UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMA, ELECTRÓNICA E INDUSTRIAL

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y COMUNICACIONES

Tema:

"SISTEMA ELECTRÓNICO AUTOMÁTICO DE ALIMENTACIÓN PARA MASCOTAS EN EL HOGAR"

Trabajo de Graduación Modalidad: Proyecto de Investigación, presentado previo a la obtención del título de Ingeniero en Electrónica y Comunicaciones.

LÍNEA DE INVESTIGACIÓN: Tecnología de la Información y Sistemas de Control.

Autor: Lenin Patricio Dávila Paredes

Tutor: Ing. Juan Pablo Pallo Noroña Mg.

AMBATO – ECUADOR

Enero 2020

APROBACIÓN DEL TUTOR

En mi calidad de tutor del trabajo de investigación sobre el tema: "SISTEMA ELECTRÓNICO AUTOMÁTICO DE ALIMENTACIÓN PARA MASCOTAS EN EL HOGAR" realizado por el señor Lenin Patricio Dávila Paredes, estudiante de la carrera de Ingeniería en Electrónica y Comunicaciones de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, considero que el informe investigativo reúne los requisitos suficientes para que continúe con los trámites y consiguiente aprobación de conformidad con el numeral 7.2 de los Lineamientos Generales para la aplicación de Instructivos de las Modalidades de Titulación de las Facultades la Universidad Técnica de Ambato.

Ambato Enero, 2020

EL TUTOR

.....

Ing. Juan Pablo Pallo Noroña Mg.

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato, para que haga uso de este Trabajo de Titulación como un documento disponible para la lectura, consulta y procesos de investigación.

Cedo los derechos de mi trabajo de Titulación, con fines de difusión pública, además autorizo su reproducción dentro de las regulaciones de la Universidad

Ambato enero, 2020

.....

Lenin Patricio Dávila Paredes

CC: 1803474921

AUTORÍA DEL TRABAJO DE TITULACIÓN

El presente proyecto de investigación titulado: "SISTEMA ELECTRÓNICO AUTOMÁTICO DE ALIMENTACIÓN PARA MASCOTAS EN EL HOGAR", es absolutamente original, auténtico y personal, en tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato enero, 2020

.....

Lenin Patricio Dávila Paredes

CC: 1803474921

APROBACIÓN DEL TRIBUNAL DE GRADO

La Comisión Calificadora del presente trabajo conformada por los señores docentes Ing. Santiago Manzano Mg., Ing. Jeovanni Brito Mg., revisó y aprobó el Informe Final del Proyecto de Investigación titulado "SISTEMA ELECTRÓNICO AUTOMÁTICO DE ALIMENTACIÓN PARA MASCOTAS EN EL HOGAR", presentado por el señor Lenin Patricio Dávila Paredes, de acuerdo al numeral 9.1 de los Lineamientos Generales para la aplicación de Instructivos de las Modalidades de Titulación de las Facultades de la Universidad Técnica de Ambato.

.....

Ing. Elsa Pilar Urrutia Urrutia Mg.
PRESIDENTA DEL TRIBUNAL

.....

Ing. Santiago Manzano Mg.
DOCENTE CALIFICADOR

.....

Ing. Geovanni Brito Mg.
DOCENTE CALIFICADOR

DEDICATORIA

*A mi familia**,** por el esfuerzo y apoyo incondicional que me brindaron durante el transcurso de toda la carrera, lo cual me motivó a seguir adelante para llegar a cumplir uno más de mis objetivos planteados, muchas gracias de todo corazón.*

A mi tutor y amigo Ing. Juan Pablo Pallo quien me guió desde los primeros semestres en la facultad y durante el transcurso del proyecto de investigación.

Lenin Patricio Dávila Paredes

AGRADECIMIENTO

Agradezco a Dios por brindarme salud y bendiciones durante toda mi vida.

A mis padres y abuelitos por su ayuda económica y sobre todo por el amor y fortaleza que me daban en los momentos difíciles.

A los docentes de la carrera de Ingeniería en Electrónica que compartieron sus conocimientos, ayudándome en el proceso de formación académica.

A mis amigos y futuros colegas con quienes viví momentos inolvidables, lo cual hizo que el transcurso de la carrera fuera muy placentera y divertida.

Lenin Patricio Dávila Paredes

ÍNDICE GENERAL DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

La finalidad del proyecto de investigación es proveer un dispensador automático de comida para mascotas, el cual permite proporcionar tres comidas diarias necesarias para su correcta alimentación. El dueño de las mascotas podrá seleccionar el horario según su criterio ya sea de forma manual o de manera automática.

Los horarios de alimentación de manera automática, se lo realiza tomando en cuenta los parámetros de peso y edad de la mascota. Estos parámetros sirven para proporcionar la cantidad de alimento indicado acorde a las marcas comerciales de croquetas en el mercado actual.

Como material de construcción se emplea acero inoxidable, debido a la adaptabilidad ya sea a ambientes internos como externos, contando con una capacidad de almacenamiento de comida de hasta 20kg, apta para el consumo de razas grandes en un periodo de 15 hasta 21 días.

El desarrollo del dispensador de comida para mascotas está diseñado en base hardware y software libre. La parte mecánica es controlada por un microcontrolador NODEMCU el cual esta codificado en Arduino. Sus respectivas interfaces tanto web como apk son desarrolladas con la integración de la plataforma de Firebase, con el programa de Android Studio, permitiendo generar un sistema de control y monitoreo basado en IOT.

La seguridad está basada en el cifrado de código QR, permitiendo asignar un dispositivo a la aplicación respectiva y el acceso de usuarios en base a la autenticación mediante el correo electrónico personal, contando con un respaldo en caso de olvidar la contraseña.

El sistema al ser probado por diferentes usuarios, presento ventajas en optimización de tiempo laboral, y genero tranquilidad en los dueños de las mascotas ya que recibían una alimentación diaria balanceada.

ABSTRACT

The purpose of the research project is to provide an automatic pet food dispenser, which will allow you to provide three daily meals necessary for proper feeding. The final consumer can select the schedule according to their criteria either manually or automatically.

The feeding schedules are automatically done, taking into account the parameters of weight and age of the pet. These parameters help us to provide the quantity of food indicated according to the commercial brands of croquettes in the current market.

The construction material used is stainless steel, a material that adapts to both internal and external environments, with a capacity of 20kg, suitable for the consumption of large breeds in a period of 15 to 21 days.

The development of the pet food dispenser is designed based on free hardware and software. Whose mechanical part is controlled by a NODEMCU microcontroller encoded in Arduino. And their respective interfaces both web and apk are developed thanks to the integration of the Firebase platform, with the Android Studio program, allowing to generate a control and monitoring system based on IOT.

Security is based on the encryption of QR code which allows a device to be assigned to the respective application and user access based on authentication through personal email, with a backup in case of forgetting the password.

The system being tested by different users, presented advantages in optimization of working time, and generated tranquility in the owners of pets as they received a balanced daily diet.

CAPÍTULO I

MARCO TEÓRICO

1.1 Antecedentes Investigativos

En diferentes países a nivel mundial se ha venido desarrollando algunos tipos de dispensador de comida para mascotas, facilitando el cuidado de las mismas, ya sea por falta de tiempo o por motivos de viajes.

El 2016 en Indonesia se desarrolló un dispensador de comida para perros usando comunicación inalámbrica, MQTT y Android client, el cual funciona por medio de un teléfono inteligente aplicando tecnología RFID, el sistema se desarrolló en base a la codificación de la placa de arduino y el módulo ESP8266 wifi basado en el lenguaje de programación LUA, su funcionamiento consistía en el registro de un usuario el cual seleccionaba un dispositivo precargado en la aplicación al igual que la selección de horario de alimentación, al momento en que se activa el dispensador y el perro se acerca por medio de la tecnología RFID se registra las veces que este se alimentara. [1]

En 2017 en Colombia se desarrolla un dispensador de mascotas controlado remotamente, el cual se basa su funcionamiento en la web, en base a lenguajes de programación como HTML5, Java Script y Python. La interfaz que se desarrolló permite al usuario su inicio de sesión y activar como desactivar el uso del dispensador ya sea en comida como para agua a través del uso de la placa raspberry pi el cual se encarga de controlar la parte mecánica de esta forma se pretende facilitar el cuidado de las mascotas y tranquilidad de sus dueños. [2]

En 2018 en Lima-Perú se estudia la viabilidad que tendría un dispensador de comida para mascotas, el cual pueda ser controlado por medio de una aplicación celular, esta idea consta de un registro por dispensador basado en códigos QR y accionamiento por medio de un click en los horarios preestablecidos, este estudio se realizó tomando en

cuenta el porcentaje de mascotas a nivel de 7 diferentes países, obteniendo como resultado de un estudio estadístico del proyecto genera ganancias usando un WACC de 32.18%. [3]

En Ecuador se han desarrollado dispensadores de comida para mascotas. En el 2017 en Cuenca se desarrolló un dispensador que podía proporcionar comida por 6 días de forma continua, contando con un respaldo de energía eléctrica, el mecanismo era accionado en base a una caja de control que se encarga de activar el controlador formado por un arduino MEGA, las ordenes enviadas eran visualizadas en un display ubicado en el propio case y de esta forma brindaban seguridad a las personas que poseen mascotas en sus hogares. [4]

1.1.1 Contextualización del problema

Actualmente se desconoce la cifra correcta de mascotas que existe en Ecuador debido a que no se ha presentado un censo que nos brinde un número aproximado, pero acorde a estudios realizados por la Secretaría de salud del Distrito Metropolitano de Quito se asume que en valores cercanos 3 de cada 5 familias presentan mascotas en casa, estos datos son tomados como referenciales debido a que en la ciudad de Ambato no se cuenta con alguna cifra respecto al estudio de mascotas en el hogar. [5]

Las personas no permanecen constantemente en casa debido a diferentes circunstancias como: estudio, trabajo, viajes por este motivo se suele descuidar la dieta de las mascotas especialmente en los perros, los cuales no son alimentados en horarios fijos o incluso en ocasiones no reciben todas sus comidas.

Al descuidar la alimentación de las mascotas estas pueden desarrollar enfermedades como: obesidad, hiperparatiroidismo, osteodistrofias, piel escamosa, diarrea, vómito, letargo y patologías cardiovasculares, por lo general estas consecuencias se presentan por el mal funcionamiento del metabolismo, además de la ausencia de calcio, fósforo, aminoácidos y proteínas que están presentes en una dieta canina balanceada. [6] [7]

1.2 Fundamentación teórica

1.2.1 Sistemas electrónicos

Los sistemas electrónicos constan de un conjunto de circuitos que trata e interactúan con señales eléctricas llegando a cumplir determinadas funciones. Estos sistemas constas de tres etapas entre las cuales tenemos: Entrada, Control y Salida mostrados en la **Figura 1.**

- Las entradas se encargan de recoger diferentes tipos de datos del medio, convirtiéndolas en señales de corriente o voltaje
- La etapa de control está integrada por los diferentes componentes los cuales están conectados entre sí y se encargan de interactuar, procesar y gestionar los datos recopilados por las entradas
- Las salidas se encargan de convertir las señales procesados en señales físicas útiles. [8]

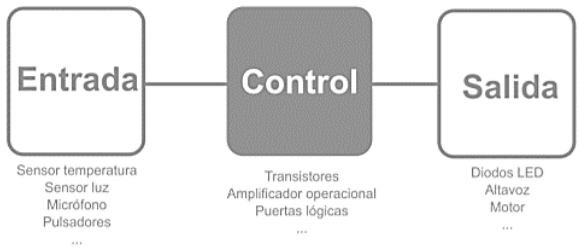


Figura 1 Estructura del Sistema Electrónico [8]

Materia prima

Entre los mejores materiales de construcción tanto para ambientes internos como externos podemos encontrar al acero inoxidable y el aluminio por lo cual en la **Tabla 1** encontramos algunas características principales ocupadas en la construcción de proyectos.

Tabla 1 Diferencias entre materiales de construcción

Diferencias	Peso	Durabilidad	Comida	Vida útil	Costo
Acero Inoxidable	Pesado	Mayor resistencia	Apto	Mayor	Mayor
Aluminio	Ligero 1/3	Menor resistencia	No apto	Menor	Menor

Elaborado por: El investigador

Sensores

Los sensores son dispositivos cuya capacidad permite medir magnitudes físicas o químicas estas se denominan variables. Entre estos podemos encontrar sensores: Digitales, Analógicos y de Comunicación por Bus. [10]

Actuadores

Los actuadores son dispositivos inherentemente mecánicos capaces de transformar energía hidráulica, neumática o eléctrica con la finalidad de generar efectos en algún elemento externo. [10]

Controlador

Los controladores son un pequeño sistema inteligente en cuya entrada se puede apreciar un sensor, indicador o una salida regulada. Entre estos tenemos controladores continuos, discontinuos. [11]

1.2.2 Sistema de respaldo eléctrico

Actualmente los sistemas de respaldo son de gran utilidad permitiendo que los equipos operen con normalidad sin interrumpir las funciones para las que fueron diseñados. Estos sistemas están formados por componentes y dispositivos electrónicos generando un suministro eléctrico continuo, el cual funciona acorde al número de baterías y equipos conectados a este. [12]

1.2.3 Internet de las cosas (IOT)

El internet de las cosas suele conocerse también como IOT debido a sus siglas en inglés, su concepto se basa en la interconexión e interacción entre parte de un mundo digital con respecto al mundo físico, lo cual en base a la tecnología que ocupemos permite conectar cosas a medios de transmisión presentes en la infraestructura de internet. [13]

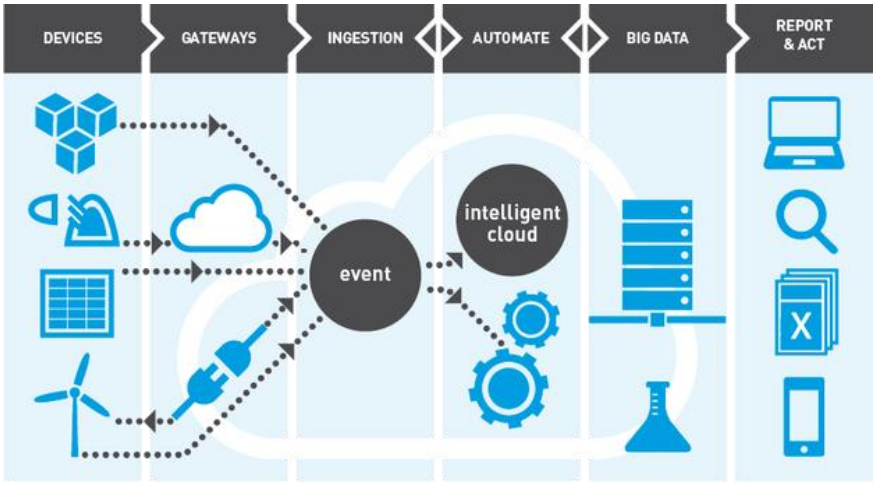


Figura 2 Estructura de un Sistema IO [13]

Conectividad

La conectividad en los dispositivos IoT puede ir desde Bluetooth de bajo consumo hasta 4G LTE (Long Term Evolution) o LTE Advanced o bien una combinación de tipos de comunicación. [14]

Fases en la integración IOT

Fase 1: Conexión. Los objetos que serán conectados en red con sensores inteligentes que empezarán a enviar información sobre sí mismos y su entorno a su centro de comunicaciones. Conectar cosas representa el comienzo de la evolución del IoT. [14]

Fase 2: Análisis y Visualización. A medida que los datos del sistema se acumulan, se empieza a ejecutar un análisis inteligente en las pilas de datos y se visualiza los resultados. [14]

Fase 3: Automatización. Para este proceso se realiza un sistema autónomo. [14]

Arquitectura

La arquitectura básica de IOT está dividida en tres capas como se muestra en la **Figura N 3:**

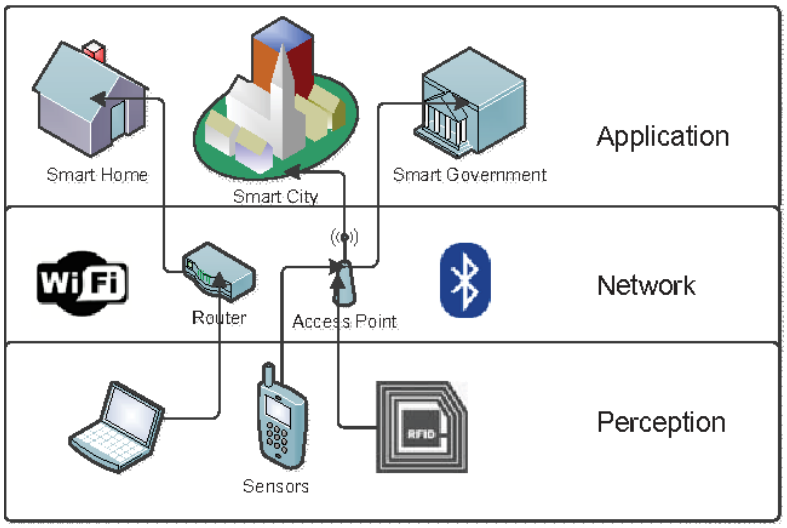


Figura 3 Estructura de un Sistema IOT [14].

- **Capa de percepción (Perception Layer):** En este primer nivel recoge las diferentes propiedades físicas de los diferentes objetos estos datos pueden ser: temperatura, humedad, localización, mediante sensores los cuales convierten las señales adquiridas y enviarlas por medio de la red. [15]
- **Capa de red (Network Layer):** En esta capa se envían los datos recogidos de la capa anterior al destino a través de la red la cual puede contar con cualquier tipo de tecnología. [15]
- **Capa de aplicación (Application Layer):** En la capa de aplicación se desarrollan diferentes aplicaciones en base al objetivo y los datos recogidos, siendo esta la capa de mayor importancia en un sistema IOT. [15]

Protocolos para IOT:

En la **Figura 4** se aprecia los diferentes tipos de protocolos con los que cuenta un sistema IOT, dentro de los cuales tenemos los siguientes: [16]

- OPC UA
- HTTP
- DDS

- CoAP
- MQTT
- AMQP

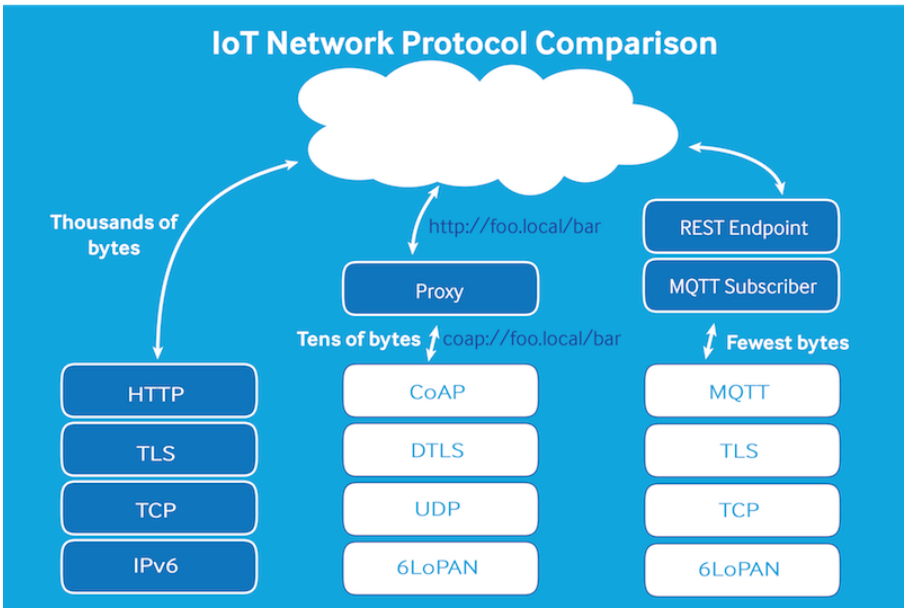


Figura 4 Protocolo de un Sistema IOT [16]

1.2.4 Mascotas

Hoy en día a nivel mundial tenemos diferentes núcleos familiares y entre ellos lo más habitual es tener algún tipo de mascota en su hogar, siendo los más comunes los perros y gatos ubicándose en el primer plano ya que en segundo plano tenemos otro tipo de mascotas como aves, peces o algún tipo de roedor. [17]

- **Perros**

  Los perros han venido siendo la principal compañía de los hombres desde tiempo atrás y se ha convertido en la mejor mascota por excelencia ya que nos ofrecen su protección, compañía y al mismo tiempo son incondicionales. [17]

- **Gatos**

  Los gatos han llegado a posicionarse en el segundo lugar debido a que es la mascota con mayor preferencia entre personas libres de compromiso y la más apta para vivir en las ciudades debido a su tamaño. [17]

Razas de perros

Las razas de perros están conformadas por animales que comparten características comunes entre sí lo cual ha dado lugar a la formación de tres grupos dividiéndolos en Pequeños, Medianos y Grandes. [18]

- **Pequeñas**: Esta raza consta con un peso entre 3 -14 kilos los cuales conllevan ventajas en cuanto a las otras razas, debido a que son tranquilos y se los puede transportar fácilmente en cuanto a paseos no requieren gran actividad física. [19]
- **Medianas:** Esta raza consta con un peso 14 -25 kilos los cuales se adaptan en espacios medianos y grandes cuyos cuidados son de facilidad al momento de atenderlos. [20]
- **Grandes:** Esta raza consta con un peso mayor a 25 kilos son ideales para compartir horas de ejercicio, ya que corren trayectos hasta de 70 km, por lo general con tendencias independientes. [21]

Factores que influyen en las necesidades nutricionales del perro

Los principales factores a tomaren cuenta para la alimentación de un perro son tamaño y peso. Además, se basa en la cantidad de calorías que consume un perro ya que por lo general los perros sedentarios consumen menos cantidad de energía. Las porciones adecuadas de ingesta diaria se describen en la **Tabla 2** [22]

Tabla 2 Porciones acorde edad – peso [22]

Tamaño del Perro	EDAD			
	1- 3(meses)	3-5(meses)	5-8(meses)	8-12(meses)
1 – 5 Kg	25 – 100 g	40 – 130 g	45 – 130 g	40 g
5 – 12 Kg	70 – 240 g	110 – 280 g	130 – 290 g	130 g
12 – 25 Kg	130 -410 g	240 – 480 g	280 – 450 g	290 – 420 g
25 – 45 Kg	210 – 500 g	410 – 760 g	480 – 840 g	450 – 700 g
45 – 70 Kg	280 – 700 g	590 – 1020 g	760 – 1100 g	840 – 1100 g

Razas de gatos

Las razas de gatos tomando en cuenta el factor de alimentación se las puede dividir principalmente en dos tipos: raza pequeña y raza grande.

