ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Mecánica y Ciencias de la Producción

Diseño de un sistema automatizado para la crianza y detección de enfermedades en pollos de engorde

PROYECTO INTEGRADOR

Previo la obtención del Título de:

Ingenieros en Mecatrónica

Presentado por:

Sheyla Andrea Benavides Salmerón

Hugo Enrique Lugmania Montesdeoca

GUAYAQUIL - ECUADOR

Año: 2022

DEDICATORIA

Dedico el presente proyecto a Dios y la Virgen María, por siempre bendecirme con las mejores herramientas y personas en mi vida.

A mis padres y hermana, por su apoyo, ánimos y preocupación durante todas las etapas de mi vida.

A mi abuelita Kely, por su preocupación, cariño y ánimos.

A mis profesores del colegio y universidad, que me dieron invaluables lecciones de vida que siempre llevo conmigo y me permitieron estar donde estoy ahora.

A mi amiga Ale Cruz y su mamá Jacqueline Monge, que me dieron los ánimos que no tenía en los momentos más difíciles en el pre.

A Carlitos Cumbe, por su apoyo desinteresado y por creer en mi cuando ni yo podía.

A mi perro Jack, mi fiel compañero de amanecidas de estudio y proyectos.

Sheyla Benavides Salmerón

DEDICATORIA

El resultado de este trabajo lo quiero dedicar a toda mi familia, en especial a mi padre, quien me ha enseñado de manera directa o indirecta a superarme cada día, a seguir esforzándome en cada una de mis metas y proyectos a pesar de las adversidades, a nunca rendirme, y a actuar con humildad y buenos valores. Sin él no sería posible todo lo que he conseguido.

Hugo Lugmania Montesdeoca

AGRADECIMIENTOS

Agradezco a Dios y la Virgen María por darme fortaleza en los tiempos difíciles.

A mis padres y hermana, por siempre apoyarme incondicionalmente en mis decisiones y metas, y por siempre alegrarse de mis logros como si fueran suyos.

A mi familia, en especial a mi abuelita Kely, por siempre pedir por mi bienestar, cuidarme y mostrarme su cariño.

A mis amigos del colegio y la universidad, por su apoyo y hacer de esta etapa de mucho sacrificio un episodio memorable de mi vida.

Al M.Sc. Efraín Terán y al Ph.D. Christian Tutivén, por guiarnos de la mejor manera en el desarrollo de este proyecto.

A Hugo, por ser el mejor compañero de equipo y un hermano para mí. Gracias por ser parte de mi vida, dejarme ser parte de la tuya y enseñarme el verdadero significado de la amistad.

Sheyla Benavides Salmerón

AGRADECIMIENTOS

Estoy agradecido con las diferentes personas que me han apoyado durante mi etapa académica, en especial mi familia; mis padres, mi hermana, mis tíos y mis abuelos, que siempre estuvieron allí, y que cada uno de ellos ha aportado de cierta manera a la gran persona que soy hoy en día.

También a mi amiga Sheyla, por todo el apoyo que me ha brindado dentro y fuera de la universidad, por todo lo que ha aportado a mi vida y mi carrera.

Así también, a mis amigos que me vienen acompañando desde el colegio, como Jorge, los cuales han hecho de este proceso algo llevadero y divertido.

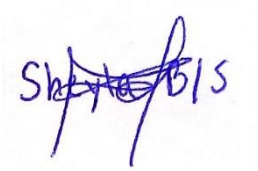
Así mismo, a los que conocí en la universidad, en especial a Julio.

Agradecer a mi tutor Christian Tutivén quien es no solo un gran profesional en su área, sino también una buena persona que nos ha apoyado durante todo este proceso académico de titulación.

Hugo Lugmania Montesdeoca

DECLARACIÓN EXPRESA

“Los derechos de titularidad y explotación, nos corresponde conforme al reglamento de propiedad intelectual de la institución; *Sheyla Andrea Benavides Salmerón* y *Hugo Enrique Lugmania Montesdeoca* damos nuestro consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual"



Sheyla Benavides Salmerón

Hugo Lugmania Montesdeoca

EVALUADORES

Firmado electrónicamente por:
EFRAIN ANDRES TERAN

Efraín Terán, M.Sc.

PROFESOR DE LA MATERIA

Firmado electrónicamente por:
CHRISTIAN JAVIER TUTIVEN GALVEZ

Christian Tutivén, Ph.D.

PROFESOR TUTOR

RESUMEN

La crianza de pollos de engorde, como la raza Cobb500, es una de las actividades económicas de mayor crecimiento en los últimos años en Ecuador. El proceso de crianza demanda un continuo monitoreo de factores ambientales como temperatura y humedad del galpón de los pollos, registro de parámetros importantes para análisis de rentabilidad, además de un control de padecimientos de alta mortalidad como la enfermedad de Newcastle. El presente proyecto consiste en el diseño de un sistema de control y monitoreo de parámetros ambientales, y detección de enfermedades en pollos de engorde Cobb500 para asegurar su óptimo desarrollo.

Se diseñó un sistema IoT y un sistema de detección de la enfermedad de Newcastle usando modelos de inteligencia artificial. El sistema IoT cuenta con sensores de temperatura y humedad junto a una Raspberry Pi que acciona el sistema mecánico para la apertura y cierre de las cortinas del galpón basándose en los factores ambientales. El sistema mecánico se compone de un sistema riel-carrito activado por un motor DC y un sistema de poleas y cables de acero. Una aplicación fue programada para el registro diario de parámetros importantes, control de enfermedades, visualización de temperatura y humedad, y control de las cortinas y luces del galpón de forma manual o automática.

El sistema diseñado permite un monitoreo constante de los pollos, además de la prevención de pérdidas por contagio de la enfermedad de Newcastle a través de la detección temprana por parte de un modelo con una precisión del 95.7%.

Palabras Clave: crianza de pollos, factores ambientales, enfermedad de Newcastle, inteligencia artificial, sistema IoT.

ABSTRACT

Broiler chicken farming, such as Cobb500 breed, is one of the fastest growing economic activities in the last years in Ecuador. The farming process demands a constant monitoring of environmental factors such as temperature and humidity inside the broiler chickens' shed, registration of important parameters for profitability analysis, along with the control of high mortality diseases such as Newcastle disease. This project consists of the design of a control and monitoring system of environmental parameters and disease detection of Cobb500 broiler chickens to ensure optimal development.

An IoT system and a Newcastle disease detection through artificial intelligence system was designed. The IoT system has temperature and humidity sensors along with a Raspberry Pi that activates the mechanical system for the opening and closing of the shed's curtains based on the ideal environmental factors. The mechanical system has a rail-trolley system activated by a DC motor and a pulley system. An application was programmed for the daily registration of important parameters, disease control, temperature and humidity visualization and curtain and lights control manually or automatically.

The designed system allows a constant monitoring of the broiler chickens and prevents the loss due to Newcastle disease contagion through early detection by a model with 95.7% precision.

Keywords: *broiler chicken farming, environmental factors, Newcastle disease, artificial intelligence, IoT system.*

ÍNDICE GENERAL

ABREVIATURAS

CDF	*Correlation Distance Fisher Criterion*
CNN	*Convolutional Neural Networks*
CONAVE	Corporación Nacional de Avicultores del Ecuador
DC	*Direct current*
DCNN	*Deep Convolutional Neural Network*
DVR	*Digital Video Recorder*
ESPOL	Escuela Superior Politécnica del Litoral
FAO	Organización de las Naciones Unidas para la Alimentación y la Agricultura
GPRS	*General Packet Radio Service*
HMM	*Hidden Markov Model*
IOT	*Internet of Things*
IR	Infrarrojo
LCD	*Liquid Crystal Display*
MAG	Ministerio de Agricultura y Ganadería
NC	*Normally closed*
NO	*Normally open*
OCDE	Organización para la Cooperación y el Desarrollo Económicos
P2P	*Peer-to-peer*
PC	*Personal Computer*
PIB	Producto Interior Bruto
SPI	*Serial Peripherical Interface*
SVM	*Support Vector Machine*
WiFi	*Wireless Fidelity*
WMFCCs	*Wavelet Transform Mel Frequency Cepstrum Coefficients*

SIMBOLOGÍA

A	Amperios
℃	Grados Celsius
g	Gramo
GB	Gigabyte
GHz	Gigahertz
HP	Horsepower
HR	Humedad Relativa
kg	Kilogramo
$\frac{\mathrm{kg}}{\mathrm{m}}$	Kilogramo por metro
$\frac{\mathrm{kg}}{\mathrm{m}^2}$	Kilogramos por metro cuadrado
m	Metro
$\frac{\mathrm{m}}{s}$	Metro sobre segundo
$\frac{\mathrm{m}}{\mathrm{s}^2}$	Metro sobre segundo al cuadrado
m^2	Metros cuadrados
mA	Miliamperios
mm	Milímetro
MP	Megapíxel
ms	Milisegundo
mW	Miliwatts
N	Newton
Nm	Newton metro
s	Segundo
V	Voltios
W	Watts

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE PLANOS

CAPÍTULO 1

1. INTRODUCCIÓN

Según la Organización para la Cooperación y el Desarrollo Económicos (OCDE) y la Organización de las Naciones Unidas para la Alimentación y la Agricultura (FAO), la carne de aves de corral se ha mantenido desde la década pasada como el principal impulsor del crecimiento de la producción de carne y, se proyecta hasta el 2030, un crecimiento a nivel mundial [1].

Dentro del Ecuador, hay empresas y microempresas de producción de carne de pollo en las 24 provincias del país y es el tipo de carne con el mayor porcentaje de crianza dentro del mismo, (71%), en comparación a las otras aves de crianzas [2]. En el país las pequeñas empresas, microempresas y personas naturales que se dedican a la crianza de pollo manejan manualmente el control y monitoreo de los parámetros ambientales, contagio de enfermedades y registro de parámetros; teniendo como resultado, altos costes de personal que supervisen las 24 horas del día, pérdidas y datos insuficientes para la proyección de rentabilidad y flujo de caja. Por ende, se busca diseñar un sistema mecatrónico que se encargue del monitoreo de los parámetros ambientales de estas aves, control de los actuadores, detección de enfermedades y registro diario de la producción.

1.1 Descripción del problema

Una granja tiene un pequeño galpón donde crían pollos de engorde Cobb500, destinados a la venta al final de un proceso de crianza de 63 días, donde ya llegan a su peso ideal. El pollo de engorde Cobb500, al igual que otros pollos de engorde, es susceptible a varios tipos de enfermedades que amenazan su óptimo desarrollo, son fácilmente transmisibles y comprenden altos índices de mortalidad en caso de no tomar las medidas necesarias ante el contagio de estas; además del peligro de que se transmitan a humanos [3].

Figura 1.1 Galpón del cliente con pollos de engorde Cobb500.

Para llegar exitosamente al final del proceso de crianza, se debe monitorear y regular diferentes parámetros durante las 24 horas del día; como la alimentación, hidratación, temperatura, humedad y luz. Además, se requiere un constante registro de las condiciones actuales de los pollos como peso actual, temperatura, dosis administradas, entre otros parámetros sugeridos por el fabricante, con el fin de monitorear el desempeño, rentabilidad del lote, y hacer proyecciones de flujo de caja [4]. La intensidad de los cuidados necesarios conlleva a gastos en cuidadores y sus descuidos humanos generan pérdidas.

Este proyecto tiene como objetivo reducir la intervención manual en el monitoreo de parámetros ambientales y estado de salud de los pollos de engorde, debido a la intensidad de cuidados que conlleva el control de estos parámetros; es agotador y poco eficiente llevar este proceso totalmente de forma manual.

1.2 Justificación del problema

Según la Corporación Nacional de Avicultores del Ecuador (CONAVE), la avicultura ecuatoriana es de gran importancia socioeconómica al ser la carne de pollo una de las proteínas de mejor calidad y de más fácil acceso para los ecuatorianos [5]. Por lo que muchos ecuatorianos emprenden en el sector avícola, criando pollos de engorde de distintas razas, tras adecuar un galpón donde se pueda alojar las aves bajo diferentes parámetros que varían basándose en la raza de pollo de engorde y son especificados en el manual de crianza. El control de estos factores influye directamente al estrés de los animales, cuyo incremento provoca que sean más susceptibles a enfermedades.

Figura 1.2 Crianza de pollos de engorde Cobb500 [6].

A pesar de la gran importancia del sector avícola para la economía del país, no hay sistemas locales comerciales para pequeños y medianos productores, que permitan facilitar y dinamizar la crianza de pollos de engorde de diferentes estirpes. El pollo de engorde de estirpe Cobb500, tiene el menor costo de peso vivo producido, su alimentación de bajo costo es más eficiente y posee la mejor tasa de crecimiento; a pesar de su popularidad y la cantidad de manuales de información que otorga la empresa Cobb para la óptima crianza de esta especie, el proceso de crianza puede ser difícil debido a la cantidad de parámetros y factores a controlar que necesitan un constante monitoreo [7].

Por ende, el presente proyecto propone el diseño de un sistema que permita el monitoreo de los parámetros ambientales y estado de salud de los pollos de engorde, orientado a pequeños y medianos productores; usando sensores, actuadores y herramientas de inteligencia artificial, controlado desde una aplicación de fácil uso y ofreciendo un registro en una base de datos para el monitoreo del desempeño del galpón.

1.3 Objetivos

1.3.1 Objetivo general

Diseñar un sistema de control y monitoreo de parámetros ambientales y detección de enfermedades en pollos de engorde Cobb500 para asegurar su óptimo desarrollo.

1.3.2 Objetivos específicos

- Diseñar un sistema de control y monitoreo de parámetros ambientales, como la temperatura y la humedad del galpón, en base a la semana de crecimiento del pollo de engorde.
- Identificar pollos enfermos de la enfermedad Newcastle en la bandada usando métodos de inteligencia artificial, previniendo posibles contagios masivos y pérdidas.
- Diseñar una aplicación móvil que despliegue los datos de los sensores, permita el control de los actuadores, registro diario y envío mensajes de alerta a los cuidadores y gerente.
- Seleccionar un motor que permita la automatización de la apertura y cierre de cortinas para regular el flujo del aire dentro del galpón

1.4 Marco teórico

La avicultura representa 3% del PIB nacional, ocupando 23% del PIB agropecuario**;** generando alrededor de 300000 empleos formales y un valor bruto anual de producción de 3700 millones de dólares en toda la cadena productiva, para el año 2021 [8]. Según las estadísticas de la Corporación

Nacional de Avicultores del Ecuador (CONAVE), en el 2021 el consumo de carne de pollo correspondió a 480 miles de toneladas, como se observa en la Figura 1.3 [9].

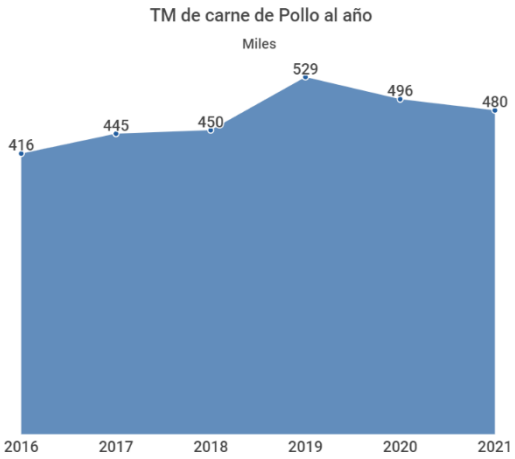


Figura 1.3 Toneladas de carne de pollo al año consumidas en Ecuador [9].

Según un reporte de la Superintendencia de Control del Poder de Mercado, la producción avícola en el Ecuador corresponde a una de las actividades productivas de mayor importancia para la economía, dividida en dos segmentos productivos: la producción de carne de pollo y de huevos; destacándose la crianza de pollos para el consumo de su carne al ser una de las proteínas más consumidas, por lo que se ha convertido en una fuente de empleo para varias familias. El alto consumo de carne de pollo se debe a la reducción de los costos o materia prima dentro del mejoramiento de procesos productivos para la crianza de pollo y su bajo precio. Además, el consumo de carne de pollo es preferido ante el consumo de carnes rojas al considerarse menos nociva para la salud [10].

1.4.1 Sectorización de la crianza de pollos de engorde en Ecuador

Según el Ministerio de Agricultura y Ganadería (MAG) en el año 2019, existían 1819 granjas avícolas en el Ecuador, entre ellas, 90 empresas registradas en la Superintendencia de Compañías; entre grandes,

pequeñas, medianas y microempresas, de las cuales, en su mayoría se concentran en Guayas, Pichincha y Tungurahua [11].

1.4.2 Crianza de pollos de engorde Cobb500

La estirpe Cobb500 es el pollo de engorde más eficiente del mundo y tiene la ventaja competitiva de menor costo por kilogramo o libra de peso vivo producido para los criadores de pollos de engorde [7]. La empresa Cobb dispone de los manuales necesarios para la crianza de los pollos de engorde Cobb500.

Actualmente, la granja del cliente ubicada Vía a la Costa, posee un área de 64 m^2 y lleva un proceso de crianza de pollos de engorde Cobb500 de forma manual, donde la corriente de aire es regulada por el cierre y apertura manual de las cortinas de plástico con una polea. El monitoreo de parámetros ambientales es dado por un solo sensor de humedad y temperatura ubicado fuera de la cama de los pollos, donde los datos son visualizados mediante una pantalla LCD.

Además, la iluminación del galpón es controlada mediante focos encendidos regularmente a las 18:00 de cada día, sin embargo, se tiene focos de respaldo con baterías cargadas mediante energía solar que son encendidos en caso de un fallo eléctrico.

El registro de las condiciones de la bandada diario es fundamental a la hora de controlar el correcto crecimiento y rentabilidad del lote, permitiendo realizar análisis de costos y flujo de caja [4]; actualmente los cuidadores realizan este registro de manera manual y semanalmente.

Figura 1.4. Crianza actual de los pollos de engorde en el galpón del cliente.

Según el manual de crianza, el galpón a alojar los pollos debe tener los siguientes componentes claves: techo con buen aislamiento, sistema de calefacción; sistema de ventilación que brinde el oxígeno suficiente, mantenga la humedad en la cama y su capacidad de enfriamiento sea la óptima para las aves, además un sistema de iluminación que brinde una distribución de luz uniforme al nivel de los pollos.

Para una ventilación ideal, el galpón debe encontrarse en un lugar con alto nivel de corriente natural de aire; para la iluminación, se recomienda que se encuentre orientado en el eje este-oeste para que haya poca incidencia de la luz solar en las paredes laterales en el momento más caluroso del día, minimizando la fluctuación de temperatura que influye directamente en el bienestar y tasa de crecimiento de las aves.

Otro factor por regular es la temperatura, usando termostatos o sondas de temperatura al nivel de las aves y en el centro del galpón; además, la temperatura no debe tener variaciones mayores a 2 ℃, y en promedio

debería ser alrededor de 30 a 32 ℃ y 30-50% HR. La temperatura depende de la humedad relativa del galpón y de su edad en días, como se describe en la Tabla 1.1 [4].

Tabla 1.1. Pautas de curva de temperatura con base en la humedad relativa del galpón [4].

	Humedad relativa [%]				
Edad [días]	**30%**	**40%**	**50%**	**60%**	**70%**
0	34 ℃	33 ℃	32 ℃	31 ℃	30 ℃
7	32 ℃	31 ℃	30 ℃	29 ℃	28 ℃
14	29 ℃	28 ℃	27 ℃	26 ℃	25 ℃
28 kg/m^2	25 ℃	24 ℃	23 ℃	22 ℃	21 ℃

Adicionalmente, la temperatura depende de la densidad de población, cuando esta supera los 28 kg/m^2, ya que puede haber exceso de calor que si no es eliminado aumentará el jadeo de las aves, disminuyendo su ingesta de alimento y su aumento de peso. Sin importar la edad del pollo de engorde, para densidad de población mayor a 28 kg/m^2, se usa la Tabla 1.2 [4].

Tabla 1.2. Guía de temperatura con base en la densidad de población [4].

Densidad kg/m^2	**Temperatura objetivo Rango (℃)**
28	22-24
30	21-23
32	20-22
34	19-21
36	18-20
38	17-19
40	16-18
42	15-17
42+	14-16

El control de la temperatura de la cama de 5 cm de espesor es importante, ya que influye directamente en el peso final de los pollos, obteniendo que el peso final aumente de 2.098 g a 2.142 g cuando la temperatura de la cama es mayor a 28 ℃, como se observa en la Figura 1.5 [4].

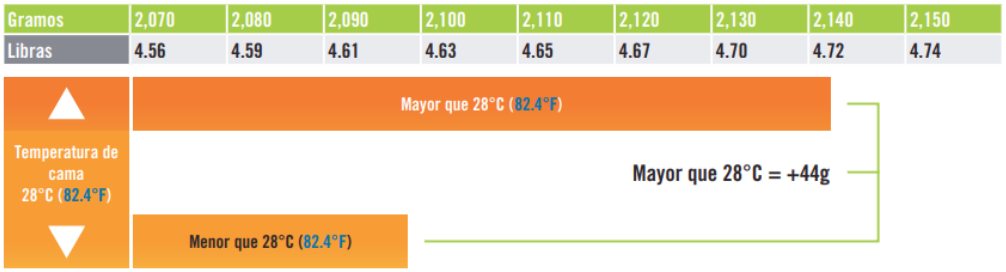


Figura 1.5. Peso final en función de la temperatura de la cama [4].

1.4.3 Enfermedades en pollos de engorde Cobb500

Las enfermedades en pollos de engorde Cobb500 es uno de los problemas más grandes debido principalmente, al desconocimiento del cuidador a la hora de identificarlas a través de simple observación. Para la prevención de estas enfermedades se recomienda: mantener limpia la zona alrededor de los gallineros, evitar mezclar grupos de aves, seguir el programa de iluminación recomendado, mantener buena ventilación, cama seca y temperatura apropiada en los gallineros, eliminar las aves muertas rápidamente, etc. [12]. Entre las enfermedades se destacan:

1.4.3.1 Bronquitis infecciosa

La bronquitis infecciosa es una enfermedad causada por el Coronavirus, cuyos síntomas son: respiración laboriosa, jadeo, ahogos, estornudos y estertores, mucosidad, secreciones de los ojos y aberturas nasales. Esta enfermedad es altamente infecciosa ya que puede transmitirse a través del aire, además de difundirse entre medios mecánicos como la ropa. Su índice de mortalidad puede ser de 5 a 60%.

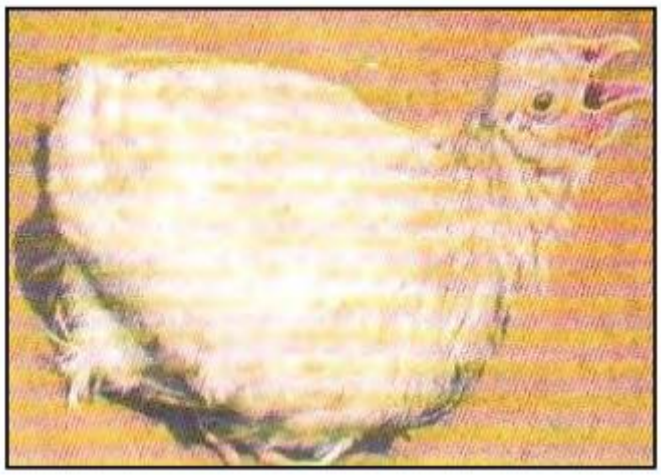


Figura 1.6 Pollo con dificultad respiratoria [12].

