando su habilidad para ejecutar código en varios lenguajes de programación, como Python, JavaScript y C++

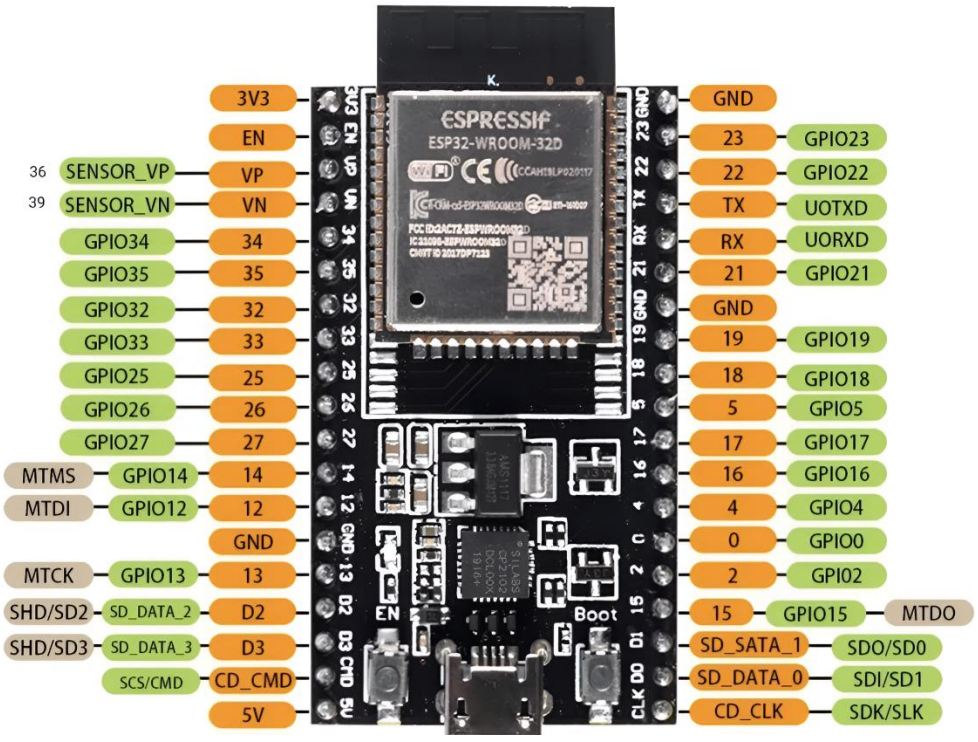


Figura 3:Sistema embebido ESP32

El sistema embebido ESP32 representa una evolución significativa respecto al ESP8266, mejorando sus capacidades tanto en comunicación como en procesamiento computacional. En el ámbito de la conectividad, posibilita la utilización de diversos protocolos inalámbricos, como WiFi, Bluetooth y BLE.

En este contexto, explicamos las especificaciones técnicas del sistema embebido ESP32:

- Especificaciones Técnicas:
  - ❖ Alimentación de 5V en corriente continua.
  - ❖ Entradas/Salidas: 3.3V en corriente continua.
  - ❖ Unidad Central de Procesamiento (CPU): Tensilica Xtensa LX6 de Doble Núcleo (32 bits).
  - ❖ Desempeño: Hasta 600 DMIPS.
  - ❖ Wifi: 802.11.
  - ❖ Bluetooth: Versión 4.2 BR/EDR y Bluetooth de Bajo Consumo de Energía.
  - ❖ Pines: 30.
  - ❖ Antena en PCB.
  - ❖ Capacidad de Almacenamiento hasta 16 MB Flash.
  - ❖ Corriente en Reposo: Menos de 5 µA.

Ahondando en las características específicas del sensor de inclinación MPU6050 y el sistema embebido ESP32, se presenta el siguiente diagrama lógico que detalla cómo se lleva a cabo el proyecto de titulación mediante un esquema por capas.

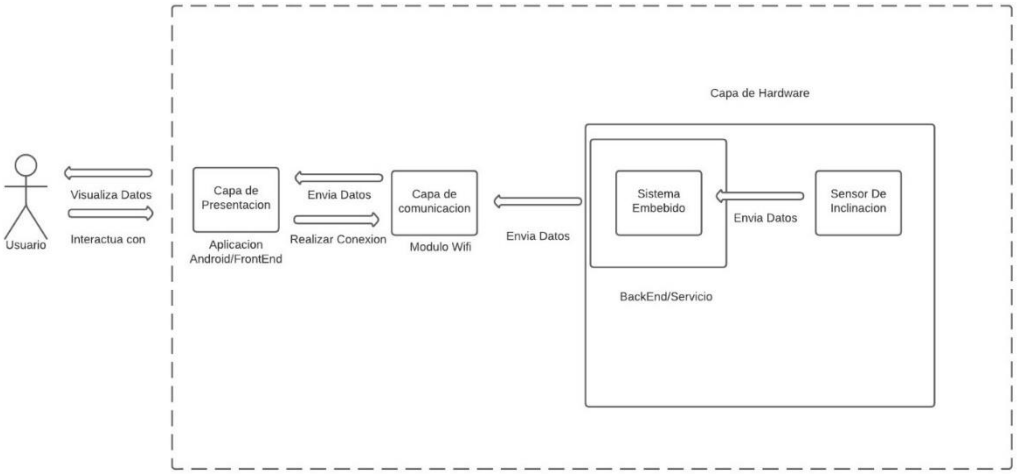


Figura 4:Esquema del Proyecto por Capas

En el actual Sprint 3, se detalla la capa de Hardware con mayor profundidad, abordando el desarrollo del sistema embebido y el sensor de inclinación. En este contexto, se describe el lenguaje de programación óptimo para el proyecto de titulación y se expone la conexión entre el sistema embebido y el sensor de inclinación. Además, se presenta la capa de comunicación, la cual se realiza mediante el módulo WIFI para la obtención de datos.

En el siguiente Sprint, correspondiente al número 4, se profundiza en la capa de presentación. Esta fase se centra en la aplicación desarrollada para Android, que facilita la interacción del usuario con el sistema.

5.3 Sprint 3

En este periodo, se detallará más a fondo la conexión entre el sistema embebido ESP32 y el sensor de inclinación MPU6050, aprovechando el módulo WIFI para la recopilación de datos.

5.3.1 Entorno de desarrollo para el sistema embebido ESP32

La elección del entorno de desarrollo para el sistema embebido ESP32 emerge como una determinación fundamental. En este contexto, se

destacan dos opciones significativas: el Arduino IDE y el Framework de Desarrollo de IoT de Espressif (ESP-IDF). Ambas alternativas presentan características particulares que requieren una evaluación minuciosa para garantizar un desarrollo eficaz y alineado con nuestros objetivos. La elección entre estas plataformas no solo define la experiencia de programación, sino que también influye en la flexibilidad y control sobre el hardware del sistema embebido ESP32, aspectos cruciales para el éxito de nuestro proyecto.

Arduino IDE: Reconocido por su facilidad de uso, este entorno ofrece un editor de código simple y una interfaz amigable, convirtiéndolo en la opción perfecta para aquellos que están dando sus primeros pasos en el desarrollo. La presencia de una comunidad extensa agrega un respaldo significativo, mientras que la incorporación de bibliotecas se realiza de manera fluida.

ESP-IDF: Desarrollado por Espressif, se presenta como la opción ideal para quienes buscan un control más detallado sobre el hardware del sistema embebido ESP32. Este framework proporciona acceso directo a funciones específicas, herramientas de depuración avanzadas y una documentación técnica exhaustiva, lo que lo convierte en una elección preferida para proyectos más avanzados.

El sistema embebido ESP32 ofrece compatibilidad con diversos lenguajes de programación, otorgando a los desarrolladores la flexibilidad de seleccionar el más acorde a sus preferencias. Entre los lenguajes de programación que se pueden utilizar con el sistema embebido ESP32 se encuentran:

C++: Es la elección habitual en la programación de microcontroladores, y el sistema embebido ESP32 no es una excepción. Este lenguaje no solo es compatible, sino que también

permite aprovechar los principios de la programación orientada a objetos para una implementación más eficiente y estructurada.

MicroPython: Es una implementación del lenguaje de programación Python optimizada específicamente para microcontroladores. Proporciona a los desarrolladores la capacidad de aprovechar la simplicidad y legibilidad inherentes a Python, especialmente adaptadas para dispositivos como el sistema embebido ESP32.

JavaScript: Algunos entornos habilitan el desarrollo en JavaScript, como el framework Espruino, lo que posibilita a los desarrolladores utilizar el lenguaje web para programar el sistema embebido ESP32

Arduino Language: Programar el sistema embebido ESP32 con el entorno de desarrollo Arduino implica la utilización de un dialecto simplificado de C++, lo que facilita la creación de aplicaciones para este microcontrolador.

La elección del entorno de desarrollo para el proyecto se centra en el uso de Arduino IDE, y nos basamos en el lenguaje de programación C++. Se ha constatado que este lenguaje cuenta con la documentación necesaria para minimizar errores al realizar la conexión entre el sistema embebido ESP32 y el sensor de inclinación MPU6050.

5.3.2 Conexión entre el sistema embebido ESP32 y el Sensor de inclinación MPU6050.

Como se presenta en la siguiente imagen el desiño del circuito de la conexión entre el sistema embebido ESP32 y el sensor de inclinación MPU6050, para realizar la conexión que sea eficiente nos basamos en la documentación ya antes descrita que se encuentra en el Sprint 2.