- Pequeña: Los gatos de raza pequeña se adapta a cualquier lugar en el ámbito doméstico y son excelentes compañeros. [23]
- Grande: Los gatos de raza grande tienen un temperamento tranquilo y afable. [24]

Las porciones de alimentación adecuadas para la ingesta diaria en un gato se describen en la **Tabla 3**

Tabla 3 Porciones acorde edad – peso [24]

Peso del Gato	Adulto	Medio
2 – 3 Kg	40 – 50	55 – 70
3 – 4 Kg	50 – 60	70 – 85
4 – 5 Kg	60 – 70	85 – 100
5 – 6 Kg	70 – 80	100 – 110
6 – 7 Kg	80 – 90	110 – 120

Enfermedades en mascotas por mala alimentación

Entre las principales enfermedades que puede tener una mascota por mala alimentación tenemos: obesidad, hiperparatiroidismo, osteodistrofias, piel escamosa, diarrea, vómito, letargo y patologías cardiovasculares. [25]

- **Obesidad:** La obesidad en las mascotas ocasiona problemas en el crecimiento de las mascotas y en adultos genera problemas articulares, cardiacos y diabetes. [26]

- **Hiperparatiroidismo:** Esta enfermedad es poco común en perros y con un menor porcentaje en un gato, se debe a la falta de calcio presentes en la comida o que conlleva problemas a nivel intestinal, renal, tracto urinario, neuromuscular y cardiaco. [27]

- **Piel escamosa:** La causa principal para esta enfermedad es la carencia de ácidos grasos en la alimentación y omega-3. [28]

- **Diarrea y Vómito:** La diarrea en las mascotas la mayoría de veces es causada por comidas que les sientan mal o debido al exceso de comida, aunque también las crónicas las pueden ocasionar paracitos. [29]

- **Letargo:** Generalmente se da por la disminución de actividad en las mascotas, además presenta cambios en su apetito, o simplemente se reúsa a comer. [30]

1.3 Objetivos

1.3.1 Objetivo General

Implementar un Sistema Electrónico automático de alimentación para mascotas en el hogar.

1.3.2 Objetivos específicos

- Analizar el tipo de alimentación diaria que necesita un perro y gato tomando en cuenta la edad y el tipo de raza.
- Diseñar el mecanismo de control del Sistema Electrónico automático de alimentación para mascotas, en base al diseño de su estructura mecanica.
- Diseñar la interface de control web y la apk basada en software libre, considerando la interaccion con una plataforma de almacenamiento en internet.

Una vez plantados los objetivos se procede con su respectivo análisis:

El objetivo del presente trabajo de investigación es implementar un Sistema Electrónico automático de alimentación para mascotas en el hogar basado en tecnología IOT, usando tanto software como hardware libre, el cual permite controlar y monitorear el sistema desde cualquier parte en que se encuentre el usuario.

Se debe analizar el método adecuado que se requiere para la correcta alimentación de las mascotas, basado en una dieta diaria saludable, la cual tenga presente el tipo de raza a la que pertenecen esta información fue obtenida en base a las siguientes actividades:

- Estudio de las razas de perros y gatos acorde a su tamaño.
- Estudio de las porciones de comida que necesita un perro y gato diariamente.

El sistema de control del presente proyecto deberá presentar su diseño correspondiente, el cual permitirá almacenar las debidas funciones que se realizara en base a su programación respectiva, para lo cual se desarrollan las siguientes actividades:

- Diseño del esquema del dispensador automático de mascotas
- Configuración del controlador del sistema.

Para que el sistema funcione de una manera remota desde cualquier dispositivo con acceso a internet, es necesario realizar el diseño de las interfaces de control tanto para la web como para los dispositivos celulares razón por la cual se sigue las siguientes actividades:

- Desarrollo de la interfaz web.
- Desarrollo de aplicación Android.

CAPÍTULO II

METODOLOGÍA

2.1 Materiales

Los materiales que conforman el presente proyecto de investigación se detallan en la **Tabla 4**, junto con la respectiva función a cumplir.

Tabla 4 Materiales utilizados en el proyecto de investigación

Material	**Grafico**	Utilidad
Acero Inoxidable		Material seleccionado para el case del proyecto, debido a durabilidad y adaptación a condiciones climáticas sin presentar corrosión.
Sensor Ultrasónico		Captar datos referentes a nivel de agua en el dispensador el cual se envía al controlador para toma de decisiones.
Servomotor		Activar el dosificador el cual permite el paso de las croquetas hacia el dispensador.
Electroválvula		Activar el control de paso de agua desde el tanque del dispensador hasta el recipiente del agua.

Nodemcu		Recibir señales de los sensores para ser almacenados en la nube y tomar decisiones en base a estos.
Cámara		Monitorear las acciones de la mascota y permite comunicarse con ella a través de la aplicación móvil.
Transformador		Transformar los 110v de entrada a 16v de salida.
Fuente		Controlar el paso de voltaje hacia la placa del controlador.
Batería		Genera energía en caso de pérdidas de electricidad en las viviendas.
Celular		Controlar las acciones tomas por parte de usuario y asignarlas al dispensador.
Firebase	Built with Firebase	Almacenar las variables q permiten la toma de decisiones en el controlador.
Android Studio	Android Studio	Software para elaboración de la interface móvil generando una apk.

Dreamweaver		Software para elaboración de la interfaz web.

Elaborado por: El investigador

2.2 Métodos

2.2.1 Modalidad de la investigación.

La modalidad presente en el dispensador automatizado para las mascotas del hogar, se fundamenta en la investigación, desarrollo e implementación aplicando los conocimientos adquiridos en el transcurso de la carrera de Ingeniería en Electrónica y Comunicaciones, permitiendo solventar los problemas que conlleva la mala alimentación de las mascotas. Para lo cual se decide aplicar las siguientes modalidades:

Investigación bibliográfica – documental.

Este tipo de investigación permite obtener datos referentes a proyectos similares realizados con anterioridad, obteniendo un sustento en los datos y estudios realizados por otros investigadores en los diferentes medios como: libros, publicaciones, artículos, revistas, buscadores en línea.

Además de los diferentes procedimientos que fueron aplicados y documentados en sus respectivos proyectos de investigación con lo cual se puede tener un soporte para futuros proyectos a realizar.

Investigación de campo.

Esta investigación se realiza en el mercado actual permitiendo recolectar información sobre los dispensadores de comida para mascota que están disponibles para los usuarios y conocer qué tipo de mecánica y forma de comunicación presentan, de esta forma proponer un proyecto de investigación mejorado que cumpla con los objetivos propuestos.

Investigación aplicada.

En este tipo de investigación se aplica los conocimientos adquiridos en el transcurso de la carrera en desarrollar un dispensador automático para mascotas, el cual se basa

en software libre, empezando desde su construcción hasta la respectiva codificación empleada.

2.2.2 Recolección de información

Para la recolección de información se toma como fuentes de información revistas indexadas, tesis y libros publicados en los últimos años con relación a dispensadores automatizados de comida y alimentación en las mascotas, lo cual tendrá un mayor nivel de confianza en la investigación.

2.2.3 Procesamiento y análisis de datos

Para el procesamiento de datos se tomó en cuenta la información obtenida en diferentes medios documentados, llegando a realizar un análisis crítico en los siguientes puntos:

- Estudio de razas de mascotas.
- Obtención de parámetros técnicos de componentes del sistema.
- Interpretación de datos en base a la solvencia del problema.
- Presentación de resultados con base al fundamento teórico.

2.2.4 Desarrollo del proyecto.

Para poder desarrollar el proyecto se debe realizar las siguientes actividades que permitan la implementación del dispensador automatizado de comida para mascotas:

- Estudio de las razas de perros y gatos acorde a su tamaño.
- Comparación del tipo de alimento para perros y gatos que hay en el mercado actual.
- Estudio de las porciones de comida que necesita un perro y gato diariamente.
- Diseño del esquema del dispensador automático para mascotas.
- Selección de los componentes y ensamblaje del sistema.
- Configuración del controlador del sistema.
- Instalación y configuración de base de datos.
- Desarrollo de interfaces web.
- Desarrollo de aplicación en sistemas celulares.
- Pruebas de funcionamiento del sistema.

CAPÍTULO III

RESULTADOS Y DISCUSIÓN

3.1 Análisis y discusión de los resultados

En base al análisis obtenido referente a las propuestas de diferentes usuarios a nivel mundial, se obtiene que se han desarrollado dos tipos de dispensadores controlados, estos son: mecánicos y automáticos.

- Los mecánicos tienen la función activar el dispensador de alimentos en base a un horario establecido tomando en cuenta el tamaño de la mascota.
- Los automáticos tienen las mismas funciones que los mecánicos con la diferencia que son controlados ya sea por una aplicación o a su vez por una página web.

En el mercado actual se tiene dispensadores que varían en tamaño, funcionalidad, materiales de construcción y costo, algunos son distribuidos a nivel nacional y otros de manera internacional. Tomando en cuenta los datos recolectados, se desarrolla un dispensador que satisfaga tanto en funcionalidad como en originalidad, sin sobrepasar los costos actuales.

3.2 Desarrollo de la Propuesta

El Proyecto de investigación se basa en el desarrollo de un dispensador automático para mascotas que pueda ser adaptado a las diferentes situaciones climáticas, por lo cual se desarrolla un case en acero inoxidable, debido a que este material es resistente a los diferentes climas, no se corroe y además es adecuado al usarlo con alimentos, puesto que no cambia las propiedades de los mismos.

El sistema presenta dos etapas una de control y otra de monitoreo. La etapa de control se desarrolla en base a la activación del proyecto desde una pagina web o una aplicación celular. Para que estas interfaces puedan ser accionadas remotamente se cuenta con una base de datos en la nube, que facilita la adquisición de las variables a

controlar. La etapa de monitoreo se llevará en base a la utilización de una cámara ip, esta permite vigilar las acciones de la mascota a la hora de cada comida.

La funcionalidad del sistema es tomar en cuenta las porciones adecuadas de comida para cada mascota, esto ayuda a tenerlas saludables y bien alimentadas previniendo posibles enfermedades. Además, en caso de pérdida de energía del sistema emplea el uso de una fuente de respaldo debido a que el dispensador tiene una capacidad de alimento de más de 15 días.

3.2.1 Análisis de factibilidad

El estudio de factibilidad del sistema, permite obtener información relevante para el desarrollo del proyecto, para lo cual se aplican tres estudios de factibilidad los cuales son detallados a continuación.

- **Factibilidad Técnica**

  El proyecto presenta una factibilidad técnica, debido a que se posee los conocimientos necesarios para su desarrollo, además se cuenta con la tecnología indicada, ya que ha sido utilizada con anterioridad en diferentes proyectos, por otra parte, se dispone de facilidad para obtener las herramientas y materiales a ser usadas ya sea de manera nacional o internacional.

- **Factibilidad Económica**

  El proyecto presenta una factibilidad económica, gracias al uso de hardware y software libre no se necesita realizar el pago de ningún tipo de licencia, además los materiales para la construcción del proyecto fuerón completamente financiados por el investigador.

- **Factibilidad Bibliográfica**

  El proyecto presenta una factibilidad Bibliográfica para su desarrollo, al contar con información referencial documentada en revistas, libros, tesis, páginas web, además se cuenta con facilidad de acceso a videos referentes a la tecnología utilizada.

3.2.2 Análisis de parámetros técnicos

- Controlador del sistema

Para la selección del controlador el primer dato a tomar en cuenta es la accesibilidad que se tiene en el mercado actual, razón por la cual se tomó en consideración las tarjetas de Arduino, NODEMCU, Rasberry PI dentro de lo cual los principales parámetros técnicos a usarse son las salidas análogas y digitales, fuente de alimentación, tecnología inalámbrica entre otras, estos datos serán detallados en la **Tabla 5**.

Tabla 5 comparación de parámetros del controlador

Parámetros	**Controlador**		
	NODEMCU 	**ARDUINO UNO** 	**RASBERRY ZERO** 
Pines I/O	16 pines (11 digital-1 análogo)	22 pines (14 digital – 8 análogo)	40 pines (28 GPIO – 12 de poder)
Corriente en los pines	12mA	40mA	50mA
Fuente de alimentación	5v	5v	5v
Puerto pwm	Si	Si	Si
Conexión inalámbrica	Si	No	Si
Costo	$8	$9	$30

Elaborado por: El investigador

Después de realizar la comparación entre los tres posibles controladores, se apresian sus respectivas similitudes, el primero en descartarse es Arduino Uno, debido a que no cuenta con un módulo incorporado que permita la comunicación inalámbrica, en cuanto a rasberry zero a pesar de cumplir con los requerimientos de nuestro proyecto su costo es un poco elevado, por lo tanto el controlador que más se ajusta a nuestras necesidades es la placa NODEMCU, la cual es seleccionada para este proyecto, sus características se detallan de una mejor manera en el **Anexo 1**.

- Sensores y Actuadores

Los sensores y actuadores son los que se encargan tanto de la detección como de control del paso de alimentos en el dispensador, para lo cual se realiza una comparación entre los actuadores que se encuentran en el mercado actual a nivel nacional como se detalla en la **Tabla 6.**

Tabla 6 parámetros de servomotores

Modelo	**Servomotor**	**Torque**	**Voltaje**	**Costo**
SG - 90		1 – 1.6 Kg/cm	3 – 7.2 v	$ 3
SG – 5010		3.5 – 6.5 Kg/cm	4.8 – 6 v	$ 10
MG – 995		8.5 – 10 Kg/cm	4.8 – 7.2 v	$ 14

RDS – 3115		13.5 -15 Kg/cm	5 – 7.2 v	$ 25

Elaborado por: El investigador

Se selecciona el servomotor **MG 995,** debido a que torque con el que cuenta es suficiente para la activación del paso de alimentos, descartando los servomotores SG ya que no tiene el suficiente torque acorde al peso que se debe controlar, mientras que el servomotor RDS se descartó debido al costo que presenta.

En lo que se refiere al accionamiento del dispensador de agua, se decide usar la electroválvula solenoide de arduino mostrada en la **Figura 5** debido a que no se maneja una gran presión de agua y a diferencia de las electroválvulas industriales esta tiene el costo más accesible

Figura 5 Electrovalvula selenoide

- Fuente de respaldo

En el mercado actual se cuenta con gran variedad de fuentes de respaldo, las cuales son de gran ayuda en proyectos que son alimentados a base de energía eléctrica, al presentarse un corte de energía se encargan de permitir que el sistema siga su funcionamiento por lo cual se ha tomado en cuenta algunas de las fuentes detalladas en la **Tabla 7.**

Tabla 7 Parámetros de fuentes de respaldo

Parámetros	**Fuentes**		
			
Voltaje de entrada	110-220 v	110-220 v	110-220 v
Voltaje de salida	12 v	12 v	12 v
Amperaje	1.5 amp	2 amp	3 amp
Costo	$ 25	$ 34	$ 40

Elaborado por: El investigador

Para el presente proyecto, se usa la fuente de 1.5amp por el factor de costos y en base a las conexiones se obtiene el amperaje deseado.

- Cámaras

Las cámaras ip son una de las mejores herramientas para el monitoreo de actividades. En la **Tabla 8** se presentan algunas opciones de cámaras disponibles en el mercado actual.

Tabla 8 Parámetros de cámaras ip

Parámetros	**Cámaras IP**		
	Camara ip Ezviz	Camara ip Ezviz	Camara IP Tubo
			
Visión nocturna	No	Si	No

Voltaje	12v	12v	12v
Almacenamiento	256gb	64gb	No
Audio bidireccional	Si	Si	No
Resolución	Hd	Hd	Hd
Costo	$ 44	$ 65	$ 89

Elaborado por: El investigador

Debido a las características presentes, se selecciona la cámara Ezviz con visión nocturna, esta permite captar los momentos en que las mascotas se alimentan, además se puede interactuar con ellas a través del audio bidireccional que presenta.

- **Base de Datos**

La determinación de la base de datos es acorde a la experiencia del investigador seleccionando a Firebase, esta plataforma es de dominio de google y se la puede usar para el desarrollo de proyectos ya sea comerciales y no comerciales, a su vez la plataforma nos ofrece un servicio de manera gratuita, con un límite de uso de 1Gb en cuanto a almacenamiento en la nube, cantidad más que suficiente para el desarrollo del proyecto de investigación, por otra parte para usuarios que pretendan desarrollar proyectos con almacenamiento de datos que superen dicha capacidad, Firebase cuenta con planes de pago ya sea por un límite de capacidad o a su vez por consumo de datos dicha información de costo y servicio se encuentra con mayor detalle en el **Anexo 2**.

3.2.3 Sistema electrónico automático de alimentación para mascotas

Este sistema se desarrolla con el fin de proveer alimento tanto a perros y gatos, teniendo en cuenta las porciones de comida acorde al tamaño y edad de cada uno de ellos. Otro tipo de funcionalidad del proyecto, es que el usuario este tranquilo en caso de no encontrarse en su hogar, ya que el dispensador tiene una capacidad de 20kg permitiendo proporcionar alimento por más de 15 días, dependiendo la raza de la mascota. Por otra parte, está conectado a una toma de agua permitiendo su flujo constante, de esta forma nos aseguramos que las mascotas tengan tanto comida como agua de forma continua.

En la **Figura N°6** se presenta el diagrama esquemático del proyecto de investigación en el cual se aprecia cada una de las etapas a desarrollar y la forma en que interactuaran entre sí.

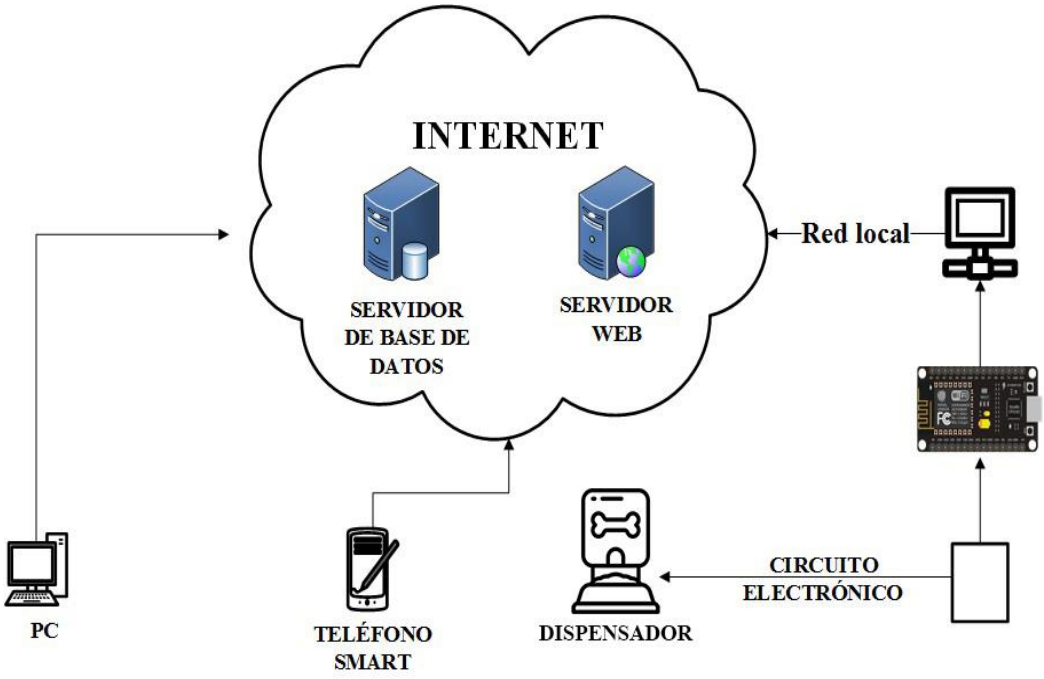


Figura 6 Esquema del Sistema electrónico automático de alimentación para mascotas

Elaborado por: El investigador

El proyecto consta de cinco faces, las cuales se aprecian en la **Figura N°7** que se las representa a través de un diagrama de bloques el respectivo funcionamiento del sistema de control.

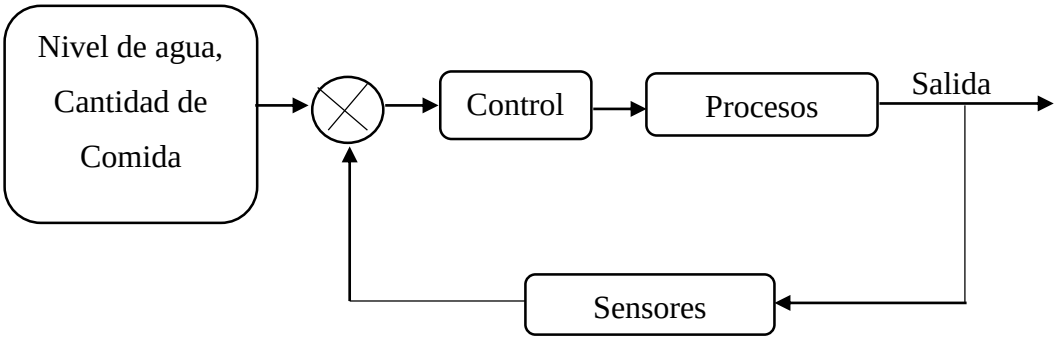


Figura 7 Diagrama de bloques del proyecto

Elaborado por: El investigador

- Sensorización

Para en control del agua se toma datos por parte de un sensor ultrasónico, el cual se encarga de tomar mediciones de la cantidad de agua que se encuentra en el recipiente, estas mediciones se realizan cada hora, con el fin de mantener el nivel indicado y que no exista carencia de ella.

- Desarrollo del sistema de control

Esta etapa se encarga de controlar prácticamente todas las funciones del sistema, tanto para comida como para agua, y de igual forma enlazar las diferentes etapas desarrolladas, utilizando la placa NODEMCU, cuyo lenguaje de programación se desarrolla en el software de Arduino.

