**UNIVERSIDAD AUTONOMA GABRIEL**

**RENÉ MORENO**

**FACULTAD DE INGENIERIA EN CIENCIAS DE LA COMPUTACION Y TELECOMUNICACIONES**

**CARRERA: INGENIERIA EN SISTEMAS**





**PROCESO DE DESARROLLO**

**Materia:**

* **Ingenieria de software II**

**Docente:**

* **Ing. Rolando Antonio Martinez Canedo**

**Estudiante:**

* **UNIV. Yaure Alvarez Wilfredo…………213021692**

Santa Cruz –Bolivia

septiembre – 2019

**Contenido**

[**Introducción** 2](#_Toc26181657)

[**CAPITULO 1: INTRODUCCION AL PROYECTO** 3](#_Toc26181658)

[**1.1.** **DEFINICION DEL PROBLEMA** 3](#_Toc26181659)

[**1.1.1.** **Situación problemática** 3](#_Toc26181660)

[**1.1.2.** **Situación deseada** 3](#_Toc26181661)

[**1.2.** **Objetivos** 3](#_Toc26181662)

[**1.2.1.** **Objetivo general** 3](#_Toc26181663)

[**1.2.2.** **Objetivo especifico** 3](#_Toc26181664)

[1.3. **Metodología** 4](#_Toc26181665)

[ **Fase de inicio** 4](#_Toc26181666)

[ **Fase de elaboración** 4](#_Toc26181667)

[ **Fase de construcción** 5](#_Toc26181668)

[ **Fase de Transición** 5](#_Toc26181669)

[**1.4.** **Alcance** 6](#_Toc26181670)

[**CAPITULO 2: AREA PROTEGIDA VALLE TUCAVACA** 7](#_Toc26181671)

[**2.1.** **Antecedente** 7](#_Toc26181673)

[**2.2.** **Administración** 7](#_Toc26181674)

[**2.3.** **Ubicación y extensión** 7](#_Toc26181675)

[**2.4.** **Valores de conservación** 8](#_Toc26181676)

[**2.5.** **Atractivos turísticos** 9](#_Toc26181677)

[**CAPITULO 3: INTELIGENCIA ARTIFICIAL, RECONOCIMIENTO DE IMÁGENES** 10](#_Toc26181678)

[**3.1.** **Inteligencia artificial** 10](#_Toc26181680)

[**3.2.** **¿Por qué es importante la inteligencia artificial?** 11](#_Toc26181681)

[**3.3.** **Cómo funciona el software de reconocimiento de imágenes** 13](#_Toc26181682)

[**3.4.** **Google Visión API** 14](#_Toc26181683)

[**3.4.1.** **Api Visión** 14](#_Toc26181684)

[**3.4.2.** **Ventajas de Google visión Detecta objetos automáticamente** 15](#_Toc26181685)

[**CAPITULO 4: ESPECIFICACION DE REQUISITOS DE SOFTWARE** 17](#_Toc26181686)

[4.1. Introducción 17](#_Toc26181691)

[**4.1.1.** **Propósito** 17](#_Toc26181692)

[**4.1.2.** **Ámbito del software** 17](#_Toc26181693)

[**4.1.3.** **Definición, acrónimos y abreviaturas** 17](#_Toc26181694)

[4.2. Descripción general del proyecto 18](#_Toc26181695)

[**4.2.1.** **Perspectiva del software** 18](#_Toc26181696)

[**4.2.2.** **Funciones del producto** 18](#_Toc26181697)

[**4.2.3.** **Características de los usuarios** 18](#_Toc26181698)

[**4.2.4.** **Restricciones** 18](#_Toc26181699)

[**4.2.5.** **Superposiciones y Dependencias** 19](#_Toc26181700)

[**4.2.6.** **Requisitos futuros** 19](#_Toc26181701)

[4.3. Requerimientos específicos 19](#_Toc26181702)

[**4.3.1.** **Requisitos funcionales** 19](#_Toc26181703)

[**4.3.2.** **Requisitos no funcionales** 25](#_Toc26181704)

[4.4. Identificar actores y casos de uso 26](#_Toc26181705)

[**4.4.1.** **Lista de casos de uso** 27](#_Toc26181706)

[**4.4.2.** **Priorizar casos de uso** 27](#_Toc26181707)

[**CAPITULO 5: ANALISIS** 29](#_Toc26181708)

[5.1. Análisis de la arquitectura 29](#_Toc26181714)

[**5.1.1.** **Identificación de paquetes** 29](#_Toc26181715)

[**5.1.2.** **Vista por paquete** 29](#_Toc26181716)

[5.2. Análisis de Casos de uso 31](#_Toc26181717)

[**5.2.1.** **Registrar Usuario** 31](#_Toc26181718)

[**5.2.2.** **Registrar Raza** 31](#_Toc26181719)

[**5.2.3.** **Registrar Habitad** 32](#_Toc26181720)

[**5.2.4.** **Registrar Especie** 32](#_Toc26181721)

[**5.2.5.** **Registrar Animal** 33](#_Toc26181722)

[**5.2.6.** **Registrar Imagen** 33](#_Toc26181723)

[**5.2.7.** **Gestionar Características** 34](#_Toc26181724)

[**CAPITULO 6: DISEÑO** 35](#_Toc26181725)

[**6.1.** **Análisis de la Arquitectura** 35](#_Toc26181726)

[**6.1.1.** **Identificar Paquetes** 35](#_Toc26181727)

[**6.1.2.** **Diseño de la arquitectura física** 36](#_Toc26181728)

[6.2. Diseñar Caso de Uso 36](#_Toc26181729)

[**Diagrama de Secuencia** 36](#_Toc26181730)

[**Registrar Usuario** 36](#_Toc26181731)

[**Registrar Raza** 37](#_Toc26181732)

[**Registrar Habitad** 38](#_Toc26181733)

[**Registrar Especie** 38](#_Toc26181734)

[**Registrar Animal** 39](#_Toc26181735)

[**Registrar Imagen** 39](#_Toc26181736)

[**Gestionar Características** 40](#_Toc26181737)

[6.3. Diseño de datos 41](#_Toc26181738)

[6.3.1. **Diseño conceptual de la base de datos** 41](#_Toc26181739)

[**6.3.2.** **Diseño lógico mapeo** 41](#_Toc26181740)

[**6.3.3.** **Diseño físico** 43](#_Toc26181741)

[**CAPITULO 7: IMPLEMENTACION** 48](#_Toc26181742)

[7.1. HERRAMIENTAS Y PLATAFORMA DE DESARROLLO 48](#_Toc26181743)

[7.1.1. LENGUAJES DE PROGRAMACION 48](#_Toc26181744)

[7.1.2. BASE DE DATOS 51](#_Toc26181745)

[7.1.3. HERRAMIENTAS CASE UML 52](#_Toc26181746)

[**CONCLUSION** 53](#_Toc26181747)

# **Introducción**

La forma de vida de los animales tiene varios componentes, entre ellos horas de actividad, organización social, espacio vital, alimentación y defensa.