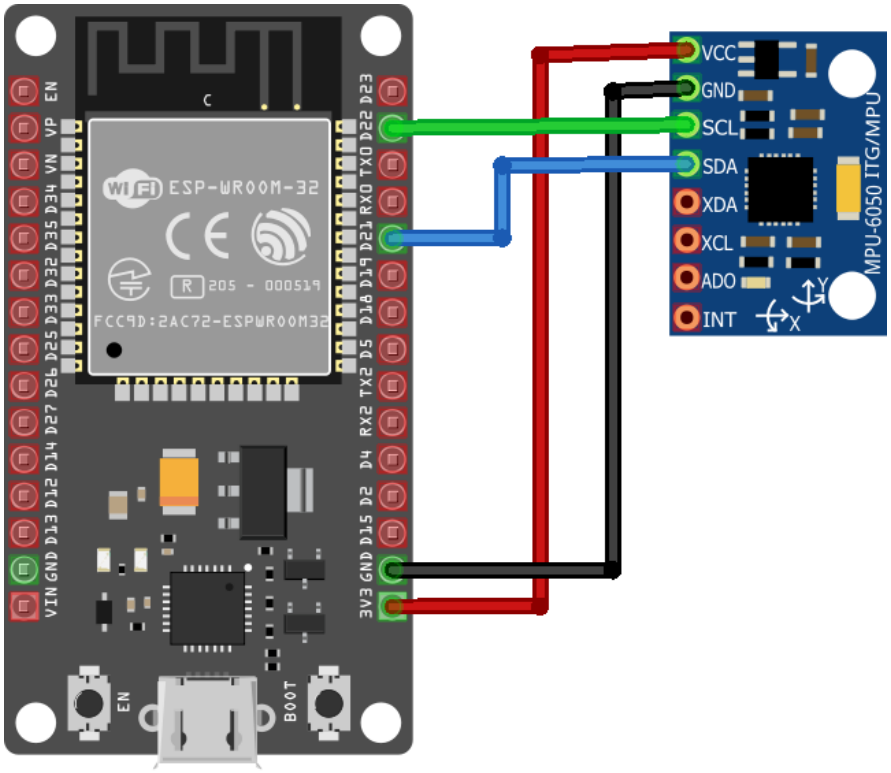


Figura 5:Circuito electrónico

Con el circuito ya ensamblado, procedemos a establecer la conexión entre el sistema embebido ESP32 y el sensor de inclinación MPU6050 mediante el entorno de desarrollo, utilizando el lenguaje de programación C++. En este proceso, haremos uso de las diferentes librerías como se muestra a continuación:

- Arduino.h: Esta librería es fundamental en cualquier programa de Arduino, ya que incluye definiciones y funciones esenciales del entorno, como las funciones de entrada/salida (E/S), el manejo del tiempo y funciones básicas de programación.

- WiFi.h: Esta librería ofrece las funciones esenciales para establecer y manejar la comunicación mediante redes Wi-Fi. Es frecuentemente empleada en proyectos que necesitan conectividad inalámbrica, como la comunicación con la nube o la conexión a una red local.

- Adafruit_MPU6050.h: Esta biblioteca está diseñada exclusivamente para el sensor MPU6050, un dispositivo que combina acelerómetro y giroscopio de 6 ejes. Facilita la comunicación con el MPU6050 para adquirir información sobre movimiento, aceleración y orientación.

- Adafruit_Sensor.h: Esta biblioteca sirve como base para varias bibliotecas de sensores de Adafruit. Ofrece una interfaz estandarizada para acceder a datos de diversos tipos de sensores, simplificando la integración de múltiples sensores en un solo proyecto.

- Arduino_JSON.h: Esta biblioteca facilita el manejo de formato JSON en proyectos de Arduino, simplificando la creación, manipulación y lectura de datos estructurados en JSON, ampliamente utilizado para el intercambio de información entre dispositivos o servicios.

- ESPAsyncWebServer.h: Esta biblioteca se emplea para desarrollar servidores web asíncronos en placas ESP8266 y ESP32. Facilita el manejo de conexiones concurrentes y operaciones asíncronas, siendo útil para construir aplicaciones web interactivas y gestionar solicitudes HTTP eficientemente.

Luego de detallar cada una de las librerías utilizadas en el proceso de conexión, se llevará a cabo la configuración para almacenar de forma

constante tanto el nombre de la red Wi-Fi como la contraseña correspondiente a dicha red. Estos valores serán posteriormente empleados en el código para establecer la conexión Wi-Fi de manera efectiva.

Después de completar la configuración para la red a la que se conectará el sistema embebido, se avanzará utilizando variables específicas para almacenar datos en formato JSON. Estas variables se emplean para capturar la información proveniente del sensor de inclinación. Además, se utilizan variables para administrar el tiempo en el programa. En este escenario, se emplea la variable “lastTime” para almacenar el tiempo (en milisegundos) de la última acción ejecutada por el programa, junto con la variable “gyroDelay” que indica el intervalo de tiempo entre lecturas relacionadas con el giroscopio.

Después de haber establecido dónde se almacenarán los datos y gestionado adecuadamente el tiempo del programa, se procederá con la configuración detallada del sensor de inclinación. Aquí, se definen variables específicas para capturar las lecturas del giroscopio, así como las desviaciones permitidas en cada uno de los ejes del sensor de inclinación. Además, se lleva a cabo la configuración de un servidor web asíncrono que escucha en el puerto 80, el puerto estándar para HTTP.

Con la configuración completa del sensor de inclinación y del servicio web, se procede a la inicialización del sensor de inclinación MPU6050 mediante la función “initMPU()”. Esta función intenta iniciar el sensor MPU6050 y, en caso de fallo de la conexión, entra en un bucle infinito. Si la conexión tiene éxito, se imprime un mensaje confirmando que el sensor MPU6050 se ha encontrado correctamente.

De manera similar, se realiza la inicialización y conexión a una red WiFi utilizando las credenciales proporcionadas. Para esto, se utiliza la función “initWiFi()”, que intenta conectar el dispositivo a la red WiFi utilizando las credenciales definidas al principio del programa. Mientras espera la conexión, se imprimen mensajes informativos en la consola serial. Una vez establecida la conexión con éxito, se imprime la dirección IP asignada al dispositivo.

Una vez completada la fase de inicialización del sensor de inclinación y la conexión a una red WiFi, se avanza a la creación de la función “getGyroReadings()”. Esta función tiene como objetivo obtener lecturas del giroscopio provenientes del sensor MPU6050. Se encarga de aplicar filtros para minimizar pequeñas variaciones y devuelve estas lecturas en formato JSON, presentadas en una cadena de texto. Con esta función, se establece una estructura eficiente para adquirir datos precisos del giroscopio y prepararlos para su posterior procesamiento.

En la configuración del servicio web, se implementan los encabezados CORS (Cross-Origin Resource Sharing), posibilitando solicitudes desde cualquier origen. Además, se especifican los métodos HTTP permitidos, que serán empleados por la aplicación desarrollada en Ionic. La incorporación de estos encabezados es fundamental para gestionar la política de mismo origen (Same-Origin Policy) y autorizar solicitudes desde dominios distintos al del servidor.

En este momento, se llevará a cabo la configuración inicial del programa. Se inicia la comunicación serial, se configuran y establecen las conexiones WiFi y del sensor MPU6050. Asimismo, se realizan los ajustes necesarios para los encabezados CORS. Se establece el manejo de solicitudes en el servidor web, especialmente en las rutas relacionadas con el giroscopio. Cuando se recibe una solicitud en esta

ruta, se ejecuta una función que obtiene las lecturas actuales del giroscopio mediante “getGyroReadings()” y envía la respuesta JSON al cliente que realizó la solicitud.

Finalmente se hace uso del bucle “loop()” que introduce un retardo de 10 milisegundos en cada iteración del bucle . Este retardo puede ser útil para limitar la frecuencia de ejecución del bucle y evitar que el programa se ejecute demasiado rápido.

Como resultado de la ejecución de este código se obtuvo lo siguiente:

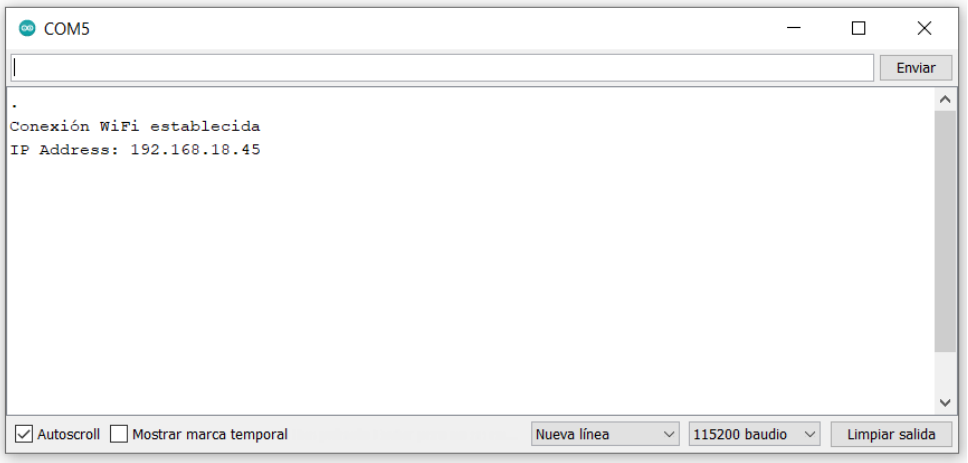


Figura 6:Dirección Ip al ejecutar el programa

Como se puede apreciar en la figura 5 , se ha obtenido una dirección IP que será utilizada en el desarrollo de la aplicación, como se detallará con mayor profundidad en el Sprint 4.omo se puede observar en la figura () se obtuvo una dirección ip la cual será ocupado al momento de realizar la aplicación como se detallara más a detalle en el Sprint 4.