Una vez pasada la etapa de sonorización del agua, estos datos son comprados en un rango de 0.5 a 19 cm de la capacidad del contenedor, si se encuentra por debajo de los 17 cm el controlador activa la electroválvula, la cual permita el llenado del recipiente hasta llegar a los 16 cm, esta acción se realiza de manera continua al paso de una hora, tiempo en el cual el sensor vuelve a tomar datos por dos minutos.

Para la etapa de la comida, los horarios de activación y los datos de almacenamiento de croquetas son enviados desde las interfaces desarrolladas tanto en Android Studio como en Html5 a través de la plataforma Firebase, ya que estos datos se manejan acorde a las decisiones del usuario y el tipo de mascota que tenga. Una vez precargado los datos el controlador verifica el horario y activa el llenado de la comida en el recipiente por medio de un servomotor, el que facilita el paso de la misma, este proceso se realiza de manera continua hasta que el dispensador presente una un nivel de comida menor a 2kg.

El desarrollo del código de programación del controlador se lo realiza en el programa arduino el cual se lo puede observar en el **Anexo 3**.

- Desarrollo de la interfaz celular

Para el desarrollo de la interfaz, se utiliza el software de Android Studio cuya licencia es gratuita, esta permite generar una apk para cualquier tipo de celular que esté basado en el sistema operativo Android, permitiendo la compatibilidad con todas versiones. El sistema automático de alimentación para mascotas consta con una interfaz celular basada en cinco Activitys su funcionamiento se se lo esquematiza en el diagrama de Activitys UML representado en la **Figura N° 8** y su respectiva programación se observa en el **Anexo 4**

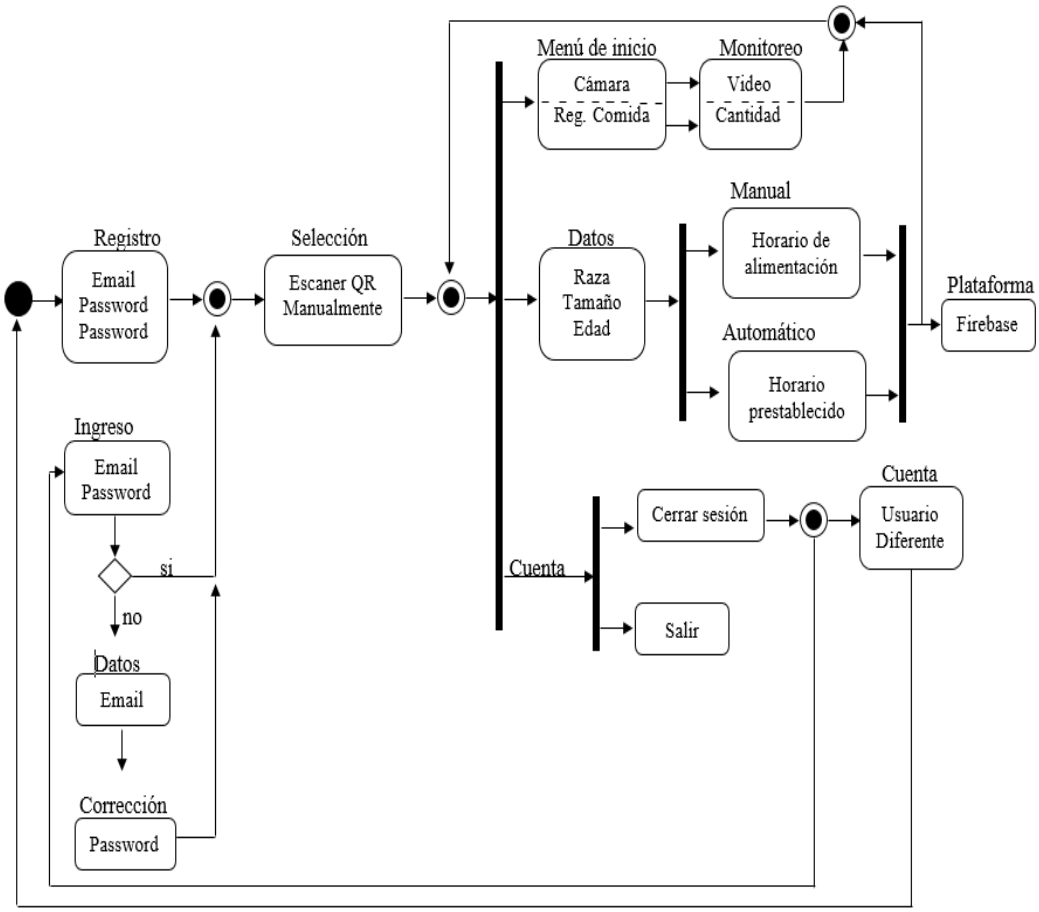


Figura 8 Diagrama de Activitys UML de la interfaz celular

Elaborado por: El investigador

Activity 1: La primera Activity muestra el ingreso del usuario a la aplicación, la cual consta de tres diferentes acciones.

- La primera se basa en el registro del usuario por medio del ingreso de datos personales a la aplicación, estos son almacenados en Firebase permitiendo acceder posteriormente al funcionamiento de la aplicación. Generando una pantalla la cual se aprecia en la **Figura N° 9.**

Figura 9 Registro de usuario

Elaborado por: El investigador

- La segunda es ingresar los datos de registro precargados anteriormente que serán comprobados en la base de datos de Firebase para su acceso, generando una pantalla que se muestra en la **Figura N° 10.**

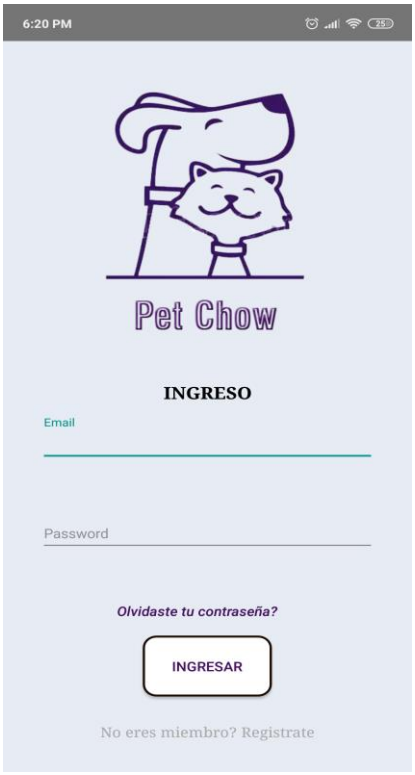


Figura 10 Acceso de usuario

Elaborado por: El investigador

- La tercera pantalla se ejecuta en caso de perder u olvidar la contraseña en esta se comprueba el correo ingresado en la base de datos, la misma que se encarga de enviar un correo electrónico con el cual podrá restablecer su respectiva clave de seguridad y en caso de ser un correo erróneo genera el mensaje respectivo obteniendo como resultado final la pantalla que se muestra en la Figura **N° 11.**

Figura 11 Recuperación de clave

Elaborado por: El investigador

Activity 2: La segunda Activity está diseñada para el registro del dispensador a utilizar esta consta de un acceso de forma automática al escanear un código de barras QR o de manera manual ingresando el número de serie del Dispensador. El funcionamiento de este layout se lo puede apreciar en la **Figura N° 12** una vez finalizado el registro se permite el acceso a la etapa de control o a su vez poder borrar el registro del mismo.

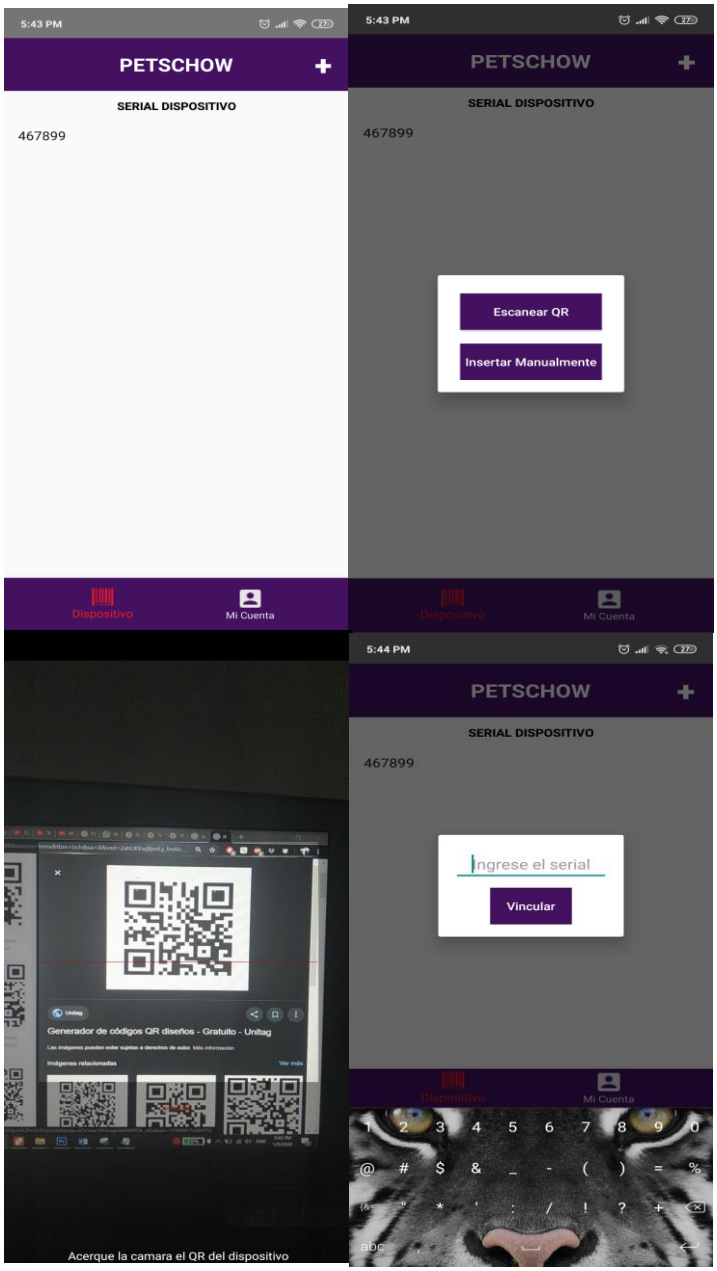


Figura 12 Registro de Dispensador
Elaborado por: El investigador

Activity 3: La tercera Activity está programada para realizar dos funciones: acceso a la cámara y registro de comida como se muestra en la **Figura N° 13**.

Figura 13 Activity 3

Elaborado por: El investigador

- La función de monitoreo se realiza por medio de la cámara ip como se muestra en la **Figura N° 14,** generado a través de una interfaz propia de la cámara, en esta se tiene un control de acceso que al ser registrados podremos dirigirnos hacia la pantalla de interacción con la cámara ip, permitiendo transmitir audio de forma bidireccional y observar las imágenes de video captado en tiempo real.

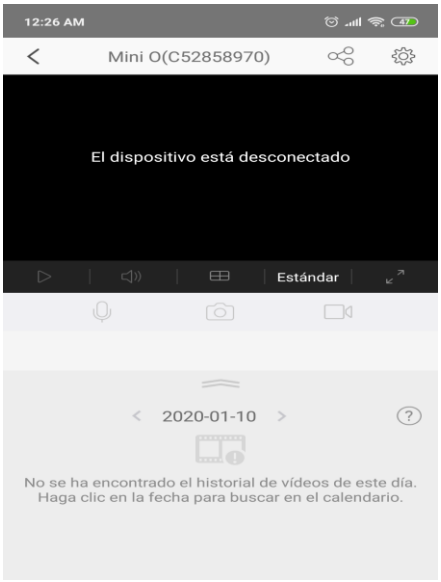


Figura 14 Interfaz de cámara ip

Elaborado por: El investigador

- Para el registro se tiene una sub pantalla como se muestra en la **Figura N° 15,** esta requiere información acerca de la cantidad de la comida provista por el usuario al dispensador, cuyos datos son almacenados en la plataforma de Firebase permitiendo que el usuario pueda visualizarlos.

Figura 15 Registro de comida

Elaborado por: El investigador

Activity 4: En esta pantalla se encuentra el ingreso de datos de las mascotas acorde a tipo, razas y edad, una vez ingresado los parámetros son almacenados en la base de datos, a continuación, podremos seleccionar el horario de alimentación ya sea de forma manual como de manera automática. Al ser ingresado de manera manual se despliega una sub pantalla para el ingreso de los horarios de comida y al ser seleccionado automático se despliega otra sub pantalla con la visualización de los horarios precargados en el sistema. La interfaz se puede observar en la **Figura N° 16.**

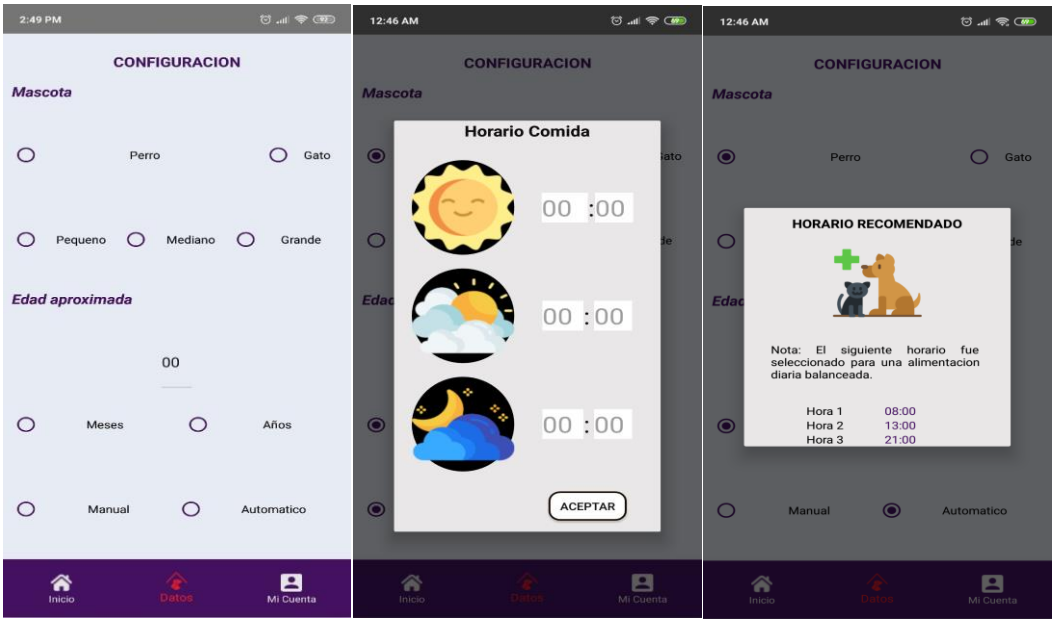


Figura 16 Selección de mascotas y horarios

Elaborado por: El investigador

Activity 5: Finalmente en esta pantalla se encuentra la opción de cerrar sesión o a su vez salir de la aplicación, su interfaz se muestra en la **Figura N° 17.**

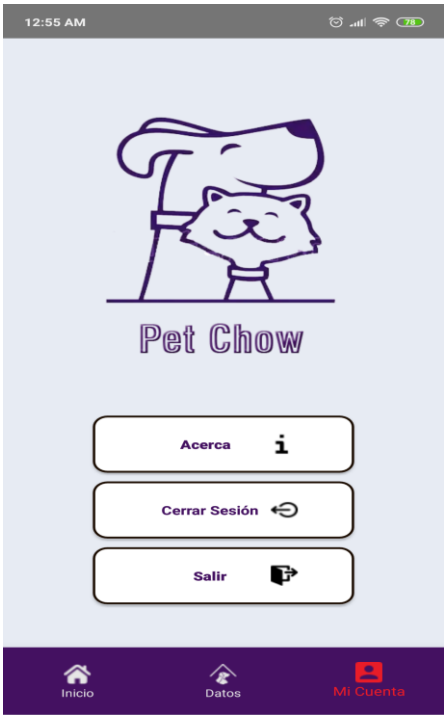


Figura 17 Cierre de sesión

Elaborado por: El investigador

- Desarrollo de la interfaz web

La interfaz web desarrollada consta de dos pantallas cuyo código de programación se observa en el **Anexo 5.**

- La primera pantalla consta del registro de usuarios cuyos datos serán almacenados en la plataforma de Firebase como se observa en la **Figura N° 18**. Para poder ingresar hacia a la etapa de control de cada uno de los dispositivos se deberá registrar previamente desde la aplicación Móvil.

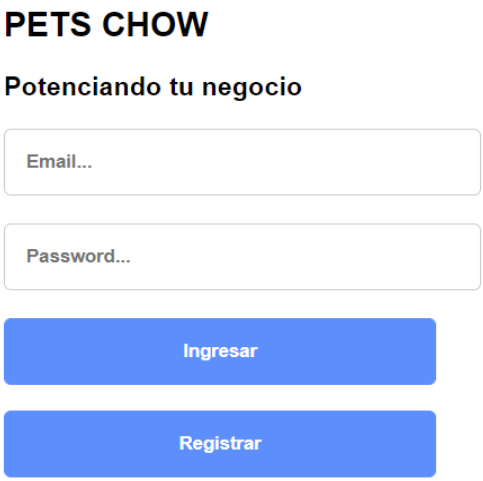


Figura 18 Pantalla de acceso

Elaborado por: El investigador

- La segunda pantalla consta de la parte de control del dispositivo en la cual al igual que en la aplicación móvil se selecciona el tipo de mascota, la raza, edad, porciones y los respectivos horarios de alimentación ya sea de forma manual o de manera automática, en caso de seleccionar la segunda opción se puede visualizar los horarios en una pantalla de notificaciones. Esta interfaz la podemos observar en la **Figura N° 19.**

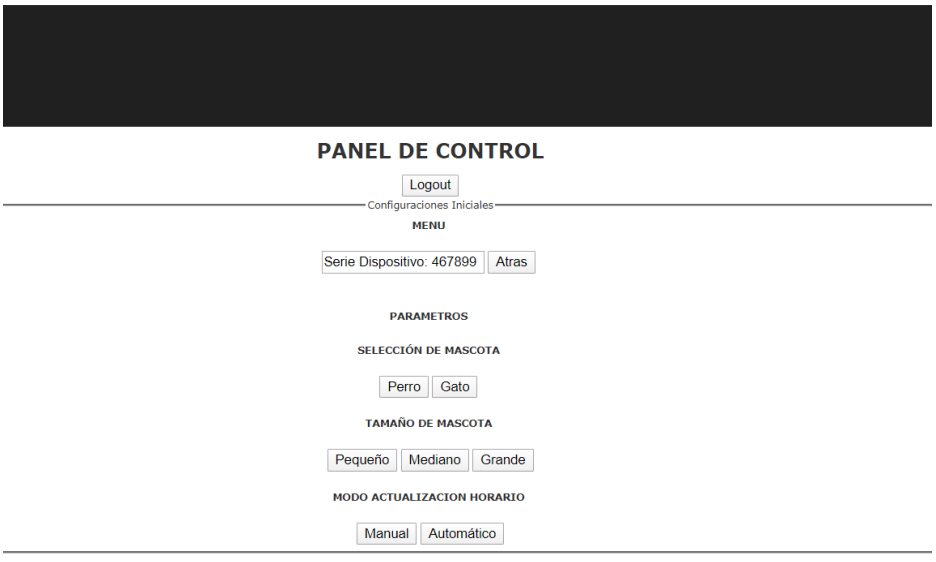


Figura 19 Pantalla de control

Elaborado por: El investigador

- Base de datos

La plataforma para el desarrollo de las interfaces de control seleccionada fue Firebase, debido a que cuenta con opción de almacenaje gratuito para el funcionamiento del proyecto, esta base contiene tres partes esenciales: Autenticación, base de datos y hosting como se puede ver en la **Figura N° 19.**

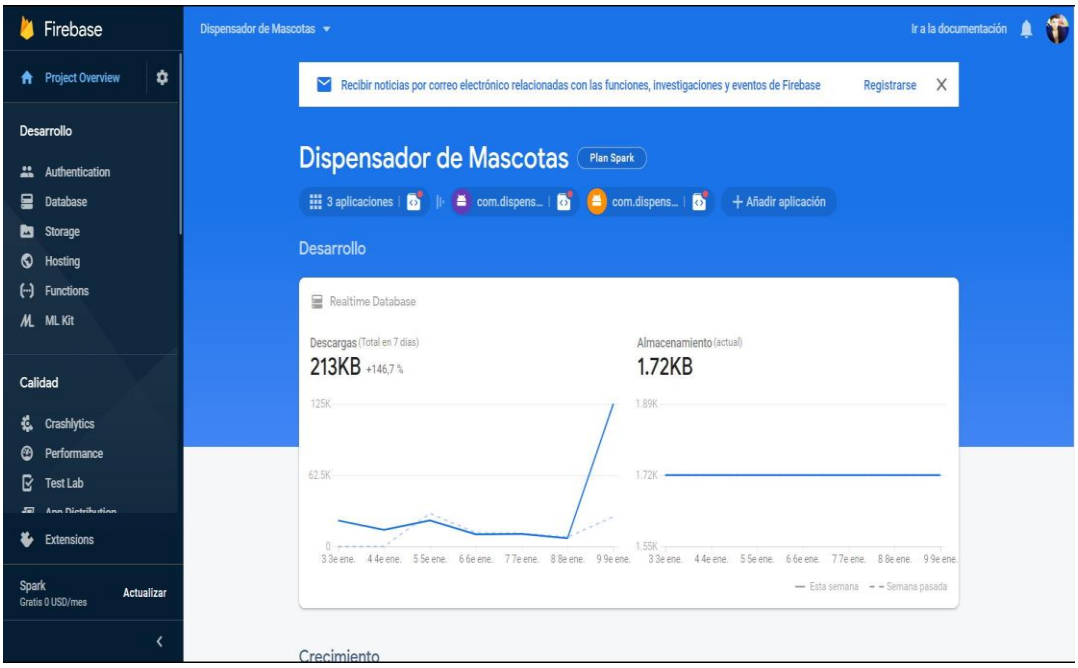


Figura 20 Plataforma de google

Elaborado por: El investigador

- En la parte de autenticación se almacenan todas las cuentas que estén registradas desde las diferentes interfaces en base al proyecto desarrollado esto se puede apreciar en la **Figura N° 21.**