Algunos animales son diurnos, es decir que caminan buscando comida de día. Por ejemplo tenemos mamíferos que son diurnos; aves como la paloma y la paraba, e insectos como las abejas que son diurnos también. Otros animales son nocturnos, es decir que caminan de noche.

Hay animales que andan siempre en grupos o en tropas de varios individuos, machos y hembras adultos, jóvenes y crías: por ejemplo, los chanchos de monte (taitetú, tropero y solitario) y los loros.

Hay también animales que andan siempre solos, o en parejas de macho y hembra, o una hembra con su cría. Por ejemplo el león (puma), el tigre (jaguar), el gato casero, el zorro, el puerco espín, el anta, la urina, la garza.

Muchos animales definen un territorio que puede ser individual o del grupo.

Las aves son más conocidas por su costumbre de construir nidos en el suelo o en árboles. La mayoría de los nidos son solitarios, construidos por una sola pareja. Pero, los loros y las parabas construyen grandes nidos juntos donde duermen varias parejas.

Cada animal tiene necesidad de comida, agua y escondites.

Para defenderse, los animales tienen una gran variedad de métodos

# **CAPITULO 1: INTRODUCCION AL PROYECTO**

## **DEFINICION DEL PROBLEMA**

### **Situación problemática**

En la actualidad, los turistas que visitan el valle de Tucavaca en Bolivia no cuentan con la información necesaria acerca de los animales, que puedan aparecen en su camino, por lo que cuando pillan un animal no saben qué acciones tomar, normalmente reaccionar de la forma más agresiva posible, provocando que el animal pueda entrar en pánico haciendo que el animal se asuste, lo cual al momento de escapar el animal pueda provocar daños al turista.

### **Situación deseada**

Brindar información al turista o persona común, de las características mas sobresalientes de los animales que existen en el valle de tucavaca, clasificándolos por especies, lograr informar acerca de las acciones a tomar en caso de ser atacado y dañado el turista.

## **Objetivos**

### **Objetivo general**

Desarrollar un software de reconocimiento de animales, que brinde información acerca de los animales, para los turistas que visiten el valle de Tucavaca.

### **Objetivo especifico**

Los objetivos específicos son:

* Recabar información acerca de los animales para objeto de prueba.
* Recolectar información de los distintos tipos de animales que existan, para así no tener fallas con los nombres, características de los animales.
* Analizar la información obtenida en la investigación.
* Recabar información de la api a utilizar para el procesamiento de la imagen.
* Diseñar una arquitectura, de tal manera que este sea soportado por el software.
* Implementar la arquitectura diseñada a través, de un lenguaje de programación.
* Realizar las pruebas necesarias para que el software no presente fallas al momento de la revisión del proyecto.

## **Metodología**

La metodología a utilizar como guía base para el desarrollo del software es el modelo del Proceso Unificado de Desarrollo de Software. Este modelo es ágil y flexible para gestionar el desarrollo de software, se basa la planificación y el diseño de software para su correcta construcción. A continuación, veremos las Fase, Flujos de trabajo y Actividades correspondiente al P.U.D.S. que se llevará a cabo durante el desarrollo del sistema. (Jacobson, Booch, & Rumbaugh, 2000)

### **Fase de inicio**

* **Captura de requisitos**
* Realizar entrevistas a los expertos en animales para recolectar información.
* Identificar y describir actores y casos de uso.
* Priorizar casos de uso.
* Detallar casos de uso.
* Realizar los prototipos de interfaces de los casos de uso e identificarlos.

**Resultado:** modelo de dominio, modelo de casos de uso.

### **Fase de elaboración**

* **Análisis**
* Análisis de paquetes.
* Análisis de casos de uso.
* Análisis de arquitectura.

**Resultado:** Especificación de requisitos de software.

* **Diseño**
* Diseño de la arquitectura.
* Diseño de casos de uso.
* Diseño de datos.

**Resultado:** Descripción de diseño de software.

### **Fase de construcción**

* **Implementación**
* Traducción de los modelos de diseño en leguaje de programación.

**Resultado:** Modelo de implementación, código fuente.

### **Fase de Transición**

* Prueba
* Comprobación del correcto funcionamiento analizando diversos aspectos: los objetos como unidades, la integración entre objetos, la implementación de todos los requisitos, entre otros.

**Resultado:** Modelo de prueba, describe como ha sido probado el sistema.

## **Alcance**

El software estará compuesto por 3 módulos:

* **Usuario**

Modulo encargado de registrar a las personas que brindaran información real sobre los animales que existen en el área protegida.

* **Procesamiento**

Modulo donde se podrá realizar el proceso de identificación del animal, mediante el uso de imágenes obtenidas.

* **Animales**

Modulo donde se podrá observar los distintos animales registrados en la base de datos del software.

# **CAPITULO 2: AREA PROTEGIDA VALLE TUCAVACA**



# **Antecedente**

Ubicado a 430 kilómetros al Este de la ciudad de Santa Cruz, la Reserva de Vida Silvestre Tucabaca es calificada como una de las reservas naturales más hermosas, tanto por la diversidad de flora y fauna, pero también por la belleza escénica que ofrecen sus serranías y donde los visitantes tienen la sensación de estar muy cerca del cielo. (Santa Cruz)

El año 2000, mediante O.M. N°013/2000, el municipio de Roboré crea la “Reserva Municipal del Valle de Tucavaca.” Luego mediante Resolución Prefectural N°08/2006 se declara “Reserva Departamental Valle de Tucavaca.” El 2009 mediante O.M. Nº 32/2009 es declarada Área Protegida Municipal y finalmente ratificada mediante Ley Municipal 001/2011 como “Reserva Municipal de Vida Silvestre Valle de Tucabaca.” Ahora, se la conoce como “Reserva de Vida Silvestre Tucabaca” por la O. M. 018/2013, que aprueba la actualización de su Plan de Manejo.

# **Administración**

La administración del área es responsabilidad del Gobierno Departamental de Santa Cruz a través de la Dirección de Áreas Protegidas, en coordinación con el Gobierno Municipal de Roboré.

# **Ubicación y extensión**

El Área Protegida se encuentra en el Municipio de Roboré, a 430 km. de la ciudad de Santa Cruz y tiene una extensión de 269.718 ha.