1.4.3.2 Newcastle

La enfermedad de Newcastle es causada por el agente Orthomixovirus y Paramyxovirus; los síntomas son similares a otras enfermedades respiratorias como tos, ahogo, respiración agitada, etc. Además de presentar parálisis de una o ambas alas, patas, cabeza y cuello torcido, se suele doblar la cabeza hacia atrás, sobre la espalda o hacia adelante, entre las patas. El índice de mortalidad varía desde 0 hasta la pérdida total del lote, el virus puede transmitirse a través de las descargas nasales, excrementos de aves infectadas y equipos contaminados.

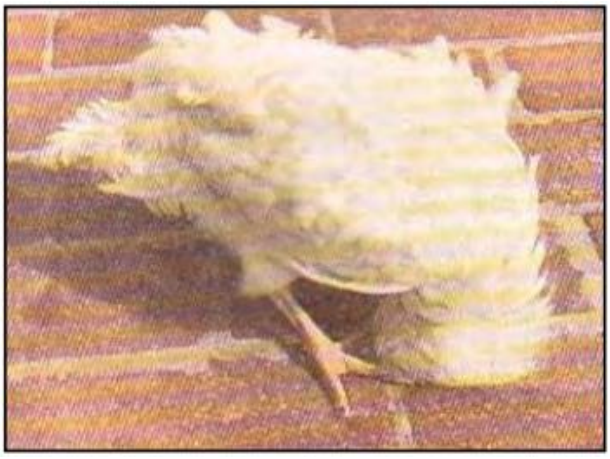


Figura 1.7. Pollo con torticolis a causa de la enfermedad de Newcastle [12].

1.4.4 Estado del arte

Dentro del país no se registra sistemas comerciales disponibles, pero fuera del país si hay proyectos para la automatización de criaderos de pollos como BAKU, el cual provee un sistema inteligente que integra software y hardware para el monitoreo de los animales y las condiciones ambientales. Consta de una aplicación que ayuda a optimizar su producción de pollos de engorde, siendo esta portable y simple de usar. Se utilizan sensores dentro de un instrumento portátil los cuales pueden ser accedidos a través de la aplicación, internet o cualquier PC, y los datos son almacenados en una base de datos. Además, el sistema alerta a través de mensajes de texto anormalidades en los parámetros ambientales. El servicio tiene un costo de 17 centavos por día [13].

Figura 1.8 Dashbord del proyecto BAKU [13].

También existen otros proyectos similares como el de Geetanjali A. Choukidar *et al.,* el cual tiene un sistema similar con sensores inalámbricos en combinación a una red GPRS usada para el monitoreo de los parámetros ambientales como la temperatura, humedad y amoniaco. Para el monitoreo de estos parámetros se usa una página web, y este consta de actuadores que permite controlar dichos parámetros, todo esto conectado a una Raspberry Pi [14].

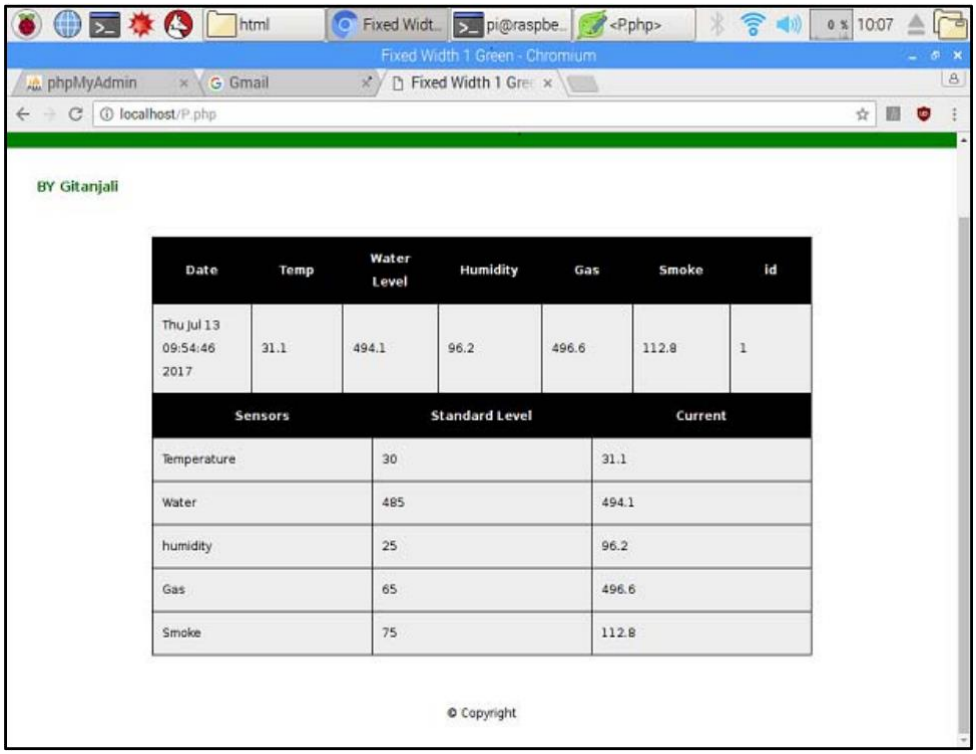


Figura 1.9. Dashboard para monitoreo de parámetros [14].

Para la detección de enfermedades en pollos de engorde se han tomado varios enfoques; como el análisis de heces de aves para determinar una relación entre heces anormales y enfermedad de los pollos de engordes usando un modelo de Deep Convolutional Neural Network (DCNN) realizado por Wang, *et al.* [15] y la detección de características distintivas en pollos de engorde inoculados con gripe aviar para automatizar la clasificación entre pollos sanos y enfermos usando un modelo de Support Vector Machine (SVM), realizado por Zhuang, *et al.* [16].

También autores como Longshen, *et al.* [17] han utilizado grabaciones de la vocalización de pollos de engorde para detectar sonidos anormales, como tos o ronquido, que pueda indicar problemas respiratorios en las aves; mediante un algoritmo de reconocimiento que usa la Transformada Ondícula de Coeficientes Cepstrales de las Frecuencias de Mel (WMFCCs), Distancia de Correlación del Criterio Fisher (CDF) y el modelo oculto de Markov (HMM). En cambio, Akomolafe, *et al.* [18] parte de imágenes de pollos de criaderos locales e imágenes en línea de pollos enfermos y sanos, para determinar el contagio de la enfermedad de

Newcastle o la influenza aviar utilizando dos configuraciones de modelo CNN.

CAPÍTULO 2

2. METODOLOGÍA

En esta sección se muestra el proceso seguido para llevar a cabo la solución, partiendo de los requerimientos del cliente y limitaciones del lugar de trabajo. Cada subsistema de la solución es detallado, junto a los métodos usados y el proceso de selección de componentes.

2.1 Proceso del diseño

El proceso de diseño consiste en las diferentes etapas del proyecto para llevar a cabo la solución, se determinaron cinco etapas principales. En la Figura 2.1, se muestra cada etapa desglosada los pasos necesarios para el cumplimiento de los objetivos del proyecto.

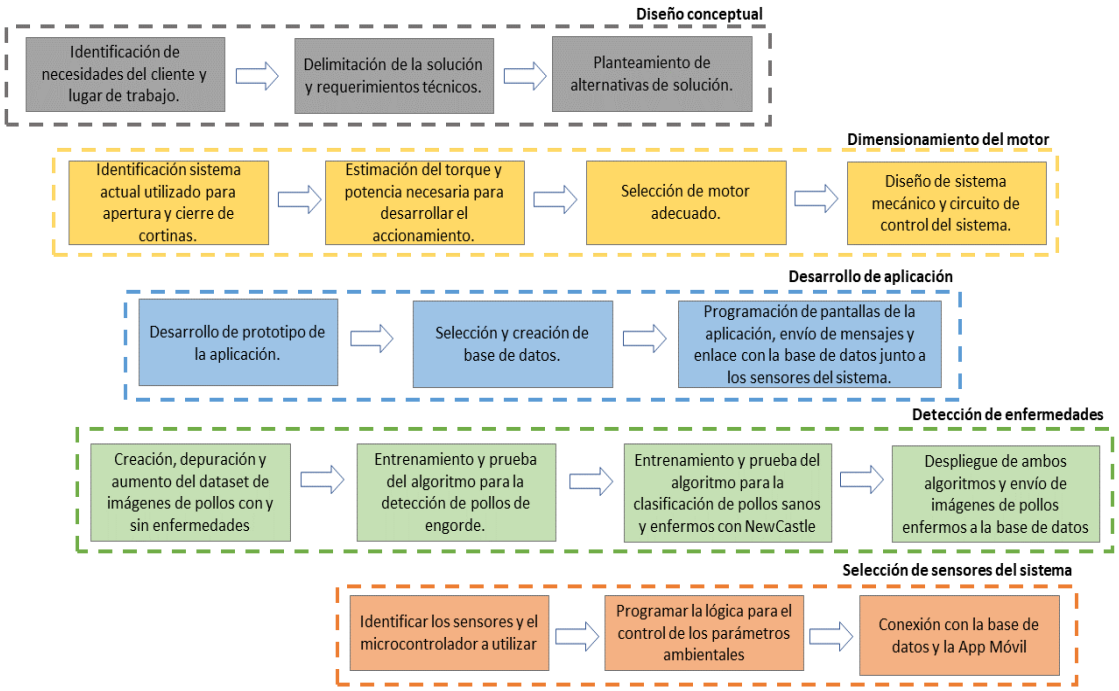


Figura 2.1 Diagrama del proceso de diseño.

2.2 Requerimientos del diseño

Los requerimientos identificados ante las necesidades del cliente se dividieron en los siguientes puntos:

- Conocer en tiempo real las condiciones ambientales del lugar.
- Recibir alertas en posible caso de detectar un pollo enfermo.
- Accionar motores que realicen la apertura de las cortinas del galpón, con el fin de regular de mejor manera las condiciones ambientales y su cierre por las noches o cuando sea necesario.
- Registrar de forma ágil los parámetros más importantes.
- Encender y apagar las luces a una determinada hora.

Tras conocer las necesidades del cliente, se definieron los requerimientos del proyecto junto con las restricciones dadas por la naturaleza de los pollos, además del espacio y ubicación del galpón.

2.2.1 Requerimientos para el desarrollo de la solución

- Se requiere de sensores que entreguen valores con relativamente buena precisión y exactitud, ya que las variaciones de temperatura provocan efectos negativos en la salud de los pollos.
- Conexión a internet y a la red eléctrica.
- Se requiere sensores inalámbricos con el fin de reducir el uso de cables y facilitar la comunicación.
- Sensores de humedad con profundidad de medición de hasta 5 cm, para medir la humedad de la cama de los pollos de engorde.
- Sensores de temperatura deben tener un rango de medición 15 a 40 ℃.
- Se requiere de una base de datos para alojar la información de los registros diarios y la información disponible en las guías.
- Se requiere una aplicación móvil de fácil uso e intuitiva.
- Se requiere de un controlador que permita el accionamiento de los actuadores, en base a las órdenes enviadas por la aplicación.
- Se requiere la implementación de un modelo de inteligencia artificial para la identificación de posibles pollos enfermos.

2.2.2 Restricciones para el desarrollo de la solución

- La cámara o cámaras para detección de enfermedades deben poder monitorear todas las aves del galpón.
- Sensores no deben intervenir con la vida diaria de los pollos, por lo que deben ser ubicados de forma discreta en el galpón.
- Debe tener conexión a internet para que se alojen los datos y se guarde la información importante dada por los sensores para el registro diario.
- Los actuadores no pueden ser ruidosos ya que induciría estrés en los pollos de engorde.
- El espacio del galpón es de 64 m^2.

2.3 Diseño conceptual

Tras identificar las necesidades del cliente y el lugar de trabajo, se delimitaron los requerimientos de la solución, para luego elaborar el diseño conceptual de la solución compuesto de dos ramas: un sistema IoT para el control de sensores y accionamiento de los actuadores manejado por la aplicación móvil, y el algoritmo de detección de enfermedades (Figura 2.3). La Figura 2.2 muestra la conexión entre los diferentes subsistemas del sistema IoT, el cual es descrito detalladamente a continuación.

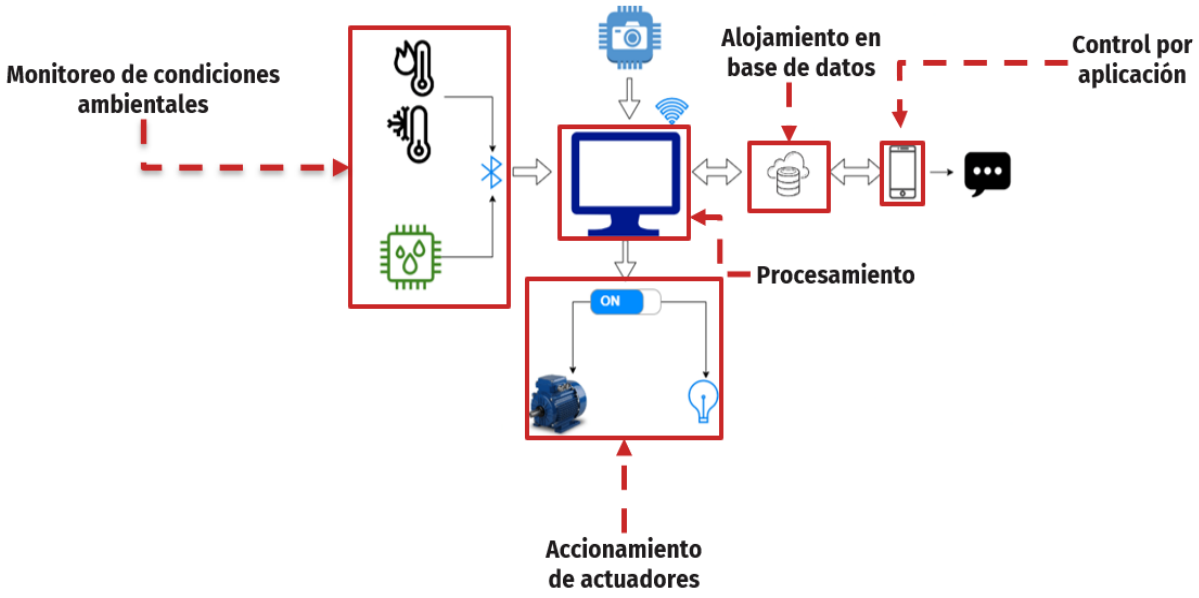


Figura 2.2 Diagrama del diseño conceptual para el sistema IoT de la solución.

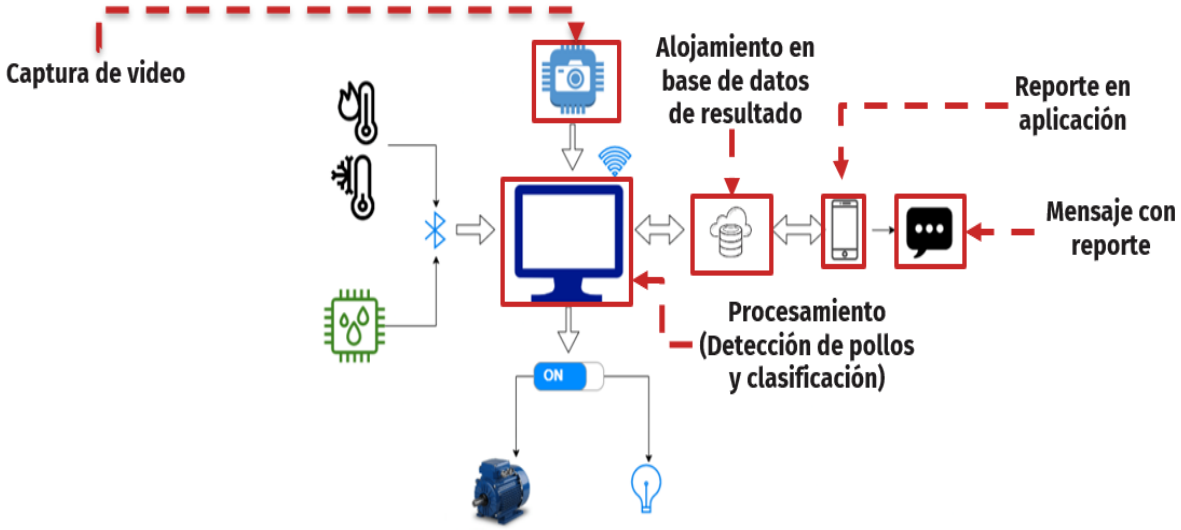


Figura 2.3 Diagrama del diseño conceptual para el sistema de detección de enfermedades.

2.3.1 Sistema IoT

Para el sistema IoT se escogió cuatro sensores de temperatura y humedad, los cuales irán conectados a un módulo WiFi que permite conectarlos a un

controlador de manera inalámbrica, esto con el fin de evitar el uso de cables.

El controlador permite el accionamiento de los actuadores a partir de las señales recibidas por los sensores, enviando como salida una señal para la apertura o cierre de las cortinas y el encendido o apagado de las luces del galpón. El control de los actuadores puede ser de forma manual o automática, con base en lo elegido por el usuario mediante la aplicación móvil desarrollada.

2.3.2 Aplicación móvil

La aplicación móvil permite el inicio de sesión para que sólo usuarios autorizados tengan acceso al control y revisión de reportes. Esta aplicación se conecta a una base de datos que permite visualizar los valores de los sensores además de cambiar el estado de los actuadores. Este estado se lo puede cambiar de manera manual o automática, donde en caso de escogerse este último se requiere de la información del peso, cantidad y edad en días de los pollos. El registro de parámetros importantes se realiza mediante una ventana de la aplicación, que respalda los registros diarios en la base de datos para su posterior análisis.

Además, las imágenes de los pollos enfermos del galpón, detectados por el algoritmo de inteligencia artificial, son desplegadas en una ventana de la aplicación junto con un mensaje de alerta al usuario.

2.3.3 Sistema mecánico

Para el diseño mecánico se optó por rediseñar el sistema actual, utilizando las varillas de acero del sistema manual actual para el cierre y apertura de las cortinas, pero haciendo que éstas sean desplazadas a través de un sistema riel-carrito, accionado por un motor; el cual controlará dos cortinas del galpón en conjunto a un sistema de poleas para la transmisión de potencia.

2.3.4 Algoritmo de detección de enfermedades

Para el diseño del algoritmo de detección de enfermedades, se formó un *dataset* de pollos sanos y enfermos con la enfermedad de Newcastle, usando imágenes disponibles en internet e imágenes del cliente. Además, un algoritmo de inteligencia artificial identifica los pollos dentro de un galpón y otro clasifica cada pollo identificado por el algoritmo previo, en pollo sano o enfermo por la enfermedad de Newcastle. Finalmente, luego de la detección dada a una frecuencia determinada, se envían las imágenes de los pollos enfermos a la base de datos para un posterior reporte al usuario, mediante la aplicación móvil.

2.4 Diseño del sistema mecánico

2.4.1 Rediseño del sistema mecánico actual del galpón

Se identificó que el sistema mecánico actual consta de poleas y una manivela encargada del control manual de las cortinas para regular los parámetros de temperatura y humedad del galpón, como se muestra en la Figura 2.4. Este sistema tiene un diseño con alta resistencia al subir o bajar, lo cual causa estancamientos que pueden reducir la vida útil del mecanismo; por lo tanto, es necesario diseñar un sistema que sea eficiente, no tenga trabas y sea de larga de duración. Además, debe ser automático, manejado con base en las condiciones ambientales, con el fin de mantener siempre un ambiente agradable para los pollos del galpón.

Figura 2.4 Manivela usada para el control manual de las cortinas del galpón.

El nuevo sistema mecánico consiste en un sistema riel-carrito con poleas y cables, accionado por un motor para el levantamiento o cierre de las cortinas del galpón. Primero se seleccionó los componentes del sistema riel-carrito, para obtener su peso estimado y luego calcular el torque y la potencia necesaria para desarrollar el accionamiento por un motor.

2.4.2 Selección de componentes

Para el sistema de riel-carrito se escogió primero el riel con modelo 164V, de la marca ARENA (Tabla 2.1).

Tabla 2.1 Modelos de riel de la marca ARENA [19].

Artículo	A	B	C	D	E	F	L (metros)	Peso(kg)
168A	28	22,5	4	10,5	0,9	8	6	4,2
164A	34	28	6	13	1,25	10	6	7
164V	33,5	26	5	12	1,25	11	6	6
170L	41	35	7	15	1,6	13	6	10,5
170R	40	34	7	15	2	13	6	13,2
172A	43,5	40	7	17	2	13	6	15
174A	49	46,5	7	19,5	2	14	6	17,5
174R	55	52	8	19	2,25	17	6	17,5
700	67	72	13	25	3,2	17	6	17,5

* Medidas Expresadas en Milímetros. ** Las medidas son aproximadas. Las especificaciones pueden ser modificadas sin previo aviso.

Para el carrito se tiene que escoger el mismo modelo del riel, el 164V (Tabla 2.2), para que pueda acoplarse perfectamente al riel. Estos carritos tienen un agujero en el centro al cual se puede acoplar accesorios, como se observa en la Figura 2.5, en el cual van acopladas las varillas de acero a las que van sujetadas las cortinas de plástico del galpón.

Figura 2.5 Carrito escogido para el sistema mecánico [20].

Tabla 2.2 Modelos de carrito de la marca ARENA [20].

Artículo	A	B	C	D	E	G	H	J	K	L	M	O	P	R	S	T	U	W	Rd.	Tca.	Pcs.	Ny.	Ac.(kg)
168	53	25	9,5	20	2,5	22	1/4	46	22	46	5	7	70	40	1,5	4	9,5	5	Plano	1/4	50	4,7	6
164	61	31	15	25	3,2	22,2	5/16	50	26	53	5,5	8	91	53	3,5	5,5	9	6,8	Plano	5/16	50	8	10,9
164 V	60	31	15	24	3,2	22,2	5/16	50	28	53	6	8	91	53	3,5	5,5	9	6,8	Redondo	5/16	50	8,7	10,5
170	82	38	19	32	4,8	22,2	7/16	66	35	79	7	11,3	91	53	3,5	6	12,3	9	Plano	7/16	20	6,7	10
172	87	39	19	38	4,8	22,2	7/16	66	38	79	8	11,3	91	53	3,5	6	12,3	9	Plano	7/16	20	--	12
174	95	45	19	45	4,8	25,4	7/16	66	44	79	10	11,3	134	85	4	6	19	9	Plano	7/16	20	--	19

* Medidas Expresadas en Milímetros. ** Las medidas son aproximadas. Las especificaciones pueden ser modificadas sin previo aviso.

2.4.3 Selección del motor

Para la selección del motor se toma en cuenta el peso a levantar según los componentes a utilizar, además del tiempo estimado para levantarlo; el cual puede ser de un minuto aproximadamente.

Debido a que se necesita una alta fuerza y un tiempo no tan elevado se optó por buscar un motorreductor, el cual tiene alto torque, pero una velocidad no demasiado alta.

2.4.3.1 Dimensionamiento del motor

Para sacar el peso de los componentes se tomaron los valores de las páginas de los proveedores, obteniendo los siguientes valores:

Masa de un carrito de acero (m_c): 10,5 kg
Cantidad de carritos de acero por cara (n_c): 2
Masa por metro de la varilla de acero 8 mm de diámetro (m_{nv}): 0.395 $\frac{kg}{m}$
Masa de la cortina (m_{ct}): 0,9 kg

El galpón cuenta con cuatro caras, donde A es la cara de menor ancho debido a la puerta ubicada en esta cara. El resto de las caras (B, C y D) tienen las mismas dimensiones, como se muestra en la Figura 2.6 y la Figura 2.7.