Figura 21 Registro de usuarios

Elaborado por: El investigador

- La segunda etapa corresponde a la base de datos en la cual fue desarrollado el almacenamiento de las variables ocupadas en el proyecto, empezando por la creación de un nodo padre el cual contiene en primer lugar el correo del usuario registrado junto con su serial necesario para su conexión con la aplicación móvil y segundo el Id perteneciente al dispensador el cual almacenara todas las variables de control enviadas desde la aplicación móvil, estas variables a su vez serán enviadas al controlador del dispensador para su adecuado funcionamiento. Estos datos se muestran en la **Figura N° 22.**

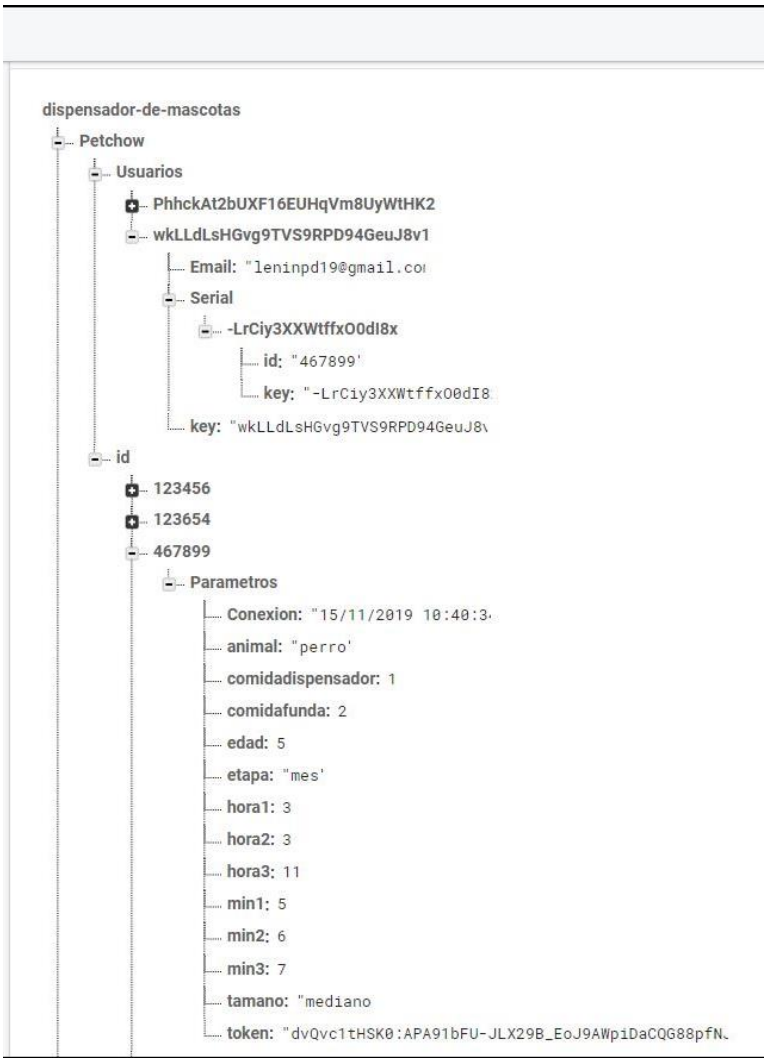


Figura 22 Base de datos de Firebase

Elaborado por: El investigador

- Finalmente se tiene el Hosting, el cual es un espacio en la plataforma de desarrollo que permite el almacenamiento de la interfaz web para su acceso a la red este se observa en la **Figura N° 23.**

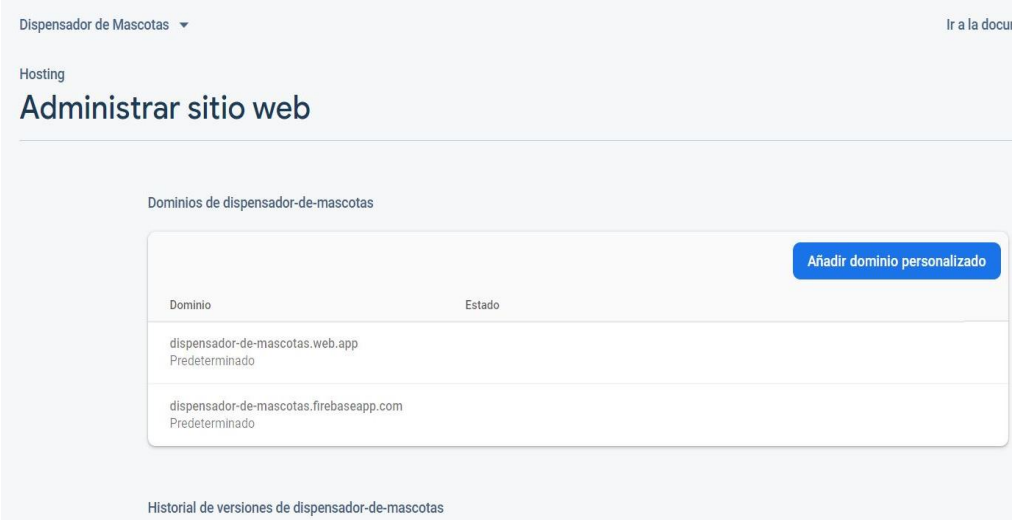


Figura 23 Hosting del Proyecto

Elaborado por: El investigador

3.2.4 Diseño de la placa del sistema de control

La placa que se usa en el sistema de control consta de los siguientes materiales:

- 1 Nodemcu
- 4 Diodos zener
- 1 7812
- 1 7805
- 2 capacitores cerámicos 104
- 2 capacitores de 2200uf
- 2 resistencias de 330Ω
- 1 transistor 3904
- 1 sensor ultrasónico

En la **Figura N° 24** se muestra el diseño de la placa con sus respectivas conexiones. Debido a que algunos materiales se los conecta externamente no se los ha incluido en el diseño virtual.

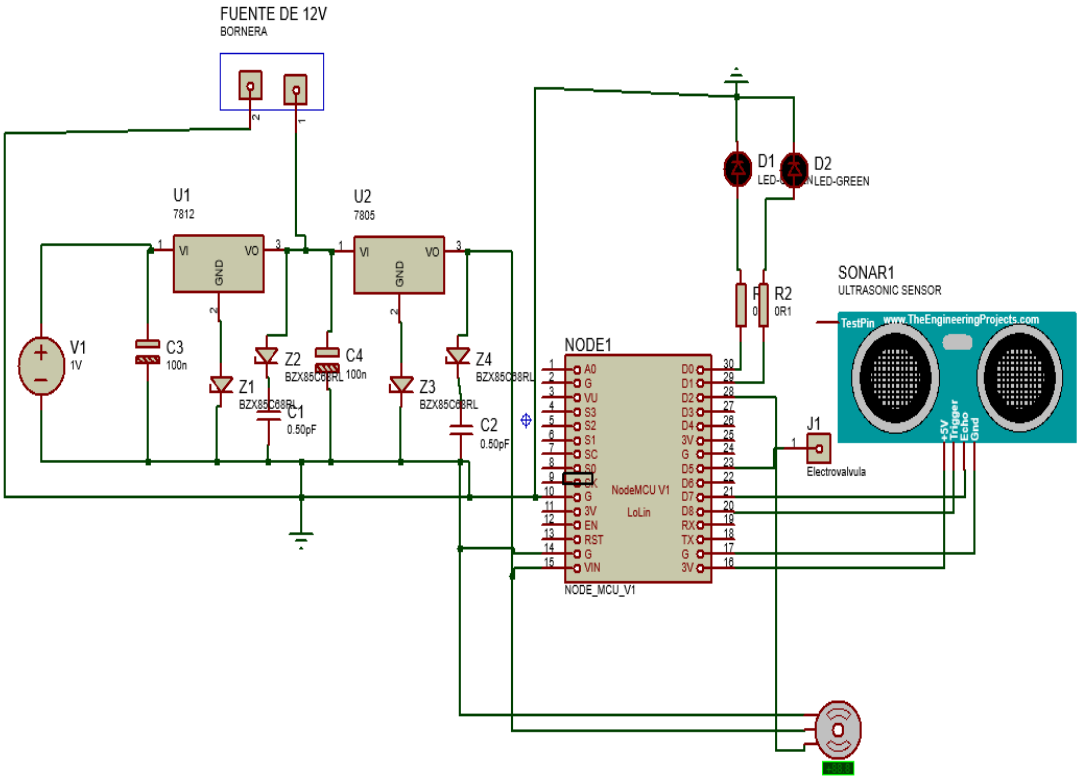


Figura 24 Diseño de la placa de control

Elaborado por: El investigador

En la **Figura N° 25** se observa el diseño del circuito impreso el cual se lo realizó en el programa de Eagle.

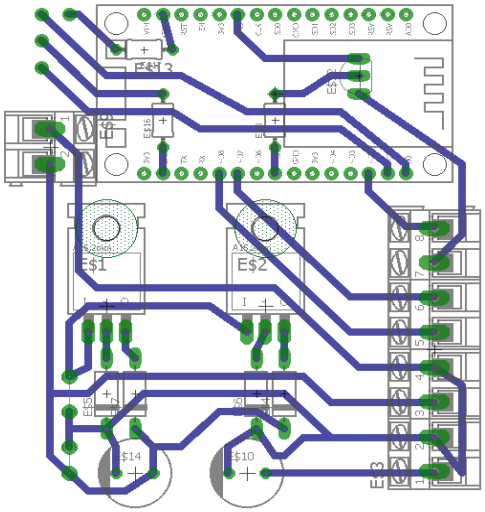


Figura 25 Diseño impreso de la placa

Elaborado por: El investigador

La **Figura N° 26** muestra la implementación del circuito impreso obteniendo el sistema de control del proyecto.

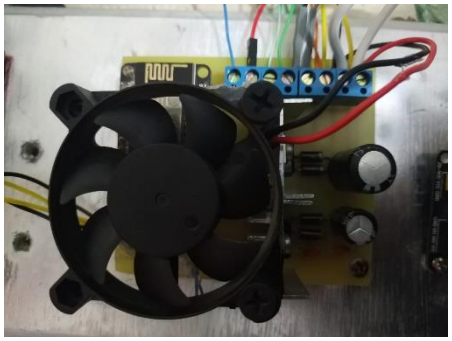


Figura 26 Placa de control del sistema
Elaborado por: El investigador

En la **Figura N° 27** se muestra las conexiones finales del circuito de control a la fuente de alimentación ubicada en la parte posterior al case del dispensador.

Figura 27 Conexiones del Sistema
Elaborado por: El investigador

3.2.5 Presupuesto

El presupuesto en cuanto a la construcción del sistema electrónico automático de alimentación para mascotas está compuesto por componentes electrónicos, además de los materiales usados para su fabricación, los mismos que se detallan en la **Tabla 9.**

Tabla 9 Costo del diseño del proyecto

TABLA DE COSTOS DEL DISPENSADOR					
Ítem	**Descripción**	**Unidad**	**Cantidad**	**Valor unitario**	**Valor total**
1	Acero inoxidable	Planchas 1.22 x 2,44 m	2	$120	$240
2	Bandejas de acero inoxidable	c/u	2	$16	$32
3	Sensor ultrasónico	c/u	1	$3.50	$3.50
4	Servomotor Mg995	c/u	1	$14	$14
5	Electroválvula	c/u	1	$11	$11
6	NODEMCU	c/u	1	$10	$10
7	Modulo relé	c/u	1	$5	$5
8	Cámara ip	c/u	1	$65	$65
9	Bateria + Fuente + regulador	kit	1	$60	$60
10	Placa	c/u	1	$20	$20
1	Doblajes de acero inoxidable				$100
TOTAL					**$560.5**

Elaborado por: El investigador

Estos costos están diseñados tomando en cuenta sus valores por unidad, los cuales fueron tomados en tiendas electrónicas a nivel nacional y para la fabricación de un solo dispensador obteniendo un total de $560.5

3.2.6 Costo del diseño

Para el cálculo del costo de diseño se toma en cuenta las horas requeridas para el desarrollo del proyecto, además el salario básico unificado para el año 2020 en Ecuador, el cual está en $400 según acuerdo ministerial Nro.MDT-2019-394 del 27 de diciembre del 2019 y finalmente los salarios de Ingeniero en Electrónica cuyo promedio es de $915 detallados en multitrabajos proporciona un promedio de $650.

Una vez obtenido un sueldo promedio se utiliza la siguiente ecuación para el cálculo de salario por hora de trabajo, el cual está basada en 21 días laborables por mes y 8 horas diarias de trabajo.

El diseño del proyecto se estima en 80 horas por lo tanto se tiene lo siguiente:

Generándonos finalmente un costo unitario del proyecto en $869.50
Un factor a tomar en cuenta es que el precio del proyecto bajaría en caso de ser comercializado, ya que los materiales al por mayor e importados bajan sus costos, de igual manera se considera que los materiales usados pueden ser remplazado por marcas con de menor valor comercial, sin afectar la funcionalidad del mismo, haciéndolo competitivo con los dispensadores que son exportados en otros países con menos funcionalidades y menor capacidad de almacenamiento de alimentos.

3.2.7 Esquema del dispensador

Debido a que el dispensador debe contener 20 kg de comida para perros o gatos, el case es de gran tamaño y de peso considerable, debido al material de construcción. Sus dimensiones se las aprecia en la **Figura N° 28 y Figura N° 29,** por otra parte, en la la **Figura N° 30** se observa el case del dispensador una vesz terminada la construcción.

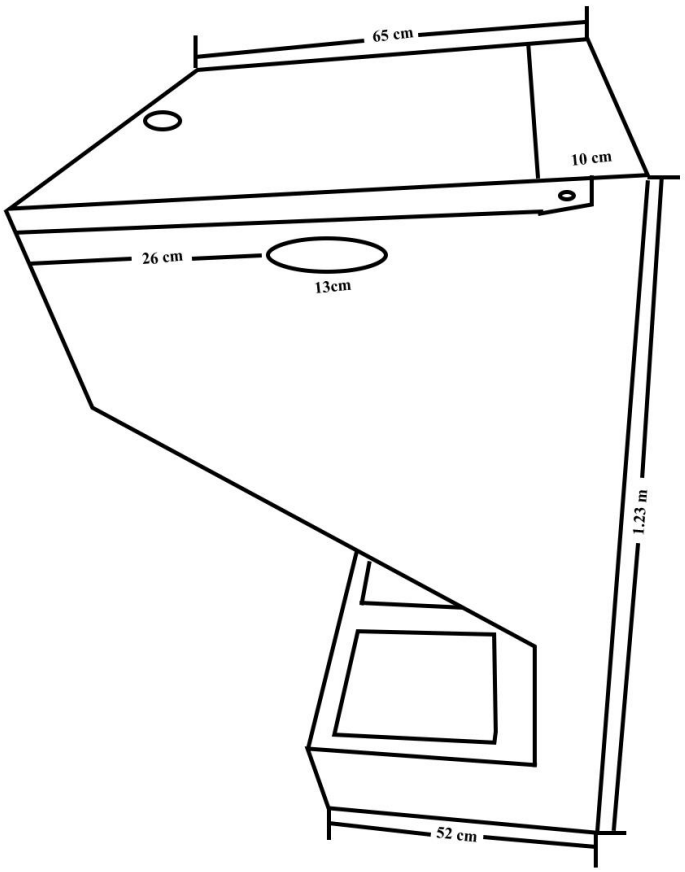


Figura 28 Vista lateral del Dispensador

Elaborado por: El investigador

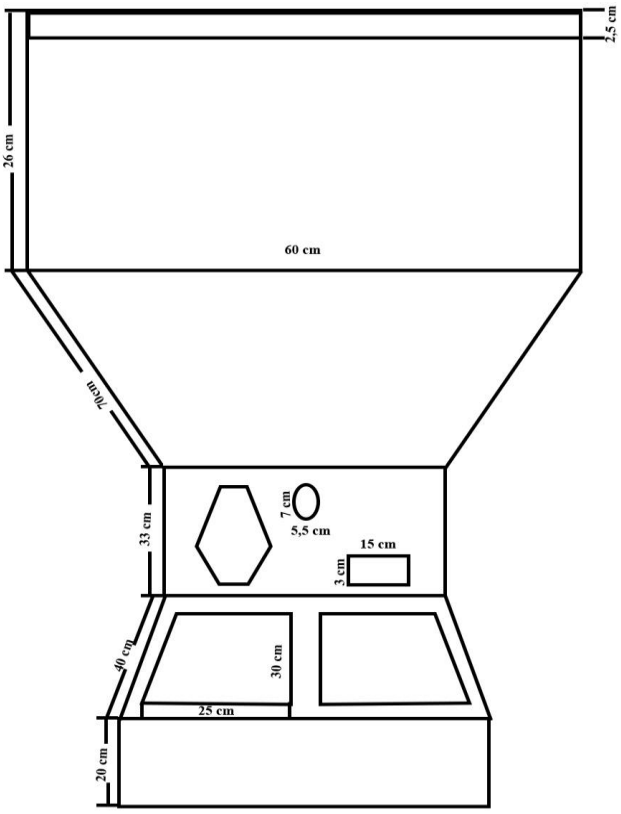


Figura 29 Vista frontal del dispensador

Elaborado por: El investigador

Figura 30 Dispensador de comida para mascotas

Fuente: El investigador

En el **Anexo 6** se muestra un manual de control del dispensador el cual contiene los pasos básicos para ponerlo en funcionamiento.

3.2.8 Pruebas de funcionamiento del dispensador

Las porciones de comida que se proporciona a cada mascota van acorte a su raza y edad en la **Tabla 10** se observa las porciones correspondientes para perros, y en la **Tabla 11** para gatos, estas se rigen en una dieta balanceada establecida en la página de alimentos de PURINA, las porciones serán activas entre dos y tres veces por día ya sea por selección manual o automática acorde a horarios programados.

Tabla 10 Horarios de alimentación automática para perros

Raza	1 – 6 meses 3 / día	7 – 12 meses 2 / día
Pequeña	90gr	140gr
Mediana	250gr	350gr
Grande	380gr	500gr

Elaborado por: El investigador

Tabla 11 Horarios de alimentación automática para gatos

Raza	1 – 6 meses 3 / día	7 – 12 meses 2 / día
Pequeña	40gr	60gr
Grande	60gr	90gr

Elaborado por: El investigador

En cuanto a pruebas de funcionamiento se realizó pruebas en tres diferentes hogares cada uno con un tipo de perro y en una casa diferente con un gato, dando un total de un mes de uso esto se aprecia en la **Figura N° 31**

Figura 31 Pruebas de funcionamiento con mascotas
Fuente: El investigador

En las pruebas realizadas se encuentra que el dispensador tiene un error respecto al dosificador del 5% de margen de apertura, cantidad que no afecta en cuanto a una dieta saludable para las mascotas.

CAPÍTULO IV

CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- El dispensador esta configurado para brindar las cantidades necesarias de alimento ya sea para perros o gatos, teniendo un margen de error del 5 % en el dosificador de alimentos. Al seleccionar la raza y edad de las mascotas permite el uso de cualquier marca de croquetas presente en el mercado nacional.

- El dispensador cuenta con la función de configurar los horarios de activación de la comida y agua, además brinda la opción de monitorear a las mascotas a través de una cámara ip, ya que cuenta con acceso remoto desde la web o aplicación celular.

- El sistema utiliza una API proporcionada por la plataforma de Firebase para facilitar el almacenamiento en su base de datos, al contar con un protocolo SSE permite las conexiones de HTTP, permitiendo enviar notificaciones cuando el dispensador de comida tenga una cantidad menor a 2 kg.

- El sistema cuenta con una fuente de respaldo, alimentado por una batería de gel de 12v a 4amp, de manera que cuando exista un corte de energía eléctrica el dispensador mantenga su funcionamiento autónomo por un periodo de tres días.

4.2 Recomendaciones

- Es recomendable que para los usuarios que no utilicen el Sistema de alimentación para mascotas fuera de sus hogares cambien el tipo de material de construcción lo cual reduciría en un gran porcentaje de su costo final sin perder su funcionalidad.

- Es recomendable cambiar el sistema del dosificador por uno de tornillo sin fin lo cual generaría menos esfuerzo por parte del motor y mejor control de tiempos de activación.

- Se recomienda desarrollar nuevos sistemas con diferentes capacidades de almacenamiento de comida lo que permite que el usuario tenga opciones a elegir acorde a sus necesidades y al tipo de raza de mascota que tenga.

- Es recomendable implementar un sistema de drenaje con respecto al control del agua, permitiendo que esta cambie con mayor constancia, siendo más gratificante para las mascotas.

BIBLIOGRAFÍA

[1] K. Karyono, "Smart dog feeder design using wireless communication, MQTT and Android client," *IEEE*, pp. 2-7, 2016.

[2] D. G. GARCIA, *DISPENSADOR MASCOTAS CLUB CONTROLADO REMOTAMENTE DESDE*, Bogota: FUNDACIÓN UNIVERSITARIA LOS LIBERTADORES, 2017.

[3] C. J. C. PEREZ, *INVESTIGACION DE PRE FACTIBILIDAD PARA LA FABRICACION DE DISPENSADOR DE COMIDA PARA MASCOTAS AUTOMATIZADO*, Lima: UNIVERSIDAD SAN IGNACIO DE LOYOLA, 2018.

[4] I. Daniel, «Dispensador automático de alimento para mascotas,» *FIGEMPA*, vol. 2, nº 2, pp. 2-8, 2017.

[5] Benítez, «3 de cada 5 familias tienen una mascota,» El Telégrafo , [En línea]. Available: https://www.eltelegrafo.com.ec/noticias/quito/1/3-de-cada-5-familias-tienen-una-mascota.. [Último acceso: 10 5 1029].

[6] «Perros malnutridos: cuatro consecuencias para su salud | EROSKI CONSUMER,» EROSKI CONSUMER, 14 11 2013. [En línea]. Available: http://www.consumer.es/web/es/mascotas/perros/alimentacion/2013/11/14/218363.php..

[7] «Desnutricion En Perros. Plan De 5 Pasos Para Recuperar A Un Perro Con Problemas De Desnutrición,» Mascotafiel, 2019. [En línea]. Available: https://mascotafiel.com/desnutricion-en-perros/..

