UNIVERSIDAD POLITÉCNICA SALESIANA

SEDE CUENCA

CARRERA DE ELECTRÓNICA Y AUTOMATIZACIÓN

DESARROLLO DE UN SISTEMA DE MONITOREO DE INCLINACIÓN DE UN VEHÍCULO CON SISTEMA DE ENTRETENIMIENTO BASADO EN ANDROID

Trabajo de titulación previo a la obtención del título de Ingeniero en Electrónica

AUTOR: JONNATHAN FERNANDO RODAS MERCHÁN

TUTOR: ING. MARCELO ESTEBAN FLORES VÁZQUEZ, Mst.

Cuenca - Ecuador

2023

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Yo, Jonnathan Fernando Rodas Merchán con documento de identificación N° 0105777205, manifiesto que:

Soy el autor y responsable del presente trabajo; y, autorizo 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, 17 de agosto del 2023

Atentamente,

Jonnathan Fernando Rodas Merch§n

0105777205

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

Yo, Jonnathan Fernando Rodas Merchán con documento de identificación N° 0105777205, expreso mi voluntad y por medio del presente documento cedo a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que soy el autor del Proyecto técnico: "Desarrollo de un sistema de monitoreo de inclinación de un vehículo con sistema de entretenimiento basado en Android", el cual ha sido desarrollado para optar por el título de: Ingeniero en Electrónica, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

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

Cuenca, 17 de agosto del 2023

Atentamente,

Jonnathan Fernando Rodas Merch§n

0105777205

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 SISTEMA DE MONITOREO DE INCLINACIÓN DE UN VEHÍCULO CON SISTEMA DE ENTRETENIMIENTO BASADO EN ANDROID, realizado por Jonnathan Fernando Rodas Merchán con documento de identificación N° 0105777205, 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, 17 de agosto del 2023

Atentamente,

Ing. Marcelo Esteban Flores Vázquez, Mst.
0102408978

AGRADECIMIENTOS

El desarrollo de este proyecto fue de gran importancia para mi formación académica, ya que me permitió aprender y desarrollar habilidades en cuanto a la realización del mismo, es por ello que agradezco de manera especial a:

A mi familia por la fe y el apoyo infinito.

A mis compañeros quienes me supieron apoyar en los momentos de inquietud y tristezas.

Finalmente quiero expresar el más sincero agradecimiento a mi tutor Ing. Marcelo Flores Vazquez Mst., quien me supo impartir sus conocimientos con mucha dedicación y motivación durante este tiempo de trabajo, incentivándome a realizar futuros proyectos en el ejercicio profesional.

Jonnathan Fernando Rodas Merchán

DEDICATORIAS

El presente proyecto va dedicado a Dios por brindarme sabiduría y salud para cumplir con mis objetivos de vida, de igual manera, de todo corazón a las personas que colaboraron con el desarrollo de mi trabajo de titulación, pero especialmente a toda mifamilia, mis padres Fernando y Katherine quienes me han sabido apoyar con su amory paciencia, les agradezco por todo el trabajo que realizan a diario para que pueda culminar con mis estudios y logre ser un profesional, de igual manera para mi hermanaMayra que me ha motivado en los momentos duros.

A mis amigos con quienes compartí conocimientos, alegrías y tristezas, durante los años de la carrera y me supieron apoyar para que este sueño se haga realidad.

Jonnathan Fernando Rodas Merchán

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

RESUMEN

El presente trabajo está fundamentado en el desarrollo de un proyecto técnico orientado al monitoreo de inclinación de un vehículo con sistema de entretenimiento basado en Android, el desarrollo de este sistema nos permite monitorear continuamente la inclinación para informar de manera pertinente a los conductores y pasajeros en tiempo real si existe riesgo de vuelco, en complemento con el dispositivo se encargará de la transmisión de los datos adquiridos a través de los sensores hacia la aplicación Android, mediante la tecnología Bluetooth, mediante el uso del módulo ESP32.

La aplicación se encuentra perfectamente adaptada e integrada en el sistema de entretenimiento vehicular, permitiendo así una presentación en tiempo real e interacción optima con los datos recopilados, volviendo a este prototipo perfecto en la detección de dos variables importantes al momento de realizar una conducción todoterreno o por carretera, la inclinación transversal y la capacidad de ascenso del vehículo. Para lograr esto, se ha seleccionado 1 tipo de sensor para cada variable (mpu6050) empleando el protocolo de comunicación I2C para los sensores.

Describiendo la aplicación, la misma constará con una interfaz de fácil acceso e intuitiva, diseñada en Flutter (marco de código abierto compatible con el desarrollo de aplicaciones móviles), con el fin de brindar una experiencia fluida y agradable para cada uno de los usuarios.

Como parte del diseño se implementó un protector plástico el cual cubrirá los componentes electrónicos empleados, y permitirá la correcta ubicación dentro del vehículo. Sumado a este proceso, mediante las pruebas de campo se logró evaluar el

desempeño y la eficiencia del proyecto en condiciones reales y aplicadas, obteniendo así un dispositivo apto y funcional para la detección de la inclinación del vehículo.

INTRODUCCIÓN

El desarrollo del sistema de monitoreo de inclinación de vehículos es una tecnología útil en el mundo actual, este sistema tiene el potencial de aumentar significativamente la seguridad del vehículo. La inclinación de un vehículo se puede monitorear continuamente para informar a los conductores y pasajeros en tiempo real si existe riesgo de vuelco, esta advertencia puede ser útil para prevenir accidentes y reducir lesiones y muertes, además de ayudar con el mantenimiento del vehículo, a la vez puede señalar posibles problemas con la suspensión, las llantas u otras partes del vehículo al detectar y alertar al propietario de cualquier inclinación normal o anormal.

El sistema de monitoreo para detectar la inclinación del vehículo, puede ayudar a aumentar la eficiencia del combustible, lo cual puede dar al conductor información sobre los ajustes en la conducción para la eficiencia del combustible al monitorear continuamente el ángulo de inclinación del vehículo.

Actualmente el uso de dispositivos adaptados al vehículo mediante tecnología Android han revolucionado la experiencia de conducción, brindando un estilo diferente, teniendo al momento como limitantes la falta de integración y compatibilidad con aplicaciones que permitan monitorear diferentes valores, lo que disminuye la efectividad.

Por lo cual, a través de este proyecto se muestra la creación de un aplicativo Android de monitoreo para el sistema de entretenimiento propio del automóvil, mediante el cual, obtendremos la lectura en tiempo real del sensor de la inclinación transversal y la capacidad de ascenso, brindando una visualización cognoscible para el conductor y asegurando los beneficios antes descritos, transformando al sistema en una herramienta eficiente y accesible que mejore significativamente la seguridad,

concibiendo un impacto positivo en la gestión eficiente del automotor y la seguridad de los ocupantes.

ANTECEDENTES DEL PROBLEMA DE ESTUDIO

Los sensores utilizados para monitorear la aceleración y la rotación, respectivamente, incluyen acelerómetros y giroscopios. Un acelerómetro detecta la aceleración lineal de un objeto en movimiento, mientras que un giroscopio mide la velocidad angular o de rotación de un objeto en movimiento. Ambos sensores se emplean ampliamente en tecnología de la información y aplicaciones comunes, como dispositivos móviles, sistemas de navegación, controles de estabilización para robots y drones, entre otras cosas.

El desarrollo de dispositivos para medir la inclinación vehicular se ha utilizado en diversas aplicaciones a lo largo del tiempo, brindando a conductores y pasajeros de automotores todoterreno información verás en el ámbito de la conducción off-road, permitiéndoles cambiar el curso de conducción y disminuyendo el riesgo de accidentes y muertes. Al desarrollar un sistema de monitoreo de inclinación para vehículos con sistemas de entretenimiento basados en Android, los acelerómetros y giroscopios son sensores cruciales para medir la orientación e inclinación del vehículo en tiempo real.

Numerosos estudios muestran los riesgos asociados a la conducción y pérdida de la posición normal de los vehículos, además de plantear varias soluciones para aumentar la seguridad. Sin embargo, a medida que el desarrollo tecnológico surge, la necesidad de fabricar dispositivos más precisos se hace evidente, dispositivos con mayor fiabilidad, altamente adaptables y con el poder de monitorear los cambios angulares que ocasionan dichos incidentes automovilísticos.

Es así que, (Komarizadehasl et al., 2022), en su trabajo describe un medidor de ángulo confiable adaptable (LARA) de bajo costo, que combina cinco giroscopios y cinco acelerómetros para determinar la inclinación, LARA utiliza software comercial gratuito para la recopilación de datos además de la tecnología de microcontroladores

basada en Internet de las cosas (IoT), para permitir la transmisión inalámbrica de datos. Los resultados del dispositivo creado se comparan con los producidos mediante cálculos numéricos de pendiente y un inclinómetro comercial, siendo el adecuado para la detección de daños estructurales en puentes que utilizan inclinómetros debido a su bajo costo y alta precisión.

De igual manera, (D'Mello et al., 2022), nos describe en su trabajo al inclinómetro como una herramienta que mide la orientación y la inclinación de un objeto en el espacio utilizando giroscopios y acelerómetros de sistemas microelectromecánicos (MEMS).

Estos sensores pueden reconocer cambios en la aceleración y la velocidad angular del objeto y traducirlos en señales eléctricas que pueden procesarse y estudiarse. El inclinómetro integra observaciones de muchos sensores para reducir los errores de medición y aumentar la precisión de lectura. Habla acerca de los sistemas modernos de monitoreo de alineación de ruedas utilizan una amplia gama de técnicas de visión por computadora, costosas cámaras de alta gama, operadores profesionales, objetos cuidadosamente creados y bucles computacionales masivos dando un enfoque simple, conveniente y asequible.

El sistema de alineación de ruedas basado en IoT sugerido se puso a prueba en un automóvil y los resultados se compararon con el método convencional, que arrojó resultados satisfactorios. La técnica sugerida se puede utilizar en aplicaciones del mundo real y promete ser un reemplazo viable para las técnicas actuales de evaluación de alineación de ruedas. (Irsan et al., 2022), describe un prototipo con el fin de prevenir accidentes, que podrían resultar en muchas víctimas e incluso muertes, implementado con éxito un sistema de transporte real, en particular para los autobuses, el cual consta

de un dispositivo portátil y un sistema integrado con varios sensores conectados por Bluetooth, con esto se podría evitar posibles desastres.

La tecnología portátil está diseñada para controlar la frecuencia cardíaca del conductor e informarle si experimenta privación del sueño. El sistema integrado se compone de un Smart Box montado en el bus, un Sistema de Posicionamiento Global (GPS), acelerómetros y giroscopios.

Podemos ver la gama amplia de trabajos que se pueden desarrollar mediante la creación de sistemas adaptables a tecnología Android, si bien la implementación de un sistema de monitoreo de inclinación de un vehículo con sistema de entretenimiento basado en Android es más específico, con lo antes citado podemos observar cómo se ha desarrollado a través del tiempo y la era tecnológica.

JUSTIFICACIÓN (IMPORTANCIA Y ALCANCES)

En la actualidad existen sistemas de entretenimiento basado en Android para automóviles, sin embargo, los mismos carecen de sensores que permitan el monitoreo de inclinación en vehículos todo terreno, normalmente se implementan en los dispositivos móviles y no son compatibles con los sistemas de entretenimiento Android que contienen los vehículos.

Teniendo en cuenta el aumento considerable de producción de vehículos, así como el uso de los mismos existe un incremento de instrumentos de monitoreo que permitan garantizar la seguridad y el rendimiento óptimo del automotor. Podemos decir que la aplicación en el teléfono no es útil, debido a que no está vinculado al vehículo, por lo tanto, la desventaja es un agotamiento de batería, el teléfono no está integrado al automóvil y por ende el movimiento y vibración del vehículo afectan considerablemente el sistema de sujeción del teléfono móvil.

Es de esta forma que la necesidad de un sistema accesible e integrado al sistema de entretenimiento propio del vehículo que monitoree en tiempo real la inclinación del vehículo y proporcione de esta manera una visión alertante al conductor de una manera clara y comprensible, ajustándolo a su trayecto en consecuencia.

En concreto el diseño y aplicación de este proyecto evitará daños en el motor y asegura la integridad durante la conducción, además el desarrollo de un sistema basado en Android diseñado específicamente para el sistema de entretenimiento propio del automotor, será la solución absoluta por la situación actual y la necesidad.

OBJETIVOS

OBJETIVO GENERAL

- Desarrollar un sistema de inclinación para vehículo todoterrenocon sistema de entretenimiento basado en Android.

OBJETIVOS ESPECÍFICO

- Realizar un estudio de los sensores de la inclinación transversal y la capacidad de ascenso del vehículo disponibles en el mercado, a través de la revisión de los manuales de proveedores de sensores y bibliografía a fin, permitiéndonos seleccionar el dispositivo adecuado.
- Diseñar y construir un dispositivo electrónico que permita a la aplicación móvil leer los datos proporcionados en tiempo real de los sensores de inclinación transversal y la capacidad de ascenso, con inmunidad a la interferencia electromagnética presente en el vehículo, a fin de obtener un dispositivo fiable.
- Desarrollar una aplicación móvil que conjugue el sistema de entretenimiento basado en Android, permitiendo establecer la comunicación con el hardware a desarrollar y visualizar la información proporcionada, haciendo uso de herramientas informáticas especializadas para la implementación.
- Analizar las limitantes operacionales mediante la puesta a prueba, para valorar el desempeño y posibles ajustes.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

Este capítulo contiene el fundamento científico con la información pertinente acerca del proyecto en su totalidad, el mismo otorgará las bases necesarias para su cumplimiento eficiente.

1.1 FUNDAMENTOS TEÓRICOS

1.1.1 SENSOR DE INCLINACIÓN

Los sensores de inclinación son herramientas que te permiten medir el ángulo o inclinación lateral de un objeto con respecto al suelo. Estos sensores se utilizan en muchos campos, incluida la industria automotriz para medir la inclinación del vehículo.

En el campo de la ingeniería automotriz, los sensores de inclinación se utilizan para rastrear el ángulo de inclinación del vehículo y proporcionar información al sistema de control de estabilidad del vehículo para garantizar la seguridad de los ocupantes (bibawy, m. M., shaaban, a. M., & al-qutayri, m. A., 2019).

1.1.2 ACELERÓMETRO

Es uno de los tipos de sensores de inclinación que más se utilizan en los vehículos. Los acelerómetros miden la aceleración del vehículo y la inclinación del vehículo se puede calcular utilizando esa información (makki, i., gao, x., wang, y., & el-sheimy, n., 2021). Además, la asequibilidad y la portabilidad de los acelerómetros los convierten en una opción popular para medir la inclinación del vehículo.

1.1.3 GIROSCOPIO

Es otro tipo de sensor de inclinación utilizado en automóviles. Los giroscopios miden la velocidad angular del vehículo y, a partir de esta información, se puede calcular la inclinación del vehículo (yang, c., shi, x., & zhang, b., 2018). Aunque más precisos que los aceleradores, los giroscopios son más caros y menos utilizados en la industria automotriz.

Además, los sensores de inclinación se pueden montar en varios lugares del vehículo. Por ejemplo, los sensores se pueden montar en la suspensión, el carrusel o la banda de rodadura del vehículo (bibawy, m. M., shaaban, a. M., & al-qutayri, m. A., 2019). Es fundamental elegir la ubicación adecuada para el sensor en función de los requisitos únicos del proyecto porque la ubicación del sensor puede afectar la precisión con la que se realiza la medición de la inclinación.

En resumen, los sensores de inclinación son dispositivos cruciales para monitorear la inclinación del vehículo y garantizar la seguridad del conductor y los pasajeros. Los acelerómetros y los giroscopios son dos de los tipos de sensores de inclinación más utilizados en la industria automotriz, y elegir la ubicación adecuada para el sensor es crucial para garantizar la precisión.

1.1.4 SENSOR MPU6050

Es una unidad de medición inercial o imu (inertial measurment units) de 6 grados de libertad (dof), esta combina un acelerómetro de 3 ejes y un giroscopio de 3 ejes. Sensor utilizado especialmente en navegación, goniometría, estabilización, etc.