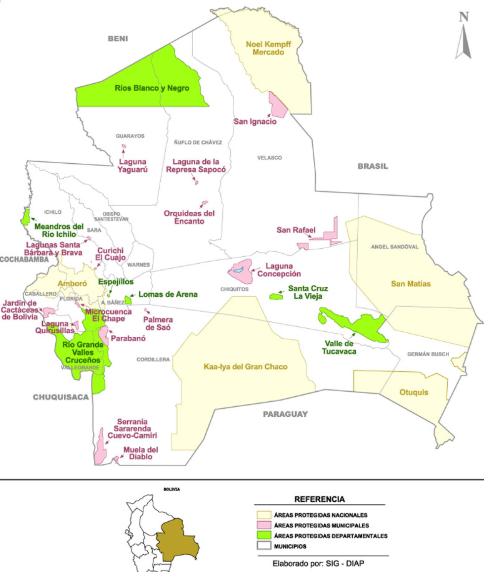


Ilustración : Ubicacion del valle tucavaca

**Fuente** (SIG-DIAP)

# **Valores de conservación**

La Reserva Departamental, protege las fuentes de agua en la serranía de Santiago y los cursos que forman la base del sistema hídrico regional; protege también el ejemplo mejor preservado del bosque seco tropical a nivel mundial, con toda la flora y fauna característica y los recursos arqueológicos existentes en la zona. A pesar de los escasos estudios realizados en cuanto a investigaciones de flora y fauna, se estima que existen más de 1500 especies de plantas características, como ser: tajibo, curupaú, verdolago, pesoé, entre otras. De la misma forma cuenta con una alta diversidad de mamíferos y especies definidas de este tipo de bosque.

# **Atractivos turísticos**

Esta zona cuenta con lugares paisajísticos de gran belleza entre los que se destacan: Santiago de Chiquitos, una de las últimas misiones fundada por los Jesuitas, en el corazón de la reserva; la Serranía de Santiago con sus farallones y formaciones rocosas que contienen numerosas sendas y miradores; la cascada de los helechos; Motacú con las pinturas rupestres; Cuevas de Miserandino, el Arco grande, el Chorro de San Luís y otros sitios de espectacular belleza, estos valores se ven potenciados por la accesibilidad del ferrocarril y por la carretera interoceánica.

# **CAPITULO 3: INTELIGENCIA ARTIFICIAL, RECONOCIMIENTO DE IMÁGENES**



# **Inteligencia artificial**

El término inteligencia artificial fue adoptado en 1956, pero se ha vuelto más popular hoy día gracias al incremento en los volúmenes de datos, algoritmos avanzados, y mejoras en el poder de cómputo y el almacenaje. (SAS, 2019)

La investigación inicial de la inteligencia artificial en la década de 1950 exploraba temas como la solución de problemas y métodos simbólicos. En la década de 1960, el Departamento de Defensa de los Estados Unidos mostró interés en este tipo de trabajo y comenzó a entrenar computadoras para que imitaran el razonamiento humano básico. Por ejemplo, la Defense Advanced Research Projects Agency (DARPA, Agencia de Proyectos de Investigación Avanzada de Defensa) realizó proyectos de planimetría de calles en la década de 1970. Y DARPA produjo asistentes personales inteligentes en 2003, mucho tiempo antes que Siri, Alexa o Cortana fueran nombres comunes.

Este trabajo inicial abrió el camino para la automatización y el razonamiento formal que vemos hoy en las computadoras, incluyendo sistemas de soporte a decisiones y sistemas de búsqueda inteligentes que pueden ser diseñados para complementar y aumentar las capacidades humanas.

Aunque las películas de Hollywood y las novelas de ciencia ficción representan la inteligencia artificial como robots semejantes a humanos que se apoderan del mundo, la evolución actual de las tecnologías IA no es tan aterradora – o así de inteligente. En su lugar, la inteligencia artificial ha evolucionado para brindar muchos beneficios específicos a todas las industrias. Siga leyendo para conocer ejemplos modernos de inteligencia artificial en las áreas de atención a la salud, comercio detallista y más.



Ilustración 2: Historia de la inteligencia artificial

Fuente (SAS, 2019)

# **¿Por qué es importante la inteligencia artificial?**

* La inteligencia artificial automatiza el aprendizaje y descubrimiento repetitivos a través de datos. La inteligencia artificial es diferente de la automatización de robots basada en hardware. En lugar de automatizar tareas manuales, la inteligencia artificial realiza tareas computarizadas frecuentes de alto volumen de manera confiable y sin fatiga. Para este tipo de automatización, la investigación humana sigue siendo fundamental para configurar el sistema y hacer las preguntas correctas.
* IA agrega inteligencia a productos existentes. En la mayoría de los casos, la inteligencia artificial no se venderá como aplicación individual. En su lugar, los productos que ya utiliza serán mejorados con recursos de inteligencia artificial, de forma muy similar en que se agregó Siri como característica a una nueva generación de productos de Apple. La automatización, las plataformas conversacionales, los bots y las máquinas inteligentes se pueden combinar con grandes cantidades de datos para mejorar muchas tecnologías en el hogar y en el lugar de trabajo, desde inteligencia de seguridad hasta análisis de las inversiones.
* La inteligencia artificial se adapta a través de algoritmos de aprendizaje progresivos para permitir que los datos llevan a cabo la programación. La inteligencia artificial encuentra estructura y regularidades en los datos de modo que el algoritmo adquiera una habilidad: el algoritmo se convierte en un clasificador o en un indicador. De tal suerte que, así como el algoritmo puede enseñarse a jugar ajedrez, también puede enseñarse qué producto recomendar después en línea. Y los modelos se adaptan cuando se les proporcionan nuevos datos. La propagación retrasada es una técnica de inteligencia artificial que permite al modelo hacer ajustes, a través de entrenamiento y datos agregados, cuando la primera respuesta no es correcta.
* La inteligencia artificial analiza más datos y datos más profundos utilizando redes neurales que tienen muchas capas ocultas. Construir un sistema de detección de fraude con cinco capas ocultas era casi imposible hace unos años. Todo eso ha cambiado con increíble poder de cómputo y el big data. Usted necesita muchos datos para entrenar modelos de aprendizaje a fondo porque éstos aprenden directamente de los datos. Cuantos más datos pueda integrarles, más precisos se vuelven.
* La inteligencia artificial logra una precisión increíble a través de redes neurales profundas – lo cual antes era imposible. Por ejemplo, sus interacciones con Alexa, Google Search y Google Photos están basadas en el aprendizaje a fondo – y se siguen volviendo más precisas cuanto más las utilizamos. En el terreno de la medicina, ahora se pueden emplear técnicas de inteligencia artificial de aprendizaje a fondo, clasificación de imágenes y reconocimiento de objetos para detectar cáncer en imágenes por resonancia magnética con la misma precisión que radiólogos altamente especializados.
* La inteligencia artificial saca el mayor provecho de los datos. Cuando los algoritmos son de autoaprendizaje, los datos mismos pueden volverse de propiedad intelectual. Las respuestas están en los datos; usted sólo tiene que aplicar inteligencia artificial para sacarlas a la luz. Como el rol de los datos es ahora más importante que nunca antes, pueden crear una ventaja competitiva. Si tiene los mejores datos en una industria competitiva, incluso si todos aplican técnicas similares, los mejores datos triunfarán.