[8] «Electrónica Digital,» Rocio Leira Rodrígues, [En línea]. Available: https://www.edu.xunta.gal/centros/cafi/aulavirtual2/pluginfile.php/39495/mod_resource/content/2/analo%CC%81gica.pdf.

[9] Obessemer, 2019. [En línea]. Available: http://www.bessemer.mx/acero-en-mexico/5-diferencias-entre-el-aluminio-y-el-acero-inoxidable/. [Último acceso: 12 12 2019].

[10] J. Crespo, «Aprendiendo Arduino,» 18 12 2016. [En línea]. Available: https://aprendiendoarduino.wordpress.com/2016/12/18/sensores-y-actuadores/. [Último acceso: 12 12 2019].

[11] J. M. Zafrilla, «PCEInst.,» PCE, [En línea]. Available: https://www.pce-iberica.es/instrumentos-de-medida/sistemas/controladores-digitales.htm. [Último acceso: 12 12 2019].

[12] A. &. C. Ingenieros, «AC & CC Ingenieros,» 22 9 2018. [En línea]. Available: https://www.ac-cc.com/blog/en-que-consiste-un-sistema-de-energia-de-respaldo-o-de-emergencia. [Último acceso: 12 12 2019].

[13] «Internet of things (IoT) en la transformación digital de las empresas,» *Incipi*, vol. 1, pp. 4-5, 2015.

[14] «Aprendiendo Arduino,» 2019. [En línea]. Available: https://aprendiendoarduino.wordpress.com/tag/iot/. . [Último acceso: 2019 12 14].

[15] A. C. Domínguez, *Diseño e implementación de una arquitectura IoT*, Sevilla, 2016.

[16] «Aprendiendo Arduino,» 2019. [En línea]. Available: https://aprendiendoarduino.wordpress.com/2018/11/17/protocolos-iot-capa-aplicacion/. [Último acceso: 2019 12 14].

[17] «Animales exóticos,» 2016. [En línea]. Available: https://animalesexoticos.blog/mascotas-mas-comunes/. [Último acceso: 16 12 2019].

[18] G. Ibañez, «Petdarling,» Guiller Ibañez, [En línea]. Available: https://www.petdarling.com/articulos/razas-de-perros/. [Último acceso: 12 16 2019].

[19] «Purina,» [En línea]. Available: https://www.purina.es/perros/razas-de-perros/razas-de-perros-pequenos. [Último acceso: 18 12 2019].

[20] «Purina,» [En línea]. Available: https://www.purina.es/perros/razas-de-perros/razas-de-perros-medianos. [Último acceso: 18 12 2019].

[21] «Purina,» [En línea]. Available: https://www.purina.es/perros/razas-de-perros/razas-de-perros-grandes. [Último acceso: 18 12 2019].

[22] «hogarmania,» 2019. [En línea]. Available: https://www.hogarmania.com/mascotas/perros/alimentacion/201705/cantidad-comida-perros-35991.html. [Último acceso: 12 20 2019].

[23] «Purina,» [En línea]. Available: https://www.purina.es/gatos/razas-de-gatos/gatos-de-raza-pequena. [Último acceso: 18 12 2019].

[24] «Purina,» [En línea]. Available: https://www.purina.es/gatos/razas-de-gatos/gatos-grandes. [Último acceso: 18 12 2019].

[25] «Desnutrición,» Mascota fiel, 2019. [En línea]. Available: http://www.consumer.es/web/es/mascotas/perros/alimentacion/2013/11/14/218363.php. . [Último acceso: 20 12 2019].

[26] «Obesidad en mascotas,» Mascotas, 16 3 1028. [En línea]. Available: https://www.infobae.com/tendencias/mascotas/2018/03/16/obesidad-en-mascotas-mas-de-la-mitad-de-los-perros-y-gatos-domesticos-sufren-de-sobrepeso/. [Último acceso: 24 12 2019].

[27] «vets & clinics,» [En línea]. Available: https://www.affinity-petcare.com/veterinary/patologias/hiperparatiroidismo-primario. [Último acceso: 24 12 2019].

[28] D. K. Becker, «Healtly pets,» MERCOLA, 2019. [En línea]. Available: https://mascotas.mercola.com/sitios/mascotas/archivo/2017/06/10/mascotas-con-piel-seca-escamosa.aspx. [Último acceso: 24 12 2019].

[29] «PURINA,» Pro Plan, 12 11 2018. [En línea]. Available: https://www.purina.es/proplan/consejos/diarrea-perros. [Último acceso: 24 12 2019].

[30] «Healthy pets,» Dra. Karen Becker, 16 9 2017. [En línea]. Available: https://mascotas.mercola.com/sitios/mascotas/archivo/2017/09/16/letargo-en-perros-y-gatos.aspxH. [Último acceso: 25 12 2019].

[31] «IOT DESIG PRO,» 30 9 2019. [En línea]. Available: https://iotdesignpro.com/articles/arduino-nano-vs-raspberry-pi-zero-w-vs-nodemcu-for-iot-projects. [Último acceso: 26 12 2019].

[32] «NAYLAM MECATRONICS,» 28 11 2017. [En línea]. Available: https://naylampmechatronics.com/blog/33_Tutorial-uso-de-servomotores-con-arduino-.html. [Último acceso: 26 12 2019].

[33] «Mercado Libre,» 1 12 2019. [En línea]. Available: https://electronica.mercadolibre.com.ec/seguridad-hogar/camara-ip-inalambrica. [Último acceso: 28 12 2019].

Anexos

Anexo 1: Especificaciones de la placa NODEMCU

Espressif Systems

ESP8266 Datasheet

Espressif Systems' Smart Connectivity Platform (ESCP) demonstrates sophisticated system-level features include fast sleep/wake context switching for energy-efficient VoIP, adaptive radio biasing for low-power operation, advance signal processing, and spur cancellation and radio co-existence features for common cellular, Bluetooth, DDR, LVDS, LCD interference mitigation.

1.2. Features

- 802.11 b/g/n
- Integrated low power 32-bit MCU
- Integrated 10-bit ADC
- Integrated TCP/IP protocol stack
- Integrated TR switch, balun, LNA, power amplifier and matching network
- Integrated PLL, regulators, and power management units
- Supports antenna diversity
- WiFi 2.4 GHz, support WPA/WPA2
- Support STA/AP/STA+AP operation modes
- Support Smart Link Function for both Android and iOS devices
- SDIO 2.0, (H) SPI, UART, I2C, I2S, IR Remote Control, PWM, GPIO
- STBC, 1x1 MIMO, 2x1 MIMO
- A-MPDU & A-MSDU aggregation & 0.4s guard interval
- Deep sleep power <10uA, Power down leakage current < 5uA
- Wake up and transmit packets in < 2ms
- Standby power consumption of < 1.0mW (DTIM3)
- +20 dBm output power in 802.11b mode
- Operating temperature range -40C ~ 125C
- FCC, CE, TELEC, WiFi Alliance, and SRRC certified

6	TOUT	I	ADC Pin (note: an internal pin of the chip) can be used to check the power voltage of VDD3P3 (Pin 3 and Pin4) or the input voltage of TOUT (Pin 6). These two functions cannot be used simultaneously.
7	CHIP_EN	I	Chip Enable. High: On, chip works properly; Low: Off, small current
8	XPD_DCDC	I/O	Deep-Sleep Wakeup; GPIO16
9	MTMS	I/O	GPIO14; HSPI_CLK
10	MTDI	I/O	GPIO12; HSPI_MISO
11	VDDPST	P	Digital/IO Power Supply (1.8V~3.3V)
12	MTCK	I/O	GPIO13; HSPI_MOSI; UART0_CTS
13	MTDO	I/O	GPIO15; HSPI_CS; UART0_RTS
14	GPIO2	I/O	UART Tx during flash programming; GPIO2
15	GPIO0	I/O	GPIO0; SPI_CS2
16	GPIO4	I/O	GPIO4
17	VDDPST	P	Digital/IO Power Supply (1.8V~3.3V)
18	SDIO_DATA_2	I/O	Connect to SD_D2 (Series R: 200Ω); SPIHD; HSPIHD; GPIO9
19	SDIO_DATA_3	I/O	Connect to SD_D3 (Series R: 200Ω); SPIWP; HSPIWP; GPIO10
20	SDIO_CMD	I/O	Connect to SD_CMD (Series R: 200Ω); SPI_CS0; GPIO11
21	SDIO_CLK	I/O	Connect to SD_CLK (Series R: 200Ω); SPI_CLK; GPIO6
22	SDIO_DATA_0	I/O	Connect to SD_D0 (Series R: 200Ω); SPI_MSIO; GPIO7
23	SDIO_DATA_1	I/O	Connect to SD_D1 (Series R: 200Ω); SPI_MOSI; GPIO8
24	GPIO5	I/O	GPIO5
25	U0RXD	I/O	UART Rx during flash programming; GPIO3
26	U0TXD	I/O	UART Tx during flash progamming; GPIO1; SPI_CS1
27	XTAL_OUT	I/O	Connect to crystal oscillator output, can be used to provide BT clock input
28	XTAL_IN	I/O	Connect to crystal oscillator input
29	VDDD	P	Analog Power 3.0V~3.6V
30	VDDA	P	Analog Power 3.0V~3.6V
31	RES12K	I	Serial connection with a 12 kΩ resistor and connect to the ground
32	EXT_RSTB	I	External reset signal (Low voltage level: Active)

Anexo 2: Planes de pago de la plataforma Firebase

Productos	Plan Spark Límites gratuitos para aficionados Sin cargo	Plan Flame Precios fijos para apps en expansión USD 25 por mes	Plan Blaze Calcula los precios de las apps a gran escala. Pago por uso ✓ Sin incluir el uso gratuito del plan Spark*
Pruebas A/B	Sin cargo		
Analytics	Sin cargo		
App Indexing	Sin cargo		
Authentication			
Autenticación telefónica: Canadá, EE.UU. y la India	10,000 por mes	10,000 por mes	USD 0.01 por verificación
Autenticación telefónica: Todos los demás países	10,000 por mes	10,000 por mes	USD 0.06 por verificación
Otros servicios de autenticación	✓	✓	✓
Cloud Firestore			
Datos almacenados	1 GiB en total	2.5 GiB en total	USD 0.18 por GiB
Network egress	10 GiB por mes	20 GiB por mes	Google Cloud pricing
Operaciones de escritura de documentos	20,000/día	100,000 por día	USD 0.18 por cada 100,000
Operaciones de lectura de documentos	50,000/día	250,000/día	USD 0.06 por cada 100,000
Operaciones de eliminación de documentos	20,000/día	100,000 por día	USD 0.02 por cada 100,000
Cloud Functions			
Invocaciones	125,000 por mes	2,000,000/mes	USD 0.40/millón
GB-segundo	40,000 por mes	400,000 por mes	USD 0.0025/mil
CPU-segundo	40,000 por mes	200,000 por mes	USD 0.01/mil
Redes de salida	Únicamente servicios de Google	5 GB por mes	USD 0.12/GB
Cloud Messaging (FCM)	Sin cargo		
Crashlytics	Sin cargo		
Dynamic Links	Sin cargo		
Hosting			
GB almacenados	1 GB	10 GB	USD 0.026/GB
GB transferidos	10 GB/mes	50 GB por mes	USD 0.15/GB
Dominio personalizado y SSL	✓	✓	✓
Varios sitios por proyecto	✓	✓	✓

Invites	Sin cargo		
Kit de AA			
API en el dispositivo	✓	✓	✓
Hosting o entrega de modelos personalizados	✓	✓	✓
AutoML Vision Edge Dataset	1K images/project	1K images/project	Google Cloud pricing
AutoML Vision Edge Training	3 hours/project	3 hours/project	15 hrs/billing account, $4.95/hour after
API de Cloud Vision	✗	✗	USD 1.50 por cada 1,000 (consulta los precios de Cloud Vision)
Supervisión del rendimiento	Sin cargo		
Predictions	Sin cargo		
Realtime Database			
Conexiones simultáneas	100	200k	200k/database
GB almacenados	1 GB	2.5 GB	USD 5 por GB
GB descargados	10 GB/mes	20 GB/mes	USD 1/GB
Varias bases de dato por proyecto	✗	✗	✓
Remote Config	Sin cargo		
Storage			
GB almacenados	5 GB	50 GB	USD 0.026/GB
GB descargados	1 GB/día	50 GB/día	USD 0.12/GB
Operaciones de carga	20,000/día	100,000 por día	USD 0.05 por cada 10,000
Operaciones de descarga	50,000/día	250,000/día	USD 0.004 por cada 10,000
Varios depósitos por proyecto	✗	✗	✓
Test Lab			
Pruebas en dispositivos virtuales	10 pruebas por día	10 pruebas por día	USD 1 por dispositivo por hora
Pruebas en dispositivos físicos	5 pruebas por día	5 pruebas por día	USD 5 por dispositivo por hora
Google Cloud Platform Usa BigQuery y otras funciones con modalidad IaaS	✗	✗	✓
Selecciona un plan	Plan Spark Sin cargo Comenzar ahora	Plan Flame USD 25 por mes Seleccionar plan	Plan Blaze Pago por uso Seleccionar plan

Anexo 3: Código de arduino

Este anexo contiene el código que se encuentra en la placa NODEMCU el mismo que se encarga del control total del dispositivo de forma mecánica, junto con los diagramas de flujo de llenado de agua en la **Figura N° 32** y comida en la **Figura N° 33**