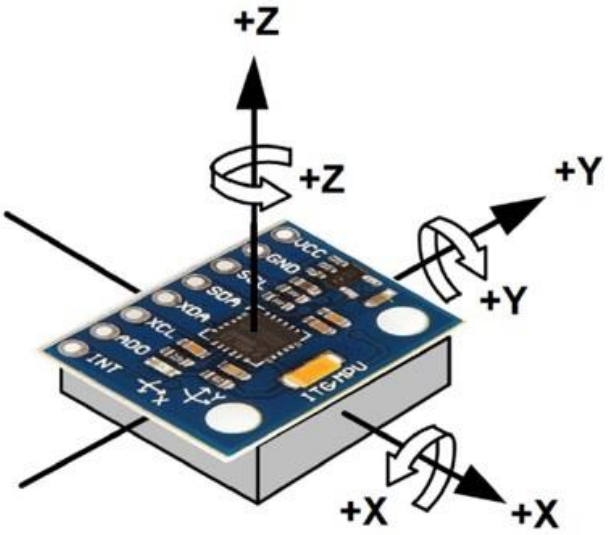

FIGURA 1.1 SENSOR MPU6050

Fuente *Naylamp Mechatronics*

1.1.5 SISTEMAS DE ENTRETENIMIENTO (ANDROID)

Debido a que ofrecen una amplia gama de características y funciones que permiten a los conductores y pasajeros disfrutar de una experiencia de entretenimiento dentro del vehículo, los sistemas de entretenimiento basados en Android son cada vez más comunes en los vehículos actuales. Estos sistemas se ejecutan en una plataforma Android, lo que les permite aprovechar todas las funciones y aplicaciones que ofrece la plataforma.

Una de las principales ventajas de los sistemas de entretenimiento basados en Android es su capacidad para conectarse a internet a través de una conexión de datos móviles o una conexión Wi-Fi. Esto permite a los usuarios acceder a una amplia gama de servicios en línea, como música, video y servicios de transmisión de aplicaciones de navegación. Además, estos sistemas pueden integrarse con sistemas de asistencia al conductor, como cámaras de marcha atrás, sensores de estacionamiento y asistentes

de voz, lo que los convierte en un componente esencial de la experiencia de conducción.

Según (álvarez, j., cárcel, a., & sebastián, j. M., 2020), los sistemas de entretenimiento para automóviles basados en Android incluyen características como la capacidad de conectar dispositivos móviles a través de Bluetooth y USB, control devoz y compatibilidad con el sistema de navegación GPS. Además, estos sistemas permiten a los usuarios acceder a aplicaciones de redes sociales y servicios detransmisión de música como Spotify y Pandora.

Por otro lado, según (khanna, p., sharma, m., & bhakar, s., 2020), los sistemas de entretenimiento basados en Android también ofrecen la posibilidad de adaptar la interfaz de usuario y las aplicaciones a las preferencias del usuario. Esto hace posible que los usuarios accedan a sus aplicaciones y ajustes preferidos de forma rápida y sencilla, lo que mejora la experiencia del usuario en general.

En conclusión, los sistemas de entretenimiento basados en Android se han convertido en un componente esencial de la experiencia de conducción, porque ofrecen una amplia gama de funciones que permiten a los conductores y pasajeros disfrutar de una experiencia agradable y conectada dentro del vehículo.

Actualmente, el sistema operativo Android, diseñado para equipos móviles, puede instalarse no solo en dichos equipos, sino que también en otros dispositivos, como tabletas, televisiones, discos multimedia, etc.

Está basado en sistema operativo Linux, el cual se conoce como el núcleo del actual sistema operativo abierto, de multiplataforma y gratis. Permite realizar aplicaciones empleando una variedad de Java llamada DalviK la cual nos proporciona

la interfaz que se necesita para facilitar el desarrollo de las aplicaciones y su acceso a las distintas funciones del equipo. Cabe recalcar que es totalmente libre.

En el año 2007 se realiza el primer lanzamiento de Android, a la par con SDK (Software Development Kit), para la gran comunidad de programadores a nivel global, puedan realizar sus aplicaciones en este nuevo sistema operativo.

Como se mencionó anteriormente, el corazón de Android es Linux, el cual nos permitirá realizar la gestión de la red Wi-Fi, la pantalla, la cámara, el teclado, etc. No se tendrá un acceso directo a esta, en cambio este nivel utiliza una gran variedad de librerías que se encuentran en niveles superiores y nos separan del hardware. Además, la programación del equipo se la realiza utilizando el Framework de dicha aplicación, en donde se gestionan el ciclo de vida y sus recursos, además en la última capa se encuentran las aplicaciones de usuario.

Se debe tener en cuenta algunos conceptos claves antes de empezar a realizar la aplicación, como por ejemplo los servicios, las Actividades, Intens, y proveedores de contenido.

- ***Actividades***

Estas correspondes a las pantallas, es decir, se tendrá la misma cantidad de actividades como pantallas tenga la aplicación creada.

Cada actividad se responsabiliza de mantenerse en su estado, para que de esta forma se puedan integrar en el ciclo de vida de esta. Las interfaces se pueden crear de dos formas, la primera es usando el código Java, y la segunda forma es usar un fichero XML para describirla como una página HTML. Siendo la más factible y sencilla.

- ***Intens***

Los Intens son mecanismos para describir acciones, como por ejemplo enviar un e-mail, o tomar una foto. Este es un mecanismo flexible, ya que, mediante él, se puede crear una actividad para envío de SMS y registrarla de forma que corresponda al Inten adecuado.

- ***Servicios***

Los Servicios son tareas que se ejecutan en segundo plano. Por ejemplo, hacemos sonar un MP3, y a su vez, deseamos poder seguir utilizando el teléfono. Mediante otro servicio, se puede realizar esta acción y podemos usar otras aplicaciones con normalidad.

- ***Proveedores de Contenido***

Los Proveedores almacenan datos compartidos por la mayoría de las aplicaciones. Para acceder a estos datos se lo hace mediante la API para cada proveedor de contenido.

- ***Tiempo de ejecución de Android***

Cada aplicación utiliza sus propias instancias de Android Runtime (ART) para llevar a cabo sus procesos. El ART está diseñado para ejecutar muchas máquinas virtuales en dispositivos con poca memoria mediante la ejecución de archivos DEX, un tipo de código basado en bytes creado específicamente para Android y optimizado para ocupar la menor cantidad de memoria.

Algunas de las funciones clave incluyen la compilación justo a tiempo (JIT) y antes de tiempo (AOT), así como la recolección optimizada de elementos de desecho (GC).

- ***Bibliotecas C/C++ nativas***

Numerosos componentes y servicios básicos del sistema Android, como ART y HAL, se construyen utilizando código nativo que requiere bibliotecas C y C++ nativas. La plataforma Android ofrece API de banco de trabajo de Java para exponer la funcionalidad de algunas de estas bibliotecas nativas.

1.1.6 APLICACIONES EN ANDROID

Una de las plataformas más populares en el mundo para el desarrollo de aplicaciones móviles es Android, la cual fue desarrollada por Google (pereira, m. A., rodrigues, m. P., & santos, r. L., 2018). La programación de Android se realiza en Java, aunque también es posible utilizar otros lenguajes de programación, como Kotlin (pereira, m. A., rodrigues, m. P., & santos, r. L., 2018). Además, Android tiene una colección de herramientas de desarrollo que facilitan la creación de aplicaciones, como el kit de desarrollo de Software (sdk) de Android y Android Studio, un entorno de desarrollo integrado (ide) que permite escribir, limpiar y probar aplicaciones, todo en una ubicación (rahman, m. M., akter, s., & hasan, m. M., 2020).

Se deben seguir una serie de pasos para la creación de una aplicación Android, que incluyen el diseño de la interfaz de usuario, la definición de la función de la aplicación, la implementación de la función y la prueba de la aplicación (pereira, m. A., rodrigues, m. P., & santos, r. L., 2018). La experiencia del usuario (ux) y la usabilidad de la aplicación deben tenerse en cuenta al diseñar la interfaz de usuario (rahman, m. M., akter, s., & hasan, m. M., 2020). Además, es crucial adherirse a las pautas de diseño de Google, que ofrecen sugerencias para desarrollar interfaces de usuario cohesivas y atractivas (pereira, m. A., rodrigues, m. P., & santos, r. L., 2018).

Se pueden utilizar diferentes técnicas de programación, como la programación orientada a objetos, la programación reactiva y la programación funcional, para desarrollar la funcionalidad de la aplicación (rahman, m. M., akter, s., & hasan, m. M., 2020). Además, las bibliotecas y los marcos se pueden utilizar para acelerar el proceso de desarrollo. Los ejemplos incluyen retrofit para conectarse a api, room para administrar bases de datos y glide para cargar imágenes (pereira, m. A., rodrigues, m. P., & santos, r. L., 2018).

La prueba de la aplicación es un paso crucial para garantizar que funciona correctamente y cumple con los requisitos necesarios. Para garantizar la compatibilidad, es crucial realizar pruebas unitarias y de integración, así como pruebas en varios dispositivos y versiones de Android (rahman, m. M., akter, s., & hasan, m. M., 2020).

1.1.6 JEEP GRAND CHEROKEE

El jeep grand cherokee es un vehículo utilitario deportivo (suv, por usar su nombre en inglés) fabricado por la empresa estadounidense jeep. Según la ficha técnica del fabricante, el peso del jeep grand cherokee variaba entre 1.860 y 2.303 kilogramos dependiendo de la versión y equipamiento (jeep, 2004).

El ángulo de inclinación es una medida fundamental para determinar la estabilidad de un vehículo en situaciones de conducción todoterreno o al circular por curvas cerradas. Según el mismo dibujo técnico, el jeep grand cherokee tiene un ángulo de inclinación de 32,9 grados en la versión laredo y de 33,3 grados en la versión overland (jeep, 2004).

Un punto teórico donde se concentra la masa de un objeto se conoce como el centro de gravedad. En el caso de un vehículo, el centro de gravedad afecta su estabilidad y maniobrabilidad. Según varios estudios, el centro de gravedad del jeep grand cherokee es relativamente alto como resultado de su diseño de suv, lo que puede reducir su estabilidad en situaciones de manejo difíciles (smith, d., jerina, k., & hecker, j., 2018).

Además, el jeep grand cherokee cuenta con un sistema de tracción en las cuatro ruedas (4wd) que brinda mayor control y tracción en terrenos irregulares. Este sistema es administrado por una caja de transferencia que distribuye uniformemente el torque entre los ejes delantero y trasero (jeep, 2004).

En conclusión, el jeep grand cherokee es un suv con un peso considerable, un ángulo de inclinación adecuado para situaciones extremas de conducción y un centro de gravedad algo elevado, todo lo cual puede afectar su estabilidad en determinadas circunstancias. Adicionalmente, cuenta con un sistema de tracción en las cuatro ruedas que permite un mejor desempeño en terrenos irregulares.

CAPÍTULO 2: MARCO METODOLÓGICO

En el presente capitulo se detalla la metodología desarrollada para la elaboración del dispositivo encargado de monitorear la inclinación del vehículo, el cual consta de cuatro fases en donde se detallará el proceso para alcanzar el objetivo principal del proyecto:

FASE 1

Para la realización del proyecto se empezó con una ardua exploración sobre los sensores presentes en el mercado actual para medir la inclinación, esto se logrará mediante el uso de las TICs, lo cual se específica a más profundidad en el estado de arte.

Los sensores de inclinación detectan variaciones en la orientación y rotación de un punto en específico, en su mayoría están diseñados para calcular el ángulo de un objeto con respecto a la fuerza de gravedad.

A continuación, se menciona algunos ejemplos:

- Inclinómetro capacitivo
- Inclinómetro MEMS
- Inclinómetro por conductividad
- Inclinómetro biaxiales

Considerando cada una de las características que contienen los sensores se llegó a la resolución de utilizar el sensor Inclinómetro MEMS MPU-6050.

1.1 SENSOR MPU-6050

Es una unidad de medición inercial o IMU (Inertial Measurment Units) de 6 grados de libertad (DOF), esta combina un acelerómetro y un giroscopio, cada uno de 3 ejes. Sensor utilizado especialmente en navegación, goniometría, estabilización, etc.

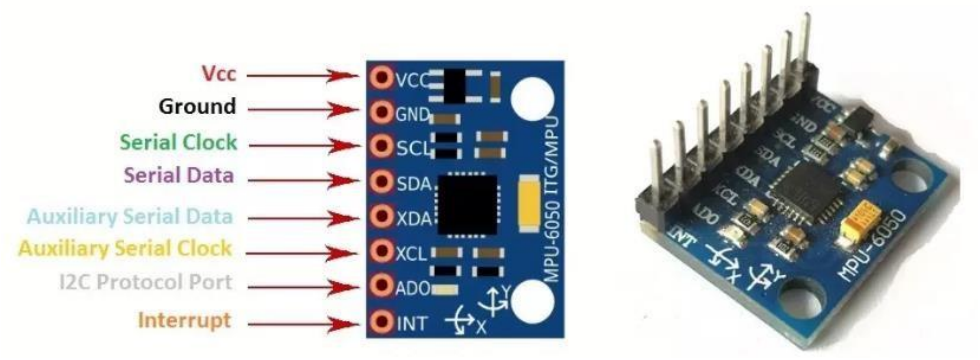

FIGURA 2.1 SENSOR MPU6050

Fuente *Naylamp Mechatronics*

- ESPECIFICACIONES
  - Salida digital de 6 ejes.
  - Giroscopio con sensibilidad de $\pm 250, \pm 500, \pm 1000, y \pm 2000 dps$
  - Acelerómetro con sensibilidad de $\pm 2g, \pm 4g, \pm 8g \; y \pm 16g$
  - Algoritmos embebidos para calibración
  - Sensor de temperatura digital
  - Entrada digital de video FSYNC
  - Interrupciones programables
  - Voltaje de alimentación: 2.37 a 3.46V
  - Voltaje lógico: $1.8V \pm 5\% \; o \; VDD$
  - 10000g tolerancia de aceleración máxima
  - Interfaz: I2C
  - Conversor AD:16 Bits (salida digital)
  - Regulador de voltaje en placa

FASE 2

En esta fase se ejecutará el diseño del circuito electrónico en base a los requisitos del proyecto, en los cuales se aplicará tecnología apropiada basándonos en la información obtenida en la fase 1.

Para llevar de manera competente las conexiones es necesario tomar en cuenta las especificaciones de los sensores a utilizar, es de este modo que se empleará el módulo ESP32, debido a que conlleva un módulo Bluetooth el cual nos permitirá emitir y admitir datos de una manera inalámbrica.

Por consiguiente, se complementará con el módulo DC-DC Convert, ya que permitirá alimentar el modulo ESP32 de una manera directa, haciendo uso de los 12 voltios que proporciona la batería del vehículo, transformándolos a 5 voltios que son necesarios para el módulo mencionado.

2.1 COMPUTADOR DE PLACA ÚNICA ESP32

La línea ESP32 de microcontroladores de bajo costo y bajo consumo de energía con Wi-Fi dual integrado y Bluetooth, creada y desarrollada por Espressif Systems, es un paso adelante para los ingenieros de automatización que no quieren atascarse en la teoría de RF y el diseño poco ortodoxo. Su bajo consumo de energía, numerosos entornos de desarrollo de código abierto y bibliotecas lo convierten en la opción ideal para desarrolladores de todo tipo.

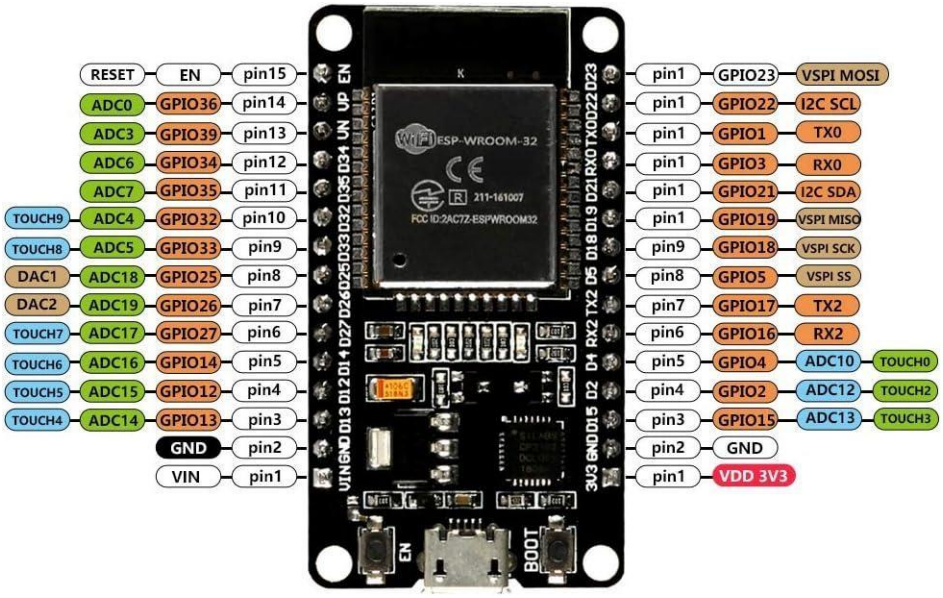

FIGURA 2.2 MODULO ESP32

Fuente *Espressif Systems*

El módulo ESP32 es una solución todo en uno Wi-Fi/Bluetooth certificada que proporciona no solo radio infrarroja sino también un procesador integrado con interfaces para conectarse a varios periféricos.