# **Cómo funciona el software de reconocimiento de imágenes**

La visión por ordenador es lo que potencia la capacidad de un escáner de código de barras para "ver" un montón de franjas en una UPC. También es la forma en que la aplicación de identidad de Apple puede saber si la cara que está mirando la cámara del dispositivo es la de su propietario o no. Básicamente, cada vez que una máquina procesa información visual sin procesar, como un archivo JPEG o una fuente de alimentación de la cámara, utiliza la visión artificial para comprender lo que está viendo.

Un programa de reconocimiento de imágenes actúa como la parte del cerebro humano que procesa la información que reciben los ojos, no como los ojos en sí.

Uno de los usos más interesantes de la visión de computadora, desde el punto de vista de la inteligencia artificial, es el reconocimiento de imágenes, que le da al equipo informático la capacidad de interpretar lo que la visión del ordenador registra, pudiendo clasificarlo. Es la esencia del reconocimiento de imágenes.

# **Google Visión API**

## **Api Visión**

La API Visión de Google Cloud ofrece modelos de aprendizaje automático preparados previamente y muy potentes a través de las API REST y RPC. Asigna etiquetas a imágenes y clasifícalas rápidamente en millones de categorías predefinidas. Detecta objetos y caras, lee texto impreso y manuscrito, y consigue metadatos de gran valor para tu catálogo de imágenes (Google Cloud, s.f.)

**¿Cómo funciona Google Visión API?,**

Básicamente Google tiene indexada una gran colección de imágenes que ha venido creciendo a lo largo de su historia. Estas imágenes son el insumo básico para un proceso riguroso de clasificación que se ha venido mejorando cada día en Google, esto permite entonces que una vez que una imagen necesita ser analizada esta API tratará de encontrar imágenes similares y obtener los metadatos de estas imágenes similares. Por ejemplo si subes una imagen de tu perro o tu gato primero la API tratará de encontrar fotos similares y encontrará por supuesto algunas fotos con perros, luego de ese conjunto de fotos con perros tratará de encontrar algunas fotos con el mismo color, tamaño de las formas entre otras posibles características. Finalmente extraerá las etiquetas de esas imágenes similares y te las presentará como si fueran las de tu imagen. Bien, probablemente a este punto dirás “Por supuesto, esto no es Inteligencia Artificial” y tal vez estés en lo correcto.

Si bien el proceso de clasificación no es un proceso “inteligente”, estrictamente hablando si es un insumo necesario para el aprendizaje de máquina (Machine Learning). También es el insumo para extraer patrones sobre las imágenes y poderlos comparar para generar conocimiento he incluso para generar conocimiento sobre lo aprendido (Deep Machine Learning), todos estos procesos hacen parte de la Google Vision API con agravante adicional, la potencia de computo de los servidores de Google (tal vez los más poderosos del mundo) han venido trabajando desde hace algún tiempo detectando patrones sobre las imágenes, procesando millones de imágenes cada día y aprendiendo de lo que nosotros como humanos creemos que hay en una imagen. (Gomez, 2017)

## **Ventajas de Google visión Detecta objetos automáticamente**

Detecta y clasifica distintos objetos, y consigue información como la posición de cada uno de ellos en la imagen. Consulta más información sobre la detección de objetos con la API Vision y AutoML Vision. (Google Cloud, s.f.)

Consigue información valiosa con el perímetro

Con AutoML Vision Edge puedes crear y desplegar modelos rápidos y de gran precisión para clasificar imágenes en el perímetro, así como desencadenar acciones en tiempo real en función de los datos locales. AutoML Vision Edge es compatible con diversos dispositivos perimetrales en caso de que se cuente con recursos limitados y la latencia sea muy importante. (Google Cloud, s.f.)

Aumenta la fluidez del proceso de venta

Gracias a la búsqueda visual de productos de la API Vision, los comercios pueden crear experiencias móviles atractivas que permitan a sus clientes subir la foto de un objeto y ver de inmediato una lista de objetos similares que pueden comprar. (Google Cloud, s.f.)

Comprende el texto y actúa en consecuencia

La API Vision utiliza la tecnología de reconocimiento óptico de caracteres (OCR) para detectar texto en imágenes en más de 50 idiomas y diferentes tipos de archivo. También forma parte de IA para interpretar documentos, que permite procesar rápidamente millones de documentos y automatizar los flujos de trabajo de los negocios. (Google Cloud, s.f.)

Detecta contenido explícito

La API Vision puede revisar tus imágenes mediante la función de búsqueda segura y estimar la probabilidad de que cualquiera de ellas incluya contenido para adultos, contenido violento o similares. (Google Cloud, s.f.)

Usa nuestro servicio de etiquetado de datos

Si tienes imágenes para AutoML Vision que todavía no se hayan etiquetado, Google cuenta con un equipo de personas que pueden ayudarte a anotar imágenes, vídeos y texto para conseguir datos de preparación de alta calidad. Más información. (Google Cloud, s.f.)

# **CAPITULO 4: ESPECIFICACION DE REQUISITOS DE SOFTWARE**



# Introducción

En la presente sección del documento se procederá a detallar los requisitos del software de distintos mecanismos.

## **Propósito**

El presente documento tiene como propósito definir las especificaciones funcionales, no funcionales para el desarrollo de un software de reconocimiento de imágenes, que permitirá reconocer las distintas especies de animales registrados en la base de datos. Éste será utilizado por cualquier usuario que tenga la aplicación móvil.

## **Ámbito del software**

El presente proyecto ha sido con el propósito de brindar información acerca de los animales, en cualquier lugar que se encuentre el usuario que tenga la aplicación del software, de esta manera el usuario podrá estar enterado en tiempo real de que tipo de animal se le aparece en el camino.