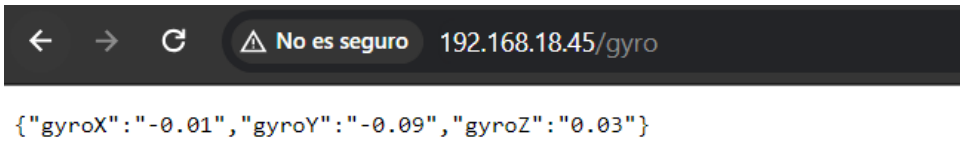


Figura 7:Ejecución del programa en la Web

Asimismo, conforme se visualiza en la figura 6, al utilizar la dirección IP junto con la ruta correspondiente, se introduce esta información en la red, y se logra obtener los datos en tiempo real provenientes del sensor de inclinación. Estos datos constituirán la base para la aplicación desarrollada en Ionic.

5.3.3 Desarrollo de la carcasa para el circuito realizado.

Para la creación de la carcasa que albergará nuestro circuito, empleamos el programa Tinkercad. En esta aplicación, llevamos a cabo el diseño teniendo en cuenta las medidas previamente tomadas de nuestro circuito conectado. El diseño resultante se presenta a continuación:

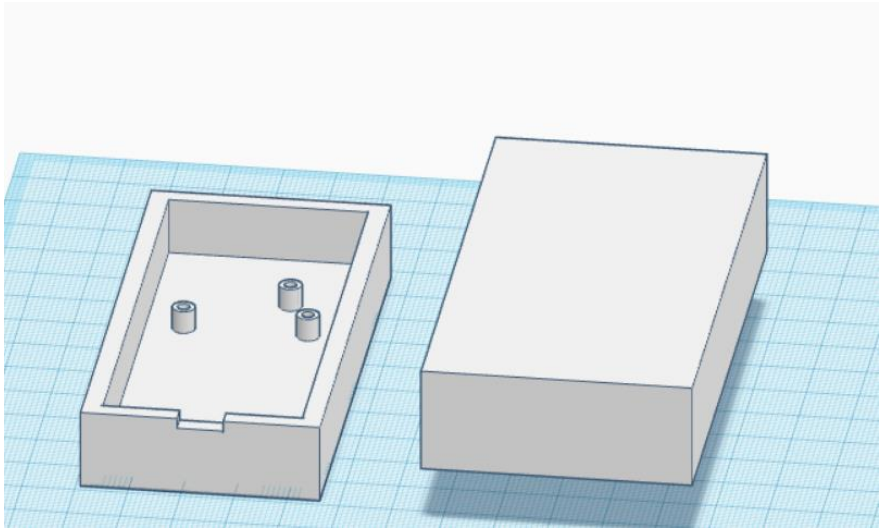


Figura 8:Diseño de la carcasa en Tinkercad

Después de completar el diseño, procedemos a configurar las especificaciones necesarias para la impresión en 3D. La finalidad de

esta carcasa es proteger nuestro circuito electrónico contra posibles riesgos y garantizar su correcta ubicación dentro del vehículo, asegurando una captura precisa de los datos.

5.4 Sprint 4

En este periodo vamos a describir todos los puntos que desarrollamos en este caso nos toco el desarrollo de la aplicación móvil para eso vamos a ir describiendo paso a paso lo que se realizó

5.4.1 Configuración de Entorno de Trabajo

Durante el proceso de establecimiento de nuestro entorno de desarrollo, nos propusimos cuidadosamente elegir una plataforma que se adaptara de manera eficiente a los requisitos únicos de nuestra aplicación. Después de una evaluación exhaustiva, tomamos la decisión de implementar Ionic como nuestro entorno principal de trabajo.

Justificación:

TypeScript y SCSS: Optamos por utilizar TypeScript como nuestro lenguaje de programación, beneficiándonos de su sistema de tipos estático que mejora la calidad y mantenimiento del código. Además, la inclusión de SCSS como preprocesador CSS nos proporciona una manera más eficiente y modular de estilizar nuestra aplicación.

Similaridad con Angular: Dado que ya contamos con experiencia en el entorno web de Angular, la similitud de Ionic con este marco de trabajo nos ofrece una transición más suave. Esto nos permite capitalizar nuestras habilidades actuales y la infraestructura existente de Angular, acelerando así nuestro proceso de desarrollo.

Enfoque en Dispositivos Móviles: La elección de Ionic se basa en su enfoque específico para el desarrollo de aplicaciones móviles. Este entorno nos proporciona herramientas y componentes diseñados específicamente para garantizar una experiencia de usuario fluida en dispositivos móviles.

Adaptabilidad a Dispositivos Móviles: Dado que nuestra aplicación se centra en dispositivos móviles, la capacidad de Ionic para adaptarse a este entorno es esencial. Esto nos asegura que nuestra aplicación siga las mejores prácticas de diseño y rendimiento para dispositivos móviles.

Integración con Angular: La integración sin complicaciones de Ionic con Angular es un aspecto crucial para nosotros. Esto simplifica la creación de componentes reutilizables y la organización estructurada del código, contribuyendo a una mayor eficiencia en el desarrollo.

La elección estratégica de Ionic como nuestro entorno de desarrollo no solo se fundamenta en su capacidad para aprovechar tecnologías modernas, sino también en su alineación con los objetivos específicos de nuestra aplicación. Este paso sienta las bases para un desarrollo eficaz y la entrega exitosa de una experiencia de usuario destacada en dispositivos móviles.

Figura 9:Logo del entorno de trabajo seleccionado

Para tener configurado el entorno de trabajo tenemos que asegurarnos de tener instalado Node en nuestro ordenador y tener un visor de código en este caso hemos utilizado Visual Studio Code, con estos requisitos

cumplidos lo único que necesitamos para crear un proyecto de Ionic es el comando “`ionic-start nombreproyecto`”

5.4.2 Diseño de la Interfaz del usuario

En esta fase ya se realizó el trabajar con los requisitos impuestos para esta aplicación web, por lo que nosotros lo hemos dividido en dos partes la capa de presentación en HTML y scss en donde realizaremos un home que será la pantalla principal del programa y la otra es la paca de typescript que se desarrolla en el archivo de home.ts., el primer punto para cumplir los requisitos es ver el desarrollo de algún inclinómetro que ya existe y aplicar una logia general para realizar la interfaz.

Figura 10:Diseño de un inclinómetro existente

Luego de tener un diseño por el cual nos podemos a realizar el siguiente punto fue seleccionar los asserts que se van a utilizar en el proyecto antes de poder armar la interfaz

El primer assert que realizamos fue el que nos pueda mostrar las medidas de inclinómetro con el cual hemos terminado usando este

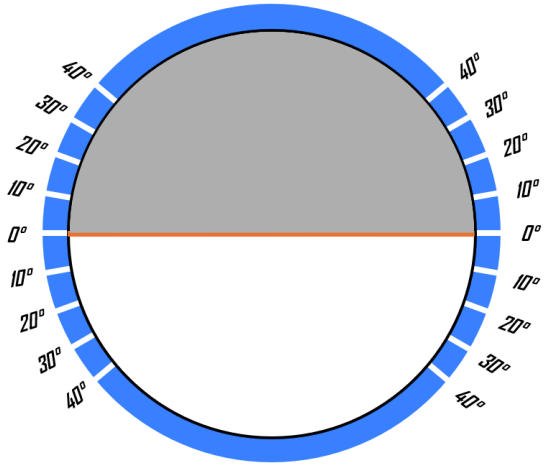


Figura 11:Diseño de la circunferencia en donde se mide los ángulos del inclinómetro

Como se puede ver la Figura 10 la realizamos en Paint con ayuda de Word ya que ahí podemos realizar círculos perfectos y podíamos ir dividiendo al círculo por grados iguales.

El hacer principal que utilizamos es el de un jeep en donde obtuvimos su vista lateral y trasera, con la cual ya podemos ir poco a poco viendo los materiales que se van a utilizar para el desarrollo de la interfaz.

Figura 12:Diseño Trasera Y Lateral de Automóvil Jeep que se utilizara.

Al obtener estos diseños de la Figura 11 lo primero que se realizo fue recórtales ya que son una misma foto y luego retiramos en blanco a su alrededor para que quede como un archivo .png con fondo transparente para su uso

La última figura que también visualizaremos en el proyecto es el logo de Cloud Computing el grupo de investigación al que va enfocado nuestro proyecto por lo que también les podremos dar una breve mención en la aplicación.