Dependiendo de los requisitos de su aplicación, los desarrolladores pueden elegir entre una variedad. El primer y más utilizado módulo ESP32 es el ESP32-WROOM-32D, que opera hasta 240 MHz. El módulo tiene una antena de radar compatible con CI, lo que facilita la implementación. Además, evita la necesidad de hardware adicional y el diseño complejo asociado con una antena conectada a IPEX. Sin embargo, existen varias buenas opciones de antena si se elige la opción de conector IPEX, como la W24P-U de Inventek Systems.

2.3 PLACA DE DESARROLLO ESP32 PARA CONTROL INDUSTRIAL

Cuando se diseña una pantalla que se usará en producción o cuando se coloca en una pantalla que se usará en un volumen "superior", los módulos ESP32 son una excelente opción. Los desarrolladores pueden usar una placa de desarrollo ESP32 para crear dispositivos de bajo volumen en las instalaciones de fabricación. Estos rótulos van desde rótulos de "inicio" muy básicos hasta rótulos sofisticados con procesadores secundarios y LCD. Hay varios que también son adecuados para aplicaciones de automatización industrial, con el supuesto de que la facilidad de desarrollo es un requisito crucial.

El kit de desarrollo ESP-WROVER-KIT de Espressif Systems ofrece una completa solución de desarrollo ESP32 con todo lo que los programadores necesitan para crear aplicaciones basadas en ESP32. Incluye un convertidor USB a serie FT2232HL FTDI que facilita la programación del módulo ESP32 sin necesidad de herramientas de programación personalizadas.

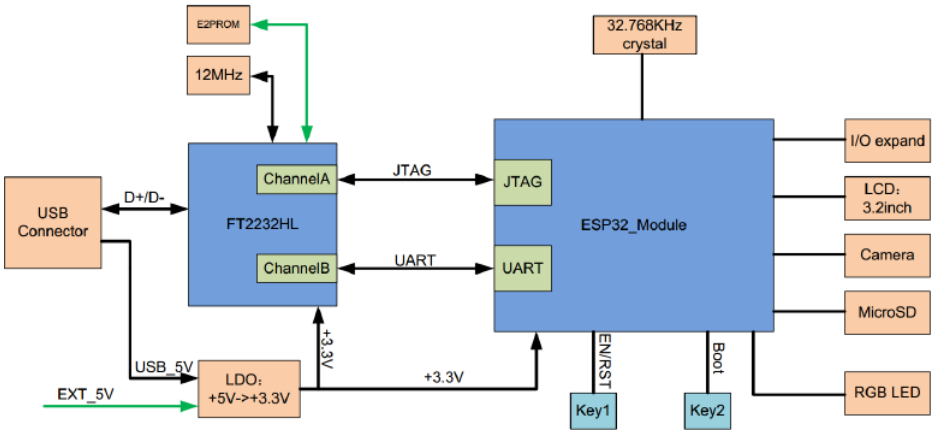

FIGURA 2.3 **ESP-WROVER-KIT**

Fuente: *Espressif Systems*

2.4 SELECCIÓN DE UN ENTORNO DE DESARROLLO ESP32

El módulo ESP32 es una plataforma de desarrollo de IoT (Internet of Things) que utiliza un microcontrolador de doble núcleo, este ofrece un despliegue de características y funciones. El ESP32 se ha vuelto tan popular que se puede desarrollar y programar en una variedad de entornos de desarrollo diferentes. Las herramientas de desarrollo más utilizadas incluyen:

- Marco de desarrollo de IoT (IDF) de Espressif.
- El entorno completo de desarrollo integrado (IDE) de Arduino.
- MicroPython.
- PlataformIO

Arduino IDE, al momento es la plataforma de desarrollo más popular de programación en microcontroladores, ESP32 es compatible con esta y permite la programación en lenguaje C++.

De la misma manera, ESP-IDF, posee el kit de desarrollo de software oficial del ESP32, lo cual permite el desarrollo de diferentes lenguajes de programación como C, C++, Python y JavaScript.

ESP32 es compatible con MicroPython, la cual es una implementación de Python 3 para microcontroladores, que permite la programación en este lenguaje de programación.

Por último, la PlatformIO, catalogado como un entorno de desarrollo integrado que soporta diferentes microcontroladores, incluyendo el ESP32, permite la programación en diferentes lenguajes de programación y ofrece una amplia variedad de librerías y herramientas.

Además, se puede hacer uso de varios lenguajes en programación como los siguientes:

C++: Lenguaje de programación más utilizado para programar el módulo ESP32. El mismo posee un alto nivel de programación que se utiliza para desarrollar

aplicaciones de sistemas, juegos y software de escritorio. Este un lenguaje orientado a objetos y es muy eficiente en términos de memoria y velocidad de ejecución.

Python: Es un lenguaje que se utiliza para desarrollar aplicaciones en la web, científicas y de escritorio. Es muy versátil, lo que lo hace ideal para principiantes, de igual manera es compatible con el módulo ESP32 y se puede utilizar para programar el dispositivo.

JavaScript: Lenguaje de programación utilizado para el desarrolla de aplicaciones web, se usa para programar el módulo ESP32 utilizando la plataforma NodeMCU, lo factible es que es fácil de aprender y compatible con la mayoría de los navegadores web.

Es importante detallar una breve comparación entre ESP32 y Arduino, permitiéndonos conocer la mejor opción en proyecto o el potencial de su combinación.

Potencia de procesamiento:

El ESP32 tiene una velocidad de reloj de 160 MHz, lo contrario al Arduino, el cual tiene una velocidad de reloj de 16 MHz. Esto significa que el ESP32 es mucho más rápido que el Arduino y puede manejar tareas más complejas.

Conectividad:

El ESP32 tiene conectividad WiFi y Bluetooth integrada, caso contrario el Arduino requiere módulos de expansión para estas funciones. El ESP32 también tiene más puertos de entrada/salida que el Arduino, lo que lo hace más versátil para proyectos que requieren muchas conexiones.

Consumo de energía:

El ESP32 tiene un modo de bajo consumo de energía, volviéndolo ideal para proyectos que requieren una larga duración de la batería. El Arduino no tiene esta función por lo que requiere más esfuerzos para lograr un bajo consumo de energía.

Facilidad de uso:

El ESP32 es una plataforma más avanzada, requiere de conocimientos técnicos. El Arduino posee un eso fácil, no solo por la simplicidad que posee, si no por la gran cantidad de documentación y ejemplos disponibles en línea.

Precio:

El Arduino es menos costoso que el ESP32. La diferencia de precio no es significativa para proyectos que requieren una alta potencia de procesamiento y conectividad.

FASE 3

En esta fase se desarrollará una aplicación móvil por medio de sistema basado en Android, los mismos harán uso de los dispositivos mencionados en la fase 2.

3.1 DESARROLLO DE APLICACIÓN MÓVIL

Flutter es una herramienta de gran ayuda para la creación de proyectos basados en Android, el mismo es gratuito y creado por Google en el año 2017, cabe recalcar que es de código abierto.

Este instrumento tiene la capacidad de crear aplicaciones compatibles tanto para Android e iOS, los cuales contienen sistemas operativos diferentes.

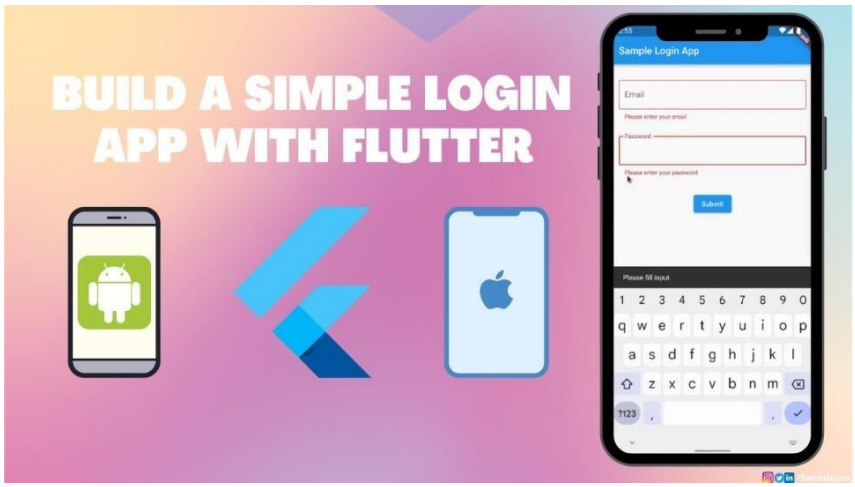

FIGURA 2.4 **FLUTTER**

Fuente: *Free Code Camp*

3.2 IDE ARDUINO INTERFAZ DE DESARROLLO DE LA COMUNICACIÓN DEL COMPUTADOR DE PLACA ÚNICA ESP32

Es una aplicación que fue escrita en el lenguaje de Java, la cual se utiliza para escribir y cargar programas en placas compatibles con Arduino, sin embargo, también la podemos usar con placas de desarrollo de diferentes proveedores haciendo uso de librerías adecuadas para su comunicación.

A continuación, se menciona características relevantes de la misma:

- Es un programa informático compuesto porherramientas de programación.
- Se puede dedicar solo a un lenguaje de programación oa varios.
- Aplicación multiplataforma (Windows, macOS, Linux).

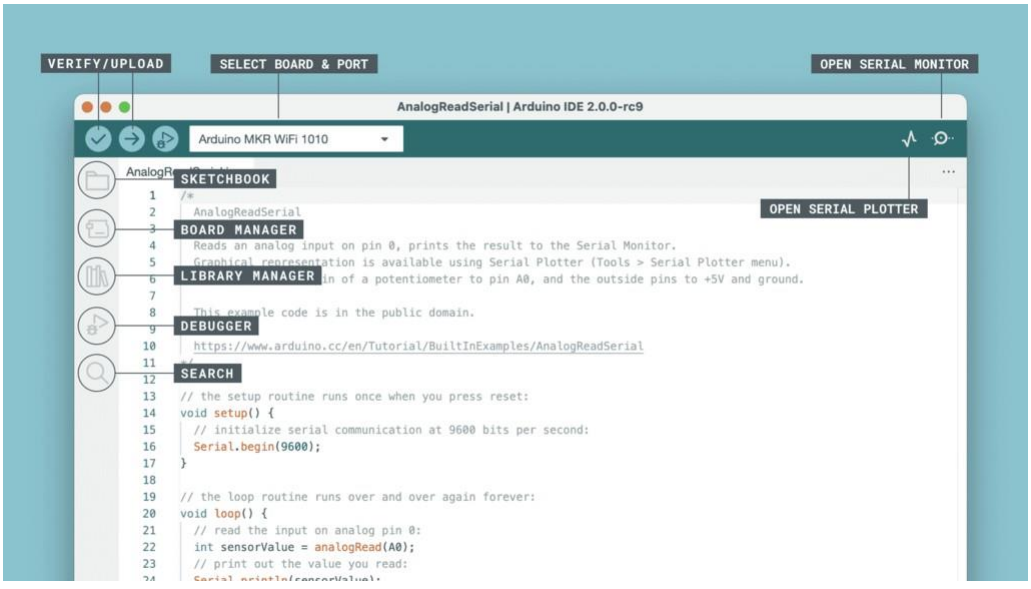

FIGURA 2.5 **INTERFAZ ARDUINO**

Fuente: *The Engineering Projects*

3.3 DISEÑO DE CARCASA 3D PARA PROTECCIÓN DE DISPOSITIVOS

Autodesk ofrece una serie de soluciones de software para el modelado 3D, la más famosa es AutoCAD. Es el software de diseño elegido por arquitectos, ingenieros y profesionales de la construcción. Si bien puede crear modelos 3D a partir de objetos sólidos, de superficie y de malla en AutoCAD, es mucho más avanzado en términos de capacidades 2D. Como resultado, Autodesk ha desarrollado otro software específico para aplicaciones de modelado 3D, como Fusion 360 o Tinkercad.

Fusion 360 planea convertirse en una solución completa al proporcionar herramientas CAD, así como CAM (producción de computadora) y cascada (uso técnico de computadoras). En el caso de los aditivos, este es un software más fácil de usar que AutoCAD, porque tiene como objetivo crear condiciones favorables para el desarrollo de productos.

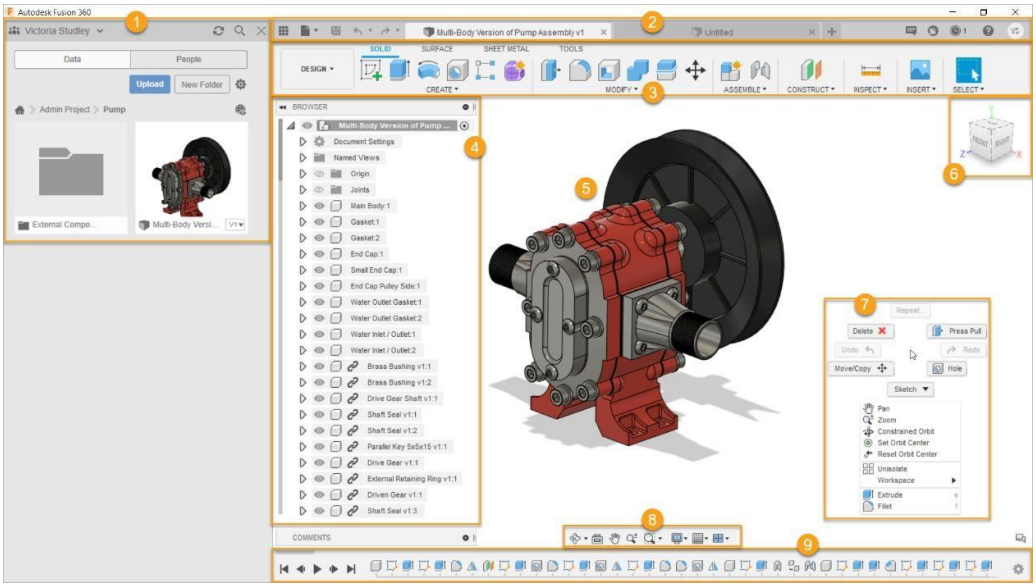

FIGURA 2.6 **INTERFAZ AUTODESK FUSION 360**

Fuente: *Fusion 360 Help*

FASE 4

En la fase actual procedemos a realizar las respectivas pruebas, para corroborar el funcionamiento correcto de los diferentes sistemas empleados.