## **Definición, acrónimos y abreviaturas**

|  |  |
| --- | --- |
| ***Nombre*** | ***Descripción*** |
| **Usuario** | Persona que usará el software |
| **IA** | Inteligencia artificial |
| **API** | Application Programming Interface(interfaz de programación de aplicaciones) |
| **ERS** | Especificación de Requisitos Software |
| **RF** | Requerimiento Funcional |
| **RNF** | Requerimiento No Funcional |

# Descripción general del proyecto

Esta sección nos presenta una descripción general del software con el fin de conocer la función que debe soportar, los datos asociados, las restricciones impuestas y cualquier otro factor que pueda influir en la construcción del mismo.

## **Perspectiva del software**

El software a realizar tiene una dependencia de un sistema de registro, dado que primero se tiene que tener los registros de los animales para posteriormente realizar las validaciones.

## **Funciones del producto**

Las funciones de la aplicación son permitir a los entes encargados a subir información certera y confiable a la aplicación, como función principal es la detección de animales en cualquier lugar que uno se encuentre.

* **Atenticación:** La autenticación a la aplicación web se tendrá como opción a través de un login y contraseña. Cada cuenta de usuario será diferente.
* **Registro de imágenes:** El registro de imágenes será llevado a cabo por personas que tengan conocimiento de los animales.
* **Verificación de imágenes:** La verificación de imágenes será llevado a cabo con las imágenes cargadas al software.

## **Características de los usuarios**

**A1: Administrador. -** Encargado de administrar el suministro de imágenes al software.

**A2: Veterinario. -** Encargado de la verificación de que los datos de los animales que se están registrado sean los correctos.

**A3: Cliente. -** Usuario final el cual podrá utilizar el software.

## **Restricciones**

Las restricciones son las siguientes:

* Por restricciones del Gobierno el software a desarrollarse será implementado utilizando únicamente tecnologías de código abierto.
* El software deberá cumplir con estándares de codificación.
* Por restricción del Framework la aplicación solo asegura su correcta funcionalidad en dispositivos multiplataforma.
* La aplicación se diseñará según el patrón de arquitectura MVC modelo vista controlador.

## **Superposiciones y Dependencias**

Se han definido las siguientes suposiciones:

* Estará implementada en un servidor web con google visión.
* Se implementará un modelo independiente, pero se realizará las pruebas sobre un gestor PostgreSQL, MYSQL y SQL-Server.
* Los dispositivos móviles cuentan con un Sistema operativo Android y se deberá activar dicho servicio para que funcione mediante servicios web.

## **Requisitos futuros**

En el proyecto a futuro se piensa aumentar nuevas funciones como ser:

* Registro de animales en peligro de extinción, con este módulo se podría identificar si un animal está en peligro de extinción para así de esta forma tratar de preservar los animales en peligro de desaparecer.
* Geolocalización de los animales mediante mapa de google maps, identificar las zonas por donde vive el animal o puede vivir dicho animal.

# Requerimientos específicos

## **Requisitos funcionales**

En este punto encapsulamos y detallamos los requisitos funcionales que tenemos.

**RF01:** El administrador podrá acceder al software mediante su correo y contraseña asignado para que ellos se encarguen de la administración de las imágenes a subir al software.

****

|  |  |
| --- | --- |
| Identificador del Requisito | RF.01 |
| Nombre del Requisito | **Gestionar Usuario** |
| Características | Registra y administra a los usuarios, para el buen manejo del software. |
| Descripción del Requisito | Es necesario que el usuario a registrarse debe cumplir con algunos requisitos que se le puedan pedir para un buen control de los usuarios a registrar. |
| Prioridad | Alta |

**RF02:** El veterinario será el encargado de realizar el registro de la raza de los animales, debido a que él tiene más conocimiento de las razas que puedan existir de un animal.



|  |  |
| --- | --- |
| Identificador del Requisito | RF.02 |
| Nombre del Requisito | **Registrar raza** |
| Características | Registrar las diferentes razas de un animal, que pueda existir en distintos lugares. |
| Descripción del Requisito | Es necesario que el veterinario tenga un amplio conocimiento de las razas de los animales, para poder registrar datos correctos en el software. |
| Prioridad | Alta |

**RF03:** En este módulo se podrá registrar los distintos habitas de los animales, con este módulo se podrá saber los lugares por donde puede vivir un animal.



|  |  |
| --- | --- |
| Identificador del Requisito | RF.03 |
| Nombre del Requisito | **Registrar habitad** |
| Características | El veterinario será el encargado de realizar el registro de habitad de un animal en específico, des esta manera se podrá observar por donde vive el animal. |
| Descripción del Requisito | Cada vez los animales emigran a distintos lugares por lo cual es necesario saber en qué zonas viven los animales. |
| Prioridad | Alta |

**RF04:** En este módulo se podrá clasificar al animal, de acuerdo al grado de agresividad que presente, con el fin de tener cuidado al momento de observar estos animales.



|  |  |
| --- | --- |
| Identificador del Requisito | RF.04 |
| Nombre del Requisito | **Registrar clasificación** |
| Características | En este módulo se podrá registrar la clasificación de acuerdo al grado de agresividad del animal para una mejor precaución al momento de observarlos. |
| Descripción del Requisito | Normalmente cuando nos dirigimos a un bosque no sabemos el grado de agresividad del animal, podemos pensar que son inofensivos, es por eso que se clasificara a los animales de acuerdo a su grado de agresividad. |
| Prioridad | Alta |

**RF05:** En este modulo se registrara las distintas especies de animales que existen, el encargado de realizar estos registros será el veterinario, el cual contara con la información necesaria de las especies existentes.



|  |  |
| --- | --- |
| Identificador del Requisito | RF.05 |
| Nombre del Requisito | **Registrar especie** |
| Características | En este módulo se registrara las distintas especies de animales que existen, de esta forma categorizar a los animales por especies. |
| Descripción del Requisito | En los bosques normalmente encontramos distintos tipos de animales los cuales no sabemos a qué especie pertenecen, este módulo nos podrá dar información a que especie pertenece un animal. |
| Prioridad | Alta |

**RF06:** En este módulo se registrará a los distintos animales que se conozcan, de esta manera se almacenara datos en el servidor de la base de datos para su posterior análisis de los animales, brindar información acerca de ellos a los usuarios finales quienes serán nuestros clientes.



|  |  |
| --- | --- |
| Identificador del Requisito | RF.06 |
| Nombre del Requisito | **Gestionar animal** |
| Características | En este módulo se registrara los distintos animales en el área de estudio, con esta información se apreciara que tipo de animales existen. |
| Descripción del Requisito | Normalmente no se tiene el conocimiento de todos los animales en un reservorio, se tratará de registrar la mayor cantidad de animales que puedan existir en el área. |
| Prioridad | Alta |