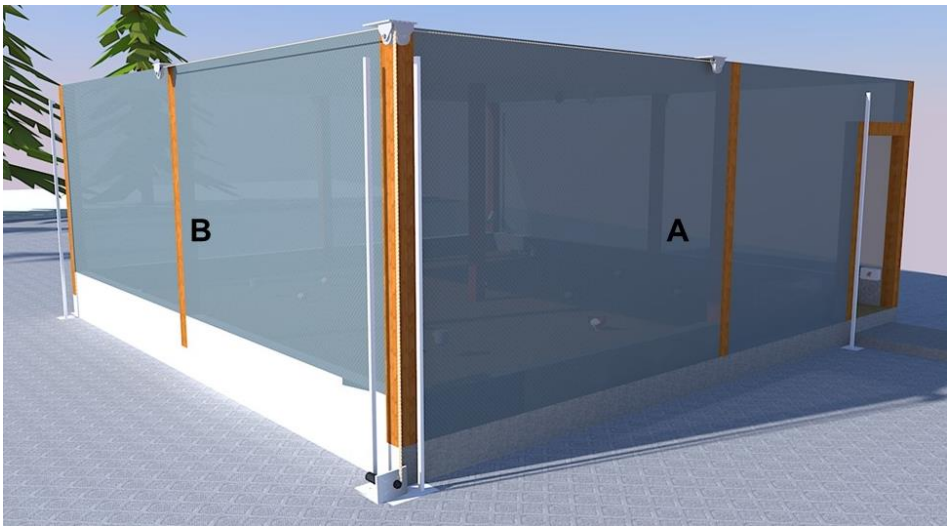


Figura 2.6 Caras A y B del galpón.

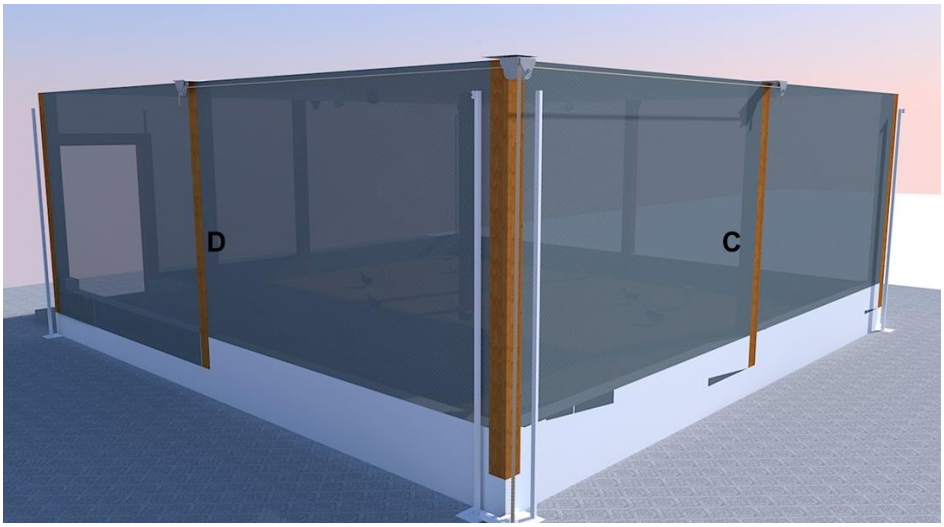


Figura 2.7 Caras C y D del galpón.

Longitud de varilla de cara A (l_{vA}): 6.42 m

$$m_{vA} = m_{nv} * l_{vA} \quad (2.1)$$

Con la ecuación 2.1, se obtiene la masa de la varilla de la cara A.

$$m_{vA} = 0.395 \frac{kg}{m} * 6.42\, m$$

$$m_{vA} = 2.536\ [kg]$$

Longitud de varilla de cara B, C y D (l_{vX}): 7.68 m

Masa de cara B, C o D con varilla de 7.68 m (m_{vX}):

$$m_{vX} = m_{nv} * l_{vX} \quad (2.2)$$

Con la ecuación 2.2, se obtiene la masa de las caras B, C o D.

$$m_{vX} = 0.395 \frac{kg}{m} * 7.68\, m$$

$$m_{vX} = 3.03\ [kg]$$

$$p_{cX} = (n_c * m_c + m_{vX} + m_{ct}) * 9.81 \frac{m}{s^2} \quad (2.3)$$

Con la ecuación 2.3, se determina el peso total a levantar para la cara B, C o D.

$$p_{cX} = (n_c * m_c + m_{vX} + m_{ct}) * 9.81 \frac{m}{s^2}$$

$$p_{cX} = (2 * 10.5\,kg + 3.03\,kg + 0.9\,kg) * 9.81 \frac{m}{s^2}$$

$$p_{cX} = 244.56\,[N]$$

Usando el peso a levantar por la cara, se debe considerar que la fuerza luego de la polea cambia por la fricción de la cuerda al deslizarse por la polea. Esta fuerza es calculada mediante la fórmula Euler-Eytelwein (ecuación 2.4), donde α es el ángulo recorrido por la cuerda, μ el coeficiente de fricción entre la cuerda y la polea, mientras que T_1 y T_2 corresponden a las tensiones [21]. El sistema de poleas se detalla en la Figura 2.8.

$$T_2 = T_1 e^{\mu\alpha} \tag{2.4}$$

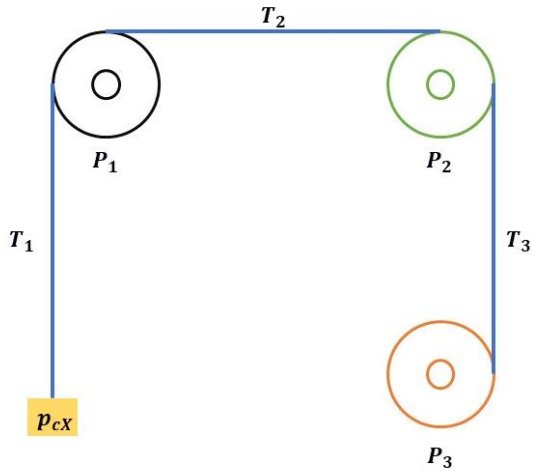


Figura 2.8 Diagrama del sistema de poleas.

Sabiendo que la cuerda es de fibra alquitranada y la polea es de aluminio [22], el valor de μ corresponde a:

$$\mu = 0.18$$

Mientras que la T_1 corresponde al peso por levantar de las caras (p_{cX}). Por ende, se tiene que:

$$T_1 = p_{cX} = 244.56\ [N]$$

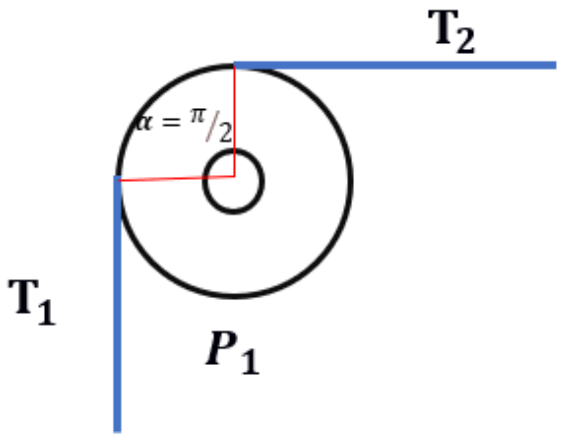


Figura 2.9 Análisis de la polea P_1.

Analizando la polea P_1, con el diagrama de la Figura 2.9, se determina T_2 usando la ecuación 2.4.

$$T_2 = 244.56 * e^{0.18(\frac{\pi}{2})}$$
$$T_2 = 324.48\ [N]$$

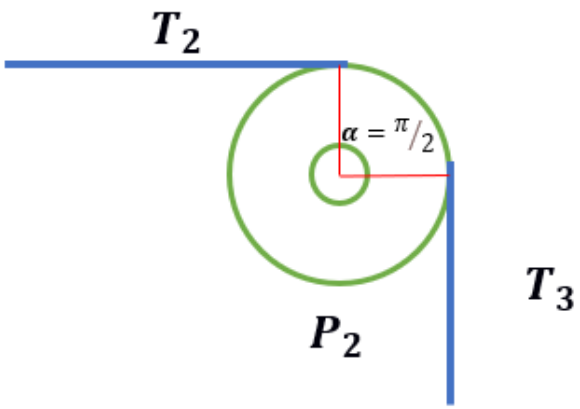


Figura 2.10 Análisis de la polea P_2.

De la misma forma se analiza la polea P_2, con el diagrama de la Figura 2.10, se determina T_3 usando la ecuación 2.4.

$$T_3 = 324.48 * e^{0.18(\frac{\pi}{2})}$$
$$T_3 = 430.51\ [N]$$

Con un factor de seguridad de 1.5:

$$p_{fsX} = T_3 * 1.5$$
$$p_{fsX} = 430.51\ N * 1.5$$
$$p_{fsX} = 645.76\ [N]$$

Peso con factor de seguridad por levantar por las dos caras:

$$p_{tX} = 2 * p_{fsX}$$
$$p_{tX} = 2 * 645.76\ N$$
$$p_{tX} = 1291.51\ [N]$$

Tiempo estimado en levantarse: 90 s

Altura por levantar: 2.6 m

$$P = F * v \tag{2.5}$$

La potencia necesaria para el funcionamiento del sistema se obtiene con la ecuación 2.5.

$$P = 1291.51\ N * \frac{2.6m}{90\ s}$$

$$P = 37.31\ W$$

$$P = 0.05\ HP$$

La potencia máxima, tomando en cuenta la fuerza para levantar la cara de mayor fuerza, necesaria para levantar dos caras puede suplirse usando un motor de 12 V a 52 W modelo MM201, de las características descritas en la Tabla 2.3.

Tabla 2.3 Motor escogido para el levantamiento de las cortinas [23].

Model	VOLTAGE (VDC)	CURRENT (mA)	SPEED (r/min)	TORQUE (Nm)	BRAKE TORQUE (Nm)	RATIO	Power (W)	LENGTH (in)
MM201	12	13000	11	45	56.5	291	52	7.8
MM201A	24	6500	11	45	56.5	291	52	7.8

Radio de polea (r_p): 17.5 mm

$$F = \frac{Torque}{r_p} \tag{2.6}$$

Con la ecuación 2.6, se determina la fuerza lineal usando los siguientes datos de la geometría de la polea.

$$F = \frac{45\ N.m}{17.5 * 10^{-3}}$$

$$F = 2571.43\ N$$

Comparando la fuerza lineal que puede dar el motor y la fuerza lineal que se requiere para levantar dos cortinas del galpón, se tiene la siguiente relación:

$$F > p_{tx}$$

$$2571.43\ N > 1291.51\ N$$

Por ende, el motor elegido puede ser utilizado para la aplicación.

2.5 Diseño de la aplicación móvil

2.5.1 Prototipado

Para el diseño de la aplicación se realizó el prototipo usando la herramienta Whimsical, permitiendo ver las diferentes pantallas y el flujo del funcionamiento (Figura 2.11). El detalle del prototipado de cada pantalla y su programación se encuentra en la sección de Apéndices.

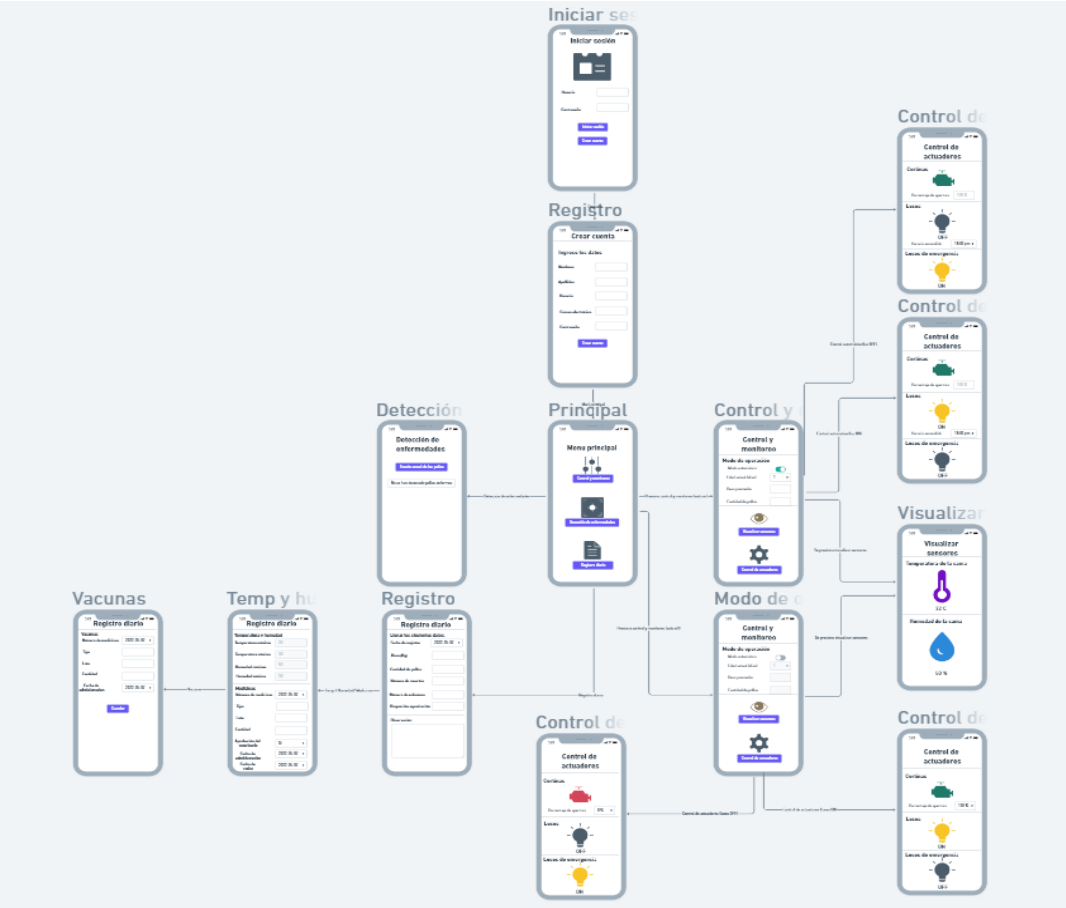


Figura 2.11 Prototipado de la aplicación móvil.

2.5.2 Diagrama entidad-relación

Se realizó el diagrama entidad-relación (Figura 2.12) con el fin de configurar la base de datos e identificar los datos a guardar y controlar en la aplicación.

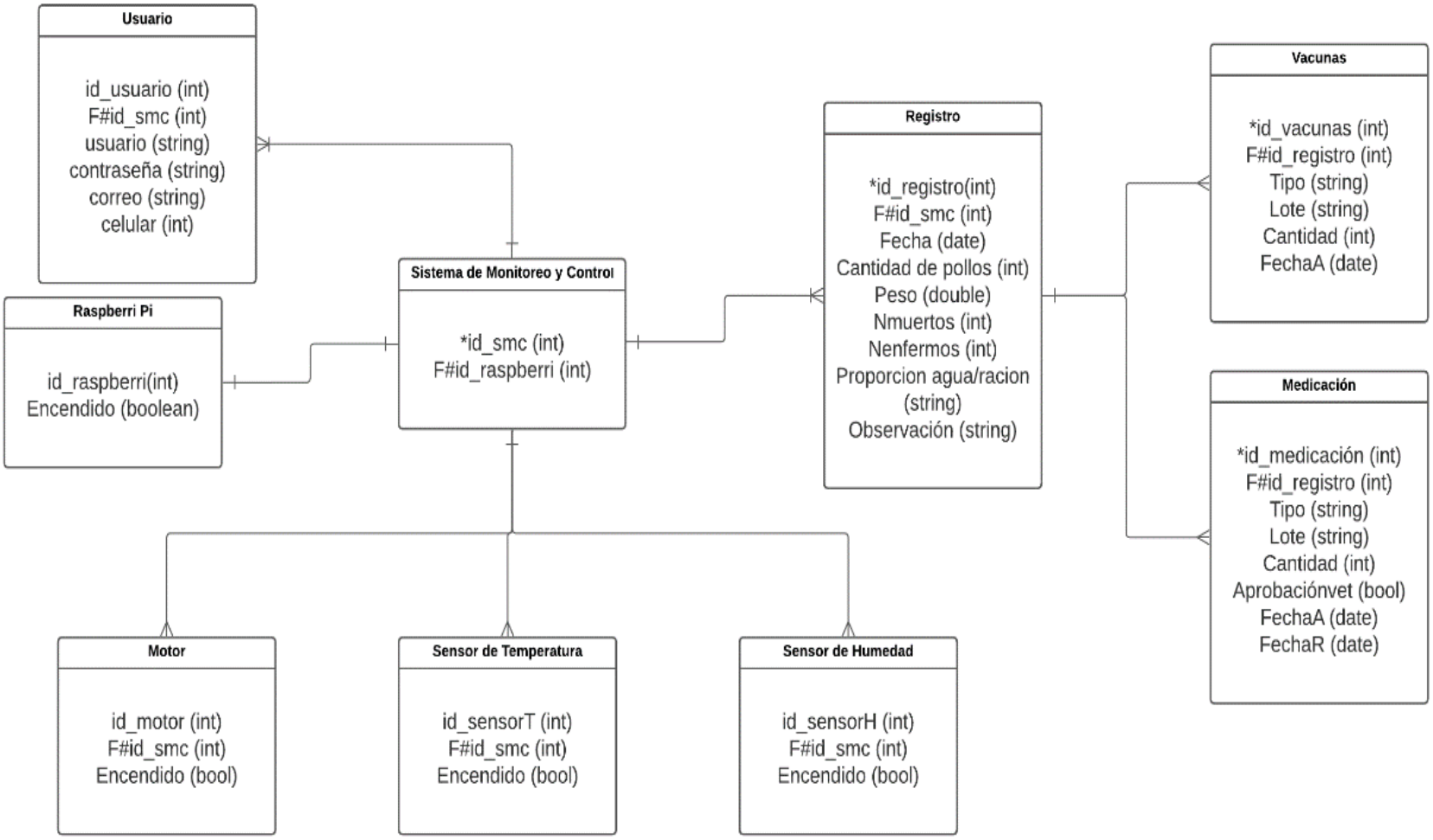


Figura 2.12 Diagrama entidad-relación.

2.6 Diseño de algoritmo para la detección de enfermedades

Debido a las numerosas pérdidas de un gran porcentaje del lote por enfermedades, especialmente por la enfermedad de Newcastle, surgió la necesidad de implementar un sistema de detección de enfermedades mediante métodos de inteligencia artificial, basándose en detectar la torticolis, síntoma característico de esta enfermedad.

La solución fue dividida en un algoritmo para la clasificación de pollos, encargada de detectar los pollos en un *frame* de video en tiempo real, para que luego las imágenes de los pollos detectados sean enviadas a un algoritmo de clasificación. El algoritmo de clasificación etiqueta a los pollos como sanos o enfermos.

Para el algoritmo de detección se utilizó el modelo YOLOv5 s, mientras que para la clasificación se optó por entrenar un modelo CNN. Además, para la entrada de video en tiempo real, se optó por escoger dos cámaras turbo bala (Tabla 2.4) usadas para vigilancia, las cuales se conectan a la Raspberry Pi 4 configurando la DVR a la que se encuentra conectada las cámaras, a la misma red de la Raspberry Pi 4 para acceder a los videos.

Tabla 2.4 Características de la cámara [24].

Cámara	
	
Modelo	Cámara Bala Turbo HD THC-B110-P
Características	• Interior/exterior IP66 • Material: plástico • Resolución máxima: 1280 x 720 (1MP) • Rango IR: Hasta 20 m IR EXIR • Lente fijo: 2.8 mm (ángulo de apertura 92 grados) • dwDR • Soporta 4 tecnologías (TVI/AHD/CVI/CVBS)

2.6.1 Creación del *dataset* de pollos enfermos y sanos

Se formó un *dataset* de pollos sanos y enfermos usando imágenes en formato jpg, obtenidas de internet, Roboflow, e imágenes tomadas propias por una cámara de celular que permitieron entrenar al algoritmo con tomas del entorno en el que se va a desenvolver. Se obtuvo como resultado un total de 662 imágenes recopiladas.

Para la detección de pollos con YOLOv5, se hizo el etiquetado de cada una de las imágenes usando la herramienta MakeSense. Este etiquetado dio

como resultado archivos .txt llamados con el mismo nombre de las imágenes, donde en cada fila del archivo se tiene la estructura de la Figura 2.13 y el número de líneas depende de la cantidad de pollos etiquetados en la imagen.

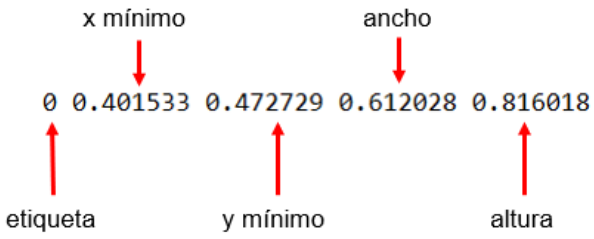


Figura 2.13 Estructura de las etiquetas de YOLO.

Debido al reducido número de imágenes del *dataset*, se acudió a técnicas de aumentación de datos o *data augmentation*, mediante la librería Albumentations. Se aplicaron técnicas para generar imágenes nuevas tras una serie de cambios, detallados a continuación:

- *Horizontal flip*: aplicado a las imágenes con una probabilidad de 1.
- *Random brightness contrast:* aplicado a las imágenes con una probabilidad de 0.8.
- *Shift Scale Rotate*: aplicado a las imágenes con una probabilidad de 0.8, usando variación en x y y en un intervalo de 0 a 0.1, además de un escalamiento límite entre 0 y 0.1. El límite de rotación definido fue entre 0 a 10 grados.

Se especificó el formato de etiquetado correspondiente a YOLO junto a un parámetro para limitar el mínimo de visibilidad de las imágenes tras los cambios aplicados a un 50%. Además, se agregaron imágenes sin etiquetado conocidas como *background* al entrenamiento, correspondientes a imágenes de otros animales, personas u objetos similares en forma o color a un pollo; esto permite que el algoritmo aprenda de mejor manera a diferenciar los pollos dentro de la imagen al entrenarse con YOLOv5.

Tras el procedimiento de *data augmentation*, se obtuvieron 1308 imágenes en total entre enfermos y sanos, tanto de manera individual como en grupo dentro de galpones; estas imágenes se dividieron 80% para entrenamiento, 10% para validación y 10% para prueba. Estas imágenes fueron utilizadas para el entrenamiento con el modelo de YOLOv5, mientras que para el modelo CNN de clasificación de pollos sanos y enfermos, se requirieron las imágenes individuales de los pollos para el entrenamiento, validación y prueba. Luego de cortar las diferentes imágenes de los pollos, se obtuvieron un total de 2239 imágenes individuales de pollos sanos y 522 imágenes individuales de pollos enfermos, manteniendo la misma proporción mencionada.

El código utilizado para el proceso de *data augmentation* se encuentra en Apéndices.

2.6.2 Algoritmo de detección de pollos de engorde

Para detectar los pollos de engorde, se escogió el algoritmo de YOLOv5 versión s, el cual es un modelo pre-entrenado con un *dataset* llamado COCO de 80 clases diferentes en la detección de objetos con alto desempeño [25]. Al tener un *dataset* relativamente pequeño, se aplicó *transfer learning*, partiendo de un modelo pre-entrenado con el fin de obtener mejores resultados para la detección de los pollos.