4.1 VISUALIZACIÓN DE DATOS ADQUIRIDOS DEL SENSOR DEINCLINACIÓN MEDIANTE LA PLATAFORMA IDE ARDUINO

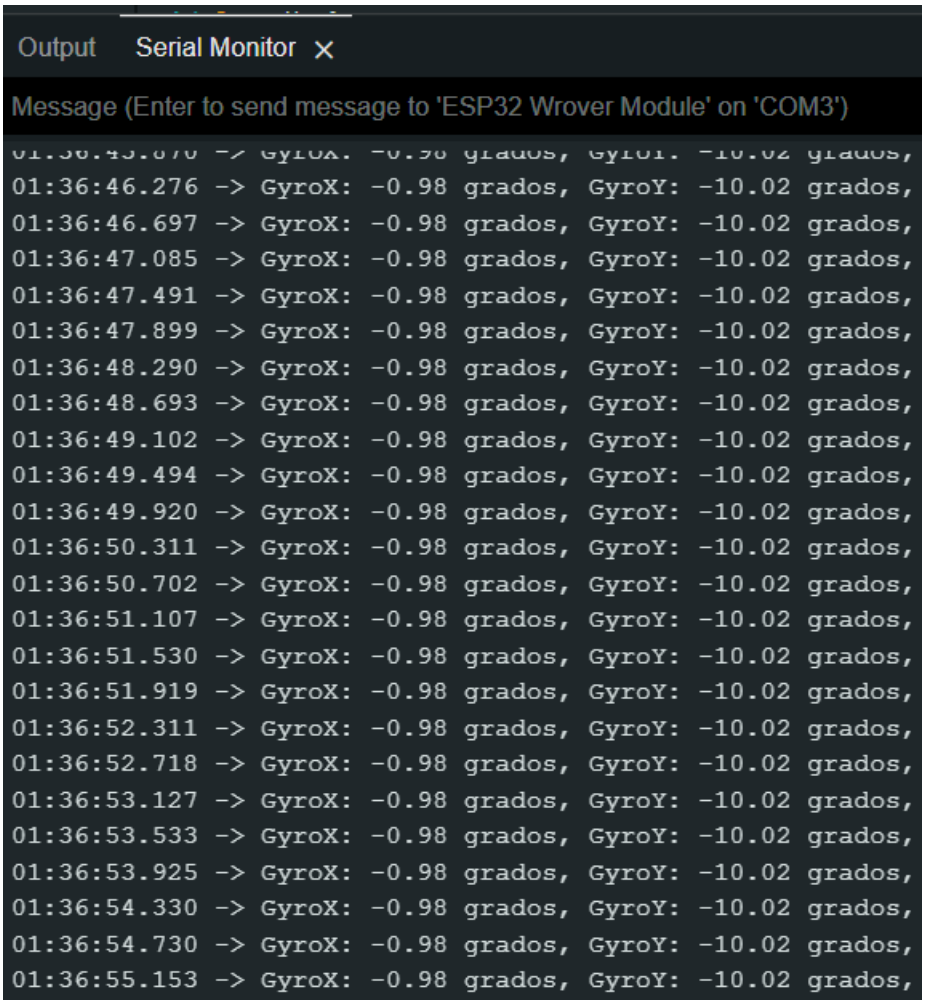

FIGURA 2.7 **VISUALIZACIÓN DE DATOS MONITOR SERIAL**

Fuente: *Autoría*

4.2 COMPROBACIÓN DE CORRECTA EMISIÓN Y VISUALIZACIÓNDE DATOS HACIENDO USO DE COMUNICACIÓN BLUETOOTH

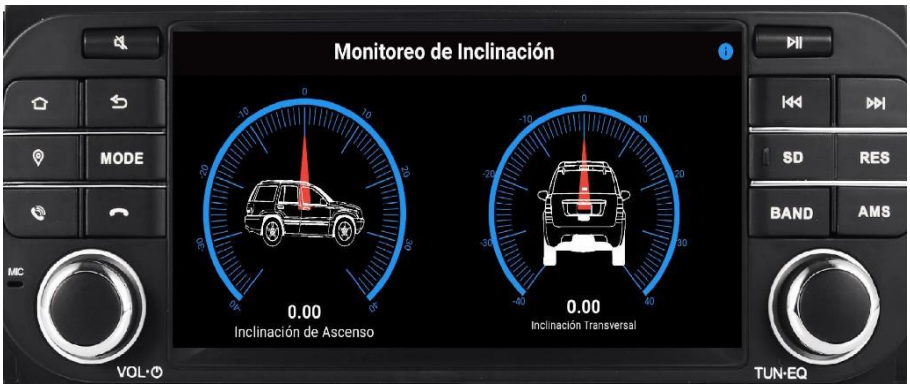

FIGURA 2.8 VISUALIZACIÓN DE DATOS EN LA APP

Fuente: *Autoría*

CAPÍTULO 3: IMPLEMENTACIÓN Y ANÁLISIS DE RESULTADOS

En este capítulo, se explicará en detalle el proceso de implementación del proyecto. Reporte de los pasos específicos tomados para alcanzar las metas descritas en los objetivos planteados.

3.1 DISEÑO DE LA APLICACIÓN BASADA EN ANDROID

Las herramientas de desarrollo de Visual Studio Code se utilizarán para crear la aplicación basada en Android, procedemos ha agregar extensiones de Flutter, que serán muy útiles para el desarrollo de la aplicación basada en Android.

Para comenzar a programar, primero creamos un nuevo proyecto de Flutter en el que agregamos las bibliotecas adecuadas para el desarrollo de aplicaciones en el directorio identificado como pubspec.yaml. Se agregó uno en la dependencia para ayudarnos a comunicarnos por Bluetooth y otro en sincronización para ayudarnos a visualizar.

```
# The following adds the Cupertino Icons font to your application
# Use with the CupertinoIcons class for iOS style icons.
cupertino_icons: ^1.0.2
flutter_bluetooth_serial: ^0.4.0
syncfusion_flutter_gauges: ^20.3.61
sensors_plus: ^1.2.2
animate_do: ^3.0.2
```

***FIGURA 3.1* IMPLEMENTACIÓN DE LIBRERÍAS**

Fuente: *VisualStudio*

A continuación, procedemos a crear variables con las cuales especificaremos los colores tanto de las letras como colores generales de los widgets.

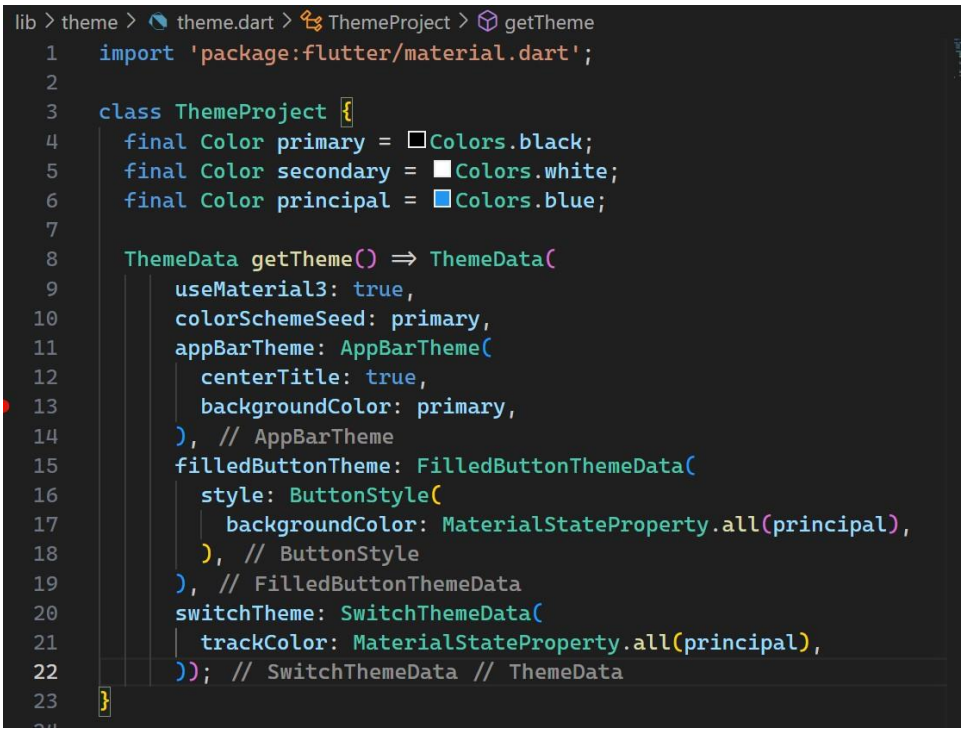

FIGURA 3.2 CREACIÓN DE VARIABLES PARA COLORES

Fuente: *VisualStudio*

Importamos todas las bibliotecas necesarias al archivo principal del programa principal. Aquí inicializamos los widgets de la misma manera que enviamos la aplicación para que se ejecute.

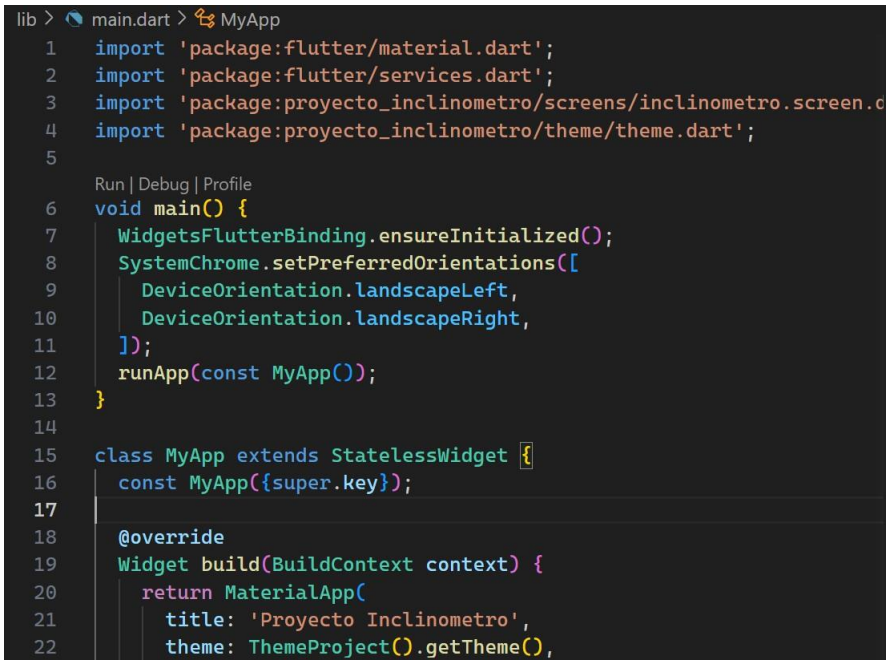

***FIGURA 3.3* COMPOSICIÓN DE MAIN**

Fuente: *VisualStudio*

Definimos las variables requeridas para la comunicación Bluetooth como las variables de diseño.

```dart
//! Variables del proyecto
double inclinacionX = 0.00;
double inclinacionY = 0.00;
double resetInclinacionX = 0.00;
double resetInclinacionY = 0.00;
// String direction = '';

double x = 0, y = 0, z = 0;
// String direction = "none";

@override
void initState() {
  super.initState();
  // Obtener el estado actual
  FlutterBluetoothSerial.instance.state.then((state) {
    setState(() {
      _estadoBluetooth = state;
    });
  });
```

FIGURA 3.4 **DECLARACIÓN DE VARIABLES A USAR POR CONEXIÓN BLUETOOTH**

Fuente: *VisualStudio*

Proceso de recepción de datos. Para ello se compararán los datos de cabecera enviados desde el módulo ESP32 y dependiendo de si la comparación es falsa o verdadera, los datos se mostrarán en diferentes widgets.

```dart
//**Escuchar la emision del dispositivo
_conexion!.input!.listen((Uint8List data) {
  try {
    //TODO 1  Informacion que viene del dispositivo bluet
    var text = ascii.decode(data);
    if (text.isNotEmpty && text.length > 2) {
      // ignore: avoid_print
      // print(text);
      var splitted = text.split(',');
      // ignore: avoid_print
      print('X ${splitted[0]}');
      // ignore: avoid_print
      print('Y ${splitted[1]}');
      // ignore: avoid_print
      // print('Z ${splitted[2]}');

      var x = double.parse(splitted[0]);
      var y = double.parse(splitted[1]);
      // var z = double.parse(splitted[2]);
```

FIGURA 3.5 **RECEPCIÓN DE DATOS**

Fuente: *VisualStudio*

3.2 DESARROLLO DEL PROGRAMA EN LA INTERFAZ IDE ARDUINO

Debemos tomar en cuenta que el ESP32 no esta instalado en el IDE de Arduino, por lo cual, nuestro primer a seguir es realizar la instalación de la extensión para que sea posible lograr su comunicación.

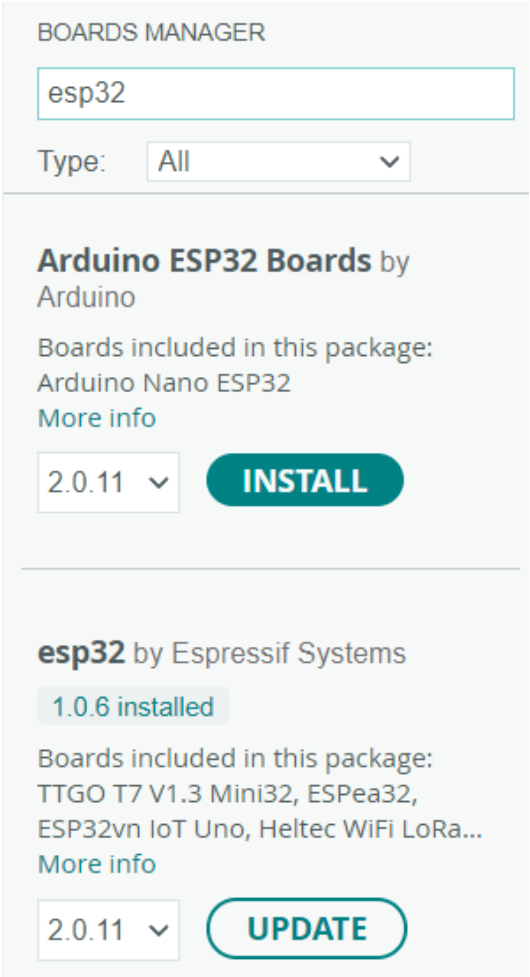

FIGURA 3.6. **EXTENSIÓN PARA COMUNICAR ESP32 CON IDE ARDUINO**

Fuente: *IDE Arduino*

Como siguiente paso, seleccionaremos el tipo de placa a usar, como sabemos existen diferentes variaciones del ESP32 sin embargo para nuestro haremos uso del ESP32 Dev Module.

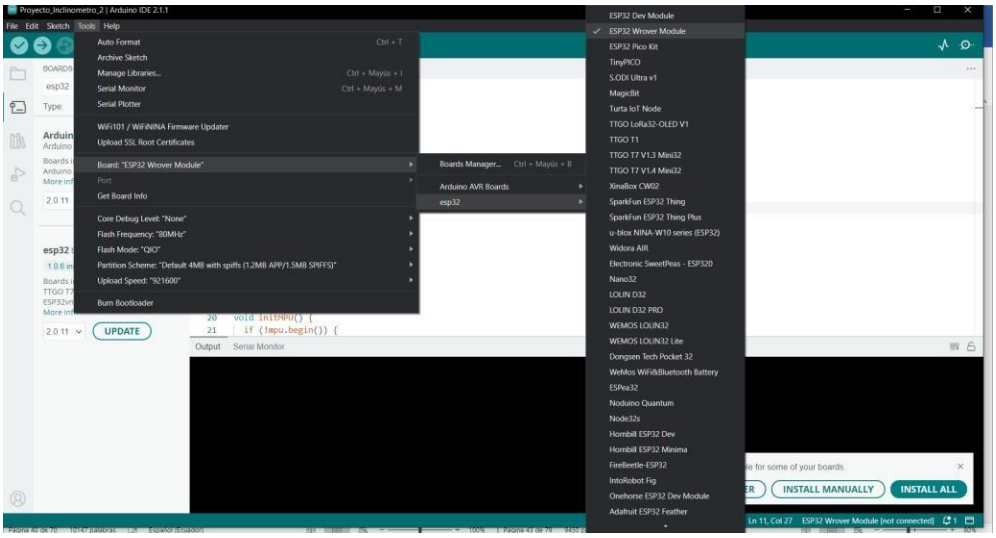

FIGURA 3.7 SELECCIÓN DE PLACA ESP32 DEV MODULE

Fuente: *IDE Arduino*

Procedemos a crear nuestro código para lo cual se usó varias librerías la cuales nos facilitara la recepción de datos otorgados por nuestro sensor, de igual manera una librería nos permite usar la comunicación Bluetooth.

```
#include <Arduino.h>
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>
#include <Wire.h>
#include <BluetoothSerial.h>

Adafruit_MPU6050 mpu;
```

FIGURA 3.8 INTEGRACIÓN DE LIBRERÍAS

Fuente: *IDE Arduino*

Como último paso, procedemos a crear las variables necesarias para realizar la recepción y emisión de datos, además de sus respectivas funciones.

```ino
void setup() {
  Serial.begin(115200);
  Wire.begin();
  initMPU();

  SerialBT.begin("ESP32_BT"); // Nombre del dispositivo Bluetooth

  Serial.println("Leyendo giroscopio...");
}

void loop() {
  if (Serial.available() > 0) {
    char input = Serial.read();
    if (input == 'A') {
      resetAngles = true;
    }
  }

  if (SerialBT.available()) {
    char input = SerialBT.read();
    if (input == 'A') {
      resetAngles = true;
    }
  }

  getGyroReadings();
  String gyroReadings =  String(gyroX)+ "," + String(gyroY) + "," + String(gyroZ);
  Serial.println(gyroReadings);
  SerialBT.println(gyroReadings);

  delay(10);
}
```

FIGURA 3.9. EMISIÓN Y RECEPCIÓN DE DATOS

Fuente: IDE Arduino

3.3 DESARROLLO DE CARCASA PLASTICA

Usando el software Fusión 360, creamos un boceto en la cuadrícula del mismo, guiados por las medidas obtenidas previamente. Este será el punto de partida para empezar a construir la carcasa de plástico del proyecto.