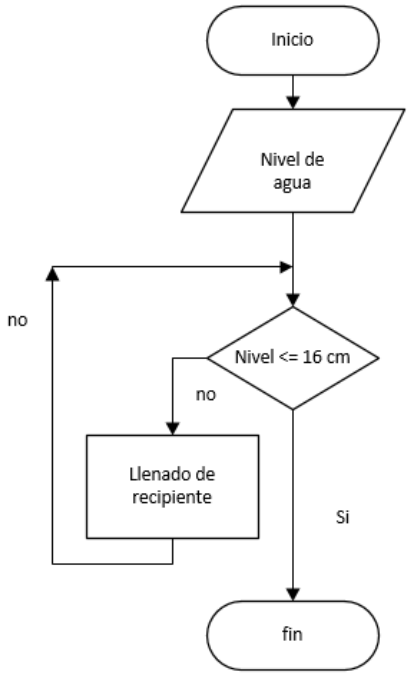


Figura 32 Digrama de flujo de llenado de agua

Elaborado por: El investigador

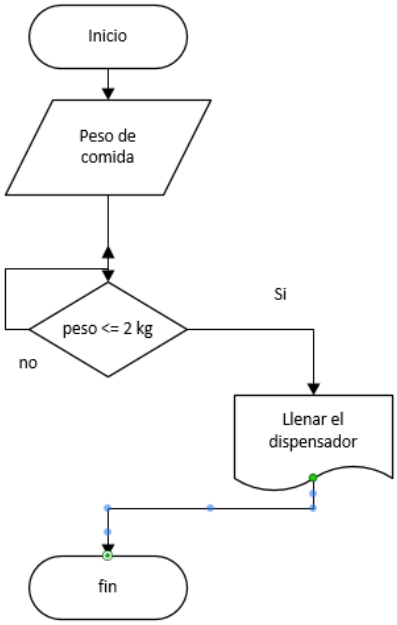


Figura 33 Diagrama de flujo de llenado de comida

Programación de Arduino

```
#include "FirebaseESP8266.h"
#include <ESP8266WiFi.h>
#include <ArduinoJson.h>
#include <Servo.h>
#include <time.h>
#define servo D2
#define ledstart D0
#define ledstop D1
#define valvula D5
#define PinTrig D8
#define PinEcho D7

#define FIREBASE_HOST "dispensador-de-mascotas.firebaseio.com"
#define FIREBASE_AUTH
"Cg3TGPyJ7AYoYjMoCmFSpCA3Ws7RTCt7IeOBZ0Tc"
#define WIFI_SSID "Virus"
#define WIFI_PASSWORD "violeta2018"
```

```
const char* hostSMS = "fcm.googleapis.com";
char API_key[]="AIzaSyAi-W5iOhCdqm2Y9lxrg20UlQMwFlaFs-4";
FirebaseData firebaseData;
Servo servo1;

String animal,tamano,hora1,min1,hora2,min2,hora3,min3;
String token,comidadispensador;
float comidadispensador2;
String edad,etapa;
String diasemana,fechai,horai,estado;
String diasem,horafinal;
String hora,segund,minutos,dia,mes;
String agua;
float comidaplato=0;
int dele;
int timezone = -5 * 3600;
int time_buffer = 5000;
int dst = 0,u;
const float VelSon = 34000.0;
float distancia;
unsigned long sendDataPrevMillis = 0;
String path = "Petchow/id/467899/Parametros";
DynamicJsonBuffer jsonBuffer;
uint16_t count = 0;
boolean val;
boolean val2=true;
void setup()
{
 Serial.begin(115200); // velocidad coomunicacion
 pinMode(ledstart, OUTPUT);  //
 pinMode(ledstop, OUTPUT);  //configuracion puerto
 pinMode(valvula, OUTPUT);
```

```
 pinMode(PinTrig, OUTPUT);
 pinMode(PinEcho, INPUT);
 digitalWrite(ledstart, HIGH);
  confwifi(); // Configuracion red  wifi

}

void loop()
{
  time_t now = time(nullptr);
  tm* p_tm = localtime(&now); // configuracion de reloj
    ////////////////////////////impresion hora
if (estado=="activado" && p_tm->tm_sec==0){
    Serial.println(estado);
    Serial.println("");
    Serial.print(diasemana);
    Serial.print("");
    Serial.print("dele");
    Serial.println(dele);


     /////////////////////////// calculo de comida por activacion///////////////////////////
    comidadispensador2=comidadispensador2-comidaplato;
    Serial.print("Comida dispensador 2   =");
    Serial.println(comidadispensador2);

     /////////////////////////// ANGULO SERVO//////////////////////////////
    servo1.attach(servo);
    servo1.write(55);
    Serial.println("giro 180");
    delay(dele);
    servo1.write(0);
    Serial.println("giro 0");
```

```
//////////////////////////// envio de comida actual a BD//////////////////////////
   if (Firebase.setFloat(firebaseData, path + "/comidadispensador",
comidadispensador2))
    {
      Serial.println("PASSED");
      Serial.println("PATH: " + firebaseData.dataPath());
      Serial.println("TYPE: " + firebaseData.dataType());
      Serial.print("VALUE: ");
       if (firebaseData.dataType() == "float")
        Serial.println(firebaseData.floatData(), 2);
      else if (firebaseData.dataType() == "json")
        Serial.println(firebaseData.jsonData());
      Serial.println("----------------------------------");
      Serial.println();
    }

    }

  ///////////////////////VERIFICACION ESTADO DE AGUA////////////////////
  if (agua=="activado"){
  ultra();
  }
  ///////////////////////VERIFICACION ESTADO DE COMIDA/////////////////
 if ((comidadispensador.toFloat()<=20)&&p_tm->tm_sec==0&&val2==true){
   val2=false;
    sendSMS();
    delay(1000);
  }
 ////////////////////////////ENVIO DE ULTIMA CONEXION//////////////////////////
 if (millis() - sendDataPrevMillis > 67000)
 {
   sendDataPrevMillis = millis();
```

```
count++;
Serial.println("------------------------------------");
Serial.println("Enviando datos");
//CREACION NODO ACORDE EL TIPO DE DATO
if (Firebase.setString(firebaseData, path + "/Conexion", fechai+" "+horai))
{
  Serial.println("PASSED");
  Serial.println("PATH: " + firebaseData.dataPath());
  Serial.println("TYPE: " + firebaseData.dataType());
  Serial.print("VALUE: ");
  if (firebaseData.dataType() == "int")
    Serial.println(firebaseData.intData());
  else if (firebaseData.dataType() == "float")
    Serial.println(firebaseData.floatData(), 5);
  else if (firebaseData.dataType() == "double")
    printf("%.9lf\n", firebaseData.doubleData());
  else if (firebaseData.dataType() == "boolean")
    Serial.println(firebaseData.boolData() == 1 ? "true" : "false");
  else if (firebaseData.dataType() == "string")
    Serial.println(firebaseData.stringData());
  else if (firebaseData.dataType() == "json")
    Serial.println(firebaseData.jsonData());
  Serial.println("------------------------------------");
  Serial.println();
}
else
{
  Serial.println("FAILED");
  Serial.println("REASON: " + firebaseData.errorReason());
  Serial.println("------------------------------------");
  Serial.println();
}
```

```
    //Pause WiFi client from all Firebase calls and use shared SSL WiFi client
    if (firebaseData.pauseFirebase(true))
    {

      WiFiClientSecure client = firebaseData.getWiFiClient();
      //Use the client to make your own http connection...
    }
    else
    {
      Serial.println("------------------------------------");
      Serial.println("Can't pause the WiFi client...");
      Serial.println("------------------------------------");
      Serial.println();
    }
    //Desenganche el cliente WiFi de la tarea de Firebase
    firebaseData.pauseFirebase(false);
  }

  if (!Firebase.readStream(firebaseData))
  {
  }
//VERIFICACION DE SINCRONIZACION
  if (firebaseData.streamTimeout())
  {

    Serial.println("Stream timeout, resume streaming...");
    Serial.println();
  }

  actualizacion();
  val2=true;
}
```

```
void actualizacion(){

   /////////////////////////// ACTUALIZACION VALORES INICOIALES
//////////////////////////
   String Animal=json["animal"];
   String Tamano=json["tamano"];
   String Hora1=json["hora1"];
   String Min1=json["min1"];
   String Hora2=json["hora2"];
   String Min2=json["min2"];
   String Hora3=json["hora3"];
   String Min3=json["min3"];
   String Token=json["token"];
   String Comidadispensador=json["comidadispensador"];
   String Edad=json["edad"];
   String Etapa=json["etapa"];
   animal=Animal;
   tamano=Tamano;
   hora1=Hora1;
   min1=Min1;
   hora2=Hora2;
   min2=Min2;
   hora3=Hora3;
   min3=Min3;
   token=Token;
   comidadispensador=Comidadispensador;
   comidadispensador2=comidadispensador.toFloat();
   edad=Edad;
   etapa=Etapa;
   }
   Serial.println("-----------------------------------");
```

```
//////////////////////////IMPORTACION DE VALORES DE BD
  if (firebaseData.dataPath()=="/animal"){
    animal=firebaseData.stringData();
  }
  if (firebaseData.dataPath()=="/tamano"){
    tamano=firebaseData.stringData();
  }
  if (firebaseData.dataPath()=="/hora1"){
    hora1=firebaseData.intData();
  }
  if (firebaseData.dataPath()=="/hora2"){
    hora2=firebaseData.intData();
  }
  if (firebaseData.dataPath()=="/hora3"){
    hora3=firebaseData.intData();
  }
  if (firebaseData.dataPath()=="/min1"){
    min1=firebaseData.intData();
  }
  if (firebaseData.dataPath()=="/min2"){
    min2=firebaseData.intData();
  }
  if (firebaseData.dataPath()=="/min3"){
    min3=firebaseData.intData();
  }

  ////////////////////////////////VARIACION DE APERTURA SERVO
  if (animal=="gato"){
   if (tamano=="pequeno"){
      if(etapa=="mes" && edad.toInt()<7){
        dele=14000;
        comidaplato=0.040;
      }else{
```

```
        dele=16000;
        comidaplato=0.060;
       }
  }else if (tamano=="grande"){
  if(etapa=="mes" && edad.toInt()<7){
        dele=19000;
        comidaplato=0.060;
       }else{
        dele=20000;
        comidaplato=0.090;
       }
  }
}

if (animal=="perro"){
  if (tamano=="pequeno"){
  if(etapa=="mes" && edad.toInt()<7){
        dele=21000;
        comidaplato=0.090;
       }else{
        dele=22000;
        comidaplato=0.140;
       }
  }else if (tamano=="mediano"){
  if(etapa=="mes" && edad.toInt()<7){
        dele=23000;
        comidaplato=0.250;
       }else{
        dele=24000;
        comidaplato=0.350;
       }
  }else if (tamano=="grande"){
   if(etapa=="mes" && edad.toInt()<7){
```

```
      dele=25000;
      comidaplato=0.380;
     }else{
      dele=26000;
      comidaplato=0.500;
     }
  }
}
////////////////////////////////////////
//////////////////////////// REPORTE DE VARIABLES EN SERIAL
Serial.print("El animal es " );
Serial.println(animal);
Serial.print("El tamano es ");
Serial.println(tamano);
Serial.print("Hora 1 es ");
Serial.print(hora1);
Serial.print(":");
Serial.println(min1);
Serial.print("Hora 2 es ");
Serial.print(hora2);
Serial.print(":");
Serial.println(min2);
Serial.print("Hora 3 es ");
Serial.print(hora3);
Serial.print(":");
Serial.println(min3);
Serial.print("Apertura Seg: ");
Serial.println(dele);
Serial.print("Comida Plato = ");
Serial.println(comidaplato);
Serial.print("Comida Dispensador 2 = ");
Serial.println(comidadispensador2);
if (token!=""){
```

```
    Serial.print("SMS ok ");
    }else{
    Serial.print("SMS fallo ");
    }

  }
}
//////////////////////////// METODO CONEXION WIFI
void confwifi(){
 // WiFi.config(ip, gateway, subnet, dns);
  Serial.print("Conectado a wifi");
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Connecting to Wi-Fi");
  while (WiFi.status() != WL_CONNECTED)
  {
    Serial.print(".");
    delay(300);
  }
  //////////////////////////// IMPRESION DIRECCION IP ////////////////////////////
  Serial.println();
  Serial.print("Connected with IP: ");
  Serial.println(WiFi.localIP());
  Serial.println();

//////////////////////////// CONEXION BD ////////////////////////////
  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
  Firebase.reconnectWiFi(true);
  Firebase.setMaxRetry (firebaseData, 5 );
  Firebase.setMaxErrorQueue (firebaseData, 30 );

  if (!Firebase.beginStream(firebaseData, path))
  {
    Serial.println("----------------------------------");
```

```
    Serial.println("Can't begin stream connection...");
    Serial.println("REASON: " + firebaseData.errorReason());
    Serial.println("------------------------------------");
    Serial.println();
  }
  //////////////////////////// CONFIGURACION HORA POR ZONA HORARIA
//////////////////////////
  configTime(timezone, dst, "pool.ntp.org","time.nist.gov");
  Serial.println("\nSincronizando tiempo");
  while(!time(nullptr)){
     Serial.print("*");
     delay(1000);
  }
  Serial.println("\nTime response....OK");
}

//////////////////////////// METODO CONSULTA DIA/////////////////////////
String diaser(int diaserver){
   switch(diaserver){
  case 1:
  diasem="Lunes";
  break;
  case 2:
  diasem="Martes";
  break;
  case 3:
  diasem="Miercoles";
  break;
  case 4:
  diasem="Jueves";
  break;
  case 5:
  diasem="Viernes";
```

```
 break;
 case 6:
 diasem="Sabado";
 break;
 case 7:
 diasem="Domingo";
 break;
 }
 return diasem;
}


//////////////////////////// METODO ACCESO HORA////
String horario(int hora, int minuto){
 String est;

  if (animal=="perro"&&((String(hora)==hora1 &&
String(minuto)==min1)||(String(hora)==hora2 &&
String(minuto)==min2)||(String(hora)==hora3 && String(minuto)==min3)))
  {
        est="activado";
  }else if (animal=="gato"&&((String(hora)==hora1 &&
String(minuto)==min1)||(String(hora)==hora3 && String(minuto)==min3)))
 {
        est="activado";

  }else  {
        est="desactivado";
 }
 return  est;
}
```

```
//////////////////////////// MEDICION NIVEL AGUA//////////////////////////
void ultra(){
  iniciarTrigger();
  unsigned long tiempo = pulseIn(PinEcho, HIGH);
  distancia = tiempo * 0.000001 * VelSon / 2.0;
  Serial.print(distancia);
  Serial.print("cm");
  Serial.println();
  delay(500);

  if(distancia >=9){
     digitalWrite(ledstop, HIGH);
     digitalWrite(valvula, LOW);
     delay(5000);
     }else{
     digitalWrite(ledstop, LOW);
     digitalWrite(valvula, HIGH);
     }
}

// METODO INICIO NIVELSA AGUA SERIAL
void iniciarTrigger()
{
  // Ponemos el Triiger en estado bajo y esperamos 2 ms
  digitalWrite(PinTrig, LOW);
  delayMicroseconds(2);
  // Ponemos el pin Trigger a estado alto y esperamos 10 ms
  digitalWrite(PinTrig, HIGH);
  delayMicroseconds(10);
  // Comenzamos poniendo el pin Trigger en estado bajo
  digitalWrite(PinTrig, LOW);
}
```

```
//////////////////////////// METODO ULTIMA CONEXION ////////////////////////////
String registro(String f1,String h1){
  Serial.print("Enviando Registro");
    StaticJsonBuffer<200> jsonBuffer;
    JsonObject& root = jsonBuffer.createObject();
    root["Hora"] = f1+h1;
    root["Acceso"] = "Si";
    //Firebase.push(Proyecto+"/Registro/"+fechai+"/"+horai+"/", root);
    //Firebase.push(Proyecto+"/Registro/"+f1+"/", root);
    //Firebase.pushJSON(firebaseData, "/mascotas/Registro", root);

    String ok;
    return ok;
}

//////////////////////////// METODO ENVIO NOTIFICACION ////////////////////////////
void sendSMS() {
  String data = "{" ;
  data = data + "\"to\": \""+ token+"\"," ;
  data = data + "\"priority\": \"high\"," ;
  data = data + "\"notification\": {" ;
  data = data + "\"body\": \"SE DEBE RECARGAR COMIDA EN EL
DISPENSADOR\"," ;
  data = data + "\"sound\": \"default\"," ;
  data = data + "\"title\" : \"ALERTA\" " ;
  data = data + "} }" ;
 WiFiClient client;
Serial.println("Send data...");
  if (client.connect(hostSMS, 80)) {
    Serial.println("Connected to the server..");
    client.println("POST /fcm/send HTTP/1.1");
```

```
client.print("Authorization:key=");//AIzaSyCvQvI2_NMNmkOEoyMnbaiuHWwsz6b-Hu0
    client.println(API_key);
    client.println("Content-Type: application/json");
    client.println("Host: fcm.googleapis.com");
    client.print("Content-Length: ");
    client.println(data.length());
    client.print("\n");
    client.print(data);

    while (client.connected()) {
      String line = client.readStringUntil('\n');
        if (line == "\r") {
        Serial.println("headers received");
        break;
      }
    }
    String line = client.readStringUntil('\n');
    Serial.println("reply was:");
    Serial.println("==========");
    Serial.println(line);
    Serial.println("==========")
  }
  Serial.println("Data sent...Reading response..");
  while (client.available()) {
   char c = client.read();
   Serial.print(c);
  }
  Serial.println("Finished!");
  client.flush();
  client.stop();
  }//fin SMS
```

Anexo 4: Código de la apk

En este anexo podemos encontrar la codificación realizada en Firebase y distribuidas en diferentes Activitys las cuales generan la apk para los dispositivos móviles, de igual forma se muestran los diagramas de flujo usados.

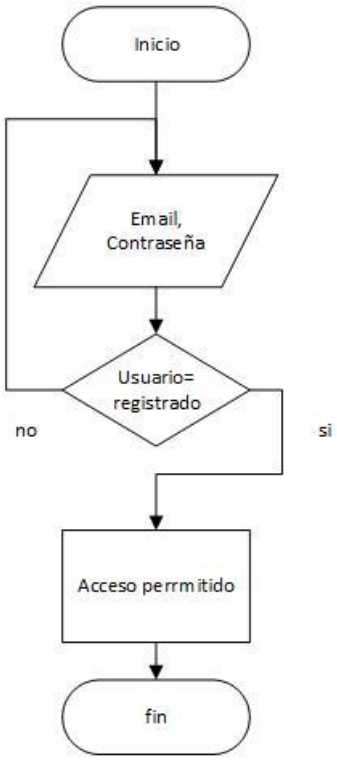


Figura 34 Diagrama de flujo de Activity 1
Elaborado por: El investigador

Activity 1 – registro de usuarios

```
package com.dispensadordemascotas.dispensador;


import android.content.Intent;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ProgressBar;
```

```
import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;



public class LoginActivity extends AppCompatActivity {
    String name, email2, uid2,token;
    private EditText inputEmail, inputPassword;
    FirebaseAuth auth;
    private ProgressBar progressBar;
    Button btnSignup, btnLogin, btnReset;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        // set the view now
        setContentView(R.layout.activity_login);



        FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();
        if (user != null) {
            //email2 = user.getEmail();
            uid2 = user.getUid();
        }

        //Get Firebase auth instance
        auth = FirebaseAuth.getInstance();
//
```

```
if (auth.getCurrentUser() != null) {
    //startActivity(new Intent(LoginActivity.this,pushActivity.class));
    //email2 = user.getEmail();
    //Intent i = new Intent(getApplicationContext(), MenuActivity.class);
   Intent i = new Intent(getApplicationContext(), Dispo.class);
    i.putExtra("uid", uid2);
    startActivity(i);
    finish();
 }


 inputEmail = (EditText) findViewById(R.id.email);
 inputPassword = (EditText) findViewById(R.id.password);
 btnSignup = (Button) findViewById(R.id.btn_signup);
 btnLogin = (Button) findViewById(R.id.btn_login);

 progressBar = (ProgressBar) findViewById(R.id.progressBar2);

 btnReset = (Button) findViewById(R.id.btn_reset_password);

 //Get Firebase auth instance
 auth = FirebaseAuth.getInstance();

 auth.getCurrentUser();


 btnSignup.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
       startActivity(new Intent(LoginActivity.this, Signup1.class));
    }
 });
```

```
        btnReset.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
              startActivity(new Intent(LoginActivity.this, ResetPassword.class));
            }
        });

        btnLogin.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                inputEmail.setEnabled(false);
                inputPassword.setEnabled(false);
                String email = inputEmail.getText().toString();
                final String password = inputPassword.getText().toString();

                if (TextUtils.isEmpty(email)) {
                    Toast.makeText(getApplicationContext(), "Ingrese una dirección de
Correo!", Toast.LENGTH_SHORT).show();
                    return;
                }

                if (TextUtils.isEmpty(password)) {
                    Toast.makeText(getApplicationContext(), "Ingrese la Contraseña!",
Toast.LENGTH_SHORT).show();
                    return;
                }

                progressBar.setVisibility(View.VISIBLE);

                //authenticate user
                auth.signInWithEmailAndPassword(email, password)
                        .addOnCompleteListener(LoginActivity.this, new
OnCompleteListener<AuthResult>() {
```

```
                @Override
                public void onComplete(@NonNull Task<AuthResult> task) {
                    // If sign in fails, display a message to the user. If sign in
succeeds
                    // the auth state listener will be notified and logic to handle the
                    // signed in user can be handled in the listener.
                    progressBar.setVisibility(View.GONE);
                    if (!task.isSuccessful()) {
                        // there was an error
                        if (password.length() < 6) {

inputPassword.setError(getString(R.string.minimum_password));
                        } else {
                            Toast.makeText(LoginActivity.this,
getString(R.string.auth_failed), Toast.LENGTH_LONG).show();
                            inputEmail.setEnabled(true);
                            inputPassword.setEnabled(true);
                        }
                    } else {
                        inputEmail.setEnabled(true);
                        inputPassword.setEnabled(true);

                        FirebaseUser user =
FirebaseAuth.getInstance().getCurrentUser();
                        name=user.getEmail();
                        uid2 = user.getUid();
                        //Intent i = new Intent(getApplicationContext(),
MenuActivity.class);
                        Intent i = new Intent(getApplicationContext(), Dispo.class);
                        i.putExtra("uid", uid2);
                        //i.putExtra("mail", name);
                        startActivity(i);
                        finish();
```

```
                    }
                }
            });
        }
    });
  }
}
```