Por ende, se inicializaron los pesos del modelo con el modelo pre-entrenado, además se entrenó un *batch* de 16 por lo que en cada iteración se escogieron 16 imágenes para entrenar el modelo y se lo entrenó por 64 *epochs*. La arquitectura del modelo YOLOv5, se describe en la Figura 2.14.

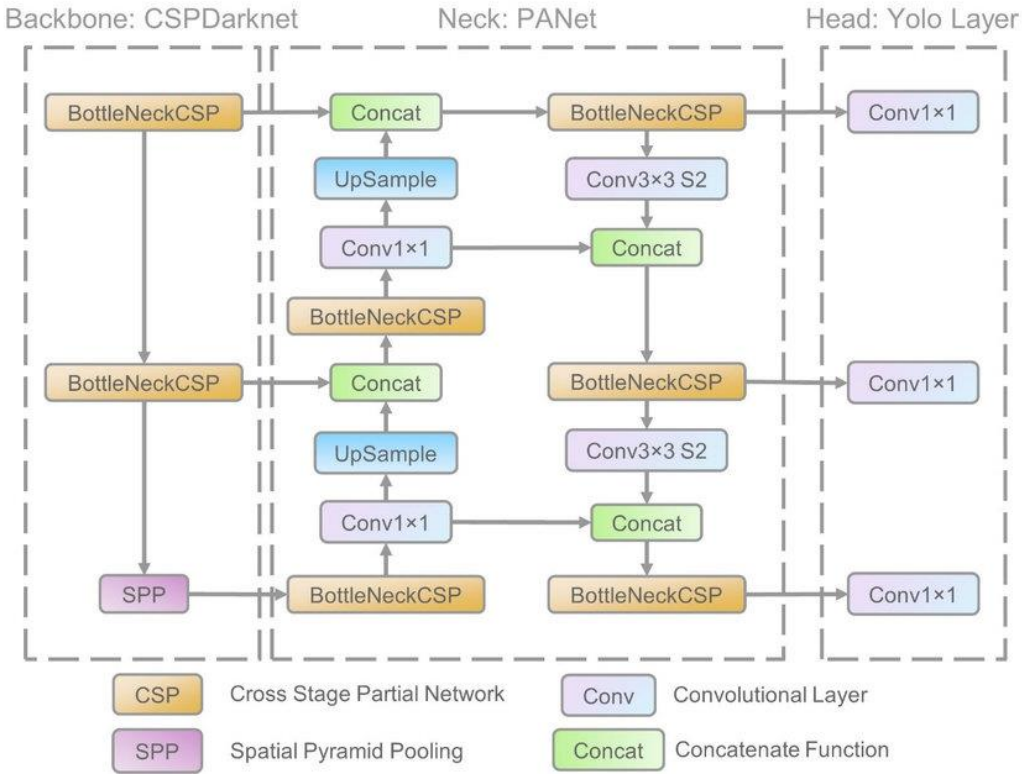


Figura 2.14 Arquitectura de YOLOv5 [26].

2.6.3 Algoritmo de clasificación de pollos enfermos y sanos

Para la clasificación se utilizó el algoritmo de CNN, el cual recibirá las imágenes de los pollos detectadas por el algoritmo de detección que usó YOLOv5, para así clasificarlos entre enfermos y sanos.

El algoritmo inicialmente cambia el tamaño de las imágenes a un tamaño de 32 x 32, además de que aplica un aumento de los datos iniciales al voltear horizontalmente las imágenes de forma aleatoria y normalizar las imágenes.

El modelo utilizado consta de tres capas convolucionales, además entre ellas se añaden capas *max pooling*, la función de activación ReLu y *dropout.* Obteniendo como resultado la configuración de la Figura 2.15.

```
Net(
  (conv1): Conv2d(3, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (conv2): Conv2d(16, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (conv3): Conv2d(32, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
  (pool): MaxPool2d(kernel_size=2, stride=2, padding=0, dilation=1, ceil_mode=False)
  (fc1): Linear(in_features=1024, out_features=500, bias=True)
  (fc2): Linear(in_features=500, out_features=2, bias=True)
  (dropout): Dropout(p=0.25, inplace=False)
)
```

Figura 2.15 Estructura del modelo CNN.

Para determinar la función de pérdida del algoritmo, se usó el criterio de *cross-entropy*, el cual permite calcular un puntaje que penaliza la probabilidad de la clase predicha, en base a que tan lejos está la predicción del valor esperado [27]. Además, se definió como optimizador al método de descenso de gradiente estocástico con una tasa de aprendizaje de 0.01, usada para modificar los atributos de la red permitiendo reducir la pérdida y mejorar la precisión del modelo durante el entrenamiento .

2.6.4 Despliegue de los algoritmos de detección y clasificación de enfermedades

Con los modelos entrenados para la detección y clasificación en archivos .pt, se hace la prueba de los dos modelos funcionando en conjunto, teniendo como entrada videos.

En la programación se obtiene un *frame* cada 5 minutos, la cual es guardada y las funciones de la clase que aloja el modelo de detección, leen la imagen y la estandarizan a un formato de 420 x 420, para luego enviar la imagen al modelo de YOLOv5 que retorna la etiqueta o etiquetas de los pollos detectados junto a las coordenadas de los pollos detectados en la imagen, con las cuales se corta las imágenes de los pollos detectados y las guarda. Una función de predicción lee las imágenes del directorio donde se encuentran las imágenes detectadas y retorna la etiqueta *sick* si el pollo es etiquetado como enfermo y *healthy* si es etiquetado como sano. Luego las imágenes con etiqueta *sick* son guardados en la base de datos de Firebase

para luego ser mostradas en la aplicación como un reporte. Por último, las imágenes son eliminadas de los directorios, ya que ya se encuentran respaldadas en la base de datos las imágenes de los pollos enfermos que se usarán para el registro.

El código del algoritmo de clasificación, detección y el despliegue de ambos se detalla en Apéndices.

2.7 Diseño del sistema IoT

Para el control y monitoreo de las condiciones ambientales, se diseñó un sistema IoT que permita visualizar en tiempo real estos parámetros, y controlarlas mediante las luces y el sistema mecánico para las cortinas.

2.7.1 Selección de sensores

La Tabla 2.5 y la Tabla 2.7 describen las características de los sensores a utilizar para medir los parámetros ambientales de temperatura y humedad, los cuales se conectarán a la computadora de control del sistema con las características de la Tabla 2.8, usando módulos WiFi (Tabla 2.6). Para el control de los motores, se utiliza un controlador de motor L298N, como el descrito en la Tabla 2.9. Además, para el control de luces del galpón, se usa relés, cuyas características se describen en la Tabla 2.10. A continuación, se detallan las especificaciones de cada dispositivo seleccionado para el sistema:

Tabla 2.5 Características del sensor de temperatura [29].

Sensor de temperatura	
	
Modelo	AM2320
Voltaje de alimentación	3.3 – 5 V DC
Rango de medición	0 – 99.9% RH (humedad) -40 – 80 ℃ (temperatura)
Precisión de medición	+-3% RH (humedad) +-0.5 ℃ (temperatura)
Resolución	0.1 %RH (humedad), 0.1℃ (temperatura)
Tamaño	3.7 x 2.1 cm

Tabla 2.6 Características del módulo WiFi [30].

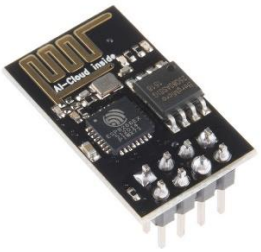
Módulo WiFi	
	
Modelo	ESP8266 WiFi Module
Características	• Gestión completa del WiFi con amplificador incluido. • 802.11 protocol y WiFi Direct (P2P) Soft-AP. • Regulador y unidad de alimentación incluidos. • Consumo en reposo < 10 mW • Soporta antena externa para mayor alcance. • Soporta el bus SPI. • 1 entrada analógica.

Tabla 2.7 Características del sensor de humedad [31].

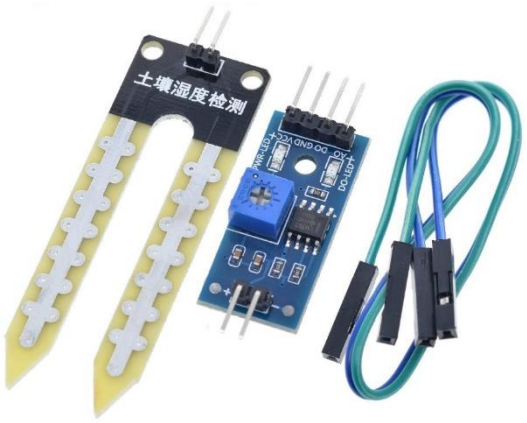
Sensor de humedad	
	
Modelo	LM393
Voltaje de alimentación	3.3 – 5 V DC
Profundidad de detección	37 mm
Tamaño	9.81 x 6.35 mm

Tabla 2.8 Características de la Raspberry Pi 4 [32].

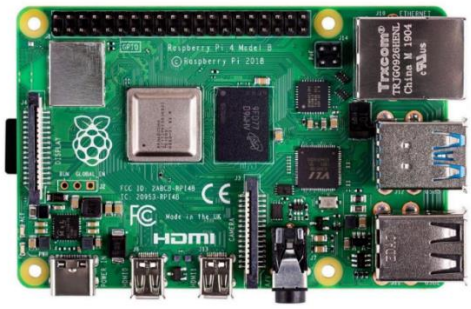
Computadora para control	
	
Modelo	Raspberry Pi 4
Procesador	Broadcom, BCM2711
Frecuencia	1.5 GHz
RAM	8 GB
Voltaje de alimentación	5 V
Temperatura de trabajo	0 - 50°C

Tabla 2.9 Controlador del motor [33].

Controlador del motor	
	
Modelo	L298N Motor Driver
Fuente de Alimentación	50 V
Voltaje de conducción	7- 35 V
Voltaje Lógico	5 V
Potencia Máxima	25 V

Tabla 2.10 Relé [34].

Relé	
	
Modelo	Módulo Relé 1 Canal
Voltaje de Operación	5V
Corriente máx	10A (NO), 5A (NC)
Tiempo de acción	10ms/ 5ms
Capacidad máx	10A/250VAC, 10A/30VDC

CAPÍTULO 3

3. RESULTADOS Y ANÁLISIS

Este capítulo presenta los resultados obtenidos en el diseño de los diferentes subsistemas que componen el sistema IoT y el de detección de enfermedades. Se muestran los resultados de prueba de algoritmos, circuitos de control y funcionamiento de la aplicación diseñada. Además, se muestra el análisis de costos de los materiales encontrados a nivel local como importados, junto con el costo total considerando mano de obra.

3.1 Aplicación móvil

Se utilizó Android Studio para el diseño y la lógica de programación de la aplicación móvil, se la conectó a la base de datos Firebase con el fin de complementar el Sistema IoT encargado del monitoreo y control de los parámetros ambientales.

La Figura 3.1 muestra la pantalla de inicio de sesión, estas credenciales serán facilitadas a los cuidadores, gerentes y supervisores del galpón; esto debido a que solo ellos tienen autorización para controlar los actuadores y revisar la información.

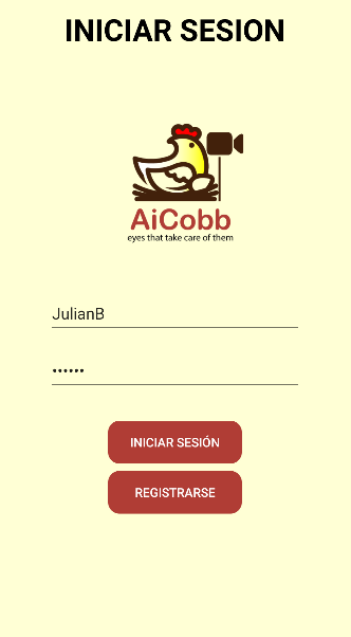


Figura 3.1 Pantalla de Inicio de Sesión.

El menú principal de la Figura 3.2 consta de 3 pantallas: Control y Monitoreo, Detección de Enfermedades y Registro Diario.

Figura 3.2 Pantalla de Menú Principal.

En la pantalla de Control y Monitoreo se puede escoger el modo de accionamiento del motor que sube/baja las cortinas; este puede ser manual o automático.

En caso de estar en modo manual se debe acceder a la pantalla de Controlar actuadores para subir o bajar las cortinas; de lo contrario, se debe ingresar la información de los pollos de edad, peso y cantidad, para con base en eso estimar la temperatura y humedad adecuada, y con ello accionar automáticamente el motor (Figura 3.3).

Figura 3.3 Pantalla de Control y Monitoreo.

La pantalla de Visualizar Sensores (Figura 3.4) permite monitorear en tiempo real el valor promedio de los 4 sensores de temperatura y humedad colocados en el galpón.

Figura 3.4 Pantalla de Visualizar Sensores.

La pantalla de Control de Actuadores permite activar o desactivar los controladores como es el motor o las luces. Así mismo, también permite asignar una hora a las luces para que se enciendan automáticamente, como se indica en la Figura 3.5.

Figura 3.5 Pantalla de Control de Actuadores (A), hora de encendido (B).

La pantalla de Detección de Enfermedades indica si existe un pollo enfermo y adicionalmente, muestra las fotos de los pollos enfermos que se hayan detectado dentro del galpón. A su vez, envía una alerta al usuario cada hora en caso de que se hayan detectado pollos enfermos (Figura 3.6).

Figura 3.6 Pantalla de Detección de Enfermedades.

La pantalla de Registro Diario, mostrada en la Figura 3.7, permite llevar el control de diario del galpón; la información queda almacenada en la base de datos para un posterior análisis de rentabilidad.

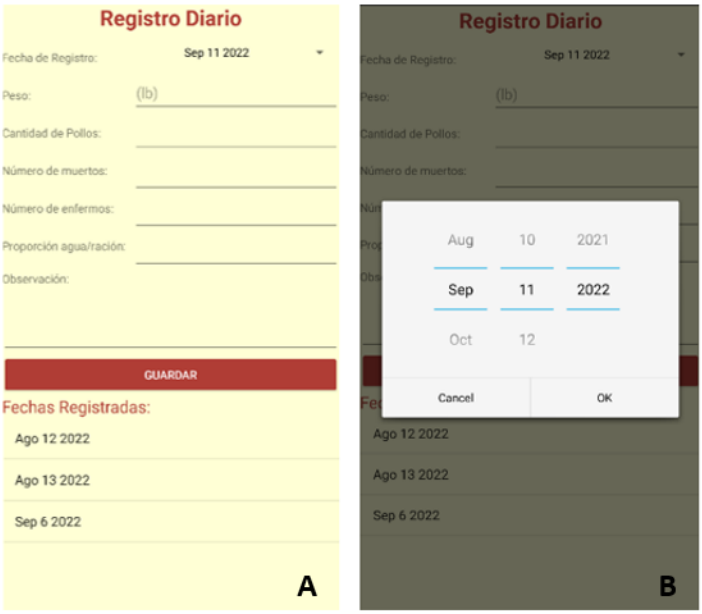


Figura 3.7 Pantalla de Registro Diario (A), acceso a fechas registradas (B).

3.2 Sistema mecánico

Mediante las medidas dadas por el proveedor de los componentes seleccionados, se modelaron los componentes del sistema para su posterior ensamble. En la Figura 3.8, se muestra el motor DC seleccionado (A) junto a la polea de 19 mm de diámetro (B).

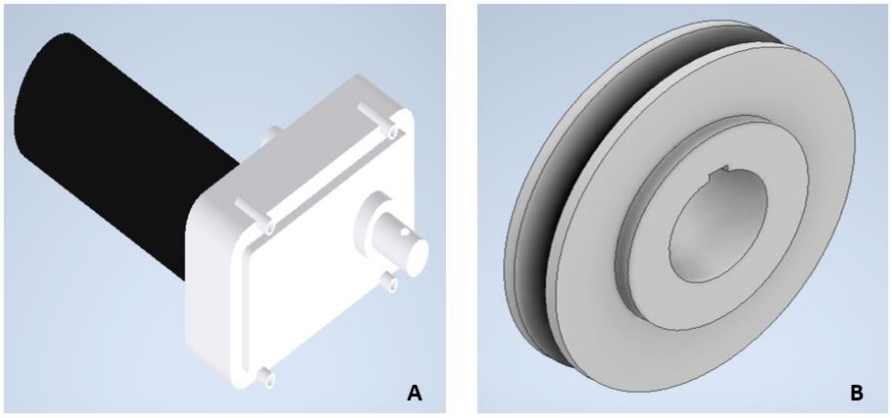


Figura 3.8 Motor DC (A), polea de 19 mm de diámetro (B).

Como se observa en la Figura 3.9, el motor junto a las poleas fue ubicado en un soporte diseñado que se encontrará empernado al concreto. Los cables de alimentación del motor se conectan luego al controlador del motor y al controlador del sistema, ambos dentro de la caja de control, cuya ubicación se muestra en la Figura 3.21. La conexión del motor al controlador del motor y al controlador del sistema, se detalla en el diagrama de la Figura 3.10.

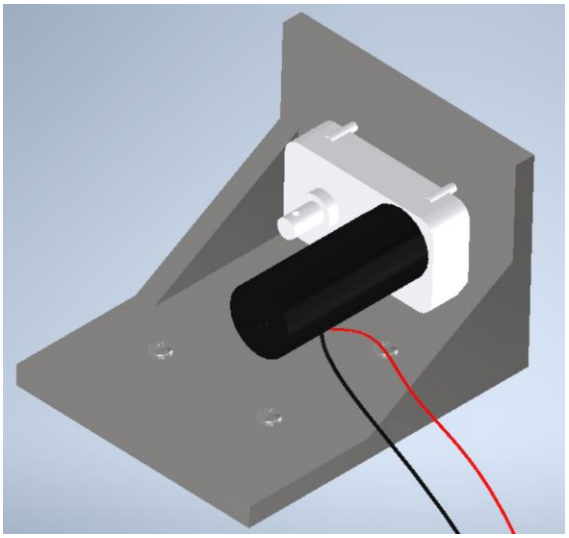


Figura 3.9 Montaje del motor DC con las poleas.

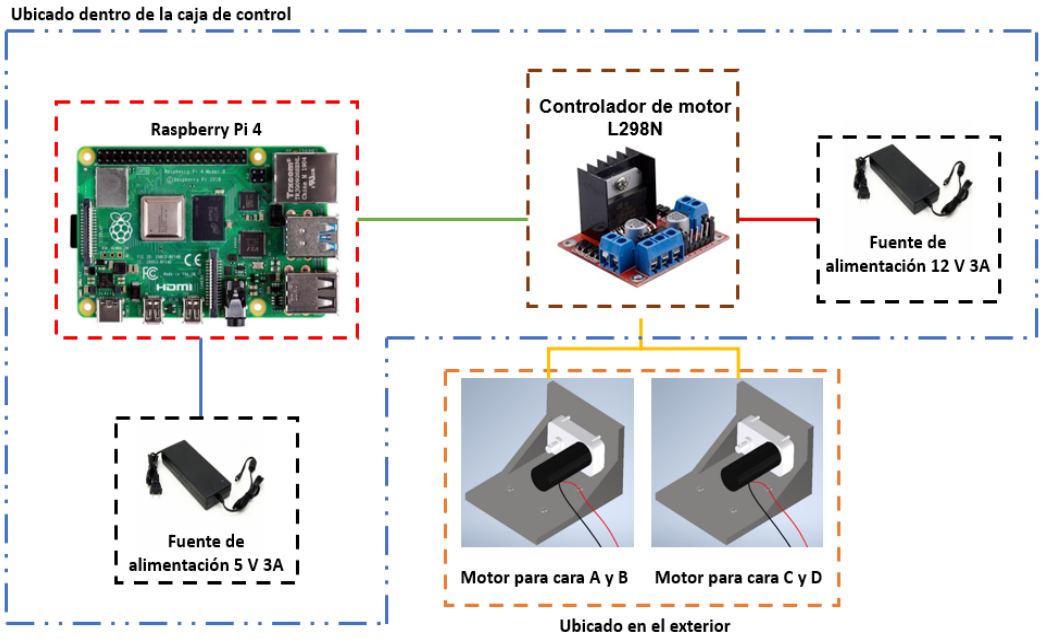


Figura 3.10 Diagrama esquemático de conexiones de los motores DC.

Además, se modeló el carrito junto con el riel y la varilla de acero con las medidas comerciales (Figura 3.11). En la Figura 3.12 y la Figura 3.13, se muestra el carrito dentro del riel.

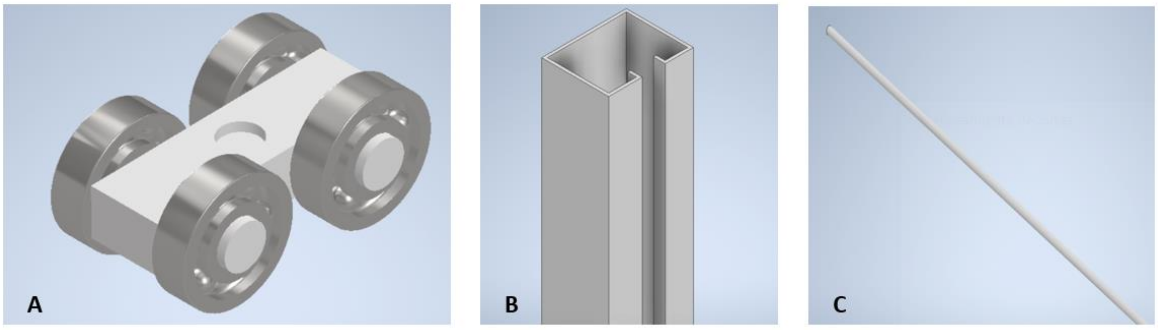


Figura 3.11 Carrito (A), riel (B), varilla (C).

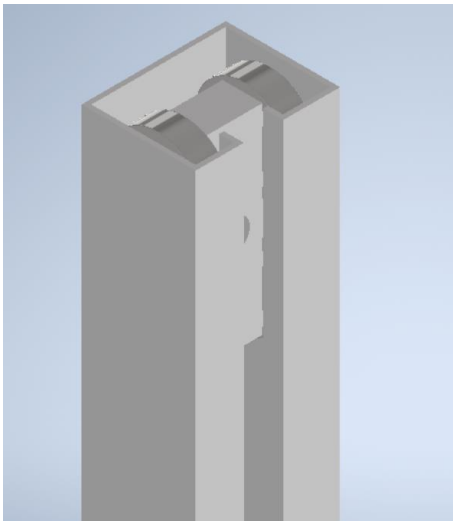


Figura 3.12 Sistema riel-carrito.

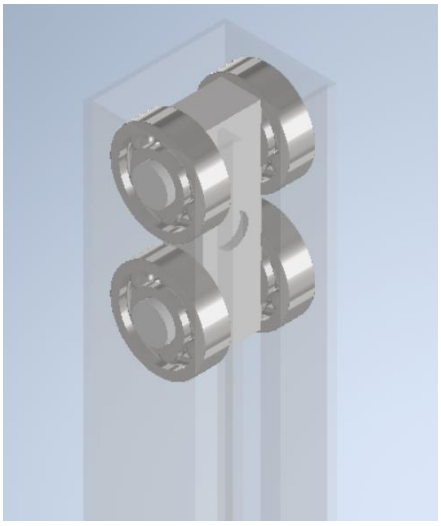


Figura 3.13 Sistema riel-carrito vista interna.