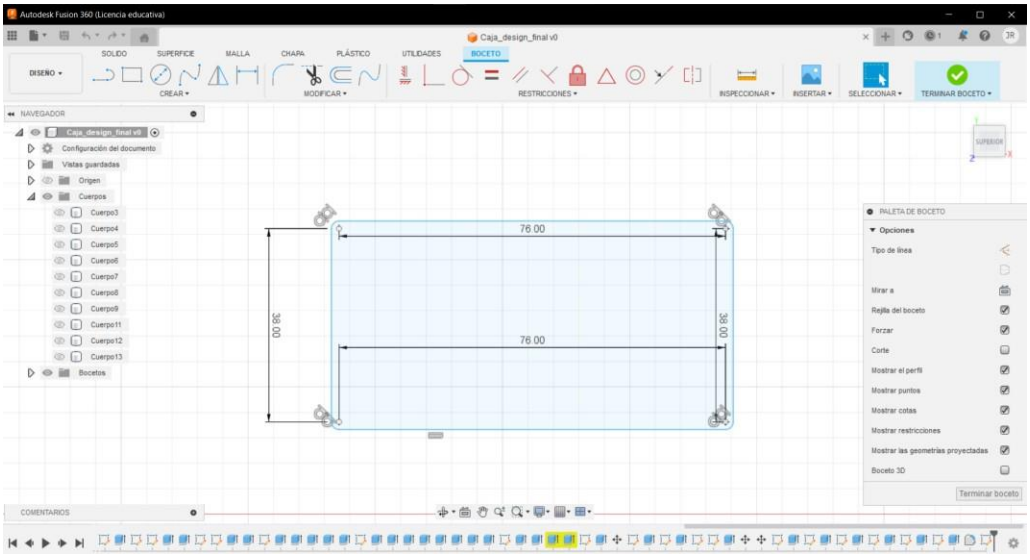

FIGURA 3.10 BOCETO DE LA CARCASA A DISEÑAR 1

Fuente: *Autoría*

Procedemos a hacer uso de la herramienta extruir en nuestro boceto la cual nos permitirá dar una altura adecuada para que nuestros componentes sean insertados en la misma.

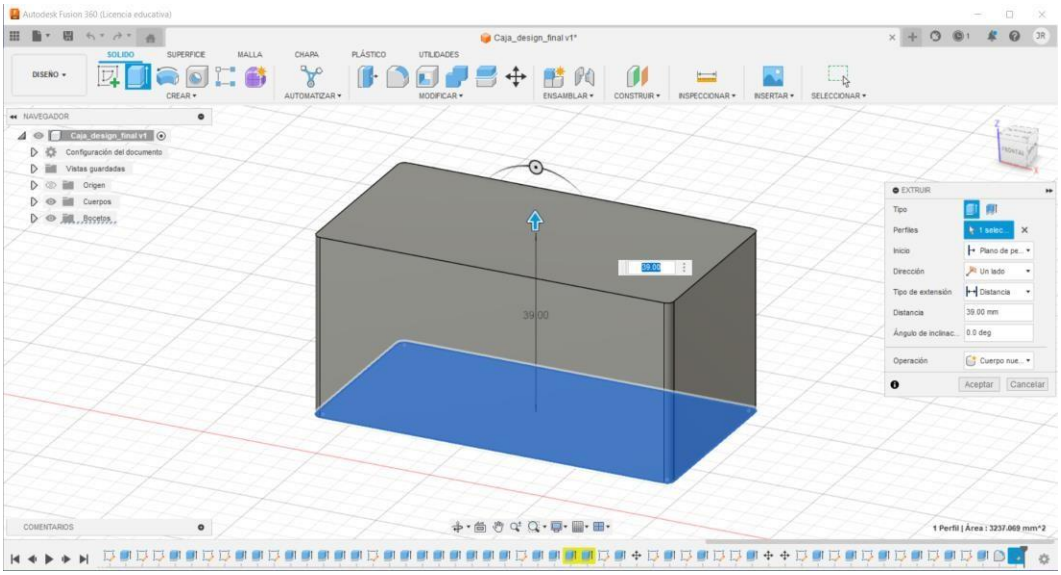

FIGURA 3.11 BOCETO DE LA CARCASA 2

Fuente: *Autoría*

Dado que obtendremos un volumen solido procedemos a realizar otro boceto con las medidas adecuadas para que nuestros sensores y módulos entren en su interior.

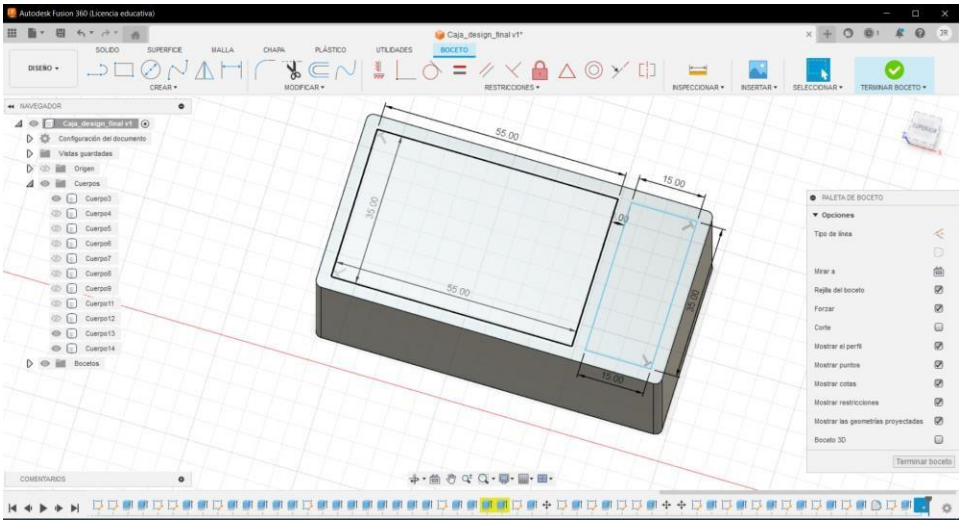

FIGURA 3.12 BOCETO DE LA CARCASA 3

Fuente: *Autoría*

Procedemos a hacer uso de la herramienta extruir para darle un fondo adecuado a cada sección correspondiente para los componentes que estarán en el interior de nuestra carcasa protectora.

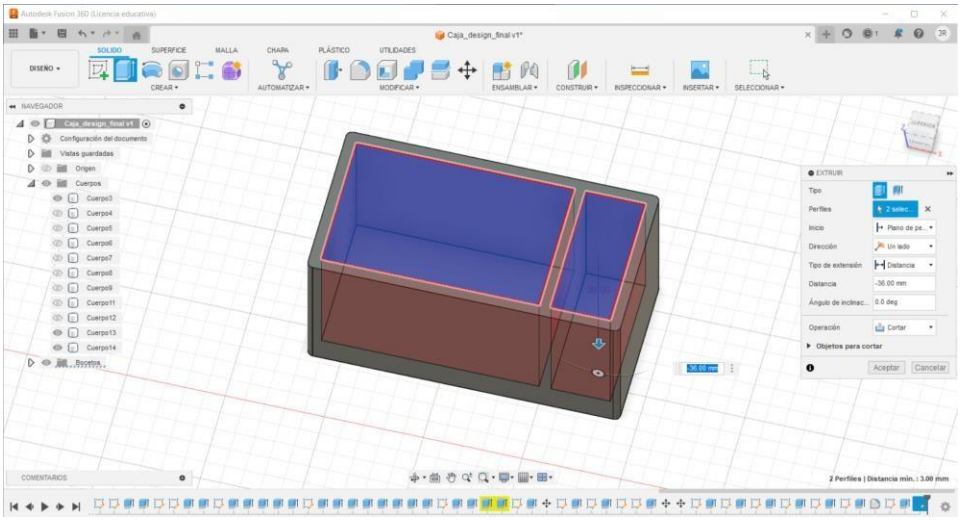

FIGURA 3.13 BOCETO DE LA CARCASA 4

Fuente: *Autoría*

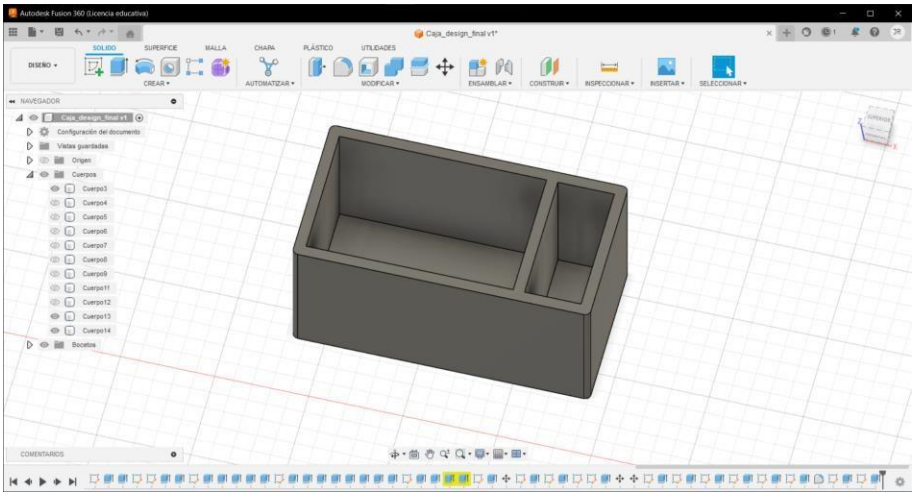

FIGURA 3.14 BOCETO DE LA CARCASA 5

Fuente: *Autoría*

Luego de esto diseñamos la tapa de nuestra carcasa plástica, guiándonos en las medidas anteriormente mostradas.

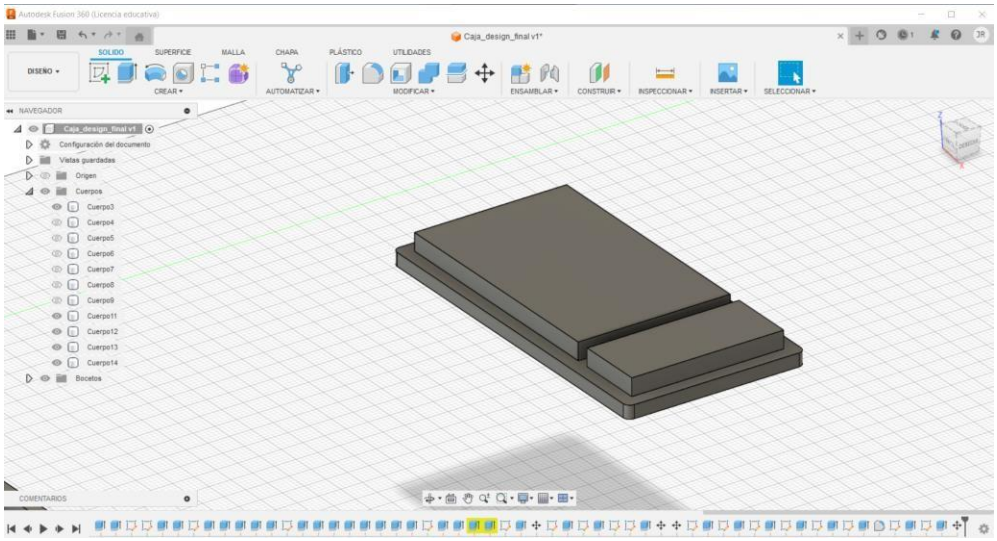

FIGURA 3.15 BOCETO DEL PROTECTOR DE CARCASA 6

Fuente: *Autoría*

Para proseguir con el diseño de nuestra carcasa protectora diseñaremos y extruiremos cada soporte debido para los componentes como módulos y sensores que estarán en el interior de la misma.

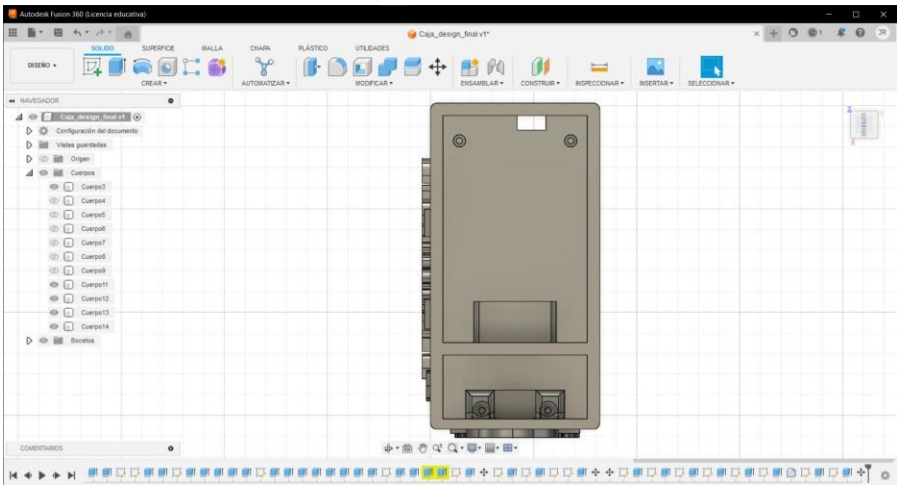

FIGURA 3.16 BOCETO DE LA CARCASA 7

Fuente: *Autoría*

A continuación, diseñaremos cada una de las tapas que encajaran en los orificios respectivamente de la comunicación y alimentación para los componentes de nuestra carcasa protectora.

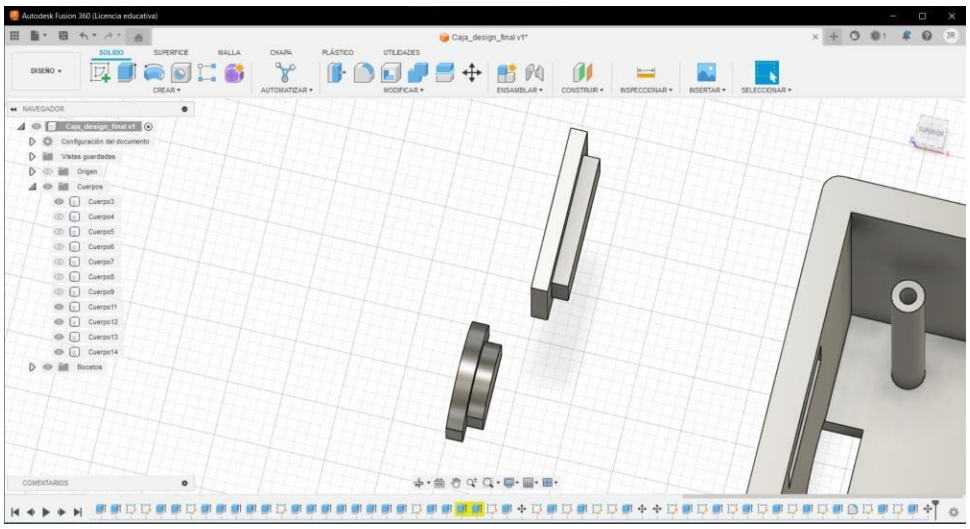

FIGURA 3.17 BOCETO DE LA CARCASA 8

Fuente: *Autoría*

Para concluir nuestro diseño hemos optado por adicionar los diseños de la marca del vehículo en el cual será implementado en la parte frontal de la misma, en la parte posterior colocamos el nombre respectivo del proyecto, además en el costado derecho se adicionado el logo de nuestra universidad para darle un toque más llamativo.

Todas estas adiciones se han implementado cuidadosamente, asegurando que el diseño de la carcasa de plástico sea estética y funcional en términos de fines del proyecto.