**RF07:** Este módulo será el encargado de subir las imágenes de los animales, de esta forma se podrá tener una extensa base de datos para el software, también estas imágenes servirán para entrenar al api de reconocimiento de animales.



|  |  |
| --- | --- |
| Identificador del Requisito | RF.06 |
| Nombre del Requisito | **Registrar imagen** |
| Características | En este módulo se registrara las imágenes de los animales, se almacenara en un servidor todas las imágenes. |
| Descripción del Requisito | El encargado de subir estas imágenes será el administrador por motivos de seguridad y no puedan subir cualquier tipo de imágenes al servidor. |
| Prioridad | Alta |

**RF08:** En este módulo se registrar las distintas características de los animales para una mejor apreciación acerca de ellos.



|  |  |
| --- | --- |
| Identificador del Requisito | RF.06 |
| Nombre del Requisito | **Gestionar características** |
| Características | En este módulo se registrara las características más sobresalientes de los animales. |
| Descripción del Requisito | El encargado de realizar el registro de las características de los animales será el administrador junto con el veterinario, de esta forma tener una veracidad más acertada de las características de los animales. |
| Prioridad | normal |

## **Requisitos no funcionales**

* **Requisito de calidad**

El software responderá a la validación en tiempos de respuestas, en búsqueda, al procesar la imagen brindará la información más certera acerca la imagen a procesar.

* **Requisito de hardware y software**

El software funcionara correctamente con el hardware y software recomendados:

* + Procesador igual o mayor Intel 3 o AMD A6, Memoria RAM de 4 GB, Almacenamiento requerido de 1 GB.
  + Se ejecutará en el sistema operativo Windows 7 en adelante, en versiones 64 bits.
  + En el móvil la versión de Android será mayor igual a Android 7
* **Requisito de seguridad**

La información manejada por el software está protegida de acceso no autorizado y divulgación. Brindado confidencialidad, integridad y disponibilidad de la información a los usuarios autorizados.

* **Requisito de usabilidad**

El software está orientado para un usuario de nivel intermedio en manejo de la computación.

* **Requisito de restricción**

El administrador podrá visualizar la información registrada en el software.

El veterinario tendrá acceso a algunas funcionalidades del software por motivos de seguridad.

# Identificar actores y casos de uso

|  |  |  |
| --- | --- | --- |
| Usuario | Características | Descripción |
| Administrador | Conocimiento acerca del software , conocimiento de la información a ser procesado en el servidor | Es el encargado de administrar el sistema deberá tener mínimo conocimientos informáticos. |
| Veterinario | Conocimiento en el área de los animales. | Encargada de registrar datos acerca de los animales que ser registren en el software. |
| Cliente | No necesitan conocimientos técnicos | El cliente usuario final para el que va dirigido el software |

## **Lista de casos de uso**

CU1: Gestionar usuario

CU2: Registrar Raza

CU3: Registrar habitad

CU4: Registra clasificación

CU5: Registrar especie

CU6: Gestionar animal

CU7: Registrar imagen

CU8: Gestionar Características

## **Priorizar casos de uso**

|  |  |  |
| --- | --- | --- |
| Casos de Uso | Estado | Prioridad |
| CU1:Gestionar usuario | Aprobado | Critico |
| CU2: Registrar raza | Aprobado | Critico |
| CU3:Registrar habitad | Aprobado | Critico |
| CU4:Registrar clasificación | Aprobado | Critico |
| CU5:Registrar especie | Aprobado | Critico |
| CU6:Gestionar animal | Aprobado | Normal |
| CU7: Registrar imagen | Aprobado | Critico |
| CU8: Gestionar características | Aprobado | Normal |

# **CAPITULO 5: ANALISIS**



# Análisis de la arquitectura

## **Identificación de paquetes**

**Paquete de usuario**

En este paquete se realizarán todos los procesos realizados por el administrador, veterinario, con el fin de precautelar la información del software.

**Paquete de galería**

En este paquete se podrá observar todas las imágenes de los animales registrados en el software, estarán disponibles para los clientes finales.

## **Vista por paquete**

**Paquete Usuario**

****

**Paquete Galeria**

****

# Análisis de Casos de uso

## **Registrar Usuario**



## **Registrar Raza**



## **Registrar Habitad**



## **Registrar Especie**



## **Registrar Animal**



## **Registrar Imagen**



## **Gestionar Características**



# **CAPITULO 6: DISEÑO**

# **Análisis de la Arquitectura**

## **Identificar Paquetes**



## **Diseño de la arquitectura física**



# Diseñar Caso de Uso

## **Diagrama de Secuencia**

## **Registrar Usuario**

## **Registrar Raza**



## **Registrar Habitad**



## **Registrar Especie**



## **Registrar Animal**



## **Registrar Imagen**



## **Gestionar Características**



# Diseño de datos

## **Diseño conceptual de la base de datos**



## **Diseño lógico mapeo**

Raza

|  |  |  |
| --- | --- | --- |
| Id  **(PK)** | Nombre | Descripción |

Habitad

|  |  |  |  |
| --- | --- | --- | --- |
| Id  **(PK)** | Nombre | Clima | Tipo\_vegetacion |

Especie

|  |  |  |
| --- | --- | --- |
| Id  **(PK)** | Nombre | Descripción |

Animal

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Id  **(PK)** | Nombre | Tiempo\_vida | idR  **(FK)** | idE  **(FK)** |