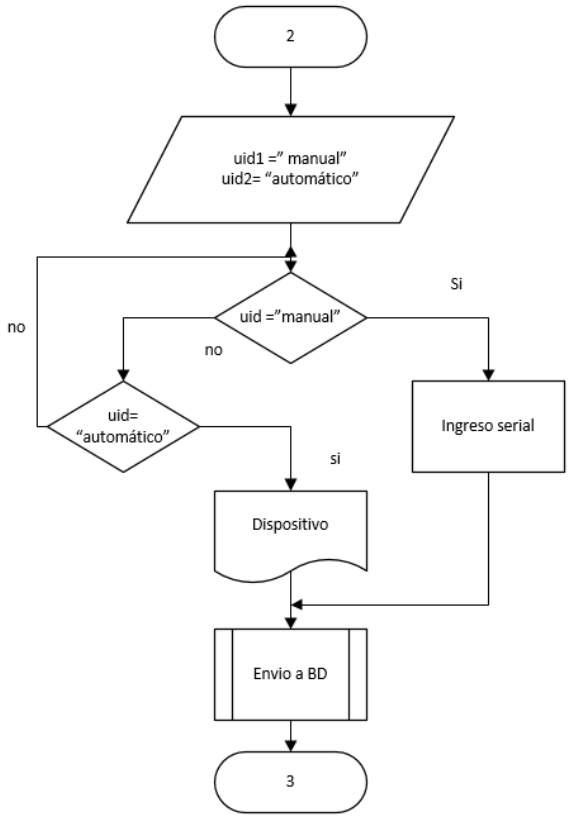


Figura 35 Diagrama de flujo de Activity 2

Elaborado por: El investigador

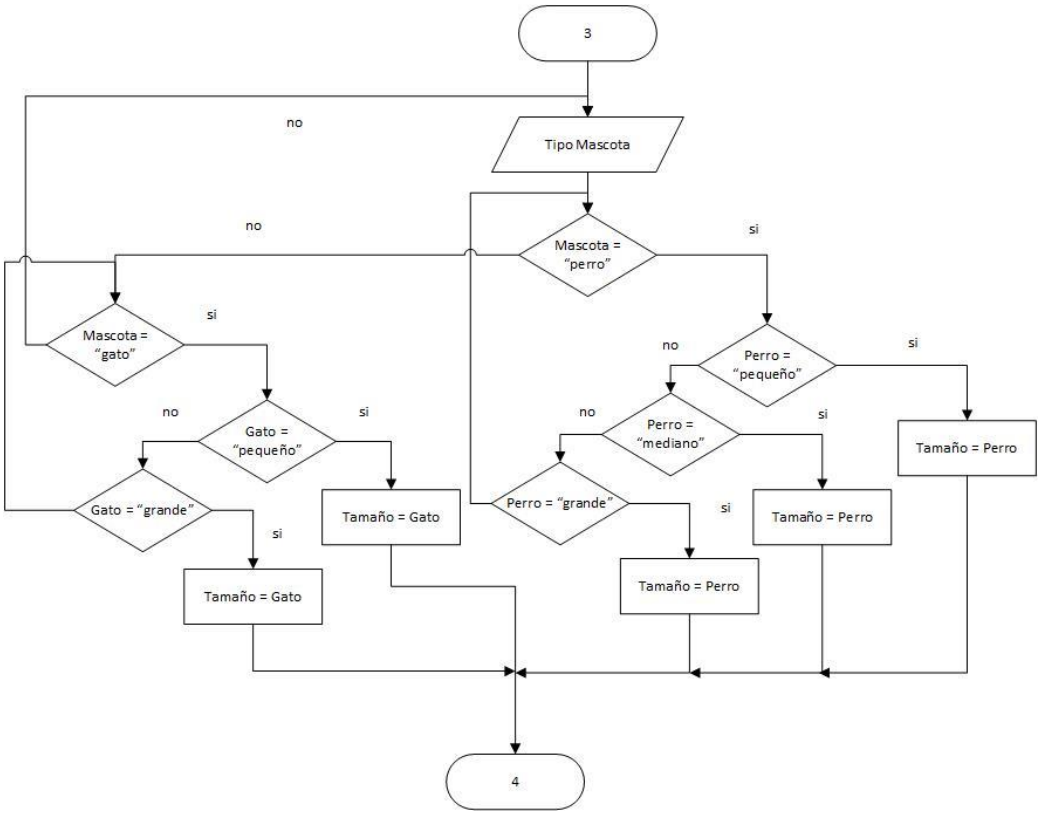


Figura 36 Diagrama de flujo Activity 3

Elaborado por: El investigador

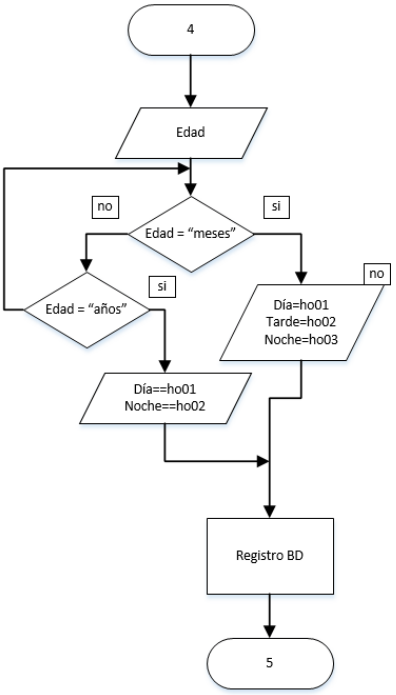


Figura 37 Diagrama de flujo Activity 4

Elaborado por: El investigador

Activity 2 – 3 – 4

```
package com.dispensadordemascotas.dispensador;

import android.app.Dialog;
import android.content.Intent;
import android.net.Uri;
import android.support.annotation.IdRes;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.text.InputType;
import android.text.TextUtils;
import android.util.Log;
import android.view.KeyEvent;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.LinearLayout;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.TextView;
import android.widget.Toast;

import com.dispensadordemascotas.dispensador.Objetos.FirebaseReferencias;
import com.dispensadordemascotas.dispensador.Objetos.Perris;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
```

```
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
import com.google.firebase.iid.FirebaseInstanceId;
import com.google.firebase.iid.InstanceIdResult;
import com.roughike.bottombar.BottomBar;
import com.roughike.bottombar.OnTabSelectListener;

import java.text.NumberFormat;

public class MenuActivity extends AppCompatActivity implements
View.OnClickListener {

    private ImageButton btncamara, btnmascota, btnregistro;
    private FirebaseAuth.AuthStateListener authListener;
    private FirebaseAuth auth;
    BottomBar menubar;
    int hoo1,hoo2,hoo3,minu1,minu2,minu3;
    float auxi;
    LinearLayout Pan1,Pan2,Pan3;
    private RadioButton
rbgato,rbperro,rbpequeno,rbmediano,rbgrande,rbmanual,rbautomatico,rbanos,rbmese
s,rdkilos,rdlibras;
    private RadioGroup S1,S2,S3;
    String link_tamano,link_mascota,link_etapa;
    Button bbeditar,bbaceptar,bbhorario;
    DatabaseReference database2;
    DatabaseReference database5;
    LinearLayout Lx2;
    EditText Edianos;
    Boolean est=true,est1=true;
    Button Editar,Aceptar,btnanadir,btnreset,btneditar2;
    EditText ho1,ho2,ho3;
    EditText mi1,mi2,mi3;
```

```
RadioGroup S5,S6,rgpesocomida;
EditText Edifunda,Edifunda2;
TextView totalcomida,totalcomida2;
FirebaseDatabase database1;
String superid,superuid;

int cf;
float cd;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_menu);
    ho1 = (EditText) findViewById(R.id.hora1);
    ho2 = (EditText) findViewById(R.id.hora2);
    ho3 = (EditText) findViewById(R.id.hora3);
    mi1 = (EditText) findViewById(R.id.minu1);
    mi2 = (EditText) findViewById(R.id.minu2);
    mi3 = (EditText) findViewById(R.id.minu3);
    rbgato=(RadioButton)findViewById(R.id.rdgato);
    rbperro=(RadioButton)findViewById(R.id.rdperro);
    rbpequeno=(RadioButton)findViewById(R.id.rdpequeno);
    rbmediano=(RadioButton) findViewById(R.id.rdmedio);
    rbgrande=(RadioButton)findViewById(R.id.rdgrande);
    rbmanual=(RadioButton) findViewById(R.id.rdmanual);
    rbautomatico=(RadioButton)findViewById(R.id.rdauto);
    S5=(RadioGroup)findViewById(R.id.s5);
    S6=(RadioGroup)findViewById(R.id.s6);
    S2=(RadioGroup)findViewById(R.id.s2);
    rbanos=(RadioButton)findViewById(R.id.rdanos);
    rbmeses=(RadioButton)findViewById(R.id.rdmeses);
    Edianos=(EditText) findViewById(R.id.edianos);

    Pan1=(LinearLayout)findViewById(R.id.pan1);
```

```
        Pan2=(LinearLayout)findViewById(R.id.pan2);
        Pan3=(LinearLayout)findViewById(R.id.pan3);
        menubar = (BottomBar) findViewById(R.id.MenuBar);
        btncamara = (ImageButton) findViewById(R.id.camara);
        btnmascota = (ImageButton) findViewById(R.id.mascota);
        btnregistro = (ImageButton) findViewById(R.id.registro);
        bbaceptar=(Button)findViewById(R.id.Bbaceptar);
        bbeditar=(Button)findViewById(R.id.Bbeditar);
        bbhorario=(Button)findViewById(R.id.Bbhorario);
        btncamara.setOnClickListener(this);
        btnmascota.setOnClickListener(this);
        btnregistro.setOnClickListener(this);


        Intent intent = getIntent();
        Bundle extras = intent.getExtras();
        if (extras != null) {
            superid = (String) extras.get("id");
            superuid=(String) extras.get("uid");
        }

        bbaceptar.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                if (TextUtils.isEmpty(Edianos.getText().toString().trim())) {
                    Toast.makeText(getApplicationContext(), "Ingrese la edad de su
mascota", Toast.LENGTH_SHORT).show();
                }
            bbeditar.setVisibility(View.VISIBLE);
            }
        });

        bbeditar.setOnClickListener(new View.OnClickListener() {
```

```
    @Override
    public void onClick(View view) {
        bbeditar.setVisibility(View.GONE);
        bbaceptar.setVisibility(View.VISIBLE);
        desbloqueo3();
    }
});


//get firebase auth instance
auth = FirebaseAuth.getInstance();
//get current user
final FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();
authListener = new FirebaseAuth.AuthStateListener() {
    @Override
    public void onAuthStateChanged(@NonNull FirebaseAuth firebaseAuth) {
        FirebaseUser user = firebaseAuth.getCurrentUser();
        if (user == null) {
            // user auth state is changed - user is null
            // launch login activity
            startActivity(new Intent(MenuActivity.this,Dispo.class));
            finish();
        }
    }
};

FirebaseInstanceId.getInstance().getInstanceId()
        .addOnCompleteListener(new OnCompleteListener<InstanceIdResult>() {
            @Override
            public void onComplete(@NonNull Task<InstanceIdResult> task) {
                if (!task.isSuccessful()) {
                    return;
                }
```

```
                    // Get new Instance ID token
                    String token = task.getResult().getToken();
                    // Log and toast
                    //Toast.makeText(MenuActivity.this, token,
Toast.LENGTH_SHORT).show();
                    database2 =
FirebaseDatabase.getInstance().getReference().child("Petchow/id/"+superid+"/Param
etros/");
                    database2.child("token").setValue(token);
                }
            });

        metobar();
        bloqueo1();


     Edianos.setOnKeyListener(new View.OnKeyListener() {
         public boolean onKey(View v, int keyCode, KeyEvent event) {
            if ((event.getAction() == KeyEvent.ACTION_DOWN) && (keyCode ==
KeyEvent.KEYCODE_ENTER)) {
                //aqui iria tu codigo al presionar el boton enter o done
                S5.clearCheck();
                S6.clearCheck();
                return true;
            }
            return false;
         }
      });
   }
   public void reset() {
      Edianos.setText("");
      S2.clearCheck();
      S5.clearCheck();
```

```
    S6.clearCheck();
}
public void bloqueo1(){
    rbpequeno.setEnabled(false);
    rbmediano.setEnabled(false);
    rbgrande.setEnabled(false);
    rbautomatico.setEnabled(false);
    rbmanual.setEnabled(false);
    rbanos.setEnabled(false);
    rbmeses.setEnabled(false);
    Edianos.setEnabled(false);
}

public void desbloqueo1(){
    rbpequeno.setEnabled(true);
    rbmediano.setEnabled(true);
    rbgrande.setEnabled(true);
}

public void desbloqueo2(){
    rbanos.setEnabled(true);
    rbmeses.setEnabled(true);
    Edianos.setEnabled(true);
}
public void desbloqueo3(){
    rbautomatico.setEnabled(true);
    rbmanual.setEnabled(true);
}

public void onRadioButtonClicked(View view) {


    boolean checked = ((RadioButton) view).isChecked();
```

```
switch(view.getId()) {

    case R.id.rdperro:
        if (checked)
            link_mascota="perro";
            rbmediano.setVisibility(View.VISIBLE);
            reset();
            bloqueo1();
            desbloqueo1();
            break;
    case R.id.rdgato:
        if (checked)
            link_mascota="gato";
            rbmediano.setVisibility(View.GONE);
            reset();
            bloqueo1();
            desbloqueo1();
        break;
    case R.id.rdpequeno:
        link_tamano="pequeno";
        hoo1=7;
        hoo2=12;
        hoo3=18;
        minu1=0;
        minu2=0;
        minu3=0;
        desbloqueo2();
        break;
    case R.id.rdmedio:
        link_tamano="mediano";
        hoo1=8;
        hoo2=13;
```

```
            hoo3=21;
            minu1=0;
            minu2=0;
            minu3=0;
            desbloqueo2();
            break;
        case R.id.rdgrande:
            link_tamano="grande";
            hoo1=5;
            hoo2=14;
            hoo3=18;
            minu1=0;
            minu2=0;
            minu3=0;
            desbloqueo2();
            break;

        case R.id.rdanos:
            link_etapa="ano";
            if (TextUtils.isEmpty(Edianos.getText().toString().trim())) {
                S5.clearCheck();
                S6.clearCheck();
                Toast.makeText(getApplicationContext(), "Ingrese la edad de su
mascota", Toast.LENGTH_SHORT).show();
                return;
            }else{
                desbloqueo3();
                S6.clearCheck();
            }
            est=false;
            break;
        case R.id.rdmeses:
            link_etapa="mes";
```

```
            if (TextUtils.isEmpty(Edianos.getText().toString().trim())) {
               S5.clearCheck();
               S6.clearCheck();
               Toast.makeText(getApplicationContext(), "Ingrese la edad de su
mascota", Toast.LENGTH_SHORT).show();
               return;
            }else{
               desbloqueo3();
               S6.clearCheck();
            }
            if (Integer.parseInt(Edianos.getText().toString())<7){
               est=true;
            }else{
               est=false;
            }
            break;

         case R.id.rdmanual:

            if ((Integer.parseInt(Edianos.getText().toString())>20)) {
               Toast.makeText(getApplicationContext(), "Error en edad de la
mascota", Toast.LENGTH_SHORT).show();
               return;
            }
            desbloqueo2();
            final Dialog mydiag1 = new Dialog(this);
            mydiag1.setTitle("Instrucciones");
            mydiag1.setContentView(R.layout.horariocomida);

            Editar =(Button) mydiag1.findViewById(R.id.editar);
            Aceptar =(Button) mydiag1.findViewById(R.id.aceptar);
            ho1 = (EditText) mydiag1.findViewById(R.id.hora1);
            ho2 = (EditText) mydiag1.findViewById(R.id.hora2);
```

```
            ho3 = (EditText) mydiag1.findViewById(R.id.hora3);
            mi1 = (EditText) mydiag1.findViewById(R.id.minu1);
            mi2 = (EditText) mydiag1.findViewById(R.id.minu2);
            mi3 = (EditText) mydiag1.findViewById(R.id.minu3);
            Lx2 =(LinearLayout)mydiag1.findViewById(R.id.lx2);
            Editar.setVisibility(View.GONE);
            Aceptar.setVisibility(View.VISIBLE);

            /*if (link_mascota.equals("gato")||est==false){
               Lx2.setVisibility(View.GONE);
            }
*/

            if (est==false){
               Lx2.setVisibility(View.GONE);
            }

            Aceptar.setOnClickListener(new View.OnClickListener() {
               @Override
               public void onClick(View view) {

                  if (TextUtils.isEmpty(ho1.getText().toString().trim())||
TextUtils.isEmpty(ho3.getText().toString().trim())
                        || TextUtils.isEmpty(mi1.getText().toString().trim()) ||
TextUtils.isEmpty(mi3.getText().toString().trim())) {
                     if (link_mascota.equals("perro")&& est==true){
                        if(TextUtils.isEmpty(ho2.getText().toString().trim())||
TextUtils.isEmpty(mi2.getText().toString().trim())  ){
                           return;
                        }

if((Double.parseDouble(ho2.getText().toString())>24)||(Double.parseDouble(mi2.get
Text().toString())>59)){
```

```
                return;
            }
        }
        Toast.makeText(getApplicationContext(), "Por favor llene los
campos solicitados", Toast.LENGTH_SHORT).show();
        return;
    }if
((Double.parseDouble(ho1.getText().toString())>24)||(Double.parseDouble(ho3.getT
ext().toString())>24)){
        Toast.makeText(getApplicationContext(), "Formato de hora
incorrecta", Toast.LENGTH_SHORT).show();
        return;
    }if
((Double.parseDouble(mi1.getText().toString())>59)||(Double.parseDouble(mi3.getT
ext().toString())>59)){
        Toast.makeText(getApplicationContext(), "Formato de minutos
incorrecta", Toast.LENGTH_SHORT).show();
        return;
    } else{
        database2 =
FirebaseDatabase.getInstance().getReference().child("Petchow/id/"+superid+"/Param
etros/");

database2.child("hora1").setValue(Integer.parseInt(ho1.getText().toString()));

database2.child("hora3").setValue(Integer.parseInt(ho3.getText().toString()));

database2.child("min1").setValue(Integer.parseInt(mi1.getText().toString()));

database2.child("min3").setValue(Integer.parseInt(mi3.getText().toString()));
        /*if (link_mascota.equals("perro")&&est==true){

database2.child("hora2").setValue(Integer.parseInt(ho2.getText().toString()));
```

```
database2.child("min2").setValue(Integer.parseInt(mi2.getText().toString()));
                    }*/

                    if (est==true){

database2.child("hora2").setValue(Integer.parseInt(ho2.getText().toString()));

database2.child("min2").setValue(Integer.parseInt(mi2.getText().toString()));
                    }


                    database5 =
FirebaseDatabase.getInstance().getReference().child("Petchow/id/"+superid+"/Param
etros/");
                    database5.child("animal").setValue(link_mascota);
                    database5.child("tamano").setValue(link_tamano);

database5.child("edad").setValue(Integer.parseInt(Edianos.getText().toString()));
                    database5.child("etapa").setValue(link_etapa);

                }

                Editar.setOnClickListener(new View.OnClickListener() {
                    @Override
                    public void onClick(View view) {
                        Aceptar.setVisibility(View.VISIBLE);
                        Editar.setVisibility(View.GONE);
                        ho1.setEnabled(true);
                        ho2.setEnabled(true);
                        ho3.setEnabled(true);
                        mi1.setEnabled(true);
                        mi2.setEnabled(true);
```

```
                    mi3.setEnabled(true);
                    ho1.setText("");
                    ho2.setText("");
                    ho3.setText("");
                    mi1.setText("");
                    mi2.setText("");
                    mi3.setText("");
                }
            });

            mydiag1.cancel();
            Toast.makeText(getApplicationContext(), "Se ha guardado el
horario", Toast.LENGTH_SHORT).show();

            // normal();
          }
        });

        mydiag1.show();
        break;
      case R.id.rdauto:
        datose();
        break;
    }
  }

  public void abrircamara() {
    Intent intent = getPackageManager().getLaunchIntentForPackage("com.ezviz");
    intent.addCategory(Intent.CATEGORY_LAUNCHER);

    startActivity(intent);

  }
```

```
    public void registro_comida() {
        final Dialog mydiag3 = new Dialog(this);
        mydiag3.setContentView(R.layout.registrocomida);
        Edifunda = (EditText) mydiag3.findViewById(R.id.edifunda);
        Edifunda2 = (EditText) mydiag3.findViewById(R.id.edifunda2);
        btnanadir=(Button)mydiag3.findViewById(R.id.Btnanadir);
        btnreset=(Button)mydiag3.findViewById(R.id.Btnreset);
        totalcomida=(TextView)mydiag3.findViewById(R.id.totalcomida);
        totalcomida2=(TextView)mydiag3.findViewById(R.id.totalcomida2);

        btneditar2=(Button)mydiag3.findViewById(R.id.Btneditar2);



        database1 = FirebaseDatabase.getInstance();
        DatabaseReference usuario_identi =
database1.getReference(FirebaseReferencias.REFERENCIAS_);
        usuario_identi.child("id/"+superid+"/Parametros/").addValueEventListener(new
ValueEventListener() {
            //Consulta nodo general
            @Override
            public void onDataChange(DataSnapshot dataSnapshot) {
                Perris perris = dataSnapshot.getValue(Perris.class);
                cd= (float) perris.getComidadispensador();
                cf= (int) perris.getComidafunda();
                totalcomida.setText(String.valueOf(cd));
                totalcomida2.setText(String.valueOf(cf));
            }
            @Override
            public void onCancelled(DatabaseError databaseError) {
            }
```

```
        });

        Edifunda.setEnabled(false);
        Edifunda2.setEnabled(false);
        database2 =
FirebaseDatabase.getInstance().getReference().child("Petchow/id/"+superid+"/Param
etros/");


        btneditar2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                btnanadir.setEnabled(true);
                Edifunda.setEnabled(true);
                Edifunda2.setEnabled(true);
                btnanadir.setVisibility(View.VISIBLE);
                btnreset.setVisibility(View.VISIBLE);
                btneditar2.setVisibility(View.GONE);
            }
        });

        btnanadir.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {

                if
(TextUtils.isEmpty(Edifunda2.getText().toString().trim())&&TextUtils.isEmpty(Edif
unda.getText().toString().trim())) {
                    Toast.makeText(getApplicationContext(), "Por favor ingrese valores",
Toast.LENGTH_SHORT).show();
                    return;
                }
```

```
            float v1,v2;
            float v3,v4;
            String comida1,comida2;



            if
(TextUtils.isEmpty(Edifunda.getText().toString().trim())&&Integer.parseInt(Edifund
a2.getText().toString())!=0) {

                if ((cf==0 && cd!=0)||(cf==0&&cd==0)){
                    v2=Integer.parseInt(Edifunda2.getText().toString());
                    v4=cd+v2;
                    v3=cf;
                    String s = String.format("%.2f", v4);
                    float number = Float.valueOf(s);
                    comida1 = String.valueOf(v3);
                    comida2 = String.valueOf(number);
                    database2.child("comidadispensador").setValue(number);
                    database2.child("comidafunda").setValue(v3);
                }else if (cf!=0){
                    v2=Integer.parseInt(Edifunda2.getText().toString());
                    v4=cd+v2;
                    v3=cf-v2;
                    if (v3<0){
                        Toast.makeText(getApplicationContext(), "Error ingreso comida,
valor maximo de comida "   + cf, Toast.LENGTH_SHORT).show();

                        return;
                    }
                    comida1 = String.valueOf(v3);
                    comida2 = String.valueOf(v4);
```

```
                NumberFormat formatter = NumberFormat.getNumberInstance();
                formatter.setMinimumFractionDigits(2);
                formatter.setMaximumFractionDigits(2);
                String s = String.format("%.2f", v4);
                float number = Float.valueOf(s);
                database2.child("comidadispensador").setValue(number);
                database2.child("comidafunda").setValue(v3);
            }else if(Integer.parseInt(Edifunda2.getText().toString())>cf && cd!=0){
                Toast.makeText(getApplicationContext(), "El valor maximo de
comida sobrante en la funda es " + cf, Toast.LENGTH_SHORT).show();
                return;
            }

        }  else if
(TextUtils.isEmpty(Edifunda2.getText().toString().trim())&&Integer.parseInt(Edifun
da.getText().toString())!=0) {
            Toast.makeText(getApplicationContext(), "Ingrese el valor a ingresar",
Toast.LENGTH_SHORT).show();
            return;
        }else{
            v1=Integer.parseInt(Edifunda.getText().toString());
            v2=Integer.parseInt(Edifunda2.getText().toString());
            if (v2>v1){
                Toast.makeText(getApplicationContext(), "Valores erroneos",
Toast.LENGTH_SHORT).show();
                return;
            }
            v3=((cf+(v1-v2)));
            v4=(cd+v2);
            ////
            String s = String.format("%.2f", v4);
            float number = Float.valueOf(s);
            comida1 = String.valueOf(v4);
```

```
            comida2 = String.valueOf(v3);
            database2.child("comidadispensador").setValue(number);
            database2.child("comidafunda").setValue(v3);
            totalcomida.setText(comida1);
            totalcomida2.setText(comida2);
        }

        btnanadir.setVisibility(View.GONE);
        btnreset.setVisibility(View.GONE);
        btneditar2.setVisibility(View.VISIBLE);
        Edifunda2.setText("");
        Edifunda.setText("");
    }
});

btnreset.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        btnanadir.setEnabled(true);
        Edifunda.setText("");
        Edifunda2.setText("");
        totalcomida.setText("");
        totalcomida2.setText("");
        database2.child("comidafunda").setValue(0);
        database2.child("comidadispensador").setValue(0);
        btnanadir.setVisibility(View.GONE);
        btnreset.setVisibility(View.GONE);
        btneditar2.setVisibility(View.VISIBLE);
    }
});
```

```
    mydiag3.show();
}

@Override
public void onClick(View view) {
    switch (view.getId()) {
        case R.id.camara:
            //Invocamos al método:
            abrircamara();
            break;
        case R.id.registro:
            registro_comida();
            break;
    }
}

@Override
public void onStart() {
    super.onStart();
    auth.addAuthStateListener(authListener);
}

public void metobar(){
    menubar.setOnTabSelectListener(new OnTabSelectListener() {
        @Override
        public void onTabSelected(@IdRes int tabId) {
            switch (tabId) {
                case R.id.tab_real:
                    Pan1.setVisibility(View.VISIBLE);
                    Pan2.setVisibility(View.GONE);
                    Pan3.setVisibility(View.GONE);
                    break;
```

```
            case R.id.tab_datos:
                Pan2.setVisibility(View.VISIBLE);
                Pan1.setVisibility(View.GONE);
                Pan3.setVisibility(View.GONE);
                break;
            case R.id.tab_cuenta:
                Pan1.setVisibility(View.GONE);
                Pan2.setVisibility(View.GONE);
                Pan3.setVisibility(View.VISIBLE);
                break;
        }
      }
   });
}

public void Cerrar_sesion (View v){
   signOut();
}

public void info_app (View v){
   Uri uri = Uri.parse("https://www.facebook.com/lenin1994");
   Intent intent = new Intent(Intent.ACTION_VIEW, uri);
   startActivity(intent);
}

public void horariocom (View v){
   final Dialog mydiag3 = new Dialog(this);
   mydiag3.setTitle("Instrucciones");
   mydiag3.setContentView(R.layout.horariocomida);
   mydiag3.show();
}

public void registroc (View v){
```

```
        final Dialog mydiag3 = new Dialog(this);
        mydiag3.setTitle("Instrucciones");
        mydiag3.setContentView(R.layout.registrocomida);
        mydiag3.show();
    }


    public void signOut() {
        auth.signOut();
    }
    public void Salirapk (View v){
        System.exit(0);
    }
    void datose(){
        database2 =
FirebaseDatabase.getInstance().getReference().child("Petchow/id/"+superid+"/Param
etros/");
        database2.child("animal").setValue(link_mascota);
        database2.child("tamano").setValue(link_tamano);

database2.child("edad").setValue(Integer.parseInt(Edianos.getText().toString()));
        database2.child("etapa").setValue(link_etapa);

        database2.child("hora1").setValue(hoo1);
        database2.child("hora3").setValue(hoo3);
        database2.child("min1").setValue(minu1);
        database2.child("min3").setValue(minu3);

        final Dialog mydiag2 = new Dialog(this);
        mydiag2.setTitle("Instrucciones");
        mydiag2.setContentView(R.layout.horariosauto);
        TextView Tx1 =(TextView) mydiag2.findViewById(R.id.tx1);
        TextView Tx2 =(TextView) mydiag2.findViewById(R.id.tx2);
```

```
TextView Tx3 =(TextView) mydiag2.findViewById(R.id.tx3);
TextView Tx4 =(TextView) mydiag2.findViewById(R.id.textView30);
mydiag2.show();

if (link_mascota.equals("perro")&&est==true){
   database2.child("hora2").setValue(hoo2);
   database2.child("min2").setValue(minu2);
   if (hoo1<10 && minu1<10) {
      Tx1.setText("0" + hoo1 + ":0" + minu1);
   }else if (hoo1>9 && minu1<10) {
      Tx1.setText("" + hoo1 + ":0" + minu1);
   }else{
      Tx1.setText("" + hoo1+":"+minu1);}
   if (hoo2<10 && minu2<10){
   Tx2.setText("0" + hoo2+":0"+minu2);}
   else if (hoo2>9 && minu2<10) {
      Tx2.setText("" + hoo2 + ":0" + minu1);
   }else{
      Tx2.setText("" + hoo2+":"+minu2);}
   if (hoo3<10 && minu3<10){
   Tx3.setText("0" + hoo3+":0"+minu3);}
   else if (hoo3>9 && minu3<10) {
      Tx3.setText("" + hoo3 + ":0" + minu3);
   }else{
      Tx3.setText("" + hoo3+":"+minu3);}

}else{

   if (hoo1<10 && minu1<10){
      Tx1.setText("0" + hoo1+":0"+minu1);}
   else if (hoo1>9 && minu1<10) {
      Tx1.setText("" + hoo1 + ":0" + minu1);
   }else{
```

```
            Tx1.setText("" + hoo1+":"+minu1);}
        if (hoo3<10 && minu3<10){
            Tx2.setText("0" + hoo3+":0"+minu3);}
        else if (hoo3>9 && minu3<10) {
            Tx2.setText("" + hoo3 + ":0" + minu3);
        }else{
            Tx2.setText("" + hoo3+":"+minu3);}
        Tx3.setVisibility(View.GONE);
        Tx4.setVisibility(View.GONE);
      }



  }


  @Override
  public void onBackPressed() {
      Intent i = new Intent(getApplicationContext(), Dispo.class);
      i.putExtra("uid", superuid);
      //i.putExtra("mail", name);
      startActivity(i);
      finish();
  }
}
```