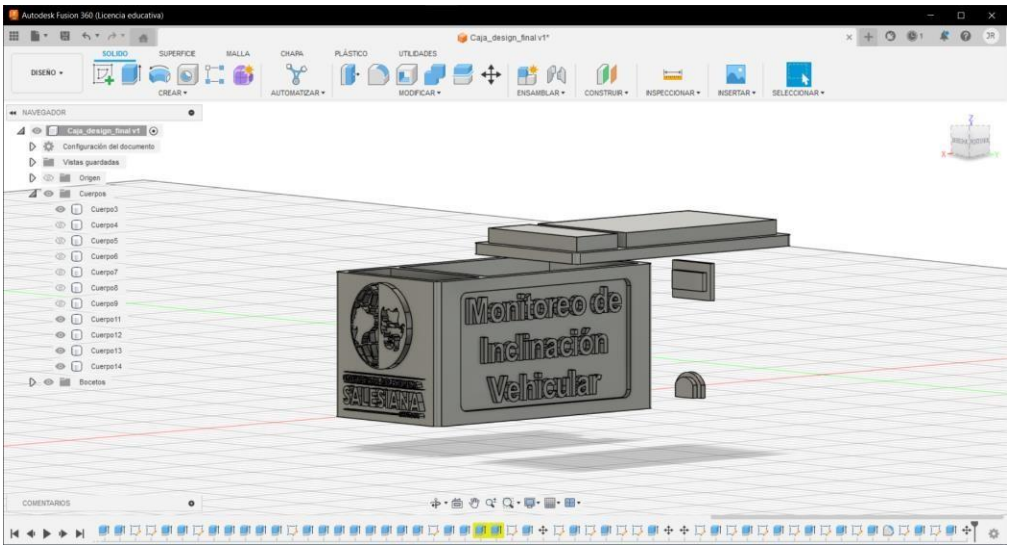

FIGURA 3.18 BOCETO FINAL DE LA CARCASA 9

Fuente: *Autoría*

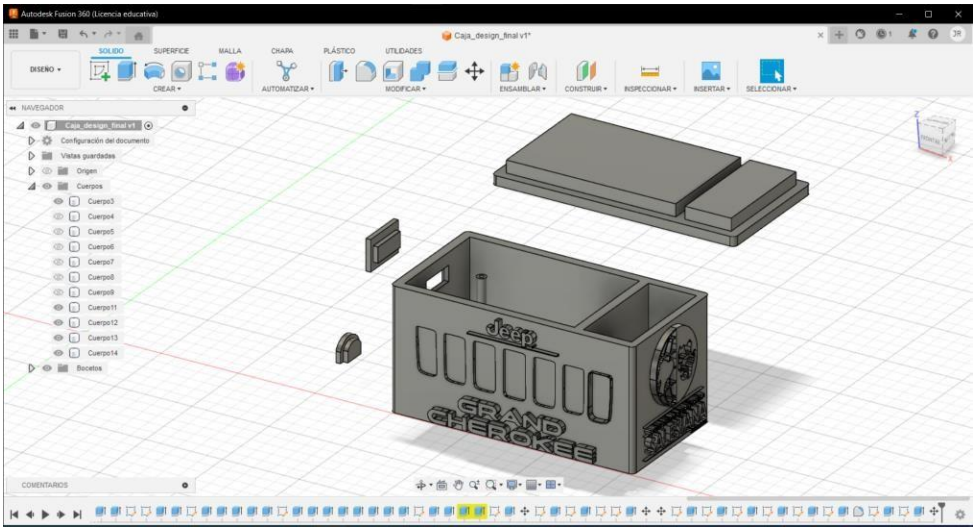

FIGURA 3.19 BOCETO FINAL DE LA CARCASA 10

Fuente: *Autoría*

3.4 DISEÑO DE CIRCUITO ELECTRÓNICO

Nuestro circuito está diseñado por 3 componente, como primer componente tendremos la fuente de alimentación transformada para lograr alimentar nuestro componente el cual lleva por nombre “Computadora de placa única”, por consiguiente, haremos uso del voltaje que otorga nuestro segundo componente (3V3, GND), para alimentar nuestro sensor de inclinación, debemos tener en cuenta que nuestro protocolo de comunicación entre el componente 2 y 3 es comunicación I2C.

En nuestra “Computadora de placa única” los pines seleccionados para lograr la comunicación antes mencionada son el D22, D21, los cuales se conectarán y comunicaran mediante los pines SDA, SCL, de nuestro sensor de inclinación.

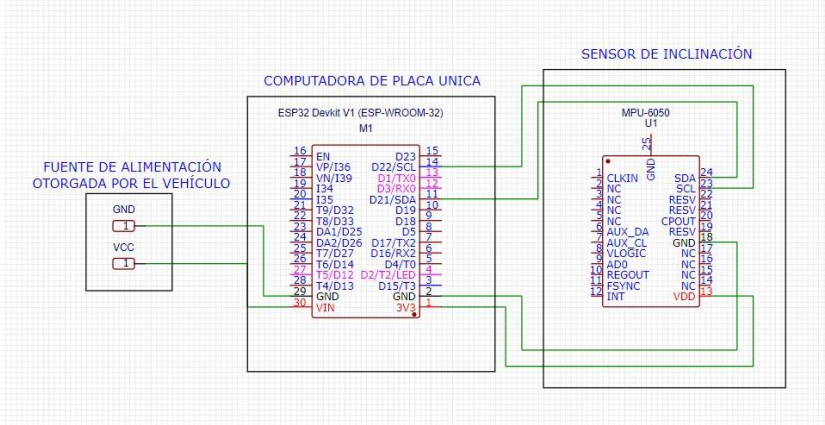

FIGURA 3.20 CONEXIÓN ENTRE ESP32 Y RESPECTIVO SENSOR

Fuente: *Autoría*

3.5 PROTOCOLO DE COMUNICACIÓN I2C

Es un protocolo el cual funciona mediante dos hilos, a través de los mismo puede llegar a conectarse con varios dispositivos donde podemosdefinir a los mismo como MAESTROS O ESCLAVOS.

Para lograr esta comunicación el protocolo asigna una dirección a cada dispositivo, por lo cual el MAESTRO es el que comando la comunicación de transferencia, en otras palabras, es quien decide con que dispositivo conectar para emisión y recepción de datos y también controla cuando se finaliza la comunicación.

Los dos hilos llevan por nombres SDA Y SCL, el primero se encarga de llevar los datos en cambio el otro hilo sobrante es el que lleva la señal de reloj.

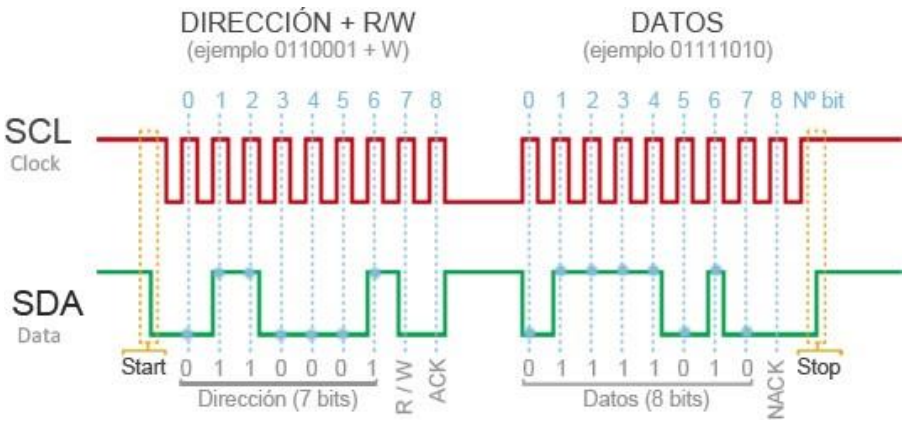

FIGURA 3.21 FUNCIONAMIENTO DE PROTOCOLO I2C

Fuente: *Bus I2C Arduino*

3.6 PRUEBAS DE COMUNICACIÓN ENTRE EL MODULO ESP32 Y APP DE MONITOREO

Para realizar las pruebas de comunicación se definieron dos etapas, la primera etapa consiste en probar su funcionamiento mediante comunicación simple enviando y leyendo datos con los dispositivos montados en la placa prototipo.

FIGURA 3.22 RECEPCIÓN DE DATOS

Fuente: *Autoría*

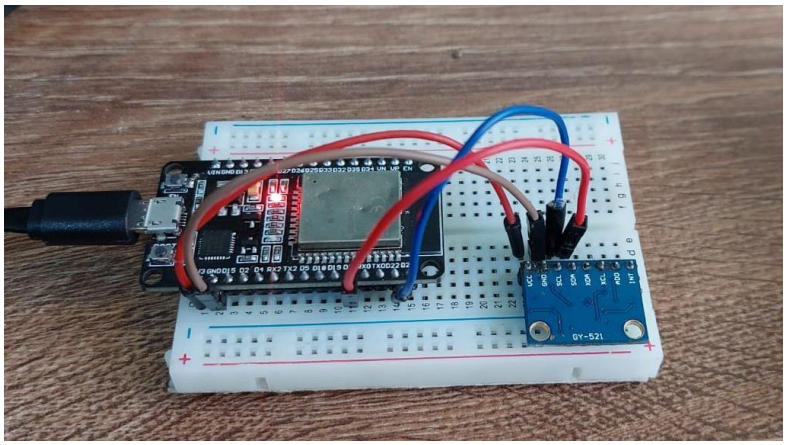

FIGURA 3.23 CONEXIÓN ENTRE ESP32 Y RESPECTIVO SENSOR EN PROTOBOARD

Fuente: *Autoría*

La segunda fase consiste en validar el funcionamiento del dispositivo y la comunicación de los datos dentro de la carcasa plástica y observar de igual forma como se cargan los datos en la aplicación la cual estará instalada en un dispositivo Android de prueba.

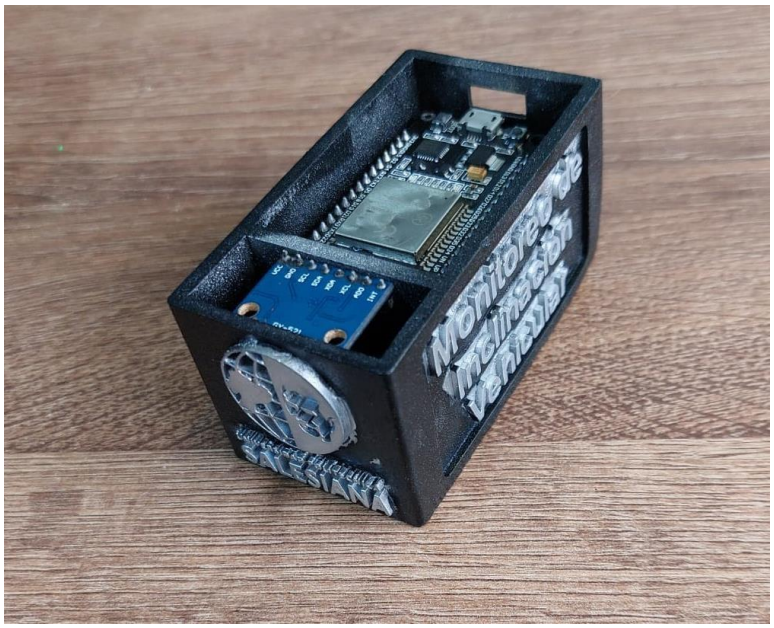

FIGURA 3.24 DISPOSITIVOS EN CARCASA PLÁSTICA 1

Fuente: *Autoría*

FIGURA 3.25 DISPOSITIVOS EN CARCASA PLÁSTICA 2

Fuente: *Autoría*

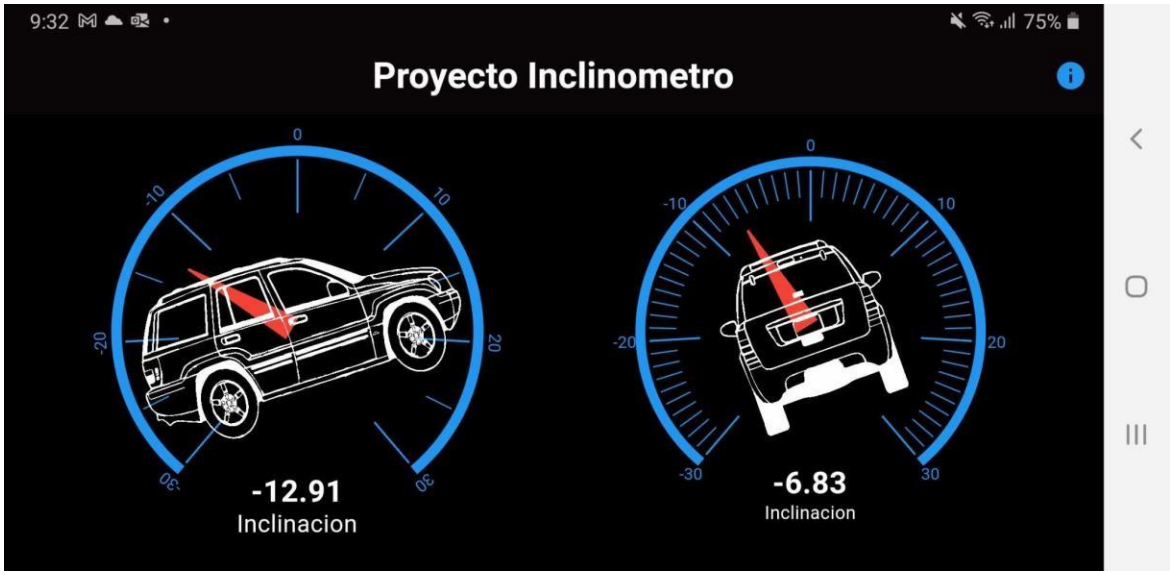

FIGURA 3.26 RECEPCIÓN DE DATOS EN PANTALLA

Fuente: *Autoría*

3.7 ANÁLISIS Y RESULTADOS

El resultado fue favorable, ya que puesto en marcha las pruebas los protocolos de comunicación no confirmaron ningún tipo de error al momento de emisión o recepción de datos.

Por último, se realizaron una serie de pruebas en diferentes tipos de terrenos para de esta manera corroborar el correcto funcionamiento del dispositivo, a su vez se comprobó que la app móvil es adaptable en diferentes tipos de sistemas de entretenimiento basados en Android.

FIGURA 3.27 VISUALIZACIÓN DE DATOS EN PANTALLA

Fuente: *Autoría*

CAPÍTULO 4: CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

En base al progreso de la presente investigación se puede determinar que el uso del módulo ESP32 DevKit V1, permite el desarrollo y control de pruebas entre hardware y software mediante diferentes comunicaciones como Bluetooth y Wi-Fi, la mismas pueden ser utilizadas según las especificaciones y requisitos definidos en el proyecto realizado; teniendo en cuenta esta información, se llevó a cabo diferentes pruebas de comunicación y recepción de datos llegando a la conclusión, de que el mejor método para el desarrollo de un sistema de inclinación para vehículo todoterreno es utilizar la comunicación Bluetooth, en donde, aplicándolo al sistema de monitoreo se obtuvo los resultados esperados con una recepción más eficaz en tiempo real de los datos registrados por los sensores.

La implementación de este proyecto se realizó mediante un estudio y revisión de manuales y de bibliografía de fuentes confiables, en donde se contempló ciertos requisitos específicos para utilizar el dispositivo adecuado al momento de la conexión con los sensores, por consiguiente, al momento de investigar y realizar pruebas se definió el tipo de comunicación correspondiente para el sensor, concluyendo que el método de comunicación adecuado será del tipo I2C.

Por otro lado, se diseñó un dispositivo electrónico para que la aplicación móvil, que se instaló en el sistema de entretenimiento con el uso de herramientas informáticas especializadas, pueda leer o procesar los datos que le proporcionen los sensores de la inclinación transversal, en donde se obtuvo como resultado un dispositivo fiable para el conductor del vehículo. Del mismo modo, la recepción de datos que serán visualizados en la aplicación móvil se lo llevó a cabo de manera inalámbrica.

Finalmente, para comprobar la veracidad y eficacia de este método se un análisis profundo sobre los límites de operación, en donde se pudo realizar pruebas de funcionamiento en diferentes ambientes, para corroborar la exactitud del sensor y del código generado, llegando a los resultados esperados y verificando la correcta funcionalidad de los mismos.

RECOMENDACIONES

Con la finalidad de una mayor duración de los componentes electrónicos internos, se recomienda adicionar al interior del protector plástico una esponja VCI FOAM, con el fin de evitar la corrosión, y la exposición a agentes externos que podrían interferir con la funcionalidad del sistema.

De igual manera se recomienda adicionar las diferentes características de medición que posee el sensor, para medir o cuantificar la inclinación a la programación del entorno IDE de Arduino, cabe recalcar, que se debería modificar la interfaz y el método de recepción de datos de la aplicación móvil desarrollada.

Po otro lado, se recomienda agregar un convertidor DC-DC exclusivo al sistema electrónico automotriz del prototipo para adaptarse a las variaciones de voltaje al encender el automóvil y evitar daños por picos de voltaje. Esto mejorará la estabilidad y confiabilidad de la conexión con la alimentación directa del automóvil.

Finalmente, se recomienda profundizar sobre la problemática principal de la investigación, ya que es un tema de interés social, que puede aportar nuevos conocimientos a la comunidad científica.