Para visualizar el diseño mecánico en el sitio a ser instalado, se utilizó el programa SketchUp. Tras modelar el galpón a escala con las medidas, se agregó el sistema mecánico junto con el sistema de poleas y los motores, como se muestra en la Figura 3.14 y la Figura 3.15. Además, en la Figura 3.16 se muestra de cerca el sistema poleas junto a las cuerdas usadas.

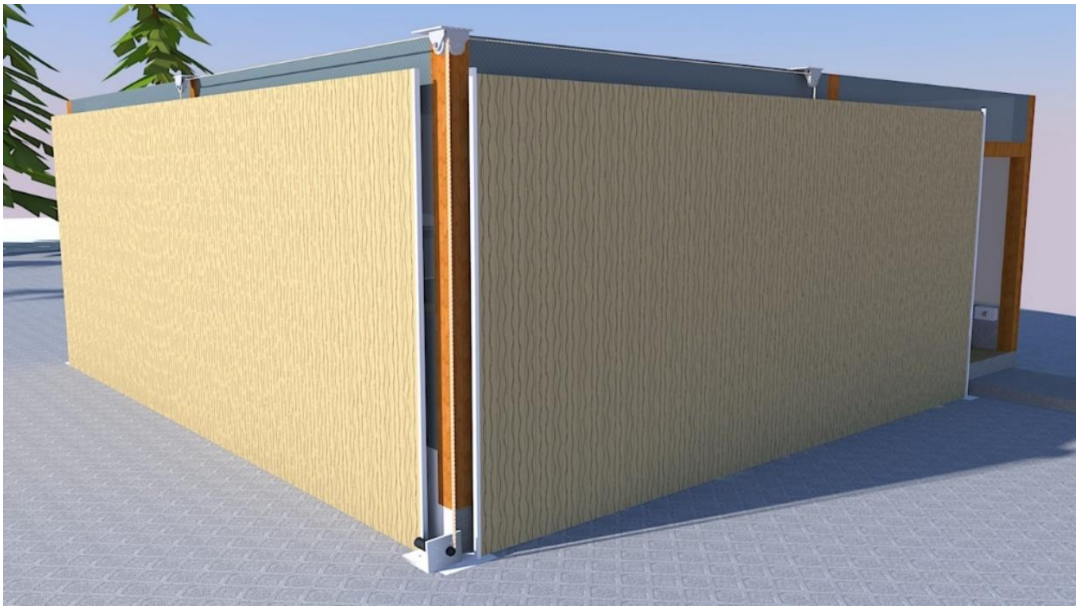


Figura 3.14 Montaje del sistema mecánico diseñado en el área de trabajo con las cortinas cerradas.

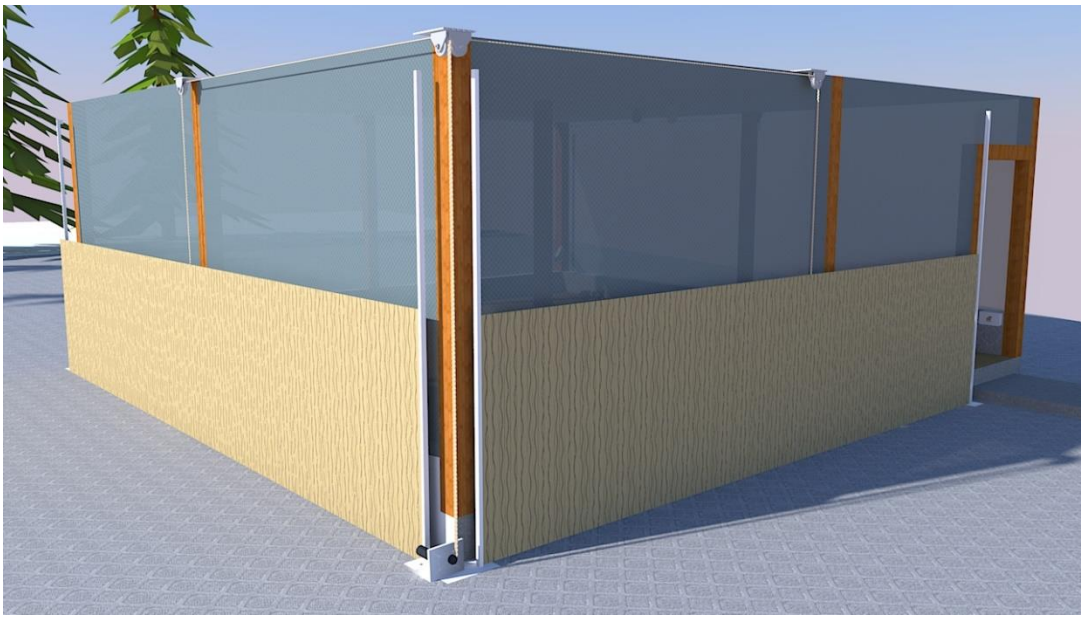


Figura 3.15 Montaje del sistema mecánico diseñado en el área de trabajo con las cortinas semiabiertas.

Figura 3.16 Sistema de poleas y cuerdas del sistema mecánico.

El material de la cuerda corresponde a fibra alquitranada, como se muestra en la Figura 3.17

Figura 3.17 Cuerda de fibra alquitranada.

3.3 Sistema IoT

Para el sistema IoT los actuadores como las luces y el motor DC del sistema mecánico se encuentran conectados a la Raspberry Pi, así mismo se tiene

conectaron los sensores. La Raspberry Pi está conectada a Firebase (Figura 3.18), donde aloja los datos de los sensores y actúa como intermediario entre la Raspberry Pi y la aplicación desarrollada.

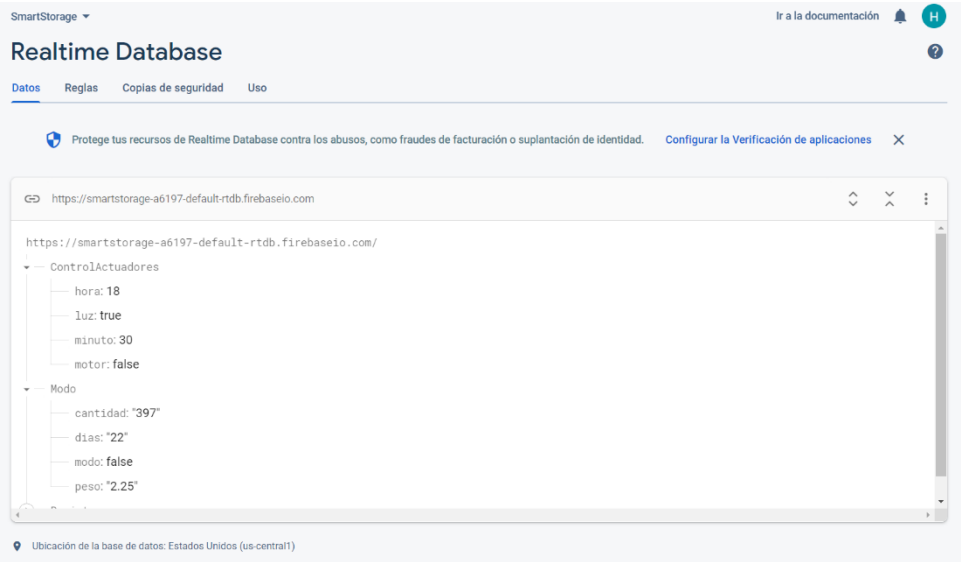


Figura 3.18 Base de datos.

Con el fin de resguardar los componentes electrónicos, se modeló una caja de control (Figura 3.19).

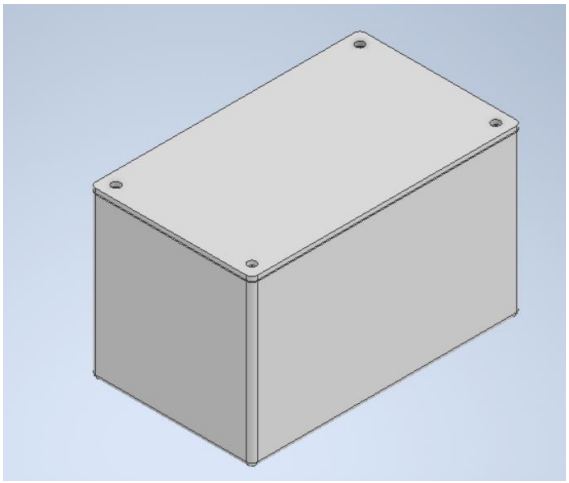


Figura 3.19 Modelado de caja de control.

En la Figura 3.20, se muestra el interior del galpón, donde también se modeló la ubicación de los componentes del sistema IoT. Se puede observar las 2

cámaras, los 4 sensores de temperatura y humedad, además de la caja de control donde se aloja la Raspberry Pi.

Figura 3.20 Sistema IoT ubicado en el área de trabajo.

En la Figura 3.21, se muestra la ubicación de la caja de control.

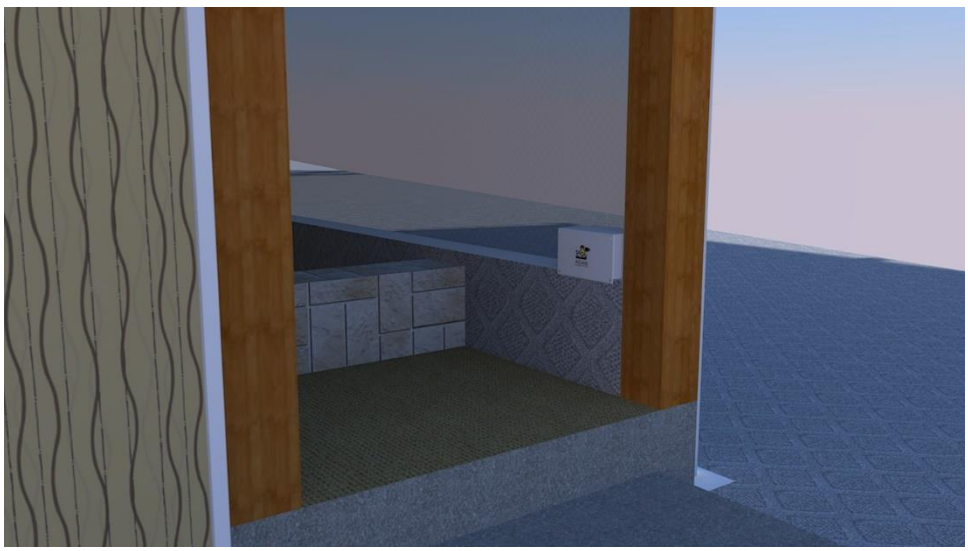


Figura 3.21 Ubicación de caja de control.

En la Figura 3.22, se muestra la ubicación de los sensores de temperatura y humedad.

Figura 3.22 Ubicación de sensores.

Para el diseño del sistema eléctrico se usó la Raspberry Pi 4 como controlador principal, y se conectó a los distintos actuadores y sensores tal como se visualiza en el diagrama de la Figura 3.23. En el diagrama se tiene los 4 sensores como entradas mientras que los motores DC controlados por un controlador para la activación y control de sentido de giro, además de los focos simulados como diodos led, actúan como salidas.

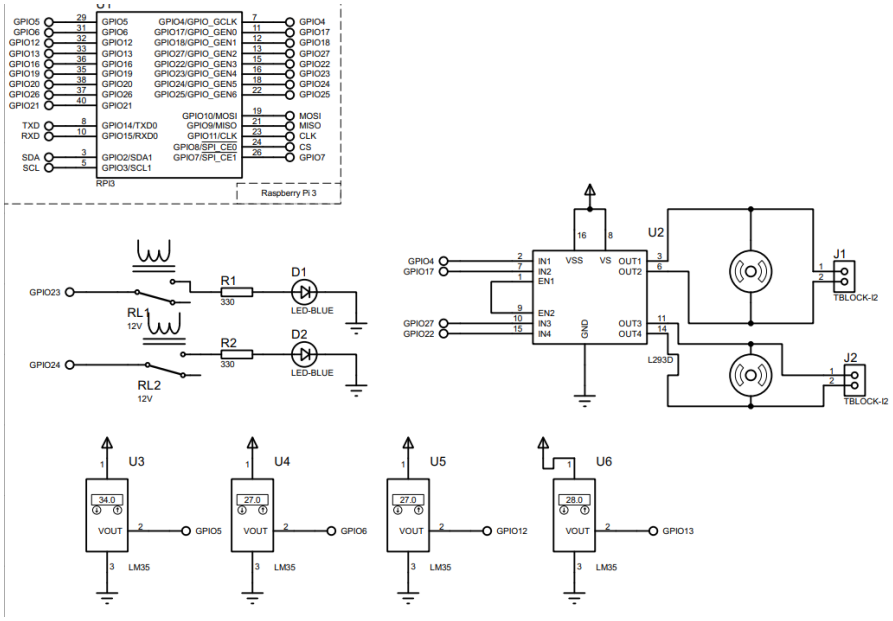


Figura 3.23 Circuito electrónico.

3.4 Sistema de detección de enfermedades

Para la captura de las imágenes en tiempo real, se tiene el esquema de conexiones de la Figura 3.24. En este diagrama, el DVR se conecta al router mediante un módulo *wireless*, permitiendo estar en la misma red que la Raspberry Pi posibilitando el envío de los videos capturados por las cámaras del galpón.

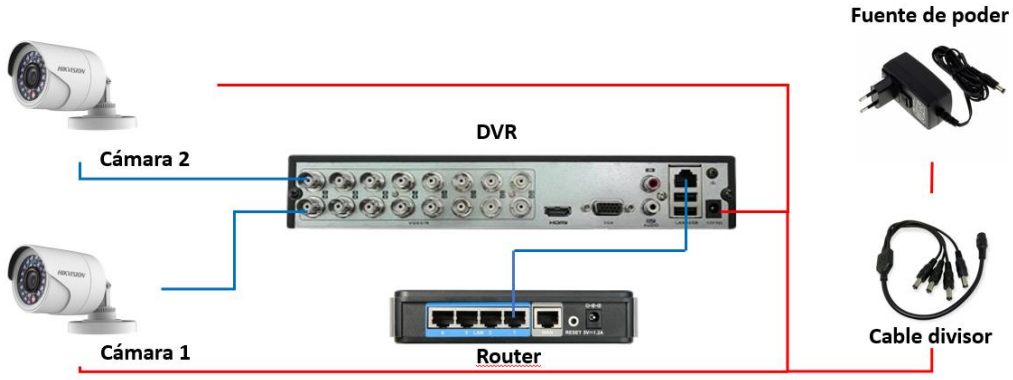


Figura 3.24 Diagrama de conexiones de las cámaras.

3.4.1 Algoritmo de detección de pollos de engorde

El análisis del desempeño del modelo de YOLOv5 (Figura 3.25) dio como resultado una precisión durante el entrenamiento del 92.5 % (mAP), mientras que el *recall* (R), indicador de que tan bien detecta los verdaderos positivos del *dataset* tiene un resultado igual de alto, equivalente al 88%. Por último, el *precision* (P) del modelo también es alto (90.9%), por lo que el desempeño del modelo es muy bueno para el desarrollo de la aplicación.

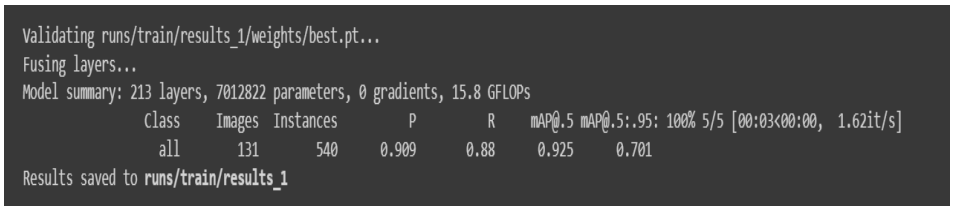


Figura 3.25 Resultados en el entrenamiento del modelo YOLOv5.

Usando para la prueba del modelo en su mayoría imágenes reales del cliente, se tiene como resultado en la Figura 3.26, que los pollos son detectados correctamente por el modelo entrenado.

Figura 3.26 Resultados en la prueba del modelo YOLOv5.

3.4.2 Algoritmo de clasificación de pollos enfermos y sanos

Para la clasificación se utilizó un modelo CNN que dio como resultado un decrecimiento en el error tanto durante entrenamiento como validación, cada vez acercándose más entre sí estos valores a medida que el algoritmo aprende a clasificar mejor en cada iteración (Figura 3.27).

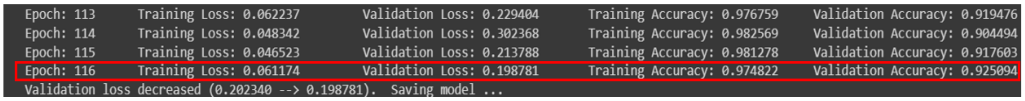


Figura 3.27 Resultados en el entrenamiento y validación del modelo CNN.

Al poner a prueba el modelo con un conjunto de imágenes nuevas, se tiene como resultado que la precisión del modelo corresponde al 95.72%, como se observa en la Figura 3.28.

Figura 3.28. Resultados en la prueba del modelo CNN.

Analizando el desempeño del modelo mediante la matriz de confusión de la Figura 3.29, es notable que el modelo predice correctamente para la mayoría de las imágenes de prueba tanto en la clase de pollos enfermos como la de sanos. Sin embargo, debido al desbalance entre las dos clases con respecto a la cantidad de imágenes, hay una mayor precisión para clasificar pollos sanos, como se muestra en la Figura 3.30.

Figura 3.29 Matriz de confusión resultante.

```
Accuracy for class: healthy is 96.6 %
Accuracy for class: sick  is 84.2 %
```

Figura 3.30 Precisión por clase.

3.5 Análisis de costos

Para el análisis de costos, se describen los costos de los componentes a adquirir a nivel nacional e internacional.

3.5.1 Costos locales

A continuación, en la Tabla 3.1, se encuentran el precio de los elementos de la solución que se encuentran en el mercado local.

Tabla 3.1 Tabla de costos en el mercado local.

Costo de elementos en el mercado local			
Sistema Mecánico			
Descripción del componente	**Cantidad**	**Costo unitario**	**Costo total**
Soporte de montaje	1	$5.95	$5.95
IFI502 Tornillos de cabeza hexagonal M6x1x20	4	$1.60	$6.40
EN ISO 7045 Tornillo M10x25	4	$2.40	$9.60
Total Sistema Mecánico			$21.95
Sistema Electrónico			
AM2320 Sensor de Temperatura	4	$8.00	$32.00
ESP8266 Wifi Module	4	$5.00	$20.00
LM393 Sensor de Humedad	4	$2.01	$8.04
Cámara Bala TURBO HD THC-B110-P	2	$13.00	$26.00
Raspberry Pi 4B, 8GB, bulk	1	$93.75	$93.75
Cables	~50		~$5.00
Cargador 5V 3A	1	$4.98	$4.98
Cargador 12V 3A	2	$4.98	$9.96
L298N Motor Driver	1	$2.86	$2.86
Relé 1 Channel	2	$0.50	$1.00
DVR	1	$43.00	$43.00
Cable splitter para 4 de alimentación	1	$2.00	$2.00
WiFi Dongle USB *wireless*	1	$6.53	$6.53
Cable BNC de 100 pies	2	$8.23	$16.46
Caja de protección	1	$10	$10
Total Sistema Electrónico			$281.58
Total			$303.53

3.5.2 Costos de artículos importados

A continuación, en la Tabla 3.2 se detalla los elementos usados para que no se encuentran en el mercado local, a estos costos se debe aumentar gastos de aduana, tarifa, transporte, etc. Por ende, al valor de $465.20 se tiene que aumentar un valor adicional, este valor podría llegar a costar

aproximadamente $150. Lo cual da como resultado un valor en costos importados de $615.20.

Tabla 3.2 Tabla de costos de artículos importados.

Costo de elementos importados			
Sistema Mecánico			
Descripción del componente	**Cantidad**	**Costo unitario**	**Costo total**
Carro 4 ruedas 164V	8	$15.00	$120.00
Riel 164V 6 metros largo	8	$21.00	$168.00
Motor DC 12V	2	$78.00	$156.00
Polea 19 mm	4	$5.30	$21.2
Total			$465.20
Total con estimación de gastos de importación			$615.20

3.5.3 Costos totales

Sumando los costos locales e importados se tiene un valor total de $918.73, precio a tomar en cuenta para los valores de venta del servicio. Por otro lado, se debe considerar el costo de la mano de obra y el costo de la tecnología e innovación que se está implementando, además de la ventaja competitiva que les da respecto a las otras microempresas.

Tomando en cuenta unas 200 horas de trabajo por todo el proyecto y un valor aproximado de $15 dólares por hora se tiene un valor de mano de obra de $3000. Por ende, el total sería igual a $3918.73 considerando costo de materiales, instalación y mano de obra, como se detalla en la Tabla 3.3.

Tabla 3.3 Costo total del proyecto.

Costo total del proyecto	
Costo de elementos en el mercado local	$303.53
Costo de elementos importados	$615.20
Costo de mano de obra	$3000
Total	$3918.73

El costo total del proyecto puede abaratarse mediante la manufactura desde cero de componentes importados, sin embargo, a pesar del costo de inversión inicial de un productor a este proyecto; este presenta muchos beneficios al reducirse la cantidad de capital a invertir en cuidadores, medicinas y revisiones veterinarias más seguidas.

CAPÍTULO 4

4. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- El conjunto de sensores utilizados para el sistema IoT mantiene un control en tiempo real de las condiciones ambientales, permitiendo detectar de forma oportuna un cambio brusco de temperatura y humedad que pueda afectar el crecimiento de los pollos de engorde.
- El sistema mecánico diseñado permite el control del cierre y apertura de las cortinas del galpón mediante la aplicación, de forma manual o automática con base en los factores ambientales ideales, que depende de la edad, cantidad y peso de los pollos del galpón. A diferencia del sistema de manivela actual del cliente, el sistema diseñado presenta una menor resistencia y control automático, lo cual permite alargar la vida del sistema al tener menor riesgo de trabas y un control más cómodo.
- La ventana de registro diario permite guardar parámetros importantes fácilmente y un acceso inmediato a registros pasados, además de un respaldo a la base de datos permitiendo usar estos en el futuro para proyección de ganancias y evaluar la rentabilidad del lote.
- El sistema de detección de enfermedades permite al usuario prevenir la pérdida parcial o hasta total del lote de pollos de engorde, mediante dos algoritmos que trabajan en conjunto para la obtención de un modelo con 95.72% de precisión. Esto le permite al usuario evitar grandes pérdidas además de gastos en medicinas y reducir la cantidad revisiones por expertos, debido a la alta precisión del algoritmo.
- El costo total del proyecto es de $3918.73 luego de considerar componentes en el mercado local, componentes importados y la mano de obra. Los costos de materiales pueden reducirse si se diseñan desde cero a nivel nacional los componentes por importar, sin embargo, el costo actual es rentable a pequeños y medianos productores. Esto se debe a la gran ventaja y ahorro que representa adquirir el sistema ya que permite reducir

inversión en varios cuidadores, costo en medicinas al tener una detección temprana de la enfermedad de Newcastle, etc.

- El proyecto realizado fue acorde a las necesidades de un cliente específico, sin embargo, esto se puede comercializar también a otros pequeños y medianos productores que se dediquen a la crianza de pollos de engorde, escalando el proyecto según las necesidades de este.
- Es importante también aspirar a diseñar sistemas similares con la crianza de otras carnes de alta demanda ya que la automatización en criaderos como por ejemplo del cerdo podría mejorar la producción de materia prima del país y ayudar al mismo tiempo a pequeños y medianos productos a acceder a un sistema asequible y competir con las grandes empresas.