Figura 13:Logo del grupo de investigación Cloud Computing

Luego de obtener todo lo necesario para trabajar fue el inicio de comenzar a trabajar para primer punto nuestra ventana principal la desarrollaremos en home.html en este archivo lo primero que vamos a realizar es el header

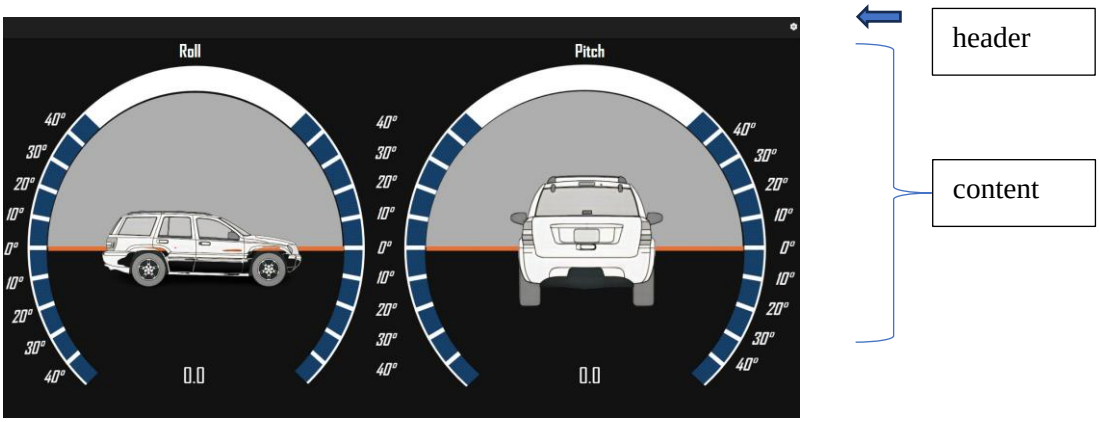


Figura 14:Interfaz Gráfica principal

En la ilustración Figura 13, se presenta la estructura que define la separación entre el encabezado (helador) y el contenido (contenté). Al analizar detenidamente el encabezado, nos encontramos con un título elocuente que proclama "Inclinómetro". En la esquina superior derecha, se despliega un menú que desempeña un papel crucial, proporcionando al usuario la capacidad de llevar a cabo acciones específicas que serán detalladas más adelante en el código.

En cuanto al contenido, se revela la vista principal que captura la esencia de la aplicación. A la izquierda, se presenta una representación visual del automóvil desde su parte trasera, mientras que, a la derecha, se muestra el automóvil en vista lateral. Dos textos informativos, denominados "Pich" y "Roll", ocupan un lugar destacado debajo de las representaciones visuales. Estos textos desempeñan la función de indicadores en tiempo real, proporcionando al usuario información precisa sobre los ángulos actuales detectados por el sensor.

Esta disposición armoniosa y visualmente atractiva permite una experiencia de usuario intuitiva y eficiente, donde la información es presentada de manera clara y accesible. Cada elemento en la interfaz contribuye a la comprensión inmediata de la aplicación, facilitando al usuario interactuar de manera efectiva con las funcionalidades ofrecidas.

Integración de Alertas y Ventanas Emergentes:

En particular, para la presentación de alertas y diversas ventanas emergentes en nuestra aplicación, optamos por la utilización de SweetAlert2, una biblioteca que ha añadido un toque distintivo a la experiencia del usuario. Esta herramienta no solo nos ha permitido implementar alertas de una manera efectiva, sino que también ha

mejorado significativamente el diseño, brindando una estética más agradable.

SweetAlert2 no se limita únicamente a notificar al usuario; va más allá, proporcionando alertas visuales y llamativas que elevan la calidad percibida de la aplicación. Su capacidad para crear alertas atractivas ha sido un aspecto clave para garantizar una interacción positiva y cómoda para aquellos que utilicen nuestra aplicación.

Esta elección deliberada de SweetAlert2 no solo cumple con la función técnica de presentar alertas, sino que también añade un valor estético y de usabilidad, creando una experiencia global más agradable para el usuario final, su uso en aplicación es muy fácil solo se debe instalar el sweetalert2.

Figura 15:Interfaz Gráfica del Acerca De realizado con sweetalert2

Queríamos que nuestra interfaz HTML fuera una algo muy bueno visual, y para lograrlo, nos sumergimos en el emocionante mundo de SCSS. ¿Recuerdas esos primeros ajustes donde modificamos los estilos? Pues sí, eso lo hicimos con SCSS, esa herramienta mágica que nos permitió darle el toque exacto a nuestros elementos.

La flexibilidad inherente de SCSS nos ha permitido modular y adaptar los estilos de manera eficiente. Al ajustar las propiedades de estilo, como colores, márgenes y tamaños de fuente, hemos logrado una armonía visual en la interfaz. La capacidad de SCSS para utilizar variables y funciones nos ha brindado una mayor coherencia en el diseño y ha simplificado la gestión de estilos complejos.

Este enfoque meticuloso en la personalización a través de SCSS no solo asegura la cohesión estilística en toda la aplicación, sino que también facilita futuras modificaciones y mejoras. La utilización de SCSS ha sido fundamental para alcanzar una interfaz HTML que no solo sea funcional, sino que también sea estéticamente agradable y adaptable a los objetivos específicos de nuestro proyecto.

5.4.3 Conexiones con el sistema embebido ESP32

Aquí, en el corazón de nuestro proyecto, nos sumergimos en la tarea crucial de conectar nuestra aplicación Ionic con el sistema embebido ESP32. La clave para esta integración es implementar la lógica necesaria que asegure una conexión eficiente y bidireccional.

Descripción Detallada:

Selección de Tecnologías: Para llevar a cabo esta conexión, hemos explorado y optado por la implementación de tecnologías como HTTP y WebSockets. Estas herramientas son fundamentales para facilitar la comunicación fluida entre la aplicación y el sistema embebido ESP32, permitiendo no solo la transmisión de datos desde el sensor hacia la aplicación, sino también la posibilidad de enviar comandos y realizar acciones específicas desde la aplicación hacia el sistema embebido ESP32.

Implementación de Lógica de Conexión: Nos hemos sumergido en la implementación de la lógica necesaria para establecer esta conexión de manera efectiva. Esto implica la configuración de solicitudes HTTP o la apertura de canales de comunicación bidireccional a través de WebSockets, según la tecnología seleccionada. Nos aseguramos de que esta lógica sea robusta, manejando posibles interrupciones de conexión y asegurando una comunicación estable.

Garantía de Bidireccionalidad: La comunicación bidireccional es clave para nuestro proyecto. Nos esforzamos por establecer un flujo constante de información entre la aplicación y el sistema embebido ESP32, permitiendo una interactividad efectiva. Esto no solo implica la transmisión de datos desde el sensor hacia la aplicación, sino también la capacidad de enviar comandos específicos desde la aplicación para realizar acciones en el sistema embebido ESP32.

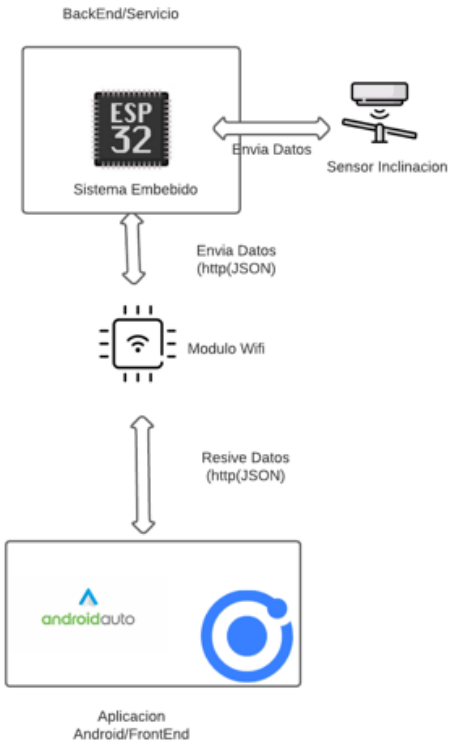


Figura 16:Arquitectura de envió de información HTTP entre servicio y aplicación

En la configuración estratégica de nuestra tesis como se puede ver en la Figura 15, el núcleo de nuestro servicio o back-end reside en el sistema embebido. Este elemento crucial desempeña un papel fundamental al transmitirnos datos mediante solicitudes HTTP a través de conexiones WiFi. En este cautivador escenario tecnológico, el sistema embebido se erige como el maestro de ceremonias, entregando sus valiosos datos hacia el frente de la acción: nuestra aplicación Ionic.

Conexión a través de WiFi: La conexión intrincada entre el sistema embebido y la aplicación Ionic se establece mediante un enlace ingenioso, donde los datos fluyen en ambos sentidos. En este contexto, el sistema embebido utiliza HTTP como el vehículo de entrega,

enviando sus datos a través de conexiones inalámbricas, específicamente WiFi.

Interacción de Datos en Formato JSON: La coreografía de datos entre ambas instancias se ejecuta con gracia, donde el sistema embebido despacha un JSON a través de una URL con protocolo HTTP hacia nuestra aplicación Ionic. Este JSON, cargado con información esencial, se convierte en el mensajero que trae consigo la riqueza de datos desde el sistema embebido hasta la interfaz de usuario de la aplicación móvil.

Visualización de Datos Dinámicos: Gracias a este flujo de información coordinado, nuestra aplicación Ionic se transforma en la receptora experta, capturando el JSON entrante y desatando una sinfonía visual. Este protocolo posibilita que los datos se integren de manera armoniosa en la interfaz, creando una experiencia inmersiva que redefine la interacción del usuario con la información proveniente del sistema embebido.