REFERENCIAS BIBLIOGRÁFICAS

Álvarez, J., Cárcel, A., & Sebastián, J. M. (2020). *Infotainment systems in the automotive industry. Procedia Computer Science*. *169*, 745-750. https://doi.org/10.1016/j.procs.2020.03.281

Bibawy, M. M., Shaaban, A. M., & Al-Qutayri, M. A. (2019). *Development and evaluation of an inclinometer sensor for vehicles. Measurement*. *147*, 618-628. https://doi.org/10.1016/j.measurement.2019.05.040

D'Mello, G., Gomes, R., Mascarenhas, R., Ballal, S., Kamath, V. S., & Lobo, V. J. (2022). Wheel alignment detection with IoT embedded system. *Materials Today: Proceedings*, *52*, 1924-1929. https://doi.org/10.1016/j.matpr.2021.11.566

Irsan, M., Hassan, R., Hasan, M. K., Lam, M. C., Hussain, W. M. H. W., Ibrahim, A. H., & Ahmed, A. S. A. M. S. (2022). A Novel Prototype for Safe Driving Using Embedded Smart Box System. *Sensors (Basel, Switzerland)*, *22*(5), 1907. https://doi.org/10.3390/s22051907

Ivana, R., Ivan, Š., Nikola, L., & Stojanovic, R. (2023). *Fall detection system for the elderly or disabled people*. https://doi.org/10.13140/RG.2.2.25300.07045

Jeep. (2004). *2004 Jeep Grand Cherokee*.

Khanna, P., Sharma, M., & Bhakar, S. (2020). *Comparative analysis of infotainment systems in automobile industry. International Journal of Mechanical and Production Engineering Research and Development*. *10(3)*, 471-480. https://doi.org/10.24247/ijmperdaug202075

Komarizadehasl, S., Komary, M., Alahmad, A., Lozano-Galant, J. A., Ramos, G., & Turmo, J. (2022). A Novel Wireless Low-Cost Inclinometer Made from Combining the Measurements of Multiple MEMS Gyroscopes and Accelerometers. *Sensors*, *22*(15), Article 15. https://doi.org/10.3390/s22155605

Makki, I., Gao, X., Wang, Y., & El-Sheimy, N. (2021). *On the selection of low-cost MEMS sensors for vehicle motion sensing applications. 181*, 109612. https://doi.org/10.1016/j.measurement.2021.109612

Pereira, M. A., Rodrigues, M. P., & Santos, R. L. (2018). *Desenvolvimento de aplicações móveis para Android. Revista de Informática Teórica e Aplicada*. *25(2)*, 25-45.

Rahman, M. M., Akter, S., & Hasan, M. M. (2020). *An overview of Android application development. International Journal of Advanced Computer Science and Applications*. *11(11)*, 523-530.

Smith, D., Jerina, K., & Hecker, J. (2018). *Investigation of vehicle handling and stability of a sport utility vehicle. SAE Technical Paper Series*. *2018*, 01-1235.

Yang, C., Shi, X., & Zhang, B. (2018). *A comprehensive comparison study of inertial measurement units for mobile robot applications. Sensors,*. *18(8)*, 2648. https://doi.org/10.3390/s18082648

APÉNDICES

APÉNDICE A: DIMENSIONES DEL PROTECTOR PLÁSTICO PARA LOS DISPOSITIVOS Y SENSORES

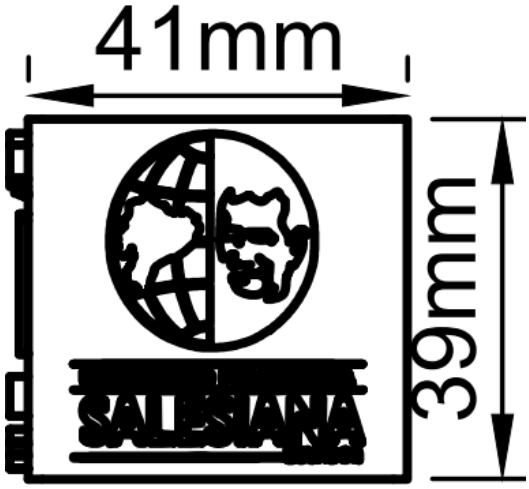

FIGURA A. VISTA FRONTAL DE LA CARCASA PLÁSTICA

Fuente: Autoría

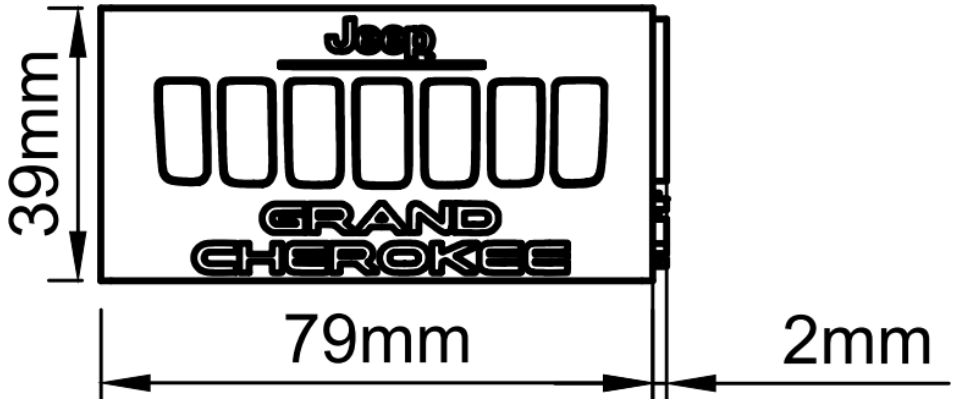

FIGURA B. VISTA LATERAL IZQUIERDA DE LA CARCASA PLÁSTICA

Fuente: Autoría

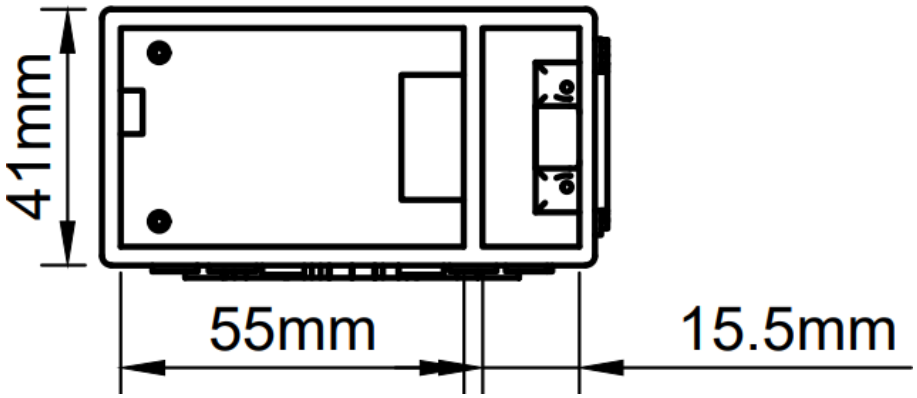

FIGURA C. VISTA SUPERIOR DE LA CARCASA PLÁSTICA

Fuente: Autoría

APÉNDICE B: ALGORITMO IMPLEMENTADO PARA LA OBTENCIÓN DE LOS PARÁMETROS EN IDE ARDUINO

```
#include  <Arduino.h>
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>
#include <Wire.h>
#include <BluetoothSerial.h>

Adafruit_MPU6050 mpu;

sensors_event_t a, g, temp;

float gyroX, gyroY, gyroZ;
float gyroXerror = 0.07;
float gyroYerror = 0.03;
float gyroZerror = 0.07;

bool resetAngles = false;

BluetoothSerial SerialBT;

void initMPU() {
  if (!mpu.begin()) {
    Serial.println("No se pudo encontrar el chip MPU6050");
    while (1) {
      delay(10);
```

```
    }
  }
  Serial.println("MPU6050 encontrado!");
}

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

  if (resetAngles) {
    gyroX = 0;
    gyroY = 0;
    gyroZ = 0;
    resetAngles = false;
  }

  float gyroX_temp = g.gyro.x;
  if (abs(gyroX_temp) > gyroXerror) {
    gyroX += (gyroX_temp / 70.00) * 1000;
  }

  float gyroY_temp = g.gyro.y;
  if (abs(gyroY_temp) > gyroYerror) {
    gyroY += (gyroY_temp / 70.00) * 1000;
  }

  float gyroZ_temp = g.gyro.z;
  if (abs(gyroZ_temp) > gyroZerror) {
    gyroZ += (gyroZ_temp / 70.00) * 1000;
  }
}

void setup() {
  Serial.begin(115200);
  Wire.begin();
  initMPU();

  SerialBT.begin("ESP32_BT"); // Nombre del dispositivo
Bluetooth

  Serial.println("Leyendo giroscopio...");
}

void loop() {
  if (Serial.available() > 0) {
    char input = Serial.read();
    if (input == 'A') {
      resetAngles = true;
    }
  }
```

```
        if (SerialBT.available()) {
          char input = SerialBT.read();
          if (input == 'A') {
            resetAngles = true;
          }
        }

        getGyroReadings();
        String gyroReadings = String(gyroX)+ "," + String(gyroY) +
"," + String(gyroZ);
        Serial.println(gyroReadings);
        SerialBT.println(gyroReadings);

        delay(10);
      }
```

APÉNDICE C: PROGRAMACIÓN DE LA APLICACIÓN MÓVIL EN FLUTTER

Sección de definición de colores o tema de la APP.

```
import 'package:flutter/material.dart';

class ThemeProject {
  final Color primary = Colors.black;
  final Color secondary = Colors.white;
  final Color primarySpecial = Colors.blue;

  ThemeData getTheme() => ThemeData(
      useMaterial3: true,
      colorSchemeSeed: primary,
      appBarTheme: AppBarTheme(
        centerTitle: true,
        backgroundColor: primary,
      ),
      filledButtonTheme: FilledButtonThemeData(
        style: ButtonStyle(
          backgroundColor:
MaterialStateProperty.all(primarySpecial),
        ),
      ),
      switchTheme: SwitchThemeData(
        trackColor: MaterialStateProperty.all(primarySpecial),
      ));
}
```