4.2 Recomendaciones

- Luego de analizar las imágenes en las que el algoritmo tiene mayor dificultad para la detección y clasificación, se recomienda diversificar más el *dataset* con el que se va a entrenar el algoritmo, ya que, debido al reducido número de imágenes en entrenamiento y validación de pollos en sus primeros días de vida, el algoritmo tiende a confundirse al clasificar los pollos de esta edad.
- Con el fin de obtener una mayor precisión del algoritmo en detectar pollos enfermos, se sugiere aumentar la cantidad de imágenes de pollos enfermos además de etiquetar de mejor manera las imágenes para que el modelo aprenda lo más característico de la enfermedad mediante un buen etiquetado.
- Es recomendable también ahondar más en las diversas enfermedades ocasionadas en los pollos de engorde, ya que existen diversos métodos de inteligencia artificial que podrían facilitar la detección de estas, permitiendo crear un sistema capaz de detectar y prevenir más enfermedades de alto riesgo.
- Para alargar la vida útil del sistema mecánico, se recomienda lubricar periódicamente los rodamientos del carrito. Además, se recomienda usar

sensores de final de carrera para accionar el motor en base a estos y prescindir de un *delay* dentro de la programación del microcontrolador.

- Es recomendable para trabajos futuros usar los parámetros de los registros y mostrar mediante la aplicación inmediatamente las estadísticas de interés.

BIBLIOGRAFÍA

[1] Organización para la Cooperación y el Desarrollo Económicos (OCDE) y Organización de las Naciones Unidas para la Alimentación y la Agricultura (FAO), «OCDE-FAO Perspectivas Agrícolas 2021-2030,» 2021.

[2] INEC, «Boletín Técnico ESPAC,» 2014. [En línea]. Available: https://www.ecuadorencifras.gob.ec/documentos/web-inec/Estadisticas_agropecuarias/espac/espac-2019/Boletin%20Tecnico%20ESPAC_2019.pdf. [Último acceso: 22 Junio 2022].

[3] R. Connelly, C. Mores y A. H. Simonne, «What are the risks of contracting diseases associated with chickens?,» The Institute of Food and Agricultural Sciences, 2019.

[4] Cobb, «Pollo de engorde: Guía de manejo,» 2018.

[5] D. E. García, Interviewee, *Diana Espín: La avicultura alimenta al Ecuador.* [Entrevista]. 11 Diciembre 2020.

[6] Poultry World, «The importance of 7-day weight,» 23 Abril 2013. [En línea]. Available: https://www.poultryworld.net/health-nutrition/the-importance-of-7-day-weight/. [Último acceso: 17 Septiembre 2022].

[7] Cobb, «Cobb500 El pollo de engorde más eficiente del mundo.,» Cobb, [En línea]. Available: https://www.cobb-vantress.com/es_MX/products/cobb500/.

[8] Vistazo, «$3.700 millones generó el sector avicultor en el 2021,» 14 Marzo 2022.

[9] CONAVE, «Estadísticas del sector avícola,» DPC Diez Punto Comunicaciones, [En línea]. Available: https://conave.org/informacion-sector-avicola-publico/.

[10] S. R. Tapia, «Estudio de Mercado Avícola enfocado a la Comercialización de Pollo en Pie, año 2012-2014,» 2017.

[11] E. Telégrafo, «$ 1.272 millones genera la producción avícola al año,» 05 Julio 2019. [En línea]. Available: https://www.eltelegrafo.com.ec/noticias/economia/1/feria-produccion-dia-pollo-ecuador. [Último acceso: Junio 2022].

[12] J. L. Houriet, «GUÍA PRÁCTICA DE ENFERMEDADES MÁS COMUNES EN AVES DE CORRAL,» 2017. [En línea]. Available: https://www.produccion-animal.com.ar/produccion_aves/enfermedades_aves/90-enfermedades.pdf. [Último acceso: 07 2022].

[13] BAKU, «Smart Farming Poultry IOT Solution,» PT Baku Inovasi Lestari., 2022. [En línea]. Available: https://baku.global/en/smart-farming-poultry-iot-solution/. [Último acceso: Junio 2022].

[14] G. A. Choukidar y N. Dawande, «Smart Poultry Farm Automation and Monitoring,» 2017. [En línea]. Available: https://www.researchgate.net/publication/327808668_Smart_Poultry_Farm_Automation_and_Monitoring_System. [Último acceso: Junio 2022].

[15] W. Jintao, S. Mingxia, L. Longshen, X. Yi y O. Cedric, «Recognition and Classification of Broiler Droppings Based on Deep Convolutional Neural Network.,» 2019.

[16] Z. Xiaolin, B. Minna, G. Jilei, W. Siyu y Z. Tiemin, «Development of an early warning algorithm to detect sick broilers,» *Computers and Electronics in Agriculture,* vol. 144, pp. 102-113, 2018.

[17] L. Longshen, L. Bo, Z. Ruqian, Y. Wen, S. Mingxia y Y. Ji, «A Novel Method for Broiler Abnormal Sound Detection Using WMFCC and HMM,» *Sensors and Applications in Agricultural and Environmental Monitoring,* 2020.

[18] O. P. Akomolage y F. B. Mediros, «Image Detection and Classification of Newcastle and Avian Flu Diseases Infected Poultry Using Machine Learning Techniques.,» *Journal of Science and Logics in ICT Research,* vol. 6, 2021.

[19] ARENA, «Guías y Rieles superiores,» [En línea]. Available: https://www.herrajesarena.com/guias-y-rieles-superiores.html. [Último acceso: 12 Septiembre 2022].

[20] ARENA, «Carro 4 Ruedas - Puertas y Portones,» [En línea]. Available: https://www.herrajesarena.com/carro-4-ruedas.html. [Último acceso: 12 Septiembre 2022].

[21] Y. Kyosev, Braiding technology for textiles: principles, design and processes., 2015.

[22] The Engineering ToolBox, «Friction - Friction Coefficients and Calculator,» [En línea]. Available: https://www.engineeringtoolbox.com/friction-coefficients-d_778.html. [Último acceso: 27 Septiembre 2022].

[23] MCMASTER, «12V 24V 50w Low Rpm High Torque Dc Gear Motor MM201,» [En línea]. Available: https://www.mcmaster-electric.com/12V-24V-50w-Low-Rpm-High-Torque-Dc-Gear-Motor-MM201-pd6339510.html. [Último acceso: 12 Septiembre 2022].

[24] SistemSeguridad, «Cámara Bala TURBO HD THC-B110-P,» [En línea]. Available: https://www.sistemseguridad.com/producto/camara-tubo-720p-turbohd-policarbonato-inteligente-ir-20mts-exterior-ip66/. [Último acceso: 13 Septiembre 2022].

[25] L. Ariel, «The practical guide for Object Detection with YOLOv5 algorithm,» Marzo 2014. [En línea]. Available: https://towardsdatascience.com/the-practical-guide-for-object-detection-with-yolov5-algorithm-74c04aac4843. [Último acceso: 12 Septiembre 2022].

[26] R. Xu, H. Lin, K. Lu, L. Cao y Y. Liu, «A Forest Fire Detection System Based on Ensemble Learning,» *Forests,* 3 Julio 2021.

[27] K. E. Koech, «Cross-Entropy Loss Function,» 2 Octubre 2020. [En línea]. Available: https://towardsdatascience.com/cross-entropy-loss-function-f38c4ec8643e. [Último acceso: 12 Septiembre 2022].

[28] A. Gupta, «A Comprehensive Guide on Deep Learning Optimizers,» *Data Science Blogathon,* 2021.

[29] AOSONG, «AM2320 Datasheet(PDF) 1 Page - List of Unclassified Manufacturers,» [En línea]. Available: https://html.alldatasheet.com/html-pdf/1132617/ETC2/AM2320/110/1/AM2320.html. [Último acceso: 12 Septiembre 2022].

[30] P. Marian, «ESP8266 Datasheet,» [En línea]. Available: https://www.electroschematics.com/esp8266-datasheet/. [Último acceso: 9 Septiembre 2022].

[31] Texas Instruments, «ALLDATASHEET,» [En línea]. Available: https://www.alldatasheet.com/datasheet-pdf/pdf/1260433/TI/LM393.html. [Último acceso: 9 Septiembre 2022].

[32] ElectroStore, «KIT BÁSICO RASPBERRY PI 8 GB RAM,» [En línea]. Available: https://grupoelectrostore.com/shop/placas-para-programacion/raspberry/raspberry-pi-4-modelo-b-8gb-ram/. [Último acceso: 13 Septiembre 2022].

[33] STMicroelectronics, «L298N Datasheet(PDF) 2 Page - STMicroelectronics,» [En línea]. Available: https://html.alldatasheet.com/html-pdf/22440/STMICROELECTRONICS/L298N/3243/2/L298N.html. [Último acceso: 13 Septiembre 2022].

[34] AVElectronics, «Módulo Relé 1 Canal,» [En línea]. Available: https://avelectronics.cc/producto/modulo-rele-1-canal/. [Último acceso: 17 Septiembre 2022].

[35] AI-SPECIALS, «Pytorch tutorials,» 2 Julio 2020. [En línea]. Available: https://github.com/gaurav67890/Pytorch_Tutorials/blob/master/cnn-scratch-inference.ipynb. [Último acceso: 15 Septiembre 2022].

[36] N. Nielsen, «Computer Vision,» 11 Enero 2022. [En línea]. Available: https://github.com/niconielsen32/ComputerVision/blob/master/deployYoloModel.py. [Último acceso: 15 Septiembre 2022].

APÉNDICES

APÉNDICE A

Prototipado inicial de aplicación

Figura A. 1 Prototipado de inicio de sesión.

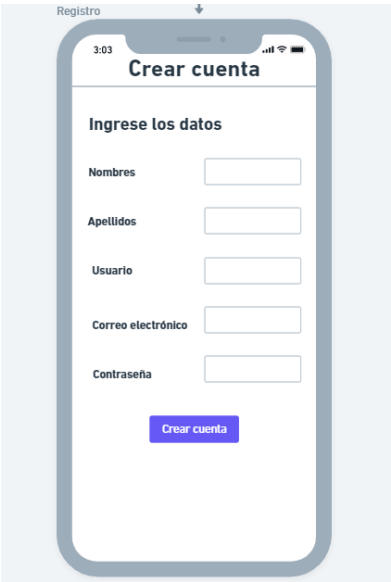


Figura A. 2 Prototipado de creación de cuenta del usuario.

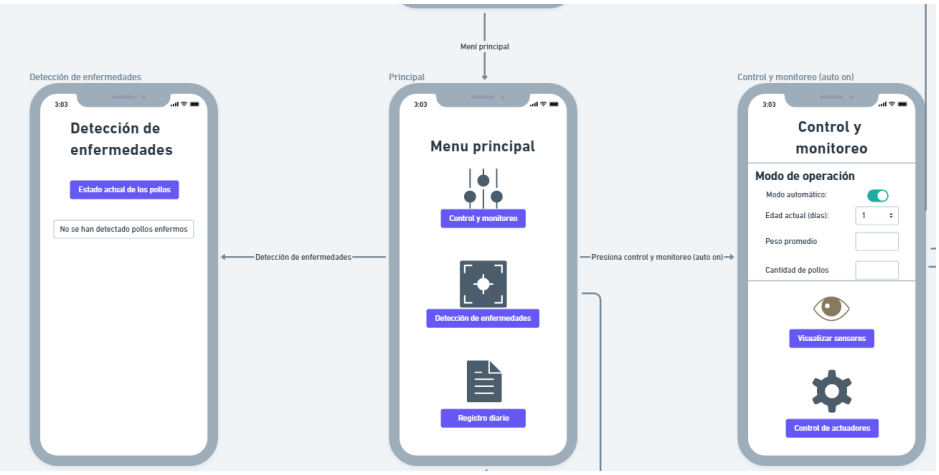


Figura A. 3 Prototipado del flujo del Menú Principal.

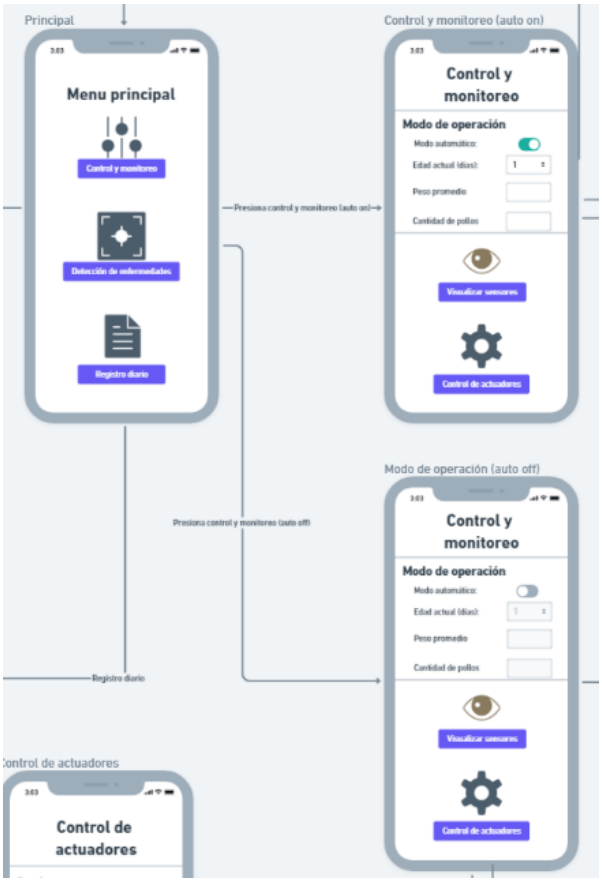


Figura A. 4 Prototipado del flujo del Menú Principal y Control y Monitoreo.

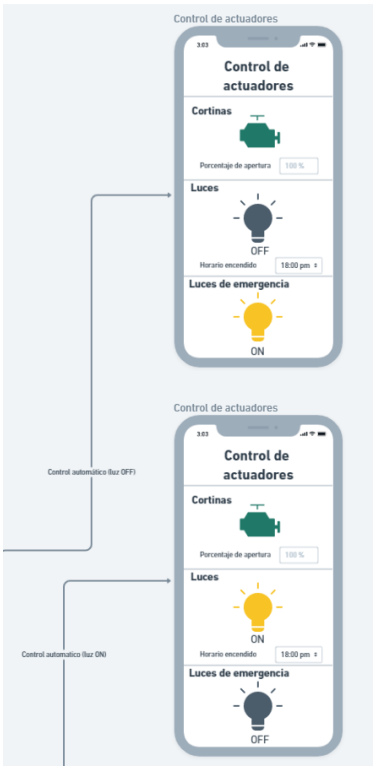


Figura A. 5 Prototipado del flujo de Control de Actuadores.

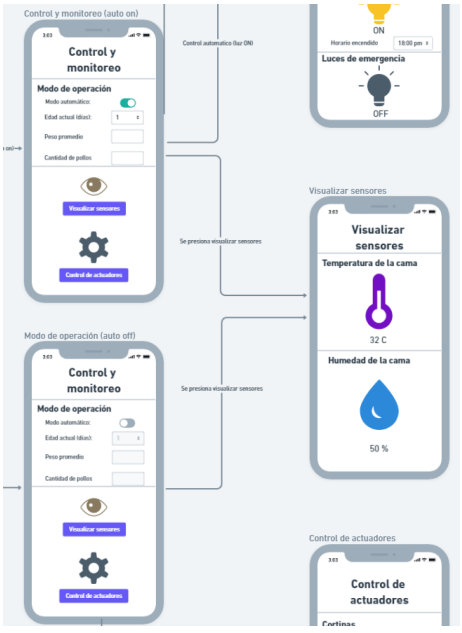


Figura A. 6 Prototipado del flujo de la pantalla para la visualización de sensores.

Figura A. 7 Prototipado del flujo del registro de parámetros importantes.

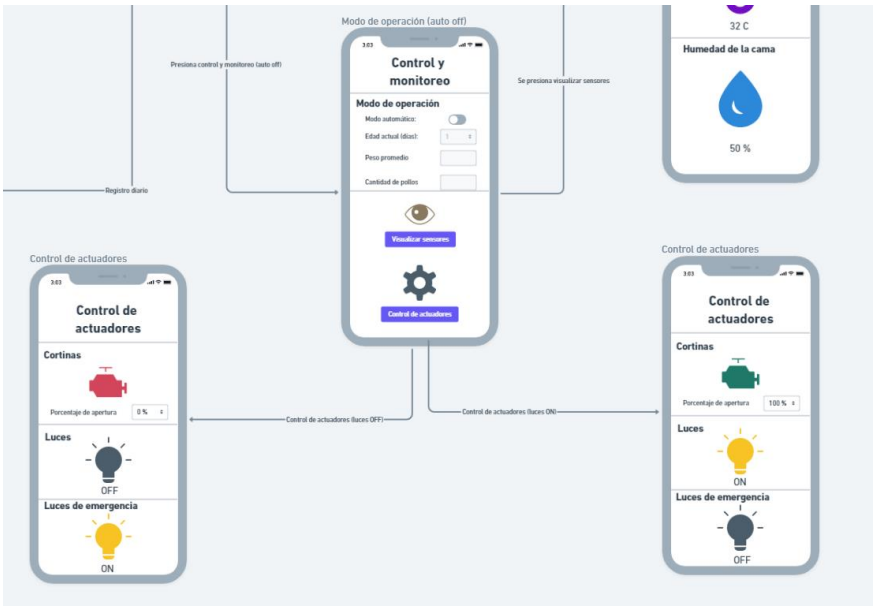


Figura A. 8 Prototipado del flujo de control de actuadores.

APÉNDICE B

Modelo de detección y clasificación de enfermedades

```
#Augmentation pipeline
transform = A.Compose([
        A.HorizontalFlip(p=1.0),
        A.RandomBrightnessContrast(p=0.8),
        A.ShiftScaleRotate(p=0.8, shift_limit_x=(0, 0.1), shift_limit_y=(0,0.1),
            scale_limit=(0, 0.1), rotate_limit=(0,10), interpolation=0, border_mode=0
            , value=(0, 0, 0), mask_value=None, rotate_method='largest_box')],
        bbox_params=A.BboxParams(format='yolo',min_visibility=0.5,label_fields
            =['class_labels']))
```

Figura B. 1 Funciones aplicadas para *data augmentation* y sus parámetros.

```
uploaded_video = st.file_uploader("Subir video", type=["mp4", "mov"])
frame_skip = 300 # display every 300 frames

if uploaded_video is not None: # run only when user uploads video
    vid = uploaded_video.name
    with open(vid, mode='wb') as f:
        f.write(uploaded_video.read()) # save video to disk

    st.title("Cámara")
    video_file = open(vid,'rb')
    video_bytes = video_file.read()
    st.video(video_bytes)
    vidcap = cv2.VideoCapture(vid) # load video from disk
    cur_frame = 0
    success = True
    path = 'C:\\Users\\sheyl\\Documentos\\frames\\'

    while success:
        success, frame = vidcap.read() # get next frame from video
        fps = vidcap.get(cv2.CAP_PROP_FPS)
        minutes = 0
        seconds = 5
        frame_id = int(fps*(minutes*60 + seconds))
        print('fps: {}'.format(fps))
        print('frame_id: {}'.format(frame_id))
        if cur_frame % frame_id == 0:
            print('frame: {}'.format(cur_frame))
            #Guarda la imagen de la frame en el tiempo indicado
            imageRGB = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
            pil_img = Image.fromarray(imageRGB)
            pil_img.save(path+dt.datetime.now().strftime('IMG-%Y-%m-%d-%H%M%S')+'.jpg')
            cur_frame = 0
        cur_frame += 1
    yolo_detection = ChickenDetection(model_name='C:\\Users\\sheyl\\Documentos\\'+'last.pt')
    yolo_detection()
```

Figura B. 2 Código utilizado para analizar *frames* a una frecuencia determinada.

```
def __call__(self):

    for file in glob.glob(path+'*.jpg'):
        fr = cv2.imread(file)
        fr = cv2.resize(fr, (420,420))
        results = self.score_frame(fr)
        fr_n= self.plot_boxes(results,fr)
        print('Detected: {}'.format(file))
    classes=["healthy","sick"]
    transform = transforms.Compose([
            transforms.Resize((32,32)),
            transforms.ToTensor(),
            transforms.Normalize((0.5,0.5,0.5),(0.5,0.5,0.5))
            ])
    def prediction(img_path,transformer):
        image = Image.open(img_path)
        image_tensor = transformer(image).float()
        image_tensor = image_tensor.unsqueeze_(0)
        if torch.cuda.is_available():
            image_tensor.cuda()
        input=Variable(image_tensor)
        output = model(input)
        index = output.data.numpy().argmax()
        pred = classes[index]
        return pred
    check= torch.load('C:\\Users\\sheyl\\Documentos\\'+'model_chicken_classification.pt')
    model = Net()
    model.load_state_dict(check)
    model.eval()
```

Figura B. 3 Parte del código para desplegar el modelo de clasificación [35].

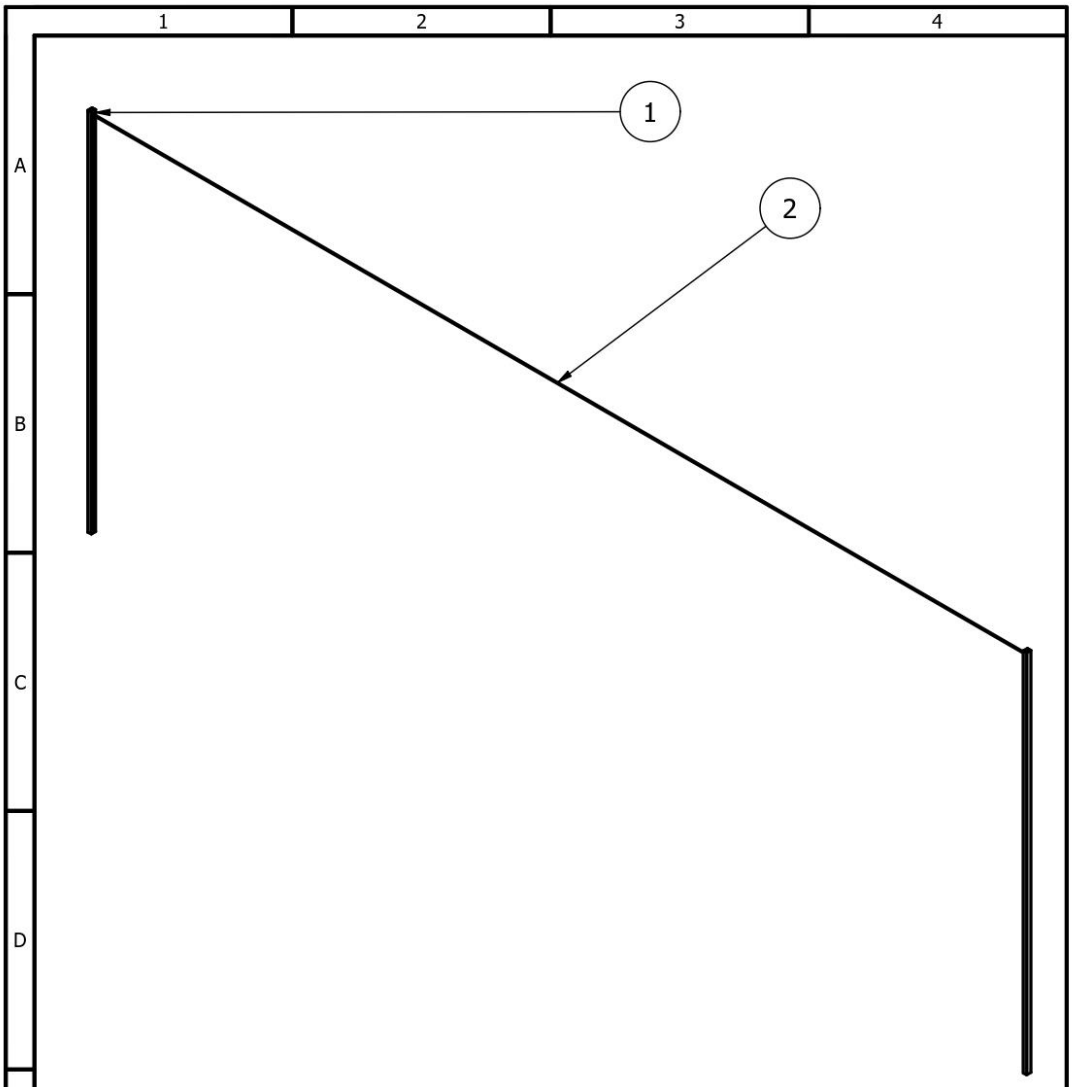
```
def plot_boxes(self, results, frame):
    labels, cord= results
    n = len(labels)
    x_shape, y_shape = frame.shape[1], frame.shape[0]
    path_write_im = 'C:\\Users\\sheyl\\Documentos\\frames_labeled\\'
    cord_l= []
    c=0
    for i in range(n):
        row = cord[i]
        if row[4] >= 0.3:
            c+=1
            x1, y1, x2, y2 = int(row[0]*x_shape), int(row[1]*y_shape), int(row[2]*x_shape), int
                (row[3]*y_shape)
            bgr = (0,255,0)
            print('x1: {}, y1: {} , x2: {} , y2: {}'.format(x1,y1,x2,y2))
            imageRGB = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
            pil_img = Image.fromarray(imageRGB[y1:y2,x1:x2]) # convert opencv frame (with type
                ()==numpy) into PIL Image
            pil_img.save(path_write_im+dt.datetime.now().strftime('IMG-%Y-%m-%d-%H%M%S')+str(c)+'
                .jpg')
        cord_l.append([row[0],row[1],row[2],row[3]])
    print(cord_l)
    return frame
```

Figura B. 4 Función utilizada para determinar las coordenadas del pollo detectado con YOLOv5 y guardar la imagen para su clasificación [36].