5.4.4 Desarrollo de funciones de la Aplicación

Nuestra aplicación consta de varias funciones, pero las más importantes son dos, las que actualizan los valores de giro de nuestras gráficas, tanto de la frontal como la lateral, ya que son las que nos van a visualizar la inclinación en tiempo real de nuestro sensor.

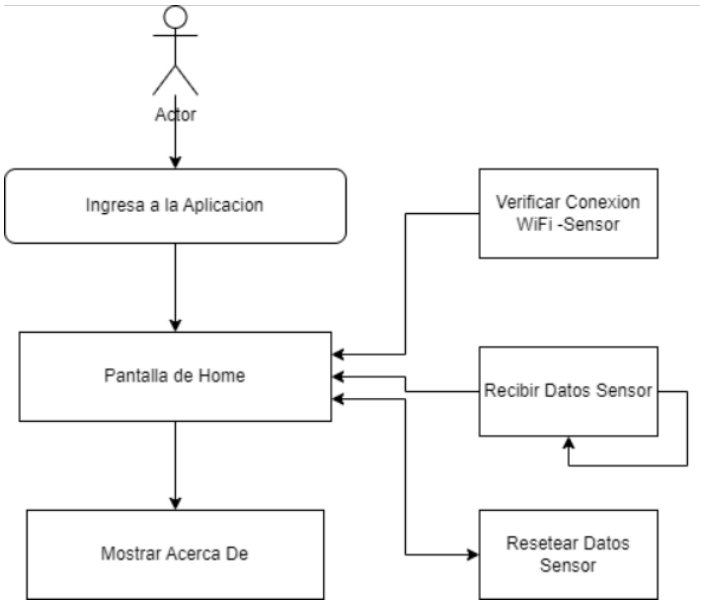


Figura 17:Diagrama de los procesos internos de la Aplicación

En la figura 16, se puede apreciar las actividades y procesos que realiza nuestra aplicación, como primer punto tenemos verificar la conexión del WiFi con el sensor MPU6050, tenemos que tener en cuenta que para poder recibir los datos tanto la aplicación como el sensor tienen que estar conectados en la misma red, por lo que hemos desarrollado un método que al iniciar la aplicación nos verifique la conexión del sistema embebido ESP32.

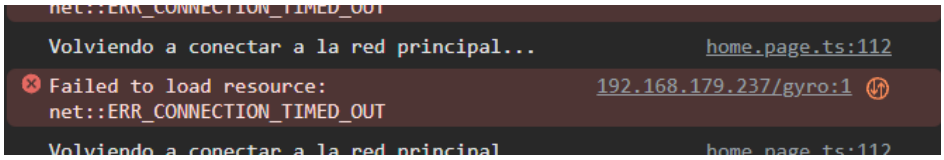


Figura 18:Control de error de Conexión de Wifi-Sensor

Como se puede ver en la figura 17 nosotros tenemos un mensaje de control que podemos ver en consola para saber si la aplicación no está conectada con la red, este mensaje lo pusimos para el desarrollo de la aplicación, pero para despliegue para que el usuario sepa que no está conectado a la misma red también pusimos un SweetAlert2 que nos va a mostrar un mensaje en pantalla para que el usuario sepa el problema.

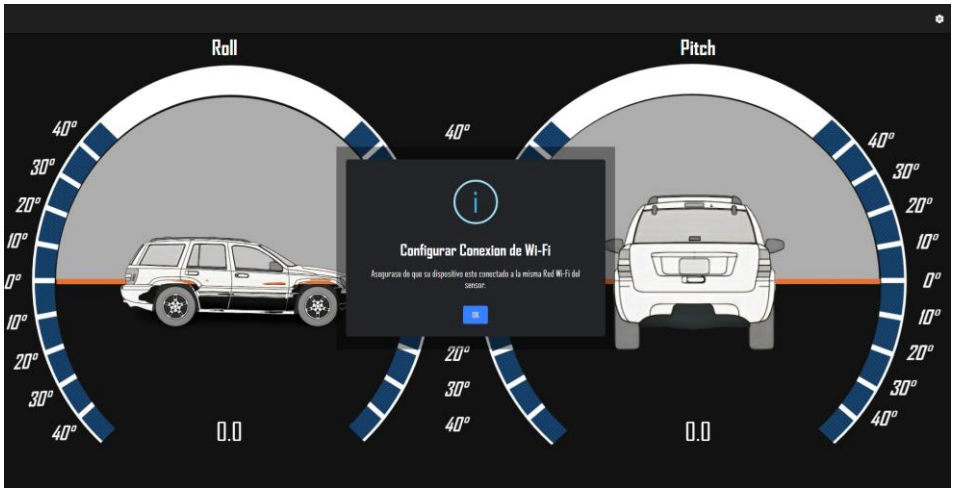


Figura 19:Alerta de Configuración de Wifi en Despliegue.

En la Figura 18 se presenta el método de recepción de datos, donde se implementa un proceso para obtener información del sensor a través de la URL correspondiente al mismo. En este caso, se recibe un objeto JSON que contiene los valores de los giros en los ejes X e Y del giroscopio. Estos valores se almacenan en variables para su posterior procesamiento.

La ejecución de este método se realiza a intervalos regulares de 10 milisegundos, lo que implica una actualización constante de los datos provenientes del sensor. Posteriormente, se han creado métodos destinados a la actualización de los valores en la interfaz gráfica de la aplicación, permitiendo que la información recibida se refleje de manera dinámica en pantalla.

Para obtener la rotación del sensor en la aplicación, se sigue un proceso específico. En primer lugar, se calcula el ángulo en radianes a partir de los valores de giro obtenidos. Luego, se realiza una conversión adecuada para obtener el valor correspondiente en grados reales. Este enfoque asegura que la representación de la rotación en la interfaz de usuario sea coherente y fácilmente comprensible para el usuario final.

El último método en consideración es el denominado "resetear". Este método se encarga exclusivamente de recibir un servicio de reset enviado desde el sistema embebido ESP32. Su funcionalidad principal consiste en restablecer todos los valores del sensor a cero. En otras palabras, este proceso garantiza que los datos registrados por el sensor sean reiniciados, proporcionando así un punto de partida consistente y neutral.

Finalmente tenemos un método para mostrar la información nuestra como desarrolladores de la aplicación.

Figura 20:Alerta de Acerca De

5.4.5 Desarrollo del despliegue de la Aplicación

En el proceso de despliegue de nuestra aplicación, la elección del entorno de trabajo de Ionic ha demostrado ser altamente beneficioso. Una herramienta clave que hemos aprovechado en este entorno es Capacitor, el cual nos ha simplificado considerablemente el despliegue de la aplicación.

Específicamente, al utilizar Capacitor para realizar el build de nuestra aplicación, hemos experimentado una mayor eficiencia en la generación de dependencias para la plataforma Android. Este procedimiento genera las dependencias directamente en el directorio raíz de nuestra aplicación, simplificando el manejo y la organización de los archivos relacionados con Android.

Este enfoque no solo agiliza el proceso de despliegue, sino que también proporciona una mayor flexibilidad y control sobre la configuración de la aplicación en la plataforma Android. Capacitor, en conjunto con Ionic, se ha convertido en una herramienta integral que mejora significativamente la productividad y facilita la gestión de las dependencias en el desarrollo de nuestra aplicación.

Para poder realizar el despliegue con un simple comando podemos realizarlo tenemos que tener en cuenta que tenemos que tener instalado Android Studio en nuestro ordenador con un sistema virtual dentro, el comando que se utiliza es “`ionic capacitor build android`”

En nuestro caso tuvimos un inconveniente con el servicio que al momento de realizar el despliegue no funcionaba y luego de verificar todas las opciones vimos que nuestro problema estaba en el AndroidManifest donde tenemos que agregar el permiso para el tráfico tanto de Http y Https, con esto nosotros pudimos finalmente llegar a la solución del error.

Uno de los puntos importantes para el despliegue de la aplicación es realizar un icono, en este caso nosotros nos hemos basado en un icono de un auto todo terreno con inclinación, ya que el icono debe dar una idea al usuario para que sirve la aplicación.

Figura 21:Figura para el Icono de la App

En la figura 20 podemos ver el icono que vamos a utilizar, para poder modificar el icono de la aplicación, se lo puede hacer luego de realizar desde Android Studio donde existe la opción de agregar un icono de nuestra preferencia como el que hemos solucionado y en este caso hemos visto que el fondo blanco va en tono con icono,y la aplicación se encarga de agregar los diferentes tamaños para el despliegue con su icono personaliza.