Característica

|  |  |  |
| --- | --- | --- |
| Id  **(PK)** | Descripción | idA  **(FK)** |

Imagen

|  |  |  |
| --- | --- | --- |
| Id  **(PK)** | Nombre | idA  **(FK)** |

Detalle

|  |  |  |
| --- | --- | --- |
| Id  **(PK)** | idH  **(FK)** | idE  **(FK)** |

## **Diseño físico**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Raza |  | | | |
| **Atributo** | **Tipo de dato** | **Longitud** | **Nulo** | **Llave** |
| Id | BigIncrement | 20 | No | PK |
| Nombre | Texto | 200 | No |  |
| Descripcion | Texto | 200 | No |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Habitad |  | | | |
| **Atributo** | **Tipo de dato** | **Longitud** | **Nulo** | **Llave** |
| Id | BigIncrement | 20 | No | PK |
| Nombre | Texto | 200 | No |  |
| Tipo\_Vegetacion | Texto | 200 | No |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Especie |  | | | |
| **Atributo** | **Tipo de dato** | **Longitud** | **Nulo** | **Llave** |
| Id | BigIncrement | 20 | No | PK |
| Nombre | Texto | 200 | No |  |
| Descripcion | Texto | 200 | no |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Animal |  | | | |
| **Atributo** | **Tipo de dato** | **Longitud** | **Nulo** | **Llave** |
| Id | BigIncrement | 20 | No | PK |
| Nombre | Text | 200 | No |  |
| Tiempo\_Vida | Text | 200 | No |  |
| IdR | Integer | 20 | No | Fk |
| IdE | Integer | 20 | No | FK |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Caracteristica |  | | | |
| **Atributo** | **Tipo de dato** | **Longitud** | **Nulo** | **Llave** |
| Id | BigIncrement | 20 | No | PK |
| Descripcion | Text | 200 | No |  |
| IdA | Integer | 20 | No | FK |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Imagen |  | | | |
| **Atributo** | **Tipo de dato** | **Longitud** | **Nulo** | **Llave** |
| Id | BigIncrement | 20 | No | PK |
| Nombre | Text | 200 | No |  |
| IdA | Integer | 20 | No | FK |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Detalle |  | | | |
| **Atributo** | **Tipo de dato** | **Longitud** | **Nulo** | **Llave** |
| Id | BigIncrement | 20 | No | PK |
| IdH | Integer | 20 | No | FK |
| IdE | Integer | 20 | No | FK |

**Script**

CREATE TABLE `animal` (

`id` bigint(20) UNSIGNED NOT NULL,

`nombre` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`tiempoVida` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`idR` bigint(20) UNSIGNED NOT NULL,

`idE` bigint(20) UNSIGNED NOT NULL,

`created\_at` timestamp NULL DEFAULT NULL,

`updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `caracteristica`

--

CREATE TABLE `caracteristica` (

`id` bigint(20) UNSIGNED NOT NULL,

`caracteristica` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`idA` bigint(20) UNSIGNED NOT NULL,

`created\_at` timestamp NULL DEFAULT NULL,

`updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `detalle`

--

CREATE TABLE `detalle` (

`id` bigint(20) UNSIGNED NOT NULL,

`idH` bigint(20) UNSIGNED NOT NULL,

`idE` bigint(20) UNSIGNED NOT NULL,

`created\_at` timestamp NULL DEFAULT NULL,

`updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `especie`

--

CREATE TABLE `especie` (

`id` bigint(20) UNSIGNED NOT NULL,

`nombre` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`descripcion` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`created\_at` timestamp NULL DEFAULT NULL,

`updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `habitad`

--

CREATE TABLE `habitad` (

`id` bigint(20) UNSIGNED NOT NULL,

`nombre` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`clima` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`tipoVegetacion` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`created\_at` timestamp NULL DEFAULT NULL,

`updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `imagen`

--

CREATE TABLE `imagen` (

`id` bigint(20) UNSIGNED NOT NULL,

`nombre` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`idA` bigint(20) UNSIGNED NOT NULL,

`created\_at` timestamp NULL DEFAULT NULL,

`updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `raza`

--

CREATE TABLE `raza` (

`id` bigint(20) UNSIGNED NOT NULL,

`nombre` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`descripcion` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`created\_at` timestamp NULL DEFAULT NULL,

`updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

-- --------------------------------------------------------

--

-- Estructura de tabla para la tabla `users`

--

CREATE TABLE `users` (

`id` bigint(20) UNSIGNED NOT NULL,

`ci` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`name` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`email` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`email\_verified\_at` timestamp NULL DEFAULT NULL,

`password` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`foto` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`sexo` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`telefono` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`direccion` varchar(255) COLLATE utf8mb4\_unicode\_ci NOT NULL,

`remember\_token` varchar(100) COLLATE utf8mb4\_unicode\_ci DEFAULT NULL,

`created\_at` timestamp NULL DEFAULT NULL,

`updated\_at` timestamp NULL DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_unicode\_ci;

# **CAPITULO 7: IMPLEMENTACION**

## HERRAMIENTAS Y PLATAFORMA DE DESARROLLO

### LENGUAJES DE PROGRAMACION

Flutter: El SDK de aplicaciones móviles Flutter es una nueva forma de crear hermosas aplicaciones móviles nativas que se separan de las aplicaciones “cookie cutter (predecibles, recicladas, no únicas)” que han sido tan comunes en el pasado. A la gente que prueba Flutter le gusta mucho; por ejemplo, mira [esto](https://medium.com/@aubykhan/how-i-built-a-cross-platform-connected-app-in-7-days-93728a987424), [esto](https://medium.com/@germansaprykin/flutter-first-steps-fffd906f093b) o [esto](https://code.tutsplus.com/tutorials/developing-an-android-app-with-flutter--cms-28270). O aquí hay una [lista de artículos y videos](https://github.com/Solido/awesome-flutter) compilados por un tercero.

Como con cualquier sistema nuevo, la gente quiere saber qué hace diferente a Flutter, o, dicho de otra manera, “¿qué hay de nuevo y emocionante en Flutter?” Esa es una pregunta justa, y este artículo tratará de responderla desde un punto de vista técnico — no sólo qué es emocionante, sino por qué.

Pero primero, un poco de historia.

**Una breve historia sobre el desarrollo de aplicaciones móviles**

El desarrollo de aplicaciones móviles es un campo de trabajo relativamente reciente. Los desarrolladores de terceros han sido capaces de crear aplicaciones móviles hace menos de una década, por lo que no es de extrañar que las herramientas sigan evolucionando.

**Los SDKs de la Plataforma**

El SDK de iOS, de Apple fue lanzado en 2008 y el SDK de Android, de Google en 2009. Estos dos SDK estaban basados en lenguajes diferentes: Objective-C y Java, respectivamente.

Tu aplicación se comunica con la plataforma para crear widgets o acceder a servicios como la cámara. Los widgets se muestran en un canvas de pantalla, y los eventos se devuelven a los widgets. Esta es una arquitectura simple, pero hay que crear aplicaciones separadas para cada plataforma porque los widgets son diferentes, por no hablar de los lenguajes nativos.

**WebViews**(Vistas Web)

Los primeros frameworks multiplataforma se basaron en JavaScript y WebViews. Algunos ejemplos incluyen una [familia de frameworks relacionados](https://en.wikipedia.org/wiki/Apache_Cordova): PhoneGap, Apache Cordova, Ionic y otros. Antes de que Apple lanzara su SDK iOS, animaron a los desarrolladores de terceros a crear aplicaciones web para el iPhone, por lo que la creación de aplicaciones multiplataforma utilizando tecnologías web era un paso obvio.