El código utilizado para los modelos de detección y clasificación, además del despliegue, se encuentran en el siguiente repositorio de github: https://github.com/sabenavi/ChickenDetection-Classification

APÉNDICE C

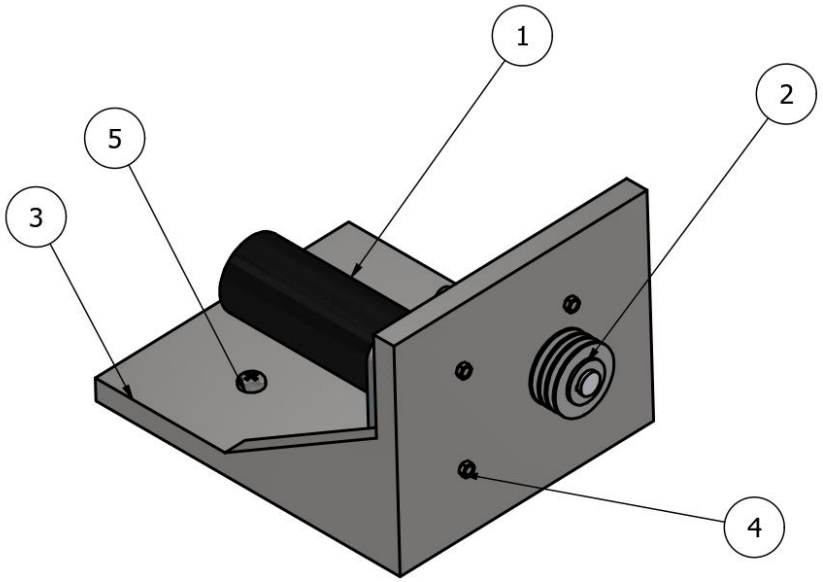
Planos mecánicos



ITEM	CANTIDAD	NOMBRE
1	2	Riel-carrito
2	1	Eje de 7 m

Tolerancias		N/A	Materiales: ACERO
	Fecha	Nombre	Denominación: SISTEMA RIEL-CARRITO CON EJE DE 7 M
Dib.	*	S.B & H.L	Escala: 0,03 : 1
Rev.	*	*	
Apro.	*	*	
			Número de dibujo: 1/12

Edición	Modificación	Fecha	Nombre



ITEM	CANTIDAD	NOMBRE
1	1	Motor
2	2	Polea de 19 mm
3	1	Soporte de montaje
4	4	IFI 502 - M6x1 x 20, IHHTSTTM
5	4	EN ISO 7045 - M10 x 25 - 4.8 - H

Edición	Modificación	Fecha	Nombre

Tolerancias: N/A

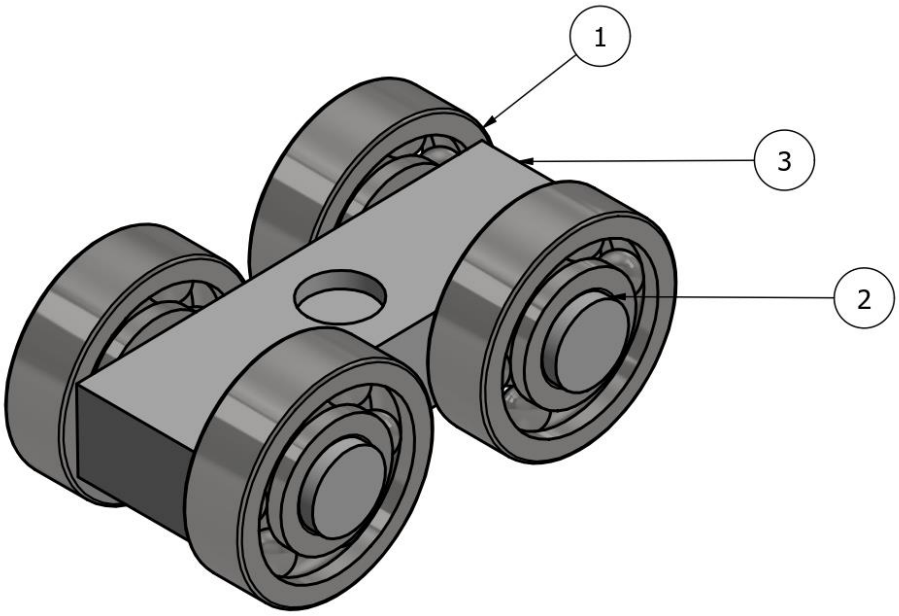
Materiales:

	Fecha	Nombre
Dib.	*	S.B & H.L
Rev.	*	*
Apro.	*	*

Denominación: MONTAJE DEL MOTOR

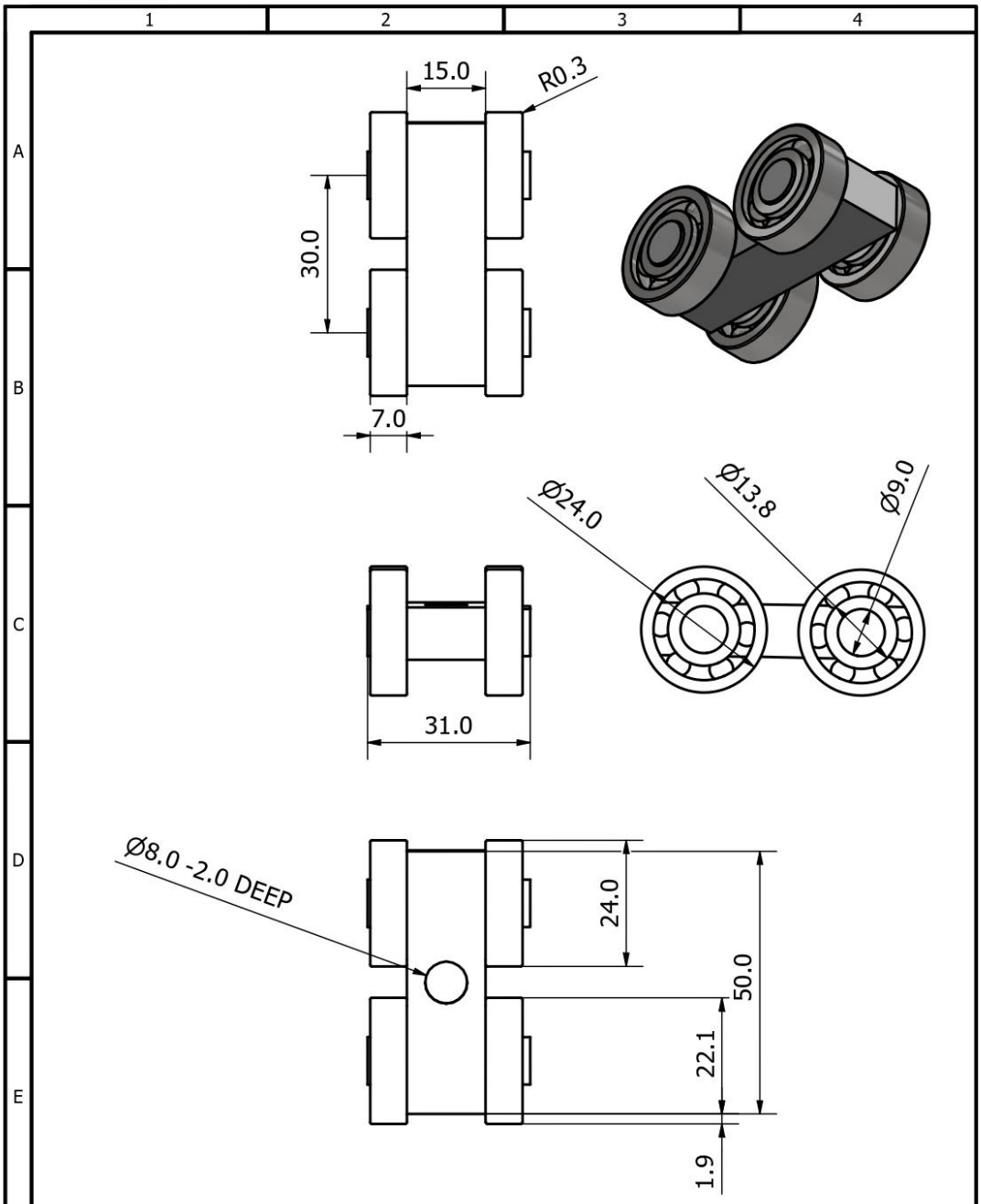
Escala: 1 : 4

Número de dibujo: 2/12

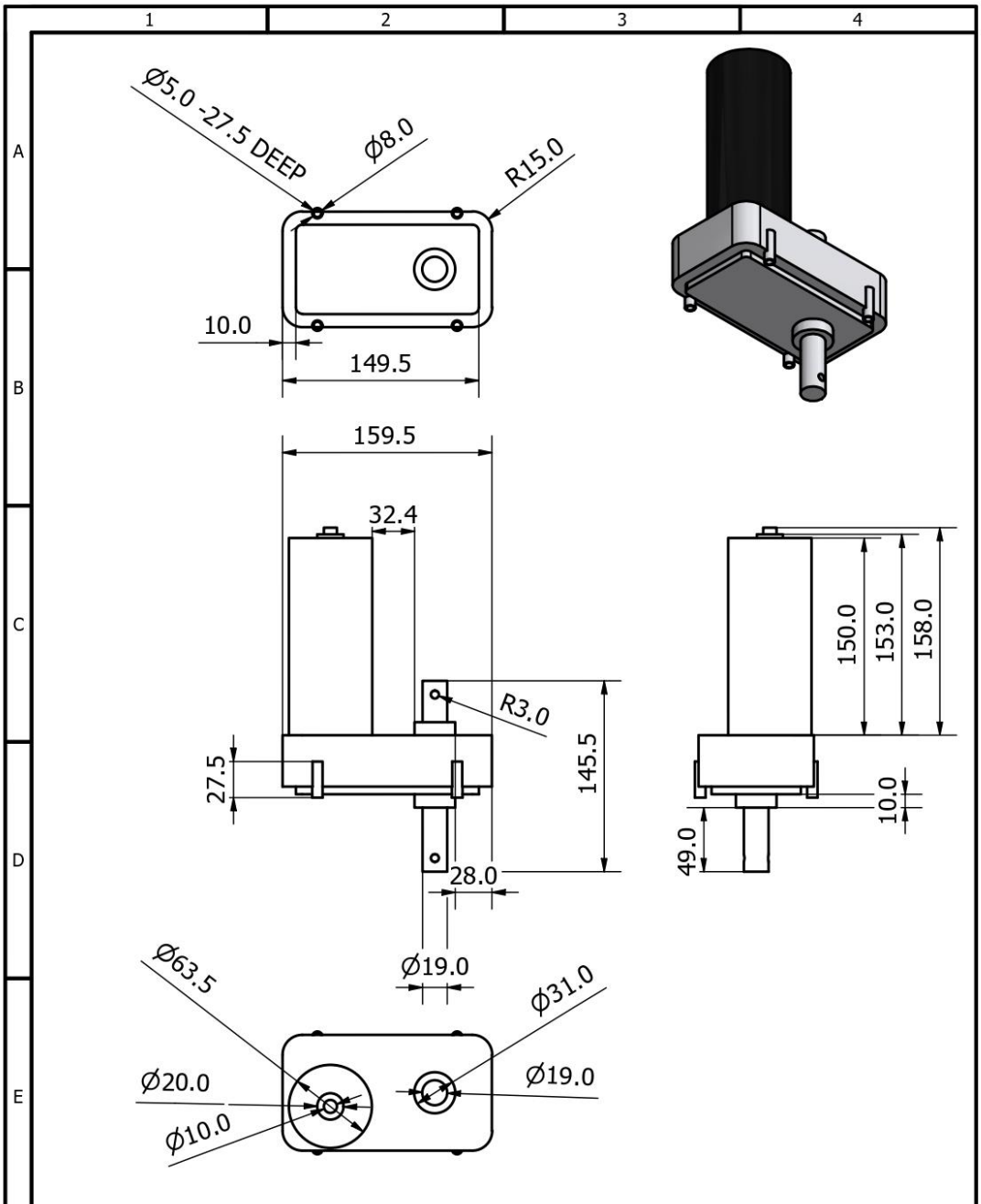


ITEM	CANTIDAD	NOMBRE	DESCRIPCIÓN
1	4	Rodamiento	Rodamiento rígido de bolas de una hilera con dos sellos RS1 SFK
2	2	Eje	
3	1	Caja	

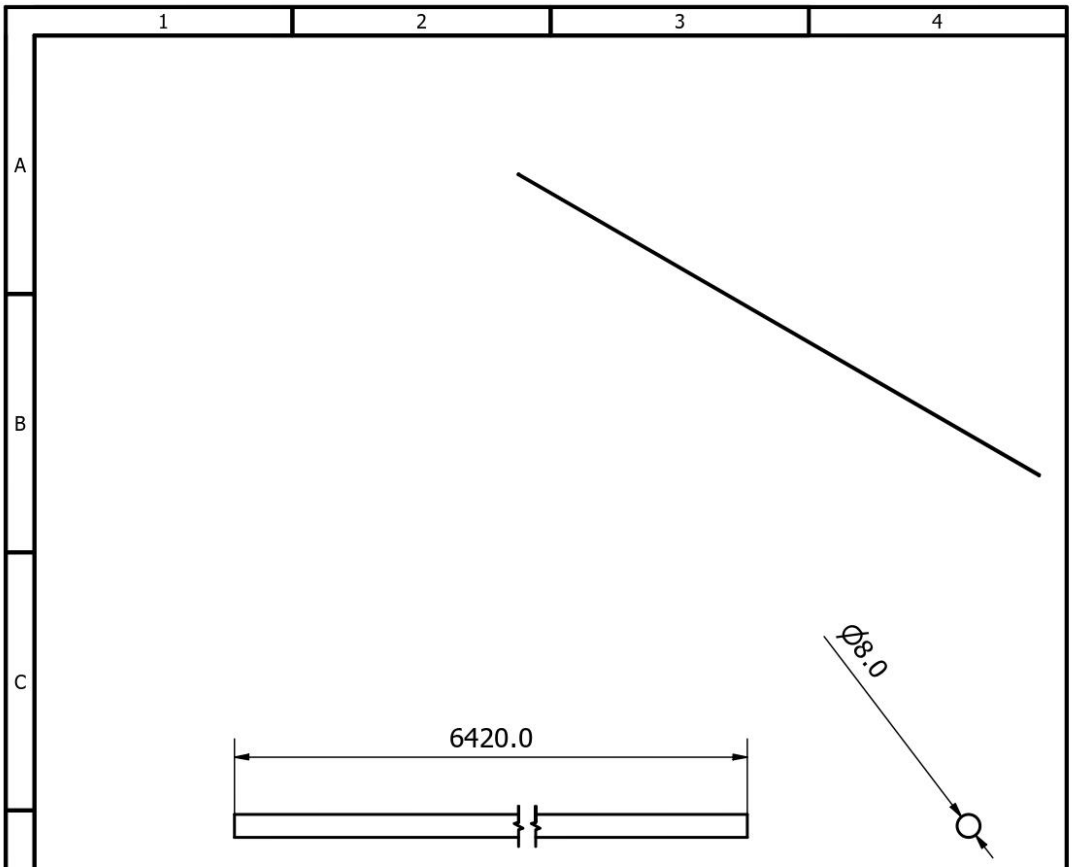
				Tolerancias		N/A	Materiales	
							ACERO	
					Fecha	Nombre	Denominación	Escala
				Dib.	*	S.B & H.L	CARRITO	2 : 1
				Rev.	*	*		
				Apro.	*	*		
							Número de dibujo	
							3/12	
Edi-ción	Modifica-ción	Fecha	Nom-bre					



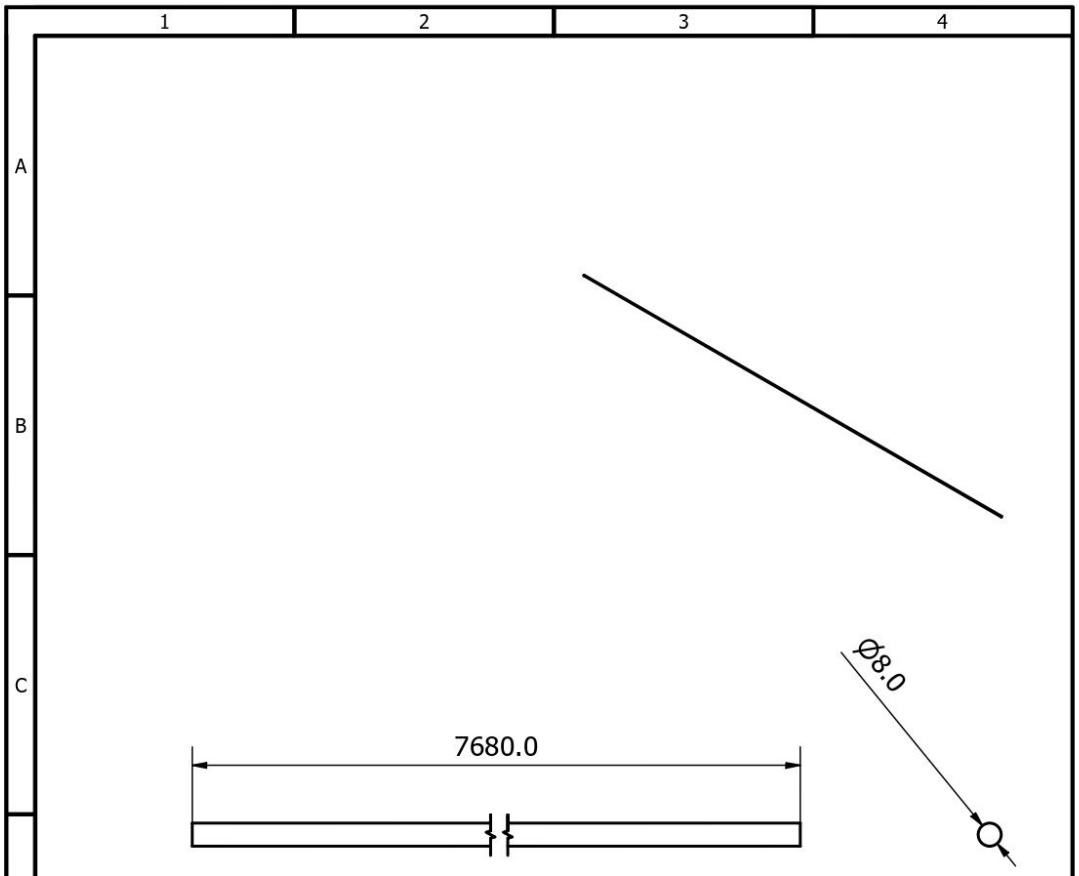
				Tolerancias		N/A	Materiales	
							ACERO	
					Fecha	Nombre	Denominación	Escala
				Dib.	*	S.B & H.L	CARRITO	1 : 1
				Rev.	*	*		
				Apro.	*	*		
							Número de dibujo	
							4/12	
Edi-ción	Modifica-ción	Fecha	Nom-bre					



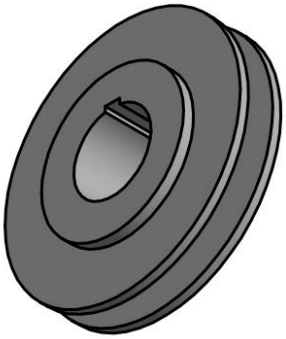
				Tolerancias		N/A	Materiales	
					Fecha	Nombre	Denominación	Escala
				Dib.	*	S.B & H.L	MOTOR DC	1 : 4
				Rev.	*	*		
				Apro.	*	*		
							Número de dibujo 5/12	
Edición	Modificación	Fecha	Nombre					

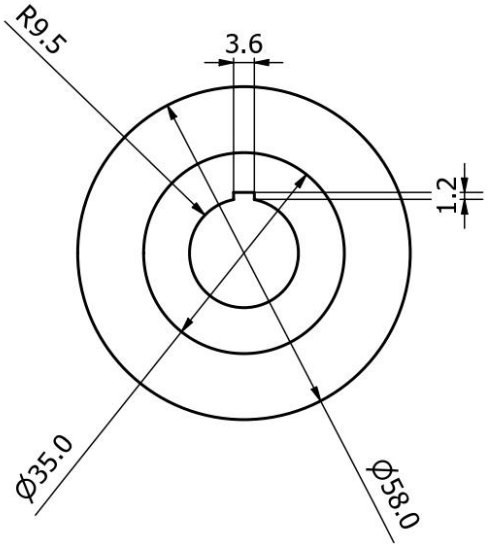


				Tolerancias		N/A	Materiales	
							ACERO	
					Fecha	Nombre	Denominación	Escala
				Dib.	*	S.B & H.L	EJE DE 6 M	1 : 50
				Rev.	*	*		
				Apro.	*	*		
							Número de dibujo	
							6/12	
Edi-ción	Modifica-ción	Fecha	Nom-bre					