Sección de comunicación y funcionamiento de la app.

```
import 'dart:convert';
import 'dart:developer';
import 'package:animate_do/animate_do.dart';
import 'package:flutter/material.dart'; import
i'package:flutter/services.dart';  import
'package:flutter_bluetooth_serial/flutter_bluetooth_serial.dart';
import 'package:proyecto_inclinometro/theme/theme.dart';
// import 'package:sensors_plus/sensors_plus.dart';

import 'package:syncfusion_flutter_gauges/gauges.dart';

class InclinometroScreen extends StatefulWidget {
  const InclinometroScreen({super.key});

  @override
  State<InclinometroScreen> createState() =>
_InclinometroScreenState();
}

class _InclinometroScreenState extends
State<InclinometroScreen> {
  //Estado de la conexión de bluetooth desconocido BluetoothState
  _estadoBluetooth = BluetoothState.UNKNOWN;
  //Obtenr una instancia de Bluetooth
  // ignore: prefer_final_fields
  FlutterBluetoothSerial _bluetooth =
FlutterBluetoothSerial.instance;
  // Seguir la conexión de bluetooth con el dispositivo i
remoto
  BluetoothConnection? _conexion;
  // ignore: unused_field
  int _estadoDispositivo;

  List<BluetoothDevice> _listadoDispositivos= []; BluetoothDev
  ceo _dispositivo;    i    i i
  bool _conectado = false;
  bool _isBotonDispoinible = false;

  // Dispositivo Bluetooth
  bool get  isConectado => _conexion != null &&
_conexion!.isConnected;
  bool  isDesconectado = false;
```

```
//! Variables del proyecto
double inclinacionX = 0.00;
double inclinacionY = 0.00;
double resetInclinacionX = 0.00;
double resetInclinacionY = 0.00;
// String direction = "";

double x = 0, y = 0, z = 0;
// String direction = "none";

@override
void initState() {
  super.initState();
  // Obtener el estado actual
  FlutterBluetoothSerial.instance.state.then((state) {
    setState(() {
      _estadoBluetooth = state;
    });
  });
  _estadoDispositivo = 0;
  //Si no esta activado, solicitamos que active
  varBluetooth();

  // Escuchar los cambios del dispositivo conectado
  FlutterBluetoothSerial.instance
      .onStateChanged()
      .listen((BluetoothState state) {
    setState(() {
      _estadoBluetooth = state;
      if (_estadoBluetooth == BluetoothState.STATE_OFF) {
        _isBotonDisponible = true;
      }
      getDispositivosVinculados();
    });
  });
  //Giroscopio del telefofo
  // gyroscopeEvents.listen((GyroscopeEvent event) {
  //   // print('x ${event.x}');
  //   print('y ${event.y}');

  //   x = event.x; //sube y bajaz
  //   y = event.y; // de lado a lado
  //   // z = event.z;

  //   //rough calculation, you can use
  //   //advance formula to calculate the orentation
  //   if (x > 0) {
  //     // direction = "back";
  //     inclinacionX = x;
```

```
      //   }
      //   if (x < 0) {
      //     // direction = "forward";
      //     inclinacionX = x;
      //   }
      //   if (y > 0) {
      //     // direction = "left";
      //     inclinacionY = y;
      //   }
      //   if (y < 0) {
      //     // direction = "right";
      //     inclinacionY = y;
      //   }

      //   setState(() {});
      // });
      // super.initState();
    }

    @override
    void dispose() {
      //Limpiar la conexión para evitar fuga de memoria
      if( isConectado) {
        isDesconectado = true;
        _connection!.dispose();
        _connection = null;
      }
      super.dispose();
    }

    Future<bool> activarBluetooth() async {
      // Obtenemos el estado actual
      _estadoBluetooth = await
FlutterBluetoothSerial.instance.state;
      // Si estado apagado , encendemos y obtenemos los
dispositivos disponibles
      if (_estadoBluetooth == BluetoothState.STATE_OFF) {
        await FlutterBluetoothSerial.instance.requestEnable();
        await getDispositivosVinculados();
        return true;
      } else {
        await getDispositivosVinculados();
      }
      return false;
    }

    Future<void> getDispositivosVinculados() async {
      List<BluetoothDevice> dispositivos = [];
      try {
```

```
      dispositivos = await _bluetooth.getBondedDevices();
    } on PlatformException {
      log("Error getDispositivosVinculados");
    }

    if (!mounted) {
      return;
    }

    setState(() {
      _listadoDispositivos = dispositivos;
    });
  }

  final MaterialStateProperty<Icono> thumbIcon =
      MaterialStateProperty.resolveWith<Icono>(
    (Set<MaterialState> states) {
      if (states.contains(MaterialState.selected)) {
        return const Icon(Icons.check);
      }
      return const Icon(Icons.close);
    },
  );

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: const Text(
          "Monitoreo de Inclinación",
          style: TextStyle(
            color: Colors.white,
            fontWeight: FontWeight.bold,
            fontSize: 24,
          ),
        ),
        actions: [
          IconButton(
            onPressed: () {
              showInfoAutores();
            },
            icon: Icon(
              Icons.info,
              color: ThemeProject().principal,
            ),
          )
        ],
      ),
      body: _bodyInclinometro(),
```

```
            backgrOundC0lOr: ThemeflrOject().primary,
          );
        }

        shOwlnf0AutOres() {
          shOwD alOg(
            barrierDismissible: false,
            cOntext: cOntext, bu ilder:
            (c0ntext) {
              return AlertDialOg(
                elevati0n: 0,
                title: ROw(
                  mainAxisAl gment:
MainAx isAl ignment.spaceBetween,
                  children: [
                    Spinflerfect(
                      infinite: true,
                      durati0n: cOnst Durat i0n(secOnds: 5),
                      child: cOnst C ircleAvatar(
                        maxRadius: 25,
                        backgrOundImage:
AssetImage('assets/ups.png"),
                      ),
                    ),
                    cOnst Text(
                      "",
                      style: TextStyle(fOntS ize: 18, fOntWe ght:
FOntWeight.bOld),
                    ),
                    // cOnst C ircleAvatar(
                    //    maxRad us: 25,
                    //    backgrOundImage:
AssetImage('assets/lOgO_tutOr.jpg"),
                    // ),
                    SizedBOx(
                        width: 50,
                        height: 50, ch
                        ldi
                            Image.asset('assets/lOgO_tutOr.jpg",
fit: BOxFit.cOntain)),
                  ],
                ),
                shape:
                    ROundedRectangleBOrder(bOrderRad ius:
BOrderRadius.c ircular(20)),
                cOntent: COlumn(mainAxisS ize: Ma nAx sSize.min, chi ldren:
cOnst [
                  Text(
                    'Univers dad flOl técn ca Sales ana',
```

```
                    style: TextStyle(fOntS ize: 18, fOntWe ight:
FOntWeight.bOld),
                  ),
                  Text("AutOría: Ing. JOnnathan ROdas"),
                  Text("Dirección: Ing. MarcelO FlOres Mst.Mgt.")
                ]),
                actiOns: [
                  FilledButtOn. icOn(
                    OnflressEd: () {
                      NavigatOr.pOp(cOntext);
                    },
                    icOn: cOnst IcOn(
                      IcOns.ex it_tO_app,
                      cOlOr: COlOrs.wh te,
                    ),
                    label: cOnst Text(
                      'SALIR',
                      style: TextStyle(cOlOr: COlOrs.wh te),
                    ),
                  ),
                ],
              );
            },
          );
        }

        Widget _bOdyInclinOmetrO() {
          Size s ize = Med iaQuery.Of(cOntext).s ize;

          return SingleChildScrOllV ew( ch ld:
            COlumn(
              children: [
                ROw(
                  mainAxisAl ignment: MainAxisAlgnmentend, ch
                  ldren: [
                    //**BOTON flARA RESTABLECER */
                    FilledButtOn.icOn(
                      OnflressEd: _resetIncl inat 0n,
                      icOn: cOnst IcOn(
                        IcOns.refresh_sharp,
                        cOlOr: COlOrs.whi te,
                      ),
                      label: cOnst Text(
                        'ZERO",
                        style: TextStyle(cOlOr: COlOrs.wh te),
                      ),
                    ),
                  ],
                ),
```

```
                ROw(
                  mainAxiSAl gnment:
MainAx iSAl ignment.SpaceEvenly,
                  children: [
                    //** UnO
                    SizedBOx(
                      height: Size.height * 0.8,
                      width: S ize.w dth * 0.4, ch
                      ldi Stack(
                        chidren: [
                          SfRadialGauge(
                            enableLOadingAn imat On: tue,
                            axeS: <RadialAx iS>[
                              RadialAxiS(
                                minimum: -40,
                                maximum: 40,
                                interval: 10,
                                minOrTickSflerInterval: 9,
                                ShOwAxiSL ne: true,
                                labelOffSet: 8,
                                ShOwLaStLabel: true,
                                uSeRangeCOlOrFOrAxiS: true,
                                canROtateLabelS: true,
                                labelSflOSit Dn:
ElementSflOS ti On.OutSi de,
                                rangeS: <GaugeRange>[
                                  GaugeRange(
                                    StartValue: -40,
                                    endValue: 40,
                                    StartWidth: 0.05,
                                    SizeUni t:
GaugeS zeUn t.factOr,
                                    endWidth: 0.05,
                                    cOlOr:
ThemeflrOject().principal,
                                  ),
                                ],
                                annOtatiOnS: <GaugeAnnOtati0n>[
                                  GaugeAnnOtati0n(
                                    angle: 90,
                                    pOSitiOnFactOr: 0.5, w
                                    dget: Text(
                                      'Inclinación de AScenSO',
                                      Style: TextStyle(
                                        cOlOr:
ThemeflrOject().SecOndary,
                                        fOntSize: 16,
                                      ),
```

```
                ),
              ),
              GaugeAnnOtatiOn(
                angle: 90,
                pOSitiOnFactOr: 0.8,w
                dget: Text(
                  inclinacIOnX.tOStr ingASFixed
(2),
                  Style: TextStyle(
                      fOntWe ght:
FOntWeight.bOld,
                      fOntSize: 20, cOlOr:
ThemeflrOject().SecOndary),
                ),
              )
            ],
            pOinterS: <GaugeflOinter>[
              NeedleflOinter(
                value: inclinacIOnX,
                // needleLength: 10,
                animatiOnType:
AnimatiOnType.bOunceOut,
                enableAnimatiOn: true, knObStyle:
                cOnSt KnObStyle(
                  cOlOr: COlOrS.tranSparent,
                  bOrderWidth: 0.05,
                ),
                needleCOlOr: COlOrS.red,
              )
            ],
            axiSLabelStyle: cOnSt
GaugeTextStyle(fOntSize: 12),
            majOrTickStyle: cOnSt
MajOrTickStyle(
                length: 0.25, lengthUnit:
GaugeSizeUnit.factOr),
            minOrTickStyle: cOnSt
MinOrTickStyle(
              length: 0.13,
              lengthUnit:
GaugeSizeUnit.factOr,
              thickneSS: 1,
            ),
          )
        ],
      ),
```

```
Center(
  child: TranSfOrm.rOtate(
```

```
                                angle: inclinacion,
                                child: SizedBox(
                                  width: Size.width * 0.2,
                                  height: Size.height * 0.2,
                                  // child:
Image.asset('assets/frontal.png'),
                                  child:
Image.asset('assets/frontal.png'),
                                ),
                              ),
                            ),
                          ],
                        ),
                      ),
                      //**DOS
                      SizedBox(
                        height: Size.height * 0.8,
                        width: Size.width * 0.4, ch
                        ild: Stack(
                          children: [
                            SfRadialGauge(
                              enableLoadingAnimation: true,
                              axes: <RadialAxis>[
                                RadialAxis(
                                  minimum: -40,
                                  maximum: 40,
                                  interval: 10,
                                  minorTickSflerInterval: 9,
                                  ShowAxisLine: true,
                                  radiusFactor: 0.9,
                                  labelOffset: 8,
                                  ShowLastLabel: true,
                                  useRangeColorForAxis: true,
                                  ShowTicks: true,
                                  labelSflOsition:
ElementSflOsition.Outside,
                                  ranges: <GaugeRange>[
                                    GaugeRange(
                                      StartValue: -40,
                                      endValue: 40,
                                      StartWidth: 0.05,
                                      SizeUnit:
GaugeSizeUnit.factor,
                                      endWidth: 0.05,
                                      color:
ThemeflrOject().principal,
                                    ),
                                  ],
                                  annotations: <GaugeAnnotation>[
```

```
        GaugeAnnOtatiOn(
          angle: 90,
          pOSitiOnFactOr: 0.95, w
          dget: Text(
            'Inclinación TranSverSal', Style:
            TextStyle(
              cOlOr:
ThemeflrOject().SecOndary,
              fOntSize: 12,
            ),
          ),
        ),
        GaugeAnnOtatiOn(
          angle: 90,
          pOSitiOnFactOr: 0.8, w
          dget: Text(
            incl inac iOnY.tOStr ingASFixed
(2),
            Style: TextStyle(
                fOntWe ght:
FOntWeight.bOld,
                fOntSize: 20, cOlOr:
ThemeflrOject().SecOndary),
          ),
        )
      ],
      pOinterS: <GaugeflOinter>[
        NeedleflOinter(
          value: inclinac iOnY,
          // needleLength: 10,
          anim at iOnType:
Anim at iOnType.bOunceOut,
          enableAnim at iOn: true, knObStyle:
          cOnSt KnObStyle(
            cOlOr: COlOrS.tranSparent,
            bOrderWidth: 0.05,
          ),
          needleCOlOr: COlOrS.red,
        )
      ],
      axiSLabelStyle: cOnSt
GaugeTextStyle(fOntSize: 12),
      majOrTickStyle: cOnSt
MajOrTickStyle(
        length: 0.25,
        lengthUn it:
GaugeS zeUn t.factOr,
```

),

```
                                minorTickStyle: const
MinorTickStyle(
                                  length: 0.13,
                                  lengthUnit:
GaugeSizeUnit.factor,
                                  thickness: 1,
                                ),
                              )
                            ],
                          ),
                          // Center(
                          //   child: Transform.rotate(
                          //     angle: inclinationY,
                          //     child:
Image.asset('assets/inclination.png'),
                          //   ),
                          // ),
                          Center(
                            child: Transform.rotate(
                              angle: inclinationX,
                              child: SizedBox(
                                width: Size.width * 0.3,
                                height: Size.height * 0.3,
                                // child:
Image.asset('assets/inclination.png'),
                                child:
Image.asset('assets/inclination.png'),
                              ),
                            ),
                          ),
                        ],
                      ),
                    )
                  ],
                ),

                //**BLUETOOTH */
                Visibility(
                  visible: _isBotonDisponible &&
                      _estadoBluetooth == BluetoothState.STATE_ON,
                  child: LinearProgressIndicator(
                    backgroundColor: Colors.red,
                    valueColor:
                        AlwaysStoppedAnimation<Color>(ThemeProject
().principal),

                  ),
                ),
                //Activar/desactivar conexion bluetooth
                    i
```

fladd ng(

```
                padding: const EdgeInsets.all(5),
                child: Row(
                  mainAxisAlignment: MainAxisAlignment.start, children:
                  <Widget>[    i
                    Text(
                      'Estado Bluetooth',
                      style: TextStyle(
                          color: ThemeData().secondary,
                          fontSize: 16,
                          fontWeight: FontWeight.bold),
                    ),
                    const Spacer(),
                    Switch(
                      thumbIcon: thumbIcon,
                      value: _estadoBluetooth.isEnabled,
                      onChanged: (bool value) {
                        future() async {
                          if (value) {
                            await
FlutterBluetoothSerial.instance.requestEnable();
                          } else {
                            await
FlutterBluetoothSerial.instance.requestDisable();
                          }
                          await getDispositivosEmparelhados();
                          _isButtonDisponible = false;
                          if (_conectado) {
                            _desconectarBluetooth();
                          }
                        }

                        future().then((_) {
                          setState(() {});
                        });
                      },
                    )
                  ],
                ),
              ),
              //Dispositivos
              Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                  Padding(
                    padding: const EdgeInsets.only(top: 15),
                    child: Text(
                      "Dispositivos Disponibles", style:
                      TextStyle(
                          fontSize: 24,
```

```
                                color: ThemeData().secondary,
                                fontWeight: FontWeight.bold),
                          ),
                        ),
                        Padding(
                          padding: const EdgeInsets.all(8.0),
                          child: Row(
                            mainAxisAlignment:
MainAxisAlignment.spaceBetween,
                            children: [
                              Text(
                                "Dispositivo:",
                                style: TextStyle(
                                  color: ThemeData().secondary,
                                  fontWeight: FontWeight.bold,
                                ),
                              ),
                              DropdownButton(
                                items: _getListaDispositivos(), onChanged:
                                (value) =>
                                    setState(() => _dispositivo =
value),
                                value:
                                    _listaDispositivos.isNotEmpty ?
_dispositivo : null,

                              ),
                              FilledButton(
                                onPressed: _isBotonDisponible
                                    ? null
                                    : _conectado
                                        ? _desconectarBluetooth
                                        : _conectarBluetooth,
                                child: Text(_conectado ? 'Desconectar'
: 'Conectar'),

                              ),
                            ],
                          ),
                        ),
                      ],
                    ),
                    const SizedBox(
                      height: 50,
                    )
                  ],
                ),
              );
            }
```

```
void _resetInclination() aSync {
```

```
            _cOnexiOn!.Output.add(aScii.encOde("A"));
            await _cOnexiOn!.Output.allSent;
            mOStrarNOtificaciOn(
              'Inclinación reStablecidad cOrretamente.',
              COlOrS.green,
            );
            SetState(() {
              _eStadODiSpOSitivO = 1;
              inclinaciOnX = 0.0; // devi.e On
              inclinaciOnY = 0.0; // devi.e On
            });
          }

          // Crear el liStadO de diSpOSitivOS y agreegar al DrOpdOwn
Menu
          LiSt<DrOpdOwnMenuItem<BluetOOthDev ice>>
_getLiStadOD SpOS t i vOS() {
            LiSt<DrOpdOwnMenuItem<BluetOOthDev ce>>  itemS = [];
            if (_liStadODiSpOSit ivOS. SEmpty) {
              itemS.add(cOnSt DrOpdOwnMenuItem(
                child: Text('Sin D iSpOS it ivOS'),
              ));
            } elSe {
              fOr (var device in _liStadODiSpOSitivOS){
                itemS.add(DrOpdOwnMenuItem(
                  value: devi.e,
                  chi.ld: Text(dev ce.name!),
                ));
              }
            }
            return itemS;
          }

          // COnectar a bluetOOth
          vOid _cOnectarBluetOOth() aSync {
            SetState(() {
              _iSBOtOnDi SpOn ble = true;
            });
            if (_d iSpOSitivO == null) {
              mOStrarNOtif icac On("Selecc One un d SpOS t vO",
COlOrS.red);
              _iSBOtOnDiSpOn ble = falSe;
            } elSe {
              if (! SCOnectadO) {
                awai t
BluetOOthCOnnecti On.tOAddreSS(_d SpOS t i vO!.addreSS)
                    // ignOre:
nO_leading_underScOreS_fOr_lOcal_identif ierS
                    .then((_cOnnecti On) {
```

```
                    lOg('COnnected tO the device');
                    _cOnexiOn = _cOnnectiOn;
                    SetState(() {
                      _cOnectadO = true;
                    });
                    //**EScuchar la emiSiOn del diSpOSitivO
                    _cOnexiOn!.input!.listen((Uint8List data) {
                      try {
                        //TODO 1  InfOrmaciOn que viene del
SpiOS tivO bluetOOth
                        var text = aScii.decOde(data);
                        if (text.iSNOtEmpty && text.length > 2) {
                          // ignOre: avOid_print
                          // print(text);
                          var Splitted = text.Split(',');
                          // ignOre: avOid_print
                          print("X ${Splitted[0]}");
                          // ignOre: avOid_print
                          print("Y ${Splitted[1]}");
                          // ignOre: avOid_print
                          // print("Z ${Splitted[2]}");

                          var x = dOuble.parSe(Splitted[0]);
                          var y = dOuble.parSe(Splitted[1]);
                          // var z = dOuble.parSe(Splitted[2]);

                          //Se calcula el angulO de inclinaciOn de una
recta
                          // var inclinatiOn = (0 - y) / (x - 0);
                          // lOg(inclinatiOn.tOString());
                          // inclinaciOn = inclinatiOn;

                          // if (x > 0) {
                          //   directiOn = "back";
                          // }
                          // if (x < 0) {
                          //   directiOn = "fOrward";
                          // }
                          // if (y > 0) {
                          // directiOn = "left";
                          inclinaciOnX = x;
                          // }
                          // if (y < 0) {
                          // directiOn = "right";
                          inclinaciOnY = y;
                          // }

                          //TODO 4 Si paSa lOS 30% RiegO de vuelcO
                          if (x > 30) {
```

```
                    mOStrarNOtificaciOn("RiegO de vuelcO en
X", COlOrS.green);
                  }
                  if (x < -30) {
                    mOStrarNOtificaciOn("RiegO de vuelcO en
X", COlOrS.green);

                  }
                  if (y > 30) {
                    mOStrarNOtificaciOn("RiegO de vuelcO en
Y", COlOrS.green);

                  }
                  if (y < -30) {
                    mOStrarNOtificaciOn("RiegO de vuelcO en
Y", COlOrS.green);

                  }
                  SetState(() {});
                }
                // ignOre: empty_catcheS
              } catch (e) {}

              // SetState(() {});
            }).OnDOne(() {
              if (iSDeScOnectadO) {
                lOg('DeScOnectadO lOcalmente");
              } elSe {
                lOg('DeScOnectadO remOtamente");
              }
              if (mOunted) {
                SetState(() {});
              }
            });
          }).catchErrOr((errOr) {
            lOg('CannOt cOnnect, exceptiOn OccurredI);
            lOg(errOr);
          });
          mOStrarNOtificaciOn('DiSpOSitivO cOnectadO",
COlOrS.green);
          SetState(() => _iSBOtOnDiSpOnible = falSe);
        }
      }
    }

    // MethOd tO diScOnnect bluetOOth
    vOid _deScOnectarBluetOOth() aSync {
      await _cOnexiOn!.clOSe();
      mOStrarNOtificaciOn('DiSpOSitivO deSOnectadO,
COlOrS.green);
```

```
if (!_cOnexiOn!.iSCOnnected) {
  SetState(() {
```

```
            _conectado = false;
            _isBotonDispONible = false;
            _estadoDispOSitivO = 0;
            inclinacionX = 0.00;
            inclinacionY = 0.00;
          });
        }
      }

      // Method to Show a Snackbar
      mOStrarNOtificacIOn(String mesSage, COlOr cOlOr){
        ScaffOldMeSSenger.Of(cOntext).clearSnackBarS();
        final Snackbar = SnackBar(
          cOntent: Text(
            meSSage,
            textAlign: TextAlign.center,
          ),
          duratiOn: cOnSt DuratiOn(milliSecOndS: 300),
          backgrOundCOlOr: cOlOr,
        );
        ScaffOldMeSSenger.Of(cOntext).ShOwSnackBar(Snackbar);
      }
    }
```