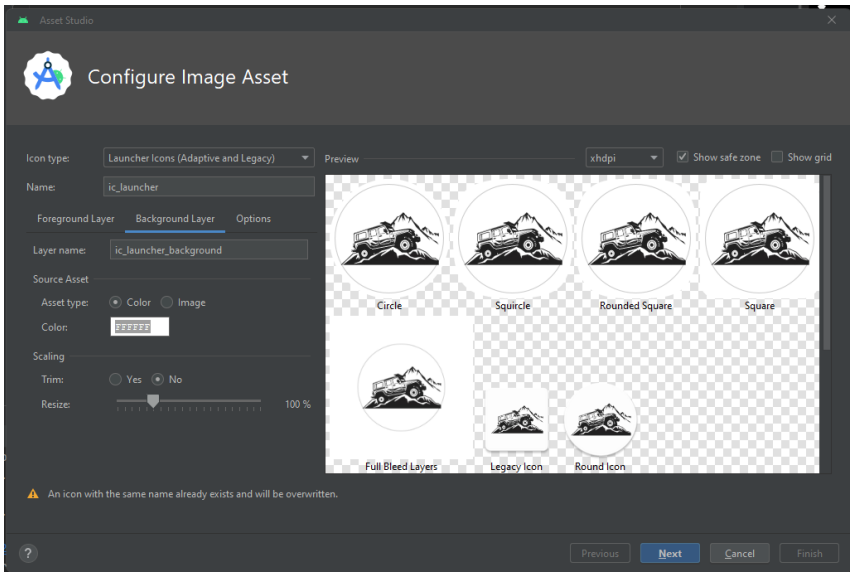


Figura 22:Proceso de Crear Icono para Aplicación

En la figura 21 se puede ver cómo se va creando el icono con sus diferentes dimensiones.

Hemos completado exitosamente el desarrollo de la aplicación móvil para el inclinómetro. Es importante destacar que la elección de Ionic desempeñó un papel crucial al simplificar significativamente el proceso. La similitud entre Ionic y Angular facilitó en gran medida el diseño, permitiendo una experiencia más fluida en la creación de la interfaz.

La implementación del despliegue mediante una simple línea de comandos, seguido de los ajustes finales en Android Studio, se reveló como un enfoque altamente eficiente. Esta metodología nos proporcionó la flexibilidad necesaria para realizar cambios y abordar posibles errores de manera eficaz. En resumen, la combinación de Ionic y Android Studio optimizó el desarrollo de la aplicación, haciendo que la experiencia fuera fluida y manejable en todas las etapas del proceso.

6 Resultados

En esta sección vamos a mostrar todos los resultados que fuimos obteniendo conforme el desarrollo de la aplicación móvil, también hablaremos un poco de los percances y también las optimizaciones que se podrían realizar a futuro.

6.1 Desempeño del sistema embebido ESP32

Indudablemente, la eficacia del sistema embebido ESP32 fue notable, aunque nos enfrentamos a uno de los primeros desafíos. Al intentar establecer una conexión WiFi con otro dispositivo, observamos que el sistema embebido ESP32 funcionaba como un punto de acceso independiente. Sin embargo, al conectar un dispositivo móvil a este punto de acceso y no recibir una respuesta de internet, se producía una desconexión automática. Para superar este inconveniente, optamos por utilizar una red WiFi externa que conectara ambos dispositivos.

En este contexto, la solución más viable fue configurar el sistema de entretenimiento del automóvil como un punto de acceso a internet y conectar nuestro ESP32 a este punto de acceso. De esta manera, logramos establecer con éxito la comunicación entre la aplicación y el sistema embebido ESP32, superando la limitación inicial y asegurando una conexión estable y funcional.

6.2 Desempeño del Sensor

El rendimiento del sensor demostró ser altamente satisfactorio. A través de nuestra investigación, identificamos que se trata de un sensor con una notable precisión, proporcionando no solo información sobre el giro, sino también sobre la temperatura y la aceleración. En el marco de nuestro proyecto, nos enfocamos específicamente en utilizarlo para medir la inclinación.

Es importante señalar que, como observación adicional, el sensor reveló ser algo sensible, requiriendo precaución en su manejo para evitar posibles daños o sobrecalentamientos. No obstante, a pesar de esta sensibilidad, el sensor ha demostrado ser altamente recomendable en virtud de los resultados positivos que hemos obtenido. Su capacidad para brindar mediciones precisas y su versatilidad para proporcionar información sobre diversos parámetros lo convierten en una elección acertada para nuestro proyecto.

6.3 Comunicación del sistema embebido ESP32-Aplicación Móvil

En esta etapa, es crucial tener presente la gestión de permisos de comunicación, incluyendo tanto el permiso HTTP como el permiso de Cors. Este aspecto se vuelve esencial para el funcionamiento óptimo de la aplicación móvil. Además, se deben conceder los permisos adecuados para el tráfico de red. En este punto, enfrentamos algunas dificultades que, tras diversas pruebas unitarias, logramos superar.

Inicialmente, aunque la aplicación en desarrollo operaba sin problemas, en el despliegue encontramos que no podía recibir datos del sensor. Identificamos que el problema residía en el tipo de solicitud que estábamos enviando: HTTP en lugar de HTTPS. Para solucionar esto, fue necesario ajustar el código base en Ionic y modificar el permiso correspondiente.

Posteriormente, a pesar de contar con los permisos de internet, nos encontramos con otra limitación: la aplicación aún no permitía la conexión con el servicio del sistema embebido ESP32. Después de un proceso de investigación, llegamos a la conclusión de que era necesario otorgar permisos de tráfico específicos desde la fase final de desarrollo, es decir, desde Android Studio. Este paso fue crucial para obtener los resultados deseados, permitiendo que la aplicación funcionara de manera satisfactoria finalmente.

6.4 Manejo de Errores y Excepciones

En esta etapa como bien se pudo mencionar en la metodología nuestro manejo de errores es simple, ya que el error prioritario que podemos tener en nuestra aplicación seria la falta de comunicación entre el sensor y la aplicación, por lo que al momento que el usuario ingrese a la aplicación en el caso de no estar en la misma red de Wi-Fi que el sensor esta mostrara un mensaje de advertencia, que indicara al usuario que deberá conectarse a la misma red, y bueno eso seria en si el manejo de errores o excepciones, ya que como el sistema pide solicitudes cada 1 ms cuando el usuario se conecte a la misma red, ya no tendrá advertencias y la aplicación funcionara adecuadamente.

Figura 23:Mensaje De Error de Conexión

A continuación, nos adentraremos en controles más allá de la gestión de errores, los cuales son requisitos fundamentales del proyecto. Se implementará un sistema de alertas dirigido al usuario en caso de que la inclinación del sensor supere los 30 grados, tanto en posición frontal como lateral. Este mensaje se

visualizará en la pantalla acompañado de una alarma sonora, asegurando que el conductor pueda percibir de inmediato cualquier situación peligrosa.

Más allá de abordar meros errores o controles, nuestra experiencia en este proyecto ha resaltado la sensibilidad delicada del sensor. La calibración precisa se vuelve crucial para garantizar que, al instalarse en el vehículo, la sensibilidad no varíe considerablemente. Este aspecto subraya la importancia de ajustar meticulosamente el sensor para mantener la coherencia en su rendimiento durante su implementación

6.5 Resultados de Caja 3D para nuestro Sistema Embebido y el sensor de inclinación

La caja realizada en 3D es muy valioso para los componentes electrónicos, como resultado de nuestra caja en 3D es realizarlo en dos capas la primera capa en donde se va a encontrar el sistema electrónico como se muestra en la figura 24.

Figura 24:Circuito en la primera capa de la caja

Como resultado de la segunda capa de caja en 3D, los componentes electrónicos quedaron totalmente protegidos, como se muestra en la figura 25, así también esto nos facilitara para colocar en un lugar adecuado del vehículo.

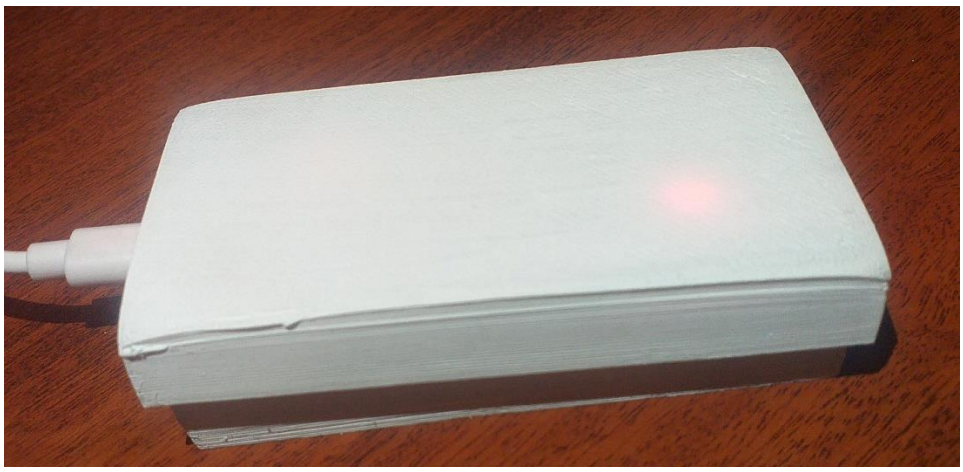


Figura 25:Cirucito totalmente protegido por la caja en 3D

6.6 Resultados de Funcionamiento

Como culminación de nuestro proyecto, nos complace informar que hemos logrado resultados altamente satisfactorios. Tras someter nuestro sistema a diversas pruebas y ajustes para gestionar posibles errores, hemos alcanzado exitosamente los objetivos establecidos. A pesar de los desafíos inherentes al manejo delicado de la sensibilidad del sensor, hemos superado estos obstáculos y obtenido un resultado final positivo.

Nuestra aplicación se destaca por ofrecer una interfaz amigable y de fácil uso, diseñada con colores que no resultan molestos para el usuario. Hemos implementado alertas efectivas para guiar al usuario en situaciones donde la aplicación podría no responder, mejorando así la experiencia de usuario.