Activity 5 – Cerrar sesión

```
package com.dispensadordemascotas.dispensador;

import android.content.Intent;
import android.support.annotation.NonNull;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
```

```
import android.text.TextUtils;
import android.view.View;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ProgressBar;
import android.widget.Spinner;
import android.widget.Toast;

import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;

public class Signup1 extends AppCompatActivity {
    private EditText inputEmail, inputPassword, inputPassword2;
    private EditText inputNombre, inputTelefono,inputBarrio;
    String name, uid,token;
    String modulo;
    String nombre,telefono,barrio,email,password,password2;
    private Button btnSignIn, btnSignUp, btnResetPassword;
    private ProgressBar progressBar;
    private FirebaseAuth auth;
    DatabaseReference mDataRef3;
    DatabaseReference mDataRef4;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

```
setContentView(R.layout.activity_signup1);

//Get Firebase auth instance
auth = FirebaseAuth.getInstance();
inputPassword2 = (EditText) findViewById(R.id.password2);

btnSignIn = (Button) findViewById(R.id.sign_in_button);
btnSignUp = (Button) findViewById(R.id.sign_up_button);
inputEmail = (EditText) findViewById(R.id.email);
inputPassword = (EditText) findViewById(R.id.password);
progressBar = (ProgressBar) findViewById(R.id.progressBar);
btnResetPassword = (Button) findViewById(R.id.btn_reset_password);

btnResetPassword.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View v) {
      startActivity(new Intent(Signup1.this, ResetPasswordActivity.class));
   }
});

btnSignIn.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View v) {
      finish();
   }
});

btnSignUp.setOnClickListener(new View.OnClickListener() {
   @Override
   public void onClick(View v) {
      email = inputEmail.getText().toString().trim();
      password = inputPassword.getText().toString().trim();
      password2 = inputPassword2.getText().toString().trim();
```

```
            if (TextUtils.isEmpty(email)) {
                Toast.makeText(getApplicationContext(), "Ingrese una dirección de
Email", Toast.LENGTH_SHORT).show();
                return;
            }

            if (TextUtils.isEmpty(password)) {
                Toast.makeText(getApplicationContext(), "Ingrese una Contraseña",
Toast.LENGTH_SHORT).show();
                return;
            }

            if (TextUtils.isEmpty(password2)) {
                Toast.makeText(getApplicationContext(), "Ingrese una Contraseña",
Toast.LENGTH_SHORT).show();
                return;
            }


            if (password.length() < 6) {
                Toast.makeText(getApplicationContext(), "La contraseña debe contener
al menos 6 caracteres", Toast.LENGTH_SHORT).show();
                return;
            }


            progressBar.setVisibility(View.VISIBLE);
            //create user
            auth.createUserWithEmailAndPassword(email, password)
                    .addOnCompleteListener(Signup1.this, new
OnCompleteListener<AuthResult>() {
```

```
                @Override
                public void onComplete(@NonNull Task<AuthResult> task) {
                    //Toast.makeText(SignupActivity.this, "Usuario Creado
Exitosamente:" + task.isSuccessful(), Toast.LENGTH_SHORT).show();
                    Toast.makeText(Signup1.this, "Usuario Creado Exitosamente",
Toast.LENGTH_SHORT).show();

                    progressBar.setVisibility(View.GONE);

                    if (!task.isSuccessful()) {
                        Toast.makeText(Signup1.this, "Autenticación fallida." +
task.getException(),
                                Toast.LENGTH_SHORT).show();
                    } else {
                        FirebaseUser user =
FirebaseAuth.getInstance().getCurrentUser();
                        //uid = user.getUid();

                        if (user != null) {
                            //email2 = user.getEmail();
                            uid = user.getUid();
                        }

                        auth = FirebaseAuth.getInstance();
                        mDataRef3 =
FirebaseDatabase.getInstance().getReference().child("Petchow/Usuarios/"+uid);
                        mDataRef3.child("Email").setValue(email);

mDataRef3.child("key").setValue(mDataRef3.getKey().toString());

                        //Intent i = new Intent(getApplicationContext(),
MenuActivity.class);
                        Intent i = new Intent(getApplicationContext(), Dispo.class);
```

```
                        i.putExtra("uid", uid);
                        startActivity(i);

                        //startActivity(new Intent(SignupActivity.this,
pushActivity.class));
                        finish();
                    }
                }
            });

        }
    });

  }
  @Override
  protected void onResume() {
    super.onResume();
    progressBar.setVisibility(View.GONE);
  }
}
```

Anexo 5: Código de la pagina web

En este anexo encontramos la codificación requería para la interfaz web

Pantalla de registro

```
<!DOCTYPE html>
<!-- saved from url=(0014)about:internet -->
<html><head><meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Petchow</title>
    <link rel="icon" type="image/png" href="https://negocitoproductivo.firebaseapp.com/images/deal.png">
    <link href="./PETCHOW_files/css" rel="stylesheet">
    <link rel="stylesheet" href="./PETCHOW_files/style.css">

    <!-- update the version number as needed -->
    <script defer="" src="./PETCHOW_files/firebase-app.js"></script>
    <!-- include only the Firebase features as you need -->
    <script defer="" src="./PETCHOW_files/firebase-auth.js"></script>
    <script defer="" src="./PETCHOW_files/firebase-database.js"></script>
    <script defer="" src="./PETCHOW_files/firebase-messaging.js"></script>
    <script defer="" src="./PETCHOW_files/firebase-storage.js"></script>
    <!-- initialize the SDK after all desired features are loaded -->
    <script defer="" src="./PETCHOW_files/init.js"></script>

  </head>
  <body>

    <div id="login_div" class="main-div" style="display: block;">
      <h2>PETS CHOW</h2>
      <h3>Potenciando tu negocio</h3>
      <input type="email" placeholder="Email..." id="email_field">
      <input type="password" placeholder="Password..." id="password_field">
```

```
    <button onclick="login()">Ingresar</button>
    <br>
    <button onclick="register()">Registrar</button>
   </div>

   <div id="user_div" class="loggedin-div" style="display: none;">
    <h3>Bienvenido</h3>
    <p id="user_para">Welcome to Firebase web login Example. You're currently
logged in.</p>
    <button onclick="logout()">Logout</button>
   </div>

    <script src="./PETCHOW_files/firebase.js"></script>
    <script src="./PETCHOW_files/autenticacion.js"></script>
    <script src="./PETCHOW_files/index.js"></script>


</body></html>
```

Pantalla de control

```
<!doctype html>
<html xmlns="http://www.w3.org/1999/xhtml" xmlns:v="urn:schemas-microsoft-
com:vml" xmlns:o="urn:schemas-microsoft-com:office:office">

<head>
        <title>CONTROL</title>
<script src="js/Chart.min.js"></script>
        <script src="https://www.gstatic.com/firebasejs/4.13.0/firebase.js"></script>
        <script
src="https://ajax.googleapis.com/ajax/libs/jquery/2.2.0/jquery.min.js"></script>
        <script type="text/javascript"
src="http://ajax.googleapis.com/ajax/libs/jquery/1.3.2/jquery.min.js"></script>
```

```
        <script type="text/javascript"
src="http://ajax.googleapis.com/ajax/libs/jqueryui/1.7.2/jquery-ui.min.js"></script>
        <script src="js/addColumnFunction.js"></script>
        <script src="js/jquery.table2excel.js"></script>
        <link rel="stylesheet" type="text/css" href="css/style.css">
  <link rel="stylesheet" type="text/css" href="css/jquery-ui-1.7.2.custom.css" />
</head>
        <style type="text/css" rel="stylesheet">
        #boton{
                height:30px;
                width:40px;
                background: #000000;
                color: white;
                border-radius: 10px;

        }
</style>
<body>

        <div id="cabecera">
                <div id="contenido-cabecera">
                        <!--img src="images/logofisei2.png" alt="Logo Demos"
height="120" /-->
                </div>
        </div>
        <center><h1>PANEL DE CONTROL</h1></center>
<button onclick="logout()">Logout</button>
<script src="./PETCHOW_files/index.js"></script>
<fieldset>
          <legend>Configuraciones Iniciales</legend>


                <div class="container">
```

```
                    <!-- Contenendor para mostrar los datos leidos -->
                    <div id="result" >
                        <!-- Creamos una tabla para visualizar los datos nuevos
-->
    <b>Elija el dispositivo</b> <br/><br/>
                        <center><table class="table2excel" data-
tableName="Test Table 1">
                            <thead>
                                <tr>
                                    <th id="ind">Estudiante</th>
                                    <th id="ind2">Hora de
ingreso</th>
                                    <th id="ind3">Dispositivo</th>
                                </tr>
                            </thead>
                            <tbody id="tabla">
                            </tbody>
                        </table>
                    </center>
                </div>
                </div>



<div id="numserial">
        <b>MENU</b> <br/><br/>
        <input type="text" name="serie" id="serie" readonly="readonly"
align="center" />
        <input type="button" id="btnatras" value="Atras">
        <br/><br/>
</div>

<div id="configinicial">
```

```
<br/><b>PARAMETROS</b> <br/><br/>

<b>SELECCIÓN DE MASCOTA</b> <br/><br/>
<input type="button" id="btnperro" value="Perro">
<input type="button" id="btngato" value="Gato">
<br/>
<br/><b>TAMAÑO DE MASCOTA</b> <br/><br/>
<input type="button" id="btnpequeno" value="Pequeño">
<input type="button" id="btnmediano" value="Mediano">
<input type="button" id="btngrande" value="Grande">
</div>

<div id="ajustes">
<br/><b>PARAMETROS</b> <br/><br/>
<input type="text" id="mascota" name="mascota" readonly="readonly"
align="center" >
<br/>
<input type="text" id="tamano" name="tamano" readonly="readonly"
align="center">
<br/>
<input type="text" id="edad" name="edad" readonly="readonly" align="center">
<br/>
<input type="text" id="comida" name="comida" readonly="readonly"
align="center">
<br/>
<input type="text" id="ho1" name="ho1" readonly="readonly" align="center">
<br/>
<input type="text" id="ho2" name="ho2" readonly="readonly" align="center">
<br/>
<input type="text" id="ho3" name="ho3" readonly="readonly" align="center">
<br/>
<br/>
<input type="button" id="btneditarp" value="Editar parametros">
```

```
</div>

<div id="modos">
<br/><b>MODO ACTUALIZACION HORARIO</b> <br/><br/>
<input type="button" id="btnmanual" value="Manual">
<input type="button" id="btnauto" value="Automático">
</div>

<div id="contenedor">
<br/><b>HORARIO</b> <br/><br/>
<label> Hora 1:</label>
<input type="number" id="thora1" name="hora1"
      min="0" max="12" placeholder="00">

<input type="number" id="tmin1" name="min1"
     min="0" max="59" placeholder="00" >
<br/>
<label> Hora 2:</label>
<input type="number" id="thora2" name="hora2"
      min="0" max="12" placeholder="00">
<input type="number" id="tmin2" name="min2"
     min="0" max="59" placeholder="00">
<br/>
<label> Hora 3:</label>
<input type="number" id="thora3" name="hora3"
                              min="0" max="12" placeholder="00">
<input type="number" id="tmin3" name="min3"
                              min="0" max="59" placeholder="00">
<br/>
<br/>
<input type="button" id="btnaceptar" value="Aceptar">
</div>
</fieldset>
```

```
<br><br>
</body>

<script>

        var config = {
                apiKey: "AIzaSyCw-zF-A2ZDdnqnNKSYkK5_tP4nwUbaQYw",
                authDomain: "dispensador-de-mascotas.firebaseapp.com",
                databaseURL: "https://dispensador-de-mascotas.firebaseio.com",
                projectId: "dispensador-de-mascotas",
                storageBucket: "dispensador-de-mascotas.appspot.com",
                messagingSenderId: "955466908194",
                appId: "1:955466908194:web:6f1bfd8dbaf98189"
 };

 firebase.initializeApp(config);
        var rootRef = firebase.database().ref();
        var i;
        var j;
        var idserial;
        var calculo = [];
        var calculo2 = [];
        var calculo3;
 var days;
        var link;
        var link2;
        var proyecto="mascotas"
        var tempRef;
 var obj;
        var animal;
        var tamano;
                var superuser;
$('#btn').hide();
```

```
$('#numserial').hide();
$('#btn3').hide();
$('#modos').hide();
$('#result').hide();
$('#ajustes').hide();
$('#configinicial').hide();
$('#contenedor').hide();
$('#btnpequeno').hide();
$('#btnmediano').hide();
$('#btngrande').hide();

$('#btnperro').click(function(){
$('#btnpequeno').show();
$('#btnmediano').show();
$('#btngrande').show();
animal="perro";
});

$('#btngato').click(function(){
$('#btnpequeno').show();
$('#btnmediano').hide();
$('#btngrande').show();
animal="gato";
});

$('#btnpequeno').click(function(){
        $('#modos').show();
        tamano="pequeno";
});

 $('#btnmediano').click(function(){
        $('#modos').show();
  tamano="mediano";
```

```
        });

        $('#btngrande').click(function(){
        $('#modos').show();
    tamano="grande";
        });

        $('#btnauto').click(function(){
   $('#contenedor').hide();
                        if(animal==="perro"){
                        switch(tamano){
                                case 'grande':
                                var a1=3,a2=3,a3=4,a4=5,a5=6,a6=7;
                                obj
={hora1:a1,hora2:a2,hora3:a3,min1:a4,min2:a5,min3:a6,};
                                break;
                                case 'mediano':
                                var a1=3,a2=9,a3=4,a4=5,a5=6,a6=7;
                                obj
={hora1:a1,hora2:a2,hora3:a3,min1:a4,min2:a5,min3:a6,};
                                break;
                                case 'pequeno':
                                var a1=3,a2=3,a3=11,a4=5,a5=6,a6=7;
                         obj
={hora1:a1,hora2:a2,hora3:a3,min1:a4,min2:a5,min3:a6,};
                                break;
                         }
                }else if (animal==="gato") {
                        switch(tamano){
                                case 'grande':
                                 var a1=1,a2=3,a3=11,a4=5,a5=6,a6=7;
                                obj
={hora1:a1,hora2:a2,hora3:a3,min1:a4,min2:a5,min3:a6,};
```

```
                                    break;
                                    case 'pequeno':
                                      var a1=4,a2=3,a3=14,a4=5,a5=6,a6=7;
                                    obj
={hora1:a1,hora2:a2,hora3:a3,min1:a4,min2:a5,min3:a6,};
                                    break;
                                }
                        }
                        tempRef = rootRef.child(link2);
                        tempRef.update(obj);
                        alert("Actualizacion de horario \nHora 1: "+a1+":"+a4
                        +"\nHora 2: "+a2+":"+a5+"\nHora 3: "+a3+":"+a6);

                        $('#pequeno').hide();
                        $('#mediano').hide();
                        $('#grande').hide();
                });



                        $(document).ready(function(){
                                superuser = window.localStorage.getItem("uiduser");
                                console.log(superuser);
                                $('#tabla').empty();
                        $('#result').show();

                                $('#ind').text("ID");
                                $('#ind2').text("KEY");
                                $('#ind3').text("Tag Id");

                                        j=0;
                                        for ( i=0; i<1; i++) {
                                                var nColumnas = $("#tabla tr:last td").length;
```

```
                                        var tempRef =
rootRef.child('Petchow/Usuarios/'+superuser+'/Serial');
                                        //alert(tempRef);
                                        if (tempRef!=""){
                                                tempRef.on("child_added",
function(snapshot, prevChildKey) {
                                                //recuperamos una captura del
objeto leido
                                                var t = snapshot.val();//t
correspondera a cada elemento dentro del nodo Temp
                        if (t.id!=""){
                                                        j=j+1
                                                        //alert(t.id);
                                                        calculo.push(t.id);

        $('#tabla').append('<tr><td>'+t.id+'</td><td>'+t.key+'</td><td>'+  "<button
type='button' " +

                "onclick='productDisplay(this);' " +

                "class='btn btn-default'>" +

                "<span class='glyphicon-remove' />" +

        "Acceder</button>"+'</td></tr>');
                                                }
                                                });
                                        }
                                }

                });

                function productDisplay(ctl) {
```

```
                var row = $(ctl).parents("tr");
                var cols = row.children("td");
                _activeId = $($(cols[0]).
                        children("button")[2]).data("id");
    idserial=$(cols[0]).text();
                        //alert(idserial);
                        var tempRef = rootRef.child("Petchow/id/"+idserial);
                        //alert(tempRef);
                        if (tempRef!=""){
                                tempRef.on("child_added", function(snapshot,
prevChildKey) {
                                //recuperamos una captura del objeto leido
                                var t = snapshot.val();//t correspondera a cada
elemento dentro del nodo Temp
                                        if (t.animal=="perro" || t.animal=="perro"){
                                        $('#ajustes').show();
                                        $('#numserial').show();
                                        $("#serie").val("Serie Dispositivo: "+idserial);
                                        $("#mascota").val("Mascota: "+t.animal);
                                        animal=t.animal;
                                        $("#tamano").val("Tamaño: "+t.tamano);
                                        tamano=t.tamano;
                                        var edad=t.etapa;
                                        if(edad==="ano"){
                                                $("#edad").val("Edad: "+t.edad+"
año(s)");
                                        }else if(edad==="meses"){
                                                $("#edad").val("Edad: "+t.edad+"
mes(s)");
                                        }
                                        $("#comida").val("C. dispensador:
"+t.comidadispensador+" kg");
                                        $("#ho1").val("Hora 1: "+t.hora1+":"+t.min1);
```

```
$("#ho2").val("Hora 2: "+t.hora2+":"+t.min2);
                                        $("#ho3").val("Hora 3: "+t.hora3+":"+t.min3);
                $('#result').hide();
                                }else if (t.animal==null){
                                                alert("No existe parametros
establecidos para este dispositivo \n Favor inicie el dispositivo");
                                }
                                });
                }

                link2="Petchow/id/"+idserial+"/Parametros";
                }

                $('#btnmanual').click(function(){
                $('#contenedor').show();
        });

                $('#btneditarp').click(function(){
         $('#ajustes').hide();
                $('#configinicial').show();
                });

                $('#btnatras').click(function(){
           $('#ajustes').hide();
                $('#modos').hide();
                $('#contenedor').hide();
                $('#result').show();
                $("#mascota").val("");
                $("#tamano").val("");
                $("#edad").val("");
                $("#comida").val("");
                $("#ho1").val("");
                $("#ho2").val("");
```

```
                $("#ho3").val("");
                $('#numserial').hide();
                $('#configinicial').hide();
                $('#pequeno').hide();
                $('#mediano').hide();
                $('#grande').hide();
                });

        $('#btnaceptar').click(function(){

                var a1=document.getElementById('thora1').value;
                var a2=document.getElementById('thora2').value;
                var a3=document.getElementById('thora3').value;
                var a4=document.getElementById('tmin1').value;
                var a5=document.getElementById('tmin2').value;
                var a6=document.getElementById('tmin3').value;

    if(animal==="perro"){
                        switch(tamano){
                                case 'grande':
                                obj
={hora1:a1,hora2:a2,hora3:a3,min1:a4,min2:a5,min3:a6,};
                                break;
                                case 'mediano':
                                obj
={hora1:a1,hora2:a2,hora3:a3,min1:a4,min2:a5,min3:a6,};
                                break;
                                case 'pequeno':
                          obj
={hora1:a1,hora2:a2,hora3:a3,min1:a4,min2:a5,min3:a6,};
                                break;
                          }
                }else if (animal==="gato") {
```

```
                        switch(tamano){
                                case 'grande':
                                obj
={hora1:a1,hora2:a2,hora3:a3,min1:a4,min2:a5,min3:a6,};
                                break;
                                case 'pequeno':
                                obj
={hora1:a1,hora2:a2,hora3:a3,min1:a4,min2:a5,min3:a6,};
                                break;
                        }
                }
                tempRef = rootRef.child(link2);
                tempRef.update(obj);
                alert("Actualizacion de horario \nHora 1: "+a1+":"+a4
                +"\nHora 2: "+a2+":"+a5+"\nHora 3: "+a3+":"+a6);

                $('#pequeno').hide();
                $('#mediano').hide();
                $('#grande').hide();
        });

</script>

</html>
```

Anexo 6: Manual de usuario

Preparación

Paso 1

Instalar la apk en el celular

Paso 2

Abrir la aplicación Petschow y registrar el usuario con correo y contraseña.

Paso 3

Registrar el dispensador en la aplicación celular por medio del escáner QR.

Paso 4

Acceder al menú Registro y registrar la cantidad de comida que se va a colocar en el Dispensador.

Paso 5

Para el encendido se deben seguir los siguientes pasos:

- Colocar un fusible de 2A.
- Conecta el cable de poder al dispensador y a una toma de 110v.

Paso 6

Conectar una manguera a la entrada de agua y llenar el tanque.

Paso 7

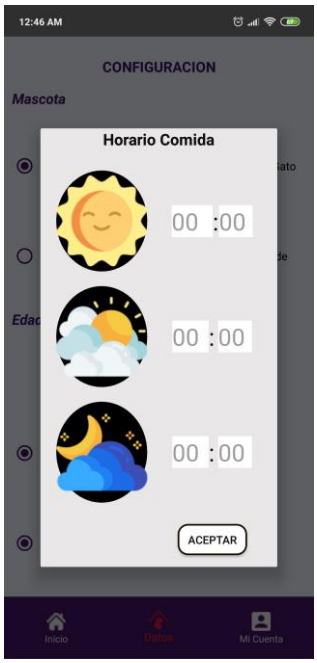
Abrir la tapa del dispensador y colocar la cantidad de comida colocada en la aplicación.

Paso 8

Encender el dispensador y vincular a una red wifi, se podrá observar como se encienden los leds indicadores y la cámara de vigilancia.

Paso 9

Configurar los horarios de comida acorde a su mascota seleccionando las opciones adecuadas.



Paso 10

Vincular la cámara ip y monitorear a su mascota en caso de ser necesario.