Al igual que React Native, Flutter también proporciona vistas de estilo reactivo. Flutter tiene un enfoque diferente para evitar los problemas de rendimiento causados por la necesidad de un puente JavaScript mediante el uso de un lenguaje de programación compilado, a saber, [Dart](https://www.dartlang.org/" \t "_blank). Dart se compila “por adelantado” (AOT) en código nativo para múltiples plataformas. Esto permite que Flutter se comunique con la plataforma sin pasar por un puente JavaScript que hace un cambio de contexto. La compilación a código nativo también mejora los tiempos de inicio de la aplicación.

El hecho de que Flutter sea el único SDK para móviles que proporciona vistas reactivas sin necesidad de un puente JavaScript debería ser suficiente para que Flutter sea interesante y merezca la pena intentarlo, pero hay algo mucho más revolucionario en Flutter, y es cómo implementa los widgets.

Compatibilidad

Una de las características más populares de Flutter es su rápida recarga en caliente. Puedes hacer un cambio en una aplicación Flutter mientras se está ejecutando, y recargará el código de la aplicación que ha cambiado y dejará que continúe desde donde la dejaste, a menudo en menos de un segundo. Si tu aplicación encuentra un error, normalmente puedes corregirlo y luego continuar como si el error nunca hubiera ocurrido. Incluso cuando tienes que hacer una recarga completa, es rápido. (Vega, 2019)

PHP : el preprocesador de hipertexto (o simplemente PHP ) es un [lenguaje de programación del lado del servidor](https://en.wikipedia.org/wiki/Server-side_scripting) diseñado para el [desarrollo web](https://en.wikipedia.org/wiki/Web_development) , y también se utiliza como un [lenguaje de programación de propósito general](https://en.wikipedia.org/wiki/General-purpose_programming_language) . Fue creado originalmente por [Rasmus Lerdorf](https://en.wikipedia.org/wiki/Rasmus_Lerdorf" \o "Rasmus Lerdorf) en 1994; la [implementación de referencia de](https://en.wikipedia.org/wiki/Reference_implementation)PHP ahora es producida por The PHP Group. PHP originalmente representaba la página de inicio personal, pero ahora representa el [inicialismo recursivo](https://en.wikipedia.org/wiki/Recursive_initialism" \o "Inicialismo recursivo)PHP: preprocesador de hipertexto.

El código PHP se puede incrustar en el código [HTML](https://en.wikipedia.org/wiki/HTML) , o se puede usar en combinación con varios [sistemas de plantillas](https://en.wikipedia.org/wiki/Web_template_system) web, sistemas de administración de contenido [web](https://en.wikipedia.org/wiki/Web_framework) y [marcos web](https://en.wikipedia.org/wiki/Web_framework) . El código PHP generalmente es procesado por un [intérprete de](https://en.wikipedia.org/wiki/Interpreter_(computing)) PHP implementado como un [módulo](https://en.wikipedia.org/wiki/Plugin_(computing)) en el servidor web o como un ejecutable de [Interfaz de puerta de enlace común](https://en.wikipedia.org/wiki/Common_Gateway_Interface) (CGI). El servidor web combina los resultados del código PHP interpretado y ejecutado, que puede ser cualquier tipo de datos, incluidas las imágenes, con la página web generada. El código PHP también se puede ejecutar con una [interfaz de línea de comandos](https://en.wikipedia.org/wiki/Command-line_interface) (CLI) y se puede usar para implementar [aplicaciones](https://en.wikipedia.org/wiki/Computer_software)[gráficas](https://en.wikipedia.org/wiki/Graphical_user_interface)[independientes](https://en.wikipedia.org/wiki/Computer_software) .

El intérprete estándar de PHP, impulsado por [Zend Engine](https://en.wikipedia.org/wiki/Zend_Engine" \o "Motor de zend) , es [un software gratuito](https://en.wikipedia.org/wiki/Free_software) publicado bajo la [Licencia PHP](https://en.wikipedia.org/wiki/PHP_License) . PHP se ha implementado ampliamente y se puede implementar en la mayoría de los servidores web en casi todos [los sistemas operativos](https://en.wikipedia.org/wiki/Operating_system) y [plataformas](https://en.wikipedia.org/wiki/Computing_platform) , sin cargo alguno.

El lenguaje PHP evolucionó sin una [especificación formal](https://en.wikipedia.org/wiki/Formal_specification) escrita o estándar hasta 2014, con la implementación original actuando como el estándar [de facto](https://en.wikipedia.org/wiki/De_facto) que otras implementaciones pretendían seguir. Desde 2014, el trabajo ha continuado para crear una especificación PHP formal.

(Colaboradores de Wikipedia, 2018)

### BASE DE DATOS

En este post quiero compartir con ustedes un programa que yo considero muy útil y fácil de utilizar. Estoy hablando del software Apache Friends Xampp. El server Xampp es un servidor independiente de plataforma que consiste principalmente en la base de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script: PHP y Perl. El nombre proviene del acrónimo deX(para cualquiera de los diferentes sistemas operativos), Apache, MySQL, PHP, Perl. Con Xampp podemos crear, editar, consultar y gestionar bases de datos en MySQL de una manera rápida y sencilla. Para obtener Xampp necesitamos descargarlo desde su página oficial: [https://www.apachefriends.org](https://www.apachefriends.org/)

Debemos descargarlo e instalarlo en nuestro equipo. Cuando esté instalado, podemos crear nuestra primera base de datos. En el siguiente video se muestra los pasos a seguir para crear una sencilla base de datos MySQL.

(García González, 2013)

### HERRAMIENTAS CASE UML

Una herramienta UML es una [aplicación](https://es.wikipedia.org/wiki/Aplicaci%C3%B3n_inform%C3%A1tica) de [software](https://es.wikipedia.org/wiki/Software) que utilizan los analistas y programadores informáticos y les facilitan el trabajo con todo tipo de [diagramas UML](https://es.wikipedia.org/w/index.php?title=Diagrama_UML&action=edit&redlink=1) o LUM ([Lenguaje Unificado de Modelado](https://es.wikipedia.org/wiki/Lenguaje_Unificado_de_Modelado)), tales como diagramas de estructura, diagramas de comportamiento o diagramas de interacción.

(Wikipedia, colaboradores de, 2017)

# **CONCLUSION**

Después de Analizar la solución propuesta e implementarla es posible concluir:

El uso correcto de las nuevas tecnologías de comunicación, los medios de notificación, y los servicios de inteligencia artificial, nos facilitan la solución del problema de falta de comunicación en la facultad.