Figura 26:Interfaz Principal de la Aplicación en el Vehículo

Además, hemos dedicado especial atención al diseño de nuestro sistema embebido, asegurándonos de que sea fácil de instalar en el vehículo. Esta

consideración facilita la utilización del sistema por parte del usuario, minimizando la posibilidad de errores durante la instalación.

Figura 27:Ventana de información de nuestra Aplicación

7 Cronograma

OE1. Realizar un estudio de los sensores de inclinación disponibles y su comunicación a través de wifi con una app móvil, a través de la revisión de bibliografía especializada a fin de seleccionar el dispositivo más adecuado.

No.	Actividad
1	Estudiar el funcionamiento de los sensores disponibles para inclinación.
2	Identificar el módulo wifi óptimo para implementación.
3	Realizar la documentación de la información más relevante.

Tabla 1:Actividad Objetivo

OE2. Diseñar y construir un sistema embebido que permita a la aplicación móvil leer los datos proporcionados por el sensor de inclinación, considerando requerimientos de inmunidad a la interferencia electromagnética presente en el vehículo, a fin de obtener un dispositivo fiable.

No.	Actividad
1	Realizar la conexión entre el sensor de inclinación y el sistema embebido a utilizar.
2	Realizar la configuración para obtener datos del sensor de movimiento en el ordenador.
3	Verificar valores de datos obtenidos para ver si son correctos.
4	Realizar un diseño en 3D para una carcasa para el circuito electrónico.
5	Instalar la carcasa 3D en el circuito electrónico.

Tabla 2:Actividad Objetivo Específico 2

OE3. Desarrollar una aplicación móvil a ser instalada en un sistema de entretenimiento basado en el sistema operativo Android, que permita establecer la comunicación a través de wifi con el sistema embebido y visualizar la información proporcionada, mediante el uso de herramientas informáticas especializadas, en aras de sentar las bases para la implementación.

No.	Actividad
1	Realizar módulo para la conexión del sistema embebido con la aplicación arduino.
2	Realizar módulo de interfaz del usuario en la aplicación de android.
3	Verificar si el funcionamiento entre la aplicación android y el sistema embebido son correctos.
4	Realizar correcciones y pulir la interfaz gráfica de la aplicación.

Tabla 3:Actividad Objetivo Específico 3

OE4. Diseñar y ejecutar un plan de experimentación, en el cual a través de la realización de pruebas de funcionamiento se puede determinar los límites de operación, el desempeño y posibles mejoras.

No.	Actividad
1	Verificar el funcionamiento de la aplicación en un sistema de entretenimiento Android en el automóvil.
2	Instalar el sistema embebido en el automóvil (Realizar calibración de inclinación).
3	Verificar si el funcionamiento es correcto.
4	Verificar si el desempeño es óptimo y realizar mejoras en caso de ser necesario.

Tabla 4:Actividad Objetivo Específico 4

Objetivo	**Actividad**	**Fecha Inicio**	**Fecha Fin**	**Horas por responsable**	**Responsables**
OE1	Ac.1 Estudiar el funcionamiento de los sensores disponible para inclinación	30/10/2023	02/11/2023	4 horas 4 horas	José Quinde Tonny Lema
OE1	Ac.2 Identificar el módulo wifi óptimo para implementación	03/11/2023	05/11/2023	4 horas 4 horas	José Quinde Tonny Lema
OE1	Ac.3 Realizar la documentación de la información más relevante.	06/11/2023	08/11/2023	4 horas 4 horas	José Quinde Tonny Lema
Total	**OE1**	**30/10/2023**	**08/11/2023**	**16 horas 16 horas**	**José Quinde Tonny Lema**
OE2	Ac.1 Realizar la conexión entre el sensor de inclinación y el sistema embebido a utilizar.	09/11/2023	13/11/2023	14 horas 14 horas	José Quinde Tonny Lema
OE2	Ac.2 Realizar la configuración para obtener datos del sensor de movimiento en el ordenador.	14/11/2023	20/11/2023	20 horas 20 horas	José Quinde Tonny Lema
OE2	Ac.3 Verificar valores	21/11/2023	22/11/2023	4 horas	José Quinde

	de datos obtenidos para ver si son correctos.			4 horas	Tonny Lema
OE2	Ac.4 Realizar un diseño en 3D para una carcasa para el circuito electrónico	23/11/2023	28/11/2023	18 horas 18 horas	José Quinde Tonny Lema
OE2	Ac.5 Instalar la carcasa 3D en el circuito electrónico	29/11/2023	30/11/2023	4 horas 4 horas	José Quinde Tonny Lema
Total	**OE2**	**09/11/2023**	**30/11/2023**	**60 horas** **60 horas**	**José Quinde** **Tonny Lema**
OE3	Ac.1 Realizar módulo para la conexión del sistema embebido con la aplicación Arduino	01/12/2023	08/12/2023	60 horas 60 horas	José Quinde Tonny Lema
OE3	Ac.2 Realizar módulo de interfaz del usuario en la aplicación de Android	09/12/2023	18/12/2023	60 horas 60 horas	José Quinde Tonny Lema
OE3	Ac. 3 Verificar si el funcionamiento entre la aplicación Android y el sistema embebido son correctos	19/12/2023	21/12/2023	20 horas 20 horas	José Quinde Tonny Lema
OE3	Ac. 4 Realizar correcciones y pulir la interfaz gráfica de la	22/12/2023	29/12/2023	40 horas 40 horas	José Quinde Tonny Lema

	aplicación.				
Total	**OE3**	**01/12/2023**	**29/12/2023**	**180 horas** **180 horas**	**José Quinde** **Tonny Lema**
OE4	Ac.1 Verificar el funcionamiento de la aplicación en un sistema de entretenimiento Android en el automóvil.	02/01/2024	05/01/2024	10 horas 10 horas	José Quinde Tonny Lema
OE4	Ac. 2 Instalar el sistema embebido en el automóvil (Realizar calibración de inclinación).	06/01/2024	08/01/2024	10 horas 10 horas	José Quinde Tonny Lema
OE4	Ac. 3 Verificar si el funcionamiento es correcto.	09/01/2024	12/01/2024	10 horas 10 horas	José Quinde Tonny Lema
OE4	Ac. 4 Realizar correcciones y pulir la interfaz gráfica de la aplicación.	13/01/2024	18/01/2024	30 horas 30 horas	José Quinde Tonny Lema
Total	**OE4**	**02/01/2024**	**18/01/2024**	**60 horas** **60 horas**	**José Quinde** **Tonny Lema**
Total	**Total por persona**	**30/11/2023**	**18/01/2024**	**300 horas** **300 horas**	**José Quinde** **Tonny Lema**
Total	**Total**	**30/11/2023**	**18/01/2024**	**600 horas**	

Tabla 5:Cronograma de Actividades

8 Presupuesto

DENOMINACIÓN	CANT.	COSTO UNITARIO	COSTO TOTAL
	Unidades	Dólares	dólares
1. Bienes			
Papel Bond A-4	1	10,00	10,00
Copias	100	0,05	5,00
2.Tecnológico			
Computadora portátil	2	1500,00	3000,00
Esp32(Sistema Embebido)	1	10,00	10,00
MPU6050 (Sensor Inclinación)	2	7,00	14,00
Módulo Sensor de Sonido	2	4,00	8,00

2. Servicios			
Servicio de transporte	2	40,00	80,00
Servicios de Internet	2	28,00	56,00
Material Bibliográfico	1	35,00	35,00
4. Personal			
Estudiante investigador	2	8 por Hora	4800,00
Asesoría especializada	1	8 por Hora	2400,00
3. Otros			
Imprevistos	2	200,00	400,00
Automóvil Todoterreno (Para instalación y pruebas)	1	24000,00	24000,00
TOTAL	1	34818,00	**34818,00**

Tabla 6:Tabla de Presupuesto

9 Conclusiones

En el transcurso de nuestro proyecto, iniciamos con una investigación exhaustiva de conceptos fundamentales, abarcando desde sensores de inclinación, sistemas embebidos, aplicaciones Android, hasta sistemas de entretenimiento en vehículos, además de explorar diversos entornos de desarrollo y lenguajes de programación. Estos fundamentos teóricos sirvieron como cimiento esencial para dirigir el desarrollo del proyecto, permitiéndonos cumplir de manera efectiva con los objetivos establecidos, respetar los plazos definidos y concretar de manera integral la propuesta inicial del proyecto.

Al adentrarnos en la implementación del servidor, subrayamos la importancia crítica de entender la interacción precisa entre el sensor de inclinación y el sistema embebido. Este entendimiento se tradujo en la capacidad de establecer una conexión efectiva entre ambos dispositivos electrónicos, un paso esencial para la obtención de datos fundamentales. Entre estos datos se destaca la dirección IP, generada mediante una confirmación adecuada en el sistema embebido, y la captura en tiempo real de los datos del sensor de inclinación. Estos conjuntos de datos contribuyen significativamente a la funcionalidad integral de nuestro proyecto.

Al concentrarnos en la creación de nuestra aplicación Android, subrayamos la importancia central de gestionar de manera precisa la información proveniente del servidor. Durante este proceso, priorizamos la tarea de presentar esta información de manera clara y accesible a través de una interfaz amigable. Esta interfaz, cuidadosamente diseñada, incorpora una representación visual del vehículo que refleja sus grados de inclinación en tiempo real cuando está en movimiento. Este enfoque integral garantiza una experiencia óptima para el usuario al proporcionar datos relevantes de manera intuitiva y visualmente atractiva.