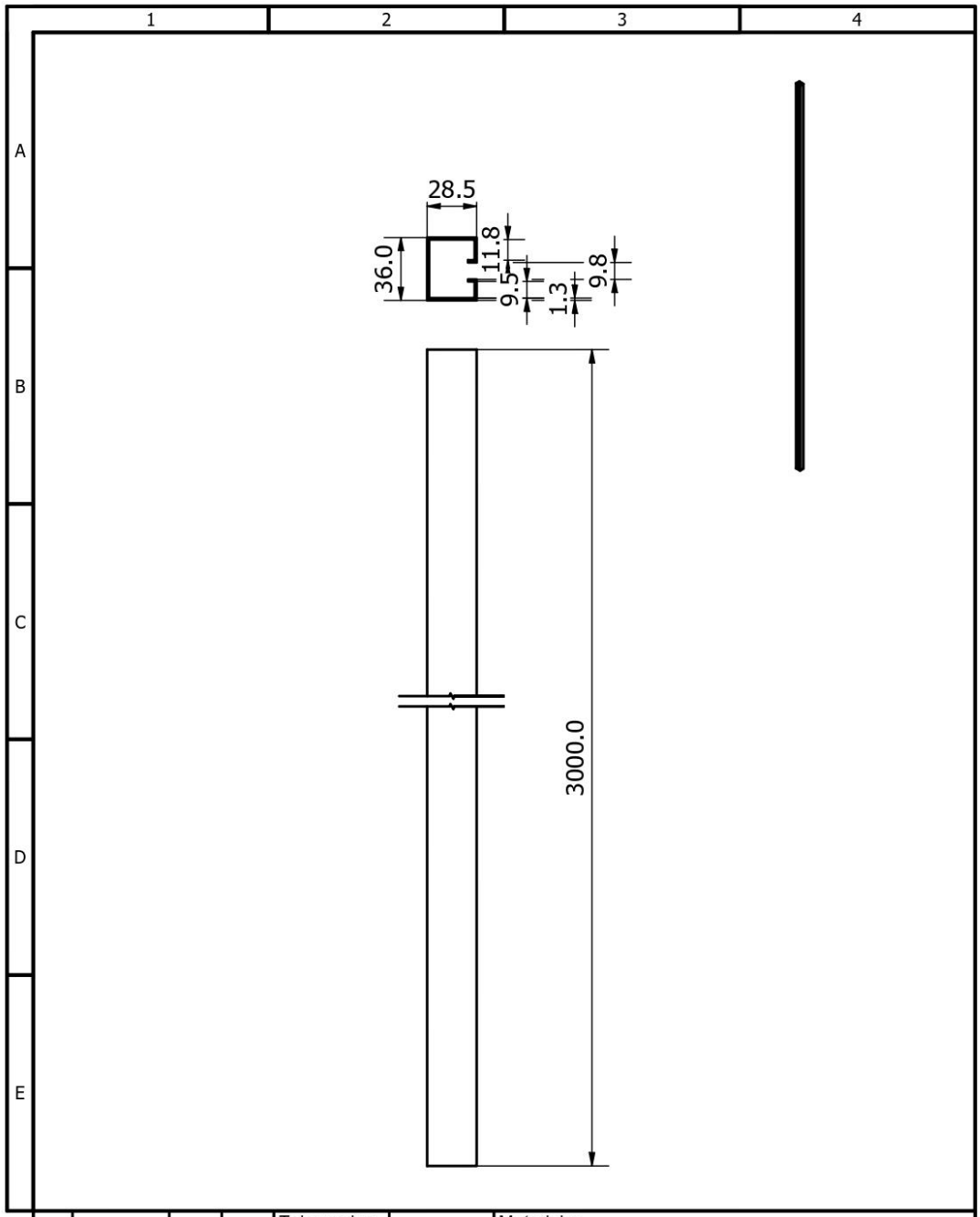


				Tolerancias		N/A	Materiales	
							ACERO	
					Fecha	Nombre	Denominación	Escala
				Dib.	*	S.B & H.L	EJE DE 7 M	1 : 75
				Rev.	*	*		
				Apro.	*	*		
							Número de dibujo	
							7/12	
Edi-ción	Modifica-ción	Fecha	Nom-bre					

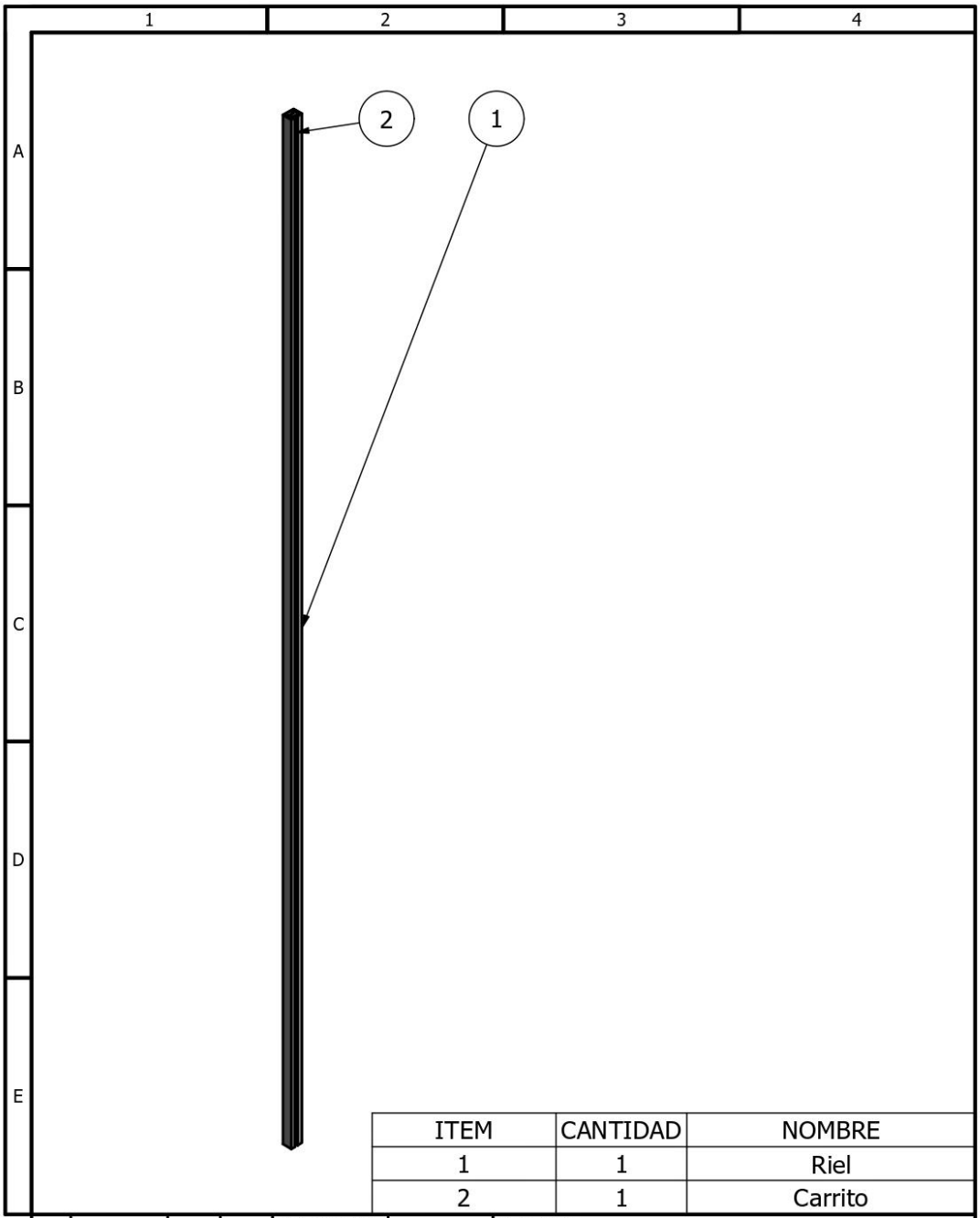




				Tolerancias		N/A	Materiales	
							ALUMINIO	
					Fecha	Nombre	Denominación	Escala
				Dib.	*	S.B & H.L	POLEA DE 19 MM	1 : 1
				Rev.	*	*		
				Apro.	*	*		
							Número de dibujo	
							8/12	
Edi-ción	Modifica-ción	Fecha	Nom-bre					

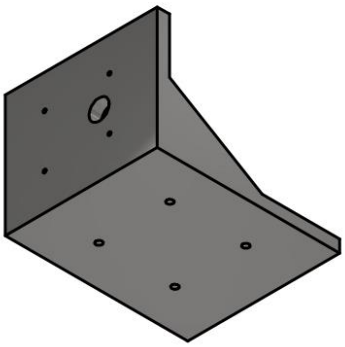


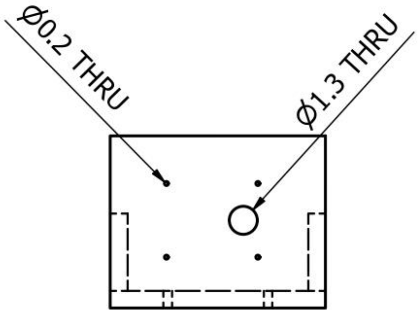
				Tolerancias		N/A	Materiales	
							ACERO	
					Fecha	Nombre	Denominación	Escala
				Dib.	*	S.B & H.L	RIEL	0,03 : 1
				Rev.	*	*		
				Apro.	*	*		
							Número de dibujo	
							9/12	
Edi-ción	Modifica-ción	Fecha	Nom-bre					

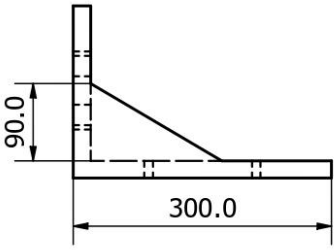


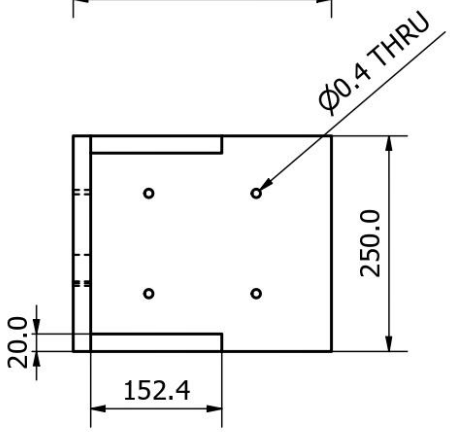
ITEM	CANTIDAD	NOMBRE
1	1	Riel
2	1	Carrito

				Tolerancias		N/A	Materiales	
							ACERO	
					Fecha	Nombre	Denominación	Escala
				Dib.	*	S.B & H.L	SISTEMA RIEL- CARRITO	0,08 : 1
				Rev.	*	*		
				Apro.	*	*		
							Número de dibujo	
							10/12	
Edi-ción	Modifica-ción	Fecha	Nom-bre					

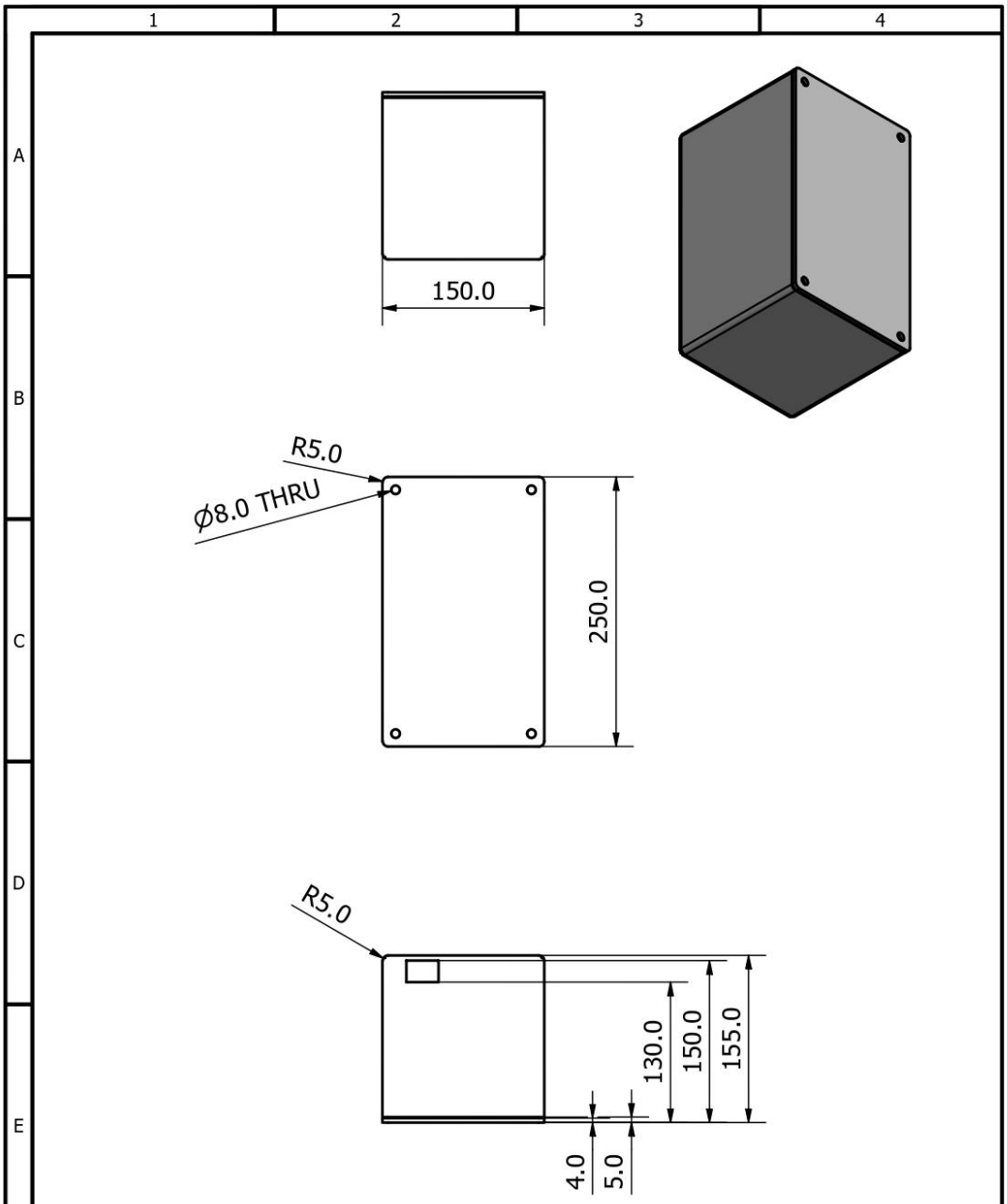








				Tolerancias		N/A	Materiales	
					Fecha	Nombre	Denominación	Escala
				Dib.	*	S.B & H.L	SOPORTE DE MONTAJE	0,15 : 1
				Rev.	*	*		
				Apro.	*	*		
							Número de dibujo 11/12	
Edi-ción	Modifica-ción	Fecha	Nom-bre					

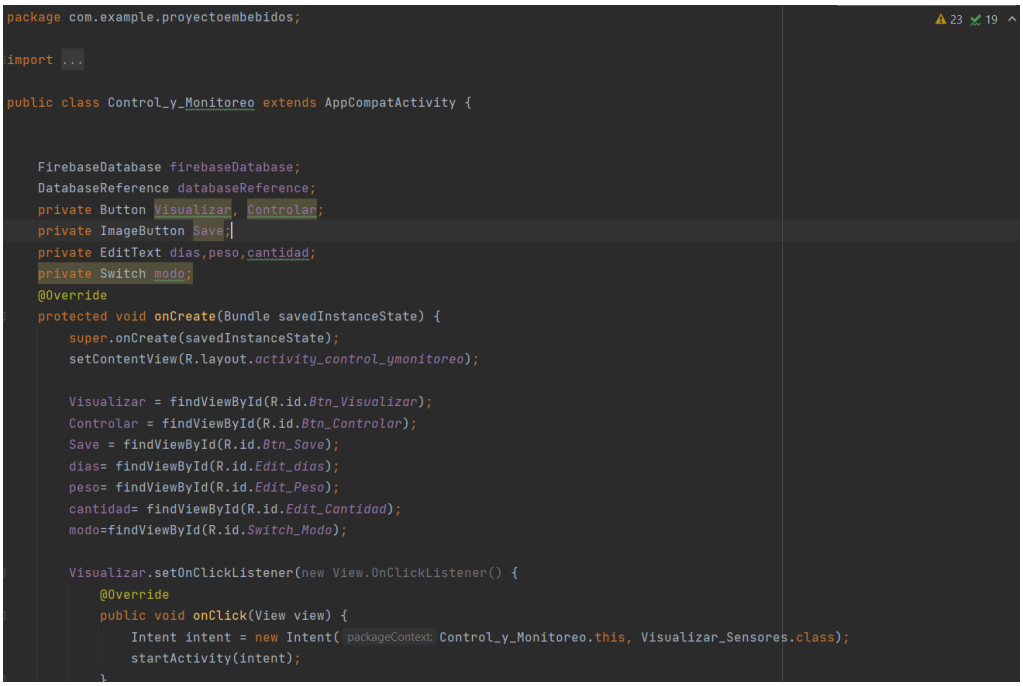


				Tolerancias		N/A	Materiales	
							ACERO	
					Fecha	Nombre	Denominación	Escala
				Dib.	*	S.B & H.L	CAJA DEL CIRCUITO	1 : 5
				Rev.	*	*		
				Apro.	*	*		
							Número de dibujo	
							12/12	
Edi-ción	Modifica-ción	Fecha	Nom-bre					

APÉNDICE D

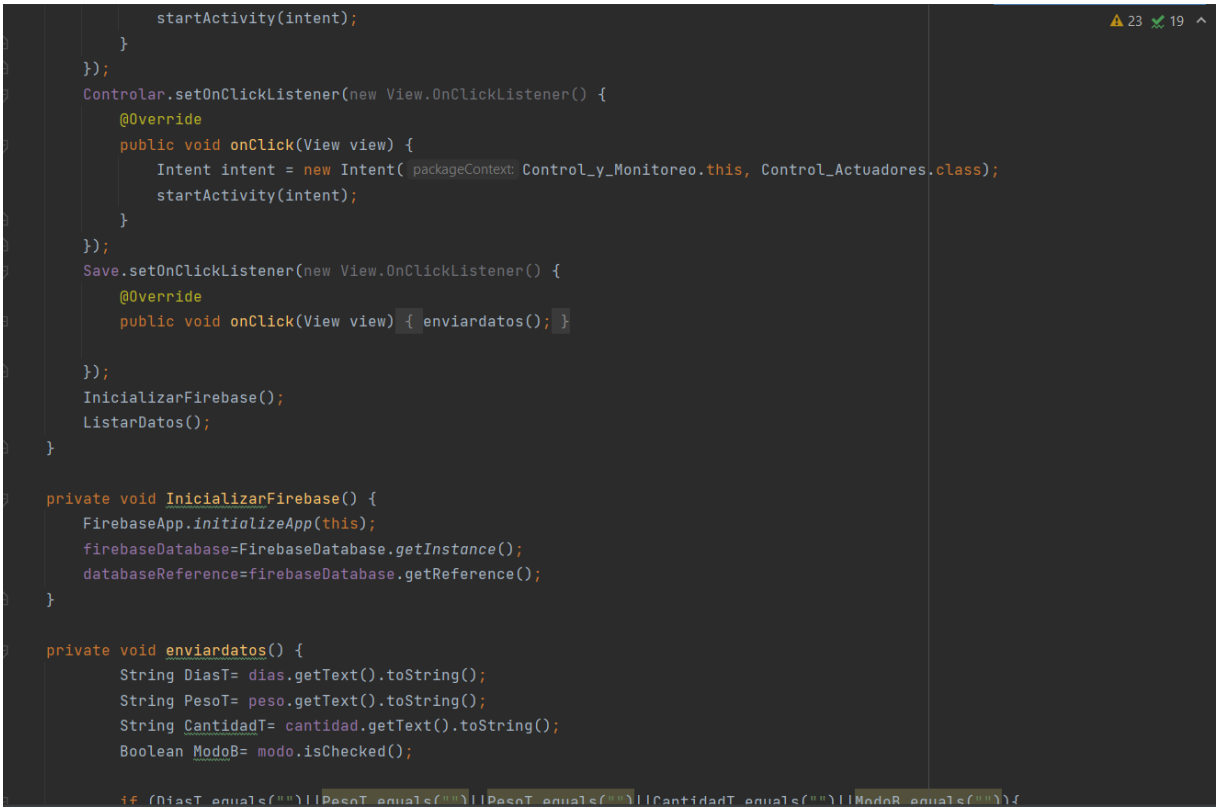
Aplicación móvil

El código presentado a continuación explica la lógica realizada para el control automático del motor. Para esto la aplicación presentará los datos guardados dentro de la base de datos de firebase, así como el modo de activación del motor. Estos datos son modificables y así mismo serán almacenados dentro de firebase.



```
package com.example.proyectoembebidos;

import ...

public class Control_y_Monitoreo extends AppCompatActivity {


    FirebaseDatabase firebaseDatabase;
    DatabaseReference databaseReference;
    private Button Visualizar, Controlar;
    private ImageButton Save;
    private EditText dias,peso,cantidad;
    private Switch modo;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_control_ymonitoreo);

        Visualizar = findViewById(R.id.Btn_Visualizar);
        Controlar = findViewById(R.id.Btn_Controlar);
        Save = findViewById(R.id.Btn_Save);
        dias= findViewById(R.id.Edit_dias);
        peso= findViewById(R.id.Edit_Peso);
        cantidad= findViewById(R.id.Edit_Cantidad);
        modo=findViewById(R.id.Switch_Modo);

        Visualizar.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Intent intent = new Intent( packageContext: Control_y_Monitoreo.this, Visualizar_Sensores.class);
                startActivity(intent);
            }
```

Figura D. 1 Definición de variables de pantalla de Control y Monitoreo de Actuadores.



```
            startActivity(intent);
        }
    });
    Controlar.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Intent intent = new Intent( packageContext: Control_y_Monitoreo.this, Control_Actuadores.class);
            startActivity(intent);
        }
    });
    Save.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) { enviardatos(); }

    });
    InicializarFirebase();
    ListarDatos();
}

private void InicializarFirebase() {
    FirebaseApp.initializeApp(this);
    firebaseDatabase=FirebaseDatabase.getInstance();
    databaseReference=firebaseDatabase.getReference();
}

private void enviardatos() {
        String DiasT= dias.getText().toString();
        String PesoT= peso.getText().toString();
        String CantidadT= cantidad.getText().toString();
        Boolean ModoB= modo.isChecked();

        if (DiasT.equals("")||PesoT.equals("")||PesoT.equals("")||CantidadT.equals("")||ModoB.equals("")){
```

Figura D. 2 Conexión de la pantalla a la base de datos de Firebase y asignación de eventos a los botones.

```
}
private void ListarDatos() {
    databaseReference.child("Modo").addValueEventListener(new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot snapshot) {
            DataSnapshot objSnapshot = snapshot;
            Control control = objSnapshot.getValue(Control.class);
            dias.setText(control.getDias());
            peso.setText(control.getPeso());
            cantidad.setText(control.getCantidad());
            modo.setChecked(control.getModo());


        }

        @Override
        public void onCancelled(@NonNull DatabaseError error) {

        }
    });
}

private void validación() {
    String DiasT= dias.getText().toString();
    String PesoT= peso.getText().toString();
    String CantidadT= cantidad.getText().toString();

    if (DiasT == ""){
        dias.setError("Campo Requerido");
    }
```

Figura D. 3 Lógica del formulario de registro de los parámetros para el control de los actuadores.

El código para el desarrollo de la aplicación se encuentra en el siguiente repositorio de github: https://github.com/Henrylm4/Aicobb