Finalmente, durante el proceso de pruebas de la aplicación, iniciamos con pruebas alfa realizadas localmente. El objetivo principal fue asegurar el correcto funcionamiento de la aplicación, llevando a cabo pruebas en emuladores para verificar la interfaz y la visualización precisa de los datos. Posteriormente, avanzamos a las pruebas beta, donde la aplicación se instaló con éxito en el sistema de entretenimiento del vehículo. Este despliegue

recibió una respuesta positiva por parte de los conductores, ya que la aplicación proporcionó información en tiempo real sobre la inclinación del vehículo, fomentando una conducción más cautelosa y contribuyendo a la prevención de posibles accidentes.

10 Recomendaciones

Lo principal debemos tener en cuenta que para manejar este tipo de aplicaciones tener que dar todos los permisos para que de esta manera se realicen las particiones hacia el servidor, esto nos facilitará tener menos errores al momento de realizar la aplicación Android.

Por otra parte, se aconseja incorporar un conversor específico en el sistema electrónico del prototipo automotriz con el fin de adecuarse a las fluctuaciones de voltaje al iniciar el vehículo y prevenir posibles daños causados por picos de voltaje. Esta medida contribuirá a fortalecer la estabilidad y confiabilidad de la conexión con la alimentación directa del automóvil.

De manera similar, con el propósito de extender la vida útil de los componentes, se sugeriría la incorporación de un recubrimiento protector de naturaleza plástica. Este recubrimiento tiene la finalidad de prevenir la corrosión, la cual podría potencialmente interferir con el rendimiento efectivo del sistema.

En cuanto a la implementación de la aplicación, se aconseja llevar a cabo pruebas exhaustivas en el entorno de desarrollo de Android Studio. Este entorno proporciona diversas pantallas, incluyendo la interfaz del sistema de entretenimiento del vehículo. Aquí, se pueden realizar pruebas que abarquen desde el funcionamiento general de la aplicación hasta la evaluación de la interfaz, permitiendo ajustes previos a la instalación en el vehículo.

11 Referencias bibliográficas

Offroad Inclinometer (3D). (2023). IronAppsDeveloper (3D) [Aplicación móvil]. Google Play Store. https://play.google.com/store/apps/details?id=com.ironappsdev.offroadincl&hl=es_EC&gl=US

Jeep Inclinometer. (2023). Fabio Paracchini [Aplicación móvil]. Google Play Store. https://play.google.com/store/apps/details?id=air.com.adobe.example.inclinometer&hl=es_EC&gl=US

Rodas Merchán, J. F. (2023). Desarrollo de un sistema de monitoreo de inclinación de un vehículo con sistema de entretenimiento basado en Android. Universidad Politecnica Salesiana. http://dspace.ups.edu.ec/handle/123456789/25783

Lara Hernández, G., Flores Cuautle, J., Granados Salazar, L., & Rodríguez Jarquin, J. (2023). Implementación De Sensor IMU Para La Asistencia De Nivel En Grúas Todo Terreno. Ideas En Ciencias De La Ingeniería, 1(2), 62-73. doi:10.36677/ideaseningenieria.v1i2.21476.

Omerovic, K., Janjatovic, J., Milosevic, M., & Maruna, T. (2016). Supporting sensor fusion in next generation Android In-Vehicle Infotainment units. En IEEE 6.ª Conferencia Internacional sobre Electrónica de Consumo 2016 - Berlín (ICCE-Berlin) (pp. 187-189). DOI: 10.1109/ICCE-Berlin.2016.7684751.

Sorriguieta Torre, P. (2017). Aplicación Android para vehículos pesados (Treball Final de Grau). UPC, Escola d'Enginyeria de Barcelona Est. Retrieved from http://hdl.handle.net/2117/112945.

García, P. A. (2019). Sistemas embebidos de tiempo real con aplicaciones en bioingeniería. Doctoral dissertation, Universidad Nacional de La Plata. http://sedici.unlp.edu.ar/handle/10915/74734

Villacres Tandazo,H(2022). Estudio comparativo sobre las herramientas de lenguaje de programación java y python en el desarrollo de aplicaciones android.

Universidad técnica de Babahoyo Facultad de administración, finanzas e informática Proceso de titulación. http://dspace.utb.edu.ec/handle/49000/11689

Calva Pérez, S., Herrera Rodríguez, L.A. y Lucio Hernández, C.E. (2022). Asistente de vídeo vigilancia para el automóvil. (Ingeniería en Sistemas Automotrices). Instituto Politécnico Nacional, Unidad Profesional Interdisciplinaria en Ingeniería y Tecnologías Avanzadas, México. http://tesis.ipn.mx/handle/123456789/30920

Paul Sebastian ,P., Paul Sebastian,T(2023). Desarrollo de un sistema de monitoreo de la concentración de oxigenación y presión atmosférica para un vehículo con un sistema de entretenimiento basado en Android. Universidad Politecnica Salesiana. http://dspace.ups.edu.ec/handle/123456789/25613

Marco Antonio,C., Alexis Bolívar,T(2020). Sistema de bloqueo inalámbrico con tecnología android para la prevención del robo de vehículos. Universidad Regional Autónoma De Los Andes. https://dspace.uniandes.edu.ec/handle/123456789/11220

12 Anexos

```
/*********
Jose Quinde-Tonny Lema
Codigo para obtener la inclinacion del sensor MPU6050 a través de una ESP32
El cual se hace host de WIFI y envía servicios get que se conectaran a la
aplicación android.
*********/
#include <Arduino.h>
#include <WiFi.h>
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>
#include <Arduino_JSON.h>
#include <ESPAsyncWebServer.h>

// Credenciales de la red WiFi
const char *ssid = "Mi 10t pro";      // Nombre de tu red WiFi
const char *password = "jose1234"; // Clave de tu red WiFi

// Configuración de dirección IP estática
//IPAddress ip(192, 168, 1, 100);     // Establece la dirección IP estática

// Json Variable to Hold Sensor Readings
JSONVar readings;

unsigned long lastTime = 0;
unsigned long gyroDelay = 10;
```

```
// Create a sensor object
Adafruit_MPU6050 mpu;

sensors_event_t a, g, temp;

float gyroX, gyroY, gyroZ;

//Gyroscope sensor deviation
float gyroXerror = 0.04;
float gyroYerror = 0.03;
float gyroZerror = 0.01;

AsyncWebServer server(80);

// Inicializar el MPU6050
void initMPU()
{
  if (!mpu.begin())
  {
    Serial.println("Conexion Fallida MPU6050");
    while (1)
    {
      delay(10);
    }
  }
  Serial.println("MPU6050 Encontrado MPU6050");
}
// Inicializar WiFi
void initWiFi()
```

```
{
  // Inicializar WiFi con las nuevas credenciales
  WiFi.begin(ssid, password);
  //WiFi.config(ip);

  Serial.println("");
  Serial.print("Conectando a la red WiFi: ");
  Serial.println(ssid);

  // Esperar a que se conecte a la red
  while (WiFi.status() != WL_CONNECTED)
  {
    delay(1000);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("Conexión WiFi establecida");
  Serial.print("IP Address: ");
  Serial.println(WiFi.localIP());
}

String getGyroReadings()
{
  mpu.getEvent(&a, &g, &temp);

  float gyroX_temp = g.gyro.x;
  if(abs(gyroX_temp) > gyroXerror)  {
    gyroX += gyroX_temp/70.00;
```

```
  }

  float gyroY_temp = g.gyro.y;
  if(abs(gyroY_temp) > gyroYerror) {
    gyroY += gyroY_temp/60.00;
  }

  float gyroZ_temp = g.gyro.z;
  if(abs(gyroZ_temp) > gyroZerror) {
    gyroZ += gyroZ_temp/70.00;
  }

  readings["gyroX"] = String(gyroX);
  readings["gyroY"] = String(gyroY);
  readings["gyroZ"] = String(gyroZ);

  String jsonString = JSON.stringify(readings);
  return jsonString;
}
void setupCORS()
{
  DefaultHeaders::Instance().addHeader("Access-Control-Allow-Origin", "*");
  //DefaultHeaders::Instance().addHeader("Access-Control-Max-Age", "600");
  DefaultHeaders::Instance().addHeader("Access-Control-Allow-Methods",
"GET,HEAD,OPTIONS,POST,PUT");
  //SDefaultHeaders::Instance().addHeader("Access-Control-Allow-Headers",
"Authorization,Content-Type,Accept,Origin");
}

void setup()
```

```
{
  Serial.begin(115200);
  initMPU();
  initWiFi();
  setupCORS(); // Configurar encabezados CORS

  // Manejar la solicitud GET en la ruta /gyro
  server.on("/gyro", HTTP_GET, [](AsyncWebServerRequest *request) {
    // Obtener la lectura actual del giroscopio
    String gyroData = getGyroReadings();

    // Enviar la respuesta JSON al cliente
    request->send(200, "application/json", gyroData);
  });
  server.on("/reset", HTTP_GET, [](AsyncWebServerRequest *request){
    gyroX=0;
    gyroY=0;
    gyroZ=0;
    request->send(200, "text/plain", "OK");
  });
  server.begin();
}

void loop()
{
  // Tu lógica principal aquí
  //Serial.println(getGyroReadings());
  delay(10);}
```